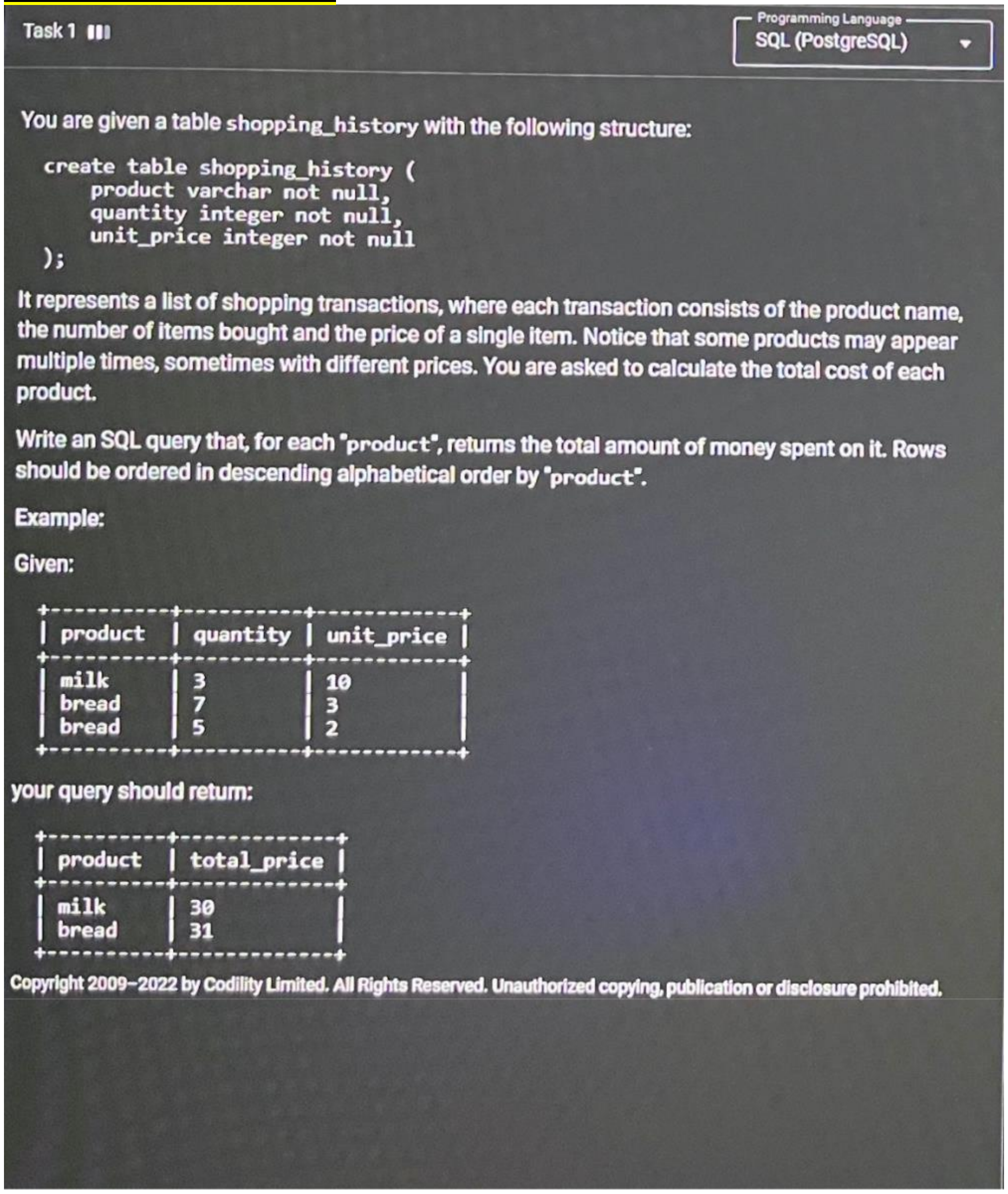
SQL PROJECT

Create the following table structure in SNOWFLAKE by creating your own warehouse. Insert some 10 rows using INSERT command (check task 3 and same way insert for all task tables) in the table by trying different values for all the columns and then check using SELECT *

Once data is loaded, performed the below task

PROBLEM STATEMENT 1:



Task 1

Programming Language: SQL (PostgreSQL)

You are given a table shopping_history with the following structure:

```
create table shopping_history (
    product varchar not null,
    quantity integer not null,
    unit_price integer not null
);
```

It represents a list of shopping transactions, where each transaction consists of the product name, the number of items bought and the price of a single item. Notice that some products may appear multiple times, sometimes with different prices. You are asked to calculate the total cost of each product.

