Common SQL Interview Questions

1. Write a query to find Year-over-Year (YoY) or Month-over-Month (MoM) growth percentage.

```
SELECT
    current_month,
    previous_month,
    (current_month - previous_month) / previous_month * 100 AS
growth_percentage
FROM
    sales_data;
```

2. Write a query to calculate the running total and running average of various products.

Running Total:

```
SELECT
    product_id,
    sales_date,
    sales_amount,
    SUM(sales_amount) OVER (PARTITION BY product_id ORDER BY sales_date) AS
running_total
FROM
    sales_data;
```

Running Average:

```
SELECT
    product_id,
    sales_date,
    sales_amount,
    AVG(sales_amount) OVER (PARTITION BY product_id ORDER BY sales_date
ROWS BETWEEN 4 PRECEDING AND CURRENT ROW) AS running_average
FROM
    sales_data;
```

3. Write a query using a self-join to get the full name of the manager and the count of people in every department, including departments with no employees.

```
SELECT
    d.department_name,
    COALESCE(m.manager_name, 'No Manager') AS manager_name,
    COUNT(e.employee_id) AS employee_count
FROM
    department d
LEFT JOIN
    employee e ON d.department_id = e.department_id
LEFT JOIN
    manager m ON d.manager_id = m.manager_id
GROUP BY
    d.department_name,
    m.manager_name;
```

4. Write a query to PIVOT data of a table in SQL.

```
SELECT *
FROM
    (SELECT product_name, month, sales_amount
     FROM sales_data) AS source_table
PIVOT
    (SUM(sales_amount) FOR month IN ([January], [February], [March])) AS
pivot_table;
```

5. Demonstrate the result of various types of joins on two tables with sample data.

Sample Data:

Table1: `1, 1, 0, 0, 1, 2, null` Table2: `1, 0, null, null, 1`

Inner Join:

```
SELECT *
FROM table1 t1
INNER JOIN table2 t2 ON t1.value = t2.value;
```

Left Join:

```
SELECT *
FROM table1 t1
LEFT JOIN table2 t2 ON t1.value = t2.value;
```

Right Join:

```
SELECT *
FROM table1 t1
RIGHT JOIN table2 t2 ON t1.value = t2.value;
```

6. Write a query to assign different categories such as L1, L2, etc., to employees based on their salary range.

```
SELECT
    employee_id,
    salary,
    CASE
        WHEN salary < 30000 THEN 'L1'
        WHEN salary BETWEEN 30000 AND 50000 THEN 'L2'
        WHEN salary > 50000 THEN 'L3'
    END AS salary_category
FROM
    employees;
```

7. Write a query to find duplicate entries in a table.

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

```
    COUNT(*) > 1;
```

8. Write a query to find the Nth highest salary in every department.

```
WITH RankedSalaries AS (
    SELECT
        department_id,
        salary,
        ROW_NUMBER() OVER (PARTITION BY department_id ORDER BY salary DESC)
AS rank
    FROM
        employees
)
SELECT
    department_id,
    salary
FROM
    RankedSalaries
WHERE
    rank = N;
```

Replace `N` with the desired rank value.