CATCHING FRAUD

Revolut Hometask-2

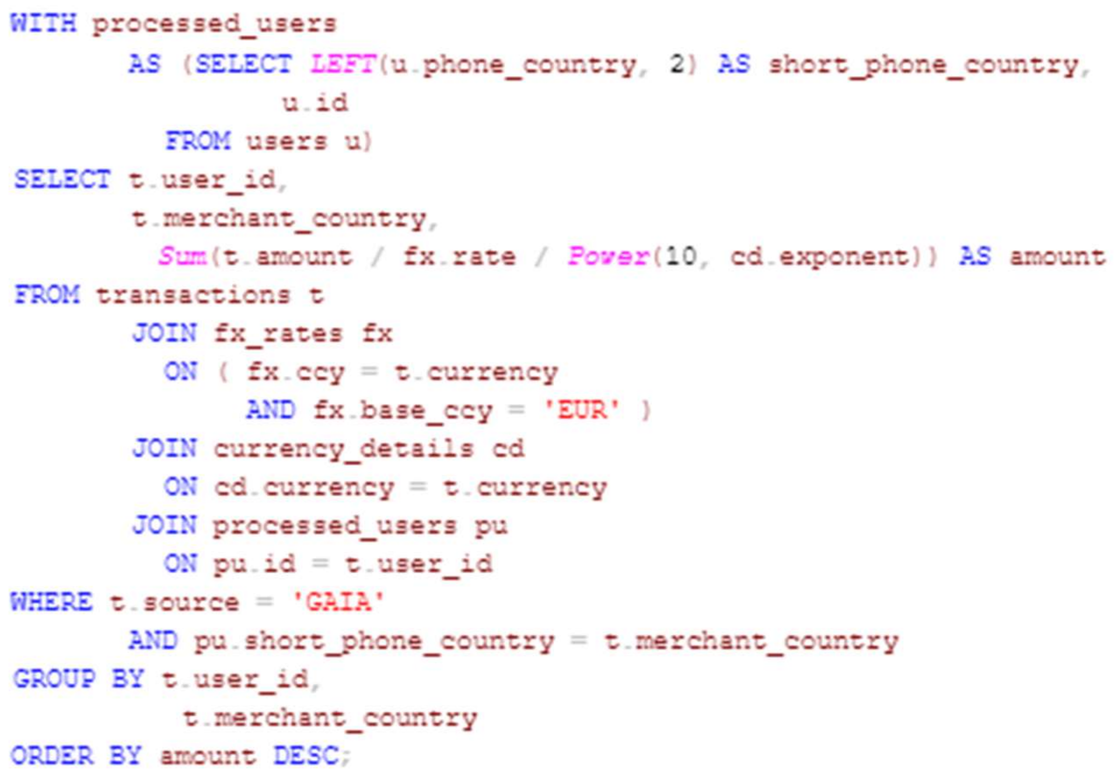
PURPOSE

Based on an unusual observation, noticed by financial crime team, the aim is to investigate the possibility of fraudulent transactions using the relevant data regarding these international monetary transactions.

