Assignment 8

Revisit assignment 2 and finish the tasks using pandas and geopandas, instead.

The tasks are:

• Find top n most common county names (n = 3 or 5, for example)

• Derive summary statistics for states, for example, number of counties, counties with max/min area

• Join fips code to get full names of states (plesae do a real join using merge)

• Plot maps to show all the counties with the most common names from first task

In [42]:
```
import json
import fiona
import pandas as pd
import geopandas as gpd
import matplotlib as mb
```

In [43]:
```
# First I downloaded the file and used the command to raed that file using geopandas:
cntyDat = gpd.read_file(r"C:\Users\njanjic\Downloads\gz_2010_us_050_00_20m-min.json", d
cntyDat
```

Out[43]:

	GEO_ID	STATE	COUNTY	NAME	LSAD	CENSUSAREA	geometry
0	0500000US01001	01	001	Autauga	County	594.436	POLYGON ((-86.49677 32.34444, -86.71790 32.402...
1	0500000US01009	01	009	Blount	County	644.776	POLYGON ((-86.57780 33.76532, -86.75914 33.840...
2	0500000US01017	01	017	Chambers	County	596.531	POLYGON ((-85.18413 32.87053, -85.12342 32.772...
3	0500000US01021	01	021	Chilton	County	692.854	POLYGON ((-86.51734 33.02057, -86.51596 32.929...
4	0500000US01033	01	033	Colbert	County	592.619	POLYGON ((-88.13999 34.58170, -88.13925 34.587...
...	...	...	...	...	...	...	...
3216	0500000US51001	51	001	Accomack	County	449.496	MULTIPOLYGON (((-75.24227 38.02721, -75.29687 ...
3217	0500000US51021	51	021	Bland	County	357.725	POLYGON ((-81.22510 37.23487, -81.20477 37.243...
3218	0500000US51027	51	027	Buchanan	County	502.763	POLYGON ((-81.96830 37.53780, -81.92787

	GEO_ID	STATE	COUNTY	NAME	LSAD	CENSUSAREA	geometry
							37.512...
3219	0500000US51037	51	037	Charlotte	County	475.271	POLYGON ((-78.44332 37.07940, -78.49303 36.891...
3220	0500000US51041	51	041	Chesterfield	County	423.297	POLYGON ((-77.85180 37.35487, -77.85515 37.418...

3221 rows × 7 columns

In [44]:
```
# Using the command with...as... I will now open the file which contains FIPS codes for
# FIPS stands for The Federal Information Processing Standard code.
with open(r"C:\Users\njanjic\Downloads\fipsToState.json", 'r') as f:
    fipData = json.load(f)
```

In [45]:
```
cntyDat.head(5)
```

Out[45]:

	GEO_ID	STATE	COUNTY	NAME	LSAD	CENSUSAREA	geometry
0	0500000US01001	01	001	Autauga	County	594.436	POLYGON ((-86.49677 32.34444, -86.71790 32.402...
1	0500000US01009	01	009	Blount	County	644.776	POLYGON ((-86.57780 33.76532, -86.75914 33.840...
2	0500000US01017	01	017	Chambers	County	596.531	POLYGON ((-85.18413 32.87053, -85.12342 32.772...
3	0500000US01021	01	021	Chilton	County	692.854	POLYGON ((-86.51734 33.02057, -86.51596 32.929...
4	0500000US01033	01	033	Colbert	County	592.619	POLYGON ((-88.13999 34.58170, -88.13925 34.587...

In [48]:
```
# In my next step I will install folium, matplotlib and mapclassify,
# which are necessary for explore() command.
pip install folium
```

```
Collecting folium
  Downloading folium-0.14.0-py2.py3-none-any.whl (102 kB)
Requirement already satisfied: requests in c:\programdata\anaconda3\lib\site-packages (f
rom folium) (2.26.0)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (from
folium) (1.20.3)
Collecting branca>=0.6.0
  Downloading branca-0.6.0-py3-none-any.whl (24 kB)
Requirement already satisfied: jinja2>=2.9 in c:\programdata\anaconda3\lib\site-packages
(from folium) (2.11.3)
Requirement already satisfied: MarkupSafe>=0.23 in c:\programdata\anaconda3\lib\site-pac
kages (from jinja2>=2.9->folium) (1.1.1)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\lib\site-package
s (from requests->folium) (3.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anaconda3\lib\site-p
```

```
ackages (from requests->folium) (2021.10.8)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\programdata\anaconda3\lib\sit
e-packages (from requests->folium) (1.26.7)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\programdata\anaconda3\lib
\site-packages (from requests->folium) (2.0.4)
Installing collected packages: branca, folium
Successfully installed branca-0.6.0 folium-0.14.0
Note: you may need to restart the kernel to use updated packages.
```

