**Case Study #1 - Danny's Diner**

**This is the solution for 1st case study of the challenge**

**CREATE** **SCHEMA** dannys\_diner;

**SET** search\_path = dannys\_diner;

**CREATE** **TABLE** sales (

"customer\_id" VARCHAR(1),

"order\_date" DATE,

"product\_id" INTEGER

);

**INSERT** **INTO** sales

("customer\_id", "order\_date", "product\_id")

**VALUES**

('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');

**CREATE** **TABLE** menu (

"product\_id" INTEGER,

"product\_name" VARCHAR(5),

"price" INTEGER

);

**INSERT** **INTO** menu

("product\_id", "product\_name", "price")

**VALUES**

('1', 'sushi', '10'),

('2', 'curry', '15'),

('3', 'ramen', '12');

**CREATE** **TABLE** members (

"customer\_id" VARCHAR(1),

"join\_date" DATE

);

**INSERT** **INTO** members

("customer\_id", "join\_date")

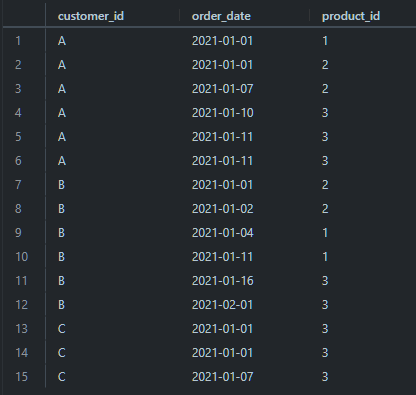
**VALUES**

('A', '2021-01-07'),

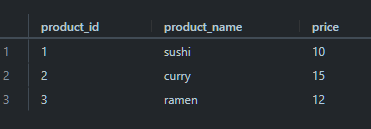
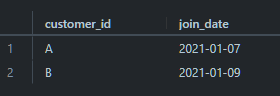
('B', '2021-01-09');

**Input Tables:**

**Sales table**



**Menu Table Members Table**

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

WITH CustomerSpending AS (

SELECT s.customer\_id, SUM(m.price) AS amount\_spent

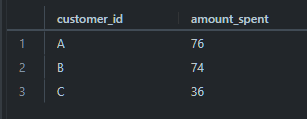
FROM sales s

INNER JOIN menu m ON s.product\_id = m.product\_id

GROUP BY s.customer\_id )

SELECT \* FROM CustomerSpending

ORDER BY amount\_spent DESC;



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

WITH CTE AS (

SELECT customer\_id, EXTRACT(DAY FROM order\_date) AS day\_visited

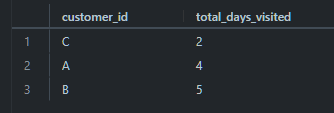
FROM sales )

SELECT customer\_id, COUNT(DISTINCT day\_visited) AS total\_days\_visited

FROM CTE

GROUP BY customer\_id

ORDER BY total\_days\_visited DESC;



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

WITH FirstPurchase AS (

SELECT customer\_id,

MIN(order\_date) AS first\_order\_date FROM sales

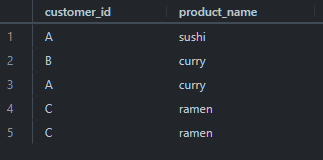
GROUP BY customer\_id )

SELECT F.customer\_id, M.product\_name

FROM FirstPurchase F

JOIN sales S ON F.customer\_id = S.customer\_id AND F.first\_order\_date = S.order\_date

JOIN menu M ON S.product\_id = M.product\_id;



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

WITH CTE AS (

SELECT customer\_id, product\_id, COUNT(\*) AS total\_purchases

FROM sales

GROUP BY customer\_id, product\_id )

SELECT M.product\_name, CTE.total\_purchases

FROM CTE

INNER JOIN menu M ON M.product\_id = CTE.product\_id

ORDER BY CTE.total\_purchases DESC

LIMIT 1;



