Project: Food Delivery Orders

This document contains SQL queries generated to perform ad-hoc analysis of food delivery orders.

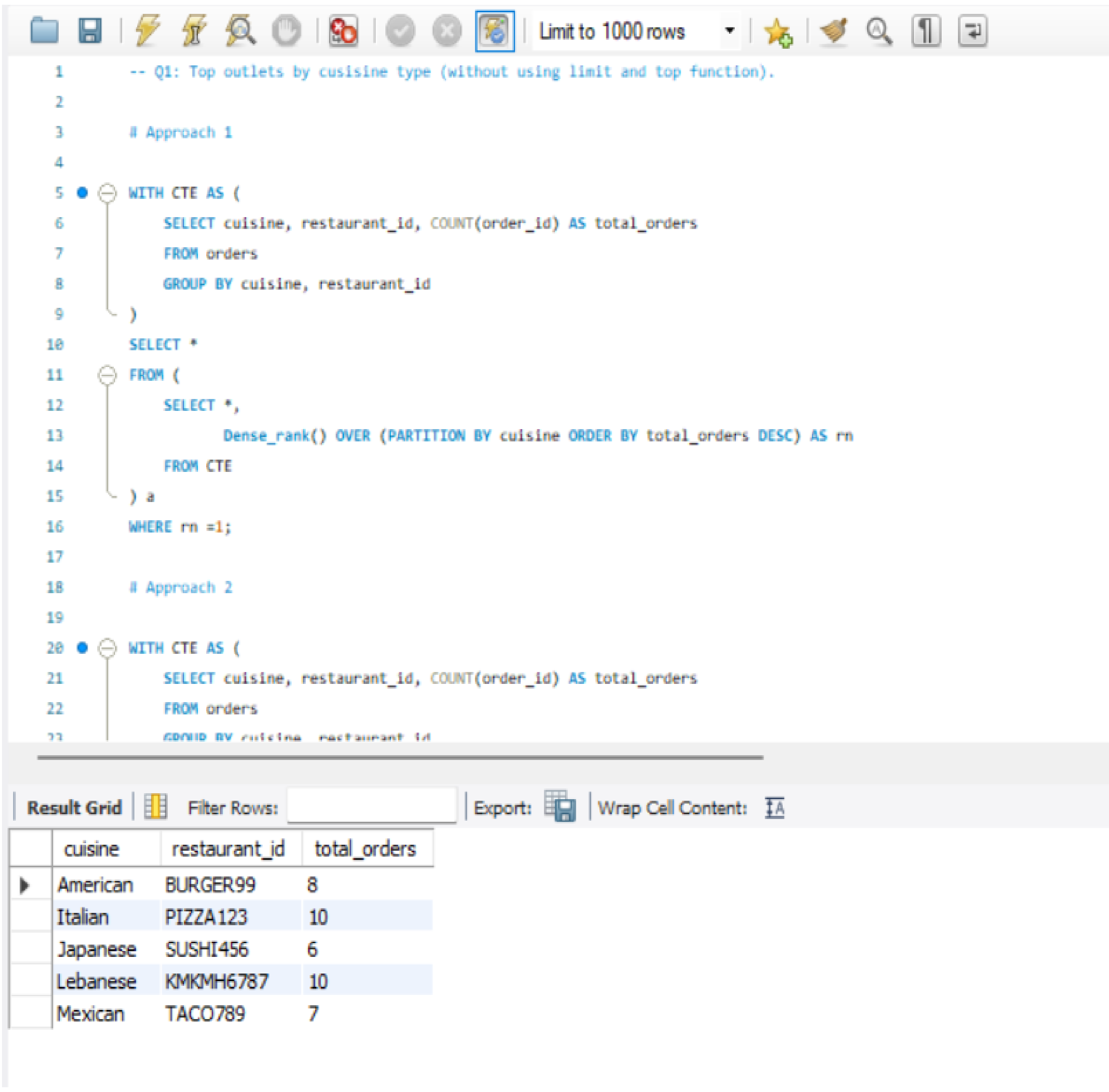
Data Analyst: Ankush Shukla

LinkedIn

GitHub: ankush0699 (github.com)

E-mail: ankushshukla0612@gmail.com

Q1: Top outlets by cuisine type (without using the limit and top functions).



