**Basic SQL Concepts**

1. **What is SQL, and why is it important in data analytics?**  
   SQL (Structured Query Language) is a programming language used to manage and manipulate relational databases. It is essential in data analytics because it allows users to query, retrieve, and process data efficiently, enabling insights from large datasets. SQL is also used to perform tasks like filtering, aggregating, and joining data.
2. **Explain the difference between INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL OUTER JOIN.**
   * **INNER JOIN**: Returns only the rows where there is a match in both tables.
   * **LEFT JOIN**: Returns all rows from the left table and matching rows from the right table. Unmatched rows in the left table will show NULL for the right table's columns.
   * **RIGHT JOIN**: Returns all rows from the right table and matching rows from the left table. Unmatched rows in the right table will show NULL for the left table's columns.
   * **FULL OUTER JOIN**: Returns all rows when there is a match in either table. Unmatched rows from both tables will show NULL for the missing data.
3. **What is the difference between WHERE and HAVING clauses?**
   * **WHERE**: Filters rows before any grouping is performed. Used with individual rows.
   * **HAVING**: Filters groups after the grouping is performed. Used with aggregated data.
4. **How do you use GROUP BY and HAVING in a query?**  
   Example:

SELECT department, COUNT(\*) as employee\_count

FROM employees

GROUP BY department

HAVING COUNT(\*) > 10;

This groups employees by department, counts them, and filters groups with more than 10 employees.

1. **Write a query to find duplicate records in a table.**

SELECT column1, column2, COUNT(\*)

FROM table\_name

GROUP BY column1, column2

HAVING COUNT(\*) > 1;

1. **How do you retrieve unique values from a table using SQL?**  
   Use the DISTINCT keyword:

SELECT DISTINCT column\_name

FROM table\_name;

1. **Explain the use of aggregate functions like COUNT(), SUM(), AVG(), MIN(), and MAX().**
   * **COUNT()**: Counts the number of rows.
   * **SUM()**: Calculates the total of a numeric column.
   * **AVG()**: Computes the average of a numeric column.
   * **MIN()**: Finds the smallest value.
   * **MAX()**: Finds the largest value.
2. **What is the purpose of a DISTINCT keyword in SQL?**  
   The DISTINCT keyword is used to remove duplicate rows from the result set, ensuring that only unique values are returned.

**Intermediate SQL**

1. **Write a query to find the second-highest salary from an employee table.**

SELECT MAX(salary)

FROM employees

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

1. **What are subqueries and how do you use them?**  
   Subqueries are nested queries inside another SQL query. They are used to perform intermediate calculations or filtering.  
   Example:

SELECT name

FROM employees

WHERE salary = (SELECT MAX(salary) FROM employees);

1. **What is a Common Table Expression (CTE)? Give an example of when to use it.**  
   A CTE is a temporary result set that simplifies complex queries by breaking them into smaller, readable parts.  
   Example:

WITH SalesCTE AS (

SELECT employee\_id, SUM(sales) AS total\_sales

FROM sales

GROUP BY employee\_id

)

SELECT employee\_id

FROM SalesCTE

WHERE total\_sales > 10000;

1. **Explain window functions like ROW\_NUMBER(), RANK(), and DENSE\_RANK().**
   * **ROW\_NUMBER()**: Assigns a unique sequential number to rows.
   * **RANK()**: Assigns ranks to rows, with gaps for ties.
   * **DENSE\_RANK()**: Assigns ranks without gaps for ties.
2. **How do you combine results of two queries using UNION and UNION ALL?**
   * **UNION**: Combines results and removes duplicates.
   * **UNION ALL**: Combines results without removing duplicates.  
     Example:

SELECT name FROM employees

UNION

SELECT name FROM contractors;

1. **What are indexes in SQL, and how do they improve query performance?**  
   Indexes are data structures that improve query performance by reducing the amount of data scanned. They are particularly useful for searching, filtering, and sorting large datasets.
2. **Write a query to calculate the total sales for each month using GROUP BY.**

SELECT MONTH(order\_date) AS month, SUM(sales) AS total\_sales

FROM orders

GROUP BY MONTH(order\_date);

**Advanced SQL**

1. **How do you optimize a slow-running SQL query?**
   * Use indexes.
   * Avoid SELECT \* and only select required columns.
   * Optimize joins and subqueries.
   * Use query execution plans to identify bottlenecks.
   * Avoid unnecessary computations in queries.
2. **What are views in SQL, and when would you use them?**  
   Views are virtual tables created using a query. They simplify complex queries, enhance security by limiting data access, and improve reusability.  
   Example:

CREATE VIEW EmployeeSales AS

SELECT employee\_id, SUM(sales) AS total\_sales

FROM sales

GROUP BY employee\_id;

1. **What is the difference between a stored procedure and a function in SQL?**
   * **Stored Procedure**: Performs operations, may return zero or more values.
   * **Function**: Returns a single value and can be used in SQL expressions.
2. **Explain the difference between TRUNCATE, DELETE, and DROP commands.**
   * **TRUNCATE**: Deletes all rows in a table without logging individual row deletions. Cannot be rolled back.
   * **DELETE**: Deletes specific rows and can be rolled back.
   * **DROP**: Removes the entire table structure and its data.
3. **What are windowing functions, and how are they used in analytics?**  
   Windowing functions operate over a set of rows related to the current row, providing analytics like running totals or rankings.
4. **How do you use PARTITION BY and ORDER BY in window functions?**  
   Example:

SELECT employee\_id, department,

RANK() OVER (PARTITION BY department ORDER BY salary DESC) AS rank

FROM employees;

1. **How do you handle NULL values in SQL, and what functions help with that?**
   * Use COALESCE(value1, value2, ...) to return the first non-NULL value.
   * Use ISNULL(column, default\_value) to replace NULL values.  
     Example:

SELECT COALESCE(salary, 0) AS adjusted\_salary

FROM employees;