Top 40 SQL Interview Questions

(INTERMEDIATE - PART 2)

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

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

2. What types of SQL functions do you know?

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

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

3. What aggregate functions do you know?

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

4. What scalar functions do you know?

- `LEN()` (in other SQL flavors – `LENGTH()`) – returns the length of a string, including the blank spaces
- `UCASE()` (in other SQL flavors – `UPPER()`) – returns a string converted to the upper case

- `LCASE()` (in other SQL flavors – `LOWER()`) – returns a string converted to the lower case
- `INITCAP(`) – returns a string converted to the title case (i.e., each word of the string starts from a capital letter)
- `MID()` (in other SQL flavors – `SUBSTR()`) – extracts a substring from a string
- `ROUND()` – returns the numerical value rounded to a specified number of decimals
- `NOW()` – returns the current date and time

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

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

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

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

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

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

- `TRIM()` – removes all the defined characters, as well as white spaces, from the left, right, or both ends of a provided string

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

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

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

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

```
SELECT * FROM table_name
ORDER BY col_1 DESC;
```

9. What set operators do you know?

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

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

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

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

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

12. What is a composite primary key?

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

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

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

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

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

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

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

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

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

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

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

18. What types of SQL relationships do you know?

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

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

In some SQL flavors, such as PostgreSQL, the BOOLEAN data type exists explicitly and takes values `TRUE`, `FALSE`, or `NULL`. In other flavors, such as Microsoft SQL

Server, the BIT datatype is used to store Boolean values as integers `1` (true) or `0` (false).

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

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

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

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

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

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

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

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

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

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

25. What is the `CASE()` function?

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

```
CASE
    WHEN condition_1 THEN value_1
    WHEN condition_2 THEN value_2
    WHEN condition_3 THEN value_3
    ...
    ELSE value
END;
```

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

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

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

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

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

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

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

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

```
INSERT INTO table_name
VALUES (value_1, value_2, ...);
```

30. How to delete a record from a table?

Using the `DELETE` statement. The syntax is:

```
DELETE FROM table_name
WHERE condition;
```

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

31. How to add a column to a table?

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

```
ALTER TABLE table_name
ADD column_name datatype;
```

32. How to rename a column of a table?

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

```
ALTER TABLE table_name
RENAME COLUMN old_column_name TO new_column_name;
```

33. How to delete a column from a table?

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

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

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

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

```
SELECT * FROM table_name
WHERE MOD(ID_column, 2) = 0;
```

To select all even records using `%`:

```
SELECT * FROM table_name
WHERE ID_column % 2 = 0;
```

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

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

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

36. How to insert many rows in a table?

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

```
INSERT INTO table_name
VALUES (value_1, value_2, ...),
      (value_3, value_4, ...),
      (value_5, value_6, ...),
      ...;
```

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

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

```
SELECT * FROM table_name
ORDER BY column_name DESC
LIMIT 1
```

```
OFFSET 5;
```

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

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

```
SELECT * FROM table_name
WHERE surname LIKE 'A_';
```

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

```
SELECT * FROM table_name
WHERE surname LIKE 'A%';
```

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

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

```
SELECT id
FROM table_name
ORDER BY id DESC
LIMIT 1;
```

or in Microsoft SQL Server:

```
SELECT TOP 1 id
FROM table_name
ORDER BY id DESC
```

40. How to select random rows from a table?

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

```
SELECT * FROM table_name
ORDER BY RAND()
LIMIT 5;
```

Key Takeaways

To sum up, we discussed the 80 essential beginner and intermediate SQL interview questions and the right answers to them. Hopefully, this information will help you to get ready for the interview and feel more confident, whether you're looking for a job in SQL or hiring candidates for an intermediate SQL position.

I hope this helps!😀