Write an SQL query that, for each "product", returns the total amount of money spent on it. Rows should be ordered in descending alphabetical order by "product".

Example:

Given:

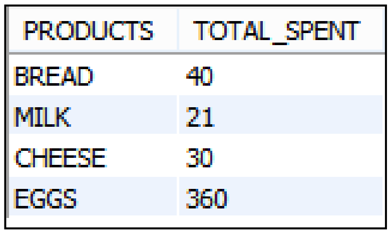
```
+----------+----------+------------+
| product  | quantity | unit_price |
+----------+----------+------------+
| milk     | 3        | 10         |
| bread    | 7        | 3          |
| bread    | 5        | 2          |
+----------+----------+------------+
```

your query should return:

```
+----------+-------------+
| product  | total_price |
+----------+-------------+
| milk     | 30          |
| bread    | 31          |
+----------+-------------+
```

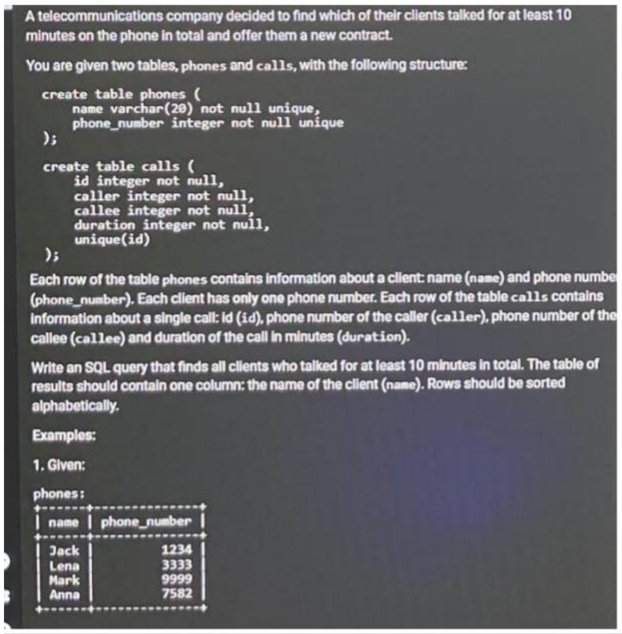
```sql
---------------------------TASK 1 : CALCULATE TOTAL COST OF EACH PRODUCT IN SHOPPING LIST------------
CREATE DATABASE INEURON_TASK
USE INEURON_TASK
CREATE TABLE SHOPPING_HISTORY(
PRODUCTS VARCHAR(10) NOT NULL,
QUANTITY INT NOT NULL,
UNIT_PRICE INT NOT NULL
);
INSERT INTO SHOPPING_HISTORY VALUES
('BREAD', 3 , 10),
('MILK',7,3),
('BREAD',5,2),
('CHEESE',2,15),
('EGGS',12,30)

SELECT PRODUCTS,SUM(QUANTITY*UNIT_PRICE) AS TOTAL_SPENT FROM SHOPPING_HISTORY
GROUP BY PRODUCTS
```

OUTPUT



PRODUCTS	TOTAL_SPENT
BREAD	40
MILK	21
CHEESE	30
EGGS	360

PROBLEM STATEMENT 2



A telecommunications company decided to find which of their clients talked for at least 10 minutes on the phone in total and offer them a new contract.

