SQL

AMERICAN EXPRESS

Interview questions for Data Analysts

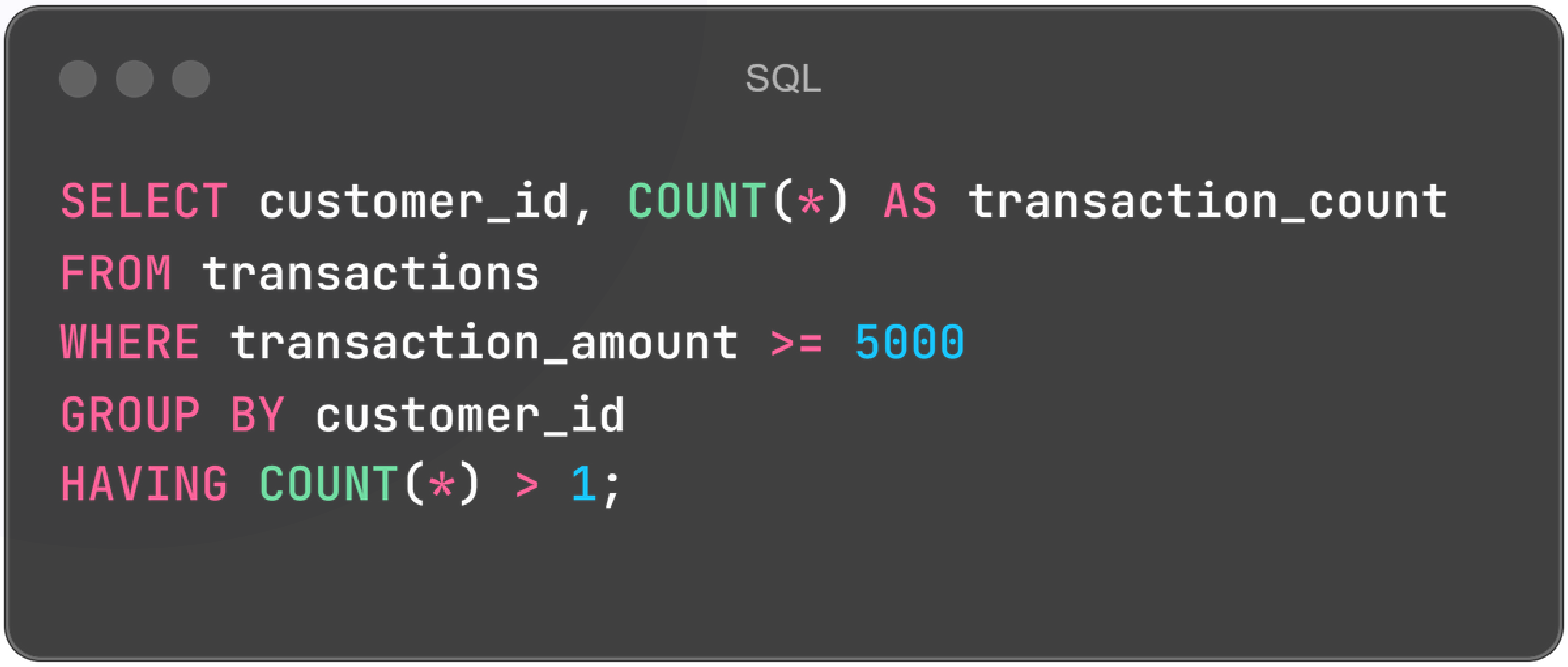
linkedin.com/in/ileonjose

1. Identify the VIP Customers for American Express

Problem Statement: Find customers who have made transactions exceeding $5000 each and have done so more than once. These customers are considered 'VIP' or 'Whale' customers.

How to Solve:

- Filter transactions with amounts greater than or equal to $5000.
- Group by customer and count the number of qualifying transactions.
- Filter groups with more than one qualifying transaction.



