120 Most Frequent SQL Interview Questions & Answers

Part 1: Basic SQL Concepts (Questions 1-20)

1. What is SQL?

Answer: SQL (Structured Query Language) is a standard language used for accessing, manipulating, and managing data stored in Relational Database Management Systems (RDBMS).

2. What is an RDBMS?

Answer: RDBMS stands for Relational Database Management System. It is a database management system based on the relational model introduced by E.F. Codd, where data is stored in tables (rows and columns). Examples: MySQL, PostgreSQL, Oracle, SQL Server.

3. What is the difference between DDL, DML, DCL, and TCL?

Answer:

- **DDL (Data Definition Language):** Defines structure (CREATE, ALTER, DROP, TRUNCATE).
- **DML (Data Manipulation Language):** Manipulates data (SELECT, INSERT, UPDATE, DELETE).
- **DCL (Data Control Language):** Controls access (GRANT, REVOKE).
- **TCL (Transaction Control Language):** Manages transactions (COMMIT, ROLLBACK, SAVEPOINT).

4. What is a Primary Key?

Answer: A Primary Key is a column (or set of columns) that uniquely identifies each row in a table. It cannot contain NULL values and must contain unique values.

5. What is a Foreign Key?

Answer: A Foreign Key is a field in one table that links to the Primary Key in another table. It is used to enforce referential integrity between the two tables.

6. What is a Unique Key?

Answer: A Unique Key constraint ensures that all values in a column are different. Unlike the Primary Key, a Unique Key allows one NULL value (in most databases).

7. What is the difference between DELETE and TRUNCATE?

Answer:

- **DELETE:** DML command. Deletes specific rows (can use WHERE). Can be rolled back. Slower.
- **TRUNCATE:** DDL command. Removes all rows from a table. Resets identity counters. Cannot be rolled back (in most systems). Faster.

8. What is the difference between CHAR and VARCHAR?

Answer:

- **CHAR:** Fixed-length character string. If the data is shorter, it adds padding spaces.
- **VARCHAR:** Variable-length character string. It stores only the characters used plus a small overhead for length.

9. What is the SELECT statement?

Answer: It is used to retrieve data from a database.

SELECT column1, column2 FROM table_name;

10. What is the DISTINCT keyword used for?

Answer: It is used to return only distinct (different) values, removing duplicates from the result set.

SELECT DISTINCT column_name FROM table_name;

11. What is the WHERE clause?

Answer: It is used to filter records that meet a certain condition.

SELECT * FROM table WHERE age > 18;

12. What are the AND, OR, and NOT operators?

Answer:

- **AND:** Displays a record if *all* conditions are true.
- **OR:** Displays a record if *any* condition is true.
- **NOT:** Displays a record if the condition is *not* true.

13. What is the ORDER BY clause?

Answer: It is used to sort the result-set in ascending (ASC) or descending (DESC) order.

SELECT * FROM table ORDER BY price DESC;

14. What are Aggregate Functions?

Answer: Functions that perform a calculation on a set of values and return a single value. Examples: COUNT(), SUM(), AVG(), MIN(), MAX().

15. What is the GROUP BY statement?

Answer: It groups rows that have the same values into summary rows, typically used with aggregate functions.

SELECT department, COUNT(*) FROM employees GROUP BY department;

16. What is the HAVING clause?

Answer: It was added to SQL because the WHERE keyword cannot be used with aggregate functions. It filters groups created by GROUP BY.

SELECT department, COUNT(*) FROM employees GROUP BY department HAVING COUNT(*) > 5;

17. What is the difference between WHERE and HAVING?

Answer:

- **WHERE:** Filters rows *before* grouping.
- **HAVING:** Filters groups *after* grouping.

18. What is the LIKE operator?

Answer: It is used in a WHERE clause to search for a specified pattern in a column.

- % represents zero, one, or multiple characters.
- _ represents a single character.

19. What is the IN operator?

Answer: It allows you to specify multiple values in a WHERE clause. It is a shorthand for multiple OR conditions.

SELECT * FROM table WHERE city IN ('Paris', 'London');

20. What is the BETWEEN operator?

Answer: It selects values within a given range. The values can be numbers, text, or dates. The range is inclusive.