1. Which item was the most popular for each customer?

WITH CTE AS (

SELECT customer\_id, product\_id, COUNT(\*) AS total

FROM sales

GROUP BY customer\_id, product\_id )

SELECT C.customer\_id, M.product\_name

FROM (

SELECT customer\_id, product\_id, total,

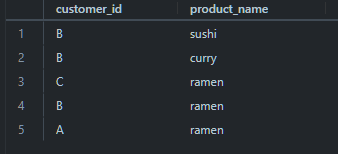
RANK() OVER (PARTITION BY customer\_id ORDER BY total DESC) AS rnk

FROM CTE

) AS C

JOIN menu M ON M.product\_id = C.product\_id

WHERE rnk = 1;



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

WITH RANK AS

(SELECT S.CUSTOMER\_ID,

M.PRODUCT\_NAME,

DENSE\_RANK() OVER (PARTITION BY S.CUSTOMER\_ID

ORDER BY S.ORDER\_DATE) AS RNK

FROM SALES S

JOIN MENU M ON M.PRODUCT\_ID = S.PRODUCT\_ID

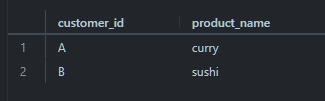
JOIN MEMBERS MEM ON MEM.CUSTOMER\_ID = S.CUSTOMER\_ID

WHERE S.ORDER\_DATE >= MEM.JOIN\_DATE )

SELECT R.CUSTOMER\_ID, R.PRODUCT\_NAME

FROM RANK R

WHERE RNK = 1;



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

WITH RANK AS

(SELECT S.CUSTOMER\_ID,

M.PRODUCT\_NAME,

DENSE\_RANK() OVER (PARTITION BY S.CUSTOMER\_ID

ORDER BY S.ORDER\_DATE) AS RNK

FROM SALES S

JOIN MENU M ON M.PRODUCT\_ID = S.PRODUCT\_ID

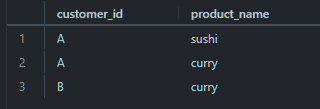
JOIN MEMBERS MEM ON MEM.CUSTOMER\_ID = S.CUSTOMER\_ID

WHERE S.ORDER\_DATE < MEM.JOIN\_DATE )

SELECT R.CUSTOMER\_ID,R.PRODUCT\_NAME

FROM RANK R

WHERE RNK = 1;



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

SELECT S.CUSTOMER\_ID,

COUNT(\*) AS TOTAL\_ITEMS,

SUM(M.PRICE) AS AMOUNT\_SPENT

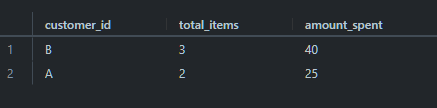
FROM SALES S

JOIN MEMBERS MS ON MS.CUSTOMER\_ID = S.CUSTOMER\_ID

AND S.ORDER\_DATE < MS.JOIN\_DATE

JOIN MENU M ON M.PRODUCT\_ID = S.PRODUCT\_ID

GROUP BY 1;



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

with pointscte as (

select \*, case when product\_id = 1 then 20\*price

                else 10\*price end as points

                from menu)

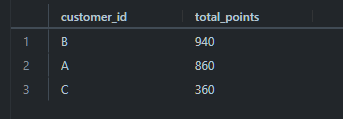
select s.customer\_id , sum(pc.points) as total\_points  from pointscte pc

right join sales s

on s.product\_id = pc.product\_id

group by  s.customer\_id

order by total\_points desc;

****

1. **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?**

**SELECT**

    s.customer\_id,

    SUM(

        CASE

            WHEN (DATE\_TRUNC('day', s.order\_date) - DATE\_TRUNC('day', me.join\_date) BETWEEN '0 days' AND '7 days') OR (m.product\_id = 1) THEN m.price \* 20

