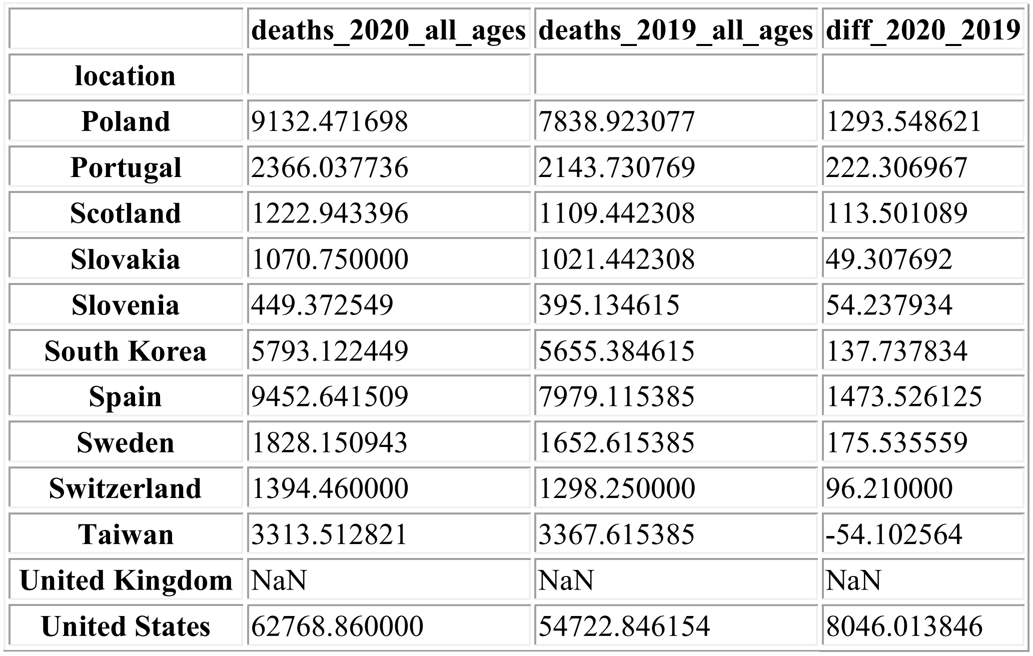
Data Wrangling Coursework 1

Answers are written in Python 3, using the Pandas library. Using these an initial DataFrame named ‘df’ created from the data set provided will be used to create all other DataFrames.

Q1 – Average Deaths

To find the average deaths, the pandas.dataframe.mean() function has been used, however, a median could have been used instead using pandas.dataframe.median(). This could produce a far different set of results.

	deaths_2020_all_ages	deaths_2019_all_ages	diff_2020_2019
location			
Austria	1726.566038	1567.019231	159.546807
Belgium	2429.622642	2085.307692	344.314949
Bulgaria	2378.773585	2073.423077	305.350508
Canada	5721.785714	5443.557692	278.228022
Chile	2402.547170	2097.673077	304.874093
Croatia	1025.479167	992.519231	32.959936
Czechia	2439.666667	2154.615385	285.051282
Denmark	1046.698113	1034.711538	11.986575
England & Wales	11586.886792	10139.115385	1447.771408
Estonia	305.301887	294.846154	10.455733
Finland	1052.634615	1035.000000	17.634615
France	12509.641509	11498.403846	1011.237663
Germany	18695.288462	18014.846154	680.442308
Greece	2478.448980	2394.961538	83.487441
Hungary	2654.442308	2485.884615	168.557692
Iceland	43.679245	43.480769	0.198476
Israel	930.584906	880.653846	49.931060
Italy	13501.886364	12364.692308	1137.194056
Latvia	542.188679	531.384615	10.804064
Lithuania	826.867925	734.307692	92.560232
Luxembourg	86.489796	82.461538	4.028257
Netherlands	3229.716981	2914.288462	315.428520
New Zealand	625.509434	655.538462	-30.029028
Northern Ireland	335.941176	300.134615	35.806561
Norway	774.679245	778.038462	-3.359216