Part 2: Joins & Relationships (Questions 21-40)

21. What is a JOIN?

Answer: A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

22. What are the different types of JOINS?

Answer:

1. **INNER JOIN:** Returns records that have matching values in both tables.
2. **LEFT (OUTER) JOIN:** Returns all records from the left table, and the matched records from the right table.
3. **RIGHT (OUTER) JOIN:** Returns all records from the right table, and the matched records from the left table.[123]
4. **FULL (OUTER) JOIN:** Returns all records when there is a match in either left or right table.[456]

23. What is a Self Join?789

Answer: A regular joi10n, but the table is joined with itself. It is useful for querying hierarchical data or co11mparing rows within the same table.12

24. What i13s a Cross Join (Cartesian Product)?

Answer: It returns the Cartesian product of the two tables (combines every row of the first table with every row of the second table).

25. What is the difference between UNION and UNION ALL?

Answer:

- **UNION:** Combines the result-set of two or more SELECT statements and **removes duplicates**.
- **UNION ALL:** Combines the result-set and **allows duplicates**. It is faster than UNION.

26. What is the INTERSECT operator?

Answer: It returns only the rows that appear in both result sets of two SELECT statements.

27. What is the MINUS (or EXCEPT) operator?

Answer: It returns the rows from the first SELECT statement that are not present in the second SELECT statement.

28. Can you join a table with itself? Give an example.

Answer: Yes (Self Join). Example: Finding employees who earn more than their managers.

SELECT e.name FROM Employee e JOIN Employee m ON e.manager_id = m.id WHERE e.salary > m.salary;

29. What happens if you drop a table that is referenced by a Foreign Key?

Answer: The database will usually throw an error preventing the drop to maintain referential integrity. You must drop the foreign key constraint or the child table first.

30. What is an Equi-Join?

Answer: A specific type of join that uses only the equality operator (=) in the join condition.

31. What is a Natural Join?

Answer: A join that creates an implicit join clause based on common columns in the two tables.

32. How do you find records in Table A that are not in Table B?

Answer: using LEFT JOIN where the right table is NULL, or using NOT EXISTS / NOT IN.

SELECT * FROM A LEFT JOIN B ON A.id = B.id WHERE B.id IS NULL;

33. What is an Alias in SQL?

Answer: A temporary name given to a table or column for the duration of a query (AS keyword). It improves readability.

34. What is a Composite Key?

Answer: A primary key that consists of two or more columns to uniquely identify a record.

35. What is Normalization?

Answer: The process of organizing data in a database to reduce redundancy and improve data integrity.

36. Explain 1NF, 2NF, and 3NF.

- **1NF:** Atomic values (no repeating groups).
- **2NF:** 1NF + No partial dependency (all non-key attributes depend on the *entire* primary key).
- **3NF:** 2NF + No transitive dependency (non-key attributes depend only on the primary key, not on other non-key attributes).

37. What is Denormalization?

Answer: The process of adding redundancy to a database (combining tables) to improve read performance, usually at the cost of write performance and storage.

38. What is a Schema?

Answer: The logical collection of database objects (tables, views, indexes) usually associated with a database user.

39. What is a View?

Answer: A virtual table based on the result-set of an SQL statement. It does not store data itself but displays data stored in other tables.

40. Why use a View?

Answer: To simplify complex queries, restrict access to specific columns (security), and present data in a specific format.

Part 3: Advanced SQL & Functions (Questions 41-70)

41. What is a Subquery?

Answer: A query nested inside another query. It can be in the SELECT, FROM, or WHERE clause.

42. What is the difference between a Correlated and Non-Correlated Subquery?

Answer:

- **Non-Correlated:** Executes once independently and passes the result to the outer query.
- **Correlated:** Executes repeatedly, once for each row processed by the outer query. It depends on the outer query for values.

43. What is the EXISTS operator?

Answer: Used to test for the existence of any record in a subquery. It returns TRUE if the subquery returns one or more records.

44. What are String Functions?

Answer: CONCAT(), SUBSTRING(), LENGTH(), UPPER(), LOWER(), TRIM(), REPLACE().

45. What are Date Functions?

Answer: NOW(), CURDATE(), DATEDIFF(), DATE_ADD(), EXTRACT().

46. What is the CASE statement?

Answer: The SQL version of IF-THEN-ELSE logic.

```
SQL
SELECT name,
CASE
   WHEN age < 18 THEN 'Minor'
   ELSE 'Adult'
END
FROM users;
```

47. What is COALESCE()?

Answer: A function that returns the first non-null value in a list. COALESCE(val1, val2, 'default').

48. What is NULLIF()?

Answer: Returns NULL if two expressions are equal; otherwise, it returns the first expression. Useful for preventing division by zero errors.

