Interviews can seem scary.

Especially programming interviews, where you may have to write some code (such as SQL) during the interview.

But, with a little revision of some SQL interview questions and some tips, they are much easier.

You can use this guide as a study guide for any upcoming interviews. It contains all of the interview questions from my SQL Interviews post.

Let's get into it!

Basic SQL Interview Questions

1. What is the difference between SQL, Oracle, MySQL, and SQL Server?

SQL is the name of the language used to query databases and follows a standard. Oracle, MySQL and SQL Server are different implementations or versions of a database management system, which implement the SQL standard and build on it in different ways.

Oracle database is targeted at large companies, SQL Server is owned by Microsoft, and MySQL is owned by Oracle but targeted toward smaller companies and systems.

2. What is the difference between SQL and PL/SQL?

SQL is the language used to query databases. You run a query, such as SELECT or INSERT, and get a result.

PL/SQL stands for Procedural Language/Structured Query Language. It's Oracle's procedural language and is built on top of SQL. It allows for more programming logic to be used along with SQL.

3. What is the difference between SQL and T-SQL?

SQL is the language used to query databases. You run a query, such as SELECT or INSERT, and get a result.

T-SQL stands for Transact-SQL and is a language and set of extensions for SQL Server that allows for further programming logic to be used with SQL.

4. What are the different DDL commands in SQL? Give a description of their purpose.

- CREATE: creates objects in the database

- ALTER: makes changes to objects in the database
- DROP: removes objects from the database
- TRUNCATE: deletes all data from a table
- COMMENT: adds comments to the data dictionary
- RENAME: renames an object in the database

5. What are the different DML commands in SQL? Give a description of their purpose.

- SELECT: retrieve or view data from the database
- INSERT: add new records into a table
- UPDATE: change existing records in a table
- DELETE: removes data from a table
- MERGE: performs an UPSERT operation, also known as insert or update.
- CALL: runs a PL/SQL procedure or Java program
- EXPLAIN PLAN: explains the way the data is loaded
- LOCK TABLE: helps control concurrency

6. What is the purpose of the BETWEEN keyword?

The BETWEEN keyword allows you to check that a value falls in between two other values in the WHERE clause.

It's the same as checking if a value is greater than or equal to one value, and less than or equal to another value.

7. What is the purpose of the IN keyword?

The IN keyword allows you to check if a value matches one of a range of values. It's often used with subqueries that return more than one row.

8. What is a view? When would you use one?

A view is a database object that allows you to run a saved query to view a set of data. You create a view by specifying a SELECT query to be used as the view, and then the view can be queried just like a table.

There are several reasons to use a view, such as to improve to security, create a layer of abstraction between the underlying tables and applications, and to simplify queries.

Related: A Guide to Views and Materialised Views

9. What's the difference between a view and a materialized view?

A view is simply an SQL query that is stored on the database, without the results. Every time a view is queried, this definition of the view's query is run. If the underlying tables have been updated, the view will load these results.

A materialized view is a query where the results have been stored in a permanent state, like a table. If the underlying tables are updated, then by default, the materialized views are not updated.

10. What is a primary key?

A primary key is a column or set of columns that uniquely identifies a row in a table. It's created on a table and ensures that the values in that column or columns must be unique and not NULL.

This is often done using some kind of numeric ID field but doesn't have to be.

Related: A Guide to Database Keys

11. What is a foreign key?

A foreign key is a field in a table that refers to a primary key in another table. It is used to link the record in the first table to the record in the second table.

12. What is a composite key?

A composite key is a primary key that is made up of two or more fields. Often, primary keys are single fields, but in some cases, a row is identified by multiple fields. This is what a composite key is.

13. What is a surrogate key?

A surrogate key is a field in a table that has been created solely for the purpose of being the primary key. It has no other purpose than an internal storage and reference number.

For example, a customer may have an account number that is unique to them, but a customer_id field might be created on the table and used as the primary key, in case business rules change and mean that the account number is no longer unique.

14. What is a unique constraint? How is it different from a primary key?

A unique constraint is a constraint on a table that says that a column or set of columns needs to have unique values.

It's different to a primary key in that a table can only have one primary key, but a table can have zero, one, or many unique constraints.

Unique constraints can also allow NULL values, but primary keys cannot.

15. What is a synonym?

A synonym is a database object that allows you to create a kind of "link" or "alias" to another database object. This is often done to hide the name of the actual object for security reasons, or to improve maintenance of the code in the future.

Related: A Guide to Synonyms

16. If a table contains duplicate rows, will a query display duplicate values by default? How can you eliminate duplicate rows from a query result?

Yes, they will be displayed by default. To eliminate duplicate records, you use the DISTINCT keyword after the word SELECT.

17. What is wrong with this query (in Oracle)?

```
SELECT SYSDATE;
```

Answer:

Even if you don't need a table to get your data, you need to add a table to your SELECT query for it to run.

