SQL Case Study: Tiny Shop Sales

This case-study uses MySQL. I have been exposed to the following areas of SQL:

- Basic Aggregations.
- CASE WHEN statements (Searched Case Expression)
- Window Functions
- Joins
- Date Time Functions
- CTEs
- Subquery

Case Study Questions

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

2) Which customer has made the most orders?

3) What's the total revenue per product?

4) Find the day with the highest revenue.

5) Find the first order (by date) for each customer.

6) Find the top 3 customers who have ordered the most distinct products

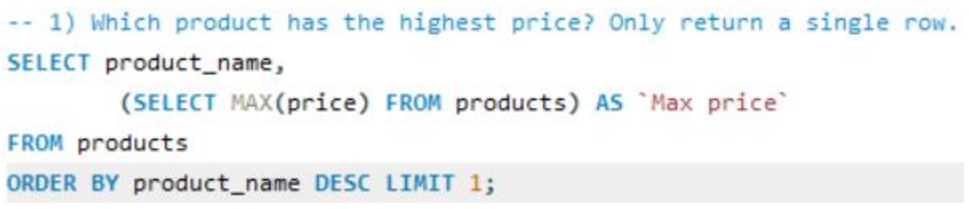
7) Which product has been bought the least in terms of quantity?

8) What is the median order total?

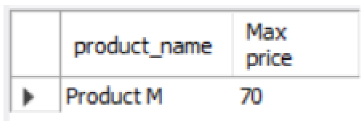
9) For each order, determine if it was 'Expensive' (total over 300), 'Affordable' (total over 100), or 'Cheap'.

10) Find customers who have ordered the product with the highest price.

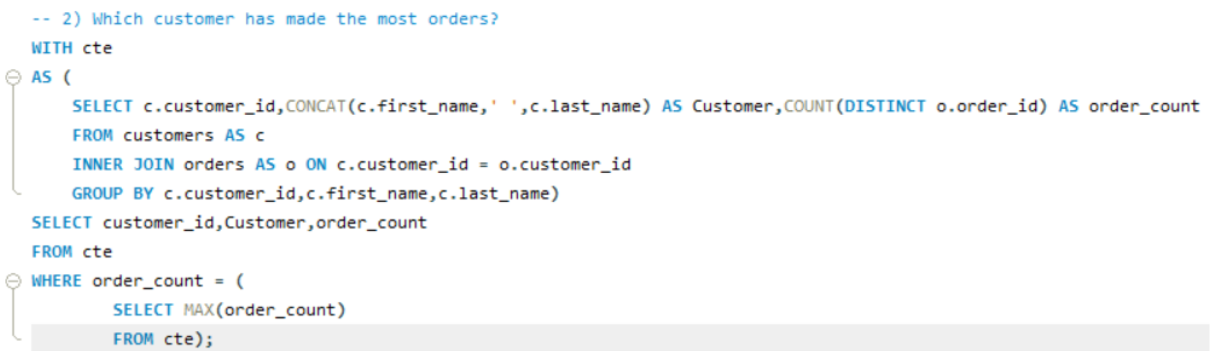
SQL Queries and their resultant output.



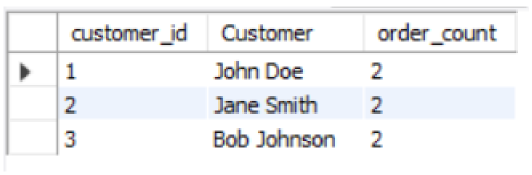
```sql
-- 1) Which product has the highest price? Only return a single row.
SELECT product_name,
        (SELECT MAX(price) FROM products) AS `Max price`
FROM products
ORDER BY product_name DESC LIMIT 1;
```