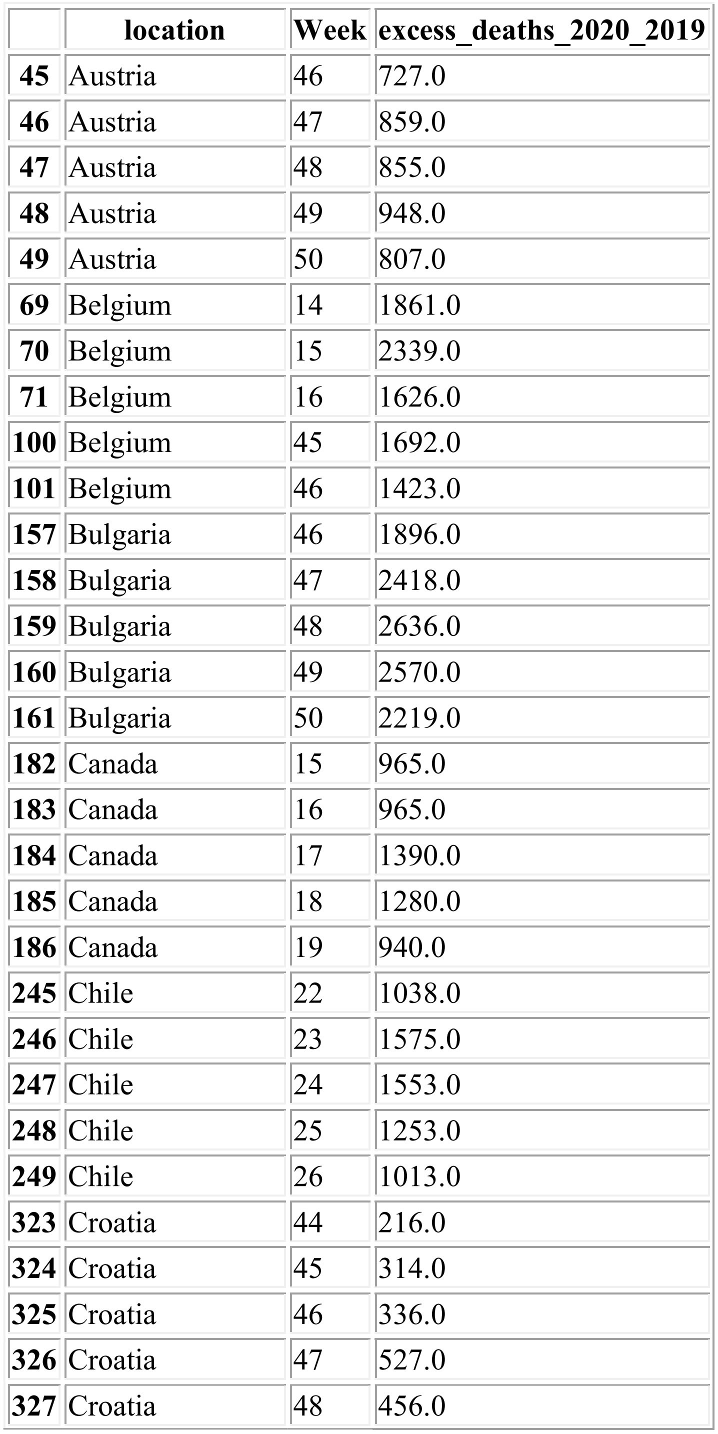
	deaths_2020_all_ages	deaths_2019_all_ages	diff_2020_2019
location			
Poland	9132.471698	7838.923077	1293.548621
Portugal	2366.037736	2143.730769	222.306967
Scotland	1222.943396	1109.442308	113.501089
Slovakia	1070.750000	1021.442308	49.307692
Slovenia	449.372549	395.134615	54.237934
South Korea	5793.122449	5655.384615	137.737834
Spain	9452.641509	7979.115385	1473.526125
Sweden	1828.150943	1652.615385	175.535559
Switzerland	1394.460000	1298.250000	96.210000
Taiwan	3313.512821	3367.615385	-54.102564
United Kingdom	NaN	NaN	NaN
United States	62768.860000	54722.846154	8046.013846

A notable result from the produced DataFrame is the lack of United Kingdom values, however, Scotland, Northern Ireland and England & Wales are listed as separate entries. These could be aggregated under United Kingdom, or dropped from the DataFrame using the pandas.dataframe.dropna() function.

Word Count: 530

Q2 – Excess Deaths 2020/2019

The rows in this DataFrame have been sorted by descending 'excess_deaths_2020_2019' and grouped by 'location', where the top 5 values for each country are taken with pandas.dataframe.head(5). These values are sorted by 'location' and 'Week' to make the data easy to read.