Limit to 1000 rows

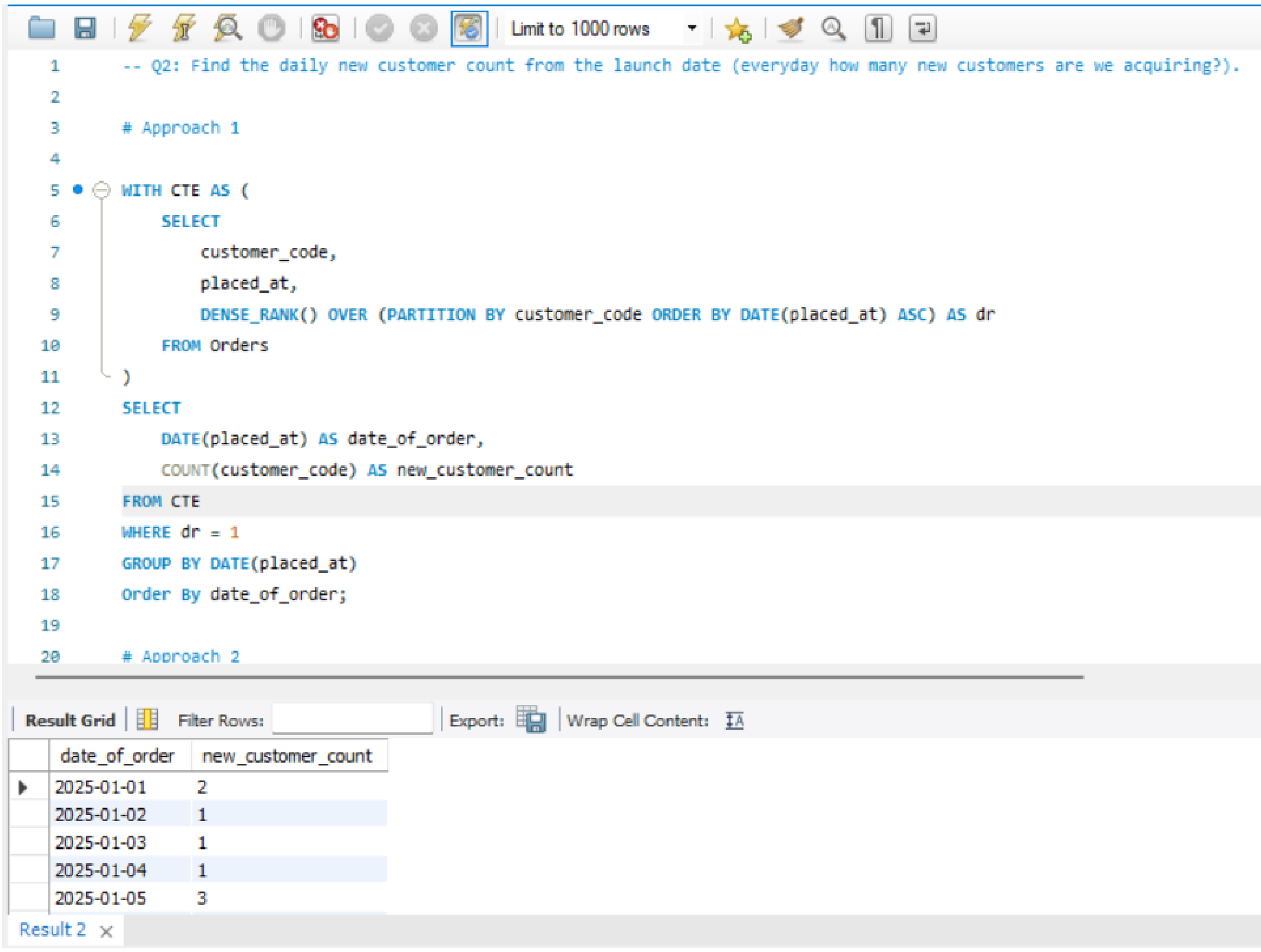
```sql
-- Q1: Top outlets by cusisine type (without using limit and top function).

# Approach 1

WITH CTE AS (
    SELECT cuisine, restaurant_id, COUNT(order_id) AS total_orders
    FROM orders
    GROUP BY cuisine, restaurant_id
)
SELECT *
FROM (
    SELECT *,
           Dense_rank() OVER (PARTITION BY cuisine ORDER BY total_orders DESC) AS rn
    FROM CTE
) a
WHERE rn =1;

# Approach 2

WITH CTE AS (
