Case Study 1 : Epic Eats

Background:

Epic Eats is a popular restaurant chain with several locations across the city. They're known for their mouth-watering burgers, crispy fries, and refreshing shakes. To better understand their customers and optimize their menu, Epic Eats needs your help with data analysis.

Problem Statement:

Rahul, the owner of Epic Eats, wants to leverage data to gain a deeper understanding of his customers' visiting patterns, spending habits, and favorite menu items. This insight will enable him to deliver a more personalized experience for his loyal customers.

Rahul aims to use these findings to inform his decision to expand the existing customer loyalty program. Additionally, he needs help generating basic datasets that his team can easily inspect without requiring SQL expertise.

Due to privacy concerns, Rahul has shared a sample of his overall customer data. He hopes this sample is sufficient for you to write fully functional SQL queries to answer his questions.

Rahul has provided three key datasets for this case study:

- sales
- menu
- members

Example Datasets

All datasets exist within the Epic Eats database schema - be sure to include this reference within your SQL scripts as you start exploring the data and answering the case study questions.

Table 1: sales

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

customer_id	order_date	product_id
AAA	2024-01-01	101
AAA	2024-01-01	102
AAA	2024-01-07	102
AAA	2024-01-10	103
AAA	2024-01-11	103
AAA	2024-01-11	103
BBB	2024-01-01	102
BBB	2024-01-02	102
BBB	2024-01-04	101
BBB	2024-01-11	101
BBB	2024-01-16	103
BBB	2024-02-01	103
CCC	2024-01-01	103
CCC	2024-01-01	103
CCC	2024-01-07	103

Table 2: menu

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

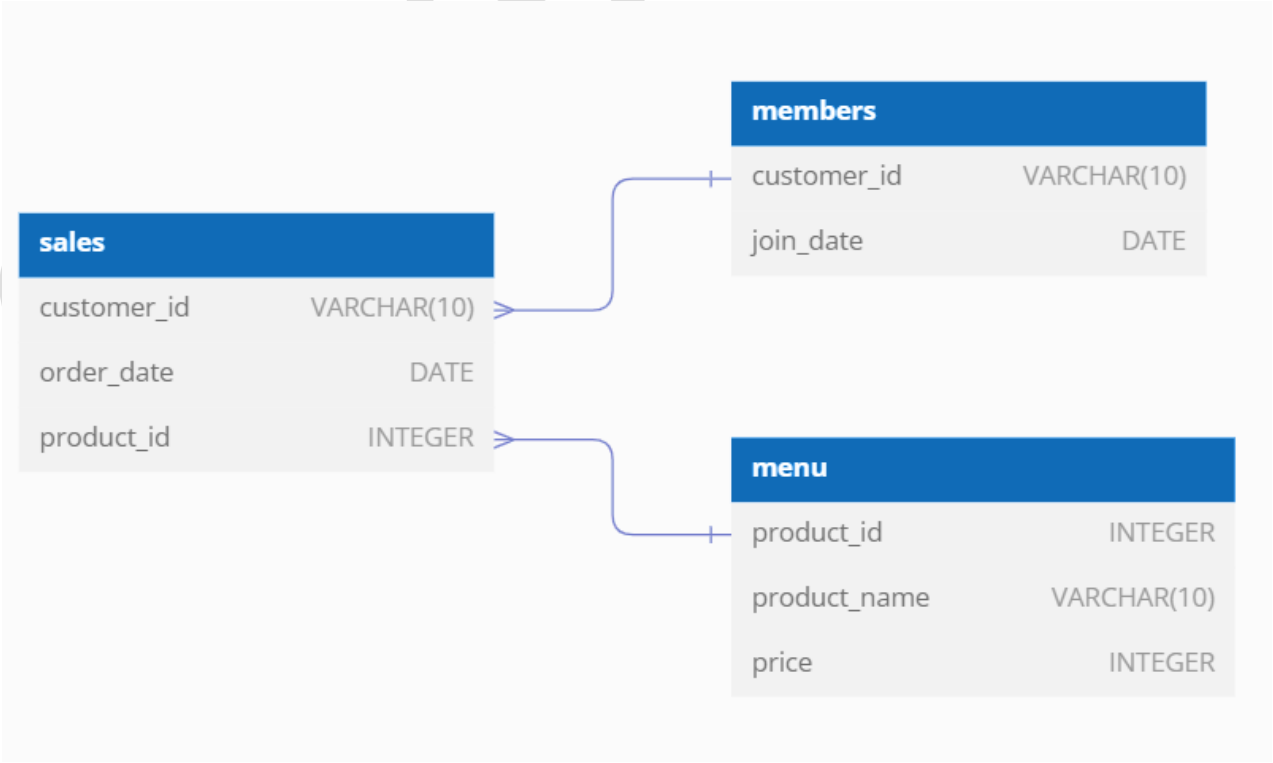
product_id	product_name	price
101	Sandwich	10
102	Pizza	15
103	Burger	12

Table 3: members

The final members table captures the `join_date` when a `customer_id` joined the beta version of the Epic Eats loyalty program.

customer_id	join_date
AAA	2024-01-07
BBB	2024-01-09

Entity Relationship Diagram



Case Study Questions

Each of the following case study questions can be answered using a single SQL statement:

1. What is the total amount each customer spent at Epic Eats?
2. How many days has each customer visited Epic Eats?
3. What was the first item from the menu purchased by each customer?
4. What is the most purchased item on the menu and how many times was it purchased by all customers?
5. Which item was the most popular for each customer?
6. Which item was purchased first by the customer after they became a member?
7. Which item was purchased just before the customer became a member?
8. What is the total items and amount spent for each member before they became a member?
9. If each $1 spent equates to 10 points and Sandwich has a 2x points multiplier - how many points would each customer have?
10. In the first week after a customer joins the program (including their join date) they earn 2x points on all items, not just Sandwich - how many points do customer AAA and BBB have at the end of January?

Bonus Questions:

Q1. Join All The Things

Recreate the table with columns:

`customer_id, order_date, product_name, price, member_status (Y/N)`

Sample Output:

customer_id	order_date	product_name	price	member_status
AAA	2024-01-01	sandwich	10	N
AAA	2024-01-01	pizza	15	N
AAA	2024-01-07	pizza	15	Y

Q2. Rank All The Things

Rahul 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.

Sample Output:

customer_id	order_date	product_name	price	member_status	ranking
AAA	2024-01-01	pizza	15	N	NULL
AAA	2024-01-01	sandwich	10	N	NULL
AAA	2024-01-07	pizza	15	Y	2
AAA	2024-01-10	burger	12	Y	3