Adrien Protzel
2/12/15

Mid Term Project

Problem Statements:

I have accumulated records of my financial transactions over the past few years, covering various categories of spending. However, I am unsure about the specifics of my spending habits. I need to analyze these records to understand where my money is going. The goal is to determine whether my spending is responsible or if there are areas where I should consider cutting back. I also want to examine trends over time to see if there are any patterns or significant changes in my spending habits. Additionally, I need to assess the necessity of each spending category to determine if there are any non-essential expenses that can be reduced or eliminated. Based on this analysis, I aim to develop a budget to ensure responsible spending and savings for future financial goals.

OSEMN Process:

Obtain:

Financial transaction records were obtained from a previous data wrangling project.

Scrub:

The data was cleaned by removing bad data in the Transfer category. Additionally, the code was broken into multiple tables to organize the data more effectively.

Explore:

The cleaned data was loaded into VSCode and SQLite to view and manipulate it locally. Using SQLite on a local database allowed for exploratory data analysis (EDA) by running SQL queries to filter, aggregate, and summarize the data.

Model:

Power BI was used to create more interesting visuals. Charts, graphs, and dashboards were developed to illustrate spending patterns, trends over time, and category-wise expenditures.

Interpret:

The visualizations created in Power BI were analyzed to draw meaningful conclusions about spending habits. High spending categories were identified, and the necessity of these expenses was assessed.

Description of data and loading:

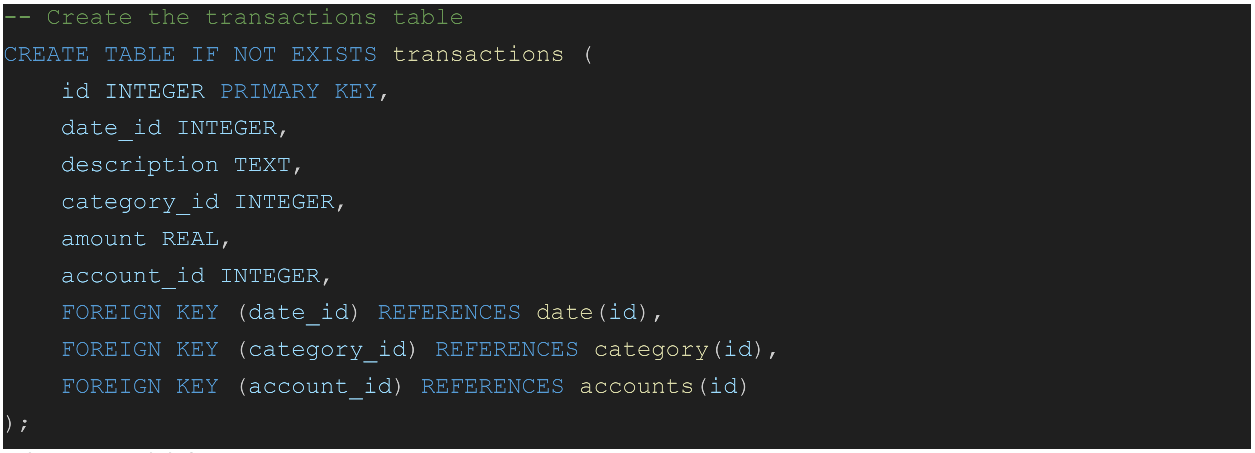
The data is now consolidated into a single CSV file that has undergone initial cleaning and further refinement. Relationship tables were used to handle repeating data such as categories, dates, and accounts, ensuring a more organized and efficient structure. Before cleaning, the data was considered to have a complexity level of 5. The records consist of various financial transactions, including the following fields: date, description, category, amount, bank, card, and type of credit or debit.

