SQL Interview Questions You Must Prepare: The Ultimate Guide

July 6th, 2021

Authors

Nathan Rosidi

Hey there, I'm the founder of StrataScratch. We're a small team trying to create resources for aspiring and current data scientists. I dabble in a bit of everything but do most of the content creation while my team works hard to improve your experience.

Categories

SQL

Interviews

Data Science Programming Languages

Share

Follow

Are you wondering what SQL interview questions you will be asked? This ultimate guide will take you through the top SQL interview questions for various data positions and the tips to approach your next SQL interview.

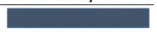
SQL is a must have tool in the arsenal of any aspiring data scientist. In this article we provide an outline to learn, prepare and ace your next SQL Interview for a Data Science role. We will explore why SQL is so widely used, then provide you a breakdown of SQL skills needed by each role viz – Data Analyst, Data Scientist, Machine Learning Engineer, etc. Further, we provide you with real interview examples from the StrataScratch Platform illustrating a few of these skills and provide you with a step-by-step learning guide to become proficient with SQL even if you are not too familiar with SQL concepts and get your dream job.

So let us start off with why SQL is so widely used in the Data Science World.

The Prevalence of SQL

One of the biggest reasons for the popularity of SQL is the tabular format for storage of data. It is easy to visualize databases as large spreadsheets with millions and millions of rows and columns. SQL allows users to quickly manipulate these tables to access information and present the results in the most common formats, tables and associated graphs and visualizations.

DB-Engines Ranking - May 2021

Rank	Name	Type	May-21	Chart May 2021
1	Oracle	Relational, Multi-model	1269.94	
2	MySQL	Relational, Multi-model	1236.38	
3	Microsoft SQL Server	Relational, Multi-model	992.66	
4	PostgreSQL	Relational, Multi-model	559.25	
5	MongoDB	Document, Multi-model	481.01	
6	IBM Db2	Relational, Multi-model	166.66	
7	Redis	Key-value, Multi-model	162.17	
8	Elasticsearch	Search engine, Multi-model	155.35	
9	SQLite	Relational	126.69	
10	Microsoft Access	Relational	115.4	

While No-SQL databases like MongoDB, Cassandra, etc have gained traction with the requirements of Big-Data and real time applications and increasing prevalence of unstructured data, SQL databases still hold the top seven positions of the top ten most popular database engines. In fact No-SQL databases have positioned themselves as No just SQL to highlight their support for SQL so as to help increase their acceptance in organizations where traditional SQL based databases are already used.

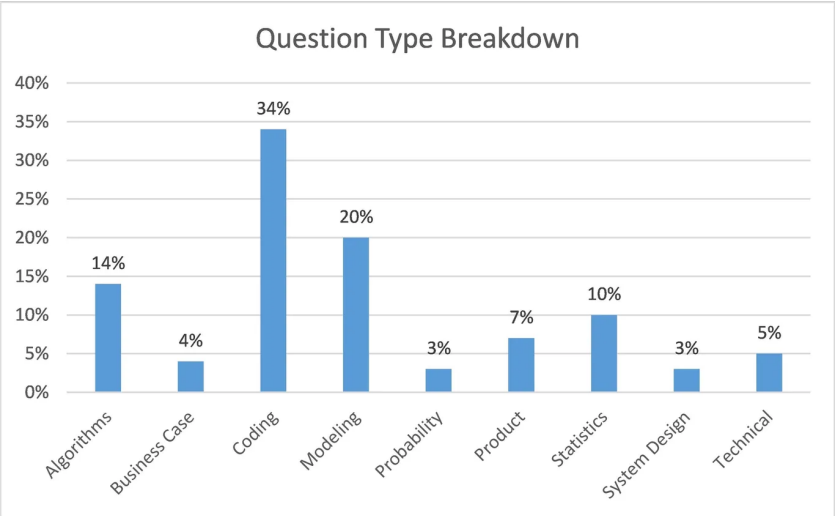
Another reason for the popularity of SQL is the ease of use. The syntax is very easy to understand, and anyone can pick it up very quickly. It is very easy to comprehend what the below statements do.

```
SELECT name FROM EMPLOYEES;
```

```
SELECT name FROM EMPLOYEES
WHERE age >= 35;
```

```
SELECT state, AVG(age) as average_age FROM EMPLOYEES
WHERE age >= 35
GROUP BY state;
```

It is pretty easy to learn SQL and a convenient yet powerful tool for the hiring managers to evaluate reasoning, coding and data science aptitudes of potential employees. Our analysis of 903 data science interview questions showed the most prominent interview question category is coding. It comprises almost a third of all the interview questions, since more often than not, the most prominent concept tested in the coding segment was writing SQL queries.



There are more SQL based interview questions asked than those on Python or R at these top companies. With increasing focus on Data management skills, you cannot go wrong with learning SQL. So what should one concentrate on while building up their SQL skills for Data Science and Allied Positions?

