1. What is SQL?

- **Answer**: SQL (Structured Query Language) is a standard programming language used for managing and manipulating relational databases.

---

2. What is a primary key?

- **Answer**: A primary key is a field (or combination of fields) that uniquely identifies each record in a table. It cannot contain `NULL` values and must have unique entries.

---

3. What is a foreign key?

- **Answer**: A foreign key is a field (or combination of fields) in one table that refers to the primary key in another table, establishing a relationship between the two tables.

---

4. What are constraints in SQL?

- **Answer**: Constraints are rules applied to table columns to enforce data integrity. Common constraints include:
  - `PRIMARY KEY`: Uniquely identifies each record.
  - `FOREIGN KEY`: Enforces referential integrity.
  - `UNIQUE`: Ensures all values in a column are distinct.
  - `CHECK`: Ensures that values in a column satisfy a specific condition.
  - `NOT NULL`: Ensures a column cannot have `NULL` values.
  - `DEFAULT`: Specifies a default value for a column.

---

5. Write a query to retrieve all records from a table named `employees`.

```
SELECT * FROM employees;
```

- **Answer**: This query selects and displays all columns and rows from the `employees` table.

6. What is the difference between `DELETE` and `TRUNCATE`?

- **Answer**:
  - `DELETE`: Removes rows from a table based on a condition. It can be rolled back (transaction-safe) and triggers can be invoked.
  - `TRUNCATE`: Removes all rows from a table, resetting the identity column. It is faster but cannot be rolled back and does not invoke triggers.

---

7. How do you find the maximum salary from an `employees` table?

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

- **Answer**: This query returns the highest salary from the `employees` table.

---

8. Write a query to fetch the second-highest salary from the `employees` table.

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

- **Answer**: The subquery finds the maximum salary, and the outer query finds the highest salary that is less than that value (i.e., the second-highest salary).

---

9. What is a `JOIN`? Explain its types.

- **Answer**: A `JOIN` clause is used to combine rows from two or more tables based on a related column. Types of joins:
  - `INNER JOIN`: Returns rows with matching values in both tables.
  - `LEFT JOIN`: Returns all rows from the left table and matching rows from the right.
  - `RIGHT JOIN`: Returns all rows from the right table and matching rows from the left.

- `FULL JOIN`: Returns rows when there is a match in either table.
- `CROSS JOIN`: Returns the Cartesian product of both tables.

---

10. Write a query to fetch employee names and department names using `JOIN`.

```
SELECT e.name, d.department_name
FROM employees e
JOIN departments d
ON e.department_id = d.id;
```

- **Answer**: This query joins the `employees` table with the `departments` table based on the `department_id`, displaying employee names and their corresponding department names.

---

11. What is a `GROUP BY` clause in SQL?

- **Answer**: The `GROUP BY` clause groups rows with the same values into summary rows. It is commonly used with aggregate functions like `COUNT()`, `SUM()`, `AVG()`, etc.

---

12. Write a query to count employees in each department.

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

- **Answer**: This query groups employees by `department_id` and counts the number of employees in each department.

---

13. What is the difference between `WHERE` and `HAVING` clauses?

- **Answer**:

- `WHERE`: Filters rows before grouping (applies to individual rows).
- `HAVING`: Filters groups after the `GROUP BY` clause (applies to aggregate functions).

---

14. Write a query to fetch departments with more than 5 employees.

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

- **Answer**: The query counts employees in each department and returns departments with more than 5 employees.

---

15. Explain `UNION` and `UNION ALL`.

- **Answer**:
  - `UNION`: Combines results of two or more `SELECT` statements and removes duplicates.
  - `UNION ALL`: Combines results and keeps all duplicates.

---

16. What is a subquery in SQL?

- **Answer**: A subquery is a query nested within another query. It is used to retrieve data that will be passed into the outer query.

---

17. Write a query to find all employees whose salary is greater than the average salary.

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

- **Answer**: This query selects all employees with a salary higher than the average salary of all employees.

---

18. What is the difference between `INNER JOIN` and `OUTER JOIN`?

- **Answer**:
  - `INNER JOIN`: Returns rows with matching values in both tables.
  - `OUTER JOIN` (Left/Right/Full): Returns matching rows plus non-matching rows from one or both tables.

---

19. Write a query to fetch the current date in SQL.

```
SELECT CURRENT_DATE;
```

- **Answer**: This query retrieves the current date from the database.

---

20. What is indexing in SQL?

- **Answer**: Indexing improves the speed of data retrieval by creating a data structure (index) on one or more columns of a table.