Objective 1

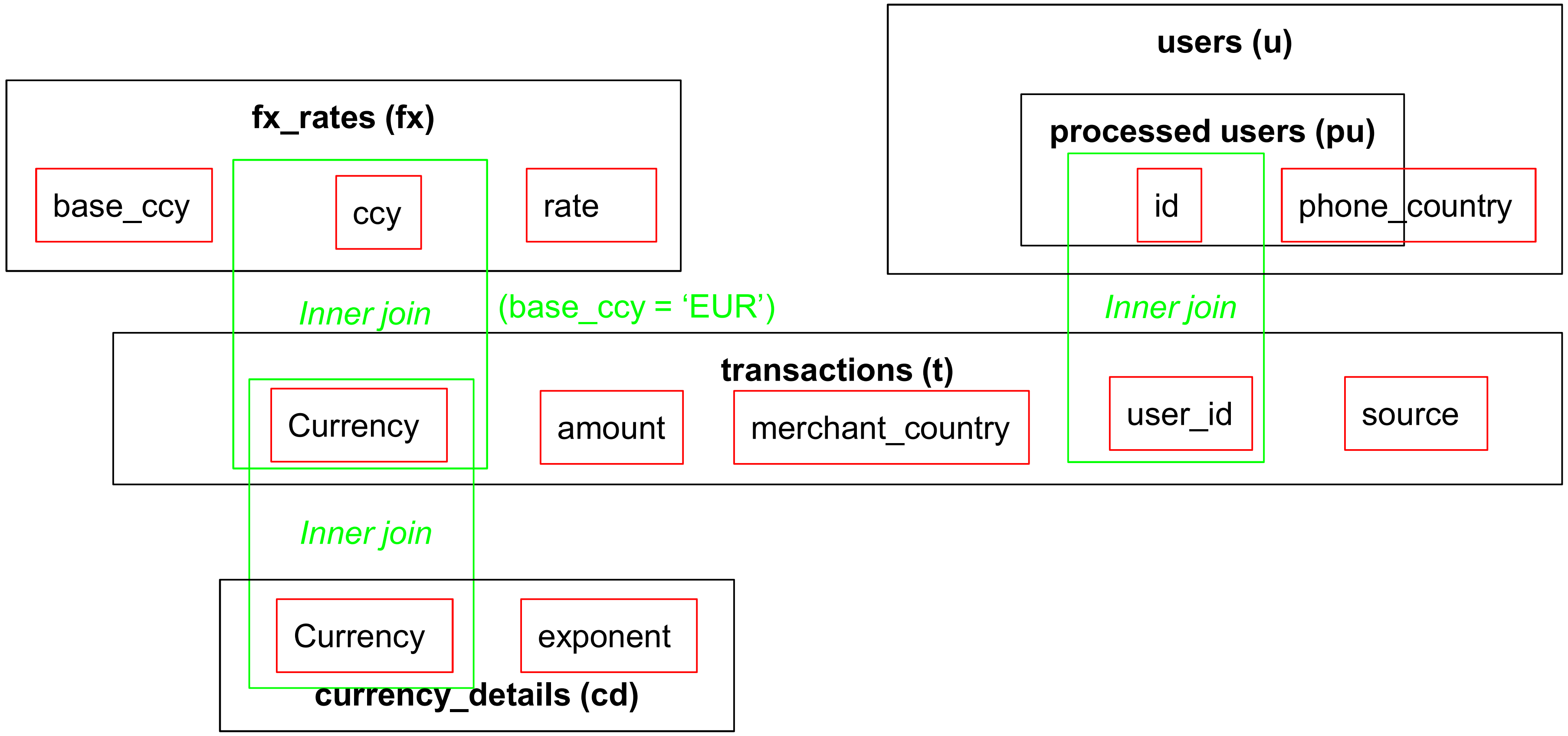
Examining the given SQL query

THE QUERY



```sql
WITH processed_users
       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
         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
GROUP BY t.user_id,
          t.merchant_country
ORDER BY amount DESC;
```

QUERY PROCESSING: INNER JOINS



QUERY PROCESSING: CONDITIONS

In the previous slide we have joined three different tables i.e *fx_rate, processed_users, currency_details* to the table *transactions*. Now, out of all the transactions only certain transactions were chosen based on the conditions described below -

1. Source of transaction must be ‘GAIA’.
2. *short_phone_country* in *processed_users* table, which is equivalent to the first two characters in *phone_country* in the *users* table, must be equal to the merchant_country of the transaction.

QUERY PROCESSING: TOTAL AMOUNT

After applying the condition and eliminating the unwanted transactions, the resulting table has limited rows now. From the resulting table, the query converts the amount used in each transaction into ‘Euro’ using the following equation -

amount = t.amount / fx.rate / Power(10, cd.exponent)

For every unique pair of user_id and merchant_country, there may be more than one transactions implying more than one amounts. All those amounts are summed up to calculate the net amount transacted by an individual.

QUERY PROCESSING: RESULT

After calculating the net amount, the query finally generates the output table which has three columns i.e. user_id, merchant_country and the net amount calculated for the pair sorted in the descending order of the amount.

Result

t.user_id *t.merchant_country* Sum(t.amount / fx.rate / Power (10, cd.exponent))

= *net amount in 'EURO'*

CONCLUSION

The aim of the query was to analyze and compare total amount of transaction made by every individual from the source ‘GAIA’.

WILL THIS QUERY WORK?

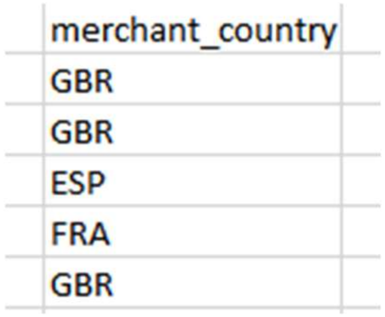
After analyzing the working of the query, it is found that it has some critical issues which would hinder the working of query in the expected manner.

MISMATCH OF PHONE COUNTRY

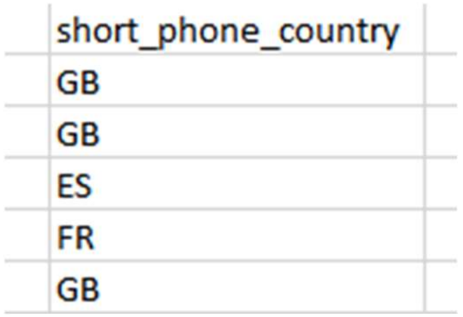
```
pu.short_phone_country = t.merchant_country
```

The condition mentioned above is the second condition of WHERE in the SQL query. Every element in ‘short_phone_country’ column in the ‘processed_user’ will not have more than two characters but the format of ‘merchant_country’ in the ‘transactions’ is such that it could always have more than two characters.

This will lead to the mismatch of the elements. For the above condition to work we have to strip the elements of merchant_country column so that only first two characters are left.



merchant_country
GBR
GBR
ESP
FRA
GBR



short_phone_country
GB
GB
ES
FR
GB

CURRENCY COMPARISON

```
( fx.ccy = t.currency AND fx.base_ccy = 'EUR' )
```

The condition mentioned above is the condition of inner join between the tables ‘transactions’ and ‘fx_rates’. When the elements in ‘base_ccy’ column will have value ‘EUR’ then no element in ‘ccy’ column will have value ‘EUR’. This implies that when t.currency will be equal to ‘EUR’ then there is no corresponding row representing fx rates for that (which should ideally be ‘1’) which will lead to either null or error result.

ANSWER

The query will work fine if the two previously stated issues are taken care of.

