Nitya CloudTech Pvt Ltd.

SQL Scenario-Based Interview Questions & Answers

1. Calculate the Rolling Average Sales Over the Last 7 Days

Q: Write a query to calculate the rolling average sales for each day over the past 7 days.

A:

```
SELECT sale_date, sales_amount,
       AVG(sales_amount) OVER (ORDER BY sale_date ROWS BETWEEN 6
PRECEDING AND CURRENT ROW) AS rolling_avg_7_days
FROM daily_sales;
```

2. Identify the Second-Highest Salary in Each Department

Q: Write a query to find the second-highest salary in each department.

A:

```
SELECT department_id, MAX(salary) AS second_highest_salary
FROM employees
WHERE salary < (
    SELECT MAX(salary)
    FROM employees e
    WHERE e.department_id = employees.department_id
)
GROUP BY department_id;
```

3. Find Orders Placed Within a Specific Timeframe

Q: Retrieve all orders placed between 9:00 AM and 5:00 PM.

A:

```
SELECT order_id, order_time
FROM orders
WHERE CAST(order_time AS TIME) BETWEEN '09:00:00' AND '17:00:00';
```

4. Detect Data Gaps for Each Product

Q: Identify dates where no sales were recorded for each product in the `sales` table.

A:

```
SELECT p.product_id, d.date
FROM products p
CROSS JOIN dates d
```

```
LEFT JOIN sales s ON p.product_id = s.product_id AND d.date =
s.sale_date
WHERE s.sale_date IS NULL;
```

5. Calculate Cumulative Sum of Sales by Month

Q: Calculate the cumulative sales by month.

A:

```
SELECT month, sales_amount,
       SUM(sales_amount) OVER (ORDER BY month) AS cumulative_sales
FROM monthly_sales;
```

6. Identify Employees in Multiple Departments

Q: Write a query to identify employees assigned to more than one department.

A:

```
SELECT employee_id
FROM employee_departments
GROUP BY employee_id
HAVING COUNT(DISTINCT department_id) > 1;
```

7. Find Products with Zero Sales in the Last Quarter

Q: List all products that had no sales in the last quarter.

A:

```
SELECT product_id, product_name
FROM products
WHERE product_id NOT IN (
    SELECT DISTINCT product_id
    FROM sales
    WHERE sale_date >= DATEADD(quarter, -1, GETDATE())
);
```

8. Count Orders with Discounts in Each Category

Q: Count the number of orders with discounts in each category.

A:

```
SELECT category_id, COUNT(*) AS discounted_orders
FROM orders
WHERE discount > 0
GROUP BY category_id;
```

9. Identify Employees Whose Tenure is Below Average

Q: Find employees with tenure less than the average tenure of all employees.

A:

```
SELECT employee_id, name, tenure
FROM employees
WHERE tenure < (SELECT AVG(tenure) FROM employees);
```

10. Identify the Most Popular Product in Each Category

Q: Find the most popular product in each category based on sales.

A:

```
SELECT category_id, product_name, MAX(sales) AS max_sales
FROM products
GROUP BY category_id, product_name
ORDER BY max_sales DESC;
```

11. Detect Orders that Exceed a Monthly Threshold

Q: Identify orders that exceeded a $10,000 monthly threshold.

A:

```
SELECT customer_id, SUM(order_amount) AS total_spent
FROM orders
GROUP BY customer_id, MONTH(order_date)
HAVING SUM(order_amount) > 10000;
```

12. Find Customers Who Have Never Ordered a Specific Product

Q: Identify customers who have never ordered product "P123".

A:

```
SELECT customer_id
FROM customers
WHERE customer_id NOT IN (
    SELECT customer_id
    FROM orders
    WHERE product_id = 'P123'
);
```

13. Calculate Each Employee's Percentage of Departmental Sales

Q: Write a query to calculate each employee's sales as a percentage of total departmental sales.

A:

```
SELECT employee_id, sales,
       sales * 100.0 / SUM(sales) OVER (PARTITION BY department_id)
AS dept_sales_percentage
FROM employee_sales;
```

14. Find Products with Sales Growth Between Two Periods

Q: Write a query to identify products with sales growth from Q1 to Q2.

A:

```
SELECT p.product_id,
       q1.sales AS q1_sales, q2.sales AS q2_sales,
       (q2.sales - q1.sales) / NULLIF(q1.sales, 0) * 100 AS
growth_rate
FROM (SELECT product_id, SUM(sales) AS sales FROM sales WHERE quarter
= 'Q1' GROUP BY product_id) q1
JOIN (SELECT product_id, SUM(sales) AS sales FROM sales WHERE quarter
= 'Q2' GROUP BY product_id) q2
ON q1.product_id = q2.product_id;
```

15. Identify Customers with Consecutive Months of Purchases

Q: Find customers with orders in consecutive months.

A:

```
SELECT customer_id, order_date,
       DATEDIFF(month, LAG(order_date) OVER (PARTITION BY customer_id
ORDER BY order_date), order_date) AS months_diff
FROM orders
WHERE months_diff = 1;
```

16. Calculate Average Order Value (AOV) by Month

Q: Write a query to calculate AOV by month.

A:

```
SELECT month, AVG(order_amount) AS avg_order_value
FROM orders
GROUP BY month;
```

17. Rank Sales Representatives by Quarterly Performance

Q: Rank sales representatives based on quarterly sales.

A:

```
SELECT sales_rep_id, quarter, total_sales,
       RANK() OVER (PARTITION BY quarter ORDER BY total_sales DESC)
AS rank
FROM quarterly_sales;
```

18. Find the Month with the Highest Revenue in Each Year

Q: Write a query to find the month with the highest revenue for each year.

A:

```
SELECT year, month, revenue
FROM (
    SELECT year, month, revenue,
           RANK() OVER (PARTITION BY year ORDER BY revenue DESC) AS
rank
    FROM monthly_revenue
) AS yearly_revenue
WHERE rank = 1;
```

19. Identify Items with Stockouts

Q: Identify items that experienced stockouts (when stock quantity was zero).

A:

```
SELECT item_id, date
FROM inventory
WHERE stock_quantity = 0;
```

20. Calculate Average Time Between Orders by Customer

Q: Write a query to calculate the average time between orders for each customer.

A:

```
SELECT customer_id,
       AVG(DATEDIFF(day, LAG(order_date) OVER (PARTITION BY
customer_id ORDER BY order_date), order_date)) AS
avg_days_between_orders
FROM orders;
```