You are given two tables, phones and calls, with the following structure:

```sql
create table phones (
    name varchar(20) not null unique,
    phone_number integer not null unique
);

create table calls (
    id integer not null,
    caller integer not null,
    callee integer not null,
    duration integer not null,
    unique(id)
);
```

Each row of the table phones contains information about a client: name (name) and phone numbe (phone_number). Each client has only one phone number. Each row of the table calls contains information about a single call: id (id), phone number of the caller (caller), phone number of the callee (callee) and duration of the call in minutes (duration).

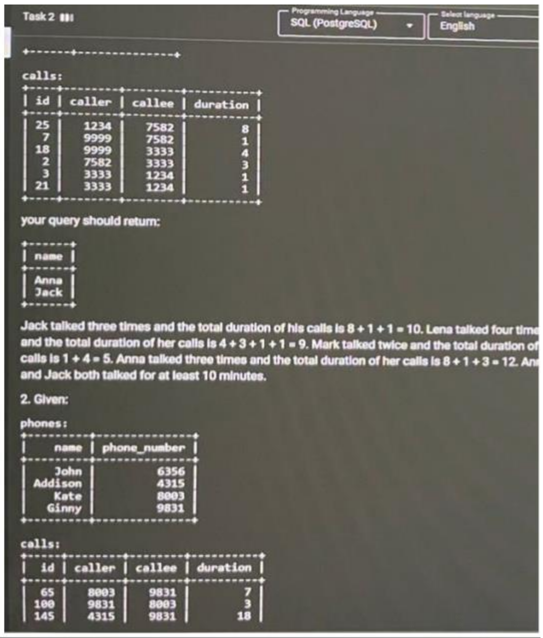
Write an SQL query that finds all clients who talked for at least 10 minutes in total. The table of results should contain one column: the name of the client (name). Rows should be sorted alphabetically.

Examples:

1. Given:

phones:

```
+------+--------------+
| name | phone_number |
+------+--------------+
| Jack |         1234 |
| Lena |         3333 |
| Mark |         9999 |
| Anna |         7582 |
+------+--------------+
```



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

calls:

```
+----+--------+--------+----------+
| id | caller | callee | duration |
+----+--------+--------+----------+
| 25 |   1234 |   7582 |        8 |
|  7 |   9999 |   7582 |        1 |
| 18 |   9999 |   3333 |        4 |
|  2 |   7582 |   3333 |        3 |
|  3 |   3333 |   1234 |        1 |
| 21 |   3333 |   1234 |        1 |
+----+--------+--------+----------+
```

your query should return:

```
+------+
| name |
+------+
| Anna |
| Jack |
+------+
```

Jack talked three times and the total duration of his calls is 8 + 1 + 1 = 10. Lena talked four time and the total duration of her calls is 4 + 3 + 1 + 1 = 9. Mark talked twice and the total duration of calls is 1 + 4 = 5. Anna talked three times and the total duration of her calls is 8 + 1 + 3 = 12. An and Jack both talked for at least 10 minutes.

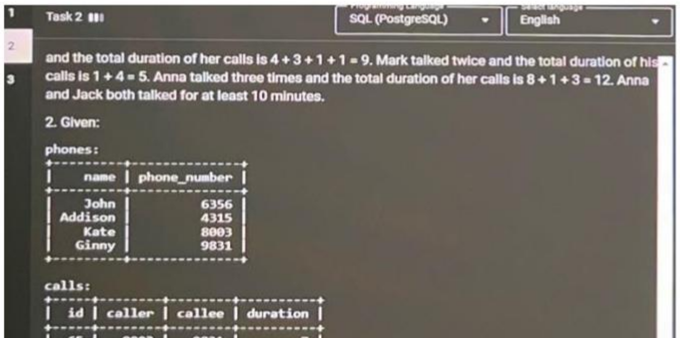
2. Given:

phones:

```
+---------+--------------+
|    name | phone_number |
+---------+--------------+
|    John |         6356 |
| Addison |         4315 |
|    Kate |         8003 |
|   Ginny |         9831 |
+---------+--------------+
```

calls:

```
+-----+--------+--------+----------+
|  id | caller | callee | duration |
+-----+--------+--------+----------+
|  65 |   8003 |   9831 |        7 |
| 100 |   9831 |   8003 |        3 |
| 145 |   4315 |   9831 |       18 |
```



and the total duration of her calls is 4 + 3 + 1 + 1 = 9. Mark talked twice and the total duration of his calls is 1 + 4 = 5. Anna talked three times and the total duration of her calls is 8 + 1 + 3 = 12. Anna and Jack both talked for at least 10 minutes.