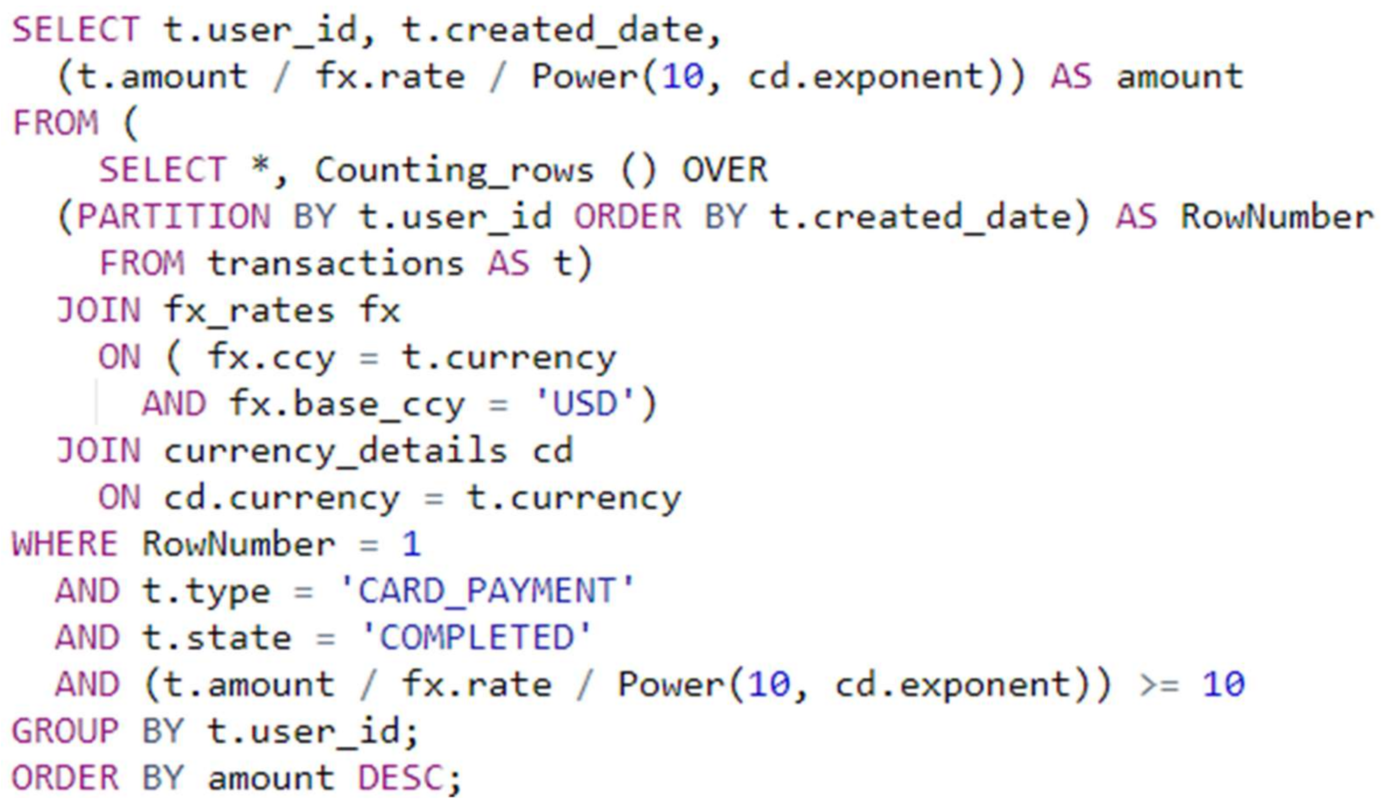
Objective 2

Writing a SQL query

AIM

To write a query to identify users whose first transaction was a successful card payment over $10 USD equivalent.

SOLUTION



```sql
SELECT t.user_id, t.created_date,
  (t.amount / fx.rate / Power(10, cd.exponent)) AS amount
FROM (
    SELECT *, Counting_rows () OVER
  (PARTITION BY t.user_id ORDER BY t.created_date) AS RowNumber
    FROM transactions AS 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 RowNumber = 1
  AND t.type = 'CARD_PAYMENT'
  AND t.state = 'COMPLETED'
  AND (t.amount / fx.rate / Power(10, cd.exponent)) >= 10
GROUP BY t.user_id;
ORDER BY amount DESC;
```

Objective 3

Finding fraudsters

INCENTIVE

The aim is to find out likely fraudsters from a set of data related to international transactions made across a long time period with the help of a given set of known fraudsters.

