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.

Created By Gaurav kathane
Linkdin- http://www.linkedin.com/in/gaurav-kathane-1a055b250/

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

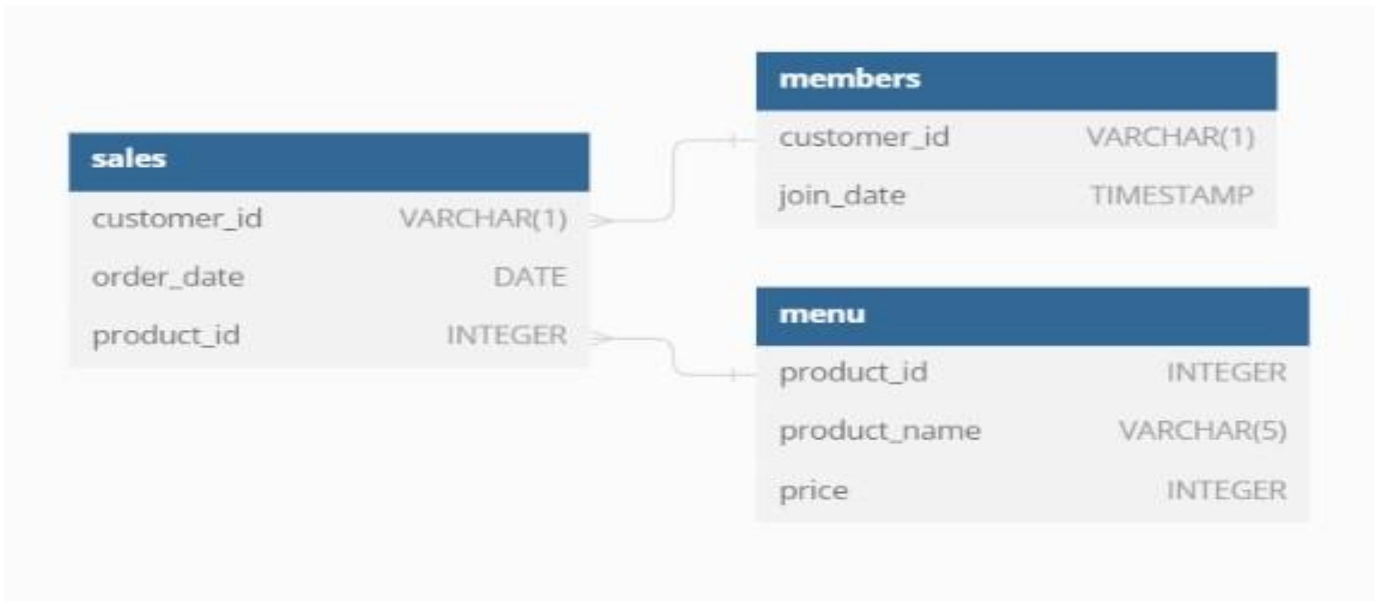


Table Creation

```
--Create Database DEMO_DB;
USE DEMO_DB;

CREATE SCHEMA dannys_dinner;
USE dannys_dinner;

--Creating sales Table

Create or replace table sales(
customer_id varchar(2),
order_date date,
product_id int
);



--Inserting data  into the sales table

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');
```

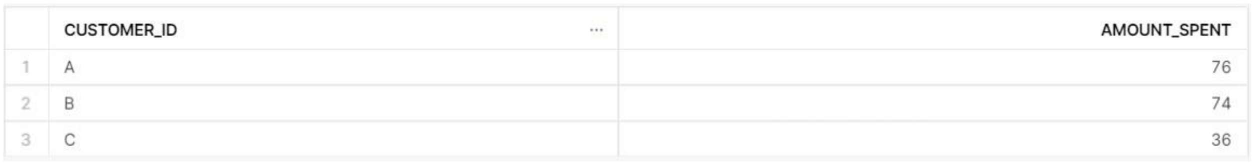
```
-- Creating Menu Table

CREATE OR REPLACE TABLE menu (
  product_id INTEGER,
  product_name VARCHAR(5),
  price INTEGER
);

--- Inserting data into the Menu Table

INSERT INTO menu
  (product_id, product_name, price)
VALUES
  ('1', 'sushi', '10'),
  ('2', 'curry', '15'),
  ('3', 'ramen', '12');


--- Creating Members Table

CREATE OR REPLACE TABLE members (
  customer_id VARCHAR(1),
  join_date DATE
);

--Inserting data into the Members Table

INSERT INTO members
  (customer_id, join_date)
VALUES
  ('A', '2021-01-07'),
  ('B', '2021-01-09');

  SELECT * FROM SALES;   --15 ROWS
  SELECT * FROM MENU;  --3 ROWS
  SELECT * FROM MEMBERS; --2 ROWS
```