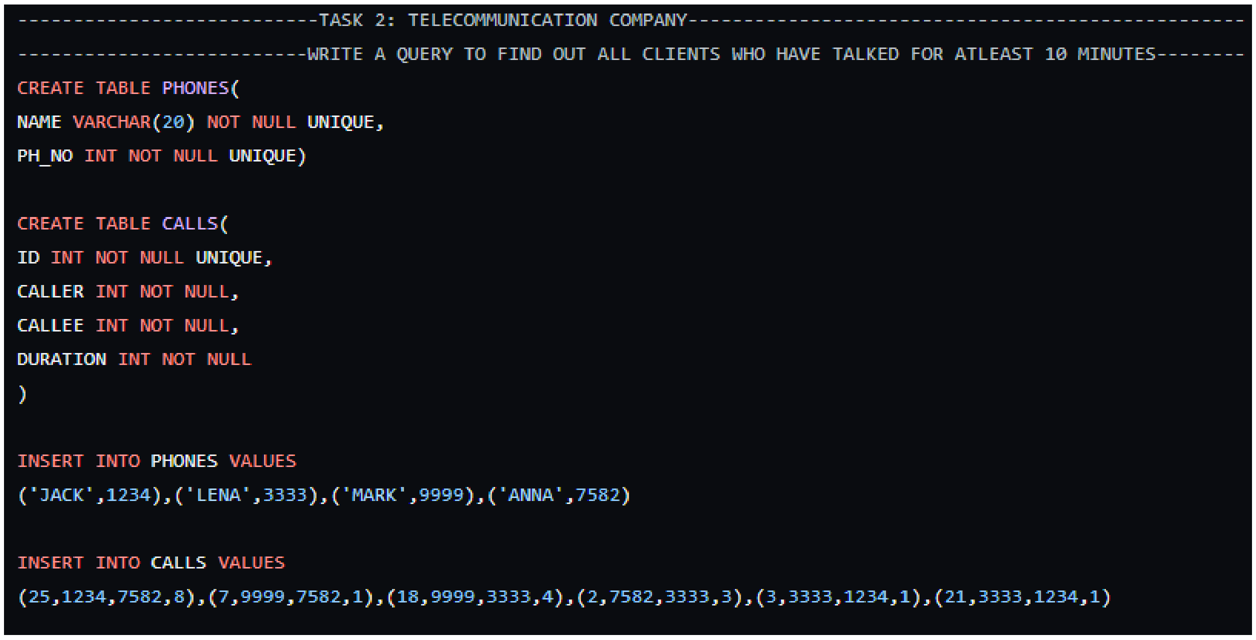
2. Given:

phones:

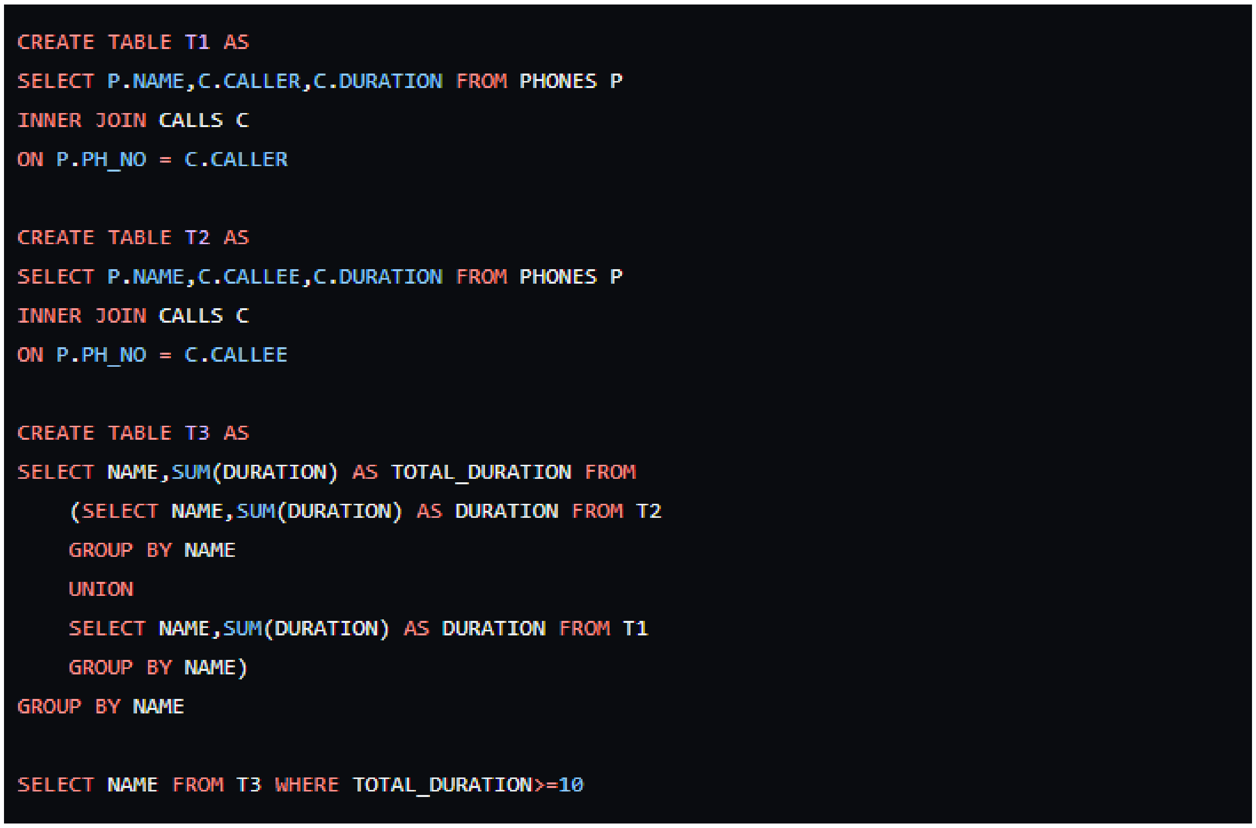
```
+---------+--------------+
|    name | phone_number |
+---------+--------------+
|    John |         6356 |
| Addison |         4315 |
|    Kate |         8003 |
|   Ginny |         9831 |
+---------+--------------+
```

calls:

```
+-----+--------+--------+----------+
|  id | caller | callee | duration |
+-----+--------+--------+----------+
```

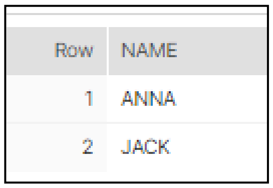


```
--------------------------TASK 2: TELECOMMUNICATION COMPANY-----------------------------------------------
-------------------------WRITE A QUERY TO FIND OUT ALL CLIENTS WHO HAVE TALKED FOR ATLEAST 10 MINUTES--------
CREATE TABLE PHONES(
NAME VARCHAR(20) NOT NULL UNIQUE,
PH_NO INT NOT NULL UNIQUE)

CREATE TABLE CALLS(
ID INT NOT NULL UNIQUE,
CALLER INT NOT NULL,
CALLEE INT NOT NULL,
DURATION INT NOT NULL
)

INSERT INTO PHONES VALUES
('JACK',1234),('LENA',3333),('MARK',9999),('ANNA',7582)

INSERT INTO CALLS VALUES
(25,1234,7582,8),(7,9999,7582,1),(18,9999,3333,4),(2,7582,3333,3),(3,3333,1234,1),(21,3333,1234,1)
```



