8WEEKSQLCHALLENGE.COM

CASE STUDY #1

THE TASTE OF SUCCESS

DATAWITHDANNY.COM

SQL

Data Analysis Project

Created By – Mandeep Singh

Context of the Work

Danny’s Diner, a restaurant specializing in sushi, curry, and ramen, opened in early 2021 with a passion for authentic Japanese cuisine. As a small business in the competitive restaurant industry, Danny is dedicated to ensuring the diner’s sustainability and profitability.

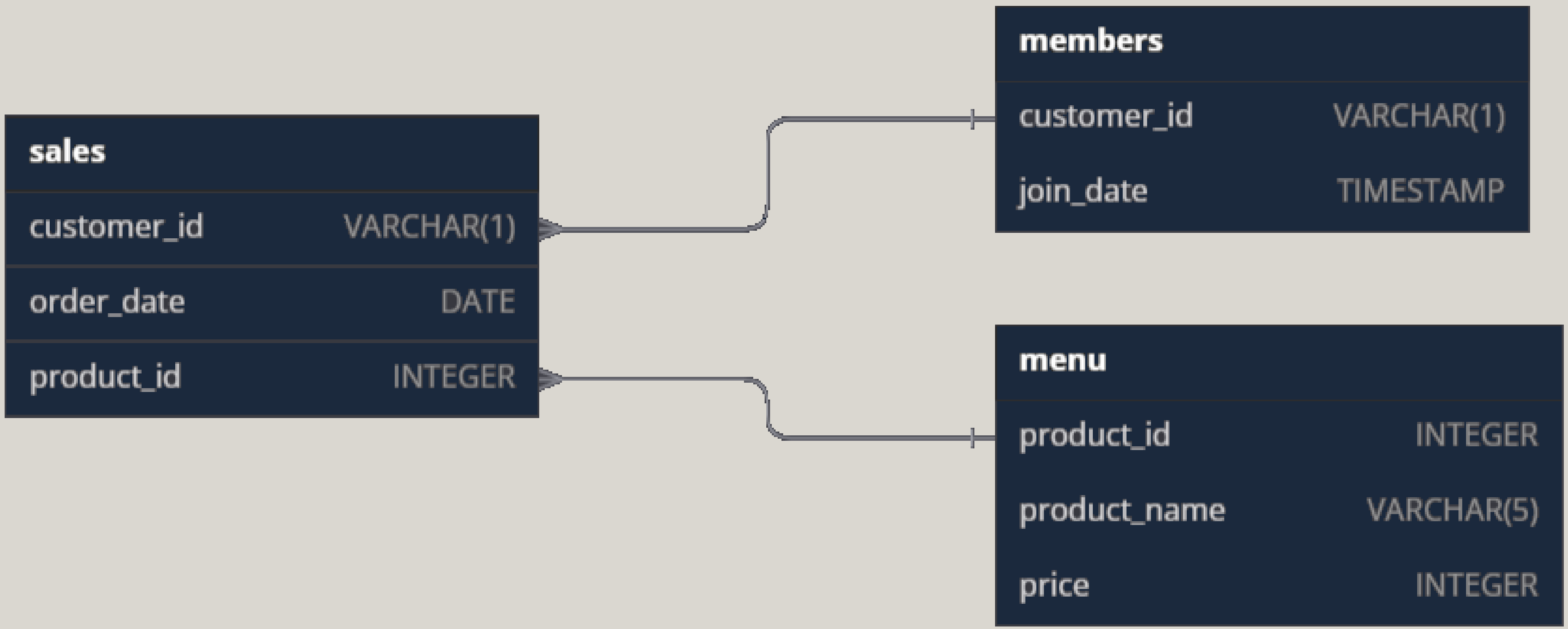
Even though Danny is collecting operational data during the first months of operation, he doesn't have deep knowledge about the behavior of customers. What he must learn includes visiting patterns, spending habits, and favorite menu items. These behaviors are critical in helping him optimize the customer experience and work out effective business strategies.

To solve this challenge, Danny wants to analyze collected data to find answers to questions that include:

How often do customers visit the restaurant?
How much do they spend usually per visit?
Which menu items are most popular among customers?

With this knowledge of how his customers behave, he hopes to have an even stronger rapport with clients, enhance the dining experience, and evaluate potential improvements to the customer loyalty program. This project will provide actionable insights that will support Danny’s Diner in achieving growth and long-term success.

Entity Relationship Diagram



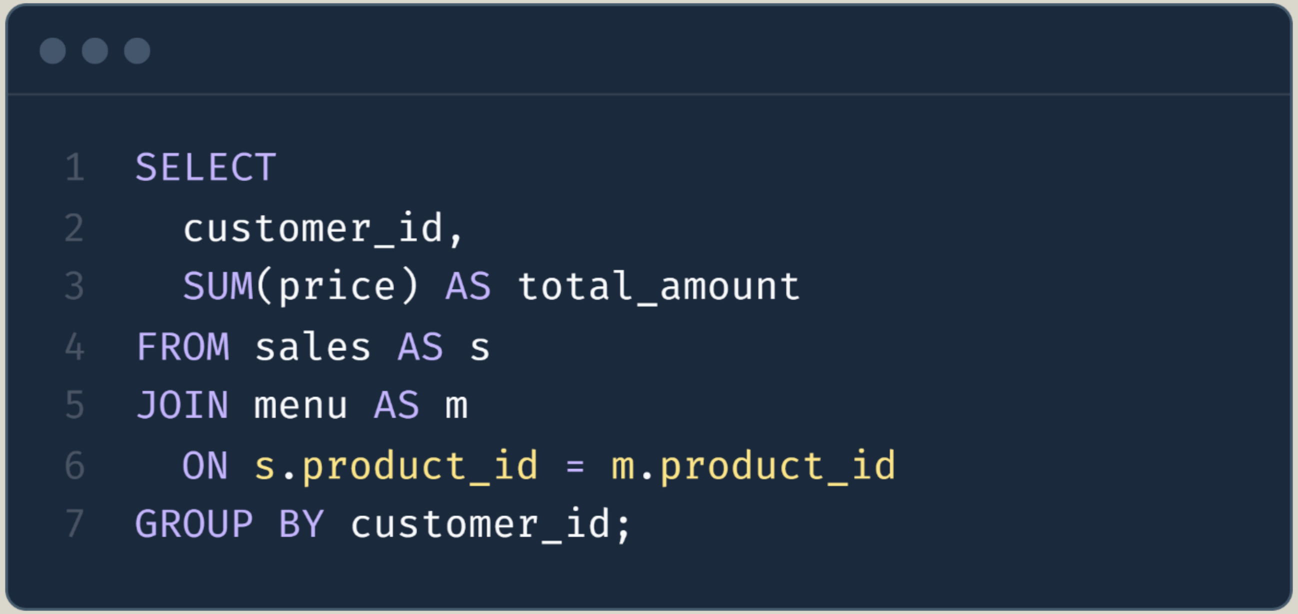
sales	
customer_id	VARCHAR(1)
order_date	DATE
product_id	INTEGER

members	
customer_id	VARCHAR(1)
join_date	TIMESTAMP

menu	
product_id	INTEGER
product_name	VARCHAR(5)
price	INTEGER

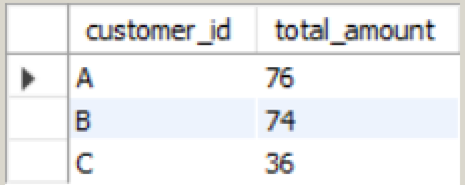
What is the total amount each customer spent at the restaurant?