In [49]:

```
pip install matplotlib
```

```
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\lib\site-packages
(3.4.3)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\site-package
s (from matplotlib) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\anaconda3\lib\site-pa
ckages (from matplotlib) (1.3.1)
Requirement already satisfied: numpy>=1.16 in c:\programdata\anaconda3\lib\site-packages
(from matplotlib) (1.20.3)
Requirement already satisfied: python-dateutil>=2.7 in c:\programdata\anaconda3\lib\site
-packages (from matplotlib) (2.8.2)
Requirement already satisfied: pyparsing>=2.2.1 in c:\programdata\anaconda3\lib\site-pac
kages (from matplotlib) (3.0.4)
Requirement already satisfied: pillow>=6.2.0 in c:\programdata\anaconda3\lib\site-packag
es (from matplotlib) (8.4.0)
Requirement already satisfied: six in c:\programdata\anaconda3\lib\site-packages (from c
ycler>=0.10->matplotlib) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [51]:

```
pip install mapclassify
```

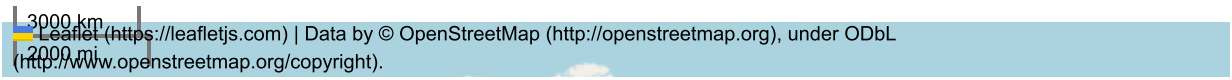
```
Collecting mapclassifyNote: you may need to restart the kernel to use updated packages.