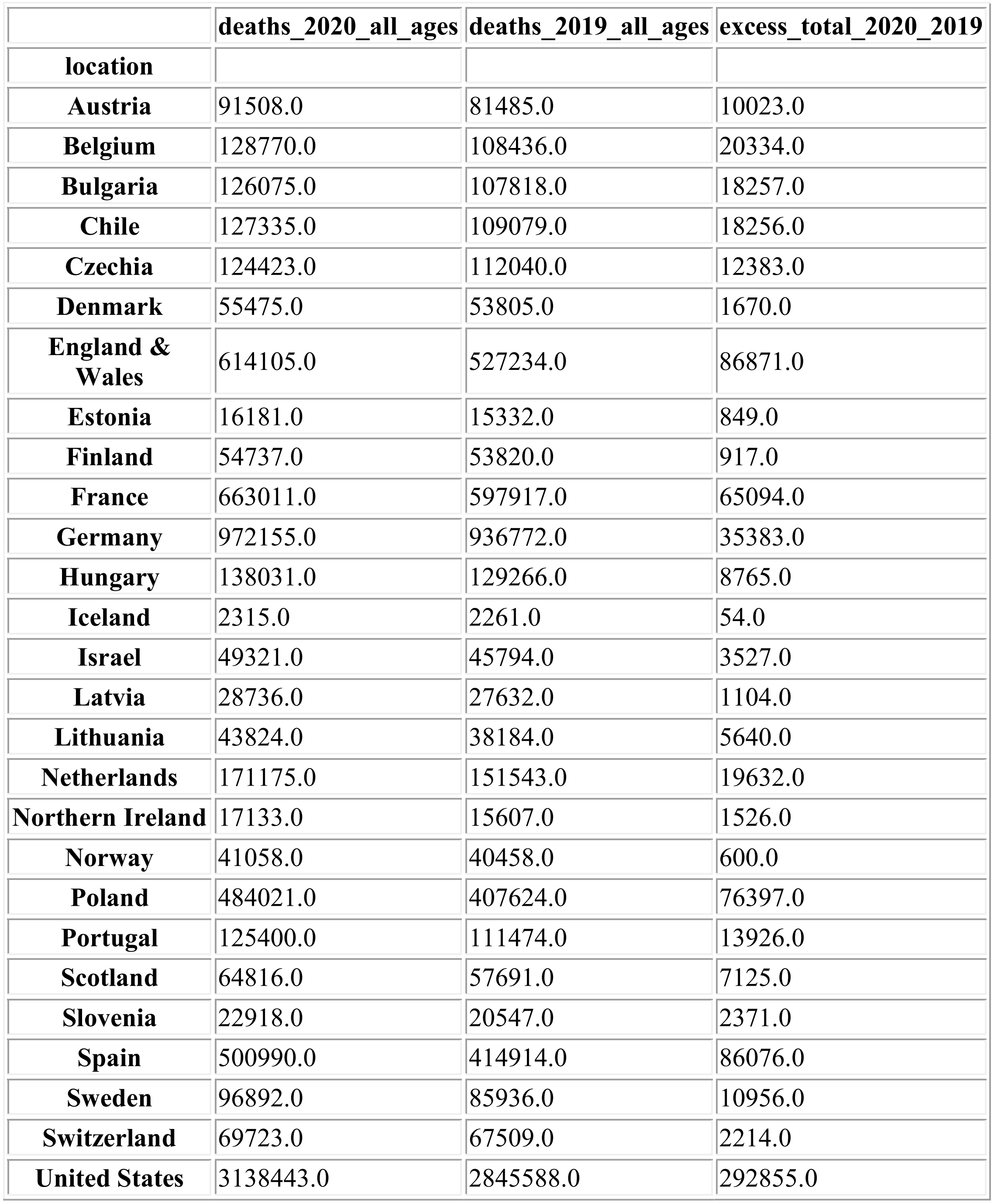
	location	Week	excess_deaths_2020_2019
45	Austria	46	727.0
46	Austria	47	859.0
47	Austria	48	855.0
48	Austria	49	948.0
49	Austria	50	807.0
69	Belgium	14	1861.0
70	Belgium	15	2339.0
71	Belgium	16	1626.0
100	Belgium	45	1692.0
101	Belgium	46	1423.0
157	Bulgaria	46	1896.0
158	Bulgaria	47	2418.0
159	Bulgaria	48	2636.0
160	Bulgaria	49	2570.0
161	Bulgaria	50	2219.0
182	Canada	15	965.0
183	Canada	16	965.0
184	Canada	17	1390.0
185	Canada	18	1280.0
186	Canada	19	940.0
245	Chile	22	1038.0
246	Chile	23	1575.0
247	Chile	24	1553.0
248	Chile	25	1253.0
249	Chile	26	1013.0
323	Croatia	44	216.0
324	Croatia	45	314.0
325	Croatia	46	336.0
326	Croatia	47	527.0
327	Croatia	48	456.0

	location	Week	excess_deaths_2020_2019
378	Czechia	43	1495.0
379	Czechia	44	2146.0
380	Czechia	45	2102.0
381	Czechia	46	1617.0
382	Czechia	47	1221.0
403	Denmark	12	101.0
404	Denmark	13	90.0
406	Denmark	15	125.0
442	Denmark	51	231.0
443	Denmark	52	173.0
461	England & Wales	14	6261.0
462	England & Wales	15	8225.0
463	England & Wales	16	13326.0
464	England & Wales	17	11938.0
465	England & Wales	18	6746.0
517	Estonia	14	55.0
534	Estonia	31	50.0
542	Estonia	39	50.0
551	Estonia	48	47.0
555	Estonia	52	56.0
574	Finland	15	122.0
576	Finland	17	141.0
585	Finland	26	126.0
596	Finland	37	133.0
606	Finland	47	143.0
628	France	13	4875.0
629	France	14	7330.0
630	France	15	6540.0
631	France	16	4108.0
660	France	45	4024.0

Within the produced data set, it can be observed that most countries have sequential or partially sequential weeks for the top 5, suggesting a spike(s) in excess deaths over this period compared to the year before. Exceptions to this include Estonia and Finland, suggesting relatively consistent rates of death compared to the year before.

