Lab Assignment 2: How to Load CSV, ASCII, and other data into Python - Rachel Holman

DS 6001: Practice and Application of Data Science

Instructions

Please answer the following questions as completely as possible using text, code, and the results of code as needed. Format your answers in a Jupyter notebook. To receive full credit, make sure you address every part of the problem, and make sure your document is formatted in a clean and professional way.

There are 11 data files attached to this lab assignment, with different extensions. First, download all of these data files, and save them in the same folder on your local machine. Your task in the following questions is to load each file into Python correctly, so that you can begin the process of data cleaning. If the variable names are included in the file, use those names to name the columns. If the variable names are not included, use these names in order:

In [1]:
```
column_names = ["Country", "Happiness score", "Whisker-high", "Whisker-low",
  "Dystopia (1.92) + residual", "Explained by: GDP per capita",
  "Explained by: Social support", "Explained by: Healthy life expectancy",
  "Explained by: Freedom to make life choices", "Explained by: Generosity",
  "Explained by: Perceptions of corruption" ]
```

If you loaded the data correctly, it will look like `data_clean.csv`, which is also attached to this lab.

Problem 0

Import the libraries you will need. Then write code to change the working directory to the folder in which you saved the data files, run the code displayed above to create the `column_names` list, load `data_clean.csv`, and display the output of the `.info()` method of `data_clean`. (1 point)

In [2]:
```
import numpy as np
import pandas as pd
import os
# set working directory
os.chdir("/Users/rachelholman/Desktop/MSDS/MSDS-SummerCourses/DS6001 - Applicat

data_clean = pd.read_csv('data_clean.csv')
data_clean.info()
data_clean.head(3)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                    Non-Null Count  Dtype
---  ------                                    --------------  -----
 0   Country                                   156 non-null    object
 1   Happiness score                           156 non-null    float64
 2   Whisker-high                              156 non-null    float64
 3   Whisker-low                               156 non-null    float64
 4   Dystopia (1.92) + residual                156 non-null    float64
 5   Explained by: GDP per capita              156 non-null    float64
 6   Explained by: Social support              156 non-null    float64
 7   Explained by: Healthy life expectancy     156 non-null    float64
 8   Explained by: Freedom to make life choices  156 non-null    float64
 9   Explained by: Generosity                  156 non-null    float64
 10  Explained by: Perceptions of corruption   156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

Out[2]:

	Country	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Expla Free to m cho
0	Finland	7.632	7.695	7.569	2.595	1.305	1.592	0.874	0
1	Norway	7.594	7.657	7.530	2.383	1.456	1.582	0.861	0
2	Denmark	7.555	7.623	7.487	2.370	1.351	1.590	0.868	0

Problem 1

Load `data1.csv`. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (1 point)

In [3]:
```
data1 = pd.read_csv('data1.csv', skiprows=2)
data1.info()
data1.head(3)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                    Non-Null Count  Dtype
---  ------                                    --------------  -----
 0   Country                                   156 non-null    object
 1   Happiness score                           156 non-null    float64
 2   Whisker-high                              156 non-null    float64
 3   Whisker-low                               156 non-null    float64
 4   Dystopia (1.92) + residual                156 non-null    float64
 5   Explained by: GDP per capita              156 non-null    float64
 6   Explained by: Social support              156 non-null    float64
 7   Explained by: Healthy life expectancy     156 non-null    float64
 8   Explained by: Freedom to make life choices  156 non-null    float64
 9   Explained by: Generosity                  156 non-null    float64
 10  Explained by: Perceptions of corruption   156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

Out[3]:

	Country	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Expla Free to m cho
0	Finland	7.632	7.695	7.569	2.595	1.305	1.592	0.874	0
1	Norway	7.594	7.657	7.530	2.383	1.456	1.582	0.861	0
2	Denmark	7.555	7.623	7.487	2.370	1.351	1.590	0.868	0

I first opened this dataset in a text editor to look at it and I noticed that the first two lines are not relevant to the code. Because of this, I utilized the `skiprows` function that I read about in the Mastering pandas - Second Edition textbook to ignore the first two lines when reading in the data.

