UN International Migrant Stock Data 2015 Revision: Data Cleaning

Yifei Chen (Yvette 1002271187)
Date: Nov 16, 2022

Overview

For this project, I applied Tidy Data Principles to clean the United Nations dataset "Trends in International Migrant Stock: The 2015 Revision" which contains 6 tables of data. In this report, I will discuss the steps and decisions I made during the cleaning process.

Four of the five Tidy Data Principles apply to the dataset and below is a summary of the violations discovered:

Tidy Principle 1 Violations: Column headers should be variable names, not values	Table 1, Table 2, Table 3, Table 4, Table 5, Table 6
Tidy Principle 2 Violations: Multiple variables should not be stored in one column	Table 1, Table 2, Table 3, Table 5, Table 6
Tidy Principle 3 Violations: Variables should not be stored in both rows and columns	Table 1, Table 2, Table 3, Table 5, Table 6
Tidy Principle 4 Violations: Multiple types of observational units should not be stored in the same table	Table 6
Tidy Principle 5 Violations: A single observational unit should not be stored in multiple tables	None

Noises occurred

1. The missing values ".."
2. The wrong data types for the variables.
3. The treatment for Principle 1 violation violates Principle 2; the treatment for Principle 2 violation violates Principle 3. The dataset was finally tidy after addressing Principles 1, 2, and 3.

Please see Appendix for the result screenshots for each table.

Table of Content

Detailed steps

Part 1: Set up

All 6 tables were set up using the same process. I will discuss the steps in detail using Table 1 as an example.

Import libraries and datasets

I used Google Colab for this project and the raw dataset is stored on my Google Drive. First I imported relevant libraries such as Pandas and Numpy, then read the dataset from Google Drive using the pd.read_excel() function. The dataset has multiple sheets, each sheet was imported to Colab separately, by specifying "sheet_name = 'table name".

The following screenshot shows the import details using Table 1 as an example.

```
#import libraries
import pandas as pd
import numpy as np

#import raw data and table 1
df_intms = pd.read_excel('/content/drive/MyDrive/Colab Notebooks/UN_MigrantStockTotal_2015.xlsx', sheet_name='Table 1')
```

High-level information

Once a table has been imported to Collab, I used a few functions to check the high-level details. This gives me a better understanding of the data and helps me plan for cleaning.

1. **head(20)** shows the first 20 rows of the sheet. The table preview contains many NaNs in the first 20 rows. Based on this given information, my next step would be to create a data frame that selects relevant rows and columns.

```
#look at first 20 items in the dataset
df_intms.head(20)
```

2. **info()** provides the following insights:
    a. Does the dataset have any null values? The answer is yes as we can see the non-null counts are less than the total row numbers.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 280 entries, 0 to 279
Data columns (total 23 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Unnamed: 0   266 non-null    object
 1   Unnamed: 1   266 non-null    object
 2   Unnamed: 2   27 non-null     object
 3   Unnamed: 3   266 non-null    object
 4   Unnamed: 4   241 non-null    object
 5   Unnamed: 5   267 non-null    object
 6   Unnamed: 6   266 non-null    object
 7   Unnamed: 7   266 non-null    object
 8   Unnamed: 8   266 non-null    object
 9   Unnamed: 9   266 non-null    float64
 10  Unnamed: 10  266 non-null    float64
 11  Unnamed: 11  267 non-null    object
 12  Unnamed: 12  266 non-null    object
 13  Unnamed: 13  266 non-null    object
 14  Unnamed: 14  266 non-null    object
 15  Unnamed: 15  266 non-null    float64
 16  Unnamed: 16  266 non-null    float64
 17  Unnamed: 17  267 non-null    object
 18  Unnamed: 18  266 non-null    object
 19  Unnamed: 19  266 non-null    object
 20  Unnamed: 20  266 non-null    object
 21  Unnamed: 21  266 non-null    float64
 22  Unnamed: 22  266 non-null    float64
dtypes: float64(6), object(17)
memory usage: 50.4+ KB
```

b. “object” and “float” data types are detected for columns which do not reflect the expected data types: integers, for international migration stock.

3. **describe()** provides a basic description of the variables in the dataset with numeric values, such as count, mean, standard deviation, etc.

```
[36] #Get high level pandas functionalities
     df_intms.info()
     df_intms.describe()
```

4. **shape** provides the dimension of the dataset, Table 1 has 280 rows and 23 columns.

```
[37] # get the size of the dataframe
     df_intms.shape
```

Part 2: Missing Values Treatment

Missing values in this dataset are presented using “..”, which is a problem because Python does not read this symbol as null or NaN.

To properly treat the missing values, I first defined the classifications of the missing values as Missing Completely at Random (MCAR) and Missing Data that's Not Missing at Random (NMAR).

Data missing completely at random (MCAR) does not contain any observable patterns to explain the missingness in the dataset. We cannot interpret whether the value was missing due to specific circumstances or not. There are a few ways to handle MCAR, for example, we could ignore it, replace it with NaN, delete the variable, correct it, or impute a reasonable data value for the missing values.

One sheet in the dataset, “Notes”, provides context to explain some of the missing values (NMAR). Countries that gained independence from 1990 to 2015 do not have data for the years prior to their independence. For example, South Sudan only has data starting from 2005 as they were previously part of Sudan. The Note explains that Sudan’s observation contains data for both Sudan and South Sudan from 1995 to 2005, which indicates that the missing data for South Sudan before 2005 are not random.

Based on the classifications, my solution is to replace “..” with “NaN”. I did not consider the other treatments because keeping the current symbol “..” for missing values can cause problems during data analysis. Deleting all observations with missing values removes useful variables that are valid and insightful. Imputing missing values with reasonable values (based on the mean or median of a dataset) is not appropriate because there is no connection among each observation (each country’s data across the years from 1990 to 2015), every country has their own migration trends. In addition, there is the practice to replace missing values with “0”. However, in the dataset, there are cells with a value of 0. If I replace missing values with “0”, we will not be able to differentiate whether the value was 0 or originally null. There is a difference between 0 and null, null does not mean 0.

I used two ways to replace “..” with NaN for this project on different tables.

1. Replace during data cleaning

```
#replace placeholder missing data with NAN
ers_df2.replace('..', np.nan, inplace=True)
ers_df2
```

- This approach is used for Tables 1, 2, 3, and 5 when I discovered the Panda pivot table automatically drops observations with only NaN values. To prevent this, I decided to replace the “..” with NaN after I created the Pivot table.
- I will discuss this in detail in the “Principle 3 Violations” section below.

2. Replace missing values when reading the dataset

```
#import raw data and table 6
missing_values = ['..'] # define missing values
ers_df = pd.read_excel('/content/drive/MyDrive/Colab Notebooks/UN_MigrantStockTotal_2015.xlsx',
                       sheet_name='Table 6',na_values = missing_values)
ers_df
```

- For tables that do not require pivoting during the cleaning process, such as Tables 4 and 6, the issue related to the pivot table automatically dropping NaN values does not apply anymore. Thus, I decided to replace the missing values as the first step in the process.

Part 3: Select specific rows or columns or both

I used the **iloc** function to select required rows and columns as we don't need all of them in the tidy version. Using Table 1 as an example, rows before line 15 are descriptive information that does not contain values. In a new data frame, I only want to include rows starting from line 15 and until the last observation, row 280.

