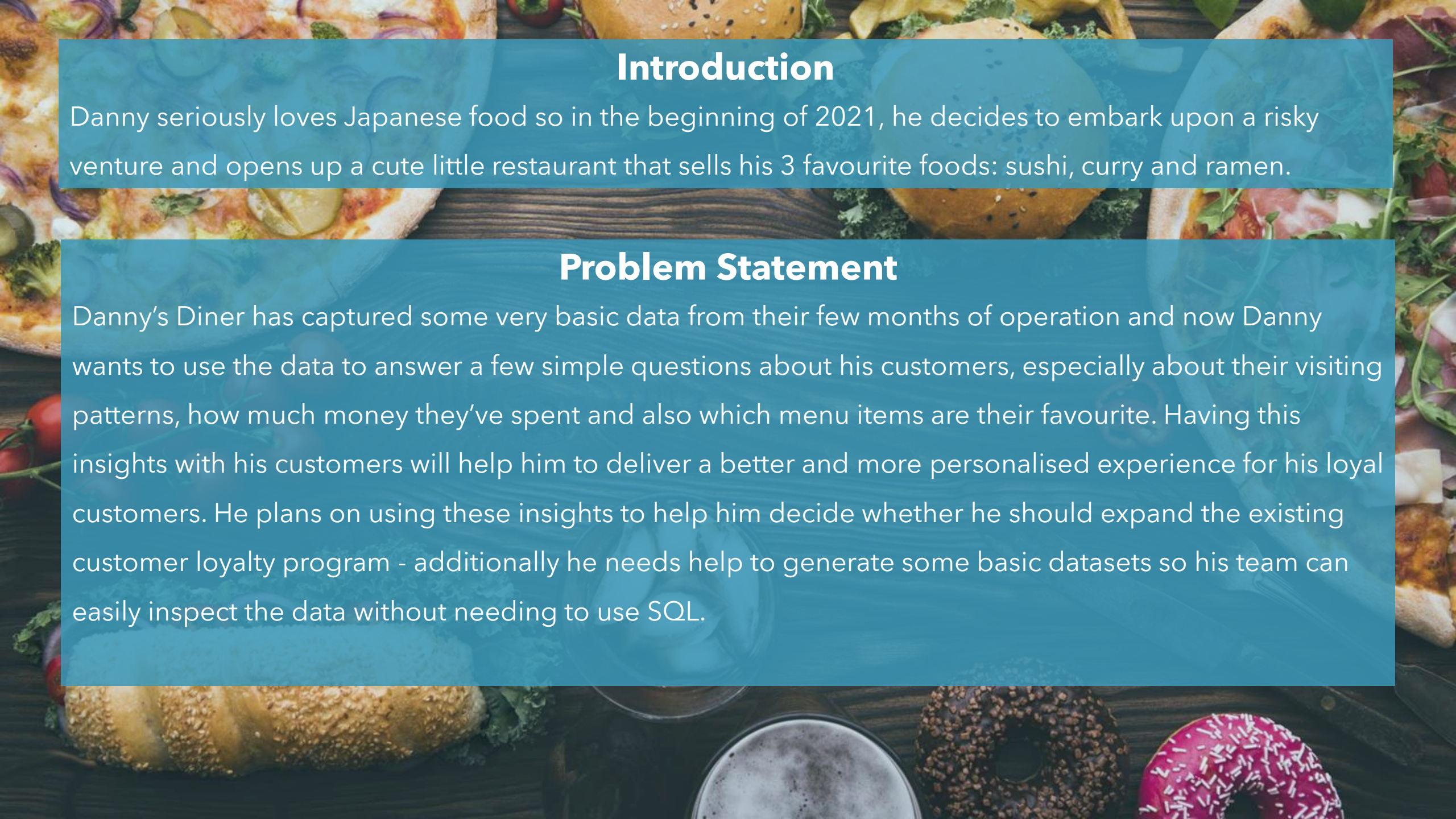
CASE STUDY #1 DANNY'S DINER
Taste of Success

