Medium

Table: RequestAccepted

```
+----------------+---------+
| Column Name    | Type    |
+----------------+---------+
| requester_id   | int     |
| accepter_id    | int     |
| accept_date    | date    |
+----------------+---------+
```

(requester_id, accepter_id) is the primary key (combination of columns with unique values) for this table. This table contains the ID of the user who sent the request, the ID of the user who received the request, and the date when the request was accepted.

Write a solution to find the people who have the most friends and the most friends number.

The test cases are generated so that only one person has the most friends.

The result format is in the following example.

Example 1:

Input:
RequestAccepted table:

```
+--------------+-------------+-------------+
| requester_id | accepter_id | accept_date |
+--------------+-------------+-------------+
| 1            | 2           | 2016/06/03  |
| 1            | 3           | 2016/06/08  |
| 2            | 3           | 2016/06/08  |
| 3            | 4           | 2016/06/09  |
+--------------+-------------+-------------+
```

Output:

```
+----+-----+
| id | num |
+----+-----+
| 3  | 3   |
+----+-----+
```

Explanation:
The person with id 3 is a friend of people 1, 2, and 4, so he has three friends in total, which is the most number than any others.

Follow up: In the real world, multiple people could have the same most number of friends. Could you find all these people in this case?

```
# Write your MySQL query statement below

WITH RA AS (
SELECT requester_id AS id
FROM RequestAccepted
UNION ALL
SELECT accepter_id AS id
FROM RequestAccepted
```

```
)

SELECT id,COUNT(*) AS num
FROM RA
GROUP BY id
ORDER BY num desc
LIMIT 1;



-- WITH RA AS (
-- SELECT requester_id AS id, COUNT(*) AS num
-- FROM RequestAccepted
-- GROUP BY requester_id
-- UNION ALL
-- SELECT accepter_id AS id, COUNT(*) AS num
-- FROM RequestAccepted
-- GROUP BY accepter_id
-- )

-- SELECT id,SUM(num) AS num
-- FROM RA
-- GROUP BY id
-- ORDER BY num desc
-- LIMIT 1;
```