```python
import geopandas
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Question 1: How to read various filetypes

```python
# CSV File 2022 Covid-19 cases and deaths by county

df = geopandas.read_file('us-counties-2022.csv')
df.head()
```

	date	county	state	fips	cases	deaths	geometry
0	2022-01-01	Autauga	Alabama	01001	11018	160	None
1	2022-01-01	Baldwin	Alabama	01003	39911	593	None
2	2022-01-01	Barbour	Alabama	01005	3860	81	None
3	2022-01-01	Bibb	Alabama	01007	4533	95	None
4	2022-01-01	Blount	Alabama	01009	11256	198	None

```python
# Shapefile of U.S. states

df2 = geopandas.read_file('tl_2022_us_state.shp')
df2.head()
```

	REGION	DIVISION	STATEFP	STATENS	GEOID	STUSPS	NAME	LSAD	MTFCC	FUNCSTAT	ALAND	AWATER	INTPTLAT
0	3	5	54	01779805	54	WV	West Virginia	00	G4000	A	62266456923	489045863	+38.6472854
1	3	5	12	00294478	12	FL	Florida	00	G4000	A	138962819934	45971472526	+28.3989775
2	2	3	17	01779784	17	IL	Illinois	00	G4000	A	143778515726	6216539665	+40.1028754
3	2	4	27	00662849	27	MN	Minnesota	00	G4000	A	206244837557	18937184315	+46.3159573
4	3	5	24	01714934	24	MD	Maryland	00	G4000	A	25151771744	6979295311	+38.9466584

```python
# GeoJSON shapefile of the United States

url = 'https://raw.githubusercontent.com/johan/world.geo.json/master/countries/USA.geo.json'
df3 = geopandas.read_file(url)
df3.head()
```

	id	name	geometry
0	USA	United States of America	MULTIPOLYGON (((-155.54211 19.08348, -155.6881...

```python
# A Geopackage of protected areas in Virginia

df4 = geopandas.read_file('PADUS2_1StateVA.gpkg')
df4.head()
```

	Category	d_Category	Own_Type	d_Own_Type	Own_Name	d_Own_Name	Loc_Own	Mang_Type	d_Mang_Typ	Mang_Name	...	GAPCdSrc
0	Fee	Fee	UNK	Unknown	UNK	Unknown	Unknown	FED		TVA	...	TNC
1	Fee	Fee	UNK	Unknown	UNK	Unknown	Unknown	FED		TVA	...	GAP
2	Fee	Fee	UNK	Unknown	UNK	Unknown		FED		BLM	...	GAP - Default
3	Fee	Fee	UNK	Unknown	UNK	Unknown	Unknown	FED		USACE	...	GAP - Default
4	Fee	Fee	UNK	Unknown	UNK	Unknown	Unknown	FED		USACE	...	GAP - Default

5 rows × 41 columns

```python
# Reading in files with geopandas is very similar to reading in files with pandas,
# you use the .read_file() function. If reading a file locally, give the data the
# appropriate file name and extension. If reading from the web, assign the name
#'url' to the link and then use .read_file() to read the url.
```

Question 2: Calculating a field

```python
df2['totalarea'] = df2['ALAND'] + df2['AWATER']
df2.head()
```

	REGION	DIVISION	STATEFP	STATENS	GEOID	STUSPS	NAME	LSAD	MTFCC	FUNCSTAT	ALAND	AWATER	INTPTLAT
0	3	5	54	01779805	54	WV	West Virginia	00	G4000	A	62266456923	489045863	+38.6472854
1	3	5	12	00294478	12	FL	Florida	00	G4000	A	138962819934	45971472526	+28.3989775
2	2	3	17	01779784	17	IL	Illinois	00	G4000	A	143778515726	6216539665	+40.1028754
3	2	4	27	00662849	27	MN	Minnesota	00	G4000	A	206244837557	18937184315	+46.3159573
4	3	5	24	01714934	24	MD	Maryland	00	G4000	A	25151771744	6979295311	+38.9466584

```python
# To calculate a field, you specify the name for the new field and specify
# the calculation using the fields that will define the values of the new field
```

Question 3: Saving a dataset