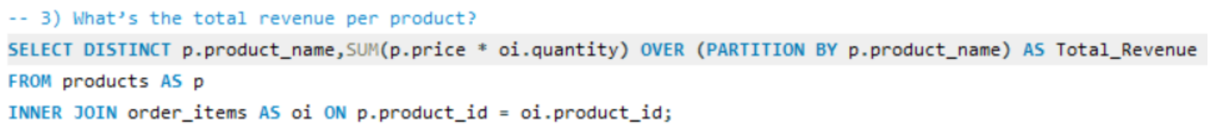
product_name	Max price
Product M	70



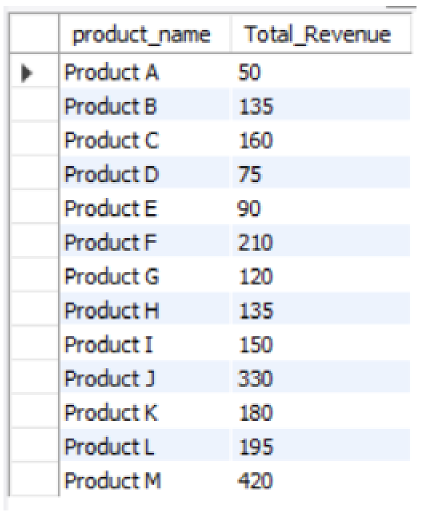
```sql
-- 2) Which customer has made the most orders?
WITH cte
AS (
    SELECT c.customer_id,CONCAT(c.first_name,' ',c.last_name) AS Customer,COUNT(DISTINCT o.order_id) AS order_count
    FROM customers AS c
    INNER JOIN orders AS o ON c.customer_id = o.customer_id
    GROUP BY c.customer_id,c.first_name,c.last_name)
SELECT customer_id,Customer,order_count
FROM cte
WHERE order_count = (
        SELECT MAX(order_count)
        FROM cte);
```



customer_id	Customer	order_count
1	John Doe	2
2	Jane Smith	2
3	Bob Johnson	2

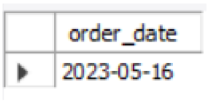


```sql
-- 3) What's the total revenue per product?
SELECT DISTINCT p.product_name,SUM(p.price * oi.quantity) OVER (PARTITION BY p.product_name) AS Total_Revenue
FROM products AS p
INNER JOIN order_items AS oi ON p.product_id = oi.product_id;
```



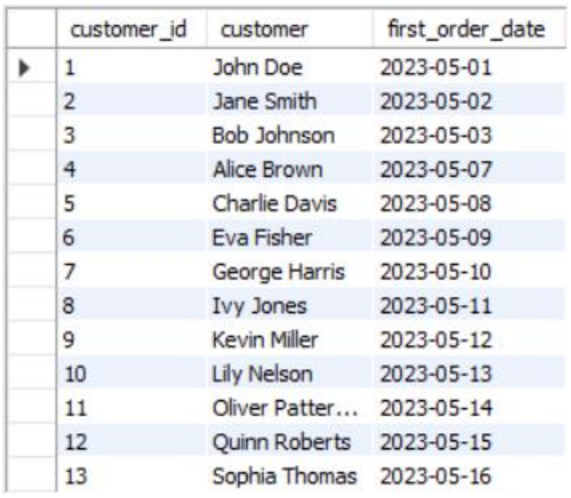
product_name	Total_Revenue
Product A	50
Product B	135
Product C	160
Product D	75
Product E	90
Product F	210
Product G	120
Product H	135
Product I	150
Product J	330
Product K	180
Product L	195
Product M	420

```sql
-- 4) Find the day with the highest revenue.
WITH cte
AS  (SELECT p.price,oi.quantity,o.order_date, SUM(p.price * oi.quantity)
        OVER (PARTITION BY o.order_date) AS total_price
    FROM products AS p
    INNER JOIN order_items AS oi
        ON p.product_id = oi.product_id
    INNER JOIN orders AS o
        ON oi.order_id = o.order_id
    GROUP BY  p.price, o.order_date, oi.quantity )
SELECT order_date
FROM cte
WHERE total_price =
    (SELECT MAX(total_price)
    FROM cte) limit 1;
```

