

SQLQuery2.sql - D:\BFV0DOU\User (60))

```
CREATE DATABASE dannys_diner

USE 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);

select * from dbo.sales
```

91 %

Results Messages

customer_id	order_date	product_id
A	2021-01-01	1

Query executed successfully.

DE

SQLQuery2.sql - D...\BFV0DOU\User (60))*

```
CREATE TABLE menu(  
    product_id INTEGER,  
    product_name VARCHAR(30),  
    price INTEGER  
)  
  
INSERT INTO menu ( product_id, product_name, price)  
VALUES  
    (1, 'sushi', 10),  
    (2, 'curry', 15),  
    (3, 'ramen', 12);  
  
select * from dbo.menu
```

91 %

Results Messages

	product_id	product_name	price
1	1	sushi	10
2	2	curry	15
3	3	ramen	12

SQLQuery2.sql - D...\BFV0DOU\User (60))*

```
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')  
  
select * from dbo.members
```

91 %

Results Messages

	customer_id	join_date
1	A	2021-01-07
2	B	2021-01-09

Object Explorer

SQLQuery1.sql - D:\BFV0DOU\User (53))* X

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

```
--select dbo.sales.customer_id, SUM(dbo.menu.price) AS total_spent from sales
left join dbo.menu on dbo.sales.product_id = dbo.menu.product_id
group by dbo.sales.customer_id
-- OR
select s.customer_id, SUM(m.price) AS total_price from sales s
inner join menu m on s.product_id = m.product_id
group by s.customer_id
```

120 %

Results Messages

	customer_id	total_price
1	A	76
2	B	74
3	C	36

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

```
select s.customer_id, count(distinct s.order_date) AS number_of_visits from sales s
group by s.customer_id
```

120 %

Results Messages

	customer_id	number_of_visits
1	A	4
2	B	6
3	C	2

--3. WHAT WAS THE FIRST ITEM FROM THE MENU PURCHASED BY EACH CUSTOMER?

```
--select s.customer_id, s.order_date, m.product_name from sales s
-- inner join menu m on s.product_id = m.product_id
-- to find out which was the first product I am going to use RANK and OVER
--select s.customer_id,
--      s.order_date,
--      m.product_name,
--      RANK() OVER(PARTITION BY s.customer_id ORDER BY s.order_date ASC) AS rank
--from sales s
-- inner join menu m on s.product_id = m.product_id
-- to filter the output I will use a COMMON TABLE EXPRESSION
WITH CTE AS (select s.customer_id,
                    s.order_date,
                    m.product_name,
                    RANK() OVER(PARTITION BY s.customer_id ORDER BY s.order_date ASC) AS rank
               from sales s
               inner join menu m on s.product_id = m.product_id)
select customer_id, product_name from CTE
where rank = 1
```

119 %

Results Messages

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

Query executed successfully.

--4. WHAT WAS THE MOST PURCHASED ITEM ON THE MENU AND HOW MANY TIMES WAS IT PURCHASED BY ALL CUSTOMERS?

```
--select COUNT(s.order_date) AS number_of_orders, m.product_name from sales s
-- inner join menu m on s.product_id = m.product_id
-- group by m.product_name
-- order by COUNT(s.order_date) DESC
-- I can limit the results just to one with TOP 1
--select TOP 1
--      COUNT(s.order_date) AS number_of_orders, m.product_name from sales s
-- inner join menu m on s.product_id = m.product_id
-- group by m.product_name
-- order by COUNT(s.order_date) DESC
```

119 %

Results Messages

	number_of_orders	product_name
1	8	ramen

--5. WHICH ITEM WAS THE MOST POPULAR FOR EACH CUSTOMER?

```
--select s.customer_id, m.product_name, COUNT(s.order_date) AS number_of_orders from sales s
INNER JOIN menu m on s.product_id = m.product_id
group by m.product_name, s.customer_id
-- I am going to use RANK OVER PARTITION BY to find this top product for each customer
--select s.customer_id, m.product_name, COUNT(s.order_date) AS number_of_orders,
RANK() OVER (PARTITION BY customer_id ORDER BY COUNT(s.order_date) DESC) AS rank
from sales s
INNER JOIN menu m on s.product_id = m.product_id
group by m.product_name, s.customer_id
```

119 %

Results Messages

	customer_id	product_name	number_of_orders	rank
1	A	ramen	3	1
2	A	curry	2	2
3	A	sushi	1	3
4	B	curry	2	1

Query executed successfully.