---

21. What is normalization? Explain its types (1NF, 2NF, 3NF, BCNF).

- **Answer**: Normalization is the process of organizing data to reduce redundancy and improve data integrity. Forms:
  - **1NF**: Eliminate duplicate columns and create tables for related data.
  - **2NF**: Remove partial dependencies (columns depend on a part of a composite key).
  - **3NF**: Remove transitive dependencies (non-key columns depend on other non-key columns).
  - **BCNF**: A stricter version of 3NF where every determinant must be a candidate key.

---

22. What is denormalization?

- **Answer**: Denormalization is the process of combining normalized tables to improve performance at the cost of introducing redundancy.

---

23. Write a query to add a new column `email` to the `employees` table.

```
ALTER TABLE employees ADD COLUMN email VARCHAR(255);
```

- **Answer**: This query adds a new `email` column to the `employees` table.

---

24. What is a stored procedure in SQL?

- **Answer**: A stored procedure is a set of SQL statements that can be stored in the database and executed as a program to perform a specific task.

---

25. Write a basic stored procedure to fetch all employees.

```
CREATE PROCEDURE GetAllEmployees()
BEGIN
   SELECT * FROM employees;
END;
```

- **Answer**: This procedure retrieves all records from the `employees` table when executed.

---

26. What are triggers in SQL?

- **Answer**: Triggers are special procedures that are automatically executed (or "triggered") in response to certain events (INSERT, UPDATE, DELETE) on a table.

---

27. Write a query to create a trigger that logs any delete action on the `employees` table.

```
CREATE TRIGGER log_delete
AFTER DELETE ON employees
FOR EACH ROW
BEGIN
   INSERT INTO log_table(action, emp_id, log_time)
   VALUES('DELETE', OLD.id, NOW());
END;
```

- **Answer**: This trigger logs the deletion of any employee by recording the action and employee ID in the `log_table`.

---

28. What is a `VIEW` in SQL?

- **Answer**: A `VIEW` is a virtual table based on the result set of an SQL query. It does not store the data itself but provides a way to simplify complex queries.

---

29. Write a query to create a view for employees with salary greater than 50,000.

```
CREATE VIEW HighSalaryEmployees AS
SELECT * FROM employees WHERE salary > 50000;
```

---

30. What is the difference between `VIEW` and `TABLE`?

- **Answer**: A `TABLE` stores data physically, while a `VIEW` is a virtual representation that dynamically pulls data from one or more tables without storing it.

---

31. What is an aggregate function? Provide examples.

- **Answer**: Aggregate functions perform calculations on a set of values and return a single value. Examples include:
  - `COUNT()`: Counts the number of rows.
  - `SUM()`: Sums up a numeric column.
  - `AVG()`: Calculates the average of a numeric column.
  - `MAX()`: Returns the maximum value.
  - `MIN()`: Returns the minimum value.

---

32. Write a query to calculate the total salary for each department.

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

- **Answer**: This query sums the salaries for each department, grouping by `department_id`.

---

33. Explain the `DISTINCT` keyword in SQL.

- **Answer**: The `DISTINCT` keyword is used to return unique values from a column, eliminating duplicate entries from the result set.

---

34. Write a query to find distinct job titles from the `employees` table.

```
SELECT DISTINCT job_title FROM employees;
```

- **Answer**: This query retrieves unique job titles from the `employees` table.

---

35. What are the ACID properties in SQL?

- **Answer**: ACID properties ensure reliable processing of database transactions:

- **Atomicity**: Ensures that all parts of a transaction are completed successfully or none at all.
- **Consistency**: Ensures the database remains in a valid state before and after the transaction.
- **Isolation**: Ensures transactions do not affect each other's execution.
- **Durability**: Ensures that once a transaction is committed, it remains permanent, even in the event of a failure.

---

36. What is a transaction in SQL?

- **Answer**: A transaction is a sequence of one or more SQL operations treated as a single unit of work, ensuring data integrity.

---

37. Explain `COMMIT`, `ROLLBACK`, and `SAVEPOINT`.

- **Answer**:
  - `COMMIT`: Saves all changes made during the current transaction.
  - `ROLLBACK`: Undoes changes made during the current transaction, restoring the database to its previous state.
  - `SAVEPOINT`: Sets a point within a transaction to which you can later roll back.

---

38. Write a query to start a transaction, update a record, and commit it.

```
START TRANSACTION;
UPDATE employees SET salary = 60000 WHERE id = 1;
COMMIT;
```

- **Answer**: This sequence starts a transaction, updates an employee's salary, and commits the change.

---

39. What is a `CASE` statement in SQL?

- **Answer**: A `CASE` statement is used to perform conditional logic in SQL queries, allowing different outputs based on specified conditions.

40. Write a query using `CASE` to categorize employees by salary.

```
SELECT name,
       CASE
          WHEN salary > 50000 THEN 'High'
          WHEN salary BETWEEN 30000 AND 50000 THEN 'Medium'
          ELSE 'Low'
       END AS salary_category
FROM employees;
```

- **Answer**: This query categorizes employees based on their salary levels.