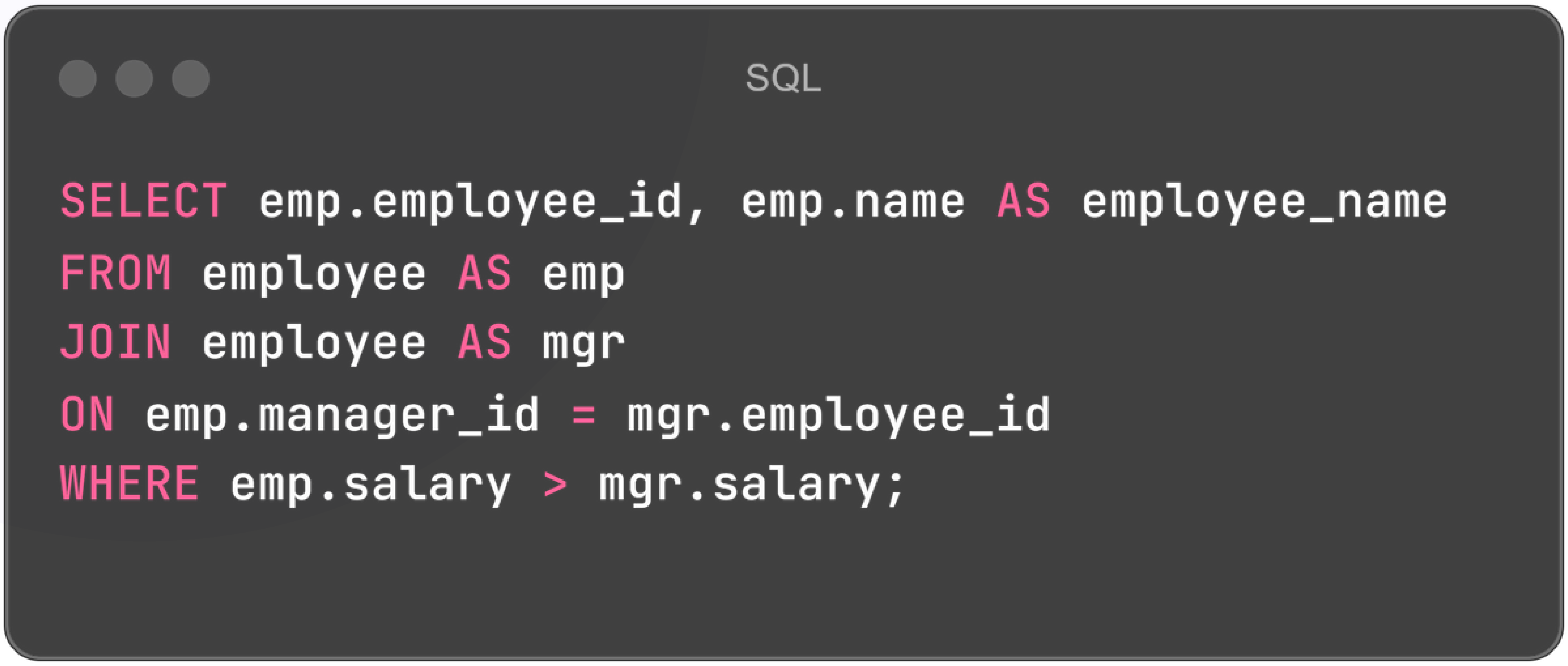
```sql
SELECT customer_id, COUNT(*) AS transaction_count
FROM transactions
WHERE transaction_amount >= 5000
GROUP BY customer_id
HAVING COUNT(*) > 1;
```

2. Employees Earning More Than Their Managers

Problem Statement: Identify employees whose salaries exceed those of their direct managers.

How to Solve:

- Perform a self-join on the employee table to compare employees with their managers.
- Filter where employee’s salary is greater than manager’s salary.



