#8weekSQLchallenge

SQL Case Study1: Danny’s Dinner – The taste of success

Background

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.

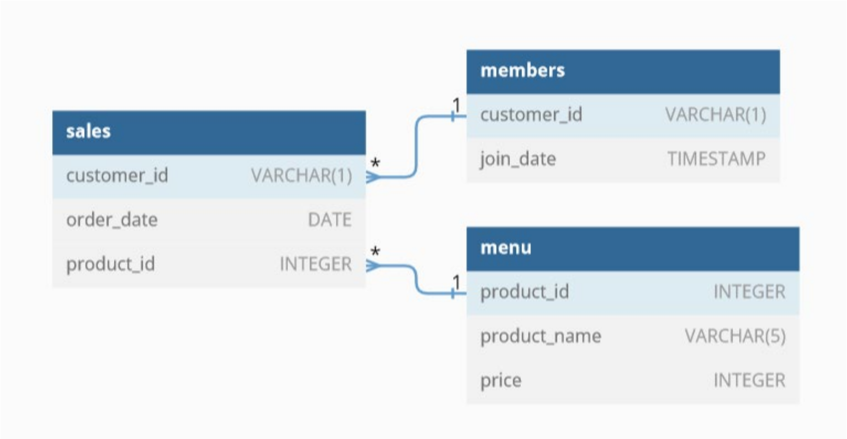
Danny’s Diner is in need of your assistance to help the restaurant stay afloat - the restaurant has captured some very basic data from their few months of operation but have no idea how to use their data to help them run the business.

Problem statement

Danny wants to use the data to answer questions about his costumers, to analyze patters and using insights to take decisions about loyalty program and improve the experience for his loyal customers.

Available information about his costumer data:

Entity Relationship DIAGRAM



sales	
customer_id	VARCHAR(1)
order_date	DATE
product_id	INTEGER

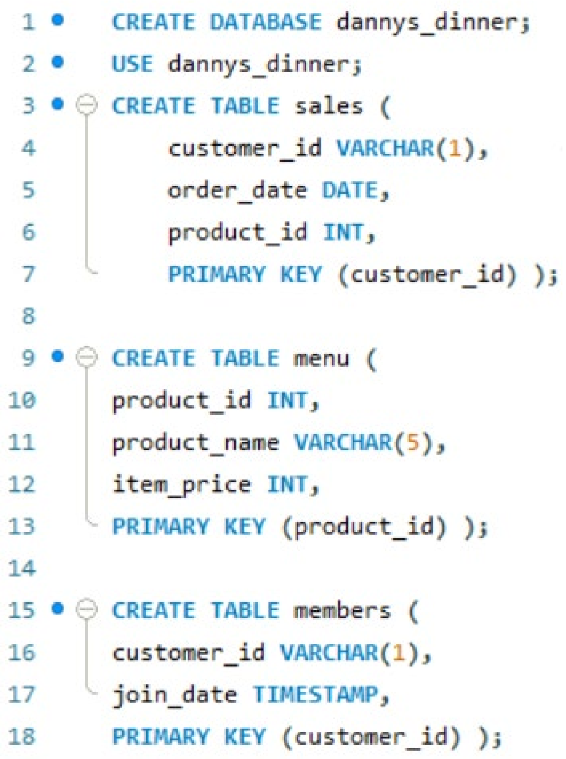
members	
customer_id	VARCHAR(1)
join_date	TIMESTAMP

menu	
product_id	INTEGER
product_name	VARCHAR(5)
price	INTEGER

Tool used for this study case: **MYSQL and Power BI for visualization**

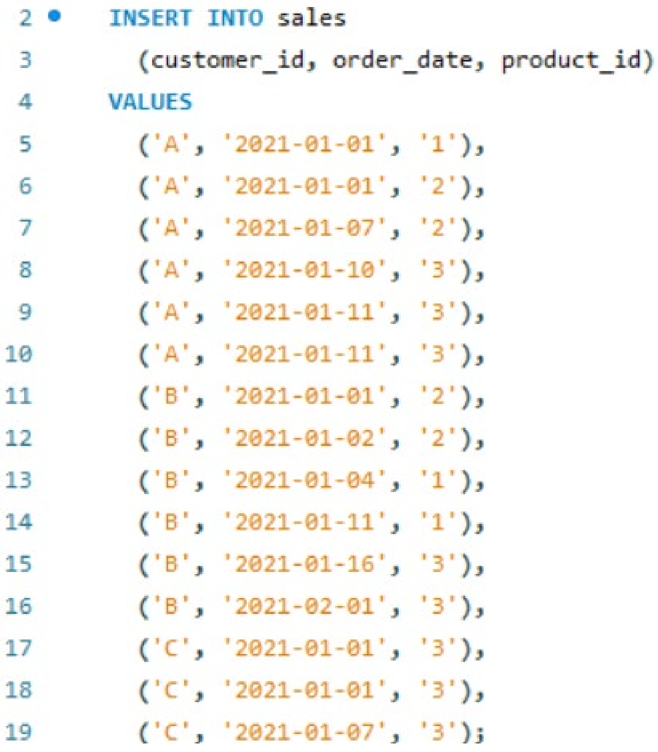
Natacha Jeria 2024-09-02

On MySQL I have created a database called dannys_dinner where in schema section, I have created 3 tables: sales, members, menu. Using the query provided by 8weeksqlchallenge.com

