Problem 1341: Movie Rating

Problem Information

Difficulty: Medium
Acceptance Rate: 0.00%
Paid Only: No

Problem Description

Table:

Movies

+---------------+---------+ | Column Name | Type | +---------------+---------+ | movie_id | int | | title | varchar | +---------------+---------+ movie_id is the primary key (column with unique values) for this table. title is the name of the movie. Each movie has a unique title.

Table:

Users

+---------------+---------+ | Column Name | Type | +---------------+---------+ | user_id | int | | name | varchar | +---------------+---------+ user_id is the primary key (column with unique values) for this table. The column 'name' has unique values.

Table:

MovieRating

+---------------+---------+ | Column Name | Type | +---------------+---------+ | movie_id | int | | user_id | int | | rating | int | | created_at | date | +---------------+---------+ (movie_id, user_id) is the primary key (column with unique values) for this table. This table contains the rating of a movie by a user in their review. created_at is the user's review date.

Write a solution to:

Find the name of the user who has rated the greatest number of movies. In case of a tie, return the lexicographically smaller user name.

Find the movie name with the

highest average

rating in

February 2020

. In case of a tie, return the lexicographically smaller movie name.

The result format is in the following example.

Example 1:

Input:

Movies table:
+----------+----------+
| movie_id | title |
+----------+----------+
1	Avengers
2	Frozen 2
3	Joker
+----------+----------+

Users table:
+---------+--------+
| user_id | name |
+---------+--------+
1	Daniel
2	Monica
3	Maria
4	James
+---------+--------+

MovieRating table:
+----------+---------+--------+------------+
| movie_id | user_id | rating | created_at |
+----------+---------+--------+------------+
1	1	3	2020-01-12
1	2	4	2020-02-11
1	3	2	2020-02-12
1	4	1	2020-01-01
2	1	5	2020-02-17
2	2	2	2020-02-01
2	3	2	2020-03-01
3	1	3	2020-02-22
3	2	4	2020-02-25
+----------+---------+--------+------------+

Output:

+----------+
| results |
+----------+
| Daniel |
| Frozen 2 |
+----------+

Explanation:

Daniel and Monica have rated 3 movies ("Avengers", "Frozen 2" and "Joker") but Daniel is smaller lexicographically. Frozen 2 and Joker have a rating average of 3.5 in February but Frozen 2 is smaller lexicographically.

Code Snippets

MySQL:

```
# Write your MySQL query statement below
```

MS SQL Server:

```
/* Write your T-SQL query statement below */
```

PostgreSQL:

```
-- Write your PostgreSQL query statement below
```

Oracle:

```
/* Write your PL/SQL query statement below */
```

Pandas:

```
import pandas as pd

def movie_rating(movies: pd.DataFrame, users: pd.DataFrame, movie_rating:
pd.DataFrame) -> pd.DataFrame:
```

Solutions

MySQL Solution:

```
# Write your MySQL query statement below
```

MS SQL Server Solution:

```
/* Write your T-SQL query statement below */
```

PostgreSQL Solution:

```
-- Write your PostgreSQL query statement below
```

Oracle Solution:

```
/* Write your PL/SQL query statement below */
```

Pandas Solution:

```python
import pandas as pd

def movie_rating(movies: pd.DataFrame, users: pd.DataFrame, movie_rating:
pd.DataFrame) -> pd.DataFrame:
```