OVERVIEW:

For these tasks we will need to perform analysis on the given data sets. We will consider three data sets as given in the information:

1. Transactions data (transactions)
2. Answers to given prop questions (answers)
3. information about the usage of content items (info)

TOOLS REQUIRED:

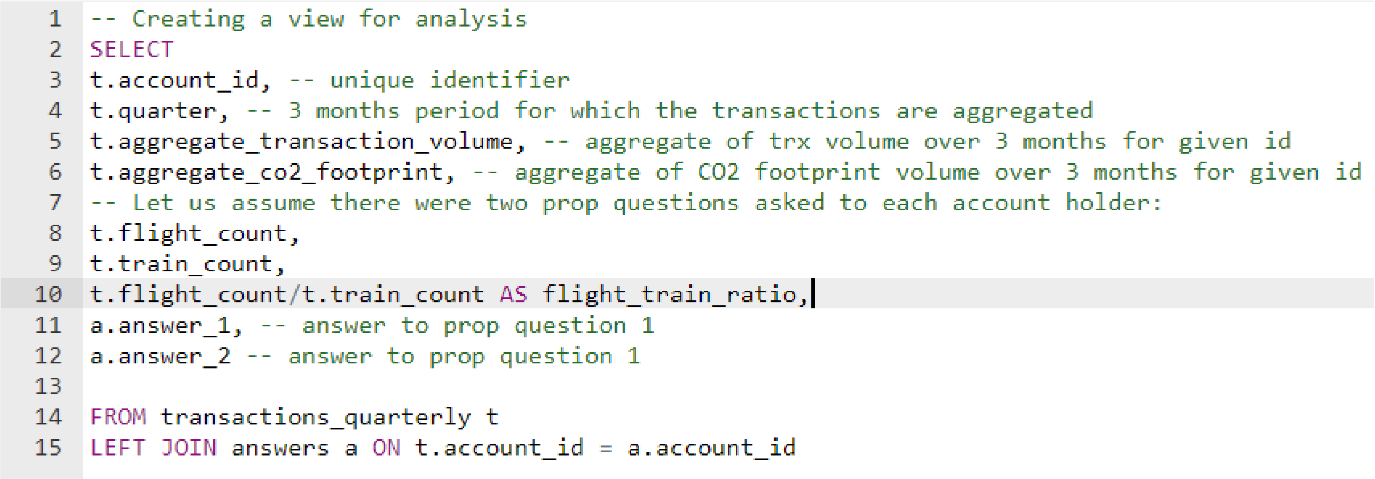
To perform the analysis, the project requires the use of SQL querying tool : such as AWS Redshift, Google Big Query, Snowflake, etc. and a Python editor such as Google Colab or Jupyter notebook, etc.

To store the data we will need access to some form of secure cloud storage and we will use Github for version control.

TASK 1:

Clustering of bank users from Example 1.

For this task, we will use the two given data tables, the first containing the transactions information aggregated over 3 months (transactions_quarterly) and the second containing the answers from bank users to particular questions (answers). We will join these two tables at account id as follows:



```sql
-- Creating a view for analysis
SELECT
t.account_id, -- unique identifier
t.quarter, -- 3 months period for which the transactions are aggregated
t.aggregate_transaction_volume, -- aggregate of trx volume over 3 months for given id
t.aggregate_co2_footprint, -- aggregate of CO2 footprint volume over 3 months for given id
-- Let us assume there were two prop questions asked to each account holder:
t.flight_count,
t.train_count,
t.flight_count/t.train_count AS flight_train_ratio,
a.answer_1, -- answer to prop question 1
a.answer_2 -- answer to prop question 1

FROM transactions_quarterly t
LEFT JOIN answers a ON t.account_id = a.account_id
```

