**SQL Interview Questions**

**1.****What is SQL?**

SQL stands for Structured Query Language. It is a language used to interact with the database, i.e to create a database, to create a table in the database, to retrieve data or update a table in the database, etc. SQL is an ANSI(American National Standards Institute) standard. Using SQL, we can do many things. For example – we can execute queries, we can insert records into a table, can update records, can create a database, can create a table, can delete a table, etc.

**2. What is a database?**

A Database is defined as a structured form of data storage in a computer or a collection of data in an organized manner and can be accessed in various ways. It is also the collection of schemas, tables, queries, views, etc. Databases help us with easily storing, accessing, and manipulating data held on a computer. The Database Management System allows a user to interact with the database.

**3. Does SQL support programming language features?**

It is true that SQL is a language, but it does not support programming as it is not a programming language, it is a command language. We do not have conditional statements in SQL like for loops or if..else, we only have commands which we can use to query, update, delete, etc. data in the database. SQL allows us to manipulate data in a database.

**4. What are the differences between SQL and PL/SQL?**

Some common differences between SQL and PL/SQL are as shown below: 

|  |  |
| --- | --- |
| **SQL** | **PL/SQL** |
| **SQL is a query execution or commanding language** | **PL/SQL is a complete programming language** |
| **SQL is a data-oriented language.** | **PL/SQL is a procedural language** |
| **SQL is very declarative in nature.** | **PL/SQL has a procedural nature.** |
| **It is used for manipulating data.** | **It is used for creating applications.** |
| **We can execute one statement at a time in SQL** | **We can execute blocks of statements in PL/SQL** |
| **SQL tells databases, what to do?** | **PL/SQL tells databases how to do.** |
| **We can embed SQL in PL/SQL** | **We can not embed PL/SQL in SQL** |

**5. What is the difference between BETWEEN and IN operators in SQL?**

**BETWEEN:**The **BETWEEN** operator is used to fetch rows based on a range of values.   
For example,

SELECT \* FROM Students

WHERE ROLL\_NO BETWEEN 20 AND 30;

This query will select all those rows from the table. Students where the value of the field ROLL\_NO lies between 20 and 30.   
**IN**   
The **IN**operator is used to check for values contained in specific sets.   
For example,

SELECT \* FROM Students

WHERE ROLL\_NO IN (20,21,23);

This query will select all those rows from the table Students where the value of the field ROLL\_NO is either 20 or 21 or 23.

**6. Write an SQL query to find the names of employees starting with ‘A’.**

The LIKE operator of SQL is used for this purpose. It is used to fetch filtered data by searching for a particular pattern in the where clause.   
The Syntax for using LIKE is, 

**SELECT column1,column2 FROM table\_name WHERE column\_name LIKE pattern;**

**LIKE**: operator name

**pattern**: exact value extracted from the pattern to get related data in

result set.

The required query is:

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

You may refer to this article [WHERE clause](https://www.geeksforgeeks.org/sql-where-clause/) for more details on the LIKE operator.

**7. What is the difference between CHAR and VARCHAR2 datatype in SQL?**

Both of these data types are used for characters, but varchar2 is used for character strings of variable length, whereas char is used for character strings of fixed length. For example, if we specify the type as char(5) then we will not be allowed to store a string of any other length in this variable, but if we specify the type of this variable as varchar2(5) then we will be allowed to store strings of variable length. We can store a string of length 3 or 4 or 2 in this variable.

**8. Name different types of case manipulation functions available in SQL.**

There are three types of case manipulation functions available in SQL. They are,

* **LOWER**: The purpose of this function is to return the string in lowercase. It takes a string as an argument and returns the string by converting it into lower case.   
  Syntax:

LOWER('string')

* **UPPER**: The purpose of this function is to return the string in uppercase. It takes a string as an argument and returns the string by converting it into uppercase.   
  Syntax:

UPPER('string')

* **INITCAP**: The purpose of this function is to return the string with the first letter in uppercase and the rest of the letters in lowercase.   
  Syntax:

INITCAP('string')

**9. What do you mean by data definition language?**

Data definition language or DDL allows to execution of queries like CREATE, DROP, and ALTER. That is those queries that define the data.

**10. What do you mean by data manipulation language?**

 Data manipulation Language or DML is used to access or manipulate data in the database.   
It allows us to perform the below-listed functions:

* Insert data or rows in a database
* Delete data from the database
* Retrieve or fetch data
* Update data in a database.

**11. What is the difference between primary key and unique constraints?**

The primary key cannot have NULL values, the unique constraints can have NULL values. There is only one primary key in a table, but there can be multiple unique constraints. The primary key creates the clustered index automatically but the unique key does not.

**12. What is the view in SQL?**

Views in SQL are a kind of virtual table. A view also has rows and columns as they are on a real table in the database. We can create a view by selecting fields from one or more tables present in the database. A View can either have all the rows of a table or specific rows based on certain conditions.

The CREATE VIEW statement of SQL is used for creating views.

Basic Syntax:

CREATE VIEW view\_name AS

SELECT column1, column2.....

FROM table\_name

WHERE condition;

**view\_name**: Name for the View

**table\_name**: Name of the table

**condition**: Condition to select rows

For more details on how to create and use view, please refer to [this](https://www.geeksforgeeks.org/sql-views/) article.

**13. What do you mean by foreign key?**

A Foreign key is a field that can uniquely identify each row in another table. And this constraint is used to specify a field as a Foreign key. That is this field points to the primary key of another table. This usually creates a kind of link between the two tables.   
Consider the two tables as shown below:   
 **Orders**

|  |  |  |
| --- | --- | --- |
| **O\_ID** | **ORDER\_NO** | **C\_ID** |
| **1** | **2253** | **3** |
| **2** | **3325** | **3** |
| **3** | **4521** | **2** |
| **4** | **8532** | **1** |

**Customers**

|  |  |  |
| --- | --- | --- |
| **C\_ID** | **NAME** | **ADDRESS** |
| **1** | **RAMESH** | **DELHI** |
| **2** | **SURESH** | **NOIDA** |
| **3** | **DHARMESH** | **GURGAON** |

As we can see clearly, that the field C\_ID in the Orders table is the primary key in the Customers’ table, i.e. it uniquely identifies each row in the Customers table. Therefore, it is a Foreign Key in the Orders table.   
Syntax:

CREATE TABLE Orders

(

O\_ID int NOT NULL,

ORDER\_NO int NOT NULL,

C\_ID int,

PRIMARY KEY (O\_ID),

FOREIGN KEY (C\_ID) REFERENCES Customers(C\_ID)

)

**14. What is a join in SQL? What are the types of joins?**

An SQL Join statement is used to combine data or rows from two or more tables based on a common field between them. Different types of Joins are: 

* **INNER JOIN**: The INNER JOIN keyword selects all rows from both tables as long as the condition is satisfied. This keyword will create the result set by combining all rows from both the tables where the condition satisfies i.e. the value of the common field will be the same.
* **LEFT JOIN**: This join returns all the rows of the table on the left side of the join and matching rows for the table on the right side of the join. For the rows for which there is no matching row on the right side, the result set will be null. LEFT JOIN is also known as LEFT OUTER JOIN
* **RIGHT JOIN**: RIGHT JOIN is similar to LEFT JOIN. This join returns all the rows of the table on the right side of the join and matching rows for the table on the left side of the join. For the rows for which there is no matching row on the left side, the result set will contain null. RIGHT JOIN is also known as RIGHT OUTER JOIN.
* **FULL JOIN**: FULL JOIN creates the result set by combining the results of both LEFT JOIN and RIGHT JOIN. The result set will contain all the rows from both tables. For the rows for which there is no matching, the result set will contain NULL values.

**15. What is an index?**

A database index is a data structure that improves the speed of data retrieval operations on a database table at the cost of additional writes and the use of more storage space to maintain the extra copy of data. Data can be stored only in one order on a disk. To support faster access according to different values, a faster search like a binary search for different values is desired. For this purpose, indexes are created on tables. These indexes need extra space on the disk, but they allow faster search according to different frequently searched values.

**16. What are table and Field?**

**Table:**A table has a combination of rows and columns. Rows are called records and columns are called fields. In MS SQL Server, the tables are being designated within the database and schema names.

**Field:**In DBMS, a database field can be defined as – a single piece of information from a record.

**17. What is a primary key?**

A [Primary Key](https://www.geeksforgeeks.org/difference-between-primary-and-candidate-key/) is one of the candidate keys. One of the candidate keys is selected as the most important and becomes the primary key. There cannot be more than one primary key in a table.

**18. What is a Default constraint?**

The **DEFAULT**constraint is used to fill a column with default and fixed values. The value will be added to all new records when no other value is provided. For more details please refer to the [SQL | Default Constraint](https://www.geeksforgeeks.org/sql-default-constraint/?ref=rp) article.

**19. What is the On Delete cascade constraint?**

An ‘ON DELETE CASCADE’ constraint is used in MySQL to delete the rows from the child table automatically when the rows from the parent table are deleted. For more details, please read [MySQL – On Delete Cascade constraint](https://www.geeksforgeeks.org/mysql-on-delete-cascade-constraint/)article.

**20. What is normalization?**

It is a process of analyzing the given relation schemas based on their functional dependencies and primary keys to achieve the following desirable properties:   
1) Minimizing Redundancy   
2) Minimizing the Insertion, Deletion, And Update Anomalies   
Relation schemas that do not meet the properties are decomposed into smaller relation schemas that could meet desirable properties.

**21. What is Denormalization?**

Denormalization is a database optimization technique in which we add redundant data to one or more tables. This can help us avoid costly joins in a relational database. Note that denormalization does not mean not doing normalization. It is an optimization technique that is applied after normalization.

In a traditional normalized database, we store data in separate logical tables and attempt to minimize redundant data. We may strive to have only one copy of each piece of data in the database.

**22. Explain  WITH clause in SQL?**

The WITH clause provides a way relationship of defining a temporary relationship whose definition is available only to the query in which the with clause occurs. SQL applies predicates in the WITH clause after groups have been formed, so aggregate functions may be used.

**23. What are all the different attributes of indexes?**

The indexing has various attributes:

* **Access Types**: This refers to the type of access such as value-based search, range access, etc.
* **Access Time**: It refers to the time needed to find a particular data element or set of elements.
* **Insertion Time**: It refers to the time taken to find the appropriate space and insert new data.
* **Deletion Time**: Time is taken to find an item and delete it as well as update the index structure.
* **Space Overhead**: It refers to the additional space required by the index.

**24. What is a Cursor?**

The cursor is a Temporary Memory or Temporary Work Station. It is Allocated by Database Server at the Time of Performing DML operations on the Table by the User. Cursors are used to store Database Tables.

**25. Write down various types of relationships in SQL?**

There are various relationships, namely:

* One-to-One Relationship.
* One to Many Relationships.
* Many to One Relationship.
* Self-Referencing Relationship.

**26. What is a query?**

An [**SQL**](https://www.geeksforgeeks.org/sql-tutorial/) query is used to retrieve the required data from the database. However, there may be multiple SQL queries that yield the same results but with different levels of efficiency. An inefficient query can drain the database resources, reduce the database speed or result in a loss of service for other users. So it is very important to optimize the query to obtain the best database performance.

**27. What is a subquery?**

In SQL a Subquery can be simply defined as a query within another query. In other words, we can say that a Subquery is a query that is embedded in the WHERE clause of another SQL query.

**28. What are the different operators available in SQL?**

There are three operators available in SQL namely:

1. Arithmetic Operators
2. Logical Operators
3. Comparison Operators

**29. What is a trigger?**

**The trigger** is a statement that a system executes automatically when there is any modification to the database. In a trigger, we first specify when the trigger is to be executed and then the action to be performed when the trigger executes. Triggers are used to specify certain integrity constraints and referential constraints that cannot be specified using the constraint mechanism of SQL.

**30. What is the difference between DELETE and TRUNCATE commands?**

|  |  |
| --- | --- |
| The DELETE statement removes rows one at a time and records an entry in the transaction log for each deleted row. | TRUNCATE TABLE removes the data by deallocating the data pages used to store the table data and records only the page deallocations in the transaction log. |
| DELETE command is slower than the identityTRUNCATE command. | While the TRUNCATE command is faster than the DELETE command. |
| To use Delete you need DELETE permission on the table. | To use Truncate on a table we need at least ALTER permission on the table. |
| The identity of the column retains the identity after using DELETE Statement on the table. | The identity of the column is reset to its seed value if the table contains an identity column. |
| The delete can be used with indexed views. | Truncate cannot be used with indexed views. |

**31. What are local and global variables and their differences?**

* **Global Variable:**

In contrast, global variables are variables that are defined outside of functions. These variables have global scope, so they can be used by any function without passing them to the function as parameters.

* **Local Variable:**

Local variables are variables that are defined within functions. They have local scope, which means that they can only be used within the functions that define them.

**32. What is a constraint?**

Constraints are the rules that we can apply to the type of data in a table. That is, we can specify the limit on the type of data that can be stored in a particular column in a table using constraints. For more details please refer to [SQL|Constraints](https://www.geeksforgeeks.org/sql-constraints/) article.

**33. What is Data Integrity?**

Data integrity is defined as the data contained in the database being both correct and consistent. For this purpose, the data stored in the database must satisfy certain types of procedures (rules). The data in a database must be correct and consistent. So, data stored in the database must satisfy certain types of procedures (rules). DBMS provides different ways to implement such types of constraints (rules). This improves data integrity in a database. For more details please refer [difference between data security and data integrity](https://www.geeksforgeeks.org/difference-between-data-security-and-data-integrity/) article.

**34. What is Auto Increment?**

Sometimes, while creating a table, we do not have a unique identifier within the table, hence we face difficulty in choosing Primary Key. So as to resolve such an issue, we’ve to manually provide unique keys to every record, but this is often also a tedious task. So we can use the  Auto-Increment feature that automatically generates a numerical Primary key value for every new record inserted. The Auto Increment feature is supported by all the Databases. For more details please refer [SQL Auto Increment](https://www.geeksforgeeks.org/sql-auto-increment/)article.

**35. What is the difference between Cluster and Non-Cluster Index?**

| **CLUSTERED INDEX** | **NON-CLUSTERED INDEX** |
| --- | --- |
| The clustered index is faster. | The non-clustered index is slower. |
| The clustered index requires less memory for operations. | The non-Clustered index requires more memory for operations. |
| In a clustered index, the index is the main data. | In the Non-Clustered index, the index is a copy of data. |
| A table can have only one clustered index. | A table can have multiple non-clustered indexes. |
| The clustered index has an inherent ability to store data on the disk. | The non-Clustered index does not have the inherent ability to store data on the disk. |
| Clustered indexes store pointers to block not data. | The non-Clustered index store both value and a pointer to the the the actual row that holds data. |
| In Clustered index leaf nodes are actual data itself. | In a Non-Clustered index, leaf nodes are not the actual data itself rather they only contain included columns. |
| In the Clustered index, the Clustered key defines the order of data within the table. | In the Non-Clustered index, the index key defines the order of data within the index. |
| A Clustered index is a type of index in which table records are physically reordered to match the index. | A Non-Clustered index is a special type of index in which the logical order of index does not match the physical stored order of the rows on the disk. |

For more details please refer [Difference between Clustered index and the No-Clustered index](https://www.geeksforgeeks.org/difference-between-clustered-and-non-clustered-index/)article.

**36. What is MySQL collation?**

A MySQL collation is a well-defined set of rules which are used to compare characters of a particular character set by using their corresponding encoding. Each character set in MySQL might have more than one collation, and has, at least, one default collation. Two character sets cannot have the same collation. For more details please refer [What are collation and character set in MySQL?](https://www.geeksforgeeks.org/what-is-collation-and-character-set-in-mysql/)article.

**37. What are user-defined functions?**

We can use User-defined functions in PL/SQL or Java to provide functionality that is not available in SQL or SQL built-in functions. SQL functions and User-defined functions can appear anywhere, that is, wherever an expression occurs.

For example, it can be used in:

* Select a list of SELECT statements.
* Condition of the WHERE clause.
* CONNECT BY, ORDER BY, START WITH, and GROUP BY
* The VALUES clause of the INSERT statement.
* The SET clause of the UPDATE statement.

**38. What are all types of user-defined functions?**

User-Defined Functions allow people to define their own T-SQL functions that can accept 0 or more parameters and return a single scalar data value or a table data type.  
Different Kinds of User-Defined Functions created are:

**1. Scalar User-Defined Function** A Scalar user-defined function returns one of the scalar data types. Text, image, and timestamp data types are not supported. These are the type of user-defined functions that most developers are used to in other programming languages. You pass in 0 to many parameters and you get a return value.

**2. Inline Table-Value User-Defined Function** An Inline Table-Value user-defined function returns a table data type and is an exceptional alternative to a view as the user-defined function can pass parameters into a T-SQL select command and, in essence, provide us with a parameterized, non-updateable view of the underlying tables.

**3. Multi-statement Table-Value User-Defined Function** A Multi-Statement Table-Value user-defined function returns a table and is also an exceptional alternative to a view, as the function can support multiple T-SQL statements to build the final result where the view is limited to a single SELECT statement. Also, the ability to pass parameters into a TSQL select command or a group of them gives us the capability to, in essence, create a parameterized, non-updateable view of the data in the underlying tables. Within the create function command you must define the table structure that is being returned. After creating this type of user-defined function, it can be used in the FROM clause of a T-SQL command, unlike the behavior found when using a stored procedure which can also return record sets.

**39. Name the function which is used to remove spaces at the end of a string?**

In SQL the spaces at the end of the string are removed by a trim function.

**Syntax:**

Trim(s)

Where s is a any string.

**40. What is a  stored procedure?**

**Stored Procedures** are created to perform one or more DML operations on databases. It is nothing but a group of SQL statements that accepts some input in the form of parameters and performs some task and may or may not return a value. For more details please refer to our[Stored procedures in the SQL](https://www.geeksforgeeks.org/what-is-stored-procedures-in-sql/) article.

**41. What are Union, minus, and Interact commands?**

Set Operations in SQL eliminate duplicate tuples and can be applied only to the relations which are union compatible. Set Operations available in SQL are :

* Set Union
* Set Intersection
* Set Difference

**UNION Operation:** This operation includes all the tuples which are present in either of the relations. For example: To find all the customers who have a loan or an account or both in a bank.

SELECT CustomerName FROM Depositor

UNION

SELECT CustomerName FROM Borrower ;

The union operation automatically eliminates duplicates. If all the duplicates are supposed to be retained, UNION ALL is used in place of UNION.

**INTERSECT Operation:**This operation includes the tuples which are present in both of the relations. For example: To find the customers who have a loan as well as an account in the bank:

SELECT CustomerName FROM Depositor

INTERSECT

SELECT CustomerName FROM Borrower ;

The Intersect operation automatically eliminates duplicates. If all the duplicates are supposed to be retained, INTERSECT ALL is used in place of INTERSECT.

**EXCEPT for Operation:**This operation includes tuples that are present in one relationship but should not be present in another relationship. For example: To find customers who have an account but no loan at the bank:

SELECT CustomerName FROM Depositor

EXCEPT

SELECT CustomerName FROM Borrower ;

The Except operation automatically eliminates the duplicates. If all the duplicates are supposed to be retained, EXCEPT ALL is used in place of EXCEPT.

**42. What is an ALIAS command?**

Aliases are the temporary names given to a table or column for the purpose of a particular SQL query. It is used when the name of a column or table is used other than its original name, but the modified name is only temporary.

* Aliases are created to make table or column names more readable.
* The renaming is just a temporary change and the table name does not change in the original database.
* Aliases are useful when table or column names are big or not very readable.
* These are preferred when there is more than one table involved in a query.

For more details, please read the [SQL | Aliases](https://www.geeksforgeeks.org/sql-aliases/)article.

**43. What is the difference between TRUNCATE and DROP statements?**

| **S.NO.** | **DROP** | **TRUNCATE** |
| --- | --- | --- |
| 1. | The DROP command is used to remove the table definition and its contents. | Whereas the TRUNCATE command is used to delete all the rows from the table. |
| 2. | In the DROP command, table space is freed from memory. | While the TRUNCATE command does not free the table space from memory. |
| 3. | DROP is a DDL(Data Definition Language) command. | Whereas the TRUNCATE is also a DDL(Data Definition Language) command. |
| 4. | In the DROP command, a view of the table does not exist. | While in this command, a view of the table exists. |
| 5. | In the DROP command, integrity constraints will be removed. | While in this command, integrity constraints will not be removed. |
| 6. | In the DROP command, undo space is not used. | While in this command, undo space is used but less than DELETE. |
| 7. | The DROP command is quick to perform but gives rise to complications. | While this command is faster than DROP. |

For more details, please read the Difference between [DROP and TRUNCATE in](https://www.geeksforgeeks.org/difference-between-drop-and-truncate-in-sql/)the [SQL](https://www.geeksforgeeks.org/difference-between-drop-and-truncate-in-sql/) article.

**44. What are aggregate and scalar functions?**

For doing operations on data SQL has many built-in functions, they are categorized into two categories and further sub-categorized into seven different functions under each category. The categories are:

**Aggregate functions:**  
These functions are used to do operations from the values of the column and a single value is returned.

**Scalar functions:**  
These functions are based on user input, these too return a single value.

For more details, please read the [SQL | Functions (Aggregate and Scalar Functions)](https://www.geeksforgeeks.org/sql-functions-aggregate-scalar-functions/) article.

**45. Which operator is used in queries for pattern matching?**

LIKE operator: It is used to fetch filtered data by searching for a particular pattern in the where clause.

**Syntax:**

**SELECT column1,column2 FROM table\_name WHERE column\_name LIKE pattern;**

LIKE: operator name

**46. Define SQL Order by the statement?**

The ORDER BY statement in SQL is used to sort the fetched data in either ascending or descending according to one or more columns.

* By default ORDER BY sorts the data in **ascending order.**
* We can use the keyword DESC to sort the data in descending order and the keyword ASC to sort in ascending order.

For more details please read [SQL | ORDER BY](https://www.geeksforgeeks.org/sql-order-by/) article.

**47. Explain SQL Having statement?**

HAVING is used to specify a condition for a group or an aggregate function used in the select statement. The WHERE clause selects before grouping. The HAVING clause selects rows after grouping. Unlike the HAVING clause, the WHERE clause cannot contain aggregate functions. (See [this](http://newtonapples.com/difference-clause-clause/)for examples). See [Having vs Where Clause?](https://www.geeksforgeeks.org/having-vs-where-clause-in-sql/)

**48. Explain SQL AND OR statement with an example?**

In SQL, the AND & OR operators are used for filtering the data and getting precise results based on conditions.

The AND and OR operators are used with the WHERE clause.

These two operators are called conjunctive operators.

**AND Operator:**This operator displays only those records where both conditions **condition 1 and condition 2 evaluate to True.**

**OR Operator:**This operator displays the records where either one of the conditions condition 1 and condition 2 evaluates to True. That is,**either condition1 is True or condition2 is True.**

For more details please read the [SQL | AND and OR](https://www.geeksforgeeks.org/sql-and-and-or-operators/)operators article.

**49. Define BETWEEN statements in SQL?**

The SQL BETWEEN condition allows you to easily test if an expression is within a range of values (inclusive). The values can be text, date, or numbers. It can be used in a SELECT, INSERT, UPDATE, or DELETE statement. The SQL BETWEEN Condition will return the records where the expression is within the range of value1 and value2.

For more details please read [SQL | Between & I operator](https://www.geeksforgeeks.org/sql-between-in-operator/) article.

**50. Why do we use  Commit and Rollback commands?**

| **COMMIT** | **ROLLBACK** |
| --- | --- |
| COMMIT permanently saves the changes made by the current transaction. | ROLLBACK undo the changes made by the current transaction. |
| The transaction can not undo changes after COMMIT execution. | Transaction reaches its previous state after ROLLBACK. |
| When the transaction is successful, COMMIT is applied. | When the transaction is aborted, ROLLBACK occurs. |

For more details please read the [Difference between Commit and Rollback in SQL](https://www.geeksforgeeks.org/difference-between-commit-and-rollback-in-sql/)article.

**51. What are ACID properties?**

A [**transaction**](https://www.geeksforgeeks.org/sql-transactions/) is a single logical unit of work that accesses and possibly modifies the contents of a database. Transactions access data using read-and-write operations. In order to maintain consistency in a database, before and after the transaction, certain properties are followed. These are called **ACID** properties. [ACID](http://en.wikipedia.org/wiki/ACID)(Atomicity, Consistency, Isolation, Durability) is a set of properties that guarantee that database transactions are processed reliably. For more details please read [ACID properties in](https://www.geeksforgeeks.org/acid-properties-in-dbms/)the [DBMS](https://www.geeksforgeeks.org/acid-properties-in-dbms/) article.

**52. What is a T-SQL?**

T-SQL is an abbreviation for Transact Structure Query Language. It is a product by Microsoft and is an extension of SQL Language which is used to interact with relational databases. It is considered to perform best with Microsoft SQL servers. T-SQL statements are used to perform the transactions to the databases. T-SQL has huge importance since all the communications with an instance of an SQL server are done by sending Transact-SQL statements to the server. Users can also define functions using T-SQL.

Types of T-SQL functions are :

* **Aggregate** functions.
* **Ranking** functions. There are different types of ranking functions.
* **Rowset** function.
* **Scalar** functions.

**53. Are NULL values the same as zero or a blank space?**

In SQL, zero or blank space can be compared with another zero or blank space. whereas one null may not be equal to another null. null means data might not be provided or there is no data.

**54. What is the need for group functions in SQL?**

In database management, group functions, also known as aggregate functions,  is a function where the values of multiple rows are grouped together as input on certain criteria to form a single value of more significant meaning.

**Various Group Functions**

1) Count()

2) Sum()

3) Avg()

4) Min()

5) Max()

For more details please read the [Aggregate functions in the SQL](https://www.geeksforgeeks.org/aggregate-functions-in-sql/) article.

**55. What is the need for a MERGE statement?**

The **MERGE** command in SQL is actually a combination of three SQL statements: **INSERT, UPDATE, and DELETE**. In simple words, the MERGE statement in SQL provides a convenient way to perform all these three operations together which can be very helpful when it comes to handling large running databases. But unlike INSERT, UPDATE, and DELETE statements MERGE statement requires a source table to perform these operations on the required table which is called a target table. For more details please read the [SQL | MERGE Statement](https://www.geeksforgeeks.org/sql-merge-statement/) article.

**56. How can you fetch common records from two tables?**

The below statement could be used to get data from multiple tables, so, we need to use join to get data from multiple tables.

**Syntax :**

SELECT tablenmae1.colunmname, tablename2.columnnmae

FROM tablenmae1

JOIN tablename2

ON tablenmae1.colunmnam = tablename2.columnnmae

ORDER BY columnname;

For more details and examples, please read [SQL | SELECT data from the Multiple Tables](https://www.geeksforgeeks.org/sql-select-data-from-multiple-tables/) article.

**57. What are the advantages of PL/SQL functions?**

The advantages of PL / SQL functions are as follows:

* We can make a single call to the database to run a block of statements. Thus, it improves the performance against running SQL multiple times. This will reduce the number of calls between the database and the application.
* We can divide the overall work into small modules which becomes quite manageable, also enhancing the readability of the code.
* It promotes reusability.
* It is secure since the code stays inside the database, thus hiding internal database details from the application(user). The user only makes a call to the PL/SQL functions. Hence, security and data hiding is ensured.

**58.  What is the SQL query to display the current date?**

CURRENT\_DATE returns to the current date. This function returns the same value if it is executed more than once in a single statement, which means that the value is fixed, even if there is a long delay between fetching rows in a cursor.

**Syntax:**

CURRENT\_DATE

or

CURRENT DATE

**59. What is ETL in SQL?**

ETL is a process in Data Warehousing and it stands for **Extract**, **Transform,**and **Load**. It is a process in which an ETL tool extracts the data from various data source systems, transforms it in the staging area, and then finally, loads it into the Data Warehouse system. These are three database functions that are incorporated into one tool to pull data out from one database and put data into another database.

**60.  What are Nested Triggers?**

A trigger can also contain INSERT, UPDATE, and DELETE logic within itself, so when the trigger is fired because of data modification it can also cause another data modification, thereby firing another trigger. A trigger that contains data modification logic within itself is called a nested trigger.

**61. How to find the available constraint information in the table?**

In SQL Server the **data dictionary** is a set of database tables used to store information about a database’s definition. One can use these data dictionaries to check the constraints on an already existing table and to change them(if possible). For more details please read [SQL | Checking Existing Constraint on a table](https://www.geeksforgeeks.org/sql-checking-existing-constraints-on-a-table-using-data-dictionaries/)article.

**62.  What is SQL injection?**

SQL injection is a technique used to exploit user data through web page inputs by injecting SQL commands as statements. Basically, these statements can be used to manipulate the application’s web server by malicious users.

* SQL injection is a code injection technique that might destroy your database.
* SQL injection is one of the most common web hacking techniques.
* SQL injection is the placement of malicious code in SQL statements, via web page input.

For more details, please read the [SQL | Injection](https://www.geeksforgeeks.org/sql-injection-2/)article.

**63. How to copy tables in SQL?**

Sometimes, in SQL, we need to create an exact copy of an already defined (or created) table. [MySQL](https://www.geeksforgeeks.org/sql-tutorial/#mysql) enables you to perform this operation. Because we may need such duplicate tables for testing the data without having any impact on the original table and the data stored in it.

CREATE TABLE Contact List(Clone\_1) LIKE Original\_table;

For more details, Please read [Cloning Table in](https://www.geeksforgeeks.org/cloning-table-in-mysql/)the [MySQL](https://www.geeksforgeeks.org/cloning-table-in-mysql/) article.

**64. Can we disable a trigger? If yes, how?**

 Yes, we can disable a trigger in PL/SQL. If consider temporarily disabling a trigger and one of the following conditions is true:

* An object that the trigger references is not available.
* We must perform a large data load and want it to proceed quickly without firing triggers.
* We are loading data into the table to which the trigger applies.
* We disable a trigger using the ALTER TRIGGER statement with the DISABLE option.
* We can disable all triggers associated with a table at the same time using the ALTER TABLE statement with the DISABLE ALL TRIGGERS option.

**65.  What is a Live Lock?**

**Livelock** occurs when two or more processes continually repeat the same interaction in response to changes in the other processes without doing any useful work. These processes are not in the waiting state, and they are running concurrently. This is different from a deadlock because in a deadlock all processes are in the waiting state.

**66.  How do we avoid getting duplicate entries in a query without using the distinct keyword?**

DISTINCT is useful in certain circumstances, but it has drawbacks that it can increase the load on the query engine to perform the sort (since it needs to compare the result set to itself to remove duplicates). We can remove duplicate entries using the following options:

* Remove duplicates using row numbers.
* Remove duplicates using self-Join.
* Remove duplicates using group by.     For more details, please read [SQL | Remove duplicates without distinct](https://www.geeksforgeeks.org/sql-remove-duplicates-without-distinct/) articles.

**67. The difference between NVL and NVL2 functions?**

These functions work with any data type and pertain to the use of null values in the expression list. These are all single-rowfunctions i.e. provide one result per row.

**NVL(expr1, expr2):** In SQL, NVL() converts a null value to an actual value. Data types that can be used are date, character, and number. Data types must match with each other. i.e. expr1 and expr2 must be of the same data type.

**Syntax:**

**NVL (expr1, expr2)**

**NVL2(expr1, expr2, expr3):** The NVL2 function examines the first expression. If the first expression is not null, then the NVL2 function returns the second expression. If the first expression is null, then the third expression is returned i.e. If expr1 is not null, NVL2 returns expr2. If expr1 is null, NVL2 returns expr3. The argument expr1 can have any data type.

**Syntax:**

**NVL2 (expr1, expr2, expr3)**

For more details please read [SQL general functions | NVL,](https://www.geeksforgeeks.org/sql-general-functions-nvl-nvl2-decode-coalesce-nullif-lnnvl-nanvl/) [NVL2, DECODE, COALESCE, NULLIF, LNNVL](https://www.geeksforgeeks.org/sql-general-functions-nvl-nvl2-decode-coalesce-nullif-lnnvl-nanvl/),[and NANVL](https://www.geeksforgeeks.org/sql-general-functions-nvl-nvl2-decode-coalesce-nullif-lnnvl-nanvl/) article.

**68. What is Case WHEN in SQL?**

Control statements form an important part of most languages since they control the execution of other sets of statements. These are found in SQL too and should be exploited for uses such as query filtering and query optimization through careful selection of tuples that match our requirements. In this post, we explore the Case-Switch statement in SQL. The CASE statement is SQL’s way of handling if/then logic.

syntax: 1

CASE case\_value

WHEN when\_value THEN statement\_list

[WHEN when\_value THEN statement\_list] ...

[ELSE statement\_list]

END CASE

syntax: 2

CASE

WHEN search\_condition THEN statement\_list

[WHEN search\_condition THEN statement\_list] ...

[ELSE statement\_list]

END CASE

For more details, please read the [SQL | Case Statement](https://www.geeksforgeeks.org/sql-case-statement/) article.

**69.  What is the difference between COALESCE() & ISNULL()?**

**COALESCE():**COALESCE function in SQL returns the first non-NULL expression among its arguments. If all the expressions evaluate to null, then the COALESCE function will return null.  
Syntax:

SELECT column(s), CAOLESCE(expression\_1,....,expression\_n)

FROM table\_name;

**ISNULL():**The ISNULL function has different uses in SQL Server and MySQL. In SQL Server, ISNULL() function is used to replace NULL values.  
**Syntax:**

SELECT column(s), ISNULL(column\_name, value\_to\_replace)

FROM table\_name;

For more details, please read the [SQL | Null functions](https://www.geeksforgeeks.org/sql-null-functions/) article.

**70. Name the operator which is used in the query for appending two strings?**

In SQL for appending two strings, the ” Concentration operator”  is used and its symbol is ” || “.

**1. What is SQL?**

It stands for Structured Query Language. A programming language used for interaction with relational database management systems (RDBMS). This includes fetching, updating, inserting, and removing data from tables.

**2. What are SQL dialects? Give some examples.**

The various versions of SQL, both free and paid, are also called SQL dialects. All the flavors of SQL have a very similar syntax and vary insignificantly only in additional functionality. Some examples are Microsoft SQL Server, PostgreSQL, MySQL, SQLite, T-SQL, Oracle, and MongoDB.

**3. What are the main applications of SQL?**

Using SQL, we can:

* create, delete, and update tables in a database
* access, manipulate, and modify data in a table
* retrieve and summarize the necessary information from a table or several tables
* add or remove certain rows or columns from a table

All in all, SQL allows querying a database in multiple ways. In addition, SQL easily integrates with other programming languages, such as Python or R, so we can use their combined power.

**4. What is an SQL statement? Give some examples.**

Also known as an SQL command. It's a string of characters interpreted by the SQL engine as a legal command and executed accordingly. Some examples of SQL statements are SELECT, CREATE, DELETE, DROP, REVOKE, and so on.

**5. What types of SQL commands (or SQL subsets) do you know?**

* **Data Definition Language (DDL)** – to define and modify the structure of a database.
* **Data Manipulation Language (DML)** – to access, manipulate, and modify data in a database.
* **Data Control Language (DCL)** – to control user access to the data in the database and give or revoke privileges to a specific user or a group of users.
* **Transaction Control Language (TCL)** – to control transactions in a database.
* **Data Query Language (DQL)** – to perform queries on the data in a database to retrieve the necessary information from it.

**6. Give some examples of common SQL commands of each type.**

* **DDL:** CREATE, ALTER TABLE, DROP, TRUNCATE, and ADD COLUMN
* **DML:** UPDATE, DELETE, and INSERT
* **DCL:** GRANT and REVOKE
* **TCL:** COMMIT, SET TRANSACTION, ROLLBACK, and SAVEPOINT
* **DQL:** – SELECT

**7. What is a database?**

A structured storage space where the data is kept in many tables and organized so that the necessary information can be easily fetched, manipulated, and summarized.

**8. What is DBMS, and what types of DBMS do you know?**

It stands for Database Management System, a software package used to perform various operations on the data stored in a database, such as accessing, updating, wrangling, inserting, and removing data. There are various types of DBMS, such as relational, hierarchical, network, graph, or object-oriented. These types are based on the way the data is organized, structured, and stored in the system.

**9. What is RDBMS? Give some examples of RDBMS.**

It stands for Relational Database Management System. It's the most common type of DBMS used for working with data stored in multiple tables related to each other by means of shared keys. The SQL programming language is particularly designed to interact with RDBMS. Some examples of RDBMS are MySQL, PostgreSQL, Oracle, MariaDB, etc.

**10. What are tables and fields in SQL?**

A table is an organized set of related data stored in a tabular form, i.e., in rows and columns. A field is another term for a column of a table.

**11. What is an SQL query, and what types of queries do you know?**

A query is a piece of code written in SQL to access the data from a database or to modify the data. Correspondingly, there are two types of SQL queries: **select** and **action** queries. The first ones are used to retrieve the necessary data (this also includes limiting, grouping, ordering the data, extracting the data from multiple tables, etc.), while the second ones are used to create, add, delete, update, rename the data, etc.

**12. What is a subquery?**

Also called an inner query; a query placed inside another query, or an outer query. A subquery may occur in the clauses such as SELECT, FROM, WHERE, UPDATE, etc. It's also possible to have a subquery inside another subquery. The innermost subquery is run first, and its result is passed to the containing query (or subquery).

**13. What types of SQL subqueries do you know?**

* **Single-row** – returns at most one row.
* **Multi-row** – returns at least two rows.
* **Multi-column** – returns at least two columns.
* **Correlated** – a subquery related to the information from the outer query.
* **Nested** – a subquery inside another subquery.

**14. What is a constraint, and why use constraints?**

A set of conditions defining the type of data that can be input into each column of a table. Constraints ensure data integrity in a table and block undesired actions.

**15. What SQL constraints do you know?**

* DEFAULT – provides a default value for a column.
* UNIQUE – allows only unique values.
* NOT NULL – allows only non-null values.
* PRIMARY KEY – allows only unique and strictly non-null values (NOT NULL and UNIQUE).
* FOREIGN KEY – provides shared keys between two and more tables.

**16. What is a join?**

A clause used to combine and retrieve records from two or multiple tables. SQL tables can be joined based on the relationship between the columns of those tables. Check out our [SQL joins](https://www.datacamp.com/tutorial/introduction-to-sql-joins) tutorial for more context.

**17. What types of joins do you know?**

* (INNER) JOIN – returns only those records that satisfy a defined join condition in both (or all) tables. It's a default SQL join.
* LEFT (OUTER) JOIN – returns all records from the left table and those records from the right table that satisfy a defined join condition.
* RIGHT (OUTER) JOIN – returns all records from the right table and those records from the left table that satisfy a defined join condition.
* FULL (OUTER) JOIN – returns all records from both (or all) tables. It can be considered as a combination of left and right joins.

**18. What is a primary key?**

A column (or multiple columns) of a table to which the PRIMARY KEY constraint was imposed to ensure unique and non-null values in that column. In other words, a primary key is a combination of the NOT NULL and UNIQUE constraints. The primary key uniquely identifies each record of the table. Each table should contain a primary key and can't contain more than one primary key.

**19. What is a unique key?**

A column (or multiple columns) of a table to which the UNIQUE constraint was imposed to ensure unique values in that column, including a possible NULL value (the only one).

**20. What is a foreign key?**

A column (or multiple columns) of a table to which the FOREIGN KEY constraint was imposed to link this column to the primary key in another table (or several tables). The purpose of foreign keys is to keep connected various tables of a database.

**21. What is an index?**

A special data structure related to a database table and used for storing its important parts and enabling faster data search and retrieval. Indexes are especially efficient for large databases, where they significantly enhance query performance.

**22. What types of indexes do you know?**

* **Unique index** – doesn't allow duplicates in a table column and hence helps maintain data integrity.
* **Clustered index** – defines the physical order of records of a database table and performs data searching based on the key values. A table can have only one clustered index.
* **Non-clustered index** – keeps the order of the table records that doesn't match the physical order of the actual data on the disk. It means that the data is stored in one place and a non-clustered index – in another one. A table can have multiple non-clustered indexes.

**23. What is a schema?**

A collection of database structural elements such as tables, stored procedures, indexes, functions, and triggers. It shows the overall database architecture, specifies the relationships between various objects of a database, and defines different access permissions for them.

**24. What is a SQL comment?**

A human-readable clarification on what a particular piece of code does. SQL code comments can be single-line (preceded by a double dash --) or span over multiple lines (as follows: /\*comment\_text\*/). When the SQL engine runs, it ignores code comments. The purpose of adding SQL code comments is to make the code more comprehensive for those people who will read it in the future.

**25. What is a SQL operator?**

A reserved character, a combination of characters, or a keyword used in SQL queries to perform a specific operation. SQL operators are commonly used with the WHERE clause to set a condition (or conditions) for filtering the data.

**26. What types of SQL operators do you know?**

* **Arithmetic** (+, -, \*, /, etc.)
* **Comparison** (>, <, =, >=, etc.)
* **Compound** (+=, -=, \*=, /=, etc.)
* **Logical** (AND, OR, NOT, BETWEEN, etc.)
* **String** (%, \_, +, ^, etc.)
* **Set** (UNION, UNION ALL, INTERSECT, and MINUS (or EXCEPT))

**27. What is an alias?**

A temporary name given to a table (or a column in a table) while executing a certain SQL query. Aliases are used to improve the code readability and make the code more compact. An alias is introduced with the AS keyword:

SELECT col\_1 AS column

FROM table\_name;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**28. What is a clause?**

A condition imposed on a SQL query to filter the data to obtain the desired result. Some examples are WHERE, LIMIT, HAVING, LIKE, AND, OR, ORDER BY, etc.

**29. What are some common statements used with the SELECT query?**

The most common ones are FROM, GROUP BY, JOIN, WHERE, ORDER BY, LIMIT, and HAVING.

**30. How to create a table?**

Using the CREATE TABLE statement. For example, to create a table with three columns of predefined datatypes, we apply the following syntax:

CREATE TABLE table\_name (col\_1 datatype,

col\_2 datatype,

col\_3 datatype);

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**31. How to update a table?**

Using the UPDATE statement. The syntax is:

UPDATE table\_name

SET col\_1 = value\_1, column\_2 = value\_2

WHERE condition;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**32. How to delete a table from a database?**

Using the DROP TABLE statement. The syntax is: DROP TABLE table\_name;.

**33. How to get the count of records in a table?**

Using the COUNT() aggregate function with the asterisk passed as its argument: SELECT COUNT(\*) FROM table\_name;.

**34. How to sort records in a table?**

Using the ORDER BY statement:

SELECT \* FROM table\_name

ORDER BY col\_1;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

We can specify that we need a descending order using the DESC keyword; otherwise, the order will be ascending by default. Also, we can sort by more than one column and specify for each one, ascending or descending order separately. For example:

SELECT \* FROM table\_name

ORDER BY col\_1 DESC, col\_3, col\_6 DESC;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**35. How to select all columns from a table?**

Using the asterisk \* with the SELECT statement. The syntax is: SELECT \* FROM table\_name;.

**36. How to select common records from two tables?**

Using the INTERSECT statement:

SELECT \* FROM table\_1

INTERSECT

SELECT \* FROM table\_1;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**37. What is the DISTINCT statement and how do you use it?**

This statement is used with the SELECT statement to filter out duplicates and return only unique values from a column of a table. The syntax is:

SELECT DISTINCT col\_1

FROM table\_name;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**38. What are entities? Give some examples.**

An entity is a real-world object, creature, place, or phenomenon for which the data can be gathered and stored in a database table. Each entity corresponds to a row in a table, while the table's columns describe its properties. Some examples of entities are bank transactions, students in a school, cars sold, etc.

**39. What are relationships? Give some examples.**

Relationships are the connections and correlations between entities, basically meaning how two or more tables of a database are related to one another. For example, we can find an ID of the same client in a table on sales data and in a customer table.

**40. What is NULL value? How is it different from zero or a blank space?**

A NULL value indicates the absence of data for a certain cell of a table. Instead, zero is a valid numeric value, and an empty string is a legal string of zero length.

**40 Top Intermediate SQL Interview Questions and Answers**

In this section, we take a look at the 40 most popular intermediate SQL questions and answers, so that you'll know what to expect from your interviewer. These questions are more suited to SQL practitioners with a few years of experience.

**1. What is a function in SQL, and why use functions?**

A database object representing a set of SQL statements frequently used for a certain task. A function takes in some input parameters, performs calculations or other manipulations on them, and returns the result. Functions help improve code readability and avoid repetition of the same code snippets.

**2. What types of SQL functions do you know?**

* **Aggregate functions** – work on multiple, usually grouped records for the provided columns of a table, and return a single value (usually by group).
* **Scalar functions** – work on each individual value and return a single value.

On the other hand, SQL functions can be built-in (defined by the system) or user-defined (created by the user for their specific needs).

**3. What aggregate functions do you know?**

* AVG() – returns the average value
* SUM() – returns the sum of values
* MIN() – returns the minimum value
* MAX() – returns the maximum value
* COUNT() – returns the number of rows, including those with null values
* FIRST() – returns the first value from a column
* LAST()– returns the last value from a column

**4. What scalar functions do you know?**

* LEN() (in other SQL flavors – LENGTH()) – returns the length of a string, including the blank spaces
* UCASE() (in other SQL flavors – UPPER()) – returns a string converted to the upper case
* LCASE() (in other SQL flavors – LOWER()) – returns a string converted to the lower case
* INITCAP() – returns a string converted to the title case (i.e., each word of the string starts from a capital letter)
* MID() (in other SQL flavors – SUBSTR()) – extracts a substring from a string
* ROUND() – returns the numerical value rounded to a specified number of decimals
* NOW() – returns the current date and time

**5. What are case manipulation functions? Give some examples.**

Case manipulation functions represent a subset of character functions, and they're used to change the case of the text data. With these functions, we can convert the data into the upper, lower, or title case.

* UCASE() (in other SQL flavors – UPPER()) – returns a string converted to the upper case
* LCASE() (in other SQL flavors – LOWER()) – returns a string converted to the lower case
* INITCAP() – returns a string converted to the title case (i.e., each word of the string starts from a capital letter)

**6. What are character manipulation functions? Give some examples.**

Character manipulation functions represent a subset of character functions, and they're used to modify the text data.

* CONCAT() – joins two or more string values appending the second string to the end of the first one
* SUBSTR() – returns a part of a string satisfying the provided start and end points
* LENGTH() (in other SQL flavors – LEN()) – returns the length of a string, including the blank spaces
* REPLACE() – replaces all occurrences of a defined substring in a provided string with another substring
* INSTR() – returns the numeric position of a defined substring in a provided string
* LPAD() and RPAD() – return the padding of the left-side/right-side character for right-justified/left-justified value
* TRIM() – removes all the defined characters, as well as white spaces, from the left, right, or both ends of a provided string

**7. What is the difference between local and global variables?**

Local variables can be accessed only inside the function in which they were declared. Instead, global variables, being declared outside any function, are stored in fixed memory structures and can be used throughout the entire program.

**8. What is the default data ordering with the ORDER BY statement, and how do you change it?**

By default, the order is ascending. To change it to descending, we need to add the DESC keyword as follows:

SELECT \* FROM table\_name

ORDER BY col\_1 DESC;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**9. What set operators do you know?**

* UNION – returns the records obtained by at least one of two queries (excluding duplicates)
* UNION ALL – returns the records obtained by at least one of two queries (including duplicates)
* INTERSECT – returns the records obtained by both queries
* EXCEPT (called MINUS in MySQL and Oracle) – returns only the records obtained by the first query but not the second one

**10. What operator is used in the query for pattern matching?**

The LIKE operator in combination with the % and \_ wildcards. The % wildcard represents any number of characters including zero, while \_ – strictly one character.

**11. What is the difference between a primary key and a unique key?**

While both types of keys ensure unique values in a column of a table, the first one identifies uniquely each record of the table, and the second one prevents duplicates in that column.

**12. What is a composite primary key?**

The primary key of a table, based on multiple columns.

**13. What is the order of appearance of the common statements in the SELECT query?**

SELECT – FROM – JOIN – ON – WHERE – GROUP BY – HAVING – ORDER BY – LIMIT

**14. In which order the interpreter executes the common statements in the SELECT query?**

FROM – JOIN – ON – WHERE – GROUP BY – HAVING – SELECT – ORDER BY – LIMIT

**15. What is a view, and why use it?**

A virtual table containing a subset of data retrieved from one or more database tables (or other views). Views take very little space, simplify complex queries, limit access to the data for security reasons, enable data independence, and summarize data from multiple tables.

**16. Can we create a view based on another view?**

Yes. This is also known as nested views. However, we should avoid nesting multiple views since the code becomes difficult to read and debug.

**17. Can we still use a view if the original table is deleted?**

No. Any views based on that table will become invalid after deleting the base table. If we try to use such a view anyway, we'll receive an error message.

**18. What types of SQL relationships do you know?**

* **One-to-one** – each record in one table corresponds to only one record in another table
* **One-to-many** – each record in one table corresponds to several records in another table
* **Many-to-many** – each record in both tables corresponds to several records in another table

**19. What are the possible values of a BOOLEAN data field?**

In some SQL flavors, such as PostgreSQL, the BOOLEAN data type exists explicitly and takes values TRUE, FALSE, or NULL. In other flavors, such as Microsoft SQL Server, the BIT datatype is used to store Boolean values as integers 1 (true) or 0 (false).

**20. What is normalization in SQL, and why use it?**

Normalization is a process of database design that includes organizing and restructuring data in a way to reduce data redundancy, dependency, duplication, and inconsistency. This leads to enhanced data integrity, more tables within the database, more efficient data access and security control, and greater query flexibility.

**21. What is denormalization in SQL, and why use it?**

Denormalization is the process opposite of normalization: it introduces data redundancy and combines data from multiple tables. Denormalization optimizes the performance of the database infrastructure in situations when read operations are more important than write operations since it helps avoid complex joins and reduces the time of query running.

**22. What is the difference between renaming a column and giving an alias to it?**

Renaming a column means permanently changing its actual name in the original table. Giving an alias to a column means giving it a temporary name while executing an SQL query, with the purpose to make the code more readable and compact.

**23. What is the difference between nested and correlated subqueries?**

A correlated subquery is an inner query nested in a bigger (outer) query that refers to the values from the outer query for its execution, meaning that a correlated subquery depends on its outer query. Instead, a non-correlated subquery doesn't rely on the data from the outer query and can be run independently of it.

**24. What is the difference between clustered and non-clustered indexes?**

While a clustered index **defines the physical order of records** of a table and performs data searching based on the key values, a non-clustered index **keeps the order of records that doesn't match the physical order of the actual data** on the disk. A table can have only one clustered index but many non-clustered ones.

**25. What is the CASE() function?**

The way to implement the *if-then-else* logic in SQL. This function sequentially checks the provided conditions in the WHEN clauses and returns the value from the corresponding THEN clause when the first condition is satisfied. If none of the conditions is satisfied, the function returns the value from the ELSE clause in case it's provided, otherwise, it returns NULL. The syntax is:

CASE

WHEN condition\_1 THEN value\_1

WHEN condition\_2 THEN value\_2

WHEN condition\_3 THEN value\_3

...

ELSE value

END;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**26. What is the difference between the DELETE and TRUNCATE statements?**

DELETE is a reversible DML (Data Manipulation Language) command used to delete one or more rows from a table based on the conditions specified in the WHERE clause. Instead, TRUNCATE is an irreversible DDL (Data Definition Language) command used to delete all rows from a table. DELETE works slower than TRUNCATE. Also, we can't use the TRUNCATE statement for a table containing a foreign key.

**27. What is the difference between the DROP and TRUNCATE statements?**

DROP deletes a table from the database completely, including the table structure and all the associated constraints, relationships with other tables, and access privileges. TRUNCATE deletes all rows from a table without affecting the table structure and constraints. DROP works slower than TRUNCATE. Both are irreversible DDL (Data Definition Language) commands.

**28. What is the difference between the HAVING and WHERE statements?**

The first one works on aggregated data after they are grouped, while the second one checks each row individually. If both statements are present in a query, they appear in the following order: WHERE – GROUP BY – HAVING. The SQL engine interprets them also in the same order.

**29. How do you add a record to a table?**

Using the INSERT INTO statement in combination with VALUES. The syntax is:

INSERT INTO table\_name

VALUES (value\_1, value\_2, ...);

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**30. How to delete a record from a table?**

Using the DELETE statement. The syntax is:

DELETE FROM table\_name

WHERE condition;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

In this way, we can also delete multiple records if they satisfy the provided condition.

**31. How to add a column to a table?**

Using the ALTER TABLE statement in combination with ADD. The syntax is:

ALTER TABLE table\_name

ADD column\_name datatype;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**32. How to rename a column of a table?**

Using the ALTER TABLE statement in combination with RENAME COLUMN ... TO ... The syntax is:

ALTER TABLE table\_name

RENAME COLUMN old\_column\_name TO new\_column\_name;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**33. How to delete a column from a table?**

Using the ALTER TABLE statement in combination with DROP COLUMN. The syntax is:

ALTER TABLE table\_name

DROP COLUMN column\_name;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**34. How to select all even or all odd records in a table?**

By checking the remainder of the division by 2. In some SQL versions (e.g., PostgreSQL and My SQL), we use the MOD function, in the others (Microsoft SQL Server and SQLite) – the modulo operator (%). To select all even records using MOD:

SELECT \* FROM table\_name

WHERE MOD(ID\_column, 2) = 0;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

To select all even records using %:

SELECT \* FROM table\_name

WHERE ID\_column % 2 = 0;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

To select all odd records, the syntax is identical in both cases, only that we would use the inequality operator <> instead of =.

**35. How to prevent duplicate records when making a query?**

Using the DISTINCT statement in combination with SELECT or creating a unique key for that table.