Problem 2

Load `data2.txt`. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (1 point)

In [4]:
```
data2 = pd.read_csv('data2.txt',  header=2, skiprows=range(3,4))
data2.info()
data2.head(3)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 158 entries, 0 to 157
Data columns (total 11 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   Country                                       158 non-null    object
 1   Happiness score                               156 non-null    float64
 2   Whisker-high                                  156 non-null    float64
 3   Whisker-low                                   156 non-null    float64
 4   Dystopia (1.92) + residual                    156 non-null    float64
 5   Explained by: GDP per capita                  156 non-null    float64
 6   Explained by: Social support                  156 non-null    float64
 7   Explained by: Healthy life expectancy         156 non-null    float64
 8   Explained by: Freedom to make life choices    156 non-null    float64
 9   Explained by: Generosity                      156 non-null    float64
 10  Explained by: Perceptions of corruption       156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.7+ KB
```

Out[4]:

	Country	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Expla Free to m cho
0	Finland	7.632	7.695	7.569	2.595	1.305	1.592	0.874	0
1	Norway	7.594	7.657	7.530	2.383	1.456	1.582	0.861	0
2	Denmark	7.555	7.623	7.487	2.370	1.351	1.590	0.868	0

Once again, I opened this data file in text editor first and noticed we had to skip the first two lines AND the 4th line. I struggled for a while trying to do everything using the `skiprows` function, but finally realized with the help of a stack overflow post that I could use `header` and `skiprows` with the `range()` option to skip rows in the middle of a dataset.

Problem 3

Load `data3.txt`. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (1 point)

In [5]:
```
data3 = pd.read_csv('data3.txt',  header=2, sep="\t")
data3.info()
data3.head(3)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                    Non-Null Count  Dtype  
---  ------                                    --------------  -----  
 0   Country                                   156 non-null    object 
 1   Happiness score                           156 non-null    float64
 2   Whisker-high                              156 non-null    float64
 3   Whisker-low                               156 non-null    float64
 4   Dystopia (1.92) + residual                156 non-null    float64
 5   Explained by: GDP per capita              156 non-null    float64
 6   Explained by: Social support              156 non-null    float64
 7   Explained by: Healthy life expectancy     156 non-null    float64
 8   Explained by: Freedom to make life choices  156 non-null    float64
 9   Explained by: Generosity                  156 non-null    float64
 10  Explained by: Perceptions of corruption   156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

Out[5]:

	Country	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Expla Free to m cho
0	Finland	7.632	7.695	7.569	2.595	1.305	1.592	0.874	0
1	Norway	7.594	7.657	7.530	2.383	1.456	1.582	0.861	0
2	Denmark	7.555	7.623	7.487	2.370	1.351	1.590	0.868	0

I opened this data file in text editor, as I do before loading any code in Python, and realized that I not only had to skip the first 2 rows but also that the data was tab delineated, not comma separated. To read in correctly, I used the `sep` function that I read about in the Mastering pandas - Second Edition textbook.

Problem 4

Load `data4.txt`. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (1 point)

In [6]:
```
data4 = pd.read_csv('data4.txt',  names=column_names, sep="$")
data4.info()
data4.head(3)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                    Non-Null Count  Dtype
---  ------                                    --------------  -----
 0   Country                                   156 non-null    object
 1   Happiness score                           156 non-null    float64
 2   Whisker-high                              156 non-null    float64
 3   Whisker-low                               156 non-null    float64
 4   Dystopia (1.92) + residual                156 non-null    float64
 5   Explained by: GDP per capita              156 non-null    float64
 6   Explained by: Social support              156 non-null    float64
 7   Explained by: Healthy life expectancy     156 non-null    float64
 8   Explained by: Freedom to make life choices  156 non-null    float64
 9   Explained by: Generosity                  156 non-null    float64
 10  Explained by: Perceptions of corruption   156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