To load the data into the database, the CSV file was kept in the same directory as the database file. A new .db file was created, and the SQLite command line tool was used to import the data. The commands .mode csv and .import file.csv file.db were executed to populate the database with the CSV data. After importing, a standard .sql file was used to execute queries and manipulate the database, allowing for efficient data analysis and management.

```
Year,Month,Date,Description,Category,Amount,Type,Bank,Card
2025,January,01/25/2025,apple store,shopping,-2.98,Credit,Bilt,Bilt
2025,January,01/25/2025,xfinity internet,utility,-43.0,Credit,Bilt,Bilt
2025,January,01/21/2025,apple store,shopping,-9.99,Credit,Bilt,Bilt
2025,January,01/14/2025,transfer,transfer,2472.93,Credit,Bilt,Bilt
2025,January,01/11/2025,apple store,shopping,-5.99,Credit,Bilt,Bilt
2025,January,01/07/2025,visible,utility,-25.0,Credit,Bilt,Bilt
```

<Snippet of cleaned data (raw data snippet is provided in data wrangling report)>

```
-- Create the date table
CREATE TABLE IF NOT EXISTS date (
    id INTEGER PRIMARY KEY,
    original_date TEXT,
    year INTEGER,
    month INTEGER,
    day INTEGER,
    month_name TEXT
);
-- Create the category table
CREATE TABLE IF NOT EXISTS category (
    id INTEGER PRIMARY KEY,
    name TEXT
);
-- Create the accounts table
CREATE TABLE IF NOT EXISTS accounts (
    id INTEGER PRIMARY KEY,
    type TEXT,
    bank TEXT,
    card TEXT
);
```



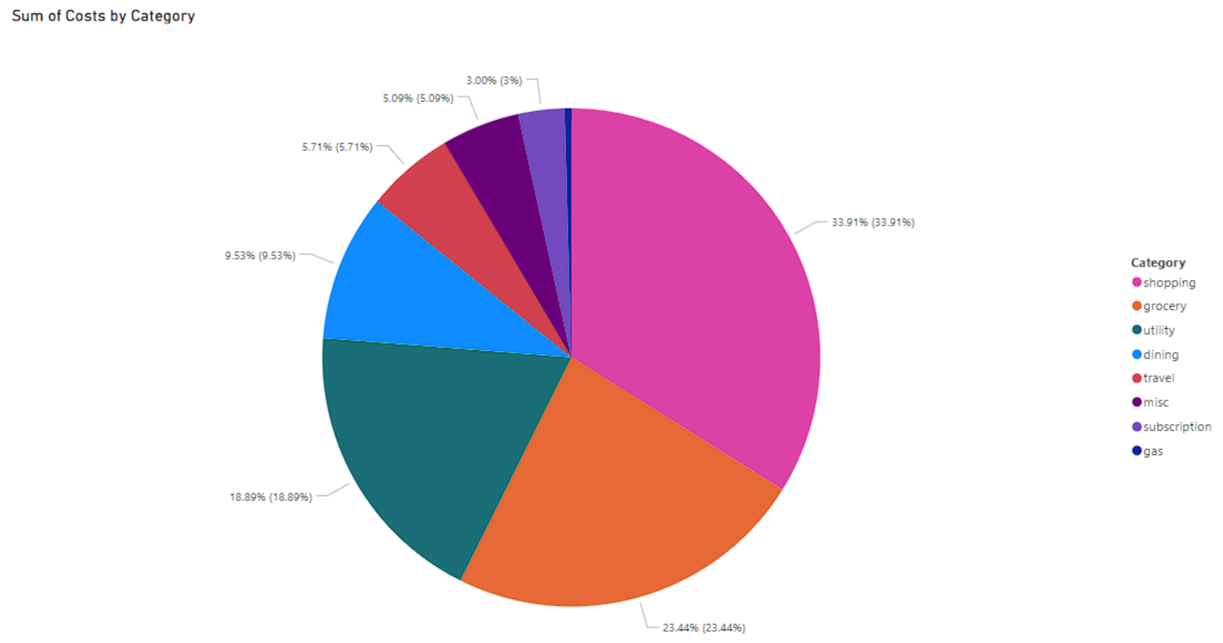
```sql
-- Create the transactions table
CREATE TABLE IF NOT EXISTS transactions (
    id INTEGER PRIMARY KEY,
    date_id INTEGER,
    description TEXT,
    category_id INTEGER,
    amount REAL,
    account_id INTEGER,
    FOREIGN KEY (date_id) REFERENCES date(id),
    FOREIGN KEY (category_id) REFERENCES category(id),
    FOREIGN KEY (account_id) REFERENCES accounts(id)
);
```