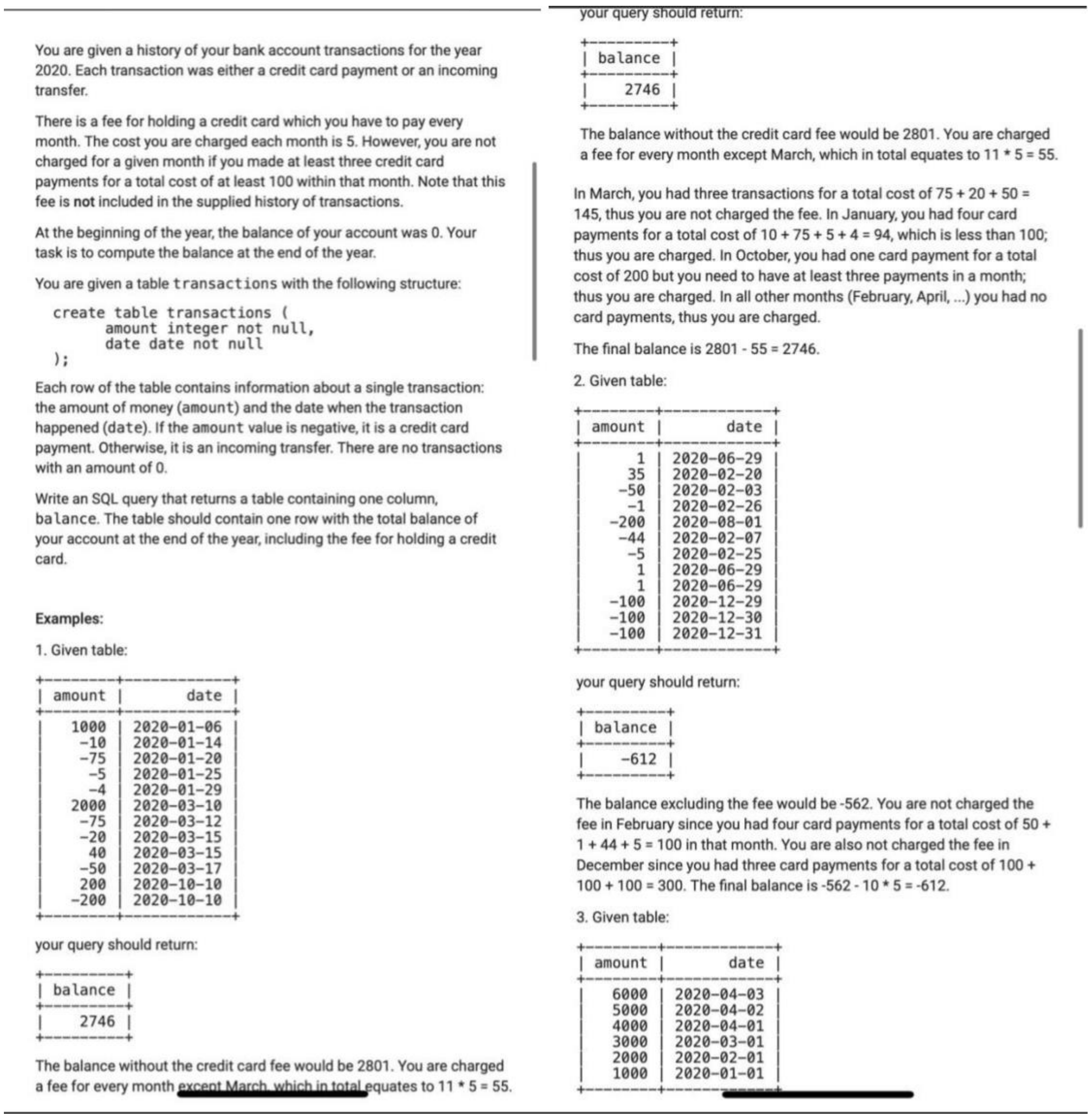
```
CREATE TABLE T1 AS
SELECT P.NAME,C.CALLER,C.DURATION FROM PHONES P
INNER JOIN CALLS C
ON P.PH_NO = C.CALLER

CREATE TABLE T2 AS
SELECT P.NAME,C.CALLEE,C.DURATION FROM PHONES P
INNER JOIN CALLS C
ON P.PH_NO = C.CALLEE

CREATE TABLE T3 AS
SELECT NAME,SUM(DURATION) AS TOTAL_DURATION FROM
    (SELECT NAME,SUM(DURATION) AS DURATION FROM T2
    GROUP BY NAME
    UNION
    SELECT NAME,SUM(DURATION) AS DURATION FROM T1
    GROUP BY NAME)
GROUP BY NAME

SELECT NAME FROM T3 WHERE TOTAL_DURATION>=10
```