In this case, you can use the DUAL table that Oracle has created.

```
SELECT SYSDATE
FROM dual;
```

Joins

18. What are the different JOIN types and what do they do?

The different join types in Oracle SQL are:

- Inner join: Returns records that exist in both tables.
- Left join/left outer join: Returns records that exist in the first table and shows NULL for those values that don't exist in the second table.

- Right join/right outer join: Returns records that exist in the second table and shows NULL for those values that don't exist in the first table.
- Full join/full outer join: Returns records that exist in both the first and second table, and shows NULL for those values that don't exist in the corresponding table.
- Cross join: Returns all combinations of all records in both tables.
- Natural join: an inner join with two tables on columns that have the same names.
- Self join: A join from one table to another record in the same table.

  Related: S QL Joins: The Complete Guide

19. What is a "cross join"?

A cross join is a type of join where the results displayed contain the records from both tables in all possible combinations. There is no field used to perform the join.

For example, if table A has 10 records and table B has 8 records, then the cross join will result in 80 (or 10 x 8) records.

The result can also be called a "cartesian product".

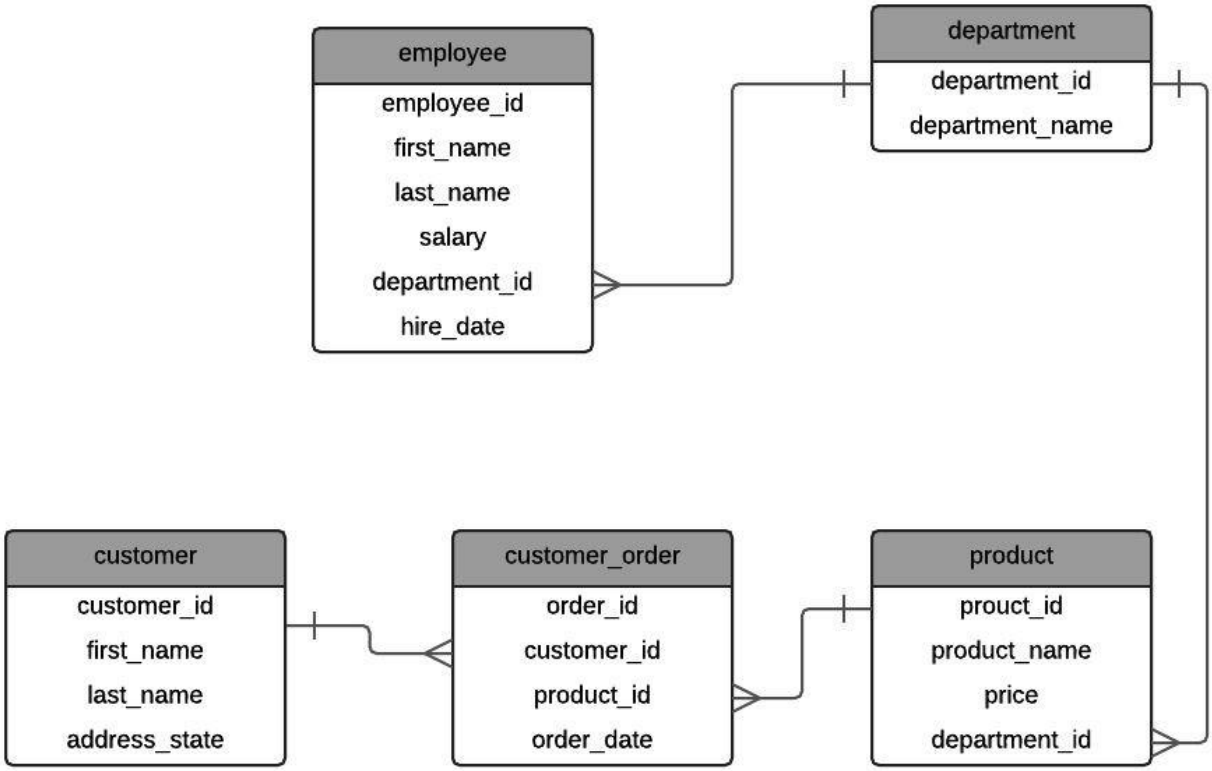
20. What is a self join and why would you use one?

A self join is a type of join where a table is joined to itself.

You would use a self join when a table has a field that refers to another record in the same table. It's often used in hierarchical structures, such as employee tables having a manager_id column where the manager_id refers to another employee record.

21. Given this ERD, write a query that shows the following information.

Entity Relationship Diagram



The customer ID, customer first and last name, the order ID of any orders the customer has placed (if any) and the date of the order. The data should be ordered by last name then first name, both in ascending order.

Answer:

```
SELECT
c.customer_id,
c.first_name,
c.last_name,
co.order_id,
co.order_date
FROM customer c
LEFT JOIN customer_order co ON c.customer_id = co.customer_id
ORDER BY c.last_name, c.first_name;
```

This question checks your ability to translate a normal English statement into a SELECT query.

