**SQL Interview Questions**

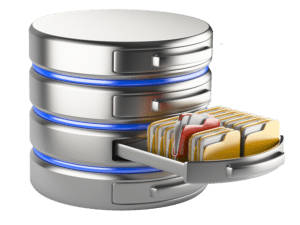
**Q1. What is the difference between DELETE and TRUNCATE statements?**

|  |  |
| --- | --- |
| **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. |

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

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

A **Database Management System** (**DBMS**) is a  software application that interacts with the user, applications and the database itself to capture and analyze 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 – Mongo

**Q4. 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

**Q5. What are joins in SQL?**

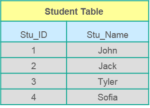
A JOIN clause is used to combine rows from two or more tables, based on a related column between them. It is used to merge two tables or retrieve data from there. There are 4 joins in SQL namely:

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

**Q6.** **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.

**Q7. What is a Primary key?**

* APrimary keyis 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.

**Q8.** **What are Constraints?**

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

**Q9. What is the difference between SQL and MySQL?**

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

**Q10. 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 a structured training from edureka! Click below to know more.

**Q11. What is a Foreign key?**

* 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 prevents actions that would destroy links between the child and parent tables.

**Q12. 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.

**Q13. What is the difference between clustered and non clustered index in SQL?**

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

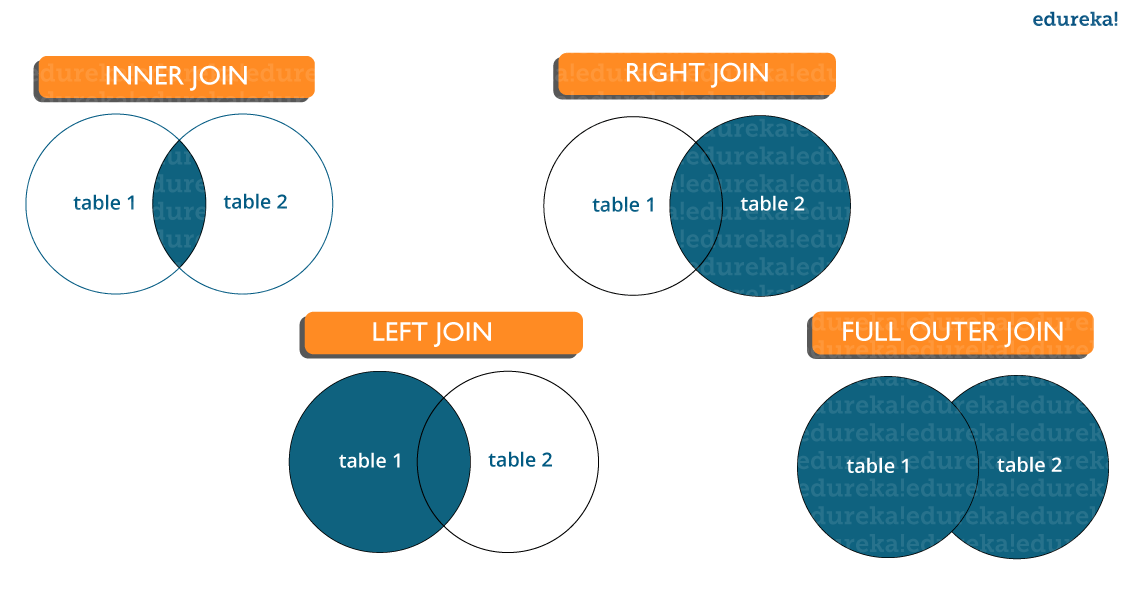
1. Clustered index is used for easy retrieval of data from the database and its faster whereas reading from non clustered index is relatively slower.
2. Clustered index alters the way records are stored in a database as it sorts out rows by the column which is set to be clustered index whereas in a non 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.

**Q14. 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.

**Q15. List the different type of joins?**

There are various types of joins which are used to retrieve data between the tables. There are four types of joins, namely:

**Inner join:** Inner Join in MySQL 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 MySQL 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.

*[Powered by Edureka](https://www.edureka.co/mymock-interview-service" \t "_blank)*[[](https://www.edureka.co/mymock-interview-service)](https://www.edureka.co/mymock-interview-service" \t "_blank)

**[80% INTERVIEW REJECTIONS HAPPEN IN FIRST 90 SECONDS](https://www.edureka.co/mymock-interview-service" \t "_blank)**

**[Take MYSQL Mock Interview](https://www.edureka.co/mymock-interview-service" \t "_blank)**

* **[Get Interviewed by Industry Experts](https://www.edureka.co/mymock-interview-service" \t "_blank)**
* **[Personalized interview feedback](https://www.edureka.co/mymock-interview-service" \t "_blank)**

[BOOK A SLOT](https://www.edureka.co/mymock-interview-service" \t "_blank)

**Right Join:** Right Join in MySQL 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.

**Q16. What do you mean by Denormalization?**

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.

**Q17. 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. 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).

**Q18. 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.

**Q19. Explain different types of index.**

There are three types of index 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.

**Q20. What is Normalization and what are the advantages of it?**

Normalization 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 a structured training from edureka! Click below to know more.

**Q21. What is the difference between DROP and TRUNCATE commands?**

DROP command removes a table and it cannot be rolled back from the database whereas TRUNCATE command removes all the rows from the table.

**Q22. 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.

* *First Normal Form (1NF)* – No repeating groups within rows
* *Second Normal Form (2NF)* – Every non-key (supporting) column value is dependent on the whole primary key.
* *Third Normal Form (3NF)* – Dependent solely on the primary key and no other non-key (supporting) column value.

**Q23. What is 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.

**Q24. What do you mean by “Trigger” in SQL?**

Trigger 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.

**Q25. 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

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.

**Q26.  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.

**Q27. 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.

**Q28. 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 =.

**Q29. 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.

**Q30. List the ways to get the count of records in a table?**

To count the number of records in a table, 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 a structured training from edureka! Click below to know more.

**Q31. 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%' |

**Q32. 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; |

**Q33. What is the need for group functions in SQL?**

*[Powered by Edureka](https://www.edureka.co/mymock-interview-service" \t "_blank)*[[](https://www.edureka.co/mymock-interview-service)](https://www.edureka.co/mymock-interview-service" \t "_blank)

**[MYSQL Mock interviews for you](https://www.edureka.co/mymock-interview-service" \t "_blank)**

* **[Interviews by Industry Experts](https://www.edureka.co/mymock-interview-service" \t "_blank)**
* **[Personalized detailed interview feedback](https://www.edureka.co/mymock-interview-service" \t "_blank)**
* **[Access to exclusive and curated content](https://www.edureka.co/mymock-interview-service" \t "_blank)**

[SCHEDULE NOW](https://www.edureka.co/mymock-interview-service" \t "_blank)

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

**Q34 . 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.

**Q35.  How can you insert NULL values in a column while inserting the data?**

NULL values can be inserted in the following ways:

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

**Q36. 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);

**Q37. Why are SQL functions used?**

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

**Q38. What is the need of 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.

**Q39. 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.

**Q40. 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.

**Q41. 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.

**Q42. 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.

**Q43. 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

**Q44. How can you fetch common records from two tables?**

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

|  |  |
| --- | --- |
| 1 | Select studentID from student. <strong>INTERSECT </strong> Select StudentID from Exam |

**Q45. 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.

**Q46. What are the different set operators available in SQL?**

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

**Q47. What is an ALIAS command?**

ALIAS name 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.

**Q48. 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.

**Q49. 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

**Q50. 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 a structured training from edureka! Click below to know more.

**Q51. How can you select unique records from a table?**

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

*[Powered by Edureka](https://www.edureka.co/mymock-interview-service" \t "_blank)*[[](https://www.edureka.co/mymock-interview-service)](https://www.edureka.co/mymock-interview-service" \t "_blank)

**[NEED HELP FOR YOUR UPCOMING INTERVIEW?](https://www.edureka.co/mymock-interview-service" \t "_blank)**

**[Take MYSQL Mock Interview](https://www.edureka.co/mymock-interview-service" \t "_blank)**

* **[Get Interviewed by Industry Experts](https://www.edureka.co/mymock-interview-service" \t "_blank)**
* **[Personalized interview feedback](https://www.edureka.co/mymock-interview-service" \t "_blank)**

[BOOK A SLOT](https://www.edureka.co/mymock-interview-service" \t "_blank)

Select DISTINCT studentID from Student

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

**Q52. 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

**Q53**. **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.

**Q54. 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.

**Q55. 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.

**Q56. 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.

**Q57. 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.

**Q58. 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.

**Q59. 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.

**Q60. 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 a structured training from edureka! Click below to know more.

**Q61. 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.

**Q62. What is Auto Increment in SQL?**

Auto increment 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 is used.

AUTO INCREMENT keyword can be used in Oracle and IDENTITY keyword can be used in SQL SERVER.

**Q63. 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.

**Q64. 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.

**Q65. 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)

**1. What is Database?**

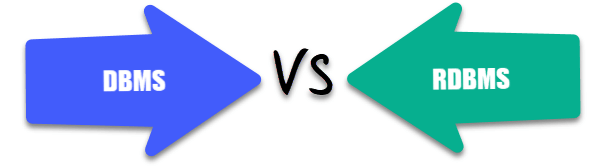
A database is an organized collection of data, stored and retrieved digitally from a remote or local computer system. Databases can be vast and complex, and such databases are developed using fixed design and modeling approaches.

**2. What is DBMS?**

DBMS stands for Database Management System. DBMS is a system software responsible for the creation, retrieval, updation and management of the database. It ensures that our data is consistent, organized and is easily accessible by serving as an interface between the database and its end users or application softwares.

**3. What is RDBMS? How is it different from DBMS?**

RDBMS stands for Relational Database Management System. The key difference here, compared to DBMS, is that RDBMS stores data in the form of a collection of tables and relations can be defined between the common fields of these tables. Most modern database management systems like MySQL, Microsoft SQL Server, Oracle, IBM DB2 and Amazon Redshift are based on RDBMS.

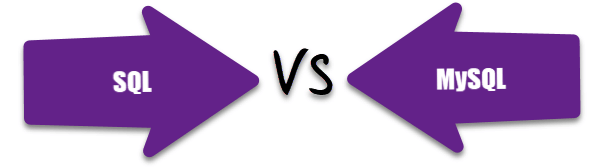


**4. What is SQL?**

SQL stands for Structured Query Language. It is the standard language for relational database management systems. It is especially useful in handling organized data comprised of entities (variables) and relations between different entities of the data.

**5. What is the difference between SQL and MySQL?**

SQL is a standard language for retrieving and manipulating structured databases. On the contrary, MySQL is a relational database management system, like SQL Server, Oracle or IBM DB2, that is used to manage SQL databases.



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

A table is an organized collection of data stored in the form of rows and columns. Columns can be categorized as vertical and rows as horizontal. The columns in a table are called fields while the rows can be referred to as records.

**7. What are Constraints in SQL?**

Constraints are used to specify the rules concerning data in the table. It can be applied for single or multiple fields in an SQL table during creation of table or after creationg using the ALTER TABLE command. The constraints are:

* **NOT NULL**- Restricts NULL value from being inserted into a column.
* **CHECK**- Verifies that all values in a field satisfy a condition.
* **DEFAULT**- Automatically assigns a default value if no value has been specified for the field.
* **UNIQUE**- Ensures unique values to be inserted into the field.
* **INDEX**- Indexes a field providing faster retrieval of records.
* **PRIMARY KEY**- Uniquely identifies each record in a table.
* **FOREIGN KEY**- Ensures referential integrity for a record in another table.

**8. What is a Primary Key?**

The PRIMARY KEY constraint uniquely identifies each row in a table. It must contain UNIQUE values and has an implicit NOT NULL constraint.   
A table in SQL is strictly restricted to have one and only one primary key, which is comprised of single or multiple fields (columns).

**CREATE** **TABLE** Students ( */\* Create table with a single field as primary key \*/*

ID INT **NOT** **NULL**

Name VARCHAR(255)

**PRIMARY** **KEY** (ID)

);

**CREATE** **TABLE** Students ( */\* Create table with multiple fields as primary key \*/*

ID INT **NOT** **NULL**

LastName VARCHAR(255)

FirstName VARCHAR(255) **NOT** **NULL**,

**CONSTRAINT** PK\_Student

**PRIMARY** **KEY** (ID, FirstName)

);

**ALTER** **TABLE** Students */\* Set a column as primary key \*/*

**ADD** **PRIMARY** **KEY** (ID);

**ALTER** **TABLE** Students */\* Set multiple columns as primary key \*/*

**ADD** **CONSTRAINT** PK\_Student */\*Naming a Primary Key\*/*

**PRIMARY** **KEY** (ID, FirstName);

Q   =>   Write a SQL statement to add PRIMARY KEY 't\_id' to the table 'teachers'.



Q   =>   Write a SQL statement to add primary key constraint 'pk\_a' for table 'table\_a' and fields 'col\_b, col\_c'.



**9. What is a UNIQUE constraint?**

A UNIQUE constraint ensures that all values in a column are different. This provides uniqueness for the column(s) and helps identify each row uniquely. Unlike primary key, there can be multiple unique constraints defined per table. The code syntax for UNIQUE is quite similar to that of PRIMARY KEY and can be used interchangeably.

**CREATE** **TABLE** Students ( */\* Create table with a single field as unique \*/*

ID INT **NOT** **NULL** **UNIQUE**

Name VARCHAR(255)

);

**CREATE** **TABLE** Students ( */\* Create table with multiple fields as unique \*/*

ID INT **NOT** **NULL**

LastName VARCHAR(255)

FirstName VARCHAR(255) **NOT** **NULL**

**CONSTRAINT** PK\_Student

**UNIQUE** (ID, FirstName)

);

**ALTER** **TABLE** Students */\* Set a column as unique \*/*

**ADD** **UNIQUE** (ID);

**ALTER** **TABLE** Students */\* Set multiple columns as unique \*/*

**ADD** **CONSTRAINT** PK\_Student */\* Naming a unique constraint \*/*

**UNIQUE** (ID, FirstName);

**10. What is a Foreign Key?**

A FOREIGN KEY comprises of single or collection of fields in a table that essentially refer to the PRIMARY KEY in another table. Foreign key constraint ensures referential integrity in the relation between two tables.   
The table with the foreign key constraint is labelled as the child table, and the table containing the candidate key is labelled as the referenced or parent table.

**CREATE** **TABLE** Students ( */\* Create table with foreign key - Way 1 \*/*

ID INT **NOT** **NULL**

Name VARCHAR(255)

LibraryID INT

**PRIMARY** **KEY** (ID)

**FOREIGN KEY** (Library\_ID) **REFERENCES** Library(LibraryID)

);

**CREATE** **TABLE** Students ( */\* Create table with foreign key - Way 2 \*/*

ID INT **NOT NULL PRIMARY KEY**

Name VARCHAR(255)

LibraryID INT **FOREIGN KEY** (Library\_ID) **REFERENCES** Library(LibraryID)

);

**ALTER** **TABLE** Students */\* Add a new foreign key \*/*

**ADD** **FOREIGN** **KEY** (LibraryID)

**REFERENCES** Library (LibraryID);

Q   =>   What type of integrity constraint does the foreign key ensure?

