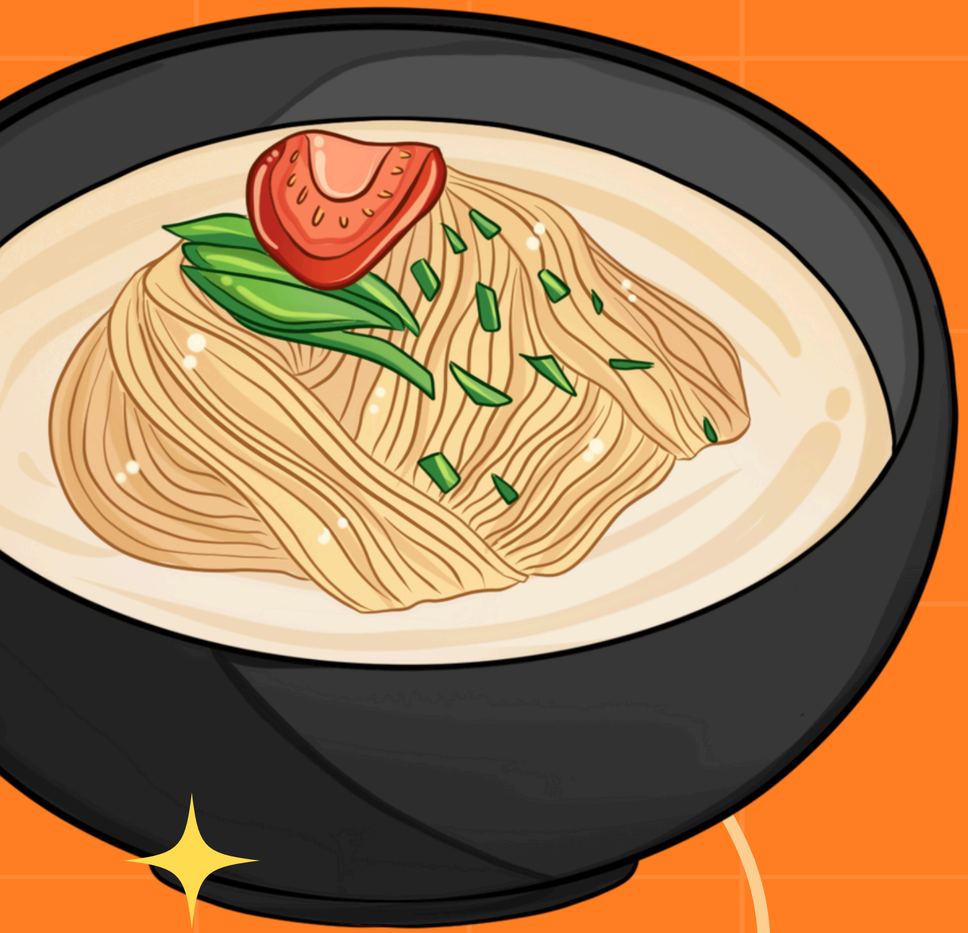
SQL PROJECT

Danny's Diner

BY RIZWAN SHAH



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 RIZWAN 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 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.