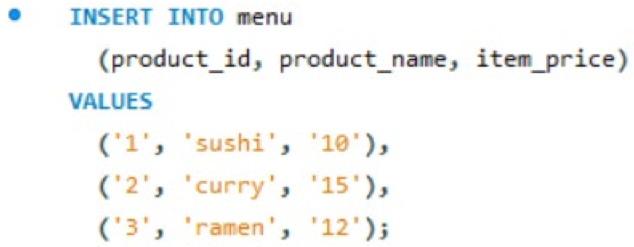


```
CREATE DATABASE dannys_dinner;
USE dannys_dinner;
CREATE TABLE sales (
    customer_id VARCHAR(1),
    order_date DATE,
    product_id INT,
    PRIMARY KEY (customer_id) );

CREATE TABLE menu (
product_id INT,
product_name VARCHAR(5),
item_price INT,
PRIMARY KEY (product_id) );

CREATE TABLE members (
customer_id VARCHAR(1),
join_date TIMESTAMP,
PRIMARY KEY (customer_id) );
```

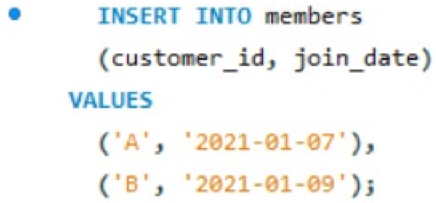
And add data to each table



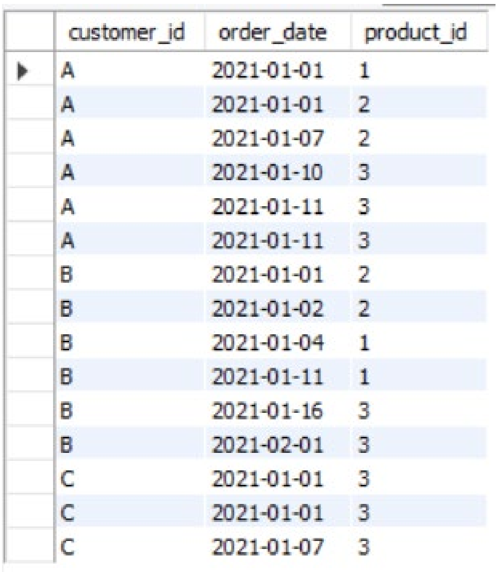
```
INSERT INTO sales
  (customer_id, order_date, product_id)
VALUES
  ('A', '2021-01-01', '1'),
  ('A', '2021-01-01', '2'),
  ('A', '2021-01-07', '2'),
  ('A', '2021-01-10', '3'),
  ('A', '2021-01-11', '3'),
  ('A', '2021-01-11', '3'),
  ('B', '2021-01-01', '2'),
  ('B', '2021-01-02', '2'),
  ('B', '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');
```



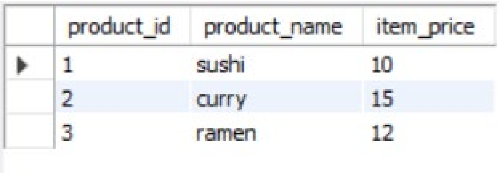
```
INSERT INTO menu
  (product_id, product_name, item_price)
VALUES
  ('1', 'sushi', '10'),
  ('2', 'curry', '15'),
  ('3', 'ramen', '12');
```



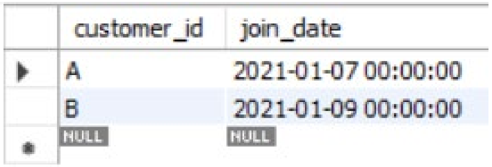
```
  INSERT INTO members
  (customer_id, join_date)
VALUES
  ('A', '2021-01-07'),
  ('B', '2021-01-09');
```



	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



	product_id	product_name	item_price
▶	1	sushi	10
	2	curry	15
	3	ramen	12



	customer_id	join_date
▶	A	2021-01-07 00:00:00
	B	2021-01-09 00:00:00
*	NULL	NULL

Clean data and format:

I have noticed that table members has join_date column showing time and we don't need that in our analysis, so I have updated the data type of the column in Alter Table

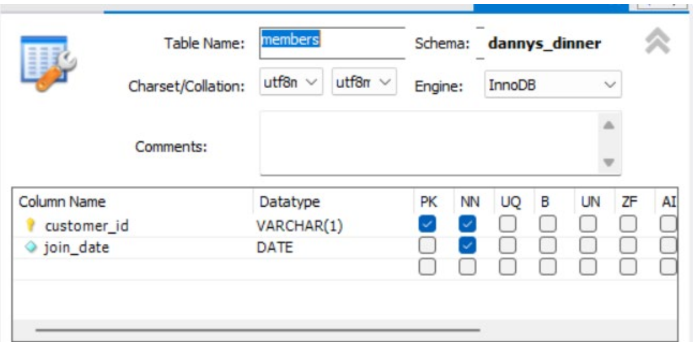
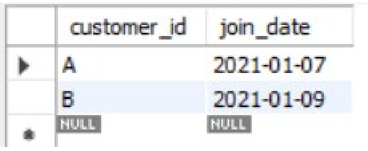


Table Name: members Schema: dannys_dinner

Charset/Collation: utf8n utf8n Engine: InnoDB

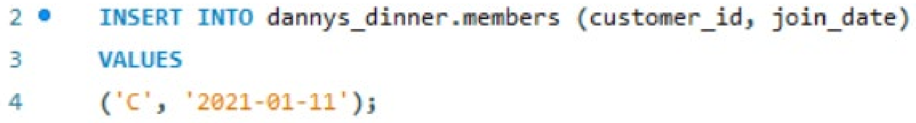
Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI
customer_id	VARCHAR(1)	☑	☑	□	□	□	□	□
join_date	DATE	□	☑	□	□	□	□	□
		□	□	□	□	□	□	□



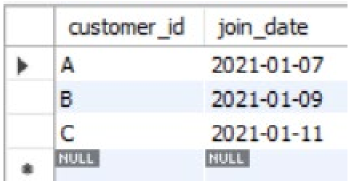
	customer_id	join_date
▶	A	2021-01-07
	B	2021-01-09
*	NULL	NULL

I have also noticed that we were missing information about join_date of customer C, so I have added the information into the table