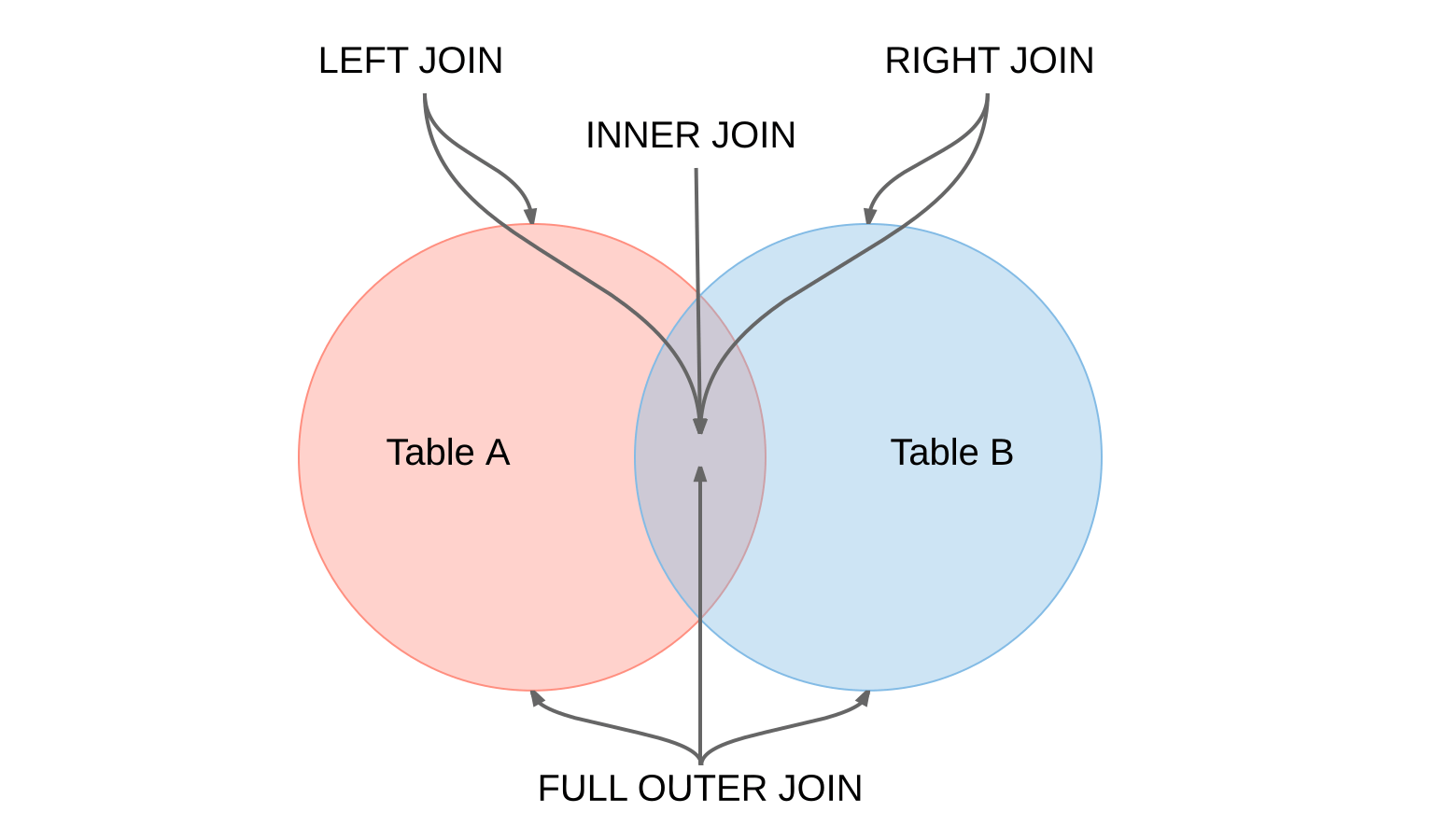


Q   =>   Write a SQL statement to add a FOREIGN KEY 'col\_fk' that references 'col\_pk' in 'table\_x'.



**11. What is a Join? List its different types.**

The SQL Join clause is used to combine records (rows) from two or more tables in a SQL database based on a related column between the two.



There are four different types of JOINs in SQL:

* **(INNER) JOIN**: Retrieves records that have matching values in both tables involved in the join. This is the widely used join for queries.
* **SELECT** \*
* **FROM** Table\_A
* **JOIN** Table\_B;
* **SELECT** \*
* **FROM** Table\_A
* **INNER JOIN** Table\_B;
* **LEFT (OUTER) JOIN**: Retrieves all the records/rows from the left and the matched records/rows from the right table.
* **SELECT** \*
* **FROM** Table\_A A
* **LEFT JOIN** Table\_B B
* **ON** A.col = B.col;
* **RIGHT (OUTER) JOIN**: Retrieves all the records/rows from the right and the matched records/rows from the left table.
* **SELECT** \*
* **FROM** Table\_A A
* **RIGHT JOIN** Table\_B B
* **ON** A.col = B.col;
* **FULL (OUTER) JOIN**: Retrieves all the records where there is a match in either the left or right table.
* **SELECT** \*
* **FROM** Table\_A A
* **FULL JOIN** Table\_B B
* **ON** A.col = B.col;

**12. What is a Self-Join?**

A **self JOIN**is a case of regular join where a table is joined to itself based on some relation between its own column(s). Self-join uses the INNER JOIN or LEFT JOIN clause and a table alias is used to assign different names to the table within the query.

**SELECT** A.emp\_id **AS** "Emp\_ID",A.emp\_name **AS** "Employee",

B.emp\_id **AS** "Sup\_ID",B.emp\_name **AS** "Supervisor"

**FROM** employee A, employee B

**WHERE** A.emp\_sup = B.emp\_id;

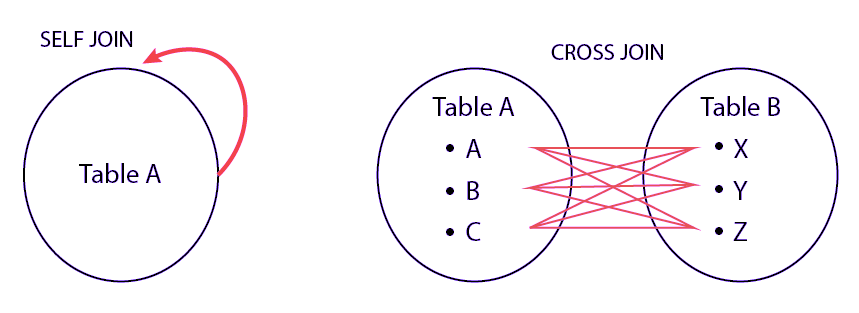
**13. What is a Cross-Join?**

Cross join can be defined as a cartesian product of the two tables included in the join. The table after join contains the same number of rows as in the cross-product of number of rows in the two tables. If a WHERE clause is used in cross join then the query will work like an INNER JOIN.

**SELECT** stu.name, sub.subject

**FROM** students **AS** stu

**CROSS** **JOIN** subjects **AS** sub;



Q   =>   Write a SQL statement to CROSS JOIN 'table\_1' with 'table\_2' and fetch 'col\_1' from table\_1 & 'col\_2' from table\_2 respectively. Do not use alias.



Q   =>   Write a SQL statement to perform SELF JOIN for 'Table\_X' with alias 'Table\_1' and 'Table\_2', on columns 'Col\_1' and 'Col\_2' respectively.



**14. What is an Index? Explain its different types.**

A database index is a data structure that provides quick lookup of data in a column or columns of a table. It enhances the speed of operations accessing data from a database table at the cost of additional writes and memory to maintain the index data structure.

**CREATE** **INDEX** index\_name */\* Create Index \*/*

**ON** table\_name (column\_1, column\_2);

**DROP** **INDEX** index\_name; */\* Drop Index \*/*

There are different types of indexes that can be created for different purposes:

* **Unique and Non-Unique Index**:

Unique indexes are indexes that help maintain data integrity by ensuring that no two rows of data in a table have identical key values. Once a unique index has been defined for a table, uniqueness is enforced whenever keys are added or changed within the index.

**CREATE** **UNIQUE** **INDEX** myIndex

**ON** students (enroll\_no);

Non-unique indexes, on the other hand, are not used to enforce constraints on the tables with which they are associated. Instead, non-unique indexes are used solely to improve query performance by maintaining a sorted order of data values that are used frequently.

* **Clustered and Non-Clustered Index**:

Clustered indexes are indexes whose order of the rows in the database correspond to the order of the rows in the index. This is why only one clustered index can exist in a given table, whereas, multiple non-clustered indexes can exist in the table.

The only difference between clustered and non-clustered indexes is that the database manager attempts to keep the data in the database in the same order as the corresponding keys appear in the clustered index.

Clustering index can improve the performance of most query operations because they provide a linear-access path to data stored in the database.

Q   =>   Write a SQL statement to create a UNIQUE INDEX "my\_index" on "my\_table" for fields "column\_1" & "column\_2".



**15. What is the difference between Clustered and Non-clustered index?**

As explained above, the differences can be broken down into three small factors -

1. Clustered index **modifies the way**records are stored in a database based on the indexed column. Non-clustered index creates a separate entity within the table which references the original table.
2. Clustered index is used for easy and **speedy retrieval**of data from the database, whereas, fetching records from the non-clustered index is relatively slower.
3. In SQL, a table can have **a single**clustered index whereas it can have multiple non-clustered indexes.

**16. 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.

**17. What is a Query?**

A query is a request for data or information from a database table or combination of tables. A database query can be either a select query or an action query.

**SELECT** fname, lname */\* select query \*/*

**FROM** myDb.students

**WHERE** student\_id = 1;

**UPDATE** myDB.students */\* action query \*/*

**SET** fname = 'Captain', lname = 'America'

**WHERE** student\_id = 1;

**18. What is a Subquery? What are its types?**

A subquery is a query within another query, also known as **nested query**or **inner query**. It is used to restrict or enhance the data to be queried by the main query, thus restricting or enhancing the output of the main query respectively. For example, here we fetch the contact information for students who have enrolled for the maths subject:

**SELECT** name, email, mob, address

**FROM** myDb.contacts

**WHERE** roll\_no **IN** (

**SELECT** roll\_no

**FROM** myDb.students

**WHERE** subject = 'Maths');

There are two types of subquery - **Correlated** and **Non-Correlated**.

* A **correlated**subquery cannot be considered as an independent query, but it can refer the column in a table listed in the FROM of the main query.
* A **non-correlated**subquery can be considered as an independent query and the output of subquery is substituted in the main query.

Q   =>   Write a SQL query to update the field "status" in table "applications" from 0 to 1.



Q   =>   Write a SQL query to select the field "app\_id" in table "applications" less than 1000.



Q   =>   Write a SQL query to fetch the field "app\_name" from "apps" where "apps.id" is equal to the above collection of "app\_id".



**19. What is the SELECT statement?**

SELECT operator in SQL is used to select data from a database. The data returned is stored in a result table, called the result-set.

**SELECT** \* **FROM** myDB.students;

**20. What are some common clauses used with SELECT query in SQL?**

Some common SQL clauses used in conjuction with a SELECT query are as follows:

* **WHERE** clause in SQL is used to filter records that are necessary, based on specific conditions.
* **ORDER BY** clause in SQL is used to sort the records based on some field(s) in ascending (**ASC**) or descending order (**DESC**).
* **SELECT** \*
* **FROM** myDB.students
* **WHERE** graduation\_year = 2019
* **ORDER** **BY** studentID **DESC**;
* **GROUP BY** clause in SQL is used to group records with identical data and can be used in conjuction with some aggregation functions to produce summarized results from the database.
* **HAVING** clause in SQL is used to filter records in combination with the GROUP BY clause. It is different from WHERE, since WHERE clause cannot filter aggregated records.
* **SELECT** **COUNT**(studentId), country
* **FROM** myDB.students
* **WHERE** country != "INDIA"
* **GROUP** **BY** country
* **HAVING** **COUNT**(studentID) > 5;

**21. What are UNION, MINUS and INTERSECT commands?**

The **UNION**operator combines and returns the result-set retrieved by two or more SELECT statements.   
The **MINUS**operator in SQL is used to remove duplicates from the result-set obtained by the second SELECT query from the result-set obtained by the first SELECT query and then return the filtered results from the first.   
The **INTERSECT**clause in SQL combines the result-set fetched by the two SELECT statements where records from one match the other and then returns this intersection of result-sets.

Certain conditions need to be met before executing either of the above statements in SQL -

* + Each SELECT statement within the clause must have the same number of columns
  + The columns must also have similar data types
  + The columns in each SELECT statement should necessarily have the same order

**SELECT** name **FROM** Students */\* Fetch the union of queries \*/*

**UNION**

**SELECT** name **FROM** Contacts;

**SELECT** name **FROM** Students */\* Fetch the union of queries with duplicates\*/*

**UNION ALL**

**SELECT** name **FROM** Contacts;

**SELECT** name **FROM** Students */\* Fetch names from students \*/*

**MINUS** */\* that aren't present in contacts \*/*

**SELECT** name **FROM** Contacts;

**SELECT** name **FROM** Students */\* Fetch names from students \*/*

**INTERSECT** */\* that are present in contacts as well \*/*

**SELECT** name **FROM** Contacts;

Q   =>   Write a SQL query to fetch "names" that are present in either table "accounts" or in table "registry".



Q   =>   Write a SQL query to fetch "names" that are present in "accounts" but not in table "registry".



Q   =>   Write a SQL query to fetch "names" from table "contacts" that are neither present in "accounts.name" nor in "registry.name".



**22. What is Cursor? How to use a Cursor?**

A database cursor is a control structure that allows for traversal of records in a database. Cursors, in addition, facilitates processing after traversal, such as retrieval, addition and deletion of database records. They can be viewed as a pointer to one row in a set of rows.

**Working with SQL Cursor**

* + **DECLARE** a cursor after any variable declaration. The cursor declaration must always be associated with a SELECT Statement.
  + Open cursor to initialize the result set. The **OPEN** statement must be called before fetching rows from the result set.
  + **FETCH** statement to retrieve and move to the next row in the result set.
  + Call the **CLOSE** statement to deactivate the cursor.
  + Finally use the **DEALLOCATE** statement to delete the cursor definition and release the associated resources.