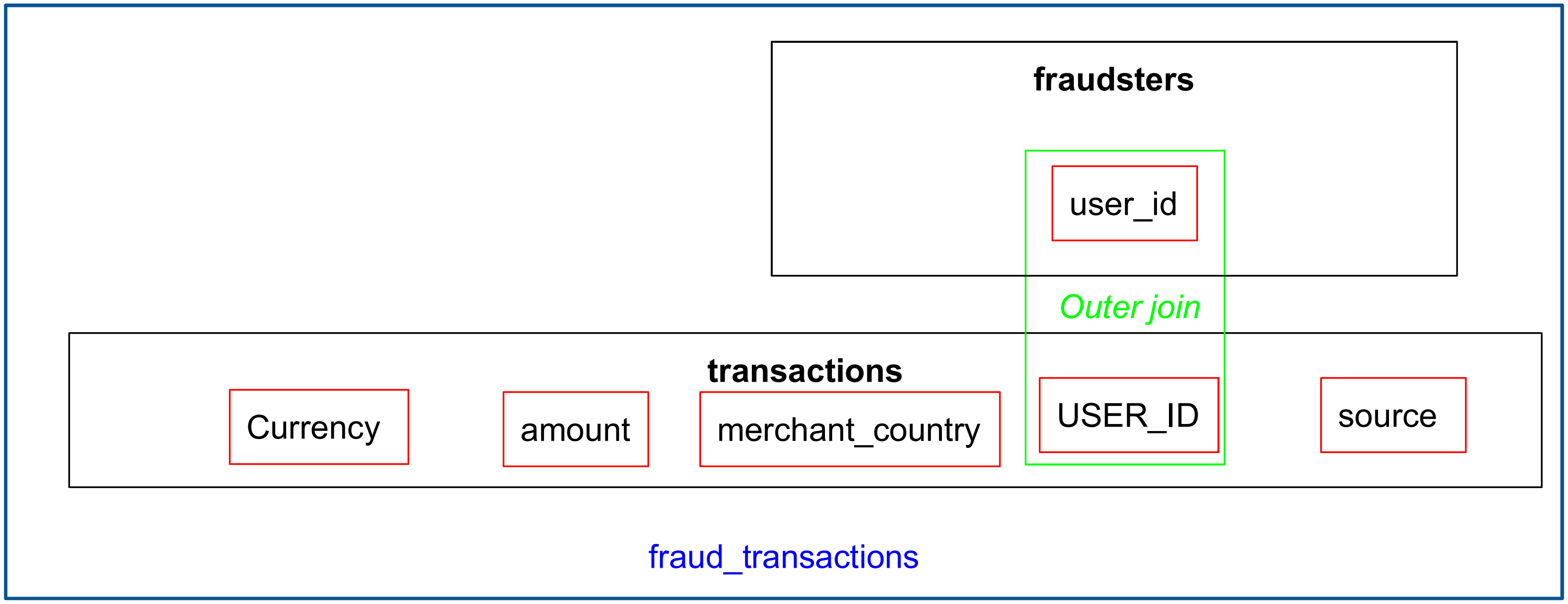
INITIALIZATION

The prediction of likely fraudsters will start with analysing the given data. The given data has two necessary tables -

1. transactions.csv - record of all the transactions made in past several years
2. fraudsters.csv - list of all the identified fraudsters

The first likely step is the outer join of these two tables to merge them into one table for easy observation and manipulation. The final table is named as ‘fraud_transactions’.

OUTER JOIN

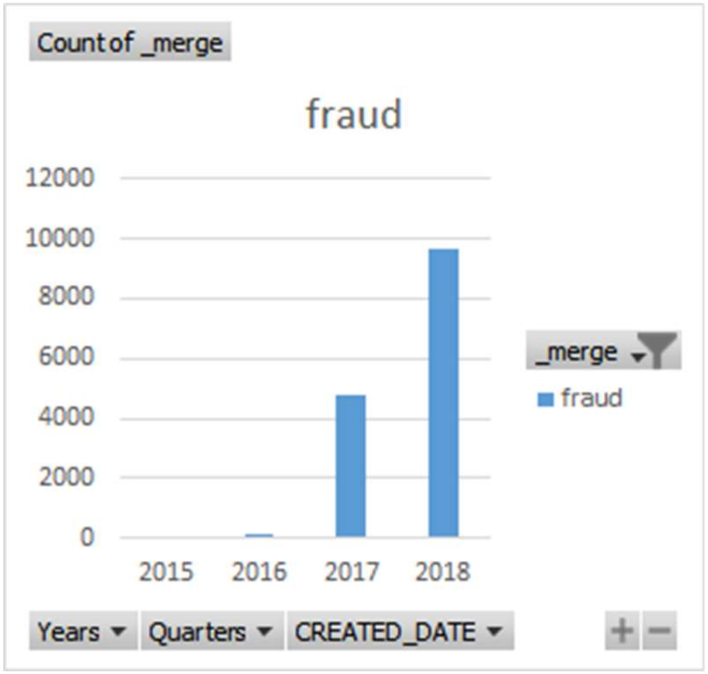

fraudsters
user_id
Outer join
transactions
Currency
amount
merchant_country
USER_ID
source
fraud_transactions

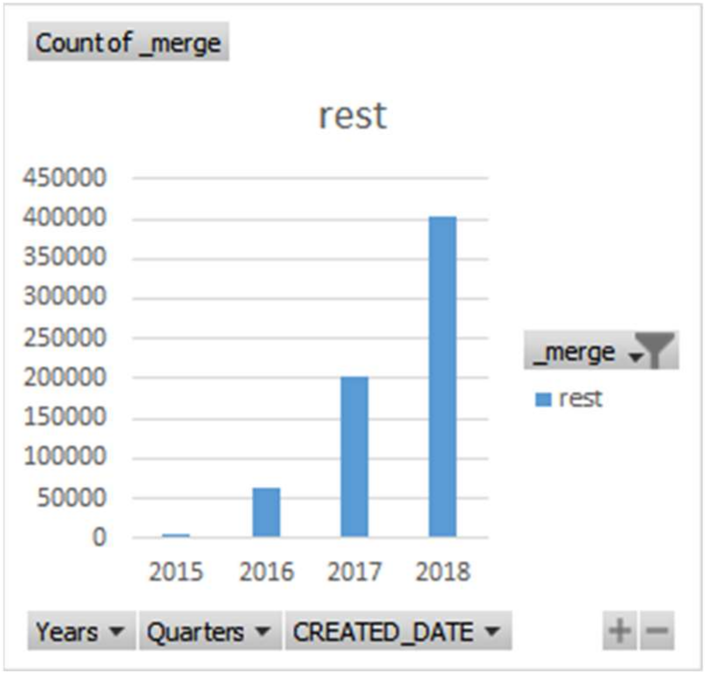
FINDING UNUSUAL BEHAVIOUR

The hypothesis is the fact that there must some similarity between identified and unidentified fraudsters making the fraud transactions. Therefore to identify the fraudsters from rest of the people we must look for any unusual trend followed by the identified fraudsters which rest of the people do not follow.

After the outer join, fraud_transactions table was used for the analysis of any unusual behaviour. The next slides presents all the unusual trends obtained while the thorough analysis of the data.

