Case Study #1 - Danny's Diner
Source: https://8weeksqlchallenge.com/case-study-1/

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 favorite foods: sushi, curry and ramen.

Danny’s Diner is in need of your assistance to help the restaurant stay afloat - the restaurant has captured some very basic data from their few months of operation but have no idea how to use their data to help them run the business.

Problem Statement
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 favorite. Having this deeper connection with his customers will help him deliver a better and more personalized 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.

Danny has provided you with a sample of his overall customer data due to privacy issues - but he hopes that these examples are enough for you to write fully functioning SQL queries to help him answer his questions!

Danny has shared with you 3 key datasets for this case study:

sales
menu
members

DANNY'S DINER

THE TASTE OF SUCCESS

Table 1: sales

The `sales` table captures all `customer_id` level purchases with an corresponding `order_date` and `product_id` information for when and what menu items were ordered.

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

Table 2: menu

The `menu` table maps the `product_id` to the actual `product_name` and `price` of each menu item.

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

Table 3: members

The final `members` table captures the `join_date` when a `customer_id` joined the beta version of the Danny's Diner loyalty program.

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

DANNY'S DINER

THE TASTE OF SUCCESS

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

Code:

```sql
SELECT
dannys_diner.sales.customer_id, SUM(dannys_diner.menu.price) AS total_amount
FROM dannys_diner.sales
JOIN dannys_diner.menu on sales.product_id=menu.product_id
GROUP BY customer_id
ORDER BY customer_id;
```

Results:

customer_id	total_amount
A	76
B	74
C	36

2. How many days has each customer visited the restaurant?

Code:

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

Results:

customer_id	days
A	4
B	6
C	2

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

Code:

```sql
SELECT customer_id, product_name, order_date
FROM dannys_diner.sales
JOIN dannys_diner.menu on sales.product_id=menu.product_id
GROUP BY customer_id, order_date, sales.product_id, product_name
ORDER BY order_date ASC;
```

Results:

customer_id	product_name	order_date
A	sushi	2021-01-01T00:00:00.000Z
A	curry	2021-01-01T00:00:00.000Z
B	curry	2021-01-01T00:00:00.000Z
C	ramen	2021-01-01T00:00:00.000Z

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

Code:

```sql
SELECT product_name, COUNT(order_date) AS number_of_purchases
FROM dannys_diner.sales
JOIN dannys_diner.menu on sales.product_id=menu.product_id
GROUP BY   product_name
ORDER BY number_of_purchases DESC
LIMIT 1;
```

Results:

product_name	number_of_purchases
ramen	8

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

Code:

```
SELECT sales.customer_id, product_name, order_date, join_date
FROM dannys_diner.sales
JOIN dannys_diner.menu on sales.product_id=menu.product_id
JOIN dannys_diner.members on sales.customer_id=members.customer_id
WHERE join_date<order_date
ORDER BY order_date ASC
LIMIT 2;
```

Results:

customer_id	product_name	order_date	join_date
A	ramen	2021-01-10T00:00:00.000Z	2021-01-07T00:00:00.000Z
B	sushi	2021-01-11T00:00:00.000Z	2021-01-09T00:00:00.000Z

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

Code:

```
SELECT t.customer_id, product_name
FROM
(SELECT sales.customer_id, product_name, order_date, join_date,
 ROW_NUMBER()
    OVER (
   PARTITION BY sales.customer_id
   ORDER BY sales.customer_id, order_date DESC
            ) AS rownumber
FROM dannys_diner.sales
JOIN dannys_diner.menu on sales.product_id=menu.product_id
JOIN dannys_diner.members on sales.customer_id=members.customer_id
WHERE join_date>order_date
 ) t
WHERE t.rownumber = 1;
```

Results:

customer_id	product_name
A	sushi
B	sushi

DANNY'S DINER

THE TASTE OF SUCCESS

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

Code:

```sql
SELECT dannys_diner.sales.customer_id, SUM(dannys_diner.menu.price) AS total_amount_spent, COUNT(product_name) AS total_items
FROM dannys_diner.sales
JOIN dannys_diner.menu on sales.product_id=menu.product_id
JOIN dannys_diner.members on sales.customer_id=members.customer_id
WHERE join_date>order_date
GROUP BY sales.customer_id
ORDER BY sales.customer_id;
```

Results:

customer_id	total_amount_spent	total_items
A	25	2
B	40	3

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

Code:

```sql
SELECT t.customer_id, SUM(total_amount_spent * CASE WHEN product_name = 'sushi' THEN 2 ELSE 1 END * 10)  AS points
FROM
(SELECT sales.customer_id, product_name, SUM(dannys_diner.menu.price) AS total_amount_spent
FROM dannys_diner.sales
JOIN dannys_diner.menu ON sales.product_id = menu.product_id
GROUP BY sales.customer_id, product_name
) t
GROUP BY t.customer_id
ORDER BY t.customer_id;
```

Results:

customer_id	points
A	860
B	940
C	360

9. 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?

Code:

```sql
SELECT sales.customer_id, SUM(price * 2 * 10)  AS points
FROM dannys_diner.sales
JOIN dannys_diner.menu ON sales.product_id = menu.product_id
JOIN dannys_diner.members on sales.customer_id=members.customer_id
WHERE join_date<=order_date
GROUP BY sales.customer_id
ORDER BY sales.customer_id;
```

Results:

customer_id	points
A	1020
B	680