SQL Usage in Data Science by Role

Not every position in Data Science uses the same SQL concepts. While some roles will focus heavily on queries and query optimization, others will tend towards data architecture and ETL processes. One can divide SQL Commands asked in Data Science Interviews into the following categories.

- Data Definition Language (DDL)
  - CREATE
  - ALTER
  - DROP
  - RENAME
  - TRUNCATE
  - COMMENT
- Data Query Language (DQL)
  - SELECT
- Data Manipulation Language (DML)
  - INSERT
  - UPDATE
  - DELETE
  - MERGE
  - CALL
  - EXPLAIN PLAN
  - LOCK TABLE
- Data Control Language (DCL)
  - GRANT
  - REVOKE

A Data Analyst or a Data Scientist will be expected to mostly be working with the SELECT statement and associated advanced level concepts like subqueries, Grouping / Rollups, Window Functions and CTEs. If you work as an analyst, you probably already use SQL. It does not matter if you are a data analyst, reporting analyst, product analyst, or even financial analyst. Your position generally requires handling the raw data and using your skills to provide the management and other stakeholders with the decision-making insight.

Data engineers work closely with data scientists and data analysts. Their main task is building and maintaining the data architecture and creating algorithms to allow data scientists and analysts easier access to data. By doing their job, they are helping data scientists to do their job too. What you cannot avoid as a data engineer is knowing SQL, often on an advanced level compared to data analysts and scientists. To be a good data

engineer, you need to be an SQL master. That is why some questions you will be asked are the same as with data analysts and scientists. Besides the DQL commands, you are expected to be proficient in database modeling as well, hence you should know the DDL, DML and DCL commands in detail as well.

Machine learning engineers are hybrid experts who are bridging the gap between data scientists and software engineers. As they serve as a bridge between those two positions, they need to have a certain set of skills from both worlds. They use those skills to design, build, and maintain the machine learning systems. To achieve that, they usually use several sets of skills:

- statistics
- mathematics
- data mining
- data analysis
- predictive analytics

Machine learning engineers usually need to know Python and/or R. However, since machine learning engineers and data scientists have some skills in common (data mining, data analysis), it is quite often required from the machine learning engineers to know SQL as well. That way, they are able to make their own analyses and use the data according to their needs. They do not need some intermediary who'll pull out the data and analyze it for them. For example, here is a question from Uber on creating a Naive Forecasting Model.

Develop a naive forecast for a new metric: "distance per dollar". Distance Per Dollar is defined as the (distance_to_travel/monetary_cost) in our dataset. Calculate the metric and measure its accuracy.

Algorithm Outline:

"Sum the "distance to travel" and "monetary cost" values at a monthly level and calculate "distance per dollar". This is the actual value for the current month.
Next, populate the forecasted value for each month. To achieve this, take the previous month's value.
Now, we have actual and forecasted values. This is our naive forecast. Now evaluate our model by calculating the root mean squared error (RMSE). RMSE is the square root of the difference mean squared differences between the actual and the forecast values. Report the RMSE rounded to the 2nd decimal place."

SQL interviews for software engineers are also usually at an intersection of various skill sets, such as computer science, engineering, and mathematics. They use those different disciplines to design, write, test, and maintain the software. Like the machine learning engineers, they will also need to work with various departments and clients. That is why they too need a high level of business and technical skills, even though their primary task is not data analysis. Why is that? When they build the interface, they have to lean on the database(s) that run in the background. They need to use those databases and analyze data during the implementation of new software.

You can refer to this excellent article on the various roles and responsibilities in Data Science to learn more. The article breaks down the most common requirements for each role that you can expect on a daily basis.

Let us look at the most frequent SQL areas tested in Data Science Interviews.

Technical Concepts tested in SQL Interview Questions

Since SQL roles differ widely, the testing areas vary quite a bit as well. Depending on the type of role that you are applying for and the organization, you can expect one or more of these SQL Data Science Interview Question types

- Fundamental SQL concepts
- SQL basics interview questions
- SQL aggregation interview questions
- Open Ended SQL interview questions
- Data Transformation interview questions
- Database Modeling interview questions
- Software Engineering SQL Interview questions

Let us look at these SQL interview question types individually to understand the similarities and distinctions among them.

Fundamental SQL Concepts

The emphasis of these questions is on your understanding and knowledge of basic database and SQL terminology. One is not required to write any code. These are the broad concepts and some example questions that one should be familiar with.

- General information about SQL
  - What is SQL?
  - What are the different flavors of SQL?
  - What is a primary key?
- Relational Databases and how they work?
  - What are the top RDBMS engines?
  - How is an RDBMS different from a No-SQL database?
- SQL sublanguages and their main keywords
  - What do DDL, DCL, and DML stand for?
  - Give examples of commands for each.
- Data types and how SQL handles it (including blanks and NULLs)
  - What are the common data types in SQL?
  - Does an SQLite database support date time objects?
- Attribute constraints
  - What are attribute constraints, and explain them?
- types of JOINs
  - What is the difference between inner join and left outer join?
  - What is the difference between UNION and UNION ALL?
- Aggregation and Rollup functions
  - When should one use a CTE over a subquery?
  - What are window functions?
- Knowledge of various SQL functions
  - What is the difference between WHERE and HAVING? Examples of where one should use one over the other
  - What does the COALESCE function do?