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 the restaurant?

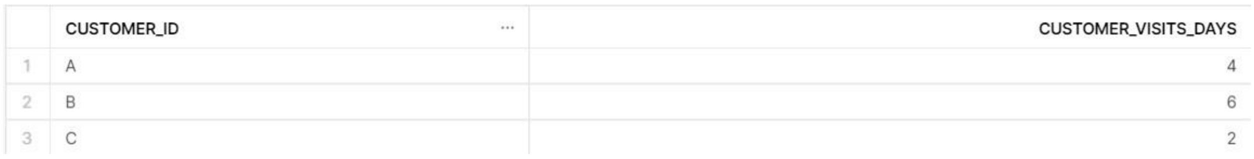
```
SELECT CUSTOMER_ID,SUM(PRICE) AS Amount_Spent
FROM SALES AS S
INNER JOIN MENU AS M ON S.PRODUCT_ID = M.PRODUCT_ID
GROUP BY CUSTOMER_ID
ORDER BY 2 DESC;
```



	CUSTOMER_ID	AMOUNT_SPENT
1	A	76
2	B	74
3	C	36

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

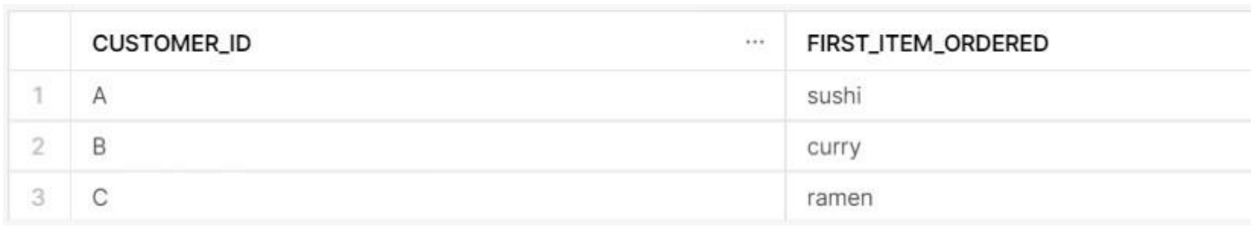
```
SELECT CUSTOMER_ID, COUNT(DISTINCT ORDER_DATE) AS
CUSTOMER_VISITS_DAYS
FROM SALES
GROUP BY 1;
```



	CUSTOMER_ID	CUSTOMER_VISITS_DAYS
1	A	4
2	B	6
3	C	2

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

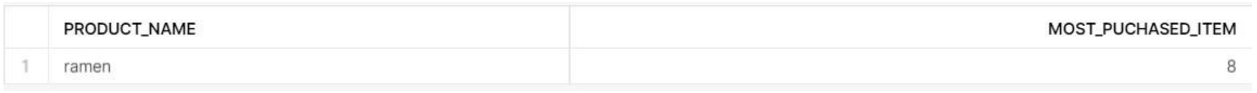
```
 SELECT CUSTOMER_ID,PRODUCT_NAME  AS FIRST_ITEM_ORDERED
 FROM
 (SELECT CUSTOMER_ID,ORDER_DATE ,PRODUCT_NAME,
 ROW_NUMBER() OVER(PARTITION BY CUSTOMER_ID ORDER BY
ORDER_DATE) AS PURCHASE_NUM
 FROM SALES AS S LEFT JOIN MENU AS M ON
S.PRODUCT_ID=M.PRODUCT_ID)
WHERE PURCHASE_NUM = 1;
```