**DECLARE** @**name** **VARCHAR**(50) */\* Declare All Required Variables \*/*

**DECLARE** db\_cursor **CURSOR** **FOR** */\* Declare Cursor Name\*/*

**SELECT** name

**FROM** myDB.students

**WHERE** parent\_name **IN** ('Sara', 'Ansh')

**OPEN** db\_cursor */\* Open cursor and Fetch data into @name \*/*

**FETCH** next

**FROM** db\_cursor

**INTO** @**name**

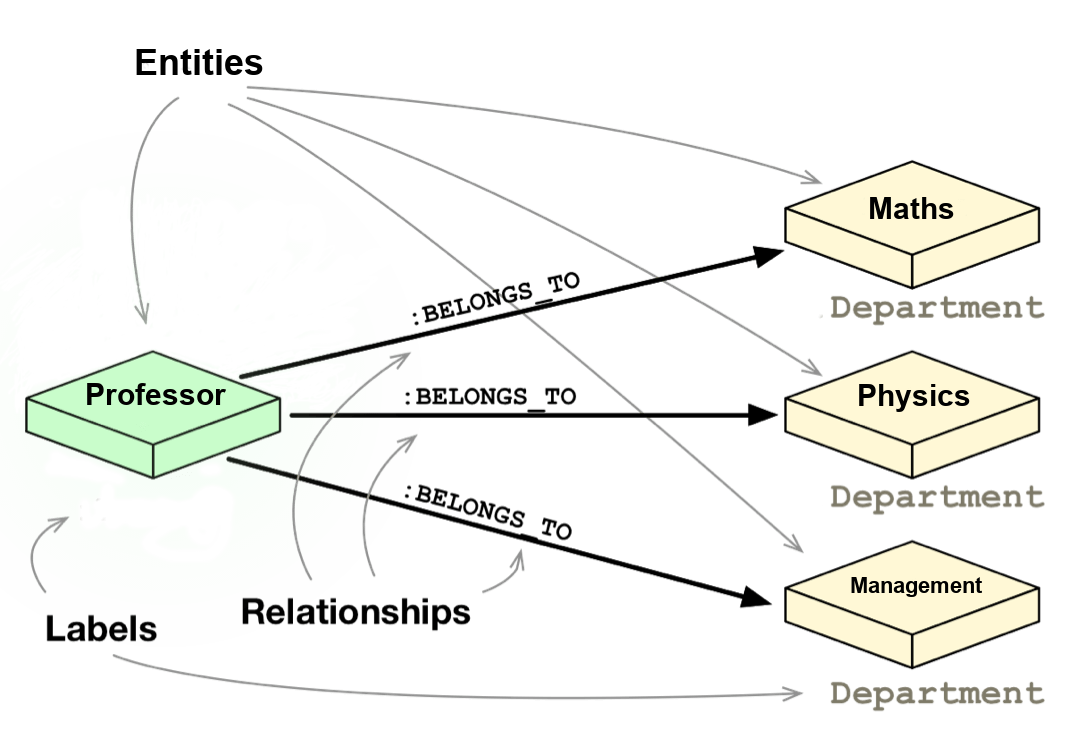
**CLOSE** db\_cursor */\* Close the cursor and deallocate the resources \*/*

**DEALLOCATE** db\_cursor

**23. What are Entities and Relationships?**

**Entity**: An entity can be a real-world object, either tangible or intangible, that can be easily identifiable. For example, in a college database, students, professors, workers, departments, and projects can be referred to as entities. Each entity has some associated properties that provide it an identity.

**Relationships**: Relations or links between entities that have something to do with each other. For example - The employees table in a company's database can be associated with the salary table in the same database.



**24. List the different types of relationships in SQL.**

* + **One-to-One** - This can be defined as the relationship between two tables where each record in one table is associated with the maximum of one record in the other table.
  + **One-to-Many** & **Many-to-One** - This is the most commonly used relationship where a record in a table is associated with multiple records in the other table.
  + **Many-to-Many** - This is used in cases when multiple instances on both sides are needed for defining a relationship.
  + **Self Referencing Relationships** - This is used when a table needs to define a relationship with itself.

**25. What is an Alias in SQL?**

An alias is a feature of SQL that is supported by most, if not all, RDBMSs. It is a temporary name assigned to the table or table column for the purpose of a particular SQL query. In addition, aliasing can be employed as an obfuscation technique to secure the real names of database fields. A table alias is also called a **correlation name**.

An alias is represented explicitly by the **AS** keyword but in some cases the same can be performed without it as well. Nevertheless, using the AS keyword is always a good practice.

**SELECT** A.emp\_name **AS** "Employee" */\* Alias using AS keyword \*/*

B.emp\_name **AS** "Supervisor"

**FROM** employee A, employee B */\* Alias without AS keyword \*/*

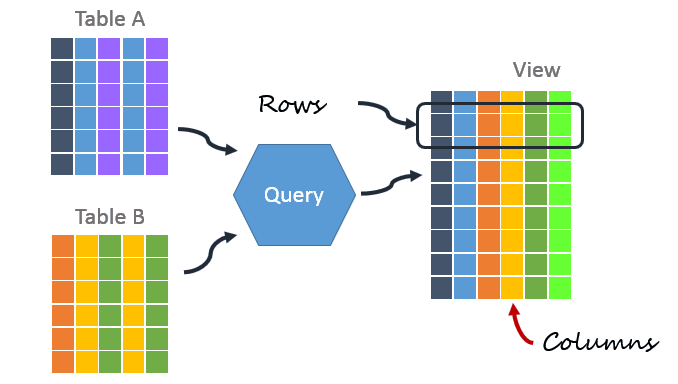
**WHERE** A.emp\_sup = B.emp\_id;

Q   =>   Write an SQL statement to select all from table "Limited" with alias "Ltd".



**26. What is a View?**

A view in SQL is a virtual table based on the result-set of an SQL statement. A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.



**27. What is Normalization?**

Normalization represents the way of organizing structured data in the database efficiently. It includes creation of tables, establishing relationships between them, and defining rules for those relationships. Inconsistency and redundancy can be kept in check based on these rules, hence, adding flexibility to the database.

**28. What is Denormalization?**

Denormalization is the inverse process of normalization, where the normalized schema is converted into a schema which has redundant information. The performance is improved by using redundancy and keeping the redundant data consistent. The reason for performing denormalization is the overheads produced in query processor by an over-normalized structure.

**29. What are the various forms of Normalization?**

Normal Forms are used to eliminate or reduce redundancy in database tables. The different forms are as follows:

* + **First Normal Form**

A relation is in first normal form if every attribute in that relation is a **single-valued attribute**. If a relation contains composite or multi-valued attribute, it violates the first normal form. Let's consider the following **students** table. Each student in the table, has a name, his/her address and the books they issued from the public library -

**Students Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student** | **Address** | **Books Issued** | **Salutation** |
| Sara | Amanora Park Town 94 | Until the Day I Die (Emily Carpenter),  Inception (Christopher Nolan) | Ms. |
| Ansh | 62nd Sector A-10 | The Alchemist (Paulo Coelho),  Inferno (Dan Brown) | Mr. |
| Sara | 24th Street Park Avenue | Beautiful Bad (Annie Ward),  Woman 99 (Greer Macallister) | Mrs. |
| Ansh | Windsor Street 777 | Dracula (Bram Stoker) | Mr. |

As we can observe, the Books Issued field has more than one values per record and to convert it into 1NF, this has to be resolved into separate individual records for each book issued. Check the following table in 1NF form -

**Students Table (1st Normal Form)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student** | **Address** | **Books Issued** | **Salutation** |
| Sara | Amanora Park Town 94 | Until the Day I Die (Emily Carpenter) | Ms. |
| Sara | Amanora Park Town 94 | Inception (Christopher Nolan) | Ms. |
| Ansh | 62nd Sector A-10 | The Alchemist (Paulo Coelho) | Mr. |
| Ansh | 62nd Sector A-10 | Inferno (Dan Brown) | Mr. |
| Sara | 24th Street Park Avenue | Beautiful Bad (Annie Ward) | Mrs. |
| Sara | 24th Street Park Avenue | Woman 99 (Greer Macallister) | Mrs. |
| Ansh | Windsor Street 777 | Dracula (Bram Stoker) | Mr. |

* + **Second Normal Form**

A relation is in second normal form if it satisfies the conditions for first normal form and does not contain any partial dependency. A relation in 2NF has **no partial dependency**, i.e., it has no non-prime attribute that depends on any proper subset of any candidate key of the table. Often, specifying a single column Primary Key is the solution to the problem. Examples -

* + - **Example 1** - Consider the above example. As we can observe, Students Table in 1NF form has a candidate key in the form of [Student, Address] that can uniquely identify all records in the table. The field Books Issued (non-prime attribute) depends partially on the Student field. Hence, the table is not in 2NF. To convert it into 2nd Normal Form, we will partition the tables into two while specifying a new ***Primary Key*** attribute to identify the individual records in the Students table. The ***Foreign Key***constraint will be set on the other table to ensure referential integrity.

**Students Table (2nd Normal Form)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student\_ID** | **Student** | **Address** | **Salutation** |
| 1 | Sara | Amanora Park Town 94 | Ms. |
| 2 | Ansh | 62nd Sector A-10 | Mr. |
| 3 | Sara | 24th Street Park Avenue | Mrs. |
| 4 | Ansh | Windsor Street 777 | Mr. |

**Books Table (2nd Normal Form)**

|  |  |
| --- | --- |
| **Student\_ID** | **Book Issued** |
| 1 | Until the Day I Die (Emily Carpenter) |
| 1 | Inception (Christopher Nolan) |
| 2 | The Alchemist (Paulo Coelho) |
| 2 | Inferno (Dan Brown) |
| 3 | Beautiful Bad (Annie Ward) |
| 3 | Woman 99 (Greer Macallister) |
| 4 | Dracula (Bram Stoker) |

* + - **Example 2** - Consider the following dependencies in relation R(W,X,Y,Z)

WX -> Y [W and X together determine Y]   
 XY -> Z [X and Y together determine Z]

Here, WX is the only candidate key and there is no partial dependency, i.e., any proper subset of WX doesn’t determine any non-prime attribute in the relation.

* + **Third Normal Form**

A relation is said to be in the third normal form, if it satisfies the conditions for second normal form and there is **no transitive dependency** between the non-prime attributes, i.e.,all non-prime attributes are determined only by the candidate keys of the relation and not by any other non-prime attribute.

* + - **Example 1** - Consider the Students Table in the above example. As we can observe, Students Table in 2NF form has a single candidate key Student\_ID (primary key) that can uniquely identify all records in the table. The field Salutation (non-prime attribute), however, depends on the Student Field rather than the candidate key. Hence, the table is not in 3NF. To convert it into 3rd Normal Form, we will once again partition the tables into two while specifying a new ***Foreign Key*** constraint to identify the salutations for individual records in the Students table. The ***Primary Key*** constraint for the same will be set on the Salutations table to identify each record uniquely.

**Students Table (3rd Normal Form)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student\_ID** | **Student** | **Address** | **Salutation\_ID** |
| 1 | Sara | Amanora Park Town 94 | 1 |
| 2 | Ansh | 62nd Sector A-10 | 2 |
| 3 | Sara | 24th Street Park Avenue | 3 |
| 4 | Ansh | Windsor Street 777 | 1 |

**Books Table (3rd Normal Form)**

|  |  |
| --- | --- |
| **Student\_ID** | **Book Issued** |
| 1 | Until the Day I Die (Emily Carpenter) |
| 1 | Inception (Christopher Nolan) |
| 2 | The Alchemist (Paulo Coelho) |
| 2 | Inferno (Dan Brown) |
| 3 | Beautiful Bad (Annie Ward) |
| 3 | Woman 99 (Greer Macallister) |
| 4 | Dracula (Bram Stoker) |

**Salutations Table (3rd Normal Form)**

|  |  |
| --- | --- |
| **Salutation\_ID** | **Salutation** |
| 1 | Ms. |
| 2 | Mr. |
| 3 | Mrs. |

* + - **Example 2** - Consider the following dependencies in relation R(P,Q,R,S,T)

P -> QR [P together determine C]   
 RS -> T [B and C together determine D]   
 Q -> S   
 T -> P

For the above relation to exist in 3NF, all possible candidate keys in above relation should be {P, RS, QR, T}.

* + **Boyce-Codd Normal Form**

A relation is in Boyce-Codd Normal Form if satisfies the conditions for third normal form and for every functional dependency, Left-Hand-Side is super key. In other words, a relation in BCNF has non-trivial functional dependencies in the form X –> Y, such that X is always a super key. For example - In the above example, Student\_ID serves as the sole unique identifier for the Students Table and Salutation\_ID for the Salutations Table, thus these tables exist in BCNF. Same cannot be said for the Books Table and there can be several books with common Book Names and same Student\_ID.

**30. What are the TRUNCATE, DELETE and DROP statements?**

**DELETE**statement is used to delete rows from a table.

**DELETE** **FROM** Candidates

**WHERE** CandidateId > 1000;

**TRUNCATE**command is used to delete all the rows from the table and free the space containing the table.

**TRUNCATE** **TABLE** Candidates;

**DROP**command is used to remove an object from the database. If you drop a table, all the rows in the table is deleted and the table structure is removed from the database.

**DROP** **TABLE** Candidates;

Q   =>   Write a SQL statement to wipe a table 'Temporary' from memory.



Q   =>   Write a SQL query to remove first 1000 records from table 'Temporary' based on 'id'.



Q   =>   Write a SQL statement to delete the table 'Temporary' while keeping its relations intact.



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

If a table is dropped, all things associated with the tables are dropped as well. This includes - the relationships defined on the table with other tables, the integrity checks and constraints, access privileges and other grants that the table has. To create and use the table again in its original form, all these relations, checks, constraints, privileges and relationships need to be redefined. However, if a table is truncated, none of the above problems exist and the table retains its original structure.

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

The TRUNCATE command is used to delete all the rows from the table and free the space containing the table.   
The DELETE command deletes only the rows from the table based on the condition given in the where clause or deletes all the rows from the table if no condition is specified. But it does not free the space containing the table.

**33. What are Aggregate and Scalar functions?**

An aggregate function performs operations on a collection of values to return a single scalar value. Aggregate functions are often used with the GROUP BY and HAVING clauses of the SELECT statement. Following are the widely used SQL aggregate functions:

* + **AVG()**- Calculates the mean of a collection of values.
  + **COUNT()**- Counts the total number of records in a specific table or view.
  + **MIN()**- Calculates the minimum of a collection of values.
  + **MAX()**- Calculates the maximum of a collection of values.
  + **SUM()**- Calculates the sum of a collection of values.
  + **FIRST()**- Fetches the first element in a collection of values.
  + **LAST()**- Fetches the last element in a collection of values.

*Note: All aggregate functions described above ignore NULL values except for the COUNT function.*

A **scalar function**returns a single value based on the input value. Following are the widely used SQL scalar functions:

* + **LEN()**- Calculates the total length of the given field (column).
  + **UCASE()**- Converts a collection of string values to uppercase characters.
  + **LCASE()**- Converts a collection of string values to lowercase characters.
  + **MID()**- Extracts substrings from a collection of string values in a table.
  + **CONCAT()**- Concatenates two or more strings.
  + **RAND()**- Generates a random collection of numbers of given length.
  + **ROUND()**- Calculates the round off integer value for a numeric field (or decimal point values).
  + **NOW()**- Returns the current data & time.
  + **FORMAT()**- Sets the format to display a collection of values.