While one may not find them in the interviews initially, these questions might be asked as follow up questions to the coding solutions submitted by you. For example, if you used an inner join in your solution, you might be asked why you did not use a left join or what would have happened if you did?

SQL Basics Interview Questions

The SQL basics interview questions will require you to put some of the above theoretical concepts into practice. That doesn't necessarily mean that these questions have to be

coding questions. They can be descriptive too. But they usually cover concepts you will need to know if you want to write a code. Those concepts are:

- using SUM(), COUNT(), AVG(), MIN(), MAX() and other aggregate functions
- GROUP BY
- CASE WHEN statement
- WHERE and HAVING
- JOINs
- UNION and UNION all

Questions Example

Here is an example from a Postmates SQL Data Science Interview

"How many customers placed an order and what is the average order amount?"

Solution: To answer this SQL interview question you'll have to use the table postmates_orders.

```
SELECT count(DISTINCT customer_id), avg(amount)
FROM postmates_orders
```

As one can see, this is really an easy one. It tests the aggregate functions COUNT() and AVG(), and it also requires the knowledge of how to use the DISTINCT clause.

Here's another one, this time from Credit Karma:

"Write a query that returns the user ID of all users that have created at least one 'Refinance' submission and at least one 'InSchool' submission."

Solution: To answer this SQL interview question you'll have to use the table loans.

```
SELECT user_id
FROM loans
WHERE TYPE in ('Refinance', 'InSchool')
GROUP BY user_id
HAVING count(DISTINCT TYPE) =2
```

The code selects the column user_id from the table loans where the value equals one of the two values: "Refinance" and "InSchool". Since we need to ensure that there are submissions in each type, we need to use the DISTINCT clause. One also needs to appreciate the difference between WHERE and HAVING. Since we are using the DISTINCT clause in the aggregate function, we have used HAVING instead of WHERE here.

SQL Aggregation Interview Questions

Aggregation functions are widely used for reporting metrics and evaluating summary results. Something Data Analysts and Data Scientists must do a lot. The purpose of aggregation is to transform data into information. This information is presented in the form of reports, dashboards, and charts. What you report are different metrics that need to be monitored. So, basically, your two main tasks will be aggregating and filtering data and performing various calculations on that data.

While strictly speaking, the data analysts and data scientists are not reporting but analyzing the data, there are a lot of instances where one needs to monitor and report metrics on a regular basis. If you are interviewing for an SQL Data Analyst or Data Scientist position, you are expected to be familiar with these concepts:

- subqueries
- joins and self-joins
- window functions
- CTEs
- GROUP BY extensions (ROLLUP, CUBE, GROUPING SETS)

Questions Example

This is a moderate SQL interview question from Zillow that tests your understanding of aggregations functions

"Write a query that identifies cities with higher than average home prices when compared to the national average. Output the city names."

Solution: To answer this question you'll have to use the table zillow_transactions.

```
SELECT city
FROM zillow_transactions a
GROUP BY city
HAVING avg(a.mkt_price) >
  (SELECT avg(mkt_price)
   FROM zillow_transactions)
ORDER BY city ASC
```

To answer this SQL interview question, we write a subquery in the HAVING clause. The inner query calculates the overall average market price of all the transactions. The main query then filters only those cities where the average market price is greater than the overall average market price. Since we need to report only the cities in the final output, only those metrics are reported.

Here is another one. This one is a little more complex from Lyft.

"Find the top 10 users that have traveled the greatest distance. Output their names and total distance traveled."

Solution: To answer this question you will have to use the tables lyft_rides_log, lyft_users.

```
SELECT name, traveled_distance
FROM
    (SELECT lu.name,
           SUM(lr.distance) AS traveled_distance,
           rank () over (order by SUM(lr.distance) desc) as rank
    FROM lyft_users AS lu
    INNER JOIN lyft_rides_log AS lr ON lu.id = lr.user_id
    GROUP BY lu.name
    ORDER BY traveled_distance DESC
    ) sq
WHERE rank <= 10
```

This SQL interview question tests subqueries, window functions, and joins. We start off by joining the two tables with an inner join as we need those users who have used the services and remove the rides with incomplete user information. We then proceed to rank the sum of the total distances travelled by these users. We finally select the top 10 users by rank and output their names and total distances travelled.

To improve your coding skills for SQL Data Science Interviews, you can refer to this video where we discuss some top data science interview questions, how to solve them and avoid common mistakes.

