Problem 262: Trips and Users

Problem Information

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

Problem Description

Table:

Trips

+-------------+----------+ | Column Name | Type | +-------------+----------+ | id | int | | client_id | int | | driver_id | int | | city_id | int | | status | enum | | request_at | varchar | +-------------+----------+ id is the primary key (column with unique values) for this table. The table holds all taxi trips. Each trip has a unique id, while client_id and driver_id are foreign keys to the users_id at the Users table. Status is an ENUM (category) type of ('completed', 'cancelled_by_driver', 'cancelled_by_client').

Table:

Users

+-------------+----------+ | Column Name | Type | +-------------+----------+ | users_id | int | | banned | enum | | role | enum | +-------------+----------+ users_id is the primary key (column with unique values) for this table. The table holds all users. Each user has a unique users_id, and role is an ENUM type of ('client', 'driver', 'partner'). banned is an ENUM (category) type of ('Yes', 'No').

The

cancellation rate

is computed by dividing the number of canceled (by client or driver) requests with unbanned users by the total number of requests with unbanned users on that day.

Write a solution to find the

cancellation rate

of requests with unbanned users (

both client and driver must not be banned

) each day between

"2013-10-01"

and

"2013-10-03"

with

at least

one trip. Round

Cancellation Rate

to

two decimal

points.

Return the result table in

any order

.

The result format is in the following example.

Example 1:

Input:

Trips table: +----+-----------+-----------+---------+---------------------+------------+ | id | client_id | driver_id | city_id | status | request_at |
+----+-----------+-----------+---------+---------------------+------------+ | 1 | 1 | 10 | 1 | completed | 2013-10-01 | | 2 | 2 | 11 | 1 | cancelled_by_driver | 2013-10-01 | | 3 | 3 | 12 | 6 | completed | 2013-10-01 | | 4 | 4 | 13 | 6 | cancelled_by_client | 2013-10-01 | | 5 | 1 | 10 | 1 | completed | 2013-10-02 | | 6 | 2 | 11 | 6 | completed | 2013-10-02 | | 7 | 3 | 12 | 6 | completed | 2013-10-02 | | 8 | 2 | 12 | 12 | completed | 2013-10-03 | | 9 | 3 | 10 | 12 | completed | 2013-10-03 | | 10 | 4 | 13 | 12 | cancelled_by_driver | 2013-10-03 |
+----+-----------+-----------+---------+---------------------+------------+ Users table:
+----------+--------+--------+ | users_id | banned | role | +----------+--------+--------+ | 1 | No | client | | 2 | Yes | client | | 3 | No | client | | 4 | No | client | | 10 | No | driver | | 11 | No | driver | | 12 | No | driver | | 13 | No | driver | +----------+--------+--------+

Output:

+------------+-------------------+ | Day | Cancellation Rate | +------------+-------------------+ | 2013-10-01 | 0.33 | | 2013-10-02 | 0.00 | | 2013-10-03 | 0.50 | +------------+-------------------+

Explanation:

On 2013-10-01: - There were 4 requests in total, 2 of which were canceled. - However, the request with Id=2 was made by a banned client (User_Id=2), so it is ignored in the calculation. - Hence there are 3 unbanned requests in total, 1 of which was canceled. - The Cancellation Rate is (1 / 3) = 0.33 On 2013-10-02: - There were 3 requests in total, 0 of which were canceled. - The request with Id=6 was made by a banned client, so it is ignored. - Hence there are 2 unbanned requests in total, 0 of which were canceled. - The Cancellation Rate is (0 / 2) = 0.00 On 2013-10-03: - There were 3 requests in total, 1 of which was canceled. - The request with Id=8 was made by a banned client, so it is ignored. - Hence there are 2 unbanned request in total, 1 of which were canceled. - The Cancellation Rate is (1 / 2) = 0.50

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 trips_and_users(trips: pd.DataFrame, users: 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
```

```python
def trips_and_users(trips: pd.DataFrame, users: pd.DataFrame) ->
pd.DataFrame:
```