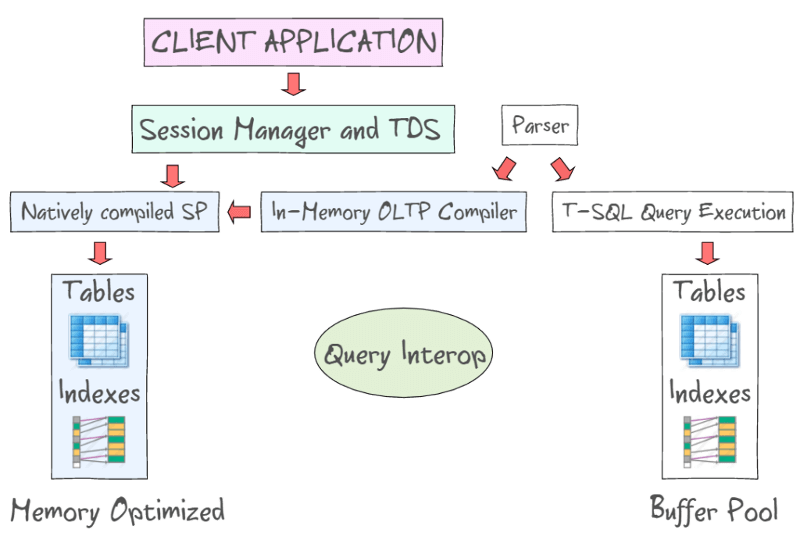
**34. What is User-defined function? What are its various types?**

The user-defined functions in SQL are like functions in any other programming language that accept parameters, perform complex calculations, and return a value. They are written to use the logic repetitively whenever required. There are two types of SQL user-defined functions:

* + **Scalar Function**: As explained earlier, user-defined scalar functions return a single scalar value.
  + **Table Valued Functions**: User-defined table-valued functions return a table as output.
    - **Inline**: returns a table data type based on a single SELECT statement.
    - **Multi-statement**: returns a tabular result-set but, unlike inline, multiple SELECT statements can be used inside the function body.

**35. What is OLTP?**

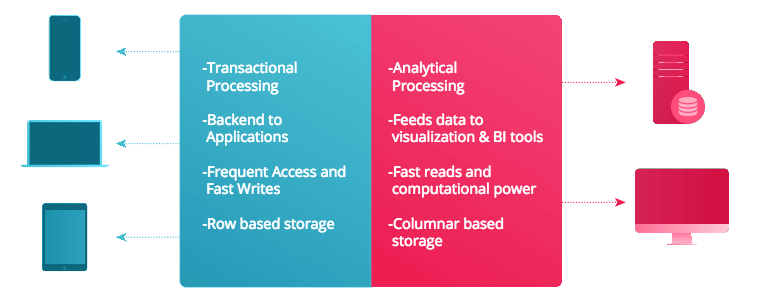
OLTP stands for Online Transaction Processing, is a class of software applications capable of supporting transaction-oriented programs. An essential attribute of an OLTP system is its ability to maintain concurrency. To avoid single points of failure, OLTP systems are often decentralized. These systems are usually designed for a large number of users who conduct short transactions. Database queries are usually simple, require sub-second response times and return relatively few records. Here is an insight into the working of an OLTP system [*Note - The figure is not important for interviews*] -



**36. What are the differences between OLTP and OLAP?**

OLTP stands for **Online Transaction Processing**, is a class of software applications capable of supporting transaction-oriented programs. An important attribute of an OLTP system is its ability to maintain concurrency. OLTP systems often follow a decentralized architecture to avoid single points of failure. These systems are generally designed for a large audience of end users who conduct short transactions. Queries involved in such databases are generally simple, need fast response times and return relatively few records. Number of transactions per second acts as an effective measure for such systems.

OLAP stands for **Online Analytical Processing**, 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. For OLAP systems, the effectiveness measure relies highly on response time. Such systems are widely used for data mining or maintaining aggregated, historical data, usually in multi-dimensional schemas.



**37. What is Collation? What are the different types of Collation Sensitivity?**

Collation refers to a set of rules that determine how data is sorted and compared. Rules defining the correct character sequence are used to sort the character data. It incorporates options for specifying case-sensitivity, accent marks, kana character types and character width. Below are the different types of collation sensitivity:

* + **Case** sensitivity: **A** and **a** are treated differently.
  + **Accent** sensitivity: **a** and **á** are treated differently.
  + **Kana** sensitivity: Japanese kana characters Hiragana and Katakana are treated differently.
  + **Width** sensitivity: Same character represented in single-byte (half-width) and double-byte (full-width) are treated differently.

**38. What is a Stored Procedure?**

A stored procedure is a subroutine available to applications that access a relational database management system (RDBMS). Such procedures are stored in the database data dictionary. The sole disadvantage of stored procedure is that it can be executed nowhere except in the database and occupies more memory in the database server. It also provides a sense of security and functionality as users who can't access the data directly can be granted access via stored procedures.

**DELIMITER** $$

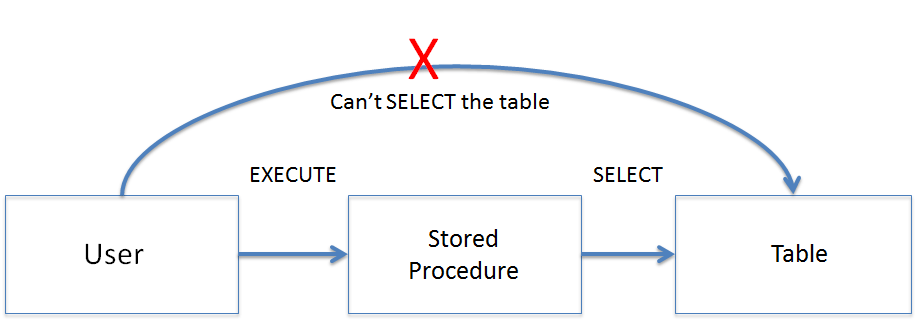
**CREATE** **PROCEDURE** FetchAllStudents()

**BEGIN**

**SELECT** \* **FROM** myDB.students;

**END** $$

**DELIMITER** ;



**39. What is a Recursive Stored Procedure?**

A stored procedure which calls itself until a boundary condition is reached, is called a recursive stored procedure. This recursive function helps the programmers to deploy the same set of code several times as and when required. Some SQL programming languages limit the recursion depth to prevent an infinite loop of procedure calls from causing a stack overflow, which slows down the system and may lead to system crashes.

**DELIMITER** $$ */\* Set a new delimiter => $$ \*/*

**CREATE** **PROCEDURE** calctotal( */\* Create the procedure \*/*

**IN** number INT, */\* Set Input and Ouput variables \*/*

**OUT** total INT

) **BEGIN**

**DECLARE** score INT **DEFAULT** **NULL**; */\* Set the default value => "score" \*/*

**SELECT** awards **FROM** achievements */\* Update "score" via SELECT query \*/*

**WHERE** id = number **INTO** score;

**IF** score **IS** **NULL** **THEN** **SET** total = 0; */\* Termination condition \*/*

**ELSE**

**CALL** calctotal(number+1); */\* Recursive call \*/*

**SET** total = total + score; */\* Action after recursion \*/*

**END** IF;

**END** $$ */\* End of procedure \*/*

**DELIMITER** ; */\* Reset the delimiter \*/*

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

Creating empty tables with the same structure can be done smartly 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. Hence, SQL prepares the new table with a duplicate structure to accept the fetched records but since no records get fetched due to the WHERE clause in action, nothing is inserted into the new table.

**SELECT** \* **INTO** Students\_copy

**FROM** Students **WHERE** 1 = 2;

**41. What is Pattern Matching in SQL?**

SQL pattern matching provides for pattern search in data if you have no clue as to what that word should be. This kind of SQL query uses wildcards to match a string pattern, rather than writing the exact word. The **LIKE** operator is used in conjunction with **SQL Wildcards** to fetch the required information.

* + **Using the % wildcard to perform a simple search**  
    The % wildcard matches zero or more characters of any type and can be used to define wildcards both before and after the pattern. Search a student in your database with first name beginning with the letter K:
  + **SELECT** \*
  + **FROM** students
  + **WHERE** first\_name **LIKE** 'K%'
  + **Omitting the patterns using the NOT keyword**  
    Use the NOT keyword to select records that don't match the pattern. This query returns all students whose first name does not begin with K.
  + **SELECT** \*
  + **FROM** students
  + **WHERE** first\_name **NOT LIKE** 'K%'
  + **Matching a pattern anywhere using the % wildcard twice**  
    Search for a student in the database where he/she has a K in his/her first name.
  + **SELECT** \*
  + **FROM** students
  + **WHERE** first\_name **LIKE** '%Q%'
  + **Using the \_ wildcard to match pattern at a specific position**  
    The \_ wildcard matches exactly one character of any type. It can be used in conjunction with % wildcard. This query fetches all students with letter K at the third position in their first name.
  + **SELECT** \*
  + **FROM** students
  + **WHERE** first\_name **LIKE** '\_\_K%'
  + **Matching patterns for specific length**  
    The \_ wildcard plays an important role as a limitation when it matches exactly one character. It limits the length and position of the matched results. For example -
  + **SELECT** \* */\* Matches first names with three or more letters \*/*
  + **FROM** students
  + **WHERE** first\_name **LIKE** '\_\_\_%'
  + **SELECT** \* */\* Matches first names with exactly four characters \*/*
  + **FROM** students

**WHERE** first\_name **LIKE** '\_\_\_\_'

**1. Compare SQL & PL/SQL**

|  |  |  |
| --- | --- | --- |
| **Criteria** | **SQL** | **PL/SQL** |
| What it is | Single query or command execution | Full programming language |
| What it comprises | Data source for reports, web pages | Application language to build, format and display report, web pages |
| Characteristic | Declarative in nature | Procedural in nature |
| Used for | Manipulating data | Creating applications |

**2. What is BCP? When is it used?**

It is a tool used to duplicate enormous quantity of information from tables and views. It does not facsimile the structures same as foundation to target.  
**BULK INSERT** command helps to bring in a data folder into a record, table or view in a user-specific arrangement.

### Watch this SQL Tutorial for Beginners video

[](https://intellipaat.com/interview-question/sql-interview-questions/#login-form)

Learn for free ! Subscribe to our youtube Channel.

[](https://www.youtube.com/user/intellipaaat?sub_confirmation=1)

**3. When is the UPDATE\_STATISTICS command used?**

This command is used, ones the processing of large data is done.  
When we delete a large number of files, alteration or reproduction takes place in the tables, to be concerned of these changes we need to restructure the indexes This is done **UPDATE\_STATISTICS**.

**4. Explain the steps needed to Create the scheduled job?**

**Steps to create a Scheduled Job :**

1. Connect to the database of SQL server in SQL Server Management Studio. On the SQL Server Agent, we will find a Jobs folder.
2. Right click on jobs and choose Add New.
3. A New Job window will come into view. Give an associated name for the same.
4. Click next on the “Steps” in the left list of options. An SQL job can have multiple steps either in the form of SQL declaration or a stored practice call.
5. Click on the “Schedules” in the left list of options. An SQL job can comprise of one or supplementary schedules. It is basically the instance at which SQL job will jog itself. We can spell out returning schedules also.

**5. What are the different types of constraints? Explain primary key, foreign key, unique key & not null constraints?**

Constraints are the rules applied on data columns on table. These are used to bound the type of data that can go into a table. This ensures the accuracy and consistency of the data. If there is any violation between the constraint and the data action, the action is aborted by the constraint. The different [types of constraints](https://intellipaat.com/tutorial/sql-tutorial/constraints/) are:

* **PRIMARY KEY:**

It is a key which helps you to find the data from the table. It must be unique and not null.

* **FOREGIN KEY**

A foreign key is a field in a relational table that matches the primary key column of another table.https://intellipaat.com/tutorial/sql-tutorial/constraints/

* **UNIQUE:**

It ensures that all the values in the column are unique.

* **Not NULL:**

It indicates that the column cannot have null value.

**6. When are we going to use truncate and delete?**

1. TRUNCATE is a DDL command, whereas DELETE is a DML command.
2. We can’t execute a trigger in case of TRUNCATE whilst with DELETE, we can accomplish a trigger.
3. TRUNCATE is quicker than DELETE, for the reason that when we use DELETE to delete the data, at that time it store the whole statistics in the rollback gap on or after where we can get the data back after removal. In case of TRUNCATE, it will not store data in rollback gap and will unswervingly rub it out. TRUNCATE do not recover the deleted data.
4. We can use any condition in WHERE clause using DELETE but it is not possible with TRUNCATE.5.If a table is referenced by any foreign key constraints, then TRUNCATE won’t work.

***Go through***[*SQL course*](https://intellipaat.com/tutorial/sql-tutorial/)***to learn more about SQL.***

**7. Explain correlated query work?**

It’s most important to be attentive of the arrange of operations in an interrelated subquery.  
**First,** a row is processed in the outer doubt.  
**Then,** for that exacting row, the subquery is executed – as a result for each row processed by the outer query, the subquery will also be processed. In correlated subquery, each time a line is worked for Emp1, the subquery will also make a decision on the exacting row’s value for Emp1.Salary and run. And the outer query will move on to the next row, and the subquery will execute for that row’s value of Emp1.Salary.  
It will persist in anticipation of the **“WHERE (1) = (… )”** state is pleased.

***Read***[*this insightful tutorial*](https://intellipaat.com/tutorial/sql-tutorial/where-clause/)***to learn usage of SQL Clauses.***

[](https://intellipaat.com/microsoft-sql-server-certification-training/?utm_source=IQ&utm_campaign=IQ_SQL_Image&utm_medium=Website)

**8. When is the Explicit Cursor Used ?**

If the developer needs to perform the row by row operations for the result set containing more than one row, then he unambiguously declares a pointer with a name. They are managed by OPEN, FETCH and CLOSE.%FOUND, %NOFOUND, %ROWCOUNT and %ISOPEN characteristics are used in all types of pointers.

**9. Find What is Wrong in this Query?**

**SELECT subject\_code, AVG (marks) FROM students WHERE AVG(marks) > 75 GROUP BY subject\_code;   
The WHERE clause cannot be used to restrict groups. Instead, the HAVING clause should be used.**

SELECT subject\_code, AVG (marks)

FROM students

HAVING AVG(marks) > 75

GROUP BY subject\_code;

**10. Write the Syntax for STUFF function in an SQL server?**

STUFF (String1, Position, Length, String2)

String1 - String to be overwritten

Position - Starting location for overwriting

Length - Length of substitute string

String2- String to overwrite.

