Medium

Table: Customers

```
+---------------+---------+
| Column Name   | Type    |
+---------------+---------+
| customer_id   | int     |
| name          | varchar |
+---------------+---------+
```
customer_id is the column with unique values for this table.
This table contains information about customers.

Table: Orders

```
+---------------+---------+
| Column Name   | Type    |
+---------------+---------+
| order_id      | int     |
| order_date    | date    |
| customer_id   | int     |
| cost          | int     |
+---------------+---------+
```
order_id is the column with unique values for this table.
This table contains information about the orders made by customer_id.
Each customer has **one order per day**.

Write a solution to find the most recent three orders of each user. If a user ordered less than three orders, return all of their orders.

Return the result table ordered by customer_name in **ascending order** and in case of a tie by the customer_id in **ascending order**. If there is still a tie, order them by order_date in **descending order**.

The result format is in the following example.

Example 1:

Input:
Customers table:
```
+-------------+-----------+
| customer_id | name      |
+-------------+-----------+
| 1           | Winston   |
| 2           | Jonathan  |
| 3           | Annabelle |
| 4           | Marwan    |
| 5           | Khaled    |
+-------------+-----------+
```
Orders table:
```
+----------+------------+-------------+------+
| order_id | order_date | customer_id | cost |
+----------+------------+-------------+------+
| 1        | 2020-07-31 | 1           | 30   |
| 2        | 2020-07-30 | 2           | 40   |
| 3        | 2020-07-31 | 3           | 70   |
| 4        | 2020-07-29 | 4           | 100  |
```

5	2020-06-10	1	1010
6	2020-08-01	2	102
7	2020-08-01	3	111
8	2020-08-03	1	99
9	2020-08-07	2	32
10	2020-07-15	1	2

Output:

customer_name	customer_id	order_id	order_date
Annabelle	3	7	2020-08-01
Annabelle	3	3	2020-07-31
Jonathan	2	9	2020-08-07
Jonathan	2	6	2020-08-01
Jonathan	2	2	2020-07-30
Marwan	4	4	2020-07-29
Winston	1	8	2020-08-03
Winston	1	1	2020-07-31
Winston	1	10	2020-07-15

Explanation:
Winston has 4 orders, we discard the order of "2020-06-10" because it is the oldest order.
Annabelle has only 2 orders, we return them.
Jonathan has exactly 3 orders.
Marwan ordered only one time.
We sort the result table by customer_name in ascending order, by customer_id in ascending order, and by order_date in descending order in case of a tie.

Follow up: Could you write a general solution for the most recent n orders?

```
# Write your MySQL query statement below
WITH CTE AS (SELECT *,RANK() OVER (PARTITION BY customer_id ORDER BY order_date DESC)
AS R
FROM Orders
JOIN Customers USING (customer_id))

SELECT name AS customer_name,customer_id,order_id,order_date
FROM CTE
WHERE R<=3
ORDER BY customer_name,customer_id,order_date DESC
```