```
#get the columns from 1 to 23 and rows from position 14 to 280
df_intms1= df_intms[df_intms.columns[0:23]].iloc[15:280]
df_intms1.head()
```

	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	Unnamed: 16	Unname
15	1	WORLD	NaN	900	NaN	152563212	160801752	172703309	191269100	221714243.0	...	87884839	97866674	114613714.0	126115435.0	748157
16	2	Developed regions	(b)	901	NaN	82378628	92306854	103375363	117181109	132560325.0	...	50536796	57217777	64081077.0	67618619.0	421152
17	3	Developing regions	(c)	902	NaN	70184584	68494898	69327946	74087991	89153918.0	...	37348043	40648897	50532637.0	58496816.0	327004
18	4	Least developed countries	(d)	941	NaN	11075966	11711703	10077824	9809634	10018128.0	...	5361902	5383009	5462714.0	6463217.0	52362
19	5	Less developed regions excluding least develop...	NaN	934	NaN	59105261	56778501	59244124	64272611	79130668.0	...	31986141	35265888	45069923.0	52033599.0	274642

Column names were excluded when creating the new data frame, so I renamed them accordingly.

```
#rename columns
df_intms1.columns=['ID','Area','Notes','Country Code','Type of data (a)','Both sexes 1990','Both sexes 1995',
                   'Both sexes 2000', 'Both sexes 2005','Both sexes 2010',
                   'Both sexes 2015', 'Male 1990', 'Male 1995', 'Male 2000', 'Male 2005',
                   'Male 2010', 'Male 2015', 'Female 1990', 'Female 1995', 'Female 2000',
                   'Female 2005', 'Female 2010', 'Female 2015']
df_intms1.head()
```

	ID	Area	Notes	Country Code	Type of data (a)	Both sexes 1990	Both sexes 1995	Both sexes 2000	Both sexes 2005	Both sexes 2010	...	Male 2000	Male 2005	Male 2010	Male 2015	Female 1990	Female 1995	Fe
15	1	WORLD	NaN	900	NaN	152563212	160801752	172703309	191269100	221714243.0	...	87884839	97866674	114613714.0	126115435.0	74815702	79064275	848
16	2	Developed regions	(b)	901	NaN	82378628	92306854	103375363	117181109	132560325.0	...	50536796	57217777	64081077.0	67618619.0	42115231	47214055	528
17	3	Developing regions	(c)	902	NaN	70184584	68494898	69327946	74087991	89153918.0	...	37348043	40648897	50532637.0	58496816.0	32700471	31850220	319
18	4	Least developed countries	(d)	941	NaN	11075966	11711703	10077824	9809634	10018128.0	...	5361902	5383009	5462714.0	6463217.0	5236216	5573685	47
19	5	Less developed regions excluding least develop...	NaN	934	NaN	59105261	56778501	59244124	64272611	79130668.0	...	31986141	35265888	45069923.0	52033599.0	27464255	26276535	272

I used the drop function, **drop()** to remove columns that are not needed for data analysis, including "Notes", "Country Code", and "Types of data".

```
# Remove unnecessary columns
df_intms1 = df_intms1.drop(['Notes','Country Code','Type of data (a)'], axis=1) #Axis 1 will act on all the COLUMNS in each ROW
df_intms1.head()
```

	ID	Area	Both sexes 1990	Both sexes 1995	Both sexes 2000	Both sexes 2005	Both sexes 2010	Both sexes 2015	Male 1990	Male 1995	Male 2000	Male 2005	Male 2010	Male 2015	Female 1990	Female 1995
15	1	WORLD	152563212	160801752	172703309	191269100	221714243.0	243700236.0	77747510	81737477	87884839	97866674	114613714.0	126115435.0	74815702	79064275
16	2	Developed regions	82378628	92306854	103375363	117181109	132560325.0	140481955.0	40263397	45092799	50536796	57217777	64081077.0	67618619.0	42115231	47214055
17	3	Developing regions	70184584	68494898	69327946	74087991	89153918.0	103218281.0	37484113	36644678	37348043	40648897	50532637.0	58496816.0	32700471	31850220
18	4	Least developed countries	11075966	11711703	10077824	9809634	10018128.0	11951316.0	5843107	6142712	5361902	5383009	5462714.0	6463217.0	5236216	5573685
19	5	Less developed regions excluding least develop...	59105261	56778501	59244124	64272611	79130668.0	91262036.0	31641006	30501966	31986141	35265888	45069923.0	52033599.0	27464255	26276535

The dataset is now ready for Tidy Data examinations.

Part 4: Tidy Data Examinations

Principle 1 violations:

Currently, we have “Year” as column headers and the corresponding migration values for different sexes as values. This violates Principle 1: column headers are values, not variable names.

All tables violate this principle, I am only using Table 1 as an example to show my process:

	ID	Area	Both sexes 1990	Both sexes 1995	Both sexes 2000	Both sexes 2005	Both sexes 2010	Both sexes 2015	Male 1990	Male 1995	Male 2000	Male 2005	Male 2010	Male 2015	Female 1990	Female 1995
15	1	WORLD	152563212	160801752	172703309	191269100	221714243.0	243700236.0	77747510	81737477	87884839	97866674	114613714.0	126115435.0	74815702	79064275
16	2	Developed regions	82378628	92306854	103375363	117181109	132560325.0	140481955.0	40263397	45092799	50536796	57217777	64081077.0	67618619.0	42115231	47214055
17	3	Developing regions	70184584	68494898	69327946	74087991	89153918.0	103218281.0	37484113	36644678	37348043	40648897	50532637.0	58496816.0	32700471	31850220
18	4	Least developed countries	11075966	11711703	10077824	9809634	10018128.0	11951316.0	5843107	6142712	5361902	5383009	5462714.0	6463217.0	5236216	5573685
19	5	Less developed regions excluding least develop...	59105261	56778501	59244124	64272611	79130668.0	91262036.0	31641006	30501966	31986141	35265888	45069923.0	52033599.0	27464255	26276535

(Table 1: International migrant stock)

I used **pd.melt** to unpivot selected values to the row axis. Year columns are converted into row values.