	CUSTOMER_ID	FIRST_ITEM_ORDERED
1	A	sushi
2	B	curry
3	C	ramen

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

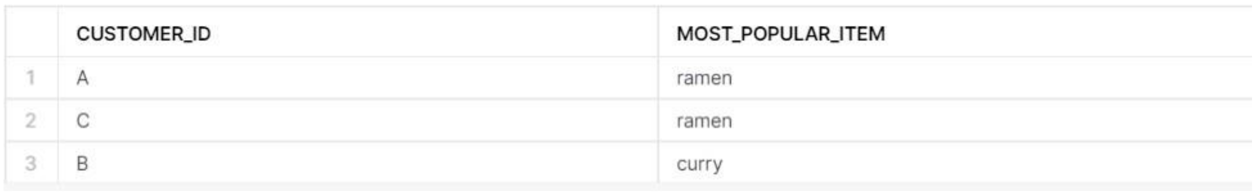
```
SELECT M.PRODUCT_NAME,COUNT(S.PRODUCT_ID)AS
Most_puchased_item
FROM SALES AS S
LEFT JOIN MENU AS M ON S.PRODUCT_ID=M.PRODUCT_ID
GROUP BY 1
ORDER BY 2 DESC
LIMIT 1;
```



	PRODUCT_NAME	MOST_PUCHASED_ITEM
1	ramen	8

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

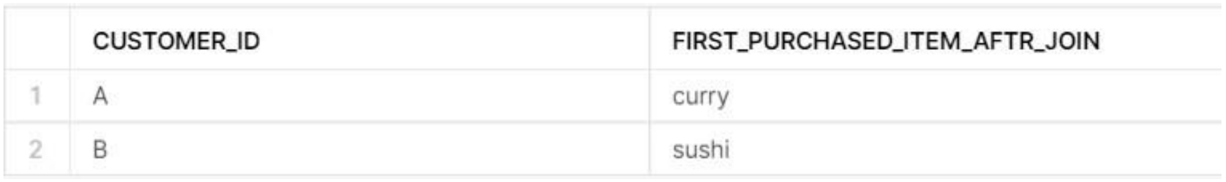
```
SELECT CUSTOMER_ID,PRODUCT_NAME AS Most_Popular_item
FROM
(SELECT S.CUSTOMER_ID,M.PRODUCT_NAME , COUNT(*) AS Total_Sales,
ROW_NUMBER() OVER(PARTITION BY S.CUSTOMER_ID ORDER BY
COUNT(*) DESC) AS RN
FROM SALES AS S
LEFT JOIN MENU AS M ON S.PRODUCT_ID = M.PRODUCT_ID
GROUP BY 1,2)
WHERE RN = 1;
```



	CUSTOMER_ID	MOST_POPULAR_ITEM
1	A	ramen
2	C	ramen
3	B	curry

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

```
SELECT CUSTOMER_ID,PRODUCT_NAME AS
First_Purchased_Item_Aftr_Join
FROM
(SELECT
S.CUSTOMER_ID,MN.PRODUCT_NAME,M.JOIN_DATE,S.ORDER_DATE,
ROW_NUMBER() OVER(PARTITION BY S.CUSTOMER_ID ORDER BY
MIN(S.ORDER_DATE) ) AS RN
FROM SALES S
LEFT JOIN MEMBERS M ON S.CUSTOMER_ID=M.CUSTOMER_ID
LEFT JOIN MENU MN  ON S.PRODUCT_ID = MN.PRODUCT_ID
WHERE ORDER_DATE >= JOIN_DATE
GROUP BY 1,2,3,4)
WHERE RN =1;
```

