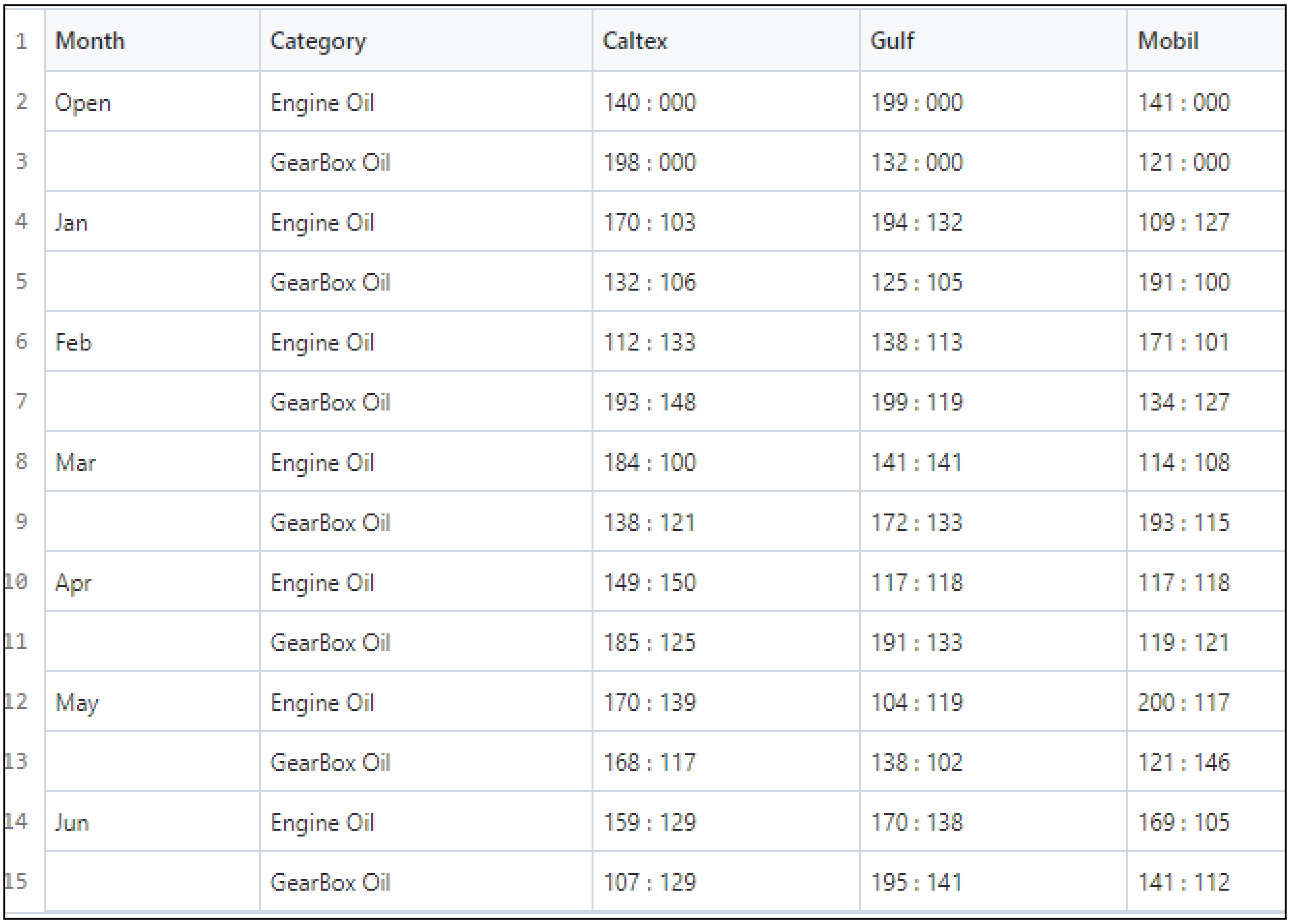
AIM 5001 Module 11 Assignment (100 Points)