<Snippet of SQL code>

Data Questions:

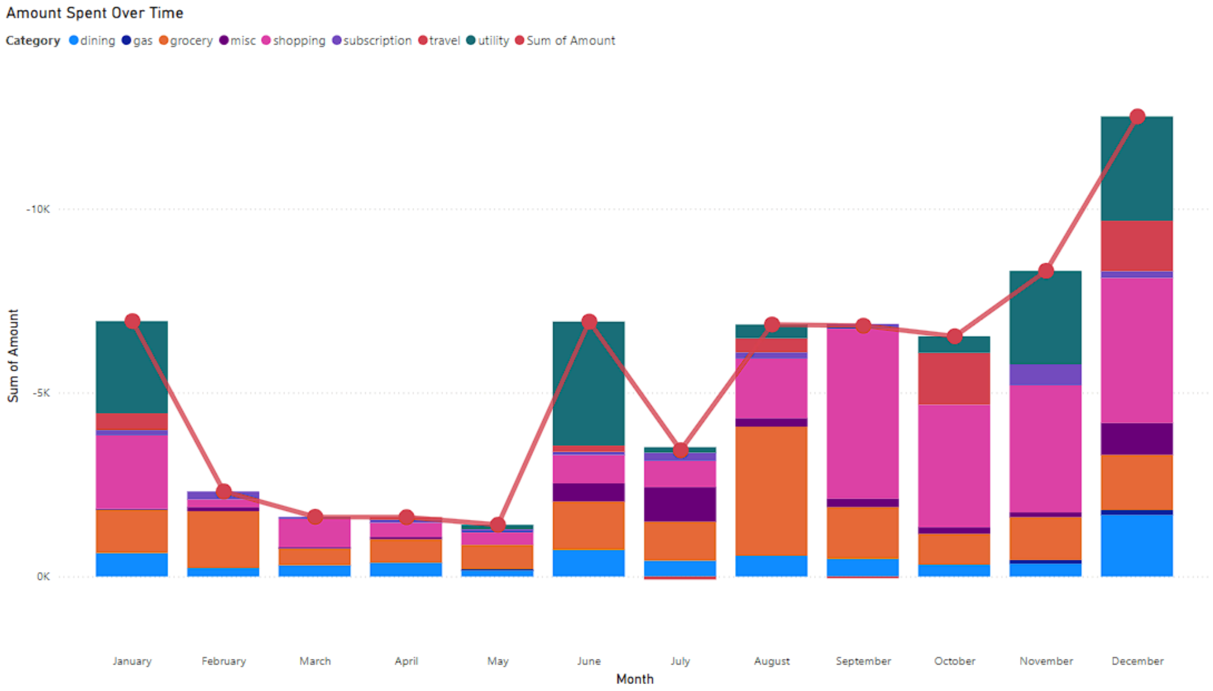
Question 1, Visualization 1:

1. Which category has the highest spending?
    a. I identified the category with the highest total spending by importing the data into Power BI and creating a pie chart. I dragged the "Category" field to the "Legend" well and the "Amount" field to the "Values" well. I excluded the "Transfer" category from the pie chart to focus on other categories. This visualization helped me see which category had the highest spending at a glance.



Question 2, Visualization 2:

2. Are there some months with abnormally high or low spending?
   a. I analyzed the data for any months where spending significantly deviated from the average by using a combination of a stacked bar chart and a line graph in Power BI. For the stacked bar chart, I dragged the "Month" field to the "Axis" well, the "Category" field to the "Legend" well, and the "Amount" field to the "Values" well. For the line graph, I dragged the "Month" field to the "Axis" well and the "Amount" field to the "Values" well. I added annotations to highlight months with significant deviations. This approach helped me visualize fluctuations in spending each month and identify any anomalies.



