MY 100 DAYS SQL CHALLENGE DAY 51-53 CASE STUDY1(DANNY'S DINER)

Introduction

Danny seriously loves Japanese food, so at the beginning of 2021, he decides to embark upon a risky venture and opens a cute little restaurant that sells his three favorite foods: sushi, curry, and ramen.

Danny’s Diner needs your assistance to help the restaurant stay afloat. The restaurant has captured some basic data from its few months of operation. Still, it has no idea how to use its 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 which menu items are their favorite. This more profound connection with his customers will help him deliver a better and more personalized experience for his loyal customers.

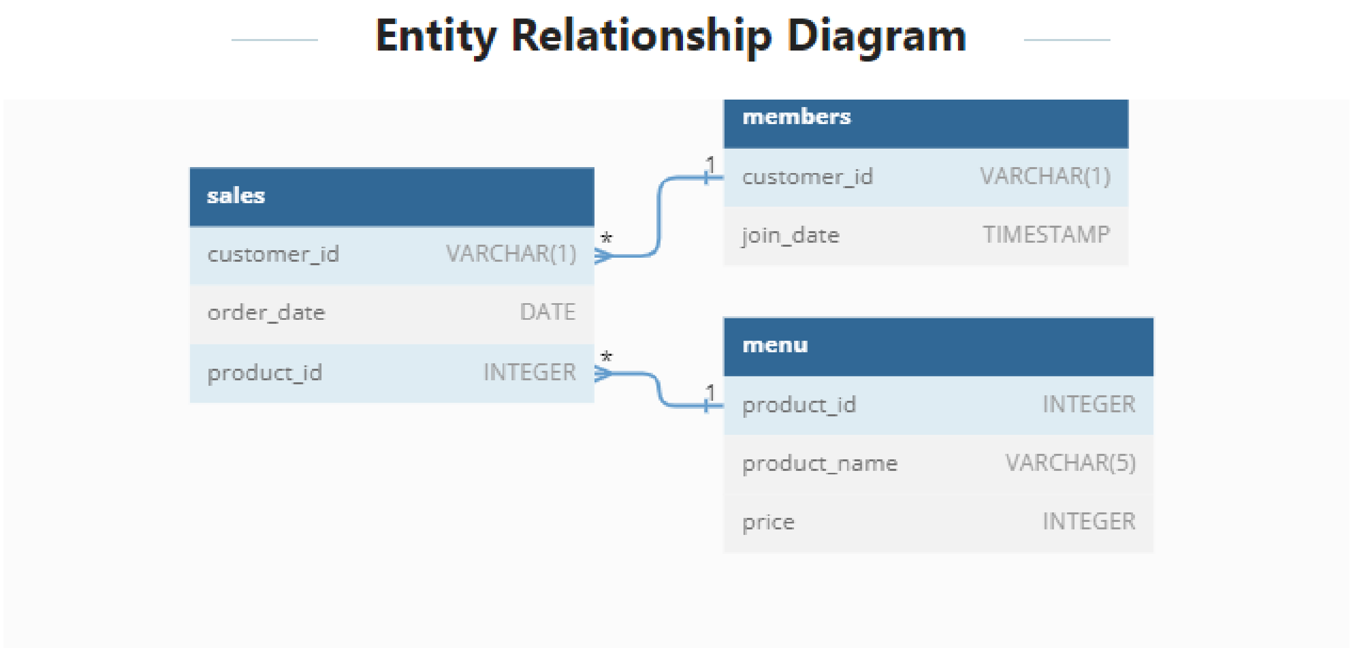
He plans to use these insights to help him decide whether to expand the existing customer loyalty program. Additionally, he needs help generating some essential datasets so his team can quickly inspect the data without using SQL.

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 three critical datasets for this case study:

- Sales
- Menu
- members

You can inspect the entity relationship diagram and example data below.