Query :



```sql
SELECT
  customer_id,
  SUM(price) AS total_amount
FROM sales AS s
JOIN menu AS m
  ON s.product_id = m.product_id
GROUP BY customer_id;
```

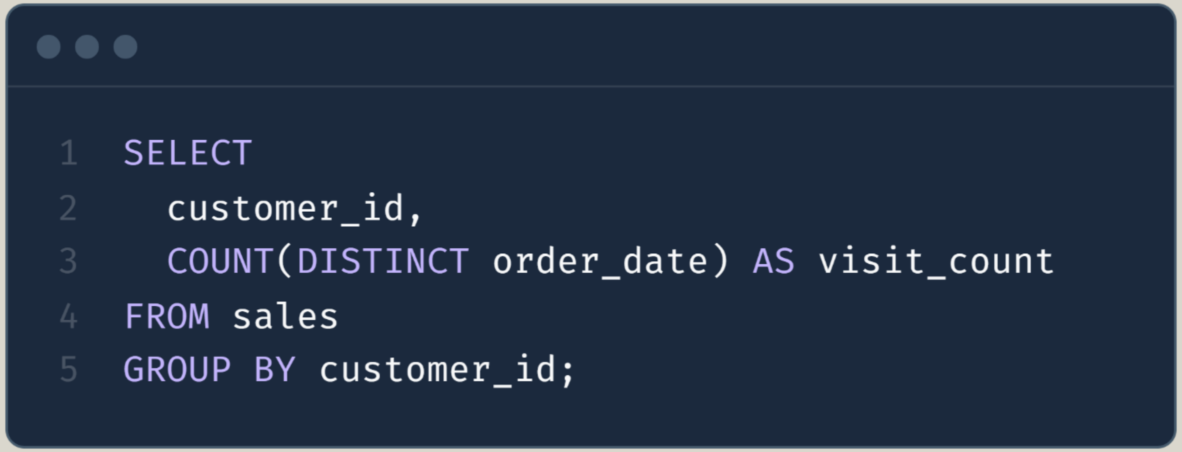
Output :



	customer_id	total_amount
▶	A	76
	B	74
	C	36

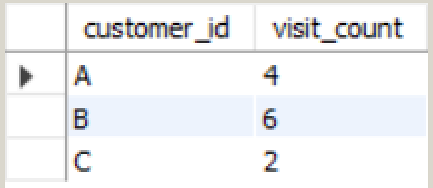
How many days has each customer visited the restaurant?

Query :



```sql
SELECT
  customer_id,
  COUNT(DISTINCT order_date) AS visit_count
FROM sales
GROUP BY customer_id;
```

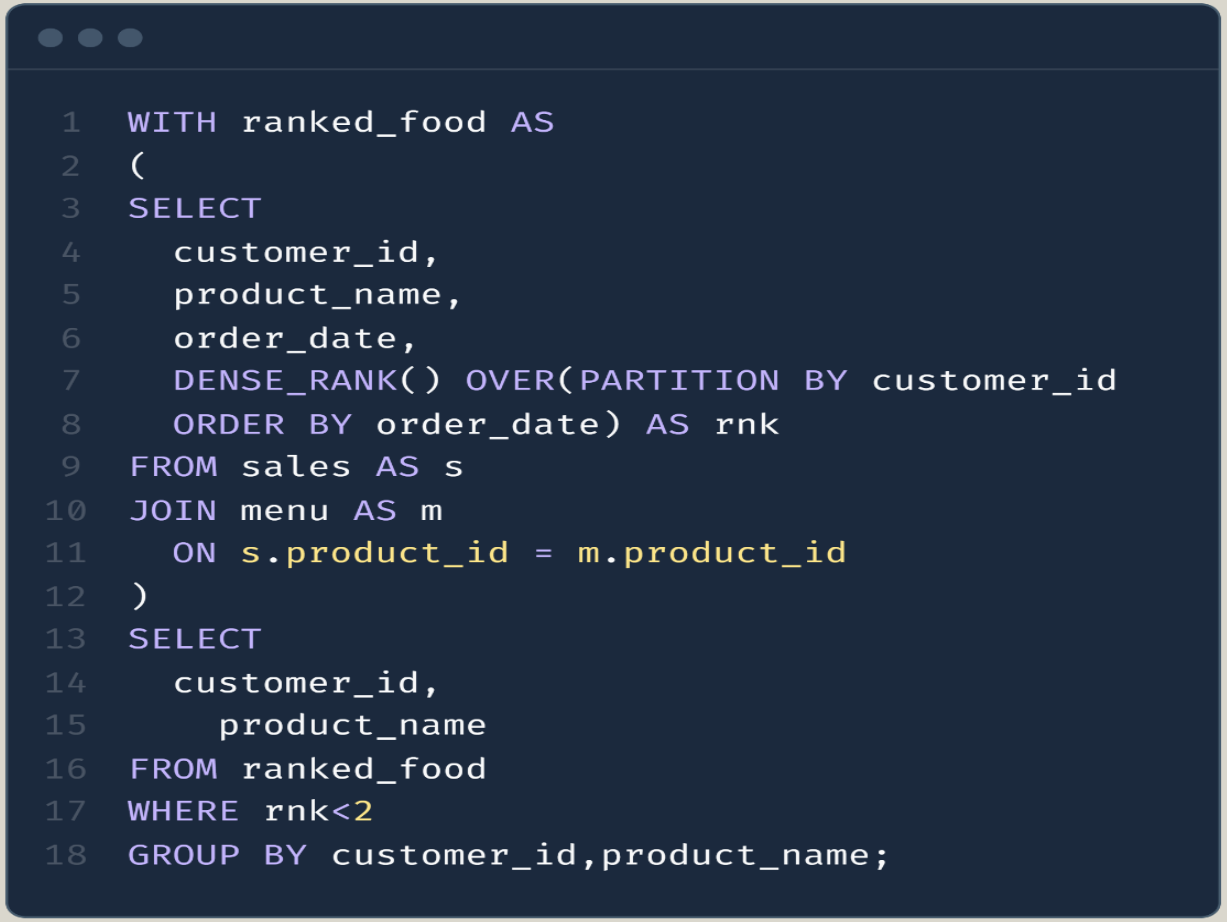
Output :



customer_id	visit_count
A	4
B	6
C	2

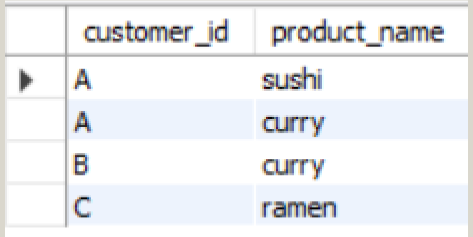
What was the first item from the menu purchased by each customer?