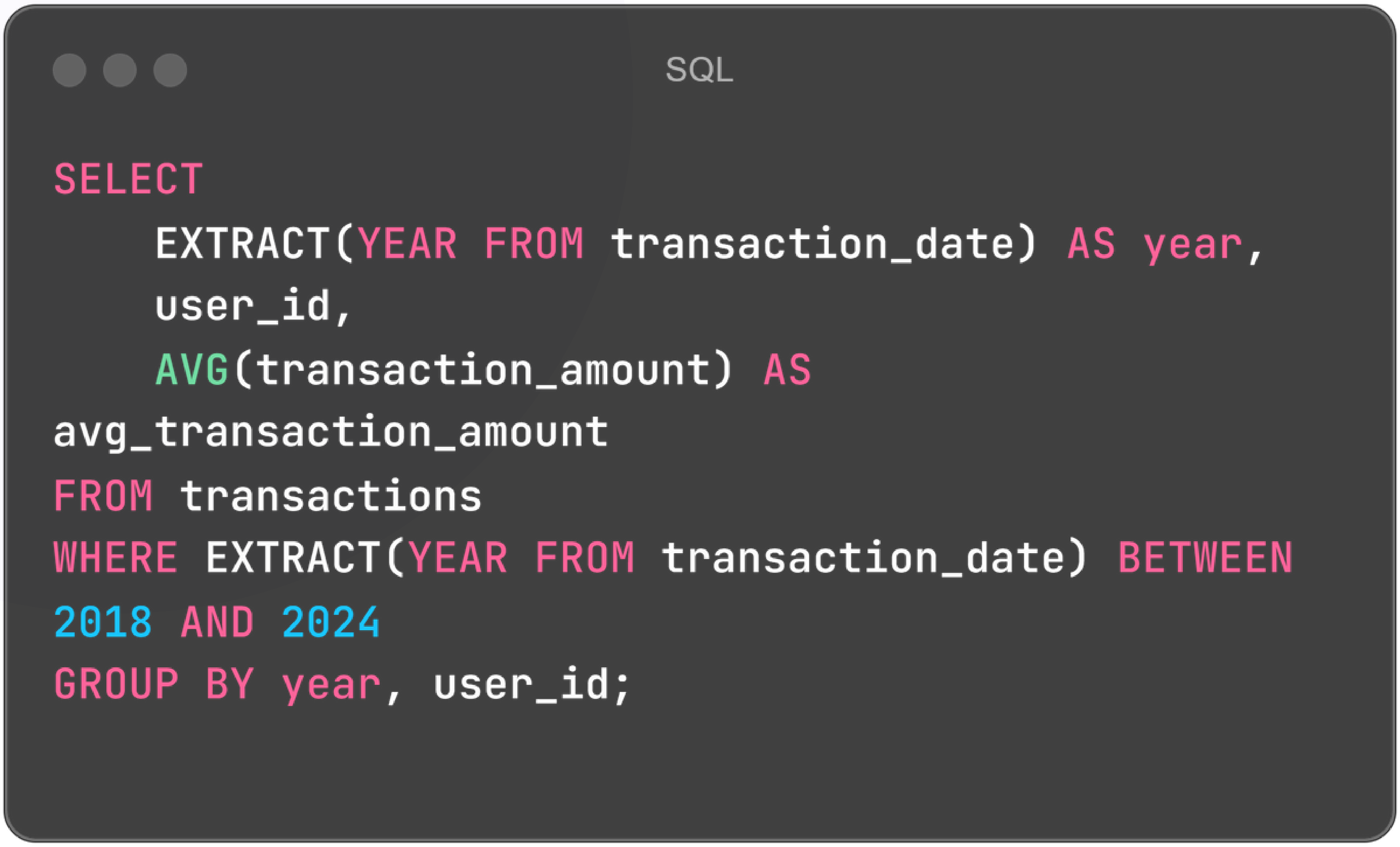
```sql
SELECT emp.employee_id, emp.name AS employee_name
FROM employee AS emp
JOIN employee AS mgr
ON emp.manager_id = mgr.employee_id
WHERE emp.salary > mgr.salary;
```

3. Calculate Average Transaction Amount per Year per Client

Problem Statement: Compute the average transaction amount for each client, segmented by year, for the years 2020 to 2024.

How to Solve:

- Extract the year from transaction dates.
- Group by client and year.
- Calculate the average transaction amount.



