1. Query to find the second highest salary from an 'employees' table:

```
SELECT MAX(salary) AS second_highest_salary

FROM employees

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

2. Retrieve the top 3 employees with the highest total sales from a 'sales' table:

```
SELECT employee_id, SUM(sales_amount) AS total_sales

FROM sales

GROUP BY employee_id

ORDER BY total_sales DESC

LIMIT 3;
```

3. Write a query to calculate the running total of 'order_amount' in an 'orders' table:

```
SELECT order_id, order_amount, SUM(order_amount) OVER (ORDER BY order_id) AS running_total

FROM orders;
```

4. Find all employees who have duplicate 'email' addresses:

```
SELECT email, COUNT(email) AS email_count

FROM employees

GROUP BY email

HAVING COUNT(email) > 1;
```

5. Retrieve the 3 most recent orders for each customer:

```
SELECT customer_id, order_id, order_date

FROM (
  SELECT customer_id, order_id, order_date,
       ROW_NUMBER() OVER (PARTITION BY customer_id ORDER BY order_date DESC) AS rnk
  FROM orders
) ranked
WHERE rnk <= 3;
```

6. Identify customers who made consecutive orders on the same day:

```
SELECT customer_id, order_id, order_date
FROM (
  SELECT customer_id, order_id, order_date,
       LEAD(order_date) OVER (PARTITION BY customer_id ORDER BY order_date) AS next_order_date
  FROM orders
) consecutive_orders
WHERE DATEDIFF(next_order_date, order_date) = 0;
```

7. Pivot table: Transform rows to columns for each 'product' and its 'sales' in a specific date range:

```
SELECT *
FROM (
  SELECT product, sale_date, sales_amount
  FROM sales
  WHERE sale_date BETWEEN '2023-01-01' AND '2023-12-31'
) AS source
PIVOT (
  SUM(sales_amount)
  FOR product IN ('Product_A', 'Product_B', 'Product_C')
) AS pivot_table;
```

8. Calculate the median 'salary' for each 'department' in the 'employees' table:

```
SELECT department_id,
       PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY salary) OVER (PARTITION BY department_id) AS median_salary
FROM employees;
```

9. Find the top 5 departments with the highest average employee salary:

```
SELECT department_id, AVG(salary) AS avg_salary
FROM employees
GROUP BY department_id
ORDER BY avg_salary DESC
LIMIT 5;
```

10. Recursive Query: Display the hierarchy of employees in an 'org_structure' table:

```
WITH RECURSIVE EmployeeHierarchy AS (
   SELECT employee_id, manager_id, employee_name, 1 AS level
   FROM org_structure
   WHERE manager_id IS NULL

   UNION ALL

   SELECT o.employee_id, o.manager_id, o.employee_name, eh.level + 1
   FROM org_structure o
   JOIN EmployeeHierarchy eh ON o.manager_id = eh.employee_id
)
SELECT * FROM EmployeeHierarchy;
```

-Pragya Rathi