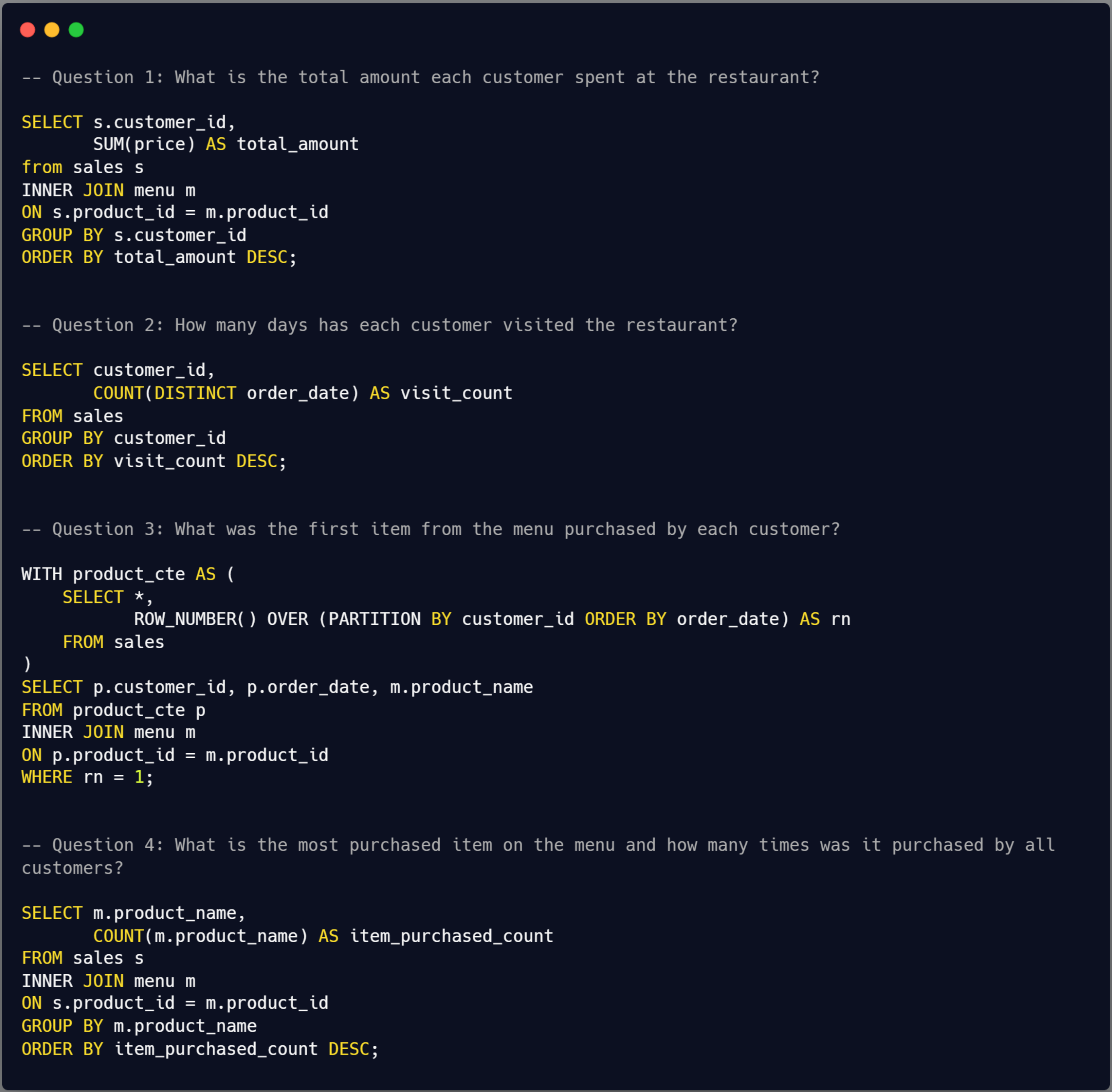
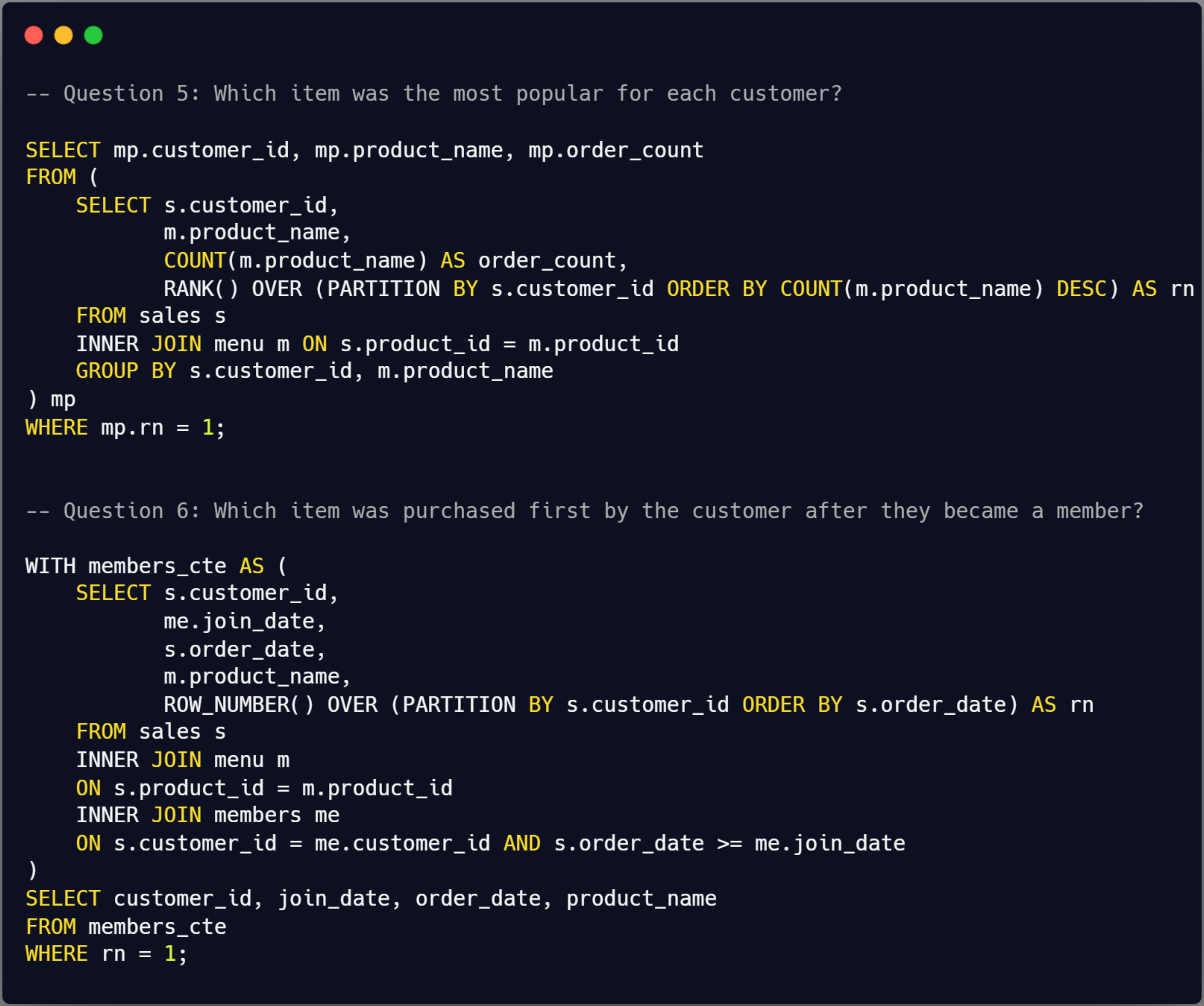
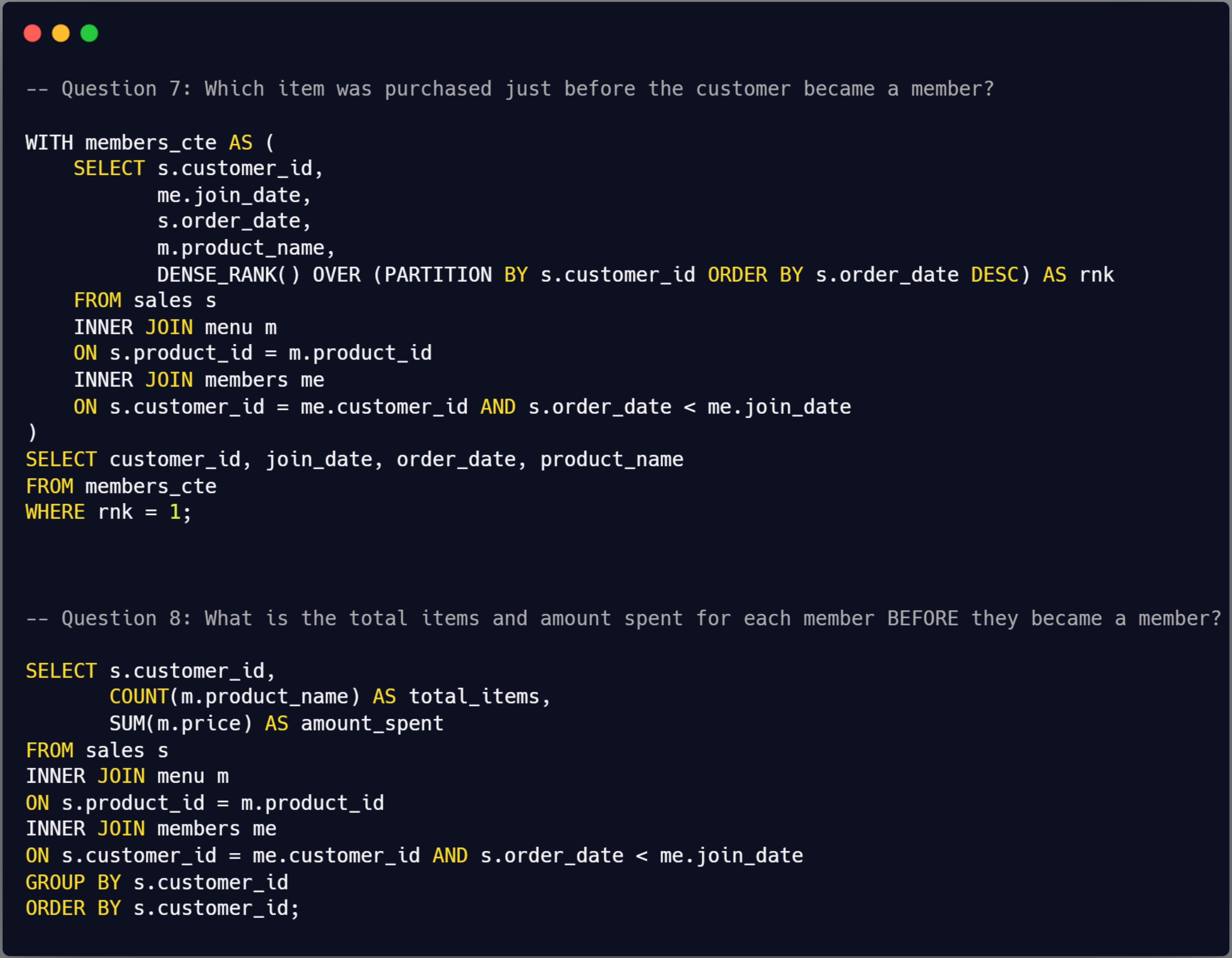


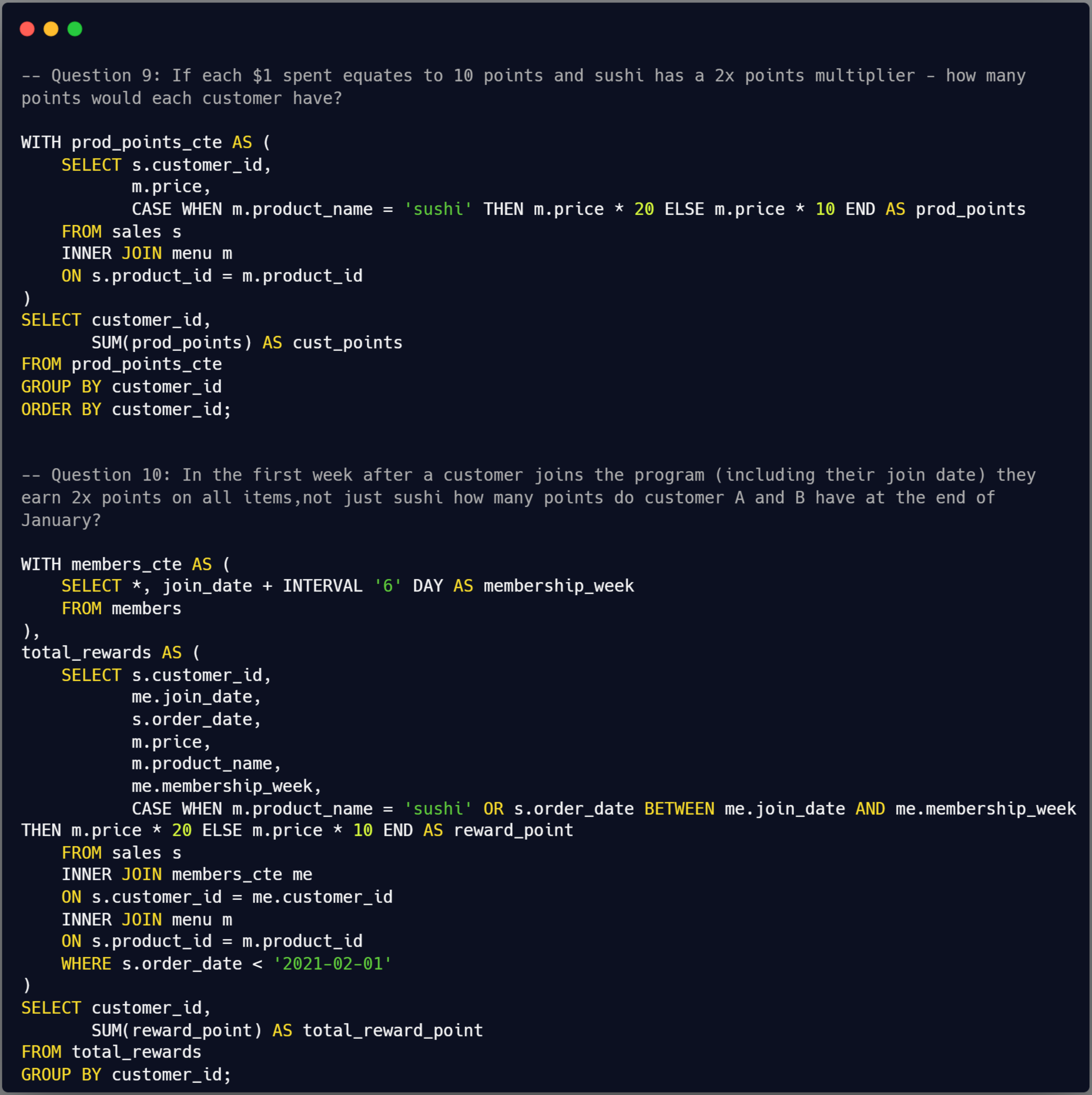
```sql
-- Question 1: What is the total amount each customer spent at the restaurant?

SELECT s.customer_id,
       SUM(price) AS total_amount
from sales s
INNER JOIN menu m
ON s.product_id = m.product_id
GROUP BY s.customer_id
ORDER BY total_amount DESC;


-- Question 2: How many days has each customer visited the restaurant?

SELECT customer_id,
       COUNT(DISTINCT order_date) AS visit_count
FROM sales
GROUP BY customer_id
ORDER BY visit_count DESC;


-- Question 3: What was the first item from the menu purchased by each customer?

WITH product_cte AS (
    SELECT *,
           ROW_NUMBER() OVER (PARTITION BY customer_id ORDER BY order_date) AS rn
    FROM sales
)
SELECT p.customer_id, p.order_date, m.product_name
FROM product_cte p
INNER JOIN menu m
ON p.product_id = m.product_id
WHERE rn = 1;


-- Question 4: What is the most purchased item on the menu and how many times was it purchased by all
customers?