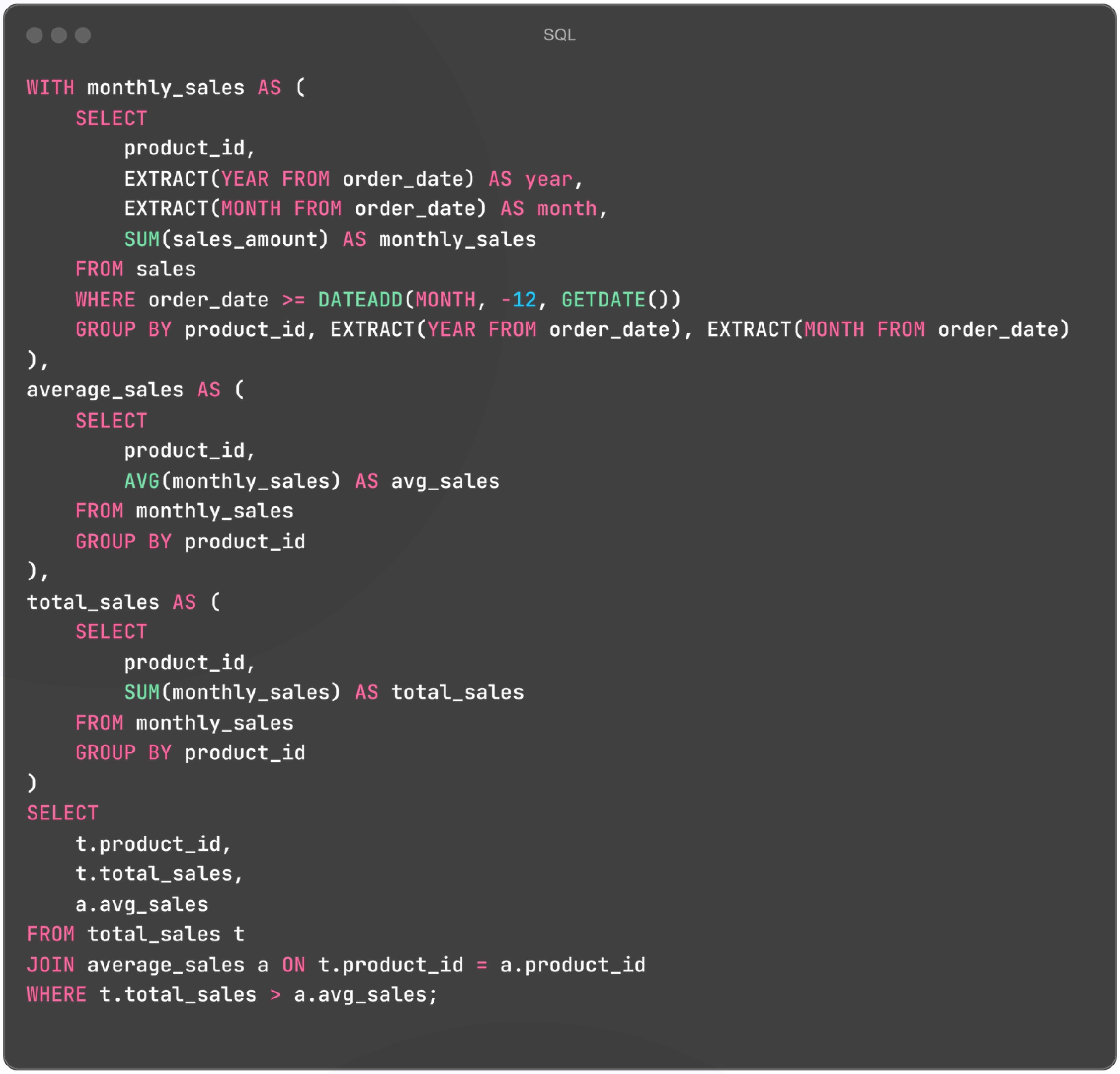
```sql
SELECT
    EXTRACT(YEAR FROM transaction_date) AS year,
    user_id,
    AVG(transaction_amount) AS
avg_transaction_amount
FROM transactions
WHERE EXTRACT(YEAR FROM transaction_date) BETWEEN
2018 AND 2024
GROUP BY year, user_id;
```

4. Find Products with Sales Greater Than Their Average Sales in the Last 12 Months

Problem Statement: Identify products whose total sales in the last 12 months exceed their average monthly sales.

How to Solve:

- Aggregate monthly sales for each product.
- Compute average sales per product.
- Compare total sales to average sales.