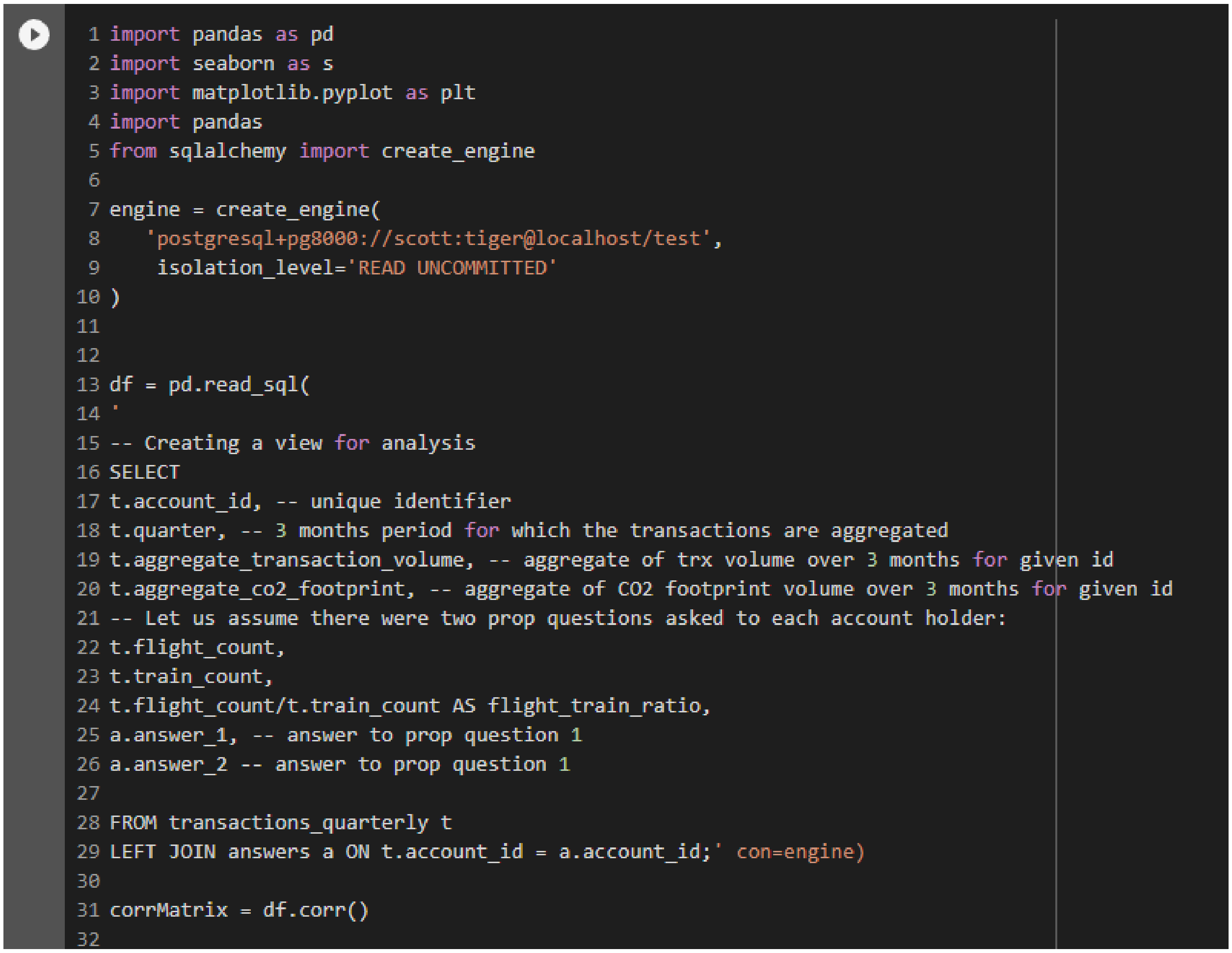
This will give us a view as follows, which can be used for further analysis:

account_id	quarter	aggregate_transaction_volume	aggregate_co2_footprint	flight_count	train_count	flight_train_ratio	answer_1	answer_2
(INT)	(DATE)	(DOUBLE)	(DOUBLE)	(INT)	(INT)	(DOUBLE)	(VARCHAR)	(VARCHAR)

Approach to clustering: To find clusters in the given data, we need to find the correlation between the different fields in the given data. This can be achieved by generating a correlation matrix using the Pandas library in Python.

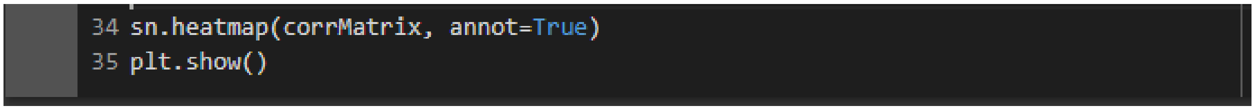
We will use the **pd.corr()** function from **Pandas** library in Python to easily generate the correlation matrix.

For this task we will need to make use of a Python editor such as **Google Colab**:



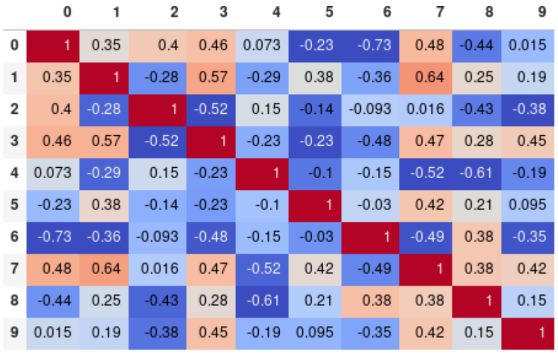
```
import pandas as pd
import seaborn as s
import matplotlib.pyplot as plt
import pandas
from sqlalchemy import create_engine

engine = create_engine(
   'postgresql+pg8000://scott:tiger@localhost/test',
    isolation_level='READ UNCOMMITTED'
)


df = pd.read_sql(
'
-- Creating a view for analysis
SELECT
t.account_id, -- unique identifier
t.quarter, -- 3 months period for which the transactions are aggregated
t.aggregate_transaction_volume, -- aggregate of trx volume over 3 months for given id
t.aggregate_co2_footprint, -- aggregate of CO2 footprint volume over 3 months for given id
-- Let us assume there were two prop questions asked to each account holder:
t.flight_count,
t.train_count,
t.flight_count/t.train_count AS flight_train_ratio,
a.answer_1, -- answer to prop question 1
a.answer_2 -- answer to prop question 1

FROM transactions_quarterly t
LEFT JOIN answers a ON t.account_id = a.account_id;' con=engine)

corrMatrix = df.corr()

```

This will give us a matrix where the the values vary between - 1 and 1, The heatmap can be plotted for better visualization using matplotlib and seaborn libraries:



```
sn.heatmap(corrMatrix, annot=True)
plt.show()
```

So, we will have a correlation matrix that looks something like this:



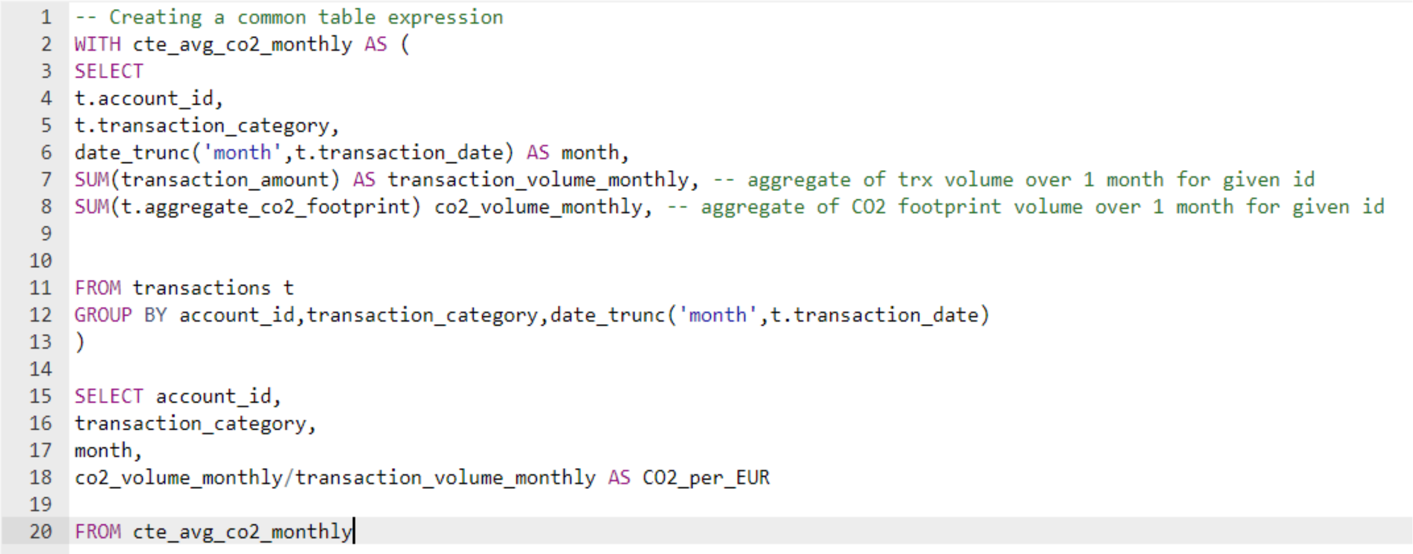
	0	1	2	3	4	5	6	7	8	9
0	1	0.35	0.4	0.46	0.073	-0.23	-0.73	0.48	-0.44	0.015
1	0.35	1	-0.28	0.57	-0.29	0.38	-0.36	0.64	0.25	0.19
2	0.4	-0.28	1	-0.52	0.15	-0.14	-0.093	0.016	-0.43	-0.38
3	0.46	0.57	-0.52	1	-0.23	-0.23	-0.48	0.47	0.28	0.45
4	0.073	-0.29	0.15	-0.23	1	-0.1	-0.15	-0.52	-0.61	-0.19
5	-0.23	0.38	-0.14	-0.23	-0.1	1	-0.03	0.42	0.21	0.095
6	-0.73	-0.36	-0.093	-0.48	-0.15	-0.03	1	-0.49	0.38	-0.35
7	0.48	0.64	0.016	0.47	-0.52	0.42	-0.49	1	0.38	0.42
8	-0.44	0.25	-0.43	0.28	-0.61	0.21	0.38	0.38	1	0.15
9	0.015	0.19	-0.38	0.45	-0.19	0.095	-0.35	0.42	0.15	1

Fig. example correlation matrix.

