8 Week SQL Challenge

Case Study #1 - Danny's Dinner

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 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 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!.

Danny has shared with you 3 key datasets for this case study:

1. Sales
2. menu
3. members

You can inspect the entity relationship diagram .

Entity Relationship Diagram

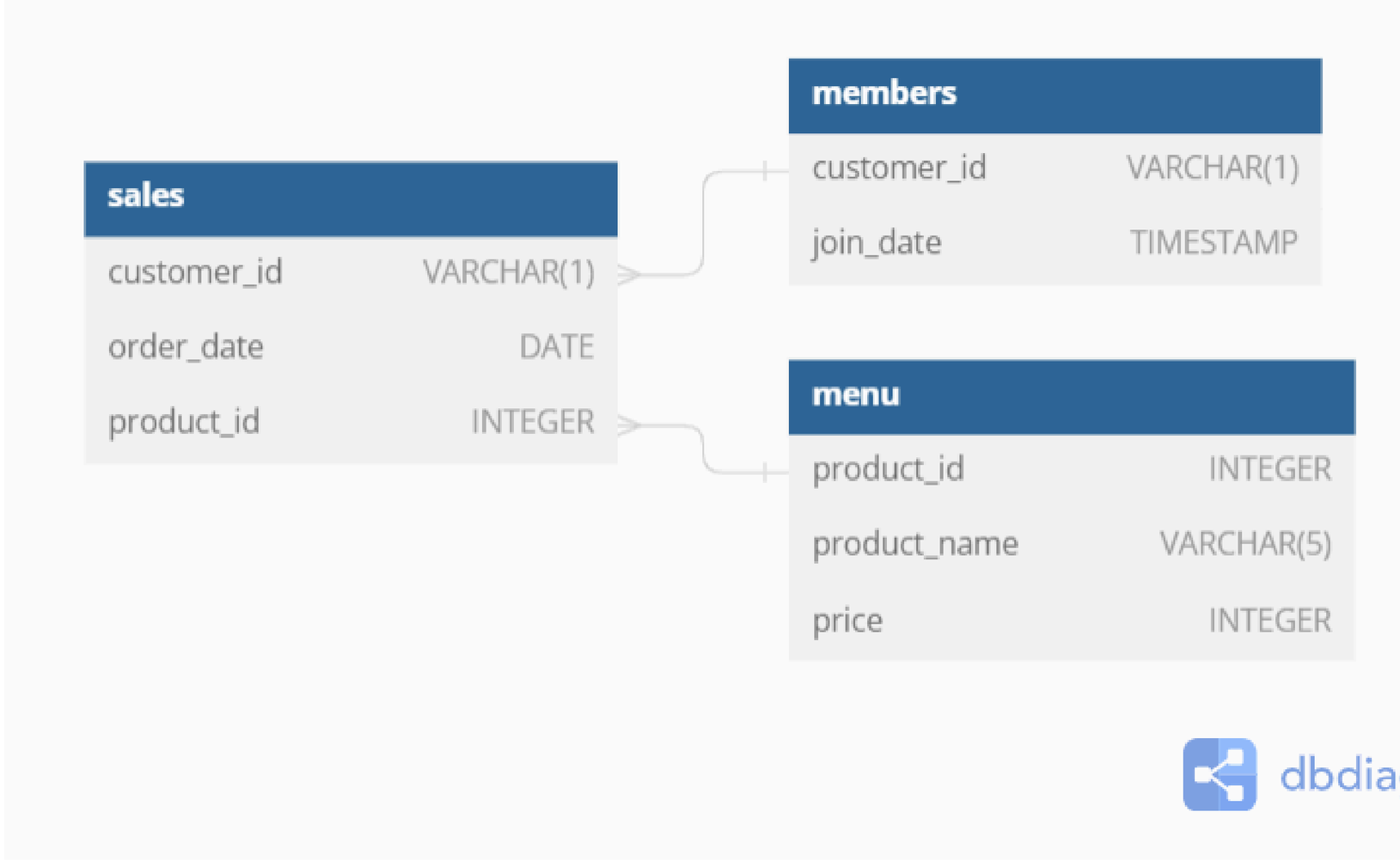

Table 1: sales

The sales table captures all customer_id level purchases with an corresponding order_date and product_id information for when and what menu items were ordered.

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

Table 2: menu

The `menu` table maps the `product_id` to the actual `product_name` and `price` of each menu item.

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

Table 3: members

The final `members` table captures the `join_date` when a `customer_id` joined the beta version of the Danny's Diner loyalty program.

