INF1340

Midterm Submission for INF1340 – UN Dataset

1002147731

KARISHMA SAVANI

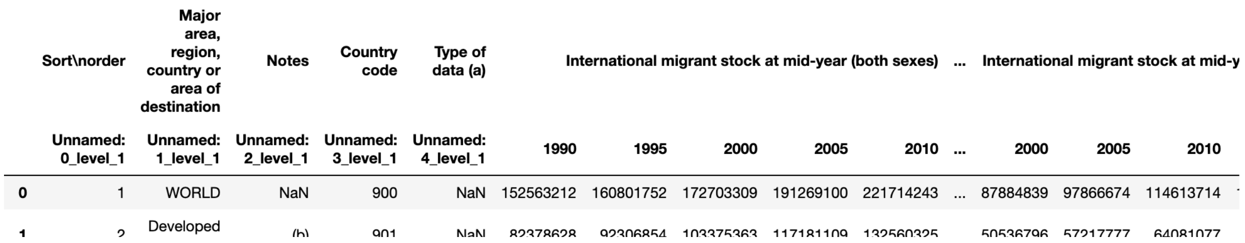
Introduction:

In order to tidy the "United Nations Trends in International Migrant Stock: The 2015 Revision" dataset, I took multiple steps to ensure that all the tidy data principles were satisfied. The overall assignment was very interesting and educative, in really understanding how various data sets are not tidy and hence hard to navigate and understand. Therefore, applying tidy data principles to the datasets is so important.

For the "United Nations Trends in International Migrant Stock: The 2015 Revision" dataset, I used Python to tidy the data keeping in mind the tidy data principles. The dataset has 6 tables of data, that were very untidy. They had multiple values in rows and columns and there was a lot of extra data that was not necessary. Furthermore, one table had many different data types in one table which made the data untidy. The UN data set has very important information for the migrant stock, but due to the untidy nature of it, it makes it hard for visualizations and further cleaning. The data needed to be cleaned to be able to make more sense of the data.

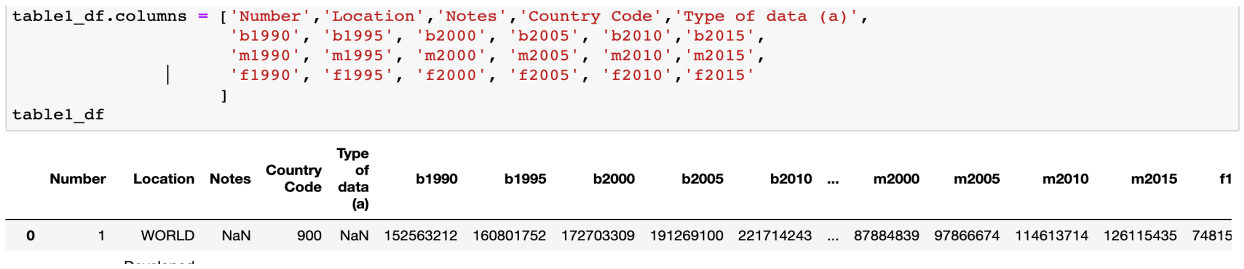
Methods and Results

I decided to import each sheet from the excel file and work on each one separately. I downloaded openxyl so that I can open the excel files and sheets on Jupyter notebooks. I also imported pandas since I used pandas dictionaries to clean this data set. I imported all the tables through the URL from Github so that they can work without the dataset having to be stored on a local computer. I skipped the first 14 rows for all the data sets and made row 15, 16 the headers since they had the titles for each column. I renamed each column to have one combined heading unlike previously where there were 2 rows with some having variables and values. Screenshots below for from this:



	Sort\norder	Major area, region, country or area of destination	Notes	Country code	Type of data (a)	International migrant stock at mid-year (both sexes)					...	International migrant stock at mid-y		
	Unnamed: 0_level_1	Unnamed: 1_level_1	Unnamed: 2_level_1	Unnamed: 3_level_1	Unnamed: 4_level_1	1990	1995	2000	2005	2010	...	2000	2005	2010
0	1	WORLD	NaN	900	NaN	152563212	160801752	172703309	191269100	221714243	...	87884839	97866674	114613714
1	2	Developed	(b)	901	NaN	82378628	92306854	103375363	117181109	132560325		50536796	57217777	64081077

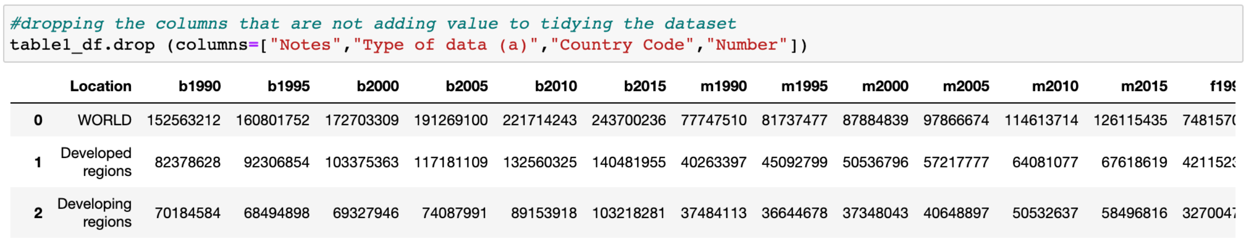
To this:



```
table1_df.columns = ['Number','Location','Notes','Country Code','Type of data (a)',
                     'b1990', 'b1995', 'b2000', 'b2005', 'b2010','b2015',
                     'm1990', 'm1995', 'm2000', 'm2005', 'm2010','m2015',
                     'f1990', 'f1995', 'f2000', 'f2005', 'f2010','f2015'
                    ]
table1_df
```

	Number	Location	Notes	Country Code	Type of data (a)	b1990	b1995	b2000	b2005	b2010	...	m2000	m2005	m2010	m2015	f1
0	1	WORLD	NaN	900	NaN	152563212	160801752	172703309	191269100	221714243	...	87884839	97866674	114613714	126115435	74815

Then, I dropped some columns, namely the notes, type of data, country code and number for each data set that were not adding value and not necessarily needed to tidy the data. The country code could have been used as the unique identifier, but I thought having the location was more meaningful for the data set since we can see the countries where more or less migration was happening.



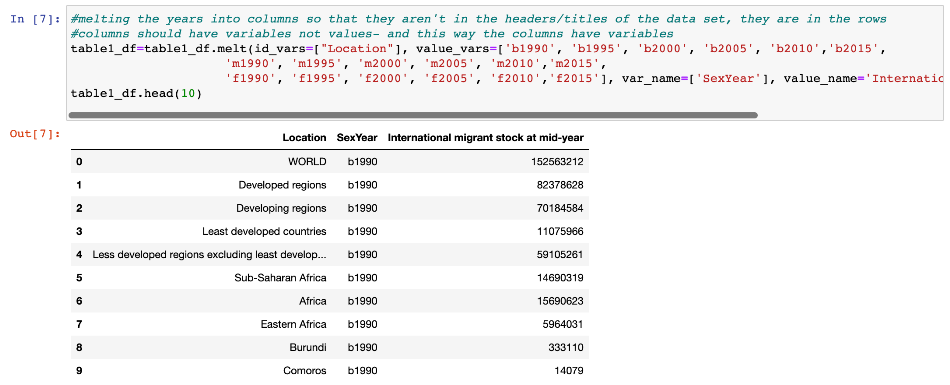
```
#dropping the columns that are not adding value to tidying the dataset
table1_df.drop (columns=["Notes","Type of data (a)","Country Code","Number"])
```