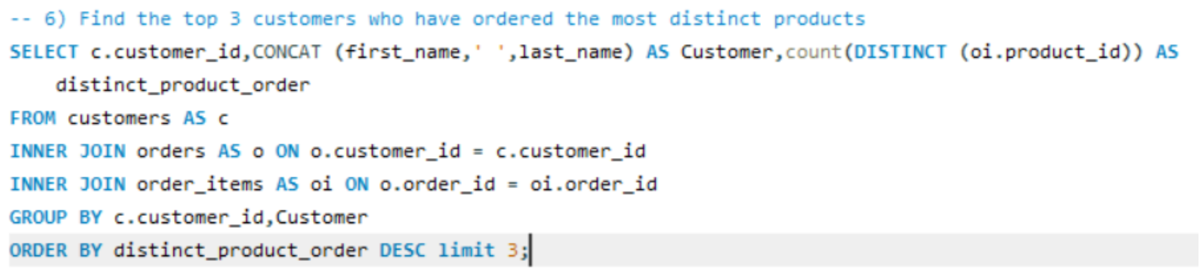


order_date
2023-05-16

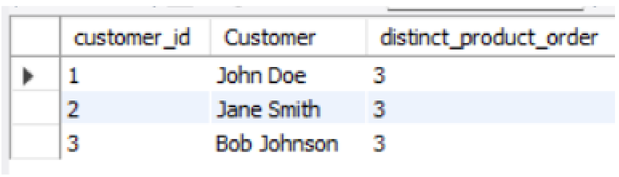
```sql
-- 5) Find the first order (by date) for each customer.
SELECT c.customer_id, concat(c.first_name,' ', c.last_name) AS customer, MIN(o.order_date) AS first_order_date
FROM customers AS c
INNER JOIN orders AS o
    ON c.customer_id = o.customer_id
GROUP BY  c.customer_id, c.first_name, c.last_name ;
```



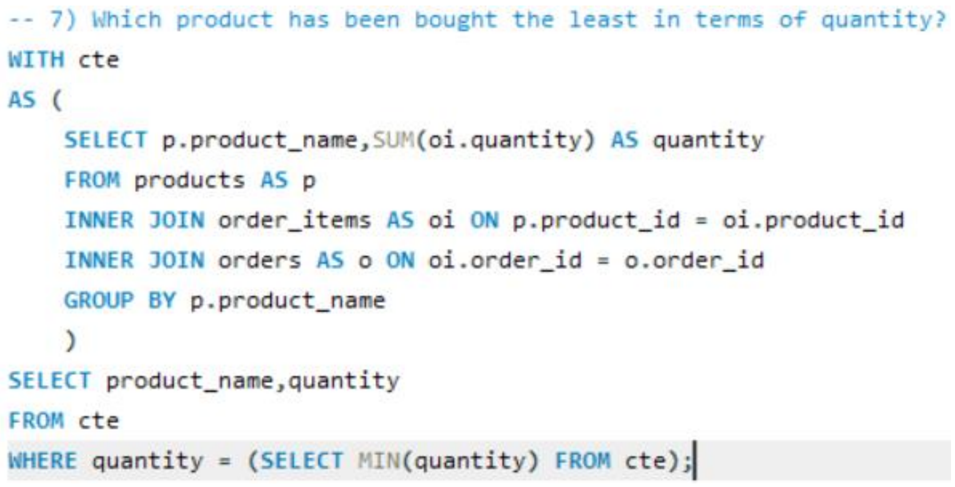
customer_id	customer	first_order_date
1	John Doe	2023-05-01
2	Jane Smith	2023-05-02
3	Bob Johnson	2023-05-03
4	Alice Brown	2023-05-07
5	Charlie Davis	2023-05-08
6	Eva Fisher	2023-05-09
7	George Harris	2023-05-10
8	Ivy Jones	2023-05-11
9	Kevin Miller	2023-05-12
10	Lily Nelson	2023-05-13
11	Oliver Patter...	2023-05-14
12	Quinn Roberts	2023-05-15
13	Sophia Thomas	2023-05-16