You should have picked up on the need for a LEFT JOIN, the need for table aliases for ambiguous columns, and the ORDER BY.

Table aliases are good to use in any case, so experienced developers will use them for every query.

You might have several different variations of this interview question for SQL interviews. Knowing your query structure and focusing on the requirement for the query are important here.

Note: If you're looking for a tool to create these kinds of diagrams, check out my g uide on 76 Data Modeling Tools Compared.

Aggregation and Grouping

22. What is an aggregate function?

An aggregate function is an SQL function that reads data from multiple rows and displays a single value. Some examples of aggregate functions are COUNT, SUM, MIN, MAX, and AVG. They are often used with a GROUP BY clause but can be used by themselves.

23. Can you nest aggregate functions?

Yes, you can have nested aggregate functions up to two levels deep. For example, you can use MAX(COUNT(*)).

24. Does COUNT return the number of columns in a table?

No, it returns the number of records in a table.

25. What's the difference between COUNT(column) and COUNT(DISTINCT column)?

COUNT(column) will return the number of non-NULL values in that column. COUNT(DISTINCT column) will return the number of unique non-NULL values in that column.

26. What is the difference between the WHERE and HAVING clauses?

The WHERE clause is run to remove data before grouping. The HAVING clause is run on data after it has been grouped.

This also means the WHERE clause cannot operate on aggregate functions calculated as part of the group.

More information: T he Difference Between the WHERE and HAVING Clause.

27. What's wrong with this query?

```
SELECT department_id, count(*)

FROM department;
```

Answer:

There is no GROUP BY clause and it will display an error. Because we have used the COUNT function, which is an aggregate function, along with a database field, we need to add a GROUP BY clause. It should GROUP BY the department_id column.

28. What's wrong with this query?

```
SELECT department_id, count(*)

FROM department

WHERE count(*) > 5

GROUP BY department_id;
```

Answer:

The WHERE clause cannot include any checks on the aggregate column – even if a GROUP BY has been performed.

This is because the WHERE happens before the grouping, so there is no way for the WHERE clause to know what the value of the COUNT function is.

To resolve this, use the HAVING clause to check for COUNT(*) > 5.

Ordering

29. What is the default sort order using ORDER BY? How can it be changed?

The default sort order is ascending. This can be changed by specifying the word DESC after any column name in the ORDER BY clause. The word ASC can be used instead to specify ascending order.

30. Can you sort a column using a column alias?

Yes, you can sort by column aliases in an ORDER BY clause.

Analytic or window functions

31. What is a window function or analytic function?

A window function or analytic function is a function that performs a calculation across a set of related rows. It's similar to an aggregate function, but a window function does not group any rows together. The window function accesses multiple rows "behind the scenes".

32. What is the difference between RANK and DENSE_RANK?

The difference between RANK and DENSE_RANK is where there is a tie or two records with the same value.

RANK will assign non-consecutive values, which means there will be gaps in numbers.

DENSE_RANK will assign consecutive values, which means there will be no gaps.

33. What's the difference between ROWNUM and ROW_NUMBER?

ROWNUM is a pseudocolumn and has no parameters, where as ROW_NUMBER is an analytical function that takes parameters.

ROWNUM is calculated on all results but before ORDER BY. ROW_NUMBER is calculated as part of the column calculation

ROWNUM is unique. ROW_NUMBER can contain duplicates.

More information: W hat's The Difference Between Oracle ROWNUM vs Oracle ROW_NUMBER?

Set Operators

34. What does UNION do? What's the difference between UNION and UNION ALL?

Union allows you to combine two sets of results into one result.

It's different to UNION ALL because UNION removes duplicate values and UNION ALL does not.

35. What's the difference between UNION and JOIN?

A join allows us to lookup data from one table in another table based on common fields (for example employees and departments). It requires us to have a field that is common in both tables.

A union allows us to combine the results of two queries into a single result. No join between the results is needed. Only the number and type of columns need to be the same.

36. What's the difference between UNION, MINUS, and INTERSECT?

They are all set operators.

But, UNION will combine the results from query1 with query2 and remove duplicate records.

MINUS will display the results of query1 and remove those that match any records from query2. INTERSECT will display the records that appear in both query1 and query2.

Subqueries

37. What is a subquery?

A subquery is a query within another query. This subquery can be in many places, such as in the FROM clause, the SELECT clause, or a WHERE clause.

It's often used if you need to use the result of one query as an input into another query.

Related: S QL Subqueries: A Complete Guide

38. What is a correlated subquery?

A correlated subquery is a subquery that refers to a field in the outer query.

Subqueries can be standalone queries (non-correlated), or they can use fields in the outer query. These fields are often used in join conditions or in WHERE clauses.

