```sql
-- Querying sales table
SELECT *
FROM sales;
```

customer_id	order_date	product_id
A	2021-01-01	1
A	2021-01-01	2
A	2021-01-07	2
A	2021-01-10	3
A	2021-01-11	3
A	2021-01-11	3
B	2021-01-01	2
B	2021-01-02	2
B	2021-01-04	1
B	2021-01-11	1
B	2021-01-16	3
B	2021-02-01	3
C	2021-01-01	3
C	2021-01-01	3
C	2021-01-07	3

```sql
-- Querying menu table
SELECT *
FROM menu;
```

product_id	product_name	price
1	sushi	10
2	curry	15
3	ramen	12

```sql
-- Querying members table
SELECT *
FROM members;
```

customer_id	join_date
A	2021-01-07
B	2021-01-09

```sql
-- CASE STUDY QUESTIONS
-- 1. Total amount spent by each customer at the restaurant
SELECT S.customer_id,
            SUM(M.price) AS total_amount_spent
FROM dannys_diner.dbo.sales AS S
INNER JOIN dannys_diner.dbo.menu AS M
ON S.product_id = M.product_id
GROUP BY S.customer_id
ORDER BY total_amount_spent DESC;
```

```
customer_id total_amount_spent
----------- ------------------
A           76
B           74
C           36
```

```sql
-- 2. Days each customer visited the restaurant
SELECT customer_id,
            COUNT(DISTINCT(order_date)) AS number_days_visited
FROM dannys_diner.dbo.sales
GROUP BY customer_id
ORDER BY number_days_visited DESC;
```

```
customer_id number_days_visited
----------- -------------------
B           6
A           4
C           2
```

```sql
-- 3. First item from the menu purchased by each customer
SELECT DISTINCT(S.customer_id),
            M.product_id,
        M.product_name
FROM (
      SELECT *,
            RANK() OVER(PARTITION BY customer_id ORDER BY order_date ASC) AS
rank
       FROM dannys_diner.dbo.sales
    ) AS S
INNER JOIN dannys_diner.dbo.menu AS M
ON S.product_id = M.product_id
WHERE rank = 1;
```

customer_id	product_id	product_name
A	1	sushi
A	2	curry
B	2	curry
C	3	ramen

```sql
-- 4. Most purchased item on the menu
SELECT TOP 1 product_id,
                    COUNT(product_id) AS num_times_purchased
FROM dannys_diner.dbo.sales
GROUP BY product_id
ORDER BY COUNT(product_id) DESC;
```

```
product_id  num_times_purchased
----------- -------------------
3           8
```

```sql
-- 5. Number of times the most purchased item was purchased by customers
WITH Most_purchased AS (
      SELECT customer_id,
            MIN(product_id) AS product_id,
            COUNT(product_id) AS times_purchased
      FROM dannys_diner.dbo.sales
      WHERE product_id IN (
               SELECT TOP 1 product_id
               FROM dannys_diner.dbo.sales
               GROUP BY product_id
               ORDER BY COUNT(product_id) DESC
               )
      GROUP BY customer_id
            )

SELECT MP.customer_id,
            MP.product_id AS most_purchased_product_id,
            M.product_name AS most_purchased_product_name,
        MP.times_purchased
FROM Most_Purchased AS MP
INNER JOIN dannys_diner.dbo.menu AS M
ON MP.product_id = M.product_id
ORDER BY MP.customer_id ASC;
```

```
customer_id most_purchased_product_id most_purchased_product_name times_purchased
----------- ---------------------------------- ------------------------------------- -------------------
A           3                                  ramen                                 3
B           3                                  ramen                                 2
C           3                                  ramen                                 3
```

```sql
-- 6. Most popular item for each customer
WITH Popular_item AS (
        SELECT customer_id,
              product_id,
              times_purchased,
              DENSE_RANK() OVER(PARTITION BY customer_id ORDER BY times_purchased
DESC) AS rank
      FROM (SELECT customer_id,
                  product_id,
                  COUNT(product_id) AS times_purchased
            FROM dannys_diner.dbo.sales
            GROUP BY customer_id, product_id
             ) AS S
                   )
```

```sql
SELECT P.customer_id,
             P.product_id AS popular_item_id,
        M.product_name AS popular_item_name
FROM Popular_item AS P
INNER JOIN dannys_diner.dbo.menu AS M
ON P.product_id = M.product_id
WHERE Rank = 1
ORDER BY P.customer_id;
```

customer_id	popular_item_id	popular_item_name
A	3	ramen
B	1	sushi
B	2	curry
B	3	ramen
C	3	ramen

```sql
-- 7. Item purchased first by the customer after they became a member
WITH item_after_member AS (
      SELECT s.customer_id,
            s.order_date,
            s.product_id,
            m.join_date,
            DENSE_RANK() OVER(PARTITION BY s.customer_id ORDER BY s.order_date
ASC) AS rank
    FROM dannys_diner.dbo.sales AS S
    LEFT JOIN dannys_diner.dbo.members AS M
    ON S.customer_id = M.customer_id AND S.order_date >= M.join_date
    WHERE m.join_date IS NOT NULL
             )

SELECT AM.customer_id,
             AM.product_id,
        MN.product_name
FROM item_after_member AS AM
INNER JOIN dannys_diner.dbo.menu AS MN
ON AM.product_id = MN.product_id
WHERE rank = 1
ORDER BY AM.customer_id;
```