SOLUTION

To solve the problem, The MySQL database software was employed. The details of the solution are described as follows;

DATABASE SETUP

The database and table were created using the following code setup;

```
Create Database
DROP DATABASE IF EXISTS dannys_diner;
CREATE DATABASE dannys_diner;

Create Table
USE dannys_diner;
CREATE TABLE sales(
                    customer_id VARCHAR(1),
                    order_date DATE,
                    product_id INTEGER
                    );
CREATE TABLE menu (
                    product_id      INTEGER,
                    product_name    VARCHAR(5),
                    price           INTEGER
                    );
CREATE TABLE members(
                    customer_id VARCHAR(1),
                    join_date   DATE
                    );
```

Inserting Dataset to the tables

The INSERT method is employed since the data we are working with is small. The code used is as follows;

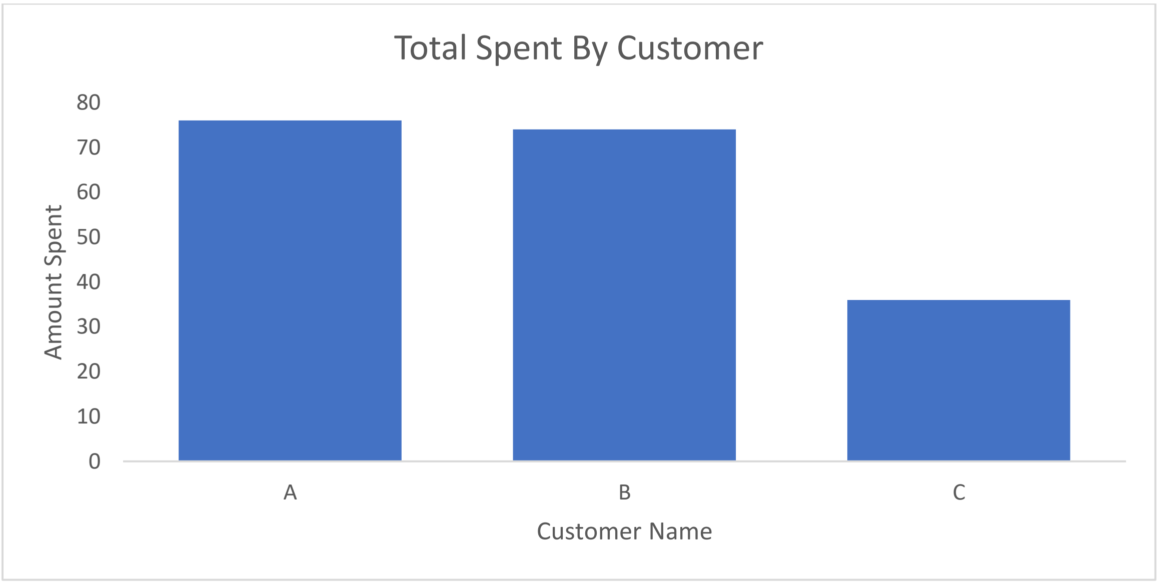
```
-- Sales Table
INSERT INTO sales
VALUES          ('A', '2021-01-01', '1'),
                ('A', '2021-01-01', '2'),
                ('A', '2021-01-07', '2'),
                ('A', '2021-01-10', '3'),
                ('A', '2021-01-11', '3'),
                ('A', '2021-01-11', '3'),
                ('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');
-- Menu Table
INSERT INTO menu
VALUES          ('1', 'sushi', '10'),
                ('2', 'curry', '15'),
                ('3', 'ramen', '12');
-- Members Table
INSERT INTO members
VALUES          ('A', '2021-01-07'),
                ('B', '2021-01-09');
```

INSIGHT GENERATION

The following queries were written to answer Dany's pertinent questions regarding the kitchen. The table retrieved was visualized using Microsoft Excel. They Include;

