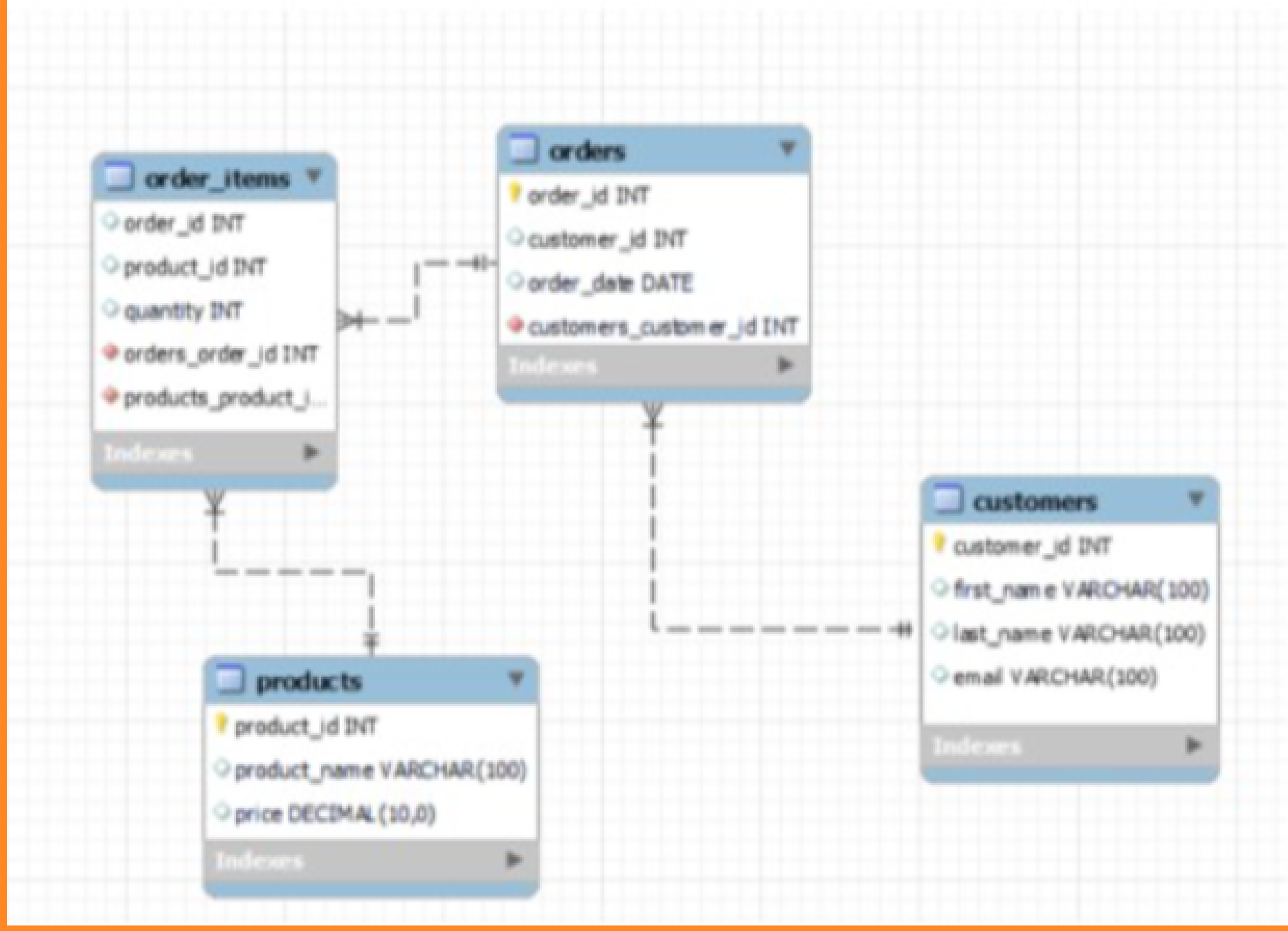
SQL CASE STUDY

DATA IN MOTION
TINY SHOP SALES

DATA IN MOTION

Mythily Ramanathan

Entity Relationship Diagram (ERD)