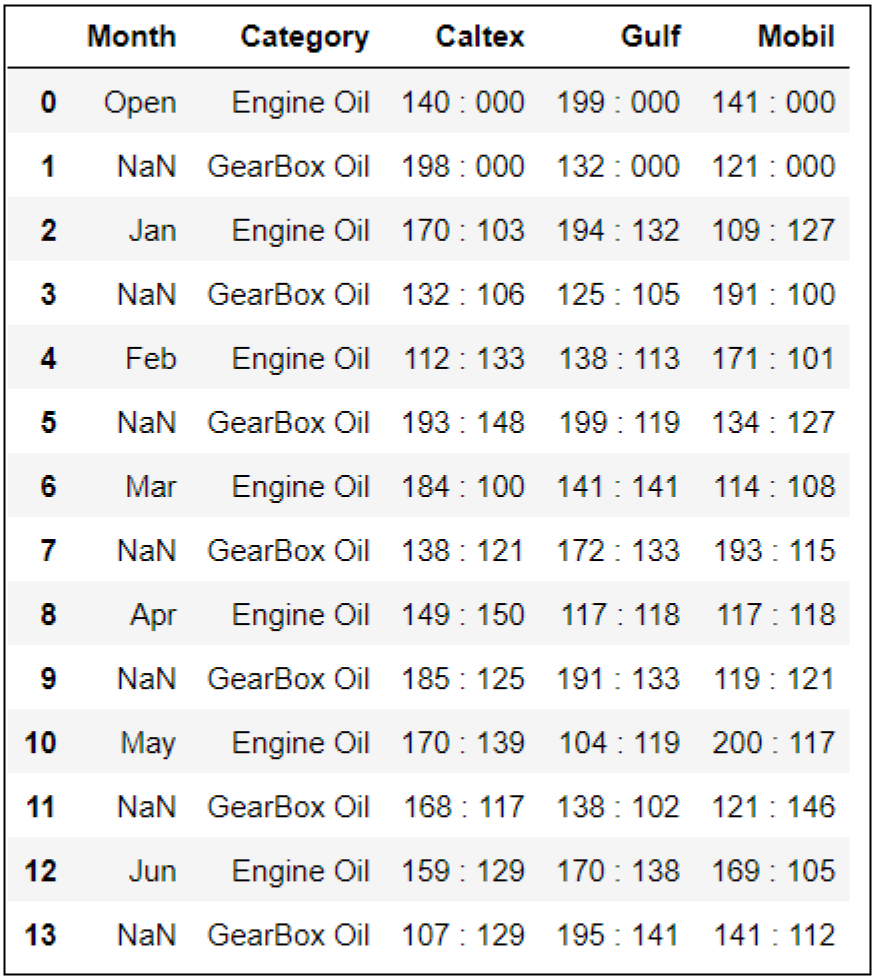
Part 1: Tidying and Reshaping Data



Month	Category	Caltex	Gulf	Mobil
Open	Engine Oil	140 : 000	199 : 000	141 : 000
	GearBox Oil	198 : 000	132 : 000	121 : 000
Jan	Engine Oil	170 : 103	194 : 132	109 : 127
	GearBox Oil	132 : 106	125 : 105	191 : 100
Feb	Engine Oil	112 : 133	138 : 113	171 : 101
	GearBox Oil	193 : 148	199 : 119	134 : 127
Mar	Engine Oil	184 : 100	141 : 141	114 : 108
	GearBox Oil	138 : 121	172 : 133	193 : 115
Apr	Engine Oil	149 : 150	117 : 118	117 : 118
	GearBox Oil	185 : 125	191 : 133	119 : 121
May	Engine Oil	170 : 139	104 : 119	200 : 117
	GearBox Oil	168 : 117	138 : 102	121 : 146
Jun	Engine Oil	159 : 129	170 : 138	169 : 105
	GearBox Oil	107 : 129	195 : 141	141 : 112

The chart above describes purchases and use of marine oil available at a major seaport. There are two types of oil available for use at the seaport: **engine oil** and **gearbox oil**. Each type of oil is provided by three distinct oil manufacturers/suppliers: **Caltex, Gulf**, and **Mobil**. The contents of the 'Caltex', 'Gulf', and 'Mobil' columns contain the number of gallons of oil purchased and consumed (e.g., **purchased : consumed**) for each month, with the '**Open**' indicator shown at the top of the chart telling us how much of each type of oil was on hand at the start of the chronological period (i.e., the ‘purchased’ amounts are the starting inventories for each type/brand of oil). The content of the chart has been re-created within the provided **M11_Data.csv** file. Get started as follows:

- Upload the provided **M11_Data.csv** file to your online AIM 5001 GitHub repository.
- Using the **pd.read_csv()** function, read the **M11_Data.csv** file from your GitHub repository into a Jupyter Notebook WITHOUT removing any empty rows or columns from the content of the file. The content of the resulting dataframe should appear as follows:



	Month	Category	Caltex	Gulf	Mobil
0	Open	Engine Oil	140 : 000	199 : 000	141 : 000
1	NaN	GearBox Oil	198 : 000	132 : 000	121 : 000
2	Jan	Engine Oil	170 : 103	194 : 132	109 : 127
3	NaN	GearBox Oil	132 : 106	125 : 105	191 : 100
4	Feb	Engine Oil	112 : 133	138 : 113	171 : 101
5	NaN	GearBox Oil	193 : 148	199 : 119	134 : 127
6	Mar	Engine Oil	184 : 100	141 : 141	114 : 108
7	NaN	GearBox Oil	138 : 121	172 : 133	193 : 115
8	Apr	Engine Oil	149 : 150	117 : 118	117 : 118
9	NaN	GearBox Oil	185 : 125	191 : 133	119 : 121
10	May	Engine Oil	170 : 139	104 : 119	200 : 117
11	NaN	GearBox Oil	168 : 117	138 : 102	121 : 146
12	Jun	Engine Oil	159 : 129	170 : 138	169 : 105
13	NaN	GearBox Oil	107 : 129	195 : 141	141 : 112

1.1 (**30 Points**): Use your knowledge of combining and reshaping data in Pandas to tidy and transform/reshape the data contained within the dataframe. To get started, think about how you would want the data to appear if it were converted to "long" format, e.g., how would you define a "single observation" for the data shown in the graphic?; How many key values are associated with each data value?; How many columns should your long format structure contain based on the information provided in the graphic shown above?; What would the column headings for the long structure be?; etc. Use your answers to these questions to guide your reshaping/transformational work on the data. **Your reshaping/transformational steps must include converting the above table to a "tidy" long format.** Additional transformational steps (e.g., filling in missing data values, renaming columns, etc.) should be performed as needed to ensure that your data is, in fact, "tidy".

1.2 (15 Points) Using your reshaped/transformed data, perform analysis to answer the following questions:

- What was the amount of oil remaining for each type/brand **at the end of the chronological period**?
- What was the most consumed brand of oil across the two separate categories/types of oil?

1.3 (15 Points) Finally, given your "tidy" long format structure, describe what, if any, changes you would make to the visual presentation of the data if you were then asked to transform your "long" data back into a "wide" format: would you mimic the structure of the graphic shown above? If not, how might you transform your "long" data to "wide" format to make its "wide" presentation easier to understand and work with? Provide an example of your recommendation and explain your rationale for preferring your specific structure.

Part 2: Using Your GroupBy and Data Aggregation Skills

Three Short Coding Challenges

Can you complete these three tasks using no more than 11 lines of code in total?

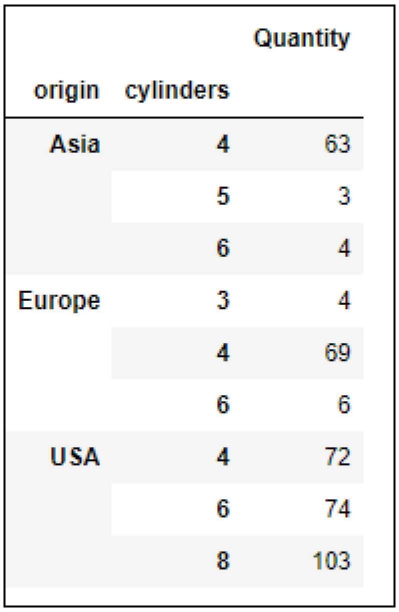
These coding challenges will give you a chance to exercise your **GroupBy/Aggregation/Split-Apply-Combine** skills based on your readings from Chapter 10 of the "**Python for Data Analytics**" textbook. See if you can answer these three questions using **no more than *11 total lines of Python code***.

For each of the three questions we'll be making use of the Auto MPG data set we explored earlier in the semester: https://archive.ics.uci.edu/ml/datasets/Auto+MPGLinks to an external site. Load the data set into your local Python environment using the code snippet shown below and then have a crack at these short coding challenges.

