**SQL Interview Questions and Answers**

### 1. What is SQL?

SQL (Structured Query Language) is a programming language used to manage and manipulate relational databases. It allows you to perform tasks like querying data, updating data, inserting data, and deleting data.

### 2. Write a query to create a table.

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DepartmentID INT,

HireDate DATE

);

### 3. Write a query to insert data into a table.

INSERT INTO Employees (EmployeeID, FirstName, LastName, DepartmentID, HireDate)

VALUES (1, 'John', 'Doe', 101, '2023-01-15');

### 4. Write a query to update data in a table.

UPDATE Employees

SET DepartmentID = 102

WHERE EmployeeID = 1;

### 5. Write a query to delete data from a table.

DELETE FROM Employees

WHERE EmployeeID = 1;

### 6. Write a query to select all columns from a table.

SELECT \* FROM Employees;

### 7. Write a query to select specific columns from a table.

SELECT FirstName, LastName FROM Employees;

### 8. Write a query to filter rows using WHERE.

SELECT \* FROM Employees

WHERE DepartmentID = 101;

### 9. Write a query to sort data using ORDER BY.

SELECT \* FROM Employees

ORDER BY HireDate DESC;

### 10. Write a query to group data using GROUP BY.

SELECT DepartmentID, COUNT(\*) AS EmployeeCount

FROM Employees

GROUP BY DepartmentID;

### 11. Write a query to filter groups using HAVING.

SELECT DepartmentID, COUNT(\*) AS EmployeeCount

FROM Employees

GROUP BY DepartmentID

HAVING COUNT(\*) > 5;

### 12. Write a query to join two tables.

SELECT Employees.EmployeeID, Employees.FirstName, Departments.DepartmentName

FROM Employees

INNER JOIN Departments ON Employees.DepartmentID = Departments.DepartmentID;

### 13. Write a query for a self-JOIN.

SELECT e1.EmployeeID, e1.FirstName, e2.ManagerID

FROM Employees e1

INNER JOIN Employees e2 ON e1.EmployeeID = e2.ManagerID;

### 14. Write a query to find duplicate rows.

SELECT FirstName, LastName, COUNT(\*)

FROM Employees

GROUP BY FirstName, LastName

HAVING COUNT(\*) > 1;

### 15. Write a query to remove duplicates.

DELETE FROM Employees

WHERE EmployeeID NOT IN (

SELECT MIN(EmployeeID)

FROM Employees

GROUP BY FirstName, LastName

);

### 16. Write a query to find the second highest salary.

SELECT MAX(Salary) AS SecondHighestSalary

FROM Employees

WHERE Salary < (SELECT MAX(Salary) FROM Employees);

### 17. Write a query to find the nth highest salary.

SELECT Salary

FROM Employees

ORDER BY Salary DESC

LIMIT 1 OFFSET n-1;

### 18. Write a query to calculate the total salary by department.

SELECT DepartmentID, SUM(Salary) AS TotalSalary

FROM Employees

GROUP BY DepartmentID;

### 19. Write a query to find employees with the same salary.

SELECT Salary, COUNT(\*) AS EmployeeCount

FROM Employees

GROUP BY Salary

HAVING COUNT(\*) > 1;

### 20. Write a query to find employees who do not have a manager.

SELECT \* FROM Employees

WHERE ManagerID IS NULL;

### 21. Write a query to find the top 5 highest-paid employees.

SELECT \* FROM Employees

ORDER BY Salary DESC

LIMIT 5;

### 22. Write a query to find employees hired in the last 6 months.

SELECT \* FROM Employees

WHERE HireDate >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH);

### 23. Write a query to find employees with a specific pattern in their name.

SELECT \* FROM Employees

WHERE FirstName LIKE 'Jo%';

### 24. Write a query to combine results from two tables using UNION.

SELECT FirstName FROM Employees

UNION

SELECT FirstName FROM Managers;

### 25. Write a query to find employees who earn more than their managers.

SELECT e1.EmployeeID, e1.FirstName, e1.Salary

FROM Employees e1

INNER JOIN Employees e2 ON e1.ManagerID = e2.EmployeeID

WHERE e1.Salary > e2.Salary;

### 26. Write a query to create an index.

CREATE INDEX idx\_employee\_name

ON Employees (FirstName, LastName);

### 27. Write a query to drop an index.

DROP INDEX idx\_employee\_name ON Employees;

### 28. Write a query to create a view.

CREATE VIEW HighPaidEmployees AS

SELECT \* FROM Employees

WHERE Salary > 100000;

### 29. Write a query to update a view.

CREATE OR REPLACE VIEW HighPaidEmployees AS

SELECT \* FROM Employees

WHERE Salary > 120000;

### 30. Write a query to delete a view.

DROP VIEW HighPaidEmployees;

### THEORY ****SQL Theory Interview Questions and Answers****

#### ****1. What is SQL?****

**Answer**: SQL (Structured Query Language) is a programming language used to communicate with and manipulate relational databases. It allows you to perform tasks like querying data, updating data, inserting data, and deleting data.

#### ****2. What is a Database?****

**Answer**: A database is an organized collection of structured data stored electronically in a computer system. It allows for efficient data retrieval, insertion, updating, and deletion.

#### ****3. What is a Relational Database?****

**Answer**: A relational database is a type of database that stores data in tables (rows and columns) and establishes relationships between these tables using keys (primary keys and foreign keys).

#### ****4. What is a Table?****

**Answer**: A table is a collection of related data organized in rows (records) and columns (fields). Each table in a database has a unique name.