**36. How to insert many rows in a table?**

Using the INSERT INTO statement in combination with VALUES. The syntax is:

INSERT INTO table\_name

VALUES (value\_1, value\_2, ...),

(value\_3, value\_4, ...),

(value\_5, value\_6, ...),

...;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**37. How to find the nth highest value in a column of a table?**

Using the OFFSET clause. For example, to find the 6th highest value from a column, we would use the following syntax:

SELECT \* FROM table\_name

ORDER BY column\_name DESC

LIMIT 1

OFFSET 5;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**38. How to find the values in a text column of a table that start with a certain letter?**

Using the LIKE operator in combination with the % and \_ wildcards. For example, we need to find all surnames in a table that start with "A". The query is:

SELECT \* FROM table\_name

WHERE surname LIKE 'A\_';

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

Here, we assume that a surname must contain at least two letters. Without this assumption (meaning that a surname can be just A), the query is as follows:

SELECT \* FROM table\_name

WHERE surname LIKE 'A%';

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**39. How to find the last id in a table?**

Using the MAX() function. Otherwise, in many SQL versions, we can use the following syntax:

SELECT id

FROM table\_name

ORDER BY id DESC

LIMIT 1;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

or in Microsoft SQL Server:

SELECT TOP 1 id

FROM table\_name

ORDER BY id DESC

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**40. How to select random rows from a table?**

Using the RAND() function in combination with ORDER BY and LIMIT. In some SQL flavors, such as PostgreSQL, it's called RANDOM(). For example, the following code will return five random rows from a table in MySQL:

SELECT \* FROM table\_name

ORDER BY RAND()

LIMIT 5;

[**POWERED BY DATACAMP WORKSPACE**](https://www.datacamp.com/workspace)

**Conclusion**

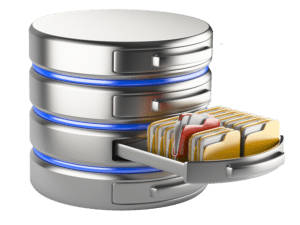
### ****Q1. What is the difference between SQL and MySQL?****

|  |  |
| --- | --- |
| **SQL vs MySQL** | |
| **SQL** | **MySQL** |
| SQL is a standard language which stands for Structured Query Language based on the English language | MySQL is a database management system. |
| SQL is the core of the relational database which is used for accessing and managing database | MySQL is an RDMS (Relational Database Management System) such as SQL Server, Informix etc. |

### ****Q2. What are the different subsets of SQL?****

* Data Definition Language (DDL) – It allows you to perform various operations on the database such as CREATE, ALTER, and DELETE objects.
* Data Manipulation Language(DML) – It allows you to access and manipulate data. It helps you to insert, update, delete and retrieve data from the database.
* Data Control Language(DCL) – It allows you to control access to the database. Example – Grant, Revoke access permissions.

### ****Q3. What do you mean by DBMS? What are its different types?****

A [**Database Management System**](https://www.edureka.co/blog/dbms-tutorial/) (**DBMS**) is a  software application that interacts with the user, applications, and the database itself to capture and analyze data. A database is a structured collection of data.

A DBMS allows a user to interact with the database. The data stored in the database can be modified, retrieved and deleted and can be of any type like strings, numbers, images, etc.

There are two types of DBMS:

* Relational Database Management System: The data is stored in relations (tables). Example – MySQL.
* Non-Relational Database Management System: There is no concept of relations, tuples and attributes.  Example – MongoDB

Let’s move to the next question in this SQL Interview Questions.

### ****Q4. What is RDBMS? How is it different from DBMS?****

A relational database management system (RDBMS) is a set of applications and features that allow IT professionals and others to develop, edit, administer, and interact with relational databases. Most commercial relational database management systems use Structured Query Language (SQL) to access the database, which is stored in the form of tables.  
The RDBMS is the most widely used database system in businesses all over the world. It offers a stable means of storing and retrieving massive amounts of data.

Databases, in general, hold collections of data that may be accessed and used in other applications. The development, administration, and use of database platforms are all supported by a database management system.

A relational database management system (RDBMS) is a type of database management system (DBMS) that stores data in a row-based table structure that links related data components. An RDBMS contains functions that ensure the data’s security, accuracy, integrity, and consistency. This is not the same as the file storage utilized by a database management system.

The following are some further distinctions between database management systems and relational database management systems:

**The number of users who are permitted to utilise the system**  
A DBMS can only handle one user at a time, whereas an RDBMS can handle numerous users.  
**Hardware and software specifications**  
In comparison to an RDBMS, a DBMS requires fewer software and hardware.  
**Amount of information**  
RDBMSes can handle any quantity of data, from tiny to enormous, whereas DBMSes are limited to small amounts.  
**The structure of the database**  
Data is stored in a hierarchical format in a DBMS, whereas an RDBMS uses a table with headers that serve as column names and rows that hold the associated values.  
**Implementation of the ACID principle**  
The atomicity, consistency, isolation, and durability (ACID) concept is not used by DBMSs for data storage. RDBMSes, on the other hand, use the ACID model to organize their data and assure consistency.  
**Databases that are distributed**  
A DBMS will not provide complete support for distributed databases, whereas an RDBMS will.  
**Programs that are managed**  
A DBMS focuses on keeping databases that are present within the computer network and system hard discs, whereas an RDBMS helps manage relationships between its incorporated tables of data.  
**Normalization of databases is supported**  
A RDBMS can be normalized , but a DBMS cannot be normalized.

### Q5. What is a Self-Join?

A self-join is a type of join that can be used to connect two tables. As a result, it is a unary relationship. Each row of the table is attached to itself and all other rows of the same table in a self-join. As a result, a self-join is mostly used to combine and compare rows from the same database table.

### Q6. What is the SELECT statement?

A SELECT command gets zero or more rows from one or more database tables or views. The most frequent data manipulation language (DML) command is SELECT in most applications. SELECT queries define a result set, but not how to calculate it, because SQL is a declarative programming language.

### Q7. What are some common clauses used with SELECT query in SQL?

The following are some frequent SQL clauses used in conjunction with a SELECT query:

**WHERE** clause: In SQL, the WHERE clause is used to filter records that are required depending on certain criteria.  
**ORDER BY** clause: The ORDER BY clause in SQL is used to sort data in ascending (ASC) or descending (DESC) order depending on specified field(s) (DESC).  
**GROUP BY** clause: GROUP BY clause in SQL is used to group entries with identical data and may be used with aggregation methods to obtain summarised database results.  
**HAVING** clause in SQL is used to filter records in combination with the GROUP BY clause. It is different from WHERE, since the WHERE clause cannot filter aggregated records.

### Q8. What are UNION, MINUS and INTERSECT commands?

The UNION operator is used to combine the results of two tables while also removing duplicate entries.  
The MINUS operator is used to return rows from the first query but not from the second query.  
The INTERSECT operator is used to combine the results of both queries into a single row.  
Before running either of the above SQL statements, certain requirements must be satisfied –  
Within the clause, each SELECT query must have the same amount of columns.  
The data types in the columns must also be comparable.  
In each SELECT statement, the columns must be in the same order.

### Q9. What is Cursor? How to use a Cursor?

After any variable declaration, DECLARE a cursor. A SELECT Statement must always be coupled with the cursor definition.

To start the result set, move the cursor over it. Before obtaining rows from the result set, the OPEN statement must be executed.

To retrieve and go to the next row in the result set, use the FETCH command.

To disable the cursor, use the CLOSE command.

Finally, use the DEALLOCATE command to remove the cursor definition and free up the resources connected with it.

### Q10. List the different types of relationships in SQL.

**There are different types of relations in the database:**

**One-to-One** – This is a connection between two tables in which each record in one table corresponds to the maximum of one record in the other.

**One-to-Many and Many-to-One** – This is the most frequent connection, in which a record in one table is linked to several records in another.

**Many-to-Many** – This is used when defining a relationship that requires several instances on each sides.

**Self-Referencing Relationships** – When a table has to declare a connection with itself, this is the method to employ.

### Q12. What is OLTP?

OLTP, or online transactional processing, allows huge groups of people to execute massive amounts of database transactions in real time, usually via the internet. A database transaction occurs when data in a database is changed, inserted, deleted, or queried.

### Q13. What are the differences between OLTP and OLAP?

OLTP stands for online transaction processing, whereas OLAP stands for online analytical processing. OLTP is an online database modification system, whereas OLAP is an online database query response system.

### Q14. How to create empty tables with the same structure as another table?

To create empty tables:  
Using the INTO operator to fetch the records of one table into a new table while setting a WHERE clause to false for all entries, it is possible to create empty tables with the same structure. As a result, SQL creates a new table with a duplicate structure to accept the fetched entries, but nothing is stored into the new table since the WHERE clause is active.

### Q15. What is PostgreSQL?

In 1986, a team lead by Computer Science Professor Michael Stonebraker created PostgreSQL under the name Postgres. It was created to aid developers in the development of enterprise-level applications by ensuring data integrity and fault tolerance in systems. PostgreSQL is an enterprise-level, versatile, resilient, open-source, object-relational database management system that supports variable workloads and concurrent users. The international developer community has constantly backed it. PostgreSQL has achieved significant appeal among developers because to its fault-tolerant characteristics.  
It’s a very reliable database management system, with more than two decades of community work to thank for its high levels of resiliency, integrity, and accuracy. Many online, mobile, geospatial, and analytics applications utilise PostgreSQL as their primary data storage or data warehouse.

### ****Q16. What are SQL comments?****

SQL Comments are used to clarify portions of SQL statements and to prevent SQL statements from being executed. Comments are quite important in many programming languages. The comments are not supported by a Microsoft Access database. As a result, the Microsoft Access database is used in the examples in Mozilla Firefox and Microsoft Edge.  
Single Line Comments: It starts with two consecutive hyphens (–).  
Multi-line Comments: It starts with /\* and ends with \*/.

### Q17. What is the usage of the NVL() function?

You may use the NVL function to replace null values with a default value. The function returns the value of the second parameter if the first parameter is null. If the first parameter is anything other than null, it is left alone.

This function is used in Oracle, not in SQL and MySQL. Instead of NVL() function, MySQL have IFNULL() and SQL Server have ISNULL() function.

Let’s move to the next question in this SQL Interview Questions.

### Q18. Explain character-manipulation functions? Explains its different types in SQL.

Change, extract, and edit the character string using character manipulation routines. The function will do its action on the input strings and return the result when one or more characters and words are supplied into it.

The character manipulation functions in SQL are as follows:

A) CONCAT (joining two or more values): This function is used to join two or more values together. The second string is always appended to the end of the first string.

B) SUBSTR: This function returns a segment of a string from a given start point to a given endpoint.

C) LENGTH: This function returns the length of the string in numerical form, including blank spaces.

D) INSTR: This function calculates the precise numeric location of a character or word in a string.

E) LPAD: For right-justified values, it returns the padding of the left-side character value.

F) RPAD: For a left-justified value, it returns the padding of the right-side character value.

G) TRIM: This function removes all defined characters from the beginning, end, or both ends of a string. It also reduced the amount of wasted space.

H) REPLACE: This function replaces all instances of a word or a section of a string (substring) with the other string value specified.

### Q19. Write the SQL query to get the third maximum salary of an employee from a table named employees.

Employee table

|  |  |
| --- | --- |
| employee\_name | salary |
| A | 24000 |
| C | 34000 |
| D | 55000 |
| E | 75000 |
| F | 21000 |
| G | 40000 |
| H | 50000 |

SELECT \* FROM(

SELECT employee\_name, salary, DENSE\_RANK()

OVER(ORDER BY salary DESC)r FROM Employee)

WHERE r=&n;

To find 3rd highest salary set n = 3

### ****Q20. What is the difference between the RANK() and DENSE\_RANK() functions?****

The RANK() function in the result set defines the rank of each row within your ordered partition. If both rows have the same rank, the next number in the ranking will be the previous rank plus a number of duplicates. If we have three records at rank 4, for example, the next level indicated is 7.

The DENSE\_RANK() function assigns a distinct rank to each row within a partition based on the provided column value, with no gaps. It always indicates a ranking in order of precedence. This function will assign the same rank to the two rows if they have the same rank, with the next rank being the next consecutive number. If we have three records at rank 4, for example, the next level indicated is 5.

**Q21. What are Tables and Fields?**

A table is a collection of data components organized in rows and columns in a relational database. A table can also be thought of as a useful representation of relationships. The most basic form of data storage is the table. An example of an Employee table is shown below.

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Name | Department | Salary |
| 1 | Rahul | Sales | 24000 |
| 2 | Rohini | Marketing | 34000 |
| 3 | Shylesh | Sales | 24000 |
| 4 | Tarun | Analytics | 30000 |

A Record or Row is a single entry in a table. In a table, a record represents a collection of connected data. The Employee table, for example, has four records.

A table is made up of numerous records (rows), each of which can be split down into smaller units called Fields(columns). ID, Name, Department, and Salary are the four fields in the Employee table above.

### Q22. What is a UNIQUE constraint?

The UNIQUE Constraint prevents identical values in a column from appearing in two records. The UNIQUE constraint guarantees that every value in a column is unique.

### Q23. What is a Self-Join?

A self-join is a type of join that can be used to connect two tables. As a result, it is a unary relationship. Each row of the table is attached to itself and all other rows of the same table in a self-join. As a result, a self-join is mostly used to combine and compare rows from the same database table.

### Q24. What is the SELECT statement?

A SELECT command gets zero or more rows from one or more database tables or views. The most frequent data manipulation language (DML) command is SELECT in most applications. SELECT queries define a result set, but not how to calculate it, because SQL is a declarative programming language.

### Q25. What are some common clauses used with SELECT query in SQL?

The following are some frequent SQL clauses used in conjunction with a SELECT query:

[[](https://www.edureka.co/microsoft-sql-server-certification-training?utm_source=blogbanner&utm_campaign=curriculum)](https://www.edureka.co/microsoft-sql-server-certification-training?utm_source=blogbanner&utm_campaign=curriculum" \t "_blank)

### [Microsoft SQL Server Certification Course](https://www.edureka.co/microsoft-sql-server-certification-training?utm_source=blogbanner&utm_campaign=curriculum" \t "_blank)

[Explore Curriculum](https://www.edureka.co/microsoft-sql-server-certification-training?utm_source=blogbanner&utm_campaign=curriculum" \t "_blank)

WHERE clause: In SQL, the WHERE clause is used to filter records that are required depending on certain criteria.  
ORDER BY clause: The ORDER BY clause in SQL is used to sort data in ascending (ASC) or descending (DESC) order depending on specified field(s) (DESC).  
GROUP BY clause: GROUP BY clause in SQL is used to group entries with identical data and may be used with aggregation methods to obtain summarised database results.  
HAVING clause in SQL is used to filter records in combination with the GROUP BY clause. It is different from WHERE, since the WHERE clause cannot filter aggregated records.

### Q26. What are UNION, MINUS and INTERSECT commands?

**The UNION operator is used to combine the results of two tables while also removing duplicate entries.**

The MINUS operator is used to return rows from the first query but not from the second query.

The INTERSECT operator is used to combine the results of both queries into a single row.  
Before running either of the above SQL statements, certain requirements must be satisfied –

Within the clause, each SELECT query must have the same amount of columns.

The data types in the columns must also be comparable.

In each SELECT statement, the columns must be in the same order.

Let’s move to the next question in this SQL Interview Questions.

### Q27. What is Cursor? How to use a Cursor?

After any variable declaration, DECLARE a cursor. A SELECT Statement must always be coupled with the cursor definition.

To start the result set, move the cursor over it. Before obtaining rows from the result set, the OPEN statement must be executed.

To retrieve and go to the next row in the result set, use the FETCH command.

To disable the cursor, use the CLOSE command.

Finally, use the DEALLOCATE command to remove the cursor definition and free up the resources connected with it.

### Q28. List the different types of relationships in SQL.

There are different types of relations in the database:  
One-to-One – This is a connection between two tables in which each record in one table corresponds to the maximum of one record in the other.  
One-to-Many and Many-to-One – This is the most frequent connection, in which a record in one table is linked to several records in another.  
Many-to-Many – This is used when defining a relationship that requires several instances on each sides.  
Self-Referencing Relationships – When a table has to declare a connection with itself, this is the method to employ.

### ****Q29. What is SQL example?****

SQL is a database query language that allows you to edit, remove, and request data from databases. The following statements are a few examples of SQL statements:

* SELECT
* INSERT
* UPDATE
* DELETE
* CREATE DATABASE
* ALTER DATABASE

### Q30. What are basic SQL skills?

SQL skills aid data analysts in the creation, maintenance, and retrieval of data from relational databases, which divide data into columns and rows. It also enables users to efficiently retrieve, update, manipulate, insert, and alter data.

The most fundamental abilities that a SQL expert should possess are:

1. Database Management
2. Structuring a Database
3. Creating SQL clauses and statements
4. SQL System SKills like MYSQL, PostgreSQL
5. PHP expertise is useful.
6. Analyze SQL data
7. Using WAMP with SQL to create a database
8. OLAP Skills

### Q31. What is schema in SQL Server? A schema is a visual representation of the database that is logical. It builds and specifies the relationships among the database’s numerous entities. It refers to the several kinds of constraints that may be applied to a database. It also describes the various data kinds. It may also be used on Tables and Views. Schemas come in a variety of shapes and sizes. Star schema and Snowflake schema are two of the most popular. The entities in a star schema are represented in a star form, whereas those in a snowflake schema are shown in a snowflake shape. Any database architecture is built on the foundation of schemas.

### Q32. How to create a temp table in SQL Server?

Temporary tables are created in TempDB and are erased automatically after the last connection is closed. We may use Temporary Tables to store and process interim results. When we need to store temporary data, temporary tables come in handy.  
  
The following is the syntax for creating a Temporary Table:  
  
CREATE TABLE #Employee (id INT, name VARCHAR(25))  
INSERT INTO #Employee VALUES (01, ‘Ashish’), (02, ‘Atul’)

Let’s move to the next question in this SQL Interview Questions.

**Q33. How to install SQL Server in Windows 11?**

Install SQL Server Management Studio In Windows 11

**Step 1:** Click on SSMS, which will take you to the SQL Server Management Studio page.

**Step 2:** Moreover, click on the SQL Server Management Studio link and tap on Save File.

**Step 3:** Save this file to your local drive and go to the folder.

**Step 4:** The setup window will appear, and here you can choose the location where you want to save the file.  
**Step 5:** Click on Install.  
**Step 6:** Close the window after the installation is complete.  
**Step 7:** Furthermore, go back to your Start Menu and search for SQL server management studio.

**Step 8:** Furthermore, double-click on it, and the login page will appear once it shows up.

**Step 9:** You should be able to see your server name. However, if that’s not visible, click on the drop-down arrow on the server and tap on Browse.

**Step 10:** Choose your SQL server and click on Connect.

After that, the SQL server will connect, and Windows 11 will run good.

### Q34. What is the case when in SQL Server?

The CASE statement is used to construct logic in which one column’s value is determined by the values of other columns.

At least one set of WHEN and THEN commands makes up the SQL Server CASE Statement. The condition to be tested is specified by the WHEN statement. If the WHEN condition returns TRUE, the THEN sentence explains what to do.

When none of the WHEN conditions return true, the ELSE statement is executed. The END keyword brings the CASE statement to a close.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | CASE  WHEN condition1 THEN result1  WHEN condition2 THEN result2  WHEN conditionN THEN resultN  ELSE result  END; |

**Q35. NoSQL vs SQL**

In summary, the following are the five major distinctions between SQL and NoSQL:

Relational databases are SQL, while non-relational databases are NoSQL.

SQL databases have a specified schema and employ structured query language. For unstructured data, NoSQL databases use dynamic schemas.

SQL databases scale vertically, but NoSQL databases scale horizontally.

NoSQL databases are document, key-value, graph, or wide-column stores, whereas SQL databases are table-based.

SQL databases excel in multi-row transactions, while NoSQL excels at unstructured data such as documents and JSON.

**Q36. What is the difference between NOW() and CURRENT\_DATE()?**  
NOW() returns a constant time that indicates the time at which the statement began to execute. (Within a stored function or trigger, NOW() returns the time at which the function or triggering statement began to execute.  
The simple difference between NOW() and CURRENT\_DATE() is that NOW() will fetch the current date and time both in format ‘YYYY-MM\_DD HH:MM:SS’ while CURRENT\_DATE() will fetch the date of the current day ‘YYYY-MM\_DD’.

Let’s move to the next question in this SQL Interview Questions.

**Q37. What is BLOB and TEXT in MySQL?**

BLOB stands for Binary Huge Objects and can be used to store binary data, whereas TEXT may be used to store a large number of strings. BLOB may be used to store binary data, which includes images, movies, audio, and applications.  
BLOB values function similarly to byte strings, and they lack a character set. As a result, bytes’ numeric values are completely dependent on comparison and sorting.  
    TEXT values behave similarly to a character string or a non-binary string. The comparison/sorting of TEXT is completely dependent on the character set collection.

**Q38. How to remove duplicate rows in SQL?**

If the SQL table has duplicate rows, the duplicate rows must be removed.

Let’s assume the following table as our dataset:

|  |  |  |
| --- | --- | --- |
| ID | Name | Age |
| 1 | A | 21 |
| 2 | B | 23 |
| 2 | B | 23 |
| 4 | D | 22 |
| 5 | E | 25 |
| 6 | G | 26 |
| 5 | E | 25 |

The following SQL query removes the duplicate ids from the  table:  
  
DELETE FROM table WHERE ID IN (  
SELECT   
ID, COUNT(ID)   
FROM   table  
GROUP BY  ID  
HAVING   
COUNT (ID) > 1);

**Q39. How to create a stored procedure using SQL Server?**

A stored procedure is a piece of prepared SQL code that you can save and reuse again and over.  
So, if you have a SQL query that you create frequently, save it as a stored procedure and then call it to run it.  
You may also supply parameters to a stored procedure so that it can act based on the value(s) of the parameter(s) given.

Stored Procedure Syntax

CREATE PROCEDURE procedure\_name

AS

sql\_statement

GO;

Execute a Stored Procedure

EXEC procedure\_name;

**Q40. What is Database Black Box Testing?**

Black Box Testing is a software testing approach that involves testing the functions of software applications without knowing the internal code structure, implementation details, or internal routes. Black Box Testing is a type of software testing that focuses on the input and output of software applications and is totally driven by software requirements and specifications. Behavioral testing is another name for it.  
  
  
**Q41. What are the different types of SQL sandbox?**

### Databases Training

### [MICROSOFT SQL SERVER CERTIFICATION COURSE](https://www.edureka.co/microsoft-sql-server-certification-training?utm_source=blogbanner&utm_campaign=trending_course" \o "Microsoft SQL Server Certification Course" \t "_blank)

### [Microsoft SQL Server Certification Course](https://www.edureka.co/microsoft-sql-server-certification-training?utm_source=blogbanner&utm_campaign=trending_course" \o "Microsoft SQL Server Certification Course" \t "_blank)

*[Reviews](https://www.edureka.co/microsoft-sql-server-certification-training?utm_source=blogbanner&utm_campaign=trending_course" \o "Microsoft SQL Server Certification Course" \t "_blank)*

**[5](https://www.edureka.co/microsoft-sql-server-certification-training?utm_source=blogbanner&utm_campaign=trending_course" \o "Microsoft SQL Server Certification Course" \t "_blank)**[(2608)](https://www.edureka.co/microsoft-sql-server-certification-training?utm_source=blogbanner&utm_campaign=trending_course" \o "Microsoft SQL Server Certification Course" \t "_blank)

### [SQL ESSENTIALS TRAINING](https://www.edureka.co/sql-essentials-training?utm_source=blogbanner&utm_campaign=trending_course" \o "SQL Essentials Training" \t "_blank)

### [SQL Essentials Training](https://www.edureka.co/sql-essentials-training?utm_source=blogbanner&utm_campaign=trending_course" \o "SQL Essentials Training" \t "_blank)

*[Reviews](https://www.edureka.co/sql-essentials-training?utm_source=blogbanner&utm_campaign=trending_course" \o "SQL Essentials Training" \t "_blank)*

**[5](https://www.edureka.co/sql-essentials-training?utm_source=blogbanner&utm_campaign=trending_course" \o "SQL Essentials Training" \t "_blank)**[(11076)](https://www.edureka.co/sql-essentials-training?utm_source=blogbanner&utm_campaign=trending_course" \o "SQL Essentials Training" \t "_blank)

### [MONGODB CERTIFICATION TRAINING COURSE](https://www.edureka.co/mongodb-certification-training?utm_source=blogbanner&utm_campaign=trending_course" \o "MongoDB Certification Training Course" \t "_blank)

### [MongoDB Certification Training Course](https://www.edureka.co/mongodb-certification-training?utm_source=blogbanner&utm_campaign=trending_course" \o "MongoDB Certification Training Course" \t "_blank)

*[Reviews](https://www.edureka.co/mongodb-certification-training?utm_source=blogbanner&utm_campaign=trending_course" \o "MongoDB Certification Training Course" \t "_blank)*

**[4](https://www.edureka.co/mongodb-certification-training?utm_source=blogbanner&utm_campaign=trending_course" \o "MongoDB Certification Training Course" \t "_blank)**[(16725)](https://www.edureka.co/mongodb-certification-training?utm_source=blogbanner&utm_campaign=trending_course" \o "MongoDB Certification Training Course" \t "_blank)

### [MYSQL DBA CERTIFICATION TRAINING](https://www.edureka.co/mysql-dba?utm_source=blogbanner&utm_campaign=trending_course" \o "MySQL DBA Certification Training" \t "_blank)

### [MySQL DBA Certification Training](https://www.edureka.co/mysql-dba?utm_source=blogbanner&utm_campaign=trending_course" \o "MySQL DBA Certification Training" \t "_blank)

*[Reviews](https://www.edureka.co/mysql-dba?utm_source=blogbanner&utm_campaign=trending_course" \o "MySQL DBA Certification Training" \t "_blank)*

**[5](https://www.edureka.co/mysql-dba?utm_source=blogbanner&utm_campaign=trending_course" \o "MySQL DBA Certification Training" \t "_blank)**[(6762)](https://www.edureka.co/mysql-dba?utm_source=blogbanner&utm_campaign=trending_course" \o "MySQL DBA Certification Training" \t "_blank)

### [APACHE CASSANDRA CERTIFICATION TRAINING](https://www.edureka.co/cassandra?utm_source=blogbanner&utm_campaign=trending_course" \o "Apache Cassandra Certification Training" \t "_blank)

### [Apache Cassandra Certification Training](https://www.edureka.co/cassandra?utm_source=blogbanner&utm_campaign=trending_course" \o "Apache Cassandra Certification Training" \t "_blank)

*[Reviews](https://www.edureka.co/cassandra?utm_source=blogbanner&utm_campaign=trending_course" \o "Apache Cassandra Certification  Training" \t "_blank)*

**[5](https://www.edureka.co/cassandra?utm_source=blogbanner&utm_campaign=trending_course" \o "Apache Cassandra Certification  Training" \t "_blank)**[(12818)](https://www.edureka.co/cassandra?utm_source=blogbanner&utm_campaign=trending_course" \o "Apache Cassandra Certification  Training" \t "_blank)

### [TERADATA CERTIFICATION TRAINING](https://www.edureka.co/teradata?utm_source=blogbanner&utm_campaign=trending_course" \o "Teradata Certification Training" \t "_blank)

### [Teradata Certification Training](https://www.edureka.co/teradata?utm_source=blogbanner&utm_campaign=trending_course" \o "Teradata Certification Training" \t "_blank)

*[Reviews](https://www.edureka.co/teradata?utm_source=blogbanner&utm_campaign=trending_course" \o "Teradata Certification Training" \t "_blank)*

**[5](https://www.edureka.co/teradata?utm_source=blogbanner&utm_campaign=trending_course" \o "Teradata Certification Training" \t "_blank)**[(2923)](https://www.edureka.co/teradata?utm_source=blogbanner&utm_campaign=trending_course" \o "Teradata Certification Training" \t "_blank)

Next

SQL Sandbox is a secure environment within SQL Server where untrusted programmes can be run. There are three different types of SQL sandboxes:

Safe Access Sandbox: In this environment, a user may execute SQL activities like as building stored procedures, triggers, and so on, but they can’t access the memory or create files.

Sandbox for External Access: Users can access files without having the ability to alter memory allocation.

Unsafe Access Sandbox: This contains untrustworthy code that allows a user to access memory.

Let’s move to the next question in this SQL Interview Questions.

**Q42. Where MyISAM table is stored?**

Prior to the introduction of MySQL 5.5 in December 2009, MyISAM was the default storage engine for MySQL relational database management system versions.  It’s based on the older ISAM code, but it comes with a lot of extra features. Each MyISAM table is split into three files on disc (if it is not partitioned). The file names start with the table name and end with an extension that indicates the file type. The table definition is stored in a.frm file, however this file is not part of the MyISAM engine; instead, it is part of the server. The data file’s suffix is.MYD (MYData). The index file’s extension is.MYI (MYIndex). If you lose your index file, you may always restore it by recreating indexes.  
  
**Q43. How to find the nth highest salary in SQL?**  
The most typical interview question is to find the Nth highest pay in a table. This work can be accomplished using the dense rank() function.  
Employee table

|  |  |
| --- | --- |
| employee\_name | salary |
| A | 24000 |
| C | 34000 |
| D | 55000 |
| E | 75000 |
| F | 21000 |
| G | 40000 |
| H | 50000 |

SELECT \* FROM(

SELECT employee\_name, salary, DENSE\_RANK()

OVER(ORDER BY salary DESC)r FROM Employee)

WHERE r=&n;

To find to the 2nd highest salary set n = 2

To find 3rd highest salary set n = 3 and so on.

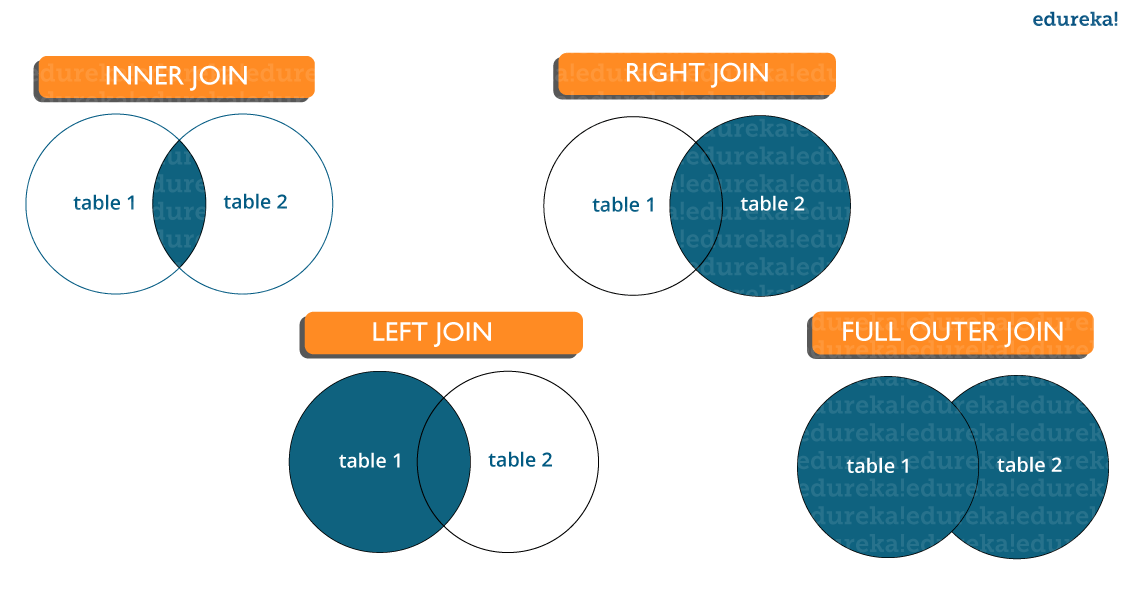
### ****Q44. What do you mean by table and field in SQL?****

A table refers to a collection of data in an organised manner in form of rows and columns. A field refers to the number of columns in a table. For example:

***Table***: StudentInformation  
***Field***: Stu Id, Stu Name, Stu Marks

### ****Q45. What are joins in SQL?****

A JOIN clause is used to combine rows from two or more tables, based on a related column between them. It is used to merge two tables or retrieve data from there. There are 4 types of joins, as you can refer to below:

****

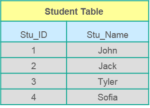
* **Inner join:** [Inner Join in SQL](https://www.edureka.co/blog/sql-joins-types) is the most common type of join. It is used to return all the rows from multiple tables where the join condition is satisfied.
* **Left Join:**  Left Join in SQL is used to return all the rows from the left table but only the matching rows from the right table where the join condition is fulfilled.
* **Right Join:** Right Join in SQL is used to return all the rows from the right table but only the matching rows from the left table where the join condition is fulfilled.
* **Full Join:** Full join returns all the records when there is a match in any of the tables. Therefore, it returns all the rows from the left-hand side table and all the rows from the right-hand side table.

Let’s move to the next question in this SQL Interview Questions.

### ****Q46.**** ****What is the difference between CHAR and VARCHAR2 datatype in SQL?****

Both Char and Varchar2 are used for characters datatype but varchar2 is used for character strings of variable length whereas Char is used for strings of fixed length. For example, char(10) can only store 10 characters and will not be able to store a string of any other length whereas varchar2(10) can store any length i.e 6,8,2 in this variable.

### ****Q47. What is a Primary key?****

* A[Primary key in SQL](https://www.edureka.co/blog/primary-key-in-sql/)is a column (or collection of columns) or a set of columns that uniquely identifies each row in the table.
* Uniquely identifies a single row in the table
* Null values not allowed

Example- In the Student table, Stu\_ID is the primary key.

### ****Q48.**** ****What are Constraints?****

[Constraints in SQL](https://www.edureka.co/blog/sql-constraints/) are used to specify the limit on the data type of the table. It can be specified while creating or altering the table statement. The sample of constraints are:

* NOT NULL
* CHECK
* DEFAULT
* UNIQUE
* PRIMARY KEY
* FOREIGN KEY

### ****Q49. What is the difference between DELETE and TRUNCATE statements?****

|  |  |
| --- | --- |
| **DELETE vs TRUNCATE** | |
| **DELETE** | **TRUNCATE** |
| Delete command is used to delete a row in a table. | Truncate is used to delete all the rows from a table. |
| You can rollback data after using delete statement. | You cannot rollback data. |
| It is a DML command. | It is a DDL command. |
| It is slower than truncate statement. | It is faster. |

### ****Q50. What is a Unique key?****

* Uniquely identifies a single row in the table.
* Multiple values allowed per table.
* Null values allowed.

Apart from this SQL Interview Questions blog, if you want to get trained from professionals on this technology, you can opt for [structured training from edureka!](https://www.edureka.co/sql-essentials-training)

### ****Q51. What is a Foreign key in SQL?****

* Foreign key maintains referential integrity by enforcing a link between the data in two tables.
* The foreign key in the child table references the primary key in the parent table.
* The [foreign key constraint](https://www.edureka.co/blog/foreign-key-sql/) prevents actions that would destroy links between the child and parent tables.

### ****Q52. What do you mean by data integrity?****

Data Integrity defines the accuracy as well as the consistency of the data stored in a database. It also defines integrity constraints to enforce business rules on the data when it is entered into an application or a database.

### ****Q53. What is the difference between clustered and non-clustered index in SQL?****

The differences between the clustered and non clustered index in SQL are :

1. Clustered index is used for easy retrieval of data from the database and its faster whereas reading from non clustered index is relatively slower.
2. Clustered index alters the way records are stored in a database as it sorts out rows by the column which is set to be clustered index whereas in a non clustered index, it does not alter the way it was stored but it creates a separate object within a table which points back to the original table rows after searching.
3. One table can only have one clustered index whereas it can have many non clustered index.

### ****Q54. Write a SQL query to display the current date?****

In SQL, there is a built-in function called **GetDate()** which helps to return the current timestamp/date.

### ****Q55.What do you understand by query optimization?****

The phase that identifies a plan for evaluation query which has the least estimated cost is known as query optimization.

The advantages of query optimization are as follows:

* The output is provided faster
* A larger number of queries can be executed in less time
* Reduces time and space complexity

**Find out our MS SQL Course in Top Cities**

|  |  |
| --- | --- |
| India | India |
| [SQL Training in Bangalore](https://www.edureka.co/microsoft-sql-server-certification-training-bangalore) | [SQL Course in Pune](https://www.edureka.co/microsoft-sql-server-certification-training-pune) |
| [SQL Training in Chennai](https://www.edureka.co/microsoft-sql-server-certification-training-chennai) | [SQL Course in Mumbai](https://www.edureka.co/microsoft-sql-server-certification-training-mumbai) |
| [SQL Training in hyderabad](https://www.edureka.co/microsoft-sql-server-certification-training-hyderabad) | [SQL Course in Kolkata](https://www.edureka.co/microsoft-sql-server-certification-training-kolkata) |

Denormalization refers to a technique which is used to access data from higher to lower forms of a database. It helps the database managers to increase the performance of the entire infrastructure as it introduces redundancy into a table. It adds the redundant data into a table by incorporating database queries that combine data from various tables into a single table.

### ****Q57. What are Entities and Relationships?****

**Entities**:  A person, place, or thing in the real world about which data can be stored in a database. Tables store data that represents one type of entity. For example – A bank database has a customer table to store customer information. The customer table stores this information as a set of attributes (columns within the table) for each customer.

**Relationships**: Relation or links between entities that have something to do with each other. For example – The customer name is related to the customer account number and contact information, which might be in the same table. There can also be relationships between separate tables (for example, customer to accounts).

Let’s move to the next question in this SQL Interview Questions.

### ****Q58. What is an Index?****

An index refers to a performance tuning method of allowing faster retrieval of records from the table. An index creates an entry for each value and hence it will be faster to retrieve data.

### ****Q59. Explain different types of index in SQL.****

There are three [types of index in SQL](https://www.edureka.co/blog/index-in-sql/) namely:

### ****Unique Index:****

This index does not allow the field to have duplicate values if the column is unique indexed. If a primary key is defined, a unique index can be applied automatically.

### ****Clustered Index:****

This index reorders the physical order of the table and searches based on the basis of key values. Each table can only have one clustered index.

#### **Non-Clustered Index:**

Non-Clustered Index does not alter the physical order of the table and maintains a logical order of the data. Each table can have many nonclustered indexes.

### ****Q60. What is Normalization and what are the advantages of it?****

[Normalization in SQL](https://www.edureka.co/blog/normalization-in-sql/) is the process of organizing data to avoid duplication and redundancy. Some of the advantages are:

* Better Database organization
* More Tables with smaller rows
* Efficient data access
* Greater Flexibility for Queries
* Quickly find the information
* Easier to implement Security
* Allows easy modification
* Reduction of redundant and duplicate data
* More Compact Database
* Ensure Consistent data after modification

Apart from this SQL Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for [structured training from edureka!](https://www.edureka.co/sql-essentials-training)

### ****Q61. What is the difference between DROP and TRUNCATE commands?****

[DROP command](https://www.edureka.co/blog/sql-commands) removes a table and it cannot be rolled back from the database whereas TRUNCATE command removes all the rows from the table.

### ****Q62. Explain different types of Normalization.****

There are many successive levels of normalization. These are called **normal forms**. Each consecutive normal form depends on the previous one.The first three normal forms are usually adequate.

Normal Forms are used in database tables to remove or decrease duplication. The following are the many forms:

**First Normal Form:**  
When every attribute in a relation is a single-valued attribute, it is said to be in first normal form. The first normal form is broken when a relation has a composite or multi-valued property.

**Second Normal Form:**

A relation is in second normal form if it meets the first normal form’s requirements and does not contain any partial dependencies. In 2NF, a relation has no partial dependence, which means it has no non-prime attribute that is dependent on any suitable subset of any table candidate key. Often, the problem may be solved by setting a single column Primary Key.

**Third Normal Form:**  
If a relation meets the requirements for the second normal form and there is no transitive dependency, it is said to be in the third normal form.

**Q63. What is OLTP?**

OLTP, or online transactional processing, allows huge groups of people to execute massive amounts of database transactions in real time, usually via the internet. A database transaction occurs when data in a database is changed, inserted, deleted, or queried.

What are the differences between OLTP and OLAP?

OLTP stands for online transaction processing, whereas OLAP stands for online analytical processing. OLTP is an online database modification system, whereas OLAP is an online database query response system.

**Q64. How to create empty tables with the same structure as another table?**

**To create empty tables:**

Using the INTO operator to fetch the records of one table into a new table while setting a WHERE clause to false for all entries, it is possible to create empty tables with the same structure. As a result, SQL creates a new table with a duplicate structure to accept the fetched entries, but nothing is stored into the new table since the WHERE clause is active.

**Q65. What is PostgreSQL?**

In 1986, a team lead by Computer Science Professor Michael Stonebraker created PostgreSQL under the name Postgres. It was created to aid developers in the development of enterprise-level applications by ensuring data integrity and fault tolerance in systems. PostgreSQL is an enterprise-level, versatile, resilient, open-source, object-relational database management system that supports variable workloads and concurrent users. The international developer community has constantly backed it. PostgreSQL has achieved significant appeal among developers because to its fault-tolerant characteristics.  
It’s a very reliable database management system, with more than two decades of community work to thank for its high levels of resiliency, integrity, and accuracy. Many online, mobile, geospatial, and analytics applications utilise PostgreSQL as their primary data storage or data warehouse.

**Q66. What are SQL comments?**

SQL Comments are used to clarify portions of SQL statements and to prevent SQL statements from being executed. Comments are quite important in many programming languages. The comments are not supported by a Microsoft Access database. As a result, the Microsoft Access database is used in the examples in Mozilla Firefox and Microsoft Edge.  
Single Line Comments: It starts with two consecutive hyphens (–).  
Multi-line Comments: It starts with /\* and ends with \*/.

Let’s move to the next question in this SQL Interview Questions.

**Q67. What is the difference between the RANK() and DENSE\_RANK() functions?**

The RANK() function in the result set defines the rank of each row within your ordered partition. If both rows have the same rank, the next number in the ranking will be the previous rank plus a number of duplicates. If we have three records at rank 4, for example, the next level indicated is 7.

[[](https://www.edureka.co/microsoft-sql-server-certification-training?utm_source=blogbanner&utm_campaign=batches)](https://www.edureka.co/microsoft-sql-server-certification-training?utm_source=blogbanner&utm_campaign=batches" \t "_blank)

### [Microsoft SQL Server Certification Course](https://www.edureka.co/microsoft-sql-server-certification-training?utm_source=blogbanner&utm_campaign=batches" \t "_blank)

[Weekday / Weekend BatchesSee Batch Details](https://www.edureka.co/microsoft-sql-server-certification-training?utm_source=blogbanner&utm_campaign=batches" \t "_blank)

The DENSE\_RANK() function assigns a distinct rank to each row within a partition based on the provided column value, with no gaps. It always indicates a ranking in order of precedence. This function will assign the same rank to the two rows if they have the same rank, with the next rank being the next consecutive number. If we have three records at rank 4, for example, the next level indicated is 5.

**Q68. What is SQL Injection?**

SQL injection is a sort of flaw in website and web app code that allows attackers to take control of back-end processes and access, retrieve, and delete sensitive data stored in databases. In this approach, malicious SQL statements are entered into a database entry field, and the database becomes exposed to an attacker once they are executed. By utilising data-driven apps, this strategy is widely utilised to get access to sensitive data and execute administrative tasks on databases. SQLi attack is another name for it.

The following are some examples of SQL injection:

* Getting access to secret data in order to change a SQL query to acquire the desired results.
* UNION attacks are designed to steal data from several database tables.
* Examine the database to get information about the database’s version and structure

**Q69. How many Aggregate functions are available in SQL?**

SQL aggregate functions provide information about a database’s data. AVG, for example, returns the average of a database column’s values.

SQL provides seven (7) aggregate functions, which are given below:

AVG(): returns the average value from specified columns.  
COUNT(): returns the number of table rows, including rows with null values.  
MAX(): returns the largest value among the group.  
MIN(): returns the smallest value among the group.  
SUM(): returns the total summed values(non-null) of the specified column.  
FIRST(): returns the first value of an expression.  
LAST(): returns the last value of an expression.

**Q70. What is the default ordering of data using the ORDER BY clause? How could it be changed?**

The ORDER BY clause in MySQL can be used without the ASC or DESC modifiers. The sort order is preset to ASC or ascending order when this attribute is absent from the ORDER BY clause.

**Q71. How do we use the DISTINCT statement? What is its use?**

The SQL DISTINCT keyword is combined with the SELECT query to remove all duplicate records and return only unique records. There may be times when a table has several duplicate records.  
The DISTINCT clause in SQL is used to eliminate duplicates from a SELECT statement’s result set.

**Q72. What are the syntax and use of the COALESCE function?**

From a succession of expressions, the COALESCE function returns the first non-NULL value. The expressions are evaluated in the order that they are supplied, and the function’s result is the first non-null value. Only if all of the inputs are null does the COALESCE method return NULL.

The syntax of COALESCE function is COALESCE (exp1, exp2, …. expn)

### ****Q73. What is the ACID property in a database?****

ACID stands for Atomicity, Consistency, Isolation, Durability. It is used to ensure that the data transactions are processed reliably in a database system.

* **Atomicity:** Atomicity refers to the transactions that are completely done or failed where transaction refers to a single logical operation of a data. It means if one part of any transaction fails, the entire transaction fails and the database state is left unchanged.
* **Consistency:** Consistency ensures that the data must meet all the validation rules. In simple words,  you can say that your transaction never leaves the database without completing its state.
* **Isolation:** The main goal of isolation is concurrency control.
* **Durability:** Durability means that if a transaction has been committed, it will occur whatever may come in between such as power loss, crash or any sort of error.

Want to upskill yourself to get ahead in your career? Check out this video in this SQL Interview Questions

## ****Top 10 Technologies to Learn in 2023 | Edureka****

This Edureka video on 𝐓𝐨𝐩 𝟏𝟎 𝐓𝐞𝐜𝐡𝐧𝐨𝐥𝐨𝐠𝐢𝐞𝐬 𝐭𝐨 𝐋𝐞𝐚𝐫𝐧 𝐢𝐧 𝟐𝟎𝟐**3** will introduce you to all the popular and trending technologies in the market which you should focus on in 2023. These are the trending technologies that you need to learn in order to have a successful career in the year 2023.



### ****Q74. What do you mean by “Trigger” in SQL?****

[Trigger in SQL](https://www.edureka.co/blog/triggers-in-sql/) is are a special type of stored procedures that are defined to execute automatically in place or after data modifications. It allows you to execute a batch of code when an insert, update or any other query is executed against a specific table.

### ****Q75. What are the different operators available in SQL?****

There are three [operators available in SQL](https://www.edureka.co/blog/sql-operators/), namely:

1. Arithmetic Operators
2. Logical Operators
3. Comparison Operators

Apart from this SQL Interview Questions blog, if you want to get trained from professionals on this technology, you can opt for structured training from edureka!

### ****Q76.  Are NULL values same as that of zero or a blank space?****

A NULL value is not at all same as that of zero or a blank space. NULL value represents a value which is unavailable, unknown, assigned or not applicable whereas a zero is a number and blank space is a character.

### ****Q77. What is the difference between cross join and natural join?****

The cross join produces the cross product or Cartesian product of two tables whereas the natural join is based on all the columns having the same name and data types in both the tables.

### ****Q78. What is subquery in SQL?****

A subquery is a query inside another query where a query is defined to retrieve data or information back from the database. In a subquery, the outer query is called as the main query whereas the inner query is called subquery. Subqueries are always executed first and the result of the subquery is passed on to the main query. It can be nested inside a SELECT, UPDATE or any other query. A subquery can also use any comparison operators such as >,< or =.

### ****Q79. What are the different types of a subquery?****

There are two types of subquery namely, Correlated and Non-Correlated.

**Correlated subquery**: These are queries which select the data from a table referenced in the outer query. It is not considered as an independent query as it refers to another table and refers the column in a table.

**Non-Correlated subquery**: This query is an independent query where the output of subquery is substituted in the main query.

Let’s move to the next question in this SQL Interview Questions.

### ****Q80. List the ways to get the count of records in a table?****

To count the number of records in a [table in SQL](https://www.edureka.co/blog/create-table-in-sql/), you can use the below commands:

SELECT \* FROM table1

SELECT COUNT(\*) FROM table1

SELECT rows FROM sysindexes WHERE id = OBJECT\_ID(table1) AND indid < 2

Apart from this SQL Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for [structured training from edureka!](https://www.edureka.co/sql-essentials-training)

### ****Q81. Write a SQL query to find the names of employees that begin with ‘A’?****

To display name of the employees that begin with ‘A’, type in the below command:

|  |  |
| --- | --- |
| 1 | SELECT \* FROM Table\_name WHERE EmpName like 'A%' |

### ****Q82. Write a SQL query to get the third-highest salary of an employee from employee\_table?****

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | SELECT TOP 1 salary  FROM(  SELECT TOP 3 salary  FROM employee\_table  ORDER BY salary DESC) AS emp  ORDER BY salary ASC; |

### ****Q83. What is the need for group functions in SQL?****

Group functions work on the set of rows and return one result per group. Some of the commonly used group functions are: AVG, COUNT, MAX, MIN, SUM, VARIANCE.

### Q84. What is a Relationship and what are they?

Relation or links are between entities that have something to do with each other. Relationships are defined as the connection between the tables in a database. There are various relationships, namely:

* One to One Relationship.
* One to Many Relationship.
* Many to One Relationship.
* Self-Referencing Relationship.

### ****Q85.  How can you insert NULL values in a column while inserting the data?****

NULL values in SQL can be inserted in the following ways:

* Implicitly by omitting column from column list.
* Explicitly by specifying NULL keyword in the VALUES clause

### ****Q86. What is the main difference between ‘BETWEEN’ and ‘IN’ condition operators?****

BETWEEN operator is used to display rows based on a range of values in a row whereas the IN condition operator is used to check for values contained in a specific set of values.

### Example of BETWEEN:

SELECT \* FROM Students where ROLL\_NO BETWEEN 10 AND 50;

**Example of IN:**

SELECT \* FROM students where ROLL\_NO IN (8,15,25);

### Q87. Why are SQL functions used?

[SQL functions](https://www.edureka.co/blog/sql-functions) are used for the following purposes:

* To perform some calculations on the data
* To modify individual data items
* To manipulate the output
* To format dates and numbers
* To convert the data types

### ****Q88. What is the need for MERGE statement?****

This statement allows conditional update or insertion of data into a table. It performs an UPDATE if a row exists, or an INSERT if the row does not exist.

### ****Q89.**** What do you mean by recursive stored procedure?

Recursive stored procedure refers to a stored procedure which calls by itself until it reaches some boundary condition. This recursive function or procedure helps the programmers to use the same set of code n number of times.

### ****Q90. What is CLAUSE in SQL?****

SQL clause helps to limit the result set by providing a condition to the query. A clause helps to filter the rows from the entire set of records.

For example – WHERE, HAVING clause.

Apart from this SQL Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for a structured training from edureka! Click below to know more.

### ****Q91. What is the difference between ‘HAVING’ CLAUSE and a ‘WHERE’ CLAUSE?****

HAVING clause can be used only with SELECT statement. It is usually used in a GROUP BY clause and whenever GROUP BY is not used, HAVING behaves like a WHERE clause.  
Having Clause is only used with the GROUP BY function in a query whereas WHERE Clause is applied to each row before they are a part of the GROUP BY function in a query.

### Q92. List the ways in which  Dynamic SQL can be executed?

Following are the ways in which dynamic SQL can be executed:

* Write a query with parameters.
* Using EXEC.
* Using sp\_executesql.

### Q93. What are the various levels of constraints?

Constraints are the representation of a column to enforce data entity and consistency. There are two levels  of a constraint, namely:

* column level constraint
* table level constraint

### ****Q94. How can you fetch common records from two tables?****

You can fetch common records from two tables using INTERSECT. For example:

Select studentID from student. <strong>INTERSECT </strong> Select StudentID from Exam

### Q95. List some case manipulation functions in SQL?

There are three case manipulation functions in SQL, namely:

* LOWER: This function returns the string in lowercase. It takes a string as an argument and returns it by converting it into lower case. Syntax:

LOWER(‘string’)

* UPPER: This function returns the string in uppercase. It takes a string as an argument and returns it by converting it into uppercase. Syntax:

UPPER(‘string’)

* INITCAP: This function returns the string with the first letter in uppercase and rest of the letters in lowercase. Syntax:

INITCAP(‘string’)

Apart from this SQL Interview Questions blog, if you want to get trained from professionals on this technology, you can opt for a structured training from edureka! Click below to know more.

### Q96. What are the different set operators available in SQL?

Some of the available set operators are – Union, Intersect or Minus operators.

### Q97. What is an ALIAS command?

[ALIAS command in SQL](https://www.edureka.co/blog/sql-commands) is the name that can be given to any table or a column. This alias name can be referred in WHERE clause to identify a particular table or a column.

For example-

Select emp.empID, dept.Result from employee emp, department as dept where emp.empID=dept.empID

In the above example, emp refers to alias name for employee table and dept refers to alias name for department table.

Let’s move to the next question in this SQL Interview Questions.

### Q98. What are aggregate and scalar functions?

Aggregate functions are used to evaluate mathematical calculation and returns a single value. These calculations are done from the columns in a table. For example- max(),count() are calculated with respect to numeric.

Scalar functions return a single value based on the input value. For example – UCASE(), NOW() are calculated with respect to string.

Let’s move to the next question in this SQL Interview Questions.

### Q99. How can you fetch alternate records from a table?

You can fetch alternate records i.e both odd and even row numbers. For example- To display even numbers, use the following command:

Select studentId from (Select rowno, studentId from student) where mod(rowno,2)=0

Now, to display odd numbers:

Select studentId from (Select rowno, studentId from student) where mod(rowno,2)=1

### Q100. Name the operator which is used in the query for pattern matching?

LIKE operator is used for pattern matching, and it can be used as -.

1. % – It matches zero or more characters.

For example- select \* from students where studentname like ‘a%’

\_ (Underscore) – it matches exactly one character.  
For example- select \* from student where studentname like ‘abc\_’

Apart from this SQL Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for [structured training from edureka!](https://www.edureka.co/sql-essentials-training)

### Q101. How can you select unique records from a table?

You can select unique records from a table by using the DISTINCT keyword.

Select DISTINCT studentID from Student

Using this command, it will print unique student id from the table Student.

### Q102. How can you fetch first 5 characters of the string?

There are a lot of ways to fetch characters from a string. For example:

Select SUBSTRING(StudentName,1,5) as studentname from student

### ****Q103****. What is the main difference between SQL and PL/SQL?

SQL is a query language that allows you to issue a single query or execute a single insert/update/delete whereas PL/SQL is Oracle’s “Procedural Language” SQL, which allows you to write a full program (loops, variables, etc.) to accomplish multiple operations such as selects/inserts/updates/deletes.

### ****Q104. What is a View?****

A view is a virtual table which consists of a subset of data contained in a table. Since views are not present, it takes less space to store. View can have data of one or more tables combined and it depends on the relationship.

Let’s move to the next question in this SQL Interview Questions.

### ****Q105. What are Views used for?****

A view refers to a logical snapshot based on a table or another view. It is used for the following reasons:

* Restricting access to data.
* Making complex queries simple.
* Ensuring data independence.
* Providing different views of same data.

### Q106. What is a Stored Procedure?

A Stored Procedure is a function which consists of many SQL statements to access the database system. Several SQL statements are consolidated into a stored procedure and execute them whenever and wherever required which saves time and avoid writing code again and again.

### Q107. List some advantages and disadvantages of Stored Procedure?

### ****Advantages****:

A Stored Procedure can be used as a modular programming which means create once, store and call for several times whenever it is required. This supports faster execution. It also reduces network traffic and provides better security to the data.

### ****Disadvantage****:

The only disadvantage of Stored Procedure is that it can be executed only in the database and utilizes more memory in the database server.

### Q108. List all the types of user-defined functions?

There are three types of user-defined functions, namely:

* Scalar Functions
* Inline Table-valued functions
* Multi-statement valued functions

Scalar returns the unit, variant defined the return clause. Other two types of defined functions return table.

Let’s move to the next question in this SQL Interview Questions.