From this we can find out which fields are interdependent, the higher the index of correlation, the higher is interdependence , for example - if the index in the matrix between flight_train_ratio and answer_1 is very high, we can plot flight_train_ratio vs answer_1 and we should be able to group them together to form **clusters**.

TASK 2:

To visualize how the average CO2 value per EUR spent evolves over time on a monthly basis, we will have to transform the transactions data in the following way

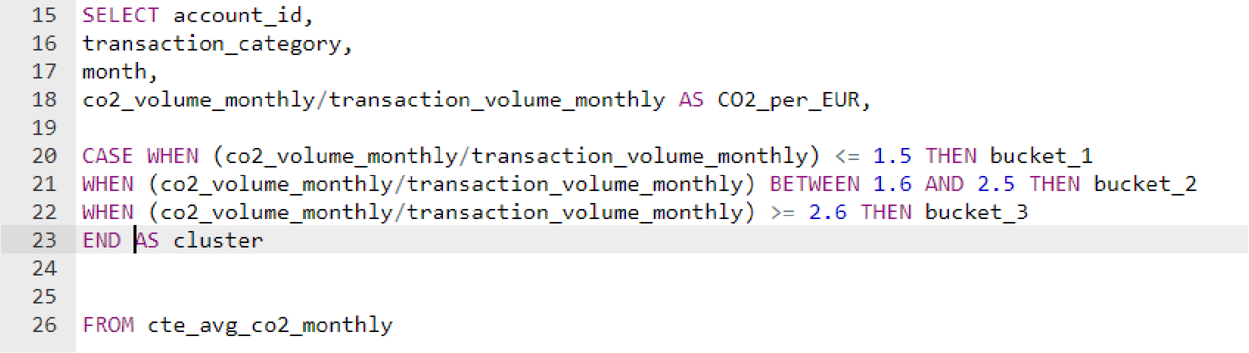


```
-- Creating a common table expression
WITH cte_avg_co2_monthly AS (
SELECT
t.account_id,
t.transaction_category,
date_trunc('month',t.transaction_date) AS month,
SUM(transaction_amount) AS transaction_volume_monthly, -- aggregate of trx volume over 1 month for given id
SUM(t.aggregate_co2_footprint) co2_volume_monthly, -- aggregate of CO2 footprint volume over 1 month for given id


FROM transactions t
GROUP BY account_id,transaction_category,date_trunc('month',t.transaction_date)
)

SELECT account_id,
transaction_category,
month,
co2_volume_monthly/transaction_volume_monthly AS CO2_per_EUR

FROM cte_avg_co2_monthly
```

The above query will give us the CO2 emission per euro for each category of transaction over a monthly timeline.

To find out how stable these values are, we need to calculate the variance of the CO2_per_EUR column. For this, we can use the VAR ([Column_Name]) function in SQL.

The best way to find clusters here is to group the transaction categories and account_ids into different buckets of CO2_per_EUR values. For example:



```
SELECT account_id,
transaction_category,
month,
co2_volume_monthly/transaction_volume_monthly AS CO2_per_EUR,

CASE WHEN (co2_volume_monthly/transaction_volume_monthly) <= 1.5 THEN bucket_1
WHEN (co2_volume_monthly/transaction_volume_monthly) BETWEEN 1.6 AND 2.5 THEN bucket_2
WHEN (co2_volume_monthly/transaction_volume_monthly) >= 2.6 THEN bucket_3
END AS cluster


FROM cte_avg_co2_monthly
```

The above way is one approach to clustering the data based on CO2 per Euro.

TASK 3:

When the amount spent for gas or diesel drops significantly and in the same period, the amount of bought train/bus tickets increases, this can be a strong indicator that a person has switched from car to public transport, however we cannot say for certain that is the case as there could be an alternative explanation: for example a person had to make a few trips outside the city by train in that period, this does not necessarily translate to switch to public transport for daily commute.

To know for certain if the reduction in amount spent for gas and increase in train/bus tickets is an indicator for switch from car to public transport, we need to know two things:

1. The frequency of buying public transport tickets (to know if the increase in amount of tickets bought is an outlier or evenly distributed)
2. Nr. of commutes made by the customer in the given period of time. This can be obtained by asking the customer a prop question, for ex.:
   “How many commutes did you make this week”

From these two data points, if we map the (ratio of nr of public transport tickets bought in a week/ Nr. of commutes made in a week) , over several weeks. If this ratio over several weeks is constant, then we can say the customer switched from car to public transport, if there is large variance in this ratio, then we can say there were spikes in ticket purchase in certain time frames due to other reasons, and the customer has not necessarily switched from car to public transport.