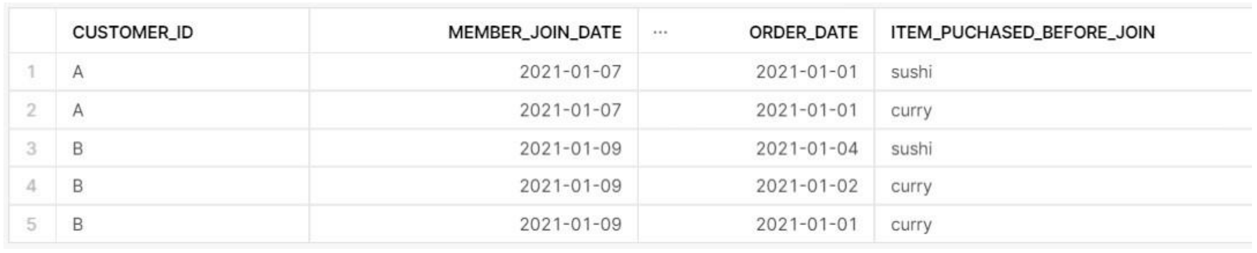


	CUSTOMER_ID	FIRST_PURCHASED_ITEM_AFTR_JOIN
1	A	curry
2	B	sushi

Created By Gaurav kathane
Linkdin- http://www.linkedin.com/in/gaurav-kathane-1a055b250/

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

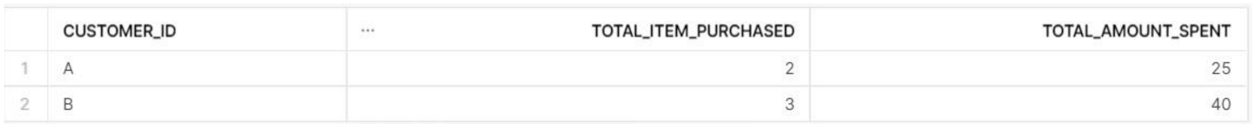
```
SELECT CUSTOMER_ID,JOIN_DATE AS Member_Join_Date,ORDER_DATE
,PRODUCT_NAME As Item_Puchased_Before_join
FROM
(SELECT
S.CUSTOMER_ID,MN.PRODUCT_NAME,M.JOIN_DATE,S.ORDER_DATE,
ROW_NUMBER() OVER(PARTITION BY S.CUSTOMER_ID ORDER BY
S.ORDER_DATE ) AS RN
FROM SALES S
LEFT JOIN MEMBERS M ON S.CUSTOMER_ID=M.CUSTOMER_ID
LEFT JOIN MENU MN  ON S.PRODUCT_ID = MN.PRODUCT_ID
WHERE ORDER_DATE < JOIN_DATE
GROUP BY 1,2,3,4)
ORDER BY 1;
```



	CUSTOMER_ID	MEMBER_JOIN_DATE	···	ORDER_DATE	ITEM_PUCHASED_BEFORE_JOIN
1	A	2021-01-07		2021-01-01	sushi
2	A	2021-01-07		2021-01-01	curry
3	B	2021-01-09		2021-01-04	sushi
4	B	2021-01-09		2021-01-02	curry
5	B	2021-01-09		2021-01-01	curry

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

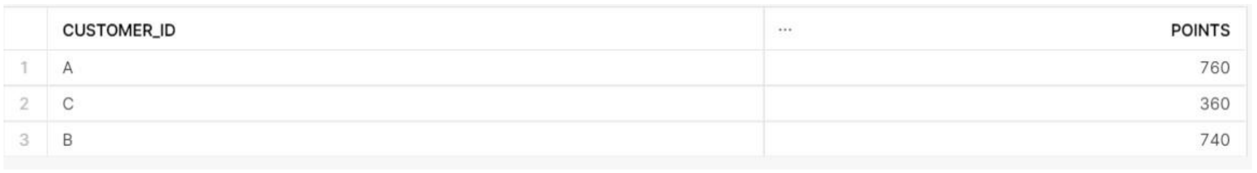
```
SELECT S.CUSTOMER_ID,COUNT(S.PRODUCT_ID)
TOTAL_ITEM_PURCHASED,SUM(M.PRICE) TOTAL_AMOUNT_SPENT
FROM SALES S
LEFT JOIN MENU M ON S.PRODUCT_ID = M.PRODUCT_ID
LEFT JOIN MEMBERS MM ON S.CUSTOMER_ID=MM.CUSTOMER_ID
WHERE S.ORDER_DATE < MM.JOIN_DATE
GROUP BY 1;
```