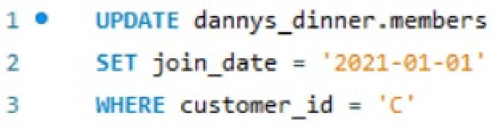
```sql
INSERT INTO dannys_dinner.members (customer_id, join_date)
VALUES
('C', '2021-01-11');
```

Final result:

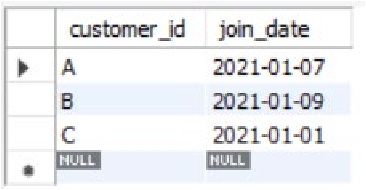


customer_id	join_date
A	2021-01-07
B	2021-01-09
C	2021-01-11
NULL	NULL

After adding the date I realized that the join date for customer C is wrong, so I updated the join_date as it follows:



```sql
UPDATE dannys_dinner.members
SET join_date = '2021-01-01'
WHERE customer_id = 'C'
```

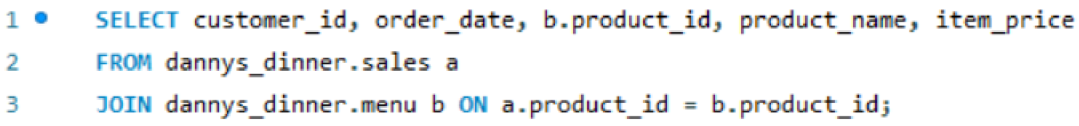


customer_id	join_date
A	2021-01-07
B	2021-01-09
C	2021-01-01
NULL	NULL

Case Study Questions

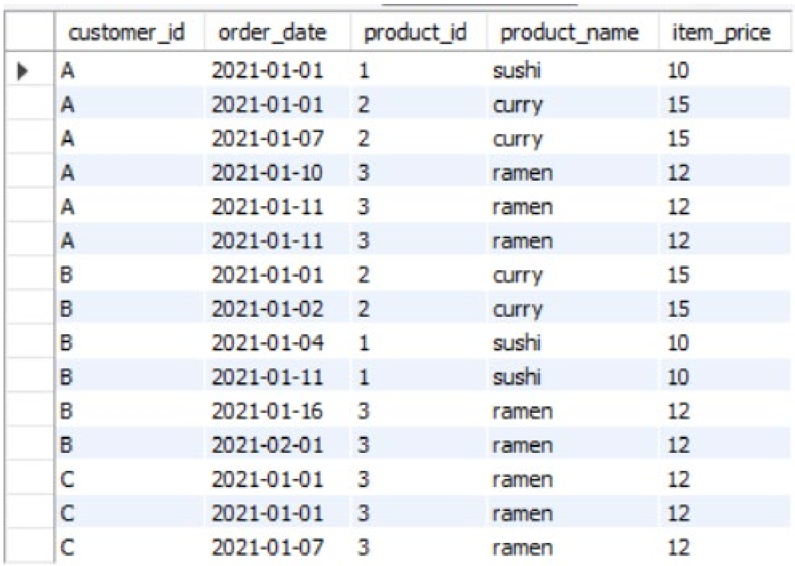
What products are purchased for each client and how many times are purchased by all customers?

Query: What product was purchased for each customer.



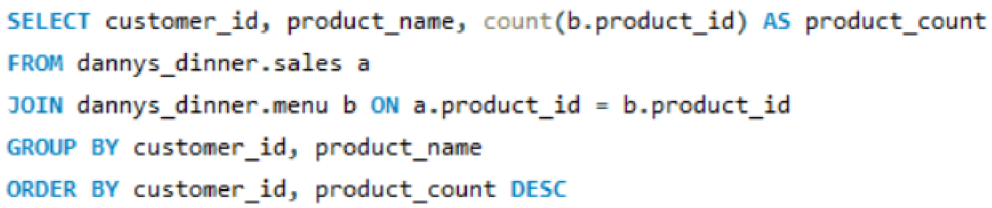
```sql
SELECT customer_id, order_date, b.product_id, product_name, item_price
FROM dannys_dinner.sales a
JOIN dannys_dinner.menu b ON a.product_id = b.product_id;
```

Result:



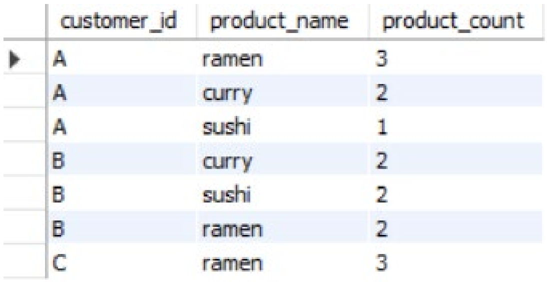
customer_id	order_date	product_id	product_name	item_price
A	2021-01-01	1	sushi	10
A	2021-01-01	2	curry	15
A	2021-01-07	2	curry	15
A	2021-01-10	3	ramen	12
A	2021-01-11	3	ramen	12
A	2021-01-11	3	ramen	12
B	2021-01-01	2	curry	15
B	2021-01-02	2	curry	15
B	2021-01-04	1	sushi	10
B	2021-01-11	1	sushi	10
B	2021-01-16	3	ramen	12
B	2021-02-01	3	ramen	12
C	2021-01-01	3	ramen	12
C	2021-01-01	3	ramen	12
C	2021-01-07	3	ramen	12