OUTPUT



Row	NAME
1	ANNA
2	JACK

PROBLEM STATEMENT 3: Output display is just one column balance



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.

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 |
+--------+------------+
```

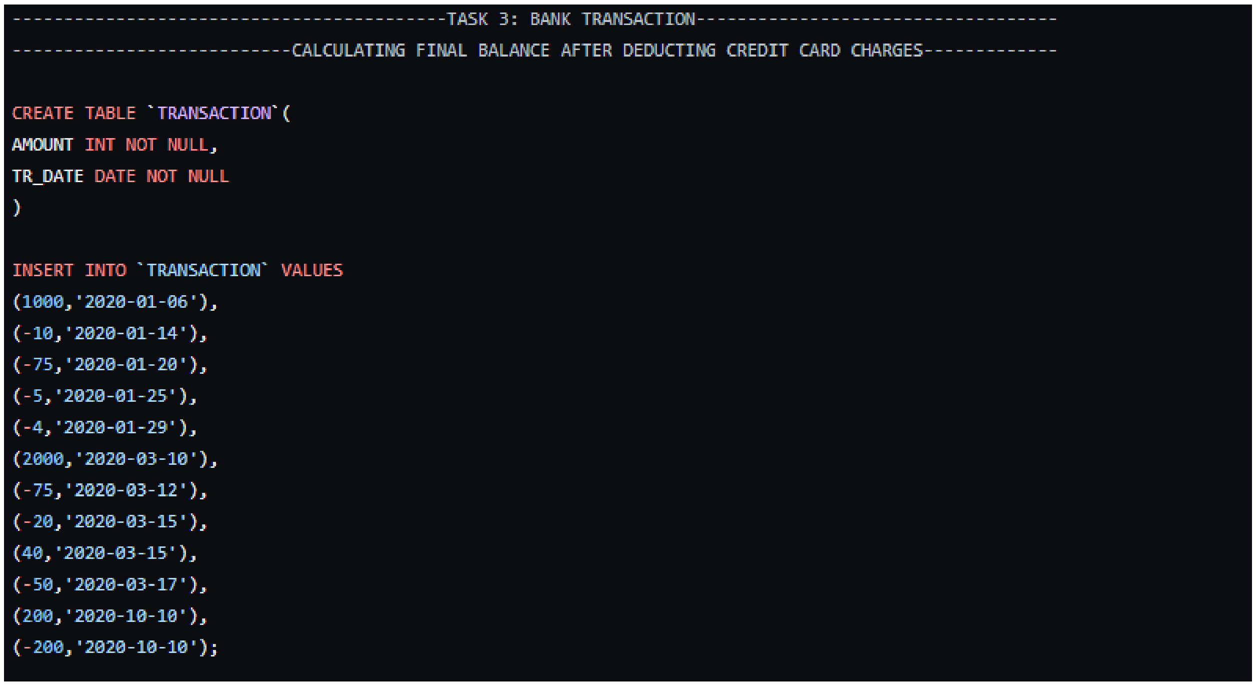
your query should return:

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

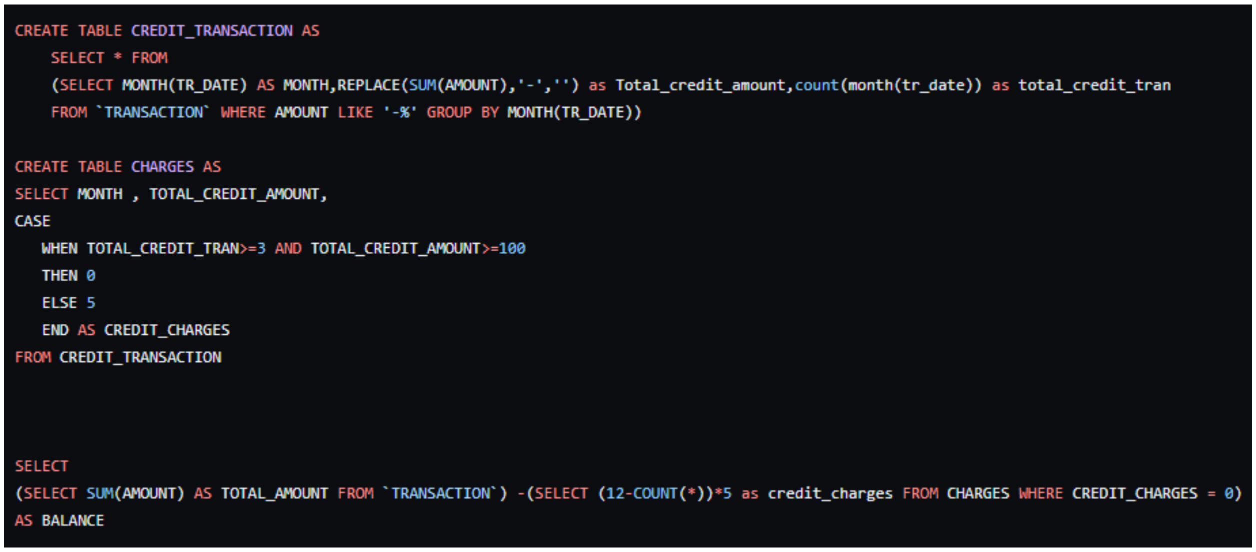
The balance excluding the fee would be -562. You are not charged the fee in February since you had four card payments for a total cost of 50 + 1 + 44 + 5 = 100 in that month. You are also not charged the fee in December since you had three card payments for a total cost of 100 + 100 + 100 = 300. The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

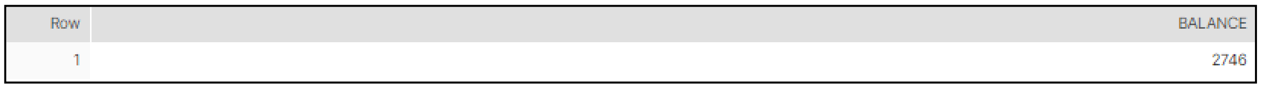