49. What are Window Functions?

Answer: Functions that perform calculations across a set of table rows that are related to the current row (e.g., ROW_NUMBER, RANK, LEAD, LAG). Unlike aggregate functions, they do not collapse rows.

50. Difference between RANK(), DENSE_RANK(), and ROW_NUMBER()?

- **ROW_NUMBER():** Unique sequential number (1, 2, 3, 4).
- **RANK():** Skips numbers for ties (1, 2, 2, 4).
- **DENSE_RANK():** Does not skip numbers for ties (1, 2, 2, 3).

51. What are LEAD() and LAG() functions?

- **LEAD():** Accesses data from the *next* row.
- **LAG():** Accesses data from the *previous* row.

52. What is a Common Table Expression (CTE)?

Answer: A temporary result set defined with the WITH clause. It makes complex queries more readable and allows for recursion.

53. What is a Recursive CTE?

Answer: A CTE that references itself. Useful for querying hierarchical data like organizational charts or family trees.

54. What is a Stored Procedure?

Answer: A prepared SQL code that you can save, so the code can be reused over and over again. It can accept parameters.

55. What is a Trigger?

Answer: A stored procedure that automatically runs (executes) when an event occurs in the database server (e.g., BEFORE INSERT, AFTER UPDATE).

56. What is a Cursor?

Answer: A database object used to retrieve, manipulate, and traverse through a result set row-by-row.

57. What is an Index?

Answer: A data structure that improves the speed of data retrieval operations on a database table.

58. What is the difference between a Clustered and Non-Clustered Index?

- **Clustered:** Physically sorts the data on the disk. Only one per table (usually the Primary Key).
- **Non-Clustered:** Creates a separate structure that points to the data. A table can have multiple.

59. What are the disadvantages of Indexes?

Answer: They slow down INSERT, UPDATE, and DELETE operations because the index must also be updated. They also consume disk space.

60. What is ACID?

Answer: Properties that guarantee database transactions are processed reliably:

- **Atomicity:** All or nothing.
- **Consistency:** Valid state to valid state.
- **Isolation:** Transactions do not interfere.
- **Durability:** Data is saved permanently after commit.

61. What is Database Locking?

Answer: A mechanism to prevent multiple users from updating the same data simultaneously, ensuring data integrity.

62. What is a Deadlock?

Answer: A situation where two or more transactions are waiting for one another to give up locks, resulting in a standstill.

63. What is Query Optimization?

Answer: The process of identifying the most efficient way to execute a SQL query (e.g., using indexes, avoiding full table scans).

64. What is the EXPLAIN command?

Answer: Used to obtain a query execution plan, showing how the database intends to execute a query (which indexes are used, join methods, etc.).

65. What is SQL Injection?

Answer: A security vulnerability where an attacker interferes with the queries an application makes to its database.

66. How do you prevent SQL Injection?

Answer: Use Prepared Statements (Parameterized Queries) and Input Validation.

67. What is a Temporary Table?

Answer: A table that exists temporarily on the database server. It is automatically deleted when the session ends.

68. What is the difference between WHERE and ON clause in Joins?

Answer: ON defines the join condition (how tables link). WHERE filters the result after the join is potentially made. In Inner Joins, they perform similarly. In Outer Joins, filtering in ON preserves rows; filtering in WHERE removes them.

69. How do you clone a table (structure only)?

Answer: SELECT * INTO new_table FROM old_table WHERE 1 = 0; (SQL Server) or CREATE TABLE new_table LIKE old_table; (MySQL).

70. How do you clone a table (structure and data)?

Answer: SELECT * INTO new_table FROM old_table;

Part 4: Practical & Scenario-Based Queries (Questions 71-100)

71. Write a query to find the second highest salary.

Answer:

SELECT MAX(salary) FROM Employee WHERE salary < (SELECT MAX(salary) FROM Employee);

72. Write a query to find the Nth highest salary.

Answer:

SELECT salary FROM Employee ORDER BY salary DESC LIMIT N-1, 1; (MySQL)

73. Write a query to find duplicate rows in a table.

Answer:

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

74. How to delete duplicate rows but keep one?

Answer: Using a CTE and ROW_NUMBER():

SQL
```
WITH CTE AS (
  SELECT *, ROW_NUMBER() OVER(PARTITION BY id ORDER BY id) as rn
  FROM table_name
)
DELETE FROM CTE WHERE rn > 1;
```

75. Write a query to get current date and time.

Answer: SELECT NOW(); (MySQL) or SELECT GETDATE(); (SQL Server).