```
#unpivot measured variables to row axis
intms_melt = pd.melt(df_intms1, id_vars =["ID", "Area"] , var_name = "SexYear", value_name = "International migrant stock")
intms_melt.head()
```

	ID	Area	SexYear	International migrant stock
0	1	WORLD	Both sexes 1990	152563212
1	2	Developed regions	Both sexes 1990	82378628
2	3	Developing regions	Both sexes 1990	70184584
3	4	Least developed countries	Both sexes 1990	11075966
4	5	Less developed regions excluding least develop...	Both sexes 1990	59105261

Principle 2 violations:
This violation appears in Tables 1, 2, 3, and 5.

After we treated the table based on Principle 1, we see that the table now violates Principle 2, that is, each column needs to consist of one and only one variable. Here, the column "SexYear" contains data for both sex and year. We will split the column into two columns. This violation is believed to be a noise as it occurred during the cleaning process.

Note that Principle 2 violation does not apply to Table 4 because it only contains data for female stock, and Table 6 because the data includes both sexes.

```
[13] #Sex and Year share one cell, which should be prevented according to tidy data 2
     intms_melt1=(intms_melt.assign(Sex = lambda x: x.SexYear.str[:-4].astype(str), Year = lambda x: x.SexYear.str[-4:].astype(str)).drop("SexYear",axis=1))

     intms_melt1
```

	ID	Area	International migrant stock	Sex	Year
0	1	WORLD	152563212	Both sexes	1990
1	2	Developed regions	82378628	Both sexes	1990
2	3	Developing regions	70184584	Both sexes	1990
3	4	Least developed countries	11075966	Both sexes	1990
4	5	Less developed regions excluding least develop...	59105261	Both sexes	1990
...	...	...	...	...	...
4765	261	Samoa	2460.0	Female	2015
4766	262	Tokelau	254.0	Female	2015
4767	263	Tonga	2604.0	Female	2015
4768	264	Tuvalu	63.0	Female	2015
4769	265	Wallis and Futuna Islands	1411.0	Female	2015

4770 rows × 5 columns

(Table 1: International migrant stock)

To make the table more readable, I then reordered the rows based on the ID and Year columns. I organized the data of the same area together, and the year is in ascending order. This styling step is helpful for the upcoming steps where I make the table into a pivot table.

```
#reorder the columns so ID and Year columns come first
intms_melt2= intms_melt1.sort_values(by =['ID', 'Year'] )
intms_melt2.head()
```

	ID	Area	International migrant stock	Sex	Year
0	1	WORLD	152563212	Both sexes	1990
1590	1	WORLD	77747510	Male	1990
3180	1	WORLD	74815702	Female	1990
265	1	WORLD	160801752	Both sexes	1995
1855	1	WORLD	81737477	Male	1995

(Table 1: International migrant stock)

Principle 3 violations:
This violation appears in Tables 1, 2, 3, and 5.

After treating the dataset using Principles 1 and 2, we see that the "Sex" column has three types of variables stored – "Both sexes", "Female", and "Male". This violates Principle 3, the variables are stored in both rows and columns. Each of the three "Sex" variables should have its own column. We will solve the problem by making a pivot table.

This is also considered as a noise that occurred during cleaning.

```
] #make a pivot table by splitting the Sex column into three columns: "Both sexes", "female", and "male"
  intms_tidy = intms_melt2.pivot_table(
    index = ['ID','Year', 'Area'],
    columns = 'Sex',
    values = 'International migrant stock', aggfunc='first').reset_index()

  intms_tidy1 = intms_tidy.set_index(['ID','Area','Year'])

  intms_tidy1
```

		Sex	Both sexes	Female	Male
ID	Area	Year			
1	WORLD	1990	152563212	74815702	77747510
		1995	160801752	79064275	81737477
		2000	172703309	84818470	87884839
		2005	191269100	93402426	97866674
		2010	221714243.0	107100529.0	114613714.0
...	...	...	...	...	...
265	Wallis and Futuna Islands	1995	1680	821	859
		2000	2015	997	1018
		2005	2365	1171	1194
		2010	2776.0	1375.0	1401.0
		2015	2849.0	1411.0	1438.0

1590 rows × 3 columns

(Table 1: International migrant stock)

This table is almost tidy, but as my last step, I will replace the ".." (null values) with NaN. Why am I doing this step so late in the cleaning process? Because during the cleanup, I found out that Panda would automatically remove observations with "NaN" values from the pivot table. For example, if I replaced ".." with "NaN" before creating the pivot table, South Sudan's data from 1990 to 2005 will be removed entirely.

ID	Area	Year	Both sexes	Female	Male
21	Rwanda	2015	441525.0	221508.0	220017.0
22	Seychelles	1990	3721.0	1517.0	2204.0
22	Seychelles	1995	5148.0	2127.0	3021.0
22	Seychelles	2000	6574.0	2737.0	3837.0
22	Seychelles	2005	8997.0	3177.0	5820.0
22	Seychelles	2010	11420.0	3616.0	7804.0
22	Seychelles	2015	12791.0	3837.0	8954.0
23	Somalia	1990	478294.0	234093.0	244201.0
23	Somalia	1995	19527.0	9245.0	10282.0
23	Somalia	2000	20087.0	9465.0	10622.0
23	Somalia	2005	20670.0	9623.0	11047.0
23	Somalia	2010	23995.0	11081.0	12914.0
23	Somalia	2015	25291.0	11531.0	13760.0
24	South Sudan	1990	NaN	NaN	NaN
24	South Sudan	1995	NaN	NaN	NaN
24	South Sudan	2000	NaN	NaN	NaN
24	South Sudan	2005	NaN	NaN	NaN
24	South Sudan	2010	257905.0	125212.0	132693.0
24	South Sudan	2015	824122.0	403173.0	420949.0
25	Uganda	1990	558307.0	268223.0	290084.0
25	Uganda	1995	634620.0	313260.0	321360.0
25	Uganda	2000	634703.0	320378.0	314325.0
25	Uganda	2005	652968.0	330547.0	322421.0
25	Uganda	2010	529160.0	267514.0	261646.0
25	Uganda	2015	749471.0	375033.0	374438.0

(Replace ".." with NaN after creating the pivot table)

126 to 150 of 1575 entries Filter

ID	Area	Year	Both sexes	Female	Male
21	Rwanda	2015	441525.0	221508.0	220017.0
22	Seychelles	1990	3721.0	1517.0	2204.0
22	Seychelles	1995	5148.0	2127.0	3021.0
22	Seychelles	2000	6574.0	2737.0	3837.0
22	Seychelles	2005	8997.0	3177.0	5820.0
22	Seychelles	2010	11420.0	3616.0	7804.0
22	Seychelles	2015	12791.0	3837.0	8954.0
23	Somalia	1990	478294.0	234093.0	244201.0
23	Somalia	1995	19527.0	9245.0	10282.0
23	Somalia	2000	20087.0	9465.0	10622.0
23	Somalia	2005	20670.0	9623.0	11047.0
23	Somalia	2010	23995.0	11081.0	12914.0
23	Somalia	2015	25291.0	11531.0	13760.0
24	South Sudan	2010	257905.0	125212.0	132693.0
24	South Sudan	2015	824122.0	403173.0	420949.0
25	Uganda	1990	558307.0	268223.0	290084.0
25	Uganda	1995	634620.0	313260.0	321360.0
25	Uganda	2000	634703.0	320378.0	314325.0
25	Uganda	2005	652968.0	330547.0	322421.0
25	Uganda	2010	529160.0	267514.0	261646.0
25	Uganda	2015	749471.0	375033.0	374438.0
26	United Republic of Tanzania	1990	574025.0	290635.0	283390.0
26	United Republic of Tanzania	1995	1106043.0	558954.0	547089.0
26	United Republic of Tanzania	2000	928180.0	457218.0	470962.0
26	United Republic of Tanzania	2005	770846.0	284072.0	486774.0

(Replace ".." with NaN before creating the pivot table. Note that South Sudan's observations from 1990 to 2005 are eliminated)

The data is tidy now.