            ELSE m.price \* 10

        END

    ) AS Points

FROM members AS me

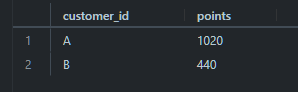
INNER JOIN sales AS s ON s.customer\_id = me.customer\_id

INNER JOIN menu AS m ON m.product\_id = s.product\_id

WHERE s.order\_date >= me.join\_date AND s.order\_date <= '2021-01-31'

GROUP BY s.customer\_id

order by 1;



**Note : you can use date\_trunc, extract and date\_part functions also**

## **Bonus Questions**

### Join All The Things

The following questions are related creating basic data tables that Danny and his team can use to quickly derive insights without needing to join the underlying tables using SQL.

Recreate the following table output using the available data:

CREATE TABLE customer\_orders AS

SELECT

s.customer\_id,

s.order\_date,

m.product\_name,

m.price,

CASE WHEN mem.customer\_id IS NOT NULL THEN 'Y' ELSE 'N' END AS member

FROM

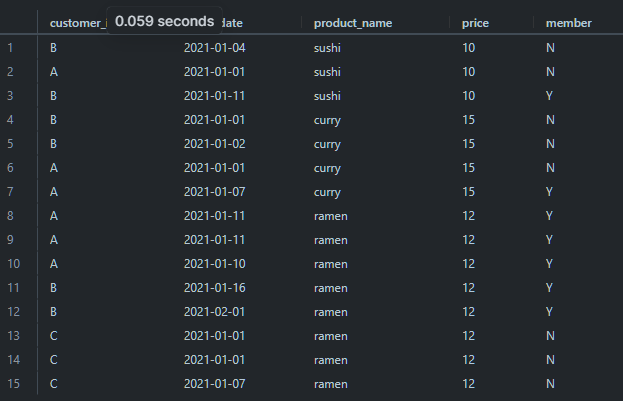
sales s

JOIN

menu m ON s.product\_id = m.product\_id

LEFT JOIN

members mem ON s.customer\_id = mem.customer\_id AND s.order\_date >= mem.join\_date;

****

### **Rank All The Things**

Danny 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.

**CREATE TABLE customer\_product\_ranking AS**

**WITH RankedProducts AS (**

**SELECT s.customer\_id, s.order\_date, m.product\_name,**

**m.price, CASE WHEN mem.customer\_id IS NOT NULL THEN 'Y' ELSE 'N'**

**END AS member,**

**ROW\_NUMBER() OVER (PARTITION BY s.customer\_id, mem.customer\_id ORDER BY s.order\_date) AS ranking**

**FROM**

**sales s**

**JOIN**

**menu m ON s.product\_id = m.product\_id**

**LEFT JOIN**

**members mem ON s.customer\_id = mem.customer\_id AND s.order\_date >= mem.join\_date**

**)**

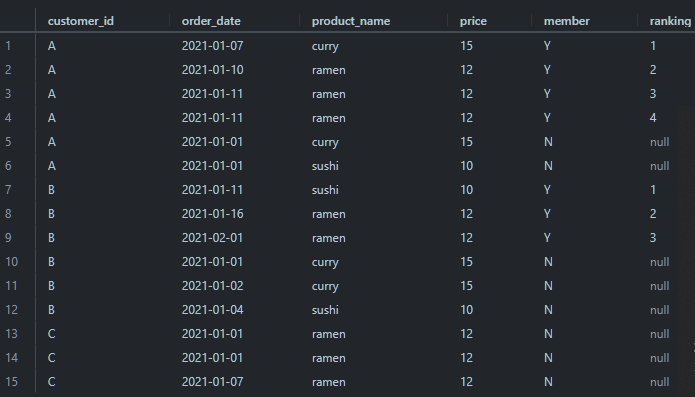
**SELECT customer\_id, order\_date, product\_name, price, member,**

**CASE WHEN member = 'Y' THEN ranking ELSE NULL**

**END AS ranking**

**FROM**

**RankedProducts;**

****