<u>Query: Total times purchased for each customer.</u>



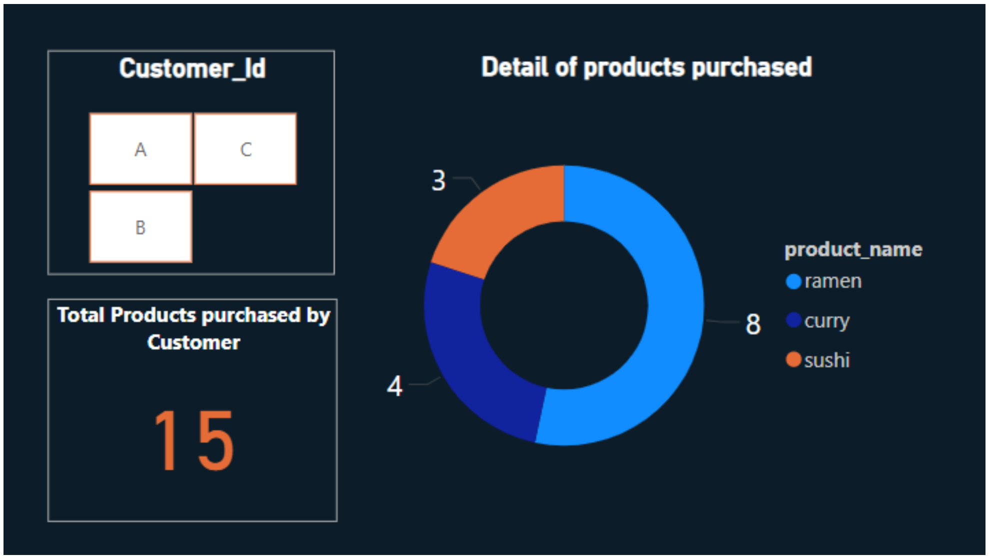
```
SELECT customer_id, product_name, count(b.product_id) AS product_count
FROM dannys_dinner.sales a
JOIN dannys_dinner.menu b ON a.product_id = b.product_id
GROUP BY customer_id, product_name
ORDER BY customer_id, product_count DESC
```

<u>Result:</u>



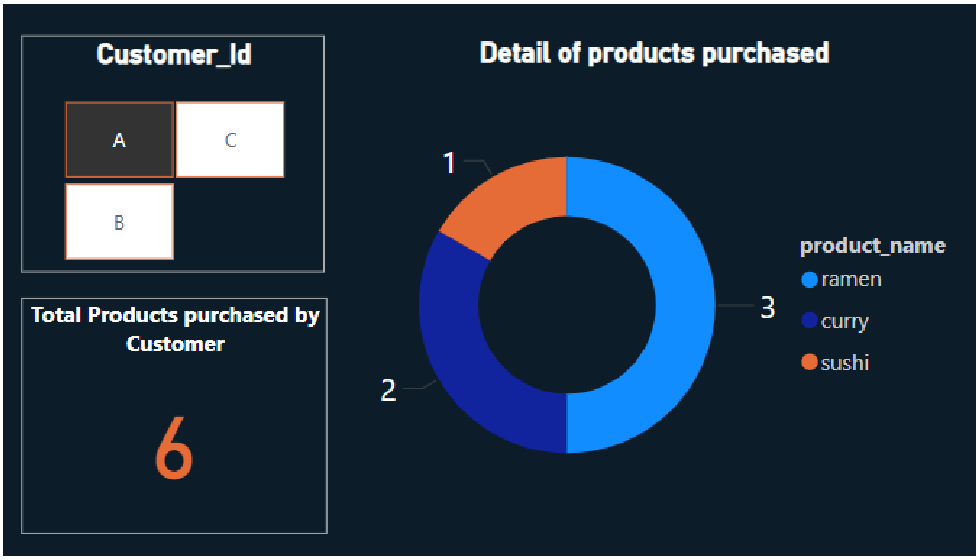
customer_id	product_name	product_count
A	ramen	3
A	curry	2
A	sushi	1
B	curry	2
B	sushi	2
B	ramen	2
C	ramen	3

Summary:



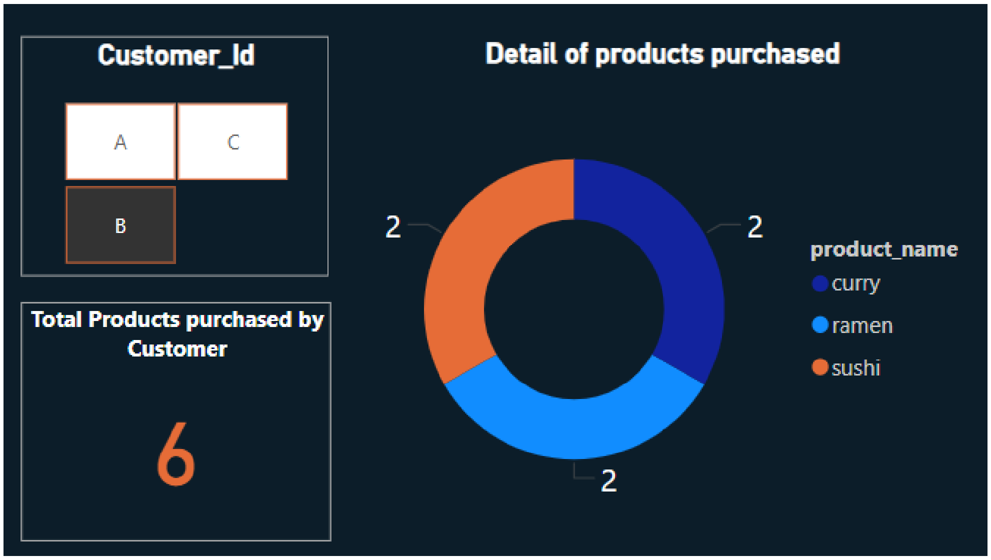
Customer_Id
A
C
B
Total Products purchased by Customer
15
Detail of products purchased
3
4
8
product_name
● ramen
● curry
● sushi

Customer A:

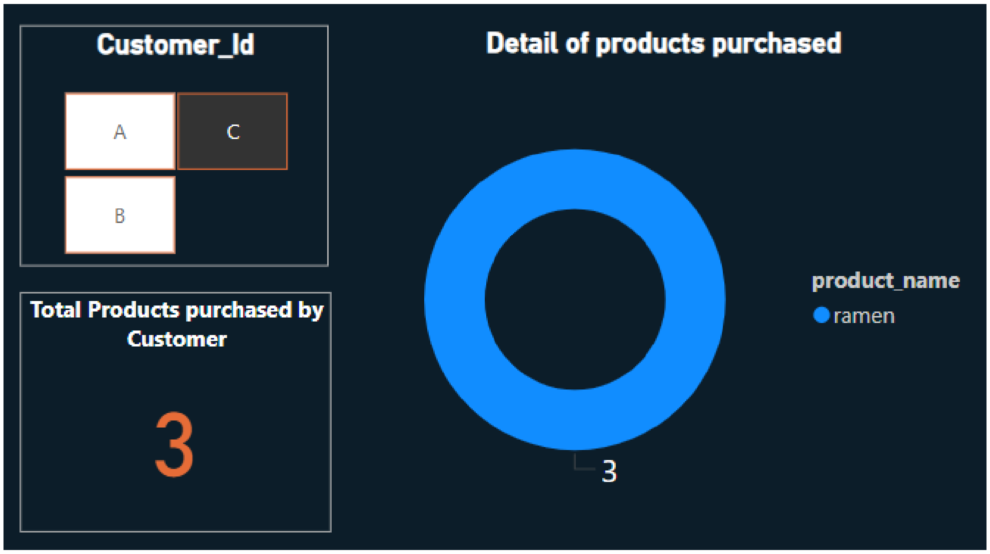


Customer_Id
A
C
B
Total Products purchased by Customer
6
Detail of products purchased
1
2
3
product_name
● ramen
● curry
● sushi

Customer B:

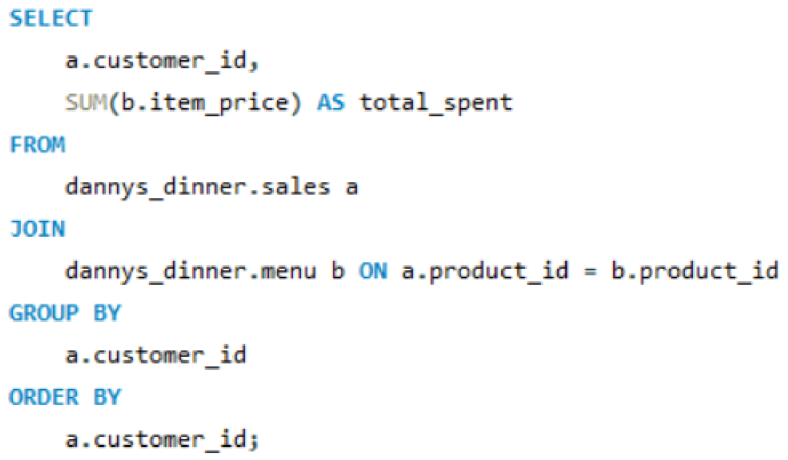


Customer C:



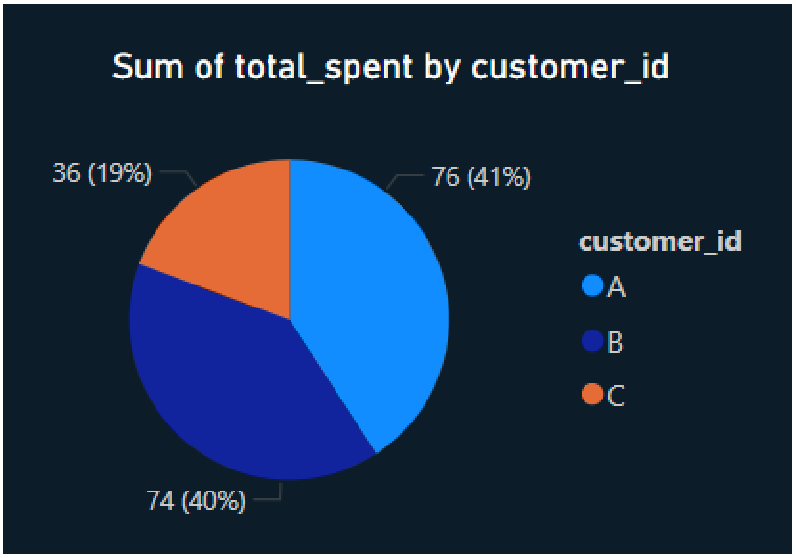
What is the total amount each customer spent at the restaurant?

Query:



```
SELECT
    a.customer_id,
    SUM(b.item_price) AS total_spent
FROM
    dannys_dinner.sales a
JOIN
    dannys_dinner.menu b ON a.product_id = b.product_id
GROUP BY
    a.customer_id
ORDER BY
    a.customer_id;
```

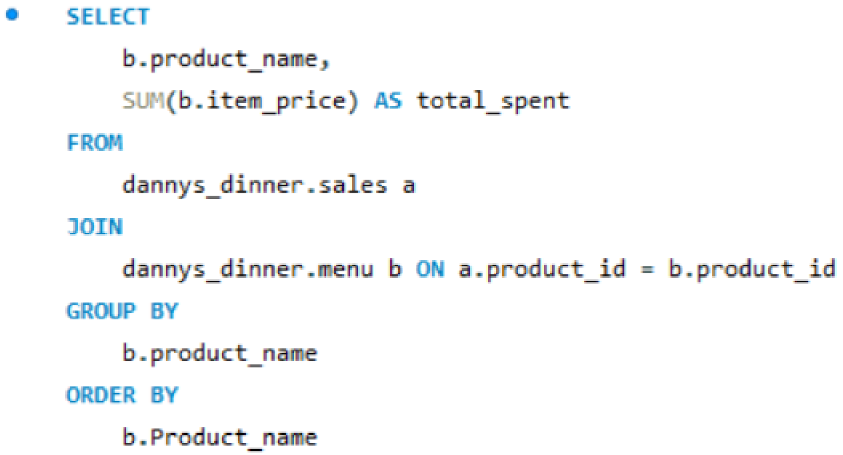
Results:



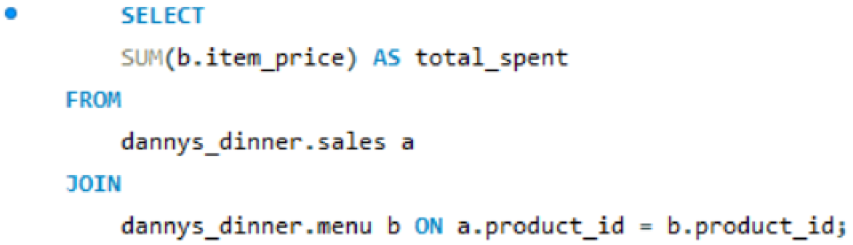
Customer A is the customer that spent the most at the restaurant.