Query :



```sql
WITH ranked_food AS
(
SELECT
  customer_id,
  product_name,
  order_date,
  DENSE_RANK() OVER(PARTITION BY customer_id
  ORDER BY order_date) AS rnk
FROM sales AS s
JOIN menu AS m
  ON s.product_id = m.product_id
)
SELECT
  customer_id,
    product_name
FROM ranked_food
WHERE rnk<2
GROUP BY customer_id,product_name;
```

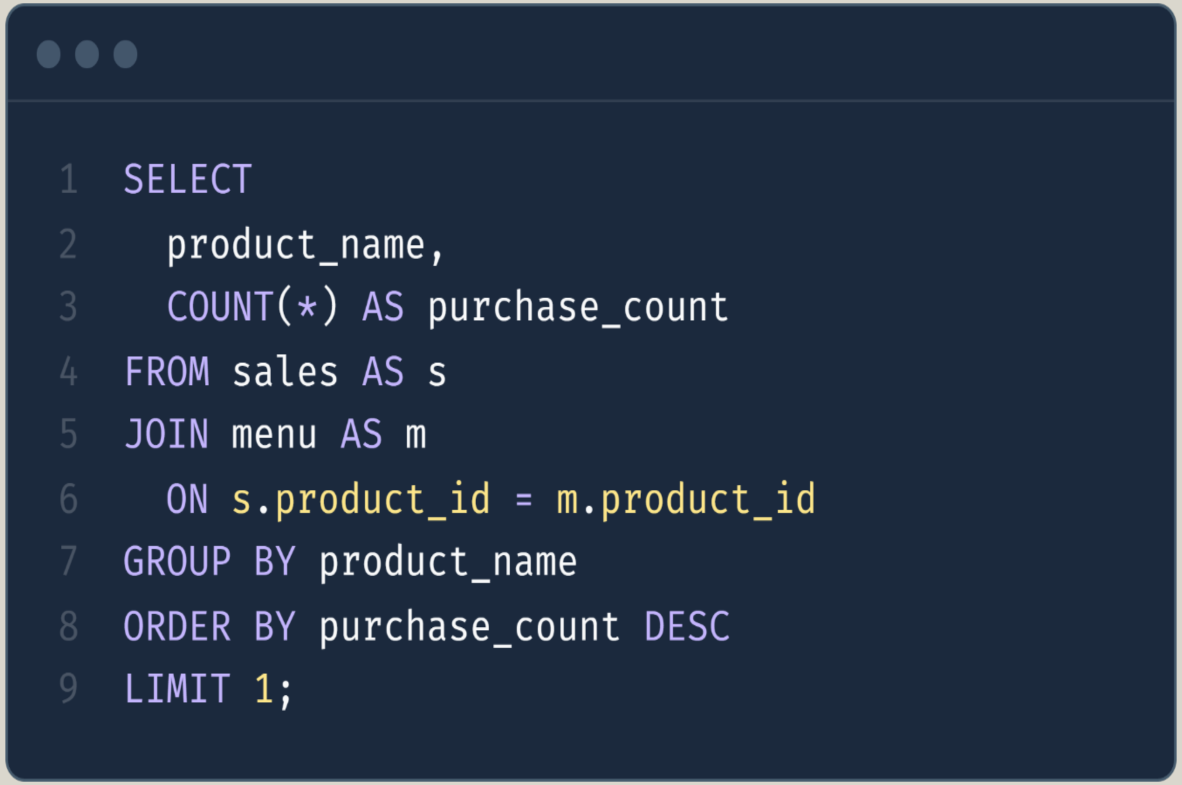
Output :



customer_id	product_name
A	sushi
A	curry
B	curry
C	ramen

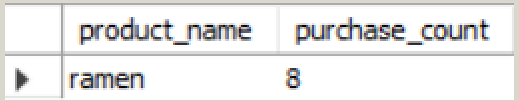
What is the most purchased item on the menu and how many times was it purchased by all customers?

Query :



```sql
SELECT
  product_name,
  COUNT(*) AS purchase_count
FROM sales AS s
JOIN menu AS m
  ON s.product_id = m.product_id
GROUP BY product_name
ORDER BY purchase_count DESC
LIMIT 1;
```

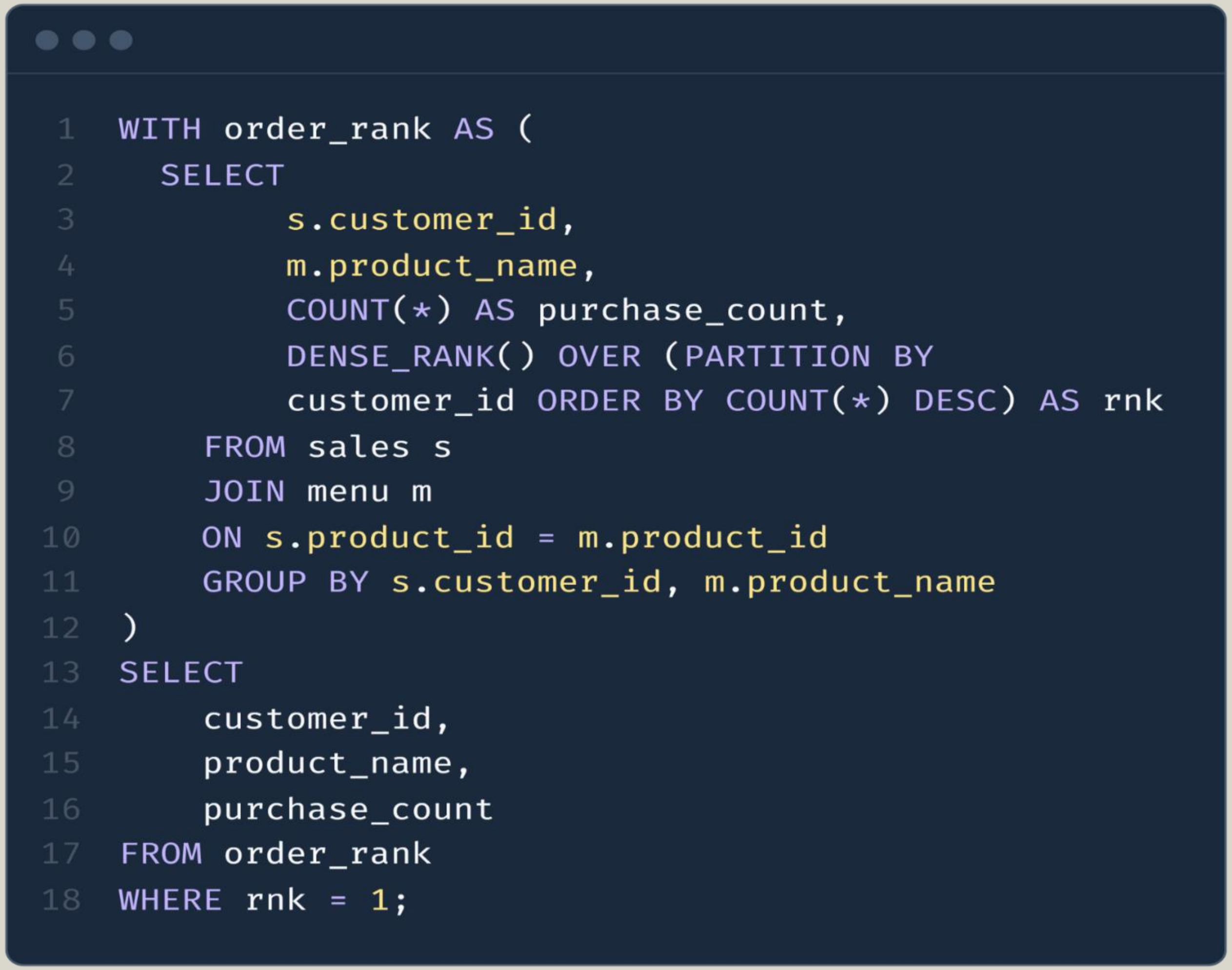
Output :



product_name	purchase_count
ramen	8

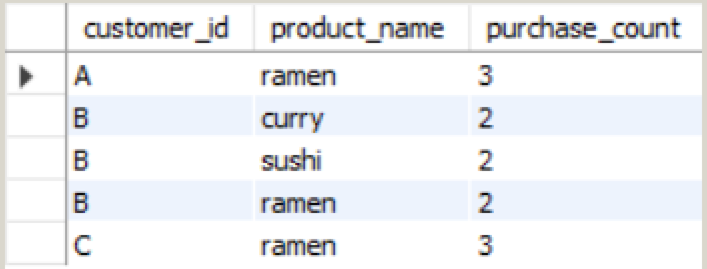
Which item was the most popular for each customer?