```
#python is not reading ".." as null, which can raise a problem during data analysis. I will change ".." to NaN
intms_tidy1.replace('..', np.nan, inplace=True)
intms_tidy1
```

			Sex	Both sexes	Female	Male
ID		Area	Year			
1		WORLD	1990	152563212.0	74815702.0	77747510.0
			1995	160801752.0	79064275.0	81737477.0
			2000	172703309.0	84818470.0	87884839.0
			2005	191269100.0	93402426.0	97866674.0
			2010	221714243.0	107100529.0	114613714.0
...		...	...	...	...	...
265		Wallis and Futuna Islands	1995	1680.0	821.0	859.0
			2000	2015.0	997.0	1018.0
			2005	2365.0	1171.0	1194.0
			2010	2776.0	1375.0	1401.0
			2015	2849.0	1411.0	1438.0

1590 rows × 3 columns

Principle 4 violations:

There are multiple types of data stored in Table 6 which violates Principle 4, multiple types of observational units should not be stored in the same table. I will discuss how I split Table 6 into 3 separate tables.

After I set Table 6 ready for cleaning (refer to the process discussed above), this is the preview of the dataset. There are three types of observational units: Estimated refugee stock, Refugee as a percentage of the international migrant stock, and Annual rate of change of the refugee stock.

	ID	Area	Both sexes 1990	Both sexes 1995	Both sexes 2000	Both sexes 2005	Both sexes 2010	Both sexes 2015	Percentage 1990	Percentage 1995	Percentage 2000	Percentage 2005	Percentage 2010	Percentage 2015	Refugee ROC 1990–1995	Refugee ROC 1995–2000	Refugee ROC 2000–2005	Refugee ROC 2005–2010	Refugee ROC 2010–2015
15	1	WORLD	18836571	17853840.0	15827803.0	13276733.0	15370755.0	19577474.0	12.346732	11.103013	9.164736	6.941389	6.932687	8.033424	-2.123497	-3.837069	-5.557223	-0.025089	2.947267
16	2	Developed regions	2014564	3609670.0	2997256.0	2361229.0	2046917.0	1954224.0	2.445494	3.910511	2.899391	2.015025	1.544140	1.391085	9.388424	-5.983348	-7.277379	-5.323293	-2.087656
17	3	Developing regions	16822007	14244170.0	12830547.0	10915504.0	13323838.0	17623250.0	23.968236	20.795958	18.507035	14.733162	14.944759	17.073768	-2.839417	-2.332154	-4.561	0.285195	2.663652
18	4	Least developed countries	5048391	5160131.0	3047488.0	2363782.0	1957884.0	3443582.0	45.56588	44.041961	30.221557	24.082430	19.533425	28.801534	-0.680327	-7.531747	-4.541459	-4.187109	7.766031
19	5	Less developed regions excluding least develop...	11773616	9084039.0	9783059.0	8551722.0	11365954.0	14179668.0	19.919743	15.999082	16.513130	13.305391	14.363526	15.537313	-4.3836	0.632489	-4.319731	1.530456	1.571047

For each observational unit, I created a new data frame consisting of the related columns, then cleaned the tables one by one.

1. Estimated refugee stock

```
#create dataframe with related columns
est_rs = ers_df2[['ID','Area','Both sexes 1990','Both sexes 1995',
                  'Both sexes 2000', 'Both sexes 2005','Both sexes 2010','Both sexes 2015']]

#year variables should not be column headers
est_rs = pd.melt(est_rs, id_vars =["ID", "Area"] , var_name = "SexYear", value_name = "Refugee Stock (Both Sexes)")
est_rs
```

	ID	Area	SexYear	Refugee Stock (Both Sexes)
0	1	WORLD	Both sexes 1990	18836571
1	2	Developed regions	Both sexes 1990	2014564
2	3	Developing regions	Both sexes 1990	16822007
3	4	Least developed countries	Both sexes 1990	5048391
4	5	Less developed regions excluding least develop...	Both sexes 1990	11773616
...	...	...	...	...
1585	261	Samoa	Both sexes 2015	0.0
1586	262	Tokelau	Both sexes 2015	0.0
1587	263	Tonga	Both sexes 2015	0.0
1588	264	Tuvalu	Both sexes 2015	0.0
1589	265	Wallis and Futuna Islands	Both sexes 2015	0.0

1590 rows × 4 columns

After I selected relevant columns and have a new data frame for this observational unit, I followed the same process mentioned in “Principle 1 Violations” and “Principle 2 Violations” sections to convert column variables into rows using pd.melt().,then split “Sex” and “Year” into separate columns.

Once Sex and Year are split into two columns, I dropped the “Sex” column because this observational unit only has data for “Both sexes”.

Then I reordered and set the index for the table.

```
#split
est_rs=(est_rs.assign(Sex = lambda x: x.SexYear.str[:-4].astype(str), Year = lambda x: x.SexYear.str[-4:].astype(str)).drop("SexYear",axis=1))

#we don't need the "sex" column
est_rs=est_rs.drop(['Sex'], axis=1)

#Order dataset based on ID and Year
est_rs= est_rs.sort_values(by =['ID', 'Year'] )

#Store the unique ID-Area in est_rs3
est_rs= est_rs.set_index(['ID','Area','Year'])

est_rs
```

The data is tidy now:

			Refugee Stock (Both Sexes)
ID	Area	Year	
1	WORLD	1990	18836571
		1995	17853840.0
		2000	15827803.0
		2005	13276733.0
		2010	15370755.0
...	...	...	...
265	Wallis and Futuna Islands	1995	0.0
		2000	0.0
		2005	0.0
		2010	0.0
		2015	0.0

1590 rows × 1 columns

ID	Area	Year	Refugee Stock (Both Sexes)
1	WORLD	1990	18836571
1	WORLD	1995	17853840.0
1	WORLD	2000	15827803.0
1	WORLD	2005	13276733.0
1	WORLD	2010	15370755.0
1	WORLD	2015	19577474.0
2	Developed regions	1990	2014564
2	Developed regions	1995	3609670.0
2	Developed regions	2000	2997256.0
2	Developed regions	2005	2361229.0
2	Developed regions	2010	2046917.0
2	Developed regions	2015	1954224.0
3	Developing regions	1990	16822007
3	Developing regions	1995	14244170.0
3	Developing regions	2000	12830547.0
3	Developing regions	2005	10915504.0
3	Developing regions	2010	13323838.0
3	Developing regions	2015	17623250.0
4	Least developed countries	1990	5048391
4	Least developed countries	1995	5160131.0
4	Least developed countries	2000	3047488.0
4	Least developed countries	2005	2363782.0
4	Least developed countries	2010	1957884.0
4	Least developed countries	2015	3443582.0
5	Less developed regions excluding least developed countries	1990	11773616

Note that a way to treat the null value is to replace it with 0, which would not work for this dataset because some observations contain both 0 and ".." in the cells. If we replace ".." with "0", we cannot differentiate the cell value that was by default 0 and cells that were previously null. 0 and null indicate different meanings.

2. **"**Refugees as a percentage of the international migrant stock" table is created using the exact process as discussed in the previous Estimated refugee stock section.