  Downloading mapclassify-2.5.0-py3-none-any.whl (39 kB)
Requirement already satisfied: numpy>=1.3 in c:\programdata\anaconda3\lib\site-packages
(from mapclassify) (1.20.3)
Requirement already satisfied: pandas>=1.0 in c:\programdata\anaconda3\lib\site-packages
(from mapclassify) (1.3.4)
Requirement already satisfied: scipy>=1.0 in c:\programdata\anaconda3\lib\site-packages
(from mapclassify) (1.7.1)
Requirement already satisfied: scikit-learn in c:\programdata\anaconda3\lib\site-package
s (from mapclassify) (0.24.2)
Requirement already satisfied: networkx in c:\programdata\anaconda3\lib\site-packages (f
rom mapclassify) (2.6.3)
Requirement already satisfied: pytz>=2017.3 in c:\programdata\anaconda3\lib\site-package
s (from pandas>=1.0->mapclassify) (2021.3)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\programdata\anaconda3\lib\si
te-packages (from pandas>=1.0->mapclassify) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-packages (f
rom python-dateutil>=2.7.3->pandas>=1.0->mapclassify) (1.16.0)
Requirement already satisfied: joblib>=0.11 in c:\programdata\anaconda3\lib\site-package
s (from scikit-learn->mapclassify) (1.1.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\programdata\anaconda3\lib\site
-packages (from scikit-learn->mapclassify) (2.2.0)
Installing collected packages: mapclassify
Successfully installed mapclassify-2.5.0
```

In [52]:

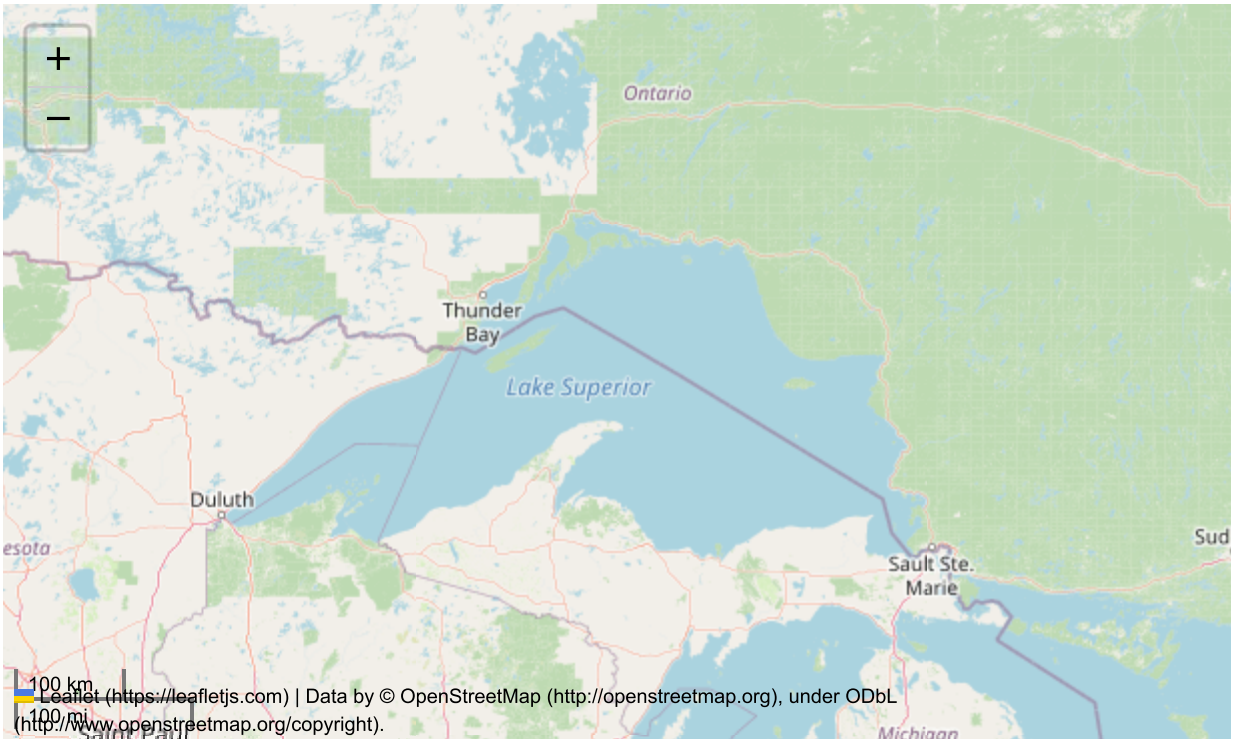
```
cntyDat.explore()
```

Out[52]: Make this Notebook Trusted to load map: File -> Trust Notebook



In [53]:
```
# If I want to represent just one state, not all of them, then I will perform:
cntyDat.query("STATE == '36'").explore()
```

Out[53]:

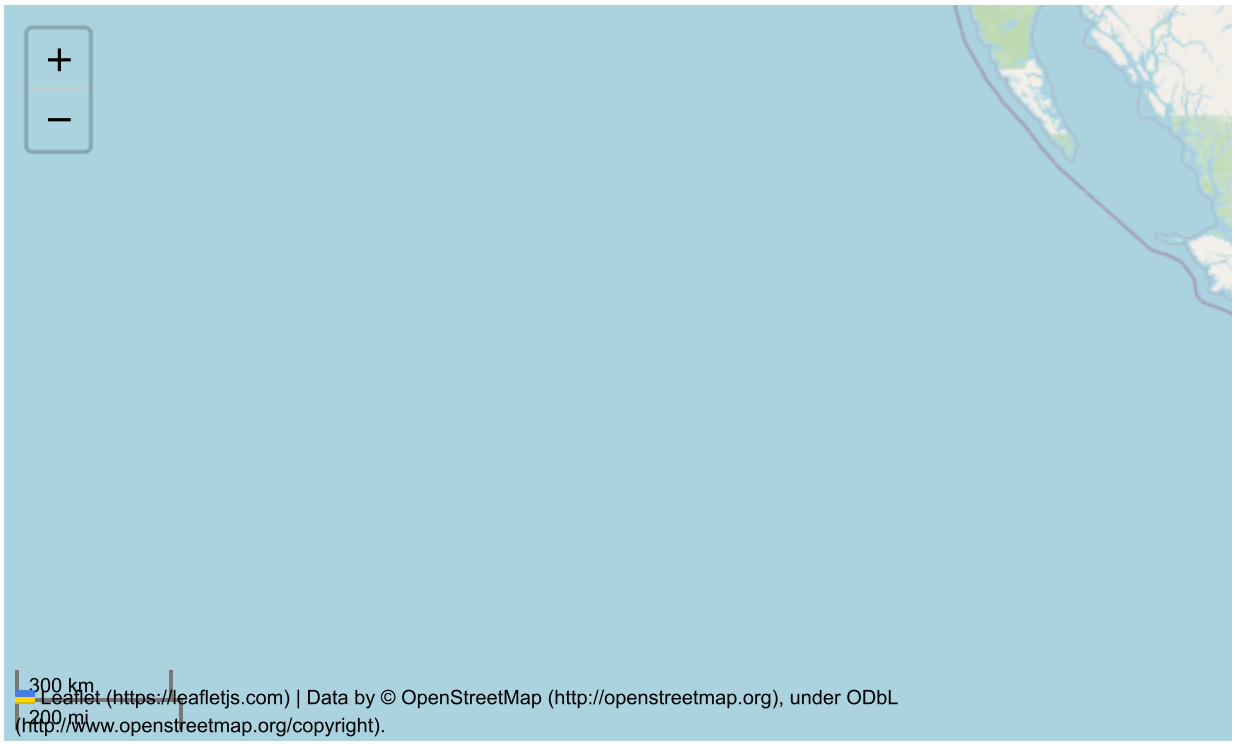


In [55]:
```
# The above one represents New York state.
# If I want to represent Alaska, I will look up for the state code list in my file.
# The next 2 examples are Alaska and Nevada.
cntyDat.query("STATE == '02'").explore()
```

Out[55]: Make this Notebook Trusted to load map: File -> Trust Notebook

In [56]:
```
cntyDat.query("STATE == '32'").explore()
```

Out[56]:



In [57]:
```
# Extracting and visualizing the specific counties from the state:
nycCntyCodes =['061', '057']
cntyDat[(cntyDat.STATE == '36') & cntyDat['COUNTY'].isin (nycCntyCodes)].explore()
```

Out[57]:

In [59]:
```
# As a result above I get 2 counties: New York and Montgomery.
# Basically, for columns I will use command filter and for rows I will use command quer
```

