 # DANNY'S DINER CASE STUDY

```
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');
```

Case Study Questions

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

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

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

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

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

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

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

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

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

-- 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 * from sales
select * from menu
select * from members
```

100 %

Results Messages

	customer_id	order_date	product_id
1	A	2021-01-01	1
2	A	2021-01-01	2
3	A	2021-01-07	2
4	A	2021-01-10	3
5	A	2021-01-11	3
6	A	2021-01-11	3
7	B	2021-01-01	2
8	B	2021-01-02	2

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

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

Sol 1)

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

select s.customer_id,SUM(m.price) as amount_spent
from sales s
left join menu m on m.product_id = s.product_id
group by s.customer_id
```

100 %

Results Messages

	customer_id	amount_spent
1	A	76
2	B	74
3	C	36

Sol 2)

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

select s.customer_id,count(distinct s.order_date) as count_days
from sales s
group by s.customer_id
```

100 %

Results Messages

	customer_id	count_days
1	A	4
2	B	6
3	C	2

Sol 3) if no two items purchased on same day

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

select t.customer_id,t.product_name from
(select s.customer_id,m.product_name,row_number() over(partition by customer_id order by customer_id) as rn
from sales s
left join menu m on m.product_id = s.product_id)t
where rn = 1
```

100 %

Results Messages

	customer_id	product_name
1	A	sushi
2	B	curry
3	C	ramen

If two items purchased on same day

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

select t.customer_id,t.product_name from
(select s.customer_id,m.product_name,rank() over(partition by s.customer_id order by order_date) as rn
from sales s
left join menu m on m.product_id = s.product_id)t
where rn = 1
```

100 %

Results | Messages

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

Sol 4)

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

select top 1 m.product_name,count(s.product_id) as no_of_items
from sales s
left join menu m on m.product_id = s.product_id
group by m.product_name
order by no_of_items desc
```

00 %

Results | Messages

	product_name	no_of_items
1	ramen	8

Sol 5)

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

with cte1 as(
select s.customer_id, m.product_name,count(s.product_id) as no_of_items,
dense_rank() over(partition by s.customer_id order by count(s.product_id) desc) as drk
from sales s
left join menu m on m.product_id = s.product_id
group by s.customer_id,m.product_name)
select customer_id, product_name
from cte1
where drk =1
```

100 %

Results | Messages

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

Sol 6)

```sql
-- 6. Which item was purchased first by the customer after they became a member?
with cte1 as(
select s.customer_id, m.product_name,s.order_date,
row_number() over(partition by s.customer_id order by s.order_date) as rn
from sales s
left join menu m on m.product_id = s.product_id
left join members mb on mb.customer_id = s.customer_id
where s.order_date > mb.join_date)
select customer_id, product_name from cte1
where rn=1
```

100 %

Results | Messages

	customer_id	product_name
1	A	ramen
2	B	sushi

Sol 7)

```sql
-- 7. Which item was purchased just before the customer became a member?
with cte1 as(
select s.customer_id, m.product_name,s.order_date,
dense_rank() over(partition by s.customer_id order by s.order_date desc) as rn
from sales s
left join menu m on m.product_id = s.product_id
left join members mb on mb.customer_id = s.customer_id
where s.order_date < mb.join_date)
select customer_id, product_name from cte1
where rn=1
```

100 %

Results | Messages

	customer_id	product_name
1	A	sushi
2	A	curry
3	B	sushi

Sol 8)

```sql
-- 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) as total_items,sum(m.price) as amount_spent
--dense_rank() over(partition by s.customer_id order by s.order_date desc) as rn
from sales s
left join menu m on m.product_id = s.product_id
left join members mb on mb.customer_id = s.customer_id
where s.order_date < mb.join_date
group by s.customer_id
```

00 %

Results | Messages

	customer_id	total_items	amount_spent
1	A	2	25
2	B	3	40

Sol 9)