Mythily Ramanathan

PostgreSQL

```sql
--1) Which product has the highest price? Only return a single row.

SELECT product_name
FROM products
ORDER BY price DESC
LIMIT 1;
```

product_name
Product M

Product M has the highest Price

Used : ORDER BY , LIMIT

Mythily Ramanathan

PostgreSQL

```sql
--2. Which customer has made the most orders?

with rank_cte AS
(
SELECT
CONCAT (first_name,' ',last_name) AS Customer,
RANK() OVER(ORDER BY COUNT(o.order_id) DESC) AS rank_num
FROM customers C
JOIN orders o
ON C.customer_id= o.customer_id
GROUP BY Customer
)
SELECT customer
FROM rank_cte
WHERE rank_num = 1
```

customer
Bob Johnson
John Doe
Jane Smith

Grouped by customer,
used rank() to reverse sort by the number of orders,
I have displayed all customers with rank =1.

Used : Windows Function - RANK(),
CTE, CONCAT

Mythily Ramanathan

PostgreSQL

```sql
--3. How many orders were placed in May, 2023?

with cte AS (
SELECT
order_id,
EXTRACT(MONTH FROM order_date) AS mon,
EXTRACT(YEAR FROM order_date) AS yr,
order_date
FROM orders
)
SELECT
COUNT(order_id) AS order_count
FROM cte
WHERE (mon = 5) AND (yr = 2023)
-- NOTE : & operator does not work in PostgreSQL. use AND instead
--      use the above format to get MONTH & YEAR.
```

order_count
16

Displaying the count of orders placed in May,2023

Used : CTE,
date-time function EXTRACT,
logical AND

Mythily Ramanathan

PostgreSQL

```sql
--4) What's the total revenue per product?

SELECT
P.product_name,
SUM(oi.quantity*P.price) AS revenue
FROM order_items oi
JOIN products P
USING (product_id)
GROUP BY product_name
ORDER BY product_name
```

product_name	revenue
Product A	50.00
Product B	135.00
Product C	160.00
Product D	75.00
Product E	90.00
Product F	210.00
Product G	120.00
Product H	135.00
Product I	150.00
Product J	330.00
Product K	180.00
Product L	195.00
Product M	420.00

Revenue = Price & Quantity

Mythily Ramanathan

```sql
--5. Find the date with the highest revenue.

SELECT
o.order_date,
SUM(oi.quantity*P.price) AS revenue
FROM order_items oi
JOIN products P
USING (product_id)
JOIN orders o
USING (order_id)
GROUP BY order_date
ORDER BY revenue DESC
LIMIT 1
```

order_date	revenue
2023-05-16	340.00

Mythily Ramanathan

```sql
--5. Find the date with the highest revenue.

SELECT
o.order_date,
SUM(oi.quantity*P.price) AS revenue,
RANK() OVER(ORDER BY(SUM(oi.quantity*P.price))DESC) AS rnk
FROM order_items oi
JOIN products P
USING (product_id)
JOIN orders o
USING (order_id)
GROUP BY order_date
LIMIT 1
```

order_date	revenue	rnk
2023-05-16	340.00	1

RANK() Window function

Mythily Ramanathan

```sql
--6. Find the product that has seen the biggest increase in sales quantity over the previous month.
with cte AS(
SELECT
order_date AS date,
split_part(product_name,'t',2) AS NAME,
quantity AS qty,
LAG(quantity,1,0)OVER(ORDER BY order_date)AS prev,
quantity-(LAG(quantity,1)OVER(PARTITION BY product_id
                              ORDER BY order_date))AS diff
FROM products P
JOIN order_items oi
USING (product_id)
JOIN orders o
USING (order_id))
SELECT NAME,diff,date,qty,prev
FROM cte
WHERE diff IS NOT NULL
ORDER BY diff DESC
LIMIT 1
```

name	diff	date	qty	prev
B	3	2023-05-04	4	1

Lag Windows Function

Mythily Ramanathan

PostgreSQL

```sql
--7. Find the first order (by date) for each customer.
with CTE AS(
SELECT
first_name,last_name,
customer_id,
order_date,
RANK() OVER(PARTITION BY customer_id ORDER BY order_date ASC)
FROM orders o
JOIN customers C
USING (customer_id)
)
SELECT
concat(first_name,' ',last_name) AS NAME,
order_date AS first_order_date
FROM cte
WHERE rank_no = 1
```

name	first_order_date
John Doe	2023-05-01
Jane Smith	2023-05-02
Bob Johnson	2023-05-03
Alice Brown	2023-05-07
Charlie Davis	2023-05-08
Eva Fisher	2023-05-09
George Harris	2023-05-10
Ivy Jones	2023-05-11
Kevin Miller	2023-05-12
Lily Nelson	2023-05-13
Oliver Patterson	2023-05-14
Quinn Roberts	2023-05-15
Sophia Thomas	2023-05-16