	Location	b1990	b1995	b2000	b2005	b2010	b2015	m1990	m1995	m2000	m2005	m2010	m2015	f19
0	WORLD	152563212	160801752	172703309	191269100	221714243	243700236	77747510	81737477	87884839	97866674	114613714	126115435	748157
1	Developed regions	82378628	92306854	103375363	117181109	132560325	140481955	40263397	45092799	50536796	57217777	64081077	67618619	421152
2	Developing regions	70184584	68494898	69327946	74087991	89153918	103218281	37484113	36644678	37348043	40648897	50532637	58496816	327004

After the preliminary cleaning, I started to focus more on the tidy data principles to clean the data. Listed below are the tidy data principles:

Tidy data principle 1: column names need to informative, variable names and not values
Tidy data principle 2: each column needs to consist of one and only one variable
Tidy data principle 3: variables need to be in cells instead of rows and columns
Tidy data principle 4: each table column needs to have a singular data type
Tidy data principle 5: the same observational unit should not be spread across multiple tables.

I implemented the 1st tidy data principle by using the melt function from the pandas library. I used it to melt the year and gender from wide to long format. I combined the year and sex in one column for now, while melting, since I thought it is easier for me to melt them together at first and then separate them. I named that column SexYear.



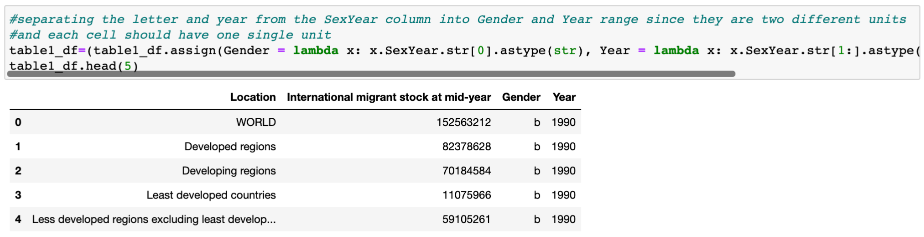
```
In [7]: #melting the years into columns so that they aren't in the headers/titles of the data set, they are in the rows
        #columns should have variables not values- and this way the columns have variables
        table1_df=table1_df.melt(id_vars=["Location"], value_vars=['b1990', 'b1995', 'b2000', 'b2005', 'b2010','b2015',
                                'm1990', 'm1995', 'm2000', 'm2005', 'm2010','m2015',
                                'f1990', 'f1995', 'f2000', 'f2005', 'f2010','f2015'], var_name=['SexYear'], value_name='Internatic
        table1_df.head(10)
```

Out[7]:

	Location	SexYear	International migrant stock at mid-year
0	WORLD	b1990	152563212
1	Developed regions	b1990	82378628
2	Developing regions	b1990	70184584
3	Least developed countries	b1990	11075966
4	Less developed regions excluding least develop...	b1990	59105261
5	Sub-Saharan Africa	b1990	14690319
6	Africa	b1990	15690623
7	Eastern Africa	b1990	5964031
8	Burundi	b1990	333110
9	Comoros	b1990	14079

After this step, I separated the letter from the year below, b is for ‘both sexes’ and 1990 is the year. I had initially named the columns as b1990, m1990, f1990 because the formatting was very different in the excel file and hence there were 2 rows of column titles/ header. Therefore, to make it one row for the header, I named the columns as combined gender and year.

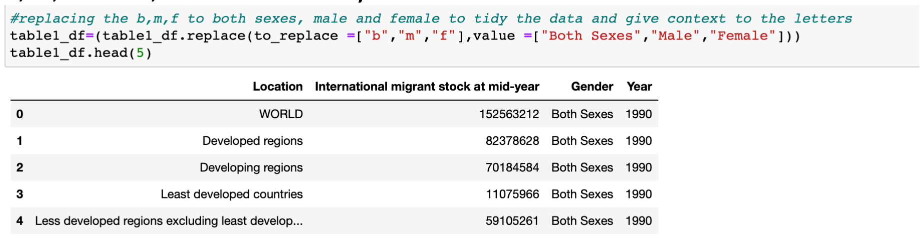
Thereafter, according to the 2nd principle, the columns should only consist of one variable. I separated the letter (indicating the gender) and the year into 2 different columns and named one column as Gender and the other column as Year.



```
#separating the letter and year from the SexYear column into Gender and Year range since they are two different units
#and each cell should have one single unit
table1_df=(table1_df.assign(Gender = lambda x: x.SexYear.str[0].astype(str), Year = lambda x: x.SexYear.str[1:].astype(
table1_df.head(5)
```