```python
# Saving the data as a CSV file, with the newly calculated field included.
# This can be done with other supported file types, not just CSV

df2.to_file('fileout')
```

Question 4: Exporting to a csv/excel file

```python
df2.to_excel('us-states.xlsx')
```

```python
# Exporting files in geopandas works like it does in pandas. The code above uses
# the .to_excel() function to export the shapefile of U.S. states to an Excel file
```

Question 5: Taking a CSV that includes latitude and longitude and making it a GeoDataFrame

```python
# Reading in a csv file with the average latitude and longitude of countries

latlong = geopandas.read_file('average-latitude-longitude-countries.csv')

# Creating a geodataframe using the .GeoDataFrame method and the .points_from_xy()
# function to define the geometry
gdf = geopandas.GeoDataFrame(latlong,geometry=geopandas.points_from_xy
                             (latlong.Longitude,latlong.Latitude))
gdf
```

	ISO 3166 Country Code	Country	Latitude	Longitude	geometry
0	AD	Andorra	42.5	1.5	POINT (1.50000 42.50000)
1	AE	United Arab Emirates	24	54	POINT (54.00000 24.00000)
2	AF	Afghanistan	33	65	POINT (65.00000 33.00000)
3	AG	Antigua and Barbuda	17.05	-61.8	POINT (-61.80000 17.05000)
4	AI	Anguilla	18.25	-63.17	POINT (-63.17000 18.25000)
...	...	...	...	...	...
235	YE	Yemen	15	48	POINT (48.00000 15.00000)
236	YT	Mayotte	-12.83	45.17	POINT (45.17000 -12.83000)
237	ZA	South Africa	-29	24	POINT (24.00000 -29.00000)
238	ZM	Zambia	-15	30	POINT (30.00000 -15.00000)
239	ZW	Zimbabwe	-20	30	POINT (30.00000 -20.00000)

240 rows × 5 columns

Question 6: Joining data with Geopandas

```python
# Reading in two csv files, one for 2015 population by county and one for 2017
pop2015 = geopandas.read_file('acs2015_county_data.csv')
pop2017 = geopandas.read_file('acs2017_county_data.csv')

# Joining the two population sets at the 'CensusId' and 'CountyId' fields
pop_joined = pop2015.merge(pop2017, how='left', left_on='CensusId', right_on='CountyId')
pop_joined.head()
```

	CensusId	State_x	County_x	TotalPop_x	Men_x	Women_x	Hispanic_x	White_x	Black_x	Native_x	...	OtherTransp_y	WorkAtHome_y
0	1001	Alabama	Autauga	55221	26745	28476	2.6	75.8	18.5	0.4	...	1.3	2.5
1	1003	Alabama	Baldwin	195121	95314	99807	4.5	83.1	9.5	0.6	...	1.1	5.6
2	1005	Alabama	Barbour	26932	14497	12435	4.6	46.2	46.7	0.2	...	1.7	1.3
3	1007	Alabama	Bibb	22604	12073	10531	2.2	74.5	21.4	0.4	...	1.7	1.5
4	1009	Alabama	Blount	57710	28512	29198	8.6	87.9	1.5	0.3	...	0.4	2.1

5 rows × 76 columns

```python
pop2015.head()
```

	CensusId	State	County	TotalPop	Men	Women	Hispanic	White	Black	Native	...	OtherTransp	WorkAtHome	MeanCommute
0	1001	Alabama	Autauga	55221	26745	28476	2.6	75.8	18.5	0.4	...	1.3	1.8	26.5
1	1003	Alabama	Baldwin	195121	95314	99807	4.5	83.1	9.5	0.6	...	1.4	3.9	26.4
2	1005	Alabama	Barbour	26932	14497	12435	4.6	46.2	46.7	0.2	...	1.5	1.6	24.1
3	1007	Alabama	Bibb	22604	12073	10531	2.2	74.5	21.4	0.4	...	1.5	0.7	28.8
4	1009	Alabama	Blount	57710	28512	29198	8.6	87.9	1.5	0.3	...	0.4	2.3	34.9

5 rows × 38 columns

