Many-to-One

An entity in A is associated to no more than one entity in B in this particular cardinality mapping. Or, we may say that any number (zero or more) of entities or things in A can be connected to a unit or thing in B.

Example: A single surgeon performs many operations in a specific institution. A many-to-one relationship is one of these relationships.

Many-to-Many

A many-to-many relationship means that each instance of one entity can be associated with multiple instances of the other entity, and each instance of the other entity can also be associated with multiple instances of the first entity. This is the most common type of relationship, and it is often used to represent relationships where the order of the entities does matter, such as a student and their courses.

Example: A student can take multiple courses, and each course can be taken by multiple students.

Introduction to SQL:

The computer programming language SQL (Structured Query Language) was developed especially for managing and changing relational databases. It provides commands and statements to connect to databases, retrieve and modify data, construct database structures, and perform numerous data tasks. More details are provided below:

Definition and Purpose of SQL:

According to its definition and intended application, SQL is a declarative language utilized for relational database management. By constructing queries, it enables users to interact with databases to access, alter, and manage structured data. No matter what database management system is used underneath, SQL provides a standardized and efficient method for working with databases.

Why SQL?

Due to its adaptability and efficiency in maintaining relational databases, SQL is frequently used in data science and analytics. The main justifications for SQL's high value are as follows:

- The core activities of inserting, updating, and deleting data in relational databases are made available to data professionals via SQ. It gives a simple and effective method for changing data.
- SQL gives customers the ability to get particular data from relational database management systems. Users can provide criteria and conditions to retrieve the desired information by creating SQL queries.
- SQL is useful for expressing the structure of stored data. Users can define the structure, data types, and relationships of database tables as well as add, change, and delete them.
- SQL gives users the ability to handle databases and their tables efficiently. In order to increase the functionality and automation of database operations, it facilitates the construction of views, stored procedures, and functions.
- SQL gives users the ability to define and edit data that is kept in a relational database Data constraints can be specified by users, preserving the integrity and consistency of the data.
- Data Security and Permissions: SQL has tools for granting access to and imposing restrictions on table fields, views, and stored procedures. Giving users the proper access rights promotes data security.

SQL constraints

- Rules for the data in a table can be specified using SQL constraints.
- The kinds of data that can be entered into a table are restricted by constraints. This guarantees the reliability and accuracy of the data in the table. The action is stopped if there is a violation between the constraint and the data action.
- Column-level or table-level constraints are both possible. Table level restrictions apply to the entire table, while column level constraints just affect the specified column.

In SQL, the following restrictions are frequently applied:

NOT NULL: A column cannot have a NULL value by using the NOT NULL flag.
UNIQUE: A unique value makes sure that each value in a column is distinct.
PRIMARY KEY: A NOT NULL and UNIQUE combination. Identifies each table row in a unique way.
FOREIGN KEY: Prevent acts that would break linkages between tables.
CHECK - Verifies if the values in a column meet a certain requirement.
DEFAULT: If no value is specified, DEFAULT sets a default value for the column.
CREATE INDEX - Used to easily create and access data from the database.

NOT NULL constraint

- A column may by default contain NULL values.
- A column must not accept NULL values according to the NOT NULL constraint.
- This forces a field to always have a value, thus you cannot add a value to this field while adding a new record or updating an existing record.
  Syntax:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255) NOT NULL,
    price int);
```

Check Constraint:

- The value range that can be entered into a column is restricted by the CHECK constraint.
- Only specific values will be permitted for a column if you define a CHECK constraint on it.
- A table's CHECK constraint can be used to restrict the values in specific columns based on the values of other columns in the same row.

Example: Create check constraint:

```
CREATE TABLE employee (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    CHECK (Age>=18));
```

SQL DEFAULT CONSTRAINT:

- A column's default value is set using the DEFAULT constraint.
- If no alternative value is supplied, the default value will be appended to all new records.
- Example: Create Default.

