MySQL case study

Afroz Arman

8WEEKSQLCHALLENGE.COM

CASE STUDY #1

DANNY'S DINER

THE TASTE OF SUCCESS

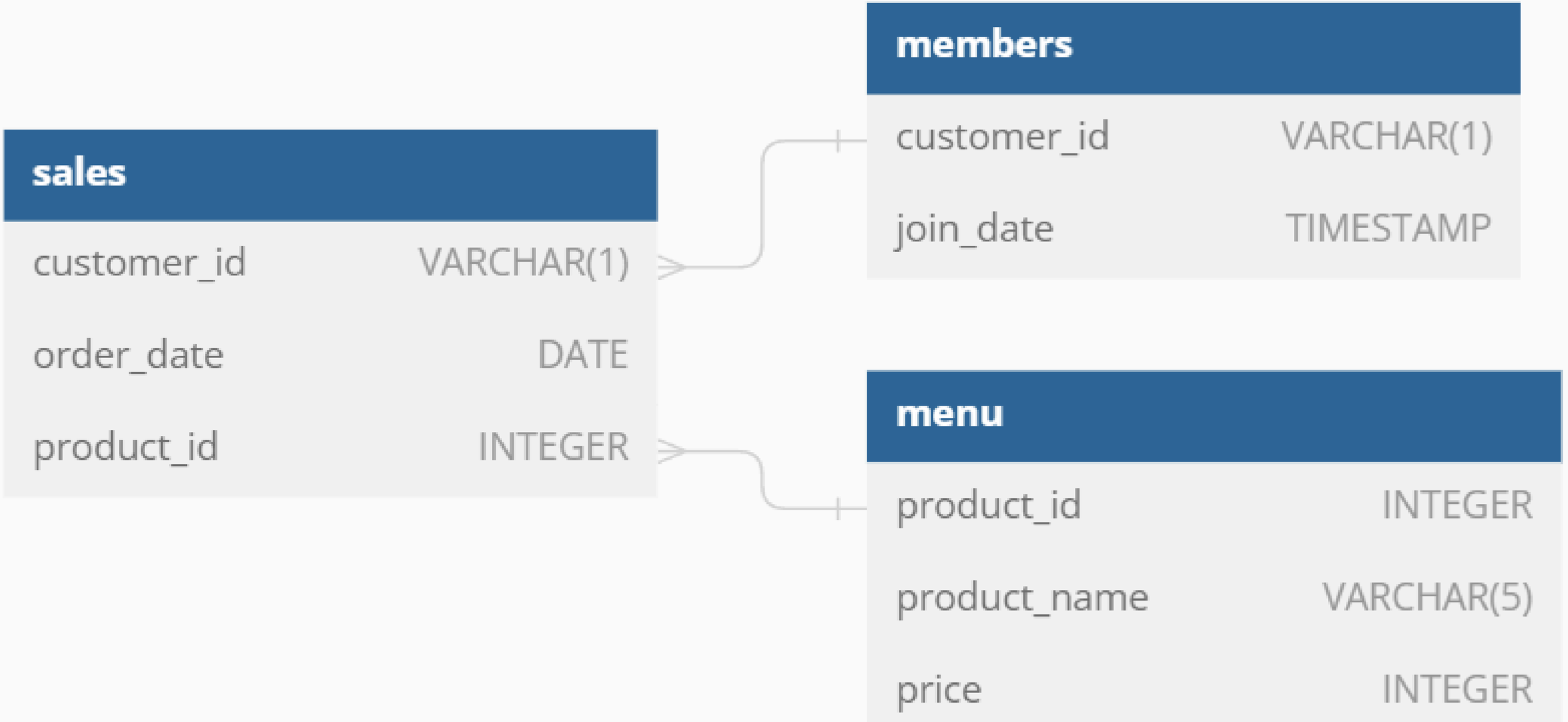
DATAWITHDANNY.COM

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.