	CUSTOMER_ID	···	TOTAL_ITEM_PURCHASED	TOTAL_AMOUNT_SPENT
1	A		2	25
2	B		3	40

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

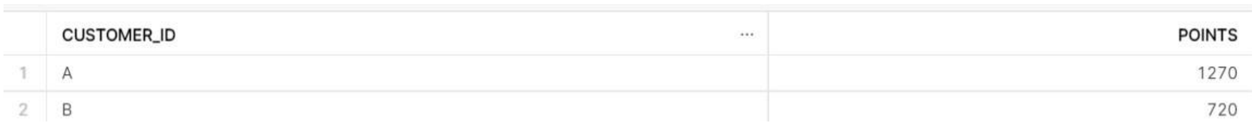
```
SELECT CUSTOMER_ID , SUM(
CASE PRODUCT_NAME
WHEN 'SUSHI' THEN (PRICE * 10)*2 ELSE PRICE * 10
END) AS  POINTS
FROM SALES S
LEFT JOIN MENU M ON S.PRODUCT_ID=M.PRODUCT_ID
GROUP BY 1;
```



	CUSTOMER_ID	...	POINTS
1	A		760
2	C		360
3	B		740

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?

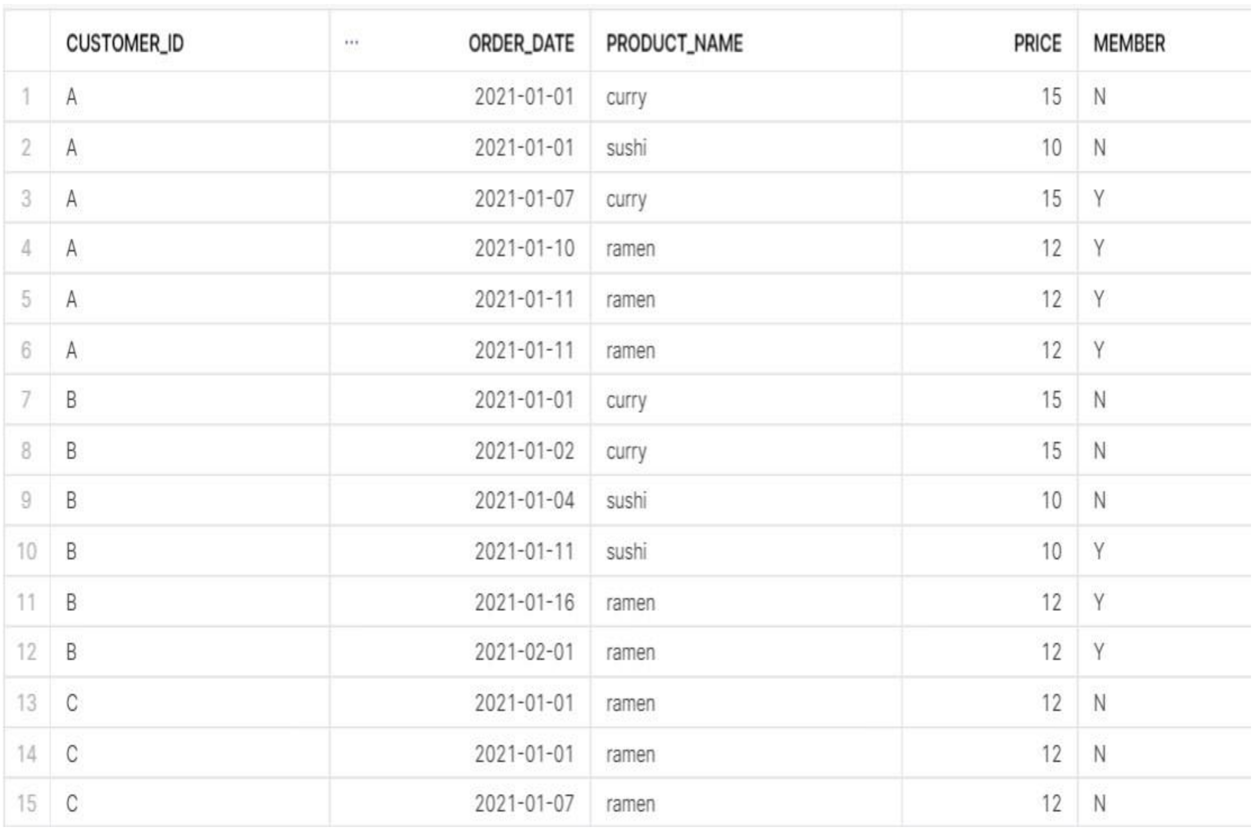
```
SELECT S.CUSTOMER_ID,
SUM(
CASE
   WHEN S.ORDER_DATE BETWEEN MM.JOIN_DATE AND
DATEADD('DAY',6,MM.JOIN_DATE) THEN PRICE * 10 *2
   WHEN M.PRODUCT_NAME = 'SHUSHI' THEN PRICE*10*2
   ELSE PRICE *10
   END
)AS POINTS
FROM SALES S
LEFT JOIN MENU M ON S.PRODUCT_ID=M.PRODUCT_ID
LEFT JOIN MEMBERS MM ON S.CUSTOMER_ID=MM.CUSTOMER_ID
WHERE
   DATE_TRUNC('MONTH',S.ORDER_DATE) = '2021-01-01'
GROUP BY S.CUSTOMER_ID
ORDER BY 1 ASC;
```