By Tejas Chaudhari



Introduction

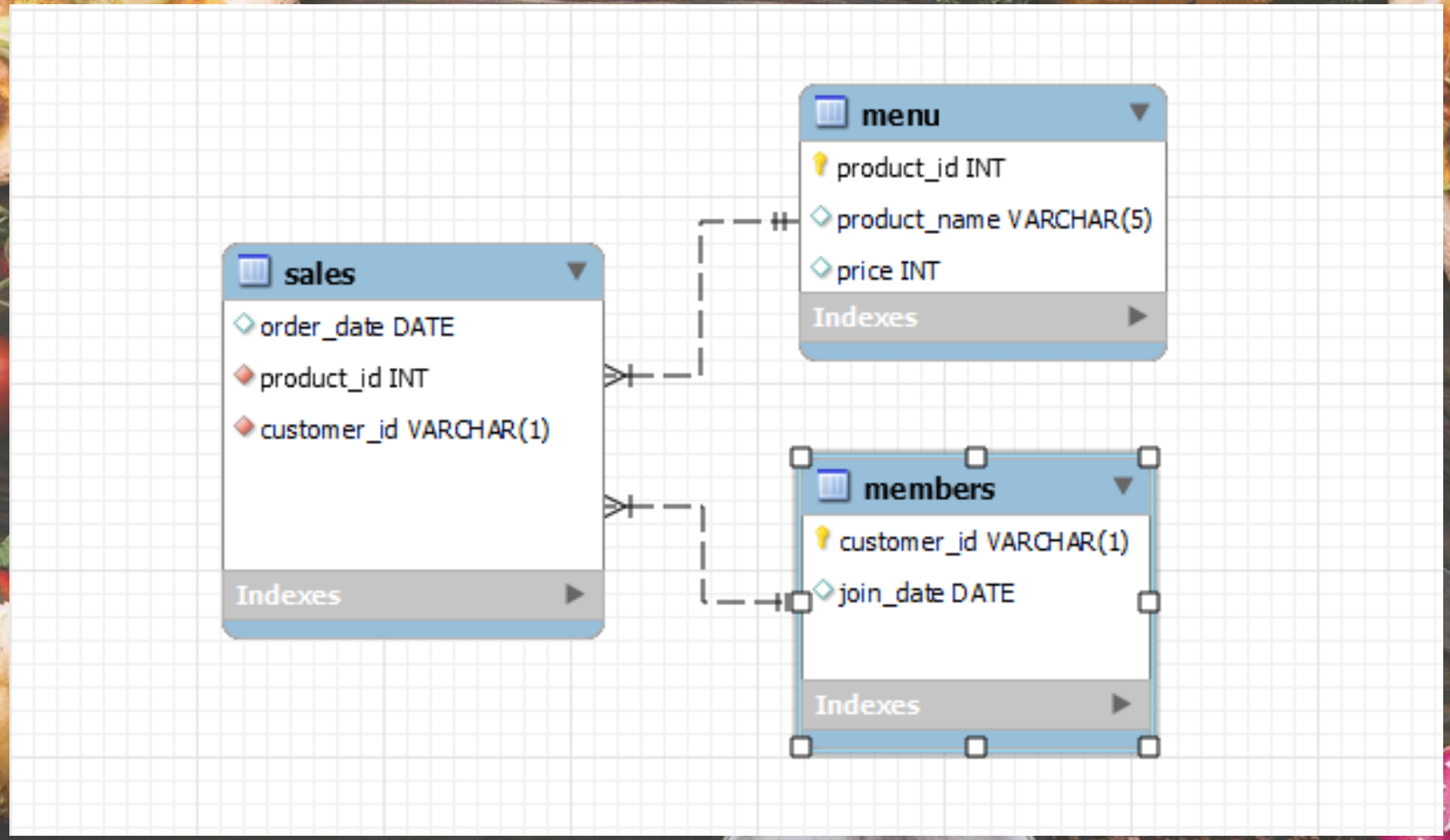
Danny seriously loves Japanese food so in the beginning of 2021, he decides to embark upon a risky venture and opens up a cute little restaurant that sells his 3 favourite foods: sushi, curry and ramen.

Problem Statement

Danny's Diner has captured some very basic data from their few months of operation and now Danny wants to use the data to answer a few simple questions about his customers, especially about their visiting patterns, how much money they've spent and also which menu items are their favourite. Having this insights with his customers will help him to deliver a better and more personalised experience for his loyal customers. He plans on using these insights to help him decide whether he should expand the existing customer loyalty program - additionally he needs help to generate some basic datasets so his team can easily inspect the data without needing to use SQL.

Dataset

- Sales : contains the transactional data for each sales with order date, product_id and customer _id.
- Menu : contains the price for each product along with its name mapped with product id.
- Members : contains the loyalty program customers along with their joining date mapped with customer_id.



Case study question #1

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

```
SELECT customer_id, SUM(price) AS total_sales
FROM sales JOIN menu
USING (product_id)
GROUP BY customer_id
ORDER BY customer_id;
```

customer_id	total_sales
A	76
B	74
C	36

Case study 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;
```

customer_id	visit_count
A	4
B	6
C	2

Case study question #3

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

```
WITH cte1 AS (
        SELECT customer_id, product_name, order_date,
        ROW_NUMBER() OVER(PARTITION BY customer_id ORDER BY order_date) AS item_rank
        FROM sales
        JOIN menu USING (product_id))
SELECT * FROM cte1 WHERE item_rank=1;
```

customer_id	product_name	order_date	item_rank
A	sushi	2021-01-01	1
B	curry	2021-01-01	1
C	ramen	2021-01-01	1

Case study question #4

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

```
SELECT product_name,COUNT(product_name) AS most_purchased_item
FROM sales
JOIN menu USING (product_id)
GROUP BY product_name
ORDER BY most_purchased_item DESC LIMIT 1;
```

product_name	most_purchased_item
ramen	8

Case study question #5

Which item was the most popular for each customer?