Q3 – Positive Excess 2020/2019

The columns in this DataFrame have been grouped by ‘location’ and totalled using the pandas.dataframe.sum() function. Pandas.dataframe.loc() is used to find all rows where 'excess_total_2020_2019' is greater than 0.



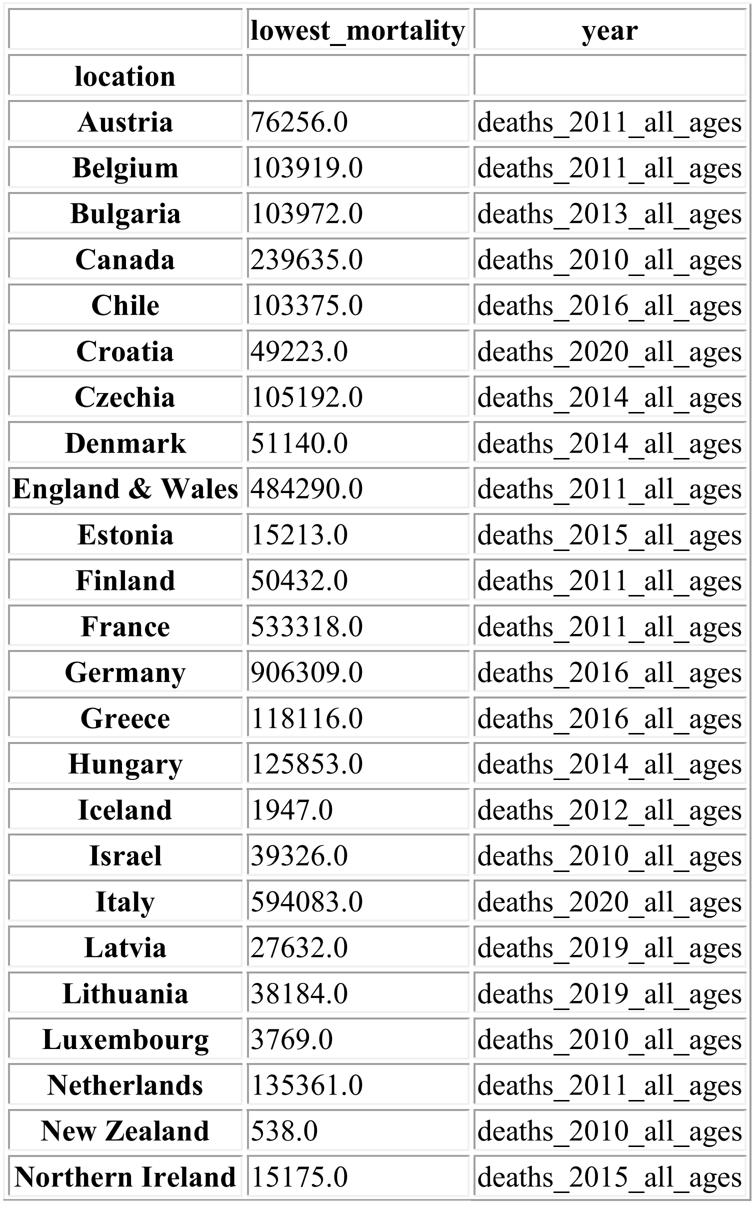
	deaths_2020_all_ages	deaths_2019_all_ages	excess_total_2020_2019
location			
Austria	91508.0	81485.0	10023.0
Belgium	128770.0	108436.0	20334.0
Bulgaria	126075.0	107818.0	18257.0
Chile	127335.0	109079.0	18256.0
Czechia	124423.0	112040.0	12383.0
Denmark	55475.0	53805.0	1670.0
England & Wales	614105.0	527234.0	86871.0
Estonia	16181.0	15332.0	849.0
Finland	54737.0	53820.0	917.0
France	663011.0	597917.0	65094.0
Germany	972155.0	936772.0	35383.0
Hungary	138031.0	129266.0	8765.0
Iceland	2315.0	2261.0	54.0
Israel	49321.0	45794.0	3527.0
Latvia	28736.0	27632.0	1104.0
Lithuania	43824.0	38184.0	5640.0
Netherlands	171175.0	151543.0	19632.0
Northern Ireland	17133.0	15607.0	1526.0
Norway	41058.0	40458.0	600.0
Poland	484021.0	407624.0	76397.0
Portugal	125400.0	111474.0	13926.0
Scotland	64816.0	57691.0	7125.0
Slovenia	22918.0	20547.0	2371.0
Spain	500990.0	414914.0	86076.0
Sweden	96892.0	85936.0	10956.0
Switzerland	69723.0	67509.0	2214.0
United States	3138443.0	2845588.0	292855.0

When shown as a percentage of the total, the sum of the UK constituents and the US occupies approximately 48% which is a huge fraction of the total positive excess deaths over this period. This could suggest that the healthcare within these regions is significantly worse that the other countries within the data set.