```sql
-- 9.  If each $1 spent equates to 10 points and sushi has a 2x points multiplier - how many points would each customer have?
with cte1 as(
select s.customer_id,m.product_name,
case
when m.product_name='sushi' then m.price*10*2
else m.price*10 end as p
from sales s
left join menu m on m.product_id = s.product_id)
select customer_id,sum(p) as points from cte1
group by customer_id
```

100 %

Results | Messages

	customer_id	points
1	A	860
2	B	940
3	C	360

Sol 10)

```sql
--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, m.product_name, m.price, order_date, join_date,
CASE
    WHEN s.order_date BETWEEN mb.join_date AND DATEADD(day, 7, mb.join_date) THEN m.price*10*2
    WHEN m.product_name = 'sushi' THEN m.price*10*2
    ELSE m.price*10
    END as points
FROM menu m
JOIN sales s
ON s.product_id = m.product_id
```

```sql
JOIN members mb
ON s.customer_id = mb.customer_id
WHERE order_date < '2021-02-01'
```

Results | Messages

	customer_id	product_name	price	order_date	join_date	points
1	A	sushi	10	2021-01-01	2021-01-07	200
2	A	curry	15	2021-01-01	2021-01-07	150
3	A	curry	15	2021-01-07	2021-01-07	300
4	A	ramen	12	2021-01-10	2021-01-07	240
5	A	ramen	12	2021-01-11	2021-01-07	240
6	A	ramen	12	2021-01-11	2021-01-07	240
7	B	curry	15	2021-01-01	2021-01-09	150
8	B	curry	15	2021-01-02	2021-01-09	150
9	B	sushi	10	2021-01-04	2021-01-09	200
10	B	sushi	10	2021-01-11	2021-01-09	200
11	B	ramen	12	2021-01-16	2021-01-09	240

```sql
SELECT customer_id, SUM(points) as total_points
FROM cte
GROUP BY customer_id
```

150 %

Results Messages

	customer_id	total_points
1	A	1370
2	B	940

```
-- BONUS Questions
-- Q.11: Determine the name and price of the product ordered by each customer on all order dates & find out whether the
-- customer was a member on the order date or not
```

```
SELECT s.customer_id, s.order_date, m.product_name, m.price,
CASE
    WHEN mb.join_date <= s.order_date THEN 'Y'
    ELSE 'N'
    END as member
FROM menu m
JOIN sales s
ON s.product_id = m.product_id
LEFT JOIN members mb
ON mb.customer_id = s.customer_id
```

%

Results Messages

customer_id	order_date	product_name	price	member
A	2021-01-01	sushi	10	N
A	2021-01-01	curry	15	N
A	2021-01-07	curry	15	Y
A	2021-01-10	ramen	12	Y
A	2021-01-11	ramen	12	Y
A	2021-01-11	ramen	12	Y
B	2021-01-01	curry	15	N
B	2021-01-02	curry	15	N
B	2021-01-04	sushi	10	N
B	2021-01-11	sushi	10	Y
B	2021-01-16	ramen	12	Y
B	2021-02-01	ramen	12	Y
C	2021-01-01	ramen	12	N
C	2021-01-01	ramen	12	N
C	2021-01-07	ramen	12	N

```sql
-- Bonus Q.12: Rank the previous output from Q.11 based on the order_date for each customer. Display NULL if customer was
-- not a member when dish was ordered.

WITH cte as
(
    SELECT s.customer_id, s.order_date, m.product_name, m.price,
CASE
    WHEN mb.join_date <= s.order_date THEN 'Y'
    ELSE 'N'
    END as member_status
FROM menu m
JOIN sales s
ON s.product_id = m.product_id
LEFT JOIN members mb
ON mb.customer_id = s.customer_id
)
SELECT *,
CASE
    WHEN cte.member_status = 'Y' THEN RANK() OVER(PARTITION BY customer_id, member_status
                                                  ORDER BY order_date)
    ELSE NULL
    END AS ranking
FROM cte
```

	customer_id	order_date	product_name	price	member_status	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