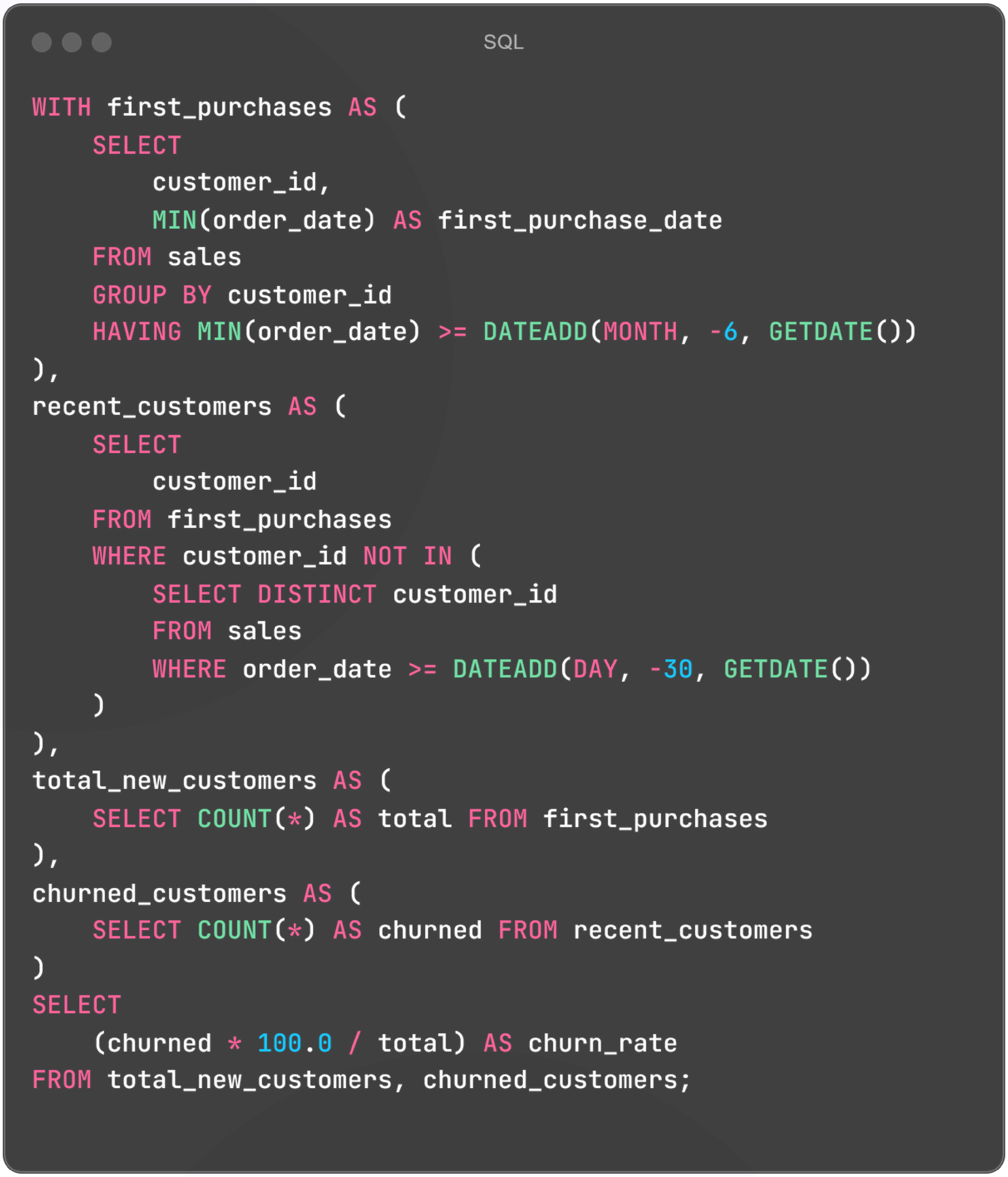
```sql
WITH monthly_sales AS (
    SELECT
        product_id,
        EXTRACT(YEAR FROM order_date) AS year,
        EXTRACT(MONTH FROM order_date) AS month,
        SUM(sales_amount) AS monthly_sales
    FROM sales
    WHERE order_date >= DATEADD(MONTH, -12, GETDATE())
    GROUP BY product_id, EXTRACT(YEAR FROM order_date), EXTRACT(MONTH FROM order_date)
),
average_sales AS (
    SELECT
        product_id,
        AVG(monthly_sales) AS avg_sales
    FROM monthly_sales
    GROUP BY product_id
),
total_sales AS (
    SELECT
        product_id,
        SUM(monthly_sales) AS total_sales
    FROM monthly_sales
    GROUP BY product_id
)
SELECT
    t.product_id,
    t.total_sales,
    a.avg_sales
FROM total_sales t
JOIN average_sales a ON t.product_id = a.product_id
WHERE t.total_sales > a.avg_sales;
```

5. Determine the Churn Rate for Customers Who Made Their First Purchase in the Last 6 Months

Problem Statement: Calculate the churn rate for customers who made their first purchase within the last 6 months but have not made any purchase in the last 30 days.

How to Solve:

- Identify customers with their first purchase in the last 6 months.
- Filter out customers who have not made a purchase in the last 30 days.
- Compute churn rate based on total new customers and churned customers.



```sql
WITH first_purchases AS (
    SELECT
        customer_id,
        MIN(order_date) AS first_purchase_date
    FROM sales
    GROUP BY customer_id
    HAVING MIN(order_date) >= DATEADD(MONTH, -6, GETDATE())
),
recent_customers AS (
    SELECT
        customer_id
    FROM first_purchases
    WHERE customer_id NOT IN (
        SELECT DISTINCT customer_id
        FROM sales
        WHERE order_date >= DATEADD(DAY, -30, GETDATE())
    )
),
total_new_customers AS (
    SELECT COUNT(*) AS total FROM first_purchases
),
churned_customers AS (
    SELECT COUNT(*) AS churned FROM recent_customers
)
SELECT
    (churned * 100.0 / total) AS churn_rate
FROM total_new_customers, churned_customers;
```

Found this helpful? Repost!

linkedin.com/in/ileonjose