```
#Create a dataframe with columns for refugee as a percentage of the international migrant stock
per_rs = ers_df2[['ID','Area','Percentage 1990', 'Percentage 1995', 'Percentage 2000', 'Percentage 2005',
                  'Percentage 2010', 'Percentage 2015']]

#column headers "year" are values not variable names. Violate principle 2. Unpivot to move year columns into rows.
per_rs = pd.melt(per_rs, id_vars =["ID", "Area"] , var_name = "PYear", value_name = "Refugee Percentage")

#Column has 2 variables - percentage and year, let's split that
per_rs=(per_rs.assign(Percentage = lambda x: x.PYear.str[:-4].astype(str), Year = lambda x: x.PYear.str[-4:].astype(str)).drop("PYear",axis=1))

#we don't need the "percentage" column because the whole observational unit refers to this.
per_rs=per_rs.drop(['Percentage'], axis=1)

#Order dataset based on ID and Year
per_rs= per_rs.sort_values(by =['ID', 'Year'] )

#Store the unique ID-Area in est_rs3
per_rs= per_rs.set_index(['ID','Area','Year'])

per_rs
```

			Refugee Percentage
ID	**Area**	**Year**	
1	**WORLD**	**1990**	12.346732
		1995	11.103013
		2000	9.164736
		2005	6.941389
		2010	6.932687
...	**...**	**...**	...
265	**Wallis and Futuna Islands**	**1995**	0.0
		2000	0.0
		2005	0.0
		2010	0.0
		2015	0.0

1590 rows × 1 columns

1 to 25 of 1590 entries Filter

ID	Area	Year	Refugee Percentage
1	WORLD	1990	12.346732054907182
1	WORLD	1995	11.103013355227622
1	WORLD	2000	9.164736386145329
1	WORLD	2005	6.941389382811964
1	WORLD	2010	6.932687224789613
1	WORLD	2015	8.033424309035137
2	Developed regions	1990	2.445493508340537
2	Developed regions	1995	3.9105113472938857
2	Developed regions	2000	2.8993910280150 6
2	Developed regions	2005	2.0150253058280923
2	Developed regions	2010	1.544139998148013
2	Developed regions	2015	1.3910854244589634
3	Developing regions	1990	23.968236386497637
3	Developing regions	1995	20.795957678482857
3	Developing regions	2000	18.507034666799445
3	Developing regions	2005	14.73316235555638
3	Developing regions	2010	14.944758793438556
3	Developing regions	2015	17.07376816321907
4	Least developed countries	1990	45.56587979247468
4	Least developed countries	1995	44.0419610226591
4	Least developed countries	2000	30.221556866037503
4	Least developed countries	2005	24.082429819324368
4	Least developed countries	2010	19.533424787369366
4	Least developed countries	2015	28.80153426096571
5	Less developed regions excluding least developed countries	1990	19.919742846580103

3. “Annual rate of change of the refugee stock” table follows the same process as well.

```
#Create data frame for this observational unit
roc_rs = ers_df2[['ID','Area','Refugee ROC 1990-1995', 'Refugee ROC 1995-2000', 'Refugee ROC 2000-2005',
                  'Refugee ROC 2005-2010', 'Refugee ROC 2010-2015']]

#column headers "year" are values not variable names. Violate principle 2. Unpivot to move year columns into rows.
roc_rs = pd.melt(roc_rs, id_vars =["ID", "Area"] , var_name = "rocYear", value_name = "Refugee ROC")

#violation 3 that column contains more than one variable, refugee rate of change and year, we will split them into two separate columns
roc_rs=(roc_rs.assign(placeholder = lambda x: x.rocYear.str[:-4].astype(str), Year = lambda x: x.rocYear.str[-4:].astype(str)).drop("rocYear",axis=1))

#Refugee rate of change column is not useful as the whole observational unit is about the Rate of change, we may remove this column
roc_rs=roc_rs.drop(['placeholder'], axis=1)

#Order dataset based on ID and Year
roc_rs= roc_rs.sort_values(by =['ID', 'Year'] )

#Store the unique ID-Area in est_rs3
roc_rs= roc_rs.set_index(['ID','Area','Year'])

roc_rs
```

			Refugee ROC
ID	Area	Year	
1	WORLD	1995	-2.123497
		2000	-3.837069
		2005	-5.557223
		2010	-0.025089
		2015	2.947267
...	...	...	...
265	Wallis and Futuna Islands	1995	NaN
		2000	NaN
		2005	NaN
		2010	NaN
		2015	NaN

1325 rows × 1 columns

1 to 25 of 1325 entries Filter

ID	Area	Year	Refugee ROC
1	WORLD	1995	-2.123497443993485
1	WORLD	2000	-3.8370685429335762
1	WORLD	2005	-5.557223280786607
1	WORLD	2010	-0.025088975388118342
1	WORLD	2015	2.9472674315195495
2	Developed regions	1995	9.388423966885876
2	Developed regions	2000	-5.983348404074283
2	Developed regions	2005	-7.2773794111666055
2	Developed regions	2010	-5.32329268351312
2	Developed regions	2015	-2.087655930965729
3	Developing regions	1995	-2.8394168886795867
3	Developing regions	2000	-2.3321542674714575
3	Developing regions	2005	-4.561000330888476
3	Developing regions	2010	0.2851952093583670 6
3	Developing regions	2015	2.6636520830871824
4	Least developed countries	1995	-0.6803269439341384
4	Least developed countries	2000	-7.531747316485774
4	Least developed countries	2005	-4.541459035747553
4	Least developed countries	2010	-4.187108632338556
4	Least developed countries	2015	7.766031376440223
5	Less developed regions excluding least developed countries	1995	-4.383599796705438
5	Less developed regions excluding least developed countries	2000	0.6324889456790788
5	Less developed regions excluding least developed countries	2005	-4.319730641824886
5	Less developed regions excluding least developed countries	2010	1.5304560436841692
5	Less developed regions excluding least developed countries	2015	1.5710467623816085

Show 25 per page

Limitations

1. Major area, region, country and area of destination are stored in the same column. I find it difficult to determine whether or not this violates principle 4 - each type of observational unit forms a table, because these string values are associated with the same type of measurement, e.g. migration stock or migration percentage. In this project, I chose to keep the major area column untouched because after splitting, it might violate Tidy Data Principle 5.

2. Data Types: Table 1 and Estimated refugee stock Table derived from Table 6 contain migration stock data. The correct data type should be an integer. I chose to keep NaN during data cleaning and as a result, I was unable to change these data types to integers using **.astype(int).** I could only change them from "object" to "float".

   All other tables have the correct data type: float. I used **.dtypes** to check the datatype and changed them accordingly.