Open Ended SQL Interview Questions

You will encounter these types of questions in Data Analyst or Data Scientist positions that require some work experience. The greatest challenge in these questions is the lack of any specified metric that needs to be calculated as in the case of SQL Aggregation Interview Questions. You will still be required to write an SQL query that will return some metric(s) as a result. However, there is one big difference. In the Open-Ended SQL Interview questions, you will be asked to find the insight.

It is entirely up to you to understand your data and what calculation answers what you are being asked. For example, you will have to find out if some product launch campaign succeeded, or new calling procedure saves the costs, or if new vehicles improved the users' satisfaction. You will have to come up with metrics to define "success", "saving", or "improvement".

Compared to the SQL Aggregation questions, these questions have this extra dimension designed to test your thinking in solving the problem. Regarding the coding part of the Open-Ended SQL interview questions, they test all the concepts you will use in the basic level and the SQL Aggregation type questions.

Questions Example

Have a look at a question asked by Facebook:

"Facebook has developed a search algorithm that will parse through user comments and present the results of the search to a user. To evaluate the performance of the algorithm, we are given a table that consists of the search result the user clicked on ('notes' column), the user's search query, and the resulting search position that was returned for the specific comment.
The higher the position, the better, since these comments were exactly what the user was searching for. Write a query that evaluates the performance of the search algorithm against each user query."

Solution: To answer this question we must use the table fb_search_results.

```
SELECT t.result_id,
       t.query,
       CASE
           WHEN t.check = FALSE THEN 1
           WHEN t.check = TRUE
                AND t.position >= 11 THEN 2
           WHEN t.check = TRUE
                AND (t.position BETWEEN 6 AND 10) THEN 3
           WHEN t.check = TRUE
                AND (t.position BETWEEN 4 AND 5) THEN 4
           WHEN t.check = TRUE
                AND t.position <=3 THEN 5
       END AS rating
```

```
FROM
  (
SELECT query,
                result_id,
                position,
                 notes,
                (regexp_replace(notes, '[^\w]+',' ','g') ilike concat
('% ', query,' %')) AS check
   FROM fb_search_results
  ) t
```

In order to solve this problem, we first need to define the criteria for evaluation. We define the rating of the match between the user's search query and the search result returned by the algorithm (5 being the highest and 1 the lowest) as follows.

1. If there is no match, a rating of 1 is given.
2. If the match happens after the first 10 characters of the query, then the rating is 2.
3. If the match happens between the 6 to 10 characters, then the rating is 3.
4. If the match is within the first five characters, but not in the first three characters, then a rating of 4 is given
5. If the match is in the first three characters, then the highest possible rating of 5 is given.

Based on these ratings, we evaluate each search string. To match the search query and the result, we use the regular expression matching function in SQL. The main query uses this check column in the CASE WHEN statements. This is an open-ended SQL interview question because you were not given the metrics which would differentiate excellent performance from not-so-great one. Therefore, you had to decide on your scale that would show how the performance was. Your scale could be different from the one in the solution above. However, it is important that you explain your assumptions and why you decided to go with a certain evaluation scale.

Data Transformation Interview Questions

These are questions that one can expect to be asked very frequently for a Data Engineer or a Machine Learning Engineer position. Though it might not be out of place in a Data Scientist Interview for positions that require some experience. Data Transformation or more generally ETL (Extract, Transform and Load) is a process used to collect data from various sources (extract), changing it according to the business rules (transform), and then loading such extracted and transformed data into a database.

When the data is extracted, it is done so from various data sources that, more often than not, store data in completely different formats.

By transformation, the data takes the format appropriate for reporting and analysis. The data is transformed via data aggregation, filtering, sorting, joining, a calculation based on the rules set for business needs, etc.

Such data is loaded into another database or table that the analysts or any other users might use.

The ETL is heavily used in data warehouses, which serves as the central source of the integrated data, with data flowing into it from one or more separate sources. If you want to perform well at the SQL job interview, these are the concepts you need to know:

- Data Definition Language (DDL) keywords
- Data Manipulation Language (DML) keywords
- Data Control Language (DCL) keywords
- Transaction Control Language (TCL) keywords
- SQL constraints
- JOINs
- indexes
- transactions
- views

- user-defined functions
- stored procedures
- triggers
- variables
- query optimization

Question Examples

This is one of the easiest and yet frequently asked questions from the Southwest Airlines Data Science SQL Interview:

"What is the difference between DELETE and TRUNCATE?"

Answer:

DELETE is a DML statement. TRUNCATE is a DDL statement. The DELETE statement can be used to delete all rows or only some rows. To delete some rows, you'll have to use the WHERE clause. While doing this, every row removed will be logged as an activity by the database. On the other hand, TRUNCATE is used only for deleting the whole table, which will be logged as only one action. That's why TRUNCATE is faster than DELETE, which shows when deleting a table with a huge amount of data. Also, you can't use TRUNCATE if there's a foreign key in the table.

Another common question to appear is

"How do you change a column name by writing a query in SQL?"