```sql
CREATE TABLE EMPLOYEE (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    City varchar(255) DEFAULT 'LONDON');
```

By utilizing operations like GETDATE, the DEFAULT constraint can also be utilized to insert system data ()

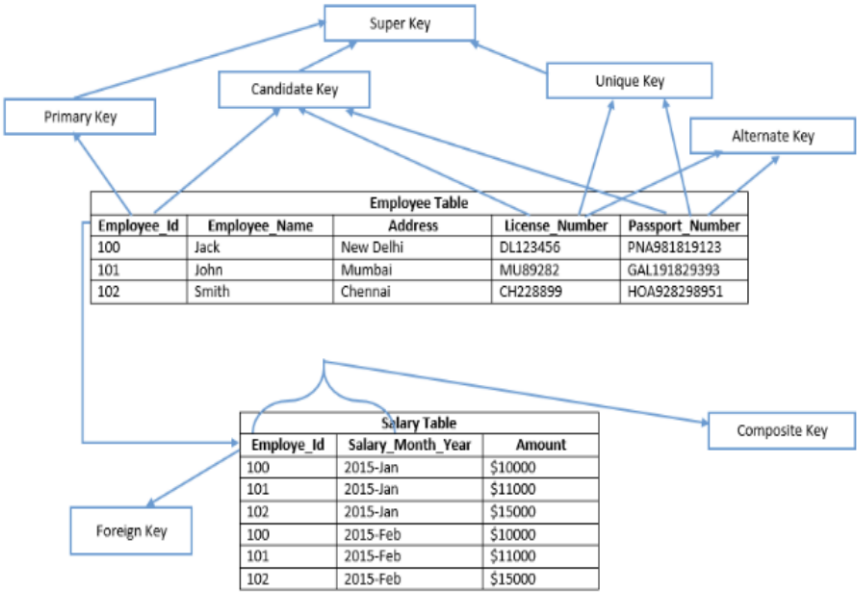
Various key types

We'll talk about keys and the many types that exist in SQL Server. Let's define keys to get this topic started.

What is the Key?

In RDBMS systems, keys are fields that take part in the following operations on tables:
To establish connections between two tables.

- To keep a table's individuality.
- To maintain accurate and consistent data in the database.
- Possibly speed up data retrieval by enabling indexes on column (s).



Types of keys

The following list includes the many key types that SQL Server supports:

- Candidate Key
- Primary Key
- Unique Key
- Alternate Key
- Composite Key
- Super Key
- Foreign Key

Candidate Key:

A candidate key is a table's primary key that has the potential to be chosen. There may be several candidate keys in a table, but only one can be chosen to serve as the main key.
Example: The candidate keys are License Number, Employee Id, , and Passport Number.

Primary Key

- The table's primary key was chosen as a candidate key to uniquely identify each record. Primary keys maintain unique values throughout the column and do not permit null values. Employee Id is the primary key of the Employee table in the example above. In SQL Server, a heap table's main key automatically builds a clustered index (a table which does not have a clustered index is known as a heap table). A table's nonclustered primary key can also be defined by explicitly specifying the kind of index.
- A table can have only one primary key, in SQL Server, the primary key can be defined using SQL commands below:
- CRETE TABLE statement (at the time of table creation) – In this case, system defines the name of primary key
- ALTER TABLE statement (using a primary key constraint) – User defines the name of the primary key
  Example: Employee_Id is a primary key of Employee table.

Unique Key

- Similar to a primary key, a unique key prevents duplicate data from being stored in a column In comparison to the primary key, it differs in the following ways:
- One null value may be present in the column.
- On heap tables, it by default creates a nonclustered index.

Alternate Key

- The alternate key is a potential primary key for the table that has not yet been chosen.
- For instance, other keys include License Number and Passport Number.

Composite Key

Each row in a table is uniquely identified by a composite key, sometimes referred to as a compound key or concatenated key. A composite key's individual columns might not be able to identify a record in a certain way. It may be a candidate key or a primary key.

Example: To uniquely identify each row in the salary database, Employee Id and Salary Month Year are merged. Each entry cannot be individually identified by the Employee Id or Salary Month Year columns alone. The Employee Id and Salary Month Year columns in the Salary database can be used to build a composite primary key.

Super Key

- A super key is a group of columns from which the table's other columns derive their functional dependence. Each row in a table is given a unique identification by a collection of columns. Additional columns that are not strictly necessary to identify each row uniquely may be included in the super key. The minimal super keys, or subset of super keys, are the primary key and candidate keys.

make up things when answering them.

Don't worry if your experience in SQL is limited: this is something your interviewer, most probably, already knows from your resume. Since they are interested in talking to you anyway, your profile was considered a good fit for their company.

Also, it's perfectly fine if you have only worked with one SQL flavor. Remember that all SQL dialects are fairly similar among themselves. Therefore, being familiar with only one of them is a solid basis for you to learn any others.

Technical SQL Interview Questions for Beginners

Now, let's move on to the technical SQL interview questions and some potential answers to them.

When answering technical questions, the best strategy is to give as precise answers as possible. It may look like an attempt to deviate from the main topic. In addition, it may provocate additional questions about which you can feel less confident.

1. What is SQL?

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

2. What are SQL dialects? Give some examples.

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

3. What are the main applications of SQL?

Using SQL, we can:

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

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

their combined power.

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

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

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

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

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

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

7. What is a database?

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

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

It stands for Database Management System, a software package used to perform various operations on the data stored in a database, such as accessing, updating,

wrangling, inserting, and removing data. There are various types of DBMS, such as relational, hierarchical, network, graph, or object-oriented. These types are based on the way the data is organized, structured, and stored in the system.

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

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

10. What are tables and fields in SQL?

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

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

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

12. What is a subquery?

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

13. What types of SQL subqueries do you know?

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

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

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

15. What SQL constraints do you know?

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

16. What is a join?

A clause used to combine and retrieve records from two or multiple tables. SQL tables can be joined based on the relationship between the columns of those tables. Check out our **SQL joins** tutorial for more context.

17. What types of joins do you know?

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

18. What is a primary key?

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

19. What is a unique key?

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

20. What is a foreign key?

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

21. What is an index?

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

22. What types of indexes do you know?

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

23. What is a schema?

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

24. What is a SQL comment?

A human-readable clarification on what a particular piece of code does. SQL code comments can be single-line (preceded by a double dash `--`) or span over multiple lines (as follows: `/*comment_text*/`). When the SQL engine runs, it ignores code