---

41. Explain `NULL` values in SQL.

- **Answer**: `NULL` represents the absence of a value in a database. It is not equivalent to zero or an empty string and is treated differently in comparisons.

---

42. Write a query to fetch records where email is `NULL`.

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

- **Answer**: This query retrieves all employees whose email address is not provided (i.e., is `NULL`).

---

43. What is the `COALESCE` function in SQL?

- **Answer**: The `COALESCE` function returns the first non-NULL value in a list of expressions.

---

44. Write a query using `COALESCE` to handle `NULL` values in a column.

```sql
SELECT name, COALESCE(email, 'No Email') AS email_address
FROM employees;
```

- **Answer**: This query replaces `NULL` email values with the string 'No Email'.

---

45. What is the difference between `COUNT(*)` and `COUNT(column_name)`?

- **Answer**: `COUNT(*)` counts all rows in a table, including `NULL` values, while `COUNT(column_name)` counts only non-NULL values in the specified column.

---

46. What is the difference between `CHAR` and `VARCHAR` in SQL?

- **Answer**:
  - `CHAR`: Fixed-length string data type. If the string is shorter than the specified length, it is padded with spaces.
  - `VARCHAR`: Variable-length string data type. It uses only the necessary space for the string's length, plus one or two bytes for length information.

---

47. Write a query to update an employee's salary by 10% where the salary is below 30,000.

```sql
UPDATE employees
SET salary = salary * 1.10
WHERE salary < 30000;
```

- **Answer**: This query increases the salary of employees earning less than 30,000 by 10%.

---

48. What is a recursive query?

- **Answer**: A recursive query is a query that references itself. It is often used to handle hierarchical data, such as organizational structures.

49. Write a recursive query to get a hierarchical structure of employees and their managers.

```
WITH RECURSIVE EmployeeHierarchy AS (
    SELECT id, name, manager_id
    FROM employees
    WHERE manager_id IS NULL
    UNION ALL
    SELECT e.id, e.name, e.manager_id
    FROM employees e
    INNER JOIN EmployeeHierarchy eh
    ON e.manager_id = eh.id
)
SELECT * FROM EmployeeHierarchy;
```

- **Answer**: This recursive query generates a hierarchy of employees based on their managers.

---

50. Explain the `EXISTS` clause in SQL.

- **Answer**: The `EXISTS` clause is used to check for the existence of rows returned by a subquery. It returns `TRUE` if the subquery returns one or more rows.

---

51. What is the purpose of the `LIMIT` clause in SQL?

- **Answer**: The `LIMIT` clause is used to specify the maximum number of records to return in the result set. It helps in pagination and controlling output size.

---

52. Write a query to retrieve the top 5 highest salaries from the `employees` table.

```
SELECT DISTINCT salary
```

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

- **Answer**: This query selects the top 5 unique highest salaries from the `employees` table.

---

53. What is a composite key?

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

---

54. Explain the `ALTER TABLE` command.

- **Answer**: The `ALTER TABLE` command is used to modify an existing table structure, allowing changes such as adding, dropping, or modifying columns and constraints.

---

55. Write a query to drop a column named `address` from the `employees` table.

```
ALTER TABLE employees DROP COLUMN address;
```

- **Answer**: This query removes the `address` column from the `employees` table.

---

56. What is data integrity?

- **Answer**: Data integrity refers to the accuracy, consistency, and reliability of data over its lifecycle. It is maintained through various means, including constraints, data validation, and database rules.

---

57. Explain the difference between `INNER JOIN` and `LEFT JOIN`.

- **Answer**:
  - `INNER JOIN`: Returns only the rows with matching values in both tables.
  - `LEFT JOIN`: Returns all rows from the left table and the matched rows from the right table. If there is no match, `NULL` values are returned for the right table's columns.

---

58. What are the advantages of using indexes?

- **Answer**: Indexes improve query performance by reducing the amount of data the database must scan. They can speed up searches, sorts, and joins, but may slow down insertions, updates, and deletions due to the overhead of maintaining the index.

---

59. Write a query to find the average salary of employees in each department.

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

- **Answer**: This query calculates the average salary for each department by grouping records based on `department_id`.

---

60. What is a `CROSS JOIN`?

- **Answer**: A `CROSS JOIN` produces the Cartesian product of two tables, meaning every row from the first table is combined with every row from the second table.

---

61. Write a query to perform a `CROSS JOIN` between `employees` and `departments`.

```
SELECT * FROM employees CROSS JOIN departments;
```

- **Answer**: This query returns all combinations of rows from the `employees` and `departments` tables.

