INF1340H: Programming for Data Science
Midterm Assignment

Data cleaning in Python with Pandas: Exploring TIDY Data Principles – UN Migrant Dataset

United Nations
Population Division
Department of Economic and Social Affairs

Trends in International Migrant Stock: The 2015 Revision
TABLE OF CONTENTS
POP/DB/MIG/Stock/Rev.2015

Suggested citation : United Nations, Department of Economic and Social Affairs, Population Division (2015). Trends in International Migrant Stock: The 2015 Revision (United Nations database, POP/DB/MIG/Stock/Rev.2015).

Instruction by Professor Shion Guha, Erina Moon, & Priyanka Verma

November 23, 2022

Assignment by Jennifer Puskar (996914629)

1. Introduction

Data cleaning is said to be one of the largest components of most data science roles. Being able to efficiently clean data is an important skillset for a becoming data scientist. The following explores a data cleaning exercise that abides by the TIDY data principles. Python with the use of pandas and numpy software libraries are common data wrangling tools and are used in this assessment to conduct the cleaning.

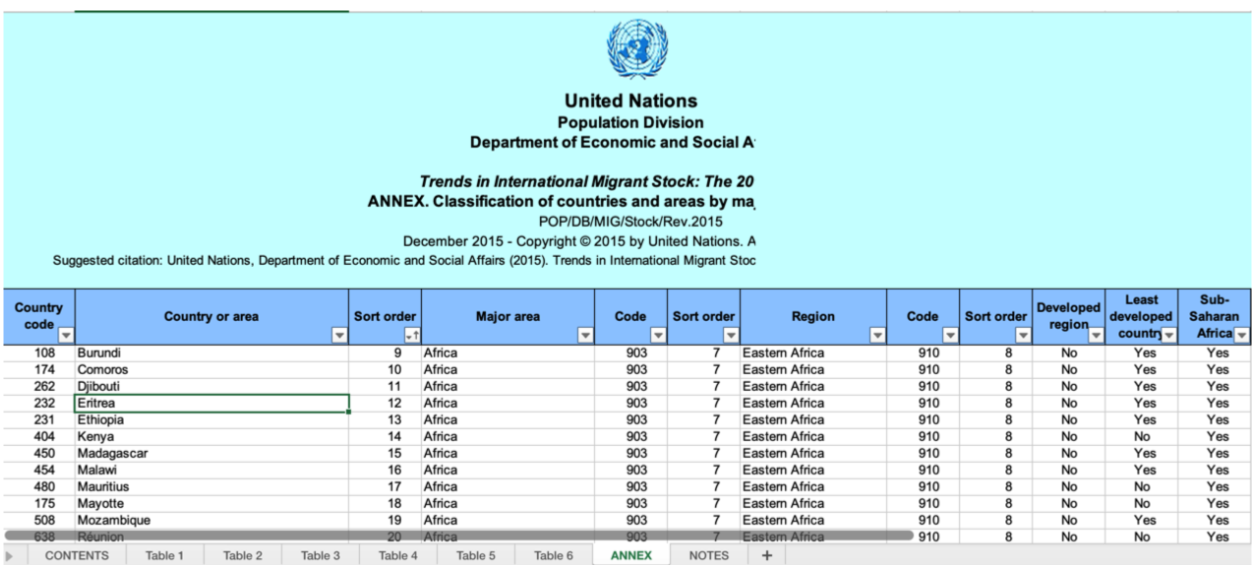
The United Nations regularly reports on a wide range of trends—international migration being one of them. This data is often presented in Microsoft Excel workbooks that are formatted in a way to make it easy for the average person to read them. That doesn't necessarily make these datasets computationally well formatted. This analysis explores what it takes to make the United Nations Trends in International Migrant Stock: The 2015 Revision TIDY from a computational lens. The next phase of this assessment will be visualizing the data, which will rely on how well the data is cleaned.

2. Methods

The UN Migrant Stock data set was cleaned according to the following five TIDY data principles:

1. Column names need to be informative, variable names and not values
2. Each column needs to consist of one and only one variable
3. Variables need to be in cells, not rows and columns
4. Each table column needs to have a singular data type
5. A single observational units must be in 1 table

Prior to loading the data, the Microsoft Excel workbook was explored to better understand the data and the context for the data. Each of the 6 tables were reviewed along with the notes and ANNEX tab (shown below).