TRANSACTIONS COUNT VS YEARS

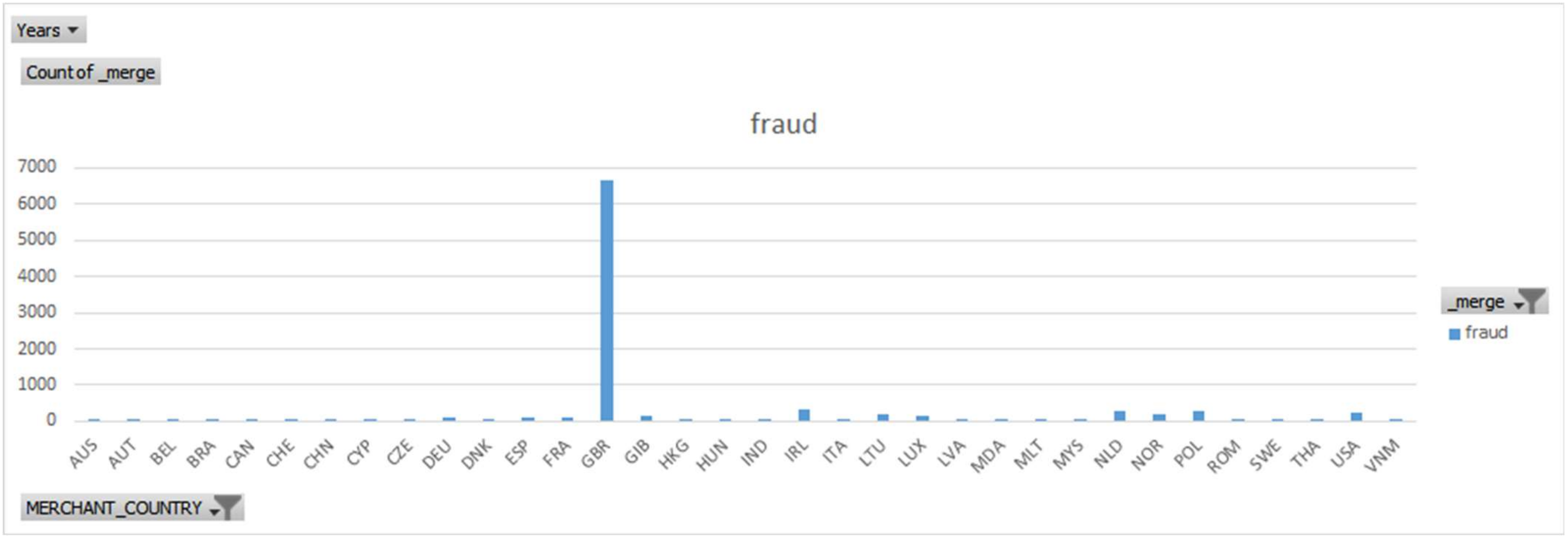




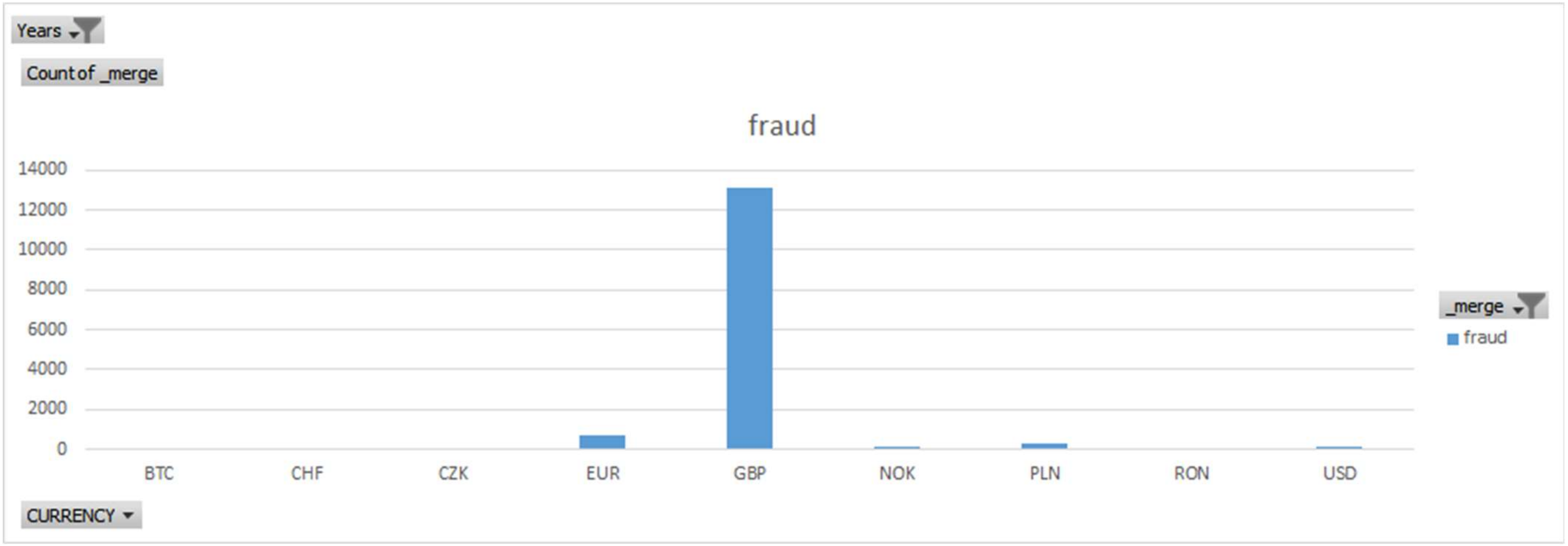
TRANSACTIONS COUNT VS YEARS

From the graph we can observe that almost every fraudulent transaction occurred in the year 2017 and 2018. Therefore, all the trends plotted in the upcoming slides is filtered by the year 2017 and 2018.

