Table: Relations

+-------------+------+  
| Column Name | Type |  
+-------------+------+  
| user\_id | int |  
| follower\_id | int |  
+-------------+------+  
(user\_id, follower\_id) is the primary key (combination of columns with unique values) for this table.  
Each row of this table indicates that the user with ID follower\_id is following the user with ID user\_id.

Write a solution to find all the pairs of users with the maximum number of common followers. In other words, if the maximum number of common followers between any two users is maxCommon, then you have to return all pairs of users that have maxCommon common followers.

The result table should contain the pairs user1\_id and user2\_id where user1\_id < user2\_id.

Return the result table in **any order**.

The result format is in the following example.

**Example 1:**

Input:   
Relations table:  
+---------+-------------+  
| user\_id | follower\_id |  
+---------+-------------+  
| 1 | 3 |  
| 2 | 3 |  
| 7 | 3 |  
| 1 | 4 |  
| 2 | 4 |  
| 7 | 4 |  
| 1 | 5 |  
| 2 | 6 |  
| 7 | 5 |  
+---------+-------------+  
Output:   
+----------+----------+  
| user1\_id | user2\_id |  
+----------+----------+  
| 1 | 7 |  
+----------+----------+  
Explanation:   
Users 1 and 2 have two common followers (3 and 4).  
Users 1 and 7 have three common followers (3, 4, and 5).  
Users 2 and 7 have two common followers (3 and 4).  
Since the maximum number of common followers between any two users is 3, we return all pairs of users with three common followers, which is only the pair (1, 7). We return the pair as (1, 7), not as (7, 1).  
Note that we do not have any information about the users that follow users 3, 4, and 5, so we consider them to have 0 followers.