### Q109. What do you mean by Collation?

Collation is defined as a set of rules that determine how data can be sorted as well as compared. Character data is sorted using the rules that define the correct character sequence along with options for specifying case-sensitivity, character width etc.

Let’s move to the next question in this SQL Interview Questions.

### Q110. What are the different types of Collation Sensitivity?

Following are the different types of collation sensitivity:

* Case Sensitivity: A and a and B and b.
* Kana Sensitivity: Japanese Kana characters.
* Width Sensitivity: Single byte character and double-byte character.
* Accent Sensitivity.

Apart from this SQL Interview Questions Blog, if you want to get trained from professionals on this technology, you can opt for[structured training from edureka!](https://www.edureka.co/sql-essentials-training)

### Q111. What are Local and Global variables?

### ****Local variables:****

These variables can be used or exist only inside the function. These variables are not used or referred by any other function.

### ****Global variables:****

These variables are the variables which can be accessed throughout the program. Global variables cannot be created whenever that function is called.

### Q112. What is Auto Increment in SQL?

Autoincrement keyword allows the user to create a unique number to get generated whenever a new record is inserted into the table.  
This keyword is usually required whenever PRIMARY KEY in SQL is used.

[AUTO INCREMENT keyword](https://www.edureka.co/blog/sql-auto-increment/) can be used in Oracle and IDENTITY keyword can be used in SQL SERVER.

### Q113. What is a Datawarehouse?

Datawarehouse refers to a central repository of data where the data is assembled from multiple sources of information. Those data are consolidated, transformed and made available for the mining as well as online processing. Warehouse data also have a subset of data called Data Marts.

### Q114. What are the different authentication modes in SQL Server? How can it be changed?

Windows mode and Mixed Mode – SQL and Windows. You can go to the below steps to change authentication mode in SQL Server:

* Click Start> Programs> Microsoft SQL Server and click SQL Enterprise Manager to run SQL Enterprise Manager from the Microsoft SQL Server program group.
* Then select the server from the Tools menu.
* Select SQL Server Configuration Properties, and choose the Security page.

### ****Q115. What are STUFF and REPLACE function?****

**STUFF Function**: This function is used to overwrite existing character or inserts a string into another string. Syntax:

STUFF(string\_expression,start, length, replacement\_characters)

where,  
string\_expression: it is the string that will have characters substituted

start: This refers to the starting position  
length: It refers to the number of characters in the string which are substituted.

replacement\_string: They are the new characters which are injected in the string.

**REPLACE function**: This function is used to replace the existing characters of all the occurrences. Syntax:

REPLACE (string\_expression, search\_string, replacement\_string)

Here every search\_string in the string\_expression will be replaced with the replacement\_string.

So this brings us to the end of the SQL interview questions blog. I hope this set of SQL Interview Questions will help you ace your job interview. **All the best for your interview!**

**SQL Interview Questions and Answers for Freshers**

**1. Define Database.**

A [database](https://intellipaat.com/blog/what-is-database/) is an organized collection of structured data that can be stored, easily accessed, managed, and retrieved digitally from a remote or local computer system. Databases can be complex and vast and are built with a fixed design and modeling approach. While smaller databases can be stored on a file system, large ones are hosted on computer clusters or cloud storage.

**2. What is the difference between DBMS and RDBMS?**

A database management system or [DBMS](https://intellipaat.com/blog/what-is-dbms/) is system software that can create, retrieve, update, and manage a database. It ensures the consistency of data and sees to it that it is organized and easily accessible by acting as an interface between the database and its end-users or [**application software**](https://intellipaat.com/blog/what-is-application-software/). DBMS can be classified into four types:

* **Hierarchical Database:** It has a treelike structure with the data being stored in a hierarchical format. A parent in a hierarchical database can have multiple children, but a child can have only one parent.
* **Network Database:**  This type of database is presented as a graph that can have many-to-many relationships, allowing entities to have multiple connections.
* **Relational Database:** It is the most widely used and easy-to-use database. It is represented as a table and the values in the columns and rows are related to each other.
* **Object-Oriented Database:** The data values and operations are stored as objects in this type of database, and these objects have multiple relationships among them.

An RDBMS stores data in the form of a collection of [tables](https://intellipaat.com/blog/tutorial/sql-tutorial/tables-in-sql/). The relations are defined between the common fields of these tables. MS SQL Server, MySQL, IBM DB2, Oracle, and Amazon Redshift are all based on RDBMS.

**DBMS Vs. RDBMS**

|  |  |  |
| --- | --- | --- |
| **Parameters** | **DBMS** | **RDBMS** |
| **Access** | Data elements need to be accessed separately | Multiple data elements can be accessed at the same time |
| **Relationship Between Data** | No relationship between data | Data in tables are related to each other |
| **Normalization** | It is not present | It is present |
| **Distributed Database** | It does not support distributed databases. | It supports distributed database |
| **Data Storage Format** | Data is stored in either a navigational or hierarchical form | Data is stored in a tabular structure with headers being the column names and the rows containing the corresponding values |
| **Amount of Data** | It deals with a small quantity of data | It deals with a larger amount of data |
| **Data Redundancy** | It is prevalent | Keys and indexes do not allow data redundancy |
| **Number of Users** | It supports a single user | It supports multiple users |
| **Data Fetching** | It is slower for large amounts of data | It is speedy due to the relational approach |
| **Data Security** | Low-security levels when it comes to data manipulation | Multiple levels of data security exist |
| **Software and Hardware Requirements** | Low | High |
| **Examples** | XML, Window Registry, etc. | MySQL, SQL Server, Oracle, [**Microsoft Access**](https://intellipaat.com/blog/what-is-microsoft-access/), PostgreSQL, etc. |

**3. What is SQL?**

[SQL](https://intellipaat.com/blog/tutorial/sql-tutorial/introduction-to-sql/) stands for Structured Query Language. It is the standard language for RDBMS and is useful in handling organized data with entities or variables with relations between them. SQL is used for communicating with databases.

According to ANSI, SQL is used for maintaining RDBMS and for performing different operations of data manipulation on different [datatypes](https://intellipaat.com/blog/tutorial/sql-tutorial/sql-data-types/) by using the [features of SQL](https://intellipaat.com/blog/tutorial/sql-tutorial/sql-features/). It is a database language that is used for the creation and deletion of databases. It can also be used, among other things, to fetch and modify the rows of a table.

**4. What is normalization and what are its types?**

[Normalization](https://intellipaat.com/blog/what-is-normalization-in-sql/) is used to reduce data redundancy and dependency by organizing fields and tables in databases. It involves constructing tables and setting up relationships between those tables according to certain rules. The redundancy and inconsistent dependency can be removed using these rules to make normalization more flexible.

The different forms of normalization are as follows:

* **First Normal Form:** If every attribute in a relation is single-valued, then it is in the first normal form. If it contains a composite or multi-valued attribute, it violates the first normal form.
* **Second Normal Form:** A relation is said to be in the second normal form if it has met the conditions for the first normal form and does not have any partial dependency, i.e., it does not have a non-prime attribute that relies on any proper subset of any candidate key of the table. Often, the solution to this problem is to specify a single-column primary key.
* **Third Normal Form:** A relation is in the third normal form when it meets the conditions for the second normal form and there is not any transitive dependency between the non-prime attributes, i.e., all the non-prime attributes are decided only by the candidate keys of the relation and not by other non-prime attributes.
* **Boyce-Codd Normal Form:** A relation is in the Boyce-Codd normal form or BCNF if it meets the conditions of the third normal form, and for every functional dependency, the left-hand side is a super key. A relation is in BCNF if and only if X is a super key for every non-trivial functional dependency in form X –> Y.

**5. What is denormalization?**

Denormalization is the opposite of normalization; redundant data is added to speed up complex queries that have multiple tables that need to be joined. Optimization of the read performance of a database is attempted by adding or grouping redundant copies of data.

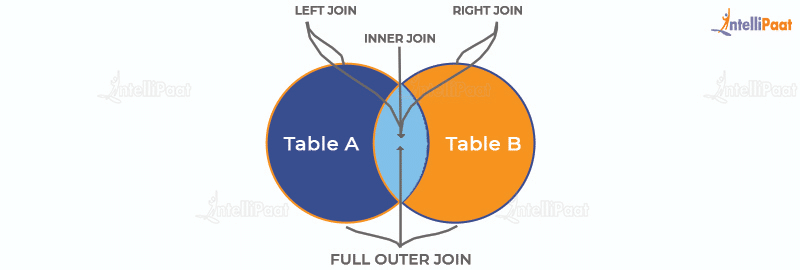
**6. What are Joins in SQL?**

[JOINS in SQL](https://intellipaat.com/blog/tutorial/sql-tutorial/joins/) is used to combine rows from two or more tables based on a related column between them. Various types of JOINS can be used to retrieve data, depending on the relationship between tables.

There are four types of Joins:

* **Inner Join**
* **Left Join**
* **Right Join**
* **Full Join**

**7. Explain the types of SQL joins.**



There are four different types of SQL Joins:

* + **(Inner) Join:** An inner join is used to retrieve the records that have matching values in tables involved in the join. It combines rows from two tables based on a related column and returns only the matching record. [Inner Join](https://intellipaat.com/blog/tutorial/sql-tutorial/inner-join-in-sql/) is mostly used to join queries.
  + SELECT \*
  + FROM Table\_A
  + JOIN Table\_B;
  + SELECT \*
  + FROM Table\_A

INNER JOIN Table\_B;

* + **Left (Outer) Join:** The use of [left join](https://intellipaat.com/blog/tutorial/sql-tutorial/left-join-sql/) is to retrieve all the records or rows from the left and the matched ones from the right.
  + SELECT \*
  + FROM Table\_A A
  + LEFT JOIN Table\_B B

ON A.col = B.col;

* + **Right (Outer) Join:** The use of [Right join](https://intellipaat.com/blog/tutorial/sql-tutorial/right-join-sql/) is to retrieve all the records or rows from the right and the matched ones from the left.
  + SELECT \*
  + FROM Table\_A A
  + RIGHT JOIN Table\_B B

ON A.col = B.col;

* **Full (Outer) Join:** The use of [Full join](https://intellipaat.com/blog/tutorial/sql-tutorial/full-join-sql/) is to retrieve the records that have a match either in the left table or the right table.
* SELECT \*
* FROM Table\_A A
* FULL JOIN Table\_B B

ON A.col = B.col;

***Ace your next SQL interview with our expert-written***[***SQL Join Interview Questions***](https://intellipaat.com/blog/interview-question/sql-joins-interview-questions/)***.***

**Get 100% Hike!**

Master Most in Demand Skills Now !

Top of Form

 +91  IN          INDIA+44  UK          UNITED KINGDOM+1  US          UNITED STATES+1  CA          CANADA------  --          ------------------------+376  AD          ANDORRA+971  AE          UNITED ARAB EMIRATES+93  AF          AFGHANISTAN+1268  AG          ANTIGUA AND BARBUDA+1264  AI          ANGUILLA+355  AL          ALBANIA+374  AM          ARMENIA+599  AN          NETHERLANDS ANTILLES+244  AO          ANGOLA+672  AQ          ANTARCTICA+54  AR          ARGENTINA+1684  AS          AMERICAN SAMOA+43  AT          AUSTRIA+61  AU          AUSTRALIA+297  AW          ARUBA+994  AZ          AZERBAIJAN+387  BA          BOSNIA AND HERZEGOVINA+1246  BB          BARBADOS+880  BD          BANGLADESH+32  BE          BELGIUM+226  BF          BURKINA FASO+359  BG          BULGARIA+973  BH          BAHRAIN+257  BI          BURUNDI+229  BJ          BENIN+590  BL          SAINT BARTHELEMY+1441  BM          BERMUDA+673  BN          BRUNEI DARUSSALAM+591  BO          BOLIVIA+55  BR          BRAZIL+1242  BS          BAHAMAS+975  BT          BHUTAN+267  BW          BOTSWANA+375  BY          BELARUS+501  BZ          BELIZE+61  CC          COCOS (KEELING ISLANDS+243  CD          CONGO, THE DEMOCRATIC REPUBLIC OF THE+236  CF          CENTRAL AFRICAN REPUBLIC+242  CG          CONGO+41  CH          SWITZERLAND+225  CI          COTE D IVOIRE+682  CK          COOK ISLANDS+56  CL          CHILE+237  CM          CAMEROON+86  CN          CHINA+57  CO          COLOMBIA+506  CR          COSTA RICA+53  CU          CUBA+238  CV          CAPE VERDE+61  CX          CHRISTMAS ISLAND+357  CY          CYPRUS+420  CZ          CZECH REPUBLIC+49  DE          GERMANY+253  DJ          DJIBOUTI+45  DK          DENMARK+1767  DM          DOMINICA+1809  DO          DOMINICAN REPUBLIC+213  DZ          ALGERIA+593  EC          ECUADOR+372  EE          ESTONIA+20  EG          EGYPT+291  ER          ERITREA+34  ES          SPAIN+251  ET          ETHIOPIA+358  FI          FINLAND+679  FJ          FIJI+500  FK          FALKLAND ISLANDS (MALVINAS+691  FM          MICRONESIA, FEDERATED STATES OF+298  FO          FAROE ISLANDS+33  FR          FRANCE+241  GA          GABON+1473  GD          GRENADA+995  GE          GEORGIA+233  GH          GHANA+350  GI          GIBRALTAR+299  GL          GREENLAND+220  GM          GAMBIA+224  GN          GUINEA+240  GQ          EQUATORIAL GUINEA+30  GR          GREECE+502  GT          GUATEMALA+1671  GU          GUAM+245  GW          GUINEA-BISSAU+592  GY          GUYANA+852  HK          HONG KONG+504  HN          HONDURAS+385  HR          CROATIA+509  HT          HAITI+36  HU          HUNGARY+62  ID          INDONESIA+353  IE          IRELAND+972  IL          ISRAEL+44  IM          ISLE OF MAN+964  IQ          IRAQ+98  IR          IRAN, ISLAMIC REPUBLIC OF+354  IS          ICELAND+39  IT          ITALY+1876  JM          JAMAICA+962  JO          JORDAN+81  JP          JAPAN+254  KE          KENYA+996  KG          KYRGYZSTAN+855  KH          CAMBODIA+686  KI          KIRIBATI+269  KM          COMOROS+1869  KN          SAINT KITTS AND NEVIS+850  KP          KOREA DEMOCRATIC PEOPLES REPUBLIC OF+82  KR          KOREA REPUBLIC OF+965  KW          KUWAIT+1345  KY          CAYMAN ISLANDS+7  KZ          KAZAKSTAN+856  LA          LAO PEOPLES DEMOCRATIC REPUBLIC+961  LB          LEBANON+1758  LC          SAINT LUCIA+423  LI          LIECHTENSTEIN+94  LK          SRI LANKA+231  LR          LIBERIA+266  LS          LESOTHO+370  LT          LITHUANIA+352  LU          LUXEMBOURG+371  LV          LATVIA+218  LY          LIBYAN ARAB JAMAHIRIYA+212  MA          MOROCCO+377  MC          MONACO+373  MD          MOLDOVA, REPUBLIC OF+382  ME          MONTENEGRO+1599  MF          SAINT MARTIN+261  MG          MADAGASCAR+692  MH          MARSHALL ISLANDS+389  MK          MACEDONIA, THE FORMER YUGOSLAV REPUBLIC OF+223  ML          MALI+95  MM          MYANMAR+976  MN          MONGOLIA+853  MO          MACAU+1670  MP          NORTHERN MARIANA ISLANDS+222  MR          MAURITANIA+1664  MS          MONTSERRAT+356  MT          MALTA+230  MU          MAURITIUS+960  MV          MALDIVES+265  MW          MALAWI+52  MX          MEXICO+60  MY          MALAYSIA+258  MZ          MOZAMBIQUE+264  NA          NAMIBIA+687  NC          NEW CALEDONIA+227  NE          NIGER+234  NG          NIGERIA+505  NI          NICARAGUA+31  NL          NETHERLANDS+47  NO          NORWAY+977  NP          NEPAL+674  NR          NAURU+683  NU          NIUE+64  NZ          NEW ZEALAND+968  OM          OMAN+507  PA          PANAMA+51  PE          PERU+689  PF          FRENCH POLYNESIA+675  PG          PAPUA NEW GUINEA+63  PH          PHILIPPINES+92  PK          PAKISTAN+48  PL          POLAND+508  PM          SAINT PIERRE AND MIQUELON+870  PN          PITCAIRN+1  PR          PUERTO RICO+351  PT          PORTUGAL+680  PW          PALAU+595  PY          PARAGUAY+974  QA          QATAR+40  RO          ROMANIA+381  RS          SERBIA+7  RU          RUSSIAN FEDERATION+250  RW          RWANDA+966  SA          SAUDI ARABIA+677  SB          SOLOMON ISLANDS+248  SC          SEYCHELLES+249  SD          SUDAN+46  SE          SWEDEN+65  SG          SINGAPORE+290  SH          SAINT HELENA+386  SI          SLOVENIA+421  SK          SLOVAKIA+232  SL          SIERRA LEONE+378  SM          SAN MARINO+221  SN          SENEGAL+252  SO          SOMALIA+597  SR          SURINAME+239  ST          SAO TOME AND PRINCIPE+503  SV          EL SALVADOR+963  SY          SYRIAN ARAB REPUBLIC+268  SZ          SWAZILAND+1649  TC          TURKS AND CAICOS ISLANDS+235  TD          CHAD+228  TG          TOGO+66  TH          THAILAND+992  TJ          TAJIKISTAN+690  TK          TOKELAU+670  TL          TIMOR-LESTE+993  TM          TURKMENISTAN+216  TN          TUNISIA+676  TO          TONGA+90  TR          TURKEY+1868  TT          TRINIDAD AND TOBAGO+688  TV          TUVALU+886  TW          TAIWAN, PROVINCE OF CHINA+255  TZ          TANZANIA, UNITED REPUBLIC OF+380  UA          UKRAINE+256  UG          UGANDA+598  UY          URUGUAY+998  UZ          UZBEKISTAN+39  VA          HOLY SEE (VATICAN CITY STATE+1784  VC          SAINT VINCENT AND THE GRENADINES+58  VE          VENEZUELA+1284  VG          VIRGIN ISLANDS, BRITISH+1340  VI          VIRGIN ISLANDS, U.S.+84  VN          VIET NAM+678  VU          VANUATU+681  WF          WALLIS AND FUTUNA+685  WS          SAMOA+381  XK          KOSOVO+967  YE          YEMEN+262  YT          MAYOTTE+27  ZA          SOUTH AFRICA+260  ZM          ZAMBIA+263  ZW          ZIMBABWE

 By providing your contact details, you agree to our [Terms of Use](https://intellipaat.com/terms-of-use/) & [Privacy Policy](https://intellipaat.com/terms-of-use-and-privacy-policy/)

Bottom of Form

**8. What are the subsets of SQL?**

[SQL queries](https://intellipaat.com/blog/tutorial/sql-tutorial/how-to-create-database-in-sql/)are divided into four main categories:

* **Data Definition Language (DDL)**  
  DDL queries are made up of SQL commands that can be used to define the structure of the database and modify it.
  + **CREATE:** Creates databases, tables, schema, etc.
  + **DROP:** Drops tables and other database objects
  + **DROP COLUMN:** Drops a column from any table structure
  + **ALTER:** Alters the definition of database objects
  + **TRUNCATE:** Removes tables, views, procedures, and other database objects
  + **ADD COLUMN:** Adds any column to the table schema
* **Data Manipulation Language (DML)**  
  These SQL queries are used to manipulate data in a database.
  + **SELECT INTO:**Selects data from one table and inserts it into another
  + **INSERT:**Inserts data or records into a table
  + **UPDATE:**Updates the value of any record in the database
  + **DELETE:**Deletes records from a table
* **Data Control Language (DCL)**  
  These SQL queries manage the access rights and permission control of the database.
  + **GRANT:** Grants access rights to database objects
  + **REVOKE:** Withdraws permission from database objects
* **Transaction Control Language (TCL)**  
  TCL is a set of commands that essentially manages the transactions in a database and the changes made by the DML statements. TCL allows statements to be grouped together into logical transactions.
  + **COMMIT:**
  + Commits an irreversible transaction, i.e., the previous image of the database before the transaction cannot be retrieved. When a transaction is committed using the COMMIT statement in SQL, it permanently saves the changes made within the transaction to the database. Once committed, the changes cannot be rolled back or undone, and the previous state of the database before the transaction cannot be retrieved without restoring from a backup or utilizing other data recovery methods.
  + **ROLLBACK:** Reverts the steps in a transaction in case of an error
  + **SAVEPOINT:** Sets a savepoint in the transaction to which rollback can be executed
  + **SET TRANSACTION:** Sets the characteristics of the transaction

**9. What are the applications of SQL?**

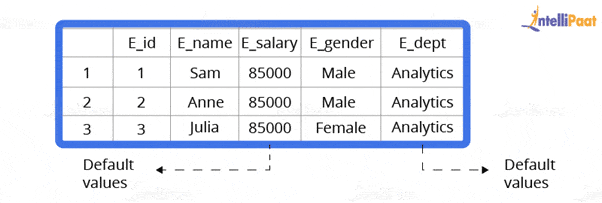
The major applications of SQL are listed below:

* Writing data integration scripts
* Setting and running analytical queries
* Retrieving subsets of information within a database for analytics applications and transaction processing
* Adding, updating, and deleting rows and columns of data in a database

**10. What is a DEFAULT constraint?**

[Constraints in SQL](https://intellipaat.com/blog/tutorial/sql-tutorial/constraints/) are used to specify some sort of rules for processing data and limiting the type of data that can go into a table. Now, let us understand what a default constraint is.

A default constraint is used to define a default value for a column so that it is added to all new records if no other value is specified. For example, if we assign a default constraint for the E\_salary column in the following table and set the default value to 85000, all the entries in this column will have the default value of 85000, unless no other value has been assigned during the insertion.



Now, let us go through how to set a default constraint. We will start by creating a new table and adding a default constraint to one of its columns.

**Code:**

create table stu1(s\_id int, s\_name varchar(20), s\_marks int default 50)

select \*stu1

**Output:**



Now, we will insert the records.

**Code:**

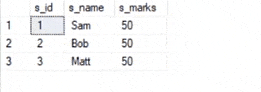
insert into stu1(s\_id,s\_name) values(1,’Sam’)

insert into stu1(s\_id,s\_name) values(2,’Bob’)

insert into stu1(s\_id,s\_name) values(3,’Matt’)

select \*from stu1

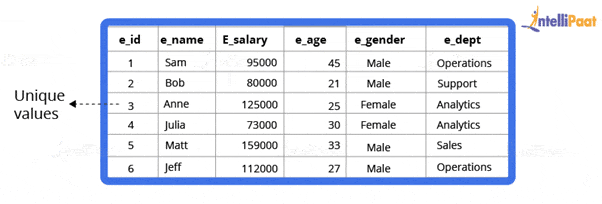
**Output:**



***Also, learn from our blog on***[***MySQL Interview Questions and Answers***](https://intellipaat.com/blog/interview-question/mysql-interview-questions/)***to crack any Interview.***

**11. What is a UNIQUE constraint?**

Unique constraints ensure that all the values in a column are different. For example, if we assign a unique constraint to the e\_name column in the following table, then every entry in this column should have a unique value.



First, we will create a table.

create table stu2(s\_id int unique, s\_name varchar(20))

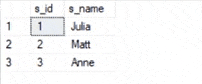
Now, we will insert the records.

insert into stu2 values(1,’Julia’)

insert into stu2 values(2,’Matt’)

insert into stu2 values(3,’Anne’)

**Output:**



A PRIMARY KEY constraint will automatically have a UNIQUE constraint. However, unlike a PRIMARY KEY, multiple UNIQUE constraints are allowed per table.

**12. What is meant by table and field in SQL?**

An organized data in the form of rows and columns is said to be a table. Simply put, it is a collection of related data in a table format.

Here rows and columns are referred to as tuples and attributes, and the number of columns in a table is referred to as a field. In the record, fields represent the characteristics and attributes and contain specific information about the data.

***Prepare yourself for PostgreSQL from our***[***PostgreSQL interview questions***](https://intellipaat.com/blog/interview-question/postgresql-interview-questions/)***blog***

**13. What is a primary key?**

A primary key is used to uniquely identify all table records. It cannot have NULL values and must contain unique values. Only one primary key can exist in one table, and it may have single or multiple fields, making it a composite key.

Now, we will write a query to demonstrate the use of a primary key for the employee table:

//

CREATE TABLE Employee (

ID int NOT NULL,

Employee\_name varchar(255) NOT NULL,

Employee\_designation varchar(255),

Employee\_Age int,

PRIMARY KEY (ID)

);

**14. What is a unique key?**

A key that can accept only a null value and cannot accept duplicate values is called a unique key. The role of a unique key is to make sure that all columns and rows are unique.

The syntax for a unique key will be the same as the primary key. So, the query using a unique key for the employee table will be as follows:

//

CREATE TABLE Employee (

ID int NOT NULL,

Employee\_name varchar(255) NOT NULL,

Employee\_designation varchar(255),

Employee\_Age int,

UNIQUE(ID)

);

**15. What is the difference between a primary key and a unique key?**

Both primary and unique keys carry unique values but a primary key cannot have a null value, while a unique key can. In a table, there cannot be more than one primary key, but there can be multiple unique keys.

**Career Transition**

[[](javascript:void(0);)](javascript:void(0);)

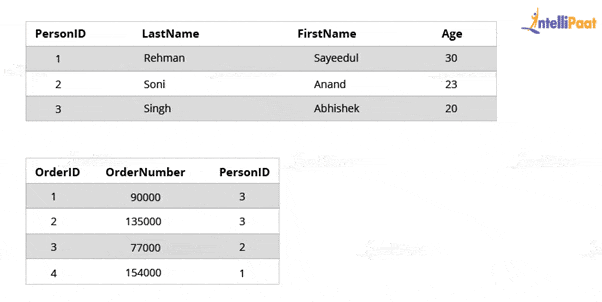
[[](javascript:void(0);)](javascript:void(0);)

[[](javascript:void(0);)](javascript:void(0);)

**16. What is a foreign key?**

A foreign key is an attribute or a set of attributes that reference the primary key of some other table. Basically, a foreign key is used to link together two tables together.

Let us create a foreign key for the following table:



CREATE TABLE Orders (

OrderID int NOT NULL,

OrderNumber int NOT NULL,

PersonID int,

PRIMARY KEY (OrderID),

FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)

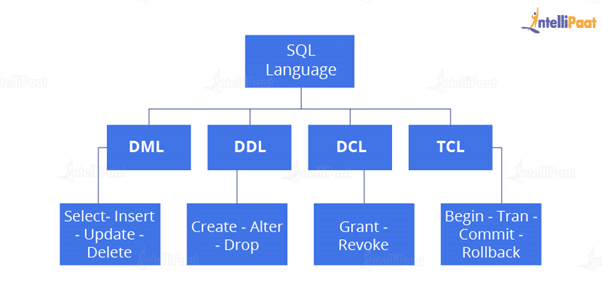
)

**17. What are the benefits of SQL database over NoSQL database?**

SQL (Structured Query Language) and NoSQL (Not Only SQL) are different database management systems, each offering unique advantages. Here, we will discuss the benefits of SQL over NoSQL in a comprehensive manner.

1. **Schema and Data Consistency:** SQL databases enforce a predefined schema, ensuring the data is structured and follows specific rules. It results in a higher level of data consistency, reducing the risk of data corruption or inconsistencies. On the other hand, NoSQL databases are schema-less, allowing for more flexibility but potentially sacrificing consistency.
2. **ACID Compliance:** SQL databases are designed to follow the ACID (Atomicity, Consistency, Isolation, Durability) principles. These properties guarantee transactional integrity, ensuring that database operations are reliable and maintain data integrity. NoSQL databases may sacrifice some of these properties for scalability and performance advantages.
3. **Advanced Querying Capabilities:** SQL databases provide a rich set of powerful querying capabilities through SQL. It allows for complex joins, aggregations, filtering, and data sorting. SQL queries are expressive and can handle complex relationships between tables efficiently. NoSQL databases may have limited querying capabilities, often requiring additional programming logic to achieve similar results.
4. **Data Integrity and Validation:** SQL databases offer built-in mechanisms for data integrity and validation through constraints, such as unique keys, foreign key relationships, and check constraints. These features ensure data quality and prevent invalid or inconsistent data insertion. NoSQL databases may require additional application-level logic to enforce data integrity.
5. **Mature Ecosystem and Tooling:** SQL databases have been around for decades and have a well-established ecosystem with many tools, libraries, and frameworks. It includes robust backup and recovery solutions, monitoring tools, and mature integration options. Being relatively newer, NoSQL databases may have a more limited tooling and ecosystem.

**18. Explain the different types of SQL commands.**



* **DDL:** DDL is that part of SQL that defines the data structure of the database in the initial stage when the database is about to be created. It is mainly used to create and restructure database objects. Commands in DDL are:
  + Create table
  + [Alter table](https://intellipaat.com/blog/tutorial/sql-tutorial/alter-command-in-sql/)
  + Drop table
* **DML:** DML is used to manipulate already existing data in a database, i.e., it helps users to retrieve and manipulate data. It is used to perform operations such as inserting data into the database through the insert command, updating data with the update command, and deleting data from the database through the delete command.
* **DCL:** DCL is used to control access to the data in the database. DCL commands are normally used to create objects related to user access and to control the distribution of privileges among users. The commands that are used in DCL are Grant and Revoke.
* **TCL:** TCL is used to control the changes made by DML commands. It also authorizes the statements to assemble in conjunction with logical transactions. The commands that are used in TCL are Commit, Rollback, Savepoint, Begin, and Transaction.

Also, Have a look at [SQL Command Cheatsheet](https://intellipaat.com/blog/tutorial/sql-tutorial/sql-commands-cheat-sheet/).

**19. What are the uses of SQL?**

The following operations can be performed by using a SQL database:

* Creating new databases
* Inserting new data
* Deleting existing data
* Updating records
* Retrieving the data
* Creating and dropping tables
* Creating functions and views
* Converting data types

**20. What is an index?**

Indexes help speed up searching in a database. If there is no index on a column in the [WHERE clause](https://intellipaat.com/blog/tutorial/sql-tutorial/where-clause/), then the SQL Server has to skim through the entire table and check each and every row to find matches, which may result in slow operations with large data.

Indexes are used to find all rows matching with some columns and then to skim through only those subsets of the data to find the matches.

**Syntax:**

CREATE INDEX INDEX\_NAME ON TABLE\_NAME (COLUMN)

**21. Explain the types of indexes.**

Single-Column Indexes: A single-column index is created for only one column of a table.

**Syntax:**

CREATE INDEX index\_name

ON table\_name(column\_name);

Composite-Column Indexes: A composite-column index is created for two or more columns of a table.

**Syntax:**

CREATE INDEX index\_name

ON table\_name (column1, column2)

Unique Indexes: A unique index is used to maintain the data integrity of a table. A unique index does not allow multiple values to be inserted into the table.

**Syntax:**

CREATE UNIQUE INDEX index

ON table\_name(column\_name)

**Courses you may like**

[](https://intellipaat.com/sql-developer-dba-training/)

[](https://intellipaat.com/data-scientist-course-training/)

[](https://intellipaat.com/business-intelligence-masters-program-training/)

**22. What are entities and relationships?**

**Entities:** An entity can be a person, place, thing, or any identifiable object for which data can be stored in a database.

For example, in a company’s database, employees, projects, salaries, etc., can be referred to as entities.

**Relationships:** A relationship between entities can be referred to as a connection between two tables or entities.

For example, in a college database, the student entity and the department entity are associated with each other.

That ends the section of basic interview questions. Let us now move on to the next section of intermediate interview questions.

**Intermediate SQL Interview Questions and Answers**

**23. What are SQL operators?**

[SQL operators](https://intellipaat.com/blog/tutorial/sql-tutorial/sql-operators/) are the special keywords or characters that perform specific operations. They are also used in SQL queries. These operators can be used within the WHERE clause of [SQL commands](https://intellipaat.com/blog/tutorial/sql-tutorial/introduction-to-sql/). Based on the specified condition, SQL operators filter the data.

The SQL operators can be categorized into the following types:

* **Arithmetic Operators:**For mathematical operations on numerical data
  + addition (+)
  + subtraction (-)
  + multiplication (\*)
  + division (/)
  + remainder/modulus (%)
* **Logical Operators:**For evaluating the expressions and returning results in True or False
  + ALL
  + AND
  + ANY
  + ISNULL
  + EXISTS
  + BETWEEN
  + IN
  + LIKE
  + NOT
  + OR
  + UNIQUE
* **Comparison Operators:**For comparisons of two values and checking whether they are the same or not
  + equal to (=)
  + not equal to (!= or <>)
  + less than (<),
  + greater than (>)
  + less than or equal to (<=)
  + greater than or equal to (>=)
  + not less than (!<)
  + not greater than (!>)
* **Bitwise Operators:**For bit manipulations between two expressions of integer type. It first performs the conversion of integers into binary bits and then applied operators
  + AND (& symbol)
  + OR (|, ^)
  + NOT (~)
* **Compound Operators:**For operations on a variable before setting the variable’s result to the operation’s result
  + Add equals (+=)
  + subtract equals (-=)
  + multiply equals (\*=)
  + divide equals (/=)
  + modulo equals (%=)
* **String Operators:**For concatenation and pattern matching of strings
  + + (String concatenation)
  + += (String concatenation assignment)
  + % (Wildcard)
  + [] (Character(s) matches)
  + [^] (Character(s) not to match)
  + \_ (Wildcard match one character)

**24. What do you mean by data integrity?**

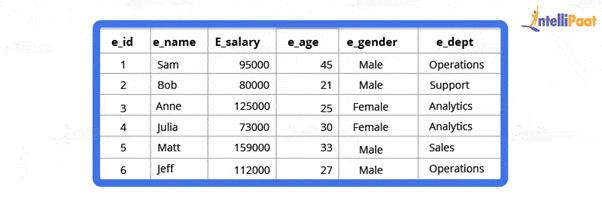
Data integrity is the assurance of the accuracy and consistency of data over its whole life cycle. It is a critical aspect of the design, implementation, and usage of systems that store, process, or retrieve data.

Data integrity also defines integrity constraints for enforcing business rules on data when it is entered into a database or application.

**25. What is a data warehouse?**

A [data warehouse](https://intellipaat.com/blog/what-is-data-warehouse/) is a large store of accumulated data, from a wide range of sources, within an organization. The data helps drive business decisions.

**26. How would you find the second highest salary from the following table?**



**Code:**

select \* from employee

select max(e\_salary) from employee where e\_salary not in (select max(e\_salary) from employee)

**Output:**



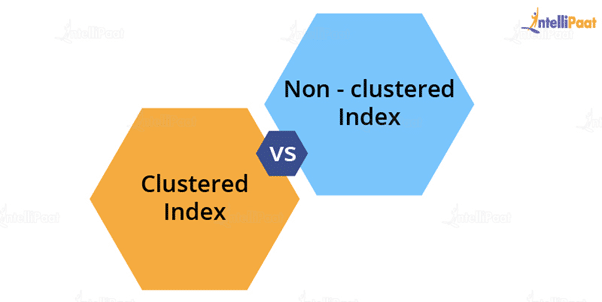
**27. Why is the FLOOR function used in SQL Server?**

The FLOOR() function helps to find the largest integer value for a given number, which can be an equal or lesser number.

***Are you planning to learn SQL Server? Here is the***[***SQL Certification training***](https://intellipaat.com/microsoft-sql-server-certification-training/)***. Enroll now!***

**28. State the differences between clustered and non-clustered indexes**

* **Clustered Index:** It is used to sort the rows of data by their key values. A clustered index is like the contents of a phone book. We can directly open the book on David’s index (for “David, Thompson”) and find information for all Davids right next to each other. Since the data are located next to each other, it helps a lot in fetching the data based on range-based queries. A clustered index is actually related to how the data is stored; only one clustered index is possible per table.
* **Non-Clustered Index:** It stores data at one location and indexes at another location. The index has pointers that point to the location of the data. As the indexes in a non-clustered index are stored in a different place, there can be many non-clustered indexes for a table.

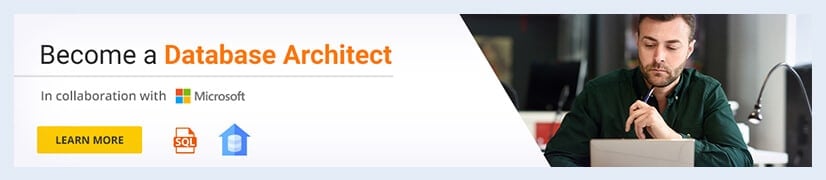


Now, we will see the major differences between clustered and non-clustered indexes:

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Clustered Index** | **Non-Clustered Index** |
| **Used For** | Sorting and storing records physically in memory | Creating a logical order for data rows; pointers are used for physical data files |
| **Methods for Storing** | Stores data in the leaf nodes of the index | Never stores data in the leaf nodes of the index |
| **Size** | Quite large | Comparatively, small |
| **Data Accessing** | Fast | Slow |
| **Additional Disk Space** | Not required | Required to store indexes separately |
| **Type of Key** | By default, the primary key of a table is a clustered index | It can be used with the unique constraint on the table that acts as a composite key |
| **Main Feature** | Improves the performance of data retrieval | Should be created on columns used in Joins |

**29. What do you know about CDC in SQL Server?**

CDC refers to change data capture. It captures recent INSERT, DELETE, and UPDATE activity applied to SQL Server tables. It records changes to SQL Server tables in a compatible format.

[](https://intellipaat.com/database-architect-training/)

**30. What is the difference between SQL and MySQL?**

Now Let’s compare the difference between [SQL and MySQL](https://intellipaat.com/blog/sql-vs-mysql-difference/).

|  |  |
| --- | --- |
| **SQL** | **MySQL** |
| It is a structured query language used in a database | It is a database management system |
| It is used for query and operating database systems, | It allows data handling, storing, and modification in an organized manner |
| It is always the same | It keeps updating |
| It supports only a single storage engine | It supports multiple storage engines |
| The server is independent | During backup sessions, the server blocks the database |

**31. State the differences between SQL and PL/SQL**

|  |  |
| --- | --- |
| **SQL** | **PL/SQL** |
| It is a database structured query language | It is a programming language for a database that uses SQL |
| It is an individual query that is used to execute DML and DDL commands | It is a block of codes used to write the entire procedure or a function |
| It is a declarative and data-oriented language | It is a procedural and application-oriented language |
| It is mainly used for data manipulation | It is used for creating applications |
| It provides interaction with the database server | It does not provide interaction with the database server |
| It cannot contain PL/SQL code | It can contain SQL because it is an extension of SQL |

**32. What is the ACID property in a database?**

The full form of [ACID](https://intellipaat.com/blog/tutorial/sql-tutorial/acid-properties-normalization/) is atomicity, consistency, isolation, and durability. ACID properties are used to check the reliability of transactions.

* + Atomicity refers to completed or failed transactions, where a transaction refers to a single logical operation on data. This implies that if any aspect of a transaction fails, the whole transaction fails and the database state remains unchanged.
* Consistency means that the data meets all validity guidelines. The transaction never leaves the database without finishing its state.
* Concurrency management is the primary objective of isolation.
* Durability ensures that once a transaction is committed, it will occur regardless of what happens in between such as a power outage, fire, or some other kind of disturbance.

***Enroll now in***[***SQL course in Bangalore***](https://intellipaat.com/sql-training-bangalore/)***to learn more in-depth about SQL.***

**33. What is the need for group functions in SQL?**

Group functions operate on a series of rows and return a single result for each group. COUNT(), MAX(), MIN(), SUM(), AVG(), and VARIANCE() are some of the most widely used group functions.

**34. What do you understand about a character manipulation function?**

Character manipulation functions are used for the manipulation of character data types.  
Some of the character manipulation functions are as follows:

**UPPER:** It returns the string in uppercase.

**Syntax:**

UPPER(‘ string’)

**Example:**

SELECT UPPER(‘demo string’) from String;

**Output:**

DEMO STRING

**LOWER:** It returns the string in lowercase.

**Syntax:**

LOWER(‘STRING’)

**Example:**

SELECT LOWER (‘DEMO STRING’) from String

**Output:**

demo string

**INITCAP:** It converts the first letter of the string to uppercase and retains others in lowercase.

**Syntax:**

Initcap(‘sTRING’)

**Example:**

SELECT Initcap(‘dATASET’) from String

**Output:**

Dataset

**CONCAT:** It is used to concatenate two strings.

**Syntax:**

CONCAT(‘str1’,’str2’)

**Example:**

SELECT CONCAT(‘Data’,’Science’) from String

**Output:**

Data Science

**LENGTH:** It is used to get the length of a string.

**Syntax:**

LENGTH(‘String’)

**Example:**

SELECT LENGTH(‘Hello World’) from String

**Output:**

11

**35. What is AUTO\_INCREMENT?**

AUTO\_INCREMENT is used in SQL to automatically generate a unique number whenever a new record is inserted into a table.

Since the primary key is unique for each record, this primary field is added as the AUTO\_INCREMENT field so that it is incremented when a new record is inserted.

The AUTO-INCREMENT value starts at 1 and is incremented by 1 whenever a new record is inserted.

Syntax:

CREATE TABLE Employee(

Employee\_id int NOT NULL AUTO-INCREMENT,

Employee\_name varchar(255) NOT NULL,

Employee\_designation varchar(255)

Age int,

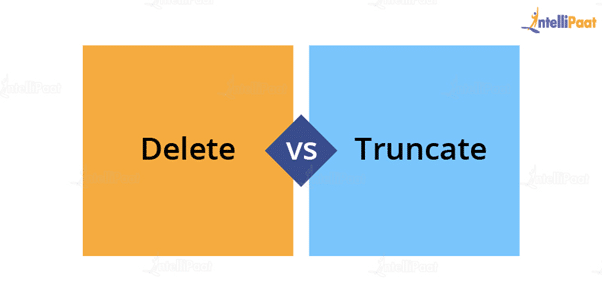
PRIMARY KEY (Employee\_id)

)

***Check out our Blog on***[***PL/SQL Interview Questions***](https://intellipaat.com/blog/interview-question/pl-sql-interview-questions/)***to crack your SQL Interview.***

**36. What is the difference between the DELETE and TRUNCATE commands?**

* **DELETE:** This query is used to delete or remove one or more existing tables.
* **TRUNCATE:**  This statement deletes all the data inside a table.

The differences between DELETE and TRUNCATE commands are the following:

* TRUNCATE is a DDL command, and DELETE is a DML command.
* With TRUNCATE, we cannot really execute and trigger, while with DELETE, we can accomplish a trigger.
* If a table is referenced by foreign key constraints, then TRUNCATE will not work. So, if we have a foreign key, we have to use the DELETE command.

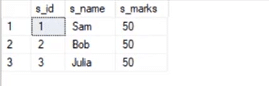
The syntax for the DELETE command is as follows:

DELETE FROM table\_name

[WHERE condition];

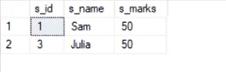
**Example:**

select \* from stu

**Output:**  


delete from stu where s\_name=’Bob’

**Output:**



The syntax for the TRUNCATE command:

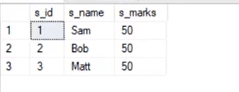
TRUNCATE TABLE

Table\_name;

**Example:**

select \* from stu1

**Output:**



truncate table stu1

**Output:**

output 8

This deletes all the records from a table.

[](https://intellipaat.com/course-cat/big-data-analytics-courses/)

**37. What is the difference between the DROP and TRUNCATE commands?**

If a table is dropped, all things associated with that table are dropped as well. This includes the relationships defined on the table with other tables, access privileges, and grants that the table has, as well as the integrity checks and constraints.

To create and use the table again in its original form, all the elements associated with the table need to be redefined.

However, if a table is truncated, there are no such problems as mentioned above. The table retains its original structure.

**38. What is a “TRIGGER” in SQL?**

The [trigger](https://intellipaat.com/blog/tutorial/sql-server-tutorial/triggers-in-sql-server/) can be defined as an automatic process that happens when an event occurs in the database server. It helps to maintain the integrity of the table. The trigger is activated when the commands, such as insert, update, and delete, are given.

The syntax used to generate the trigger function is as follows:

CREATE TRIGGER trigger\_name

**39. Where are usernames and passwords stored in SQL Server?**

In SQL Server, usernames and passwords are stored in the main database in the sysxlogins table.

**40. What are the types of relationships in SQL Server databases?**

Relationships are developed by interlinking the columns of one table with the column of another table. There are three different types of relationships, which are as follows:

* One-to-one relationship
* Many-to-one relationship
* Many-to-many relationship

**41. What are the third-party tools that are used in SQL Server?**

The following is the list of third-party tools that are used in SQL Server:

* SQL CHECK
* SQL DOC 2
* SQL Backup 5
* SQL Prompt
* Litespeed 5.0

**42. How can you handle expectations in SQL Server?**

TRY and CATCH blocks handle exceptions in SQL Server. Put the SQL statement in the TRY block and write the code in the CATCH block to handle expectations. If there is an error in the code in the TRY block, then the control will automatically move to the CATCH block.

**43. How many authentication modes are there in SQL Server? What are they?**

Two authentication modes are available in SQL Server. They are as follows:

* **Windows Authentication Mode:** It allows authentication for Windows but not for SQL Server.
* **Mixed Mode:** It allows both types of authentication—Windows and SQL Server.

**44. What is a function in SQL Server?**

A function is an SQL Server database object. It is basically a set of SQL statements that allow input parameters, perform processing, and return results only. A function can only return a single value or table. The ability to insert, update, and delete records in database tables is not available.

**45. Mention different types of replication in SQL Server?**

In SQL Server, three different types of replications are available:

* Snapshot replication
* Transactional replication
* Merge replication

**46. Which command is used to find out the SQL Server version?**

The following command is used to identify the version of SQL Server:

Select SERVERPROPERTY('productversion')

**47. What is the COALESCE function?**

The COALESCE function takes a set of inputs and returns the first non-null value.

**Syntax:**

COALESCE(val1,val2,val3,……,nth val)

**Example:**

SELECT COALESCE(NULL, 1, 2, ‘MYSQL’)

**Output:**

1

**48. Can we link SQL Server with others?**

Yes, SQL Server can be linked with other database systems using various methods. One common method is through the use of linked servers. Linked servers allow SQL Server to establish connections and access data from other database platforms. By configuring appropriate settings and creating the necessary connections, SQL Server can interact with databases such as MySQL, Oracle, PostgreSQL, and more, enabling data integration and querying across multiple systems.

***Also, check out the blog on***[***PostgreSQL vs. MySQL***](https://intellipaat.com/blog/postgresql-vs-mysql/)***.***

**49. What is SQL Server Agent?**

SQL Server Agent plays an important role in the daily work of SQL Server administrators or DBAs. This is one of the important parts of SQL Server. The aim of the server agent is to easily implement tasks using a scheduler engine that enables the tasks to be performed at scheduled times. SQL Server Agent uses SQL Server to store scheduled management task information.

**50. What do you know about magic tables in SQL Server?**

A magic table can be defined as a provisional logical table that is developed by an SQL Server for tasks such as insert, delete, or update (DML) operations. The operations recently performed on the rows are automatically stored in magic tables. Magic tables are not physical tables; they are just temporary internal tables.

**51. What are some common clauses used with SELECT queries in SQL?**

There are many SELECT statement clauses in SQL. Some of the most commonly used clauses with SELECT queries are as follows:

* **FROM**The FROM clause defines the tables and views from which data can be interpreted. The tables and views listed must exist at the time the question is given.
* **WHERE**The WHERE clause defines the parameters that are used to limit the contents of the results table. You can test for basic relationships or for relationships between a column and a series of columns using subselects.
* **GROUP BY**The GROUP BY clause is commonly used for aggregate functions to produce a single outcome row for each set of unique values in a set of columns or expressions.
* **ORDER BY**The ORDER BY clause helps in choosing the columns on which the table’s result should be sorted.
* **HAVING**The HAVING clause filters the results of the GROUP BY clause by using an aggregate function.

**52. What is wrong with the following SQL query?**

SELECT gender, AVG(age) FROM employee WHERE AVG(age)>30 GROUP BY gender

When this command is executed, it gives the following error:

Msg 147, Level 16, State 1, Line 1

Aggregation may not appear in the WHERE clause unless it is in a subquery contained in the HAVING clause or a select list; the column being aggregated is an outer reference.

Msg 147, Level 16, State 1, Line 1

Invalid column name ‘gender’.

This means that whenever we work with aggregate functions and use the GROUP BY clause, we cannot use the WHERE clause. Therefore, instead of the WHERE clause, we should use the HAVING clause.

When we use the HAVING clause, the GROUP BY clause should come first, followed by the HAVING clause.

select e\_gender, avg(e\_age) from employee group by e\_gender having avg(e\_age)>30

**Output:**



**53. What do you know about the stuff() function?**

The stuff() function deletes a part of the string and then inserts another part into the string, starting at a specified position.

**Syntax:**

STUFF(String1, Position, Length, String2)

Here, String1 is the one that will be overwritten. The position indicates the starting location for overwriting the string. Length is the length of the substitute string, and String2 is the string that will overwrite String1.

**Example:**

select stuff(‘SQL Tutorial’,1,3,’Python’)

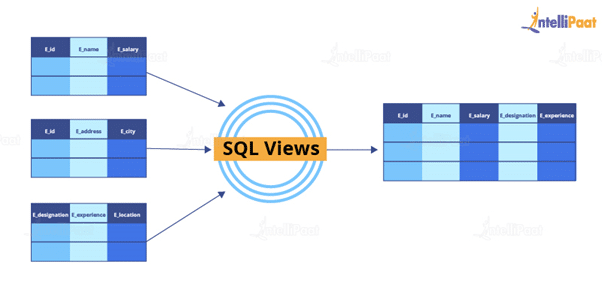
This will change ‘SQL Tutorial’ to ‘Python Tutorial’

**Output:**

Python Tutorial

**54. What are views? Give an example.**

Views are virtual tables used to limit the tables that we want to display. Views are nothing but the result of an SQL statement that has a name associated with it. Since views are not physically present, they take less space to store.

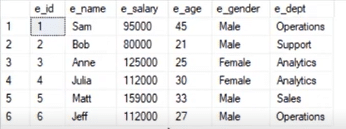


Let us consider an example. In the following employee table, say we want to perform multiple operations on the records with the gender “Female”. We can create a view-only table for the female employees from the entire employee table.

Now, let us implement it on SQL Server.

This is the employee table:

select \* from employee



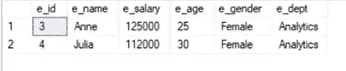
Now, we will write the syntax for the view.

**Syntax:**

create view female\_employee as select \* from employee where e\_gender=’Female’

select \* from female\_employee

**Output:**



**55. What are the types of views in SQL?**

In SQL, the views are classified into four types. They are the following:

* **Simple View:** It is a view based on a single table and does not have a GROUP BY clause or other features.
* **Complex View:** It is a view built from several tables and includes a GROUP BY clause as well as functions.
* **Inline View:** It is a view built on a subquery in the FROM clause, which provides a temporary table and simplifies a complicated query.
* **Materialized View:** It is a view that saves both the definition and the details. It builds data replicas by physically preserving them.

**Advanced SQL Interview Questions and Answers For Experienced**

**56. What is a stored procedure? Give an example.**

A stored procedure is a prepared SQL code that can be saved and reused. In other words, we can consider a stored procedure to be a function consisting of many SQL statements to access the database system. We can consolidate several SQL statements into a stored procedure and execute them whenever and wherever required.

A stored procedure can be used as a means of modular programming, i.e., we can create a stored procedure once, store it, and call it multiple times as required. This also supports faster execution when compared to executing multiple queries.

**Syntax:**

CREATE PROCEDURE procedure\_name

AS

Sql\_statement

GO;

To execute we will use this:

EXEC procedure\_name

**Example:**

We are going to create a stored procedure that will help us extract the age of the employees.

create procedure employee\_age

as

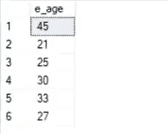
select e\_age from employee

go

Now, we will execute it.

exec employee\_age

**Output:**



**57. Explain Inner Join with an example.**

Inner Join gives us those records that have matching values in two tables.

Let us assume that we have two tables: Table A and Table B. When we apply Inner Join to these two tables, we will get only records common to both Table A and Table B.

**Syntax:**

SELECT columns

FROM table1

INNER JOIN table2

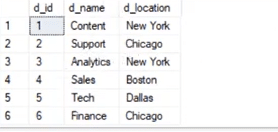
ON table1.column\_x=table2.column\_y;

**Example:**

select \* from employee

select \* from department

**Output:**



Now, we will apply Inner Join to both these tables, where the e\_dept column in the employee table is equal to the d\_name column of the department table.

**Syntax:**

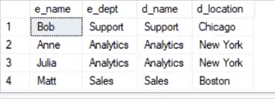
select employee.e\_name, employee.e\_dept, department.d\_name, department.d\_location

from employee inner join department

on

employee.e\_dept=department.d\_name

**Output:**



After applying Inner Join, we have only those records where the departments match in both tables. As we can see, the matched departments are Support, Analytics, and Sales.

**58. State the differences between views and tables.**

|  |  |
| --- | --- |
| **Views** | **Tables** |
| A view is a virtual table that is extracted from a database. | A table is structured with a set number of columns and a boundless number of rows. |
| A view does not hold the data itself. | A table contains data and stores it in databases. |
| A view is utilized to query certain information contained in a few distinct tables. | A table holds fundamental client information and cases of a characterized object. |
| In a view, we will get frequently queried information. | In a table, changing the information in the database changes the information that appears in the view. |

**59. What do you understand about a temporary table? Write a query to create a temporary table**

A temporary table helps us store and process intermediate results. Temporary tables are created and can be automatically deleted when they are no longer used. They are very useful in places where temporary data needs to be stored.

**Syntax:**

CREATE TABLE #table\_name();

The below query will create a temporary table:

create table #book(b\_id int, b\_cost int)

Now, we will insert the records.

insert into #book values(1,100)

insert into #book values(2,232)

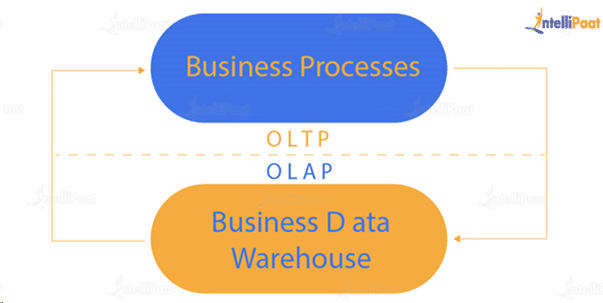
select \* from #book

**Output:**



**60. Explain the difference between OLTP and OLAP.**

**OLTP:** It stands for online transaction processing, and we can consider it a category of software applications that are efficient for supporting transaction-oriented programs. One of the important attributes of the OLTP system is its potential to maintain consistency. The OLTP system often follows decentralized planning to avoid single points of failure. This system is generally designed for a large audience of end users to perform short transactions. The queries involved in such databases are generally simple, require a fast response time, and, in comparison, return only a few records. So, the number of transactions per second acts as an effective measure for those systems.