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

Answer: SELECT * FROM Employee WHERE name LIKE 'A%';

77. How to fetch alternate records from a table?

Answer:

- Even: SELECT * FROM table WHERE id % 2 = 0;
- Odd: SELECT * FROM table WHERE id % 2 <> 0;

78. Write a query to find the 3 highest earners in each department.

Answer:

SQL
```
SELECT department, name, salary FROM (
  SELECT *, DENSE_RANK() OVER(PARTITION BY department ORDER BY salary DESC) as rank
  FROM employees
) as t
WHERE rank <= 3;
```

79. Write a query to calculate the cumulative sum of salary.

Answer:

```
SELECT id, salary, SUM(salary) OVER(ORDER BY id) as cumulative_sum FROM employees;
```

80. How to select the first 5 records from a table?

Answer:

- MySQL/PostgreSQL: `SELECT * FROM table LIMIT 5;`
- SQL Server: `SELECT TOP 5 * FROM table;`

81. Write a query to find employees hired in the last 30 days.

Answer: `SELECT * FROM Employees WHERE hire_date >= DATE_SUB(CURDATE(), INTERVAL 30 DAY);`

82. How do you change a column name in a table?

Answer: `ALTER TABLE table_name RENAME COLUMN old_name TO new_name;`

83. How do you add a new column to a table?

Answer: `ALTER TABLE table_name ADD column_name datatype;`

84. How do you drop a column from a table?

Answer: `ALTER TABLE table_name DROP COLUMN column_name;`

85. Write a query to find common records between two tables.

Answer: `SELECT * FROM TableA INTERSECT SELECT * FROM TableB;`

86. Write a query to calculate the age from a Date of Birth.

Answer: `SELECT TIMESTAMPDIFF(YEAR, dob, CURDATE()) AS age FROM users;`

87. How to find rows that contain only numeric data in a VARCHAR column?

Answer: SELECT * FROM table WHERE column REGEXP '^[0-9]+$';

88. Write a query to combine First Name and Last Name.

Answer: SELECT CONCAT(First_Name, ' ', Last_Name) AS Full_Name FROM Employees;

89. What does the <> operator mean?

Answer: It means "Not Equal To" (Same as !=).

90. What is the standard port for MySQL and PostgreSQL?

Answer: MySQL: 3306, PostgreSQL: 5432.

91. Can a Primary Key be based on multiple columns?

Answer: Yes, this is called a Composite Primary Key.

92. What is Data Integrity?

Answer: Ensuring the accuracy and consistency of data (Entity, Referential, Domain integrity).

93. What is a "Safe Update"?

Answer: A mode in some databases (like MySQL) that prevents UPDATE or DELETE statements without a WHERE clause that uses a KEY.

94. What is a Bitmap Index?

Answer: A special index used for columns with low cardinality (few unique values like Gender: M/F). Common in Data Warehousing.

95. What is OLTP vs OLAP?

Answer:

- **OLTP (Online Transaction Processing):** Fast, day-to-day operations (Bank ATM).
- **OLAP (Online Analytical Processing):** Complex analysis, historical data (Business Intelligence).

96. What is a Pivot Table (in SQL context)?

Answer: A technique to rotate rows into columns (Transposing).

97. How to handle NULL values in calculations?

Answer: Use COALESCE(column, 0) or IFNULL(column, 0) to convert NULL to zero before calculating.

98. Write a query to update a salary by 10% for a specific department.

Answer: UPDATE Employees SET salary = salary * 1.10 WHERE department = 'IT';

99. How to find the number of days between two dates?

Answer: SELECT DATEDIFF(date1, date2);

100. Write a query to randomly select 1 row.

Answer: SELECT * FROM table ORDER BY RAND() LIMIT 1;

Part 5: Deep Dive into JOINS (New Questions 101-120)

101. What is the output of an INNER JOIN if there are NULL values in the join column?

Answer: NULL values never match with anything (not even other NULLs). Rows with NULL in the joining column will be excluded from the result of an INNER JOIN.

102. If Table A has 5 rows (all 1s) and Table B has 3 rows (all 1s), how many rows will a CROSS JOIN return?

Answer: A Cross Join is a Cartesian product (5×3). It will return 15 rows.

103. Using the same tables (A=5 rows of 1s, B=3 rows of 1s), how many rows will an INNER JOIN return?

Answer: Since every row in A matches every row in B (1=1), it behaves exactly like a Cross Join. It will return 15 rows.

