```
--Case Study 1
--Danny's Diner
--Completed by: Srikanth Thella
--GitHub: https://github.com/srikanththella
--LinkedIn: https://www.linkedin.com/in/srikanth-thella-82231a216/



--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


--Case Study Questions:

--What is the total amount each customer spent at the restaurant?
--How many days has each customer visited the restaurant?
--What was the first item from the menu purchased by each customer?
--What is the most purchased item on the menu and how many times was it purchased by all
  customers?
--Which item was the most popular for each customer?
--Which item was purchased first by the customer after they became a member?
--Which item was purchased just before the customer became a member?
--What is the total items and amount spent for each member before they became a member?
--If each $1 spent equates to 10 points and sushi has a 2x points multiplier  - how many points
  would each customer have?
--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?

--Topics Covered:
--Common Table Expressions
--Group By Aggregates
--Window Functions for ranking
--Table Joins


create database DataWithDanny;

use DataWithDanny;

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');


select * from members;
select * from menu;
select * from sales;


--=======================================solutions==============================
```

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

select S.customer_id,SUM(M.price) as "Total Amount" from sales as S
left join menu as M
on S.product_id = M.product_id
group by S.customer_id

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

select customer_id,COUNT(distinct(order_date)) as "No. of Days visited" from sales
group by customer_id

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

with first_order_cte as(
select customer_id,order_date,product_name,
RANK() OVER(PARTITION BY customer_id ORDER BY order_date ASC) as "Rank",
ROW_NUMBER() OVER(PARTITION BY customer_id ORDER BY order_date ASC) as "Row Number",
DENSE_RANK() OVER(PARTITION BY customer_id ORDER BY order_date ASC) as "Dense Rank"
from sales
left join menu on sales.product_id = menu.product_id
)
select customer_id,product_name from first_order_cte
where Rank =1

--4.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(1) as "Orders Count" from sales left join menu on
  sales.product_id = menu.product_id
group by product_name
order by COUNT(1) DESC

--5.Which item was the most popular for each customer?
with popular_item as(
select customer_id,product_name,COUNT(1) as "Orders_Count" ,
RANK() OVER(PARTITION BY customer_id ORDER BY COUNT(1) DESC) as "Rank"
from sales left join menu on sales.product_id = menu.product_id
group by customer_id ,product_name
)
select customer_id,product_name from popular_item
where Rank = 1

--6.Which item was purchased first by the customer after they became a member?
with member_product as(
select S.customer_id ,order_date,product_id,
RANK() OVER(PARTITION BY S.customer_id ORDER BY order_date ASC) as "RNK"
from sales as S left join members as M on S.customer_id = M.customer_id
where order_date>=join_date
)
select MP.customer_id,MP.order_date,M.product_name from member_product as MP
left join menu as M on MP.product_id = M.product_id
where MP.RNK = 1

--or
with mem_orders as (
select sales.customer_id,order_date,sales.product_id,product_name,join_date,
```

```
RANK() OVER(PARTITION BY sales.customer_id ORDER BY order_date ASC) as "RNK"
from sales
left join menu on sales.product_id = menu.product_id
left join members on sales.customer_id = members.customer_id
where order_date>=join_date
)
select customer_id,order_date,product_name from mem_orders
where RNK = 1

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

with before_mem as (
select sales.customer_id,sales.order_date,sales.product_id,menu.product_name,members.join_date,
RANK() OVER(PARTITION BY sales.customer_id ORDER BY order_date DESC) as "RNK"
from sales
left join menu on sales.product_id = menu.product_id
left join members on sales.customer_id = members.customer_id
where order_date<join_date
)
select customer_id,order_date,product_name from before_mem
where RNK = 1

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

select S.customer_id,COUNT(1) as "Item_Count",SUM(price) as "Total Amount" from sales as S
left join menu as M on S.product_id = M.product_id
left join members as MM on S.customer_id = MM.customer_id
where order_date<join_date
group by S.customer_id

--9.If each $1 spent equates to 10 points and sushi has a 2x points multiplier  - how many
  points would each customer have?
with points_mem as (
select S.customer_id,order_date,S.product_id,product_name,price,
case
when S.product_id = 1 then price*10*2
else price*10
end as "points"
from sales as S left join menu as M on S.product_id = M.product_id
)
select customer_id,SUM(points) as "Total Points" from points_mem
group by customer_id;

--or

select S.customer_id,
SUM(case
when S.product_id = 1 then price*10*2
else price*10
end) as "points"
from sales as S left join menu as M on S.product_id = M.product_id
group by S.customer_id


--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?
```

```sql
select S.customer_id,order_date,S.product_id,product_name,price,join_date ,
case
when order_date between join_date and DATEADD(DAY,6,join_date) then price*10*2
when S.product_id = 1 then price*2*10
else price*10
end as "Points"
from sales as S
left join menu as M on S.product_id = M.product_id
left join members as MM on S.customer_id = MM.customer_id
where DATEPART(month,order_date)=1

---final answer (take inner join instead of left join for only results of A and B)

select S.customer_id,
SUM(case
when order_date between join_date and DATEADD(DAY,6,join_date) then price*10*2
when S.product_id = 1 then price*2*10
else price*10
end) as "Points"
from sales as S
left join menu as M on S.product_id = M.product_id
left join members as MM on S.customer_id = MM.customer_id
where DATEPART(month,order_date)=1
group by S.customer_id


--Bonus Questions
--Join All the Things

select S.customer_id,order_date,product_name,price,
case
when order_date<join_date or join_date IS NULL then 'N'
else 'Y'
end as member
from sales as S
left join menu as M on S.product_id = M.product_id
left join members as MM on S.customer_id = MM.customer_id
order by S.customer_id,order_date,price desc

--Rank all the things

select S.customer_id,order_date,product_name,price,
case
when order_date<join_date or join_date IS NULL then 'N'
else 'Y'
end as member,
case
when join_date IS NULL or order_date<join_date then NULL
else RANK() OVER(PARTITION BY S.customer_id,(case
when order_date<join_date or join_date IS NULL then 'N'
else 'Y'
end) ORDER BY order_date)
end as "ranking"
from sales as S
left join menu as M on S.product_id = M.product_id
left join members as MM on S.customer_id = MM.customer_id
order by S.customer_id,order_date,price desc
```

```sql
--or
with ranking_tab as (
select S.customer_id,order_date,product_name,price,
case
when order_date<join_date or join_date IS NULL then 'N'
else 'Y'
end as member
from sales as S
left join menu as M on S.product_id = M.product_id
left join members as MM on S.customer_id = MM.customer_id
)
select customer_id,order_date,product_name,price,member,
case
when member ='N' then NULL
else RANK() OVER(PARTITION BY customer_id,member ORDER BY order_date)
end as "ranking"
from ranking_tab
order by customer_id,order_date,price desc
```