Query :



```sql
WITH order_rank AS (
  SELECT
        s.customer_id,
        m.product_name,
        COUNT(*) AS purchase_count,
        DENSE_RANK() OVER (PARTITION BY
        customer_id ORDER BY COUNT(*) DESC) AS rnk
    FROM sales s
    JOIN menu m
    ON s.product_id = m.product_id
    GROUP BY s.customer_id, m.product_name
)
SELECT
    customer_id,
    product_name,
    purchase_count
FROM order_rank
WHERE rnk = 1;
```

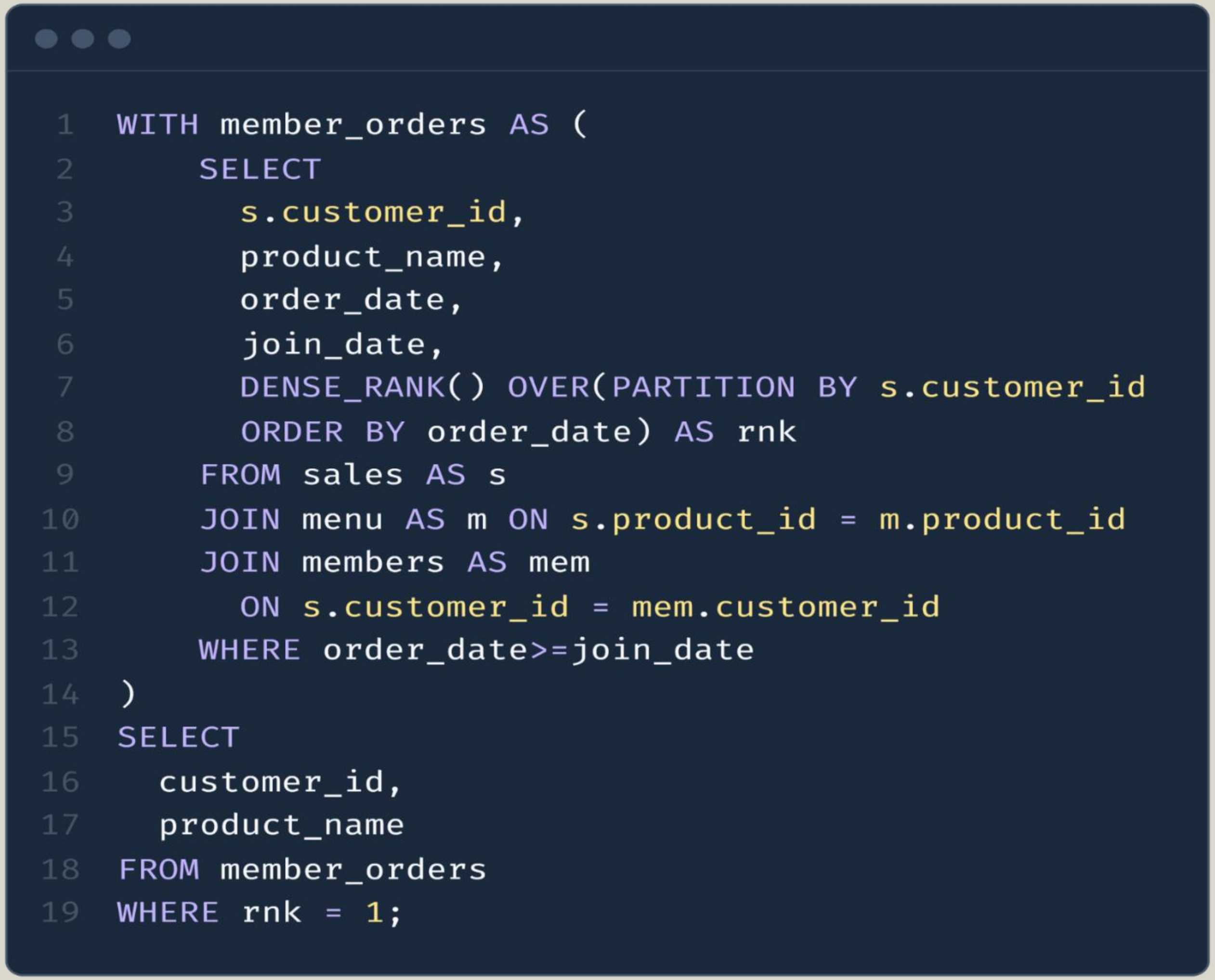
Output :



	customer_id	product_name	purchase_count
▶	A	ramen	3
	B	curry	2
	B	sushi	2
	B	ramen	2
	C	ramen	3

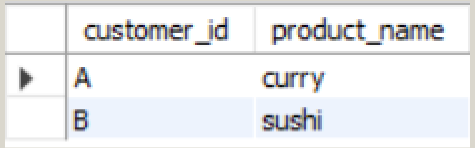
Which item was purchased first by the customer after they became a member?

Query :



```sql
WITH member_orders AS (
    SELECT
      s.customer_id,
      product_name,
      order_date,
      join_date,
      DENSE_RANK() OVER(PARTITION BY s.customer_id
      ORDER BY order_date) AS rnk
    FROM sales AS s
    JOIN menu AS m ON s.product_id = m.product_id
    JOIN members AS mem
      ON s.customer_id = mem.customer_id
    WHERE order_date>=join_date
)
SELECT
  customer_id,
  product_name
FROM member_orders
WHERE rnk = 1;
```