    SELECT cuisine, restaurant_id, COUNT(order_id) AS total_orders
    FROM orders
    GROUP BY cuisine, restaurant_id
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

cuisine	restaurant_id	total_orders
American	BURGER99	8
Italian	PIZZA123	10
Japanese	SUSHI456	6
Lebanese	KMKMH6787	10
Mexican	TACO789	7

Q2: Find the daily new customer count from the launch date (how many new customers are we acquiring?).

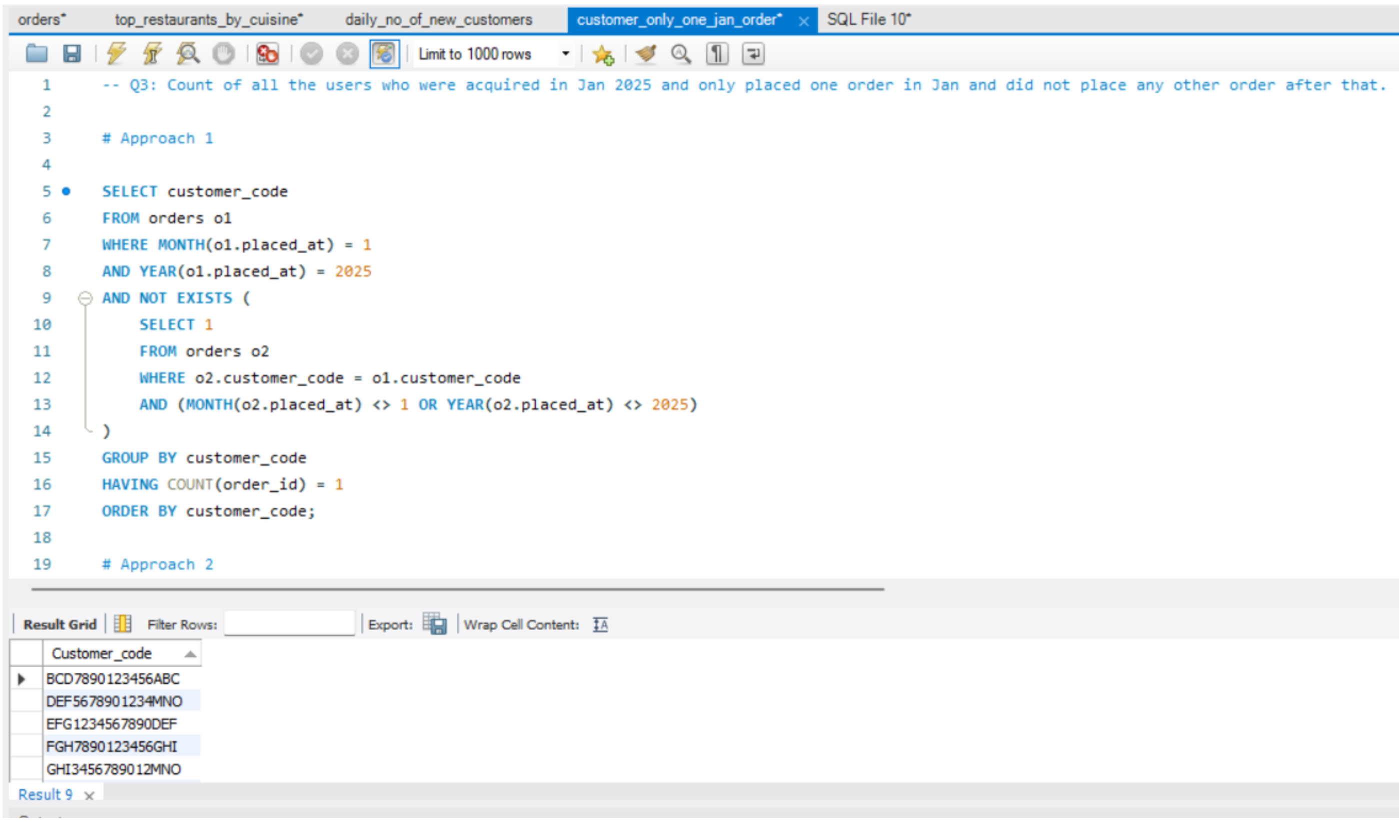
—



