**Q1) You are provided with the following datasets:**

* **olist\_order\_items\_dataset.csv: order\_id, order\_item\_id, product\_id, price**
* **olist\_products\_dataset.csv: product\_id,** **product\_category\_name, product\_weight\_g, product\_length\_cm, product\_height\_cm, product\_width\_cm**
* **product\_category\_name\_translation.csv: product\_category\_name, product\_category\_name\_english**

**🡪 Your task is to use these datasets to find association rules among the products:**

1. My first step to completing the task was to first look at each dataset and understand where things were and how they correlated.
2. Once I was acquainted with the datasets, my goal before beginning the Association Rules was to ultimately have a dataframe that consisted of rows with the order\_id and the columns with each product\_category\_name\_english that was purchased during the order.
3. To do this, I first started off by creating a dictionary out of the product\_category\_name\_translation.csv:
   1. temp\_dict\_english\_name = pd.Series(prod\_category.product\_category\_name\_english.values,index=prod\_category.product\_category\_name).to\_dict()
4. With this, I then used this dictionary to compare the values of the “product\_category\_name” column in the olist\_products dataset.
   1. If a value in the cell of the column (the Spanish name) matched a key in the dictionary, I would add the value of that key to a cell in the same row in a new column to the dataset. This is what the outcome looked like:
   2. Table

      Description automatically generated
5. Now that I had the English name of the same row as the product\_id, I repeated this similar process, except the dictionary looked like:
   1. {product\_id: prod\_category\_name\_english}
6. With a dictionary created with the product\_id and English name, I then compared the this dictionary with the product\_id column of the olist\_order\_items dataset and added the English name of the product\_id to a new column.
   1. The outcome looked like this:
   2. Text, table

      Description automatically generated
7. **NOTE**: I did notice there were NaN values. Throughout the analysis, I made sure to inform myself of these and handled them at a later time because they did not cause an issue at the time.
8. Next, I created another dictionary for the order\_id that was of the form:
   1. {order\_id: []}
   2. The empty list would be filled with the products that were purchased during that order.
9. To add to the products to the correct key, I iterated through the rows of olist\_order\_items and compared the order\_id with that of the dictionary. If the key equaled the cell value, I added the product to the value list for the order.
10. Once this dictionary was complete, I turned it into a data frame that looks like this:
    1. Table

       Description automatically generated with low confidence
    2. **NOTE:** Due to all the NaN values, I replaced those values with 0. This way, when running the association rules, I knew there could not be an association between 0🡪 product\_id and knew to discard those rules.
11. Finally, I used the apriori function to run the association rules and manipulated my min\_support, min\_confidence, and min\_lift to fulfill my rules.
12. The following is the list of Association Rules that I found:
    1. Text

       Description automatically generated
    2. Text

       Description automatically generated
    3. Text

       Description automatically generated
    4. And many more

**Q2) You will use nltk to explore the Herman Melville novel Moby Dick. Write the python code for answering the following questions:**

1. **Import the libraries required and Set up Data. Use the link ”https://www.gutenberg.org/files/2701/old/moby10b.txt” to access the ebook.**
2. **Find how many tokens (words and punctuation symbols) are in the text. A token is a linguistic unit such as a word, punctuation mark, or alpha-numeric strings.**
3. **Find how many of the tokens found in 1.2) are unique.**
4. **Find how many tokens are unique after removing stopwords.**
5. **What is the lexical diversity of the given text input? (i.e. ratio of unique tokens to the total number of tokens)**
6. **What percentage of tokens is 'whale' or 'Whale'?**
7. **What are the 20 most frequently occurring (unique) tokens in the text? What is their frequency?**
8. **What tokens have a length of greater than 5 and frequency of more than 150?**
9. **Find the longest word in the text and that word's length.**
10. **What unique words (not punctuation) have a frequency of more than 2000? What is their frequency?**
11. **What is the average number of tokens per sentence?**

\*\*All of my answers for this are answered in the comments of the code. Please refer to that for the answers\*\*