---

62. What is a SQL injection?

- **Answer**: SQL injection is a security vulnerability that occurs when an attacker can manipulate a SQL query by injecting malicious input through user input fields, potentially allowing unauthorized access to the database.

---

63. How can you prevent SQL injection?

- **Answer**: To prevent SQL injection:
  - Use prepared statements or parameterized queries.
  - Validate and sanitize user inputs.
  - Limit database user permissions.
  - Use ORM frameworks that handle data safely.

---

64. What is a database view? Can it be updated?

- **Answer**: A database view is a virtual table based on a SQL query that can simplify complex queries. While some views are updatable, updates depend on the complexity of the underlying query. Simple views that directly map to a single table can usually be updated.

---

65. What is the difference between `RANK()` and `DENSE_RANK()`?

- **Answer**:
  - `RANK()`: Assigns a rank to each row within a partition, with gaps in the ranking for ties.
  - `DENSE_RANK()`: Similar to `RANK()`, but without gaps in the ranking for ties.

---

66. Write a query to use `RANK()` to rank employees by salary.

```
SELECT name, salary, RANK() OVER (ORDER BY salary DESC) AS salary_rank
```

```
FROM employees;
```

- **Answer**: This query ranks employees based on their salaries, with the highest salary getting rank 1.

---

67. What is a schema in a database?

- **Answer**: A schema is a logical container for database objects like tables, views, indexes, and procedures. It defines the structure and organization of data.

---

68. Explain the `WITH` clause (Common Table Expression).

- **Answer**: The `WITH` clause defines a temporary result set that can be referenced within a `SELECT`, `INSERT`, `UPDATE`, or `DELETE` statement, often simplifying complex queries.

---

69. Write a query using a CTE to find employees with salaries greater than the average.

```
WITH AvgSalary AS (
    SELECT AVG(salary) AS avg_salary FROM employees
)
SELECT * FROM employees
WHERE salary > (SELECT avg_salary FROM AvgSalary);
```

- **Answer**: This query uses a CTE to calculate the average salary and then selects employees earning more than that average.

---

70. What are SQL data types? Give examples.

- **Answer**: SQL data types specify the kind of data that can be stored in a column. Examples include:
  - `INT`: Integer values.
  - `VARCHAR(n)`: Variable-length string.

- `DATE`: Date values.
- `FLOAT`: Floating-point numbers.
- `BOOLEAN`: True/false values.

---

71. What is the purpose of the `GROUP BY` clause?

- **Answer**: The `GROUP BY` clause groups rows that have the same values in specified columns into aggregated data, often used with aggregate functions.

---

72. Write a query to find the number of employees in each department and order by the count.

```
SELECT department_id, COUNT(*) AS employee_count
FROM employees
GROUP BY department_id
ORDER BY employee_count DESC;
```

- **Answer**: This query counts employees in each department and orders the results by the count in descending order.

---

73. What is a `SELF JOIN`?

- **Answer**: A `SELF JOIN` is a regular join that joins a table with itself. It is used when you need to compare rows within the same table.

---

74. Write a query to perform a `SELF JOIN` to find employees and their managers.

```
SELECT e1.name AS Employee, e2.name AS Manager
FROM employees e1
LEFT JOIN employees e2 ON e1.manager_id = e2.id;
```

- **Answer**: This query retrieves a list of employees along with their corresponding managers.

---

75. What is the difference between `WHERE` and `AND`?

- **Answer**: `WHERE` is used to specify conditions for filtering rows in a query, while `AND` is a logical operator used within the `WHERE` clause to combine multiple conditions.

---

76. Write a query to find employees with a salary between 30,000 and 50,000.

```
SELECT * FROM employees
WHERE salary BETWEEN 30000 AND 50000;
```

- **Answer**: This query retrieves employees whose salaries fall within the specified range.

---

77. What are stored functions in SQL?

- **Answer**: Stored functions are similar to stored procedures but return a single value. They can be used in SQL statements just like built-in functions.

---

78. Write a simple stored function to calculate the annual salary.

```
CREATE FUNCTION CalculateAnnualSalary(monthly_salary DECIMAL(10,2))
RETURNS DECIMAL(10,2)
BEGIN
    RETURN monthly_salary * 12;
END;
```

- **Answer**: This function takes a monthly salary and returns the annual salary.

---

79. What is the `EXPLAIN` statement used for?

- **Answer**: The `EXPLAIN` statement provides information about how a SQL query will be executed, including details on the chosen indexes, join types, and estimated rows. It helps optimize queries for better performance.

---

80. Write a query to use `EXPLAIN` to analyze a SELECT statement.

```
EXPLAIN SELECT * FROM employees WHERE department_id = 1;
```

- **Answer**: This query analyzes the execution plan for selecting employees from department 1.