TRANSACTIONS COUNT VS MERCHANT COUNTRY



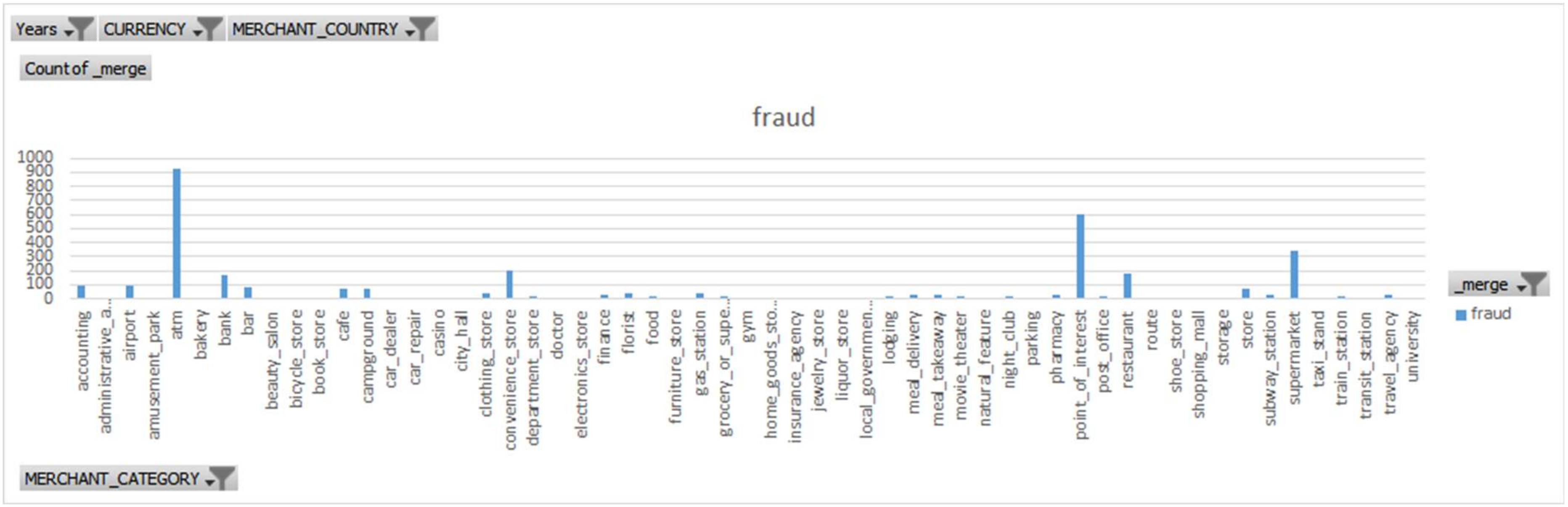
TRANSACTIONS COUNT VS CURRENCY



TRANSACTIONS COUNT VS MERCHANT COUNTRY AND CURRENCY

From the chart, we have seen that most of the fraudulent transactions were from the country ‘GBR’ which has the currency ‘GBP’. Therefore, to identify potential new fraudsters we will focus on the country ‘GBR’ and currency ‘GBP’. All the trends in the upcoming slides are filtered by the merchant_country ‘GBR’ and currency ‘GBP’.