Out[6]:

	Country	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Expla Free to m cho
0	Finland	7.632	7.695	7.569	2.595	1.305	1.592	0.874	0
1	Norway	7.594	7.657	7.530	2.383	1.456	1.582	0.861	0
2	Denmark	7.555	7.623	7.487	2.370	1.351	1.590	0.868	0

When looking at the data in text editor, I saw that the variables were $ delineated, so I once again employed the `sep` function. Also, there were no header column names, so I used the `names` function to assign them to the list given at the top of this assignment.

Problem 5

Load `data5.csv`. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (1 point)

In [7]:
```
data5 = pd.read_csv('data5.csv', skipfooter=2, engine="python")
data5.info()
data5.tail(3)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                     Non-Null Count  Dtype  
---  ------                                     --------------  -----  
 0   Country                                    156 non-null    object 
 1   Happiness score                            156 non-null    float64
 2   Whisker-high                               156 non-null    float64
 3   Whisker-low                                156 non-null    float64
 4   Dystopia (1.92) + residual                 156 non-null    float64
 5   Explained by: GDP per capita               156 non-null    float64
 6   Explained by: Social support               156 non-null    float64
 7   Explained by: Healthy life expectancy      156 non-null    float64
 8   Explained by: Freedom to make life choices 156 non-null    float64
 9   Explained by: Generosity                   156 non-null    float64
 10  Explained by: Perceptions of corruption    156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

Out[7]:

	Country	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Exp Fre to cl
153	South Sudan	3.254	3.385	3.123	1.691	0.337	0.608	0.177	
154	Central African Republic	3.083	3.227	2.939	2.487	0.024	0.000	0.010	
155	Burundi	2.905	3.074	2.735	1.752	0.091	0.627	0.145	

When I opened this file in text editor, at first it seemed perfectly normal and clean, but then I noticed the last two lines are not part of the data. I tried removing them with the `skiprows` function, but was removing data insted. so I looked at the pandas.read_csv API reference and discovered the `skipfooter` parameter. This worked, but gave warning advising me to inclide `engine="python"`, so I did.

Problem 6

Load `data6.dat`. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (1 point)

In [8]:
```
data6 = pd.read_csv('data6.dat', na_values= 999.000)
data6.info()
data6.head(3)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                Non-Null Count  Dtype  
---  ------                                --------------  -----  
 0   Country                               145 non-null    object 
 1   Happiness score                       142 non-null    float64
 2   Whisker-high                          135 non-null    float64
 3   Whisker-low                           136 non-null    float64
 4   Dystopia (1.92) + residual            145 non-null    float64
 5   Explained by: GDP per capita          137 non-null    float64
 6   Explained by: Social support          134 non-null    float64
 7   Explained by: Healthy life expectancy 142 non-null    float64
 8   Explained by: Freedom to make life choices 140 non-null    float64
 9   Explained by: Generosity              145 non-null    float64
 10  Explained by: Perceptions of corruption 143 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

Out[8]:

	Country	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Expla Free to m cho
0	Finland	7.632	7.695	7.569	2.595	NaN	NaN	NaN	C
1	Norway	7.594	7.657	7.530	NaN	NaN	1.582	NaN	0
2	Denmark	7.555	7.623	7.487	2.370	1.351	1.590	NaN	0

I opened this .dat file in text editor and noticed it was comma separated, so I used `read_csv` instead of `read_fwf`. This data loaded smoothly, but I had to turn to the pandas.read_csv API reference to find the `na_values` parameter which allowed me to replace the 999.000 values with NaN.