```
# Since 2 selected counties have too many data, I want to see only the name and geometr
# Therefore, I need to perform follwing:
cntyDat.loc[(cntyDat.STATE == '36') & cntyDat['COUNTY'].isin(nycCntyCodes)][['NAME', 'g
```

Out[59]:

In [60]:

```
# With command loc above I chose a specific columns of interest.
# However, it is possible to perform subsetting w/o using loc command:
cntyDat[(cntyDat.STATE == '36') & cntyDat['COUNTY'].isin(nycCntyCodes)][['NAME', 'geome
```

Out[60]:

In [61]:

```
# Converting JSON dictionary to pandas DataFrame (columns are "fip" and "st_name"):
pd.DataFrame.from_dict(fipData, orient="index", columns=['st_name'])\
```

```
    .rename_axis('st_fip')\
    .reset_index()
```

Out[61]:

	st_fip	st_name
0	01	Alabama
1	02	Alaska
2	04	Arizona
3	05	Arkansas
4	06	California
5	08	Colorado
6	09	Connecticut
7	10	Delaware
8	11	District of Columbia
9	12	Florida
10	13	Georgia
11	15	Hawaii
12	16	Idaho
13	17	Illinois
14	18	Indiana
15	19	Iowa
16	20	Kansas
17	21	Kentucky
18	22	Louisiana
19	23	Maine
20	24	Maryland
21	25	Massachusetts
22	26	Michigan
23	27	Minnesota
24	28	Mississippi
25	29	Missouri
26	30	Montana
27	31	Nebraska
28	32	Nevada
29	33	New Hampshire
30	34	New Jersey
31	35	New Mexico

In [64]:
```
pd.DataFrame([(k, v) for k,v in fipData.items()], columns=['fip', 'state'])
```

Out[64]:

	fip	state
8	01	Alabama
1	02	Alaska
2	04	Arizona
3	05	Arkansas
4	06	California
5	08	Colorado
6	09	Connecticut
7	10	Delaware
[illegible]	[illegible]	District of Columbia
9	12	Florida
10	13	Geogia

	fip	stat
11	15	Hawaii
12	16	Idaho
13	17	Illinois
14	18	Indiana
15	19	Iowa
16	20	Kansas
17	21	Kentucky
18	22	Louisiana
19	23	Maine
20	24	Maryland
21	25	Massachusetts
22	26	Michigan
23	27	Minnesota
24	28	Mississippi
25	29	Missouri
26	30	Montana
27	31	Nebraska
28	32	Nevada
29	33	New Hampshire
30	34	New Jersey
31	35	New Mexico
32	36	New York
33	37	North Carolina
34	38	North Dakota
35	39	Ohio
36	40	Oklahoma
37	41	Oregon
38	42	Pennsylvania
39	44	Rhode Island
40	45	South Carolina
41	46	South Dakota
42	47	Tennessee
43	48	Texas
44	49	Utah

	fip	state
45	50	Vermont
46	51	Virginia
47	53	Washington
48	54	West Virginia
49	55	Wisconsin
50	56	Wyoming

In [65]:
```
list(zip(fipData.keys(), fipData.values()))
```

Out[65]:
```
[('01', 'Alabama'),
 ('02', 'Alaska'),
 ('04', 'Arizona'),
 ('05', 'Arkansas'),
 ('06', 'California'),
 ('08', 'Colorado'),
 ('09', 'Connecticut'),
 ('10', 'Delaware'),
 ('11', 'District of Columbia'),
 ('12', 'Florida'),
 ('13', 'Geogia'),
 ('15', 'Hawaii'),
 ('16', 'Idaho'),
 ('17', 'Illinois'),
 ('18', 'Indiana'),
 ('19', 'Iowa'),
 ('20', 'Kansas'),
 ('21', 'Kentucky'),
 ('22', 'Louisiana'),
 ('23', 'Maine'),
 ('24', 'Maryland'),
 ('25', 'Massachusetts'),
 ('26', 'Michigan'),
 ('27', 'Minnesota'),
 ('28', 'Mississippi'),
 ('29', 'Missouri'),
 ('30', 'Montana'),
 ('31', 'Nebraska'),
 ('32', 'Nevada'),
 ('33', 'New Hampshire'),
 ('34', 'New Jersey'),
 ('35', 'New Mexico'),
 ('36', 'New York'),
 ('37', 'North Carolina'),
 ('38', 'North Dakota'),
 ('39', 'Ohio'),
 ('40', 'Oklahoma'),
 ('41', 'Oregon'),
 ('42', 'Pennsylvania'),
 ('44', 'Rhode Island'),
 ('45', 'South Carolina'),
 ('46', 'South Dakota'),
 ('47', 'Tennessee'),
 ('48', 'Texas'),
```