Answer: Assuming you are using PostgreSQL. For a hypothetical table say product, one of the columns is named year, but I want to rename it to description. The query that will do that is:

```
ALTER TABLE product
RENAME year TO description;
```

Another example of Data Transformation SQL Interview question will be:

"How do you create a stored procedure?"

Answer: We will solve this for Microsoft SQL Server. For example, if you are using a table named employee. Your procedure should help you get the employees that work in a certain department. The code would be

```
CREATE PROCEDURE employee_deparment @deparment nvarchar(50)
AS
SELECT * FROM employees WHERE department = @department
GO;
```

Once the procedure is created, I can invoke it in the following manner:

```
EXEC employee_deparment @department = 'Accounting';
```

Database Modeling Interview Questions

These questions are designed to test how good you are at database design or database modeling. What is meant by that? You need to show the ability to design and build the database from scratch according to the business processes and business needs. This requires a high level of both technical and business knowledge. You will be working with both technical and non-technical colleagues. So, you need to understand both the business side of their requirement and how to, in the soundest way, technically cater to their business needs regarding the data. Generally, this is a process that goes through these steps (at least in the ideal world):

1. defining the database purpose
2. collecting and defining users' requirements
3. creating a conceptual model
4. creating the logical model
5. creating the physical model

Question Examples

One of the typical questions that occur in the SQL interviews is this one by Audible:

"Can you walk us through how you would build a recommendation system?"

Answer: Since there is a wide variety of approaches to answer this question, we will leave you to come up with your own way of building one.

The database design question can also include SQL coding, such as this one from Facebook:

"Write a SQL query to compute a frequency table of a certain attribute involving two joins. What if you want to GROUP or ORDER BY some attribute? What changes would you need to make? How would you account for NULLs?"

Answer: Due to the nature of the question, we will let you answer this one on your own.

Software Engineering SQL Interview Questions

These are questions that require SQL knowledge, but usually, what is being asked may not be widely applicable in practice. These questions come up during the interview because even though as a software engineer, you might not be writing SQL codes every day, you still need to interact with your peers who use SQL daily and follow what they are trying to achieve and implement their needs and SQL logic into the software development. These questions test your logical skills more than coding skills. One such question that could be asked during your interview could be

Imagine you're working with two tables. The one is the product table, which has the following data:

- id
- product_name
- manufacturer_id

The second table is manufacturer with the following data:

- id
- manufacturer

There are 8 records in the first table and 4 in the second one.

How many rows will the following SQL code return:

```
SELECT *
FROM product, manufacturer
```

Answer: The query will return 32 rows. Whenever the WHERE clause is omitted, the default result is CROSS JOIN or a Cartesian product. This means the query will return every combination of rows from the first table with every combination of rows from the second table.

The SQL Interview Questions Asked by the FAANG Companies

FAANG is an acronym for the five most famous tech companies: Facebook, Amazon, Apple, Netflix, and Google. Why would you specially prepare for the questions asked by those companies, except being in awe of the possibility of working for them? They might seem or even be attractive, but that is not the main reason why you would pay special attention if you want to work at those companies.

The main reason is their SQL interview questions are a bit different. As tech companies, their business heavily relies on data. And where there is data, there is SQL which the FAANG companies often use. Hence they want to be absolutely certain that their employees know SQL in depth. You will always get SQL interview questions with a little twist. The twist being their questions are more practical and concerning a case study with real problems and data a certain company is facing in their everyday business. These are arguably the next level of the Open Ended SQL Interview Questions that we saw earlier.

Have a look at this example from Google:

"Find the email activity rank for each user. Email activity rank is defined by the total number of emails sent. The user with the highest number of emails sent will have a rank of 1, and so on. Output the user, total emails, and their activity rank. Order records by the total emails in descending order. Sort users with the same rank score in alphabetical order.
In your rankings, return a unique value (i.e., a unique percentile) even if multiple users have the same number of emails."

Solution: To answer this question, you'll need to use the google_gmail_emails table.

```
SELECT  from_user,
        count(*) as total_emails,
        row_number() OVER ( order by count(*) desc)
    FROM google_gmail_emails
    GROUP BY from_user
    order by 3, 1
```

As you can see, this question tests your aggregate functions and window functions knowledge, along with the GROUP BY and ORDER BY clauses. But they also do that on real-life problems you'll probably have to work on if you get a job.

Here is another example of such question, this time from Netflix:

"Find the nominee who has won the most Oscars.
Output the nominee's name alongside the result.
Order the result based on the number of wins in descending order."

Solution: To answer this question, you will need to use the oscar_nominees table.

```
SELECT
    nominee,
    count(winner) AS n_times_won
FROM oscar_nominees
WHERE
    winner = true
GROUP BY
    nominee
```

```
    ORDER BY
        n_times_won DESC
```

Again, this question tests some usual concepts. But the problem set is something that you expect to work on a daily basis. If you work at Netflix on an SQL job, you will for sure analyze some data that contains some Oscar nominations and winners.