Load the Data

```
import pandas as pd
import numpy as np

# load the data set
auto_df = pd.read_csv("https://raw.githubusercontent.com/jtopor/DAV-5400/master/Week7/auto-mpg.data", delim_whitespace = True, header = None)

# add meaningful column names
auto_df.columns = ['mpg', 'cylinders', 'displacement', 'horsepower',
            'weight', 'acceleration', 'model', 'origin', 'car_name']

# replace '?' in horsepower column with 'NaN'
auto_df.horsepower.replace('?', np.nan, inplace = True)

# convert the column to numeric
auto_df["horsepower"] = pd.to_numeric(auto_df["horsepower"])

# replace origin values using a dict
auto_df.origin.replace({1: 'USA',
                2: 'Asia',
                3: 'Europe'}, inplace = True)

auto_df.head(10)
```

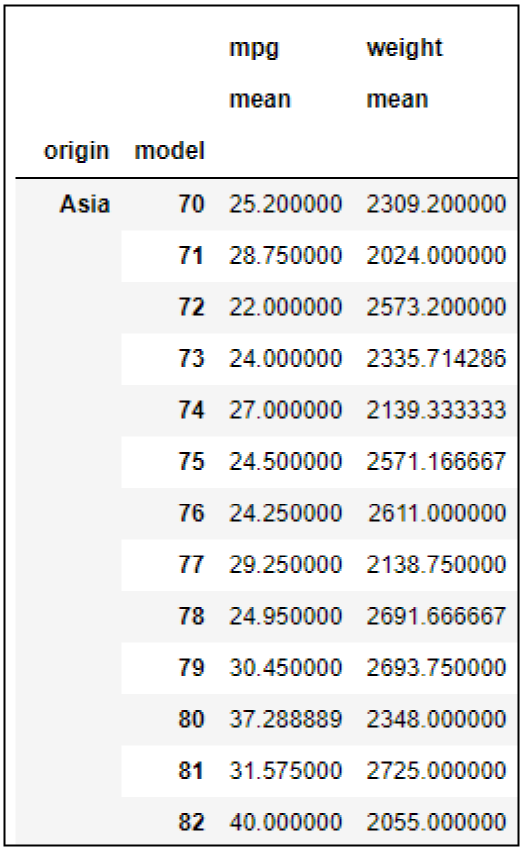
2.1 (**12 Points**): You've been asked to generate a quick report that tells us how many vehicles of each '**Origin**'/'**Cylinder**' grouping within the data set have been manufactured. For each origin/cylinder grouping, your output should include the origin, number of cylinders, and quantity. The output of your report should appear as shown in the graphic below.



		Quantity
origin	cylinders	
Asia	4	63
	5	3
	6	4
Europe	3	4
	4	69
	6	6
USA	4	72
	6	74
	8	103

You are allowed to use **no more than two (2) lines** of Python/Pandas code to generate this report in its entirety (i.e., you **MUST** produce the results for all of the rivers at once) and you **MUST** use Pandas' groupby and/or aggregation functionality to accomplish the task. **Be sure to include a brief narrative explaining how your proposed code would accomplish the task.**