```
('49', 'Utah'),
('50', 'Vermont'),
('51', 'Virginia'),
('53', 'Washington'),
('54', 'West Virginia'),
('55', 'Wisconsin'),
('56', 'Wyoming')]
```

In [66]:
```
# The lenght of the file:
len(cntyDat)
```

Out[66]:
```
3221
```

In [67]:
```
cntyDat.head(5)
cntyDat.dtypes
```

Out[67]:
```
GEO_ID          object
STATE           object
COUNTY          object
NAME            object
LSAD            object
CENSUSAREA     float64
geometry      geometry
dtype: object
```

In [68]:
```
# Joining pandas DataFrames:
fipDF=pd.DataFrame([(k, v) for k,v in fipData.items()], columns=['fip', 'stName'])
fipDF.head
```

Out[68]:
```
<bound method NDFrame.head of    fip                stName
0   01                Alabama
1   02                 Alaska
2   04                Arizona
3   05               Arkansas
4   06             California
5   08               Colorado
6   09            Connecticut
7   10               Delaware
8   11  District of Columbia
9   12                Florida
10  13                 Geogia
11  15                 Hawaii
12  16                  Idaho
13  17               Illinois
14  18                Indiana
15  19                   Iowa
16  20                 Kansas
17  21               Kentucky
18  22              Louisiana
19  23                  Maine
20  24               Maryland
21  25          Massachusetts
22  26               Michigan
23  27              Minnesota
24  28            Mississippi
25  29               Missouri
26  30                Montana
```

```
27  31              Nebraska
28  32                Nevada
29  33         New Hampshire
30  34            New Jersey
31  35            New Mexico
32  36              New York
33  37        North Carolina
34  38          North Dakota
35  39                  Ohio
36  40              Oklahoma
37  41                Oregon
38  42          Pennsylvania
39  44          Rhode Island
40  45        South Carolina
41  46          South Dakota
42  47             Tennessee
43  48                 Texas
44  49                  Utah
45  50               Vermont
46  51              Virginia
47  53            Washington
48  54         West Virginia
49  55             Wisconsin
50  56               Wyoming>
```

In [69]:

```
#The first five rows containing fip and stName:
fipDF.head(5)
```

Out[69]:

	fip	stName
0	01	Alabama
1	02	Alaska
2	04	Arizona
3	05	Arkansas
4	06	California

In [101…

