# Problem 3497: Analyze Subscription Conversion

## Problem Information

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

## Problem Description

Table:

UserActivity

+------------------+---------+ | Column Name | Type | +------------------+---------+ | user_id | int | | activity_date | date | | activity_type | varchar | | activity_duration| int | +------------------+---------+ (user_id, activity_date, activity_type) is the unique key for this table. activity_type is one of ('free_trial', 'paid', 'cancelled'). activity_duration is the number of minutes the user spent on the platform that day. Each row represents a user's activity on a specific date.

A subscription service wants to analyze user behavior patterns. The company offers a

7

-day

free trial

, after which users can subscribe to a

paid plan

or

cancel

. Write a solution to:

Find users who converted from free trial to paid subscription

Calculate each user's

average daily activity duration

during their

free trial

period (rounded to

2

decimal places)

Calculate each user's

average daily activity duration

during their

paid

subscription period (rounded to

2

decimal places)

Return

the result table ordered by

user_id

in

ascending

order

.

The result format is in the following example.

Example:

Input:

UserActivity table:

| user_id | activity_date | activity_type | activity_duration |
|---|---|---|---|
| 1 | 2023-01-01 | free_trial | 45 |
| 1 | 2023-01-02 | free_trial | 30 |
| 1 | 2023-01-05 | free_trial | 60 |
| 1 | 2023-01-10 | paid | 75 |
| 1 | 2023-01-12 | paid | 90 |
| 1 | 2023-01-15 | paid | 65 |
| 2 | 2023-02-01 | free_trial | 55 |
| 2 | 2023-02-03 | free_trial | 25 |
| 2 | 2023-02-07 | free_trial | 50 |
| 2 | 2023-02-10 | cancelled | 0 |
| 3 | 2023-03-05 | free_trial | 70 |
| 3 | 2023-03-06 | free_trial | 60 |
| 3 | 2023-03-08 | free_trial | 80 |
| 3 | 2023-03-12 | paid | 50 |
| 3 | 2023-03-15 | paid | 55 |
| 3 | 2023-03-20 | paid | 85 |
| 4 | 2023-04-01 | free_trial | 40 |
| 4 | 2023-04-03 | free_trial | 35 |
| 4 | 2023-04-05 | paid | 45 |
| 4 | 2023-04-07 | cancelled | 0 |

Output:

| user_id | trial_avg_duration | paid_avg_duration |
|---|---|---|
| 1 | 45.00 | 76.67 |
| 3 | 70.00 | 63.33 |
| 4 | 37.50 | 45.00 |

Explanation:

User 1:

Had 3 days of free trial with durations of 45, 30, and 60 minutes.

Average trial duration: (45 + 30 + 60) / 3 = 45.00 minutes.

Had 3 days of paid subscription with durations of 75, 90, and 65 minutes.

Average paid duration: (75 + 90 + 65) / 3 = 76.67 minutes.

User 2:

Had 3 days of free trial with durations of 55, 25, and 50 minutes.

Average trial duration: (55 + 25 + 50) / 3 = 43.33 minutes.

Did not convert to a paid subscription (only had free_trial and cancelled activities).

Not included in the output because they didn't convert to paid.

User 3:

Had 3 days of free trial with durations of 70, 60, and 80 minutes.

Average trial duration: (70 + 60 + 80) / 3 = 70.00 minutes.

Had 3 days of paid subscription with durations of 50, 55, and 85 minutes.

Average paid duration: (50 + 55 + 85) / 3 = 63.33 minutes.

User 4:

Had 2 days of free trial with durations of 40 and 35 minutes.

Average trial duration: (40 + 35) / 2 = 37.50 minutes.

Had 1 day of paid subscription with duration of 45 minutes before cancelling.

Average paid duration: 45.00 minutes.

The result table only includes users who converted from free trial to paid subscription (users 1, 3, and 4), and is ordered by user_id in ascending order.

## 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 analyze_subscription_conversion(user_activity: 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 analyze_subscription_conversion(user_activity: pd.DataFrame) ->
pd.DataFrame:
```