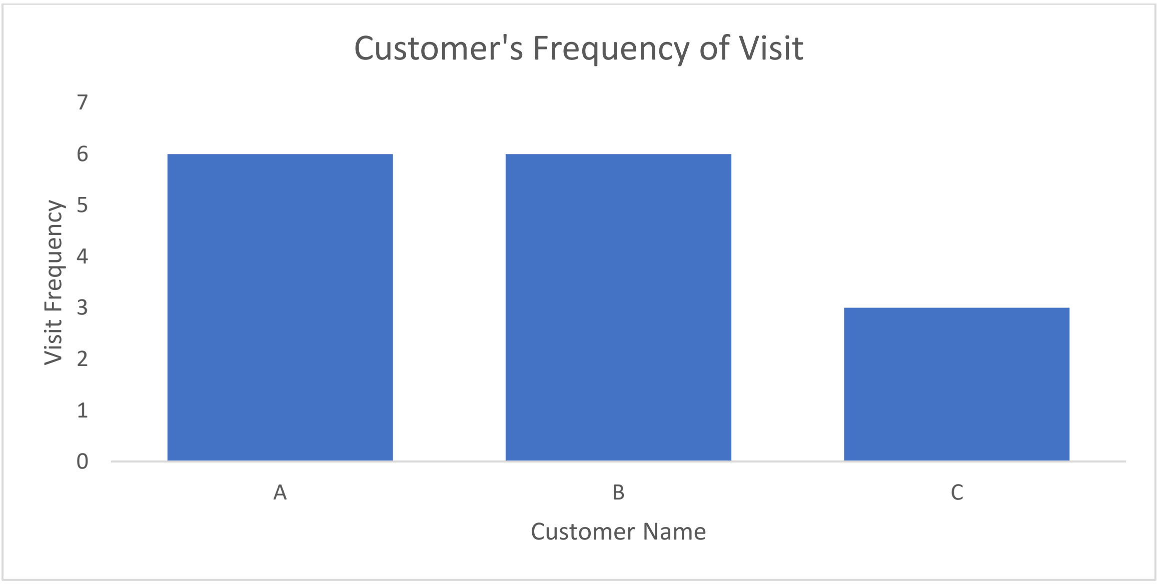
1. What is the total amount each customer spent at the restaurant?

```
SELECT sl.customer_id,
        SUM(mn.price) AS tot_spent
FROM members mm
RIGHT JOIN sales sl
ON mm.customer_id = sl.customer_id
RIGHT JOIN menu mn
ON sl.product_id = mn.product_id
GROUP BY sl.customer_id
ORDER BY sl.customer_id;
```



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

```
SELECT customer_id,
        COUNT(order_date) AS days_visited
FROM sales
GROUP BY customer_id;
```



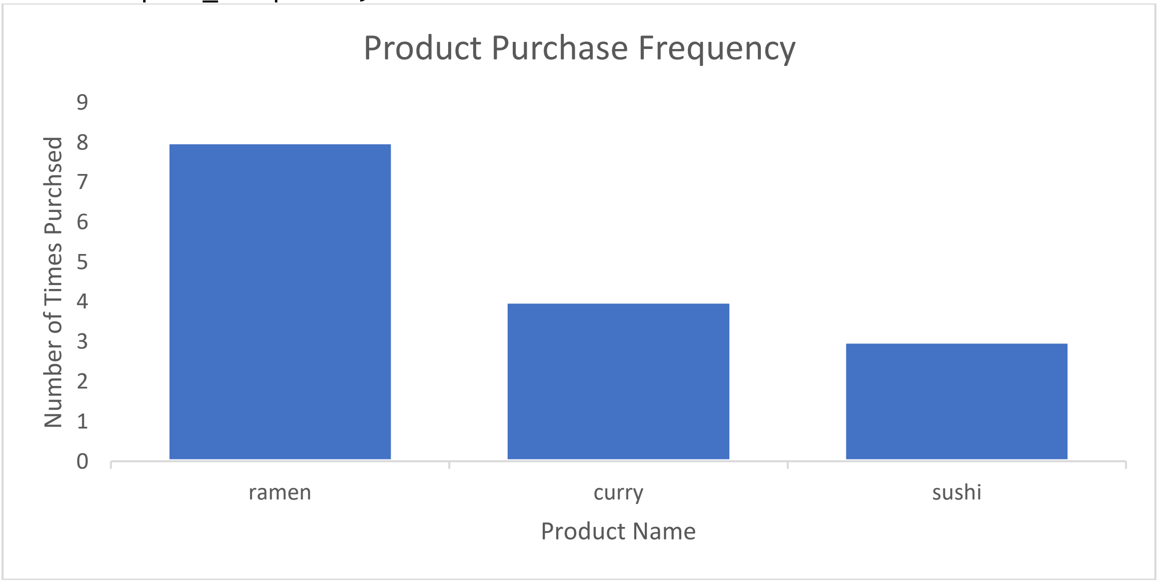
3. What was the first item from the menu purchased by each customer?

```
WITH item_order_cte AS
                    (SELECT DISTINCT sl.customer_id,
                            sl.order_date,
                            mn.product_name,
                            DENSE_RANK() OVER (PARTITION BY sl.customer_id
                                                ORDER BY sl.order_date ASC)
                                                AS date_order_rnk
                    FROM sales sl
                    LEFT JOIN menu mn
                    ON sl.product_id = mn.product_id)
SELECT customer_id, product_name
FROM item_order_cte
WHERE date_order_rnk = 1;
```