Entity Relationship Diagram



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:

sales

menu

members

Table 1: sales

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

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

Table 3: members

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

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


```sql
select s.customer_id, sum(m.price) from sales s
join menu m
using (product_id)
group by s.customer_id
```

Result:

Result Grid | Filter Rows:

customer_id	sum(m.price)
A	76
B	74
C	36

How many days has each customer visited the restaurant?

```sql
select customer_id, count(distinct(order_date)) as visited_days from sales
group by customer_id
```

Result:

	customer_id	visited_days
	A	4
▶	B	6
	C	2

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

```sql
With table1 as ( select customer_id,product_name, row_number()
over (partition by customer_id order by order_date)
as Row_no from sales s
Join menu m
using (product_id)
)
select * from table1
where Row_no = 1
```

Result:

customer_id	product_name	Row_no
A	sushi	1
B	curry	1
C	ramen	1

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

```sql
select m.product_name, count(s.product_id) as item_ordered from sales s
Join menu m
using (product_id)
group by product_name
order by item_ordered desc
Limit 1
```

Result:

	product_name	item_ordered
▶	ramen	8

Which item was the most popular for each customer?

```sql
with cte1 as(
select s.customer_id, m.product_name, count(s.product_id) as item_ordered
from sales s
Join menu m
using (product_id)
group by m.product_name, s.customer_id
order by item_ordered desc
),
cte2 as(select customer_id, product_name, item_ordered, dense_rank() over(partition by
customer_id order by item_ordered desc) as drank from cte1)
select * from cte2  where drank = 1
```

Result:

customer_id	product_name	item_ordered	drank
A	ramen	3	1
B	curry	2	1
B	sushi	2	1
B	ramen	2	1
C	ramen	3	1

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

```sql
with cte1 as (
select s.customer_id, s.order_date, m.product_name,
row_number() over (partition by customer_id order by order_date) as rnk
from menu m
JOIN sales s
using (product_id)
join members mb
using(customer_id)
where order_date>=join_date
)
select * from cte1
where rnk=1
```

Result:

customer_id	order_date	product_name	rnk
A	2021-01-07	curry	1
B	2021-01-11	sushi	1

Which item was purchased just before the customer became a member

```sql
with cte1 as (
select s.customer_id, s.order_date, m.product_name,
row_number() over (partition by customer_id order by order_date) as rnk
from menu m
JOIN sales s
using (product_id)
join members mb
using(customer_id)
where s.order_date < mb.join_date
)
select * from cte1
where rnk=1
```

Result:

	customer_id	order_date	product_name	rnk
▶	A	2021-01-01	sushi	1
	B	2021-01-01	curry	1

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

```sql
select s.customer_id, count(s.product_id) as items,
sum(m.price) as amount
from sales s
join menu m
using (product_id)
join members mb
using (customer_id)
where order_date < join_date
group by s.customer_id
```

Result:

customer_id	items	amount
B	3	40
A	2	25

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

```sql
select s.customer_id, s.order_date, m.product_name, m.price, mb.join_date,
CASE
when product_name = 'sushi' then price*20
else price*10
end as earning_points
from sales s
join menu m
using (product_id)
join members mb
using (customer_id)
```

Result:

customer_id	order_date	product_name	price	join_date	earning_points
B	2021-01-11	sushi	10	2021-01-09	200
B	2021-01-04	sushi	10	2021-01-09	200
A	2021-01-01	sushi	10	2021-01-07	200
B	2021-01-02	curry	15	2021-01-09	150
B	2021-01-01	curry	15	2021-01-09	150

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?

```sql
with cte1 as (select s.customer_id,
CASE
when order_date >= join_date and DATE_ADD(join_date, interval 6 day)
then price*20
end as earning_points
from sales s
join menu m
using (product_id)
join members mb
using (customer_id)
)
select customer_id, sum(earning_points) as total_points
from cte1
group by customer_id
```

Result:

	customer_id	total_points
▶	B	680
	A	1020

Thanks