**OLAP:**It stands for online analytical processing, and it is a category of software programs that are identified by a comparatively lower frequency of online transactions. For OLAP systems, the efficiency of computing depends heavily on the response time. Hence, such systems are generally used for data mining or maintaining aggregated historical data, and they are usually used in multidimensional schemas.

**61. What is Hybrid OLAP?**

Hybrid OLAP (HOLAP) uses a combination of multidimensional data structures and relational database tables to store multidimensional data. The aggregations for a HOLAP partition are stored by analysis services in a multidimensional structure. The facts are stored in a relational database.

**62. What do you understand by Self Join? Explain using an example**

[Self Join in SQL](https://intellipaat.com/blog/self-join-in-sql/) is used for joining a table with itself. Here, depending on some conditions, each row of the table is joined with itself and with other rows of the table.

**Syntax:**

SELECT a.column\_name, b.column\_name

FROM table a, table b

WHERE condition

**Example:**

Consider the customer table given below.

**Example:**

Consider the customer table given below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Name** | **Age** | **Address** | **Salary** |
| 1 | Anand | 32 | Ahmedabad | 2,000.00 |
| 2 | Abhishek | 25 | Delhi | 1,500.00 |
| 3 | Shivam | 23 | Kota | 2,000.00 |
| 4 | Vishal | 25 | Mumbai | 6,500.00 |
| 5 | Sayeedul | 27 | Bhopal | 8,500.00 |
| 6 | Amir | 22 | MP | 4,500.00 |
| 7 | Arpit | 24 | Indore | 10,000.00 |

We will now join the table using Self Join:

SQL> SELECT a.ID, b.NAME, a.SALARY

FROM CUSTOMERS a, CUSTOMERS b

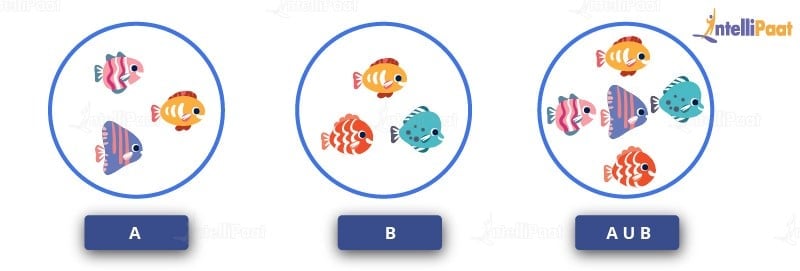
WHERE a.SALARY < b.SALARY;

**Output:**

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Salary** |
| 2 | Anand | 1,500.00 |
| 2 | Abhishek | 1,500.00 |
| 1 | Vishal | 2,000.00 |
| 2 | Vishal | 1,500.00 |
| 3 | Vishal | 2,000.00 |
| 6 | Vishal | 4,500.00 |
| 1 | Sayeedul | 2,000.00 |
| 2 | Sayeedul | 1,500.00 |
| 3 | Sayeedul | 2,000.00 |
| 4 | Sayeedul | 6,500.00 |
| 6 | Sayeedul | 4,500.00 |
| 1 | Amir | 2,000.00 |
| 2 | Amir | 1,500.00 |
| 3 | Amir | 2,000.00 |
| 1 | Arpit | 2,000.00 |
| 2 | Arpit | 1,500.00 |
| 3 | Arpit | 2,000.00 |
| 4 | Arpit | 6,500.00 |
| 5 | Arpit | 8,500.00 |
| 6 | Arpit | 4,500.00 |

**63. What is the difference between Union and Union All operators?**

The [union operator](https://intellipaat.com/blog/tutorial/sql-tutorial/union-operator/) is used to combine the result set of two or more select statements. For example, the first select statement returns the fish shown in Image A, and the second statement returns the fish shown in Image B. The Union operator will then return the result of the two select statements as shown in Image A U B. If there is a record present in both tables, then we will get only one of them in the final result.



**Syntax:**

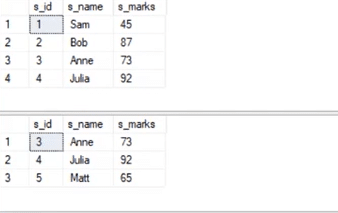
SELECT column\_list FROM table1

**Union:**

SELECT column\_list FROM table2

Now, we will execute it in the SQL Server.

These are the two tables in which we will use the Union operator.

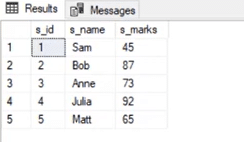


select \* from student\_details1

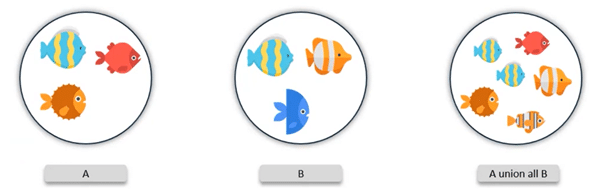
**Union:**

select \* from student\_details2

**Output:**



The Union All operator gives all the records from both tables including the duplicates.



Let us implement it in the SQL Server.

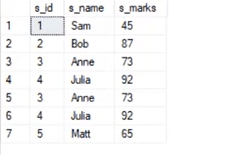
**Syntax:**

select \* from student\_details1

**Union All:**

select \* from student\_details2

**Output:**



**64. What is a database cursor? How to use a database cursor?**

A database cursor is a control that allows you to navigate around a table’s rows or documents. It can be referred to as a pointer for a row in a set of rows. Cursors are extremely useful for database traversal operations such as extraction, insertion, and elimination.

* After any variable declaration, DECLARE a cursor. A SELECT statement must always be aligned with the cursor declaration.
* To initialize the result set, OPEN statements must be called before fetching the rows from the result table.
* To grab and switch to the next row in the result set, use the FETCH statement.
* To deactivate the cursor, use the CLOSE expression.
* Finally, use the DEALLOCATE clause to uninstall the cursor description and clear all the resources associated with it.

Here is an example SQL cursor:

DECLARE @name VARCHAR(50)

DECLARE db\_cursor CURSOR FOR

SELECT name

From myDB.company

WHERE employee\_name IN (‘Jay’, ‘Shyam’)

OPEN db\_cursor

FETCH next

FROM db\_cursor

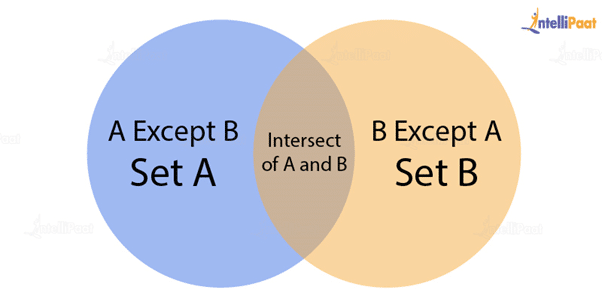
Into @name

Close db\_cursor

DEALLOCATE db\_cursor

**65. What is the use of the INTERSECT operator?**

The INTERSECT operator helps combine two select statements and returns only those records that are common to both select statements. After we get Table A and Table B over here, and if we apply the INTERSECT operator on these two tables, then we will get only those records that are common to the result of the select statements of these two tables.



**Syntax:**

SELECT column\_list FROM table1

INTERSECT

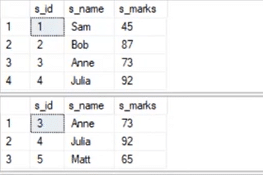
SELECT column\_list FROM table2

Now, let us take a look at an example of the INTERSECT operator.

select \* from student\_details1

select \* from student\_details1

**Output:**



select \* from student\_details1

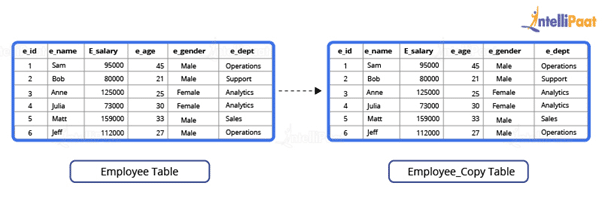
intersect

select \* from student\_details2

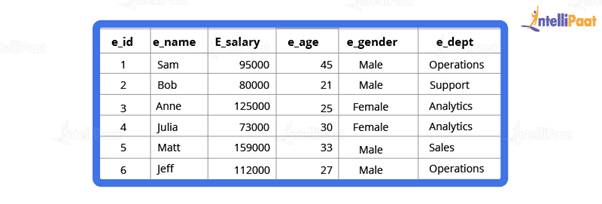
**Output:**



**66. How can you copy data from one table to another table?**



Here, we have our employee table.



We have to copy this data into another table. For this purpose, we can use the INSERT INTO SELECT operator. Before we go ahead and do that, we will have to create another table that will have the same structure as the above-given table.

**Syntax:**

create table employee\_duplicate(

e\_id int,

e\_name varchar(20),

e\_salary int,

e\_age int,

e\_gender varchar(20)

e\_dept varchar(20)

)

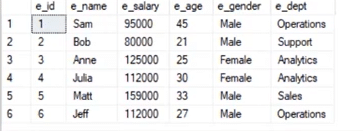
For copying the data, we will use the following query:

insert into employee\_duplicate select \* from employees

Let us take a look at the copied table.

select \* from employee\_duplicate

**Output:**



**67. What is the difference between BETWEEN and IN operators in SQL?**

The BETWEEN operator is employed to identify rows that fall within a specified range of values, encompassing numerical, textual, or date values. It returns the count of values that exist between the two defined boundaries.

On the other hand, the IN operator serves as a condition operator utilized for searching values within a predetermined range. When multiple values are available for selection, the IN operator is utilized.

***Check out how to use***[***IN and BETWEEN Operators in SQL***](https://intellipaat.com/blog/tutorial/sql-tutorial/sql-between/)***with examples.***

**68. Describe how to delete duplicate rows using a single statement but without any table creation.**

Let us create an employee table where the column names are ID, NAME, DEPARTMENT, and EMAIL. Below are the SQL scripts for generating the sample data:

CREATE TABLE EMPLOYEE

(

ID INT,

NAME Varchar(100),

DEPARTMENT INT,

EMAIL Varchar(100)

)

INSERT INTO EMPLOYEE VALUES (1,'Tarun',101,'tarun@intellipaat.com')

INSERT INTO EMPLOYEE VALUES (2,'Sabid',102,'sabid@intellipaat.com')

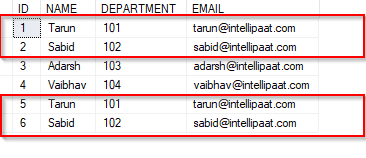
INSERT INTO EMPLOYEE VALUES (3,'Adarsh',103,'adarsh@intellipaat.com')

INSERT INTO EMPLOYEE VALUES (4,'Vaibhav',104,'vaibhav@intellipaat.com')

–These are the duplicate rows:

INSERT INTO EMPLOYEE VALUES (5,'Tarun',101,'tarun@intellipaat.com')

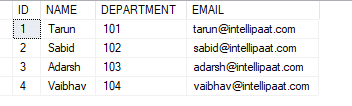
INSERT INTO EMPLOYEE VALUES (6,'Sabid',102,'sabid@intellipaat.com')



We can see the duplicate rows in the above table.

DELETE e1 FROM EMPLOYEE e1, EMPLOYEE e2 WHERE e1.name = e2.name AND e1.id > e2.id

The SQL query above will delete the rows, where the name fields are duplicated, and it will retain only those unique rows in which the names are unique and the ID fields are the lowest, i.e., the rows with IDs 5 and 6 are deleted, while the rows with IDs 1 and 2 are retained.



**69. Can you identify the employee who has the third-highest salary from the given employee table (with salary-related data)?**

Consider the following employee table. In the table, Sabid has the third-highest salary (60,000).

|  |  |
| --- | --- |
| **Name** | **Salary** |
| Tarun | 70,000 |
| Sabid | 60,000 |
| Adarsh | 30,000 |
| Vaibhav | 80,000 |

Below is a simple query to find out which employee who has the third-highest salary. The functions RANK, DENSE RANK, and ROW NUMBER are used to obtain the increasing integer value by imposing the ORDER BY clause in the SELECT statement, based on the ordering of the rows. The ORDER BY clause is necessary when the RANK, DENSE RANK, or ROW NUMBER functions are used. On the other hand, the PARTITION BY clause is optional.

WITH CTE AS

(

    SELECT Name, Salary, RN = ROW\_NUMBER() OVER (ORDER BY Salary DESC) FROM EMPLOYEE

)

SELECT Name, Salary FROM CTE WHERE RN =3

**70. What is the difference between HAVING and WHERE clauses?**

The main difference between the ‘HAVING’ and ‘WHERE’ clauses in SQL is that the ‘WHERE’ clause operates on individual rows of data, while the ‘HAVING’ clause is used to filter aggregated data. The [**‘WHERE’ clause**](https://intellipaat.com/blog/tutorial/sql-tutorial/where-clause/) cannot be used with aggregate functions, whereas the ‘HAVING’ clause specifically filters results based on aggregate conditions.

Let us consider the employee table below.

|  |  |  |
| --- | --- | --- |
| **Name** | **Department** | **Salary** |
| Tarun | Production | 50,000 |
| Tarun | Testing | 60,000 |
| Sabid | Marketing | 70,000 |
| Adarsh | Production | 80,000 |
| Vaibhav | Testing | 90,000 |

The following will select the data on a row-by-row basis:

SELECT Name, Salary FROM Employee WHERE Salary >=50000

**Output:**

|  |  |
| --- | --- |
| **Name** | **Salary** |
| Tarun | 50,000 |
| Tarun | 60,000 |
| Sabid | 70,000 |
| Adarsh | 80,000 |
| Vaibhav | 90,000 |

The HAVING clause, on the other hand, operates on the aggregated results.

SELECT Department, SUM(Salary) AS total FROM Employee GROUP BY Department

**Output:**

|  |  |
| --- | --- |
| **Department** | **Total** |
| Marketing | 70,000 |
| Production | 130,000 |
| Testing | 150,000 |

Now, let us see the output when we apply HAVING to the above query.

SELECT Department, SUM(Salary) AS total FROM Employee GROUP BY Department HAVING SUM(Salary)>70000

**Output:**

|  |  |
| --- | --- |
| **Department** | **Total** |
| Production | 130,000 |
| Testing | 150,000 |

**71. Explain database white box testing and black box testing.**

The [white box testing](https://intellipaat.com/blog/white-box-penetration-testing/) method mainly deals with the internal structure of a particular database, where users hide specification details. The white box testing method involves the following:

* As the coding error can be detected by testing the white box, it can eliminate internal errors.
* To check for the consistency of the database, it selects the default table values.
* This method verifies the referential integrity rule.
* It helps perform the module testing of database functions, triggers, views, and SQL queries.

The black box testing method generally involves interface testing and database integration. The black box testing method involves the following:

* Mapping details
* Verification of incoming data
* Verification of outgoing data from the other query functions

**72. How can you create empty tables with the same structure as another table?**

This can be achieved by fetching the records of one table into a new table using the INTO operator while fixing a WHERE clause to be false for all records. In this way, SQL prepares the new table with a duplicate structure to accept the fetched records. However, there are no records that will get fetched due to the WHERE clause in action. Therefore, nothing is inserted into the new table, thus creating an empty table.

SELECT \* INTO Students\_copy

FROM Students WHERE 1 = 2;

**Basic SQL Interview Questions**

All set to kickstart your career in SQL? Look no further and start your professional career with these SQL interview questions for freshers. We will start with the basics and slowly move towards slightly advanced questions to set the pace. If you are an experienced professional, this section will help you brush up on your SQL skills.

**What is SQL?**

The acronym SQL stands for Structured Query Language. It is the typical language used for relational database maintenance and a wide range of data processing tasks. The first SQL database was created in 1970. It is a database language used for operations such as database creation, deletion, retrieval, and row modification. It is occasionally pronounced “sequel.” It can also be used to manage structured data, which is made up of variables called entities and relationships between those entities.

**What is Database?**

A [database](https://www.mygreatlearning.com/academy/learn-for-free/courses/crash-course-on-database-management#gl_blog_id=15225) is a system that helps in collecting, storing and retrieving data. Databases can be complex, and such databases are developed using design and modelling approaches.

**What is DBMS?**

[DBMS](https://www.mygreatlearning.com/blog/dbms-tutorial/) stands for Database Management System which is responsible for the creating, updating, and managing of the database.

**What is RDBMS? How is it different from DBMS?**

RDBMS stands for Relational Database Management System that stores data in the form of a collection of tables, and relations can be defined between the common fields of these tables.

**How to create a table in SQL?**

The command to create a table in SQL is extremely simple:

CREATE TABLE table\_name (

column1 datatype,

column2 datatype,

column3 datatype,

....

);

We will start off by giving the keywords, CREATE TABLE, and then we will give the name of the table. After that in braces, we will list out all the columns along with their [data types](https://www.mygreatlearning.com/blog/types-of-data/).

For example, if we want to create a simple employee table:

CREATE TABLE employee (

name varchar(25),

age int,

gender varchar(25),

....

);

**How to delete a table in SQL?**

There are two ways to delete a table from SQL: DROP and TRUNCATE. The DROP TABLE command is used to completely delete the table from the database. This is the command:

DROP TABLE *table\_name;*

The above command will completely delete all the data present in the table along with the table itself.

But if we want to delete only the data present in the table but not the table itself, then we will use the truncate command:

DROP TABLE *table\_name ;*

**How to change a table name in SQL?**

This is the command to change a table name in SQL:

**ALTER TABLE table\_name**

**RENAME TO new\_table\_name;**

We will start off by giving the keywords ALTER TABLE, then we will follow it up by giving the original name of the table, after that, we will give in the keywords RENAME TO and finally, we will give the new table name.

For example, if we want to change the “employee” table to “employee\_information”, this will be the command:

ALTER TABLE employee

RENAME TO employee\_information;

**How to delete a row in SQL?**

We will be using the DELETE query to delete existing rows from the table:

DELETE FROM table\_name

WHERE [condition];

We will start off by giving the keywords DELETE FROM, then we will give the name of the table, and after that we will give the WHERE clause and give the condition on the basis of which we would want to delete a row.

For example, from the employee table, if we would like to delete all the rows, where the age of the employee is equal to 25, then this will be the command:

DELETE FROM employee

WHERE [age=25];

**How to create a database in SQL?**

A database is a repository in SQL, which can comprise multiple tables.

This will be the command to create a database in sql:

CREATE DATABASE *database\_name.*

**What is Normalization in SQL?**

Normalization is used to decompose a larger, complex table into simple and smaller ones. This helps us in removing all the redundant data.

Generally, in a table, we will have a lot of redundant information which is not required, so it is better to divide this complex table into multiple smaller tables which contain only unique information.

**First normal form:**

A relation schema is in 1NF, if and only if:

* All attributes in the relation are atomic(indivisible value)
* And there are no repeating elements or groups of elements.

**Second normal form:**

A relation is said to be in 2NF, if and only if:

* It is in 1st Normal Form.
* No partial dependency exists between non-key attributes and key attributes.

**Third Normal form:**

A relation R is said to be in 3NF if and only if:

* It is in 2NF.
* No transitive dependency exists between non-key attributes and key attributes through another non-key attribute

**What is join in SQL?**

Joins are used to combine rows from two or more tables, based on a related column between them.

Types of Joins:

• **INNER JOIN** − Returns rows when there is a match in both tables.

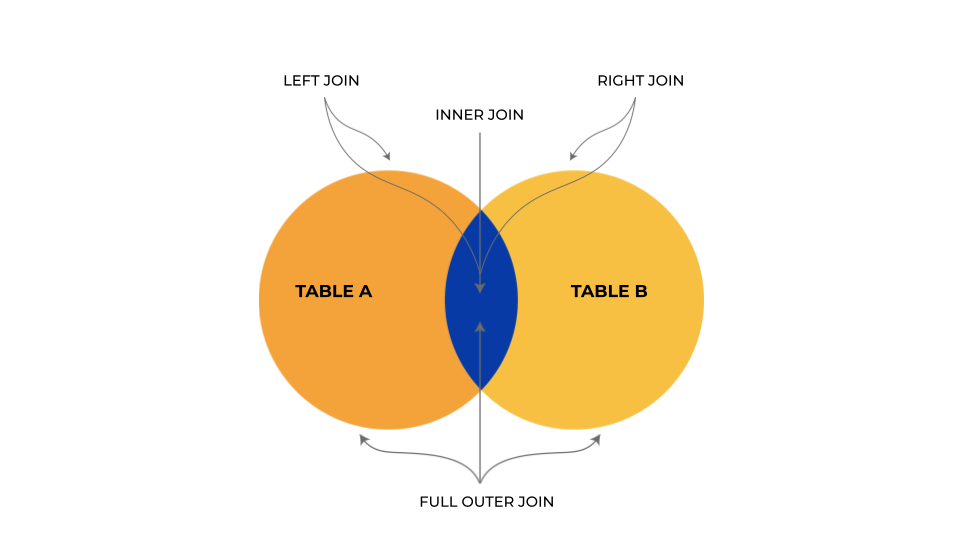
• **LEFT JOIN** − Returns all rows from the left table, even if there are no matches in the right table.

• **RIGHT JOIN** − Returns all rows from the right table, even if there are no matches in the left table.

• **FULL OUTER JOIN** − Returns rows when there is a match in one of the tables.

• **SELF JOIN** − Used to join a table to itself as if the table were two tables, temporarily renaming at least one table in the SQL statement.

• **CARTESIAN JOIN (CROSS JOIN)** − Returns the Cartesian product of the sets of records from the two or more joined tables.



**INNER JOIN:**

The INNER JOIN creates a new result table by combining column values of two tables (table1 and table2) based upon the join-predicate. The query compares each row of table1 with each row of table2 to find all pairs of rows which satisfy the join-predicate.

**SYNTAX:**

SELECT table1.col1, table2.col2,…, table1.coln

FROM table1

INNER JOIN table2

ON table1.commonfield = table2.commonfield;

**LEFT JOIN:**

The LEFT JOIN returns all the values from the left table, plus matched values from the right table or NULL in case of no matching join predicate.

**SYNTAX:**

SELECT table1.col1, table2.col2,…, table1.coln

FROM table1

LEFT JOIN table2

ON table1.commonfield = table2.commonfield;

**RIGHT JOIN:**

The RIGHT JOIN returns all the values from the right table, plus matched values from the left table or NULL in case of no matching join predicate.

**SYNTAX:**

SELECT table1.col1, table2.col2,…, table1.coln

FROM table1

RIGHT JOIN table2

ON table1.commonfield = table2.commonfield;

**FULL OUTER JOIN:**

The FULL OUTER JOIN combines the results of both left and right outer joins. The joined table will contain all records from both the tables and fill in NULLs for missing matches on either side.

**SYNTAX:**

SELECT table1.col1, table2.col2,…, table1.coln

FROM table1

Left JOIN table2

ON table1.commonfield = table2.commonfield;

Union

SELECT table1.col1, table2.col2,…, table1.coln

FROM table1

Right JOIN table2

ON table1.commonfield = table2.commonfield;

**SELF JOIN:**

The SELF JOIN joins a table to itself; temporarily renaming at least one table in the SQL statement.

**SYNTAX:**

SELECT a.col1, b.col2,..., a.coln

FROM table1 a, table1 b

WHERE a.commonfield = b.commonfield;

**Joins in SQL**

[](https://www.mygreatlearning.com/academy/learn-for-free/courses/joins-in-sql-1?gl_blog_id=85199)

[GREAT LEARNING ACADEMY](https://www.mygreatlearning.com/academy/learn-for-free/courses/joins-in-sql-1?gl_blog_id=85199" \t "_blank)

**[Joins in SQL](https://www.mygreatlearning.com/academy/learn-for-free/courses/joins-in-sql-1?gl_blog_id=85199" \t "_blank)**

[1.5 Hours Online](https://www.mygreatlearning.com/academy/learn-for-free/courses/joins-in-sql-1?gl_blog_id=85199" \t "_blank)

[Land your dream job](https://www.mygreatlearning.com/academy/learn-for-free/courses/joins-in-sql-1?gl_blog_id=85199" \t "_blank)

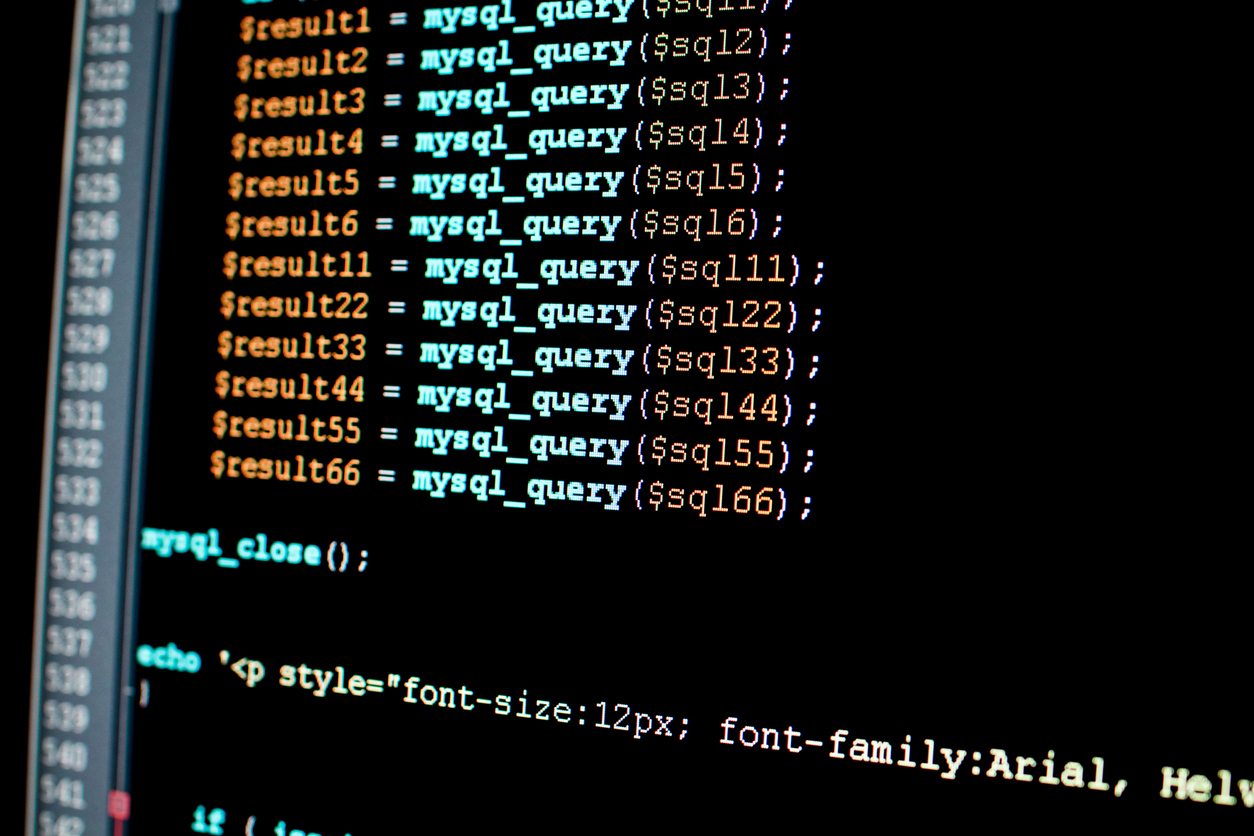
[](https://www.mygreatlearning.com/academy/learn-for-free/courses/sql-projects-for-beginners?gl_blog_id=85199)

[GREAT LEARNING ACADEMY](https://www.mygreatlearning.com/academy/learn-for-free/courses/sql-projects-for-beginners?gl_blog_id=85199" \t "_blank)

**[SQL Projects for Beginners](https://www.mygreatlearning.com/academy/learn-for-free/courses/sql-projects-for-beginners?gl_blog_id=85199" \t "_blank)**

[0.5 Hours Online](https://www.mygreatlearning.com/academy/learn-for-free/courses/sql-projects-for-beginners?gl_blog_id=85199" \t "_blank)

[Stand out to recruiters](https://www.mygreatlearning.com/academy/learn-for-free/courses/sql-projects-for-beginners?gl_blog_id=85199" \t "_blank)

[](https://www.mygreatlearning.com/academy/learn-for-free/courses/advanced-sql?gl_blog_id=85199)

[GREAT LEARNING ACADEMY](https://www.mygreatlearning.com/academy/learn-for-free/courses/advanced-sql?gl_blog_id=85199" \t "_blank)

**[Advanced SQL](https://www.mygreatlearning.com/academy/learn-for-free/courses/advanced-sql?gl_blog_id=85199" \t "_blank)**

[1.0 Hourse Online](https://www.mygreatlearning.com/academy/learn-for-free/courses/advanced-sql?gl_blog_id=85199" \t "_blank)

[Land your dream job](https://www.mygreatlearning.com/academy/learn-for-free/courses/advanced-sql?gl_blog_id=85199" \t "_blank)



**How to insert a date in SQL?**

If the RDBMS is MYSQL, this is how we can insert date:

*"INSERT INTO tablename (col\_name, col\_date) VALUES ('DATE: Manual Date', '2020-9-10')";*

**What is Primary Key in SQL?**

Primary Key is a constraint in SQL. So, before understanding what exactly is a primary key, let’s understand what exactly is a constraint in SQL. Constraints are the rules enforced on data columns on a table. These are used to limit the type of data that can go into a table. Constraints can either be column level or table level.

Let’s look at the different types of constraints which are present in SQL:

|  |  |
| --- | --- |
| **Constraint** | **Description** |
| **NOT NULL** | Ensures that a column cannot have a NULL value. |
| **DEFAULT** | Provides a default value for a column when none is specified. |
| **UNIQUE** | Ensures that all the values in a column are different |
| **PRIMARY** | Uniquely identifies each row/record in a database table |
| **FOREIGN** | Uniquely identifies a row/record in any another database table |
| **CHECK** | The CHECK constraint ensures that all values in a column satisfy certain conditions. |
| **INDEX** | Used to create and retrieve data from the database very quickly. |

You can consider the Primary Key constraint to be a combination of UNIQUE and NOT NULL constraint. This means that if a column is set as a primary key, then this particular column cannot have any null values present in it and also all the values present in this column must be unique.

**How do I view tables in SQL?**

To view tables in SQL, all you need to do is give this command:

Show tables;

**What is PL/SQL?**

PL SQL stands for Procedural language constructs for Structured Query Language. [PL SQL](https://www.mygreatlearning.com/blog/pl-sql-tutorial/) was introduced by Oracle to overcome the limitations of plain sql. So, pl sql adds in procedural language approach to the plain vanilla sql.

One thing to be noted over here is that pl sql is only for oracle databases. If you don’t have an [Oracle database](https://www.mygreatlearning.com/academy/learn-for-free/courses/oracle-sql/#gl_blog_id=15225), then you cant work with PL SQL. However, if you wish to learn more about Oracle, you can also take up[free oracle courses](https://www.mygreatlearning.com/oracle/free-courses#gl_blog_id=15225)and enhance your knowledge.

While, with the help of sql, we were able to DDL and DML queries, with the help of PL SQL, we will be able to create functions, triggers and other procedural constructs.

**How can I see all tables in SQL?**

Different database management systems have different queries to see all the tables.

To see all the tables in MYSQL, we would have to use this query:

**show** **tables**;

This is how we can see all tables in ORACLE:

SELECT

table\_name

FROM

User\_tables;

This is how we can extract all tables in SQL Server:

SELECT

\*

FROM

Information\_schema.tables;

**What is ETL in SQL?**

ETL stands for Extract, Transform and Load. It is a three-step process, where we would have to start off by extracting the data from sources. Once we collate the data from different sources, what we have is raw data. This raw data has to be transformed into the tidy format, which will come in the second phase. Finally, we would have to load this tidy data into tools which would help us to find insights.

**How to install SQL?**

SQL stands for Structured Query Language and it is not something you can install. To implement sql queries, you would need a relational database management system. There are different varieties of relational database management systems such as:

* ORACLE
* MYSQL
* SQL Server

Hence, to implement sql queries, we would need to install any of these Relational Database Management Systems.

**What is the update command in SQL?**

The update command comes under the DML(Data Manipulation Langauge) part of sql and is used to update the existing data in the table.

UPDATE employees

SET last\_name=‘Cohen’

WHERE employee\_id=101;

With this update command, I am changing the last name of the employee.

**How to rename column name in SQL Server?**

[Rename column in SQL](https://www.mygreatlearning.com/blog/how-to-rename-column-name-in-sql/): When it comes to SQL Server, it is not possible to rename the column with the help of ALTER TABLE command, we would have to use sp\_rename.

**What are the types of SQL Queries?**

We have four types of SQL Queries:

* **DDL** (Data Definition Language): the creation of objects
* **DML** (Data Manipulation Language): manipulation of data
* **DCL** (Data Control Language): assignment and removal of permissions
* **TCL**(Transaction Control Language): saving and restoring changes to a database

Let’s look at the different commands under DDL:

|  |  |
| --- | --- |
| **Command** | **Description** |
| **CREATE** | Create objects in the database |
| **ALTER** | Alters the structure of the database object |
| **DROP** | Delete objects from the database |
| **TRUNCATE** | Remove all records from a table permanently |
| **COMMENT** | Add comments to the data dictionary |
| **RENAME** | Rename an object |

**Write a Query to display the number of employees working in each region?**

SELECT region, COUNT(gender) FROM employee GROUP BY region;

**What are Nested Triggers?**

Triggers may implement DML by using INSERT, UPDATE, and DELETE statements. These triggers that contain DML and find other triggers for data modification are called Nested Triggers.

**Write SQL query to fetch employee names having a salary greater than or equal to 20000 and less than or equal 10000.**

By using BETWEEN in the where clause, we can retrieve the Employee Ids of employees with salary >= 20000 and <=10000.

SELECT FullName FROM EmployeeDetails WHERE EmpId IN (SELECT EmpId FROM EmployeeSalary WHERE Salary BETWEEN 5000 AND 10000)

**Given a table Employee having columns empName and empId, what will be the result of the SQL query below? select empName from Employee order by 2 asc;**

“Order by 2” is valid when there are at least 2 columns used in SELECT statement. Here this query will throw error because only one column is used in the SELECT statement.

**What is OLTP?**

OLTP stands for Online Transaction Processing. And is a class of software applications capable of supporting transaction-oriented programs. An essential attribute of an OLTP system is its ability to maintain concurrency.

**What is Data Integrity?**

Data Integrity is the assurance of accuracy and consistency of data over its entire life-cycle, and is a critical aspect to the design, implementation and usage of any system which stores, processes, or retrieves data. It also defines integrity constraints to enforce business rules on the data when it is entered into an application or a database.

**What is OLAP?**

OLAP stands for Online Analytical Processing. And a class of software programs which are characterized by relatively low frequency of online transactions. Queries are often too complex and involve a bunch of aggregations.

**Find the Constraint information from the table?**

There are so many times where user needs to find out the specific constraint information of the table. The following queries are useful, SELECT \* From User\_Constraints; SELECT \* FROM User\_Cons\_Columns;

**Can you get the list of employees with same salary?**

Select distinct e.empid,e.empname,e.salary from employee e, employee e1 where e.salary =e1.salary and e.empid != e1.empid

**What is an alternative for the TOP clause in SQL?**

1. ROWCOUNT function   
2. Set rowcount 3  
3. Select \* from employee order by empid desc Set rowcount 0

**Will the following statement gives an error or 0 as output? SELECT AVG (NULL)**

Error. Operand data type NULL is invalid for the Avg operator.

**What is the Cartesian product of the table?**

The output of Cross Join is called 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.

**What is a schema in SQL?**

Our database comprises of a lot of different entities such as tables, stored procedures, functions, database owners and so on. To make sense of how all these different entities interact, we would need the help of schema. So, you can consider schema to be the logical relationship between all the different entities which are present in the database.

Once we have a clear understanding of the schema, this helps in a lot of ways:

* We can decide which user has access to which tables in the database.
* We can modify or add new relationships between different entities in the database.

Overall, you can consider a schema to be a blueprint for the database, which will give you the complete picture of how different objects interact with each other and which users have access to different entities.

**How to delete a column in SQL?**

To delete a column in SQL we will be using DROP COLUMN method:

ALTER TABLE employees

DROP COLUMN age;

We will start off by giving the keywords ALTER TABLE, then we will give the name of the table, following which we will give the keywords DROP COLUMN and finally give the name of the column which we would want to remove.

**What is a unique key in SQL?**

Unique Key is a constraint in SQL. So, before understanding what exactly is a primary key, let’s understand what exactly is a constraint in SQL. Constraints are the rules enforced on data columns on a table. These are used to limit the type of data that can go into a table. Constraints can either be column level or table level.

**Unique Key:**

Whenever we give the constraint of unique key to a column, this would mean that the column cannot have any duplicate values present in it. In other words, all the records which are present in this column have to be unique.

**How to implement multiple conditions using the WHERE clause?**

We can implement multiple conditions using AND, OR operators:

SELECT \* FROM employees WHERE first\_name = ‘Steven’ AND salary <=10000;

In the above command, we are giving two conditions. The condition ensures that we extract only those records where the first name of the employee is ‘Steven’ and the second condition ensures that the salary of the employee is less than $10,000. In other words, we are extracting only those records, where the employee’s first name is ‘Steven’ and this person’s salary should be less than $10,000.

**What is the difference between SQL vs PL/SQL?**

|  |  |  |
| --- | --- | --- |
| **BASIS FOR COMPARISON** | **SQL** | **PL/SQL** |
| **Basic** | In SQL you can execute a single query or a command at a time. | In PL/SQL you can execute a block of code at a time. |
| **Full form** | Structured Query Language | Procedural Language, an extension of SQL. |
| **Purpose** | It is like a source of data that is to be displayed. | It is a language that creates an application that displays data acquired by SQL. |
| **Writes** | In SQL you can write queries and commands using DDL, DML statements. | In PL/SQL you can write a block of code that has procedures, functions, packages or variables, etc. |
| **Use** | Using SQL, you can retrieve, modify, add, delete, or manipulate the data in the database. | Using PL/SQL, you can create applications or server pages that display the information obtained from SQL in a proper format. |
| **Embed** | You can embed SQL statements in PL/SQL. | You can not embed PL/SQL in SQL |

**What is the difference between SQL having vs where?**

|  |  |  |
| --- | --- | --- |
| S. No. | Where Clause | Having Clause |
| 1 | The WHERE clause specifies the criteria which individual records must meet to be selected by a query. It can be used without the GROUP by clause | The HAVING clause cannot be used without the GROUP BY clause |
| 2 | The WHERE clause selects rows before grouping | The HAVING clause selects rows after grouping |
| 3 | The WHERE clause cannot contain aggregate functions | The HAVING clause can contain aggregate functions |
| 4 | WHERE clause is used to impose a condition on SELECT statement as well as single row function and is used before GROUP BY clause | HAVING clause is used to impose a condition on GROUP Function and is used after GROUP BY clause in the query |
| 5 | SELECT Column,AVG(Column\_nmae)FROM Table\_name WHERE Column > value GROUP BY Column\_nmae | SELECT Columnq, AVG(Coulmn\_nmae)FROM Table\_name WHERE Column > value GROUP BY Column\_nmae Having column\_name>or<value |

**SQL Interview Questions for Experienced**

Planning to switch your career to SQL or just need to upgrade your position? Whatever your reason, this section will better prepare you for the SQL interview. We have compiled a set of advanced SQL questions that may be frequently asked during the interview.

**What is SQL injection?**

SQL injection is a hacking technique which is widely used by black-hat hackers to steal data from your tables or databases. Let’s say, if you go to a website and give in your user information and password, the hacker would add some malicious code over there such that, he can get the user information and password directly from the database. If your database contains any vital information, it is always better to keep it secure from SQL injection attacks.

**What is a trigger in SQL?**

A trigger is a stored program in a database which automatically gives responses to an event of DML operations done by inserting, update, or delete. In other words, is nothing but an auditor of events happening across all database tables.

Let’s look at an example of a trigger:

CREATE TRIGGER bank\_trans\_hv\_alert

BEFORE UPDATE ON bank\_account\_transaction

FOR EACH ROW

begin

if( abs(:new.transaction\_amount)>999999)THEN

RAISE\_APPLICATION\_ERROR(-20000, 'Account transaction exceeding the daily deposit on SAVINGS account.');

end if;

end;



**How to insert multiple rows in SQL?**

To insert multiple rows in SQL we can follow the below syntax:

INSERT INTO table\_name (column1, column2,column3...)

VALUES

(value1, value2, value3…..),

(value1, value2, value3….),

...

(value1, value2, value3);

We start off by giving the keywords INSERT INTO then we give the name of the table into which we would want to insert the values. We will follow it up with the list of the columns, for which we would have to add the values. Then we will give in the VALUES keyword and finally, we will give the list of values.

Here is an example of the same:

INSERT INTO employees (

name,

age,

salary)

VALUES

(

'Sam',

21,

75000

),

(

' 'Matt',

32,

85000 ),

(

'Bob',

26,

90000

);

In the above example, we are inserting multiple records into the table called employees.

**How to find the nth highest salary in SQL?**

This is how we can find the nth highest salary in SQL SERVER using TOP keyword:

*SELECT TOP 1 salary FROM ( SELECT DISTINCT TOP N salary FROM #Employee ORDER BY salary DESC ) AS temp ORDER BY salary*

This is how we can find the nth highest salary in MYSQL using LIMIT keyword:

*SELECT salary FROM Employee ORDER BY salary DESC LIMIT N-1, 1*

**How to copy table in SQL?**

We can use the SELECT INTO statement to copy data from one table to another. Either we can copy all the data or only some specific columns.

This is how we can copy all the columns into a new table:

SELECT \*

INTO newtable

FROM oldtable

WHERE condition;

If we want to copy only some specific columns, we can do it this way:

SELECT column1, column2, column3, ...

INTO newtable

FROM oldtable

WHERE condition;

**How to add a new column in SQL?**

We can add a new column in SQL with the help of alter command:

ALTER TABLE employees ADD COLUMN contact INT(10);

This command helps us to add a new column named as contact in the employees table.

**How to use LIKE in SQL?**

The LIKE operator checks if an attribute value matches a given string pattern. Here is an example of LIKE operator

SELECT \* FROM employees WHERE first\_name like ‘Steven’;

With this command, we will be able to extract all the records where the first name is like “Steven”.

**If we drop a table, does it also drop related objects like constraints, indexes, columns, default, views and sorted procedures?**

Yes, SQL server drops all related objects, which exists inside a table like constraints, indexex, columns, defaults etc. But dropping a table will not drop views and sorted procedures as they exist outside the table.

**Can we disable a trigger? If yes, How?**

Yes, we can disable a single trigger on the database by using “DISABLE TRIGGER triggerName ON<>. We also have an option to disable all the trigger by using, “DISABLE Trigger ALL ON ALL SERVER”.

**What is a Live Lock?**

A live lock is one where a request for an exclusive lock is repeatedly denied because a series of overlapping shared locks keep interferring. A live lock also occurs when read transactions create a table or page.

**How to fetch alternate records from a table?**

Records can be fetched for both Odd and Even row numbers – To display even numbers –

Select employeeId from (Select rowno, employeeId from employee) where mod(rowno,2)=0

To display odd numbers –

Select employeeId from (Select rowno, employeeId from employee) where mod(rowno,2)=1

**Define COMMIT and give an example?**

When a COMMIT is used in a transaction, all changes made in the transaction are written into the database permanently.

Example:

BEGIN TRANSACTION; DELETE FROM HR.JobCandidate WHERE JobCandidateID = 20; COMMIT TRANSACTION;

The above example deletes a job candidate in a SQL server.

**Can you join the table by itself?**

A table can be joined to itself using self join, when you want to create a result set that joins records in a table with other records in the same table.

**Explain Equi join with an example.**

When two or more tables have been joined using equal to operator then this category is called an equi join. Just we need to concentrate on the condition is equal to (=) between the columns in the table.

Example:

Select a.Employee\_name,b.Department\_name from Employee a,Employee b where a.Department\_ID=b.Department\_ID

**How do we avoid getting duplicate entries in a query?**

The SELECT DISTINCT is used to get distinct data from tables using a query. The below SQL query selects only the DISTINCT values from the “Country” column in the “Customers” table:

SELECT DISTINCT Country FROM Customers;

**How can you create an empty table from an existing table?**

Lets take an example:

Select \* into studentcopy from student where 1=2

Here, we are copying the student table to another table with the same structure with no rows copied.

**Write a Query to display odd records from student table?**

SELECT \* FROM (SELECT \*, ROW\_NUMBER() OVER (ORDER BY student\_no) AS RowID FROM student) WHERE row\_id %2!=0

**Explain Non-Equi Join with an example?**

When two or more tables are joining without equal to condition, then that join is known as Non Equi Join. Any operator can be used here, that is <>,!=,<,>,Between.

Example:

Select b.Department\_ID,b.Department\_name from Employee a,Department b where a.Department\_id <> b.Department\_ID;

**How can you delete duplicate records in a table with no primary key?**

By using the SET ROWCOUNT command. It limits the number of records affected by a command. Let’s take an example, if you have 2 duplicate rows, you would SET ROWCOUNT 1, execute DELETE command and then SET ROWCOUNT 0.

**Difference between NVL and NVL2 functions?**

Both the NVL(exp1, exp2) and NVL2(exp1, exp2, exp3) functions check the value exp1 to see if it is null. With the NVL(exp1, exp2) function, if exp1 is not null, then the value of exp1 is returned; otherwise, the value of exp2 is returned, but case to the same data type as that of exp1. With the NVL2(exp1, exp2, exp3) function, if exp1 is not null, then exp2 is returned; otherwise, the value of exp3 is returned.

**What is the difference between clustered and non-clustered indexes?**

1. Clustered indexes can be read rapidly rather than non-clustered indexes.
2. Clustered indexes store data physically in the table or view whereas, non-clustered indexes do not store data in the table as it has separate structure from the data row.

**What does this query says? GRANT privilege\_name ON object\_name TO {user\_name|PUBLIC|role\_name} [WITH GRANT OPTION];**

The given syntax indicates that the user can grant access to another user too.

**Where MyISAM table is stored?**

Each MyISAM table is stored on disk in three files.

1. The “.frm” file stores the table definition.
2. The data file has a ‘.MYD’ (MYData) extension.
3. The index file has a ‘.MYI’ (MYIndex) extension.

**What does myisamchk do?**

It compresses the MyISAM tables, which reduces their disk or memory usage.

**What is ISAM?**

ISAM is abbreviated as Indexed Sequential Access Method. It was developed by IBM to store and retrieve data on secondary storage systems like tapes.

**What is Database White box testing?**

White box testing includes: Database Consistency and ACID properties Database triggers and logical views Decision Coverage, Condition Coverage, and Statement Coverage Database Tables, Data Model, and Database Schema Referential integrity rules.

**What are the different types of SQL sandbox?**

There are 3 different types of SQL sandbox:

* Safe Access Sandbox: Here a user can perform SQL operations such as creating stored procedures, triggers etc. but cannot have access to the memory as well as cannot create files.
* External Access Sandbox: Users can access files without having the right to manipulate the memory allocation.
* Unsafe Access Sandbox: This contains untrusted codes where a user can have access to memory.

**What is Database Black Box Testing?**

This testing involves:

* Data Mapping
* Data stored and retrieved
* Use of Black Box testing techniques such as Equivalence Partitioning and Boundary Value Analysis (BVA).

**Explain Right Outer Join with Example?**

This join is usable, when user wants all the records from Right table (Second table) and only equal or matching records from First or left table. The unmatched records are considered as null records. Example: Select t1.col1,t2.col2….t ‘n’col ‘n.’. from table1 t1,table2 t2 where t1.col(+)=t2.col;

**What is a Subquery?**

A SubQuery is a SQL query nested into a larger query. Example: SELECT employeeID, firstName, lastName FROM employees WHERE departmentID IN (SELECT departmentID FROM departments WHERE locationID = 2000) ORDER BY firstName, lastName;

**SQL Interview Questions for Developers**

**How to find duplicate records in SQL?**

There are multiple ways to find duplicate records in SQL. Let’s see how can we find duplicate records using group by:

SELECT

x,

y,

COUNT(\*) occurrences

FROM z1

GROUP BY

x,

y

HAVING

COUNT(\*) > 1;

We can also find duplicates in the table using rank:

SELECT \* FROM ( SELECT eid, ename, eage, Row\_Number() OVER(PARTITION BY ename, eage ORDER By ename) AS Rank FROM employees ) AS X WHERE Rank>1

**What is Case WHEN in SQL?**

If you have knowledge about other [programming languages](https://www.mygreatlearning.com/blog/what-are-the-best-programming-languages-to-learn/), then you’d have learnt about if-else statements. You can consider Case WHEN to be analogous to that.

In Case WHEN, there will be multiple conditions and we will choose something on the basis of these conditions.

Here is the syntax for CASE WHEN:

CASE

WHEN condition1 THEN result1

WHEN condition2 THEN result2

WHEN conditionN THEN resultN

ELSE result

END;

We start off by giving the CASE keyword, then we follow it up by giving multiple WHEN, THEN statements.

**How to find 2nd highest salary in SQL?**

Below is the syntax to find 2nd highest salary in SQL:

SELECT name, MAX(salary)

FROM employees

WHERE salary < (SELECT MAX(salary)

FROM employees);

**How to delete duplicate rows in SQL?**

There are multiple ways to delete duplicate records in SQL.

Below is the code to delete duplicate records using rank:

alter table emp add sid int identity(1,1)

delete e

from emp e

inner join

(select \*,

RANK() OVER ( PARTITION BY eid,ename ORDER BY id DESC )rank

From emp )T on e.sid=t.sid

where e.Rank>1

alter table emp

drop column sno

Below is the syntax to delete duplicate records using groupby and min:

alter table emp add sno int identity(1,1)

delete E from emp E

left join

(select min(sno) sno From emp group by empid,ename ) T on E.sno=T.sno

where T.sno is null

alter table emp

drop column sno

**What is cursor in SQL?**

Cursors in SQL are used to store database tables. There are two types of cursors:

* Implicit Cursor
* Explicit Cursor

**Implicit Cursor:**

These implicit cursors are default cursors which are automatically created. A user cannot create an implicit cursor.

**Explicit Cursor:**

Explicit cursors are user-defined cursors. This is the syntax to create explicit cursor:

DECLARE cursor\_name CURSOR FOR SELECT \* FROM table\_name

We start off by giving by keyword DECLARE, then we give the name of the cursor, after that we give the keywords CURSOR FOR SELECT \* FROM, finally, we give in the name of the table.

**How to create a stored procedure using SQL Server?**

If you have worked with other languages, then you would know about the concept of Functions. You can consider stored procedures in SQL to be analogous to functions in other languages. This means that we can store a SQL statement as a stored procedure and this stored procedure can be invoked whenever we want.

This is the syntax to create a stored procedure:

CREATE PROCEDURE procedure\_name

AS

sql\_statement

GO;

We start off by giving the keywords CREATE PROCEDURE, then we go ahead and give the name of this stored procedure. After that, we give the AS keyword and follow it up with the SQL query, which we want as a stored procedure. Finally, we give the GO keyword.

Once, we create the stored procedure, we can invoke it this way:

EXEC procedure\_name;

We will give in the keyword EXEC and then give the name of the stored procedure.

Let’s look at an example of a stored procedure:

CREATE PROCEDURE employee\_location @location nvarchar(20)

AS

SELECT \* FROM employees WHERE location = @location

GO;

In the above command, we are creating a stored procedure which will help us to extract all the employees who belong to a particular location.

EXEC employee\_location @location = 'Boston';

With this, we are extracting all the employees who belong to Boston.

**How to create an index in SQL?**

We can create an index using this command:

CREATE INDEX index\_name

ON table\_name (column1, column2, column3 ...);

We start off by giving the keywords CREATE INDEX and then we will follow it up with the name of the index, after that we will give the ON keyword. Then, we will give the name of the table on which we would want to create this index. Finally, in parenthesis, we will list out all the columns which will have the index. Let’s look at an example:

CREATE INDEX salary

ON Employees (Salary);

In the above example, we are creating an index called a salary on top of the ‘Salary’ column of the ‘Employees’ table.

Now, let’s see how can we create a unique index:

CREATE UNIQUE INDEX index\_name

ON table\_name (column1, column2,column3 ...);

We start off with the keywords CREATE UNIQUE INDEX, then give in the name of the index, after that, we will give the ON keyword and follow it up with the name of the table. Finally, in parenthesis, we will give the list of the columns which on which we would want this unique index.

**How to change the column data type in SQL?**

We can change the data type of the column using the alter table. This will be the command:

ALTER TABLE table\_name

MODIFY COLUMN column\_name datatype;

We start off by giving the keywords ALTER TABLE, then we will give in the name of the table. After that, we will give in the keywords MODIFY COLUMN. Going ahead, we will give in the name of the column for which we would want to change the datatype and finally we will give in the data type to which we would want to change.

[**How to Rename Column Name in SQL?**](https://www.mygreatlearning.com/blog/how-to-rename-column-name-in-sql/)

**Difference between SQL and NoSQL databases?**

SQL stands for structured query language and is majorly used to query data from relational databases. When we talk about a SQL database, it will be a relational database.

But when it comes to the NoSQL databases, we will be working with non-relational databases.

Want to learn more about NoSQL databases? Check out the [NoSQL course](https://www.mygreatlearning.com/academy/learn-for-free/courses/introduction-to-nosql).

**SQL Joins Interview Questions**

**How to change column name in SQL?**

The command to change the name of a column is different in different RDBMS.

This is the command to change the name of a column in MYSQL:

*ALTER TABLE Customer CHANGE Address Addr char(50);*

IN MYSQL, we will start off by using the ALTER TABLE keywords, then we will give in the name of the table. After that, we will use the CHANGE keyword and give in the original name of the column, following which we will give the name to which we would want to rename our column.

This is the command to change the name of a column in ORACLE:

*ALTER TABLE Customer RENAME COLUMN Address TO Addr;*

In ORACLE, we will start off by using the ALTER TABLE keywords, then we will give in the name of the table. After that, we will use the RENAME COLUMN keywords and give in the original name of the column, following which we will give the TO keyword and finally give the name to which we would like to rename our column.

When it comes to SQL Server, it is not possible to rename the column with the help of ALTER TABLE command, we would have to use sp\_rename.

**What is a view in SQL?**

A view is a database object that is created using a Select Query with complex logic, so views are said to be a logical representation of the physical data, i.e Views behave like a physical table and users can use them as database objects in any part of SQL queries.