In RANK() Windows Function,
partition by gives for each customer,
order by date Asc gives rank 1 for the first order for each customer

Mythily Ramanathan

PostgreSQL

```sql
--8. Find the top 3 customers who have ordered
--the most distinct products
with CTE AS(
SELECT
first_name,last_name,
C.customer_id,
COUNT(DISTINCT product_id) AS distinct_prod_count
FROM orders o
JOIN customers C
  USING (customer_id)
JOIN order_items oi
  USING (order_id)
GROUP BY C.customer_id,1,2
)
SELECT
concat(first_name,' ',last_name) AS NAME,distinct_prod_count
FROM cte
ORDER BY customer_id
LIMIT 3
```

name	distinct_prod_count
John Doe	3
Jane Smith	3
Bob Johnson	3

Used : CTE,
aggregate COUNT,
DISTINCT

Mythily Ramanathan

PostgreSQL

```sql
--9. What is the median order total?

with cte AS(
SELECT
order_id,
SUM(oi.quantity*P.price) AS order_total
FROM order_items oi
JOIN products P
USING (product_id)
GROUP BY order_id
),
cte2 AS(
SELECT *,
RANK() OVER(ORDER BY order_total ASC) AS rn_asc,
RANK() OVER(ORDER BY order_total DESC) AS rn_desc
FROM cte
)
SELECT round(AVG(order_total),2) AS median_order_total FROM ct
WHERE ABS(rn_asc-rn_desc) <=1
```

median_order_total
112.50

The median is the middle value in a set of data.

I have used nested CTEs.

First cte calculates order_total,
cte2 ranks the order_total by ASC & DESC
to find the middle values`s rank,
Finally we find avg of the 2 middle values to get the Median

Mythily Ramanathan

PostgreSQL

```sql
--10. For each order, determine if it was
--'Expensive' (total over 300),
--'Affordable' (total over 100), or 'Cheap'.


with cte AS(
SELECT
order_id,
SUM(oi.quantity*P.price) AS order_total
FROM order_items oi
JOIN products P
USING (product_id)
GROUP BY order_id
)
SELECT *,
CASE WHEN (order_total > 300) THEN 'Expensive'
  WHEN (order_total > 100) THEN 'Affordable'
  ELSE 'Cheap'
  END
FROM cte
ORDER BY order_id--)

```

order_id	order_total	case
1	35.00	Cheap
2	75.00	Cheap
3	50.00	Cheap
4	80.00	Cheap
5	50.00	Cheap
6	55.00	Cheap
7	85.00	Cheap
8	145.00	Affordable
9	140.00	Affordable
10	285.00	Affordable
11	275.00	Affordable
12	80.00	Cheap
13	185.00	Affordable
14	145.00	Affordable
15	225.00	Affordable
16	340.00	Expensive

Case Statements

Mythily Ramanathan

PostgreSQL

```sql
--11. Find customers who have ordered the product
--with the highest price.

with cte AS(
SELECT
first_name,last_name,
RANK() OVER(ORDER BY price DESC) AS rn
FROM orders o
JOIN customers C
  USING (customer_id)
JOIN order_items oi
  USING (order_id)
JOIN products P
  USING (product_id)
)
SELECT
concat(first_name,' ',last_name) AS NAME
FROM cte
WHERE rn =1
```

name
Ivy Jones
Sophia Thomas

RANK() Window function

Mythily Ramanathan

PostgreSQL

```sql
--12. Which product has been bought the most in terms of
--quantity?
with cte AS(
SELECT
product_name,
SUM(quantity) AS total_qty
FROM products
JOIN order_items oi
  USING (product_id)
 GROUP BY product_id
)
SELECT
*
FROM cte
ORDER BY total_qty DESC
LIMIT 1
```

product_name	total_qty
Product B	9

Common Table Expression (CTE)

Mythily Ramanathan