39. Given these two queries and result sets, what will the result of this query be?

Explain your answer.

```
SELECT *

FROM employee;
```

EMPLOYEE_ ID	FIRST_ NAME	LAST_ NAME	SALARY	DEPARTMENT_ ID	MANAGER_ ID	HIRE_ DATE
1	Michelle	Foster	48000	8	162	27/Aug/11
2	Cheryl	Turner	79000	3	99	02/Jan/12
3	Carolyn	Hudson	47000	7	199	04/Dec/16
4	Patrick	Berry	51000	3	159	12/Oct/11
5	Doris	Powell	117000	1		15/Nov/11
6	Jessica	Elliott	21000	7	70	02/Jul/10
7	Sean	Burns	76500	6	37	03/Oct/10
8	Ann	Bowman	34000	7	187	20/May/10
9	Kathleen	Jones	92000	7	131	15/Mar/15
84	Jennifer	Long	36000	(null)	4	25/Nov/14

```
SELECT *

FROM department;
```

DEPARTMENT_ID	DEPARTMENT_NAME
1	Executive
2	Sales
3	Customer Support
4	Hardware Development
5	Software Development
6	Marketing
7	Finance
8	Legal
9	Maintenance

What will the result of this query be?

```
SELECT *
FROM department
WHERE department_id NOT IN (
  SELECT department_id
  FROM employee
);
```

Answer:

This will return an empty result set. This is because of how the NOT IN command treats NULL values.

If the set of data inside the NOT IN subquery contains any values that have a NULL value, then the outer query returns no rows.

To avoid this issue, add a check for NULL to the inner query:

```
SELECT *
FROM department
WHERE department_id NOT IN (
  SELECT department_id
  FROM employee
  WHERE department_id IS NOT NULL
);
```

40. Write a query to display the 5th highest employee salary in the employee table

```
SELECT *
FROM (
  SELECT
employee_id,
first_name,
last_name,
salary,
```

```
  DENSE_RANK() OVER (ORDER BY salary DESC NULLS LAST) rank_val
  FROM employee
)
WHERE rank_val = 5;
```

This could also be done using the ROW_NUMBER function. It's one of those interview questions in SQL that can have multiple answers, but as long as you provide an answer to it, you should be OK.

Database Design

41. What is cardinality?

Cardinality refers to the uniqueness of values in a column. High cardinality means that there is a large percentage of unique values. Low cardinality means there is a low percentage of unique values.

42. How can you create an empty table from an existing table?

You can use the CREATE TABLE AS SELECT command.

The SELECT statement will contain all of the columns that you want to have in your new table. To ensure it is empty, add a WHERE clause that evaluates to FALSE, such as WHERE 1=0.

43. What is normalisation?

Normalisation is the process of organising your data into tables that adhere to certain rules. It aims to make the process of selecting, inserting, updating, and deleting data more efficient and reduce data issues that may appear otherwise.

There are three popular normal forms, named first/second/third normal form. Third normal form is commonly used as a goal, but there are normal forms after third normal form that are occasionally used.

Related: D atabase Normalization: A Step-By-Step-Guide With Examples

44. What is denormalisation?

Denormalisation is the process of converting a normalised database into a series of tables that are not normalised. These denormalised tables often contain records that refer to the same value, so updating them is not as efficient. However, the aim of this process is usually to prepare the data for a data warehouse, so the goal is the efficient reading of data.

It often results in a smaller number of tables, each of which has more columns than normalised tables.

45. What do OLTP and OLAP mean and how are they different?

OLTP stands for OnLine Transaction Processing and refers to databases that are designed for regular transactions of inserting, updating, and deleting data. This often includes a normalised database and is linked to an application used during business hours for people to do their job.

OLAP stands for OnLine Analytical Processing and refers to databases that are designed for analysis and reporting. They are focused on SELECT queries and often contain denormalised database designs. They are often used by reporting systems to analyse data from other OLTP systems.

Functions

46. What are the case manipulation functions in Oracle SQL?

To change the case of a string in Oracle SQL you can use UPPER, LOWER, or INITCAP. Read more here.

47. Which function or functions returns the remainder of a division operation?

The MOD function and REMAINDER function both return the remainder of a division operator.

This is one of the SQL interview questions which is Oracle specific, as the REMAINDER function does not exist in other database management systems.

48. What does the NVL function do, and how is it different from NVL2?

The NVL function checks if a value is NULL, and returns the value if it is not NULL. If the value is NULL, it returns a different value which you can specify.

NVL2 is slightly different in that you specify both the value to return if the checked value is NULL and if it is not NULL.

NVL takes two parameters and NVL2 takes three.

49. How can you perform conditional logic in an SQL statement?

This can be done using either the DECODE function or the CASE statement.

The CASE statement is more flexible and arguably easier to read than the DECODE function.

50. How can you search for a value in a column when you don't have the exact match to search for?

If you don't know the exact match, you can use wildcards along with LIKE. The wildcards are the % symbol for any number of characters, and the _ symbol for a single character.