What to study for your SQL Data Science Interviews?

You would have probably noticed that the technical SQL interview questions overlap with other SQL questions. That is because one does not work without the other. There is no point in knowing the theory without being able to put it into practice, i.e., the SQL code. Conversely, you need to describe the technical concepts behind the code that you wrote. While the SQL concepts you should know depend on your position, years of experience, and the company you want to work at, we have looked at some concepts that are useful across roles. While this is not an exhaustive list, it is definitely something that you are expected to know if you are attending an SQL Data Science Interview.

SQL & Database Generalities

SQL definition

SQL stands for “Structured Query Language”. It is a programming language used for creating database structure, retrieving and manipulating data in it.

Types of the SQL commands

- Data Definition Language (DDL)
  - CREATE
  - ALTER
  - DROP
  - RENAME
  - TRUNCATE
  - COMMENT
- Data Query Language (DQL)
  - SELECT
- Data Manipulation Language (DML)
  - INSERT
  - UPDATE
  - DELETE
  - MERGE
  - CALL
  - EXPLAIN PLAN
  - LOCK TABLE
- Data Control Language (DCL)
  - GRANT
  - REVOKE

Relational database

A relational database is one based on the relational data model. This means the database is a collection of relations. Those relations are shown as tables, which consist of columns, rows, and values. The relational database aims to minimize or completely avoid data redundancy, leading to data integrity and speeding up its retrieval.

Relationships in the database

The relationship defines the type of connection between the tables in the database. There are three main types of relationships:

- one-to-one relationship (1:1)
- one-to-many relationship (1:N) or many-to-one relationship (N:1)
- many-to-many relationship (M:N)

Database normalization

Database normalization is a process of organizing data in the database to achieve its purpose: data integrity, its non-redundancy, and speed of retrieval.

Constraints

The constraints are the rules that define what type of data can and can't be entered as a value in the database. The most common attributes are:

- NOT NULL
- CHECK
- UNIQUE
- PRIMARY KEY
- FOREIGN KEY

Indexes

The indexes are structures in the databases created to tune the database performance. They are used to speed up data retrieval. The types of indexes are:

- clustered index
- non-clustered index
- unique index
- filtered index
- columnstore index
- hash index

View

A view is a virtual table containing data from one or more tables resulting from a SQL statement.

Stored procedure

A stored procedure is an SQL code consisting of one or several SQL statements that are saved and can be called and executed whenever required.

Trigger

A trigger is a special type of a stored procedure. It is automatically executed (triggered) whenever some special event occurs in the database.

Joining Tables & Queries

Inner join

An Inner join returns only those rows where the data from one table matches the data from the second table.

Left outer join

The left join is a table join that will retrieve all the rows from the left table and only the matching rows from the right table.

Right outer join

This join is the one that returns all the rows from the right table and only the matching rows from the left table.

Full outer join

The full outer join will join the data so that the result will include all the rows from one table and all the rows from the second table.

Cross join

This results in a Cartesian product. This means it will return all the combinations of rows from one table with all the combinations of rows from the other table.

Union

This is an SQL command that will combine the result of one query with the result of another query. Therefore, it will show only unique records.

Union all

This one also combines the results from two or more queries. The difference between UNION and UNION ALL is it will also include duplicates.

Aggregating and Grouping Data

Aggregate functions

The aggregate functions perform a calculation on a data set and return a single value as a result. Example of the aggregate functions are:

- COUNT()
- SUM()
- MIN()
- MAX()
- AVG()
- STDEV()
- VAR()

GROUP BY clause

The GROUP BY clause allows you to group data according to the defined (one or more) criteria.

Filtering & Ordering Data

DISTINCT clause

The DISTINCT clause is a clause that will return only distinct or unique values, i.e., there will be no duplicate values in the result.

WHERE clause

The WHERE clause is used to filter data according to the specified criteria.

HAVING clause

The HAVING clause also filters data according to the specified criteria. The difference compared to the WHERE clause is that the HAVING clause works with the aggregate functions. Therefore, if used, it always follows the GROUP BY clause and precedes the ORDER BY clause.

ORDER BY clause

The ORDER BY clause is used to order the query result according to a certain data column.

CASE statement

The CASE statement returns a defined value based on certain criteria. It is the SQL statement that allows you to apply the IF-THEN logic. Instead of IF, you use WHEN. And for THEN, you use THEN.

Subqueries, Common Table Expressions (CTEs) & Window Functions

Subquery

A subquery is a query found within the query. It can occur in a SELECT clause, FROM clause, or WHERE clause.

CTE

A CTE or a Common Table Expression is a temporary result set returned by a query and used by another query. In that way, it's similar to subquery. But the main difference is CTE can be named and can reference itself.