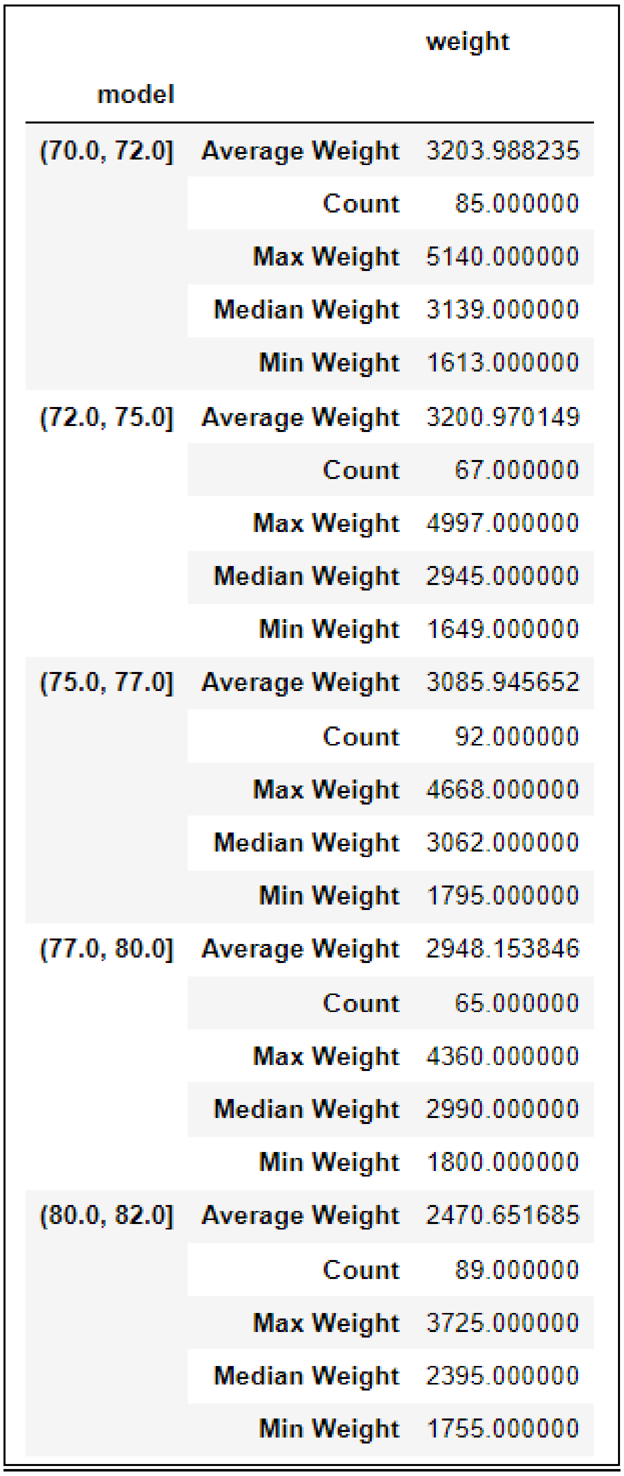
2.2 (14 Points): You've been asked to generate a second report that shows the **average miles per gallon** and **average vehicle weight** for each '**origin**'/'**model**' vehicle grouping within the data set. Your output should be similar to the output shown in the graphic below for 'Asia', and your report should include similar content for each of the origins contained within the data set.



		mpg	weight
		mean	mean
origin	model		
Asia	70	25.200000	2309.200000
	71	28.750000	2024.000000
	72	22.000000	2573.200000
	73	24.000000	2335.714286
	74	27.000000	2139.333333
	75	24.500000	2571.166667
	76	24.250000	2611.000000
	77	29.250000	2138.750000
	78	24.950000	2691.666667
	79	30.450000	2693.750000
	80	37.288889	2348.000000
	81	31.575000	2725.000000
	82	40.000000	2055.000000

You are allowed to use **<u>no more than three (3) lines</u>** of Python/Pandas code and you **MUST** use Pandas' groupby and/or aggregation functionality to accomplish the task. **Be sure to include a brief narrative explaining how your proposed code would accomplish the task.**

2.3 (14 Points) Finally, you've been asked to generate one last report that shows the average weight, count, minimum weight, maximum weight, and median weight of vehicles manufactured during 5 equal length time periods (1970–1972; 1972-1975; 1975-1977; 1977-1980; 1980-1982). The output of your report should appear as shown in the graphic below.



		weight
model		
(70.0, 72.0]	**Average Weight**	3203.988235
	Count	85.000000
	Max Weight	5140.000000
	Median Weight	3139.000000
	Min Weight	1613.000000
(72.0, 75.0]	**Average Weight**	3200.970149
	Count	67.000000
	Max Weight	4997.000000
	Median Weight	2945.000000
	Min Weight	1649.000000
(75.0, 77.0]	**Average Weight**	3085.945652
	Count	92.000000
	Max Weight	4668.000000
	Median Weight	3062.000000
	Min Weight	1795.000000
(77.0, 80.0]	**Average Weight**	2948.153846
	Count	65.000000
	Max Weight	4360.000000
	Median Weight	2990.000000
	Min Weight	1800.000000
(80.0, 82.0]	**Average Weight**	2470.651685
	Count	89.000000
	Max Weight	3725.000000
	Median Weight	2395.000000
	Min Weight	1755.000000

You are allowed to use **no more than six (6) lines** of Python/Pandas code and you **must** use Pandas' groupby and/or aggregation functionality to accomplish the task. **Be sure to include a brief narrative explaining how your proposed code would accomplish the task.**

Save all of your work for this assignment within **a single Jupyter Notebook** and upload / submit it within the provided M11 Assignment Canvas submission portal. Be sure to save your Notebook using the following nomenclature: **first initial_last name_M11_assn**" (e.g., J_Smith_M11_assn).