SELECT m.product_name,
       COUNT(m.product_name) AS item_purchased_count
FROM sales s
INNER JOIN menu m
ON s.product_id = m.product_id
GROUP BY m.product_name
ORDER BY item_purchased_count DESC;
```



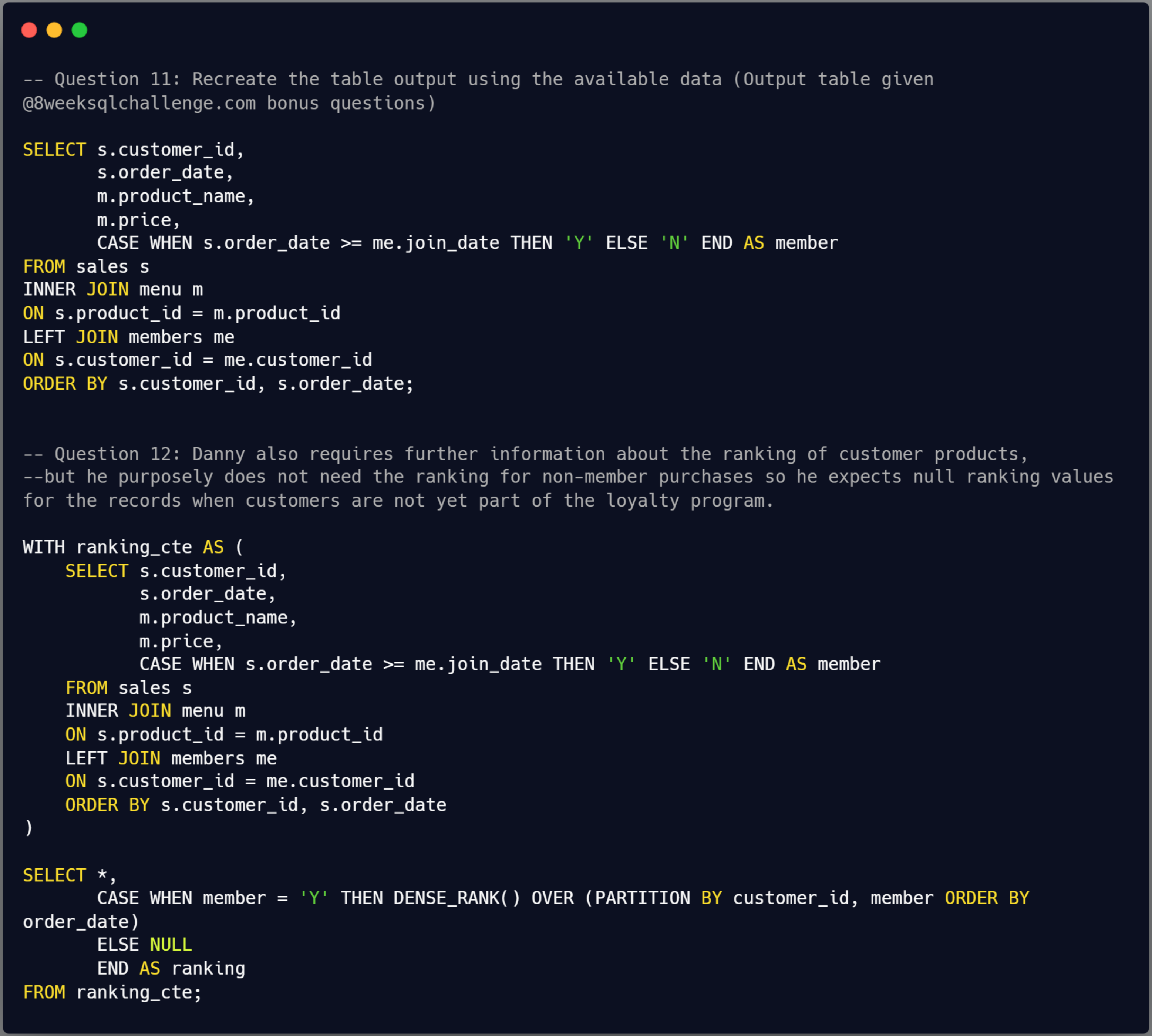
```sql
-- Question 5: Which item was the most popular for each customer?

SELECT mp.customer_id, mp.product_name, mp.order_count
FROM (
    SELECT s.customer_id,
           m.product_name,
           COUNT(m.product_name) AS order_count,
           RANK() OVER (PARTITION BY s.customer_id ORDER BY COUNT(m.product_name) DESC) AS rn
    FROM sales s
    INNER JOIN menu m ON s.product_id = m.product_id
    GROUP BY s.customer_id, m.product_name
) mp
WHERE mp.rn = 1;


-- Question 6: Which item was purchased first by the customer after they became a member?