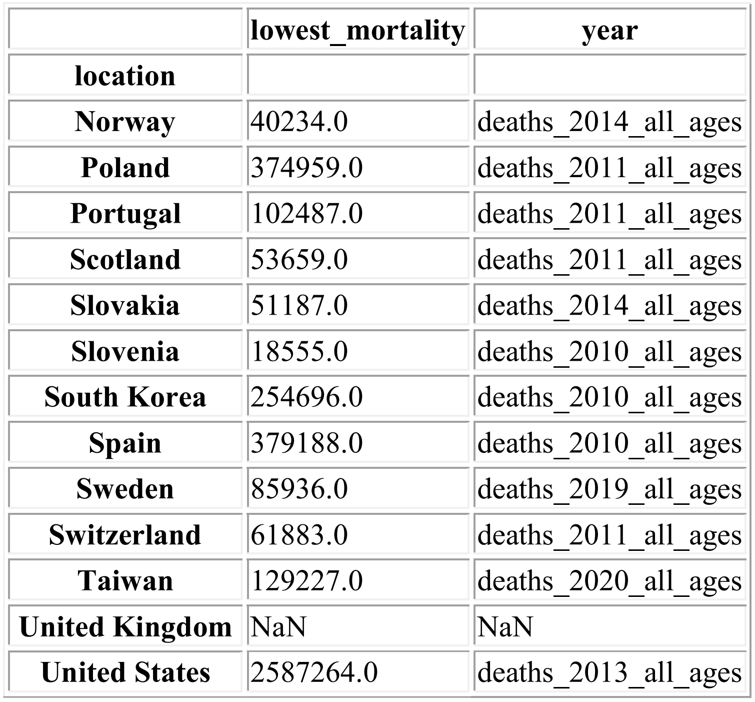
Word Count: 530

Q4 – Lowest Mortality

Df_del_4 consists of all full years (therefore excluding 2021) and 'location' columns from 'df'. Each row has been grouped by location and summed to give each year as a total figure. The minimum non-zero year has been selected using pandas.dataframe[dataframe>0].min(axis=1). The minimum, non-zero value has been selected as otherwise the method would return years with no data and thus 0 total deaths. The column index of the minimum value is then selected using pandas.dataframe[dataframe>0].idxmin(axis=1). One key weakness of this approach is that it will still count incomplete year datasets.



	lowest_mortality	year
location		
Austria	76256.0	deaths_2011_all_ages
Belgium	103919.0	deaths_2011_all_ages
Bulgaria	103972.0	deaths_2013_all_ages
Canada	239635.0	deaths_2010_all_ages
Chile	103375.0	deaths_2016_all_ages
Croatia	49223.0	deaths_2020_all_ages
Czechia	105192.0	deaths_2014_all_ages
Denmark	51140.0	deaths_2014_all_ages
England & Wales	484290.0	deaths_2011_all_ages
Estonia	15213.0	deaths_2015_all_ages
Finland	50432.0	deaths_2011_all_ages
France	533318.0	deaths_2011_all_ages
Germany	906309.0	deaths_2016_all_ages
Greece	118116.0	deaths_2016_all_ages
Hungary	125853.0	deaths_2014_all_ages
Iceland	1947.0	deaths_2012_all_ages
Israel	39326.0	deaths_2010_all_ages
Italy	594083.0	deaths_2020_all_ages
Latvia	27632.0	deaths_2019_all_ages
Lithuania	38184.0	deaths_2019_all_ages
Luxembourg	3769.0	deaths_2010_all_ages
Netherlands	135361.0	deaths_2011_all_ages
New Zealand	538.0	deaths_2010_all_ages
Northern Ireland	15175.0	deaths_2015_all_ages



	lowest_mortality	year
location		
Norway	40234.0	deaths_2014_all_ages
Poland	374959.0	deaths_2011_all_ages
Portugal	102487.0	deaths_2011_all_ages
Scotland	53659.0	deaths_2011_all_ages
Slovakia	51187.0	deaths_2014_all_ages
Slovenia	18555.0	deaths_2010_all_ages
South Korea	254696.0	deaths_2010_all_ages
Spain	379188.0	deaths_2010_all_ages
Sweden	85936.0	deaths_2019_all_ages
Switzerland	61883.0	deaths_2011_all_ages
Taiwan	129227.0	deaths_2020_all_ages
United Kingdom	NaN	NaN
United States	2587264.0	deaths_2013_all_ages

If 2021 had been included, many rows would list ‘deaths_2021_all_ages’ as the value for ‘year’. This is because the year is not complete, thus has less values to sum.