```sql
-- Q2: Find the daily new customer count from the launch date (everyday how many new customers are we acquiring?).

# Approach 1

WITH CTE AS (
    SELECT
        customer_code,
        placed_at,
        DENSE_RANK() OVER (PARTITION BY customer_code ORDER BY DATE(placed_at) ASC) AS dr
    FROM Orders
)
SELECT
    DATE(placed_at) AS date_of_order,
    COUNT(customer_code) AS new_customer_count
FROM CTE
WHERE dr = 1
GROUP BY DATE(placed_at)
Order By date_of_order;

# Approach 2
```

date_of_order	new_customer_count
2025-01-01	2
2025-01-02	1
2025-01-03	1
2025-01-04	1
2025-01-05	3

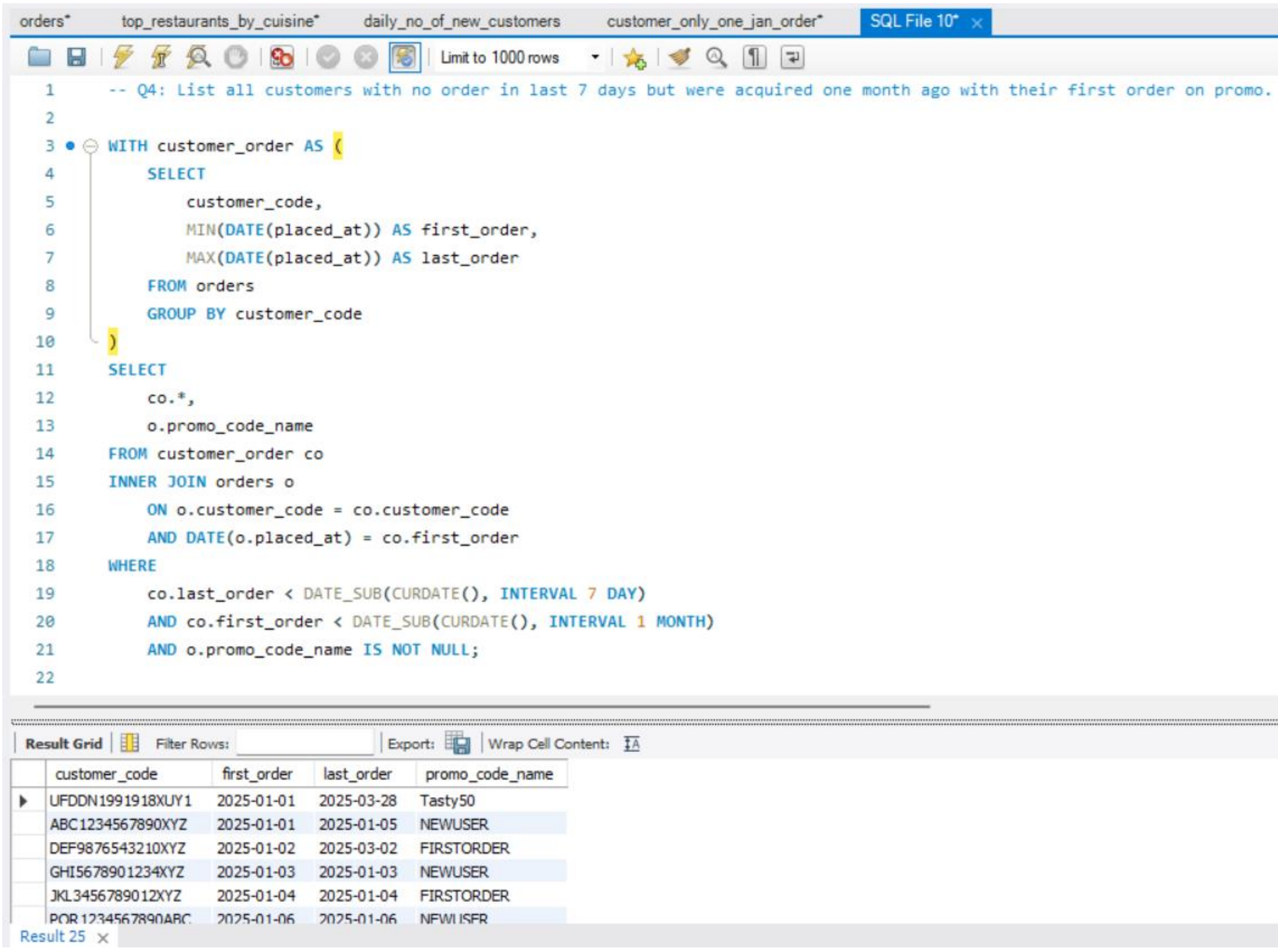
Q3: Count of all the users who were acquired in Jan 2025 and only placed one order in Jan and did not place any other order after that.