customer_id	product_name
A	sushi
A	curry
B	curry
C	ramen

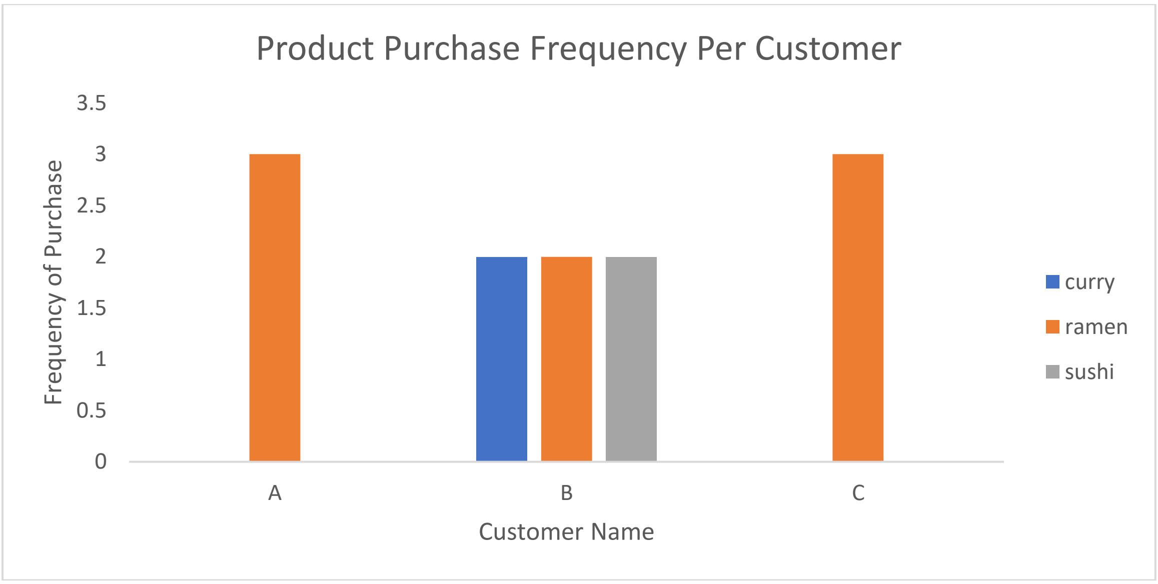
4. What is the most purchased item on the menu and how many times was it purchased by all customers?

```
SELECT mn.product_name,
        COUNT(*) AS purc_freq
FROM menu mn
JOIN sales sl
ON mn.product_id = sl.product_id
GROUP BY mn.product_name
ORDER BY purc_freq DESC;
```



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

```
WITH pop_prod_cte AS
                (SELECT sl.customer_id,
                        mn.product_name,
                        COUNT(*)AS prd_purc_freq,
                        DENSE_RANK() OVER(PARTITION BY sl.customer_id
                                     ORDER BY COUNT(*) DESC) AS freq_rank
                FROM members mm
                RIGHT JOIN sales sl
                ON mm.customer_id = sl.customer_id
                LEFT JOIN menu mn
                ON mn.product_id = sl.product_id
                GROUP BY sl.customer_id, mn.product_name)
SELECT customer_id, product_name, prd_purc_freq
FROM pop_prod_cte
WHERE freq_rank = 1;
```



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

```
WITH mmb_purc_cte AS
                    (SELECT mm.customer_id,
                            mm.join_date,
                            mn.product_name,
                            sl.order_date,
                            DENSE_RANK() OVER(PARTITION BY sl.customer_id
                                          ORDER BY order_date) AS membr_purc_rank
                    FROM members mm
                    RIGHT JOIN sales sl
                    ON mm.customer_id = sl.customer_id
                    LEFT JOIN menu mn
                    ON sl.product_id = mn.product_id
                    WHERE sl.order_date > mm.join_date)
SELECT customer_id, product_name
FROM mmb_purc_cte
WHERE membr_purc_rank =1;
```