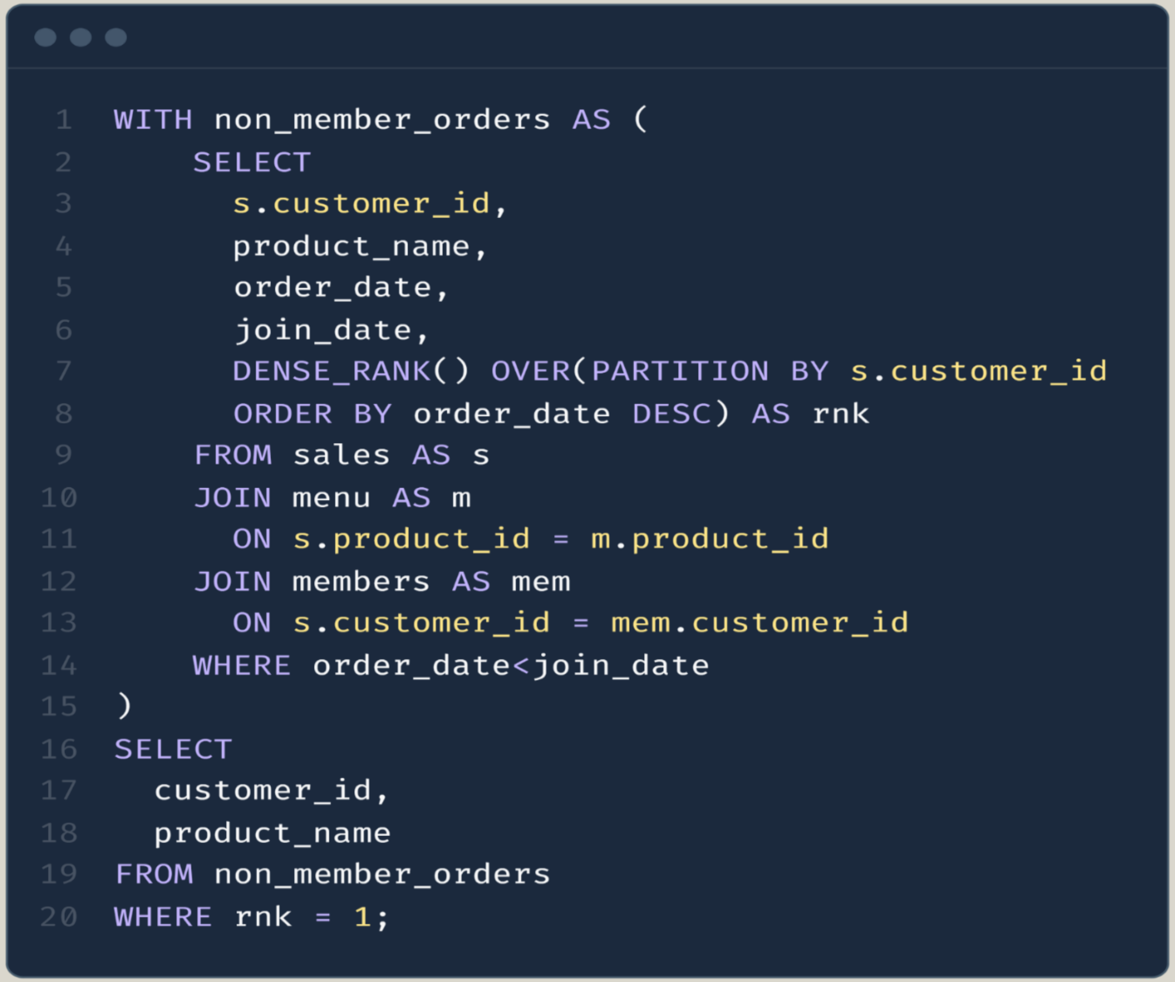
Output :



customer_id	product_name
A	curry
B	sushi

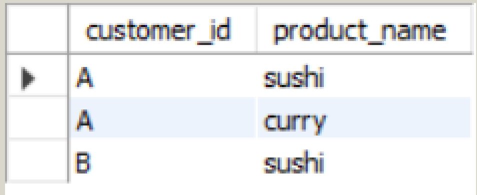
Which item was purchased just before the customer became a member?

Query :



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

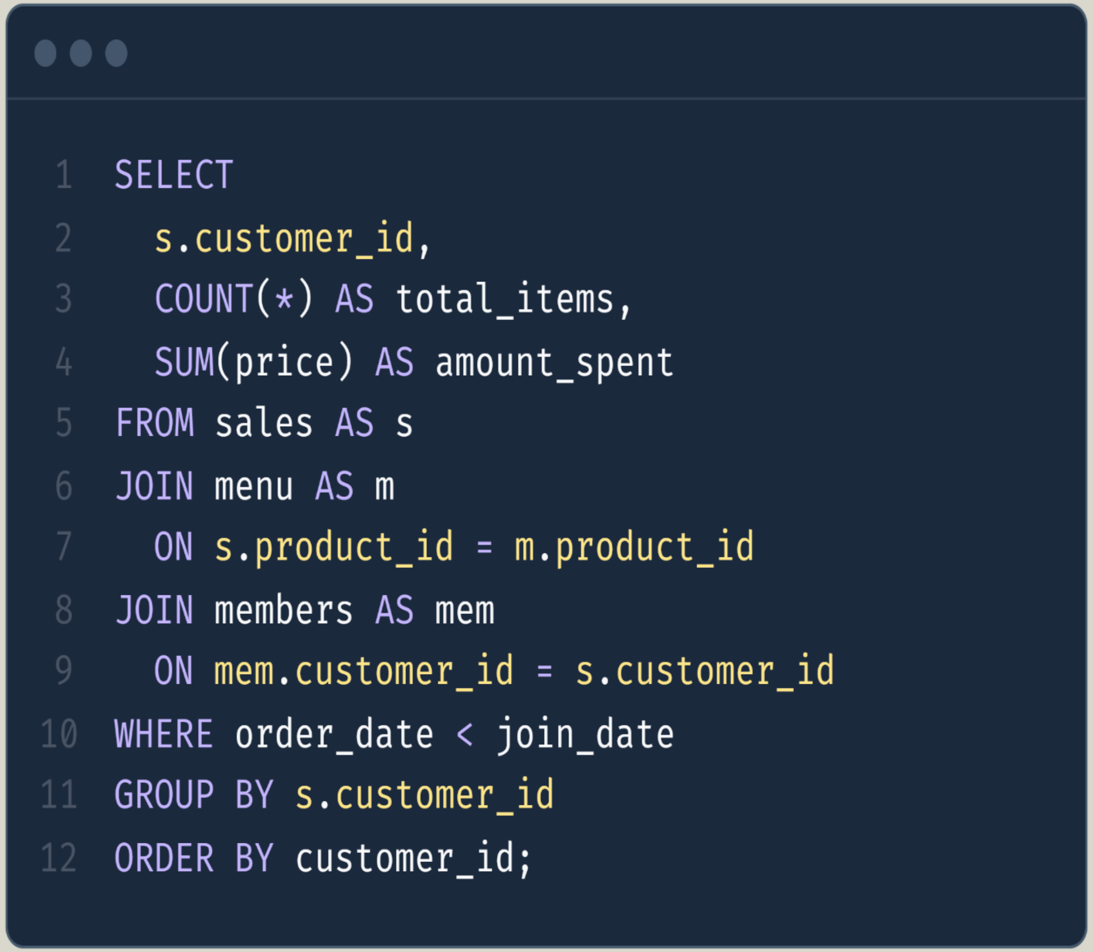
Output :



customer_id	product_name
A	sushi
A	curry
B	sushi

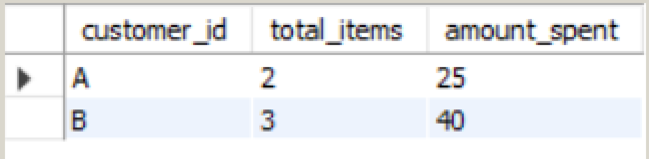
What is the total items and amount spent for each member before they became a member?

Query :



```sql
SELECT
  s.customer_id,
  COUNT(*) AS total_items,
  SUM(price) AS amount_spent
FROM sales AS s
JOIN menu AS m
  ON s.product_id = m.product_id
JOIN members AS mem
  ON mem.customer_id = s.customer_id
WHERE order_date < join_date
GROUP BY s.customer_id
ORDER BY customer_id;
```