```sql
-- Q3: Count of all the users who were acquired in Jan 2025 and only placed one order in Jan and did not place any other order after that.

# Approach 1

SELECT customer_code
FROM orders o1
WHERE MONTH(o1.placed_at) = 1
AND YEAR(o1.placed_at) = 2025
AND NOT EXISTS (
    SELECT 1
    FROM orders o2
    WHERE o2.customer_code = o1.customer_code
    AND (MONTH(o2.placed_at) <> 1 OR YEAR(o2.placed_at) <> 2025)
)
GROUP BY customer_code
HAVING COUNT(order_id) = 1
ORDER BY customer_code;

# Approach 2
```

Customer_code
BCD7890123456ABC
DEF5678901234MNO
EFG1234567890DEF
FGH7890123456GHI
GHI3456789012MNO

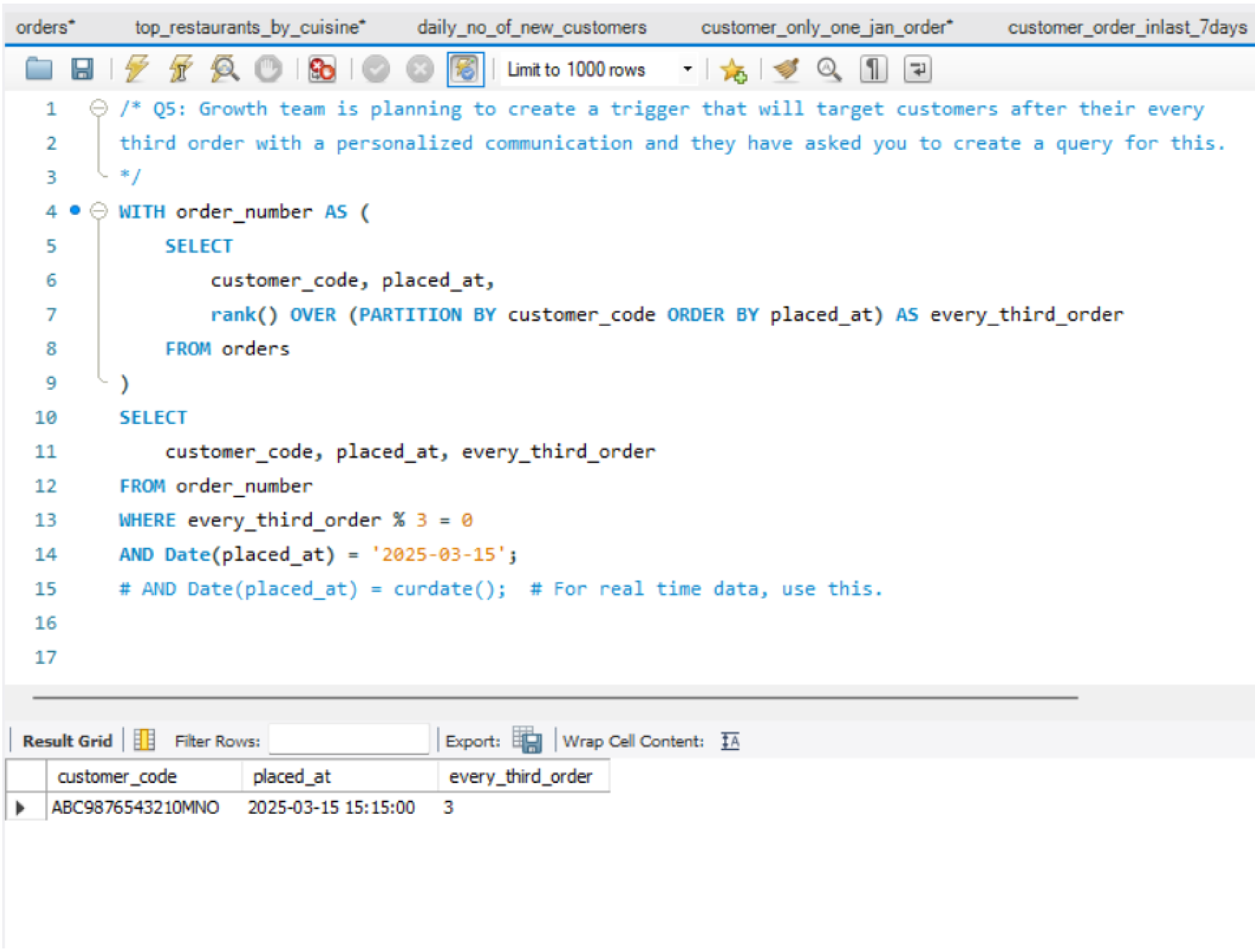
Q4: List all customers with no orders in last 7 days but were acquired one month ago with their first order on promo.



