Task

You are given a history of your bank account transactions for the year 2020. Each transaction was either a credit card payment or an incoming transfer.

There is a fee for holding a credit card which you have to pay every month. The cost you are charged each month is 5. However, you are not charged for a given month if you made at least three credit card payments for a total cost of at least 100 within that month. Note that this fee is **not** included in the supplied history of transactions.

At the beginning of the year, the balance of your account was 0. Your task is to compute the balance at the end of the year.

You are given a table `transactions` with the following structure:

```
create table transactions (
      amount integer not null,
      date date not null
);
```

Each row of the table contains information about a single transaction: the amount of money (`amount`) and the date when the transaction happened (`date`). If the `amount` value is negative, it is a credit card payment. Otherwise, it is an incoming transfer. There are no transactions with an amount of 0.

Write an SQL query that returns a table containing one column, `balance`. The table should contain one row with the total balance of your account at the end of the year, including the fee for holding a credit card.

Examples:

1. Given table:

```
+--------+------------+
| amount |       date |
+--------+------------+
|   1000 | 2020-01-06 |
|    -10 | 2020-01-14 |
|    -75 | 2020-01-20 |
|     -5 | 2020-01-25 |
|     -4 | 2020-01-29 |
|   2000 | 2020-03-10 |
|    -75 | 2020-03-12 |
|    -20 | 2020-03-15 |
|     40 | 2020-03-15 |
|    -50 | 2020-03-17 |
|    200 | 2020-10-10 |
|   -200 | 2020-10-10 |
+--------+------------+
```

your query should return:

```
+---------+
| balance |
+---------+
|    2746 |
+---------+
```

The balance without the credit card fee would be 2801. You are charged a fee for every month except March, which in total equates to 11 * 5 = 55.

In March, you had three transactions for a total cost of 75 + 20 + 50 = 145, thus you are not charged the fee. In January, you had four card payments for a total cost of 10 + 75 + 5 + 4 = 94, which is less than 100; thus you are charged. In October, you had one card payment for a total cost of 200 but you need to have at least three payments in a month; thus you are charged. In all other months (February, April, ...) you had no card payments, thus you are charged.

The final balance is 2801 - 55 = 2746.

2. Given table:

```
+--------+------------+
| amount |       date |
+--------+------------+
|      1 | 2020-06-29 |
|     35 | 2020-02-20 |
|    -50 | 2020-02-03 |
|     -1 | 2020-02-26 |
|   -200 | 2020-08-01 |
|    -44 | 2020-02-07 |
|     -5 | 2020-02-25 |
|      1 | 2020-06-29 |
|      1 | 2020-06-29 |
|   -100 | 2020-12-29 |
|   -100 | 2020-12-30 |
|   -100 | 2020-12-31 |
+--------+------------+
```

your query should return:

```
+---------+
| balance |
+---------+
|    -612 |
+---------+
```

The balance excluding the fee would be -562. You are not charged the fee in February since you had four card payments for a total cost of 50 + 1 + 44 + 5 = 100 in that month. You are also not charged the fee in December since you had three card payments for a total cost of 100 + 100 + 100 = 300. The final balance is -562 - 10 * 5 = -612.

3. Given table:

```
+--------+------------+
| amount |       date |
+--------+------------+
|   6000 | 2020-04-03 |
|   5000 | 2020-04-02 |
|   4000 | 2020-04-01 |
|   3000 | 2020-03-01 |
|   2000 | 2020-02-01 |
|   1000 | 2020-01-01 |
+--------+------------+
```

Your query should return:

```
+---------+
```

```
+---------+
| balance |
+---------+
|   20940 |
+---------+
```

You earned 21000 but you are charged a fee for every month.
The final balance is 21000 - 12 * 5 = 20940.

Assume that:

- column `date` contains only dates between 2020-01-01 and 2020-12-31;
- column `amount` contains only non-zero values.

SOLUTION

```sql
-- Implement your solution here
SELECT sum(amount_per_month) -
       sum(5) FILTER (WHERE negative_per_month > -100 OR negative_count < 3)
FROM (SELECT sum(amount) AS amount_per_month,
             sum(amount) FILTER (WHERE amount < 0) AS negative_per_month,
             month_start,
             count(*) FILTER (WHERE amount < 0) AS negative_count
      FROM (SELECT coalesce(t.amount, 0) AS amount,
                   coalesce(date_trunc('month', CAST (t.date AS timestamp)),
dates.d) AS month_start
            FROM generate_series(
                    TIMESTAMP '2020-01-01',
                    TIMESTAMP '2020-12-01',
                    INTERVAL '1 month'
                  ) AS dates (d)
               LEFT JOIN transactions AS t
                  ON dates.d = date_trunc('month', CAST (t.date AS timestamp))
            ) AS gaps_filled
      GROUP BY month_start
     ) AS sums_per_month;
```

```
Your test case:
insert into transactions values ('1000', '2020-01-06');
insert into transactions values ('-10', '2020-01-14');
insert into transactions values ('-75', '2020-01-20');
insert into transactions values ('-5', '2020-01-25');
insert into transactions values ('-4', '2020-01-29');
insert into transactions values ('2000', '2020-03-10');
insert into transactions values ('-75', '2020-03-12');
insert into transactions values ('-20', '2020-03-15');
insert into transactions values ('40', '2020-03-15');
insert into transactions values ('-50', '2020-03-17');
insert into transactions values ('200', '2020-10-10');
insert into transactions values ('-200', '2020-10-10');

Returned value:
+------+
| 2746 |
+------+
```

Your code is syntactically correct and works properly on the example test.
Note that the example tests are not part of your score. On submission at least 8 test cases not shown here will assess your solution.

Task 3

SQL (PostgreSQL)

Task score ?
88%

Correctness ?
88%

Performance ?
—

Example test cases ?
Passed 3 out of 3

Correctness test cases
Passed 8 out of 9

Submission date
2023-02-16 23:29 EET