84_Friend Requests II: Who Has the Most Friends Medium - Solution

Source - https://leetcode.com/problems/friend-requests-ii-who-has-the-most-friends/description/

Running Notes:

- find the people who have the most friends and the most friends number.
- we have 2 fields that have caught my interest while reading the question -

  `| requester_id | accepter_id |`
- Now I need to find the people who have most friends and to count how many friends they have I need to consider 2 cases
  - I can send a request and someone can accept (my id is there in requester_id)
  - Someone can send me friend request and I can accept it (my id will be there in accepter_id)
- So my thought process says that I can find the total count of how many times my id appears in either `requester_id | accepter_id` to find my friends count

```
-- Write your PostgreSQL query statement below
WITH requester_CTE AS (
   SELECT requester_id,
   COUNT(requester_id) as friends_based_on_requests
   FROM RequestAccepted
   GROUP BY requester_id
),

accepter_CTE AS (
```

```
    SELECT accepter_id,
    COUNT(accepter_id) as friends_based_on_accepts
    FROM RequestAccepted
    GROUP BY accepter_id
),

main_CTE AS (SELECT *,
COALESCE(friends_based_on_requests,0) + COALESCE(friends_based_on_acce
FROM requester_CTE as r
FULL OUTER JOIN accepter_CTE as a
ON r.requester_id = a.accepter_id)

SELECT
COALESCE(requester_id,accepter_id) as id,
total_friends as num
FROM main_CTE
WHERE
total_friends = (SELECT MAX(total_friends) FROM main_CTE)
```

My code breakdown:

- I calculated friendships based on requests (`requester_CTE`) first
- This counts how many times each `requester_id` appears in the table.
- Compute friendships based on accepted requests (accepter_CTE)
- This counts how many times each accepter_id appears in the table.
- Merge Both Counts with a FULL OUTER JOIN (main_CTE)
- `COALESCE(friends_based_on_requests, 0) + COALESCE(friends_based_on_accepts, 0)` adds the counts from both sides.
- The COALESCE(requester_id, accepter_id) ensures we always get a valid id.

Alternative approach:

```sql
-- Write your PostgreSQL query statement below
WITH requester_CTE AS (
    SELECT requester_id as id,
    COUNT(requester_id) as num
    FROM RequestAccepted
    GROUP BY requester_id
),

accepter_CTE AS (
    SELECT accepter_id as id,
    COUNT(accepter_id) as num
    FROM RequestAccepted
    GROUP BY accepter_id
),

main_CTE AS (SELECT *
FROM requester_CTE
UNION ALL
SELECT *
FROM accepter_CTE)

SELECT id,
SUM(num) AS num
FROM main_CTE
GROUP BY id
ORDER BY SUM(num) DESC
LIMIT 1
```

alternate query is more optimized than my first one. Instead of using FULL OUTER JOIN, it uses UNION ALL, which is faster and simpler.

Written By,
Harshee Pitroda