Let’s look at the types of Views:

* Simple View
* Complex View
* Inline View
* Materialized View

**Simple View:**

Simple views are created with a select query written using a single table. Below is the command to create a simple view:

Create VIEW Simple\_view as Select \* from BANK\_CUSTOMER ;

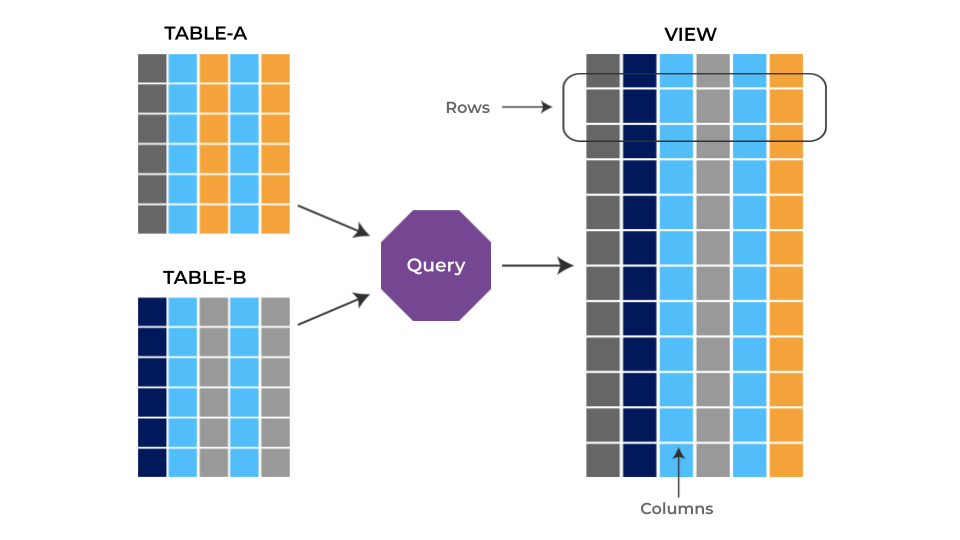
**Complex View:**

Create VIEW Complex\_view as SELECT bc.customer\_id , ba.bank\_account From Bank\_customer bc JOIN Bank\_Account ba Where bc.customer\_id = ba.customer\_id And ba.balance > 300000

**Inline View:**

A subquery is also called an inline view if and only if it is called in FROM clause of a SELECT query.

SELECT \* FROM ( SELECT bc.customer\_id , ba.bank\_account From Bank\_customer bc JOIN Bank\_Account ba Where bc.customer\_id = ba.customer\_id And ba.balance > 300000)



**How to drop a column in SQL?**

To drop a column in SQL, we will be using this command:

ALTER TABLE employees

DROP COLUMN gender;

We will start off by giving the keywords ALTER TABLE, then we will give the name of the table, following which we will give the keywords DROP COLUMN and finally give the name of the column which we would want to remove.

**How to use BETWEEN in SQL?**

The BETWEEN operator checks an attribute value within a range. Here is an example of BETWEEN operator:

SELECT \* FROM employees WHERE salary between 10000 and 20000;

With this command, we will be able to extract all the records where the salary of the employee is between 10000 and 20000.

**Advanced SQL Interview Questions**

**What are the subsets of SQL?**

* DDL (Data Definition Language): Used to define the data structure it consists of the commands like CREATE, ALTER, DROP, etc.
* DML (Data Manipulation Language): Used to manipulate already existing data in the database, commands like SELECT, UPDATE, INSERT
* DCL (Data Control Language): Used to control access to data in the database, commands like GRANT, REVOKE.

**Difference between CHAR and VARCHAR2 datatype in SQL?**

CHAR is used to store fixed-length character strings, and VARCHAR2 is used to store variable-length character strings.

**How to sort a column using a column alias?**

By using the column alias in the ORDER BY instead of where clause for sorting

**Difference between COALESCE() & ISNULL() ?**

COALESCE() accepts two or more parameters, one can apply 2 or as many parameters but it returns only the first non NULL parameter.

ISNULL() accepts only 2 parameters.

The first parameter is checked for a NULL value, if it is NULL then the 2nd parameter is returned, otherwise, it returns the first parameter.

**What is “Trigger” in SQL?**

A trigger allows you to execute a batch of SQL code when an insert,update or delete command is run against a specific table as Trigger is said to be the set of actions that are performed whenever commands like insert, update or delete are given.

**Write a Query to display employee details along with age.**

SELECT \* DATEDIFF(yy, dob, getdate()) AS 'Age' FROM employee

**Write a Query to display employee details along with age?**

SELECT SUM(salary) FROM employee

**Write an SQL query to get the third maximum salary of an employee from a table named employee\_table.**

SELECT TOP 1 salary FROM ( SELECT TOP 3 salary FROM employee\_table ORDER BY salary DESC ) AS emp ORDER BY salary ASC;

**What are aggregate and scalar functions?**

Aggregate functions are used to evaluate mathematical calculations and return single values. This can be calculated from the columns in a table. Scalar functions return a single value based on input value.

Example -. Aggregate – max(), count – Calculated with respect to numeric. Scalar – UCASE(), NOW() – Calculated with respect to strings.

**What is a deadlock?**

It is an unwanted situation where two or more transactions are waiting indefinitely for one another to release the locks.

**Explain left outer join with example.**

Left outer join is useful if you want all the records from the left table(first table) and only matching records from 2nd table. The unmatched records are null records. Example: Left outer join with “+” operator Select t1.col1,t2.col2….t ‘n’col ‘n.’. from table1 t1,table2 t2 where t1.col=t2.col(+);

**What is SQL injection?**

SQL injection is a code injection technique used to hack data-driven applications.

**What is a UNION operator?**

The UNION operator combines the results of two or more Select statements by removing duplicate rows. The columns and the data types must be the same in the SELECT statements.

**Explain SQL Constraints.**

[SQL Constraints](https://www.mygreatlearning.com/blog/sql-constraints/) are used to specify the rules of data type in a table. They can be specified while creating and altering the table. The following are the constraints in SQL: NOT NULL CHECK DEFAULT UNIQUE PRIMARY KEY FOREIGN KEY

**What is the ALIAS command?**

This command provides another name to a table or a column. It can be used in the WHERE clause of a SQL query using the “as” keyword.

**What are Group Functions? Why do we need them?**

Group functions work on a set of rows and return a single result per group. The popularly used group functions are AVG, MAX, MIN, SUM, VARIANCE, and COUNT.

**How can dynamic SQL be executed?**

* By executing the query with parameters
* By using EXEC
* By using sp\_executesql

**What is the usage of NVL() function?**

This function is used to convert the NULL value to the other value.

**Write a Query to display employee details belongs to ECE department?**

SELECT EmpNo, EmpName, Salary FROM employee WHERE deptNo in (select deptNo from dept where deptName = ‘ECE’)

**What are the main differences between #temp tables and @table variables and which one is preferred?**

1. SQL server can create column statistics on #temp tables.

2. Indexes can be created on #temp tables

3. @table variables are stored in memory up to a certain threshold

**What is CLAUSE?**

SQL clause is defined to limit the result set by providing conditions to the query. This usually filters some rows from the whole set of records. Example – Query that has WHERE condition.

**What is a recursive stored procedure?**

A stored procedure calls by itself until it reaches some boundary condition. This recursive function or procedure helps programmers to use the same set of code any number of times.

**What does the BCP command do?**

The Bulk Copy is a utility or a tool that exports/imports data from a table into a file and vice versa.

**What is a Cross Join?**

In SQL cross join, a combination of every row from the two tables is included in the result set. This is also called cross product set. For example, if table A has ten rows and table B has 20 rows, the result set will have 10 \* 20 = 200 rows provided there is a NOWHERE clause in the SQL statement.

**Which operator is used in query for pattern matching?**

LIKE operator is used for pattern matching, and it can be used as- 1. % – Matches zero or more characters. 2. \_(Underscore) – Matching exactly one character.

**Write a SQL query to get the current date?**

SELECT CURDATE();

**State the case manipulation functions in SQL?**

* LOWER: converts all the characters to lowercase.
* UPPER: converts all the characters to uppercase.
* INITCAP: converts the initial character of each word to uppercase

**How to add a column to an existing table?**

ALTER TABLE Department ADD (Gender, M, F)

**Define lock escalation?**

A query first takes the lowest level lock possible with the smallest row level. When too many rows are locked, the lock is escalated to a range or page lock. If too many pages are locked, it may escalate to a table lock.

**How to store Videos inside SQL Server table?**

By using FILESTREAM datatype, which was introduced in SQL Server 2008.

**State the order of SQL SELECT?**

The order of SQL SELECT clauses is: SELECT, FROM, WHERE, GROUP BY, HAVING, ORDER BY. Only the SELECT and FROM clauses are mandatory.

**What is the difference between IN and EXISTS?**

**IN:** Works on List result set Doesn’t work on subqueries resulting in Virtual tables with multiple columns Compares every value in the result list.

**Exists:** Works on Virtual tables Is used with co-related queries Exits comparison when the match is found

**How do you copy data from one table to another table?**

INSERT INTO table2 (column1, column2, column3, …) SELECT column1, column2, column3, … FROM table1 WHERE condition;

**List the ACID properties that make sure that the database transactions are processed**

ACID (Atomicity, Consistency, Isolation, Durability) is a set of properties that guarantee that database transactions are processed reliably.

**What will be the output of the following Query, provided the employee table has 10 records?**

BEGIN TRAN TRUNCATE TABLE Employees ROLLBACK SELECT \* FROM Employees

This query will return 10 records as TRUNCATE was executed in the transaction. TRUNCATE does not itself keep a log but BEGIN TRANSACTION keeps track of the TRUNCATE command.

**What do you mean by Stored Procedures? How do we use it?**

A stored procedure is a collection of SQL statements that can be used as a function to access the database. We can create these stored procedures earlier before using it and can execute them wherever required by applying some conditional logic to it. Stored procedures are also used to reduce network traffic and improve performance.

**What does GRANT command do?**

This command is used to provide database access to users other than the administrator in SQL privileges.

**What does the First normal form do?**

First Normal Form (1NF): It removes all duplicate columns from the table. It creates a table for related data and identifies unique column values.

**How to add e record to the table?**

INSERT syntax is used to add a record to the table. INSERT into table\_name VALUES (value1, value2..);

**What are the different tables present in MySQL?**

There are 5 tables present in MYSQL.

* MyISAM
* Heap
* Merge
* INNO DB
* ISAM

**What is BLOB and TEXT in MySQL?**

BLOB stands for the large binary objects. It is used to hold a variable amount of data. TEXT is a case-insensitive BLOB. TEXT values are non-binary strings (character strings).

**What is the use of mysql\_close()?**

Mysql\_close() cannot be used to close the persistent connection. Though it can be used to close a connection opened by mysql\_connect().

**Write a query to find out the data between ranges?**

In day-to-day activities, the user needs to find out the data between some range. To achieve this user needs to use Between..and operator or Greater than and less than the operator.

Query 1: Using Between..and operator

Select \* from Employee where salary between 25000 and 50000;

Query 2: Using operators (Greater than and less than)

Select \* from Employee where salary >= 25000 and salary <= 50000;

**How to calculate the number of rows in a table without using the count function?**

There are so many system tables which are very important. Using the system table user can count the number of rows in the table. following query is helpful in that case, Select table\_name, num\_rows from user\_tables where table\_name=’Employee’;

**What is wrong with the following query? SELECT empName FROM employee WHERE salary <> 6000**

The following query will not fetch a record with the salary of 6000 but also will skip the record with NULL.

**Will the following statements execute? if yes what will be output? SELECT NULL+1 SELECT NULL+’1′**

Yes, no error. The output will be NULL. Performing any operation on NULL will get the NULL result.

**SQL Server Interview Questions**

**What is an SQL server?**

SQL server has stayed on top as one of the most popular database management products ever since its first release in 1989 by Microsoft Corporation. The product is used across industries to store and process large volumes of data. It was primarily built to store and process data that is built on a relational model of data.

SQL Server is widely used for data analysis and also scaling up of data. SQL Server can be used in conjunction with Big Data tools such as [Hadoop](https://www.mygreatlearning.com/academy/learn-for-free/courses/introduction-to-hadoop/?gl_blog_id=15225).

SQL Server can be used to process data from various data sources such as Excel, Table, .Net Framework application, etc.

**How to install SQL Server?**

* Click on the below SQL Server official release link to access the latest version: https://www.microsoft.com/en-in/sql-server/sql-server-downloads
* Select the type of SQL Server edition that you want to install. SQL Server can be used on a Cloud Platform or as an open-source edition(Express or Developer) in your local computer system.
* Click on the Download Now button.
* Save the .exe file on your system. Right-click on the .exe file and click on Open.
* Click on ‘Yes’ to allow the changes to be made on your system and have SQL Server Installed.
* Once the installation is complete, restart your system, if required, and launch the SQL Server Management Studio application from the START menu.

**How to create a stored procedure in SQL Server?**

A Stored Procedure is nothing but a frequently used SQL query. Queries such as a SELECT query, which would often be used to retrieve a set of information many times within a database, can be saved as a Stored Procedure. The Stored Procedure, when called, executes the SQL query saved within the Stored Procedure.

Syntax to create a Stored Proc:

|  |  |
| --- | --- |
| 1  2  3  4 | **CREATE** **PROCEDURE** PROCEDURE\_NAME  **AS**  SQL\_QUERY (GIVE YOUR OFTEN USED QUERY HERE)  GO; |

Stored procedures can be user-defined or built-in. Various parameters can be passed onto a Stored Procedure.

**How to install SQL Server 2008?**

* Click on the below SQL Server official release link: https://www.microsoft.com/en-in/sql-server/sql-server-downloads
* Click on the search icon and type in – SQL Server 2008 download
* Click on the result link to download and save SQL Server 2008.
* Select the type of SQL Server edition that you want to install. SQL Server can be used on a Cloud Platform or as an open-source edition(Express or Developer) in your local computer system.
* Click on the Download Now button.
* Save the .exe file on your system. Right-click on the .exe file and click on Open.
* Click on ‘Yes’ to allow the changes to be made on your system and have SQL Server installed.
* Once the installation is complete, restart your system, if required, and launch the SQL Server Management Studio application.

**How to install SQL Server 2017?**

* Click on the below SQL Server official release link: https://www.microsoft.com/en-in/sql-server/sql-server-downloads
* Click on the search icon and type in – SQL Server 2017 download
* Click on the result link to download and save SQL Server 2017.
* Select the type of SQL Server edition that you want to install. SQL Server can be used on a Cloud Platform or as an open-source edition(Express or Developer) in your local computer system.
* Click on the Download Now button.
* Save the .exe file on your system. Right-click on the .exe file and click on Open.
* Click on ‘Yes’ to allow the changes to be made on your system and have SQL Server installed.
* Once the installation is complete, restart your system, if required, and launch the SQL Server Management Studio application from the START menu.

**How to restore the database in SQL Server?**

Launch the SQL Server Management Studio application and from the Object Explorer window pane, right-click on Databases and click on Restore. This would automatically restore the database.

**How to install SQL Server 2014?**

* Click on the below SQL Server official release link: https://www.microsoft.com/en-in/sql-server/sql-server-downloads
* Click on the search icon and type in – SQL Server 2014 download
* Click on the result link to download and save SQL Server 2014.
* Select the type of SQL Server edition that you want to install. SQL Server can be used on a Cloud Platform or as an open-source edition(Express or Developer) in your local computer system.
* Click on the Download Now button.
* Save the .exe file on your system. Right-click on the .exe file and click on Open.
* Click on ‘Yes’ to allow the changes to be made on your system and have SQL Server Installed.
* Once the installation is complete, restart your system, if required, and launch the SQL Server Management Studio application from the START menu.

**How to get the connection string from SQL Server?**

Launch the SQL Server Management Studio. Go to the Database for which you require the Connection string. Right-click on the database and click on Properties. In the Properties window that is displayed, you can view the Connection String property.

Connection strings help connect databases to another staging database or any external source of data.

**How to install SQL Server 2012?**

* Click on the below SQL Server official release link: https://www.microsoft.com/en-in/sql-server/sql-server-downloads
* Click on the search icon and type in – SQL Server 2012 download
* Click on the result link to download and save SQL Server 2012.
* Select the type of SQL Server edition that you want to install. SQL Server can be used on a Cloud Platform or as an open-source edition(Express or Developer) in your local computer system.
* Click on the Download Now button.
* Save the .exe file on your system. Right-click on the .exe file and click on Open.
* Click on ‘Yes’ to allow the changes to be made on your system and have SQL Server Installed.
* Once the installation is complete, restart your system, if required, and launch the SQL Server Management Studio application from the START menu.

**What is cte in SQL Server?**

CTEs are Common Table Expressions that are used to create temporary result tables from which data can be retrieved/ used. The standard syntax for a CTE with a SELECT statement is:

|  |  |
| --- | --- |
| 1  2  3  4  5 | WITH RESULT AS  (SELECT COL1, COL2, COL3  FROM EMPLOYEE)  SELECT COL1, COL2 FROM RESULT  CTEs can be used with Insert, Update or Delete statements as well. |

**Few examples of CTEs are given below:**

Query to find the 10 highest salaries.

with result as

(select distinct salary, dense\_rank() over (order by salary desc) as salary rank from employees)

select result. salary from result where the result.salaryrank = 10

Query to find the 2nd highest salary

with the result as

(select distinct salary, dense\_rank() over (order by salary desc) as salaryrank from employees)

select result. salary from result where the result.salaryrank = 2

In this way, CTEs can be used to find the nth highest salary within an organisation.

**How to change the SQL Server password?**

Launch your SQL Server Management Studio. Click on the Database connection for which you want to change the login password. Click on Security from the options that get displayed.

Click on Logins and open your database connection. Type in the new password for login and click on ‘OK’ to apply the changes.

**How to delete duplicate records in SQL Server?**

Select the duplicate records in a table HAVING COUNT(\*)>1

Add a delete statement to delete the duplicate records.

**Sample Query to find the duplicate records in a table-**

(SELECT COL1, COUNT(\*) AS DUPLICATE

FROM EMPLOYEE

GROUP BY COL1

HAVING COUNT(\*) > 1)

**How to uninstall SQL Server?**

In Windows 10, go to the START menu and locate the SQL Server.

Right-click and select uninstall to uninstall the application.

**How to check SQL Server version?**

You can run the below query to view the current version of SQL Server that you are using.

|  |  |
| --- | --- |
| 1 | SELECT @@version; |

**How to rename column name in SQL Server?**

From the Object Explorer window pane, go to the table where the column is present and choose Design. Under the Column Name, select the name you want to rename and enter the new name. Go to the File menu and click Save.

**What is the stored procedure in SQL Server?**

A Stored Procedure is nothing but a frequently used SQL query. Queries such as a SELECT query, which would often be used to retrieve a set of information many times within a database, can be saved as a Stored Procedure. The Stored Procedure, when called, executes the SQL query saved within the Stored Procedure.

Syntax to create a Stored Proc:

|  |  |
| --- | --- |
| 1  2  3  4 | **CREATE** **PROCEDURE** PROCEDURE\_NAME  **AS**  SQL\_QUERY (GIVE YOUR OFTEN USED QUERY HERE)  GO; |

You can execute the Stored Proc by using the command***Exec Procedure\_Name;***

**How to create a database in SQL Server?**

After installing the required version of SQL Server, it is easy to create new databases and maintain them.

1. Launch the SQL Server Management Studio
2. In the Object Explorer window pane, right-click on Databases and select ‘New Database’
3. Enter the Database Name and click on ‘Ok’.
4. Voila! Your new database is ready for use.

**What is an index in SQL Server?**

Indexes are database objects which help in retrieving records quickly and more efficiently. Column indexes can be created on both Tables and Views. By declaring a Column as an index within a table/ view, the user can access those records quickly by executing the index. Indexes with more than one column are called Clustered indexes.

Syntax:

|  |  |
| --- | --- |
| 1  2 | CREATE INDEX INDEX\_NAME  ON TABLE\_NAME(COL1, COL2); |

The syntax to drop an Index is DROP INDEX INDEX\_NAME;

Indexes are known to improve the efficiency of SQL Select queries.

**How to create the table in SQL Server?**

Tables are the fundamental storage objects within a database. A table is usually made up of

Rows and Columns. The below syntax can be used to create a new table with 3 columns.

CREATE TABLE *TABLE\_NAME(*

*COLUMN1 DATATYPE,*

*COLUMN2 DATATYPE,*

*COLUMN3 DATATYPE*

*);*

Alternatively, you can right-click on Table in the Object Explorer window pane and select ‘New -> Table’.

You can also define the type of Primary/ Foreign/ Check constraint when creating a table.

**How to connect to SQL Server?**

* Launch the SQL Server Management Studio from the START menu.
* In the dialogue box shown below, select the Server Type as Database Engine and Server Name as the name of your laptop/ desktop system.
* Select the appropriate Authentication type and click on the *Connect*button.
* A secure connection would be established, and the list of the available Databases will be loaded in the Object Explorer window pane.

**How to delete duplicate rows in SQL Server?**

Select the duplicate records in a table HAVING COUNT(\*)>1

Add a delete statement to delete the duplicate records.

**Sample Query to find the duplicate records in a table –**

(SELECT COL1, COUNT(\*) AS DUPLICATE

FROM EMPLOYEE

GROUP BY COL1

HAVING COUNT(\*) > 1);

**How to download SQL Server?**

The Express and Developer versions (open-source versions) of the latest SQL Server release can be downloaded from the official Microsoft website. The link is given below for reference.  
https://www.microsoft.com/en-in/sql-server/sql-server-downloads

**How to connect SQL Server management studio to the local database?**

* Launch the SQL Server Management Studio from the START menu.
* In the dialogue box shown below, select the Server Type as Database Engine and Server Name as the name of your laptop/ desktop system and click on the *Connect*button.
* Select the Authentication as ‘Windows Authentication.
* A secure connection would be established, and the list of the available Databases will be loaded in the Object Explorer window pane.

**How to download SQL Server 2014?**

* Both the Express and Developer versions (free editions) of SQL Server can be downloaded from the official Microsoft website. The link is given below for reference.
* Click on the link below: https://www.microsoft.com/en-in/sql-server/sql-server-downloads
* Click on the search icon and type in – SQL Server 2014 download
* Click on the result link to download and save SQL Server 2014.

**How to uninstall SQL Server 2014?**

From the START menu, type SQL Server. Right-click on the app and select uninstall to uninstall the application from your system. Restart the system, if required, for the changes to get affected.

**How to find server names in SQL Server?**

Run the query SELECT @@version; to find the version and name of the SQL Server you are using.

**How to start SQL Server?**

Launch the SQL Server Management Studio from the START menu. Login using Windows Authentication. In the Object Explorer window pane, you can view the list of databases and corresponding objects.

**What is the case when in SQL Server?**

Case When statements in SQL are used to run through many conditions and to return a value when one such condition is met. If none of the conditions is met in the When statements, then the value mentioned in the Else statement is returned.

Syntax:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | CASE  **WHEN** CONDITION1 **THEN** RESULT1    **WHEN** CONDITION2 **THEN** RESULT2    **ELSE**  RESULT  **END**; |

Sample query:

HOW MANY HEAD OFFICES/ BRANCHES ARE THERE IN CANADA

select   
sum (   
case   
when region\_id >=  5 AND region\_id <= 7 then    
1  
else   
0  
end ) as Canada  
from company\_regions;  
Nested CASE statement:  
SELECT  
SUM (  
CASE  
WHEN rental\_rate = 0.99 THEN  
1  
ELSE  
0  
END  
) AS "Mass",  
SUM (  
CASE  
WHEN rental\_rate = 2.99 THEN  
1  
ELSE  
0  
END  
) AS "Economic",  
SUM (  
CASE  
WHEN rental\_rate = 4.99 THEN  
1  
ELSE  
0  
END  
) AS " Luxury"  
FROM  
film;

**How to install SQL Server management studio?**

Launch Google and in the Search toolbar, type in SQL Server Management Studio download.

Go to the routed website and click on the link to download. Once the download is complete, open the .exe file to install the content of the file. Once the installation is complete, refresh or restart the system, as required.

Alternatively, once SQL Server is installed and launched, it will prompt the user with an option to launch SQ Server Management Studio.

**How to write a stored procedure in SQL Server?**

A Stored Procedure is nothing but a frequently used SQL query. Queries such as a SELECT query, which would often be used to retrieve a set of information many times within a database, can be saved as a Stored Procedure. The Stored Procedure, when called, executes the SQL query saved within the Stored Procedure.

Syntax to create a Stored Proc:

|  |  |
| --- | --- |
| 1  2  3  4 | **CREATE** **PROCEDURE** PROCEDURE\_NAME  **AS**  SQL\_QUERY (GIVE YOUR OFTEN USED QUERY HERE)  GO; |

You can execute the Stored Proc by using the command***Exec Procedure\_Name;***

**How to open SQL Server?**

Launch the SQL Server Management Studio from the START menu. Login using Windows Authentication. In the Object Explorer window pane, you can view the list of databases and corresponding objects.

**How to use SQL Server?**

SQL Server is used to retrieve and process various data that is built on a relational model.

Some of the common actions that can be taken on the data are CREATE, DELETE, INSERT, UPDATE, SELECT, REVOKE, etc.

SQL Server can also be used to import and export data from different data sources. SQL Server can also be connected to various other databases/ .Net frameworks using Connection Strings.

SQL Server can also be used in conjunction with Big Data tools like Hadoop.

**What is a function in SQL Server?**

Functions are pre-written codes that return a value and which help the user achieve a particular task concerning viewing, manipulating, and processing data.

Examples of a few functions are:

AGGREGATE FUNCTIONS:

* MIN()- Returns the minimum value
* MAX()- Returns the maximum value
* AVG()- Returns the average value
* COUNT()

**STRING FUNCTIONS:**

* COALESCE()
* CAST()
* CONCAT()
* SUBSTRING()

**DATE FUNCTIONS:**

* GETDATE()
* DATEADD()
* DATEDIFF()

There are many types of functions such as Aggregate Functions, Date Functions, String Functions, Mathematical functions, etc.

**How to find nth highest salary in SQL Server without using a subquery**

Query to find the 10 highest salaries. For up-gradation of the b10 band.

*with result as*

(select distinct salary, dense\_rank() over (order by salary desc) as salaryrank from employees)

select result.salary from result where result.salaryrank = 10

Query to find the 2nd highest salary

with the result as

(select distinct salary, dense\_rank() over (order by salary desc) as salary rank from employees)

select result.salary from result where result.salaryrank = 2

In this way, by replacing the salary rank value, we can find the nth highest salary in any organisation.

**How to install SQL Server in Windows 10?**

Click on the below SQL Server official release link: https://www.microsoft.com/en-in/sql-server/sql-server-downloads

Click on the search icon and type in - SQL Server 2012 download

Click on the result link to download and save SQL Server 2012.

Select the type of the SQL Server edition that you want to install. SQL Server can be used on a Cloud Platform or as an open-source edition(Express or Developer) in your local computer system.

Click on the Download Now button.

Save the .exe file on your system. Right-click on the .exe file and click on Open.

Click on ‘Yes’ to allow the changes to be made on your system and have SQL Server Installed

**How to create a temp table in SQL Server?**

Temporary tables can be used to retain the structure and a subset of data from the original table from which they were derived.

**Syntax:**

|  |  |
| --- | --- |
| 1  2  3 | **SELECT** COL1, COL2  **INTO** TEMPTABLE1  **FROM** ORIGTABLE; |

Temporary tables do not occupy any physical memory and can be used to retrieve data faster.

**PostgreSQL Interview Questions**

**What is PostgreSQL?**

PostgreSQL is one of the most widely and popularly used languages for Object-Relational Database Management systems. It is mainly used for large web applications. It is an open-source, object-oriented, -relational database system. It is extremely powerful and enables users to extend any system without problem. It extends and uses the SQL language in combination with various features for safely scaling and storage of intricate data workloads.

**List different datatypes of PostgreSQL?**

Listed below are some of the new data types in PostgreSQL

* UUID
* Numeric types
* Boolean
* Character types
* Temporal types
* Geometric primitives
* Arbitrary precision numeric
* XML
* Arrays etc

**What are the Indices of PostgreSQL?**

Indices in PostgreSQL allow the database server to find and retrieve specific rows in a given structure. Examples are B-tree, hash, GiST, SP-GiST, GIN and BRIN.  Users can also define their indices in PostgreSQL. However, indices add overhead to the data manipulation operations and are seldom used

**What are tokens in PostgreSQL?**

Tokens in PostgreSQL act as the building blocks of a source code. They are composed of various special character symbols. Commands are composed of a series of tokens and terminated by a semicolon(“;”). These can be a constant, quoted identifier, other identifiers, keyword or a constant. Tokens are usually separated by whitespaces.

**How to create a database in PostgreSQL?**

Databases can be created using 2 methods

* First is the CREATE DATABASE SQL Command

We can create the database by using the syntax:-

|  |  |
| --- | --- |
| 1 | CREATE DATABASE <dbname>; |

* The second is by using the createdb command

We can create the database by using the syntax:-

|  |  |
| --- | --- |
| 1 | createdb [option...] <dbname> [description] |

Various options can be taken by the createDB command based on the use case.

**How to create a table in PostgreSQL?**

You can create a new table by specifying the table name, along with all column names and their types:

CREATE TABLE [IF NOT EXISTS] table\_name (

column1 datatype(length) column\_contraint,

column2 datatype(length) column\_contraint,

.

.

.

columnn datatype(length) column\_contraint,

table\_constraints

);

**How can we change the column datatype in PostgreSQL?**

The column the data type can be changed in PostgreSQL by using the ALTER TABLE command:

ALTER TABLE table\_name

ALTER COLUMN column\_name1 [SET DATA] TYPE new\_data\_type,

ALTER COLUMN column\_name2 [SET DATA] TYPE new\_data\_type,

...;

**Compare ‘PostgreSQL’ with ‘MongoDB’**

|  |  |
| --- | --- |
| **PostgreSQL** | **MongoDB** |
| PostgreSQL is an SQL database where data is stored as tables, with structured rows and columns. It supports concepts like referential integrity entity-relationship and JOINS. PostgreSQL uses SQL as its querying language. PostgreSQL supports vertical scaling. This means that you need to use big servers to store data. This leads to a requirement of downtime to upgrade. It works better if you require relational databases in your application or need to run complex queries that test the limit of SQL. | MongoDB, on the other hand, is a NoSQL database. There is no requirement for a schema, therefore it can store unstructured data. Data is stored as BSON documents and the document’s structure can be changed by the user. MongoDB uses JavaScript for querying. It supports horizontal scaling, as a result of which additional servers can be added as per the requirement with minimal to no downtime.  It is appropriate in a use case that requires a highly scalable distributed database that stores unstructured data |

**What is Multi-Version concurrency control in PostgreSQL?**

MVCC or better known as Multi-version concurrency control is used to implement transactions in PostgreSQL. It is used to avoid unwanted locking of a database in the system. while querying a database each transaction sees a version of the database. This avoids viewing inconsistencies in the data, and also provides transaction isolation for every database session. MVCC locks for reading data do not conflict with locks acquired for

**How do you delete the database in PostgreSQL?**

Databases can be deleted in PostgreSQL using the syntax

|  |  |
| --- | --- |
| 1 | DROP DATABASE [IF EXISTS] <database\_name>; |

Please note that only databases having no active connections can be dropped.

**What does a schema contain?**

* Schemas are a part of the database that contains tables. They also contain other kinds of named objects, like data types, functions, and operators.
* The object names can be used in different schemas without conflict; Unlike databases, schemas are separated more flexibly. This means that a user can access objects in any of the schemas in the database they are connected to, till they have privileges to do so.
* Schemas are highly beneficial when there is a need to allow many users access to one database without interfering with each other. It helps in organizing database objects into logical groups for better manageability. Third-party applications can be put into separate schemas to avoid conflicts based on names.

**What is the square root operator in PostgreSQL?**

It is denoted by ‘|/” and returns the square root of a number. Its syntax is

|  |  |
| --- | --- |
| 1 | Select |/ <number> |

Egs:- Select |/16

**How are the stats updated in Postgresql?**

To update statistics in PostgreSQL a special function called an explicit ‘vacuum’ call is made. Entries in pg\_statistic are updated by the ANALYZE and VACUUM ANALYZE commands

**What Is A Candid?**

The CTIDs field exists in every PostgreSQL table. It is unique for every record of a table and exactly shows the location of a tuple in a particular table. A logical row’s CTID changes when it is updated, thus it cannot be used as a permanent row identifier. However, it is useful when identifying a row within a transaction when no update is expected on the data item.

**What is Cube Root Operator (||/) in PostgreSQL?**

It is denoted by ‘|/” and returns the square root of a number. Its syntax is

|  |  |
| --- | --- |
| 1 | Select |/ <number> |

Egs:- Select |/16

**Explain Write-Ahead Logging?**

Write-ahead logging is a method to ensure data integrity. It is a protocol that ensures writing the actions as well as changes into a transaction log. It is known to increase the reliability of databases by logging changes before they are applied or updated onto the database. This provides a backup log for the database in case of a crash.

**What is a non-clustered index?**

A non-clustered index in PostgreSQL is a simple index, used for fast retrieval of data, with no certainty of the uniqueness of data. It also contains pointers to locations where other parts of data are stored

**How is security ensured in PostgreSQL?**

PostgreSQL uses 2 levels of security

* Network-level security uses Unix Domain sockets, TCP/IP sockets, and firewalls.
* Transport-level security which uses SSL/TLS to enable secure communication with the database
* Database-level security with features like roles and permissions, row-level security (RLS), and auditing.

**SQL Practice Questions**

**PART 1**

This covers SQL basic query operations like creating databases forms scratch, creating a table, inserting values etc.

It is better to get hands-on in order to have practical experience with SQL queries. A small error/bug will make you feel surprised and next time you will get there!

Let’s get started!

1) Create a Database bank

|  |  |
| --- | --- |
| 1  2 | **CREATE** **DATABASE** bank;  use bank |

2) Create a table with the name “bank\_details” with the following columns

— Product  with string data type

— Quantity with numerical data type

— Price with real number data type

— purchase\_cost with decimal data type

— estimated\_sale\_price with data type float

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | **Create** **table** bank\_details(  Product **CHAR**(10) ,  quantity **INT**,  price **Real** ,  purchase\_cost **Decimal**(6,2),  estimated\_sale\_price  **Float**); |

3) Display all columns and their datatype and size in Bank\_details

|  |  |
| --- | --- |
| 1 | Describe bank\_details; |

4) Insert two records into Bank\_details.

— 1st record with values —

— Product: PayCard

— Quantity: 3

— price: 330

— Puchase\_cost: 8008

— estimated\_sale\_price: 9009

— Product: PayPoints —

— Quantity: 4

— price: 200

— Puchase\_cost: 8000

— estimated\_sale\_price: 6800

|  |  |
| --- | --- |
| 1  2 | **Insert** **into** Bank\_detailsvalues ( 'paycard' , 3 , 330, 8008, 9009);  **Insert** **into** Bank\_detailsvalues ( 'paypoints' , 4 , 200, 8000, 6800); |

5) Add a column: Geo\_Location to the existing Bank\_details table with data type varchar and size 20

|  |  |
| --- | --- |
| 1 | **Alter** **table** Bank\_details **add**  geo\_location **Varchar**(20); |

6) What is the value of Geo\_location for a product : “PayCard”?

|  |  |
| --- | --- |
| 1 | **Select** geo\_location  **from** Bank\_details **where** Product = 'PayCard'; |

7) How many characters does the  Product : “paycard” have in the Bank\_details table.

|  |  |
| --- | --- |
| 1 | **select** char\_length(Product) **from** Bank\_details **where** Product = 'PayCard'; |

8) Alter the Product field from CHAR to VARCHAR in Bank\_details

|  |  |
| --- | --- |
| 1 | **Alter** **table**  bank\_details **modify** PRODUCT **varchar**(10); |

9) Reduce the size of the Product field from 10 to 6 and check if it is possible

|  |  |
| --- | --- |
| 1 | **Alter** **table** bank\_details **modify** product **varchar**(6); |

10) Create a table named as Bank\_Holidays with below fields

— a) Holiday field which displays only date

— b) Start\_time field which displays hours and minutes

— c) End\_time field which also displays hours and minutes and timezone

|  |  |
| --- | --- |
| 1  2  3  4 | **Create** **table** bank\_holidays (              Holiday  **date** ,              Start\_time datetime ,              End\_time **timestamp**); |

11) Step 1: Insert today’s date details in all fields of Bank\_Holidays

— Step 2: After step1, perform the below

— Postpone Holiday to next day by updating the Holiday field

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | -- Step1:  **Insert** **into** bank\_holidays  **values** ( **current\_date**(),  **current\_date**(),  **current\_date**() );    -- Step 2:  **Update** bank\_holidays  **set** holiday = DATE\_ADD(Holiday , INTERVAL 1 DAY); |

Update the End\_time with current European time.

|  |  |
| --- | --- |
| 1 | **Update** Bank\_Holidays **Set** End\_time = utc\_timestamp(); |

12)  Display output of PRODUCT field as NEW\_PRODUCT in  Bank\_details table

|  |  |
| --- | --- |
| 1 | **Select** PRODUCT **as** NEW\_PRODUCT **from** bank\_details; |

13)  Display only one record from bank\_details

|  |  |
| --- | --- |
| 1 | **Select** \* **from** Bank\_details limit 1; |

15) Display the first five characters of the Geo\_location field of Bank\_details.

|  |  |
| --- | --- |
| 1 | **SELECT** substr(Geo\_location  , 1, 5)  **FROM** `bank\_details`; |

**PART 2**

— ——————————————————–

# Datasets Used: cricket\_1.csv, cricket\_2.csv

— cricket\_1 is the table for cricket test match 1.

— cricket\_2 is the table for cricket test match 2.

— ——————————————————–

Find all the players who were present in the test match 1 as well as in the test match 2.

|  |  |
| --- | --- |
| 1  2  3 | **SELECT** \* **FROM** cricket\_1  **UNION**  **SELECT** \* **FROM** cricket\_2; |

Write a MySQl query to find the players from the test match 1 having popularity higher than the average popularity.

|  |  |
| --- | --- |
| 1 | **select** player\_name , Popularity **from** cricket\_1 **WHERE** Popularity > (**SELECT** AVG(Popularity) **FROM** cricket\_1); |

  Find player\_id and player name that are common in the test match 1 and test match 2.

|  |  |
| --- | --- |
| 1  2 | **SELECT** player\_id , player\_name **FROM** cricket\_1  **WHERE** cricket\_1.player\_id IN (**SELECT** player\_id **FROM** cricket\_2); |

Retrieve player\_id, runs, and player\_name from cricket\_1 and cricket\_2 table and display the player\_id of the players where the runs are more than the average runs.

|  |  |
| --- | --- |
| 1 | **SELECT** player\_id , runs , player\_name **FROM** cricket\_1 **WHERE**  cricket\_1.RUNS > (**SELECT** AVG(RUNS) **FROM** cricket\_2); |

Write a query to extract the player\_id, runs and player\_name from the table “cricket\_1” where the runs are greater than 50.

|  |  |
| --- | --- |
| 1  2 | **SELECT** player\_id , runs , player\_name **FROM** cricket\_1  **WHERE** cricket\_1.Runs > 50 ; |

Write a query to extract all the columns from cricket\_1 where player\_name starts with ‘y’ and ends with ‘v’.

|  |  |
| --- | --- |
| 1 | **SELECT** \* **FROM** cricket\_1 **WHERE** player\_name LIKE 'y%v'; |

Write a query to extract all the columns from cricket\_1 where player\_name does not end with ‘t’.

|  |  |
| --- | --- |
| 1  2 | **SELECT** \* **FROM** cricket\_1 **WHERE** player\_name NOT LIKE '%t'; |

# Dataset Used: cric\_combined.csv

Write a MySQL query to create a new column PC\_Ratio that contains the popularity to charisma ratio.

|  |  |
| --- | --- |
| 1  2  3  4 | **ALTER** **TABLE** cric\_combined  **ADD** **COLUMN** PC\_Ratio float4;    **UPDATE** cric\_combined **SET** PC\_Ratio =  (Popularity / Charisma); |

 Write a MySQL query to find the top 5 players having the highest popularity to charisma ratio

|  |  |
| --- | --- |
| 1 | **SELECT** Player\_Name , PC\_Ratio  **FROM** cric\_combined **ORDER** **BY**  PC\_Ratio **DESC** LIMIT 5; |

Write a MySQL query to find the player\_ID and the name of the player that contains the character “D” in it.

|  |  |
| --- | --- |
| 1 | **SELECT** Player\_Id ,  Player\_Name **FROM** cric\_combined **WHERE** Player\_Name LIKE '%d%'; |

Dataset Used: new\_cricket.csv

Extract the Player\_Id and Player\_name of the players where the charisma value is null.

|  |  |
| --- | --- |
| 1 | **SELECT** Player\_Id , Player\_Name **FROM** new\_cricket **WHERE** Charisma  **IS** NULL; |

Write a MySQL query to impute all the NULL values with 0.

|  |  |
| --- | --- |
| 1 | **SELECT** IFNULL(Charisma, 0) **FROM** new\_cricket; |

Separate all Player\_Id into single numeric ids (example PL1 =  1).

|  |  |
| --- | --- |
| 1  2 | **SELECT** Player\_Id, SUBSTR(Player\_Id,3)  **FROM**  new\_cricket; |

Write a MySQL query to extract Player\_Id, Player\_Name and charisma where the charisma is greater than 25.

|  |  |
| --- | --- |
| 1 | **SELECT** Player\_Id , Player\_Name , charisma **FROM** new\_cricket **WHERE** charisma > 25; |

# Dataset Used: churn1.csv

Write a query to count all the duplicate values from the column “Agreement” from the table churn1.

|  |  |
| --- | --- |
| 1 | **SELECT** Agreement, COUNT(Agreement) **FROM** churn1 **GROUP** **BY** Agreement **HAVING** COUNT(Agreement) > 1; |

Rename the table churn1 to “Churn\_Details”.

|  |  |
| --- | --- |
| 1 | RENAME **TABLE** churn1 **TO** Churn\_Details; |

Write a query to create a new column new\_Amount that contains the sum of TotalAmount and MonthlyServiceCharges.

|  |  |
| --- | --- |
| 1  2  3  4  5 | **ALTER** **TABLE** Churn\_Details  **ADD** **COLUMN** new\_Amount **FLOAT**;  **UPDATE** Churn\_Details **SET** new\_Amount = (TotalAmount + MonthlyServiceCharges);    **SELECT** new\_Amount **FROM** CHURN\_DETAILS; |

 Rename column new\_Amount to Amount.

|  |  |
| --- | --- |
| 1  2  3 | **ALTER** **TABLE** Churn\_Details CHANGE new\_Amount Amount **FLOAT**;    **SELECT** AMOUNT **FROM** CHURN\_DETAILS; |

Drop the column “Amount” from the table “Churn\_Details”.

|  |  |
| --- | --- |
| 1 | **ALTER** **TABLE** Churn\_Details **DROP** **COLUMN** Amount ; |

Write a query to extract the customerID, InternetConnection and gender from the table “Churn\_Details ” where the value of the column “InternetConnection” has ‘i’ at the second position.

|  |  |
| --- | --- |
| 1 | **SELECT** customerID, InternetConnection,  gender **FROM** Churn\_Details **WHERE** InternetConnection LIKE '\_i%'; |

Find the records where the tenure is 6x, where x is any number.

|  |  |
| --- | --- |
| 1 | **SELECT** \* **FROM** Churn\_Details **WHERE** tenure LIKE '6\_'; |

**Part 3**

# DataBase = Property Price Train

Dataset used: Property\_Price\_Train\_new

Write An MySQL Query To Print The First Three Characters Of  Exterior1st From Property\_Price\_Train\_new Table.

|  |  |
| --- | --- |
| 1 | **Select** substring(Exterior1st,1,3) **from** Property\_Price\_Train\_new; |

Write An MySQL Query To Print Brick\_Veneer\_Area Of Property\_Price\_Train\_new Excluding Brick\_Veneer\_Type, “None” And “BrkCmn” From Property\_Price\_Train\_new Table.

|  |  |
| --- | --- |
| 1 | **Select**  Brick\_Veneer\_Area, Brick\_Veneer\_Type **from** Property\_Price\_Train\_new  **where** Brick\_Veneer\_Type not in ('None','BrkCmn'); |

Write An MySQL Query to print Remodel\_Year , Exterior2nd of the Property\_Price\_Train\_new Whose Exterior2nd Contains ‘H’.

|  |  |
| --- | --- |
| 1 | **Select** Remodel\_Year , Exterior2nd **from** Property\_Price\_Train\_new **where** Exterior2nd like '%H%' ; |

Write MySQL query to print details of the table Property\_Price\_Train\_new whose Remodel\_year from 1983 to 2006

|  |  |
| --- | --- |
| 1 | **select** \* **from** Property\_Price\_Train\_new **where** Remodel\_Year between 1983 and 2006; |

Write MySQL query to print details of Property\_Price\_Train\_new whose Brick\_Veneer\_Type ends with e and contains 4 alphabets.

|  |  |
| --- | --- |
| 1 | **Select** \* **from** Property\_Price\_Train\_new **where** Brick\_Veneer\_Type like '\_\_\_\_e'; |

Write MySQl query to print nearest largest integer value of column Garage\_Area from Property\_Price\_Train\_new

|  |  |
| --- | --- |
| 1 | **Select** ceil(Garage\_Area) **from** Property\_Price\_Train\_new; |

Fetch the 3 highest value of column Brick\_Veneer\_Area from Property\_Price\_Train\_new table

|  |  |
| --- | --- |
| 1 | **Select** Brick\_Veneer\_Area **from** Property\_Price\_Train\_new **order** **by** Brick\_Veneer\_Area **desc** limit 2,1; |

Rename column LowQualFinSF to Low\_Qual\_Fin\_SF fom table Property\_Price\_Train\_new

|  |  |
| --- | --- |
| 1 | **Alter** **table** Property\_Price\_Train\_new change LowQualFinSF Low\_Qual\_Fin\_SF **varchar**(150); |

Convert Underground\_Full\_Bathroom (1 and 0) values to true or false respectively.

# Eg. 1 – true ; 0 – false

|  |  |
| --- | --- |
| 1 | **SELECT** CASE **WHEN** Underground\_Full\_Bathroom = 0 **THEN** 'false' **ELSE** 'true' **END** **FROM** Property\_Price\_Train\_new; |

Extract total Sale\_Price for each year\_sold column of Property\_Price\_Train\_new table.

|  |  |
| --- | --- |
| 1 | **Select** Year\_Sold, sum(Sale\_Price) **from** Property\_Price\_Train\_new **group** **by** Year\_Sold; |

Extract all negative values from W\_Deck\_Area

|  |  |
| --- | --- |
| 1 | **Select** W\_Deck\_Area **from** Property\_Price\_Train\_new **where** W\_Deck\_Area < 0; |

Write MySQL query to extract Year\_Sold, Sale\_Price whose price is greater than 100000.

|  |  |
| --- | --- |
| 1 | **Select** Sale\_Price , Year\_Sold **from** Property\_Price\_Train\_new **group** **by** Year\_Sold **having** Sale\_Price  >  100000; |

Write MySQL query to extract Sale\_Price and House\_Condition from Property\_Price\_Train\_new and Property\_price\_train\_2 perform inner join. Rename the table as PPTN and PPTN2.

|  |  |
| --- | --- |
| 1 | **Select** Sale\_Price , House\_Condition **from** Property\_Price\_Train\_new **AS** PPTN **inner** join Property\_price\_train\_2 **AS** PPT2 **on** PPTN.ID= PPTN2.ID; |

Count all duplicate values of column Brick\_Veneer\_Type from tbale Property\_Price\_Train\_new

|  |  |
| --- | --- |
| 1 | **Select** Brick\_Veneer\_Type, count(Brick\_Veneer\_Type) **from** Property\_Price\_Train\_new **group** **by** Brick\_Veneer\_Type **having** count(Brick\_Veneer\_Type) > 1; |

# DATABASE Cricket

Find all the players from both matches.

|  |  |
| --- | --- |
| 1  2  3 | **SELECT** \* **FROM** cricket\_1  **UNION**  **SELECT** \* **FROM** cricket\_2; |

Perform right join on cricket\_1 and cricket\_2.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | **SELECT**      cric2.Player\_Id, cric2.Player\_Name, cric2.Runs, cric2.Charisma, cric1.Popularity  **FROM**      cricket\_1 **AS** cric1          RIGHT JOIN      cricket\_2 **AS** cric2 **ON** cric1.Player\_Id = cric2.Player\_Id; |

 Perform left join on cricket\_1 and cricket\_2

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | **SELECT**   cric1.Player\_Id, cric1.Player\_Name, cric1.Runs, cric1.Popularity, cric2.Charisma  **FROM**      cricket\_1 **AS** cric1          LEFT JOIN      cricket\_2 **AS** cric2 **ON** cric1.Player\_Id = cric2.Player\_Id; |

Perform left join on cricket\_1 and cricket\_2.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | **SELECT**      cric1.Player\_Id, cric1.Player\_Name, cric1.Runs, cric1.Popularity, cric2.Charisma  **FROM**      cricket\_1 **AS** cric1  **INNER** JOIN      cricket\_2 **AS** cric2 **ON** cric1.Player\_Id = cric2.Player\_Id; |

Create a new table and insert the result obtained after performing inner join on the two tables cricket\_1 and cricket\_2.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | **CREATE** **TABLE** Players1And2 **AS**  **SELECT**      cric1.Player\_Id, cric1.Player\_Name, cric1.Runs, cric1.Popularity, cric2.Charisma  **FROM**      cricket\_1 **AS** cric1  **INNER** JOIN      cricket\_2 **AS** cric2 **ON** cric1.Player\_Id = cric2.Player\_Id; |

Write MySQL query to extract maximum runs of players get only top two players

|  |  |
| --- | --- |
| 1 | **select** Player\_Name, Runs **from** cricket\_1 **group** **by** Player\_Name **having** **max**(Runs) limit 2; |

**PART 4**

# Pre-Requisites

# Assuming Candidates are familiar with “Group by” and “Grouping functions” because these are used along with JOINS in the questionnaire.