**11. Name some commands that can be used to manipulate text in T-SQL code. For example, a command that obtains only a portion of the text or replace a text string, etc.**

* **CHARINDEX( findTextData, textData, [startingPosition] )** – Returns the starting position of the specified expression in a character string. The starting position is optional.
* **LEFT( character\_expression , integer\_expression )** – Returns the left part of a character string with the specified number of characters.
* **LEN( textData )** – Returns integer value of the length of the string, excluding trailing blanks.
* **LOWER ( character\_expression )** – Returns a character expression after converting uppercase character data to lowercase.
* **LTRIM( textData)** – Removes leading blanks. PATINDEX( findTextData, textData ) – Returns integer value of the starting position of text found in the string.
* **REPLACE( textData, findTextData, replaceWithTextData )** – Replaces occurrences of text found in the string with a new value.
* **REPLICATE( character\_expression , integer\_expression )** – Repeats a character expression for a specified number of times.
* **REVERSE( character\_expression )** – Returns the reverse of a character expression.
* **RTRIM( textData)** – Removes trailing blanks. SPACE( numberOfSpaces ) – Repeats space value specified number of times.
* **STUFF( textData, start , length , insertTextData )** – Deletes a specified length of characters and inserts another set of characters at a specified starting point.
* **SUBSTRING( textData, startPosition, length )** – Returns portion of the string.
* **UPPER( character\_expression )** – Returns a character expression with lowercase character data converted to uppercase.

**12. What are the three ways that Dynamic SQL can be executed?**

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

***Get a clear understanding of SQL in***[*this riveting blog*](https://intellipaat.com/blog/nosql-vs-sql-what-is-better/)***.***

**13. In what version of SQL Server were synonyms released? How do synonyms work and explain its use cases? Synonyms were released with SQL Server 2005.**

* Synonyms enable the reference of another object (View, Table, Stored Procedure or Function) potentially on a different server, database or schema in your environment. In simple words, the original object that is referenced in the whole code is using a completely different underlying object, but no coding changes are necessary. Think of this as an alias as a means to simplify migrations and application testing without the need to make any dependent coding changes.
* Synonyms can offer a great deal of value when converting underlying database objects without breaking front end or middle tier code. This could be useful during a re-architecture or upgrade project.

***Become Master of SQL by going through this***[*SQL training*](https://intellipaat.com/microsoft-sql-server-certification-training/#course-content)***.***

**14. If you are a SQL Developer, how can you delete duplicate records in a table with no primary key?**

Use the SET ROWCOUNT command. For instance,  
if you have 2 duplicate rows, you would SET ROWCOUNT 1, execute DELETE command and then SET ROWCOUNT 0.

**15. Is it possible to import data directly from T-SQL commands without using SQL Server Integration Services? If so, what are the commands?**

**Yes, six commands are available to import data directly in the T-SQL language. These commands include :**

* **BCP :** The bulk copy (bcp) command of Microsoft SQL Server provides you with the ability to insert large numbers of records directly from the command line. In addition to being a great tool for command-line aficionados, bcp is a powerful tool for those seeking to insert data into a SQL Server database from within a batch file or other programmatic method.
* **Bulk Insert :** The BULK INSERT statement was introduced in SQL Server 7 and allows you to interact with bcp (bulk copy program) via a script.
* **OpenRowSet :** The OPENROWSET function can be referenced in the FROM clause of a query as if it were a table name. The OPENROWSET function can also be referenced as the target table of an INSERT, UPDATE, or DELETE statement, subject to the capabilities of the OLE DB provider. Although the query might return multiple result sets, OPENROWSET returns only the first one.
* **OPENDATASOURCE :** Provides ad hoc connection information as part of a four-part object name without using a linked server name.
* **OPENQUERY :** Executes the specified pass-through query on the specified linked server. This server is an OLE DB data source. OPENQUERY can be referenced in the FROM clause of a query as if it were a table name.
* **Linked Servers :** Configure a linked server to enable the SQL Server Database Engine to execute commands against OLE DB data sources outside of the instance of SQL Server. Typically linked servers are configured to enable the Database Engine to execute a Transact-SQL statement that includes tables in another instance of SQL Server, or another database product such as Oracle.

**16. What is the native system stored procedure to execute a command against all databases?**

* The sp\_MSforeachdb system stored procedure accepts the **@Command** parameter which can be exetecuted against all databases. The ‘?’ is used as a placeholder for the database name to execute the same command.
* The alternative is to use a cursor to process specific commands against each database.

**17. How can a SQL Developer prevent T-SQL code from running on a production SQL Server?**

Use IF logic with the **@@SERVERNAME function** compared against a string with a RETURN command before any other logic.

**18. How do you maintain database integrity where deletions from one table will automatically cause deletions in another table?**

You can create a trigger that will automatically delete elements in the second table when elements from the first table are removed.

**19. What port does SQL server run on?**

1433 is the standard port for SQL server.

***Go through***[*this SQL Video*](https://intellipaat.com/microsoft-sql-server-certification-training/#course-preview)***to get clear understanding of SQL.***

**20. What is the SQL CASE statement used for? Explain with an example?**

It allows you to embed an if-else like clause in the SELECT clause.

SELECT Employee\_Name, CASE Location

WHEN 'alex' THEN Bonus \* 2

WHEN 'robin' THEN Bonus \*, 5

ELSE Bonus

END

"New Bonus"

FROM Intellipaat\_employee;

***Read***[***this blog***](https://intellipaat.com/blog/sql-optimization-techniques/)***to learn why SQL Optimization has always been a important aspect of database management.***

**21. What are the risks of storing a hibernate-managed object in cache? How do you overcome the problems?**

The primary problem here is that the object will outlive the session it came from. Lazily loaded properties won’t get loaded if needed later. To overcome the problem, perform cache on the object’s id and class and then retrieve the object in the current session context.

**22. When is the use of UPDATE\_STATISTICS command ?**

Updating statistics ensures that queries compile with up-to-date statistics. However, updating statistics causes queries to recompile. We recommend not updating statistics too often because there is a performance tradeoff between improving query plans and the time it takes to recompile queries. The specific tradeoffs depend on your application. UPDATE STATISTICS can use tempdb to sort the sample of rows for building statistics.

**Syntax**

UPDATE STATISTICS table\_or\_indexed\_view\_name

[

{

{ index\_or\_statistics\_\_name }

| ( { index\_or\_statistics\_name } [ ,...n ] )

}

]

[ WITH

[

FULLSCAN

| SAMPLE number { PERCENT | ROWS }

| RESAMPLE

[ ON PARTITIONS ( { | } [, …n] ) ]

| [ ,...n ]

]

[ [ , ] [ ALL | COLUMNS | INDEX ]

[ [ , ] NORECOMPUTE ]

[ [ , ] INCREMENTAL = { ON | OFF } ]

] ;

::=

[ STATS\_STREAM = stats\_stream ]

[ ROWCOUNT = numeric\_constant ]

[ PAGECOUNT = numeric\_contant ]

**23. What is SQL Profiler?**

Microsoft SQL Server Profiler is a graphical user interface to SQL Trace for monitoring an instance of the Database Engine or Analysis Services. You can capture and save data about each event to a file or table to analyze later.

Use SQL Profiler to monitor only the events in which you are interested.

If traces are becoming too large, you can filter them based on the information you want, so that only a subset of the event data is collected. Monitoring too many events adds overhead to the server and the monitoring process and can cause the trace file or trace table to grow very large, especially when the monitoring process takes place over a long period of time.

These are described in SQL’s online reference guide and on [SQL community](https://intellipaat.com/community/sql).

**24. What command using Query Analyzer will give you the version of SQL server and operating system?**

SELECT SERVERPROPERTY (‘productversion’), SERVERPROPERTY (‘productlevel’), SERVERPROPERTY (‘edition’).

**25. What does it mean to have QUOTED\_IDENTIFIER ON? What are the implications of having it OFF?**

When **SET QUOTED\_IDENTIFIER** is **ON,** identifiers can be delimited by double quotation marks, and literals must be delimited by single quotation marks. When **SET QUOTED\_IDENTIFIER** is **OFF**, identifiers cannot be quoted and must follow all Transact-SQL rules for identifiers.

**26. What is the STUFF function and how does it differ from the REPLACE function in SQL?**

**Stuff function :** – This function is used to replace string from the given start position, passed as 2nd argument with string passed as last argument. In [Stuff function](https://intellipaat.com/tutorial/architectural-capability-framework/), 3rd argument defines the number of characters which are going to be replaced.  
**Syntax :-**

STUFF ( character\_expression , start , length , replaceWith\_expression )

**For example :-**

Select Stuff ('Intellipaat', 3, 3, 'abc')

This query will return the string **"Iabcllipaat"**. In this example, Stuff function replaces the string **"Intellipaat"** onwards the 3rd position('nte') with 'abc'.

**Replace Function :**– Replace function is used to replace all occurrence of a specified with the string passed as last argument.  
**Syntax :-**

REPLACE ( string\_expression , string\_pattern , string\_replacement )

**For example :-**

Select Replace ('Abcabcabc', 'bc', 'xy')

This query will return the string **Axyaxyaxy**. In this example, Replace function replaces the occurrence of each **'bc'** string with **'xy'**.

**27. How to get @@ERROR and @@ROWCOUNT at the same time?**

If @@Rowcount is checked after Error checking statement then it will have 0 as the value of @@Recordcount as it would have been reset. And if @@Recordcount is checked before the error-checking statement then @@Error would get reset. To get @@error and @@rowcount at the same time do both in same statement and store them in local variable.

SELECT @RC = @@ROWCOUNT, [@ER](https://intellipaat.com/members/er/) = @@ERROR

**28. What is de-normalization in SQL database administration? Give examples**

De-normalization is used to optimize the readability and performance of the database by adding redundant data. It covers the inefficiencies in the relational database software.  
De-normalization logical data design tend to improve the query responses by creating rules in the database which are called as constraints.  
**Examples include the following :**

* Materialized views for implementation purpose such as :
* Storing the count of “many” objects in one-to-many relationship.
* Linking attribute of one relation with other relations.
* To improve the performance and scalability of web applications.

**29. Can you explain about buffer cash and log Cache in SQL Server?**

* **Buffer Cache :** [Buffer cache](https://intellipaat.com/tutorial/oracle-dba-tutorial/database-maintenance/) is a memory pool in which data pages are read. The ideal performance of the buffer cache is indicated as: 95% indicates that pages that were found in the memory are 95% of time. Another 5% is need physical disk access.  
  If the value falls below 90%, it is the indication of more physical memory requirement on the server.
* **Log Caches :** Log cache is a memory pool used to read and write the log pages. A set of cache pages are available in each log cache. The synchronization is reduced between log and data buffers by managing log cache separately from the buffer cache.

**30. Describe how to use Linked Server.**

MS SQL Server supports the connection to different OLE DB on an ad hoc basis. This persistent connection is referred as Linked Server.  
**Following are the steps to use Linked Server for any OLE DB. You can refer this to use an MS-Excel workbook.**

1. Open SQL Server Management Studio in SQL Server.
2. Expand Server Objects in Object Explorer.
3. Right-click on Linked Servers. Click on New Linked Server.
4. Select General page in the left pane and
   * Type any name for the linked server in the first text box.
   * Select the Other Data Source option.
   * Click on Microsoft Jet 4.0 OLE DB Provider from the Provider list.
   * Type the Excel as the name of the OLE DB data source.
   * Type the full path and file name of the Excel file in Data Source box.
   * Type the Excel version no. (7.0, 8.0 etc) in the Provider String. Use Excel 8.0 for Excel 2000, Excel 2002 or Excel 97.
   * To create a linked server click on OK.

**31. How to find second highest salary of an Employee?**

There are many ways to find second highest salary of Employees in SQ. You can either use SQL Join or Subquery to solve this problem.  
**Here is SQL query using Subquery :**

Select MAX(Salary) from Intellipaat\_emplyee WHERE Salary NOT IN ( select MAX(Salary) from Intellipaat\_employee.

**32. Explain how to send email from SQL database.**

SQL Server has a feature for sending mails. Stored procedures can also be used for sending mail on demand. With SQL Server 2005, MAPI client is not needed for sending mails.  
**The following is the process for sending emails from database.**

* Make sure that the SQL Server Mail account is configured correctly and enable Database Mail.
* Write a script to send an e-mail. The following is the script.
* USE [YourDB]
* EXEC msdb.dbo.sp\_send\_dbmail
* @recipients = 'xyz@intellipaat.com; abc@intellipaat.com;pqr@intellipaat.com’
* @body = ' A warm wish for your future endeavor',
* @subject = 'This mail was sent using Database Mail' ;

GO

**33. How to make remote connection in database?**

**The following is the process to make a remote connection in database :**

1. Use SQL Server Surface Area Configuration Tool for enabling the remote connection in database.
2. Click on Surface Area Configuration for Services and Connections.
3. Click on SQLEXPRESS/Database Engine/RemoteConnections.
4. Select the radio button: Local and Remote Connections and select ‘Using TCP/IP only’ under Local and Remote Connections.
5. Click on OK button / Apply button

**34. What is the purpose of OPENXML clause SQL server stored procedure?**

OPENXML parses the XML data in SQL Server in an efficient manner. It’s primary ability is to insert XML data to the RDB. It is also possible to query the data by using OpenXML. The path of the XML element needs to be specified by using ‘xpath’.  
**The following is a procedure for retrieving xml data:**

DECLARE @index int

DECLARE @xmlString varchar(8000)

SET @xmlString ='

abc

9343463943/PhoneNo>

xyz

9342673212

'

EXEC sp\_xml\_preparedocument @index OUTPUT, @xmlString

SELECT \* FROM OPENXML (@index, 'Persons/Person') WITH (id varchar(10), Name varchar(100) 'Name' , PhoneNo varchar(50) 'PhoneNo')

EXEC sp\_xml\_removedocument @index

The above code snippet results the following:

15201 abc 9343463943

15202 xyz 9342673212

**35. How to store pdf file in SQL Server?**

Create a column as type ‘blob’ in a table. Read the content of the file and save in ‘blob’ type column in a table.  
**Or**  
Store them in a folder and establish the pointer to link them in the database.

**36. Explain the use of keyword WITH ENCRYPTION. Create a Store Procedure with Encryption.**

It is a way to convert the original text of the stored procedure into encrypted form. The stored procedure gets obfuscated and the output of this is not visible to

CREATE PROCEDURE Abc

WITH ENCRYPTION

AS

<< SELECT statement>>

GO

WITH ENCRYPTION indicates that SQL Server will convert the original text of CREATE PROCEDURE statement to an encrypted format. Users that do not have no access to system tables or database files cannot retrieve the encrypted text. However, the text will be available to privileged users.

**Example:**

**CREATE PROCEDURE salary\_sum  
WITH ENCRYTION  
AS  
SELECT sum(salary)  
FROM employee  
WHERE emp\_dept LIKE Develop**

**37. What is lock escalation?**

Lock escalation is used to convert row locks and page locks into table locks thereby “escalating” the smaller or finer locks. This increases the system performance as each lock is nothing but a memory structure. Too many locks would mean more consumption of memory. Hence, escalation is used.  
Lock escalation from SQL Server 7.0 onwards is dynamically managed by SQL Server. It is the process of converting a lot of low level locks into higher level locks.