	CUSTOMER_ID	POINTS
1	A	1270
2	B	720

Join All The Things

```
CREATE VIEW Master_Table as
(SELECT S.CUSTOMER_ID,S.ORDER_DATE,M.PRODUCT_NAME,M.PRICE,
CASE
    WHEN MM.JOIN_DATE <=S.ORDER_DATE THEN 'Y'
    ELSE 'N'
    END AS MEMBER
FROM SALES S
LEFT JOIN MENU AS M ON S.PRODUCT_ID = M.PRODUCT_ID
LEFT JOIN MEMBERS AS MM ON S.CUSTOMER_ID = MM.CUSTOMER_ID
ORDER BY 1,2,4 DESC);
```

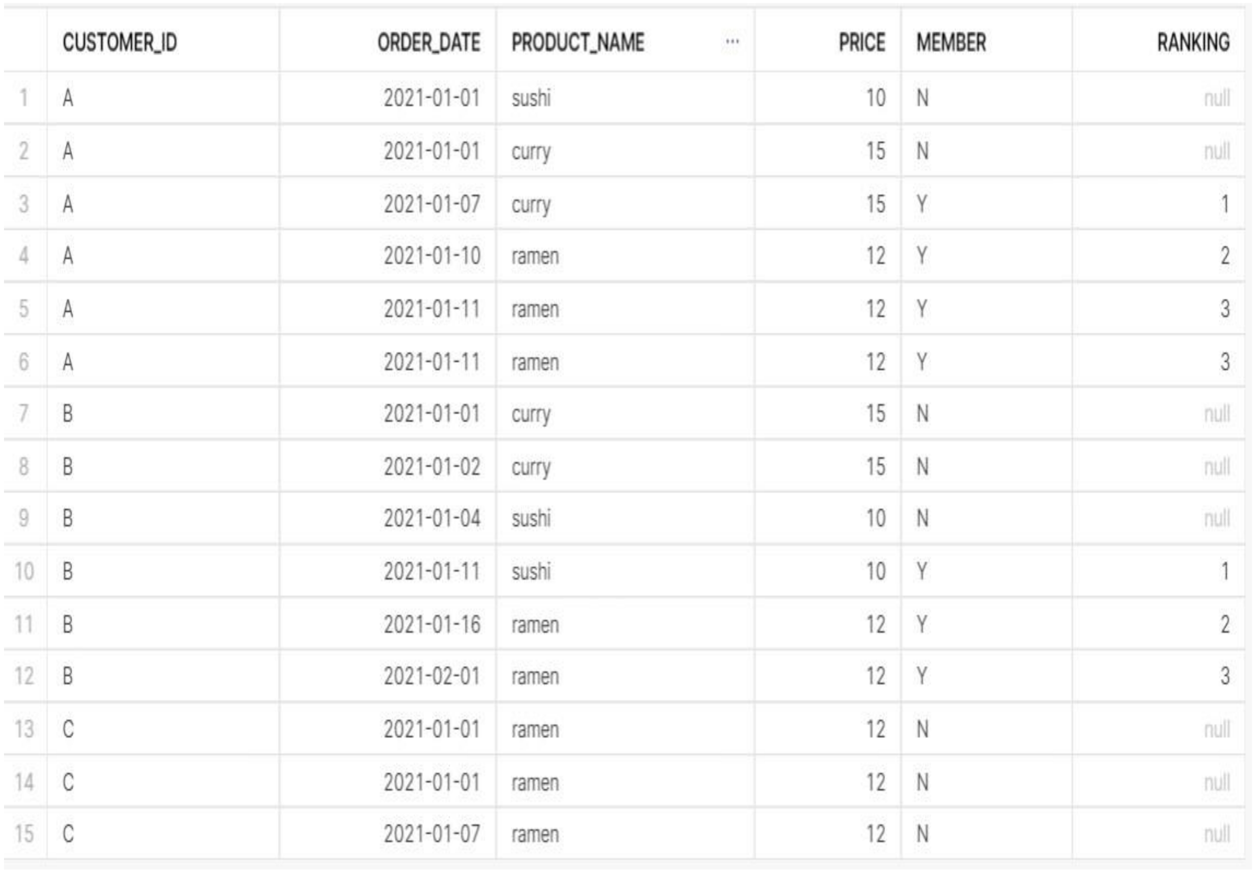


	CUSTOMER_ID	ORDER_DATE	PRODUCT_NAME	PRICE	MEMBER
1	A	2021-01-01	curry	15	N
2	A	2021-01-01	sushi	10	N
3	A	2021-01-07	curry	15	Y
4	A	2021-01-10	ramen	12	Y
5	A	2021-01-11	ramen	12	Y
6	A	2021-01-11	ramen	12	Y
7	B	2021-01-01	curry	15	N
8	B	2021-01-02	curry	15	N
9	B	2021-01-04	sushi	10	N
10	B	2021-01-11	sushi	10	Y
11	B	2021-01-16	ramen	12	Y
12	B	2021-02-01	ramen	12	Y
13	C	2021-01-01	ramen	12	N
14	C	2021-01-01	ramen	12	N