Word Count: 530

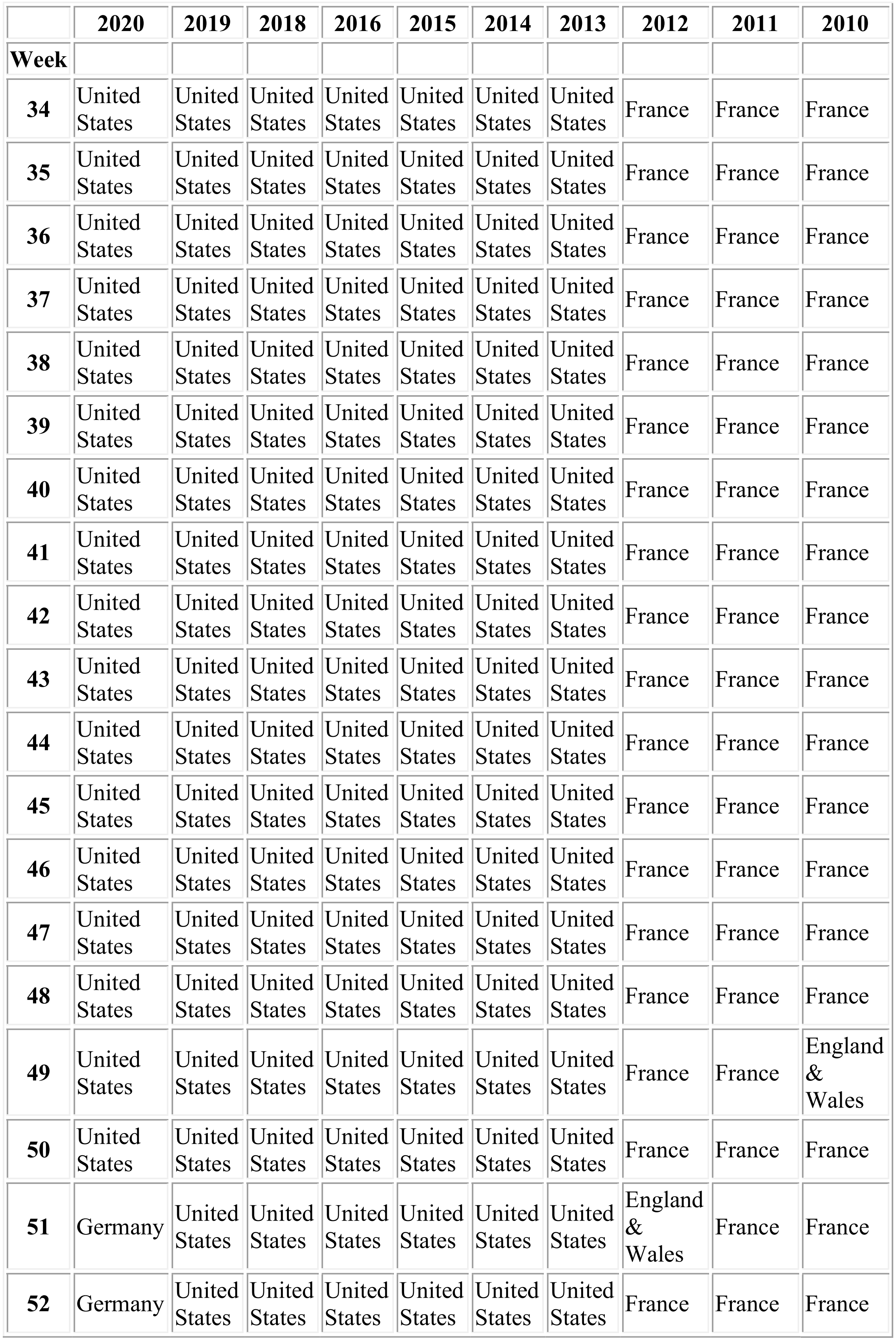
Q5 – Weekly Highest Mortality

All full years were added to an array as they are used multiple times throughout the code. Furthermore, using full years avoids the number of rows being severely cropped when rows with NaN (float null) values are dropped, as would happen if 'deaths_2021_all_ages' is added. Pandas.dataframe.astype(int) has been used to convert all values to integers as pandas.dataframe.iloc[] requires integers to function properly.