# Create below DB objects

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  166  167  168  169  170  171  172  173 | **CREATE** **TABLE** BANK\_CUSTOMER ( customer\_id **INT** ,                  customer\_name **VARCHAR**(20),                  Address     **VARCHAR**(20),                  state\_code  **VARCHAR**(3) ,                  Telephone   **VARCHAR**(10) );  **INSERT** **INTO** BANK\_CUSTOMER **VALUES** (123001,"Oliver", "225-5, Emeryville", "CA" , "1897614500");  **INSERT** **INTO** BANK\_CUSTOMER **VALUES** (123002,"George", "194-6,New brighton","MN" , "1897617000");  **INSERT** **INTO** BANK\_CUSTOMER **VALUES** (123003,"Harry", "2909-5,walnut creek","CA" , "1897617866");  **INSERT** **INTO** BANK\_CUSTOMER **VALUES** (123004,"Jack", "229-5, Concord",      "CA" , "1897627999");  **INSERT** **INTO** BANK\_CUSTOMER **VALUES** (123005,"Jacob", "325-7, Mission Dist","SFO", "1897637000");  **INSERT** **INTO** BANK\_CUSTOMER **VALUES** (123006,"Noah", "275-9, saint-paul" ,  "MN" , "1897613200");  **INSERT** **INTO** BANK\_CUSTOMER **VALUES** (123007,"Charlie","125-1,Richfield",   "MN" , "1897617666");  **INSERT** **INTO** BANK\_CUSTOMER **VALUES** (123008,"Robin","3005-1,Heathrow",     "NY" , "1897614000");    **CREATE** **TABLE** BANK\_CUSTOMER\_EXPORT ( customer\_id **CHAR**(10),  customer\_name **CHAR**(20),  Address **CHAR**(20),  state\_code  **CHAR**(3) ,  Telephone  **CHAR**(10));    **INSERT** **INTO** BANK\_CUSTOMER\_EXPORT **VALUES** ("123001 ","Oliver", "225-5, Emeryville", "CA" , "1897614500") ;  **INSERT** **INTO** BANK\_CUSTOMER\_EXPORT **VALUES** ("123002 ","George", "194-6,New brighton","MN" , "189761700");  **CREATE** **TABLE** Bank\_Account\_Details(Customer\_id **INT**,                                  Account\_Number **VARCHAR**(19),                                  Account\_type **VARCHAR**(25),                                  Balance\_amount **INT**,                                  Account\_status **VARCHAR**(10),                                  Relationship\_type **varchar**(1) ) ;  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123001, "4000-1956-3456",  "SAVINGS" , 200000 ,"ACTIVE","P");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123001, "5000-1700-3456", "RECURRING DEPOSITS" ,9400000 ,"ACTIVE","S");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123002, "4000-1956-2001",  "SAVINGS", 400000 ,"ACTIVE","P");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123002, "5000-1700-5001",  "RECURRING DEPOSITS" ,7500000 ,"ACTIVE","S");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123003, "4000-1956-2900",  "SAVINGS" ,750000,"INACTIVE","P");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123004, "5000-1700-6091", "RECURRING DEPOSITS" ,7500000 ,"ACTIVE","S");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123004, "4000-1956-3401",  "SAVINGS" , 655000 ,"ACTIVE","P");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123005, "4000-1956-5102",  "SAVINGS" , 300000 ,"ACTIVE","P");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123006, "4000-1956-5698",  "SAVINGS" , 455000 ,"ACTIVE" ,"P");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123007, "5000-1700-9800",  "SAVINGS" , 355000 ,"ACTIVE" ,"P");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123007, "4000-1956-9977",  "RECURRING DEPOSITS" , 7025000,"ACTIVE" ,"S");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123007, "9000-1700-7777-4321",  "Credit Card" ,0  ,"INACTIVE", "P");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123007, '5900-1900-9877-5543', "Add-on Credit Card" ,   0   ,"ACTIVE", "S");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123008, "5000-1700-7755",  "SAVINGS"      ,0      ,"INACTIVE","P");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123006, '5800-1700-9800-7755', "Credit Card"   ,0     ,"ACTIVE", "P");  **INSERT** **INTO** Bank\_Account\_Details  **VALUES** (123006, '5890-1970-7706-8912', "Add-on Credit Card"   ,0      ,"ACTIVE", "S");    # **CREATE** Bank\_Account **Table**:  # **Create** **Table**  **CREATE** **TABLE** BANK\_ACCOUNT ( Customer\_id **INT**,                      Account\_Number **VARCHAR**(19),               Account\_type **VARCHAR**(25),               Balance\_amount **INT** ,              Account\_status **VARCHAR**(10), Relation\_ship **varchar**(1) ) ;  # **Insert** records:  **INSERT** **INTO** BANK\_ACCOUNT  **VALUES** (123001, "4000-1956-3456",  "SAVINGS"            , 200000 ,"ACTIVE","P");  **INSERT** **INTO** BANK\_ACCOUNT  **VALUES** (123001, "5000-1700-3456",  "RECURRING DEPOSITS" ,9400000 ,"ACTIVE","S");  **INSERT** **INTO** BANK\_ACCOUNT  **VALUES** (123002, "4000-1956-2001",  "SAVINGS"            , 400000 ,"ACTIVE","P");  **INSERT** **INTO** BANK\_ACCOUNT  **VALUES** (123002, "5000-1700-5001",  "RECURRING DEPOSITS" ,7500000 ,"ACTIVE","S");  **INSERT** **INTO** BANK\_ACCOUNT  **VALUES** (123003, "4000-1956-2900",  "SAVINGS"            ,750000,"INACTIVE","P");  **INSERT** **INTO** BANK\_ACCOUNT  **VALUES** (123004, "5000-1700-6091",  "RECURRING DEPOSITS" ,7500000 ,"ACTIVE","S");  **INSERT** **INTO** BANK\_ACCOUNT  **VALUES** (123004, "4000-1956-3401",  "SAVINGS"            , 655000 ,"ACTIVE","P");  **INSERT** **INTO** BANK\_ACCOUNT  **VALUES** (123005, "4000-1956-5102",  "SAVINGS"            , 300000 ,"ACTIVE","P");  **INSERT** **INTO** BANK\_ACCOUNT  **VALUES** (123006, "4000-1956-5698",  "SAVINGS"            , 455000 ,"ACTIVE" ,"P");  **INSERT** **INTO** BANK\_ACCOUNT  **VALUES** (123007, "5000-1700-9800",  "SAVINGS"            , 355000 ,"ACTIVE" ,"P");  **INSERT** **INTO** BANK\_ACCOUNT  **VALUES** (123007, "4000-1956-9977",  "RECURRING DEPOSITS" , 7025000,"ACTIVE" ,"S");  **INSERT** **INTO** BANK\_ACCOUNT  **VALUES** (123007, "9000-1700-7777-4321",  "CREDITCARD"    ,0      ,"INACTIVE","P");  **INSERT** **INTO** BANK\_ACCOUNT  **VALUES** (123008, "5000-1700-7755",  "SAVINGS"            ,NULL   ,"INACTIVE","P");          # **CREATE** **TABLE** Bank\_Account\_Relationship\_Details    **CREATE** **TABLE** Bank\_Account\_Relationship\_Details                                  ( Customer\_id **INT**,                                  Account\_Number **VARCHAR**(19),                                  Account\_type **VARCHAR**(25),                                  Linking\_Account\_Number **VARCHAR**(19));  **INSERT** **INTO** Bank\_Account\_Relationship\_Details  **VALUES** (123001, "4000-1956-3456",  "SAVINGS" , "");  **INSERT** **INTO** Bank\_Account\_Relationship\_Details  **VALUES** (123001, "5000-1700-3456",  "RECURRING DEPOSITS" , "4000-1956-3456");  **INSERT** **INTO** Bank\_Account\_Relationship\_Details  **VALUES** (123002, "4000-1956-2001",  "SAVINGS" , "" );  **INSERT** **INTO** Bank\_Account\_Relationship\_Details  **VALUES** (123002, "5000-1700-5001",  "RECURRING DEPOSITS" , "4000-1956-2001" );  **INSERT** **INTO** Bank\_Account\_Relationship\_Details  **VALUES** (123003, "4000-1956-2900",  "SAVINGS" , "" );  **INSERT** **INTO** Bank\_Account\_Relationship\_Details  **VALUES** (123004, "5000-1700-6091",  "RECURRING DEPOSITS" , "4000-1956-2900" );  **INSERT** **INTO** Bank\_Account\_Relationship\_Details  **VALUES** (123004, "5000-1700-7791",  "RECURRING DEPOSITS" , "4000-1956-2900" );  **INSERT** **INTO** Bank\_Account\_Relationship\_Details  **VALUES** (123007, "5000-1700-9800",  "SAVINGS" , "" );  **INSERT** **INTO** Bank\_Account\_Relationship\_Details  **VALUES** (123007, "4000-1956-9977",  "RECURRING DEPOSITS" , "5000-1700-9800" );  **INSERT** **INTO** Bank\_Account\_Relationship\_Details  **VALUES** (NULL, "9000-1700-7777-4321",  "Credit Card" , "5000-1700-9800" );  **INSERT** **INTO** Bank\_Account\_Relationship\_Details  **VALUES** (NULL, '5900-1900-9877-5543', 'Add-on Credit Card', '9000-1700-7777-4321' );  **INSERT** **INTO** Bank\_Account\_Relationship\_Details  **VALUES** (NULL, '5800-1700-9800-7755', 'Credit Card', '4000-1956-5698' );  **INSERT** **INTO** Bank\_Account\_Relationship\_Details  **VALUES** (NULL, '5890-1970-7706-8912', 'Add-on Credit Card', '5800-1700-9800-7755' );        # **CREATE** **TABLE** BANK\_ACCOUNT\_TRANSACTION    **CREATE** **TABLE** BANK\_ACCOUNT\_TRANSACTION (                  Account\_Number **VARCHAR**(19),                  Transaction\_amount **Decimal**(18,2) ,                  Transcation\_channel **VARCHAR**(18) ,                  Province **varchar**(3) ,                  Transaction\_Date **Date**) ;      **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "4000-1956-3456",  -2000, "ATM withdrawl" , "CA", "2020-01-13");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "4000-1956-2001",  -4000, "POS-Walmart"   , "MN", "2020-02-14");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "4000-1956-2001",  -1600, "UPI transfer"  , "MN", "2020-01-19");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "4000-1956-2001",  -6000, "Bankers cheque", "CA", "2020-03-23");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "4000-1956-2001",  -3000, "Net banking"   , "CA", "2020-04-24");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "4000-1956-2001",  23000, "cheque deposit", "MN", "2020-03-15");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "5000-1700-6091",  40000, "ECS transfer"  , "NY", "2020-02-19");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "5000-1700-7791",  40000, "ECS transfer"  , "NY", "2020-02-19");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "4000-1956-3401",   8000, "Cash Deposit"  , "NY", "2020-01-19");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "4000-1956-5102",  -6500, "ATM withdrawal" , "NY", "2020-03-14");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "4000-1956-5698",  -9000, "Cash Deposit"  , "NY", "2020-03-27");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "4000-1956-9977",  50000, "ECS transfer"  , "NY", "2020-01-16");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "9000-1700-7777-4321",  -5000, "POS-Walmart", "NY", "2020-02-17");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "9000-1700-7777-4321",  -8000, "Shopping Cart", "MN", "2020-03-13");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "9000-1700-7777-4321",  -2500, "Shopping Cart", "MN", "2020-04-21");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( "5800-1700-9800-7755", -9000, "POS-Walmart","MN", "2020-04-13");  **INSERT** **INTO** BANK\_ACCOUNT\_TRANSACTION **VALUES** ( '5890-1970-7706-8912', -11000, "Shopping Cart" , "NY" , "2020-03-12") ;        # **CREATE** **TABLE** BANK\_CUSTOMER\_MESSAGES    **CREATE** **TABLE** BANK\_CUSTOMER\_MESSAGES (                  Event **VARCHAR**(24),                  Customer\_message **VARCHAR**(75),                  Notice\_delivery\_mode **VARCHAR**(15)) ;      **INSERT** **INTO** BANK\_CUSTOMER\_MESSAGES **VALUES** ( "Adhoc", "All Banks are closed due to announcement of National strike", "mobile" ) ;  **INSERT** **INTO** BANK\_CUSTOMER\_MESSAGES **VALUES** ( "Transaction Limit", "Only limited withdrawals per card are allowed from ATM machines", "mobile" );  **INSERT** **INTO** `bank\_account\_transaction`(`Account\_Number`, `Transaction\_amount`, `Transcation\_channel`, `Province`, `Transaction\_Date`) **VALUES**  ('4000-1956-9977' ,    10000.00     ,'ECS transfer',     'MN' ,    '2020-02-16' ) ;    -- inserted for queries after 17th  **INSERT** **INTO** `bank\_account\_transaction`(`Account\_Number`, `Transaction\_amount`, `Transcation\_channel`, `Province`, `Transaction\_Date`) **VALUES**  ('4000-1956-9977' ,    40000.00     ,'ECS transfer',     'MN' ,    '2020-03-18' ) ;    **INSERT** **INTO** `bank\_account\_transaction`(`Account\_Number`, `Transaction\_amount`, `Transcation\_channel`, `Province`, `Transaction\_Date`) **VALUES**  ('4000-1956-9977' ,    60000.00     ,'ECS transfer',     'MN' ,    '2020-04-18' ) ;    **INSERT** **INTO** `bank\_account\_transaction`(`Account\_Number`, `Transaction\_amount`, `Transcation\_channel`, `Province`, `Transaction\_Date`) **VALUES**  ('4000-1956-9977' ,    20000.00     ,'ECS transfer',     'MN' ,    '2020-03-20' ) ;    -- inserted for queries after 24th    **INSERT** **INTO** `bank\_account\_transaction`(`Account\_Number`, `Transaction\_amount`, `Transcation\_channel`, `Province`, `Transaction\_Date`) **VALUES**  ('4000-1956-9977' ,    49000.00     ,'ECS transfer',     'MN' ,    '2020-06-18' ) ;          # **CREATE** **TABLE** BANK\_INTEREST\_RATE    **CREATE** **TABLE** BANK\_INTEREST\_RATE(                  account\_type **varchar**(24),                  interest\_rate **decimal**(4,2),                  month **varchar**(2),                  year  **varchar**(4)                  )   ;    **INSERT**  **INTO** BANK\_INTEREST\_RATE **VALUES** ( "SAVINGS" , 0.04 , '02' , '2020' );  **INSERT**  **INTO** BANK\_INTEREST\_RATE **VALUES** ( "RECURRING DEPOSITS" , 0.07, '02' , '2020' );  **INSERT**  **INTO** BANK\_INTEREST\_RATE **VALUES**   ( "PRIVILEGED\_INTEREST\_RATE" , 0.08 , '02' , '2020' );      # Bank\_holidays:    **Insert** **into** bank\_holidays **values**( '2020-05-20', now(), now() ) ;    **Insert** **into** bank\_holidays **values**( '2020-03-13' , now(), now() ) ; |

Print customer Id, customer name and average account\_balance maintained by each customer for all of his/her accounts in the bank.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | **Select** bc.customer\_id , customer\_name, avg(ba.Balance\_amount) **as** All\_account\_balance\_amount  **from** bank\_customer bc  **inner** join  Bank\_Account\_Details ba  **on** bc.customer\_id = ba.Customer\_id  **group** **by** bc.customer\_id, bc.customer\_name; |

Print customer\_id , account\_number and balance\_amount ,

#condition that if balance\_amount is nil then assign transaction\_amount  for account\_type = “Credit Card”

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | **Select** customer\_id , ba.account\_number,  Case **when** ifnull(balance\_amount,0) = 0 **then**   Transaction\_amount **else** balance\_amount **end**  **as** balance\_amount  **from** Bank\_Account\_Details ba  **inner** join  bank\_account\_transaction bat  **on** ba.account\_number = bat.account\_number  and account\_type = "Credit Card"; |

Print customer\_id , account\_number and balance\_amount ,

# conPrint account number,  balance\_amount, transaction\_amount from Bank\_Account\_Details and bank\_account\_transaction

# for all the transactions occurred during march,2020 and april, 2020

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | **Select**  ba.Account\_Number, Balance\_amount, Transaction\_amount, Transaction\_Date  **from** Bank\_Account\_Details ba  **inner** join  bank\_account\_transaction bat  **on** ba.account\_number = bat.account\_number  And ( Transaction\_Date between "2020-03-01" and "2020-04-30");  -- or use below condition --  # (date\_format(Transaction\_Date , '%Y-%m')  between "2020-03" and "2020-04"); |

Print all of the customer id, account number,  balance\_amount, transaction\_amount from bank\_customer,

# Bank\_Account\_Details and bank\_account\_transaction tables where excluding all of their transactions in march, 2020  month

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | **Select**  ba.Customer\_id,  ba.Account\_Number, Balance\_amount, Transaction\_amount, Transaction\_Date  **from** Bank\_Account\_Details ba  Left join bank\_account\_transaction bat  **on** ba.account\_number = bat.account\_number  And NOT ( date\_format(Transaction\_Date , '%Y-%m') = "2020-03" ); |

Print only the customer id, customer name, account\_number, balance\_amount who did transactions during the first quarter.

# Do not display the accounts if they have not done any transactions in the first quarter.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | **Select**  ba.Customer\_id,  ba.Account\_Number, Balance\_amount , transaction\_amount , transaction\_date **from**  Bank\_Account\_Details ba  **Inner** join bank\_account\_transaction bat  **on** ba.account\_number = bat.account\_number  And ( date\_format(Transaction\_Date , '%Y-%m') <= "2020-03" ); |

Print account\_number, Event adn Customer\_message from BANK\_CUSTOMER\_MESSAGES and Bank\_Account\_Details to display an “Adhoc”

# Event for all customers who have  “SAVINGS” account\_type account.

|  |  |
| --- | --- |
| 1  2  3  4  5 | **SELECT** Account\_Number, Event , Customer\_message  **FROM** Bank\_Account\_Details  CROSS JOIN  BANK\_CUSTOMER\_MESSAGES  **ON** Event  = "Adhoc"  And ACCOUNT\_TYPE = "SAVINGS"; |

Print Customer\_id, Account\_Number, Account\_type, and display deducted balance\_amount by

# subtracting only negative transaction\_amounts for Relationship\_type = “P” ( P – means  Primary , S – means Secondary )

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | **SELECT**      ba.Customer\_id,      ba.Account\_Number,      (Balance\_amount + IFNULL(transaction\_amount, 0)) deducted\_balance\_amount    **FROM** Bank\_Account\_Details ba  LEFT JOIN bank\_account\_transaction bat  **ON** ba.account\_number = bat.account\_number  AND Relationship\_type = "P"; |

Display records of All Accounts, their Account\_types, the transaction amount.

# b) Along with the first step, Display other columns with the corresponding linking account number, account types

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | **SELECT**  br1.Account\_Number primary\_account ,          br1.Account\_type primary\_account\_type,          br2.Account\_Number Seconday\_account,          br2.Account\_type Seconday\_account\_type  **FROM** `bank\_account\_relationship\_details` br1  LEFT JOIN `bank\_account\_relationship\_details` br2  **ON** br1.account\_number = br2.linking\_account\_number; |

Display records of All Accounts, their Account\_types, the transaction amount.

# b) Along with the first step, Display other columns with corresponding linking account number, account types

# c) After retrieving all records of accounts and their linked accounts, display the transaction amount of accounts appeared in another column.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | **SELECT** br1.Account\_Number primary\_account\_number ,  br1.Account\_type         primary\_account\_type,  br2.Account\_Number      secondary\_account\_number,  br2.Account\_type        secondary\_account\_type,  bt1.Transaction\_amount   primary\_acct\_tran\_amount  **from** bank\_account\_relationship\_details br1  LEFT JOIN bank\_account\_relationship\_details br2  **on** br1.Account\_Number = br2.Linking\_Account\_Number  LEFT JOIN bank\_account\_transaction bt1  **on** br1.Account\_Number  = bt1.Account\_Number; |

Display all saving account holders have “Add-on Credit Cards” and “Credit cards”

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | **SELECT**  br1.Account\_Number  primary\_account\_number ,  br1.Account\_type  primary\_account\_type,  br2.Account\_Number secondary\_account\_number,  br2.Account\_type secondary\_account\_type  **from** bank\_account\_relationship\_details br1  JOIN bank\_account\_relationship\_details br2  **on** br1.Account\_Number = br2.Linking\_Account\_Number  and br2.Account\_type like '%Credit%' ; |

That covers the most asked or SQL practiced questions.

**Frequently Asked Questions in SQL**

**1. How do I prepare for the SQL interview?**

Many online sources can help you prepare for an SQL interview. You can go through brief tutorials and free online courses on SQL (eg.: SQL basics on [Great Learning Academy](https://www.mygreatlearning.com/academy)) to revise your knowledge of SQL. You can also practice projects to help you with practical aspects of the language. Lastly, many blogs such as this list all the probable questions that an interviewer might ask.

**2. What are the 5 basic SQL commands?**

The five basic SQL commands are:

* Data Definition Language (DDL)
* Data Manipulation Language (DML)
* Data Control Language (DCL)
* Transaction Control Language (TCL)
* Data Query Language (DQL)

**3. What are basic SQL skills?**

SQL is a vast topic and there is a lot to learn. But the most basic skills that an SQL professional should know are:

* How to structure a database
* Managing a database
* Authoring SQL statements and clauses
* Knowledge of popular database systems such as MySQL
* Working knowledge of [PHP](https://www.mygreatlearning.com/php/free-courses/?gl_blog_id=15225)
* SQL data analysis
* Creating a database with WAMP and SQL

**4. How can I practice SQL?**

There are some platforms available online that can help you practice SQL such as SQL Fiddle, SQLZOO, W3resource, Oracle LiveSQL, DB-Fiddle, Coding Groud, GitHub and others. Also take up a [Oracle SQL](https://www.mygreatlearning.com/academy/learn-for-free/courses/oracle-sql?gl_blog_id=15225) to learn more.

**5. Where can I practice SQL questions?**

There are some platforms available online that can help you practice SQL such as SQL Fiddle, SQLZOO, W3resource, Oracle LiveSQL, DB-Fiddle, Coding Groud, GitHub and others.

You can also refer to articles and blogs online that list the most important SQL interview questions for preparation.

**6. What is the most common SQL command?**

Some of the most common SQL commands are:

* CREATE DATABASE
* ALTER DATABASE
* CREATE TABLE
* ALTER TABLE
* DROP TABLE
* CREATE INDEX
* DROP INDEX

**7. How are SQL commands classified?**

SQL Commands are classified under four categories, i.e.,

* Data Definition Language (DDL)
* Data Query Language (DQL)
* Data Manipulation Language (DML)
* Data Control Language (DCL)

**8. What are basic SQL commands?**

Basic SQL commands are:

* CREATE DATABASE
* ALTER DATABASE
* CREATE TABLE
* ALTER TABLE
* DROP TABLE
* CREATE INDEX
* DROP INDEX

**9. Is SQL coding?**

Yes, SQL is a coding language/ programming language that falls under the category of domain-specific programming language. It is used to access relational databases such as MySQL.

**10. What is SQL example?**

SQL helps you update, delete, and request information from databases. Some of the examples of SQL are in the form of the following statements:

* SELECT
* INSERT
* UPDATE
* DELETE
* CREATE DATABASE
* ALTER DATABASE

**11. What is SQL code used for?**

SQL code is used to access and communicate with a database. It helps in performing tasks such as updating and retrieving data from the databases.

## ****BASIC QUESTIONS TO REVISE CONCEPTS****

### ****1. How to CREATE a table in SQL?****

To create a table in SQL, you all need is table’s name, the column’s name, the data type of the column, and the size of the table.  
Here is the syntax you must follow to create a table in SQL.

**Syntax**

CREATE TABLE table\_name(

column\_name\_1 datatype(size) column\_constraints,

column\_name\_2 datatype(size) column\_constraints,

…..

…..

);

**Copy code**

Let’s take an example to get a better understanding of creating a table in SQL.

**Problem Statement**: Create an Employee Table, having column names EmployeeID, Name, Gender, Department, Salary,

CREATE TABLE Employee(

EmployeeID int PRIMARY KEY,

Name VARCHAR(100) NOT NULL,

Gender text NOT NULL,

Department VARCHAR(30),

Salary VARCHAR (20)

);

**Copy code**

Here is the output.

**Output**



**Must Read:** [How to CREATE a table in SQL?](https://www.shiksha.com/online-courses/articles/how-to-create-table-in-sql/)

Now, let’s see how to insert the data into table.

### ****2. How do INSERT data into the table in SQL?****

To INSERT the data into the created table, we have to use INSERT INTO clause.

**Syntax**

INSERT INTO table\_name (column\_name1, column\_name2, …..) VALUES (value1, value2, ……)

**Copy code**

Now, let’s insert the data in the above Employee table.

INSERT INTO Employee (EmployeeID, Name, Gender, Department, Salary)

VALUES

(1001, 'Ajay', 'M', 'Engineering', 45000),

(1002, 'Babloo', 'M', 'Engineering', 25000),

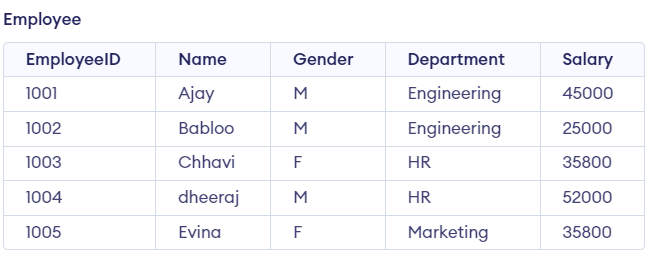
(1003, 'Chhavi', 'F', 'HR', 35800),

(1004, 'dheeraj', 'M', 'HR', 52000),

(1005, 'Evina', 'F', 'Marketing', 35800);

**Copy code**

**Output**



**Must Read:** [*How to use INSERT clause in SQL?*](https://www.shiksha.com/online-courses/articles/how-to-use-insert-in-sql/)

### ****3.**** ****How to SELECT the record from the table in SQL?****

To select the record from the table, we have to use SELECT statement in SQL.

**Syntax**

SELECT column\_name

FROM table\_name

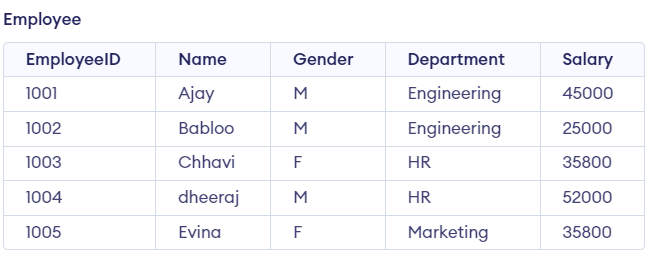
**Copy code**

**Example-1: Print all the record from the Employee Table.**

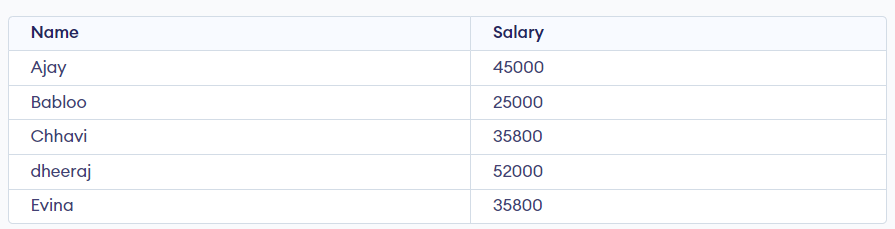
SELECT \*

FROM Employee;

**Copy code**



**Example-2: Print the Name and Salary of the employees from the Employee Table.**



**Must Read:** [*How to use SELECT statement in SQL?*](https://www.shiksha.com/online-courses/articles/select-statement-in-sql/)

### ****4. How to filter rows in a table using the WHERE clause in SQL?****

WHERE clause in SQL is used to filter out the records or to specify a condition while extracting the records from a single table or joining multiple tables.  
WHERE clause follows the SELECT and FROM clause.

**Syntax**

SELECT column\_name

FROM table\_name

WHERE conditions;

**Copy code**

**Example: Find the complete detail of the EmployeeID is 1003.**

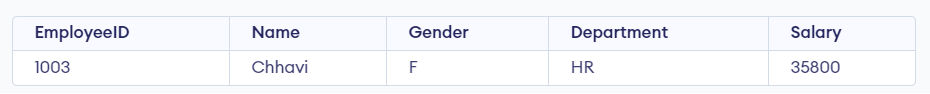
SELECT \*

FROM Employee

WHERE EmployeeID = 1003;

**Copy code**

**Output**



**Must Read:** [How to use WHERE clause in SQL?](https://www.shiksha.com/online-courses/articles/how-to-use-where-clause-in-sql/)

### ****5. How to group filter data based on the specific condition using GROUP BY?****

GROUP BY clause in SQL is used to group all the rows that have the same value by one or more columns. It follows the WHERE clause in the SELECT statement and precedes the ORDER BY clause.  
Note: It is used in conjunction with the aggregate function.

**Syntax**

SELECT column\_names

FROM table\_name

WHERE conditions

GROUP BY column\_names

ORDER BY column\_names;

**Copy code**

**Example: Use the GROUP BY clause to count the number of employees in each department.mployee table.**

SELECT Department, COUNT(EmployeeID) AS 'Number of Employee'

FROM Employee

GROUP BY Department

**Copy code**

**Output**



**Must Read:** [*How to use GROUP BY clause in SQL?*](https://www.shiksha.com/online-courses/articles/how-to-use-group-by-in-sql/)

### ****6. How to filter groups in a table using the HAVING clause in SQL?****

The HAVING clause in SQL is quite similar to the WHERE clause. Similar to the WHERE clause, it filters the data but in a different way.

* It filters the result obtained after the GROUP BY clause.
* It can include one or more conditions.
* ORDER of Execution: After the HAVING clause and before the ORDER BY clause.

**Syntax**

SELECT column\_names

FROM table\_name

WHERE conditions

GROUP BY column\_name

HAVING conditions

ORDER BY column\_name;

**Copy code**

**Example: Using HAVING clause determine the department having number of employees greater than 1.**

SELECT Department, Salary, COUNT(EmployeeID) AS 'Number of Employee'

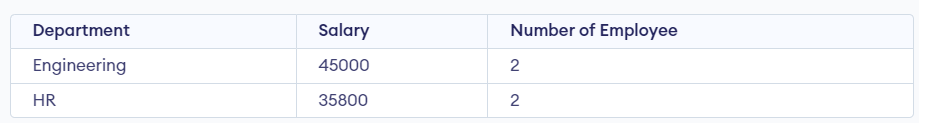
FROM Employee

GROUP BY Department

HAVING COUNT(EmployeeID) > 1;

**Copy code**

**Output**



**Must Read:** [*How to use HAVING clause in SQL?*](https://www.shiksha.com/online-courses/articles/how-to-use-having-clause-in-sql/)

### ****7. How to Sort the data using the ORDER BY clause in SQL?****

SQL ORDER BY clause is used to arrange the output either in the Ascending or Descending Order.

**Syntax**

SELECT column\_list

FROM table\_name

WHERE conditions

ORDER BY column\_names [ASC | DESC];

**Copy code**

Note:

* ASC: Ascending Order (By Default)
* DESC: Descending Order

**Example: List the record of all the employees in decreasing value of their salary.**

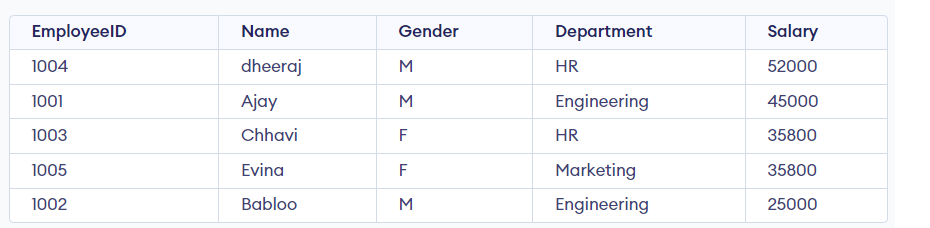
SELECT \*

FROM Employee

ORDER BY Salary DESC;

**Copy code**

**Output**



**Must Read:** [*How to use ORDER BY clause in SQL?*](https://www.shiksha.com/online-courses/articles/how-to-use-order-by-clause-in-sql/)

### ****How to restrict the records using the LIMIT clause in SQL?****

The LIMIT clause in SQL is used to set an upper limit on the number of tuples returned for any given query.  
It is an efficient way to avoid long-running queries as it stops processing and returns the result as soon as the requirement is fulfilled. This reduces the processing time.

**Example: Find the record of top 3 employees who are getting highest salary**

SELECT \*

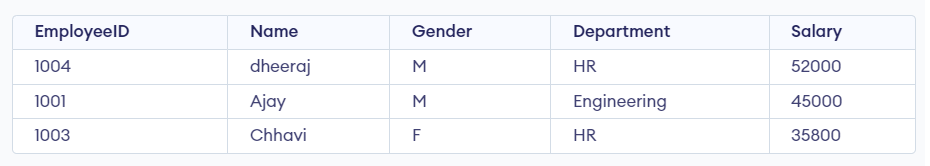
FROM Employee

ORDER BY Salary DESC

LIMIT 3;

**Copy code**

**Output**



**Must Read:** [*How to use LIMIT clause in SQL?*](https://www.shiksha.com/online-courses/articles/how-to-use-limit-clause-in-sql/)

Till now, you get enough understanding of how to create a table, how to insert the data into the table, how to filter the records using WHERE and HAVING clause, and you also know how to sort the data.

Now, we will learn how to remove the records from the table.

So, let’s start with removing duplicates from the table.

### ****How to remove the duplicate entries using the DISTINCT clause in SQL?****

Now, let’s consider you have 1000 of records and you have to find distinct department in the company using the Employee table. Then, here DISTINCT commands come into action.

To remove the duplicate values from the dataset, we use the DISTINCT clause in SQL. The DISTINCT clause returns only unique values (rows).  
Note:

* It can be used with aggregate functions like COUNT, AVG, MAX, MIN, etc.
* It can only be operated on a single column.
* Includes NULL value as a distinct value.
* If the SQL query consists of an ORDER BY clause or any aggregate function, then the limit clause is the last to be evaluated.

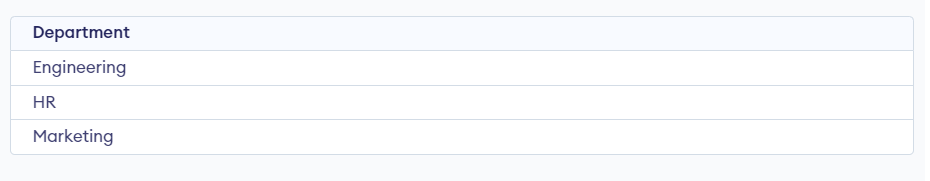
**Example: Find all the departments** **from the Employee table that are distinct.**

SELECT DISTINCT Department

FROM Employee

**Copy code**

**Output**



**Must Read:** [How to use DISTINCT clause in SQL?](https://www.shiksha.com/online-courses/articles/how-to-use-distinct-in-sql/)

## ****Leet Code SQL Practice Problem****

### ****Easy****

#### **Question-1: Write a solution to find the**

customer\_number

#### Copy code

#### **for the customer who has placed the largest number of orders.**

|  |  |
| --- | --- |
| **Column Name** | **Type** |
| order\_name | int |
| customer\_name | int |

* **order\_number** is the primary key (column with unique values) for this table.
* This table contains information about the order ID and the customer ID.

The test cases are generated so that **exactly one customer** will have placed more orders than any other customer.

The result format is in the following example.

**Example:**

Input:

Orders Table:

+--------------+-----------------+

| order\_number | customer\_number |

+--------------+-----------------+

| 1 | 1 |

| 2 | 2 |

| 3 | 3 |

| 4 | 3 |

+--------------+-----------------+

**Output:**

+-----------------+

| customer\_number |

+-----------------+

| 3 |

+-----------------+

**Explanation to the test case:**  
The customer with number 3 has two orders, which is greater than either customer 1 or 2 because each of them only has one order.  
So the result is customer\_number 3.

**Solution**

select customer\_number from Orders group by customer\_number order by count(customer\_number) desc limit 1;

**Copy code**

**Explanation** **to the solution**

Since we have to find the customer\_number for the customer who has placed the largest number of orders.  
So, let’s start with:

* SELECT column customer\_number FROM tables Orders.
* GROUP BY column customer\_number
* COUNT the number of occurrences of values in the column customer\_number
* ORDER BY the COUNT in DESC (descending) order
* and since we have to find the customer number who has placed the maximum order, so LIMIT the result by 1.

#### **Question-2: Report for every three-line segments whether they can form a triangle. Return the result table in any order.**

**Table:**

Trian

**Copy code**

gle

**Copy code**

+-------------+------+

| Column Name | Type |

+-------------+------+

| x | int |

| y | int |

| z | int |

+-------------+------+

* In SQL, (x, y, z) is the primary key column for this table.

Each row of this table contains the lengths of three-line segments.

The result format is in the following example.

**Example:**

**Input:**

Triangle table:

+----+----+----+

| x | y | z |

+----+----+----+

| 13 | 15 | 30 |

| 10 | 20 | 15 |

+----+----+----+

**Output:**

+----+----+----+----------+

| x | y | z | triangle |

+----+----+----+----------+

| 13 | 15 | 30 | No |

| 10 | 20 | 15 | Yes |

+----+----+----+----------+

**Solution**

select

x,y,z,

case when (x+y) > z and (x+z) > y and (y+z) > x then 'Yes' else 'No' end as triangle

from Triangle

**Copy code**

**Explanation to Solution**

Here, we are given three sides of the triangle and using those sides, we have to check whether they can form a triangle or not. And for that, we will use an important property of a triangle:

* **Sum of two sides of a triangle is always greater than the third side.**

Now, let’s check the input set in the given example.

* in the first set, the combination 13+15 < 30, i.e., it fails to be a triangle.
* in the second set, all the combination 10+20 > 15, 20+15 > 10, 10+15 > 20 satisfies the above given condition.

Now, to find the solution of the above problem we will use CASE, WHEN, THEN clause.

#### **Question-3: Write a solution to find the names of all the salespersons who did not have any orders related to the company with the name “RED”. Return the result table in any order.**

**Table:**

SalesPerson

**Copy code**

+-----------------+---------+

| Column Name | Type |

+-----------------+---------+

| sales\_id | int |

| name | varchar |

| salary | int |

| commission\_rate | int |

| hire\_date | date |

+-----------------+---------+

* sales\_id is this table’s primary key (column with unique values).

Each row of this table indicates a salesperson’s name and ID alongside their salary, commission rate, and hire date.

**Table:**

Company

**Copy code**

+-------------+---------+

| Column Name | Type |

+-------------+---------+

| com\_id | int |

| name | varchar |

| city | varchar |

+-------------+---------+

com\_id is this table’s primary key (column with unique values).

Each row of this table indicates a company’s name and ID and the city in which the company is located.

**Table:**

Orders

**Copy code**

+-------------+------+

| Column Name | Type |

+-------------+------+

| order\_id | int |

| order\_date | date |

| com\_id | int |

| sales\_id | int |

| amount | int |

+-------------+------+

* order\_id is this table’s primary key (column with unique values).
* com\_id is a foreign key (reference column) to com\_id from the Company table.
* sales\_id is a foreign key (reference column) to sales\_id from the SalesPerson table.

Each row of this table contains information about one order. This includes the company’s ID, the salesperson’s ID, the order’s date, and the amount paid.

The result format is in the following example.

**Example:**

**Input:**

SalesPerson table:

+----------+------+--------+-----------------+------------+

| sales\_id | name | salary | commission\_rate | hire\_date |

+----------+------+--------+-----------------+------------+

| 1 | John | 100000 | 6 | 4/1/2006 |

| 2 | Amy | 12000 | 5 | 5/1/2010 |

| 3 | Mark | 65000 | 12 | 12/25/2008 |

| 4 | Pam | 25000 | 25 | 1/1/2005 |

| 5 | Alex | 5000 | 10 | 2/3/2007 |

+----------+------+--------+-----------------+------------+

Company table:

+--------+--------+----------+

| com\_id | name | city |

+--------+--------+----------+

| 1 | RED | Boston |

| 2 | ORANGE | New York |

| 3 | YELLOW | Boston |

| 4 | GREEN | Austin |

+--------+--------+----------+

Orders table:

+----------+------------+--------+----------+--------+

| order\_id | order\_date | com\_id | sales\_id | amount |

+----------+------------+--------+----------+--------+

| 1 | 1/1/2014 | 3 | 4 | 10000 |

| 2 | 2/1/2014 | 4 | 5 | 5000 |

| 3 | 3/1/2014 | 1 | 1 | 50000 |

| 4 | 4/1/2014 | 1 | 4 | 25000 |

+----------+------------+--------+----------+--------+

**Output:**

+------+

| name |

+------+

| Amy |

| Mark |

| Alex |

+------+

**Explanation to Example**

According to orders 3 and 4 in the Orders table, it is easy to tell that only salesperson John and Pam have sales to company RED, so we report all the other names in the table salesperson.

**Solution**

Here, we will not use the concept of JOINS rather than Subquery.

select salesperson.name

from orders o join company c on (o.com\_id = c.com\_id and c.name = 'RED')

right join salesperson on salesperson.sales\_id = o.sales\_id

where o.sales\_id is null

**Copy code**

#### **Question-4: Actors and Directors Who Cooperated At Least Three Times**

**Write a solution to find all the pairs (actor\_id, director\_id) where the actor has cooperated with the director at least three times. Return the result table in any order.**

**Table:**

ActorDirector

**Copy code**

+-------------+---------+

| Column Name | Type |

+-------------+---------+

| actor\_id | int |

| director\_id | int |

| timestamp | int |

+-------------+---------+

* timestamp is this table’s primary key (column with unique values).

The result format is in the following example.

**Example:**

**Input:**

ActorDirector table:

+-------------+-------------+-------------+

| actor\_id | director\_id | timestamp |

+-------------+-------------+-------------+

| 1 | 1 | 0 |

| 1 | 1 | 1 |

| 1 | 1 | 2 |

| 1 | 2 | 3 |

| 1 | 2 | 4 |

| 2 | 1 | 5 |

| 2 | 1 | 6 |

+-------------+-------------+-------------+

**Output:**

+-------------+-------------+

| actor\_id | director\_id |

+-------------+-------------+

| 1 | 1 |

+-------------+-------------+

**Explanation**

The only pair is (1, 1) where they cooperated exactly 3 times.

**Solution**

select actor\_id, director\_id

from(

select actor\_id,director\_id,

count(timestamp) as cooperated

from ActorDirector

group by actor\_id,director\_id)

table1

where cooperated>=3;

**Copy code**

You can use HAVING clause, rather than WHERE clause. It makes you write the simple and readable code.

select actor\_id,director\_id

from ActorDirector

group by actor\_id,director\_id

Having count(timestamp)>=3;

**Copy code**

#### **Question-5: Product Sales Analysis**

**Write a solution to report the product\_name, year, and price for each sale\_id in the Sales table. Return the resulting table in any order.**

Table:

Sales

**Copy code**

+-------------+-------+

| Column Name | Type |

+-------------+-------+

| sale\_id | int |

| product\_id | int |

| year | int |

| quantity | int |

| price | int |

+-------------+-------+

* (sale\_id, year) is the primary key (combination of columns with unique values) of this table.
* product\_id is a foreign key (reference column) to Product table.
* Each row of this table shows a sale on the product product\_id in a certain year. Note that the price is per unit.

Table:

Product

**Copy code**

+--------------+---------+

| Column Name | Type |

+--------------+---------+

| product\_id | int |

| product\_name | varchar |

+--------------+---------+

* product\_id is the primary key (column with unique values) of this table. Each row of this table indicates the product name of each product.

The result format is in the following example.

**Example 1:**

**Input:**

Sales table:

+---------+------------+------+----------+-------+

| sale\_id | product\_id | year | quantity | price |

+---------+------------+------+----------+-------+

| 1 | 100 | 2008 | 10 | 5000 |

| 2 | 100 | 2009 | 12 | 5000 |

| 7 | 200 | 2011 | 15 | 9000 |

+---------+------------+------+----------+-------+

Product table:

+------------+--------------+

| product\_id | product\_name |

+------------+--------------+

| 100 | Nokia |

| 200 | Apple |

| 300 | Samsung |

+------------+--------------+

**Output:**

+--------------+-------+-------+

| product\_name | year | price |

+--------------+-------+-------+

| Nokia | 2008 | 5000 |

| Nokia | 2009 | 5000 |

| Apple | 2011 | 9000 |

+--------------+-------+-------+

**Explanation of Example:**

* From sale\_id = 1, we can conclude that Nokia was sold for 5000 in the year 2008.
* From sale\_id = 2, we can conclude that Nokia was sold for 5000 in the year 2009.
* From sale\_id = 7, we can conclude that Apple was sold for 9000 in the year 2011.

**Solution**

SELECT P.product\_name ,S.year ,S.price

FROM Sales S

LEFT JOIN Product P

ON S.product\_id =P.product\_id

**Copy code**

#### **Question-6: Project Employees**

Write an SQL query that reports the average experience years of all the employees for each project, rounded to 2 digits. Return the result table in any order.

Table:

Project

**Copy code**

+-------------+---------+

| Column Name | Type |

+-------------+---------+

| project\_id | int |

| employee\_id | int |

+-------------+---------+

* (project\_id, employee\_id) is the primary key of this table.
* employee\_id is a foreign key to Employee table.

Each row of this table indicates that the employee with employee\_id is working on the project with project\_id.

Table:

Employee

**Copy code**

+------------------+---------+

| Column Name | Type |

+------------------+---------+

| employee\_id | int |

| name | varchar |

| experience\_years | int |

+------------------+---------+

* employee\_id is the primary key of this table. It’s guaranteed that experience\_years is not NULL.

Each row of this table contains information about one employee.

The query result format is in the following example.

**Example:**

**Input:**

Project table:

+-------------+-------------+

| project\_id | employee\_id |

+-------------+-------------+

| 1 | 1 |

| 1 | 2 |

| 1 | 3 |

| 2 | 1 |

| 2 | 4 |

+-------------+-------------+

Employee table:

+-------------+--------+------------------+

| employee\_id | name | experience\_years |

+-------------+--------+------------------+

| 1 | Khaled | 3 |

| 2 | Ali | 2 |

| 3 | John | 1 |

| 4 | Doe | 2 |

+-------------+--------+------------------+

**Output:**

+-------------+---------------+

| project\_id | average\_years |

+-------------+---------------+

| 1 | 2.00 |

| 2 | 2.50 |

+-------------+---------------+

**Explanation of Example:**

The average experience years for the first project is (3 + 2 + 1) / 3 = 2.00 and for the second project is (3 + 2) / 2 = 2.50

**Solution**

select project\_id,

round(sum(e.experience\_years) / count(e.experience\_years),2) as average\_years

from Project p

join Employee e on p.employee\_id = e.employee\_id

group by project\_id

**Copy code**

Note: You can also use AVG function rather than defining the average function as we have done.

#### **Question-7: Game Play Analysis**

**Write a solution to find the first login date for each player. Return the result table in any order.**

Table:

Activity

**Copy code**

+--------------+---------+

| Column Name | Type |

+--------------+---------+

| player\_id | int |

| device\_id | int |

| event\_date | date |

| games\_played | int |

+--------------+---------+

* (player\_id, event\_date) is the primary key (combination of columns with unique values) of this table.
* This table shows the activity of players in some games.

Each row records a player who logged in and played many games (possibly 0) before logging out on some using some device.

The result format is in the following example.

**Example:**

**Input:**

Activity table:

+-----------+-----------+------------+--------------+

| player\_id | device\_id | event\_date | games\_played |

+-----------+-----------+------------+--------------+

| 1 | 2 | 2016-03-01 | 5 |

| 1 | 2 | 2016-05-02 | 6 |

| 2 | 3 | 2017-06-25 | 1 |

| 3 | 1 | 2016-03-02 | 0 |

| 3 | 4 | 2018-07-03 | 5 |

+-----------+-----------+------------+--------------+

**Output:**

+-----------+-------------+

| player\_id | first\_login |

+-----------+-------------+

| 1 | 2016-03-01 |

| 2 | 2017-06-25 |

| 3 | 2016-03-02 |

+-----------+-------------+

**Solution**

select player\_id, min(event\_date) as first\_login

from activity

group by player\_id

**Copy code**

#### **Ques-8: User Activity Past 30 Days**

**Write a solution to find the daily active user count for a period of 30 days ending 2019-07-27 inclusively. A user was active on someday if they made at least one activity on that day. Return the result table in any order.**

Table:

Activity

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| user\_id | int |

| session\_id | int |

| activity\_date | date |

| activity\_type | enum |

+---------------+---------+

* This table may have duplicate rows.
* The activity\_type column is an ENUM (category) of type (‘open\_session’, ‘end\_session’, ‘scroll\_down’, ‘send\_message’).
* The table shows the user activities for a social media website.
* Note that each session belongs to exactly one user.

The result format is in the following example.

**Example:**

**Input:**

Activity table:

+---------+------------+---------------+---------------+

| user\_id | session\_id | activity\_date | activity\_type |

+---------+------------+---------------+---------------+

| 1 | 1 | 2019-07-20 | open\_session |

| 1 | 1 | 2019-07-20 | scroll\_down |

| 1 | 1 | 2019-07-20 | end\_session |

| 2 | 4 | 2019-07-20 | open\_session |

| 2 | 4 | 2019-07-21 | send\_message |

| 2 | 4 | 2019-07-21 | end\_session |

| 3 | 2 | 2019-07-21 | open\_session |

| 3 | 2 | 2019-07-21 | send\_message |

| 3 | 2 | 2019-07-21 | end\_session |

| 4 | 3 | 2019-06-25 | open\_session |

| 4 | 3 | 2019-06-25 | end\_session |

+---------+------------+---------------+---------------+

**Output:**

+------------+--------------+

| day | active\_users |

+------------+--------------+

| 2019-07-20 | 2 |

| 2019-07-21 | 2 |

+------------+--------------+

**Solution**

SELECT activity\_date AS day, COUNT(DISTINCT user\_id) AS active\_users

FROM Activity

WHERE (activity\_date > "2019-06-27" AND activity\_date <= "2019-07-27")

GROUP BY activity\_date;

**Copy code**

**Explanation of the code**

To solve the given problem, we must remember to count users distinctly, as they can log in multiple times during the day. One more thing, we have to count the users datewise and in between a specific period of date. So, start with

* Select the column to display, i.e., activity\_day
* Define the condition of the dates using the WHERE clause
* Group the result by date, i.e., activity\_date.

#### **Question-9: Article View**

**Write a solution to find all the authors who viewed at least one of their own articles. Return the result table sorted by id in ascending order.**

Table:

Views

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| article\_id | int |

| author\_id | int |

| viewer\_id | int |

| view\_date | date |

+---------------+---------+

* This table has no primary key (column with unique values); the table may have duplicate rows.
* Each row of this table indicates that some viewers viewed an article (written by some author) on some date.
* Note that equal author\_id and viewer\_id indicate the same person.

The result format is in the following example.

**Example:**

**Input:**

Views table:

+------------+-----------+-----------+------------+

| article\_id | author\_id | viewer\_id | view\_date |

+------------+-----------+-----------+------------+

| 1 | 3 | 5 | 2019-08-01 |

| 1 | 3 | 6 | 2019-08-02 |

| 2 | 7 | 7 | 2019-08-01 |

| 2 | 7 | 6 | 2019-08-02 |

| 4 | 7 | 1 | 2019-07-22 |

| 3 | 4 | 4 | 2019-07-21 |

| 3 | 4 | 4 | 2019-07-21 |

+------------+-----------+-----------+------------+

**Output:**

+------+

| id |

+------+

| 4 |

| 7 |

+------+

**Solution**

select distinct author\_id as id from Views

where author\_id = viewer\_id

order by id;

**Copy code**

#### **Question-10: Average Selling Price**

**Write an SQL query to find the average selling price for each product. average\_price should be rounded to 2 decimal places. Return the result table in any order.**

Table:

Prices

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| product\_id | int |

| start\_date | date |

| end\_date | date |

| price | int |

+---------------+---------+

* (product\_id, start\_date, end\_date) is the primary key for this table.
* Each row of this table indicates the price of the product\_id in the period from start\_date to end\_date.
* For each product\_id, there will be no two overlapping periods. That means there will be no two intersecting periods for the same product\_id.

Table:

UnitsSold

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| product\_id | int |

| purchase\_date | date |

| units | int |

+---------------+---------+

* There is no primary key for this table; it may contain duplicates.
* Each row of this table indicates each product sold’s date, units, and product\_id.

The query result format is in the following example.

**Example:**

**Input:**

Prices table:

+------------+------------+------------+--------+

| product\_id | start\_date | end\_date | price |

+------------+------------+------------+--------+

| 1 | 2019-02-17 | 2019-02-28 | 5 |

| 1 | 2019-03-01 | 2019-03-22 | 20 |

| 2 | 2019-02-01 | 2019-02-20 | 15 |

| 2 | 2019-02-21 | 2019-03-31 | 30 |

+------------+------------+------------+--------+

UnitsSold table:

+------------+---------------+-------+

| product\_id | purchase\_date | units |

+------------+---------------+-------+

| 1 | 2019-02-25 | 100 |

| 1 | 2019-03-01 | 15 |

| 2 | 2019-02-10 | 200 |

| 2 | 2019-03-22 | 30 |

+------------+---------------+-------+

**Output:**

+------------+---------------+

| product\_id | average\_price |

+------------+---------------+

| 1 | 6.96 |

| 2 | 16.96 |

+------------+---------------+

**Explanation of the Example**

* Average selling price = Total Price of Product / Number of products sold.
* Average selling price for product 1 = ((100 \* 5) + (15 \* 20)) / 115 = 6.96
* Average selling price for product 2 = ((200 \* 15) + (30 \* 30)) / 230 = 16.96

**Solution**

SELECT

u.product\_id,

ROUND(SUM(p.price \* u.units) / SUM(u.units), 2) AS average\_price

FROM unitssold u

JOIN prices p ON 1=1

AND p.product\_id = u.product\_id

AND u.purchase\_date BETWEEN p.start\_date AND p.end\_date

GROUP BY u.product\_id

**Copy code**

#### **Question-11: Students and Examination**

**Write a solution to find the number of times each student attended each exam. Return the result table ordered by student\_id and subject\_name.**

Table:

Students

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| student\_id | int |

| student\_name | varchar |

+---------------+---------+

* student\_id is the primary key (column with unique values) for this table.
* Each row of this table contains the ID and the name of one student in the school.

Table:

Subjects

**Copy code**

+--------------+---------+

| Column Name | Type |

+--------------+---------+

| subject\_name | varchar |

+--------------+---------+

* subject\_name is the primary key (column with unique values) for this table.
* Each row of this table contains the name of one subject in the school.

Table:

Examinations

**Copy code**

+--------------+---------+

| Column Name | Type |

+--------------+---------+

| student\_id | int |

| subject\_name | varchar |

+--------------+---------+

* This table has no primary key (column with unique values). It may contain duplicates.
* Each student from the Students table takes every course from the Subjects table.
* Each row of this table indicates that a student with ID student\_id attended the exam of subject\_name.

The result format is in the following example.

**Example:**

**Input:**

Students table:

+------------+--------------+

| student\_id | student\_name |

+------------+--------------+

| 1 | Alice |

| 2 | Bob |

| 13 | John |

| 6 | Alex |

+------------+--------------+

Subjects table:

+--------------+

| subject\_name |

+--------------+

| Math |

| Physics |

| Programming |

+--------------+

Examinations table:

+------------+--------------+

| student\_id | subject\_name |

+------------+--------------+

| 1 | Math |

| 1 | Physics |

| 1 | Programming |

| 2 | Programming |

| 1 | Physics |

| 1 | Math |

| 13 | Math |

| 13 | Programming |

| 13 | Physics |

| 2 | Math |

| 1 | Math |

+------------+--------------+

**Output:**

+------------+--------------+--------------+----------------+

| student\_id | student\_name | subject\_name | attended\_exams |

+------------+--------------+--------------+----------------+

| 1 | Alice | Math | 3 |

| 1 | Alice | Physics | 2 |

| 1 | Alice | Programming | 1 |

| 2 | Bob | Math | 1 |

| 2 | Bob | Physics | 0 |

| 2 | Bob | Programming | 1 |

| 6 | Alex | Math | 0 |

| 6 | Alex | Physics | 0 |

| 6 | Alex | Programming | 0 |

| 13 | John | Math | 1 |

| 13 | John | Physics | 1 |

| 13 | John | Programming | 1 |

+------------+--------------+--------------+----------------+

**Explanation of Example**

* The result table should contain all students and all subjects.
* Alice attended the Math exam 3 times, the Physics exam 2 times, and the Programming exam 1 time.
* Bob attended the Math exam 1 time, the Programming exam 1 time, and did not attend the Physics exam.
* Alex did not attend any exams.
* John attended the Math exam 1 time, the Physics exam 1 time, and the Programming exam 1 time.

**Solution**

SELECT s.student\_id, s.student\_name, sub.subject\_name, COALESCE(e.attended\_exams, 0) AS attended\_exams

FROM Students s

CROSS JOIN Subjects sub

LEFT JOIN (

SELECT student\_id, subject\_name, COUNT(\*) AS attended\_exams

FROM Examinations

GROUP BY student\_id, subject\_name

) e USING (student\_id, subject\_name)

ORDER BY s.student\_id, sub.subject\_name;

**Copy code**

#### **Question-12: Replace Employee ID With The Unique Identifier**

**Write a solution to show the unique ID of each user, If a user does not have a unique ID replace just show null. Return the result table in any order.**

Table:

Employees

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| id | int |

| name | varchar |

+---------------+---------+

* id is the primary key (column with unique values) for this table.
* Each row of this table contains the id and the name of an employee in a company.

Table:

EmployeeUNI

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| id | int |

| unique\_id | int |

+---------------+---------+

* (id, unique\_id) is the primary key (combination of columns with unique values) for this table.
* Each row of this table contains the id and the corresponding unique id of an employee in the company.

The result format is in the following example.

**Example:**

**Input:**

Employees table:

+----+----------+

| id | name |

+----+----------+

| 1 | Alice |

| 7 | Bob |

| 11 | Meir |

| 90 | Winston |

| 3 | Jonathan |

+----+----------+

EmployeeUNI table:

+----+-----------+

| id | unique\_id |

+----+-----------+

| 3 | 1 |

| 11 | 2 |

| 90 | 3 |

+----+-----------+

**Output:**

+-----------+----------+

| unique\_id | name |

+-----------+----------+

| null | Alice |

| null | Bob |

| 2 | Meir |

| 3 | Winston |

| 1 | Jonathan |

+-----------+----------+

**Explanation of Example:**

* Alice and Bob do not have a unique ID, We will show null instead.
* The unique ID of Meir is 2.
* The unique ID of Winston is 3.
* The unique ID of Jonathan is 1.

**Solution**

select

eu.unique\_id as unique\_id, e.name as name

from Employees e left join EmployeeUNI eu on e.id = eu.id

**Copy code**

#### **Question-13: Top Travellers**

**Write a solution to report the distance traveled by each user. Return the result table ordered by travelled\_distance in descending order, if two or more users traveled the same distance, order them by their name in ascending order.**

Table:

Users

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| id | int |

| name | varchar |

+---------------+---------+

* id is the column with unique values for this table.
* name is the name of the user.

Table:

Rides

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| id | int |

| user\_id | int |

| distance | int |

+---------------+---------+

* id is the column with unique values for this table.
* user\_id is the id of the user who travelled the distance “distance”.

The result format is in the following example.