Window functions

The window functions are SQL functions performing calculations over the defined set of rows (a window). Compared to the aggregate functions, which return a single value as a result, the window functions allow you to add the aggregated value to each row in a separate column. This means the rows are not grouped and all the rows are kept as a query result. The window functions are:

- row_number()
- rank()
- dense_rank()
- percent_rank()
- cume_dist()
- lead()
- lag()
- ntile()
- first_value()
- last_value()
- nth_value()
- avg()
- count()
- min()
- max()
- sum()

How to Organize your SQL Data Science Interview Solution?

Being good at SQL is the prerequisite to do well at the job interview. However, it is not the only skill. Questions can be tricky, designed to put you off or doubt your knowledge by being seemingly too complicated or too simple. That's why it's important to have a clear strategy on what to do in certain situations.

1. Make Sure you Understand What is Required

If you don't understand what the question is and what is expected from you, you will likely get the wrong answer. To avoid that, make sure you understand what is asked of you. Repeat the requirements out loud and ask the interviewer to confirm you understood the question correctly. Don't be afraid to do that. The interviewers are people too. They can also unintentionally be unclear, make a mistake, or forget to give you enough details for you to answer correctly.

2. Outlay Your Approach

Before you start answering, especially if you are writing SQL code, outlay your approach. That way, you will be able to find the solution faster and or find the holes in the code you intended to write. You should do that to allow the interviewer to lead you through in case you missed the point of the question. It is always better to be corrected before presenting the final solution.

3. Try to Visualize the Output

This is something that can help you in writing the problem-solving code. Sometimes, when you clarify how the output should look and compare it with initial data, the approach and the solution reveal themselves.

4. Write the SQL Code

At some point, you will need to start writing the code. As we discussed, you should not jump headfirst into it. But you cannot keep procrastinating on the best approach to write it. After you have gone through all those previous steps, and you are still not sure if you have the right solution, simply start writing the code.

One of the reasons is, sometimes there is no solution at all. Meaning the question is too complex to be solved in the time you're being given. In such cases, the interviewer is not interested in your solution. Instead, he or she is interested in your way of thinking and how you approach the problem. There are usually multiple ways to use SQL for problem-solving, and this is what some interviewers are interested in: the process, not the goal.

5. Code in Logical Parts

When you're writing the code, pay attention to its structure. Divide the code into logical parts. That way, you will make your code easier to read, which is also one of the requirements to get the job. There is no point in writing a correct code that is a mess, and nobody can read it and understand it after you write it. Not even you!! If your code is divided into logical parts, it will be easier for you to explain to the interviewer what you did.

6. Optimize Code

It is also important to have the code optimization in mind. If your code is complex, of course, you are not going to be able to optimize it as you write. But you should pay attention to some general optimization guidelines, so your code is reasonably optimized. You can also discuss with the interviewer what other steps you will have to take to optimize your code in the aftermath. This is also the job requirement, similarly to the previous point. There is no point writing the code that will get you the required result, but takes forever to execute.

7. Explain Your Answer and Assumptions

Even if you did not get the required answer, it does not mean you failed the interview. That is why you should always know why you did something and explain why you did it. Maybe you did not get the answer to the question they asked, but you did get the answer to some questions. So make sure that you state your assumption and explain why you did what you did. Again, they may be looking exactly for that: the right reasoning in line with assumptions, even though the assumptions were wrong. That also shows you know what you are doing, even if it is not what they asked.

Also, one of the reasons for explaining the assumptions is there may be a loophole in the question. So imagine pointing at it right there at the interview by simply explaining why you did something while you thought you were all wrong.

Here is the video where Nate from StrataScratch shares some tips on how to organize your SQL interview solution:

Bonus SQL Interview Tips

Here are some additional tips that might help you to be a success at the upcoming SQL interview.

Get to Know the Your Potential Employer

This is important in general, not only for the SQL part of the interview. It is important to be informed about your future employer, their products, and their industry. It is especially important when the SQL questions are regarded. Why is that?

As we discussed earlier, the FAANG companies will usually ask you very practical SQL coding questions that will have you use the same data and solve the same problems as you would have to when you get employed. The FAANG companies are not the only ones who do that. So when you prepare for the interview, try to think which data is important to this company, how their database could look like, etc. When you practice the SQL questions, try to find the real questions from the companies you are interested in or at least from their competitors. If the companies are in the same industry, it's quite likely the data they use will be more or less the same.

Be Prepared for a Whiteboard

It is quite usual to be asked to write SQL code on a whiteboard. It can be shocking to some people, which is understandable. You are probably used to writing code in a real RDBMS, on real data, which allows you to regularly check if the query works. Not even the greatest masters of SQL can write a code without running it to see if it works at all or if it returns the desired result. However, in the SQL interview, the criteria are a little higher.

While it can be scary, it is also understandable. Writing the code on a whiteboard shows that you know how to write your code. Reading your (or someone else's code) is also

important. This is the skill that is also tested on a whiteboard. If you can read a code and say if it will give you the desired result without relying on the database to tell you that, then working with a real database and SQL environment will be easier for you.