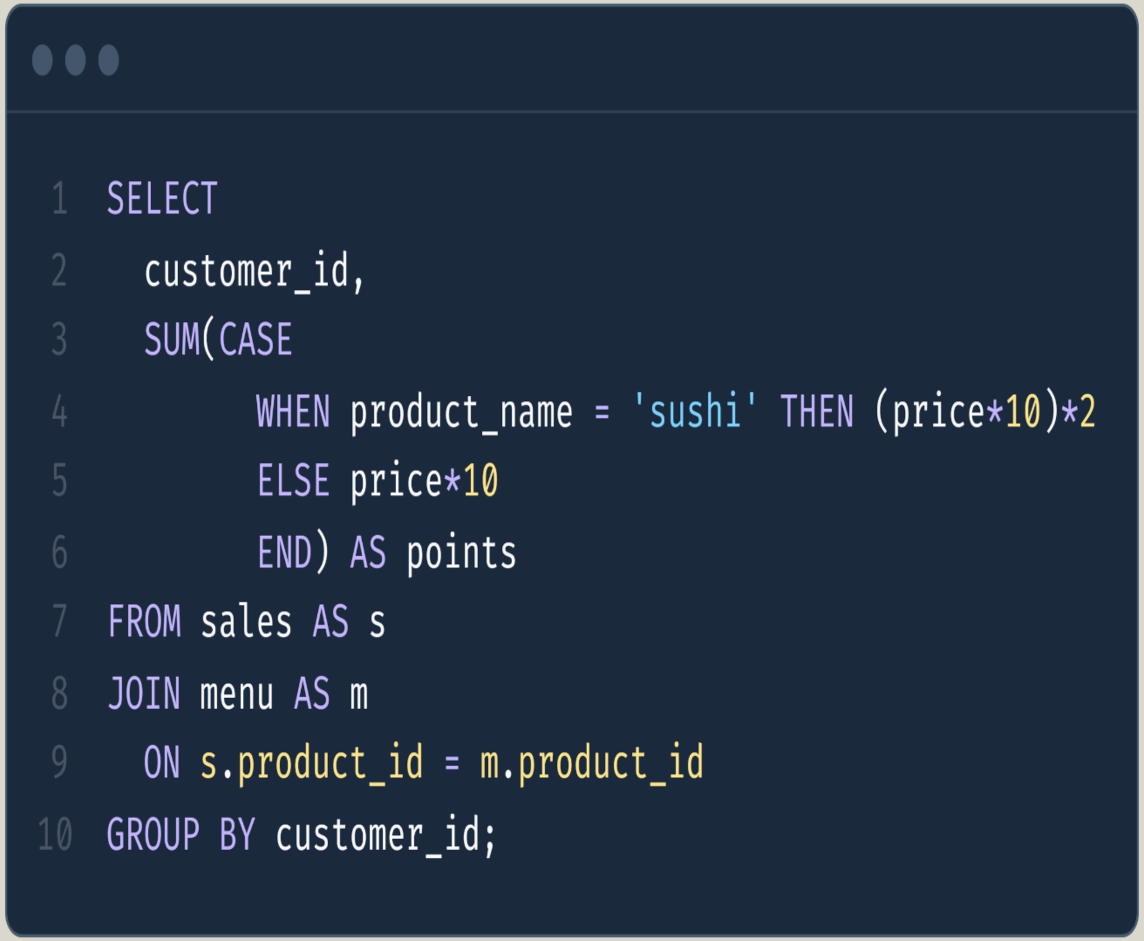
Output :



customer_id	total_items	amount_spent
A	2	25
B	3	40

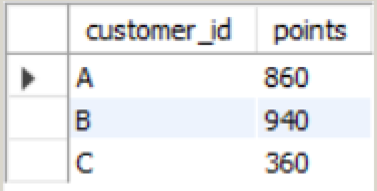
If each $1 spent equates to 10 points and sushi has a 2x points multiplier - how many points would each customer have?

Query :



```sql
SELECT
  customer_id,
  SUM(CASE
        WHEN product_name = 'sushi' THEN (price*10)*2
        ELSE price*10
        END) AS points
FROM sales AS s
JOIN menu AS m
  ON s.product_id = m.product_id
GROUP BY customer_id;
```

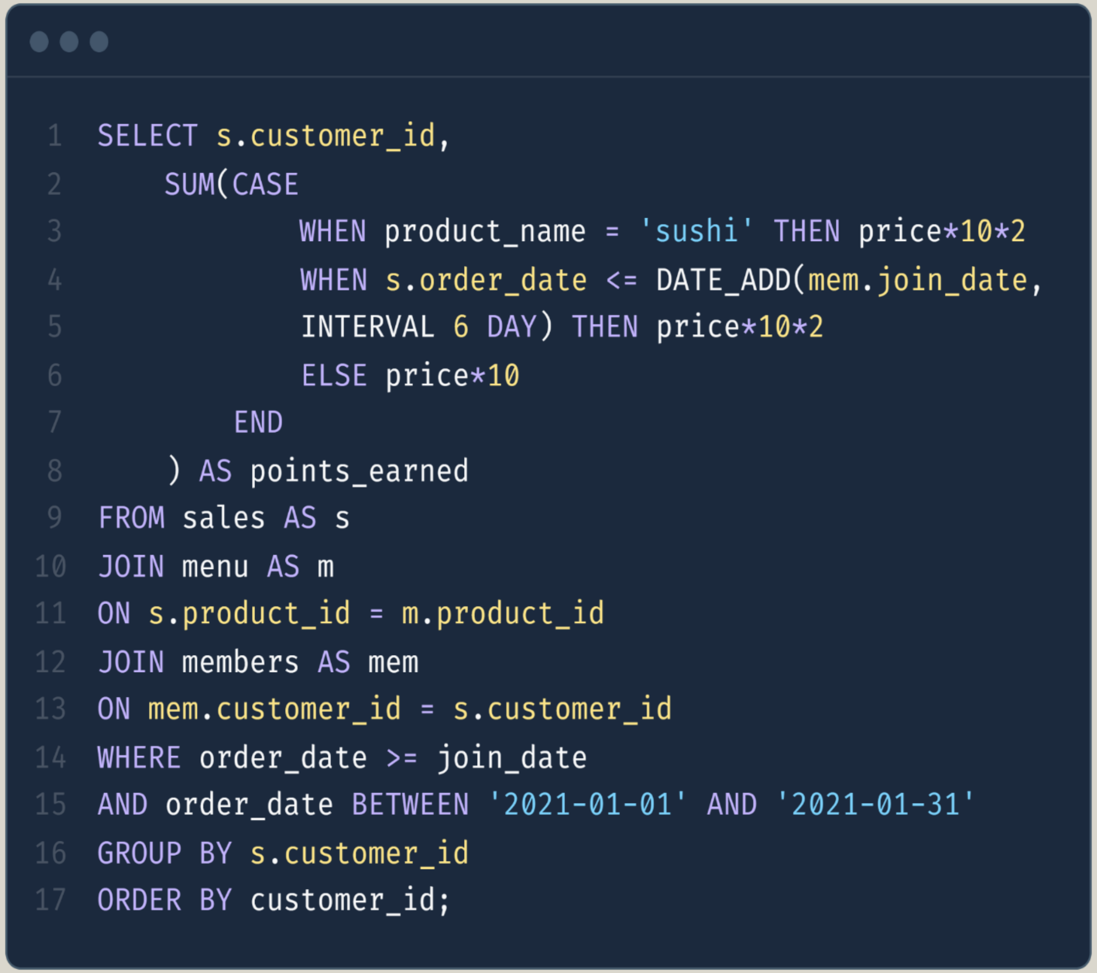
Output :



customer_id	points
A	860
B	940
C	360

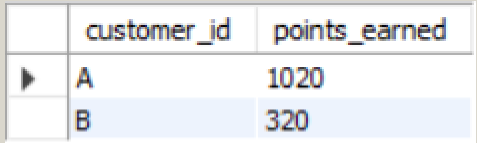
In the first week after a customer joins the program (including their join date) they earn 2x points on all items, not just sushi - how many points do customer A and B have at the end of January?