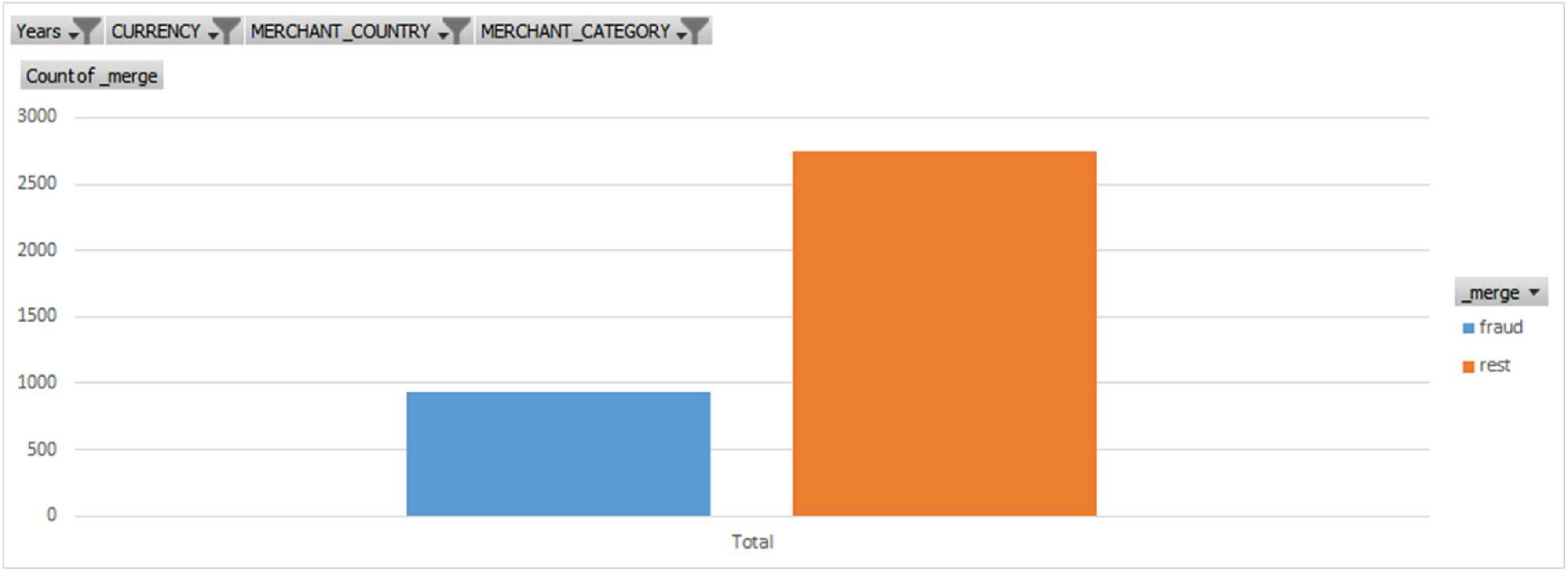
TRANSACTIONS COUNT VS MERCHANT CATEGORY



TRANSACTIONS COUNT VS MERCHANT CATEGORY

From the chart, we can see that atm is the most frequent place for fraudulent transactions Therefore, to better analyze the fraudster’s data the further trends were plotted only for ‘atm’.

TRANSACTIONS COUNT AFTER APPLYING FILTERS



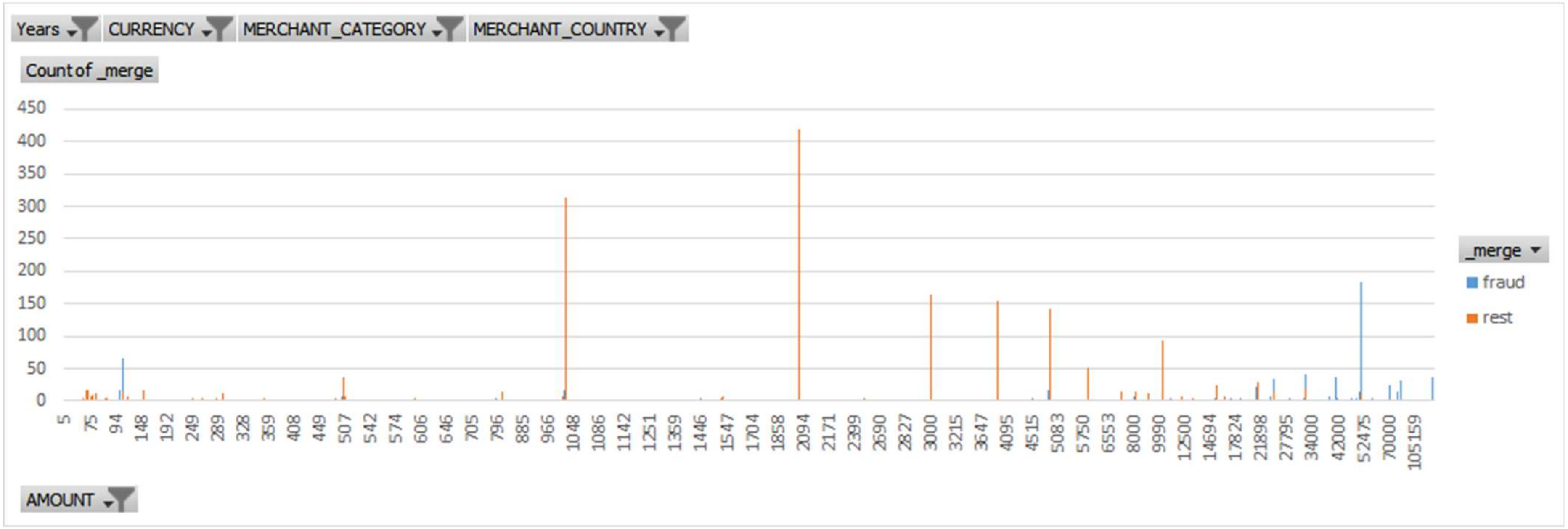
TRANSACTIONS COUNT AFTER APPLYING FILTERS

In the previous graph we have compared the count of fraudulent transactions after applying the filters mentioned in the previous slides. Those filters are also listed below -

1. Year - 2017 and 2018
2. merchant_country - GBR
3. currency - ‘GBP’
4. merchant_category - atm

We can see that the count of fraudulent transactions is very comparable with the rest count. Now we will see how the count varies with ‘amount of transaction’ with these filters.