	Location	International migrant stock at mid-year	Gender	Year
0	WORLD	152563212	b	1990
1	Developed regions	82378628	b	1990
2	Developing regions	70184584	b	1990
3	Least developed countries	11075966	b	1990
4	Less developed regions excluding least develop...	59105261	b	1990

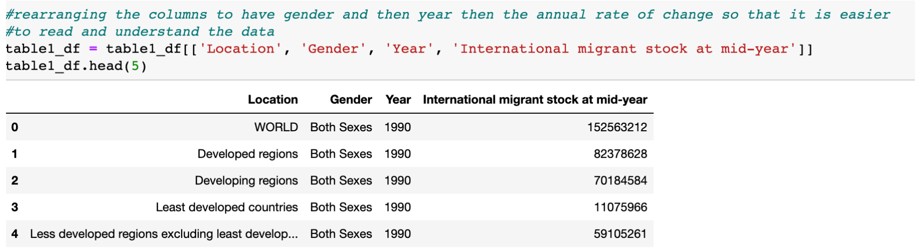
To tidy the data further, I replaced the letters b,m,f with both sexes, male and female respectively. This is also important since, now there is no need for a legend to explain what b,m,f mean, and rather they are named and all clean.



```
#replacing the b,m,f to both sexes, male and female to tidy the data and give context to the letters
table1_df=(table1_df.replace(to_replace =["b","m","f"],value =["Both Sexes","Male","Female"]))
table1_df.head(5)
```

	Location	International migrant stock at mid-year	Gender	Year
0	WORLD	152563212	Both Sexes	1990
1	Developed regions	82378628	Both Sexes	1990
2	Developing regions	70184584	Both Sexes	1990
3	Least developed countries	11075966	Both Sexes	1990
4	Less developed regions excluding least develop...	59105261	Both Sexes	1990

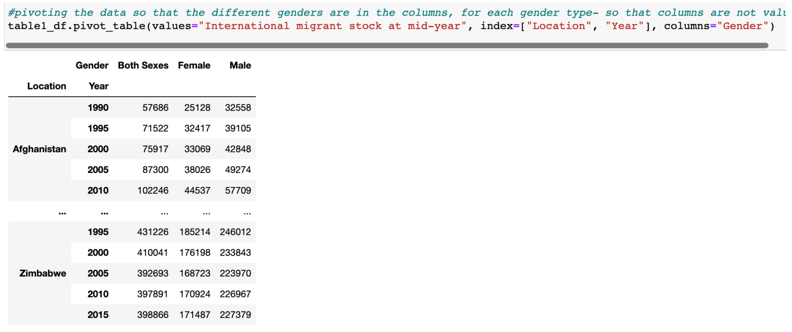
I also rearranged the data so that the order is Location, Gender, Year and international migrant stock or whichever value was being measured across all the tables. This makes it easier when going through the data to understand, which location, which gender, which year and then the result.



```
#rearranging the columns to have gender and then year then the annual rate of change so that it is easier
#to read and understand the data
table1_df = table1_df[['Location', 'Gender', 'Year', 'International migrant stock at mid-year']]
table1_df.head(5)
```

	Location	Gender	Year	International migrant stock at mid-year
0	WORLD	Both Sexes	1990	152563212
1	Developed regions	Both Sexes	1990	82378628
2	Developing regions	Both Sexes	1990	70184584
3	Least developed countries	Both Sexes	1990	11075966
4	Less developed regions excluding least develop...	Both Sexes	1990	59105261