**Example:**

**Input:**

Users table:

+------+-----------+

| id | name |

+------+-----------+

| 1 | Alice |

| 2 | Bob |

| 3 | Alex |

| 4 | Donald |

| 7 | Lee |

| 13 | Jonathan |

| 19 | Elvis |

+------+-----------+

Rides table:

+------+----------+----------+

| id | user\_id | distance |

+------+----------+----------+

| 1 | 1 | 120 |

| 2 | 2 | 317 |

| 3 | 3 | 222 |

| 4 | 7 | 100 |

| 5 | 13 | 312 |

| 6 | 19 | 50 |

| 7 | 7 | 120 |

| 8 | 19 | 400 |

| 9 | 7 | 230 |

+------+----------+----------+

**Output:**

+----------+--------------------+

| name | travelled\_distance |

+----------+--------------------+

| Elvis | 450 |

| Lee | 450 |

| Bob | 317 |

| Jonathan | 312 |

| Alex | 222 |

| Alice | 120 |

| Donald | 0 |

+----------+--------------------+

**Explanation to Example:**

* Elvis and Lee travelled 450 miles; Elvis is the top traveller as his name is alphabetically smaller than Lee’s.
* Bob, Jonathan, Alex, and Alice have only one ride, and we just order them by the total distance of the ride.
* Donald did not have any rides; the distance travelled by him is 0.

**Solution**

SELECT DISTINCT u.name,

IFNULL(SUM(distance) OVER (PARTITION BY user\_id), 0) as travelled\_distance

FROM Rides r

RIGHT JOIN Users u

ON r.user\_id = u.id

ORDER BY travelled\_distance DESC, name

**Copy code**

#### **Question-14: Group Sold Product By Date**

**Write a solution to find for each date the number of different products sold and their names. The sold product names for each date should be sorted lexicographically. Return the result table ordered by sell\_date.**

Table

Activities

**Copy code**

:

+-------------+---------+

| Column Name | Type |

+-------------+---------+

| sell\_date | date |

| product | varchar |

+-------------+---------+

* This table has no primary key (column with unique values). It may contain duplicates.
* Each row of this table contains the product name and the date it was sold in a market.

The result format is in the following example.

**Example:**

**Input:**

Activities table:

+------------+------------+

| sell\_date | product |

+------------+------------+

| 2020-05-30 | Headphone |

| 2020-06-01 | Pencil |

| 2020-06-02 | Mask |

| 2020-05-30 | Basketball |

| 2020-06-01 | Bible |

| 2020-06-02 | Mask |

| 2020-05-30 | T-Shirt |

+------------+------------+

**Output:**

+------------+----------+------------------------------+

| sell\_date | num\_sold | products |

+------------+----------+------------------------------+

| 2020-05-30 | 3 | Basketball,Headphone,T-shirt |

| 2020-06-01 | 2 | Bible,Pencil |

| 2020-06-02 | 1 | Mask |

+------------+----------+------------------------------+

**Explanation to Example:**

* For 2020-05-30, Sold items were (headphones, Basketballs, and T-shirts); we sorted them lexicographically and separated them by a comma.
* For 2020-06-01, the Sold items were (Pencil, Bible); we sorted them lexicographically and separated them by a comma.
* For 2020-06-02, the Sold item is (Mask), we just return it.

**Solution**

SELECT sell\_date,

COUNT(DISTINCT(product)) AS num\_sold,

GROUP\_CONCAT(DISTINCT product ORDER BY product ASC SEPARATOR ',') AS products

FROM Activities

GROUP BY sell\_date

ORDER BY sell\_date ASC

**Copy code**

**Explanation**

Here, in the above problem, the only thing was, how to aggregate the product name in one cell. In MySQL, this can be done by **GROUP\_CONCAT** clause, in which you can also specify the sorting mechanism for group concatenation.

#### **Question-15: Find users with valid E-mail**

Write a solution to find the users who have **valid emails**.

A valid e-mail has a prefix name and a domain where:

* **The prefix name** is a string that may contain letters (upper or lower case), digits, underscore ‘\_’, period ‘.’, and/or dash ‘-‘. The prefix name **must** start with a letter.
* **The domain** is ‘@leetcode.com’.

Return the result table in **any order**.

Table:

Users

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| user\_id | int |

| name | varchar |

| mail | varchar |

+---------------+---------+

* user\_id is the primary key (column with unique values) for this table.
* This table contains information of the users signed up in a website. Some e-mails are invalid.

The result format is in the following example.

**Example:**

**Input:**

Users table:

+---------+-----------+-------------------------+

| user\_id | name | mail |

+---------+-----------+-------------------------+

| 1 | Winston | winston@leetcode.com |

| 2 | Jonathan | jonathanisgreat |

| 3 | Annabelle | bella-@leetcode.com |

| 4 | Sally | sally.come@leetcode.com |

| 5 | Marwan | quarz#2020@leetcode.com |

| 6 | David | david69@gmail.com |

| 7 | Shapiro | .shapo@leetcode.com |

+---------+-----------+-------------------------+

**Output:**

+---------+-----------+-------------------------+

| user\_id | name | mail |

+---------+-----------+-------------------------+

| 1 | Winston | winston@leetcode.com |

| 3 | Annabelle | bella-@leetcode.com |

| 4 | Sally | sally.come@leetcode.com |

+---------+-----------+-------------------------+

**Explanation to Example:**

* The mail of user 2 does not have a domain.
* The mail of user 5 has the # sign, which is not allowed.
* The mail of user 6 does not have the leetcode domain.
* The mail of user 7 starts with a period.

**Solution**

SELECT \*

FROM Users

WHERE mail REGEXP '^[a-zA-Z][a-zA-Z0-9\_.-]\*@leetcode[.]com$';

**Copy code**

**Explanation**

The regular expression may be broken down into chunks and explained as follows (refer to the documentation immediately below if needed):

* ^[a-zA-Z]: The email must start with an alphanumeric letter (i.e., not digit).
* [a-zA-Z0-9\_.-]\*:
  + The next zero or more characters must be either alphanumeric (letters a-z, A-Z, or digits 0-9)
  + or a \_ or .
  + or a –

@leetcode[.]com$: + @leetcode: The next sequence of characters must exactly match @leetcode + [.]: The . character within a [] pair does not have special meaning so it must be matched exactly after @leetcode. + com$: The string must end with com.

### ****Medium****

#### **Question-1: Manager with at least 5 Direct Reports**

**Write a solution to find managers with at least five direct reports. Return the result table in any order.**

Table:

Employee

**Copy code**

+-------------+---------+

| Column Name | Type |

+-------------+---------+

| id | int |

| name | varchar |

| department | varchar |

| managerId | int |

+-------------+---------+

* id is the primary key (column with unique values) for this table.
* Each row of this table indicates the name of an employee, their department, and the id of their manager.
* If managerId is null, then the employee does not have a manager.
* No employee will be the manager of themself.

The result format is in the following example.

**Example:**

**Input:**

Employee table:

+-----+-------+------------+-----------+

| id | name | department | managerId |

+-----+-------+------------+-----------+

| 101 | John | A | None |

| 102 | Dan | A | 101 |

| 103 | James | A | 101 |

| 104 | Amy | A | 101 |

| 105 | Anne | A | 101 |

| 106 | Ron | B | 101 |

+-----+-------+------------+-----------+

**Output:**

+------+

| name |

+------+

| John |

+------+

**Solution**

Select m.name

from employee as e

inner join employee as m

on e.managerId=m.id

group by e.managerId

having count(e.id)>=5

**Copy code**

#### **Question-2: Tree Node**

**Each node in the tree can be one of three types:**

* **“Leaf”**: if the node is a leaf node.
* **“Root”**: if the node is the root of the tree.
* **“Inner”**: If the node is neither a leaf node nor a root node.

**Write a solution to report the type of each node in the tree.**

Table:

Tree

**Copy code**

+-------------+------+

| Column Name | Type |

+-------------+------+

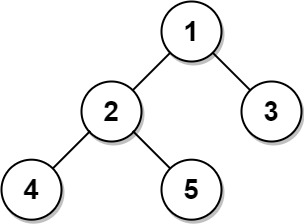
| id | int |

| p\_id | int |

+-------------+------+

* id is the column with unique values for this table.
* Each row of this table contains information about a node’s id and its parent node’s id in a tree.
* The given structure is always a valid tree.

The result format is in the following example.



**Input:**

Tree table:

+----+------+

| id | p\_id |

+----+------+

| 1 | null |

| 2 | 1 |

| 3 | 1 |

| 4 | 2 |

| 5 | 2 |

+----+------+

**Output:**

+----+-------+

| id | type |

+----+-------+

| 1 | Root |

| 2 | Inner |

| 3 | Leaf |

| 4 | Leaf |

| 5 | Leaf |

+----+-------+

**Explanation to Example:**

* Node 1 is the root node because its parent node is null, and it has child nodes 2 and 3.
* Node 2 is an inner node because it has parent node 1 and child nodes 4 and 5.
* Nodes 3, 4, and 5 are leaf nodes because they have parent nodes, and they do not have child nodes.

**Solution**

SELECT DISTINCT t1.id, (

CASE

WHEN t1.p\_id IS NULL THEN 'Root'

WHEN t1.p\_id IS NOT NULL AND t2.id IS NOT NULL THEN 'Inner'

WHEN t1.p\_id IS NOT NULL AND t2.id IS NULL THEN 'Leaf'

END

) AS Type

FROM tree t1

LEFT JOIN tree t2

ON t1.id = t2.p\_id

**Copy code**

#### **Question-3: Customer Who Bought All Product**

**Write a solution to report the customer ids from the Customer table that bought all the products in the Product table. Return the result table in any order.**

Table:

Customer

**Copy code**

+-------------+---------+

| Column Name | Type |

+-------------+---------+

| customer\_id | int |

| product\_key | int |

+-------------+---------+

* This table may contain duplicates rows.
* customer\_id is not NULL.
* product\_key is a foreign key (reference column) to Product table.

Table:

Product

**Copy code**

+-------------+---------+

| Column Name | Type |

+-------------+---------+

| product\_key | int |

+-------------+---------+

* product\_key is the primary key (column with unique values) for this table.

The result format is in the following example.

**Example:**

**Input:**

Customer table:

+-------------+-------------+

| customer\_id | product\_key |

+-------------+-------------+

| 1 | 5 |

| 2 | 6 |

| 3 | 5 |

| 3 | 6 |

| 1 | 6 |

+-------------+-------------+

Product table:

+-------------+

| product\_key |

+-------------+

| 5 |

| 6 |

+-------------+

**Output:**

+-------------+

| customer\_id |

+-------------+

| 1 |

| 3 |

+-------------+

The customers who bought all the products (5 and 6) are customers with IDs 1 and 3.

**Solution**

select customer\_id

from customer c

group by customer\_id

having count(distinct product\_key)=(select count(distinct product\_key) from product)

**Copy code**

#### **Question-4: Game Play Analysis**

**Write a solution to report the fraction of players that logged in again on the day after the day they first logged in, rounded to 2 decimal places. In other words, you need to count the number of players that logged in for at least two consecutive days starting from their first login date, then divide that number by the total number of players.**

Table:

Activity

**Copy code**

+--------------+---------+

| Column Name | Type |

+--------------+---------+

| player\_id | int |

| device\_id | int |

| event\_date | date |

| games\_played | int |

+--------------+---------+

* (player\_id, event\_date) is the primary key (combination of columns with unique values) of this table.
* This table shows the activity of players of some games.
* Each row is a record of a player who logged in and played a number of games (possibly 0) before logging out on someday using some device.

The result format is in the following example.

**Example:**

**Input:**

Activity table:

+-----------+-----------+------------+--------------+

| player\_id | device\_id | event\_date | games\_played |

+-----------+-----------+------------+--------------+

| 1 | 2 | 2016-03-01 | 5 |

| 1 | 2 | 2016-03-02 | 6 |

| 2 | 3 | 2017-06-25 | 1 |

| 3 | 1 | 2016-03-02 | 0 |

| 3 | 4 | 2018-07-03 | 5 |

+-----------+-----------+------------+--------------+

**Output:**

+-----------+

| fraction |

+-----------+

| 0.33 |

+-----------+

**Explanation to Example**

Only the player with id 1 logged back in after the first day he had logged in so the answer is 1/3 = 0.33

**Solution**

SELECT

ROUND(COUNT(DISTINCT player\_id) / (SELECT COUNT(DISTINCT player\_id) FROM Activity), 2) AS fraction

FROM

Activity

WHERE

(player\_id, DATE\_SUB(event\_date, INTERVAL 1 DAY))

IN (

SELECT player\_id, MIN(event\_date) AS first\_login FROM Activity GROUP BY player\_id

)

**Copy code**

#### **Explanation of SQL query for Game Play Analysis**

Here, we have to report the fraction of players that logged in again on the day after the day they first logged in (rounding to two decimal places).

So, we have to calculate:

* The number of players who logged in on consecutive days
  + To calculate this, we need to find the first login date for each player and check if there is a login on the day after their first login.
    - Write a sub-query to calculate the total number of distinct players using the activity table (this will give you the denominator for the fraction)
    - In the main query, we use the WHERE clause to filter the rows where the player’s ID and the date of the event match the player’s first login date. (This will give you the players who logged in on consecutive days.)
    - Count the distinct player ID in the filtered row (this will give you the numerator for calculating the fraction).
    - Calculate the value of the fraction (numerator/denominator) and use the ROUND function to round the result to 2 decimal places.

#### **Question-5: Product Price At a Given Date**

**Write an SQL query to find the prices of all products on 2019-08-16. Assume the price of all products before any change is 10. Return the result table in any order**

Table:

Products

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| product\_id | int |

| new\_price | int |

| change\_date | date |

+---------------+---------+

* (product\_id, change\_date) is the primary key of this table.
* Each row of this table indicates that the price of some product was changed to a new price at some date.

The query result format is in the following example.

**Example:**

**Input:**

Products table:

+------------+-----------+-------------+

| product\_id | new\_price | change\_date |

+------------+-----------+-------------+

| 1 | 20 | 2019-08-14 |

| 2 | 50 | 2019-08-14 |

| 1 | 30 | 2019-08-15 |

| 1 | 35 | 2019-08-16 |

| 2 | 65 | 2019-08-17 |

| 3 | 20 | 2019-08-18 |

+------------+-----------+-------------+

**Output:**

+------------+-------+

| product\_id | price |

+------------+-------+

| 2 | 50 |

| 1 | 35 |

| 3 | 10 |

+------------+-------+

**Solution**

select distinct product\_id, 10 as price

from Products

group by product\_id

having (min(change\_date) > "2019-08-16")

union

select p2.product\_id, new\_price

from Products p2

where (p2.product\_id, p2.change\_date) in (

select product\_id, max(change\_date) as recent\_date

from Products

where change\_date <= "2019-08-16"

group by product\_id

)

**Copy code**

Note: Inspite of using UNION clause you can also use LEFT JOIN with a subquery to get the same above result.

SELECT T1.product\_id, IFNULL(T2.new\_price,10) AS price

FROM (SELECT DISTINCT product\_id FROM Products) AS T1

LEFT JOIN (SELECT product\_id, new\_price

FROM Products

WHERE (product\_id, change\_date) IN (

SELECT product\_id, MAX(change\_date) AS last\_date

FROM Products

WHERE change\_date <= '2019-08-16'

GROUP BY product\_id)) AS T2

ON T1.product\_id = T2.product\_id;

**Copy code**

#### **Question-6: Monthly Transaction**

**Write an SQL query to find for each month and country, the number of transactions and their total amount, the number of approved transactions and their total amount. Return the result table in any order.**

The query result format is in the following example.

**Example:**

**Input:**

Transactions table:

+------+---------+----------+--------+------------+

| id | country | state | amount | trans\_date |

+------+---------+----------+--------+------------+

| 121 | US | approved | 1000 | 2018-12-18 |

| 122 | US | declined | 2000 | 2018-12-19 |

| 123 | US | approved | 2000 | 2019-01-01 |

| 124 | DE | approved | 2000 | 2019-01-07 |

+------+---------+----------+--------+------------+

**Output:**

+----------+---------+-------------+----------------+--------------------+-----------------------+

| month | country | trans\_count | approved\_count | trans\_total\_amount | approved\_total\_amount |

+----------+---------+-------------+----------------+--------------------+-----------------------+

| 2018-12 | US | 2 | 1 | 3000 | 1000 |

| 2019-01 | US | 1 | 1 | 2000 | 2000 |

| 2019-01 | DE | 1 | 1 | 2000 | 2000 |

+----------+---------+-------------+----------------+--------------------+-----------------------+

**Solution**

SELECT

DATE\_FORMAT(trans\_date, '%Y-%m') AS month,

country,

COUNT(\*) AS trans\_count,

SUM(IF(state = 'approved', 1, 0)) AS approved\_count,

SUM(amount) AS trans\_total\_amount,

SUM(IF(state = 'approved', amount, 0)) AS approved\_total\_amount

FROM Transactions

GROUP BY DATE\_FORMAT(trans\_date, '%Y-%m'), country;

**Copy code**

**Expalnation**

Here, we have to calculate the number of transactions and their total amount, the number of approved transactions, and their total amount for each month and country.  
To do that we have used aggregate functions like COUNT and SUM and at the end, we have grouped the transactions by month and country to get the desired result.

Here we have used the DATE\_FORMAT function of SQL to extract the month from the trans\_date column and assign it an alias Month.

#### **Question-7: Last Person to fit in Bus**

There is a queue of people waiting to board a bus. However, the bus has a weight limit of

1000

**Copy code**

**kilograms**, so there may be some people who cannot board.

Write a solution to find the person\_name of the **last person** that can fit on the bus without exceeding the weight limit. The test cases are generated such that the first person does not exceed the weight limit.

Table:

Queue

**Copy code**

+-------------+---------+

| Column Name | Type |

+-------------+---------+

| person\_id | int |

| person\_name | varchar |

| weight | int |

| turn | int |

+-------------+---------+

* person\_id column contains unique values.
* This table has information about all people waiting for a bus.
* The person\_id and turn columns will contain all numbers from 1 to n, where n is the number of rows in the table.
* turn determines the order of which the people will board the bus, where turn=1 denotes the first person to board and turn=n denotes the last person to board.
* weight is the weight of the person in kilograms.

The result format is in the following example.

**Example:**

**Input:**

Queue table:

+-----------+-------------+--------+------+

| person\_id | person\_name | weight | turn |

+-----------+-------------+--------+------+

| 5 | Alice | 250 | 1 |

| 4 | Bob | 175 | 5 |

| 3 | Alex | 350 | 2 |

| 6 | John Cena | 400 | 3 |

| 1 | Winston | 500 | 6 |

| 2 | Marie | 200 | 4 |

+-----------+-------------+--------+------+

**Output:**

+-------------+

| person\_name |

+-------------+

| John Cena |

+-------------+

**Explanation to Example**

The following table is ordered by the turn for simplicity.

+------+----+-----------+--------+--------------+

| Turn | ID | Name | Weight | Total Weight |

+------+----+-----------+--------+--------------+

| 1 | 5 | Alice | 250 | 250 |

| 2 | 3 | Alex | 350 | 600 |

| 3 | 6 | John Cena | 400 | 1000 | (last person to board)

| 4 | 2 | Marie | 200 | 1200 | (cannot board)

| 5 | 4 | Bob | 175 | \_\_\_ |

| 6 | 1 | Winston | 500 | \_\_\_ |

+------+----+-----------+--------+--------------+

**Solution**

select person\_name from

(select person\_name, weight, turn, sum(weight) over(order by turn) as cum\_sum

from queue) x

where cum\_sum <= 1000

order by turn desc limit 1;

**Copy code**

#### **Question-8: Restaurant Growth**

**You are the restaurant owner and want to analyze a possible expansion (there will be at least one customer daily). Compute the moving average of how much the customer paid in a seven-day window (i.e., current day + 6 days before). average\_amount should be rounded to two decimal places.**

**Return the result table ordered by visited\_on in ascending order.**

Table:

Customer

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| customer\_id | int |

| name | varchar |

| visited\_on | date |

| amount | int |

+---------------+---------+

* In SQL,(customer\_id, visited\_on) is the primary key for this table.
* This table contains data about customer transactions in a restaurant.
* visited\_on is when the customer with ID (customer\_id) has visited the restaurant.
* amount is the total paid by a customer

The result format is in the following example.

**Example:**

**Input:**

Customer table:

+-------------+--------------+--------------+-------------+

| customer\_id | name | visited\_on | amount |

+-------------+--------------+--------------+-------------+

| 1 | Jhon | 2019-01-01 | 100 |

| 2 | Daniel | 2019-01-02 | 110 |

| 3 | Jade | 2019-01-03 | 120 |

| 4 | Khaled | 2019-01-04 | 130 |

| 5 | Winston | 2019-01-05 | 110 |

| 6 | Elvis | 2019-01-06 | 140 |

| 7 | Anna | 2019-01-07 | 150 |

| 8 | Maria | 2019-01-08 | 80 |

| 9 | Jaze | 2019-01-09 | 110 |

| 1 | Jhon | 2019-01-10 | 130 |

| 3 | Jade | 2019-01-10 | 150 |

+-------------+--------------+--------------+-------------+

**Output:**

+--------------+--------------+----------------+

| visited\_on | amount | average\_amount |

+--------------+--------------+----------------+

| 2019-01-07 | 860 | 122.86 |

| 2019-01-08 | 840 | 120 |

| 2019-01-09 | 840 | 120 |

| 2019-01-10 | 1000 | 142.86 |

+--------------+--------------+----------------+

**Explanation to Example**

* 1st moving average from 2019-01-01 to 2019-01-07 has an average\_amount of (100 + 110 + 120 + 130 + 110 + 140 + 150)/7 = 122.86
* 2nd moving average from 2019-01-02 to 2019-01-08 has an average\_amount of (110 + 120 + 130 + 110 + 140 + 150 + 80)/7 = 120
* 3rd moving average from 2019-01-03 to 2019-01-09 has an average\_amount of (120 + 130 + 110 + 140 + 150 + 80 + 110)/7 = 120
* 4th moving average from 2019-01-04 to 2019-01-10 has an average\_amount of (130 + 110 + 140 + 150 + 80 + 110 + 130 + 150)/7 = 142.86

**Solution**

*# Write your MySQL query statement below*

SELECT

visited\_on,

(

SELECT SUM(amount)

FROM customer

WHERE visited\_on BETWEEN DATE\_SUB(c.visited\_on, INTERVAL 6 DAY) AND c.visited\_on

) AS amount,

ROUND(

(

SELECT SUM(amount) / 7

FROM customer

WHERE visited\_on BETWEEN DATE\_SUB(c.visited\_on, INTERVAL 6 DAY) AND c.visited\_on

),

2

) AS average\_amount

FROM customer c

WHERE visited\_on >= (

SELECT DATE\_ADD(MIN(visited\_on), INTERVAL 6 DAY)

FROM customer

)

GROUP BY visited\_on;

**Copy code**

**Explanation to Solution**

Here, we have to calculate the amount and average amount of how much the customer paid in a seven-day window.

To do that, select the visited date from the customer table.

* Create a subquery that sums the amount for each visited\_on date by considering the previous 7 days.
* Use the WHERE clause on visited\_on to filter the customer BETWEEN the given date.
* Similarly, write another subquery to calculate the average amount.
* To find the average of 7 days, divide the sum(amount)/7.
* Use the WHERE clause to filter the record where there will be at least one customer.
* Finally, group the result by the column visited\_on.

#### **Question-9: Movie Rating**

**Write a solution to:**

* **Find the user’s name who has rated the greatest number of movies. In case of a tie, return the lexicographically smaller user name.**
* **Find the movie name with the highest average rating in February 2020. In case of a tie, return the lexicographically smaller movie name.**

Table:

Movies

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| movie\_id | int |

| title | varchar |

+---------------+---------+

* movie\_id is the primary key (column with unique values) for this table.
* title is the name of the movie.

Table:

Users

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| user\_id | int |

| name | varchar |

+---------------+---------+

* user\_id is the primary key (column with unique values) for this table

Table:

MovieRating

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| movie\_id | int |

| user\_id | int |

| rating | int |

| created\_at | date |

+---------------+---------+

* (movie\_id, user\_id) is the primary key (column with unique values) for this table.
* This table contains the rating of a movie by a user in their review.
* created\_at is the user’s review date.

The result format is in the following example.

**Example:**

**Input:**

Movies table:

+-------------+--------------+

| movie\_id | title |

+-------------+--------------+

| 1 | Avengers |

| 2 | Frozen 2 |

| 3 | Joker |

+-------------+--------------+

Users table:

+-------------+--------------+

| user\_id | name |

+-------------+--------------+

| 1 | Daniel |

| 2 | Monica |

| 3 | Maria |

| 4 | James |

+-------------+--------------+

MovieRating table:

+-------------+--------------+--------------+-------------+

| movie\_id | user\_id | rating | created\_at |

+-------------+--------------+--------------+-------------+

| 1 | 1 | 3 | 2020-01-12 |

| 1 | 2 | 4 | 2020-02-11 |

| 1 | 3 | 2 | 2020-02-12 |

| 1 | 4 | 1 | 2020-01-01 |

| 2 | 1 | 5 | 2020-02-17 |

| 2 | 2 | 2 | 2020-02-01 |

| 2 | 3 | 2 | 2020-03-01 |

| 3 | 1 | 3 | 2020-02-22 |

| 3 | 2 | 4 | 2020-02-25 |

+-------------+--------------+--------------+-------------+

**Output:**

+--------------+

| results |

+--------------+

| Daniel |

| Frozen 2 |

+--------------+

**Explanation to Example**

* Daniel and Monica have rated 3 movies (“Avengers”, “Frozen 2” and “Joker”) but Daniel is smaller lexicographically.
* Frozen 2 and Joker have a rating average of 3.5 in February but Frozen 2 is smaller lexicographically.

**Solution**

select name as results

from (

select user\_id, name, count(rating) rating\_num

from Movie\_Rating left join Users using(user\_id)

group by user\_id, name

order by rating\_num desc, name asc limit 1) a

union

select title as results

from (

select movie\_id, title, avg(rating) avg\_rating

from Movie\_Rating left join Movies using(movie\_id)

where created\_at like '2020-02-%'

group by movie\_id, title

order by avg\_rating desc, title asc limit 1) b

**Copy code**

#### **Question-10: Capital Gain/Loss**

**Write a solution to report the Capital gain/loss for each stock. The Capital gain/loss of a stock is the total gain or loss after buying and selling the stock one or many times.**

**Return the result table in any order.**

able:

Stocks

**Copy code**

+---------------+---------+

| Column Name | Type |

+---------------+---------+

| stock\_name | varchar |

| operation | enum |

| operation\_day | int |

| price | int |

+---------------+---------+

* (stock\_name, operation\_day) is the primary key (combination of columns with unique values) for this table.
* The operation column is an ENUM (category) of type (‘Sell’, ‘Buy’)
* Each row of this table indicates that the stock with stock\_name had an operation on the day operation\_day with the price.
* Each ‘Sell’ operation for a stock is guaranteed to have a corresponding ‘Buy’ operation in the previous day, and ‘Buy’ operation for a stock is guaranteed to have a corresponding ‘Sell’ operation on an upcoming day.

The result format is in the following example.

**Example:**

**Input:**

Stocks table:

+---------------+-----------+---------------+--------+

| stock\_name | operation | operation\_day | price |

+---------------+-----------+---------------+--------+

| Leetcode | Buy | 1 | 1000 |

| Corona Masks | Buy | 2 | 10 |

| Leetcode | Sell | 5 | 9000 |

| Handbags | Buy | 17 | 30000 |

| Corona Masks | Sell | 3 | 1010 |

| Corona Masks | Buy | 4 | 1000 |

| Corona Masks | Sell | 5 | 500 |

| Corona Masks | Buy | 6 | 1000 |

| Handbags | Sell | 29 | 7000 |

| Corona Masks | Sell | 10 | 10000 |

+---------------+-----------+---------------+--------+

**Output:**

+---------------+-------------------+

| stock\_name | capital\_gain\_loss |

+---------------+-------------------+

| Corona Masks | 9500 |

| Leetcode | 8000 |

| Handbags | -23000 |

+---------------+-------------------+

**Solution**

SELECT stock\_name, SUM(

CASE

WHEN operation = 'Buy' THEN -price

ELSE price

END

) AS capital\_gain\_loss

FROM Stocks

GROUP BY stock\_name

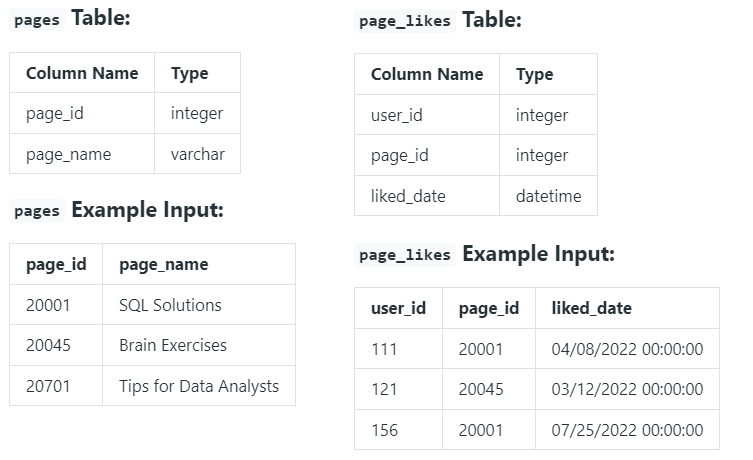
**Copy code**

## ****FAANG QUESTION FOR PRACTICE****

**EASY**

#### **1. Facebook – Pages with No Likes**

**Assume you’re given two tables containing data about Facebook Pages and their respective likes (as in “Like a Facebook Page”).** **Write a query to return the IDs of the Facebook pages with zero likes. The output should be sorted in ascending order based on the page IDs.**



**Solution**

SELECT p1.page\_id

FROM pages p1

LEFT JOIN page\_likes p2

ON p1.page\_id = p2.page\_id

WHERE p2.page\_id IS NULL

ORDER BY p2.page\_id

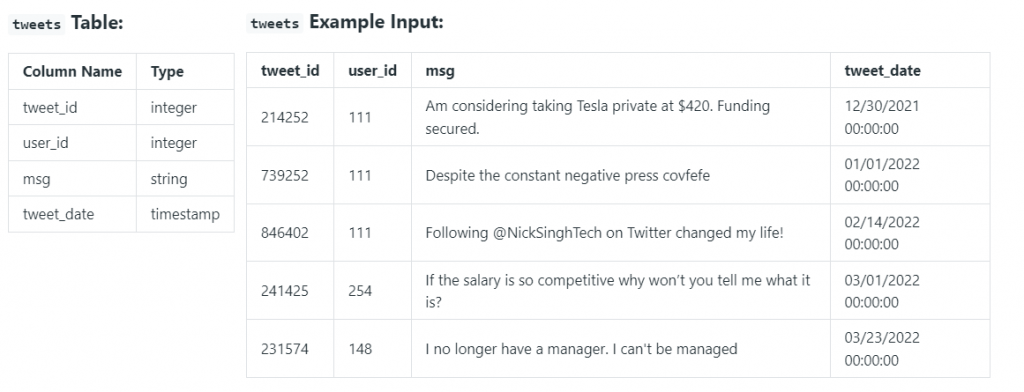
**Copy code**

**Hint**

* Use LEFT JOIN between pages and page\_likes tables.
* Check the NULL values.

#### **2. Twitter – Histogram of Tweets**

**Assume you’re given a table of Twitter tweet data and write a query to obtain a histogram of tweets posted per user in 2022. Output the tweet counts per user as the bucket and the number of Twitter users who fall into that bucket. In other words, group the users by the number of tweets they posted in 2022 and count the number of users in each group.**



**Solution**

SELECT tweet\_bucket, count(tweet\_bucket) AS users\_num

FROM(

SELECT COUNT(user\_id) AS tweet\_bucket

FROM tweets

WHERE DATE\_PART('year', tweet\_date)=2022

GROUP BY user\_id) as twt

GROUP BY tweet\_bucket

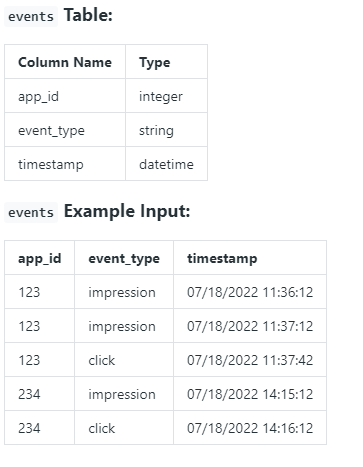
**Copy code**

#### **3. Facebook – App Click-through Rate (CTR)**

**Assume you have an events table on Facebook app analytics. Write a query to calculate the click-through rate (CTR) for the app in 2022 and round the results to 2 decimal places.**

Definition and note:

* Percentage of click-through rate (CTR) = 100.0 \* Number of clicks / Number of impressions
* To avoid integer division, multiply the CTR by 100.0, not 100.



**Solution**

SELECT

app\_id,

ROUND(100.0 \*

sum(case WHEN event\_type = 'click' then 1 ELSE 0 END) /

sum(case WHEN event\_type = 'impression' then 1 ELSE 0 END),2) as ctr

FROM events

WHERE timestamp BETWEEN '2022-01-01' AND '2022-12-31'

GROUP BY app\_id;;

**Copy code**

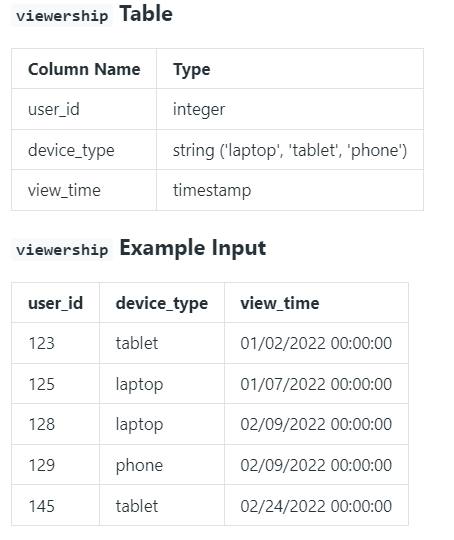
**Hint**

Use CASE, WHEN, and THEN to calculate the impression count and click count. You can use:

* sum(case WHEN event\_type = ‘impression’ then 1 ELSE 0 END) as impression\_count,
* sum(case WHEN event\_type = ‘click’ then 1 ELSE 0 END) as click\_count,

#### **4. New York Times – Laptop vs Mobile Viewership**

**Assume you’re given the table on user viewership categorized by device type, where the three types are laptop, tablet, and phone. Write a query that calculates the total viewership for laptops and mobile devices, where mobile is defined as the sum of tablet and phone viewership. Output the total viewership for laptops as**laptop\_reviews**and the total viewership for mobile devices as**mobile\_views**.**



**Solution**

SELECT

SUM(CASE WHEN device\_type='laptop' THEN 1 ELSE 0 END) AS laptop\_reviews,

SUM(CASE WHEN device\_type IN ('phone','tablet') THEN 1 ELSE 0 END) AS mobile\_reviews

FROM viewership

**Copy code**

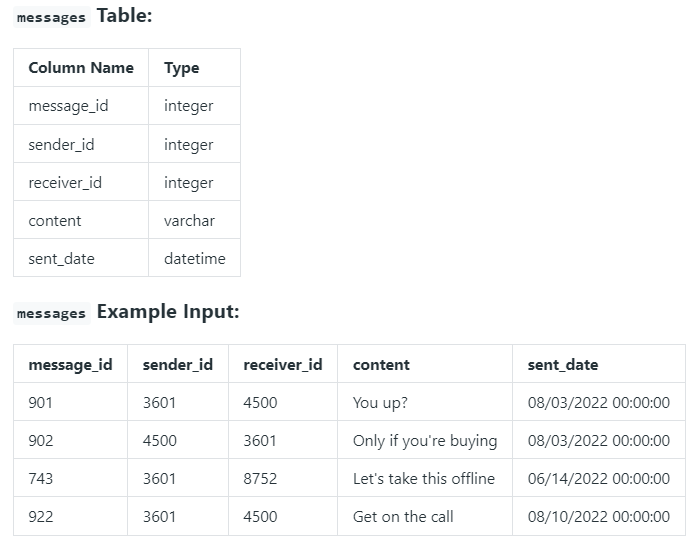
**Hint**

Use CASE, WHEN, and THEN as done in the previous question.

#### **5. Microsoft – Teams Power Users**

**Write a query to identify the top 2 Power Users who sent the most messages on Microsoft Teams in August 2022. Display these 2 users’ IDs and the total number of messages they sent. Output the results in descending order based on the count of the messages.**

Assumption: No two users have sent the same number of messages in August 2022.



**Solution**

SELECT sender\_id, count(message\_id) AS message\_count

FROM messages

where sent\_date >= '08/01/2022' and sent\_date < '09/01/2022'

group by sender\_id

order by message\_count DESC

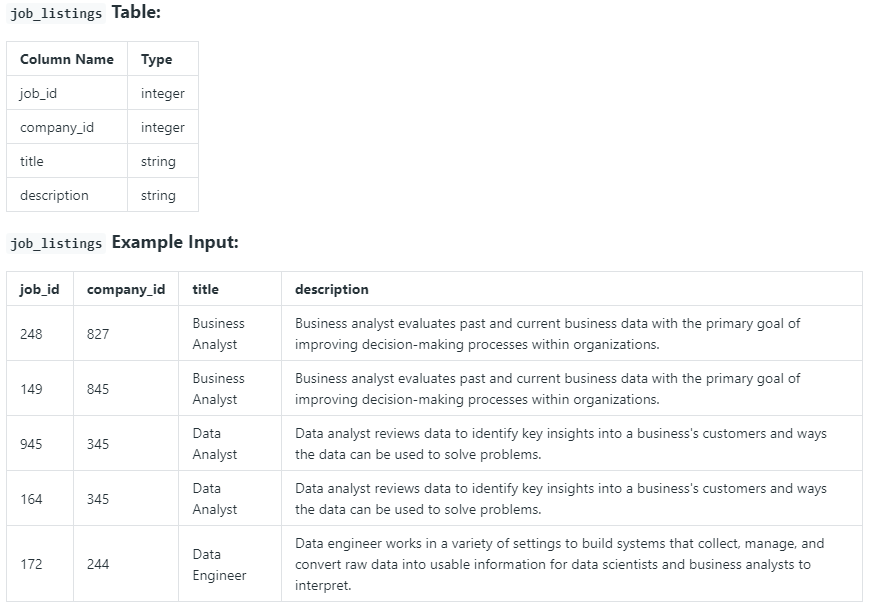
LIMIT 2;

**Copy code**

#### **6. Linkedin – Duplicate Job Listings**

**Assume you’re given a table containing job postings from various companies on the LinkedIn platform. Write a query to retrieve the count of companies that have posted duplicate job listings.**

Definition: Duplicate job listings are defined as two job listings within the same company that share identical titles and descriptions.



**Solution**

SELECT

COUNT(DISTINCT company\_id) AS duplicate\_companies

FROM (

SELECT

company\_id,

COUNT(job\_id) AS job\_count

FROM job\_listings

GROUP BY company\_id, title, description

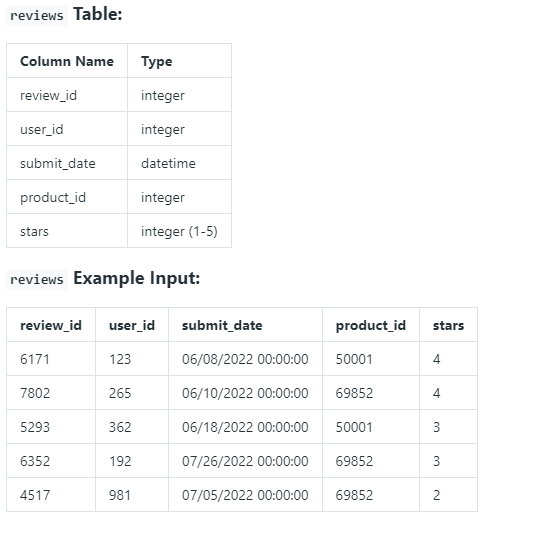
) AS duplicate\_count

WHERE job\_count > 1

**Copy code**

#### **7. Amazon – Average Review Rating**

**Given the reviews table, write a query to retrieve the average star rating for each product, grouped by month. The output should display the month as a numerical value, product ID, and average star rating rounded to two decimals. Sort the output first by month and then by product ID.**



**Solution**

SELECT EXTRACT(MONTH FROM submit\_date) AS month,

product\_id AS product,

ROUND(AVG(stars), 2) AS avg\_value

FROM reviews

GROUP BY EXTRACT(MONTH FROM submit\_date),

product\_id

ORDER BY month,

product;

**Copy code**

#### **8. JPMorgan – Cards Issued Difference**

**Your team at JPMorgan Chase is preparing to launch a new credit card, and to gain some insights, you’re analyzing how many credit cards were issued each month.**

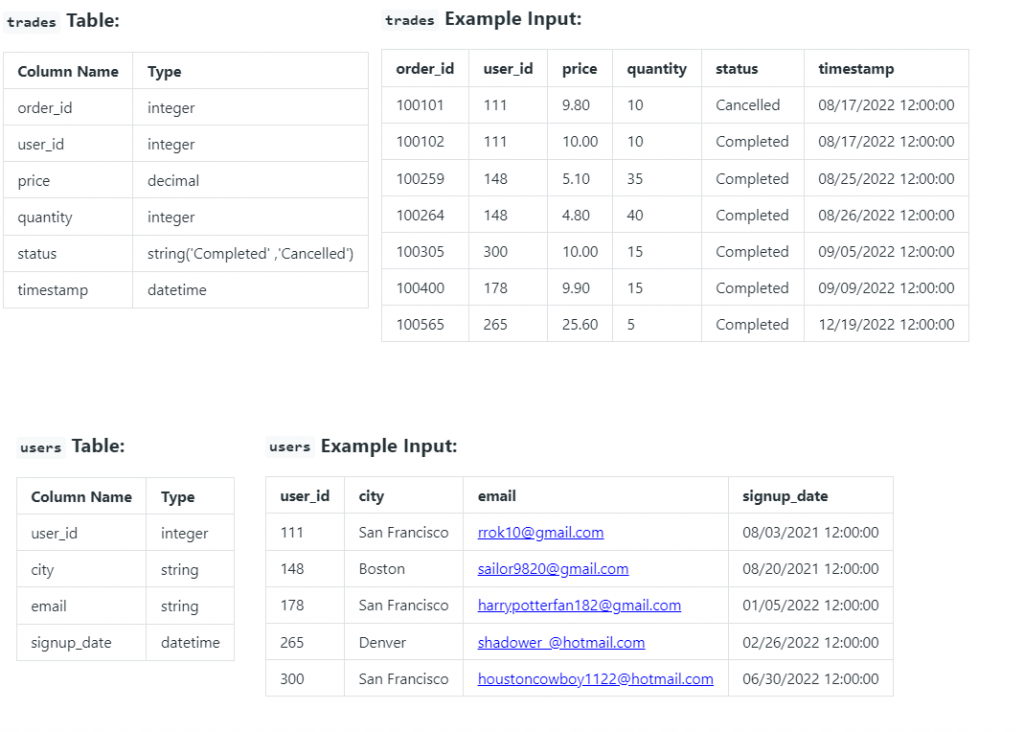
**Write a query that outputs the name of each credit card and the difference in the number of issued cards between the month with the highest and lowest issuance cards. Arrange the results based on the largest disparity.**



#### **9. Cities With Completed Trades**

**Assume you’re given the tables containing completed trade orders and user details in a Robinhood trading system.**

**Write a query to retrieve the top three cities that have the highest number of completed trade orders listed in descending order. Output the city name and the corresponding number of completed trade orders.**



**Solution**

SELECT u.city, COUNT(t.order\_id) as total\_orders

FROM trades t

LEFT JOIN users u

on t.user\_id = u.user\_id

WHERE t.status = 'Completed'

GROUP BY city

ORDER BY total\_orders DESC

LIMIT 3

**Copy code**

#### **10. Alibaba – Compressed Mean**

**You’re trying to find the mean number of items per order on Alibaba, rounded to 1 decimal place using tables which includes information on the count of items in each order (**

item\_count

**Copy code**

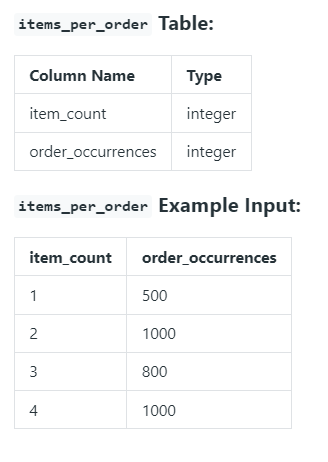
**table) and the corresponding number of orders for each item count (**

order\_occurrences

**Copy code**

**table).**

**There are a total of 500 orders with one item per order, 1000 orders with two items per order, and 800 orders with three items per order.**



**Solution**

SELECT ROUND(1.0\*SUM(ITEM\_COUNT\*ORDER\_OCCURENCES)/SUM(ORDER\_OCCURENCES),1) AS MEAN

FROM ITEMS\_PER\_ORDER;

**Copy code**

**Explanation**

Here, we multiplied 1.0 by the sum to convert the integer to decimal format.

## Basic Level SQL Interview Questions

Below is the list of the most important basic MySQL and SQL interview questions.

## Q1. What is DBMS?

**Ans.** A Database Management System (DBMS) is system software for creating and managing databases. It serves as an interface between databases and end-users or application programs so that data is consistently organized and remains easily accessible. It allows end-users to create, read, update, and delete data in a database. There are two types of DBMS:

* Relational Database Management System (RDBMS): In RDBMS, the data is stored in relations (tables). Example – MySQL.
* Non-Relational Database Management System (often called NoSQL databases): It stores data in a non-tabular form.  Example – MongoDB

[[](https://www.shiksha.com/online-courses/articles/what-are-the-advantages-of-dbms/)](https://www.shiksha.com/online-courses/articles/what-are-the-advantages-of-dbms/)

**[What are the Advantages of DBMS?](https://www.shiksha.com/online-courses/articles/what-are-the-advantages-of-dbms/)**

[A database management system (DBMS) is a software program that allows users to interact with a database. Let's explore advantages](https://www.shiksha.com/online-courses/articles/what-are-the-advantages-of-dbms/)**[...read more](https://www.shiksha.com/online-courses/articles/what-are-the-advantages-of-dbms/)**

[[](https://www.shiksha.com/online-courses/articles/top-8-disadvantages-of-dbms/)](https://www.shiksha.com/online-courses/articles/top-8-disadvantages-of-dbms/)

**[Top 8 Disadvantages of DBMS](https://www.shiksha.com/online-courses/articles/top-8-disadvantages-of-dbms/)**

[DBMS can be costly, complex to set up & maintain. Also, may pose performance issues, & to data security. Explore](https://www.shiksha.com/online-courses/articles/top-8-disadvantages-of-dbms/)**[...read more](https://www.shiksha.com/online-courses/articles/top-8-disadvantages-of-dbms/)**

***Explore courses related to SQL:***

|  |  |
| --- | --- |
| [Popular Technology Courses](https://www.shiksha.com/online-courses/technology-courses-certification-training-ct139?fftid=hamburger) | [Popular Database Administration Courses](https://www.shiksha.com/online-courses/database-administration-courses-certification-training-st605-tg1347) |
| [Popular Programming Courses](https://www.shiksha.com/online-courses/programming-courses-certification-training-st619) | [Popular Data Visualization Courses](https://www.shiksha.com/online-courses/data-visualization-courses-certification-training-st583-tg1061) |

## Q2. What is SQL?

**Ans.** SQL (structured querying language) is a computer language used to create, update, and modify a database. It is the standard language for Relational Database System. All the RDMS like MySQL, MS Access, Oracle, and SQL Server use SQL as their standard database language.

## Q3. What is MySQL?

And. MySQL is an open-source relational database management system (RDBMS) that is developed and distributed by Oracle Corporation. Supported by various operating systems, such as Windows, Unix, Linux, etc., MySQL can be used to develop different types of applications. Known for its speed, reliability, and flexibility, MySQL is mainly used for developing web applications.

## Q4. What are the subsets of SQL? Explain them.

**Ans.** The following are the three subsets of SQL:

1. **Data Definition Language (DDL)** – It allows end-users to CREATE, ALTER, and DELETE database objects.
2. **Data Manipulation Language (DML)** – With this, you can access and manipulate data. It allows you to Insert, Update, Delete, and Retrieve data from the database.
3. **Data Control Language (DCL)** – This lets you control access to the database. It includes the Grant and Revoke permissions to manipulate or modify the database.

## Q5. What is the primary key?

**Ans.** A primary key constraint uniquely identifies each row/record in a database table. Primary keys must contain unique values. Null value and duplicate values are not allowed to be entered in the primary key column. A table can have only one primary key. It can consist of single or multiple fields.

## Q6. What is a foreign key?

**Ans.** A foreign key (often called the referencing key) is used to link two tables together. It is a column or a combination of columns whose values match a Primary Key in a different table. It acts as a cross-reference between tables because it references the primary key of another table and established a link between them.

**Learn –**[**what is Database Administration**](https://www.shiksha.com/online-courses/what-is-database-administration-st605-tg1347)**?**

## Q7. What is RDBMS?

**Ans.** Relational Database Management System or RDBMS is based on the relational database model and is among the most popular database management systems.

**Also Read:**[**What is the Difference Between DBMS and RDBMS?**](https://www.shiksha.com/online-courses/articles/difference-between-dbms-and-rdbms/)

## Q8. What are the features of MySQL?

**Ans.** Here are some of the **important features** of MySQL:

* It is reliable and easy to use
* It supports standard SQL (Structured Query Language)
* MySQL is secure as it consists of a data security layer that protects sensitive data from unauthorized users
* MySQL has a flexible structure and supports a large number of embedded applications
* It is one of the very fast database languages
* It is a suitable database software for both large and small applications
* MySQL offers very high-performance results compared to [other databases](https://www.shiksha.com/online-courses/databases-courses-certification-training-st605?fftid=hamburger)
* It is supported by many well-known programming languages, such as PHP, [Java](https://www.shiksha.com/online-courses/java-courses-certification-training-st619-tg17), and C++
* It is free to download and use

**Also Read>>**[**PHP Interview Questions and Answers**](https://www.shiksha.com/online-courses/articles/php-interview-questions-and-answers/)

## Q9. What are the disadvantages of MySQL?

**Ans.** The **disadvantages**of MySQL are:

* It is hard to make MySQL scalable
* It does not support a very large database size as efficiently
* MySQL does not support SQL check constraint
* It is prone to data corruption

**Check out the best**[**Database Administration Courses**](https://www.shiksha.com/online-courses/database-administration-courses-certification-training-st605-tg1347)

## Q10. What are the differences between MySQL vs SQL?

**Ans.** This is one of the frequently asked **SQL interview questions**.

The [differences between MySQL and SQL](https://www.shiksha.com/online-courses/articles/difference-between-sql-and-mysql/) are:

|  |  |
| --- | --- |
| **MySQL** | **SQL** |
| 1. It is a relational database that uses SQL to query a database | 1. It is a query language |
| 2. MySQL supports multiple storage engines and plug-in storage engines | 2. SQL supports a single storage engine |
| 3. It is a database that stores the existing data in a database in an organized manner. | 3. SQL is used to access, update, and manipulate the data stored in a database |
| 4. Supports many platforms | 4. Supports only Linux and Windows |
| 5. It has a complex syntax | 5. It has a simpler syntax |

## Q11. What is a unique key?

**Ans.** A unique key is a set of one or more than one field/column of a table that uniquely identifies a record in a database table. A primary key is a special kind of unique key.

## Q12. Explain the different types of indexes in SQL.

**Ans.** There are three types of indexes in SQL:

1. **Unique Index –**It does not allow a field to have duplicate values if the column is unique indexed.
2. **Clustered Index –** This index defines the order in which data is physically stored in a table. It reorders the physical order of the table and searches based on key values. There can be only one clustered index per table.
3. **Non-Clustered Index –** It does not sort the physical order of the table and maintains a logical order of the data. Each table can have more than one non-clustered index.

## Q13. What is the difference between TRUNCATE and DELETE?

**Ans.** This is one of the most commonly asked SQL interview questions. The difference between TRUNCATE and DELETE are:

|  |  |
| --- | --- |
| **DELETE** | **TRUNCATE** |
| Delete command is used to delete a specified row in a table. | Truncate is used to delete all the rows from a table. |
| You can roll back data after using the delete statement. | You cannot roll back data. |
| It is a DML command. | It is a DDL command. |
| It is slower than a truncate statement. | It is faster. |

## Q14. What is the difference between:

**SELECT \* FROM MyTable WHERE MyColumn <> NULL**

**SELECT \* FROM MyTable WHERE MyColumn IS NULL**

**Ans.** The first syntax will not work because NULL means ‘no value’, and you cannot use scalar value operators. This is why there is a separate IS – a NULL predicate in SQL.

***Also explore:***

* Paid and [free online courses](https://www.shiksha.com/online-courses/coursera-courses-certification-training-v475) by Coursera
* Popular online [Udemy courses](https://www.shiksha.com/online-courses/udemy-courses-certification-training-v21)
* Top online [edX courses](https://www.shiksha.com/online-courses/edx-courses-certification-training-v117)

## Q15. What is the difference between CHAR and VARCHAR?

**Ans.** CHAR is a fixed-length character data type, while VARCHAR is a variable-length character data type.

## Q16. What is a subquery in SQL? What are the different types of subquery?

**Ans.** A subquery is a query within another query. When there is a query within a query, the outer query is called the main query, while the inner query is called a subquery. There are two types of a subquery:

* Correlated subquery: It obtains values from its outer query before it executes. When the subquery returns, it passes its results to the outer query.
* Non-Correlated subquery: It executes independently of the outer query. The subquery executes first and then passes its results to the outer query. Both inner and outer queries can run separately.

**Also explore:**[Understanding Subqueries in SQL](https://www.shiksha.com/online-courses/articles/subqueries-in-sql/)

## Q17. What is collation sensitivity?

**Ans.** Collation sensitivity defines the rules to sort and compare the strings of character data, based on correct character sequence, case sensitivity, character width, and accent marks, among others.

## Q18. What are the different types of collation sensitivity?

**Ans.** There are four types of collation sensitivity, which include –

* Accent sensitivity
* Case sensitivity
* Kana sensitivity
* Width sensitivity

## Q19. What is a “scheduled job” or “scheduled task”?

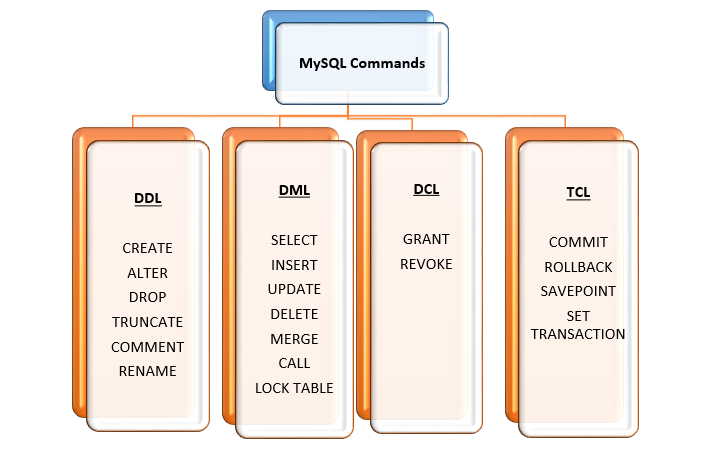
**Ans.**Scheduled job or task allows automated task management on regular or predictable cycles. One can schedule administrative tasks and decide the order of the tasks.