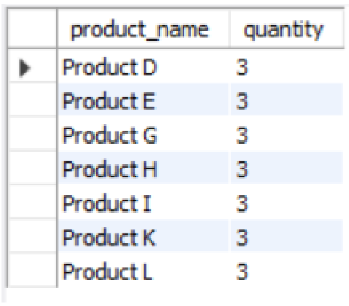
```sql
-- 6) Find the top 3 customers who have ordered the most distinct products
SELECT c.customer_id,CONCAT (first_name,' ',last_name) AS Customer,count(DISTINCT (oi.product_id)) AS
    distinct_product_order
FROM customers AS c
INNER JOIN orders AS o ON o.customer_id = c.customer_id
INNER JOIN order_items AS oi ON o.order_id = oi.order_id
GROUP BY c.customer_id,Customer
ORDER BY distinct_product_order DESC limit 3;
```



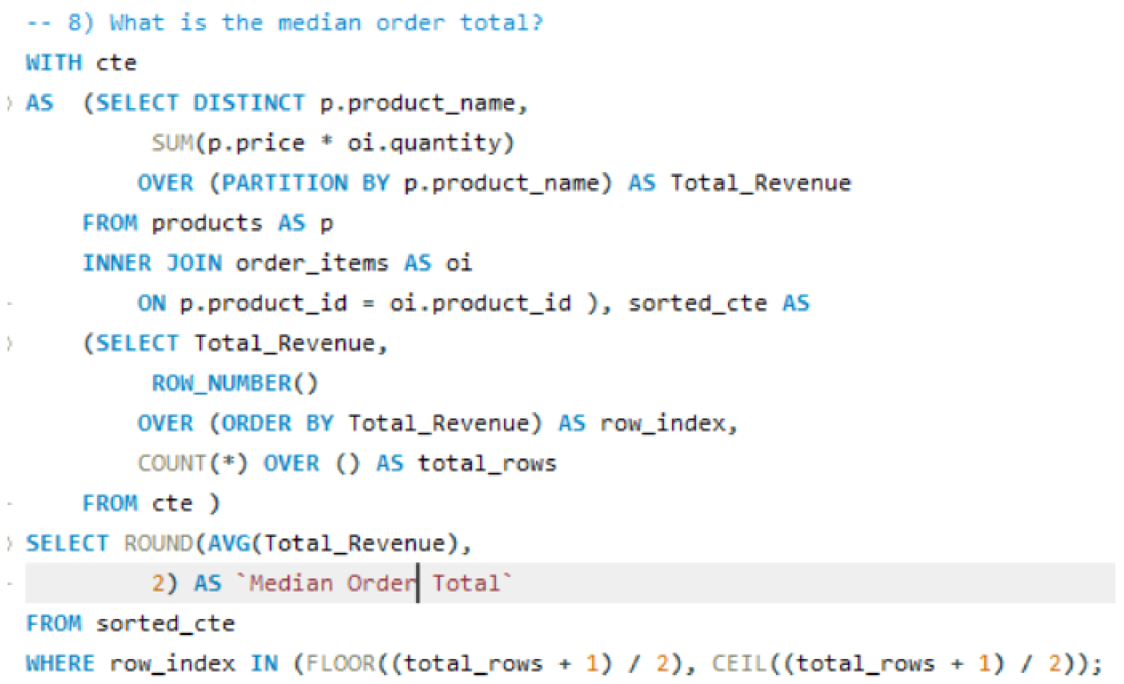
customer_id	Customer	distinct_product_order
1	John Doe	3
2	Jane Smith	3
3	Bob Johnson	3



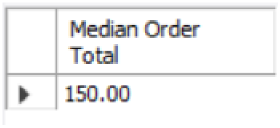
```sql
-- 7) Which product has been bought the least in terms of quantity?
WITH cte
AS (
    SELECT p.product_name,SUM(oi.quantity) AS quantity
    FROM products AS p
    INNER JOIN order_items AS oi ON p.product_id = oi.product_id
    INNER JOIN orders AS o ON oi.order_id = o.order_id
    GROUP BY p.product_name
    )
SELECT product_name,quantity
FROM cte
WHERE quantity = (SELECT MIN(quantity) FROM cte);
```