Inserting and Updating Data

51. What does the MERGE statement do?

The MERGE statement allows you to check a set of data for a condition, and UPDATE a record if it exists or INSERT a record if it doesn't exist.

Related: T he MERGE Statement in Oracle

52. Can you insert a NULL value into a column with the INSERT statement?

Yes, you can. You can do this by:

- Leaving the column out of the list of columns in the INSERT statement; or
- Specifying the value of NULL for the column in the VALUES clause

53. Can you INSERT data from one table into another table? If so, how?

Yes, you can do this using an INSERT INTO SELECT query. You start by writing an INSERT INTO statement, along with the columns you want, and then instead of the VALUES clause, you write a SELECT query.

This SELECT query can select data from the same table, or another table, or a combination of tables using JOINs, just like a regular SELECT query.

54. What happens if you don't have a WHERE clause in an UPDATE statement?

All records in the table will be updated. You need to be sure that's what you want to do.

55. What happens if you don't have a WHERE clause in a DELETE statement?

All records will be deleted from the table. It will still run, there will be no error. You need to be sure that's what you want to do.

56. What's the difference between DROP and DELETE?

DROP is used to remove database objects from the database, such as tables or views. DELETE is used to remove data from a table.

Also, DROP is a DDL statement and DELETE is a DML statement, which means DELETE can be rolled back but DROP cannot.

57. What's the difference between TRUNCATE and DELETE?

There are several differences.

TRUNCATE deletes all records from a table and you cannot specify a WHERE clause, but DELETE allows you to specify a WHERE clause if you want.

TRUNCATE does not allow for rollbacks, and DELETE does.

TRUNCATE is often faster because it does not generate an undo log, but DELETE does.

Other SQL Interview Questions

58. What is a clustered index?

A clustered index is a type of index that reorders how the records are stored on the disk. This allows for fast retrieval of data that uses this index.

A table can only have one clustered index. An alternative is a non-clustered index, which does not order the records on a disk but does offer other benefits of indexes.

Related: O racle SQL Indexes - The Definitive Guide

59. What is DCL? Provide an explanation of some of the commands.

DCL stands for Data Control Language. The commands that come under DCL are:

- GRANT: give access privileges to a user
- REVOKE: withdraw access privileges from a user

60. What is TCL? Provide an explanation of some of the commands.

TCL stands for Transaction Control Language and it contains statements to manage changes made by DML statements. It includes:

- COMMIT: saves the data to the database
- ROLLBACK: undo the modifications made since the last COMMIT
- SAVEPOINT: create a point in a transaction that you can ROLLBACK to
- SET TRANSACTION: change the transaction options, such as isolation level
- SET ROLE: sets the current active role

61. What is an execution plan? How can you view the execution plan?

An execution plan is a graphic or text visualisation of how the database's optimiser will run a query. They are useful for helping a developer understand and analyse the performance of their query.

To find the execution plan of a query, add the words "EXPLAIN PLAN FOR" before your query. The query won't run, but the execution plan for the query will be displayed.

62. Is NULL the same as a zero or blank space? If not, what is the difference?

No, they are different. NULL represents an unknown value. Zero represents the number zero, and a blank space represents a character string with no data.

NULL is compared differently to a zero and a blank space and must use comparisons like IS NULL or IS NOT NULL.

63. What's the difference between ANY and ALL?

The ANY keyword checks that a value meets at least one of the conditions in the following set of values. The ALL keyword checks that a value meets all of the conditions in the following set of values.

64. What's the difference between VARCHAR2 and CHAR?

VARCHAR2 does not pad spaces at the end of a character string, but CHAR does. CHAR values are always the maximum length, but VARCHAR2 values are variable length.

65. List the ACID properties and explain what they are.

ACID stands for Atomicity, Consistency, Isolation, and Durability. They are a set of properties that ensure that database transactions are processed reliably.

- Atomicity means that each transaction be atomic, which means "all or nothing". Either the entire transaction gets saved, or none of it gets saved.
- Consistency means that any transaction will bring the database from one consistent state to another. Data must be valid according to all business rules.

- Isolation means that transactions that are executed at the same time will give the same results as transactions executed one after the other. The effects of one transaction may not be visible to another transaction.
- Durability means that once a transaction has been committed, it remains committed. This is even if there is a disaster, such as power loss or other errors.

This SQL interview question should be relevant to all database management systems. It's not Oracle specific.

66. How can you create an auto-increment column in Oracle, in version 11 or earlier? What about in Oracle 12c?

In Oracle 11g, you can create an auto-increment column by using a combination of a sequence and a BEFORE INSERT trigger.

The sequence is used to generate new values, and the BEFORE INSERT trigger will read these new values and put them into the required column whenever you INSERT a new record.

In Oracle 12c, you can define a column as an Identity column by putting the words GENERATED AS IDENTITY after the column in the CREATE TABLE statement. This means new values are generated automatically.

67. What's the difference between % and _ for pattern matching (e.g. in the LIKE operator)?

The difference is the % sign will match one or more characters, but the _ sign will match only one character.