customer_id	product_name
A	ramen
B	sushi

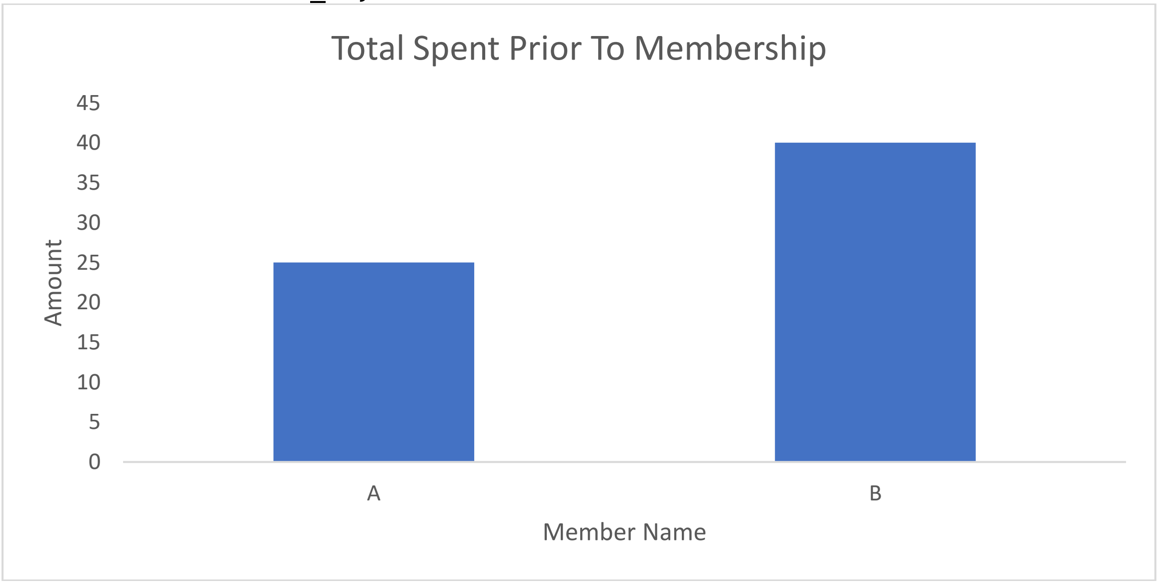
7. Which item was purchased just before the customer became a member?

```
WITH mmb_purc_cte AS
                    (SELECT mm.customer_id,
                            mm.join_date,
                            mn.product_name,
                            sl.order_date,
                            DENSE_RANK() OVER(PARTITION BY sl.customer_id
                                                  ORDER BY join_date) AS membr_purc_rank
                    FROM members mm
                    RIGHT JOIN sales sl
                    ON mm.customer_id = sl.customer_id
                    LEFT JOIN menu mn
                    ON sl.product_id = mn.product_id
                    WHERE sl.order_date < mm.join_date)
SELECT DISTINCT customer_id, product_name
FROM mmb_purc_cte
WHERE membr_purc_rank =1;
```

customer_id	product_name
A	sushi
A	curry
B	curry
B	sushi

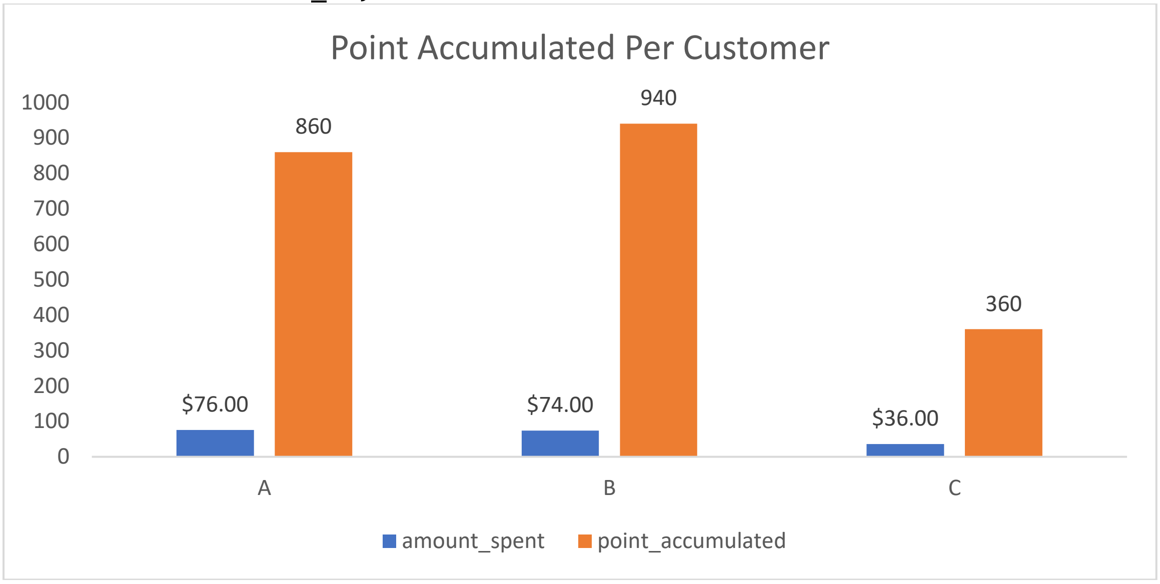
8. What is the total items and amount spent for each member before they became a member?

```
SELECT mm.customer_id,
        COUNT(sl.product_id) AS tot_item,
        SUM(mn.price) AS amnt_spent
FROM members mm
JOIN sales sl
ON mm.customer_id = sl.customer_id
JOIN menu mn
ON sl.product_id = mn.product_id
WHERE mm.join_date > sl.order_date
GROUP BY mm.customer_id
ORDER BY mm.customer_id;
```