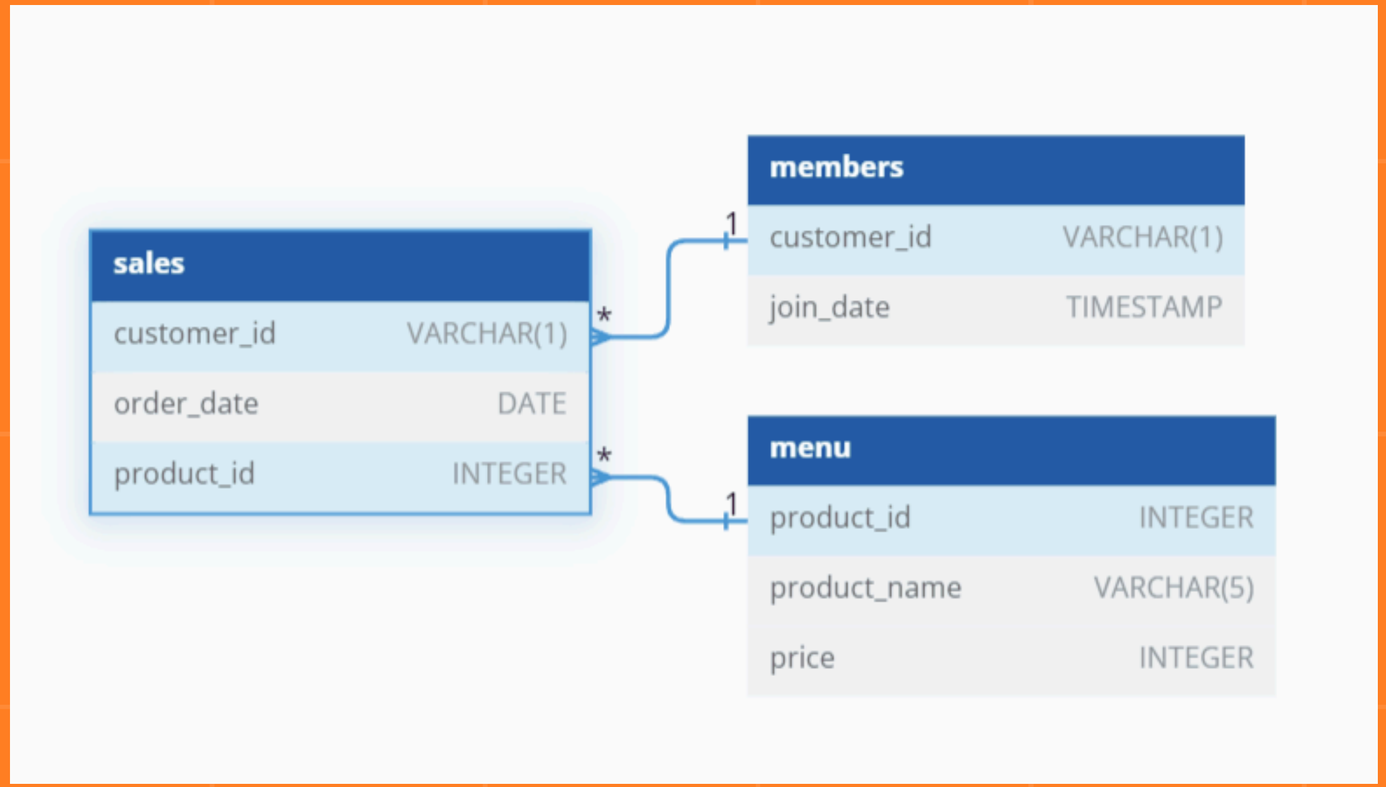
He plans on using these insights to help him decide whether he should expand the existing customer loyalty program - additionally he needs help to generate some basic datasets so his team can easily inspect the data without needing to use SQL.

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!

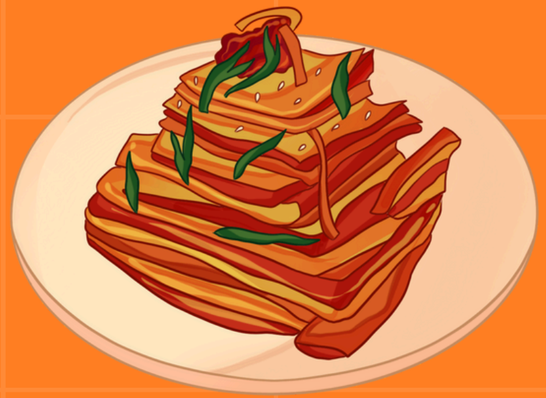
Entity Relationship Diagram

Danny has shared 3 key datasets for this case study:

- sales
- menu
- members



Case Study Questions



Q1.What is the total amount each customer spent at the restaurant?

ANSWER:

```
select s.customer_id,sum(price) as total_spent from sales as s
join menu as m
on s.product_id=m.product_id
group by s.customer_id ;
```

OUTPUT:

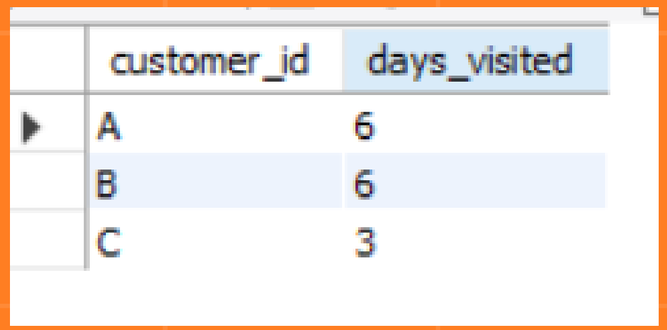
customer_id	total_spent
A	76
B	74
C	36

Q2.How many days has each customer visited the restaurant?

ANSWER:

```
select customer_id,count(order_date) as days_visited
from sales
group by customer_id;
```

OUTPUT:



customer_id	days_visited
A	6
B	6
C	3

Q3.What was the first item from the menu purchased by each customer?

ANSWER:

```
with cte as (
select s.customer_id,s.order_date,m.product_id,m.product_name from sales as s
join menu as m
on s.product_id=m.product_id)
select * from(
select customer_id,order_date,product_name,
dense_rank() over (partition by customer_id order by order_date) as rn
from cte ) as a
where rn=1;
```

OUTPUT:

customer_id	order_date	product_name	rn
A	2021-01-01	sushi	1
A	2021-01-01	curry	1
B	2021-01-01	curry	1
C	2021-01-01	ramen	1
C	2021-01-01	ramen	1

Q4.What is the most purchased item on the menu and how many times was it purchased by all customers?