68. What is a CTE?

CTE stands for Common Table Expression. It's a SELECT query that returns a temporary result set that you can use within another SQL query.

They are often used to break up complex queries to make them simpler. They use the WITH clause in SQL. An example of a CTE would be:

```
WITH cte_car_model (make_name,
model_name) AS (   SELECT ma.make_name,
mo.model_name
  FROM car_make ma
  INNER JOIN car_model mo ON mo.model_id = ma.model_id
```

```
)
SELECT make_name,
model_name
FROM cte_car_model;
```

This is a simple example, but more complicated queries will benefit from CTEs, both in performance and readability.

69. What is a temp table, and when would you use one?

A temp table (or temporary table) is a database table that exists temporarily on the system. It allows you to store the results of a query for use later in a session. They are useful if you have a large number of results and you want to use them again.

A temporary table, by default, is only accessible by you. Global temporary tables can be accessed by others.

Temporary tables are automatically deleted when the connection that created them is closed.

Top 100 advanced SQL questions and answers for query writing!

1. How to retrieve the second-highest salary of an employee?

```
SELECT MAX(salary)
FROM employees
WHERE salary < (SELECT MAX(salary) FROM employees);
```

2. How to get the nth highest salary in ?

```
SELECT salary
FROM (SELECT salary, DENSE_RANK() OVER (ORDER BY salary DESC) AS rank
    FROM employees) AS ranked_salaries
WHERE rank = N;
```

3. How do you fetch all employees whose salary is greater than the average salary?

```
SELECT *
FROM employees
WHERE salary > (SELECT AVG(salary) FROM employees);
```

4. Write a query to display the current date and time in .

```
SELECT CURRENT_TIMESTAMP;
```

5. How to find duplicate records in a table?

```
SELECT column_name, COUNT(*)
FROM table_name
GROUP BY column_name
HAVING COUNT(*) > 1;
```

6. How can you delete duplicate rows in ?

```
WITH CTE AS (
  SELECT column_name,
       ROW_NUMBER() OVER (PARTITION BY column_name ORDER BY column_name) AS row_num
  FROM table_name
)
DELETE FROM CTE WHERE row_num > 1;
```

7. How to get the common records from two tables?

```
SELECT *
FROM table1
INTERSECT
SELECT *
FROM table2;
```

8. How to retrieve the last 10 records from a table?

```
SELECT *
FROM employees
ORDER BY employee_id DESC
LIMIT 10;
```

9. How do you fetch the top 5 employees with the highest salaries?

```
SELECT *
FROM employees
ORDER BY salary DESC
LIMIT 5;
```

10. How to calculate the total salary of all employees?

```
SELECT SUM(salary)
FROM employees;
```

11. How to write a query to find all employees who joined in the year 2020?

```
SELECT *
FROM employees
WHERE YEAR(join_date) = 2020;
```

12. Write a query to find employees whose name starts with 'A'.

```
SELECT *
FROM employees
WHERE name LIKE 'A%';
```

13. How can you find the employees who do not have a manager?

```
SELECT *
FROM employees
WHERE manager_id IS NULL;
```

14. How to find the department with the highest number of employees?

```
SELECT department_id, COUNT(*)
FROM employees
GROUP BY department_id
ORDER BY COUNT(*) DESC
LIMIT 1;
```

15. How to get the count of employees in each department?

```
SELECT department_id, COUNT(*)
FROM employees
GROUP BY department_id;
```

16. Write a query to fetch employees having the highest salary in each department.

```
SELECT department_id, employee_id, salary
FROM employees AS e
WHERE salary = (SELECT MAX(salary)
FROM employees
WHERE department_id = e.department_id);
```

17. How to write a query to update the salary of all employees by 10%?

```
UPDATE employees
SET salary = salary * 1.1;
```

18. How can you find employees whose salary is between 50,000 and 1,00,000?

```
SELECT *
FROM employees
WHERE salary BETWEEN 50000 AND 100000;
```

19. How to find the youngest employee in the organization?

```
SELECT *
FROM employees
ORDER BY birth_date DESC
LIMIT 1;
```

20. How to fetch the first and last record from a table?

```
(SELECT * FROM employees ORDER BY employee_id ASC LIMIT 1)
UNION ALL
(SELECT * FROM employees ORDER BY employee_id DESC LIMIT 1);
```

21. Write a query to find all employees who report to a specific manager.

```
SELECT *
FROM employees
WHERE manager_id = ?;
```

22. How can you find the total number of departments in the company?

```
SELECT COUNT(DISTINCT department_id)
FROM employees;
```

23. How to find the department with the lowest average salary?

```
SELECT department_id, AVG(salary)
FROM employees
GROUP BY department_id
```

```
ORDER BY AVG(salary) ASC
LIMIT 1;
```