United Nations
Population Division
Department of Economic and Social A

Trends in International Migrant Stock: The 20
ANNEX. Classification of countries and areas by ma
POP/DB/MIG/Stock/Rev.2015
December 2015 - Copyright © 2015 by United Nations. A
Suggested citation: United Nations, Department of Economic and Social Affairs (2015). Trends in International Migrant Stoc

Country code	Country or area	Sort order	Major area	Code	Sort order	Region	Code	Sort order	Developed region	Least developed country	Sub-Saharan Africa
108	Burundi	9	Africa	903	7	Eastern Africa	910	8	No	Yes	Yes
174	Comoros	10	Africa	903	7	Eastern Africa	910	8	No	Yes	Yes
262	Djibouti	11	Africa	903	7	Eastern Africa	910	8	No	Yes	Yes
232	Eritrea	12	Africa	903	7	Eastern Africa	910	8	No	Yes	Yes
231	Ethiopia	13	Africa	903	7	Eastern Africa	910	8	No	Yes	Yes
404	Kenya	14	Africa	903	7	Eastern Africa	910	8	No	No	Yes
450	Madagascar	15	Africa	903	7	Eastern Africa	910	8	No	Yes	Yes
454	Malawi	16	Africa	903	7	Eastern Africa	910	8	No	Yes	Yes
480	Mauritius	17	Africa	903	7	Eastern Africa	910	8	No	No	Yes
175	Mayotte	18	Africa	903	7	Eastern Africa	910	8	No	No	Yes
508	Mozambique	19	Africa	903	7	Eastern Africa	910	8	No	Yes	Yes
638	Réunion	20	Africa	903	7	Eastern Africa	910	8	No	No	Yes

CONTENTS | Table 1 | Table 2 | Table 3 | Table 4 | Table 5 | Table 6 | ANNEX | NOTES

Tools and libraries used
Jupyter Notebook was used to work with Python version 3.10. 4 —the current stable release, the fourth maintenance release of Python 3.10, published on March 24, 2022. The panadas and numpy libraries were loaded to begin.

Loading Key Libraries and data

```
In [1]: #Loading pandas, etc.
        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
        import plotly
```

```
In [2]: #Loading excel, Table#1 (2nd Sheet) as Sheet#1 is info
        UNmst_df1 = pd.read_excel('UN_MigrantStockTotal_2015.xlsx', sheet_name = 'Table 1')
```

Exploratory
Data was explored using simple prompts such as len, head, and shape, to understand the size and format of the first table and to ensure the data loaded correctly.

Exploratory

```
In [44]: #Let's see length
         len(UNmst_df1)
Out[44]: 280
```

```
In [45]: #Let's see the top 15 rows
         UNmst_df1.head(5)
Out[45]:
```

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	...	Unnamed: 13	Unnamed: 14	Unnamed: 15	Unna
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	
3	NaN	NaN	NaN	NaN	United Nations	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	
4	NaN	NaN	NaN	NaN	Population Division	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	

5 rows × 23 columns

```
In [46]: UNmst_df1.shape
Out[46]: (280, 23)
```

Cleaning
The data cleaning began with applying the TIDY data principles to Tabe 1 first, and then to tables 2-6. The following describes what was done to the data sets to abide to the TIDY data principles.

The first three principals were looked at first.

Principle 1: Column names need to be informative, variable names and not values
Principle 2: Each column needs to consist of one and only one variable
Principle 3: Variables need to be in cells, not rows and columns

The data set did not begin with useful headers, so the first 15 rows were dropped to get to the main data measures.

```
In [56]: #Dropping rows to get to data. I have decided to drop the header columns for data cleaning (will rename myself).
         UNmst_df1.drop(UNmst_df1.index[0:15], inplace=True)
         UNmst_df1.head(5)
```

Out[56]:

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	...	Unnamed: 13	Unnamed: 14	Unnamed: 15
15	1	WORLD	NaN	900	NaN	152563212	160801752	172703309	191269100	221714243.0	...	87884839	97866674	114613714.0
16	2	Developed regions	(b)	901	NaN	82378628	92306854	103375363	117181109	132560325.0	...	50536796	57217777	64081077.0
17	3	Developing regions	(c)	902	NaN	70184584	68494898	69327946	74087991	89153918.0	...	37348043	40648897	50532637.0
18	4	Least developed countries	(d)	941	NaN	11075966	11711703	10077824	9809634	10018128.0	...	5361902	5383009	5462714.0
19	5	Less developed regions excluding least develop...	NaN	934	NaN	59105261	56778501	59244124	64272611	79130668.0	...	31986141	35265888	45069923.0

5 rows × 23 columns

Columns were assigned named based on review of the Excel workbook.

```
In [58]: #Let's look at the shape of the data now. It has less rows now (266 vs. 280)
         UNmst_df1.shape

Out[58]: (265, 23)
```

```
In [59]: #We need headers.
         UNmst_df1.columns = ['Sort order','Major area, region, country','Notes','Country code','Type of data','T1990','T1995','
                   ]
         UNmst_df1.columns

Out[59]: Index(['Sort order', 'Major area, region, country', 'Notes', 'Country code',
                'Type of data', 'T1990', 'T1995', 'T2000', 'T2005', 'T2010', 'T2015',
                'M1990', 'M1995', 'M2000', 'M2005', 'M2010', 'M2015', 'F1990', 'F1995',
                'F2000', 'F2005', 'F2010', 'F2015'],
               dtype='object')
```

Columns that contained variables needed to be converted into cells. The melt function was very useful for this.

```
In [90]: #Let's transpose columns (years).
         UNmst_df1 = UNmst_df1.melt(id_vars = ['Sort order','Major area, region, country','Notes','Country code','Type of data']
                         var_name = ["SexYear"],
                         value_name = "Migrant Population",)
         UNmst_df1.head(5)
```

Out[90]:

	Sort order	Major area, region, country	Notes	Country code	Type of data	SexYear	Migrant Population
0	1	WORLD	NaN	900	NaN	T1990	152563212
1	2	Developed regions	(b)	901	NaN	T1990	82378628
2	3	Developing regions	(c)	902	NaN	T1990	70184584
3	4	Least developed countries	(d)	941	NaN	T1990	11075966
4	5	Less developed regions excluding least develop...	NaN	934	NaN	T1990	59105261

Multiple variables were stored in one cell and needed to be separated. The lambda function was used to separate sex from year.

```
In [73]: and year need to be separated.
_df1=(UNmst_df1.assign(Sex = lambda x: x.SexYear.str[0].astype(str), Year = lambda x: x.SexYear.str[1:].astype(str)).dr
_df1.head(5)
```

Out[73]:

	Sort order	Major area, region, country	Notes	Country code	Type of data	Migrant Population	Sex	Year
0	1	WORLD	NaN	900	NaN	152563212	T	1990
1	2	Developed regions	(b)	901	NaN	82378628	T	1990
2	3	Developing regions	(c)	902	NaN	70184584	T	1990
3	4	Least developed countries	(d)	941	NaN	11075966	T	1990
4	5	Less developed regions excluding least develop...	NaN	934	NaN	59105261	T	1990

All of the steps listed above were also completed for tables 2-6.

Steps unique to Table 1

To preserve the integrity of the data set, the "Type of Data" column values were also separated into separate columns. This likely isn't needed but was done just in case and to practice using if-else statement. This was only done for Table 1 since the Type of Data values are consistent across the tables and could be joined based on the "Country code".