**38. What is Failover clustering overview?**

Failover clustering is mainly used for data availability. **Typically, in a failover cluster, there are two machines.**

* One machine provides the basic services and the second is available to run the service when the primary system fails.
* The primary system is monitored periodically to check if it works. This monitoring may be performed by the failover computer or an independent system also called as cluster controller. In an event of failure of primary computer, the failover system takes control.

**39. What is Builtin/Administrator?**

The Builtin/Administrator account is basically used during some setup to join some machine in the domain. It should be disabled immediately thereafter. For any disaster recovery, the account will be automatically enabled. It should not be used for normal operations.

**40. What XML support does the SQL server extend?**

**SQL Server (server-side) supports 3 major elements :**

1. Creation of XML fragments: This is done from the relational data using FOR XML to the select query.
2. Ability to shred xml data to be stored in the database.
3. Finally, storing the xml data.

Client-side XML support in SQL Server is in the form of SQLXML. **It can be described in terms of :**

* **XML Views :** providing bidirectional mapping between XML schemas and relational tables.
* **Creation of XML Templates :** allows creation of dynamic sections in XML.

SQL server can return XML document using FOR XML clause. XML documents can be added to SQL Server database and you can use the OPENXML clause to display the data from the document as a relational result set. SQL Server 2000 supports XPath queries.

##### **Q1) What is the Difference between MySql Vs SQL Server Performance?**

**Ans.**

|  |  |  |
| --- | --- | --- |
| **MySQl Server Vs SQL Server** | | |
| **Function** | **MS SQL Server** | **MySql** |
| Developer | Microsoft | Oracle |
| License | Commercial | OpenSource |
| Cloud-based | No | No |
| Implementation Language | C++ | C & C++ |
| XML Support | Yes | Yes |
| Supported Programming Lang | It supports C#, PHP, Python, Ruby, R, Visual Basic, Java etc | It supports c++, C#, Java, PHP, Perl, Python, Ruby, Tcl, Delphi, D etc |
| Server-Side Scripting. | Net & TransaScripting. | Yes, it supports |
| Concurreny | It supports ,yes | It supports ,yes |
| Consistency Concept's | It enables immediate consistency | It enables immediate consistency |
| Transaction concepts | It supports ACID | It supports ACID |

##### **Q2) What is normalization? Explain different levels of normalization?**

**Ans.** It is the way to eliminate redundant data

1. Reduces null value
2. Enables efficient indexing
3. 1NF – Removes duplicated attributes, Attribute data should be atomic, and attribute should be same kind.
4. 2NF – Should be in 1NF and each non-key is fully dependent on the primary key.
5. 3NF – Should be in 2NF and all the non-key attributes which are not dependent on the primary key should be removed. All the attributes which are dependent on the other non-key attributes should also be removed. Normalization is done in OLTP.

##### **Q3) What is denormalization and when would you go for it?**

**Ans.** It is the reverse process of normalization. It increases the query performance by reducing the joins. It is used for OLAP applications.

##### **Q4) How do you implement one-to-one, one-to-many and many-to-many relationships while designing tables?**

**Ans.** Relationships in sql server are explained below

1. One to One –It can be implemented as a single table. Rarely it is implemented in two tables. For each instance in the first entity there is one and only one in the second entity and vice versa.
2. One to Many –For each instance in the first entity there can be one or more in the second entity. For each instance in the second entity there can be one and only one instance in the first entity.
3. Many to Many –For each instance in the first entity there can be one or more instance in the second entity and moreover, for each instance in the second entity there can be one or more instance in the first entity.

##### **Q5) Difference between Primary key and Unique key.**

**Ans. Primary Key**

* 1.Enforces uniqueness of the column in a table
* 2.Default clustered index
* 3.Does not Allow nulls

**Unique Key**

* 1.Enforces uniqueness of the column in a table.
* 2.Default non-clustered index.
* 3.Allows one null value

##### **Q6) Define following keys:**

**Ans.** Candidate key, Alternate key, Composite key.

* 1.Candidate key –Key which can uniquely identify a row in table.
* 2.Alternate key –If the table has more than one candidate keys and when one becomes a primary key the rest becomes alternate keys.
* 3.Composite key –More than one key uniquely identify a row in a table.

##### **Q7) What are defaults? Is there a column to which a default can’t be bound?**

**Ans.**  1.It is a value that will be used by a column if no value is supplied to that column while inserting data.  
         2.I can’t be assigned for identity and timestamp values.

##### **Q8) What are user defined data types and when you should go for them?**

**Ans.** Lets you extend the base SQL server data types by providing a descriptive name and format to the database.

E.g. Flight\_num appears in many tables and all these tables have varchar(8)  
Create a user defined data-type

##### **Q9) What is a transaction and what are ACID properties?**

**Ans.** A transaction is a logical unit of work in which, all the steps must be performed or none. ACID stands for Atomicity, Consistency, Isolation, and Durability. These are the properties of a transaction.

##### **Q10) What part does database design have to play in the performance of a SQL Server-based application?**

**Ans.** It plays a very major part. When building a new system, or adding to an existing system, it is crucial that the design is correct. Ensuring that the correct data is captured and is placed in the appropriate tables, that the right relationships exist between the tables and that data redundancy is eliminated is an ultimate goal when considering performance. Planning a design should be an iterative process, and constantly reviewed as an application is developed. It is rare, although it should be the point that everyone tries to achieve, when the initial design and system goals are not altered, no matter how slightly. Therefore, a designer has to be on top of this and ensure that the design of the database remains efficient..

##### **Q11) What can a developer do during the logical and physical design of a database in order to help ensure that their database and SQL Server-based application will perform well?**

**Ans.** A developer must investigate volumes of data (capacity planning), what types of information will be stored, and how that data will be accessed. If you are dealing with an upgrade to an existing system, analyzing the present data and where existing data volumes occur, how that data is accessed and where the current response bottlenecks are occurring, can help you search for problem areas in the design.

A new system would require a thorough investigation of what data will be captured, and looking at volumes of data held in other formats also will aid design. Knowing your data is just as important as knowing the constituents of your data. Also, constantly revisit your design. As your system is built, check relationships, volumes of data, and indexes to ensure that the physical design is still at its optimum. Always be ready to check your system by using tools like the SQL Server Profiler.

##### **Q12) What are the main steps in Data Modeling?**

**Ans.**

* 1.Logical – Planning, Analysis and Design
* 2.Physical – Design, Implementation and Maintenance

### ****DATABASE DEVELOPMENT / PROGRAMMING****

##### **Q13) What are cursors? Explain different types of cursors. What are the disadvantages of cursors? How can you avoid cursors?**

**Ans.** Cursors allow row-by-row processing of the result sets.

**Types of cursors:**

Static – Makes a temporary copy of the data and stores in tempdb and any modifications on the base table does not reflected in data returned by fetches made by the cursor.  
Dynamic – Reflects all changes in the base table.  
Forward-only – specifies that cursor can only fetch sequentially from first to last.  
Keyset-driven – Keyset is the set of keys that uniquely identifies a row is built in a tempdb.

**Disadvantages of cursors:**

Each time you fetch a row from the cursor, it results in a network roundtrip, whereas a normal SELECT query makes only one roundtrip, however large the result set is. Cursors are also costly because they require more resources and temporary storage (results in more IO operations). Further, there are restrictions on the SELECT statements that can be used with some types of cursors.  
Most of the times set-based operations can be used instead of cursors.  
Here is an example:  
If you have to give a flat hike to your employees using the following criteria:

Salary between 30000 and 40000 — 5000 hike  
Salary between 40000 and 55000 — 7000 hike  
Salary between 55000 and 65000 — 9000 hike

In this situation, many developers tend to use a cursor, determine each employee’s salary and update his salary according to the above formula. But the same can be achieved by multiple update statements or can be combined in a single UPDATE statement as shown below:

UPDATE tbl\_emp SET salary =  
CASE  
WHEN salary BETWEEN 30000 AND 40000 THEN salary + 5000  
WHEN salary BETWEEN 40000 AND 55000 THEN salary + 7000  
WHEN salary BETWEEN 55000 AND 65000 THEN salary + 10000  
END

another situation in which developers tend to use cursors: You need to call a stored procedure when a column in a particular row meets certain condition. You don’t have to use cursors for this. This can be achieved using WHILE loop, as long as there is a unique key to identify each row.

##### **Q14) Write down the general syntax for a SELECT statement covering all the options.**

**Ans.** Here’s the basic syntax: (Also checkout SELECT in books online for advanced syntax).

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | SELECT select\_list  [HDEV:INTO new\_table\_]  FROM table\_source  [HDEV:WHERE search\_condition]  [HDEV:GROUP BY group\_by\_expression]  [HDEV:HAVING search\_condition]  [ORDER BY order\_expression [ASC | HDEV:DESC] ] |

##### **Q15) What is a Join and explain different types of Joins?**

**Ans.** Joins are used in queries to explain how different tables are related. Joins also let you select data from a table depending upon data from another table.  
Types of joins: INNER JOINs, OUTER JOINs, CROSS JOINs. OUTER JOINs are further classified as LEFT OUTER JOINS, RIGHT OUTER JOINS and FULL OUTER JOINS.

##### **Q16) Can you have a nested transaction?**

**Ans.** Yes, very much. Check out BEGIN TRAN, COMMIT, ROLLBACK, SAVE TRAN and @@TRANCOUNT

##### Q17) What is an extended stored procedure? Can you instantiate a COM object by using T-SQL?

**Ans.** An extended stored procedure is a function within a DLL (written in a programming language like C, C++ using Open Data Services (ODS) API) that can be called from T-SQL, just the way we call normal stored procedures using the EXEC statement.  
Yes, you can instantiate a COM (written in languages like VB, VC++) object from T-SQL by using sp\_OACreate stored procedure. Also see books online for sp\_OAMethod, sp\_OAGetProperty, sp\_OASetProperty, sp\_OADestroy.

##### Q18) What is the system function to get the current user’s userid?

**Ans.** USER\_ID(). Also check out other system functions like USER\_NAME(), SYSTEM\_USER, SESSION\_USER, CURRENT\_USER, USER, SUSER\_SID(), HOST\_NAME().

##### Q19) What are triggers? How many triggers you can have on a table? How to invoke a trigger on demand?

**Ans.** Triggers are special kind of stored procedures that get executed automatically when an INSERT, UPDATE or DELETE operation takes place on a table. In SQL Server 6.5 you could define only 3 triggers per table, one for INSERT, one for UPDATE and one for DELETE. From SQL Server 7.0 onwards, this restriction is gone, and you could create multiple triggers per each action. But in 7.0 there’s no way to control the order in which the triggers fire. In SQL Server 2000 you could specify which trigger fires first or fires last using sp\_settriggerorder.  
Triggers can’t be invoked on demand. They get triggered only when an associated action (INSERT, UPDATE, DELETE) happens on the table on which they are defined. Triggers are generally used to implement business rules, auditing. Triggers can also be used to extend the referential integrity checks, but wherever possible, use constraints for this purpose, instead of triggers, as constraints are much faster.  
Till SQL Server 7.0, triggers fire only after the data modification operation happens. So in a way, they are called post triggers. But in SQL Server 2000 you could create pre triggers also – INSTEAD OF triggers.  
Virtual tables – Inserted and Deleted form the basis of trigger architecture.

##### Q20) What is a self join? Explain it with an example.

**Ans.** Self join is just like any other join, except that two instances of the same table will be joined in the query. Here is an example: Employees table which contains rows for normal employees as well as managers. So, to find out the managers of all the employees, you need a self join.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | CREATE TABLE emp  (  empid int,  mgrid int,  empname char(10)  )  INSERT emp SELECT 1,2,’Vyas’  INSERT emp SELECT 2,3,’Mohan’  INSERT emp SELECT 3,NULL,’Shobha’  INSERT emp SELECT 4,2,’Shridhar’  INSERT emp SELECT 5,2,’Sourabh’  SELECT t1.empname [HDEV:Employee], t2.empname [HDEV:Manager]  FROM emp t1, emp t2  WHERE t1.mgrid = t2.empid |

Here’s an advanced query using a LEFT OUTER JOIN that even returns the employees without managers (super bosses)

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | SELECT t1.empname [HDEV:Employee], COALESCE(t2.empname, ‘No manager’) [HDEV:Manager]  FROM emp t1  LEFT OUTER JOIN  emp t2  ON  t1.mgrid = t2.empid |

Explore SQL Server Sample Resumes! Download & Edit, Get Noticed by Top Employers![**Download Now!**](https://mindmajix.com/sql-server-sample-resumes)

##### Q21) Write a SQL Query to find first Week Day of month?

|  |  |
| --- | --- |
| 1 | SELECT DATENAME(dw, DATEADD(dd, – DATEPART(dd, GETDATE()) + 1, GETDATE())) AS FirstDay |

##### Q22) How to find 6th highest salary from Employee table?

|  |  |
| --- | --- |
| 1 | SELECT TOP 1 salary FROM (SELECT DISTINCT TOP 6 salary FROM employee ORDER BY salary DESC) a ORDER BY salary |

##### Q23) How can I enforce to use particular index?

You can use index hint (index=index\_name) after the table name. SELECT au\_lname FROM authors (index=aunmind)

##### Q24) What is ORDER BY and how is it different than clustered index?

The ORDER BY clause sorts query results by one or more columns up to 8,060 bytes. This will happen by the time when we retrieve data from database. Clustered indexes will physically sort data, while inserting/updating the table.

##### Q25) What is the difference between a UNION and a JOIN?

#### Subscribe to our youtube channel to get new updates..!

A JOIN selects columns from 2 or more tables. A UNION selects rows.

##### Q26) What is the Referential Integrity?

Referential integrity refers to the consistency that must be maintained between primary and foreign keys, i.e. every foreign key value must have a corresponding primary key value

##### Q27) What is the purpose of UPDATE STATISTICS?

It updates information about the distribution of key values for one or more statistics groups (collections) in the specified table or indexed view.

##### Q28) What is the use of SCOPE\_IDENTITY() function?

It returns the most recently created identity value for the tables in the current execution scope.

##### Q29) What do you consider are the best reasons to use stored procedures in your application instead of passing Transact-SQL code directly to SQL Server?

First and foremost, a stored procedure is a compiled set of code, where passing T-SQL through languages such as VB, Visual FoxPro, etc., means that the set of code needs to be compiled first. Although T-SQL within VB, etc., can be prepared before running, this is still slower than using a stored procedure. Then, of course, there is the security aspect, where, by building a stored procedure, you can place a great deal of security around it. When dealing with sensitive data, you can use an encrypted stored procedure to hide sensitive columns, calculations, and so on. Finally, by using a stored procedure, I feel that transactional processing becomes a great deal easier and, in fact, using nested transactions become more insular and secure. Having to deal with transactions within code that may have front end code, will slow up a transaction and therefore a lock will be held for longer than necessary.