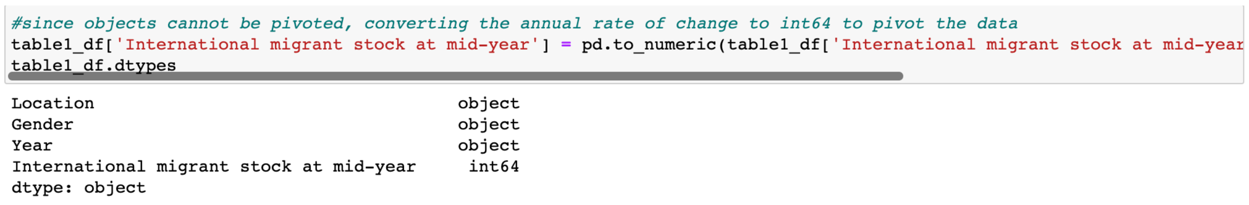
The third principle is about variables need to be in cells, not rows and columns. In order to satisfy this principle, I pivoted the gender to rows so that the data isn't too long whereby you need to finish looking at both sexes to see male or female. Additionally, if I had kept the gender as columns, each year would have 3 different genders associated to it in the long format. By pivoting the gender to the header, the columns only go by the year and each year has its gender data next to it. This isn't against any principles and the variables are in the header and not affecting the data set.



```
#pivoting the data so that the different genders are in the columns, for each gender type- so that columns are not valu
table1_df.pivot_table(values="International migrant stock at mid-year", index=["Location", "Year"], columns="Gender")
```

Location	Year	Both Sexes	Female	Male
Afghanistan	1990	57686	25128	32558
	1995	71522	32417	39105
	2000	75917	33069	42848
	2005	87300	38026	49274
	2010	102246	44537	57709
...	...	...	...	...
Zimbabwe	1995	431226	185214	246012
	2000	410041	176198	233843
	2005	392693	168723	223970
	2010	397891	170924	226967
	2015	398866	171487	227379

Here the years are by the location and the gender is back in wide format, this is much cleaner and easier to navigate. However, to pivot the data, I had to ensure the data types were consistent, since objects cannot be pivoted. So, I changed the data to integer – int64 where there were no decimals and for where there were, I changed them to float64.

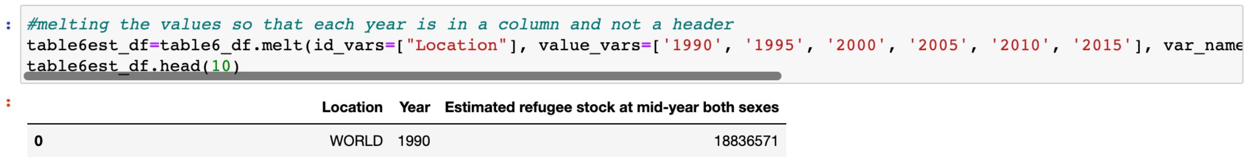


```
#since objects cannot be pivoted, converting the annual rate of change to int64 to pivot the data
table1_df['International migrant stock at mid-year'] = pd.to_numeric(table1_df['International migrant stock at mid-year
table1_df.dtypes
```

```
Location                                   object
Gender                                     object
Year                                       object
International migrant stock at mid-year     int64
dtype: object
```

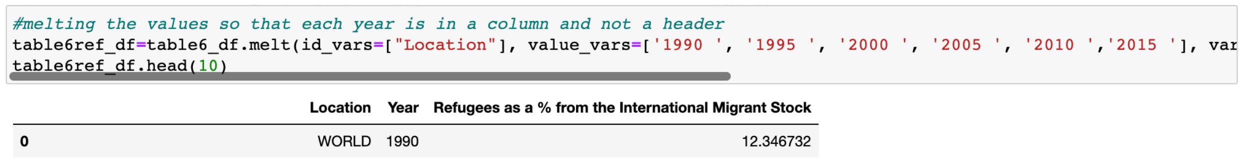
For tables 4 and 6 whereby I did not need to pivot, I left the data as object and only changed the ones that weren't object to object. Table 4 was all female data and Table 6 data was split into 3 tables.