WITH members_cte AS (
    SELECT s.customer_id,
           me.join_date,
           s.order_date,
           m.product_name,
           ROW_NUMBER() OVER (PARTITION BY s.customer_id ORDER BY s.order_date) AS rn
    FROM sales s
    INNER JOIN menu m
    ON s.product_id = m.product_id
    INNER JOIN members me
    ON s.customer_id = me.customer_id AND s.order_date >= me.join_date
)
SELECT customer_id, join_date, order_date, product_name
FROM members_cte
WHERE rn = 1;
```



```sql
-- Question 7: Which item was purchased just before the customer became a member?

WITH members_cte AS (
    SELECT s.customer_id,
           me.join_date,
           s.order_date,
           m.product_name,
           DENSE_RANK() OVER (PARTITION BY s.customer_id ORDER BY s.order_date DESC) AS rnk
    FROM sales s
    INNER JOIN menu m
    ON s.product_id = m.product_id
    INNER JOIN members me
    ON s.customer_id = me.customer_id AND s.order_date < me.join_date
)
SELECT customer_id, join_date, order_date, product_name
FROM members_cte
WHERE rnk = 1;



-- Question 8: What is the total items and amount spent for each member BEFORE they became a member?

SELECT s.customer_id,
       COUNT(m.product_name) AS total_items,
       SUM(m.price) AS amount_spent
FROM sales s
INNER JOIN menu m
ON s.product_id = m.product_id
INNER JOIN members me
ON s.customer_id = me.customer_id AND s.order_date < me.join_date
GROUP BY s.customer_id
ORDER BY s.customer_id;
```



```
-- Question 9: If each $1 spent equates to 10 points and sushi has a 2x points multiplier - how many
points would each customer have?

WITH prod_points_cte AS (
    SELECT s.customer_id,
           m.price,
           CASE WHEN m.product_name = 'sushi' THEN m.price * 20 ELSE m.price * 10 END AS prod_points
    FROM sales s
    INNER JOIN menu m
    ON s.product_id = m.product_id
)
SELECT customer_id,
       SUM(prod_points) AS cust_points
FROM prod_points_cte
GROUP BY customer_id
ORDER BY customer_id;


-- Question 10: In the first week after a customer joins the program (including their join date) they
earn 2x points on all items,not just sushi how many points do customer A and B have at the end of
January?

WITH members_cte AS (
    SELECT *, join_date + INTERVAL '6' DAY AS membership_week
    FROM members
),
total_rewards AS (
    SELECT s.customer_id,
           me.join_date,
           s.order_date,
           m.price,
           m.product_name,
           me.membership_week,
           CASE WHEN m.product_name = 'sushi' OR s.order_date BETWEEN me.join_date AND me.membership_week
THEN m.price * 20 ELSE m.price * 10 END AS reward_point
    FROM sales s
    INNER JOIN members_cte me
    ON s.customer_id = me.customer_id
    INNER JOIN menu m
    ON s.product_id = m.product_id
    WHERE s.order_date < '2021-02-01'
)
SELECT customer_id,
       SUM(reward_point) AS total_reward_point
FROM total_rewards
GROUP BY customer_id;
```



```
-- Question 11: Recreate the table output using the available data (Output table given
@8weeksqlchallenge.com bonus questions)

SELECT s.customer_id,
       s.order_date,
       m.product_name,
       m.price,
       CASE WHEN s.order_date >= me.join_date THEN 'Y' ELSE 'N' END AS member
FROM sales s
INNER JOIN menu m
ON s.product_id = m.product_id
LEFT JOIN members me
ON s.customer_id = me.customer_id
ORDER BY s.customer_id, s.order_date;


-- Question 12: Danny also requires further information about the ranking of customer products,
--but he purposely does not need the ranking for non-member purchases so he expects null ranking values
for the records when customers are not yet part of the loyalty program.

WITH ranking_cte AS (
    SELECT s.customer_id,
           s.order_date,
           m.product_name,
           m.price,
           CASE WHEN s.order_date >= me.join_date THEN 'Y' ELSE 'N' END AS member
    FROM sales s
    INNER JOIN menu m
    ON s.product_id = m.product_id
    LEFT JOIN members me
    ON s.customer_id = me.customer_id
    ORDER BY s.customer_id, s.order_date
)

SELECT *,
       CASE WHEN member = 'Y' THEN DENSE_RANK() OVER (PARTITION BY customer_id, member ORDER BY
order_date)
       ELSE NULL
       END AS ranking
FROM ranking_cte;
```