Problem 7

Load `data7.xlsx`, which is an Excel file. Keep only the sheet named "Data". Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (2 points)

In [9]:
```
data7 = pd.read_excel('data7.xlsx', sheet_name =1)
data7.info()
data7.head(3)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                     Non-Null Count  Dtype
---  ------                                     --------------  -----
 0   Country                                    156 non-null    object
 1   Happiness score                            156 non-null    float64
 2   Whisker-high                               156 non-null    float64
 3   Whisker-low                                156 non-null    float64
 4   Dystopia (1.92) + residual                 156 non-null    float64
 5   Explained by: GDP per capita               156 non-null    float64
 6   Explained by: Social support               156 non-null    float64
 7   Explained by: Healthy life expectancy      156 non-null    float64
 8   Explained by: Freedom to make life choices 156 non-null    float64
 9   Explained by: Generosity                   156 non-null    float64
 10  Explained by: Perceptions of corruption    156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

Out[9]:

	Country	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Expla Free to m cho
0	Finland	7.632	7.695	7.569	2.595	1.305	1.592	0.874	0
1	Norway	7.594	7.657	7.530	2.383	1.456	1.582	0.861	0
2	Denmark	7.555	7.623	7.487	2.370	1.351	1.590	0.868	0

Because this is an .xlsx file, I opened it in Excel and noticed that the data was set up perfectly but was on its own sheet. For this type of data, I needed to use `read_excel` to load it in, and specify that the `sheet_name` I wanted to read was at index 1.

Problem 8

Load `data8.dta`, which is a Stata 13 file. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (2 points)

In [10]:
```
data8 = pd.read_stata('data8.dta')
data8.info()
data8.head(3)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   country                           156 non-null    object
 1   happinessscore                    156 non-null    float32
 2   whiskerhigh                       156 non-null    float32
 3   whiskerlow                        156 non-null    float32
 4   dystopia192residual               156 non-null    float32
 5   explainedbygdppercapita           156 non-null    float32
 6   explainedbysocialsupport          156 non-null    float32
 7   explainedbyhealthylifeexpectancy  156 non-null    float32
 8   explainedbyfreedomtomakelifechoi  156 non-null    float32
 9   explainedbygenerosity             156 non-null    float32
 10  explainedbyperceptionsofcorrupti  156 non-null    float32
dtypes: float32(10), object(1)
memory usage: 8.5+ KB
```

Out[10]:

	country	happinessscore	whiskerhigh	whiskerlow	dystopia192residual	explainedbygdpperc
0	Finland	7.632	7.695	7.569	2.595	
1	Norway	7.594	7.657	7.530	2.383	
2	Denmark	7.555	7.623	7.487	2.370	

When I opened this data file in text editor it looked like nonsense, so I knew it would not use `read_csv` or `read_excel`. Then I remembered working with this sort of data file in our live coding lesson and implemented `read_stata` just like we did earlier today!

Problem 9

Load `data9.sav`, which is an SPSS file. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (2 points)

In [11]:
```
data9 = pd.read_spss('data9.sav')
data9.info()
data9.head(3)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   country      156 non-null    object 
 1   happiness    156 non-null    float64
 2   whiskerhigh  156 non-null    float64
 3   whiskerlow   156 non-null    float64
 4   dystopia     156 non-null    float64
 5   gdpPC        156 non-null    float64
 6   socsupport   156 non-null    float64
 7   lifeexp      156 non-null    float64
 8   lifechoice   156 non-null    float64
 9   generous     156 non-null    float64
 10  corrupt      156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

Out[11]:

	country	happiness	whiskerhigh	whiskerlow	dystopia	gdpPC	socsupport	lifeexp	lifecho
0	Finland	7.632	7.695	7.569	2.595	1.305	1.592	0.874	0.6
1	Norway	7.594	7.657	7.530	2.383	1.456	1.582	0.861	0.6
2	Denmark	7.555	7.623	7.487	2.370	1.351	1.590	0.868	0.6

For this data set, I googled what .sav files needed to be loaded into Python and learned that they are SPSS, so I implemented `read_spss`. This turned out to be a bit tricky because I had to open my console and install `pyreadstat` in order for the function to run without error.