```
#check the datatype
roc_rs.dtypes
```

```
Refugee ROC    object
dtype: object
```

```
#change datatype to float
roc_rs=roc_rs.astype('float')
roc_rs.info()
```

```
<class 'pandas.core.frame.DataFrame'>
MultiIndex: 1325 entries, (1, 'WORLD', '1995') to (265, 'Wallis and Futuna Islands', '2015')
Data columns (total 1 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Refugee ROC  890 non-null    float64
dtypes: float64(1)
memory usage: 37.5+ KB
```

Conclusion

This report discussed the data cleaning process for the UN International Migrant Stock Data 2015 Revision database, using the Tidy Data Principles.

- I started by previewing the dataset using head(20) and getting some statistical insights using methods like info(), describe(), shape().
- I moved forward to select relevant rows and columns using iloc and drop functions.
- The missing values and incorrect data types are noises during data cleaning. I was successful at dealing with the missing values by replacing ".." with "NaN". Whereas my attempt to correct the data types to integers was unsuccessful due to NaN values, converting to float was not an issue.
- I was not sure if I should treat the "major area" column based on the Tidy Data Principle. Ideally, the observational units should be divided into different tables based on their category.

- The dataset was cleaned referencing four Tidy Data Principles and the project outcome includes 5 cleaned tables generated from Tables 1 to 5 and 3 new tables derived from Table 6.

Appendix

Table 1: International migrant stock at mid-year by sex and by major area, region, country or area, 1990-2015

		Sex	Both sexes	Female	Male
ID	Area	Year			
1	WORLD	1990	152563212.0	74815702.0	77747510.0
		1995	160801752.0	79064275.0	81737477.0
		2000	172703309.0	84818470.0	87884839.0
		2005	191269100.0	93402426.0	97866674.0
		2010	221714243.0	107100529.0	114613714.0
...	...	...	...	...	...
265	Wallis and Futuna Islands	1995	1680.0	821.0	859.0
		2000	2015.0	997.0	1018.0
		2005	2365.0	1171.0	1194.0
		2010	2776.0	1375.0	1401.0
		2015	2849.0	1411.0	1438.0

1 to 25 of 1590 entries Filter

ID	Area	Year	Both sexes	Female	Male
1	WORLD	1990	152563212.0	74815702.0	77747510.0
1	WORLD	1995	160801752.0	79064275.0	81737477.0
1	WORLD	2000	172703309.0	84818470.0	87884839.0
1	WORLD	2005	191269100.0	93402426.0	97866674.0
1	WORLD	2010	221714243.0	107100529.0	114613714.0
1	WORLD	2015	243700236.0	117584801.0	126115435.0
2	Developed regions	1990	82378628.0	42115231.0	40263397.0
2	Developed regions	1995	92306854.0	47214055.0	45092799.0
2	Developed regions	2000	103375363.0	52838567.0	50536796.0
2	Developed regions	2005	117181109.0	59963332.0	57217777.0
2	Developed regions	2010	132560325.0	68479248.0	64081077.0
2	Developed regions	2015	140481955.0	72863336.0	67618619.0
3	Developing regions	1990	70184584.0	32700471.0	37484113.0
3	Developing regions	1995	68494898.0	31850220.0	36644678.0
3	Developing regions	2000	69327946.0	31979903.0	37348043.0
3	Developing regions	2005	74087991.0	33439094.0	40648897.0
3	Developing regions	2010	89153918.0	38621281.0	50532637.0
3	Developing regions	2015	103218281.0	44721465.0	58496816.0
4	Least developed countries	1990	11075966.0	5236216.0	5843107.0
4	Least developed countries	1995	11711703.0	5573685.0	6142712.0
4	Least developed countries	2000	10077824.0	4721920.0	5361902.0
4	Least developed countries	2005	9809634.0	4432371.0	5383009.0
4	Least developed countries	2010	10018128.0	4560536.0	5462714.0
4	Least developed countries	2015	11951316.0	5493028.0	6463217.0
5	Less developed regions excluding least developed countries	1990	59105261.0	27464255.0	31641006.0

Table 2: Total population at mid-year by sex and by major area, region, country or area, 1990-2015 (thousands)

		Sex	Both sexes	Female	Male
ID	Area	Year			
1	WORLD	1990	5309667.699	2639243.998	2670423.701
		1995	5735123.084	2848487.191	2886635.893
		2000	6126622.121	3042084.459	3084537.662
		2005	6519635.850	3234553.601	3285082.249
		2010	6929725.043	3435768.139	3493956.904
...	...	...	...	...	...
265	Wallis and Futuna Islands	1995	14.143	NaN	NaN
		2000	14.497	NaN	NaN
		2005	14.246	NaN	NaN
		2010	13.565	NaN	NaN
		2015	13.151	NaN	NaN

1590 rows × 3 columns

ID	Area	Year	Both sexes	Female	Male
1	WORLD	1990	5309667.699	2639243.998	2670423.701
1	WORLD	1995	5735123.084	2848487.191	2886635.893
1	WORLD	2000	6126622.121	3042084.459	3084537.662
1	WORLD	2005	6519635.85	3234553.601	3285082.249
1	WORLD	2010	6929725.04300001	3435768.139	3493956.904
1	WORLD	2015	7349472.099	3642266.346	3707205.753
2	Developed regions	1990	1144463.062	589207.436	555255.626
2	Developed regions	1995	1169761.211	601492.755	568268.456
2	Developed regions	2000	1188811.731	610801.513	578010.218
2	Developed regions	2005	1208919.509	620957.296	587962.213
2	Developed regions	2010	1233375.711	633420.235	599955.476
2	Developed regions	2015	1251351.086	642053.938	609297.148
3	Developing regions	1990	4165204.637	2050036.562	2115168.075
3	Developing regions	1995	4565361.873	2246994.436	2318367.437
3	Developing regions	2000	4937810.39	2431282.946	2506527.444
3	Developing regions	2005	5310716.341	2613596.305	2697120.036
3	Developing regions	2010	5696349.33200001	2802347.904	2894001.428
3	Developing regions	2015	6098121.013	3000212.408	3097908.605
4	Least developed countries	1990	510057.629	256015.073	254042.556
4	Least developed countries	1995	585189.354	293162.612	292026.742
4	Least developed countries	2000	664386.087	332903.612	331482.475
4	Least developed countries	2005	752804.951	377047.236	375757.715
4	Least developed countries	2010	847254.847	424857.315	422397.532
4	Least developed countries	2015	954157.804	478126.625	476031.179
5	Less developed regions excluding least developed countries	1990	3655147.008	1794021.489	1861125.519

Table 3: International migrant stock as a percentage of the total population, 1990-2015

		Sex	Both sexes	Female	Male
ID	Area	Year			