**Check out the best**[**Programming courses**](https://www.shiksha.com/online-courses/programming-courses-certification-training-st619?fftid=hamburger)**.**

## Q20. Can you name different types of MySQL commands?

**Ans.** SQL commands are divided into the following –

* Data Definition Language (DDL)
* Data Manipulation Language (DML)
* Data Control Language (DCL)
* Transaction Control Language (TCL)



## Q21. Explain different DDL commands in MySQL.

**Ans.** DDL commands include –

* CREATE – Used to create the database or its objects like table, index, function, views, triggers, etc.
* DROP – Used to delete objects
* ALTER – Used to change database structures
* TRUNCATE – It erases all records from a table, excluding its database structure
* COMMENT – Used to add comments to the data dictionary
* RENAME – Used to rename a database object

## Q22. Explain different DML commands in MySQL.

**Ans.** This is one of the most popularly asked **SQL interview questions**.

DML commands include –

* SELECT – Used to select specific database data
* INSERT – Used to insert new records into a table
* UPDATE – It helps in updating existing records
* DELETE – Used to delete existing records from a table
* MERGE – Used to UPSERT operation (insert or update)
* CALL – It is used when you need to call a PL/SQL or Java subprogram
* EXPLAIN PLAN – Used to interpret data access path
* LOCK TABLE – Used to control concurrency

## Q23. Explain different DCL commands in MySQL.

**Ans.** DCL commands are –

* GRANT – It provides user access privileges to the database
* DENY – Used to deny permissions to users
* REVOKE – Used to withdraw user access by using the GRANT command

## Q24. Explain different TCL commands in MySQL.

**Ans.** DCL commands include –

* COMMIT – Used to commit a transaction
* ROLLBACK – Used to roll back a transaction
* SAVEPOINT – Used to roll back the transaction within groups
* SET TRANSACTION – Used to specify transaction characteristics

## Q25. What are the different types of Database relationships in MySQL?

**Ans.** There are three types of Database Relationship –

* One-to-one – Both tables can have only one record
* One-to-many – The single record in the first table can be related to one or more records in the second table
* Many-to-many – Each record in both the tables can be related to any number of records

## Q26. What is Normalization?

**Ans.** Normalization is a database design technique to organize tables to reduce data redundancy and data dependency.

**Also explore:**[Introduction to Normalization – SQL Tutorial](https://www.shiksha.com/online-courses/articles/introduction-to-normalization-sql-tutorial/)

## Q27. What are the different types of Normalization?

**Ans.** There are six different types of Normalization –

* First Normal Form (1NF)
* Second Normal Form (2NF)
* Third Normal Form (3NF)
* Boyce-Codd Normal Form (BCNF)
* Fourth Normal Form (4NF)
* Fifth Normal Form (5NF)

## Q28. What is Denormalization?

**Ans.** Denormalization is a database optimization technique for increasing a database infrastructure performance by adding redundant data to one or more tables.

**Check out the list of**[**Top Online IT Courses**](https://www.shiksha.com/online-courses/articles/top-online-courses-for-it-professionals/)**.**

## Q29. Is MySQL query case-sensitive?

**Ans.** MySQL queries are not case-sensitive by default. The following queries are the same.

**SELECT \* FROM `table` WHERE `column` = ‘value’**

**SELECT \* FROM `table` WHERE `column` = ‘VALUE’**

**SELECT \* FROM `table` WHERE `column` = ‘VaLuE’**

## Q30. How many TRIGGERS are allowed in the MySQL table?

**Ans.** 6 triggers are allowed in the MySQL table:

* BEFORE INSERT
* AFTER INSERT
* BEFORE UPDATE
* AFTER UPDATE
* BEFORE DELETE
* AFTER DELETE

## Q31. What are the different column comparison operators in MySQL?

**Ans.**The =, <>, <=, <, >=, >, <<, >>, < = >, AND, OR or LIKE operator are the comparison operators in MySQL.

Comparisons operators are generally used with SELECT statements. They are used to compare one expression to another value or expression.

## Q32. What syntax can we use to get a version of MySQL?

**Ans.** By using the given query in your phpmyadmin-

**SELECT version();**

## Q33. What is Auto Increment in SQL?

**Ans.** Auto Increment allows a unique number to be generated whenever a new record is created in a table. Generally, it is the PRIMARY KEY field that we want to be created automatically every time a new record is inserted.

## Q34. SQL Server runs in which TCP/IP port? Can it be changed?

**Ans.** SQL Server runs on port 1433, and it can be changed from the Network Utility TCP/IP properties.

## Q35. Name symmetric key encryption algorithms supported in the SQL server.

**Ans.** SQL Server supports several symmetric key encryption algorithms, such as DES, Triple DES, RC2, RC4, 128-bit RC4, DESX, 128-bit AES, 192-bit AES, and 256-bit AES.

## Q36. What is Database Relationship?

**Ans.** A Database Relationship is defined as the connection between two relational database tables. The primary table has a foreign key that references the primary key of another table. There are three types of Database Relationship –

* One-to-one
* One-to-many
* Many-to-many

## Q37. What is faster between a table variable and a temporary table?

**Ans.** Between these, a table variable is faster mostly as it is stored in memory, whereas a temporary table is stored on disk. In case the size of the table variable exceeds memory size, then both the tables perform similarly.

**Also Read>>**[**Top Tableau Interview Questions and Answers**](https://www.shiksha.com/online-courses/articles/tableau-interview-questions-answers/)

## Q38. Mention the command used to get back the privileges offered by the GRANT command?

**Ans.** REVOKE command is used to get back the privileges offered by the GRANT command.

## ****Q39. What is a Clause in SQL?****

**Ans.** A clause in SQL is a part of a query that allows users to filter or customize how they want their data to be queried to them. It lets users limit the result set by providing a condition to the query. When there is a large amount of data stored in the database, Clause can be used to query and get data required by the user. The clause function helps filter and analyze data quickly.

**For Example –** WHERE clause, HAVING clause.

## ****Q40. Explain the ‘WHERE’ Clause and the ‘HAVING’ Clause.****

**Ans.** It is one of the most important SQL interview questions.

**The WHERE clause** is used to filter the records from the table or used while joining more than one table. It returns the particular value from the table if the specified condition in the WHERE clause is satisfied. It is used with SELECT, INSERT, UPDATE, and DELETE queries to filter data from the table or relation.

**For Example:**

SELECT \* FROM employees

WHERE working\_hour > 9;

**The HAVING clause** is used to filter the records from the groups based on the given condition in the HAVING Clause. It can only be used with the SELECT statement. It returns only those values from the groups in the final result that fulfills certain conditions.

**For Example:**

SELECT name, SUM(working\_hour) AS “Total working hours”

FROM employees GROUP BY name

HAVING SUM(working\_hour) > 6;

### Free SQL Courses

[**View all**](https://www.shiksha.com/online-courses/free-sql-courses-certification-training-st605-tg1361)

[[](https://www.shiksha.com/online-courses/managing-relational-databases-course-courl6057)CourseraManaging Relational Databases**Go to Free Course**](https://www.shiksha.com/online-courses/managing-relational-databases-course-courl6057)[[](https://www.shiksha.com/online-courses/sql-a-practical-introduction-for-querying-databases-course-courl5464)CourseraSQL: A Practical Introduction for Querying Databases**Go to Free Course**](https://www.shiksha.com/online-courses/sql-a-practical-introduction-for-querying-databases-course-courl5464)[[University](https://www.shiksha.com/online-courses/advanced-mysql-topics-course-courl5224)MetaCourseraAdvanced MySQL Topics**Go to Free Course**](https://www.shiksha.com/online-courses/advanced-mysql-topics-course-courl5224)[[University](https://www.shiksha.com/online-courses/implement-polyglot-persistence-using-sql-and-nosql-databases-course-courl5049)NIITCourseraImplement Polyglot Persistence Using SQL and NoSQL Databases**Go to Free Course**](https://www.shiksha.com/online-courses/implement-polyglot-persistence-using-sql-and-nosql-databases-course-courl5049)[[University](https://www.shiksha.com/online-courses/introduction-to-software-programming-and-databases-course-courl4994)IBMCourseraIntroduction to Software, Programming, and Databases**Go to Free Course**](https://www.shiksha.com/online-courses/introduction-to-software-programming-and-databases-course-courl4994)

## ****Q41. Explain the SELECT statement?****

**Ans.**The SQL SELECT statement helps select data from a database. It returns a result set of records, from one or more tables.

**Syntax:**

SELECT \* FROM myDB.employees;

## ****Q42. What are the differences between the ‘WHERE’ Clause and the ‘HAVING’ Clause?****

**Ans.** Below are the major differences between the ‘WHERE’ Clause and the ‘HAVING’ Clause:

|  |  |
| --- | --- |
| **WHERE Clause** | **HAVING Clause** |
| It performs filtration on individual rows based on the specified condition. | HAVING clause performs filtration on groups based on the specified condition. |
| It can be used without GROUP BY Clause. | It is always used with the GROUP BY Clause. |
| WHERE Clause is applied in row operations. | HAVING is applied in column operations. |
| We cannot use the WHERE clause with aggregate functions. | This clause works with aggregate functions. |
| WHERE comes before GROUP BY | HAVING comes after GROUP BY. |
| This clause acts as a pre-filter. | HAVING clause acts as a post-filter. |
| WHERE Clause can be used with SELECT, INSERT, UPDATE, and DELETE statements. | This Clause can only be used with the SELECT statement. |

**Explore**[**Popular Online Courses**](https://www.shiksha.com/online-courses/)

Now, let’s move on to advanced-level **SQL interview questions**.

## Advanced Level SQL Interview Questions

The following are the commonly asked advanced-level SQL and MySQL interview questions.

## Q43. How to find:

**duplicate records with one field?**

**duplicate records with more than one field?**

**Ans.** **Finding duplicate records with one field:**

SELECT COUNT(field)

FROM table\_name

GROUP BY field

HAVING COUNT(field) > 1

**Finding duplicate records with more than one field:**

SELECT field1,field2,field3, COUNT(\*)

FROM table\_name

GROUP BY field1,field2,field3

HAVING COUNT(\*) > 1

***Also Read>>***[***Top Web Developer Interview Questions and Answers***](https://www.shiksha.com/online-courses/articles/top-20-web-developer-interview-questions-answers/)

## Q44. What is a constraint, and how many levels of constraints are there?

**Ans.** Constraints are the representation of a column to enforce data entity and consistency. There are two levels of constraint –

* Column level – Limits only column data
* Table level – Limits whole table data

Following are the most used constraints that can be applied to a table:

* NOT NULL
* UNIQUE
* CHECK
* DEFAULT
* PRIMARY KEY
* FOREIGN KEY

## Q45. What are the authentication modes in SQL Server?

**Ans.** SQL Server has two authentication modes –

* Windows Mode – Default. This SQL Server security model is integrated with Windows
* Mixed Mode – Supports authentication both by Windows and by SQL Server

We can change modes by selecting tools of SQL Server configuration properties and then hover over the security page.

## Q46.  What is PL/SQL?

**Ans.** PL/SQL or Procedural Language for SQL was developed by Oracle. It is an extension of SQL and enables the programmer to write code in a procedural format. Both PL/SQL and SQL run within the same server process and have features like – robustness, security, and portability of the Oracle Database.

## Q47. What is SQL Profiler?

**Ans.** SQL Server Profiler is a graphical user interface for creating and saving data about each event of a file. It also allows a system administrator to analyze and replay trace results when a problem is being diagnosed. SQL Server Profiler is used to:

* Examine the problem queries to find the cause of the problem
* Diagnose slow-running queries
* Determine the Transact-SQL statements that lead to a problem
* Monitor the performance of the SQL Server
* Correlate performance counters to diagnose problems

## Q48. What is the SQL Server Agent?

**Ans.** SQL Server Agent is a Microsoft Windows service that executes day-to-day tasks or jobs of SQL Server Database Administrator (DBA). This service enables the implementation of tasks on a scheduled date and time.

***Also, Read>>***[***Top Hadoop Interview Questions and Answers***](https://www.shiksha.com/online-courses/articles/hadoop-interview-questions-answers/)

## Q49. What is Data Integrity?

**Ans.** Data integrity attributes to the accuracy, completeness, and consistency of the data in a database. It also refers to the safety and security of data and is maintained by a collection of processes, rules, and standards that were implemented during the design phase. Three types of data integrity are:

* Column Integrity
* Entity Integrity
* Referential Integrity

## Q50. What is the difference between Rename and Alias?

**Ans.**Rename is actually changing the name of an object. Alias is giving another name (additional name) to an existing object. Rename involves changing the name of a database object and giving it a permanent name whereas Alias is a temporary name given to a database object.

**Syntax of a table Alias:**

SELECT column1, column2….

FROM table\_name AS alias\_name

WHERE [condition];

**Syntax of a table Rename:**

RENAME TABLE {tbl\_name} TO {new\_tbl\_name};

## Q51.  Which are the main steps in Data Modeling?

Ans. Following are the main steps in Data Modeling:

* Identify and analyze business requirement
* Create a quality conceptual and logical data model
* Select the target database to create scripts for physical schema using a data modeling tool

## Q52. What is Referential Integrity?

**Ans.** Referential integrity is a relational database concept that suggests that the accuracy and consistency of data should be maintained between primary and foreign keys.

**Explore**[**Free Online Courses with Certificates**](https://www.shiksha.com/online-courses/free-online-courses-certificates)

## Q53. What is Business Intelligence?

**Ans.** Business intelligence (BI) includes technologies and practices for collecting, integrating, analyzing, and presenting business information. It combines business analytics, data mining, data visualization, data tools and infrastructure, and best practices.

## Q54. Mention the types of privileges available in SQL?

**Ans.** Following are the types of privileges used in SQL:

**System Privilege:** It deals with an object of a specific type and indicates actions on it which include admin that helps users to perform administrative tasks, alter any cache group, and alter any index.

**Object Privilege:** It helps users to perform actions on an object using commands like table, view, and indexes. There are other object privileges used in SQL EXECUTE, INSERT, SELECT, FLUSH, LOAD, INDEX, UPDATE, DELETE, REFERENCES, etc.

## Q55. What is the difference between a clustered and non-clustered index?

**Ans.** **Clustered Index –** A clustered index is used to order the rows in a table. It has leaf nodes consisting of data pages. A table can possess only one clustered index.

**Non-clustered Index –** A non-clustered index stores the data and indices at different places. It also has leaf nodes that contain index rows. A table can possess numerous non-clustered indexes.

## Q56. What is ERD?

**Ans.** ERD or Entity Relationship Diagram is a visual representation of the database structures and shows a relationship between the tables. The ER Diagrams have three basic elements:

* **Entities** – An entity is a person, place, thing, or event for which data is collected.
* **Attributes** – It refers to the data we want to collect for an entity. It is a property, trait, or characteristic of an entity, relationship, or another attribute.
* **Relationships** – It describes how entities interact.

## Q57. How will you find the unique values, if a value in the column is repeatable?

**Ans.** To find the unique values when the value in the column is repeatable, we can use DISTINCT in the query, such as:

**SELECT DISTINCT user\_firstname FROM users;**

We can also ask for several distinct values by using:

**SELECT COUNT (DISTINCT user\_firstname) FROM users;**

## Q58. Explain database white box testing.

**Ans.** White Box Testing is concerned with the internal structure of the database. The users are unaware of the specification details.

* Database white box testing includes testing of database triggers and logical views that support database refactoring.
* Validates database tables, data models, database schema
* Performs module testing of database functions and SQL queries
* Select default table values to check on database consistency
* Adheres to referential integrity rules

## Q59. Exhibit the students who are having the same batch ID and study in the same department as student ids, 1002 and 1004.

**Ans.**

select x.student\_id ,

x.department\_id

from students x

where (department\_id, batch\_id)

in (Select department\_id , batch\_id

from students

where student\_id in (1002,1004))

and x.student\_id not in (1002, 1004)

## Q60. What is the ACID property in SQL?

**Ans.** ACID is short for Atomicity, Consistency, Isolation, Durability. It ensures Data Integrity during a transaction.

**Atomicity:** It means either all the operations (insert, update, delete) inside a transaction take place or none.  So, if one part of any transaction fails, the entire transaction fails and the database state is left unchanged.

**Consistency:** Consistency ensures that the data must meet all the validation rules. Irrespective of whatever happens in the middle of the transaction, Consistency property will never leave your database in a half-completed state.

**Isolation:** It means that every transaction is individual. One transaction can’t access the result of other transactions until the transaction is completed.

**Durability:** It implies that maintaining updates of committed transactions is important. These updates must never be lost. It refers to the ability of the system to recover committed transaction updates if either the system or the storage media fails.

***Also Read>>***[***Top MongoDB Interview Questions and Answers***](https://www.shiksha.com/online-courses/articles/top-mongodb-interview-questions-and-answers/)

## Q61. Explain string functions in SQL?

**Ans.** SQL string functions are used for string manipulation.

Following are the extensively used SQL string functions:

* UPPER(): Converts character data to upper case
* LOWER(): Converts character data to lower case
* SUBSTRING() : Extracts characters from a text field
* RTRIM(): Removes all whitespace at the end of the string
* LEN(): Returns the length of the value in a text field
* REPLACE(): Updates the content of a string.
* LTRIM(): Removes all whitespace from the beginning of the string
* CONCAT(): Concatenates function combines multiple character strings

## Q62. What are the differences between the Primary key and the Unique key?

**Ans.** Differences between the Primary key and the Unique key are:

|  |  |
| --- | --- |
| **Primary Key** | **Unique Key** |
| Enforces column uniqueness in a table | Determines a row that isn’t a primary key |
| Does not allow NULL values | Accepts one NULL value |
| Has only one primary key | Has more than one unique key |
| Creates clustered index | Creates non-clustered index |
| Primary Key on CREATE TABLE Syntax: CREATE TABLE Students (    ID int NOT NULL PRIMARY KEY,    LastName varchar(255) NOT NULL,    FirstName varchar(255),    Age int); | Unique Key on CREATE TABLE Syntax: CREATE TABLE Students (    ID int NOT NULL UNIQUE,    LastName varchar(255) NOT NULL,    FirstName varchar(255),    Age int); |

Try explaining these differences in the answer for this basic SQL interview question.

## Q63. Write the SQL query to convert the string to UPPERCASE and LOWERCASE.

**Ans.** The SQL query used to convert the string to UPPERCASE and LOWERCASE is:

STRING UPPER(“naukrilearning”); => NAUKRILEARNING

STRING LOWER(“LEARNERS”); => learners

## Q64. What is the procedure to hide a specific table name of the schema?

**Ans.** By using SYNONYMS, we can hide a specific table name of the schema.

Syntax:

CREATE SYNONYM STU for STUDENTS;

After creating the above synonym, we can access the data of the STUDENTS table using STU as the table name below

SELECT \* from STU;

## Q65. What is the syntax to eliminate duplicate rows?

**Ans.** By using the DISTINCT keyword, we can eliminate duplicate records.

Syntax:

SELECT DISTINCT CLASS\_ID

FROM STUDENTS;

## Q66. Find out nth highest salary from emp table?

**Ans.** Syntax:

select salary from

(select salary, rownum EP from

(select salary from employees

order by salary desc))

where EP=n;

## Q67. Name the encryption mechanisms in the SQL server.

**Ans.** This is one of the most popular SQL interview questions. The encryption mechanism used in SQL servers are –

* Transact-SQL functions – Individual items can be encrypted as they are inserted or updated using Transact-SQL functions.
* Asymmetric keys – It is made up of a private key and the corresponding public key. Each key can decrypt data encrypted by the other.
* Symmetric keys – It is used for both encryption and decryption.
* Certificates – Also known as a public key certificate, it binds the value of a public key to the identity of the person, device, or service that holds the corresponding private key.
* Transparent Data Encryption – It is a special case of encryption using a symmetric key that encrypts an entire database using that symmetric key.

## Q68. What is the procedure to pass variables in a SQL routine?

**Ans.** Variables can be passed to a SQL routine by using:

* “&” symbol
* SQLPLUS command

***Also Read>>***[***Top Database Interview Questions and Answers***](https://www.shiksha.com/online-courses/articles/database-interview-questions-answers/)

## Q69. Can a view be updated/inserted/deleted? If yes, at what conditions?

**Ans.** It is not possible to add the data through a view if the view contains the following:

* Group by clause
* Group functions
* DISTINCT keyword
* Columns defined by expressions
* Pseudo column ROWNUM keyword
* NOT NULL column in the base table that is not selected by the view.

## Q70. How can you create an SQL table from another table without copying any values from the old table?

**Ans. Syntax:**

CREATE TABLE new\_table

  AS (SELECT \*

      FROM old\_table WHERE 1=2);

This will create a new table with the same structure as the old table with no rows copied.

## Q71. Explain what is an inline view?

**Ans.** An inline view is a SELECT statement in the FROM-clause of another SELECT statement. In-line views are used to reduce complex queries by removing join operations and summarizing multiple separate queries into a single query.

**Syntax:**

SELECT SALARY FROM

(SELECT SALARY, ROWNUN EP FROM

(SELECT SALARY FROM EMPLOYEES ORDER BY SALARY DESC) )

WHERE EP=7

## Q72. What command is used to create a table by copying the structure of another table?

**Ans.** **Syntax:**

CREATE TABLE STU AS

SELECT \* FROM STUDENTS

WHERE 1=2

Invalid Condition

We have to give the invalid condition in the where clause, where the whole data will copy to the new table (STU table).

## Q73. Mention the use of the DROP option in the ALTER TABLE command.

**Ans.** The use of the DROP option in the ALTER TABLE command is to drop a particular COLUMN.

**Syntax:**

ALTER TABLE TABLE\_NAME

DROP COLUMN COLUMN\_NAME

## Q74. What are the aggregate functions in SQL?

**Ans.** SQL aggregate functions allow us to return a single value, which is calculated from values in a column.

Following are the aggregate functions in SQL:

* AVG() : This function returns the average value
* COUNT(): This function returns the number of rows
* MAX() : It returns the largest value
* MIN() : This function returns the smallest value
* ROUND(): This function rounds a numeric field to the number of decimals specified
* SUM() : It returns the sum

## Q75. Write the SQL query to update the student names by removing leading and trailing spaces.

**Ans.** This can be done by using ‘Update’ command with ‘LTRIM’ and ‘RTRIM’ function.

**Syntax:**

UPDATE StudentDetails

SET FullName = LTRIM(RTRIM(FullName));

## Q76. Write the SQL query to fetch alternate records from a table

**Ans.** Records can be fetched for odd and even row numbers:

* **Syntax to fetch even numbers:**

Select employeeId from (Select rowno, employeeId from employee) where mod(rowno,2)=0

* **Syntax to fetch odd numbers:**

Select employeeId from (Select rowno, employeeId from employee) where mod(rowno,2)=1

## Q77. How do you return a hundred books starting from the 15th?

**Ans.** The syntax will be:

**SELECT book\_title FROM books LIMIT 15, 100.**

The first number in LIMIT is the offset, and the second is the number.

## Q78. How will the query select all teams that lost either 1, 3, 5, or 7 games?

**Ans.** We will use-

**SELECT team\_name FROM teams WHERE team\_lost IN (1, 3, 5, 7)**

## Q79. How will you delete a column?

**Ans.** We can delete a column by –

**ALTER TABLE techpreparation\_answers DROP answer\_user\_id.**

## Q80. What is the meaning of this query – Select User\_name, User\_isp From Users Left Join Isps Using (user\_id)?

**Ans.** It means:

**SELECT user\_name, user\_isp FROM users LEFT JOIN isps WHERE users.user\_id=isps.user\_id**

## Q81. How will you see all indexes defined for a table?

**Ans.** By using:

**SHOW INDEX FROM techpreparation\_questions;**

## Q82. How would you change a table to InnoDB?

**Ans.** By using:

**ALTER TABLE techpreparation\_questions ENGINE InnoDB;**

## Q83. Name the default port for the MySQL server.

**Ans.** The default port for the MySQL server is **3306**.

## Q84. What is the possible way to add five minutes to a date?

**Ans.** By using:

**ADDDATE(techpreparation\_publication\_date, INTERVAL 5 MINUTE)**

## Q85. What is the possible way to convert between Unix timestamps and Mysql timestamps?

**Ans.** Example:

**UNIX\_TIMESTAMP converts from MySQL timestamp to Unix timestamp**

**FROM\_UNIXTIME converts from Unix timestamp to MySQL timestamp**

## Q86. How do you implement Enums and sets internally in MySQL?

**Ans.** To implement an ENUM column, use the given syntax:

**CREATE TABLE table\_name ( … col ENUM (‘value1′,’value2′,’value3’), … );**

## Q87. How can we restart SQL Server in the single user or the minimal configuration modes?

**Ans.**The command line SQLSERVER.EXE used with ‘–m’ will restart SQL Server in the single-user mode.

The command line SQLSERVER.EXE used with ‘–f’ will restart it in the minimal configuration mode.

## Q88. What is the use of the tee command in Mysql?

**Ans.**Tee is a UNIX command that takes the standard output of a Unix command and writes it to both the terminal and a file. Tee followed by a filename turns on MySQL logging to a specified file. It can be paused by a command note.

## Q89. Is it possible to save your connection settings to a conf file?

**Ans.** Yes, it is possible, and you can name it ~/.my.conf. You can also change the permissions on the file to 600 so that it’s not readable by others.

## Q90. How to convert numeric values to character strings?

**Ans.** We can convert numeric values to character strings by using the CAST(value AS CHAR) function, as shown in the following examples:

SELECT CAST(4123.45700 AS CHAR) FROM DUAL;

4123.45700

## Q91. Use mysqldump to create a copy of the database?

**Ans.** mysqldump -h mysqlhost -u username -p mydatabasename > dbdump.sql

## Q92. What are federated tables?

**Ans.** Federated tables allow access to the tables situated on other databases on other servers in MySQL. It lets you access data from a remote MySQL database without using replication or cluster technology. Querying a local FEDERATED table pulls the data from the remote (federated) tables. Data is not stored on the local tables.

## Q93. What are the different groups of data types in MySQL?

**Ans.** There are three groups of data types in MySQL, as listed below:

**String Data Types –** BINARY, VARBINARY, TINYBLOB, CHAR, NCHAR, VARCHAR, NVARCHAR, TINYTEXT, BLOB, TEXT, MEDIUMBLOB, LONGBLOB, LONGTEXT, ENUM, SET, MEDIUMTEXT.

**Numeric Data Types –** MEDIUMINT, INTEGER, BIGINT, FLOAT, BIT, TINYINT, BOOLEAN, SMALLINT, DOUBLE, REAL, DECIMAL.

**Date and Time Data Types –** TIMESTAMP, TIME, DATE, DATETIME, YEAR.

## Q94. What is the procedure to concatenate two character strings?

**Ans.** To concatenate various character strings into one, you can use the CONCAT() function. Example:

**SELECT CONCAT(’Naukri’,’ Learning’) FROM DUAL;**

**Shiksha Online**

**SELECT CONCAT(‘Learner’,’Thing’) FROM DUAL;**

**Learner Thing**

## Q95. What is the procedure to change the database engine in Mysql?

**Ans.** By using:

**ALTER TABLE EnterTableName ENGINE = EnterEngineName;**

## Q96. What is the default storage engine in MySQL?

**Ans.** InnoDB is the default storage engine in MySQL.

## Q97. What is COALESCE?

**Ans.** COALESCE returns the first non-NULL expression within its arguments from more than one column in the arguments.

The syntax for COALESCE is –

**COALESCE (expression 1, expression 2, … expression n)**

## Q98. What syntax is used to create an index in MySQL?

**Ans.** By using-

**CREATE INDEX [index name] ON [table name]([column name]);**

## ****Q99. How to store videos in SQL Server table?****

**Ans.** We use the FILESTREAM datatype to store videos in SQL server table.

## ****Q100. Explain the use of the NVL() function.****

**Ans.** The NVL()function converts the Null value to the other value.

## Q101. What are the different storage engines/table types present in MySQL?

**Ans.** MySQL supports two types of tables: transaction-safe tables (InnoDB and BDB) and non-transaction-safe tables (HEAP, ISAM, MERGE, and MyISAM).

* MyISAM: This is a default table type that is based on the Indexed Sequential Access Method (ISAM). It extends the former ISAM storage engine. These tables are optimized for compression and speed.
* HEAP: It allows fast data access. However, the data will be lost if there is a crash. HEAP table cannot have BLOB, TEXT, and AUTO\_INCREMENT fields.
* BDB: It supports transactions using COMMIT and ROLLBACK. It is slower than the others.
* InnoDB: These tables fully support ACID-compliant and transactions.
* MERGE: Also known as the MRG\_MyISAM engine, MERGE is a virtual table that combines multiple MyISAM tables that have a similar structure to one table.

## Q102. What are the differences between and MyISAM and InnoDB?

**Ans.** The following are the differences between and MyISAM and InnoDB

|  |  |
| --- | --- |
| **MyISAM** | **InnoDB** |
| No longer supports transactions | Supports transactions |
| It supports Table-level Locking | It helps in Row-level Locking |
| No longer assist ACID (Atomicity, Consistency, Isolation, and Durability) | Supports ACID property |
| Supports FULLTEXT index | Does not support FULLTEXT index |

## Q103. What drivers are available in MySQL?

**Ans.** Below are the drivers available in MySQL:

* PHP Driver
* C WRAPPER
* ODBC Driver
* JDBC Driver
* PYTHON Driver
* RUBY Driver
* PERL Driver
* CAP11PHP Driver
* Ado.net5.mxj

## Q104. What is a Join? What are the different types of joins in MySQL?

**Ans.** Join is a query that retrieves related columns or rows. There are four types of joins in MySQL:

* Inner Join – it returns the rows if there is at least one match in two tables.
* Left Join – returns all the rows from the left table even if there is no match in the right table.
* Right Join – returns all the rows from the right table even if no matches exist in the left table.
* Full Join – would return rows when there is at least one match in the tables.

## ****Q105. What is a pattern matching operator in SQL?****

**Ans.** The pattern matching operator in SQL allows you to perform a pattern search in data if you have no clue as to what that word should be. Rather than writing the exact word, this operator uses wildcards to match a string pattern. The LIKE operator is used with SQL Wildcards to get the required information.

LIKE operator is used for pattern matching in the below format:

* % – It matches zero or more characters.

**For Example –** To search for any employee in the database with the last name beginning with the letter A

SELECT \*

FROM employees

WHERE last\_name LIKE ‘A%’

* \_ (Underscore) – it matches exactly one character.

**For Example –** This example matches only if A appears at the third position of the last name

SELECT \*

FROM employees

WHERE last\_name LIKE ‘\_ \_A%’

## ****Q106.**** ****What is a Stored Procedure? What are its advantages and disadvantages?****

**Ans.** A Stored Procedure is an SQL function that consists of several SQL statements to access the database system. It can be stored for later use and can be used many times. If you have to perform a particular task, repeatedly, you won’t have to write the statements repeatedly, you will just have to call the stored procedure. This saves time and avoids writing code again.

**Syntax: To create a stored procedure**

CREATE PROCEDURE procedure\_name

AS

sql\_statement

GO;

**Syntax: To execute a stored procedure**

EXEC procedure\_name;

**Advantages of Stored Procedure:**

1. Execution becomes fast and efficient as stored procedures are compiled once and stored in executable form.
2. A Stored Procedure can be used as modular programming. Once created and stored, it can be called repeatedly, whenever required.
3. Maintaining a procedure on a server is easier than maintaining copies on different client machines.
4. Better security.

**Disadvantages of Stored Procedure:**

1. It can be executed only in the database and utilizes more memory in the database server.
2. Any data errors in handling stored procedures are not generated until runtime.
3. Version control is not supported.

## ****Q107. Explain the STUFF and REPLACE functions.****

**Ans.** This is one of the commonly asked **SQL interview questions**.

The **STUFF function** deletes a substring of a certain length of a string and replaces it with a new string. It inserts the string at a given position and deletes the number of characters specified from the original string.

**Syntax:**

STUFF (string\_expression, start, length, replacement\_string)

**Parameters:**

* string\_expression: the main string in which the stuff is to be applied.
* start: starting position of the character in string\_expression.
* length: length of characters that need to be replaced.
* replacement\_string: a new string that is to be applied to the main string.

The **REPLACE function**replaces all occurrences of a specific string value with another string.

**Syntax:**

REPLACE (string\_expression, search\_string, replacement\_string)

**Parameters:**

* string\_expression: the main string that contains the substring to be replaced.
* Search\_string: to locate the substring.
* replacement\_string: the new replacement string.

## ****Q108. What is a Database Cursor?****

**Ans.** A database cursor is a mechanism that allows for traversal over the records in a database. Cursors also allow processing after traversal, like retrieval, addition, and deletion of database records. A cursor is behaviorally similar to the programming language iterator.

**How to use a Database Cursor in SQL Procedures**

1. Declare variables.
2. Declare a cursor that defines a result set. The cursor declaration must always be associated with a SELECT Statement.
3. Open the cursor to initialize the result set.
4. FETCH statement to retrieve and move to the next row in the result set.
5. Close the cursor.
6. Deallocate the cursor.

## ****Q109. What are SQL Scalar functions? Name some.****

**Ans.** An SQL scalar function returns a single value based on the user input. Below are some of the commonly used scalar functions:

|  |  |  |
| --- | --- | --- |
| **SQL Scalar Function** | **Format** | **Description** |
| LCASE() | SELECT LCASE(column\_name) FROM table\_name; | converts the value of a field to lowercase |
| UCASE() | SELECT UCASE(column\_name) FROM table\_name; | converts the value of a field to uppercase |
| LEN() | SELECT LENGTH(column\_name) FROM table\_name; | returns the total length of the value in a text field |
| ROUND() | SELECT ROUND(column\_name,decimals) FROM table\_name; | rounds a numeric field to the number of decimals specified |
| NOW() | SELECT NOW() FROM table\_name; | returns the current system date and time |
| FORMAT() | SELECT FORMAT(column\_name,format) FROM table\_name; | sets the format to display a collection of values |

## ****Q110. What is the difference between SQL and PL/SQL?****

**Ans.** This is an important question that you must prepare for your **SQL interview**.

Below are some of the major differences between SQL and PL/SQL:

|  |  |
| --- | --- |
| **SQL** | **PL/SQL** |
| SQL is a database Structured Query Language. | PL/SQL or Procedural Language/Structured Query Language is a database programming language using SQL. It is a dialect of SQL to enhance SQL capabilities. |
| It was developed by IBM Corporation and first appeared in 1974. | It was developed by Oracle Corporation in the early 90s. |
| Data variables are not available. | Data variables are available. |
| SQL is a declarative language. | PL/SQL is a procedural language. |
| It is data-oriented. | PL/SQL is application-oriented. |
| It can execute only a single query at a time. | It can execute a whole block of code at a time. |
| SQL can directly interact with the database server. | PL/SQL cannot directly interact with the database server. |
| It can be embedded in PL/SQL. | It cannot be embedded in SQL. |
| SQL is used to write queries, DDL, and DML statements. | It is used to write program blocks, functions, procedures triggers, and packages |
| + More 1 Rows |  |

## ****Q111. Explain SQL comments.****

**Ans.**SQL comments help in explaining the sections of the SQL statements. They also help in avoiding the execution of SQL statements. There are three types of comments:

* Single line comments: They start and end within a single line. Single line comments start with –. The text between — and the end of the line is not executed.
* Multi-line comments: These comments start in one line and end in a different one. Any text between /\* and \*/ will not be executed.
* Inline comments: They are an extension of multi-line comments. We can write the comments between the statements enclosed within ‘/\*’ and ‘\*/.’

**Basics MySQL Interview Questions**

**1. What features does MySQL have in it?**

Below are some different features of MySQL.

* **Multithreading** in MySQL makes it scalable. Any size of data can be handled by MySQL. The default 4 GB file size limit can be increased to suit our needs.
* Many embedded systems are supported by MySQL that’s why it is flexible.
* **Operating systems** like Windows, macOS, and others are compatible with MYSQL.
* It has fewer memory leakage issues, which improves memory efficiency.
* The **partitioning** feature of MySQL he4lps to improve the performance of large databases.

**2. Why MySQL is used?**

The **MySQL Server** is quick, dependable, and simple to use. MySQL is user friendly and open-source language so we can easily download it from the Internet.

**3. How to get the current MySQL version?**

VERSION command is used to get the current [MySQL Version](https://www.geeksforgeeks.org/mysql-version-function/).

**Syntax:**

*SELECT VERSION ();*

**4. Explain the differences between SQL and MySQL?**

| **SQL** | **MySQL** |
| --- | --- |
| SQL is present in multiple languages. | MySQL is present only in English. |
| SQL programming language used to create database systems. | MySQL supports multiple programming languages. |
| SQL supports [XML](https://www.geeksforgeeks.org/xml-basics/). | It doesn’t support XML. |
| Users can define functions in SQL. | MySQL doesn’t support [User define functions](https://www.geeksforgeeks.org/python-user-defined-functions/). |
| SQL is developed by Microsoft Corporation. | MySQL is developed by Oracle Corporation. |

**5. What is the MySQL server’s default port?**

**3306** is [MySQL server](https://www.geeksforgeeks.org/how-to-stop-mysql-server-on-windows-and-linux/)‘s default port..

**6. How can we learn batch mode in MySQL?**

Below is the syntax used to run [batch](https://www.geeksforgeeks.org/batch-command-in-linux-with-examples/) mode.

*mysql <batch-file>;*

*mysq <batch-file> mysql.out*

**7. How many different tables are present in MySQL?**

There are 5 types of tables present in MySQL.

* [Heap](https://www.geeksforgeeks.org/heap-sort/) table
* [merge](https://www.geeksforgeeks.org/merge-sort/) table
* MyISAM table
* INNO DB table
* ISAM table

**8. What are the differences between CHAR and VARCHAR data types in MySQL?**

* Storage and retrieval have been different for CHAR and VARCHAR.
* Column length is fixed in CHAR but VARCHAR length is variable.
* CHAR is faster than VARCHAR.
* CHAR datatype can hold a maximum of 255 characters while VARCHAR can store up to 4000 characters.

**9. What is Difference between CHAR\_LENGTH and LENGTH?**

[LENGTH](https://www.geeksforgeeks.org/length-function-in-mysql/) is byte count whereas [CHAR\_LENGTH](https://www.geeksforgeeks.org/char_length-function-in-mysql/) is character count. The numbers are the same for Latin characters but different for Unicode and other encodings.

**Syntax of CHAR\_LENGTH:**

*SELECT CHAR\_LENGTH(column\_name) FROM table\_name;*

**Syntax of LENGTH:**

*SELECT LENGTH(column\_name) FROM table\_name;*

**10. What do you understand by % and \_ in the like statement?**

‘\_’ corresponds to only one character but ‘%’ corresponds to zero or more characters in the [LIKE](https://www.geeksforgeeks.org/sql-like/) statement.

**11. How many index columns can be created in a table?**

There are **16** indexed columns can be created in a table.

**12. What are string types available for columns?**

There are six string types available for the column.

* [SET](https://www.geeksforgeeks.org/sets-in-javascript/)
* [BOLB](https://www.geeksforgeeks.org/blob-full-form/)
* TEXT
* [ENUM](https://www.geeksforgeeks.org/enumeration-enum-c/)
* CHAR
* VARCHAR

**13. Explain the main difference between FLOAT and DOUBLE?**

* [FLOAT](https://www.geeksforgeeks.org/css-float/) stored floating point number with 8 place accuracy. The size of FLOAT is 4 bytes.
* [DOUBLE](https://www.geeksforgeeks.org/difference-float-double-c-cpp/) also stored floating point numbers with 18 place accuracy. The size of DOUBLE is 8 bytes.

**14. Explain the differences between BOLB and TEXT.**

**BOLB:**

A [BOLB](https://www.geeksforgeeks.org/retrieve-image-and-file-stored-as-a-blob-from-mysql-table-using-python/) is a large object in binary form that can hold a variable amount of data. Sorting and comparing in BLOB values are case-sensitive.

There are four types of BOLB.

* TINYBOLB
* BOLB
* MEDIUMBOLB
* LONGBOLB

**TEXT:**

[Sorting](https://www.geeksforgeeks.org/sorting-algorithms/) and comparison are performed in case-insensitive for TEXT values. we can also say a TEXT is case-insensitive BOLB.

There are four types of TEXT.

* TINYTEXT
* TEXT
* MEDIUMTEXT
* LONGTEXT

**15. Explain the difference between having and where clause in MySQL.**

* [WHERE](https://www.geeksforgeeks.org/sql-where-clause/) statement is used to filter rows but [HAVING](https://www.geeksforgeeks.org/sql-having-clause-with-examples/) statement is used to filter groups.
* [GROUP BY](https://www.geeksforgeeks.org/sql-group-by/) is not used with WHERE. HAVING clause is used with GROUP BY.

**16. Explain REGEXP.**

[REGEXP](https://www.geeksforgeeks.org/mysql-regular-expressions-regexp/) is a pattern match where the pattern is matched anywhere in the search value.

For more detail you refer to our [MySQL | Regular expressions](https://www.geeksforgeeks.org/mysql-regular-expressions-regexp/) Article.

**17. How can we add a column in MySQL?**

A **column** is a series of table cells that store a value for table’s each row. we can add column by using [ALTER TABLE](https://www.geeksforgeeks.org/sql-alter-add-drop-modify/)statement.

*ALTER TABLE tab\_name*

*ADD COLUMN col\_name col\_definition [FIRST|AFTER exist\_col];*

**18. How to delete columns in MySQL?**

We can remove columns in MySQL by using [ALTER TABLE](https://www.geeksforgeeks.org/sql-alter-rename/) statement.

**Syntax:**

***ALTER******TABLE****table\_name****DROP******COLUMN****column1, column2….;*

**19. How to delete a table in MySQL?**

We can delete a table by using [DROP TABLE](https://www.geeksforgeeks.org/hive-drop-table/) statement. This statement deletes complete data of table.

*DROP TABLE table-name;*

**20. How are mysql\_fetch\_array() and mysql\_fetch\_object() different from each another?**

mysql\_fetch\_array() Gets a result row as a related array or a regular [array](https://www.geeksforgeeks.org/array-data-structure/) from database. mysql\_fetch\_object gets a result row as an [object](https://www.geeksforgeeks.org/object-class-in-java/) from the database.

**21. How to get the top 10 rows?**

The following query will be used to get top 10 rows.

*SELECT \* FROM table\_name LIMIT 0,10;*

**22. How does NOW() differ from CURRENT\_DATE()?**

current year, month, and date with hours, minutes, and seconds is shown by using [NOW()](https://www.geeksforgeeks.org/now-function-in-mysql/) command while [CURRENT\_DATE](https://www.geeksforgeeks.org/current_date-function-in-mysql/) shows current year current month, and current date.

**Syntax:**

*SELECT NOW();*

*SELECT CURRENT\_DATE();*

**23. What is the use of the ‘DISTINCT’ keyword in MySQL?**

the [DISTINCT](https://www.geeksforgeeks.org/postgresql-distinct-on-expression/) keyword allows for the removal of all duplicate records and the retrieval of unique records.**The**DISTINCT keyword is used with the SELECT statement.

**Syntax:**

*SELECT DISTINCT colu1, colum2..*

*FROM table\_name;*

**24. Which storage engines are used in MySQL?**

[Storage engines](https://www.geeksforgeeks.org/how-to-manipulate-mysql-storage-engines/) are also called table types. Data is stored in a file using multiple techniques.

Below are some techniques.

* Locking Level
* [Indexing](https://www.geeksforgeeks.org/indexing-in-databases-set-1/)
* Storage mechanism
* Capabilities and functions

**25. How to create a table in MySQL?**

The [CREAT TABLE](https://www.geeksforgeeks.org/postgresql-create-table/) command will be used to create a table in MySQL.

**Syntax:**

*CREATE TABLE ‘Employee’ (‘Employee\_Name’ VARCHAR(128), ‘Employee\_ID’ VARCHAR(128), ‘Employee\_Salary’ VARCHAR(16), ‘Designation’ CHAR(4)) ;*

**26. How to insert data in MySQL table?**

We can add data to a table using the [INSERT INTO](https://www.geeksforgeeks.org/sql-insert-statement/) statement .

**Syntax:**

***INSERT******INTO****table\_name ( field1, field2, field3 )*

***VALUES****( value1, value2, value3 );*

**Intermediate MySQL Interview Questions**

**27. Write a statement to find duplicate rows In the MySQL table?**

The below statement is used to find duplicate rows.

*SELECT Table\_Name, Category*

*FROM Product*

*GROUP BY Name, Category*

*HAVING COUNT(id) > 1;*

**28. What types of relationships are used in MySQL?**

There are three types of [relationships](https://www.geeksforgeeks.org/recursive-relationships-in-er-diagrams/) used in MySQL.

**One-to-one:** Elements with a [one to one](https://www.geeksforgeeks.org/hibernate-one-to-one-mapping/) relationship can be included as columns in the table.

**One-to-many:** One to many or many to one relationships both are same. It will occur when one row in a table is related to multiple rows in different table.

**Many-to-many:**Many rows in a table are related to many rows in different table is called many to many relationship.

**29. How to insert Date in MySQL?**

We can use INSERT statement to insert date in MySQL table. MySQL default date format is YYYY-MM-DD. Automatic MySQL consist many data types to store dates.

* DATE
* DATETIME
* TIMESTAMP
* YEAR

**Syntax:**

***INSERT******INTO****table\_name (column\_name, column\_date)****VALUES****(‘DATE: Manual Date’, ‘2023-5-20’);*

**30. What is join? Tell different join in MySQL.**

[Joins](https://www.geeksforgeeks.org/sql-join-set-1-inner-left-right-and-full-joins/) are used to connect two or more tables. It returns only same values in all tables.

There are four different ways to join MySQL tables.

* Inner Join
* left Join
* Right Join
* Full Join

**31. What is a primary key? How to drop the primary key in MySQL?**

A primary key in MySQL is a single field or a group of fields that are used to uniquely identify each record in a table. A primary key cannot be null or empty. [ALTER TABLE](https://www.geeksforgeeks.org/sql-alter-add-drop-modify/) statement is used to delete a primary key from a table.

**Syntax:**

***ALTER******TABLE****table\_name****DROP******PRIMARY******KEY****;*

**32. What is a heap table in MySQL?**

A [heap](https://www.geeksforgeeks.org/heap-data-structure/) table is usually used for temporary and fast temporary storage.

* BOLB or TEXT fields are not permitted in the heap table.
* [comparison operator](https://www.geeksforgeeks.org/javascript-comparison-operators/) like =, <,>, = >,=< can be used only.
* Heap table didn’t support the AUTO\_INCREMENT command.
* Indexes should be NOT NULL in the heap table.

**33. What is the main difference between the primary key and the candidate key?**

The primary key uniquely identified each row of a table. only one primary key is available for a table.

* A primary is also a [candidate key](https://www.geeksforgeeks.org/difference-between-primary-and-candidate-key/).
* Candidate key that can be used for all [foreign key](https://www.geeksforgeeks.org/postgresql-foreign-key/) references.

For mor detail you can see: [Difference between Primary and Candidate Key](https://www.geeksforgeeks.org/difference-between-primary-and-candidate-key/)

**34. What is the difference between DELETE and TRUNCATE commands in MySQL?**

[**DELETE**](https://www.geeksforgeeks.org/sql-delete-statement/)Command is used to delete rows from the table depending on given the condition. [TRUNCATE](https://www.geeksforgeeks.org/sql-drop-truncate/) command is used to DELETE all rows from the table. DELETE command is a [Data Manipulation](https://www.geeksforgeeks.org/data-manipulattion-in-python-using-pandas/) Language command. TRUNCATE command is a Data Definition Language command.

For More detail you can see : [Difference between DELETE and TRUNCATE](https://www.geeksforgeeks.org/difference-between-delete-and-truncate/)

**35. What is InnoDB?**

A SQL storage database is called InnoDB database. The InnoDB offers [ACID transactions](https://www.geeksforgeeks.org/acid-properties-in-dbms/), row-level locking, and foreign key support. InnoDB is owned by Oracle Corporation.

**36. What is the difference between UNION and UNION ALL in MySQL?**

During combining the results of more than one SELECT statement, the [UNION](https://www.geeksforgeeks.org/union-and-union-all-in-ms-sql-server/) operator deletes duplicate rows between the various SELECT statements. The [UNION ALL](https://www.geeksforgeeks.org/union-union_all-functions-in-dplyr-package-in-r/) also combines the result set of more than one SELECT statement, but it does not delete duplicate rows.

**37. What is a ‘timestamp’ in MySQL?**

In MySQL, When a row is added to or updated in a table, a data type “[timestamp](https://www.geeksforgeeks.org/get-current-timestamp-using-python/)” automatically records the time.

**38. What is the use of ENUMs in MySQL?**

ENUM is a string [object](https://www.geeksforgeeks.org/objects-in-javascript/) that can be used when creating tables to specify a set of predefined values.

*CREATE table size(name ENUM(‘Small’, ‘Medium’, ‘Large’);*

For more detail refer to those article on [Enumerator (Enum) in MySQL](https://www.geeksforgeeks.org/enumerator-enum-in-mysql/)

**39. How can you control max size of heap in MySQL?**

MySQL config variable *max\_heap\_table\_size* can be used to control the max size of [heap](https://www.geeksforgeeks.org/heap-sort/).

**Syntax:**

*SET max\_heap\_table\_size = M*

**40. What is a view? How to create a view?**

A database object that has no value is called view. Rows and columns exist in a view. A view is virtual table. it is created by combining one or more tables. The difference of a view and a table is that views are definition that build on other tables. If the underlying table changes, the View will also reflect those same changes.

The below syntax is used to create a view.

**Syntax:**

***CREATE******VIEW****view\_name****AS***

***SELECT****columns*

***FROM****tables*

*[****WHERE****conditions];*

**41. Where MyISAM table will be stored and also give MyISAM formats of storage?**

Every MyISAM table is stored on [disk](https://www.geeksforgeeks.org/c-look-disk-scheduling-algorithm/).

There are three storage formats can be used .

* The .frm file can be used to store table definition.
* The .MYD( MYData) extension can be used for data files.
* The .MYI(MYIndex) extension can be used to Index files.

**42. How can we save images in MySQL?**

In MySQL, Blobs can be used to store images. All [database images](https://www.geeksforgeeks.org/how-to-upload-image-into-database-and-display-it-using-php/) are first converted into blobs then saved and then they will be added to the database, and finally, it will later be stored on the disk.

**43. What are trigger and how many TRIGGERS are available in MySQL table?**

A [trigger](https://www.geeksforgeeks.org/sql-trigger-student-database/) is a procedural code in a database. Triggers are automatically triggered when specific events occur on a particular table. During column updating triggers are invoked automatically.

SIX triggers are available in MySQL table.

* BEFORE INSERT
* AFTER INSERT
* BEFORE UPDATE
* AFTER UPDATE
* BEFORE DELETE
* AFTER DELETE

For more detail you can see: [Different types of MySQL Triggers (with examples) – GeeksforGeeks](https://www.geeksforgeeks.org/different-types-of-mysql-triggers-with-examples/)

**44. What are the different characteristics of MySQL MyISAM Static and MyISAM Dynamic?**

* Width of all fields is fixed in MyISAM [Static](https://www.geeksforgeeks.org/static-keyword-cpp/) table whereas width of all fields is not fixed in MyISAM [Dynamic](https://www.geeksforgeeks.org/dynamic-programming/). In MyISAM Dynamic table width will be like TEXT, BOLD, etc.
* In case of corruption MyISAM static table is easy to store.

**Advanced MySQL Interview Questions**

**45. What are Access Control Lists?**

A list of permissions known as an [Access Control List](https://www.geeksforgeeks.org/access-lists-acl/) is connected to an object. It is MySQL server security model helps in troubleshooting issues like users being unable to connect. MySQL holds the ACL’s cached in memory. ACL’s also called grant tables. MySQL verifies the authentication data and permissions against the ACLs. It predetermined order whenever a user tries to log in or execute a command.

**46. What is Normalization and list the different types of normalization?**

[**Normalization**](https://www.geeksforgeeks.org/introduction-of-database-normalization/) is used to avoid duplication and redundancy. it is a process of organizing data. There are many normal forms of normalization. which are also called successive levels. The first three regular forms are sufficient.

**First Normal Form (1NF):** There are no repeating groups within rows.

**Second Normal form(2NF):** Value of every supporting column depending on the whole primary key.

**Third Normal Form(3NF):**It depends only on the *primary key* and no other value of non-key column.

**47. What are various ways to create an index?**

There are many options to create an index as below:

* [T-SQL](https://www.geeksforgeeks.org/difference-between-t-sql-and-pl-sql/) statements can be used to create an index.
* The SQL Server Management Studio is available for use. we can use this to browse to the table where the index will be created, and then right-click on the Indexes node. We must select the New Index option over here.
* We can identify the index indirectly by specifying the [PRIMARY KEY](https://www.geeksforgeeks.org/difference-between-primary-key-and-unique-key/) and the [UNIQUE](https://www.geeksforgeeks.org/postgresql-unique-index/) constraint in the [CREATE TABLE](https://www.geeksforgeeks.org/python-sqlite-create-table/) or [ALTER TABLE](https://www.geeksforgeeks.org/sql-alter-add-drop-modify/) statement.

**48. What are a clustered index and a non clustered index?**

**Cluster Index:** An index type used to arrange data in a table is called a [clustered index](https://www.geeksforgeeks.org/clustering-indexing-in-databases/). The table’s data are stored in a specific order based on the clustered index.

**Non Cluster Index:** A [non-clustered index](https://www.geeksforgeeks.org/sql-queries-on-clustered-and-non-clustered-indexes/) is also a type of index used to organize data in a table. The table’s data are not stored in a specific order based on the non clustered index.

For more details, Check our latest article on [Difference between Clustered and Non-clustered index](https://www.geeksforgeeks.org/difference-between-clustered-and-non-clustered-index/).

**49. How to validate emails using a single query?**

We can use the [regular expressions function](https://www.geeksforgeeks.org/regular-expressions-in-c/) (REGEXP\_LIKE) to validate emails. Below is the example of validate emails using a single query.

***SELECT***

*Email*

***FROM***

*Vehicle*

***where****NOT REGEXP\_LIKE(Email, ‘[A-Z0-9.\_%+-]+@[A-Z0-9.-]+.[A-Z]{2,4}’, ‘i’);*

For detail you can check our latest article on [Regular expressions (Regexp)](https://www.geeksforgeeks.org/mysql-regular-expressions-regexp/).

**50. How can you handle the –secure-file-priv in MySQL?**

The MySQL Server is restricted from loading directories using the [LOAD DATA INFILE](https://www.geeksforgeeks.org/how-to-import-timestamp-from-a-csv-file-in-mysql/) command by the -secure-file-priv option. Use the SHOW VARIABLES LIKE “secure\_file\_priv” command to view the directory that has been configured.

There are two options to handle as below.

* Either transfer your file to the directory that secure-file-priv specifies.
* Or you can turn off secure-file-priv. This must be removed at the beginning and cannot be disabled later.