customer_id	product_id	product_name
A	2	curry
B	1	sushi

```sql
-- 8. Item purchased just before the customer became a member
WITH item_before_member AS (
      SELECT s.customer_id,
            s.order_date,
            s.product_id,
            m.join_date,
            DENSE_RANK() OVER(PARTITION BY s.customer_id ORDER BY s.order_date
DESC) AS rank
```

```
    FROM dannys_diner.dbo.sales AS S
    LEFT JOIN dannys_diner.dbo.members AS M
    ON S.customer_id = M.customer_id AND S.order_date < M.join_date
    WHERE m.join_date IS NOT NULL
            )

SELECT BM.customer_id,
            BM.product_id,
        MN.product_name
FROM item_before_member AS BM
INNER JOIN dannys_diner.dbo.menu AS MN
ON BM.product_id = MN.product_id
WHERE rank = 1
ORDER BY BM.customer_id;
```

customer_id	product_id	product_name
A	1	sushi
A	2	curry
B	1	sushi

```
-- 9. The total items and amount spent for each member before they became a member
SELECT S.customer_id,
            COUNT(S.product_id) AS total_items_purchased,
        SUM(MN.price) AS total_amount_spent
FROM dannys_diner.dbo.sales AS S
LEFT JOIN dannys_diner.dbo.members AS M
ON S.customer_id = M.customer_id
      AND S.order_date < M.join_date
LEFT JOIN dannys_diner.dbo.menu AS MN
ON S.product_id = MN.product_id
WHERE m.join_date IS NOT NULL
GROUP BY S.customer_id
ORDER BY S.customer_id;
```

customer_id	total_items_purchased	total_amount_spent
A	2	25
B	3	40

```
-- 9.1 Scenario: Each $1 spent equates to 10 points and sushi has a 2x points
multiplier.
-- Calculating points each customer would have earned
SELECT S.customer_id,
        SUM(CASE
                    WHEN M.product_name = 'sushi' THEN M.price * 20
                    ELSE M.price * 10
            END) AS points
FROM dannys_diner.dbo.sales AS S
INNER JOIN dannys_diner.dbo.menu AS M
ON S.product_id = M.product_id
GROUP BY S.customer_id
ORDER BY S.customer_id;
```

```
customer_id points
----------- -----------
A           860
B           940
C           360
```

```sql
-- Scenario: In the first week after a customer joins the program (including their
join date) they earn 2x points on all items, not just sushi.
-- Calculating the points customer A and B have at the end of January
WITH january_points AS (
        SELECT S.customer_id,
                S.order_date,
                MN.product_name,
                MN.price,
                MS.join_date,
                CASE
                    WHEN MN.product_name = 'sushi' THEN MN.price * 20
                    WHEN S.order_date < MS.join_date THEN MN.price * 10
                    WHEN S.order_date = MS.join_date THEN MN.price * 20
                    WHEN S.order_date <= DATEADD(day, 6, MS.join_date) THEN
MN.price * 20
                    ELSE MN.price * 10
                END AS points
        FROM dannys_diner.dbo.sales AS S
        INNER JOIN dannys_diner.dbo.menu AS MN
        ON S.product_id = MN.product_id
        LEFT JOIN dannys_diner.dbo.members AS MS
        ON S.customer_id = MS.customer_id
        WHERE MS.join_date IS NOT NULL
                AND S.order_date < '2021-2-1'
                )

SELECT customer_id,
            SUM(points) AS total_january_points
FROM january_points
GROUP BY customer_id;
```

```
customer_id total_january_points
----------- --------------------
A           1370
B           820
```

```sql
-- Bonus Question 1: Joining all the things
SELECT S.customer_id,
            S.order_date,
        MN.product_name,
        MN.price,
        CASE
            WHEN S.order_date < MS.join_date THEN 'N'
           WHEN S.order_date >= MS.join_date THEN 'Y'
            ELSE 'N'
        END AS member
FROM dannys_diner.dbo.sales AS S
INNER JOIN dannys_diner.dbo.menu AS MN
```

```
ON S.product_id = MN.product_id
LEFT JOIN dannys_diner.dbo.members AS MS
ON S.customer_id = MS.customer_id
ORDER BY S.customer_id ASC, S.order_date ASC, MN.product_name ASC;
```

customer_id	order_date	product_name	price	member
A	2021-01-01	curry	15	N
A	2021-01-01	sushi	10	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

```
-- Bonus Question 2: Joining and Ranking all the things
WITH all_things AS (
    SELECT S.customer_id,
            S.order_date,
            MN.product_name,
            MN.price,
            CASE
                WHEN S.order_date < MS.join_date THEN 'N'
                WHEN S.order_date >= MS.join_date THEN 'Y'
                ELSE 'N'
            END AS member
    FROM dannys_diner.dbo.sales AS S
    INNER JOIN dannys_diner.dbo.menu AS MN
    ON S.product_id = MN.product_id
    LEFT JOIN dannys_diner.dbo.members AS MS
    ON S.customer_id = MS.customer_id
             )

SELECT *,
             CASE
             WHEN member = 'N' THEN NULL
            ELSE RANK() OVER(PARTITION BY customer_id, member ORDER BY order_date
ASC)
        END AS rank
FROM all_things;
```

customer_id	order_date	product_name	price	member	rank
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