1. **Find Duplicate records in a table**

**SELECT col1, COUNT(\*)**

**FROM users**

**GROUP BY col1**

**HAVING COUNT(\*)>1;**

1. **Retrieve second highest salary from the emp table**

==============================================================

**WITH cte AS(**

**SELECT emp\_id, name,salary,**

**DENSE\_RANK() OVER(ORDER BY salary DESC) AS rnk**

**FROM employees**

**)**

**SELECT emp\_id, name, salary**

**FROM cte**

**WHERE rnk=2;**

=============================================================

**-- Step 1: Prepare the statement**

**PREPARE get\_nth\_salary(int) AS**

**WITH cte AS (**

**SELECT emp\_id, name, salary,**

**DENSE\_RANK() OVER (ORDER BY salary DESC) AS rnk**

**FROM employees**

**)**

**SELECT emp\_id, name, salary**

**FROM cte**

**WHERE rnk = $1;**

**-- Step 2: Execute it with a desired N (e.g., 3 for 3rd highest)**

**EXECUTE get\_nth\_salary(3);**

1. **Find employees without department**

**SELECT emp\_id, emp\_name**

**FROM employees e**

**LEFT JOIN department d**

**ON e.dept\_id = d.dept\_id**

**WHERE d.dept\_id IS NULL;**

1. **Calculate the total revenue per product**

**SELECT product\_id, SUM(quantity \* price)**

**FROM Sales s**

**JOIN Products p**

**USING(product\_id)**

**GROUP BY product\_id;**

1. **Get the top 3 highest paid employees**

**WITH cte AS(**

**SELECT emp\_id, emp\_name, salary,**

**DENSE\_RANK() OVER(ORDER BY salary DESC) AS rnk**

**FROM employees**

**)**

**SELECT emp\_id, emp\_name, salary**

**FROM cte**

**WHERE rnk<=3;**

1. **Customers who made purchases but never returned products**

**SELECT DISTINCT c.customer\_id**

**FROM customers c**

**JOIN purchases p**

**ON p. customer\_id = c. customer\_id**

**LEFT JOIN returns r**

**ON r.purchase\_id = P. purchase \_id**

**WHERE r.return\_id IS NULL;**

1. **Show the count of orders per customers**

**SELECT c.customer\_id, c.customer\_name, COUNT(\*)**

**FROM customers c**

**JOIN orders o**

**ON o.customer\_id = c.customer\_id**

**GROUP BY c.customer\_id, c.customer\_name;**

1. **Identify the most selling product**

**WITH cte AS (**

**SELECT**

**product\_id,**

**SUM(quantity) AS total\_quantity,**

**DENSE\_RANK() OVER (ORDER BY SUM(quantity) DESC) AS rnk**

**FROM sales**

**GROUP BY product\_id**

**)**

**SELECT p.product\_id**

**FROM products p**

**JOIN cte ON p.product\_id = cte.product\_id**

**WHERE cte.rnk = 1;**

1. **Get monthly sales revenue and order count**

**SELECT DATE\_TRUNC('MONTH', order\_date) AS month,**

**SUM(quantity\*price) AS revenue,**

**COUNT(order\_id) AS order\_count**

**FROM sales**

**GROUP BY month;**

1. **Rank employees by salary with in each department**

**SELECT department, emp\_id, salary,**

**DENSE\_RANK() OVER(PARTITION BY department ORDER BY salary DESC) AS rnk**

**FROM employees e**

**JOIN departments d**

**USING(department\_id);**

1. **Find customers who placed orders every month in 2023**

**SELECT customer\_id**

**FROM customers c**

**JOIN orders**

**USING(customer\_id)**

**WHERE (SELECT COUNT(DISTINCT DATE\_TRUNC('MONTH', ORDER\_DATE))**

**FROM orders o1**

**WHERE c.customer\_id = o1.customer\_id**

**AND o1.order\_date >= '2023-01-01' AND o1.order\_date < '2024-01-01') = 12**

**;**

1. **Find moving average of sales over the last 3 days**

**SELECT order\_date, amount,**

**ROUND(AVG(amount) OVER(ORDER BY order\_date ROWS BETWEEN 2 PRECEDING AND CURRENT ROW),2) AS avg\_3\_days**

**FROM sales;**

1. **Identify the first and last order date of each customer**

**SELECT customer\_id, MIN(order\_date) AS first\_order,**

**MAX(order\_date) AS last\_order**

**FROM orders**

**GROUP BY customer\_id;**

1. **Retrieve customers who made consecutive purchases (2 Days)**

**WITH cte AS(**

**SELECT customer\_id, purchase\_date,**

**LEAD(purchase\_date) OVER(PARTITION BY customer\_id ORDER BY purchase\_date) AS nxt\_purchase**

**FROM purchases**

**)**

**SELECT customer\_id, purchase\_date, nxt\_purchase**

**FROM cte**

**WHERE nxt\_purchase IS NOT NULL**

**AND (nxt\_purchase - purchase\_date) <= INTERVAL '2 days';**

1. **Find churned customers (no orders in last 6 months)**

**SELECT customer\_name**

**FROM customers c**

**WHERE NOT EXISTS (**

**SELECT 1**

**FROM orders o**

**WHERE o.customer\_id = c.customer\_id AND o.order\_date>= CURRENT\_DATE - INTERVAL '180 DAYS'**

**)**

**;**

1. **Calculate cumulative revenue per day**

**SELECT order\_date::date AS order\_day, SUM(quantity\*price) AS revenue,**

**SUM(SUM(quantity\*price)) OVER(ORDER BY order\_date::date) AS cum\_revenue**

**FROM orders**

**GROUP BY order\_day**

**ORDER BY order\_day;**

1. **Identify top performing departments by average salary**

**SELECT d.department\_name,**

**AVG(e.salary) AS avg\_department\_salary**

**FROM employees e**

**JOIN departments d ON e.department\_id = d.department\_id**

**GROUP BY d.department\_name**

**HAVING AVG(e.salary) > (**

**SELECT AVG(salary) FROM employees**

**)**

**ORDER BY avg\_department\_salary DESC;**

1. **Find customers who ordered more than the average number of orders per customer**

**SELECT customer\_id, COUNT(order\_id)**

**FROM orders**

**JOIN customers**

**USING(customer\_id)**

**GROUP BY customer\_id**

**HAVING COUNT(order\_id)>(**

**SELECT AVG(order\_count)**

**FROM (**

**SELECT COUNT(order\_id) AS order\_count**

**FROM orders**

**GROUP BY customer\_id**

**) AS sub**

**);**

1. **Calculate revenue generated by new customers (first time orders)**

**SELECT customer\_id, SUM(quantity\*price) AS revenue**

**FROM orders**

**GROUP BY customer\_id**

**HAVING COUNT(order\_id)=1;**

1. **Find the percentage of employees in each department**

**SELECT department\_id,**

**ROUND(COUNT(employee\_id)\*100/(SELECT COUNT(employee\_id) FROM employees),2)||'%' AS per\_emp**

**FROM employees**

**GROUP BY department\_id;**

1. **Retrieve maximum salary difference within each department**

**SELECT department\_id, MAX(salary)-MIN(salary) AS salary\_diff**

**FROM employees**

**GROUP BY department\_id**

1. **Find products that contribute to 80% of the revenue**