```
------------------------------------------TASK 3: BANK TRANSACTION----------------------------------
--------------------------CALCULATING FINAL BALANCE AFTER DEDUCTING CREDIT CARD CHARGES------------

CREATE TABLE `TRANSACTION`(
AMOUNT INT NOT NULL,
TR_DATE DATE NOT NULL
)

INSERT INTO `TRANSACTION` VALUES
(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');
```



```sql
CREATE TABLE CREDIT_TRANSACTION AS
    SELECT * FROM
    (SELECT MONTH(TR_DATE) AS MONTH,REPLACE(SUM(AMOUNT),'-','') as Total_credit_amount,count(month(tr_date)) as total_credit_tran
    FROM `TRANSACTION` WHERE AMOUNT LIKE '-%' GROUP BY MONTH(TR_DATE))

CREATE TABLE CHARGES AS
SELECT MONTH , TOTAL_CREDIT_AMOUNT,
CASE
   WHEN TOTAL_CREDIT_TRAN>=3 AND TOTAL_CREDIT_AMOUNT>=100
   THEN 0
   ELSE 5
   END AS CREDIT_CHARGES
FROM CREDIT_TRANSACTION



SELECT
(SELECT SUM(AMOUNT) AS TOTAL_AMOUNT FROM `TRANSACTION`) -(SELECT (12-COUNT(*))*5 as credit_charges FROM CHARGES WHERE CREDIT_CHARGES = 0)
AS BALANCE
```

OUTPUT



Row	BALANCE
1	2746