##### Q30) What are some techniques for writing fast performing stored procedures?

Fast performing stored procedures are like several other areas within T-SQL. Revisiting stored procedures every six months or so, to ensure that they are still running at their optimum performance is essential. However, actual techniques themselves include working with as short a transaction area as possible, as lock contention will certainly impact performance. Recompiling your stored procedures after index additions if you are unable or not wishing to restart SQL Server, will also ensure that a procedure is using the correct index, if that stored procedure is accessing the table which has received the new index. If you have a T-SQL command that joins several tables, and it takes a long time to return a value, first of all check out the indexes. But what you may find tends to help, is to break down the code and try to determine which join it is that is causing the performance problem. Then analyze this specific join and see why it is a problem.  
Always check out a stored procedure’s performance as you build it up by using the SHOWPLAN commands.  
Also, try to use EXISTS, rather than a JOIN statement. An EXISTS statement will only join on a table until one record is found, rather than joining all the records . Also, try to look at using sub queries when you are trying to find a handful of values in the sub query statement, and there is no key on the column you are looking up on.

###### Q31) When should SQL Server-based cursors be used, and not be used?

SQL Server cursors are perfect when you want to work one record at a time, rather than taking all the data from a table as a single bulk. However, they should be used with care as they can affect performance, especially when the volume of data increases. From a beginner’s viewpoint, I really do feel that cursors should be avoided every time because if they are badly written, or deal with too much data, they really will impact a system’s performance. There will be times when it is not possible to avoid cursors, and I doubt if many systems exist without them. If you do find you need to use them, try to reduce the number of records to process by using a temporary table first, and then building the cursor from this. The lower the number of records to process, the faster the cursor will finish. Always try to think “out of the envelope”.

###### **Q32) What alternatives do developers have over using SQL Server-based cursors? In other words, how can developers perform the same function as a cursor without using a cursor?**

Perhaps one of the performance gains least utilized by developers starting out in SQL Server are temporary tables. For example, using one or more temporary tables to break down a problem in to several areas could allow blocks of data to be processed in their own individual way, and then at the end of the process, the information within the temporary tables merged and applied to the underlying data. The main area of your focus should be, is there an alternative way of doing things? Even if I have to break this down into several chunks of work, can I do this work without using cursors, and so result in faster performance. Another area that you can look at is the use of CASE statements within your query. By using a CASE statement, you can check the value within a column and make decisions and operations based on what you have found. Although you will still be working on a whole set of data, rather than a subset found in a cursor, you can use CASE to leave values, or records as they are, if they do not meet the right criteria. Care should be taken here though, to make sure that by looking at all the data, you will not be creating a large performance impact. Again, look at using a subset of the data by building a temporary table first, and then merging the results in afterwards. However, don’t get caught out with these recommendations and do any of them in every case. Cursors can be faster if you are dealing with small amounts of data. However, what I have found, to be rule number one, is get as little data in to your cursor as is needed.

###### Q33) If you have no choice but to use a SQL Server-based cursor, what tips do you have in order to optimize them?

Perhaps the best performance gain is when you can create a cursor asynchronously rather than needing the whole population operation to be completed before further processing can continue. Then, by checking specific global variables settings, you can tell when there is no further processing to take place. However, even here, care has to be taken. Asynchronous population should only occur on large record sets rather than those that only deal with a small number of rows. Use the smallest set of data possible. Break out of the cursor loop as soon as you can. If you find that a problem has occurred, or processing has ended before the full cursor has been processed, then exit. If you are using the same cursor more than once in a batch of work, and this could mean within more than one stored procedure, then define the cursor as a global cursor by using the GLOBAL keyword, and not closing or deallocating the cursor until the whole process is finished. A fair amount of time will be saved, as the cursor and the data contained will already be defined, ready for you to use.  
DATABASE PERFORMANCE OPTIMIZATION / TUNING

###### Q34) What are the steps you will take to improve performance of a poor performing query?

This is a very open ended question and there could be lot of reasons behind the poor performance of a query. But some general issues that you could talk about would be:

1. No indexes
2. No Table scans
3. Missing or out of date statistics
4. Blocking
5. Excess recompilations of stored procedures

###### Q35) What is an ER Diagram?

An ER diagram or Entity-Relationship diagram is a special picture used to represent the requirements and assumptions in a system from a top down perspective. It shows the relations between entities (tables) in a database.

###### Q36) What is a prime attribute?

A prime attribute is an attribute that is part of a candidate key.

###### Q37) What are the properties of a transaction?

The ACID properties. Atomicity, Consistency, Isolation, and Durability.

###### Q38) What is a non-prime attribute?

A non-prime attribute is an attribute that is not a part of a candidate key.

###### Q39) What is Atomicity?

This means the transaction finish completely, or it will not occur at all.

###### Q40) What is Consistency?

Consistency means that the transaction will repeat in a predictable way each time it is performed.

###### Q41) What is Isolation?

The data the transactions are independent of each other. The success of one transaction doesn’t depend on the success of another.

###### Q42) What is Durability?

Guarantees that the database will keep track of pending changes so that the server will be able to recover if an error occurs.

###### Q43) What is a DBMS?

A DBMS is a set of software programs used to manage and interact with databases.

###### Q44) What is a RDBMS?

It is a set of software programs used to interact with and manage relational databases. Relational databases are databases that contain tables.

###### Q45) What is business intelligence?

Refers to computer-based techniques used in identifying, extracting, and analyzing business data, such as sales revenue by products and/or departments, or by associated costs and incomes.

###### Q46) What is normalization?

Database normalization is the process of organizing the fields and tables of a relational database to minimize redundancy and dependency.

###### Q47) What is a relationship?

The way in which two or more concepts/entities are connected, or the state of being connected.

###### Q48) What are the different types of relationships?

One to one, one to many, many to many, many to fixed cardinality.

###### Q49) What is the difference between a OLTP and database?

An OLTP is the process of gathering the data from the users, and a database is the initial information.

###### Q50) What are the different kinds of relationships?

Identifying and non-identifying.

###### Q51) What is an entity?

Something that exists by itself, although it need not be of material existence.

###### Q52) What is a conjunction table?

A table that is composed of foreign keys that points to other tables.

###### Q53) What is a relational attribute?

An attribute that would not exist if it were not for the existence of a relation.

###### Q54) What are associative entities?

An associative entity is a conceptual concept. An associative entity can be thought of as both an entity and a relationship since it encapsulates properties from both. It is a relationship since it is serving to join two or more entities together, but it is also an entity since it may have its own properties.

###### Q55) What is the difference between a derived attribute, derivedpersistent attribute, and computed column?

A derived attribute is a attribute that is obtained from the values of other existing columns and does not exist on it’s own. A derived persistent attribute is a derived attribute that is stored. A computed attribute is a attribute that is computed from internal system values.

###### Q56) What are the types of attributes?

Simple, composite (split into columns), multi-valued (becomes a separate table), derived, computed, derived persistent.

###### Q57) Is the relationship between a strong and weak entity always identifying?

Yes, this is the requirement.

###### Q58) Do stand alone tables have cardinality?

No.

###### Q59) What is a simple key?

It is a key that in composed of one attribute.  
Give/ recite the types of UDF functions.  
Scalar, In-line, Multi

###### Q60) Describe what you know about PK, FK, and UK.

Primary keys – Unique clustered index by default, doesn’t accept null values, only one primary key per table.  
Foreign Key – References a primary key column. Can have null values. Enforces referential integrity.  
Unique key – Can have more than one per table. Can have null values. Cannot have repeating values. Maximum of 999 clustered indexes per table.

##### Q61) What do you mean by CTEs? How will you use it?

CTEs also known as common table expressions are used to create a temporary table that will only exist for the duration of a query. They are used to create a temporary table whose content you can reference in order to simplify a queries structure.

##### Q62) What is a sparse column?

It is a column that is optimized for holding null values.

##### Q63) What would the command: DENY CREATE TABLE TO Peter do?

It wouldn’t allow the user Peter to perform the operation CREATE TABLE regardless of his role.

##### Q64) What does the command: GRANT SELECT ON project TO Peter do?

It will allow the SELECT operation on the table ‘project’ by Peter.

##### Q65) What does the command: REVOKE GRANT SELECT ON project TO Peter do?

It will revoke the permission granted on that table to Peter.

##### Q66) New commands in SQL 2008?

Database encryption, CDCs tables – For on the fly auditing of tables, Merge operation, INSERT INTO – To bulk insert into a table from another table, Hierarchy attributes, Filter indexes, C like operations for numbers, resource management, Intellisense – For making programming easier in SSMS, Execution Plan Freezing – To freeze in place how a query is executed.  
What is new in SQL 2008 R2?  
PowerPivot, maps, sparklines, data bars, and indicators to depict data.

##### Q67) What is faster? A table variable or temporary table?

A table variable is faster in most cases since it is held in memory while a temporary table is stored on disk. However, when the table variable’s size exceeds memory size the two table types tend to perform similarly.

##### Q68) How big is a tinyint, smallint, int, and bigint?

1 byte, 2 bytes, 4 bytes, and 8 bytes.

##### Q69) What does @@trancount do?

It will give you the number of active transactions for the current user.

Q70) What are the drawbacks of CTEs?

It is query bound.

##### Q71) What is the transaction log?

t keeps a record of all activities that occur during a transaction and is used to roll back changes

##### Q72) What are before images, after images, undo activities and redo activities in relation to transactions?

Before images refers to the changes that are rolled back on if a transaction is rolled back. After images are used to roll forward and enforce a transaction. Using the before images is called the undo activity. Using after images is called the redo activity.

##### Q73) What are shared, exclusive and update locks?

A shared lock, locks a row so that it can only be read. An exclusive lock locks a row so that only one operation can be performed on it at a time. An update lock basically has the ability to convert a shared lock into an exclusive lock.

##### Q74) What does WITH TIES do?

If you use TOP 3 WITH TIES \*, it will return the rows, that have a similarity in each of their columns with any of the column values from the returned result set.

##### Q75) How can you get a deadlock in SQL?

By concurrently running the same resources that access the same information in a transaction.

##### Q76) What is LOCK\_TIMEOUT used for?

It is used for determining the amount of time that the system will wait for a lock to be released.

##### Q77) What is the ANY predicate used for?

|  |  |
| --- | --- |
| 1 | SELECT \* FROM emp\_table WHERE enter\_date > ANY (SELECT enter\_date FROM works\_on) |

##### Q78) What is the ALL predicate used for?

|  |  |
| --- | --- |
| 1 | SELECT \* FROM emp\_table WHERE enter\_date > ALL (SELECT enter\_date FROM works\_on) |

##### Q79) What are some control flow statements in SQL?

while, if, case, for each etc..

##### Q80) What is the EXISTS function used for?

It is used to determine whether a query returns one or more rows. If it does, the EXIST function returns TRUE, otherwise, it will return FALSE.

## SQL Query Interview Questions with Answers

**Inner Join:** It is used to retrieve matching records from both the tables

**Department:**

|  |  |
| --- | --- |
| **Department\_No** | **Department\_Name** |
| 10 | ECE |
| 20 | ECE |
| 30 | CSE |
| 40 | IT |

**Employee Details:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Employee\_No** | **Emp\_Name** | **Address** | **Age** | **Department\_No** | **Salary** |
| 1 | Anil | Hyderabad | 23 | 10 | 20000 |
| 2 | Sunil | Hyderabad | 22 | 10 | 21000 |
| 3 | Ajay | Chennai | 24 | 20 | 23000 |
| 4 | Vijay | Chennai | 25 | 30 | 22000 |
| 5 | James | Hyderabad | 24 | 50 | 230000 |

##### Q1) Write a Query to display employee details who are working in ECE department?

|  |  |
| --- | --- |
| 1  2  3  4 | SELECT employee.employee\_name, employee.address, employee.salary, employee.age,  FROM Department D  INNER JOIN Employees E  ON department.D\_no=employee.D\_no WHERE department.D\_name= ‘ECE’ |

##### Q2) Write a Query to display employee details?

|  |  |
| --- | --- |
| 1 | SELECT \* FROM employee; |

##### Q3)  Write a Query to display employee details along with department\_name?

|  |  |
| --- | --- |
| 1  2  3  4 | SELECT employee.employee\_no, employee.employee\_name, employee.address, employee.salary, employee.age, department.department\_name  FROM department D  INNER JOIN employee E  ON department.D\_no=employee.D\_no |

##### Q4) Write a Query to display employee details whose sal>20000 and who is working in ECE department?

|  |  |
| --- | --- |
| 1  2  3  4  5 | SELECT employee.employee\_no, employee.employee\_name, employee.address, employee.salary, employee.age  FROM department D  INNER JOIN employee E  ON dept.D\_no=emp.D\_no  WHERE dept.D\_name=’ECE’ and E.salary>20000 |

##### 5) Write a Query to display employee details along with department\_name and who is working in ECE department, whose name starts with a?

|  |  |
| --- | --- |
| 1  2  3  4  5 | SELECT emp.e\_no, emp.e\_name, emp.address, emp.salary, emp.age, dept.dname  FROM department D  INNER JOIN employee E  ON dept.D\_no=emp.D\_no  WHERE dept.D\_name=’ECE’ and emp.E\_name like ‘a%’ |

##### Q6) Write a Query to display employee details along with department\_name and whose age between 20 and 24?

|  |  |
| --- | --- |
| 1  2  3  4  5 | SELECT emp.e\_no, emp.e\_name, emp.address, emp.salary, emp.age, dept.d\_name  FROM department D  INNER JOIN employee E  ON dept.D\_no=emp.D\_no  WHERE E.age between 20 and 24 |

##### Q7) Write a Query to display employee details along with department\_name and who are staying in hyderabad?

|  |  |
| --- | --- |
| 1  2  3  4  5 | SELECT emp.e\_no, emp.e\_name, emp.address, emp.salary, emp.age, dept.d\_name  FROM department D  INNER JOIN employee E  ON dept.D\_no=emp.D\_no  WHERE E.address=’hyd’ |

##### Q8)  Write a Query to display employee details whose salary>20000 and whose age>20 & who is working in ECE department?

|  |  |
| --- | --- |
| 1  2  3  4  5 | SELECT emp.e\_no, emp.e\_name, emp.address, emp.salary, emp.age, dept.d\_name  FROM department D  INNER JOIN employee E  ON dept.D\_no=emp.D\_no  WHERE E.age>20 and E.salary>20000 and dept.D\_name=’ECE’ |

**State Table:**

|  |  |
| --- | --- |
| **State ID** | **State Name** |
| S1 | Telangana |
| S2 | AP |
| S3 | Tamil Nadu |
| S4 | Karnataka |
| S5 | Kerala |

**City**

|  |  |  |
| --- | --- | --- |
| **City ID** | **City Name** | **State ID** |
| 1 | Hyderabad | S1 |
| 2 | Vizag | S2 |
| 3 | Vijayawada | S2 |
| 4 | Chennai | S3 |
| 5 | Madhurai | S3 |
| 6 | Bangalore | S4 |