ANSWER:

```
select customer_id,count(s.product_id) as purchased ,product_name
from sales as s
join menu as m
on s.product_id=m.product_id
where m.product_id=(
 select s.product_id from sales as s
group by s.product_id
order by count(s.product_id) desc
limit 1)
group by  customer_id;
```

OUTPUT:

customer_id	purchased	product_name
A	3	ramen
B	2	ramen
C	3	ramen

Q5.Which item was the most popular for each customer?

ANSWER:

```
with cte as (
select customer_id,count(s.product_id) as number_of_times_purchased,product_name
from sales as s
join menu as m
on s.product_id=m.product_id
group by  customer_id,product_name
 )
select * from (
select *,dense_rank() over (partition by customer_id order by cte.number_of_times_purchased desc  ) as dn
from cte ) a
where dn=1;
```

OUTPUT:

customer_id	number_of_times_purchased	product_name	dn
A	3	ramen	1
B	2	curry	1
B	2	sushi	1
B	2	ramen	1
C	3	ramen	1

Q6.Which item was purchased first by the customer after they became a member?

ANSWER:

```
select s.customer_id,m.join_date,product_name from members as m
join sales as s
on s.order_date=m.join_date
join menu as mu
on s.product_id=mu.product_id ;
```

OUTPUT:

customer_id	join_date	product_name
A	2021-01-07	curry
C	2021-01-07	ramen

Q7.Which item was purchased just before the customer became a member?

ANSWER 1 :

```
with cte as (
select  s.customer_id,s.order_date,s.product_id,product_name
from sales as s
join members as m
on  s.customer_id=m.customer_id
join menu as mu
on s.product_id=mu.product_id
where s.order_date < m.join_date

order by customer_id,order_date desc)
select * from(
select *,dense_rank() over (partition by cte.customer_id order by cte.order_date desc ) as dn
from cte  ) as a
where dn=1 ;
```

OUTPUT:

customer_id	order_date	product_id	product_name	dn
A	2021-01-01	1	sushi	1
A	2021-01-01	2	curry	1
B	2021-01-04	1	sushi	1

Q7.Which item was purchased just before the customer became a member?

ANSWER 2. :

```
select  s.customer_id,s.order_date,s.product_id,product_name
from sales as s
join members as m
on  s.customer_id=m.customer_id
join menu as mu
on s.product_id=mu.product_id
where s.order_date < m.join_date
and s.order_date=
(select max(order_date) from sales as s2
 where s.customer_id=s2.customer_id
 and  s2.order_date < m.join_date )
 order by customer_id;
```

OUTPUT:

customer_id	order_date	product_id	product_name
A	2021-01-01	1	sushi
A	2021-01-01	2	curry
B	2021-01-04	1	sushi

Q8.What is the total items and amount spent for each member before they became a member?

ANSWER :

```
select s.customer_id,count(s.product_id) as total_items ,sum(m.price) as total_amount
from sales as s
join members as mem
on s.customer_id=mem.customer_id
join menu as m
on s.product_id=m.product_id
where s.order_date < mem.join_date
group by s.customer_id
order by customer_id;
```

OUTPUT:

customer_id	total_items	total_amount
A	2	25
B	3	40

Q9.If each $1 spent equates to 10 points and sushi has a 2x points multiplier - how many points would each customer have?

ANSWER :

```sql
select s.customer_id,
sum(case when product_name='sushi' then price* 10 *2
     else price *10  end) as total_points
from sales as s
join menu as m
on s.product_id=m.product_id
group by s.customer_id
order by total_points desc;
```

OUTPUT:

customer_id	total_points
B	940
A	860
C	360

Q10.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?

ANSWER :

```sql
select s.customer_id,
sum(case when order_date between join_date  and adddate(mem.join_date,interval 7 day) then price *10 * 2
    else price *10 end) as total_points
from sales as s
join menu as m
on s.product_id=m.product_id
join members as mem
on s.customer_id=mem.customer_id
where s.customer_id in ('A','B')
and s.order_date between mem.join_date  and '2021-01-31'
group by s.customer_id
order by total_points desc;
```

OUTPUT:

customer_id	total_points
A	1020
B	440

Bonus Questions

The following questions are related creating basic data tables that Danny and his team can use to quickly derive insights without needing to join the underlying tables using SQL.