One other measure that was taken for cleaning Table 1 was the creation of a new column based on NaN values found in the type of data. This helped distinguish which rows were countries, which may be helpful once it comes to data visualization.

```
In [92]: #We need to distinguish between countries and regions. This could help with Pivots of table splits later.
UNmst_df1['Major Area/Region vs. Country?'] = np.where(UNmst_df1['Type of data'].notna(), 'Country', 'Major Area or Reg
UNmst_df1.head(15)
```

Out[92]:

	Sort order	Major area, region, country	Notes	Country code	Type of data	Migrant Population	Sex	Year	Major Area/Region vs. Country?
0	1	WORLD	NaN	900	NaN	152563212	T	1990	Major Area or Region
1	2	Developed regions	(b)	901	NaN	82378628	T	1990	Major Area or Region
2	3	Developing regions	(c)	902	NaN	70184584	T	1990	Major Area or Region
3	4	Least developed countries	(d)	941	NaN	11075966	T	1990	Major Area or Region
4	5	Less developed regions excluding least develop...	NaN	934	NaN	59105261	T	1990	Major Area or Region
5	6	Sub-Saharan Africa	(e)	947	NaN	14690319	T	1990	Major Area or Region
6	7	Africa	NaN	903	NaN	15690623	T	1990	Major Area or Region
7	8	Eastern Africa	NaN	910	NaN	5964031	T	1990	Major Area or Region
8	9	Burundi	NaN	108	B R	333110	T	1990	Country
9	10	Comoros	NaN	174	B	14079	T	1990	Country
10	11	Djibouti	NaN	262	B R	122221	T	1990	Country
11	12	Eritrea	NaN	232	I	11848	T	1990	Country
12	13	Ethiopia	NaN	231	B R	1155390	T	1990	Country
13	14	Kenya	NaN	404	B R	297292	T	1990	Country
14	15	Madagascar	NaN	450	C	23917	T	1990	Country

Once basic cleaning had been conducted on tables 1-5 the last two principles were explored:

Principle 4: Each table column needs to have a singular data type
Principle 5: A single observational units must be in 1 table

Table 6 contained various different measure and annual ranges, so was separated into three different tables.

In [173]:
```
#Df six has a lot of different data and measurements in one table. Let's split into three separate tables.
UNmst_df6RefStock = UNmst_df6[['Major area, region, country, or area of destination','Country code','Year','Sex','Refug
UNmst_df6RefStock.head(5)
```

Out[173]:

	Major area, region, country, or area of destination	Country code	Year	Sex	Refugees as a percentage of the international migrant stock
0	WORLD	900	1990	T	12.346732
1	Developed regions	901	1990	T	2.445494
2	Developing regions	902	1990	T	23.968236
3	Least developed countries	941	1990	T	45.56588
4	Less developed regions excluding least develop...	934	1990	T	19.919743

In [174]:
```
UNmst_df6RefP = UNmst_df6[['Major area, region, country, or area of destination','Country code','RefugeePerTotal','Refu
UNmst_df6RefP.head(5)
```

Out[174]:

	Major area, region, country, or area of destination	Country code	RefugeePerTotal	Refugees as a percentage of the international migrant stock
0	WORLD	900	P1990	12.346732
1	Developed regions	901	P1990	2.445494
2	Developing regions	902	P1990	23.968236
3	Least developed countries	941	P1990	45.56588
4	Less developed regions excluding least develop...	934	P1990	19.919743

In [175]:
```
UNmst_df6RefP=(UNmst_df6RefP.assign(Description = lambda x: x.RefugeePerTotal.str[0].astype(str), Year = lambda x: x.Re
UNmst_df6RefP.drop('Description', axis=1, inplace=True)
UNmst_df6RefP.head(5)
```

Out[175]:

	Major area, region, country, or area of destination	Country code	Refugees as a percentage of the international migrant stock	Year
0	WORLD	900	12.346732	1990
1	Developed regions	901	2.445494	1990
2	Developing regions	902	23.968236	1990
3	Least developed countries	941	45.56588	1990
4	Less developed regions excluding least develop...	934	19.919743	1990

In [176]:
```
UNmst_df6ROC = UNmst_df6[['Major area, region, country, or area of destination','Country code','AnnualROC','Annual rate
UNmst_df6ROC.head(5)
```

Out[176]:

	Major area, region, country, or area of destination	Country code	AnnualROC	Annual rate of change of the refugee stock
0	WORLD	900	R1990-95	-2.123497
1	Developed regions	901	R1990-95	9.388424
2	Developing regions	902	R1990-95	-2.839417
3	Least developed countries	941	R1990-95	-0.680327
4	Less developed regions excluding least develop...	934	R1990-95	-4.3836