	2020	**2019**	**2018**	**2016**	**2015**	**2014**	**2013**	**2012**	**2011**	**2010**
Week										
1	United States	United States	United States	United States	United States	United States	United States	France	England & Wales	England & Wales
2	United States	United States	United States	United States	United States	United States	United States	France	England & Wales	England & Wales
3	United States	United States	United States	United States	United States	United States	United States	France	England & Wales	England & Wales
4	United States	United States	United States	United States	United States	United States	United States	France	France	France
5	United States	United States	United States	United States	United States	United States	United States	France	France	France
6	United States	United States	United States	United States	United States	United States	United States	France	France	France
7	United States	United States	United States	United States	United States	United States	United States	France	France	France
8	United States	United States	United States	United States	United States	United States	United States	France	France	France
9	United States	United States	United States	United States	United States	United States	United States	France	France	France
10	United States	United States	United States	United States	United States	United States	United States	France	France	France
11	United States	United States	United States	United States	United States	United States	United States	France	France	France
12	United States	United States	United States	United States	United States	United States	United States	France	France	France
13	United States	United States	United States	United States	United States	United States	United States	France	France	France
14	United States	United States	United States	United States	United States	United States	United States	France	France	France

	2020	2019	2018	2016	2015	2014	2013	2012	2011	2010
Week										
15	United States	United States	United States	United States	United States	United States	United States	France	France	France
16	United States	United States	United States	United States	United States	United States	United States	England & Wales	France	France
17	United States	United States	United States	United States	United States	United States	United States	France	France	France
18	United States	United States	United States	United States	United States	United States	United States	France	France	France
19	United States	United States	United States	United States	United States	United States	United States	France	England & Wales	France
20	United States	United States	United States	United States	United States	United States	United States	France	France	France
21	United States	United States	United States	United States	United States	United States	United States	France	France	France
22	United States	United States	United States	United States	United States	United States	United States	France	France	France
23	United States	United States	United States	United States	United States	United States	United States	France	France	France
24	United States	United States	United States	United States	United States	United States	United States	England & Wales	France	France
25	United States	United States	United States	United States	United States	United States	United States	France	France	France
26	United States	United States	United States	United States	United States	United States	United States	France	France	France
27	United States	United States	United States	United States	United States	United States	United States	France	France	France
28	United States	United States	United States	United States	United States	United States	United States	France	France	France
29	United States	United States	United States	United States	United States	United States	United States	France	France	France
30	United States	United States	United States	United States	United States	United States	United States	France	France	France
31	United States	United States	United States	United States	United States	United States	United States	France	France	France
32	United States	United States	United States	United States	United States	United States	United States	France	France	France
33	United States	United States	United States	United States	United States	United States	United States	France	France	France

Word Count: 530



	2020	2019	2018	2016	2015	2014	2013	2012	2011	2010
Week										
34	United States	United States	United States	United States	United States	United States	United States	France	France	France
35	United States	United States	United States	United States	United States	United States	United States	France	France	France
36	United States	United States	United States	United States	United States	United States	United States	France	France	France
37	United States	United States	United States	United States	United States	United States	United States	France	France	France
38	United States	United States	United States	United States	United States	United States	United States	France	France	France
39	United States	United States	United States	United States	United States	United States	United States	France	France	France
40	United States	United States	United States	United States	United States	United States	United States	France	France	France
41	United States	United States	United States	United States	United States	United States	United States	France	France	France
42	United States	United States	United States	United States	United States	United States	United States	France	France	France
43	United States	United States	United States	United States	United States	United States	United States	France	France	France
44	United States	United States	United States	United States	United States	United States	United States	France	France	France
45	United States	United States	United States	United States	United States	United States	United States	France	France	France
46	United States	United States	United States	United States	United States	United States	United States	France	France	France
47	United States	United States	United States	United States	United States	United States	United States	France	France	France
48	United States	United States	United States	United States	United States	United States	United States	France	France	France
49	United States	United States	United States	United States	United States	United States	United States	France	France	England & Wales
50	United States	United States	United States	United States	United States	United States	United States	France	France	France
51	Germany	United States	United States	United States	United States	United States	United States	England & Wales	France	France
52	Germany	United States	United States	United States	United States	United States	United States	France	France	France

Word Count: 530

United States makes up a majority of the weeks with highest mortality and it can be assumed that where United States does not occupy the highest mortality of a given week, data is missing from that week for United States. This can be seen in weeks 51 and 52 of 2020 where data is missing for United States thus Germany had the highest mortality for those weeks.