DESKTOP-BFV

-- 6. WHICH ITEM WAS PURCHASED FIRST BY THE CUSTOMER AFTER THEY BECAME A MEMBER?

```
--select m.product_name, s.customer_id, s.order_date from sales s
inner join members mem on s.customer_id = mem.customer_id
inner join menu m on s.product_id = m.product_id
where mem.join_date <= s.order_date
-- to find out which was the first purchased item I am going to use RANK () OVER PARTITION BY
--select m.product_name, s.customer_id, s.order_date,
RANK() OVER (PARTITION BY s.customer_id ORDER BY s.order_date ASC) AS rank
from sales s
inner join members mem on s.customer_id = mem.customer_id
inner join menu m on s.product_id = m.product_id
where mem.join_date <= s.order_date
-- to filter the output I will use a COMMON TABLE EXPRESSION, then SELECT from CTE where rank = 1
WITH CTE AS(
select m.product_name, s.customer_id, s.order_date,
RANK() OVER (PARTITION BY s.customer_id ORDER BY s.order_date ASC) AS rank
from sales s
inner join members mem on s.customer_id = mem.customer_id
inner join menu m on s.product_id = m.product_id
where mem.join_date <= s.order_date)
select customer_id, product_name from CTE
where rank = 1
```

119 %

Results Messages

	customer_id	product_name
1	A	curry
2	B	sushi

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

```
WITH CTE AS(
select m.product_name, s.customer_id, s.order_date,
RANK() OVER (PARTITION BY s.customer_id ORDER BY s.order_date DESC) AS rank
from sales s
inner join members mem on s.customer_id = mem.customer_id
inner join menu m on s.product_id = m.product_id
where mem.join_date > s.order_date)
select customer_id, product_name from CTE
where rank = 1
```

119 %

Results Messages

	customer_id	product_name
1	A	sushi
2	A	curry
3	B	sushi

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

```
select COUNT(s.product_id) AS total_number_of_products, SUM(m.price) AS total_amount_spent, mem.customer_id from sales s
inner join members mem on s.customer_id = mem.customer_id
inner join menu m on s.product_id = m.product_id
where s.order_date < mem.join_date
group by mem.customer_id
```

119 %

Results Messages

	total_number_of_products	total_amount_spent	customer_id
1	2	25	A
2	3	40	B

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

```
select s.customer_id,  
SUM(CASE WHEN m.product_name = 'sushi' THEN m.price * 20  
ELSE m.price * 10  
END) as total_points  
from sales s  
inner join menu m on s.product_id = m.product_id  
group by s.customer_id
```

100 %

Results Messages

	customer_id	total_points
1	A	860
2	B	940
3	C	360

-- 9. different conditional in order to pivot into columns for sushi and non-sushi products

```
select s.customer_id,  
SUM (CASE WHEN m.product_name = 'sushi' THEN m.price * 20 ELSE NULL END) as TOTAL_POINTS_FOR_SUSHI,  
SUM (CASE WHEN m.product_name != 'sushi' THEN m.price * 10 ELSE NULL END) as TOTAL_POINTS_FOR_NON_SUSHI_PRODUCTS  
from sales s  
inner join menu m on s.product_id = m.product_id  
group by s.customer_id
```

100 %

Results Messages

	customer_id	TOTAL_POINTS_FOR_SUSHI	TOTAL_POINTS_FOR_NON_SUSHI_PRODUCTS
1	A	200	660
2	B	400	540
3	C	NULL	360

```
-- 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 mem.join_date AND DATEADD(day, 7, mem.join_date) THEN m.price * 20  
-- if this condition is not true, it's going to check this other condition where it check if your product name = sushi  
WHEN m.product_name = 'sushi' THEN m.price * 20 ELSE m.price * 10 END) AS total_points  
from sales s  
INNER JOIN menu m on m.product_id = s.product_id  
-- i am going to do a left join I want all the records that are in the Sales table and the matching records from the Members table  
LEFT JOIN members mem on s.customer_id = mem.customer_id  
WHERE s.customer_id IN ('A', 'B') AND s.order_date <= '2021-01-31'  
group by s.customer_id
```

119 %

Results Messages

	customer_id	total_points
1	A	1370
2	B	940

```
--11. Recreate the table output using the available data. Danny requires me to create a basic data tables that his team can see to  
-- quickly derive insights without needing to write SQL.
```

```
select s.customer_id, s.order_date, m.product_name, m.price,  
CASE WHEN s.order_date >= mem.join_date THEN 'Y' ELSE 'N' END AS member  
from sales s  
inner join menu m on s.product_id = m.product_id  
-- I am going to use left join because I want all the records that are in the Sales table plus only the matching records  
-- that are in the Members table  
left join members mem on s.customer_id = mem.customer_id  
order by s.customer_id, s.order_date
```

119 %

Results Messages

	customer_id	order_date	product_name	price	member
1	A	2021-01-01	sushi	10	N
2	A	2021-01-01	curry	15	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. Rank all the things:
-- Danny also requires further information about the ranking of products. He purposely does not need the ranking of
-- non members, so he expects NULL ranking values for customers who are not yet part of the loyalty program.
WITH customer_data AS (select s.customer_id, s.order_date, m.product_name, m.price,
CASE
    WHEN s.order_date < mem.join_date THEN 'N'
    WHEN s.order_date >= mem.join_date THEN 'Y'
    ELSE 'N' END AS member
from sales s
left join members mem on s.customer_id = mem.customer_id
inner join menu m on s.product_id = m.product_id)

SELECT *,
CASE WHEN member = 'N' THEN NULL
ELSE RANK() OVER(PARTITION BY customer_id, member ORDER BY order_date)
END AS ranking
FROM customer_data
ORDER BY customer_id, order_date

```

89 %

Results Messages

	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