1811. Find Interview Candidates

Problem Description

In this problem, we have two SQL tables: Contests and Users.

The Contests table has the following columns:

- contest_id (integer): the primary key for this table, representing the contest ID.
- gold_medal (integer): the user ID of the gold medalist.
- silver_medal (integer): the user ID of the silver medalist.
- bronze_medal (integer): the user ID of the bronze medalist.

The Users table has the following columns:

- user_id (integer): the primary key for this table, representing the user ID.
- mail (varchar): the email address of the user.
- name (varchar): the name of the user.

The problem requires us to write an SQL query to report the name and the mail of all interview candidates. A user is considered an interview candidate if **at least one** of these two conditions is true:

1. The user won any medal in three or more consecutive contests.
2. The user won the gold medal in three or more different contests (not necessarily consecutive).

Example

Let's walk through the example given in the problem description to understand the requirements clearly:

Contests table:

```
1  +------------+-----------+-------------+-------------+
2  | contest_id | gold_medal | silver_medal | bronze_medal |
3  +------------+-----------+-------------+-------------+
4  | 190        | 1         | 5           | 2           |
5  | 191        | 2         | 3           | 5           |
6  | 192        | 5         | 2           | 3           |
7  | 193        | 1         | 3           | 5           |
8  | 194        | 4         | 5           | 2           |
9  | 195        | 4         | 2           | 1           |
10 | 196        | 1         | 5           | 2           |
11 +------------+-----------+-------------+-------------+
```

Users table:

```
1  +---------+-------------------+-------+
2  | user_id | mail              | name  |
3  +---------+-------------------+-------+
4  | 1       | sarah@leetcode.com | Sarah |
5  | 2       | bob@leetcode.com   | Bob   |
6  | 3       | alice@leetcode.com | Alice |
7  | 4       | hercy@leetcode.com | Hercy |
8  | 5       | quarz@leetcode.com | Quarz |
9  +---------+-------------------+-------+
```

Result table:

```
1  +-------+-------------------+
2  | name  | mail              |
3  +-------+-------------------+
4  | Sarah | sarah@leetcode.com |
5  | Bob   | bob@leetcode.com   |
6  | Alice | alice@leetcode.com |
7  | Quarz | quarz@leetcode.com |
8  +-------+-------------------+
```

Explanation:

- Sarah won 3 gold medals (190, 193, and 196), so we include her in the result table.
- Bob won a medal in 3 consecutive contests (190, 191, and 192), so we include him in the result table.
  - Note that he also won a medal in 3 other consecutive contests (194, 195, and 196).
- Alice won a medal in 3 consecutive contests (191, 192, and 193), so we include her in the result table.
- Quarz won a medal in 5 consecutive contests (190, 191, 192, 193, and 194), so we include them in the result table.

Approach

To solve this problem, we can combine the information from both tables and use the aggregation feature of SQL to count the number of consecutive contest wins and the number of gold medals won by each user.

1. First, we will create a temporary table that stores all the medals won by each user, along with their contest IDs.
2. In the next step, we will count the number of gold medals won by each user and join the results with the Users table.
3. We will then add a column with the number of consecutive contest wins for each user by counting their consecutive contests.
4. Finally, we will select only users who meet at least one of the two conditions and return their name and email.

SQL Query

Here's the SQL query that implements the approach described above:

```
1  WITH medal_counts AS (
2      SELECT user_id, contest_id, contest_id - ROW_NUMBER() OVER (PARTITION BY user_id ORDER BY contest_id) AS group_id
3      FROM (
4          SELECT gold_medal AS user_id, contest_id FROM Contests
5          UNION ALL
6          SELECT silver_medal AS user_id, contest_id FROM Contests
7          UNION ALL
8          SELECT bronze_medal AS user_id, contest_id FROM Contests
9      ) medals
10 )
11 , gold_medals AS (
12     SELECT user_id, COUNT(*) AS gold_medal_count
13     FROM Contests
14     GROUP BY user_id
15 )
16 SELECT u.name, u.mail
17 FROM Users u
18 JOIN gold_medals gm ON u.user_id = gm.user_id
19 WHERE gm.gold_medal_count >= 3
20 OR EXISTS (
21     SELECT 1
22     FROM medal_counts
23     WHERE user_id = u.user_id
24     GROUP BY user_id, group_id
25     HAVING COUNT(*) >= 3
26 )
```

The query first creates a temporary table with all medals won by each user and another temporary table with the count of gold medals won by each user. Then it selects users who have won 3 or more gold medals or have won any medals in 3 or more consecutive contests.## Alternative Approach

Another approach is to first find all users who won a medal in three or more consecutive contests, and then find all users who won the gold medal in three or more different contests. Finally, we can union these results and join them with the Users table to obtain their name and email.

SQL Query

Here's the SQL query that implements the alternative approach:

```
1  WITH medals AS (
2      SELECT DISTINCT user_id, contest_id
3      FROM (
4          SELECT gold_medal AS user_id, contest_id FROM Contests
5          UNION ALL
6          SELECT silver_medal AS user_id, contest_id FROM Contests
7          UNION ALL
8          SELECT bronze_medal AS user_id, contest_id FROM Contests
9      ) medals
10 )
11 , consecutive_medals AS (
12     SELECT user_id
13     FROM (
14         SELECT user_id, contest_id, contest_id - ROW_NUMBER() OVER (PARTITION BY user_id ORDER BY contest_id) AS group_id
15         FROM medals
16     ) medal_group
17     GROUP BY user_id, group_id
18     HAVING COUNT(*) >= 3
19 )
20 , gold_medalists AS (
21     SELECT user_id
22     FROM Contests
23     GROUP BY user_id
24     HAVING COUNT(gold_medal) >= 3
25 )
26
27 SELECT u.name, u.mail
28 FROM Users u
29 WHERE EXISTS (SELECT 1 FROM consecutive_medals WHERE user_id = u.user_id)
30 OR EXISTS (SELECT 1 FROM gold_medalists WHERE user_id = u.user_id)
```

This query first creates a temporary table with all unique medal winning user_ids and contest_ids. It then finds users who won a medal in three or more consecutive contests and users who won the gold medal in three or more different contests. Finally, it selects users that meet either of the conditions and returns their name and email.

Got a question? Ask the Teaching Assistant anything you don't understand.