Question 3, Visualization 3:

3. What specific expenses comprise that high expenditure?
    a. I broke down the category with the highest spending and listed the specific expenses that contributed to the high expenditure by creating a treemap in Power BI. I dragged the "Category" field to the "Group" well, the "Description" field to the "Details" well, and the "Amount" field to the "Values" well. I focused on the largest category to see the specific expenses that contributed to the high expenditure. This visualization provided a clear picture of the major contributors to the highest spending category.

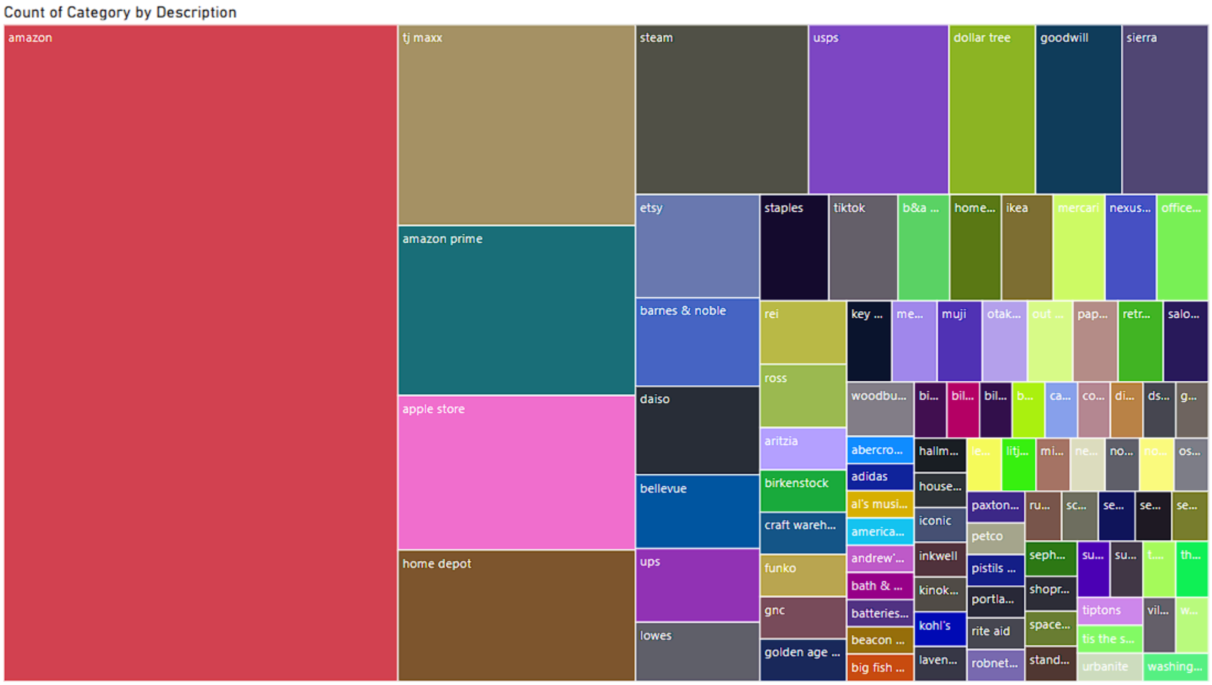


Table Schema:

(temp_transactions is the same structure as transactions table and is later deleted)

<date>

Column Name	Data Type	Description
id	INTEGER	Primary key
original_date	TEXT	Original date string
year	INTEGER	Year part of the date
month	INTEGER	Month part of the date
day	INTEGER	Day part of the date
month_name	TEXT	Name of the month

<category>

Column Name	Data Type	Description
id	INTEGER	Primary key
name	TEXT	Name of the category

<accounts>

Column Name	Data Type	Description
id	INTEGER	Primary key
type	TEXT	Type of account
bank	TEXT	Bank associated with the account
card	TEXT	Card associated with the account

<transactions>

Column Name	Data Type	Description
id	INTEGER	Primary key
date_id	INTEGER	Foreign key referencing date(id)
description	TEXT	Description of the transaction
category_id	INTEGER	Foreign key referencing category(id)
amount	REAL	Amount of the transaction
account_id	INTEGER	Foreign key referencing accounts(id)