Word Count: 530

Code Listing

Set-up

```
# Import Relevant Libraries

import pandas as pd


# Initialise Dataframe 'df'

df = pd.read_csv('excess_mortality.csv')
```

Q1

```
# Select the columns that we want, in this case, location, deaths_2020_all_ages

# and deaths_2019_all_ages are the columns that we need to start with.

df_del_1 = df[['location', 'deaths_2020_all_ages', 'deaths_2019_all_ages']]


# Group the rows based on location and treat each location as a data set, then

# get the mean value for the selected columns.

df_del_1 = df_del_1.groupby('location').mean()


# Add a column for the difference betwen 2020 and 2019.

df_del_1['diff_2020_2019'] = df_del_1['deaths_2020_all_ages'] - df_del_1['deaths_2019_all_ages']


# Print Values

df_del_1
```

Q2

```
# Select the Columns that we need

df_del_2 = df[['location', 'Week']]


# Add Excess Deaths 2020 - 2019 column and set it equal to the difference between 2020 and 2019
```

```
df_del_2['excess_deaths_2020_2019'] = df['deaths_2020_all_ages'] -
df['deaths_2019_all_ages']


# Sort the values by descending value of excess deaths and group them by location

# Then take the top 5 values of each location group (head) and then sort all of these by
country alphabetically

df_del_2_top_5 = df_del_2.sort_values('excess_deaths_2020_2019',
ascending=False).groupby('location').head(5).sort_values(['location', 'Week'])


#Print Results

df_del_2_top_5.head(60)
```

Q3

```
# Take the three columns that we need (location, deaths for 2020 and deaths for 2019)

df_del_3 = df[['location', 'deaths_2020_all_ages', 'deaths_2019_all_ages']]


# Group the values by country/location and sum all of the values, giving columns:

# location x, sum of deaths for 2020 in location x, sum of deaths in 2019 in location x

df_del_3 = df_del_3.groupby('location').sum()


# Add a new column to the data set, excess_total_2020_2019, equal to the difference
between

# the sums of 2020 and 2019 for each country/location

df_del_3['excess_total_2020_2019'] = df_del_3['deaths_2020_all_ages'] -
df_del_3['deaths_2019_all_ages']


# Locate all rows where the the difference is positive, ie above 0

df_del_3_pos_excess = df_del_3.loc[(df_del_3['excess_total_2020_2019'] > 0)]


# Print Results

df_del_3_pos_excess
```

Word Count: 530

Q4

```
# Include Columns for locations and FULL YEARS (2021 excluded as not complete)

df_del_4 = df[['location', 'deaths_2020_all_ages', 'deaths_2019_all_ages',
'deaths_2018_all_ages', 'deaths_2017_all_ages', 'deaths_2016_all_ages',
'deaths_2015_all_ages', 'deaths_2014_all_ages', 'deaths_2013_all_ages',
'deaths_2012_all_ages', 'deaths_2011_all_ages', 'deaths_2010_all_ages']]


# Group rows by location and sum the values for each column (year)

df_del_4 = df_del_4.groupby('location').sum()


# Get the value from the column with the minimum, assuming that years with 0 as the sum
to be incomplete,

# thus exclude them from the minimum count

df_del_4['lowest_mortality'] = df_del_4[df_del_4>0].min(axis=1)


# Get the column index for the minimum, non-zero total (to match the value of
lowest_mortality)

df_del_4['year'] = df_del_4[df_del_4>0].idxmin(axis=1)


# Drop the unnecessary columns

df_del_4 = df_del_4.drop(columns=['deaths_2020_all_ages', 'deaths_2019_all_ages',
'deaths_2018_all_ages', 'deaths_2017_all_ages', 'deaths_2016_all_ages',
'deaths_2015_all_ages', 'deaths_2014_all_ages', 'deaths_2013_all_ages',
'deaths_2012_all_ages', 'deaths_2011_all_ages', 'deaths_2010_all_ages'])


# Print Results

df_del_4
```

Q5

```
# Array of column names for years for for loop

years = ['deaths_2020_all_ages', 'deaths_2019_all_ages', 'deaths_2018_all_ages',
'deaths_2016_all_ages', 'deaths_2015_all_ages', 'deaths_2014_all_ages',
'deaths_2013_all_ages', 'deaths_2012_all_ages', 'deaths_2011_all_ages',
'deaths_2010_all_ages']

# Include Columns for locations, weeks and FULL YEARS (2021 excluded as not complete)
df_del_5 = df[['Week', 'location']+years]

# Group rows by week and get the index of the maximum value,
# then drop null (float nan) values and convert all values to int
# so iloc doesnt cause errors. Crops weeks to 52.
df_del_5 = df_del_5.groupby('Week').idxmax().dropna().astype(int)

# Iterate through years array (foreach):
for year in years:
    # For each row in df_del_5
    for week in range(len(df_del_5[year])):
        # Creates a new column for each year and at cell coords (week+1, year)
        # take the current value and use it as the index to find the row in
        # df where location can be taken from
        df_del_5.at[week+1,year[7:11]] = df.iloc[int(df_del_5.at[week+1, year])]['location']

# Drop all of the deaths_20XX_all_ages columns
df_del_5 = df_del_5.drop(columns=years)

# Print Results
df_del_5
```

Word Count: 530