1	WORLD	1990	2.873310	2.834740	2.911430
		1995	2.803806	2.775658	2.831583
		2000	2.818899	2.788169	2.849206
		2005	2.933739	2.887645	2.979124
		2010	3.199467	3.117222	3.280341
...	...	...	...	...	...
265	Wallis and Futuna Islands	1995	11.878668	NaN	NaN
		2000	13.899427	NaN	NaN
		2005	16.601151	NaN	NaN
		2010	20.464431	NaN	NaN
		2015	21.663752	NaN	NaN

1590 rows × 3 columns

1 to 25 of 1590 entries Filter

ID	Area	Year	Both sexes	Female	Male
1	WORLD	1990	2.873309981879527	2.8347398746267793	2.911429746930635
1	WORLD	1995	2.8038064684018558	2.7756584354603824	2.831582507451347
1	WORLD	2000	2.8188993149753947	2.788169465481629	2.8492062224656345
1	WORLD	2005	2.9337390062974147	2.8876450206644764	2.9791240091413616
1	WORLD	2010	3.199466668940384	3.117222253279647	3.280341376528896
1	WORLD	2015	3.315887627264533	3.2283416375942324	3.4019000671312347
2	Developed regions	1990	7.19801544805122	7.1477765599686025	7.2513262567104535
2	Developed regions	1995	7.891085217391433	7.849480248519368	7.935122656183471
2	Developed regions	2000	8.695688333512921	8.650693535528292	8.743235746742457
2	Developed regions	2005	9.693044750095103	9.656595129208371	9.731539839618911
2	Developed regions	2010	10.747765163343647	10.811029426617544	10.68097209933625
2	Developed regions	2015	11.226422110605018	11.348475834751442	11.097806582872762
3	Developing regions	1990	1.6850212682599583	1.5951164777323616	1.7721576570221256
3	Developing regions	1995	1.500316949792865	1.4174587835962045	1.580624253738602
3	Developing regions	2000	1.404022036577229	1.3153509365339002	1.4900312817001815
3	Developing regions	2005	1.3950658676311327	1.2794284234343527	1.5071222807081621
3	Developing regions	2010	1.5651062251250263	1.3781758126773969	1.7461165191933694
3	Developing regions	2015	1.692624347400762	1.4906099608398127	1.8882679723212816
4	Least developed countries	1990	2.171512662542687	2.045276451359565	2.300050468709659 7
4	Least developed countries	1995	2.0013527108697193	1.9012264087754818	2.1034758522217802
4	Least developed countries	2000	1.5168625889662857	1.4184045560911485	1.6175521797947237
4	Least developed countries	2005	1.303077774258687	1.1755479358559733	1.4325744449451956
4	Least developed countries	2010	1.182422034582943	1.0734276753596674	1.2932637115881633
4	Least developed countries	2015	1.2525513023000963	1.1488646966690048	1.3577297633271201
5	Less developed regions excluding least developed countries	1990	1.6170419649507022	1.530876590297074	1.7001005938063223

Table 4: Female migrants as a percentage of the international migrant stock by major area, region, country or area, 1990-2015

			Female Stock Percentage
ID	Area	Year	
1	WORLD	1990	49.039150
		1995	49.168790
		2000	49.112244
		2005	48.832993
		2010	48.305660
...	...	...	...
265	Wallis and Futuna Islands	1995	48.869048
		2000	49.478908
		2005	49.513742
		2010	49.531700
		2015	49.526150

1 to 2

ID	Area	Year	Female Stock Percentage
1	WORLD	1990	49.03914975256289
1	WORLD	1995	49.168789529109105
1	WORLD	2000	49.11224370344867
1	WORLD	2005	48.83299288803053
1	WORLD	2010	48.305660272804396
1	WORLD	2015	48.249769031819895
2	Developed regions	1990	51.12397720437879
2	Developed regions	1995	51.149024101720556
2	Developed regions	2000	51.113307336100966
2	Developed regions	2005	51.1715006895864
2	Developed regions	2010	51.658931886294035
2	Developed regions	2015	51.86668707735452
3	Developing regions	1990	46.59209919944813
3	Developing regions	1995	46.50013494435746
3	Developing regions	2000	46.12844436498955
3	Developing regions	2005	45.13429713595554
3	Developing regions	2010	43.31977984411184
3	Developing regions	2015	43.327077884585194
4	Least developed countries	1990	47.2611548557615
4	Least developed countries	1995	47.57166388267656
4	Least developed countries	2000	46.82668932474215
4	Least developed countries	2005	45.15740603012823
4	Least developed countries	2010	45.499573491631956
4	Least developed countries	2015	45.942752093153
5	Less developed regions excluding least developed countries	1990	46.46668424321822

Table 5: Annual rate of change of the migrant stock by sex and by major area, region, country or area, 1990-2015 (percentage)

		Sex	Both sexes	Female	Male
ID	Area	Year			
1	WORLD	1990-1995	1.051865	1.104667	1.000922
		1995-2000	1.428058	1.405044	1.450294
		2000-2005	2.042124	1.928080	2.151575
		2005-2010	2.954160	2.737012	3.159228
		2010-2015	1.890991	1.867837	1.912603
...	...	...	...	...	...
265	Wallis and Futuna Islands	1990-1995	3.617880	3.886601	3.364378
		1995-2000	3.636508	3.884553	3.396526
		2000-2005	3.203177	3.217252	3.189382
		2005-2010	3.204660	3.211913	3.197545
		2010-2015	0.519140	0.516899	0.521340

1325 rows × 3 columns

1 to 25 of 1325 entries Filter

ID	Area	Year	Both sexes	Female	Male
1	WORLD	1990-1995	1.0518647425166152	1.1046669305252277	1.0009217335145362
1	WORLD	1995-2000	1.428057863098461	1.4050439284500462	1.4502939644990827
1	WORLD	2000-2005	2.042123832765301	1.928079870326455	2.151575465608218
1	WORLD	2005-2010	2.9541604406421347	2.737011950980451	3.1592276505737047
1	WORLD	2010-2015	1.8909914000254764	1.8678373432812825	1.9126034560463894
2	Developed regions	1990-1995	2.2758472453040235	2.2856433391010755	2.2655954630023367
2	Developed regions	1995-2000	2.264965364366877	2.2509947198217244	2.2795827653075498
2	Developed regions	2000-2005	2.5070802663846417	2.5298376478130304	2.4832586438960855
2	Developed regions	2005-2010	2.4663429338210405	2.6559501678821253	2.2656894584023703
2	Developed regions	2010-2015	1.1608244305289614	1.241096530144353	1.0746852876885695
3	Developing regions	1990-1995	-0.4873885006770395	-0.5269038419309634	-0.45297967095805947
3	Developing regions	1995-2000	0.24177650697458386	0.08126769765679266	0.3802461525144045
3	Developing regions	2000-2005	1.3281073659923524	0.8923604058508654	1.6938237519984363
3	Developing regions	2005-2010	3.7022167470843788	2.8815550248910444	4.352954064150006