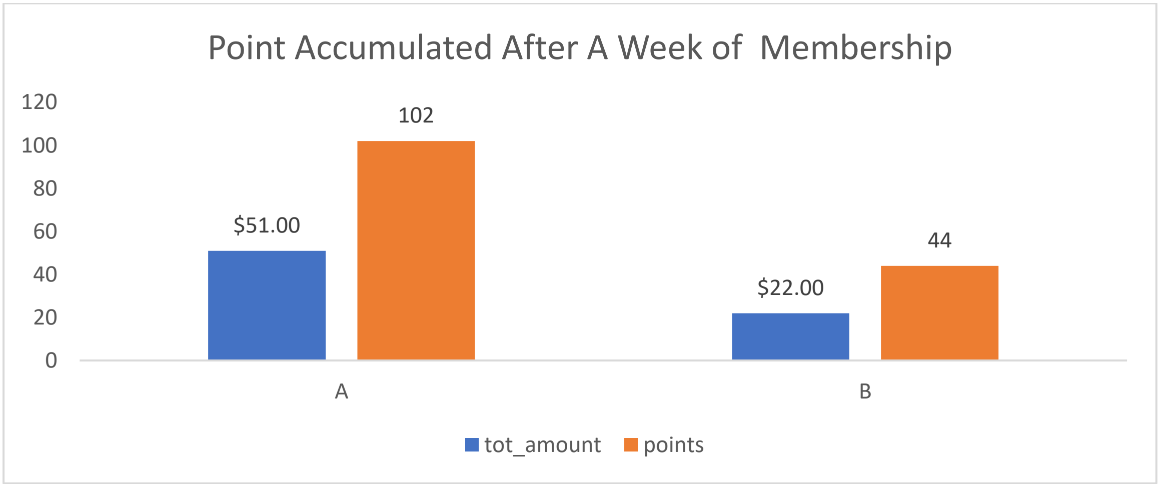
9. **If each $1 spent equates to 10 points and sushi has a 2x points multiplier how many points would each customer have?**

```
SELECT sl.customer_id,
        SUM(mn.price) AS init_price,
        SUM(CASE WHEN mn.product_name = 'sushi'
                THEN (mn.price*20)
            ELSE (mn.price*10) END) AS rev_points
FROM sales sl
JOIN menu mn
ON sl.product_id = mn.product_id
GROUP BY sl.customer_id;
```



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 customers A and B have at the end of January?**

```
SELECT mm.customer_id,
        mm.join_date,
        SUM(mn.price) AS tot_amount,
        SUM(mn.price)*2 AS points
FROM members mm
JOIN sales sl
ON mm.customer_id = sl.customer_id
JOIN menu mn
ON sl.product_id = mn.product_id
WHERE sl.order_date BETWEEN mm.join_date AND mm.join_date+7
GROUP BY mm.customer_id, mm.join_date
ORDER BY mm.customer_id;
```


Point Accumulated After A Week of Membership
$51.00
$22.00
A
B
tot_amount
points