```python
pop2017.head()
```

	CountyId	State	County	TotalPop	Men	Women	Hispanic	White	Black	Native	...	OtherTransp	WorkAtHome	MeanCommute
0	1001	Alabama	Autauga County	55036	26899	28137	2.7	75.4	18.9	0.3	...	1.3	2.5	25.8
1	1003	Alabama	Baldwin County	203360	99527	103833	4.4	83.1	9.5	0.8	...	1.1	5.6	27.0
2	1005	Alabama	Barbour County	26201	13976	12225	4.2	45.7	47.8	0.2	...	1.7	1.3	23.4
3	1007	Alabama	Bibb County	22580	12251	10329	2.4	74.6	22.0	0.4	...	1.7	1.5	30.0
4	1009	Alabama	Blount County	57667	28490	29177	9.0	87.4	1.5	0.3	...	0.4	2.1	35.0

5 rows × 38 columns

```python
# Joining data with geopandas works similar to pandas, here I used df.merge()
# to join the two tables. You can also use pd.concat()
```

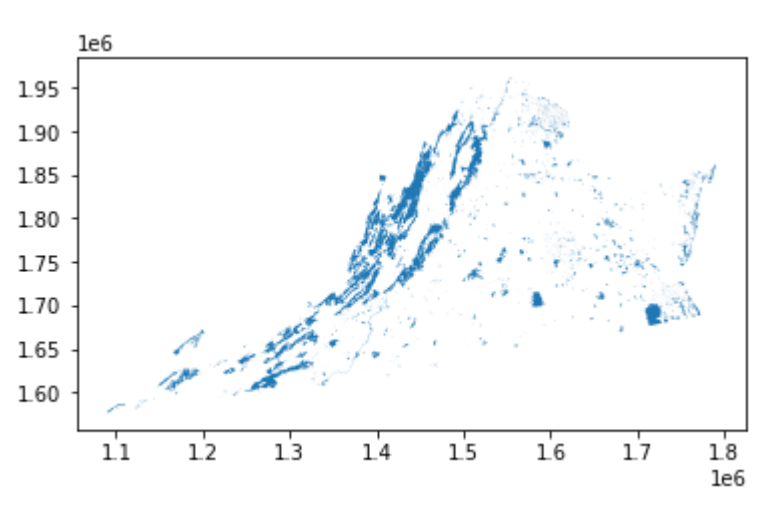
Question 7: Creating plots to display data

```python
# Plotting the GEOJson shapefile of the U.S. imported in Question 1
df3.plot()
```

<AxesSubplot:>

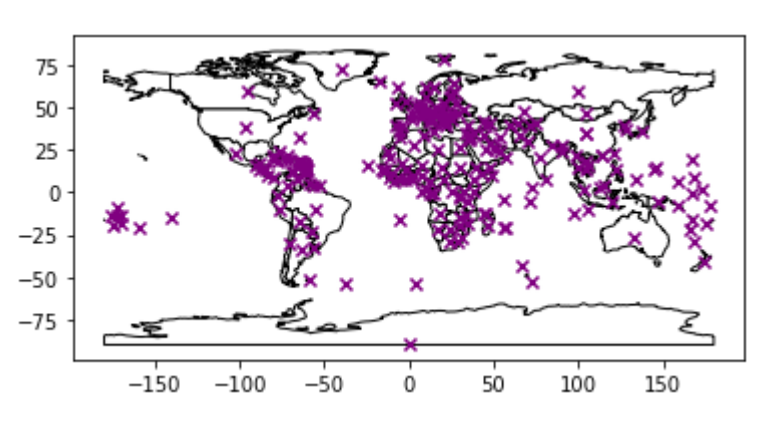
```python
# Plotting the geopackage file of portected areas in Virginia imported in
# Question 1
df4.plot()
```

<AxesSubplot:>