```
WITH cte1 AS(
          SELECT customer_id,product_name,
          COUNT(product_name) AS number_of_times_purchased,
          DENSE_RANK() OVER(PARTITION BY customer_id
          ORDER BY COUNT(product_name) DESC) AS rank_num
          FROM sales JOIN menu USING (product_id)
          GROUP BY customer_id,product_name)
SELECT * FROM CTE1 WHERE rank_num = 1;
```

customer_id	product_name	number_of_times_purchased	rank_num
A	ramen	3	1
B	curry	2	1
B	sushi	2	1
B	ramen	2	1
C	ramen	3	1

Case study question #6

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

```
WITH cte1 AS(
    SELECT product_name,s.customer_id,order_date,join_date,me.product_id,
    DENSE_RANK() OVER(PARTITION BY s.customer_id ORDER BY s.order_date) AS item_rank
    FROM sales s
    JOIN members m USING (customer_id)
    JOIN menu me USING (product_id)
    WHERE order_date>=join_date)
SELECT customer_id, product_name, order_date FROM cte1 WHERE item_rank=1;
```

customer_id	product_name	order_date
A	curry	2021-01-07
B	sushi	2021-01-11

Case study question #7

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

```
WITH product_before_member AS(
        SELECT product_name, s.customer_id, order_date, join_date, m.product_id,
        DENSE_RANK() OVER(PARTITION BY s.customer_id
        ORDER BY s.order_date DESC) AS item_rank
        FROM menu m
        JOIN sales s USING (product_id)
        JOIN members me USING (customer_id)
        WHERE order_date < join_date )
SELECT customer_id, product_name, order_date, join_date
FROM product_before_member WHERE item_rank=1;
```

customer_id	product_name	order_date	join_date
A	sushi	2021-01-01	2021-01-07
A	curry	2021-01-01	2021-01-07
B	sushi	2021-01-04	2021-01-09

Case study question #8

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

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

customer_id	total_items	amount_spent
A	2	25
B	3	40

Case study question #9

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

```
SELECT customer_id,
        SUM(CASE
                WHEN product_name = 'sushi' THEN price*20
                ELSE price*10
        END) AS customer_points
FROM menu m JOIN sales s USING (product_id)
GROUP BY customer_id ORDER BY customer_id;
```

customer_id	customer_points
A	860
B	940
C	360

Case study 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 cte1 AS(
          SELECT join_date, DATE_ADD(join_date, INTERVAL 7 DAY) AS program_last_date,
          customer_id FROM members)
SELECT s.customer_id,
        SUM(CASE
                WHEN order_date BETWEEN join_date AND program_last_date THEN price*20
                WHEN order_date NOT BETWEEN join_date AND program_last_date
                     AND product_name = 'sushi' THEN price*20
                WHEN order_date NOT BETWEEN join_date AND program_last_date
                     AND product_name != 'sushi' THEN price*10
            END) AS customer_points
FROM menu m JOIN sales s USING (product_id)
JOIN cte1 ON cte1.customer_id = s.customer_id
AND order_date<='2021-01-31' AND order_date >=join_date
GROUP BY s.customer_id ORDER BY s.customer_id;
```

customer_id	customer_points
A	1020
B	440

Case study bonus question #1
Join all the things and show loyalty program member or not

```sql
SELECT customer_id,order_date,product_name,price,
    CASE WHEN join_date IS NULL THEN 'N'
         WHEN join_date > order_date THEN 'N'
         ELSE 'Y'
    END AS member
FROM menu m
LEFT JOIN sales s USING (product_id)
LEFT JOIN members me USING (customer_id)
ORDER BY customer_id, order_date;
```

customer_id	order_date	product_name	price	member
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

Case study bonus question #2
Join all the things and rank loyalty program member

```sql
WITH cte1 AS(SELECT customer_id,order_date,product_name,price,
    CASE WHEN join_date IS NULL THEN 'N'
         WHEN join_date > order_date THEN 'N'
         ELSE 'Y'
    END AS member
FROM menu m
LEFT JOIN sales s USING (product_id)
LEFT JOIN members me USING (customer_id)
ORDER BY customer_id, order_date)
SELECT *,
    CASE WHEN member != 'N' THEN
        DENSE_RANK() OVER(PARTITION BY customer_id,member ORDER BY order_date)
        ELSE NULL
    END AS ranklist
FROM cte1;
```

customer_id	order_date	product_name	price	member	ranklist
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 found can help danny in following ways:

- Customer report : know high value customers by number of visits and total sales. Taking customer feedback of those customers. Know better performing and improvement needed areas.

- Product performance : know the most selling dish overall and diving deep into at each customer level. Devising the menu price changes according to it.

- Evaluate loyalty program : know top selling dish before and after becoming loyalty program members. Understand the effectiveness of loyalty program. Whether to continue the program or not. Changes to be made.

- Devise Membership acquisition and retention strategy based on loyalty program evaluation.

- Loyalty points : Encourage spending with points multipliers.

- Maintain reports of customer for quicker evaluation.


THANK YOU!