Catching Fraud

1. Analysis of the query

```
WITH processed_users -- Creation of a subtable
      AS (SELECT LEFT(u.phone_country, 2) AS short_phone_country, --Selection
of the first two characters of the phone number
                u.id
         FROM   users u)
SELECT t.user_id,
       t.merchant_country,
        Sum(t.amount / fx.rate / Power(10, cd.exponent)) AS amount -- Convert
amount to Euro
FROM   transactions t
       JOIN fx_rates fx
         ON ( fx.ccy = t.currency
                 AND fx.base_ccy = 'EUR' ) -- Get the exchange rate info to
convert all amounts in Euros
       JOIN currency_details cd
         ON cd.currency = t.currency
       JOIN processed_users pu
         ON pu.id = t.user_id
WHERE  t.source = 'GAIA'
          AND pu.short_phone_country = t.merchant_country -- Filter to get
transaction in the country of the phone of the user
GROUP  BY t.user_id,
          t.merchant_country
ORDER  BY amount DESC;
```

The goal of this query is to get the amount of money spent, in Euros, by users in the country of their phone. It sorts amounts from the highest to the lowest, filtering only expenses coming from source GAIA.

The query itself is working as it is correctly written. But it give us no results as the condition:

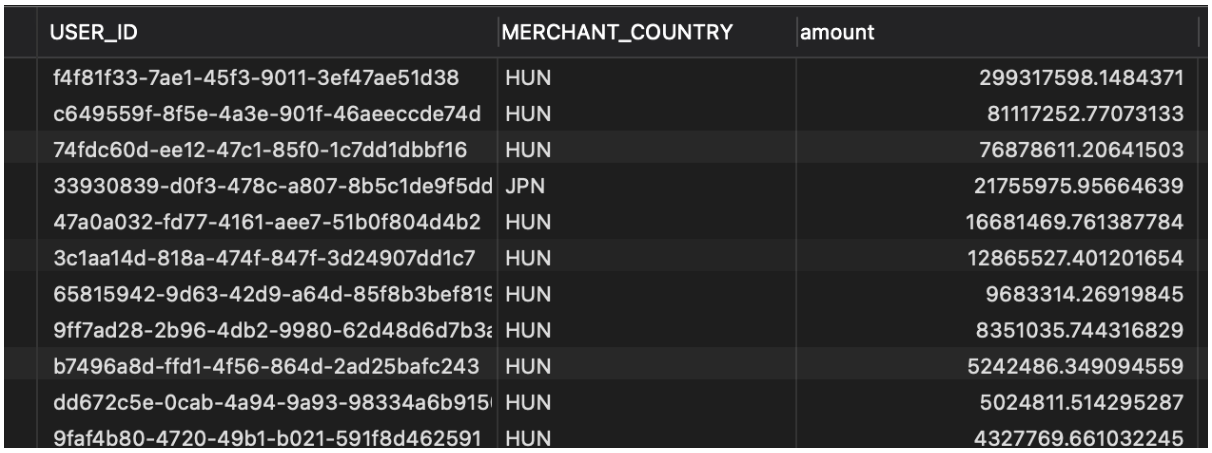
```
AND pu.short_phone_country = t.merchant_country
```

doesn't send us anything as short_phone_country has only two characters, while merchant_country has three.

We fix the query :

```
WITH processed_users -- Creation of a subtable
     AS (SELECT LEFT(u.phone_country, 2) AS short_phone_country,
                u.id
         FROM   users u)
SELECT t.user_id,
       t.merchant_country,
          Sum(t.amount / fx.rate / Power(10, cd.exponent)) AS amount FROM
transactions t
       JOIN fx_rates fx
         ON ( fx.ccy = t.currency
              AND fx.base_ccy = 'EUR' )
       JOIN currency_details cd
```

```sql
         ON cd.currency = t.currency
       JOIN processed_users pu
         ON pu.id = t.user_id
WHERE  t.source = 'GAIA'
       AND pu.short_phone_country = LEFT(t.merchant_country, 2)
GROUP  BY t.user_id,
          t.merchant_country
ORDER  BY amount DESC;
```