```python
# Plots the point of average latitude and longitude for every country and
# some territories

world = geopandas.read_file(geopandas.datasets.get_path('naturalearth_lowres'))
base = world.plot(color='white', edgecolor='black')
gdf.plot(ax=base,marker='x',color='purple')
```

<AxesSubplot:>

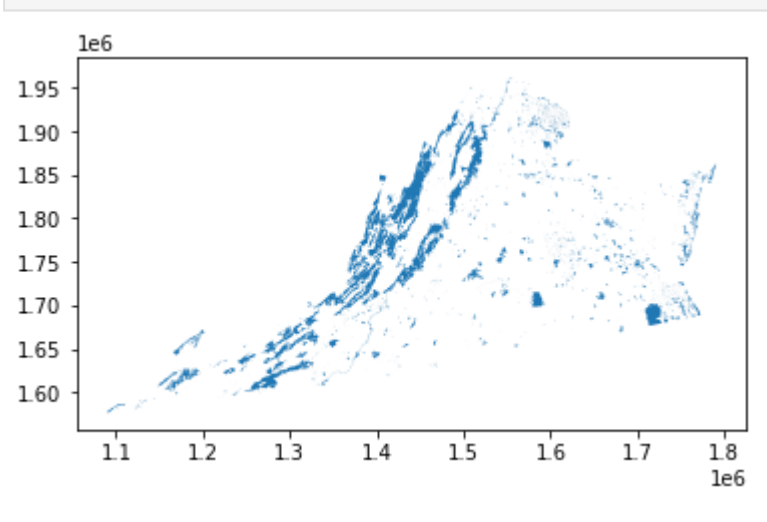


```python
# You can plot data with geopandas using the .plot() function. Here, I made
# two simple plots of files by using the .plot() function and nothing else.
# The last plot shows average lat/long location of countries, using the default
# geopandas world map as a base map.
```

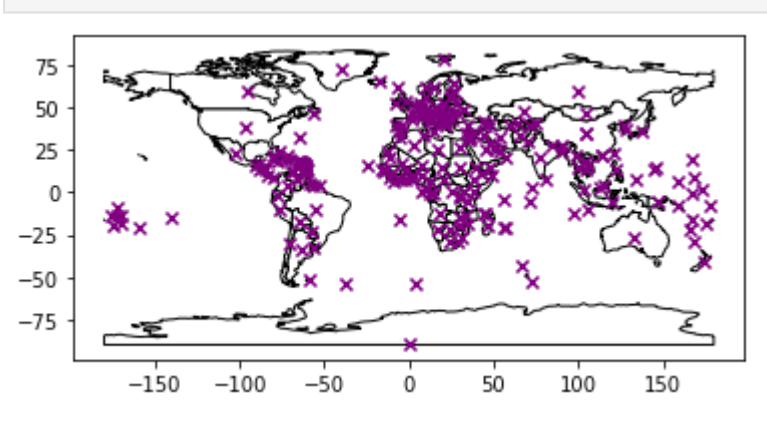
Question 8: Saving plots to a disk

```python
df3.plot()

plt.savefig('United_States.png', dpi=300, bbox_inches='tight', pad_inches=0)
```

```python
df4.plot()

plt.savefig('VA_Protected_Areas', dpi=300, bbox_inches='tight', pad_inches=0)
```



```python
world = geopandas.read_file(geopandas.datasets.get_path('naturalearth_lowres'))
base = world.plot(color='white', edgecolor='black')
gdf.plot(ax=base,marker='x',color='purple')

plt.savefig('avg_latlong_countries.png', dpi=300, bbox_inches='tight', pad_inches=0)
```

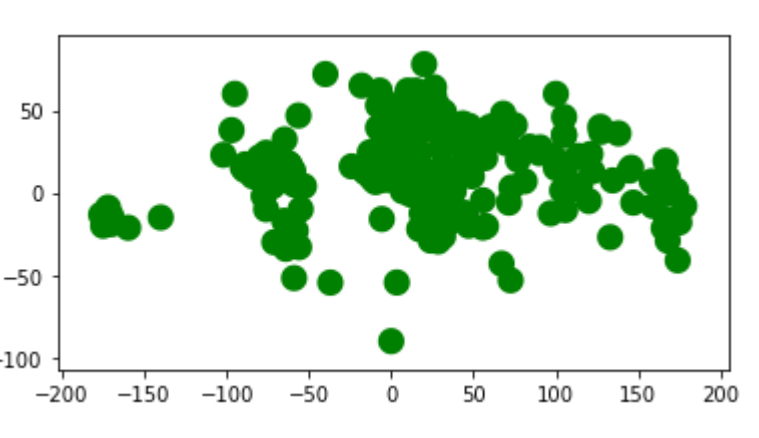