Query :



```sql
SELECT s.customer_id,
    SUM(CASE
            WHEN product_name = 'sushi' THEN price*10*2
            WHEN s.order_date <= DATE_ADD(mem.join_date,
            INTERVAL 6 DAY) THEN price*10*2
            ELSE price*10
        END
    ) AS points_earned
FROM sales AS s
JOIN menu AS m
ON s.product_id = m.product_id
JOIN members AS mem
ON mem.customer_id = s.customer_id
WHERE order_date >= join_date
AND order_date BETWEEN '2021-01-01' AND '2021-01-31'
GROUP BY s.customer_id
ORDER BY customer_id;
```

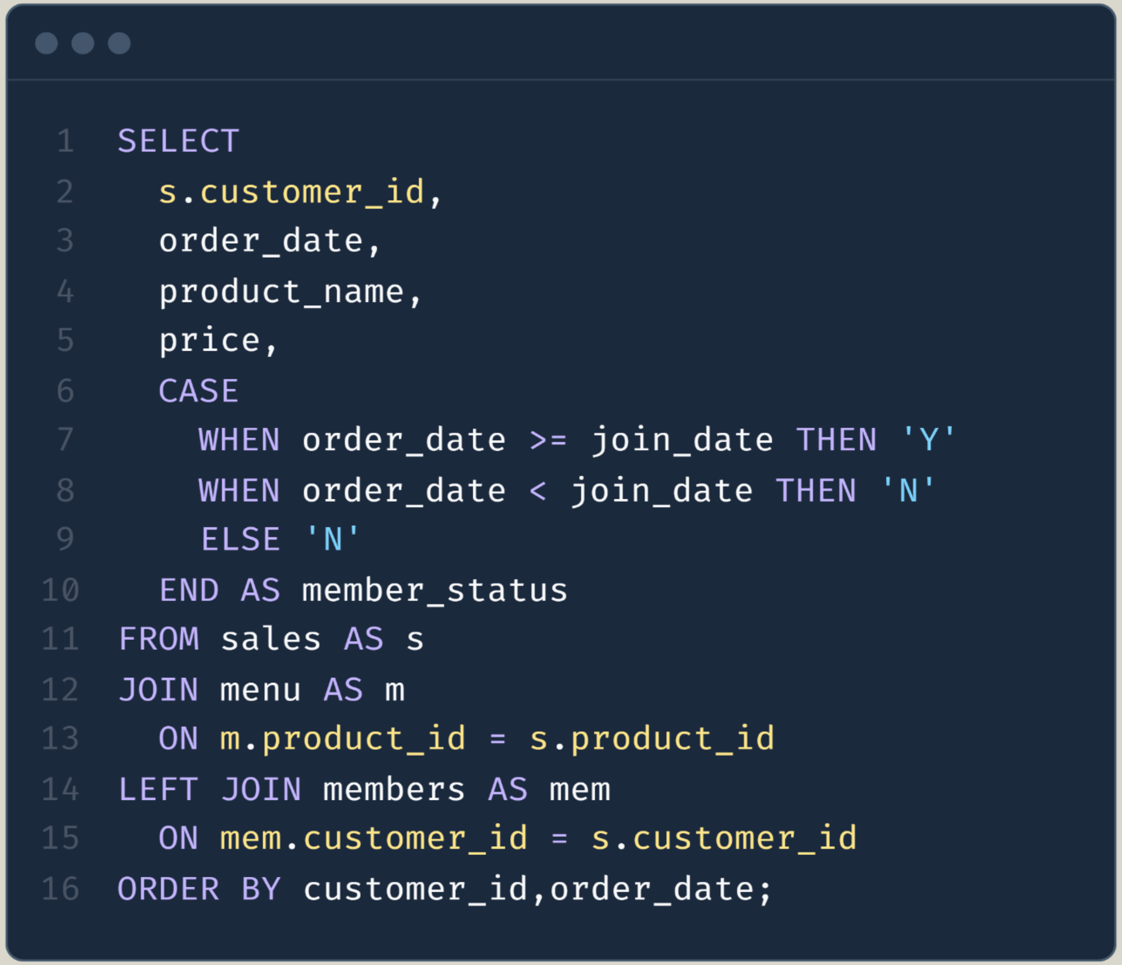
Output :



	customer_id	points_earned
▶	A	1020
	B	320

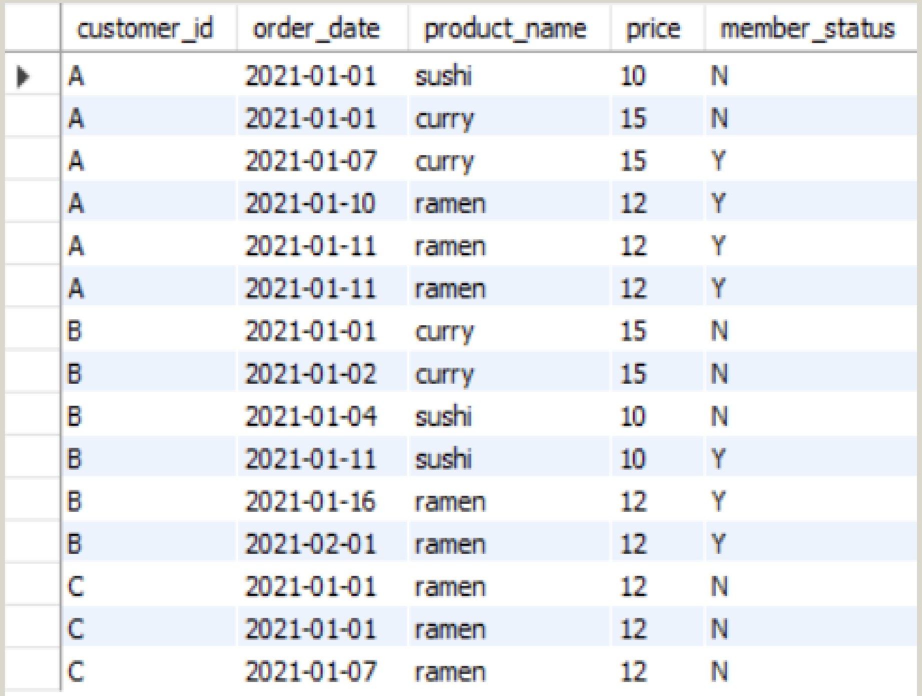
Join All The Things: Recreate the table with: customer_id, order_date, product_name, price, member (Y/N)

Query :