```sql
-- Q4: List all customers with no order in last 7 days but were acquired one month ago with their first order on promo.

WITH customer_order AS (
    SELECT
        customer_code,
        MIN(DATE(placed_at)) AS first_order,
        MAX(DATE(placed_at)) AS last_order
    FROM orders
    GROUP BY customer_code
)
SELECT
    co.*,
    o.promo_code_name
FROM customer_order co
INNER JOIN orders o
    ON o.customer_code = co.customer_code
    AND DATE(o.placed_at) = co.first_order
WHERE
    co.last_order < DATE_SUB(CURDATE(), INTERVAL 7 DAY)
    AND co.first_order < DATE_SUB(CURDATE(), INTERVAL 1 MONTH)
    AND o.promo_code_name IS NOT NULL;
```

customer_code	first_order	last_order	promo_code_name
UFDDN1991918XUY1	2025-01-01	2025-03-28	Tasty50
ABC1234567890XYZ	2025-01-01	2025-01-05	NEWUSER
DEF9876543210XYZ	2025-01-02	2025-03-02	FIRSTORDER
GHI5678901234XYZ	2025-01-03	2025-01-03	NEWUSER
JKL3456789012XYZ	2025-01-04	2025-01-04	FIRSTORDER
PQR1234567890ABC	2025-01-06	2025-01-06	NEWUSER

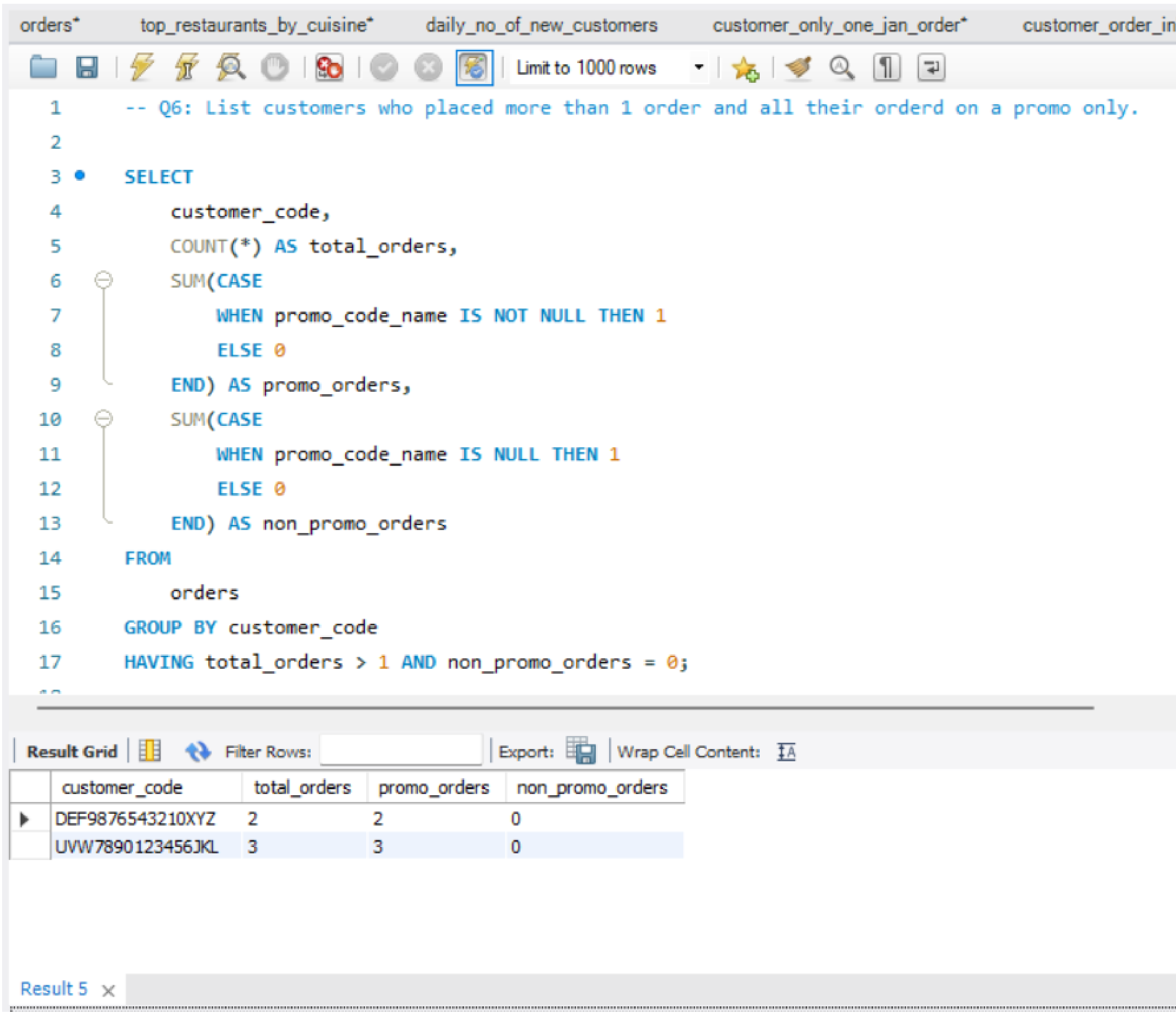
Q5: The Growth team is planning to create a trigger that will target customers after their every third order with personalized communication, and they have asked you to create a query for this.


orders*
top_restaurants_by_cuisine*
daily_no_of_new_customers
customer_only_one_jan_order*
customer_order_inlast_7days
Limit to 1000 rows
Result Grid
Filter Rows:
Export:
Wrap Cell Content:

