

Problem 2230: The Users That Are Eligible for Discount

Problem Information

Difficulty: Easy

Acceptance Rate: 0.00%

Paid Only: No

Problem Description

Table:

Purchases

+-----+-----+ | Column Name | Type | +-----+-----+ | user_id | int | |
time_stamp | datetime | | amount | int | +-----+-----+ (user_id, time_stamp) is the
primary key (combination of columns with unique values) for this table. Each row contains
information about the purchase time and the amount paid for the user with ID user_id.

A user is eligible for a discount if they had a purchase in the inclusive interval of time

[startDate, endDate]

with at least

minAmount

amount. To convert the dates to times, both dates should be considered as the

start

of the day (i.e.,

endDate = 2022-03-05

should be considered as the time

2022-03-05 00:00:00

).

Write a solution to report the IDs of the users that are eligible for a discount.

Return the result table ordered by

user_id

.

The result format is in the following example.

Example 1:

Input:

Purchases table: +-----+-----+-----+ | user_id | time_stamp | amount |
+-----+-----+-----+ | 1 | 2022-04-20 09:03:00 | 4416 | | 2 | 2022-03-19 19:24:02
| 678 | | 3 | 2022-03-18 12:03:09 | 4523 | | 3 | 2022-03-30 09:43:42 | 626 |
+-----+-----+-----+ startDate = 2022-03-08, endDate = 2022-03-20,
minAmount = 1000

Output:

+-----+ | user_id | +-----+ | 3 | +-----+

Explanation:

Out of the three users, only User 3 is eligible for a discount. - User 1 had one purchase with at least minAmount amount, but not within the time interval. - User 2 had one purchase within the time interval, but with less than minAmount amount. - User 3 is the only user who had a purchase that satisfies both conditions.

Important Note:

This problem is basically the same as

The Number of Users That Are Eligible for Discount

.

Code Snippets

MySQL:

```
CREATE PROCEDURE getUserIDs(startDate DATE, endDate DATE, minAmount INT)
BEGIN
  # Write your MySQL query statement below.

END
```

MS SQL Server:

```
CREATE PROCEDURE getUserIDs(@startDate DATE, @endDate DATE, @minAmount INT)
AS
BEGIN
  /* Write your T-SQL query statement below. */

END
```

PostgreSQL:

```
CREATE OR REPLACE FUNCTION getUserIDs(startDate DATE, endDate DATE, minAmount
INT)
RETURNS TABLE (user_id INT) AS $$
BEGIN
  RETURN QUERY (
    -- Write your PostgreSQL query statement below.

  );
END;
$$ LANGUAGE plpgsql;
```

Oracle:

```

CREATE FUNCTION getUserIDs(startDate IN DATE, endDate IN DATE, minAmount IN
NUMBER)
RETURN SYS_REFCURSOR IS result SYS_REFCURSOR;
BEGIN
/* Write your PL/SQL query statement below */

RETURN result;
END;

```

Pandas:

```

import pandas as pd
from datetime import datetime

def find_valid_users(purchases: pd.DataFrame, start_date: datetime, end_date:
datetime, min_amount: int) -> pd.DataFrame:

```

Solutions

MySQL Solution:

```

CREATE PROCEDURE getUserIDs(startDate DATE, endDate DATE, minAmount INT)
BEGIN
# Write your MySQL query statement below.

END

```

MS SQL Server Solution:

```

CREATE PROCEDURE getUserIDs(@startDate DATE, @endDate DATE, @minAmount INT)
AS
BEGIN
/* Write your T-SQL query statement below. */

END

```

PostgreSQL Solution:

```

CREATE OR REPLACE FUNCTION getUserIDs(startDate DATE, endDate DATE, minAmount
INT)
RETURNS TABLE (user_id INT) AS $$

```

```
BEGIN
RETURN QUERY (
-- Write your PostgreSQL query statement below.

);
END;
$$ LANGUAGE plpgsql;
```

Oracle Solution:

```
CREATE FUNCTION getUserIDs(startDate IN DATE, endDate IN DATE, minAmount IN
NUMBER)
RETURN SYS_REFCURSOR IS result SYS_REFCURSOR;
BEGIN
/* Write your PL/SQL query statement below */

RETURN result;
END;
```

Pandas Solution:

```
import pandas as pd
from datetime import datetime

def find_valid_users(purchases: pd.DataFrame, start_date: datetime, end_date:
datetime, min_amount: int) -> pd.DataFrame:
```