Question:

Given the following tables:

1. **Customers**
    - customer_id (INT, Primary Key)
    - customer_name (VARCHAR)
    - city (VARCHAR)
2. **Orders**
    - order_id (INT, Primary Key)
    - customer_id (INT, Foreign Key referencing Customers)
    - order_date (DATE)
3. **Order_Items**
    - order_id (INT, Foreign Key referencing Orders)
    - product_id (INT, Foreign Key referencing Products)
    - quantity (INT)
    - price_per_unit (DECIMAL)
4. **Products**
    - product_id (INT, Primary Key)
    - product_name (VARCHAR)
    - category_id (INT, Foreign Key referencing Categories)
5. **Categories**
    - category_id (INT, Primary Key)
    - category_name (VARCHAR)

Task:

For each customer who placed an order in **both 2022 and 2023**, identify their **most frequently purchased product category in 2023** based on the **total quantity ordered**. If there is a tie, return the category with the highest total revenue. If there is still a tie, return the category that appears first alphabetically.

Expected Output:

- customer_name
- city
- top_category_2023 (most frequently purchased category in 2023)
- total_quantity_2023 (total quantity ordered in that category)
- total_revenue_2023 (total revenue from that category)

Constraints:

- Customers should have placed at least one order in both 2022 and 2023.

- If a customer ordered multiple categories in 2023, return only **one** category per customer based on the tie-breaker rules.

**** Query:**

// Grouping all the transactions made in a day into a single variable (helps in understanding the "streak")

WITH TXNS_PER_DAY AS(

SELECT CUSTOMER_ID, TRANSACTION_DATE, COUNT(*) AS TOTAL_TXNS_DAY

FROM TRANSACTIONS

GROUP BY CUSTOMER_ID, TRANSACTION_DATE

)

// Determining if there's at least a single transaction made on the following day

, NEXT_DAY AS(

SELECT A.,*

CASE WHEN B.TOTAL_TXNS_DAY > 0 THEN 1

ELSE 0

END AS NEXT_DAY_TXN_MADE

FROM TXNS_PER_DAY AS A

LEFT JOIN TXNS_PER_DAY AS B

ON A.CUSTOMER_ID = B.CUSTOMER_ID

AND A.TRANSACTION_DATE = DATE_ADD(B.TRANSACTION_DATE,INTERVAL -1 DAY)

)

// Determining the streaks for each customer

, MAX_COUNT_PER_CUSTOMER AS(

*SELECT *,*

CASE WHEN NEXT_DAY_TXN_MADE =1 THEN (1+ (SUM(NEXT_DAY_TXN_MADE) OVER (PARTITION BY CUSTOMER_ID, NEXT_DAY_TXN_MADE ORDER BY TRANSACTION_DATE)))

ELSE 1

END AS CONSECUTIVE_TXNS

FROM NEXT_DAY

)

```
// Selecting the max streak/ longest one for each of the customers
SELECT CUSTOMER_ID, MAX(CONSECUTIVE_TXNS) AS LONGEST_TRANSACTION_STREAK
FROM MAX_COUNT_PER_CUSTOMER
GROUP BY CUSTOMER_ID
;
```