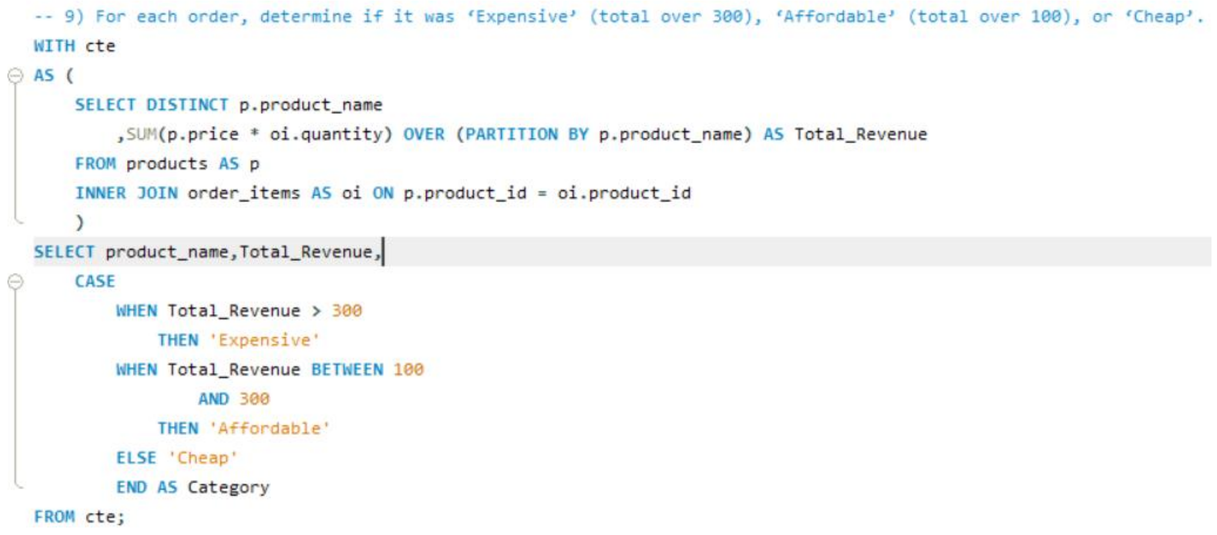
product_name	quantity
Product D	3
Product E	3
Product G	3
Product H	3
Product I	3
Product K	3
Product L	3



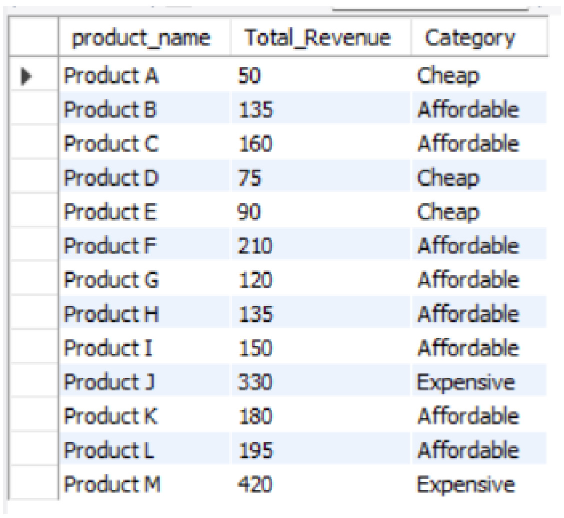
```sql
-- 8) What is the median order total?
WITH cte
AS  (SELECT DISTINCT p.product_name,
         SUM(p.price * oi.quantity)
        OVER (PARTITION BY p.product_name) AS Total_Revenue
    FROM products AS p
    INNER JOIN order_items AS oi
        ON p.product_id = oi.product_id ), sorted_cte AS
    (SELECT Total_Revenue,
         ROW_NUMBER()
        OVER (ORDER BY Total_Revenue) AS row_index,
        COUNT(*) OVER () AS total_rows
    FROM cte )
SELECT ROUND(AVG(Total_Revenue),
         2) AS `Median Order Total`
FROM sorted_cte
WHERE row_index IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2));
```