TRANSACTION COUNT VS AMOUNT



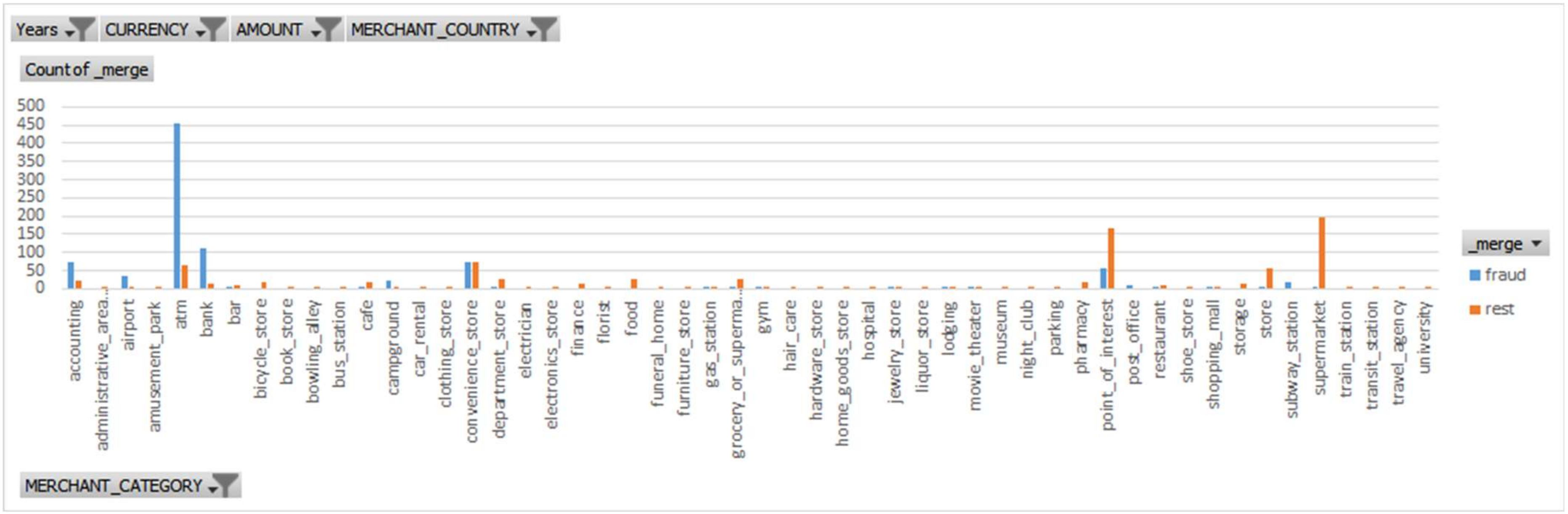
TRANSACTION COUNT VS AMOUNT

The previous chart is very crucial for identifying the pattern used by the fraudsters in their transactions. We can observe that certain transaction amounts are used much more frequently by fraudsters than by the rest people. Majority of these amounts are listed in the following array -

[100, 25000, 30000, 40000, 50000, 70000, 80000, 288900]

Now, we will also use this amount filter and once again analyze the merchant_category distribution which was primarily ‘atm’ before.

TRANSACTIONS COUNT VS MERCHANT CATEGORY

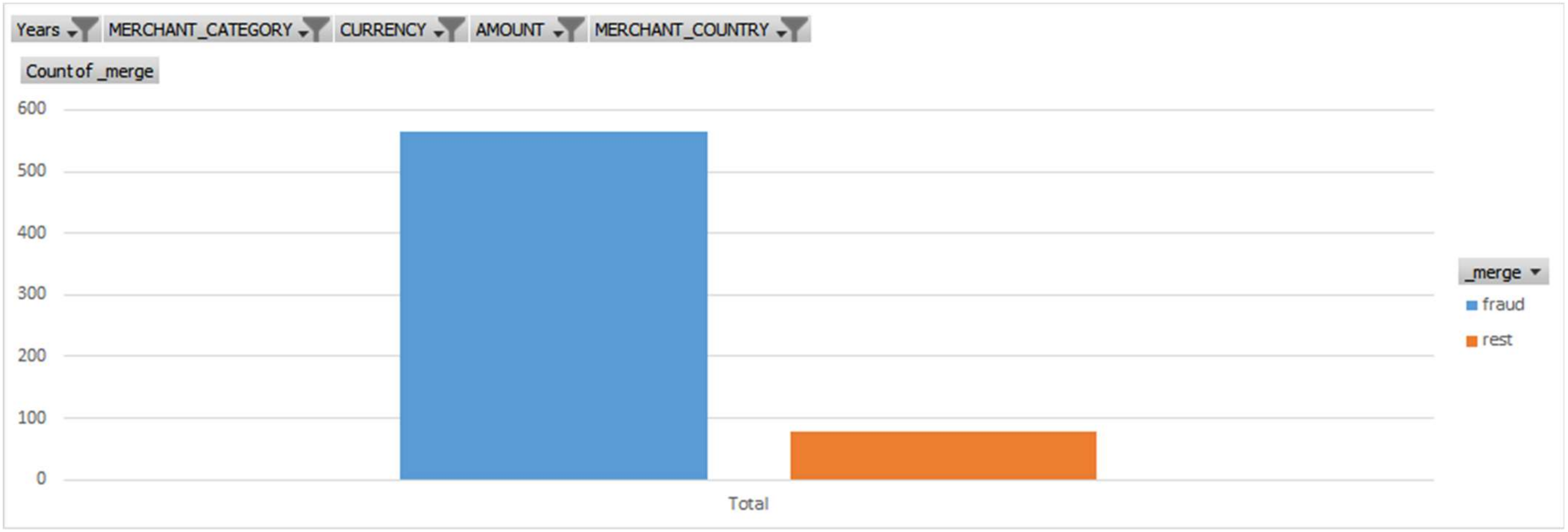


TRANSACTIONS COUNT VS MERCHANT CATEGORY

We can now see that, not only the atm plays an important role in fraudulent transactions but ‘bank’ also plays a significant role in it as its relative count of fraudulent transactions is also very high.

Therefore, the filter has been updated from ‘atm’ to ‘atm’ and ‘bank’.

TRANSACTIONS COUNT WITH THE NEW FILTERS



TRANSACTIONS COUNT WITH THE NEW FILTERS

In the previous chart, we can observe that under the given circumstances the count of fraudulent transactions is very high than the count of rest transactions. It is highly probable that any transaction made in the rest transactions category is fraudulent. Therefore, user ids of all the rest transaction were obtained and the top 5 frequent transactions were reported.

ANSWER

USER ID	Count of transactions
f590dcd7-b3b7-493b-9a1f-210fc319485d	9
50552988-e421-4a57-ba08-2f7f9dc6cae4	7
ca027a38-c6eb-4f66-8649-000948986833	4
01a29ba5-4c7d-4da7-aa56-59138d4a0859	4
79638a4e-e6df-43ae-8d82-14832364544b	3