104. When would you use a FULL OUTER JOIN over a LEFT or RIGHT join?

Answer: Use a FULL OUTER JOIN when you want to see all data from both tables, regardless of whether there is a match. For example, to find "Orphan" records in both tables simultaneously (e.g., Customers without Orders AND Orders without valid Customers).

105. Write a query to find all employees who do NOT have a manager.

Answer: This requires a Self Join or a simple WHERE clause if the manager_id is in the same table.

SELECT name FROM Employees WHERE manager_id IS NULL;

106. Explain "Semi Join" vs. "Anti Join".

- **Semi Join:** Returns rows from the first table where a match exists in the second table (often used with EXISTS). It doesn't duplicate rows from the first table like an Inner Join might.
- **Anti Join:** Returns rows from the first table where NO match exists in the second table (often used with NOT EXISTS).

107. How can you simulate a FULL OUTER JOIN in MySQL (which doesn't support it directly)?

Answer: You must UNION a LEFT JOIN and a RIGHT JOIN.

SQL
SELECT * FROM A LEFT JOIN B ON A.id=B.id
UNION
SELECT * FROM A RIGHT JOIN B ON A.id=B.id;

108. Scenario: You have a table Students and Courses. How do you find students who have NOT enrolled in any course?

Answer: Use a LEFT JOIN where the right side is NULL.

SELECT s.name FROM Students s LEFT JOIN Enrollments e ON s.id = e.student_id WHERE e.course_id IS NULL;

109. What is the performance difference between JOIN and WHERE clause for joining tables (Implicit Join)?

Answer: In modern SQL optimizers, there is usually no performance difference between explicit (INNER JOIN) and implicit (WHERE table1.id = table2.id) joins. However, explicit JOIN syntax is preferred for readability and is required for Outer Joins.

110. Can you join a table on a specific condition other than equality (Non-Equi Join)? Give an example.

Answer: Yes.

SELECT e.name, s.grade FROM Employees e JOIN SalaryGrades s ON e.salary BETWEEN s.min_salary AND s.max_salary;

111. What happens to the result set if you use LEFT JOIN and filter by a column in the right table in the WHERE clause?

Answer: It effectively turns the LEFT JOIN into an INNER JOIN. The WHERE clause filters out the NULLs generated by the Left Join. (To preserve the Left Join behavior, put that condition in the ON clause).

112. How does a CROSS JOIN differ from an INNER JOIN with a condition ON 1=1?

Answer: Functionally, they are identical; both produce a Cartesian product. However, CROSS JOIN syntax explicitly states the intent to produce a product, whereas INNER JOIN ON 1=1 looks like a hack.

113. If Table A has 10 rows and Table B has 0 rows, how many rows will SELECT * FROM A LEFT JOIN B ON A.id=B.id return?

Answer: 10 rows. A Left Join always returns all rows from the left table. The columns from Table B will simply be NULL.

114. If Table A has 10 rows and Table B has 0 rows, how many rows will SELECT * FROM A INNER JOIN B ON A.id=B.id return?

Answer: 0 rows. An Inner Join requires a match in both tables.

115. Write a query to find pairs of cities that have the same name but are in different countries.

Answer: Self Join.

SELECT A.city, A.country, B.country FROM Cities A JOIN Cities B ON A.city = B.city AND A.country <> B.country;

116. What is a "Rolling Join" (or ASOF Join)?

Answer: Commonly used in time-series data (like finance). It joins a row to the closest matching timestamp in another table without requiring an exact match. (Supported natively in Snowflake/Pandas, requires complex logic in standard SQL).

117. What is a "Lateral Join" (or CROSS APPLY)?

Answer: It allows a subquery in the JOIN clause to reference columns from the preceding tables in the FROM clause. Useful for applying a function to every row of a table (e.g., getting the "Top 3 orders" for each customer).

118. How do you find records where the join key is NULL in both tables?

Answer: Join keys that are NULL will never match. You would have to select them separately using WHERE key IS NULL from both tables and UNION them, as standard joins drop NULL keys.

119. Scenario: You want to update Table A with values from Table B based on a matching ID. How do you write this?

Answer:

SQL
UPDATE TableA

```
SET TableA.value = TableB.value
FROM TableA
JOIN TableB ON TableA.id = TableB.id;
```

120. Explain the order of execution for a query with Joins, Where, and Group By.

Answer:

1. **FROM / JOIN:** Tables are combined first.
2. **WHERE:** Rows are filtered.
3. **GROUP BY:** Rows are grouped.
4. **HAVING:** Groups are filtered.
5. **SELECT:** Columns are selected.
6. **ORDER BY:** Final sorting.