What is the amount of sale for each product, what is the favourite product? and what is the total amount of sales for the company?

Query:

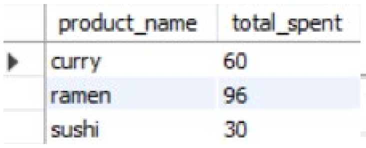


```
SELECT
    b.product_name,
    SUM(b.item_price) AS total_spent
FROM
    dannys_dinner.sales a
JOIN
    dannys_dinner.menu b ON a.product_id = b.product_id
GROUP BY
    b.product_name
ORDER BY
    b.Product_name
```

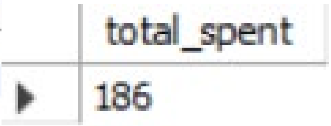


```
    SELECT
    SUM(b.item_price) AS total_spent
FROM
    dannys_dinner.sales a
JOIN
    dannys_dinner.menu b ON a.product_id = b.product_id;
```

Results:

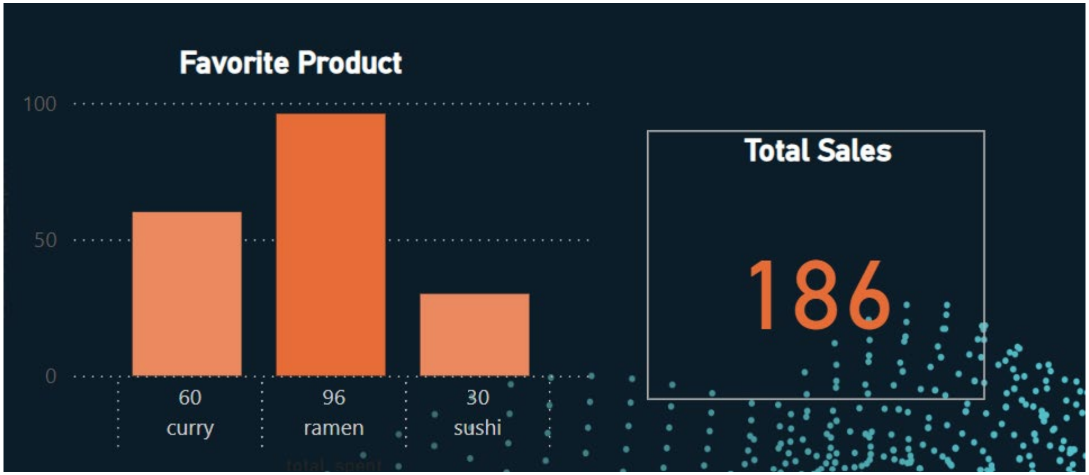


product_name	total_spent
curry	60
ramen	96
sushi	30



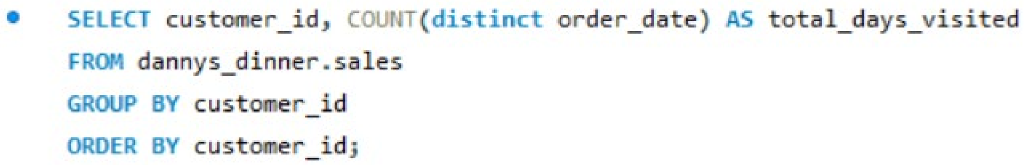
total_spent
186

Visualization:



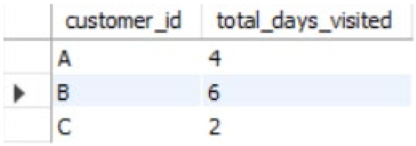
How many days has each customer visited the restaurant?

Query:



```
SELECT customer_id, COUNT(distinct order_date) AS total_days_visited
FROM dannys_dinner.sales
GROUP BY customer_id
ORDER BY customer_id;
```

Results:

