Problem 1264: Page Recommendations

Problem Information

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

Problem Description

Table:

Friendship

+---------------+---------+ | Column Name | Type | +---------------+---------+ | user1_id | int | | user2_id | int | +---------------+---------+ (user1_id, user2_id) is the primary key (combination of columns with unique values) for this table. Each row of this table indicates that there is a friendship relation between user1_id and user2_id.

Table:

Likes

+-------------+---------+ | Column Name | Type | +-------------+---------+ | user_id | int | | page_id | int | +-------------+---------+ (user_id, page_id) is the primary key (combination of columns with unique values) for this table. Each row of this table indicates that user_id likes page_id.

Write a solution to recommend pages to the user with

user_id = 1

using the pages that your friends liked. It should not recommend pages you already liked.

Return result table in

any order

without duplicates.

The result format is in the following example.

Example 1:

Input:

Friendship table: +----------+----------+ | user1_id | user2_id | +----------+----------+ | 1 | 2 | | 1 | 3 | | 1 | 4 | | 2 | 3 | | 2 | 4 | | 2 | 5 | | 6 | 1 | +----------+----------+ Likes table: +---------+---------+ | user_id | page_id | +---------+---------+ | 1 | 88 | | 2 | 23 | | 3 | 24 | | 4 | 56 | | 5 | 11 | | 6 | 33 | | 2 | 77 | | 3 | 77 | | 6 | 88 | +---------+---------+

Output:

+------------------+ | recommended_page | +------------------+ | 23 | | 24 | | 56 | | 33 | | 77 | +------------------+

Explanation:

User one is friend with users 2, 3, 4 and 6. Suggested pages are 23 from user 2, 24 from user 3, 56 from user 3 and 33 from user 6. Page 77 is suggested from both user 2 and user 3. Page 88 is not suggested because user 1 already likes it.

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 page_recommendations(friendship: pd.DataFrame, likes: 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:

```
import pandas as pd

def page_recommendations(friendship: pd.DataFrame, likes: pd.DataFrame) ->
pd.DataFrame:
```