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

Case Study Questions

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

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

	customer_id	total_amt
1	A	76
2	B	74
3	C	36

Case Study Questions

Q2 How many days has each customer visited the restaurant?

```
select s.customer_id , COUNT(distinct order_date) as days from sales s
inner join
menu m  on s.product_id = m.product_id
group by s.customer_id ;
```

	customer_id	days
1	A	4
2	B	6
3	C	2

Case Study Questions

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

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

	customer_id	product_name
1	A	sushi
2	B	curry
3	C	ramen

Case Study Questions

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

```
select Top 1 product_name,COUNT(product_name) as times_purchsed from sales s
inner join
menu m  on s.product_id = m.product_id
group by product_name
order by times_purchsed desc ;
```

	product_name	times_purchsed
1	ramen	8

Case Study Questions

Q5 Which item was the most popular for each customer?

```
with cte as (select s.customer_id,product_name , Count(product_name) as order_count ,
DENSE_RANK() over (partition by customer_id order by Count(product_name) desc ) as rnk
from sales s
inner join
menu m  on s.product_id = m.product_id
group by customer_id , product_name)
select customer_id,product_name,order_count from cte
where rnk = 1 ;
```

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

Case Study Questions

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

```
with cte as (select mem.customer_id , product_name , order_date , join_date,
DENSE_RANK() over (partition by mem.customer_id order by s.order_date) as rnk  from sales s
inner join
members mem on mem.customer_id =s.customer_id
inner join
menu m on m.product_id =s.product_id
where order_date >= join_date)
select customer_id,product_name, order_date, join_date from cte
where rnk = 1 ;
```

	customer_id	product_name	order_date	join_date
1	A	curry	2021-01-07	2021-01-07
2	B	sushi	2021-01-11	2021-01-09

Case Study Questions

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

```
with cte as (select mem.customer_id , product_name , order_date , join_date,
DENSE_RANK() over (partition by mem.customer_id order by s.order_date desc) as rnk  from sales s
inner join
members mem on mem.customer_id =s.customer_id
inner join
menu m on m.product_id =s.product_id
where order_date < join_date)
select customer_id,product_name, order_date, join_date from cte
where rnk = 1 ;
```

	customer_id	product_name	order_date	join_date
1	A	sushi	2021-01-01	2021-01-07
2	A	curry	2021-01-01	2021-01-07
3	B	sushi	2021-01-04	2021-01-09

Case Study Questions

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

```
select  mem.customer_id , STRING_AGG(product_name , ' , ') as items , COUNT(product_name) as total_items
, sum(price) as amt_spent from sales s
inner join
members mem on mem.customer_id =s.customer_id
inner join
menu m on m.product_id =s.product_id
where order_date < join_date
group by mem.customer_id ;
```

	customer_id	items	total_items	amt_spent
1	A	sushi , curry	2	25
2	B	curry , curry , sushi	3	40

Case Study Questions

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

```
with cte as (select * ,
case when product_id = 1 then 20 * price else price*10 End as Points from menu)
select customer_id , Sum(Points) as total_points from sales s
inner join
cte c on c.product_id = s.product_id
group by customer_id ;
```

	customer_id	total_points
1	A	860
2	B	940
3	C	360

Case Study Questions

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?

```
with cte as (select s.customer_id ,order_date, join_date , price,
case when (DATEDIFF (DAY, join_date,order_date) between 0 and  7)  then price * 20
when (DATEDIFF (WEEK, join_date,order_date) != 1) then 0 else 0 end as points
from sales s
inner join
members mem on mem.customer_id=s.customer_id
inner join
menu m on m.product_id = s.product_id)
select customer_id , SUM(points) as Point from cte
group by customer_id;
```

	customer_id	Point
1	A	1020
2	B	440

Bonus Questions

Join All The Things

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.

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

	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	B	2021-02-01	ramen	12	Y
13	C	2021-01-01	ramen	12	N
14	C	2021-01-01	ramen	12	N
15	C	2021-01-07	ramen	12	N

Bonus Questions

Rank All The Things

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.

```
with cte as (select s.customer_id,s.order_date ,
m.product_name,m.price ,
(Case when order_date >= join_date
then 'Y' else 'N' end ) as Member from sales s
inner join
menu m on m.product_id = s.product_id
left join
members mem on mem.customer_id=s.customer_id )
select * ,
(Case when Member = 'Y' then dense_rank ()
over(partition by customer_id,Member
order by order_date)
else null end) as ranking
from cte ;
```

	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
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