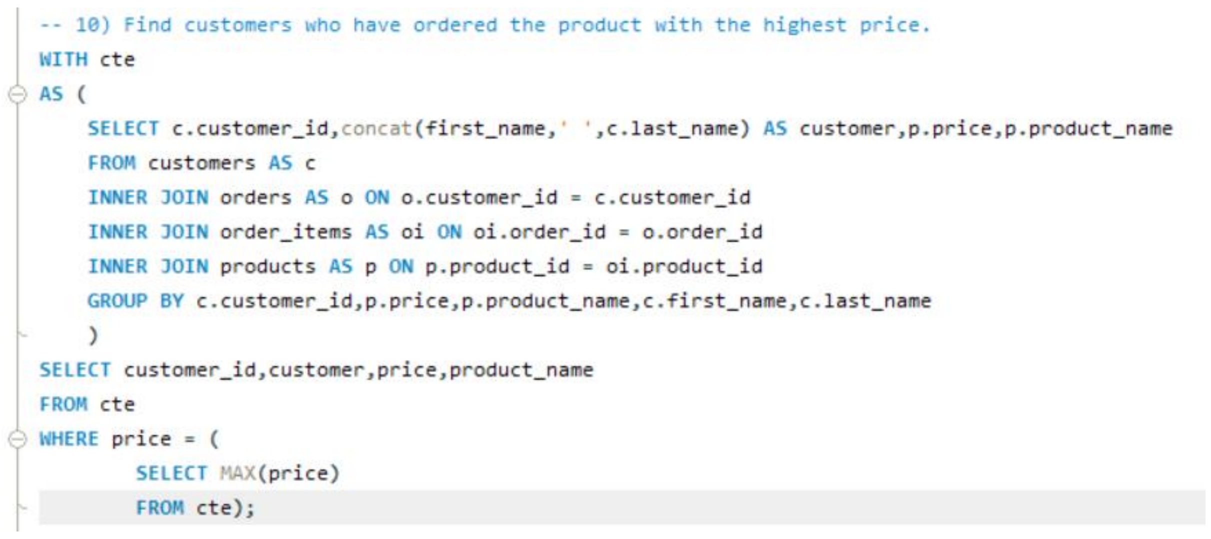
Median Order Total
150.00



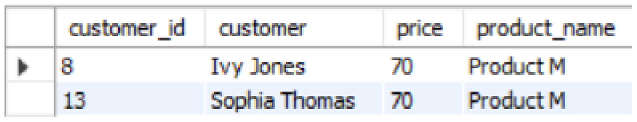
```sql
-- 9) For each order, determine if it was 'Expensive' (total over 300), 'Affordable' (total over 100), or 'Cheap'.
WITH cte
AS (
    SELECT DISTINCT p.product_name
        ,SUM(p.price * oi.quantity) OVER (PARTITION BY p.product_name) AS Total_Revenue
    FROM products AS p
    INNER JOIN order_items AS oi ON p.product_id = oi.product_id
    )
SELECT product_name,Total_Revenue,
    CASE
        WHEN Total_Revenue > 300
            THEN 'Expensive'
        WHEN Total_Revenue BETWEEN 100
                AND 300
            THEN 'Affordable'
        ELSE 'Cheap'
        END AS Category
FROM cte;
```



product_name	Total_Revenue	Category
Product A	50	Cheap
Product B	135	Affordable
Product C	160	Affordable
Product D	75	Cheap
Product E	90	Cheap
Product F	210	Affordable
Product G	120	Affordable
Product H	135	Affordable
Product I	150	Affordable
Product J	330	Expensive
Product K	180	Affordable
Product L	195	Affordable
Product M	420	Expensive



```sql
-- 10) Find customers who have ordered the product with the highest price.
WITH cte
AS (
    SELECT c.customer_id,concat(first_name,' ',c.last_name) AS customer,p.price,p.product_name
    FROM customers AS c
    INNER JOIN orders AS o ON o.customer_id = c.customer_id
    INNER JOIN order_items AS oi ON oi.order_id = o.order_id
    INNER JOIN products AS p ON p.product_id = oi.product_id
    GROUP BY c.customer_id,p.price,p.product_name,c.first_name,c.last_name
    )
SELECT customer_id,customer,price,product_name
FROM cte
WHERE price = (
        SELECT MAX(price)
        FROM cte);
```



customer_id	customer	price	product_name
8	Ivy Jones	70	Product M
13	Sophia Thomas	70	Product M