The <u>4th tidy data principle is that each table column needs to have a singular data type</u>. Table 6 had multiple types of data stored in one table such as the Estimated refugee stock at mid-year (both sexes), Refugees as a percentage of the international migrant stock, and Annual rate of change of the refugee stock. The last one wasn't specific years and rather it was year range, so to ensure that there aren't different types of data in one table, I split the data set into the aforementioned 3 data sets.



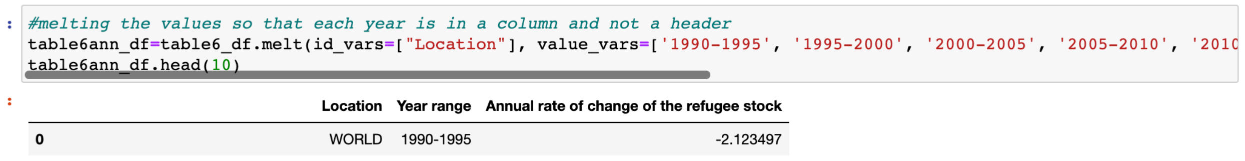
```
#melting the values so that each year is in a column and not a header
table6est_df=table6_df.melt(id_vars=["Location"], value_vars=['1990', '1995', '2000', '2005', '2010', '2015'], var_name
table6est_df.head(10)
```

	Location	Year	Estimated refugee stock at mid-year both sexes
0	WORLD	1990	18836571



```
#melting the values so that each year is in a column and not a header
table6ref_df=table6_df.melt(id_vars=["Location"], value_vars=['1990 ', '1995 ', '2000 ', '2005 ', '2010 ','2015 '], var
table6ref_df.head(10)
```

	Location	Year	Refugees as a % from the International Migrant Stock
0	WORLD	1990	12.346732



```
#melting the values so that each year is in a column and not a header
table6ann_df=table6_df.melt(id_vars=["Location"], value_vars=['1990-1995', '1995-2000', '2000-2005', '2005-2010', '2010
table6ann_df.head(10)
```

	Location	Year range	Annual rate of change of the refugee stock
0	WORLD	1990-1995	-2.123497

Discussions

This assignment was an eye opener for me. I learnt about the different ways to clean data, that I did not know before. Earlier, I would focus on sex and the year and try to make sense of it one by one, and maybe think that it would be better if it were easy to read. Through this assignment, I was able to ensure that the years were melted to a long format and the gender up in the header as a wide format, made it easier to read, understand and for future visualizations. This has helped me understand the UN Dataset as well. Furthermore, if I see any data sets that are untidy going forward, I can clearly and confidently state that this needs to be tidied and how it needs to be tidied according to the principles of tidy data.

When starting out with this assignment, I did not feel very confident with Python and was unsure how I would manage to complete the assignment. However, I learnt a lot while doing the assignment. I learnt to be more patient and be more independent in my search online to seek help regarding the code. Furthermore, using python to clean the data is helpful even after this assignment.

I had some challenges with the data set, whereby I couldn't reduce the decimal numbers to 2 decimals on some of the tables. Additionally, I think it might have been interesting to merge some of the data sets that were going by years and year ranges, but I was unsure whether that would be useful. Secondly, I could have filtered by the countries and the regions (world, less, developed etc…) but I didn't have the capacity nor the knowledge to do so, but would have liked to research it further, to make it tidier.

Conclusion

In conclusion, the assignment was challenging but very useful in understanding and more importantly applying the tidy data principles to datasets. I learnt a lot about the nuances in a data set that can be easily ignored but that can be improved such as putting the year in a long format from wide 'melting it'. Additionally, cleaning up the data to have 1 header from 2 headers really cleaned the visual. There were many challenges while cleaning this dataset, and I was able to resolve some of them, and some are left unresolved. The unresolved ones needed more research and expertise that would come with experience and resolving the challenges I was able to make me feel good too.