#### ****5. What is a Primary Key?****

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

#### ****6. What is a Foreign Key?****

**Answer**: A foreign key is a column (or a set of columns) in one table that refers to the primary key in another table. It is used to enforce referential integrity between two tables.

#### ****7. What is a Composite Key?****

**Answer**: A composite key is a combination of two or more columns that uniquely identify a row in a table. It is used when a single column is not sufficient to uniquely identify a row.

#### ****8. What is a Unique Key?****

**Answer**: A unique key is a column (or a set of columns) that ensures all values in the column are unique. Unlike a primary key, it can contain NULL values.

#### ****9. What is an Index?****

**Answer**: An index is a database object that improves the speed of data retrieval operations on a table. It is created on one or more columns of a table.

#### ****10. What is the difference between a Clustered and a Non-Clustered Index?****

**Answer**:

**Clustered Index**: Determines the physical order of data in a table. A table can have only one clustered index.

**Non-Clustered Index**: Does not alter the physical order of data. A table can have multiple non-clustered indexes.

#### ****11. What is a View?****

**Answer**: A view is a virtual table based on the result set of an SQL query. It does not store data itself but displays data from one or more tables.

#### ****12. What is a Stored Procedure?****

**Answer**: A stored procedure is a precompiled collection of SQL statements that can be executed as a single unit. It is used to encapsulate and reuse SQL logic.

#### ****13. What is a Trigger?****

**Answer**: A trigger is a database object that automatically executes a set of SQL statements when a specific event (e.g., INSERT, UPDATE, DELETE) occurs on a table.

#### ****14. What is Normalization?****

**Answer**: Normalization is the process of organizing data in a database to reduce redundancy and improve data integrity. It involves dividing large tables into smaller tables and defining relationships between them.

#### ****15. What are the different Normal Forms?****

**Answer**:

**1NF (First Normal Form)**: Each column contains atomic values, and each row is unique.

**2NF (Second Normal Form)**: The table is in 1NF, and all non-key columns are fully dependent on the primary key.

**3NF (Third Normal Form)**: The table is in 2NF, and there are no transitive dependencies.

**BCNF (Boyce-Codd Normal Form)**: A stronger version of 3NF.

**4NF (Fourth Normal Form)**: The table is in BCNF, and there are no multi-valued dependencies.

#### ****16. What is Denormalization?****

**Answer**: Denormalization is the process of combining tables to improve read performance at the cost of write performance. It is used in data warehousing and reporting systems.

#### ****17. What is a Transaction?****

**Answer**: A transaction is a sequence of SQL operations performed as a single logical unit of work. It ensures data integrity by following the ACID properties (Atomicity, Consistency, Isolation, Durability).

#### ****18. What are the ACID Properties?****

**Answer**:

**Atomicity**: Ensures that all operations in a transaction are completed successfully or none are.

**Consistency**: Ensures that the database remains in a valid state before and after the transaction.

**Isolation**: Ensures that concurrent transactions do not interfere with each other.

**Durability**: Ensures that the results of a transaction are permanently stored in the database.

#### ****19. What is a Deadlock?****

**Answer**: A deadlock occurs when two or more transactions are waiting for each other to release locks on resources, resulting in a standstill.

#### ****20. What is the difference between****CHAR****and****VARCHAR****?****

**Answer**:

**CHAR**: Fixed-length character data. Pads spaces if the data is shorter than the defined length.

**VARCHAR**: Variable-length character data. Does not pad spaces.

#### ****21. What is the difference between****DELETE****and****TRUNCATE****?****

**Answer**:

**DELETE**: Removes rows from a table based on a condition. It can be rolled back and fires triggers.

**TRUNCATE**: Removes all rows from a table. It cannot be rolled back and does not fire triggers.

#### ****22. What is the difference between****DROP****and****TRUNCATE****?****

**Answer**:

**DROP**: Deletes the entire table structure and data.

**TRUNCATE**: Deletes all rows from a table but retains the table structure.

#### ****23. What is the difference between****WHERE****and****HAVING****?****

**Answer**:

**WHERE**: Filters rows before grouping (used with SELECT, UPDATE, DELETE).

**HAVING**: Filters groups after grouping (used with GROUP BY).

#### ****24. What is the difference between****INNER JOIN****and****LEFT JOIN****?****

**Answer**:

**INNER JOIN**: Returns only matching rows from both tables.

**LEFT JOIN**: Returns all rows from the left table and matching rows from the right table.

#### ****25. What is a Subquery?****

**Answer**: A subquery is a query nested inside another query. It is used to return data that will be used in the main query.

#### ****26. What is a Correlated Subquery?****

**Answer**: A correlated subquery is a subquery that depends on the outer query for its values. It is executed once for each row processed by the outer query.

#### ****27. What is the difference between****UNION****and****UNION ALL****?****

**Answer**:

**UNION**: Combines the result sets of two queries and removes duplicates.

**UNION ALL**: Combines the result sets of two queries without removing duplicates.

#### ****28. What is the difference between****IN****and****EXISTS****?****

**Answer**:

**IN**: Checks if a value matches any value in a list or subquery.

**EXISTS**: Checks if a subquery returns any rows.

#### ****29. What is a Self-JOIN?****

**Answer**: A self-JOIN is a join where a table is joined with itself. It is useful for querying hierarchical data or comparing rows within the same table.

#### ****30. What is a Cursor?****

**Answer**: A cursor is a database object used to retrieve and manipulate data row-by-row. It is typically used in stored procedures