```python
# Plots made with geopandas can be saved using plt.savefig(). Here
# I saved the three plots I made, using bbox to specify the bounding
# area of the image and pad_inches to set no padding around the image area.
```

Question 9: Applying a buffer with Geopandas

```python
# Using the .buffer() function locates all points within a specified
# distance of an object. Here, the buffer locates all areas within 8 degrees
# of the average latitude and longitude points for each country/territory
# in the Geodataframe. Then, using matplotlib .plot() function, a plot of
# the points and their surrounding buffers is made.

gdf.buffer(distance=8).plot(color='green')
```

<AxesSubplot:>



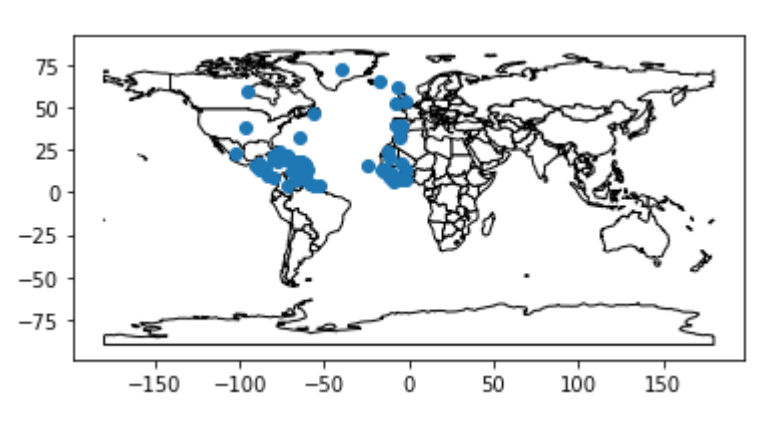
Question 10: Using Pandas indexing (.loc and .iloc) and spatial indexing (.cx) to find data

```python
# locates the first 10 rows in the Geodataframe and returns the
# names of the countries

gdf.iloc[:10]['Country']
```

```
0                 Andorra
1    United Arab Emirates
2             Afghanistan
3     Antigua and Barbuda
4                Anguilla
5                 Albania
6                 Armenia
7    Netherlands Antilles
8                  Angola
9     Asia/Pacific Region
Name: Country, dtype: object
```

```python
# Geopandas lets you locate data based on coordinate location using
# the .cx() index function. This code, using the geodataframe of each
# country's average latitude and longitude, locates the countries whose
# average latitude is north of the equator and whose average longitude
# is west of the prime meridian.

base = world.plot(color='white', edgecolor='black')
northwest = gdf.cx[:0, 0:]
northwest.plot(ax=base,)
```

<AxesSubplot:>



Question 11: Demonstrating a 'group' operation with Pandas syntax and doing a spatial 'group' using .dissolve

```python
# Using pandas to group the states by region

regions = covid_2020.groupby('State Region')
regions
```

<pandas.core.groupby.generic.DataFrameGroupBy object at 0x000002135D77A910>

```python
covid_2020 = geopandas.read_file('1-1-21 US covid19.csv')
covid_2020.head()
```

	State/Territory	Total Cases	Confirmed Cases	Probable Cases	Cases in Last 7 Days	Case Rate per 100000	Total Deaths	Confirmed Deaths	Probable Deaths	Deaths in Last 7 Days	Death Rate per 100000	State Region	geometry
0	Alaska	44966	null	null	1605	6147	202	null	null	6	27	West	None
1	Alabama	356820	287173	69647	22251	7277	4774	4174	600	187	97	South	None
2	Arkansas	222430	null	null	14489	7371	3637	null	null	261	120	South	None
3	American Samoa	3	null	null	0	5	0	null	null	0	0	Other	None
4	Arizona	512489	488303	24186	39216	7041	8718	7892	826	539	119	West	None

```python
# Using the .dissolve() function, the states and territories are all
# grouped into their respective region. Then, using .loc, I display
# the column with how many states/territories are in each region.

regions = covid_2020.dissolve(by='State Region', aggfunc='count')
regions.loc[:,['State/Territory']]
```

State Region	State/Territory
Midwest	12
Northeast	10
Other	8
South	17
West	13