**SQL Interview Questions**

1. What is Database?

2. What is DBMS?

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

4. What is SQL?

5. What is the difference between SQL and MySQL?

6. What are Tables and Fields?

7. What are Constraints in SQL?

8. What is a Primary Key?

9. What is a UNIQUE constraint?

10. What is a Foreign Key?

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

12. What is a Self-Join?

13. What is a Cross-Join?

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

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

16. What is Data Integrity?

17. What is a Query?

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

19. What is the SELECT statement?

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

21. What are UNION, MINUS and INTERSECT commands?

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

23. What are Entities and Relationships?

24. List the different types of relationships in SQL.

25. What is an Alias in SQL?

26. What is a View?

27. What are the TRUNCATE, DELETE and DROP statements?

28. What is the difference between DROP and TRUNCATE statements?

29. What is the difference between DELETE and TRUNCATE statements?

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.

1. **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 software.

1. **What is RDBMS?**

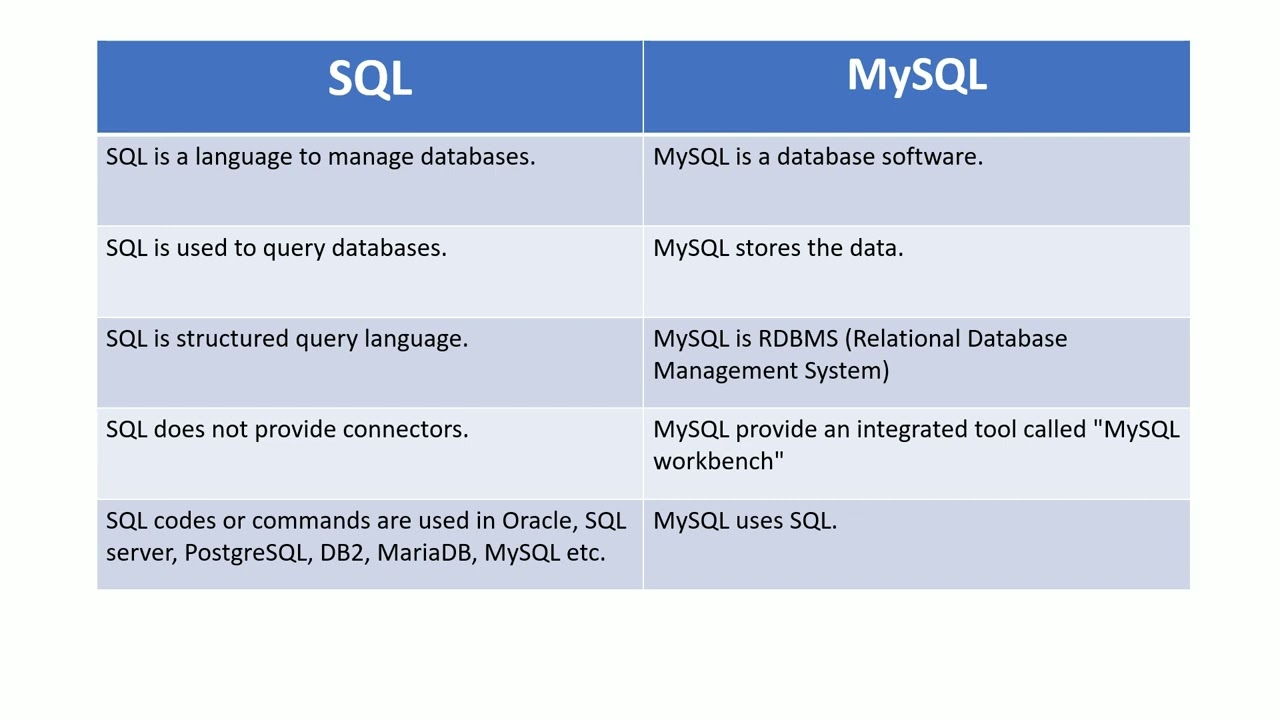
How is it different from DBMS? RDBMS stands for Relational Database Management System. The key difference here, compared to DBMS, is that RDBMS stores data in the form of a collection of tables, and relations can be defined between the common fields of these tables. Most modern database management systems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

1. **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.

1. **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(protocols/rules) 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 the creation of the table or are creating 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);*

**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 refers 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 labeled as the child table, and the table containing the candidate key is labeled 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);*

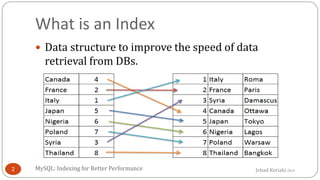
**14. What is an Index? Explain its different types.**

A database index is a data structure that provides a 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 \*/*



**16. What is Data Integrity?**

Data Integrity is the assurance of accuracy and consistency of data over its entire lifecycle and is a critical aspect of 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 a 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');*

**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 conjunction 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 the 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;*

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

A database cursor is a control structure that allows for the traversal of records in a database. Cursors, in addition, facilitates processing aer 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:**

1. DECLARE a cursor aer any variable declaration. The cursor declaration must

always be associated with a SELECT Statement.

2. Open cursor to initialize the result set. The OPEN statement must be called

before fetching rows from the result set.

3. FETCH statement to retrieve and move to the next row in the result set.

4. Call the CLOSE statement to deactivate the cursor.

5. 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 employee's 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;*

**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 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 are deleted and the table structure is removed from the database.

*DROP TABLE Candidates;*

**28. 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.

**29. 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.