```sql
/* Q5: Growth team is planning to create a trigger that will target customers after their every
third order with a personalized communication and they have asked you to create a query for this.
*/
WITH order_number AS (
    SELECT
        customer_code, placed_at,
        rank() OVER (PARTITION BY customer_code ORDER BY placed_at) AS every_third_order
    FROM orders
)
SELECT
    customer_code, placed_at, every_third_order
FROM order_number
WHERE every_third_order % 3 = 0
AND Date(placed_at) = '2025-03-15';
# AND Date(placed_at) = curdate();  # For real time data, use this.
```

customer_code	placed_at	every_third_order
ABC9876543210MNO	2025-03-15 15:15:00	3

Q6: List customers who placed more than 1 order and all their orders on a promo only.



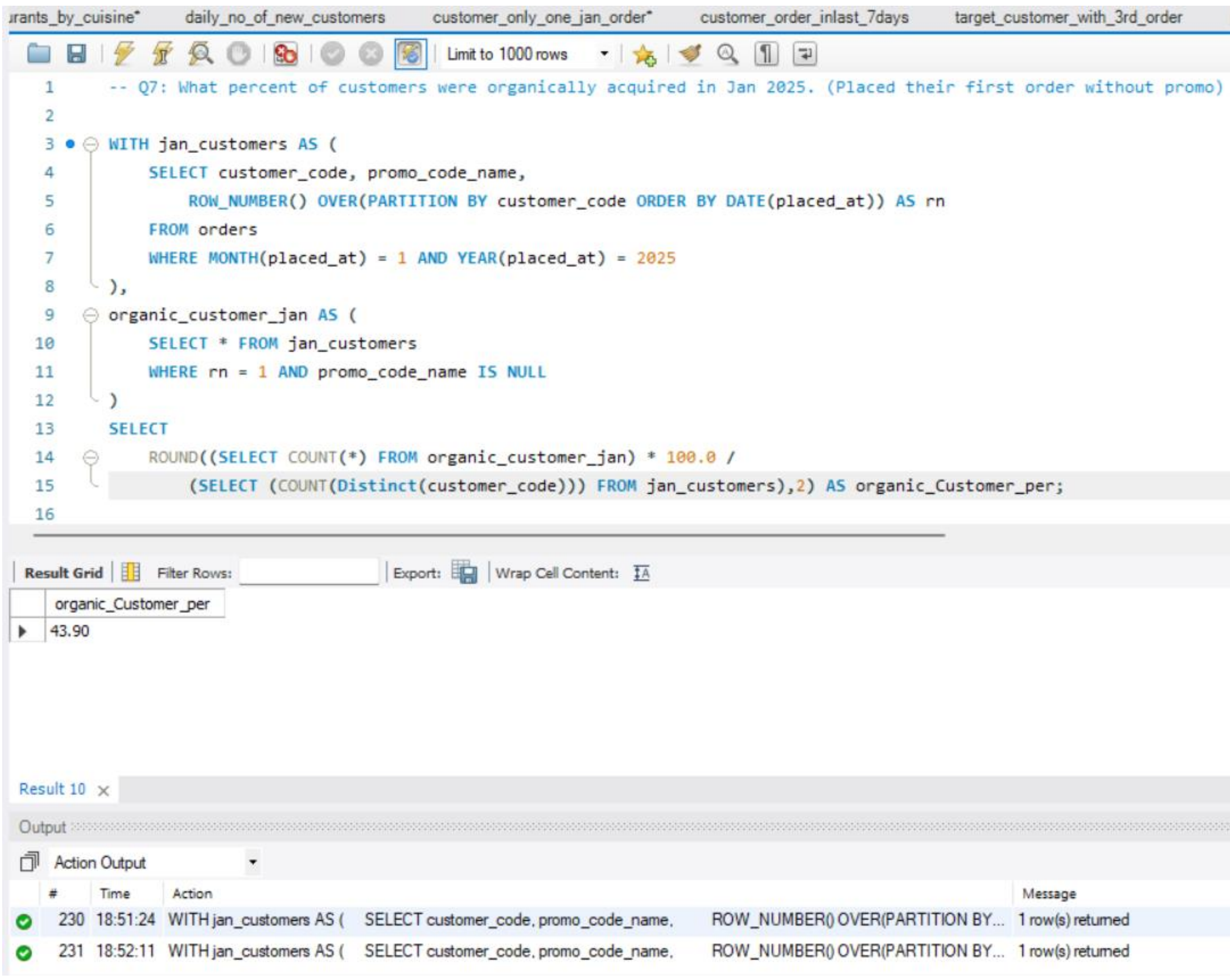
orders* top_restaurants_by_cuisine* daily_no_of_new_customers customer_only_one_jan_order* customer_order_in
Limit to 1000 rows