24. How to delete all employees from a department in one query?

```
DELETE FROM employees
WHERE department_id = ?;
```

25. How to display all employees who have been in the company for more than 5 years?

```
SELECT *
FROM employees
WHERE DATEDIFF(CURDATE(), join_date) > 1825;
```

26. How to find the second-largest value from a table?

```
SELECT MAX(column_name)
FROM table_name
WHERE column_name < (SELECT MAX(column_name) FROM table_name);
```

27. How to write a query to remove all records from a table but keep the table structure?

```
TRUNCATE TABLE table_name;
```

28. Write a query to get all employee records in XML format.

```
SELECT employee_id, name, department_id
FROM employees
FOR XML AUTO;
```

29. How to get the current month's name from ?

```
SELECT MONTHNAME(CURDATE());
```

30. How to convert a string to lowercase in ?

```
SELECT LOWER('STRING_VALUE');
```

31. How to find all employees who do not have any subordinates?

```
SELECT *
FROM employees
WHERE employee_id NOT IN (SELECT manager_id FROM employees WHERE manager_id IS NOT NULL);
```

32. Write a query to calculate the total sales per customer in a sales table.

```
SELECT customer_id, SUM(sales_amount)
FROM sales
GROUP BY customer_id;
```

33. How to write a query to check if a table is empty?

```
SELECT CASE
  WHEN EXISTS (SELECT 1 FROM table_name)
  THEN 'Not Empty'
  ELSE 'Empty'
END;
```

34. How to find the second highest salary for each department?

```
SELECT department_id, salary
FROM (SELECT department_id, salary,
        DENSE_RANK() OVER (PARTITION BY department_id ORDER BY salary DESC) AS rank
    FROM employees) AS ranked_salaries
WHERE rank = 2;
```

35. Write a query to fetch employees whose salary is a multiple of 10,000.

```
SELECT *
FROM employees
WHERE salary % 10000 = 0;
```

36. How to fetch records where a column has null values?

```
SELECT *
FROM employees
WHERE column_name IS NULL;
```

37. How to write a query to find the total number of employees in each job title?

```
SELECT job_title, COUNT(*)
FROM employees
GROUP BY job_title;
```

38. Write a query to fetch all employees whose names end with ‘n’.

```
SELECT *
FROM employees
WHERE name LIKE '%n';
```

39. How to find all employees who work in both departments 101 and 102?

```
SELECT employee_id
FROM employees
WHERE department_id IN (101, 102)
GROUP BY employee_id
HAVING COUNT(DISTINCT department_id) = 2;
```

40. Write a query to fetch the details of employees with the same salary.

```
SELECT *
FROM employees
WHERE salary IN (SELECT salary
        FROM employees
        GROUP BY salary
        HAVING COUNT(*) > 1);
```

41. How to update salaries of employees based on their department?

```
UPDATE employees
SET salary = CASE
  WHEN department_id = 101 THEN salary * 1.10    WHEN
department_id = 102 THEN salary * 1.05
  ELSE salary
END;
```

42. How to write a query to list all employees without a department?

```
SELECT *
FROM employees
WHERE department_id IS NULL;
```

43. Write a query to find the maximum salary and minimum salary in each department.

```
SELECT department_id, MAX(salary), MIN(salary)
FROM employees
GROUP BY department_id;
```

44. How to list all employees hired in the last 6 months?

```
SELECT *
FROM employees
```

```
WHERE hire_date > ADDDATE(CURDATE(), INTERVAL -6 MONTH);
```

45. Write a query to display department-wise total and average salary.

```
SELECT department_id, SUM(salary) AS total_salary, AVG(salary) AS avg_salary
FROM employees
GROUP BY department_id;
```

46. How to find employees who joined the company in the same month and year as their manager?

```
SELECT e.employee_id, e.name
FROM employees e
JOIN employees m ON e.manager_id = m.employee_id
WHERE MONTH(e.join_date) = MONTH(m.join_date)
  AND YEAR(e.join_date) = YEAR(m.join_date);
```

47. Write a query to count the number of employees whose names start and end with the same letter.

```
SELECT COUNT(*)
FROM employees
WHERE LEFT(name, 1) = RIGHT(name, 1);
```

48. How to retrieve employee names and salaries in a single string?

```
SELECT CONCAT(name, ' earns ', salary) AS employee_info
FROM employees;
```

49. How to find employees whose salary is higher than their manager's salary?

```
SELECT e.employee_id, e.name
FROM employees e
JOIN employees m ON e.manager_id = m.employee_id
```

```
WHERE e.salary > m.salary;
```

50. Write a query to get employees who belong to departments with less than 3 employees.

```
SELECT *
FROM employees
WHERE department_id IN (SELECT department_id
                        FROM employees
                        GROUP BY department_id
                        HAVING COUNT(*) < 3);
```

51. How to write a query to find employees with the same first name?

```
SELECT *
FROM employees
WHERE first_name IN (SELECT first_name
                     FROM employees
                     GROUP BY first_name
                     HAVING COUNT(*) > 1);
```