Recreate the following table output using the available data:

TABLE LIKE THIS

customer_id	order_date	product_name	price	member
A	2021-01-01	curry	15	N
A	2021-01-01	sushi	10	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

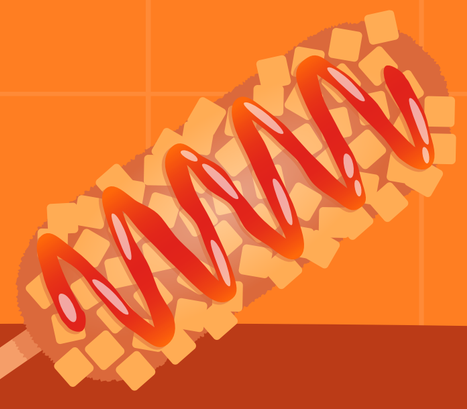
```sql
select s.customer_id,s.order_date,m.product_name,m.price,
(case when s.customer_id=mem.customer_id  and order_date>= join_date then 'Y'
      else 'N' end ) as member
from sales as s
left join members as mem
on s.customer_id=mem.customer_id
join menu as m
on m.product_id=s.product_id
```

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.

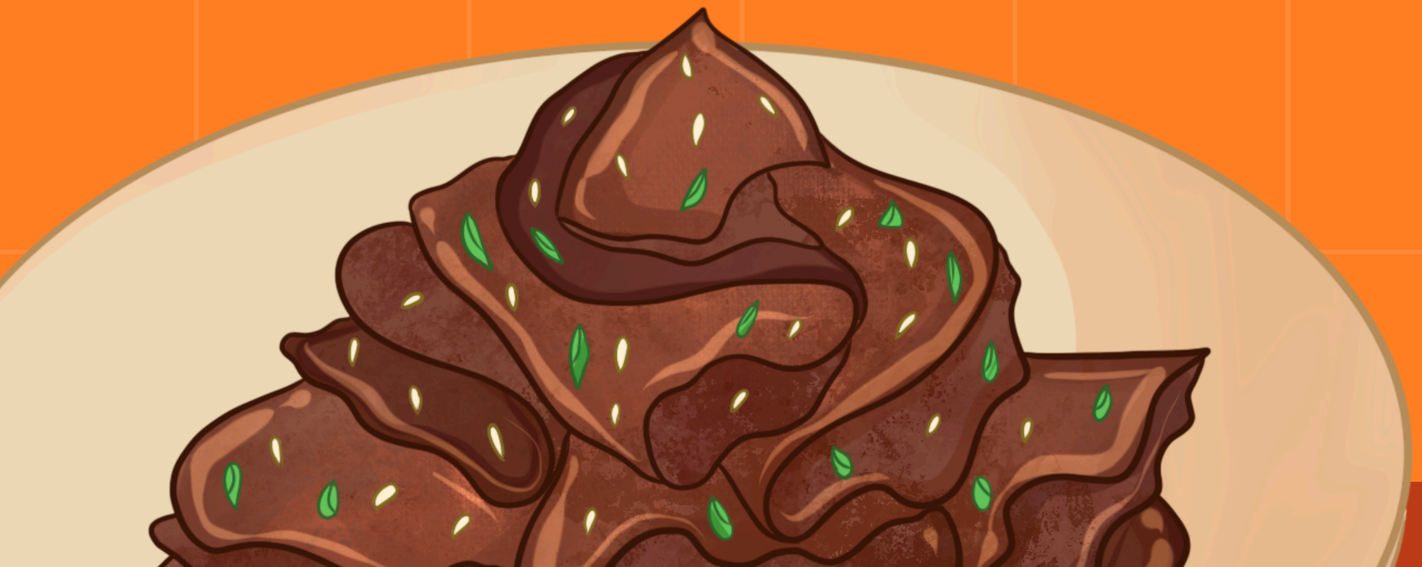
customer_id	order_date	product_name	price	member	ranking
A	2021-01-01	curry	15	N	null
A	2021-01-01	sushi	10	N	null
A	2021-01-07	curry	15	Y	1
A	2021-01-10	ramen	12	Y	2
A	2021-01-11	ramen	12	Y	3
A	2021-01-11	ramen	12	Y	3
B	2021-01-01	curry	15	N	null
B	2021-01-02	curry	15	N	null
B	2021-01-04	sushi	10	N	null
B	2021-01-11	sushi	10	Y	1
B	2021-01-16	ramen	12	Y	2
B	2021-02-01	ramen	12	Y	3
C	2021-01-01	ramen	12	N	null
C	2021-01-01	ramen	12	N	null
C	2021-01-07	ramen	12	N	null

```sql
with cte as (
select s.customer_id,s.order_date,m.product_name,m.price,
(case when s.customer_id=mem.customer_id  and order_date>= join_date then 'Y'
     else 'N' end ) as member

from sales as s
left join members as mem
on s.customer_id=mem.customer_id
join menu as m
on m.product_id=s.product_id
order by customer_id )
select *,
(case when member='Y' then dense_rank() over (partition by cte.customer_id,member order by  cte.order_date )
  else 'null' end) as ranking
  from cte
```



Thank You



RIZWAN SHAH