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

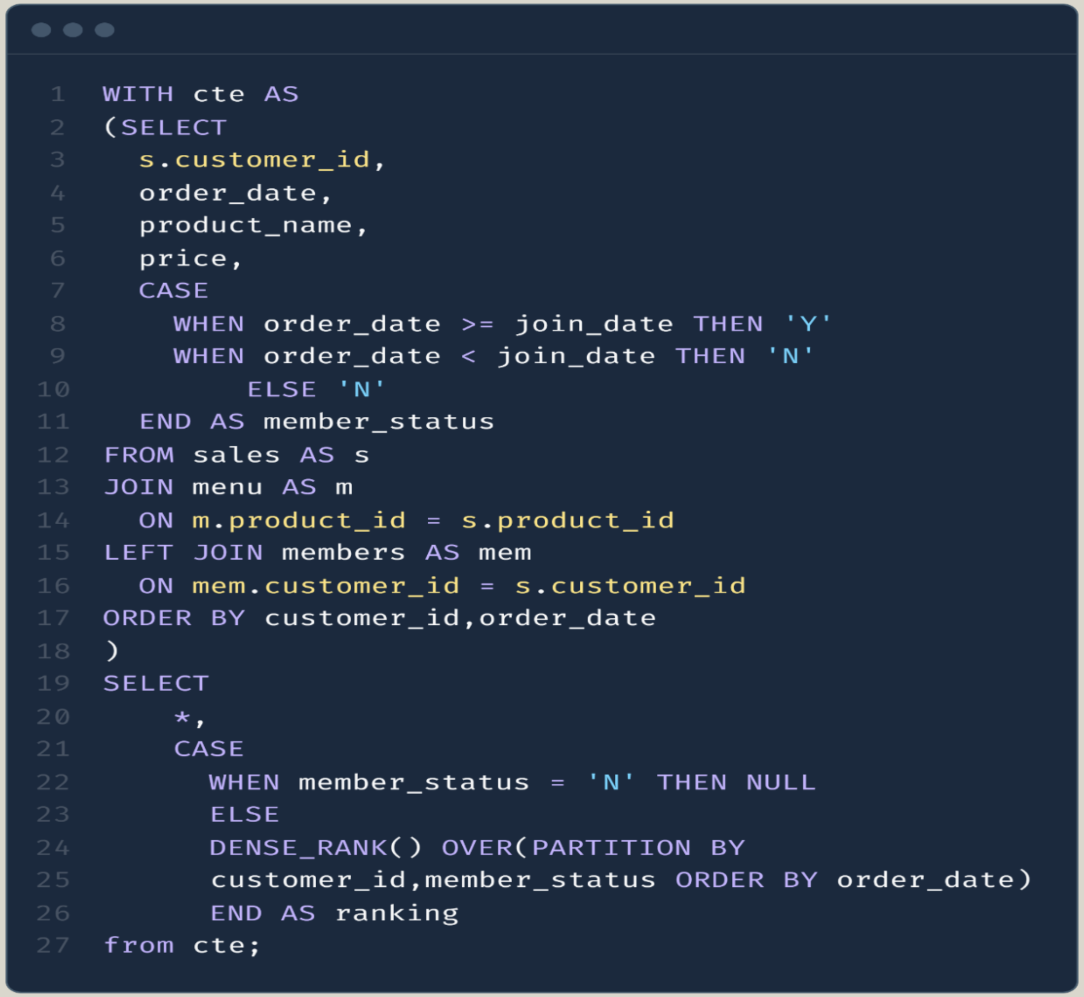
Output :



customer_id	order_date	product_name	price	member_status
A	2021-01-01	sushi	10	N
A	2021-01-01	curry	15	N
A	2021-01-07	curry	15	Y
A	2021-01-10	ramen	12	Y
A	2021-01-11	ramen	12	Y
A	2021-01-11	ramen	12	Y
B	2021-01-01	curry	15	N
B	2021-01-02	curry	15	N
B	2021-01-04	sushi	10	N
B	2021-01-11	sushi	10	Y
B	2021-01-16	ramen	12	Y
B	2021-02-01	ramen	12	Y
C	2021-01-01	ramen	12	N
C	2021-01-01	ramen	12	N
C	2021-01-07	ramen	12	N

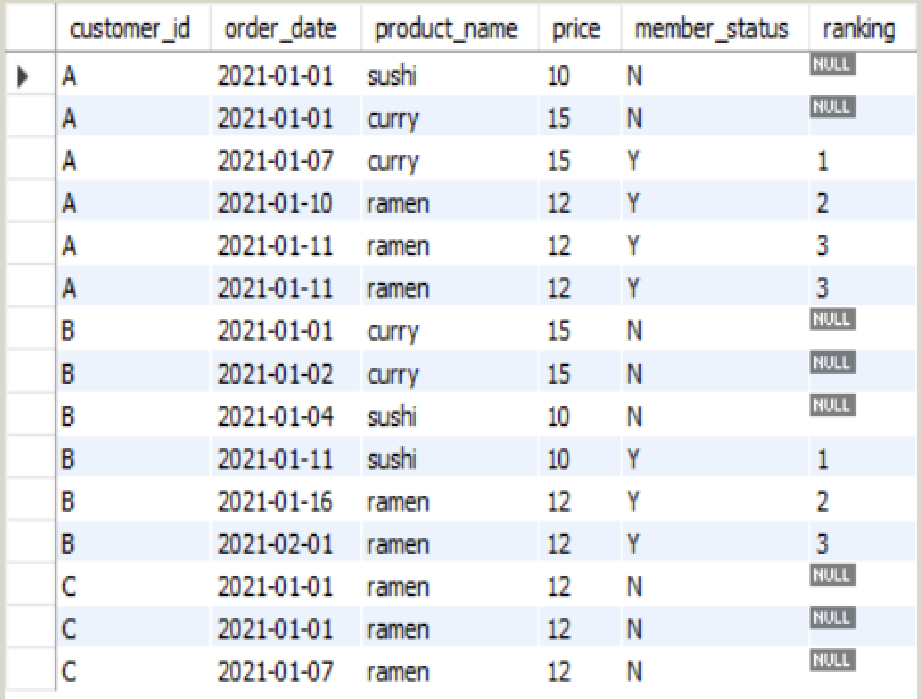
Rank All The Things : Danny also requires further information about the ranking of customer products, but he purposely does not need the ranking for non-member purchases so he expects null ranking values for the records

Query :



```
WITH cte AS
(SELECT
  s.customer_id,
  order_date,
  product_name,
  price,
  CASE
    WHEN order_date >= join_date THEN 'Y'
    WHEN order_date < join_date THEN 'N'
        ELSE 'N'
  END AS member_status
FROM sales AS s
JOIN menu AS m
  ON m.product_id = s.product_id
LEFT JOIN members AS mem
  ON mem.customer_id = s.customer_id
ORDER BY customer_id,order_date
)
SELECT
    *,
    CASE
      WHEN member_status = 'N' THEN NULL
      ELSE
      DENSE_RANK() OVER(PARTITION BY
      customer_id,member_status ORDER BY order_date)
      END AS ranking
from cte;
```

Output :



	customer_id	order_date	product_name	price	member_status	ranking
▶	A	2021-01-01	sushi	10	N	NULL
	A	2021-01-01	curry	15	N	NULL
	A	2021-01-07	curry	15	Y	1
	A	2021-01-10	ramen	12	Y	2
	A	2021-01-11	ramen	12	Y	3
	A	2021-01-11	ramen	12	Y	3
	B	2021-01-01	curry	15	N	NULL
	B	2021-01-02	curry	15	N	NULL
	B	2021-01-04	sushi	10	N	NULL
	B	2021-01-11	sushi	10	Y	1
	B	2021-01-16	ramen	12	Y	2
	B	2021-02-01	ramen	12	Y	3
	C	2021-01-01	ramen	12	N	NULL
	C	2021-01-01	ramen	12	N	NULL
	C	2021-01-07	ramen	12	N	NULL

Insights

Customer Visit Frequency: On average, customers visit the restaurant approximately 4 times. Such a high repeat customer frequency means that the customers generally enjoy dining there.

Average Spend Per Visit: The customers spend approximately $15.50 per visit. It may further be used for marketing to boost the spend per visit.

Menu item preferences: While customers' preferences may vary in the first visit-by choosing items such as curry, ramen, and sushi. Ramen is found to be the preferred order after frequent visits. This would imply long-term demand for it. Therefore, ramen can be marketed even more.

Recommendations

Opportunities in Menu Enlargement: Since the customers are at first diversified, the restaurant could benefit from introducing additional menu items. This will then attract a wider audience while promoting frequent visits due to the exploration of new options by customers.

Membership Benefits: To instill loyalty and encourage repeat visits, the restaurant can benefit membership program members by providing some privileges and benefits that they do not get from ordinary customers that may persuade them to join the membership program and increase their hours of visitation to the restaurant.

Thank You !