Problem 10

Load `data10.xpt`, which is a SAS file. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (If some of the country names display as `b'Finland'`, don't worry aout that.) (2 points)

In [12]:
```
data10 = pd.read_sas('data10.xpt')
data10.columns=column_names
data10.info()
data10.head(3)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                    Non-Null Count  Dtype
---  ------                                    --------------  -----
 0   Country                                   156 non-null    object
 1   Happiness score                           156 non-null    float64
 2   Whisker-high                              156 non-null    float64
 3   Whisker-low                               156 non-null    float64
 4   Dystopia (1.92) + residual                156 non-null    float64
 5   Explained by: GDP per capita              156 non-null    float64
 6   Explained by: Social support              156 non-null    float64
 7   Explained by: Healthy life expectancy     156 non-null    float64
 8   Explained by: Freedom to make life choices  156 non-null    float64
 9   Explained by: Generosity                  156 non-null    float64
 10  Explained by: Perceptions of corruption   156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

Out[12]:

	Country	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Exp Fr t c
0	b'Finland'	7.632	7.695	7.569	2.595	1.305	1.592	0.874	
1	b'Norway'	7.594	7.657	7.530	2.383	1.456	1.582	0.861	
2	b'Denmark'	7.555	7.623	7.487	2.370	1.351	1.590	0.868	

I used Google to figure out what the .xpt extention is and determined that it should be read in using `read_sas`. Once the data was read in, I changed the column names by adding the line `data6.columns = column_names` just so I can refer to this if I forget how in the future.

Problem 11

Please load the `data11.txt` file, which is a fixed width file. The columns are defined as follows:

Variable	Width	Start	End
Country	24	1	24
Happiness score	5	25	29
Whisker-high	5	30	34
Whisker-low	5	35	39
Dystopia (1.92) + residual	5	40	44
Explained by: GDP per capita	5	45	49
Explained by: Social support	5	50	54
Explained by: Healthy life expectancy	5	55	59

Variable	Width	Start	End
Explained by: Freedom to make life choices	5	60	64
Explained by: Generosity	5	65	69
Explained by: Perceptions of corruption	5	70	74

Then save the this loaded data frame as a CSV file on your local machine. Be sure to use a unique filename so as not to overwrite any existing files. (5 points)

In [13]:
```
widths = [24,5,5,5,5,5,5,5,5,5,5]
data11 = pd.read_fwf('data11.txt', widths=widths, header=None, names=column_nam
data11.info()
data11.head(3)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                    Non-Null Count  Dtype
---  ------                                    --------------  -----
 0   Country                                   156 non-null    object
 1   Happiness score                           156 non-null    float64
 2   Whisker-high                              156 non-null    float64
 3   Whisker-low                               156 non-null    float64
 4   Dystopia (1.92) + residual                156 non-null    float64
 5   Explained by: GDP per capita              156 non-null    float64
 6   Explained by: Social support              156 non-null    float64
 7   Explained by: Healthy life expectancy     156 non-null    float64
 8   Explained by: Freedom to make life choices  156 non-null    float64
 9   Explained by: Generosity                  156 non-null    float64
 10  Explained by: Perceptions of corruption   156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

Out[13]:

	Country	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Expla Free to m cho
0	Finland	7.632	7.695	7.569	2.595	1.305	1.592	0.874	0
1	Norway	7.594	7.657	7.530	2.383	1.456	1.582	0.861	0
2	Denmark	7.555	7.623	7.487	2.370	1.351	1.590	0.868	0

I recognized this fixed-width file as being similar to an example we covered in the live coding session, so I utilized tools we learned in class to properly load it. This involved saving the widths as a list to initialize in `read_fwf`, as well as assigning the column names with the `names` parameter.