## Problem Statement:

Danny is deciding whether he should expand the existing customer loyalty program. We are going to create functional SQL queries to aid his decision making.

## Entity Relationship Diagram & Dataset:

Source: https://8weeksqlchallenge.com/case-study-1/

## Inspiration: https://www.datawithdanny.com/

/\* --------------------

Case Study Questions

--------------------\*/

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

-- left join sales with menu. We use left join to reserve the left row of sales and keep customer\_id who has been making purchases

-- use sum() to compute the total amount each customer spent

**select** s.customer\_id, **sum**(price) **as** amount\_spent

**from** dannys\_diner.sales s

**left join** dannys\_diner.menu m **on** s.product\_id = m.product\_id

**group by** s.customer\_id

**order by** amount\_spent desc;

**Output:**



Looks like customer A, and customer B spent the most.

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

**select** s.customer\_id, **count**(**distinct** order\_date) total\_visit

**from** dannys\_diner.sales s

**left join** dannys\_diner.members m **on** s.customer\_id = m.customer\_id

**left join** dannys\_diner.menu mu **on** mu.product\_id = s.product\_id

**group by** s.customer\_id

**order by** total\_visit **desc**;

**Output:**

**A picture containing table

Description automatically generated**

Customer B and A visited the restaurant more frequently than others.

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

////

modifying

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

/// modifying

with sales\_date

as

(select customer\_id, min(date(order\_date)) as dt

from dannys\_diner.sales

group by 1)

select s.customer\_id, s.order\_date

from dannys\_diner.sales s

inner join sales\_date sd

on s.customer\_id = sd.customer\_id

and s.order\_date = sd.dt

inner join dannys\_diner.menu m

on m.product\_id = s.product\_id;

-- 5. Which item was the most popular for each customer?

/// modifying

with

purchase as

(select s.customer\_id, m.product\_name, count(s.product\_id) as total

from dannys\_diner.sales s

inner join dannys\_diner.menu m

on m.product\_id = s.product\_id

group by 1, 2

order by 1),

pur as

(select customer\_id, max(total) as pop

from purchase

group by customer\_id),

final as

(select p.customer\_id, p.product\_name, pi.pop

from purchase p

inner join pur pi

on p.customer\_id = pi.customer\_id

and p.total = pi.pop)

select \* from final;

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

with

cte as

(select s.customer\_id, s.order\_date, s.product\_id, m.join\_date,

min(order\_date) over(partition by s.customer\_id) as first\_date

from dannys\_diner.sales s

inner join dannys\_diner.members m

on s.customer\_id = m.customer\_id

where m.join\_date <= s.order\_date)

select c.customer\_id, order\_date, first\_date, product\_name

from cte c

left join dannys\_diner.menu m

on c.product\_id = m.product\_id

where order\_date = first\_date

order by c.customer\_id;

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

-- find the product bought before customer become member

**with** cte

**as**

(**select** s.customer\_id, s.order\_date, s.product\_id,

**min**(order\_date) **over**(**partition** **by** s.customer\_id) **as** first\_order\_date

**from** dannys\_diner.sales s

**inner join** dannys\_diner.members m **on** s.customer\_id = m.customer\_id

**where** order\_date < join\_date)

**select** customer\_id, product\_name

**from** cte c

**left join** dannys\_diner.menu m **on** c.product\_id = m.product\_id

**where** order\_date = first\_order\_date

**order** **by** c.customer\_id;

**Output:**

Table

Description automatically generated

Customer 1 ordered sushi and curry, whereas customer 2 ordered curry before he/she became the member.

Ah ha! Sushi and curry are the favorite items in the restaurant

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

**with**

sales\_members **as**

(**select** m.customer\_id, s.product\_id

**from** dannys\_diner.members m

**inner join** dannys\_diner.sales s **on** m.customer\_id = s.customer\_id

**where** order\_date < join\_date)

**select** c.customer\_id, **count**(m.product\_id), **sum**(price)

**from** sales\_members c

**left join** dannys\_diner.menu m **on** c.product\_id = m.product\_id

**group by** c.customer\_id

**order by** c.customer\_id;

**output:**

Table

Description automatically generated

Customer B bought 3 items and spent 40. customer A spent less at 25 with 2 items.

Ah ha! Customer B could be a very potential customer

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

-- create a case statement to compute the point

**select** s.customer\_id,

**sum**(**case**

**when** m.product\_name = 'sushi' **then** m.price \* 10 \* 2

**else** m.price \*10

**end**) as point

**from** dannys\_diner.sales s

**inner join** dannys\_diner.menu m **on** s.product\_id = m.product\_id

**group by** s.customer\_id

**order by** customer\_id;

**Output:**

We got the total point for client A is 860, client B is 940, client C is 360

Background pattern, table

Description automatically generated

-- 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?

-- Query:

-- create a timeline table that contains join\_date, and first\_week(after become member)

-- left join timeline table with sales and menu

-- create a case statement to compute the point

**with**

timeline **as**

(**select**

customer\_id,

join\_date,

join\_date + 6 as first\_week,

**extract**(month from join\_date) as month

**from** dannys\_diner.members)

**select** t.customer\_id,

sum(**case**

**when** product\_name = 'sushi' **then** (price \* 2 \* 10)

**when** order\_date **between** join\_date

**and** first\_week then (price \* 10 \* 2)

**else** price \* 10

**end**) as january\_points

**from** timeline t

**left join** dannys\_diner.sales s **on** s.customer\_id = t.customer\_id

**left join** dannys\_diner.menu m **on** m.product\_id = s.product\_id

**where** month = 1

**group by** t.customer\_id

**order by** t.customer\_id;

**output:**

Background pattern

Description automatically generated with medium confidence