```
-- Q6: List customers who placed more than 1 order and all their orderd on a promo only.

SELECT
    customer_code,
    COUNT(*) AS total_orders,
    SUM(CASE
        WHEN promo_code_name IS NOT NULL THEN 1
        ELSE 0
    END) AS promo_orders,
    SUM(CASE
        WHEN promo_code_name IS NULL THEN 1
        ELSE 0
    END) AS non_promo_orders
FROM
    orders
GROUP BY customer_code
HAVING total_orders > 1 AND non_promo_orders = 0;
```

Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Result 5

customer_code	total_orders	promo_orders	non_promo_orders
DEF9876543210XYZ	2	2	0
UVW7890123456JKL	3	3	0

Q7: What percent of customers were organically acquired in Jan 2025? (Placed their first order without promo)



urants_by_cuisine* daily_no_of_new_customers customer_only_one_jan_order* customer_order_inlast_7days target_customer_with_3rd_order

Limit to 1000 rows

```sql
-- Q7: What percent of customers were organically acquired in Jan 2025. (Placed their first order without promo)

WITH jan_customers AS (
    SELECT customer_code, promo_code_name,
        ROW_NUMBER() OVER(PARTITION BY customer_code ORDER BY DATE(placed_at)) AS rn
    FROM orders
    WHERE MONTH(placed_at) = 1 AND YEAR(placed_at) = 2025
),
organic_customer_jan AS (
    SELECT * FROM jan_customers
    WHERE rn = 1 AND promo_code_name IS NULL
)
SELECT
    ROUND((SELECT COUNT(*) FROM organic_customer_jan) * 100.0 /
        (SELECT (COUNT(Distinct(customer_code))) FROM jan_customers),2) AS organic_Customer_per;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

organic_Customer_per
43.90

Result 10 ×

Output

Action Output

```
#    Time      Action                                                                                  Message
230  18:51:24  WITH jan_customers AS (  SELECT customer_code, promo_code_name,  ROW_NUMBER() OVER(PARTITION BY...  1 row(s) returned
231  18:52:11  WITH jan_customers AS (  SELECT customer_code, promo_code_name,  ROW_NUMBER() OVER(PARTITION BY...  1 row(s) returned
```