In [177]:
```
UNmst_df6ROC=(UNmst_df6ROC.assign(Description = lambda x: x.AnnualROC.str[0].astype(str), Year = lambda x: x.AnnualROC.
UNmst_df6ROC.drop('Description', axis=1, inplace=True)
UNmst_df6ROC.head(5)
```

Out[177]:

	Major area, region, country, or area of destination	Country code	Annual rate of change of the refugee stock	Year
0	WORLD	900	-2.123497	1990-95
1	Developed regions	901	9.388424	1990-95
2	Developing regions	902	-2.839417	1990-95
3	Least developed countries	941	-0.680327	1990-95
4	Less developed regions excluding least develop...	934	-4.3836	1990-95

Final data frames

The following shows the final data frames for tables 1-6, including the three data frames for table 6.

In [178]:
```
UNmst_df1.head(5)
```
Out[178]:

	Sort order	Major area, region, country	Notes	Country code	Migrant Population	Sex	Year	Major Area/Region vs. Country?	B Data (Foreign-born)	C Data (Foreign citizens)	R Data (Refugees)	I Data (Imputation)
0	1	WORLD	NaN	900	152563212	T	1990	Major Area or Region				
1	2	Developed regions	(b)	901	82378628	T	1990	Major Area or Region				
2	3	Developing regions	(c)	902	70184584	T	1990	Major Area or Region				
3	4	Least developed countries	(d)	941	11075966	T	1990	Major Area or Region				
4	5	Less developed regions excluding least develop...	NaN	934	59105261	T	1990	Major Area or Region				

In [179]:
```
UNmst_df2.head(5)
```
Out[179]:

	Sort order	Major area, region, country or area of destination	Notes	Country code	Population at Midyear	Sex	Year
0	1	WORLD	NaN	900	5309667699.0	T	1990
1	2	Developed regions	(b)	901	1144463062.0	T	1990
2	3	Developing regions	(c)	902	4165204637.0	T	1990
3	4	Least developed countries	(d)	941	510057629.0	T	1990
4	5	Less developed regions excluding least develop...	NaN	934	3655147008.0	T	1990

In [180]:
```
UNmst_df3.head(5)
```
Out[180]:

	Sort order	Major area, region, country or area of destination	Notes	Country code	Type of data	International migrant stock as a percentage of the total population	Sex	Year
0	1	WORLD	NaN	900	NaN	2.87331	T	1990
1	2	Developed regions	(b)	901	NaN	7.198015	T	1990
2	3	Developing regions	(c)	902	NaN	1.685021	T	1990
3	4	Least developed countries	(d)	941	NaN	2.171513	T	1990
4	5	Less developed regions excluding least develop...	NaN	934	NaN	1.617042	T	1990

In [181]:
```
UNmst_df4.head(5)
```
Out[181]:

	Sort order	Major area, region, country or area of destination	Notes	Country code	Type of data	Female migrants as a percentage of the international migrant stock	Sex	Year
0	1	WORLD	NaN	900	NaN	49.03915	F	1990
1	2	Developed regions	(b)	901	NaN	51.123977	F	1990
2	3	Developing regions	(c)	902	NaN	46.592099	F	1990
3	4	Least developed countries	(d)	941	NaN	47.261155	F	1990
4	5	Less developed regions excluding least develop...	NaN	934	NaN	46.466684	F	1990

In [182]:
```
UNmst_df5.head(5)
```
Out[182]:

	Sort order	Major area, region, country or area of destination	Notes	Country code	Type of data	Annual rate of change of the migrant stock	Sex	Year
0	1	WORLD	NaN	900	NaN	1.051865	T	1990-95
1	2	Developed regions	(b)	901	NaN	2.275847	T	1990-95
2	3	Developing regions	(c)	902	NaN	-0.487389	T	1990-95
3	4	Least developed countries	(d)	941	NaN	1.118175	T	1990-95
4	5	Less developed regions excluding least develop...	NaN	934	NaN	-0.803244	T	1990-95

In [183]:
```
UNmst_df6RefStock.head(5)
```
Out[183]:

	Major area, region, country, or area of destination	Country code	Year	Sex	Refugees as a percentage of the international migrant stock
0	WORLD	900	1990	T	12.346732
1	Developed regions	901	1990	T	2.445494
2	Developing regions	902	1990	T	23.968236
3	Least developed countries	941	1990	T	45.56588
4	Less developed regions excluding least develop...	934	1990	T	19.919743

```
In [184]: UNmst_df6RefP.head(5)
```

Out[184]:

	Major area, region, country, or area of destination	Country code	Refugees as a percentage of the international migrant stock	Year
0	WORLD	900	12.346732	1990
1	Developed regions	901	2.445494	1990
2	Developing regions	902	23.968236	1990
3	Least developed countries	941	45.56588	1990
4	Less developed regions excluding least develop...	934	19.919743	1990

```
In [185]: UNmst_df6ROC.head(5)
```

Out[185]:

	Major area, region, country, or area of destination	Country code	Annual rate of change of the refugee stock	Year
0	WORLD	900	-2.123497	1990-95
1	Developed regions	901	9.388424	1990-95
2	Developing regions	902	-2.839417	1990-95
3	Least developed countries	941	-0.680327	1990-95
4	Less developed regions excluding least develop...	934	-4.3836	1990-95

3. Discussion

The following outlines how different units of analysis, spitting data sets, and text "fluff" were addressed.

Text 'fluff', descriptive features, and dropping info

Certain text was deleted if it did not provide significant context to the data. An example of this would the first 15 rows of data. After some discussion with colleagues, it seems this analysis may have left more descriptive data measures than other analyses, for example the sort order. The data set was quite small. Had this been larger with more variables, items such as notes may have been dropped to reduce future error.

Units of analysis

An effort was made to keep units of analysis the same. For example, one of the tables stored data in the thousands, where other data sets did not. To avoid error, all measures for that set were multiplied by 1,000 to make the data consistent with the other data frames.

```
In [168]: #Df2 data was in thousands. Let's make consistent with others by multiplying by 1,000.
          UNmst_df2['Population at Midyear'] = UNmst_df2['Population at Midyear'].apply(lambda x: x*1000)
          UNmst_df2.head(5)
```

Out[168]:

	Sort order	Major area, region, country or area of destination	Notes	Country code	Population at Midyear	Sex	Year
0	1	WORLD	NaN	900	5309667699.0	T	1990
1	2	Developed regions	(b)	901	1144463062.0	T	1990
2	3	Developing regions	(c)	902	4165204637.0	T	1990
3	4	Least developed countries	(d)	941	510057629.0	T	1990
4	5	Less developed regions excluding least develop...	NaN	934	3655147008.0	T	1990

Splitting data sets

Admittedly, this was the most difficult part of the analysis decision wise. There are some benefits to developing one large data set, and some to creating smaller tables that could be joined based on a common unique ID. DF6 was split into different datasets, for example, since different measures were being computed.

```
In [173]: #Df six has a lot of different data and measurements in one table. Let's split into three separate tables.
          UNmst_df6RefStock = UNmst_df6[['Major area, region, country, or area of destination','Country code','Year','Sex','Refug
          UNmst_df6RefStock.head(5)
```

Out[173]:

	Major area, region, country, or area of destination	Country code	Year	Sex	Refugees as a percentage of the international migrant stock
0	WORLD	900	1990	T	12.346732
1	Developed regions	901	1990	T	2.445494
2	Developing regions	902	1990	T	23.968236
3	Least developed countries	941	1990	T	45.56588
4	Less developed regions excluding least develop...	934	1990	T	19.919743

```
In [174]: UNmst_df6RefP = UNmst_df6[['Major area, region, country, or area of destination','Country code','RefugeePerTotal','Refu
          UNmst_df6RefP.head(5)
```

Out[174]:

	Major area, region, country, or area of destination	Country code	RefugeePerTotal	Refugees as a percentage of the international migrant stock
0	WORLD	900	P1990	12.346732
1	Developed regions	901	P1990	2.445494
2	Developing regions	902	P1990	23.968236
3	Least developed countries	941	P1990	45.56588
4	Less developed regions excluding least develop...	934	P1990	19.919743

```
In [175]: UNmst_df6RefP=(UNmst_df6RefP.assign(Description = lambda x: x.RefugeePerTotal.str[0].astype(str), Year = lambda x: x.Re
          UNmst_df6RefP.drop('Description', axis=1, inplace=True)
          UNmst_df6RefP.head(5)
```

Out[175]:

	Major area, region, country, or area of destination	Country code	Refugees as a percentage of the international migrant stock	Year
0	WORLD	900	12.346732	1990
1	Developed regions	901	2.445494	1990
2	Developing regions	902	23.968236	1990
3	Least developed countries	941	45.56588	1990
4	Less developed regions excluding least develop...	934	19.919743	1990

What’s missing?