52. How to write a query to delete employees who have been in the company for more than 15 years?

```
DELETE FROM employees
WHERE DATEDIFF(CURDATE(), join_date) > 5475;
```

53. Write a query to list all employees working under the same manager.

```
SELECT *
FROM employees
WHERE manager_id = ?;
```

54. How to find the top 3 highest-paid employees in each department?

```
SELECT *
FROM (SELECT *,
        DENSE_RANK() OVER (PARTITION BY department_id ORDER BY salary DESC) AS rank
    FROM employees) AS ranked_employees
WHERE rank <= 3;
```

55. Write a query to list all employees with more than 5 years of experience in each department.

```
SELECT *
FROM employees
WHERE DATEDIFF(CURDATE(), join_date) > 1825;
```

56. How to list all employees in departments that have not hired anyone in the past 2 years?

```
SELECT *
FROM employees
WHERE department_id IN (SELECT department_id
                        FROM employees
                        GROUP BY department_id
                        HAVING MAX(hire_date) < ADDDATE(CURDATE(), INTERVAL -2 YEAR));
```

57. Write a query to find all employees who earn more than the average salary of their department.

```
SELECT *
FROM employees e
WHERE salary > (SELECT AVG(salary)
                FROM employees
                WHERE department_id = e.department_id);
```

58. How to list all managers who have more than 5 subordinates?

```
SELECT *
FROM employees
WHERE employee_id IN (SELECT manager_id
                      FROM employees
                      GROUP BY manager_id
                      HAVING COUNT(*) > 5);
```

59. Write a query to display employee names and hire dates in the format "Name - MM/DD/YYYY".

```
SELECT CONCAT(name, ' - ', DATE_FORMAT(hire_date, '%m/%d/%Y')) AS employee_info
FROM employees;
```

60. How to find employees whose salary is in the top 10%?

```
SELECT *
FROM employees
WHERE salary >= (SELECT PERCENTILE_CONT(0.9)
                 WITHIN GROUP (ORDER BY salary ASC)
                 FROM employees);
```

61. Write a query to display employees grouped by their age brackets (e.g., 20-30, 31-40, etc.).

```
SELECT CASE
    WHEN age BETWEEN 20 AND 30 THEN '20-30'        WHEN age
BETWEEN 31 AND 40 THEN '31-40'
    ELSE '41+'
  END AS age_bracket,
  COUNT(*)
FROM employees
GROUP BY age_bracket;
```

62. How to find the average salary of the top 5 highest-paid employees in each department?

```
SELECT department_id, AVG(salary)
FROM (SELECT department_id, salary,
        DENSE_RANK() OVER (PARTITION BY department_id ORDER BY salary DESC) AS rank
    FROM employees) AS ranked_employees
WHERE rank <= 5
GROUP BY department_id;
```

63. How to calculate the percentage of employees in each department?

```
SELECT department_id,
    (COUNT(*) * 100.0 / (SELECT COUNT(*) FROM employees)) AS percentage
FROM employees
GROUP BY department_id;
```

64. Write a query to find all employees whose email contains the domain '@example.com'.

```
SELECT *
FROM employees
WHERE email LIKE '%@example.com';
```

65. How to retrieve the year-to-date sales for each customer?

```
SELECT customer_id, SUM(sales_amount)
FROM sales
WHERE sale_date BETWEEN '2024-01-01' AND CURDATE()
GROUP BY customer_id;
```

66. Write a query to display the hire date and day of the week for each employee.

```
SELECT name, hire_date, DAYNAME(hire_date) AS day_of_week
FROM employees;
```

67. How to find all employees who are older than 30 years?

```
SELECT *
FROM employees
WHERE DATEDIFF(CURDATE(), birth_date) / 365 > 30;
```

68. Write a query to display employees grouped by their salary range (e.g., 020K, 20K-50K).

```
SELECT CASE
        WHEN salary BETWEEN 0 AND 20000 THEN '0-20K'
        WHEN salary BETWEEN 20001 AND 50000 THEN '20K-50K'
        ELSE '50K+'
      END AS salary_range,
      COUNT(*)
FROM employees
GROUP BY salary_range;
```

69. How to list all employees who do not have a bonus?

```
SELECT *
FROM employees
WHERE bonus IS NULL;
```

70. Write a query to display the highest, lowest, and average salary for each job role.

```
SELECT job_role, MAX(salary) AS highest_salary, MIN(salary) AS lowest_salary, AVG(salary) AS
avg_salary
FROM employees
GROUP BY job_role;
```

Conclusion

Interviews for SQL-heavy positions (ETL developer, BI developer, data analyst, for example) can seem daunting because of the wide range of technical questions that can be asked.

But they don't have to be. Knowing your basics and a good level of SQL needed for the position you're applying for is all the interviewer will want to know.

Study these questions and understand the topics, and along with any experience you have, you'll be well prepared for an SQL interview.

Good luck!