**Blood Group Details**

|  |  |
| --- | --- |
| **Blood Group ID** | **Blood Group** |
| B1 | A+ve |
| B2 | B+ve |
| B3 | AB +ve |
| B4 | A -ve |
| B5 | O +ve |

**Donor Details**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Donor ID** | **Donor Name** | **Phone Number** | **City ID** | **Blood Group ID** |
| D1 | Anil | 9999 | 1 | B1 |
| D2 | Sunil | 8888 | 1 | B1 |
| D3 | Ajay | 7777 | 2 | B1 |
| D4 | John | 6666 | 4 | B3 |
| D5 | James | 5555 | 4 | B5 |

##### Q9) Write a Query to display city names belongs to AP?

|  |  |
| --- | --- |
| 1  2  3  4  5 | SELECT C.City\_Name  FROM State S  INNER JOIN City C  ON S.State\_ID  WHERE S.State\_Name ‘AP’ |

##### Q10) Write a Query to display Donor\_ID, Donor\_Name, Phone No, City?

|  |  |
| --- | --- |
| 1  2  3  4 | SELECT D.Donor\_ID, D\_Name, D\_Phone No, C.City\_Name  FROM Donor D  INNER JOIN City C  ON D.City\_ID=C.City\_ID |

###### Q11) Write a Query to display Donor\_ID, Donor\_Name, Phone No, Blood Group?

|  |  |
| --- | --- |
| 1  2  3  4 | SELECT D.Donor\_ID, D\_Name, D\_Phone No, B.Blood\_Group  FROM Donor D  INNER JOIN Blood B  ON D.Blood\_ID=B.Blood\_ID; |

###### Q12) Write a Query to display Donor\_ID, Donor\_Name, Phone No and who are staying in hyderabad?

|  |  |
| --- | --- |
| 1  2  3  4  5 | SELECT D.Donor\_ID, D\_Name, D\_Phone No, C.City\_Name  FROM Donor D  INNER JOIN City C  ON C.City\_ID=D.City\_ID  WHERE C.City\_Name=’hyderabad’ |

###### Q13) Write a Query to display donor details whose blood group is A +ve?

|  |  |
| --- | --- |
| 1  2  3  4  5 | SELECT D.Donor\_ID, D\_Name, D\_Phone No  FROM Donor D  INNER JOIN Blood B  ON D.Donor\_ID=B.Blood\_ID  WHERE B.Blood\_Group=’A+ve’ |

###### Q14) Write a Query to display Donor\_ID, Donor\_Name, Phone No, City, Blood Group?

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | SELECT D.Donor\_ID, D\_Name, D\_Phone No, C.City\_Name B.Blood\_Group  FROM Blood B  INNER JOIN Donor D  ON D.Blood\_ID=B.Donor\_Name  INNER JOIN City C  ON D.City\_ID=C.City\_ID |

###### Q15) Write a Query to display Donor\_Name, Phone No, Blood Group of the donors who is staying in hyderabad and whose blood group is A+ve?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | SELECT D.Donor\_Name, D. Phone\_Number, B.Blood\_Group  FROM Donor D  INNER JOIN Blood B  ON D.Blood\_ID=B.Blood\_ID  INNER JOIN City C  ON D.City\_ID=C.City\_ID  WHERE C.City\_Name=’hyderabad’ and B.Blood\_Group=’A+ve’ |

**Outer Join** A join that includes rows even if they do not have related rows in the joined table is an Outer Join.. You can create three different outer join to specify the unmatched rows to be included:

* Left Outer Join
* Right Outer Join
* Full Outer Join

**Employee Details Table**

|  |  |  |
| --- | --- | --- |
| **Employee\_No** | **Employee\_Name** | **Dept\_No** |
| 101 | Anil | 10 |
| 102 | Sunil | 20 |
| 103 | Ajay | 30 |
| 104 | Vijay | 40 |
| 105 | Null | Null |

**Department Details Table**

|  |  |
| --- | --- |
| **Dept\_No** | **Depat\_Name** |
| 10 | EEE |
| 20 | EEE |
| 30 | CSE |
| Null | Null |
| 50 | IT |

###### Q16) Write a Query to display only left records?

|  |  |
| --- | --- |
| 1  2  3  4  5 | SELECT e.\*  FROM Employee E  LEFT OUTER JOIN Department D  ON E.D\_no  WHERE D.D\_No IS NULL |

###### Q17) Write a Query to display employee details where employee no is 101?

|  |  |
| --- | --- |
| 1  2  3 | SELECT \*  FROM Employee E  WHERE E\_No=101 |

###### Q18) Write a Query to display employee details where employee number is null?

|  |  |
| --- | --- |
| 1  2  3 | SELECT \*  FROM Employee E  WHERE E\_No IS NULL |

##### Q19) Write a Query to display only right records?

|  |  |
| --- | --- |
| 1  2  3  4  5 | SELECT D.\*  FROM Employee E  RIGHT OUTER JOIN Department D  ON E.D.No=D.D\_No  WHERE E.D\_No IS NULL |

##### Q20) Write a Query to display all the records from the table except matching records?

|  |  |
| --- | --- |
| 1  2  3  4  5 | SELECT E.\*, D.\*  FROM Employee E  FULL JOIN Department D  ON E.D\_No=D.D\_No  WHERE E.D\_No IS NULL or D.D\_No IS NULL |

**Department Details Table**

|  |  |
| --- | --- |
| **Dept\_No** | **Dept\_Name** |
| 1 | ECE |
| 2 | CSE |
| 3 | EEE |

**Course Details Table**

|  |  |  |
| --- | --- | --- |
| **Course\_ID** | **Course\_Name** | **Cr** |
| 1 | EDC | 4 |
| 2 | PDC | 4 |
| 3 | SS | 4 |
| 4 | DAA | 4 |
| 5 | OS | 4 |

**Student Details Table**

|  |  |
| --- | --- |
| **Student\_No** | **Student\_Name** |
| 101 | Anil |
| 102 | Sunil |
| 103 | Ajay |
| 104 | Vijay |
| 105 | John |

**Enroll Details Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Enroll\_Date** | **Student\_No** | **Dpet\_No** | **S\_ID** |
| 1/2/2014 | 101 | 10 | S1 |
| 3/2/2016 | 102 | 10 | S1 |
| 3/2/2016 | 103 | 10` | S1 |
| 3/2/2016 | 104 | 20 | S2 |
| 3/2/2016 | 105 | 20 | S2 |

**Address Table**

|  |  |
| --- | --- |
| **Emp\_No** | **Address** |
| E1 | Hyderabad |
| E2 | Vizag |
| E3 | Hyderabad |
| E4 | Bangalore |
| E5 | Hyderabad |

**Employee Details Table**

|  |  |
| --- | --- |
| **Emp\_No** | **Emp\_Name** |
| E1 | Arun |
| E2 | Kiran |
| E3 | Kumar |
| E4 | Anus |
| E5 | James |

**Semester Details Table**

|  |  |
| --- | --- |
| **Semester** | **Sn** |
| S1 | 1 |
| S2 | 2-1 |
| S3 | 2-2 |
| S4 | 3-1 |
| S5 | 3-2 |
| S6 | 4-1 |
| S7 | 4-2 |

**Course Department Details**

|  |  |
| --- | --- |
| **Dept\_No** | **Course\_ID** |
| 10 | 1 |
| 10 | 2 |
| 10 | 3 |
| 20 | 4 |
| 20 | 5 |

**Syllabus Table**

|  |  |  |
| --- | --- | --- |
| **Dept\_No** | **Course\_ID** | **S\_ID** |
| 10 | 1 | S1 |
| 10 | 2 | S1 |
| 10 | 3 | S1 |
| 20 | 4 | S2 |
| 20 | 5 | S2 |

**Instructor Details Table**

|  |  |
| --- | --- |
| **Emp\_No** | **Dept\_No** |
| E1 | 10 |
| E2 | 10 |
| E3 | 10 |
| E4 | 20 |
| E5 | 30 |

**Course Instructor Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Course\_ID** | **Emp\_No** | **S\_ID** | **Dept\_No** |
| 1 | E1 | S1 | 10 |
| 1 | E1 | S1 | 20 |
| 1 | E2 | S1 | 30 |
| 2 | E3 | S1 | 10 |
| 4 | E4 | S2 | 20 |
| 5 | E4 | S2 | 20 |
| 5 | E5 | S1 | 10 |

##### Q) Write a query to display Student No, Student Name, Enroll Date, Department Name?

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | SELECT S.Student\_No,   S.Student\_Name, S.Enroll\_Date, D.Dept\_Name  FROM Student S  INNER JOIN Enroll E  ON S.Student\_No=E.Student\_No  INNER JOIN Department D  ON D.Dept\_No=E.Dept\_No |

##### Q) Write a query to display Employee Number, Employee Name and address, department name?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | SELECT E.Emp\_No, E.Emp\_Name, A.Address, D.Dept\_Name  FROM Employee E  INNER JOIN Address A  ON E.Emp\_No=A.Emp\_No  INNER JOIN Instructor I  ON A.Emp\_No=I.Emp\_No  INNER JOIN Department D  ON I.Dept\_No=D.Dept\_No |

##### Q) Write a query to display course name belongs to ECE department?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | SELECT C.Course\_Name  FROM Department D  INNER JOIN Course Department CD  ON D.Dept\_NO=CD.Dept\_NO  INNER JOIN Course C  ON CD.CourseDept\_ID=C.Course\_ID  WHERE D.Dept\_Name=’ECE’ |

##### Q) ) Write a query to display student number, student name, enroll date, dept name,  semester name?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | SELECT S.Student\_No, S.Student\_Name, S.Enroll\_Date, D.Dpet\_Name, Sem.Student\_Name  FROM Enroll E  INNER JOIN Student S  ON S.Student\_No=E.Student\_No  INNER JOIN Deprtment D  ON E.Dept\_No=D.Dept\_No  INNER JOIN Semester SE  ON E.Student\_ID=Sem.Student\_ID |

###### Q) Write a query to display the syllabus of ECE department 1st year?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | SELECT C.Course\_Name  FROM Department D  INNER JOIN Syllabus Sy  ON D.Dept\_No=Sy.Dept\_No  INNER JOIN Course C  ON Sy.Course\_ID=C.Course\_ID  INNER JOIN Semester Se  ON Syllabus\_Sy\_ID=Se\_Sy\_ID  WHERE D.Dept\_Name=’ECE’ and Se.Semester=’1’ |

###### Q)  Write a query to display the employee names and faculty names of ECE dept 1st year?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | SELECT E.Emp\_Name  FROM Employee E  INNER JOIN Course Instructor Ci  ON E.Emp\_No=Ci.Emp\_No  INNER JOIN Semester Se  ON Se.Student\_ID=Ci.Student\_ID  INNER JOIN Dept D  ON Ci.Dept\_No=D.Dept\_No  WHERE D.Dept\_Name=’ECE’ and Se.Student\_Name=’1’ |

###### Q) ) Write a query to display student details who enrolled for ECE department?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | SELECT S.Student\_NO, S.Student\_Name, S.Enroll\_Date  FROM Student S  INNER JOIN Enroll E  ON S.Student\_No=E.Student\_No  INNER JOIN Department D  ON E.Dept\_No=D.Dept\_No  WHERE D.Dept\_Name=’ECE’ |

###### Q) ) Write a query to display student details along with dept name who are enrolled in ECE department first year?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | SELECT S.Student\_No, S.Student\_Name, S.Enroll\_Date, D.Dept\_Name  FROM Student S  INNER JOIN Enrollment E  ON S.Student\_No=E.Student\_No  INNER JOIN Department D  ON D.Dept\_No=E.Dept\_No  INNER JOIN Semester Se  ON E.Student\_ID=Se.Student\_ID  WHERE D.Dept\_Name=’ECE’ and Se.Student\_Name=’1’ |

###### Q) ) Write a query to display employee name who is teaching EDC?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | SELECT E.Emp\_Name  FROM Employee E  INNER JOIN Course Instructor Ci  ON E.Emp\_No=Ci.Emp\_No  INNER JOIN Course C  ON Ci.Course\_ID=C.Course\_ID  WHERE C.Course\_Name=’EDC’ |

###### Q) ) Write a query to display employee details along with dept name who are staying in Hyderabad?

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | SELECT E.Emp\_No, Emp\_Name, D.Dept\_Name  FROM Employee E  INNER JOIN Address A  ON E.Emp\_No=A.Emp\_No  INNER JOIN Instructor I  ON A.Emp\_No=I.Emp\_No  INNER JOIN Department D  ON I.Dept\_No=D.Dept\_No  WHERE A.Address=’hyderabad’ |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Emp\_No** | **Emp\_Name** | **Salary** | **Age** | **Dept\_Name** |
| 101 | Anil | 20,000 | 22 | ECE |
| 102 | Sunil | 23000 | 23 | EEE |
| 103 | Vijay | 32000 | 24 | CSE |

**Using Range Operator**:: BETWEEN, NOT BETWEEN

##### Q) Write a Query to display employee details whose salary > 20000 and whose age >23?

|  |  |
| --- | --- |
| 1  2 | SELECT \* FROM Employee  WHERE Salary>20000 AND Age>23; |

###### Q) Write a Query to display employee details whose salary >20000 and who is working in ECE department?

|  |  |
| --- | --- |
| 1  2 | SELECT \* FROM Employee  WHERE Salary>20000 AND Dept\_Name=’ECE’ |

###### Q) Write a Query to display employee details whose age is BETWEEN 18 and 22?

|  |  |
| --- | --- |
| 1  2 | SELECT \* FROM Employee Details  WHERE Age BETWEEN 18 AND 22; |

###### Q) Write a Query to display employee details whose salary range BETWEEN 20000 and 23000?

|  |  |
| --- | --- |
| 1  2 | SELECT \* FROM Employee  WHERE Salary BETWEEN 20000 AND 23000; |

###### Q) Write a Query to display employee details whose age is NOT BETWEEN 18 & 22?

|  |  |
| --- | --- |
| 1  2 | SELECT \* FROM Employee  WHERE Age NOT BETWEEN 18 AND 22; |

**Using String Operators**:: LIKE, NOT LIKE

##### Q) Write a Query to display employee details whose name starts with a?

|  |  |
| --- | --- |
| 1  2 | SELECT \* FROM Employee  WHERE Emp\_Name LIKE ‘a%’ |

a%  ----> starts with a  
%a  ----> ends with a

###### Q) Write a Query to display employee details and whose age>20 & whose name starts with a?

|  |  |
| --- | --- |
| 1  2 | SELECT \* FROM Employee  WHERE Salary>20000 AND Age>20 AND Emp\_Name LIKE ‘a%’ |

###### Q) Write a Query to display employee details whose name not starts with a?

|  |  |
| --- | --- |
| 1  2 | SELECT \* FROM employee  WHERE Emp\_Name NOT LIKE ‘a%’ |