3	Developing regions	2010-2015	2.929633741752306	2.933002838189001	2.927058412886359
4	Least developed countries	1990-1995	1.1181747061423195	1.2491463004482026	1.0000732507360885
4	Least developed countries	1995-2000	-3.0011391246419965	-3.3168183707581083	-2.7189515113557117
4	Least developed countries	2000-2005	-0.5396362842953523	-1.2656168243647048	0.07857497246459404
4	Least developed countries	2005-2010	0.4191370226801371	0.5701101030075331	0.29396448696282274
4	Least developed countries	2010-2015	3.5269268330280505	3.7207898253194553	3.3636289120042537
5	Less developed regions excluding least developed countries	1990-1995	-0.80324377014423	-0.8841801077171564	-0.7332559407516964
5	Less developed regions excluding least developed countries	1995-2000	0.8501770309210522	0.7334019345311066	0.9502314937034967
5	Less developed regions excluding least developed countries	2000-2005	1.629339638854774	1.243623967396138	1.9522687175941946
5	Less developed regions excluding least developed countries	2005-2010	4.159338546917266	3.2123583491025	4.90597954260659 8
5	Less developed regions excluding least developed countries	2010-2015	2.852687474882951	2.825127386194658	2.873490325882706 5

Table 6: Estimated refugee stock at mid-year by major area, region, country or area, 1990-2015

1. Estimated refugee stock at mid-year (both sexes) as a separate dataset

			Refugee Stock (Both Sexes)
ID	**Area**	**Year**	
1	**WORLD**	**1990**	18836571
		1995	17853840.0
		2000	15827803.0
		2005	13276733.0
		2010	15370755.0
...	...	...	...
265	**Wallis and Futuna Islands**	**1995**	0.0
		2000	0.0
		2005	0.0
		2010	0.0
		2015	0.0

1590 rows × 1 columns

1 to 25 of 1590 entries Filter

ID	Area	Year	Refugee Stock (Both Sexes)
1	WORLD	1990	18836571
1	WORLD	1995	17853840.0
1	WORLD	2000	15827803.0
1	WORLD	2005	13276733.0
1	WORLD	2010	15370755.0
1	WORLD	2015	19577474.0
2	Developed regions	1990	2014564
2	Developed regions	1995	3609670.0
2	Developed regions	2000	2997256.0
2	Developed regions	2005	2361229.0
2	Developed regions	2010	2046917.0
2	Developed regions	2015	1954224.0
3	Developing regions	1990	16822007
3	Developing regions	1995	14244170.0
3	Developing regions	2000	12830547.0
3	Developing regions	2005	10915504.0
3	Developing regions	2010	13323838.0
3	Developing regions	2015	17623250.0
4	Least developed countries	1990	5048391
4	Least developed countries	1995	5160131.0
4	Least developed countries	2000	3047488.0
4	Least developed countries	2005	2363782.0
4	Least developed countries	2010	1957884.0
4	Least developed countries	2015	3443582.0
5	Less developed regions excluding least developed countries	1990	11773616

2. Refugees as a percentage of the international migrant stock as a separate dataset

			Refugee Percentage
ID	Area	Year	
1	WORLD	1990	12.346732
		1995	11.103013
		2000	9.164736
		2005	6.941389
		2010	6.932687
...	...	...	...
265	Wallis and Futuna Islands	1995	0.0
		2000	0.0
		2005	0.0
		2010	0.0
		2015	0.0

1590 rows × 1 columns

1 to 25 of 1590 entries Filter

ID	Area	Year	Refugee Percentage
1	WORLD	1990	12.346732054907182
1	WORLD	1995	11.103013355227622
1	WORLD	2000	9.164736386145329
1	WORLD	2005	6.941389382811964
1	WORLD	2010	6.932687224789613
1	WORLD	2015	8.033424309035137
2	Developed regions	1990	2.445493508340537
2	Developed regions	1995	3.9105113472938857
2	Developed regions	2000	2.89939102801506
2	Developed regions	2005	2.0150253058280923
2	Developed regions	2010	1.544139998148013
2	Developed regions	2015	1.3910854244589634
3	Developing regions	1990	23.968236386497637
3	Developing regions	1995	20.795957678482857
3	Developing regions	2000	18.507034666799445
3	Developing regions	2005	14.73316235555638
3	Developing regions	2010	14.944758793438556
3	Developing regions	2015	17.07376816321907
4	Least developed countries	1990	45.56587979247468
4	Least developed countries	1995	44.0419610226591
4	Least developed countries	2000	30.221556866037503
4	Least developed countries	2005	24.082429819324368
4	Least developed countries	2010	19.533424787369366
4	Least developed countries	2015	28.80153426096571
5	Less developed regions excluding least developed countries	1990	19.919742846580103

Show 25 per page

1 2 10 60 64

3. The annual rate of change of the refugee stock as a separate dataset

ID	Area	Year	Refugee ROC
1	WORLD	1995	-2.123497
		2000	-3.837069
		2005	-5.557223
		2010	-0.025089
		2015	2.947267
...	...	...	...
265	Wallis and Futuna Islands	1995	NaN
		2000	NaN
		2005	NaN
		2010	NaN
		2015	NaN

1325 rows × 1 columns

1 to 25 of 1325 entries Filter

ID	Area	Year	Refugee ROC
1	WORLD	1995	-2.123497443993485
1	WORLD	2000	-3.8370685429335762
1	WORLD	2005	-5.557223280786607
1	WORLD	2010	-0.025088975388118342
1	WORLD	2015	2.9472674315195495
2	Developed regions	1995	9.388423966885876
2	Developed regions	2000	-5.983348404074283
2	Developed regions	2005	-7.2773794111666055
2	Developed regions	2010	-5.32329268351312
2	Developed regions	2015	-2.087655930965729
3	Developing regions	1995	-2.8394168886795867
3	Developing regions	2000	-2.3321542674714575
3	Developing regions	2005	-4.561000330888476
3	Developing regions	2010	0.28519520935836706
3	Developing regions	2015	2.6636520830871824
4	Least developed countries	1995	-0.6803269439341384
4	Least developed countries	2000	-7.531747316485774
4	Least developed countries	2005	-4.541459035747553
4	Least developed countries	2010	-4.187108632338556
4	Least developed countries	2015	7.766031376440223
5	Less developed regions excluding least developed countries	1995	-4.383599796705438
5	Less developed regions excluding least developed countries	2000	0.6324889456790788
5	Less developed regions excluding least developed countries	2005	-4.319730641824886
5	Less developed regions excluding least developed countries	2010	1.5304560436841692
5	Less developed regions excluding least developed countries	2015	1.5710467623816085

Show 25 per page

1 2 10 50 53