The data frames are technically not TIDY data yet, since the Major area, region, country, or area of destination does not contain unique data. To fix this, that column could be dropped, and the country code could be linked to the Annex table, which could act as an index. This would be helpful for producing data visualizations by country, region, etc.

Lessons Learned

Reduce the urge to delete data without understanding it first

One key lesson learned during this exercise was that the temptation to remove or drop data can be strong, but sometimes it might be best to wait. This occurred with the “Sort Order”, as there was temptation to drop it early on without fully understanding its meaning. It wasn’t until the ANNEX tab was further explored that it became a useful way to organize the countries into their regions, which may come in handy for visualization.

Documentation and descriptive naming conventions make a difference

The data cleaning took place over the course of a number of weeks. Proper documentation helped to ensure it was clear what had been done and where cleaning was left off. In hindsight, the data frames could have been given more descriptive and simpler dames.

Review source materials for context

It was helpful to explore the Excel workbook prior to conducting the data cleaning and throughout the data cleaning. This exercise would have been significantly more challenging had the Excel workbook have not been so easy to read for the average person. It is suspected that most data sets will not come with such clean notes.

Leverage libraries and develop skills building and using functions
Lengthy code made data cleaning redundant, tedious, and prone to error. The experience was better when I was able to use the lambda or anther more concise function.

Outstanding questions and further exploration

The following highlight questions and areas of further exploration with respect to data cleaning:

- When do indexes need to be reset?
- What are the best practices for managing decimal places (keep consistent or maintain precision)?
- When is it best to join vs. separate data sets?
- How can the lambda function be used to build more efficient functions?
- How can one make data cleaning more efficient?
- What else can numpy and pandas do?

4. Conclusion & Next Steps

The cleaning of the UN migrant stock data was more challenging than expected, especially given how clean the Excel Workbook looked on initial inspection. Data cleaning is often touted as being 80% of a data scientists' job—or at least a good portion of it. This exercise certainly proved why that is the case. Though the formatting made the data easier to read for the average person, computationally there was a lot that needed to be addressed—and there are further improvements that could be done to make this a better computational data set. It is helpful to have an excel version of the data/view it in its original form. Certain formatting helped provide context on what the data was about that was not as intuitive when viewed in the kernel.

The pandas and numpy libraries both proved to be useful tools to wrangle with data. Pandas was helpful in the reshaping and pivoting of datasets. It was also helpful in the merging and joining of datasets. Numpy had some useful built-in functions and, and overall seemed quick and powerful when dealing with data structures. Further exploration of the capabilities and functions would enhance the data cleaning skill set of any becoming data scientist.

Online sources, like Stack overflow, often contained helpful tricks or more efficient functions to complete the data cleaning. Once familiarity is achieved with basics such and for loops and if-else statements, it would be wise to delve deeper into lambda and other functions to develop more efficient code.

The next phase of the project will be visualizing the data. It is expected that the visualizations and process will emphasize any errors made in the data cleaning.

References

Guha, S. (2022, November). INF1340H: Programming for Data Science course resources. Retrieved from: https://github.com/shionguha/inf1340-programmingfordatascience-fa22/blob/main/notebooks/tidydata.ipynb

2022, August 9. “How to make a new column based on nan conditional?” Stack overflow. Retrieved from: https://stackoverflow.com/questions/73285591/how-to-make-a-new-column-based-on-nan-conditional

2022, September 1. “Pandas Vs NumPy: What’s The Difference? [2022]”. InterviewBit. Retrieved from https://www.interviewbit.com/blog/pandas-vs-numpy/