15	C	2021-01-07	ramen	12	N

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.

```
SELECT *,(
CASE
   WHEN MEMBER = 'N' THEN NULL
   ELSE RANK() OVER(PARTITION BY CUSTOMER_ID,MEMBER ORDER
ORDER_DATE)
END ) AS RANKING
FROM MASTER_TABLE;
```



	CUSTOMER_ID	ORDER_DATE	PRODUCT_NAME	PRICE	MEMBER	RANKING
1	A	2021-01-01	sushi	10	N	null
2	A	2021-01-01	curry	15	N	null
3	A	2021-01-07	curry	15	Y	1
4	A	2021-01-10	ramen	12	Y	2
5	A	2021-01-11	ramen	12	Y	3
6	A	2021-01-11	ramen	12	Y	3
7	B	2021-01-01	curry	15	N	null
8	B	2021-01-02	curry	15	N	null
9	B	2021-01-04	sushi	10	N	null
10	B	2021-01-11	sushi	10	Y	1
11	B	2021-01-16	ramen	12	Y	2
12	B	2021-02-01	ramen	12	Y	3
13	C	2021-01-01	ramen	12	N	null
14	C	2021-01-01	ramen	12	N	null
15	C	2021-01-07	ramen	12	N	null

Created By Gaurav kathane
Linkdin- http://www.linkedin.com/in/gaurav-kathane-1a055b250/