Write a Clean Code

We are not talking about your handwriting. There is not much you can do if your handwriting is messy. But that does not mean your code has to be unreadable. When you write a code, try to format it so that it is easier for you and the interviewers to read it and check your solution.

"Code is read more often than it is written. Code should always be written in a way that promotes readability.
- Guido Van Rossum, the creator of Python"

Regularly use spacing and line breaks to make your code easier to read. If you need to (re)name tables and columns, be consistent with the naming convention you choose. Add comments whenever needed. Try to use aliases whenever possible, but try to make them sound logical and not some random decision when you do.

Here are also some useful tips from Nate on how to organize lengthy SQL codes.

Write in the Company's SQL Dialect Only if you are comfortable with it

If you have experience with multiple SQL databases (Oracle, PostgreSQL, Microsoft SQL Server, MySQL), try to adapt and write in a dialect of the database that is used at your future employer. That would be nice and could show your versatility, especially if you know what dialect they prefer. However, if you are familiar with only one dialect, do not think that it is the end of the interview. For example, if you were using only PostgreSQL and the company is using Microsoft SQL Server, there may be different keywords for the same command in those two databases. It is also possible that PostgreSQL has some functions that aren't allowed in Microsoft SQL Server and vice versa.

Ask the interviewer if it is possible to write a code in, say, PostgreSQL instead of Microsoft SQL Server since you are more familiar with it. It is always better if you know several dialects. But it is also better if you write in a familiar dialect, even though "the wrong one", than mess up the code just because you were too afraid to ask if you can write in a dialect you are comfortable with. The differences between the dialects are not that huge. So if you know SQL, you'll easily and quickly adapt to a new database.

Communication. Confidence. Collaboration.

While the interview is evaluating you, there are other things that employers look for in an employee besides just coding ability. You will be working as a part of a team and hence your ability to confidently put across your ideas, be open to positive feedback about your codes and ability to work as a team is equally important. Employers try to gage these even during something as technical as coding.

It is vital that you ask for help in case you are stuck. Asking for help shows confidence and is not a sign of weakness. Keep the interviewer in the loop regarding what your thought process is so that she might be able to help you in case you are stuck or omitted some information unintentionally. Listen to any explanations provided to ensure that you have taken care of all the edge cases that might arise.

These are skills that every Data Scientist must possess and unfortunately not a lot of candidates focus on them. An interviewer is more likely to hire a Data Scientist with sound basic understanding of SQL and willing to adapt to changes and pick up additional skills on the way over a prodigious but rigid one. The employer is looking at the potential for a long term relationship. If you give the right signals, you might just land your dream job.

Check out these 5 tips on how to prepare for a Data Science interview.

We have also gathered some advanced level SQL questions asked by real companies in 2021 that you can find in our Advanced SQL Interview Questions You Must Know How to Answer article.

Conclusion

In this article, we looked at the various aspects of an SQL Data Science Interview. We started off by understanding why SQL is so popular in the Data Science world and the different roles that are available in the industry. We then provided a detailed overview of the type of questions that you can expect for each position and what to learn for your becoming proficient with SQL for Data Science Interviews.

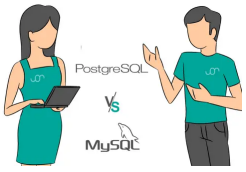
Even if you have just started with SQL, all that it takes to become interview ready is persistence, patience and lots of practice. If you do not have much real-life experience with SQL, it is very important that you practice writing SQL codes. Do it a lot, and regularly. Continuity is very important. Try to answer as many as possible SQL interview questions, be it hypothetical or, even better, the real ones from the company you want to work at. Only by writing a lot of code, you'll gain experience, grasp some typical problems that need to be solved by SQL, and the syntax will become like second nature.

Even if you are vastly experienced in SQL and use it in a business environment, it is always good to prepare for an interview and brush up your skills. Nobody knows everything about SQL. Generally, people know what they need, what they regularly use every day. So it is possible that after several years at your job, you became a master of a certain aspect of SQL. Do not let it make you think you know it all. It could be that your new job will be asking you to know some different SQL functions and possibilities, the ones you are not exactly versed at.

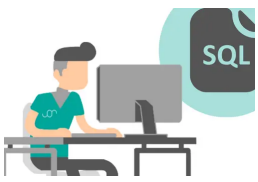
In the StrataScratch community, you will be able to compete and collaborate with other aspiring data scientists working towards achieving their dream job at top tech companies and hottest start-ups across the world. We have over 400+ Real Life SQL Data Science Interview questions categorized by difficulty levels, frequency, recency, companies, et al. Or you can choose questions according to the topics, such as Window Functions, Manipulating Text, Manipulating Datetime. Join a community of over 20,000 learners and be interview ready.

Related Posts:

Data Analyst Interview Questions and Answers



Postgres vs MySQL: Which is Better for Analytics?



SQL Interview Questions for the Data Analyst Position