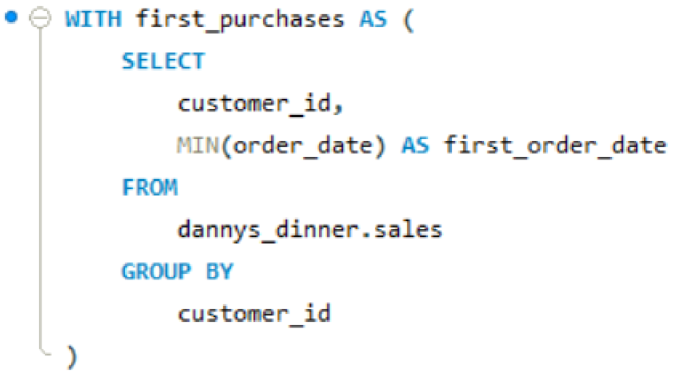


customer_id	total_days_visited
A	4
B	6
C	2

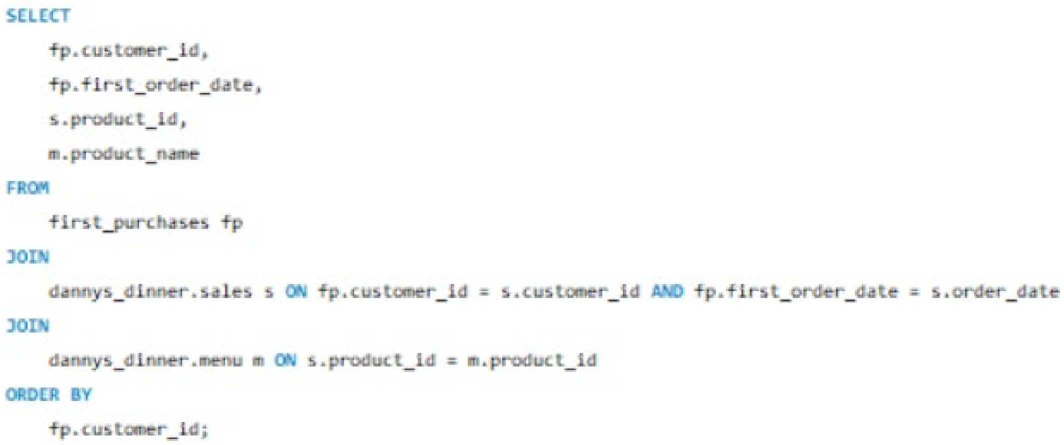
Costumer B is the one who visited the restaurant the most.

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

Query:

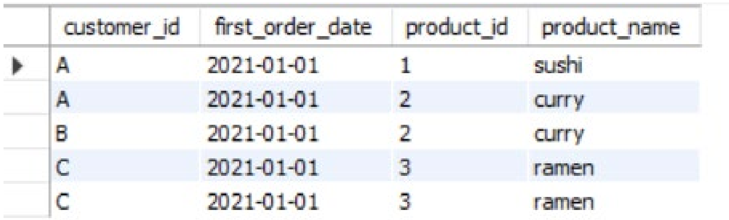


```
WITH first_purchases AS (
    SELECT
        customer_id,
        MIN(order_date) AS first_order_date
    FROM
        dannys_dinner.sales
    GROUP BY
        customer_id
)
```



```
SELECT
    fp.customer_id,
    fp.first_order_date,
    s.product_id,
    m.product_name
FROM
    first_purchases fp
JOIN
    dannys_dinner.sales s ON fp.customer_id = s.customer_id AND fp.first_order_date = s.order_date
JOIN
    dannys_dinner.menu m ON s.product_id = m.product_id
ORDER BY
    fp.customer_id;
```

Results:

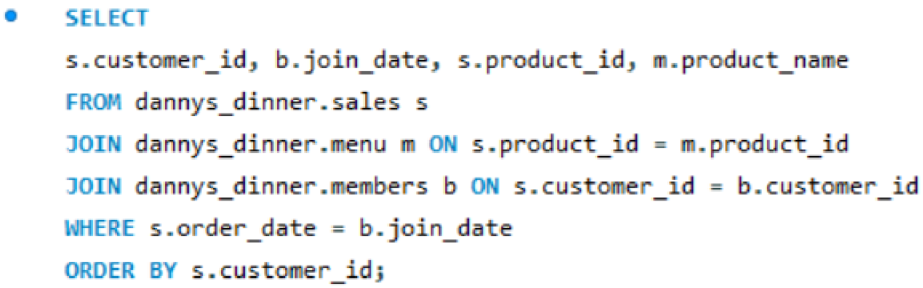


customer_id	first_order_date	product_id	product_name
A	2021-01-01	1	sushi
A	2021-01-01	2	curry
B	2021-01-01	2	curry
C	2021-01-01	3	ramen
C	2021-01-01	3	ramen

Customer A bought curry and sushi as first items purchased.
Customer B bought curry.
Customer C bought ramen.

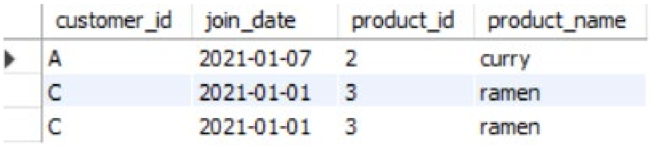
Which item was purchased first by the customer at the same day they became a member?

Query:



```
SELECT
s.customer_id, b.join_date, s.product_id, m.product_name
FROM dannys_dinner.sales s
JOIN dannys_dinner.menu m ON s.product_id = m.product_id
JOIN dannys_dinner.members b ON s.customer_id = b.customer_id
WHERE s.order_date = b.join_date
ORDER BY s.customer_id;
```

Results:

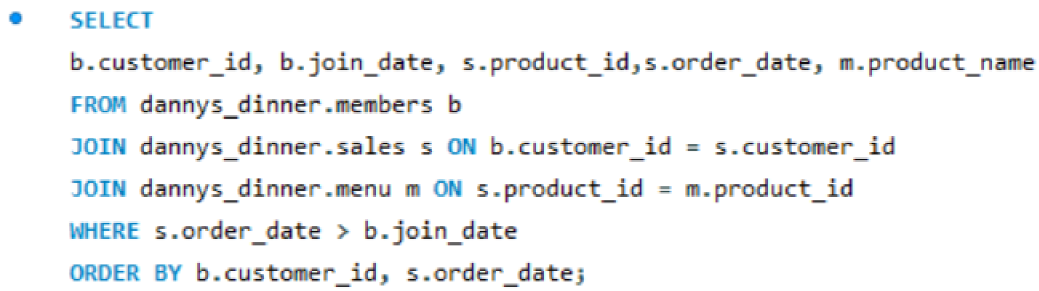


customer_id	join_date	product_id	product_name
A	2021-01-07	2	curry
C	2021-01-01	3	ramen
C	2021-01-01	3	ramen

The results shows that member A purchased curry
Member B did not buy any product the day became member.
Member C purchased ramen twice.

