Arpan Ghosh
https://www.linkedin.com/in/itsarpan/

Danny's Diner Problem | 8 Weeks SQL Challenge

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. 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 favourite. Having this deeper connection with his customers will help him 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.

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

Entity Relationship Diagram

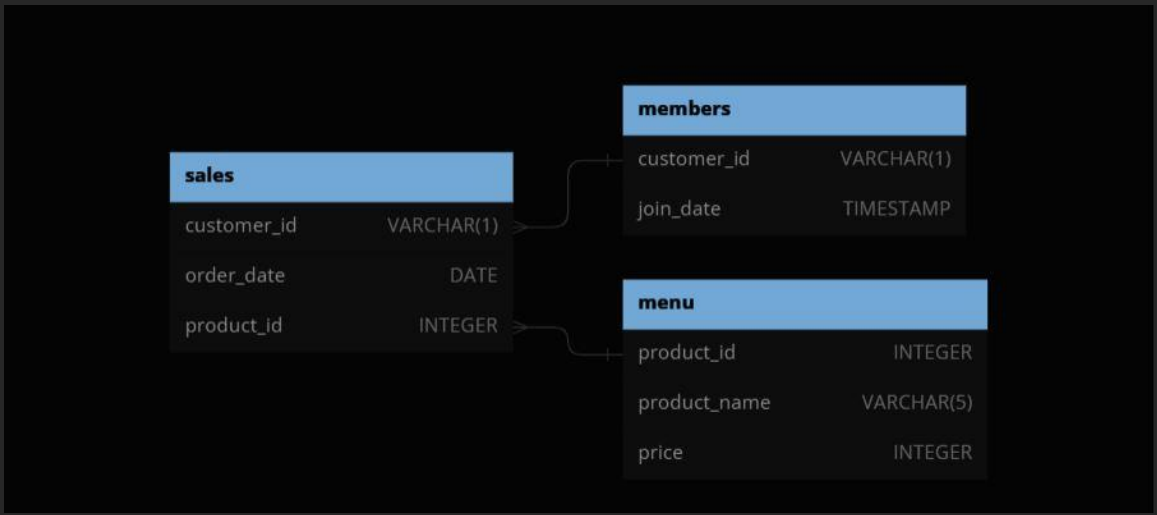

Table 1: sales

The sales table captures all customer_id level purchases with a 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

Case Study Question

1.

```
1    -- 1. what is the total amount each cumtomer spent at the resturant?
2
3 •  select customer_id, sum(price) as total_spend from sales as s
4    inner join menu as m on s.product_id = m.product_id
5    group by customer_id;
```

2.

```
7    -- 2. How many days has each customer visited the resturant?
8 •  select
9    customer_id,
10   count(distinct order_date) as date_visited
11   from sales
12   group by customer_id;
```

3.

```sql
-- 3. What was the first item from the men u perchased each customer?
WITH CTE AS (
  SELECT
    customer_id,
    order_date,
    product_name,
    RANK() OVER(PARTITION BY CUSTOMER_ID ORDER BY order_date) as rnk,
    ROW_NUMBER() OVER(PARTITION BY customer_id ORDER BY order_date ASC) as rn
  FROM
    SALES as S
    INNER JOIN MENU as M on S.product_id = M.product_id
)
SELECT
  customer_id,
  product_name
FROM
  CTE
WHERE
  rnk = 1;
```

4.

```sql
-- 4. what is the most purchased item on the menu and how many times was it purchased by all customers?
SELECT
  product_name,
  COUNT(order_date) as orders
FROM
  SALES as S
  INNER JOIN MENU as M on S.product_id = M.product_id
GROUP BY
  product_name
ORDER BY
  COUNT(order_date) DESC
LIMIT 1;

```

5.

```sql
-- 5. Which item was the most popular for each customer?
WITH CTE AS (
  SELECT
    product_name,
    customer_id,
    COUNT(order_date) as orders,
    RANK() OVER(PARTITION BY customer_id ORDER BY COUNT(order_date) DESC) as rnk,
    ROW_NUMBER() OVER(PARTITION BY customer_id ORDER BY COUNT(order_date) DESC) as rn
  FROM
    SALES as S
    INNER JOIN MENU as M on S.product_id = M.product_id
  GROUP BY
    product_name,
    customer_id
)
SELECT
  customer_id,
  product_name
FROM
  CTE
WHERE rnk = 1;

```

6.

```sql
-- 6. Which item was purchased first by the customer after they became a member?
WITH CTE AS (
  SELECT
    S.customer_id,
    order_date,
    join_date,
    product_name,
    RANK() OVER(PARTITION BY S.customer_id ORDER BY order_date ASC) as rnk,
    ROW_NUMBER() OVER(PARTITION BY S.customer_id ORDER BY order_date) as rn
  FROM
    SALES as S
    INNER JOIN MENU as M on S.product_id = M.product_id
    INNER JOIN MEMBERS as MEM ON MEM.customer_id = S.customer_id
  WHERE
    order_date >= join_date
  ORDER BY
    order_date ASC
)
SELECT
  customer_id,
  product_name
FROM
  CTE
WHERE
  rnk = 1;
```

7.

```sql
-- 7. Which item was purchased just before the customer became a member?
WITH CTE AS (
  SELECT
    S.customer_id,
    order_date,
    join_date,
    product_name,
    RANK() OVER(PARTITION BY S.customer_id ORDER BY order_date ASC) as rnk,
    ROW_NUMBER() OVER(PARTITION BY S.customer_id ORDER BY order_date) as rn
  FROM
    SALES as S
    INNER JOIN MENU as M on S.product_id = M.product_id
    INNER JOIN MEMBERS as MEM ON MEM.customer_id = S.customer_id
  WHERE
    order_date < join_date
  ORDER BY
    order_date ASC
)
SELECT
  customer_id,
  product_name
FROM
  CTE
WHERE
  rnk = 1;
```

8.

```sql
-- 8. What is the total items and amount spent for each member before they became a member?
SELECT
  S.customer_id,
  COUNT(M.product_id) as total_items,
  SUM(M.price) as amount_spent
FROM
  SALES as S
  INNER JOIN MENU as M on S.product_id = M.product_id
  INNER JOIN MEMBERS as MEM ON MEM.customer_id = S.customer_id
WHERE
  order_date < join_date
GROUP BY
  S.customer_id;

```

9.

```sql
-- 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 product_name
      WHEN 'sushi' THEN price * 10 * 2
      ELSE price * 10
    END
  ) as points
FROM
  MENU as M
  INNER JOIN SALES as S ON S.product_id = M.product_id
GROUP BY
  customer_id;
```

Thank you