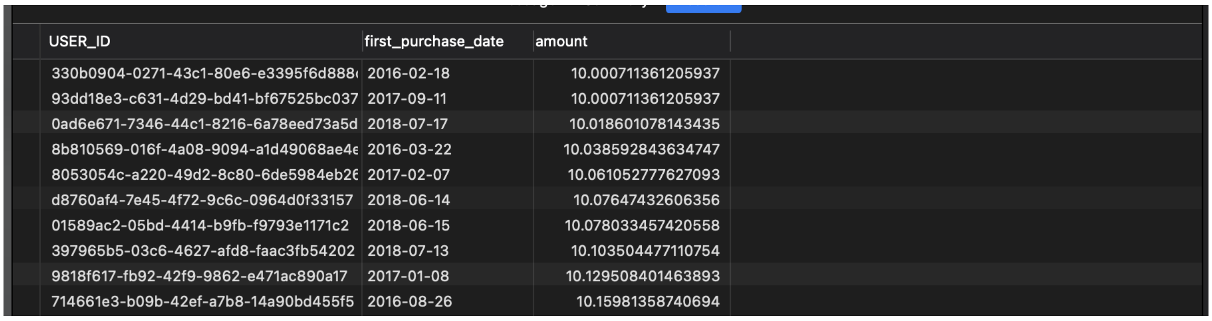


USER_ID	MERCHANT_COUNTRY	amount
f4f81f33-7ae1-45f3-9011-3ef47ae51d38	HUN	299317598.1484371
c649559f-8f5e-4a3e-901f-46aeeccde74d	HUN	81117252.77073133
74fdc60d-ee12-47c1-85f0-1c7dd1dbbf16	HUN	76878611.20641503
33930839-d0f3-478c-a807-8b5c1de9f5dd	JPN	21755975.95664639
47a0a032-fd77-4161-aee7-51b0f804d4b2	HUN	16681469.761387784
3c1aa14d-818a-474f-847f-3d24907dd1c7	HUN	12865527.401201654
65815942-9d63-42d9-a64d-85f8b3bef819	HUN	9683314.26919845
9ff7ad28-2b96-4db2-9980-62d48d6d7b3a	HUN	8351035.744316829
b7496a8d-ffd1-4f56-864d-2ad25bafc243	HUN	5242486.349094559
dd672c5e-0cab-4a94-9a93-98334a6b915	HUN	5024811.514295287
9faf4b80-4720-49b1-b021-591f8d462591	HUN	4327769.661032245

2. Catching users with first succeed purchase over 10 USD

```sql
with completed_transaction as (
   select t."USER_ID"
,t."CREATED_DATE"
,(t."AMOUNT" / fx."rate" / Power(10, cd.exponent)) AS amount
FROM   transactions t
      JOIN fx_rates fx
        ON ( fx."ccy" = t."CURRENCY"
             AND fx."base_ccy" = 'USD' )
            JOIN currency_details cd
        ON cd."currency" = t."CURRENCY"
where t."STATE" ='COMPLETED'
and (t."AMOUNT" / fx."rate" / Power(10, cd.exponent))>10)
,
first_date as (
   select "USER_ID"
   ,min("CREATED_DATE") AS first_purchase_date
   from completed_transaction ct
   GROUP BY "USER_ID")

SELECT
fd."USER_ID"
,fd."first_purchase_date"
,ct."amount"
from first_date fd
LEFT  JOIN  completed_transaction  ct  on  (fd."first_purchase_date"=ct."CREATED_DATE"
and fd."USER_ID"=ct."USER_ID")
ORDER  BY ct."amount";
```



USER_ID	first_purchase_date	amount
330b0904-0271-43c1-80e6-e3395f6d888	2016-02-18	10.000711361205937
93dd18e3-c631-4d29-bd41-bf67525bc037	2017-09-11	10.000711361205937
0ad6e671-7346-44c1-8216-6a78eed73a5d	2018-07-17	10.018601078143435
8b810569-016f-4a08-9094-a1d49068ae4	2016-03-22	10.038592843634747
8053054c-a220-49d2-8c80-6de5984eb2	2017-02-07	10.061052777627093
d8760af4-7e45-4f72-9c6c-0964d0f33157	2018-06-14	10.07647432606356
01589ac2-05bd-4414-b9fb-f9793e1171c2	2018-06-15	10.078033457420558
397965b5-03c6-4627-afd8-faac3fb54202	2018-07-13	10.103504477110754
9818f617-fb92-42f9-9862-e471ac890a17	2017-01-08	10.129508401463893
714661e3-b09b-42ef-a7b8-14a90bd455f5	2016-08-26	10.15981358740694

3. Fraudster

I didn’t have time to realize this exercise but I would apply the following strategy.

1. We are missing labelled data of people that are not fraudsters, we only have Fraudster people. We cannot build a Supervised predictive model. If we consider not identified people as non fraudster we could catch false positive and give a wrong information to our sistem

2. I would transform the dataset to have a list of features that could describe each users, such as:

    a. Prefered transaction type
    b. Average transaction amount
    c. Amount of first transaction
    d. Percentage of transaction in the country

3. I would use all of these features to build a non supervised clusterization probably using the K-Means method.

4. I would see the percentage of Fraudsters in each group. Depending on the result I would be able to identify people with really similar behaviours than our identified fraudsters and probably catch them as Fraudsters.