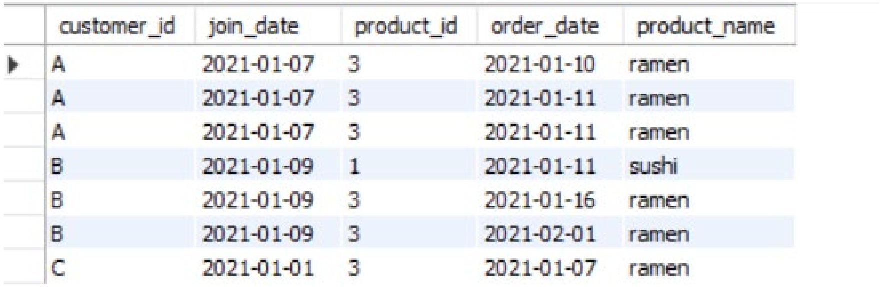
Which items were purchased by the customers after the day they became members?

Query:



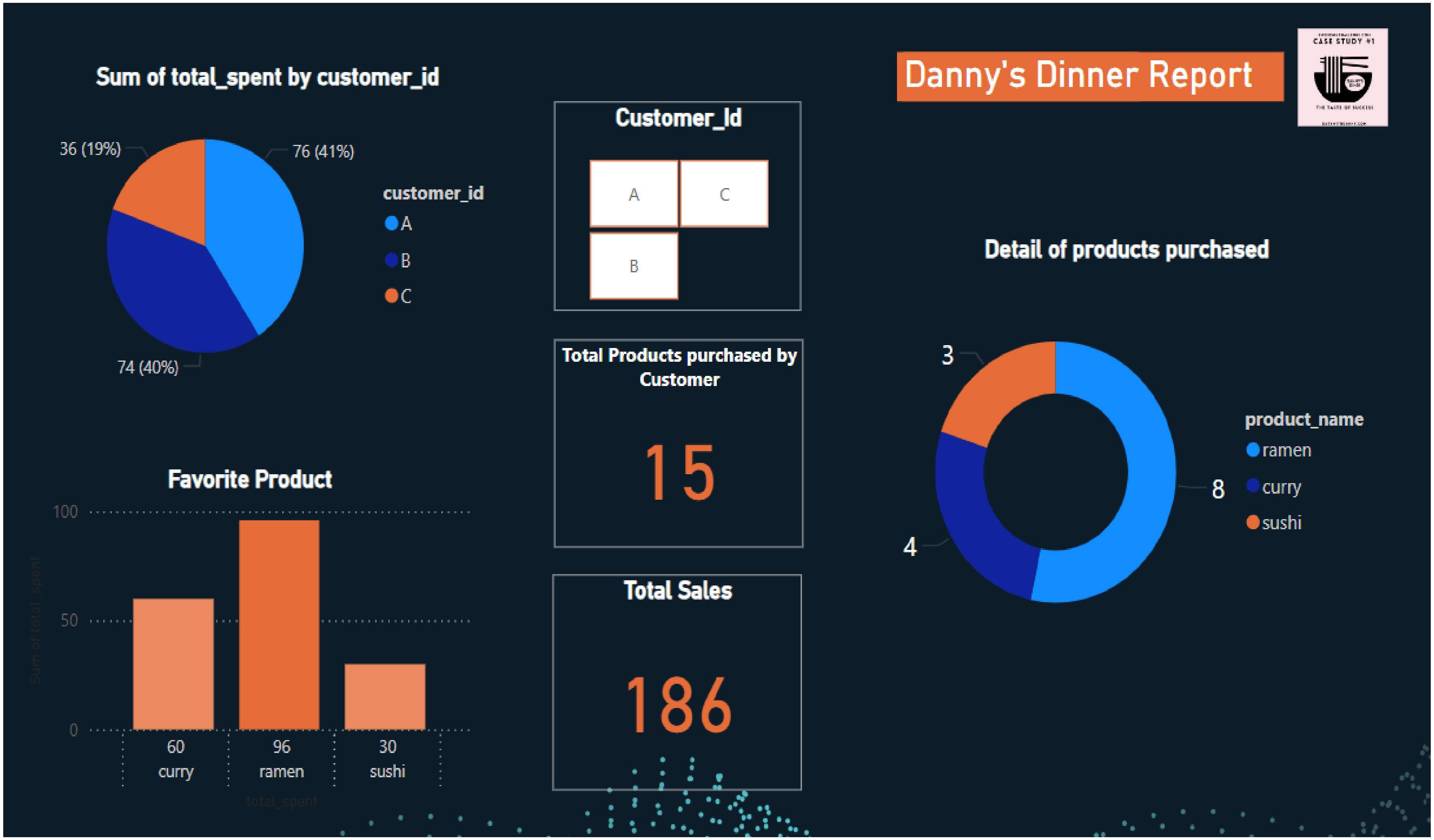
```
• SELECT
  b.customer_id, b.join_date, s.product_id,s.order_date, m.product_name
  FROM dannys_dinner.members b
  JOIN dannys_dinner.sales s ON b.customer_id = s.customer_id
  JOIN dannys_dinner.menu m ON s.product_id = m.product_id
  WHERE s.order_date > b.join_date
  ORDER BY b.customer_id, s.order_date;
```

Results:



	customer_id	join_date	product_id	order_date	product_name
▶	A	2021-01-07	3	2021-01-10	ramen
	A	2021-01-07	3	2021-01-11	ramen
	A	2021-01-07	3	2021-01-11	ramen
	B	2021-01-09	1	2021-01-11	sushi
	B	2021-01-09	3	2021-01-16	ramen
	B	2021-01-09	3	2021-02-01	ramen
	C	2021-01-01	3	2021-01-07	ramen

Customer A purchased ramen 3 times after became member,
Customer B purchased sushi and ramen twice
Customer C purchased ramen once.



Danny's Dinner Report

Sum of total_spent by customer_id

Customer_Id

Total Products purchased by Customer

15

Total Sales

186

Favorite Product

Detail of products purchased