```
# Now I will merge my datasets with a condition: columns STATE and COUNTY on the right
cntyDat.merge(fipDF, left_on='STATE', right_on='fip')
```

Out[101…

	GEO_ID	STATE	COUNTY	NAME	LSAD	CENSUSAREA	geometry	fip	stNam
0	0500000US01001	01	001	Autauga	County	594.436	POLYGON ((-86.49677 32.34444, -86.71790 32.402...	01	Alaban
1	0500000US01009	01	009	Blount	County	644.776	POLYGON ((-86.57780 33.76532, -86.75914 33.840...	01	Alaban
2	0500000US01017	01	017	Chambers	County	596.531	POLYGON ((-85.18413	01	Alaban

	GEO_ID	STATE	COUNTY	NAME	LSAD	CENSUSAREA	geometry	fip	stNam
							32.87053, -85.12342 32.772...		
3	0500000US01021	01	021	Chilton	County	692.854	POLYGON ((-86.51734 33.02057, -86.51596 32.929...	01	Alaban
4	0500000US01033	01	033	Colbert	County	592.619	POLYGON ((-88.13999 34.58170, -88.13925 34.587...	01	Alaban
...	...	...	...	...	...	...	...	...	
3138	0500000US44001	44	001	Bristol	County	24.164	POLYGON ((-71.22480 41.71050, -71.22787 41.705...	44	Rhoc Islar
3139	0500000US44003	44	003	Kent	County	168.528	POLYGON ((-71.78968 41.72457, -71.45534 41.732...	44	Rhoc Islar
3140	0500000US44005	44	005	Newport	County	102.386	MULTIPOLYGON (((-71.38359 41.46478, -71.38928 ...	44	Rhoc Islar
3141	0500000US44007	44	007	Providence	County	409.502	POLYGON ((-71.79682 41.92855, -71.79924 42.008...	44	Rhoc Islar
3142	0500000US44009	44	009	Washington	County	329.235	MULTIPOLYGON (((-71.58955 41.19656, -71.58023 ...	44	Rhoc Islar

3143 rows × 9 columns

In [102...

```
# The desrepancy in the total number of rows (3143 instead of 3221) is because there ar
# So I have to do a little different:
cntyDatMerged = cntyDat.merge(fipDF, how='left', left_on='STATE', right_on='fip')
cntyDatMerged
```

Out[102...

	GEO_ID	STATE	COUNTY	NAME	LSAD	CENSUSAREA	geometry	fip	stNam
0	0500000US01001	01	001	Autauga	County	594.436	POLYGON ((-86.49677 32.34444,	01	Alaban

	GEO_ID	STATE	COUNTY	NAME	LSAD	CENSUSAREA	geometry	fip	stNam
							-86.71790 32.402...		
1	0500000US01009	01	009	Blount	County	644.776	POLYGON ((-86.57780 33.76532, -86.75914 33.840...	01	Alabam
2	0500000US01017	01	017	Chambers	County	596.531	POLYGON ((-85.18413 32.87053, -85.12342 32.772...	01	Alabam
3	0500000US01021	01	021	Chilton	County	692.854	POLYGON ((-86.51734 33.02057, -86.51596 32.929...	01	Alabam
4	0500000US01033	01	033	Colbert	County	592.619	POLYGON ((-88.13999 34.58170, -88.13925 34.587...	01	Alabam
...	...	...	...	...	...	...	...	...	
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	MULTIPOLYGON (([illegible] -75.24227 38.62721, -75.25687 ...	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	POLYGON ((-81.22519 37.23487, -81.26477 37.243...	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	POLYGON ((-81.96830 37.53785, -81.92787 37.512...	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	POLYGON ((-78.44222 37.07540, -78.43383 36.891...	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	POLYGON ((-77.55190 37.35487, -77.55515 37.418...	[illegible]	[illegible]

3221 rows × 9 columns

In [103...

```
# Now we have a total match.
# However, in order to show all data that do not have a match for fip:
cntyDatMerged.loc[cntyDatMerged['fip'].isna()]
```

Out[103...

	GEO.ID	STATE	COUNTY	NAME	LSAD	CENSUSAREA	geometry	fip	stfips
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	POLYGON ((-66.58678 18.48495, -66.58818 18.489...	NaN	NaN
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	POLYGON ((-66.31485 18.15269, -66.32234 18.165...	NaN	NaN
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	POLYGON ((-66.06610 18.36332, -66.05274 18.304...	NaN	NaN
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	POLYGON ((-66.03656 18.43022, -66.03547 18.450...	NaN	NaN
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	POLYGON ((-66.16913 18.22725, -66.24046 18.184...	NaN	NaN
...	...	...	...	...	...	...	...	...	...
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	POLYGON ((-66.67049 18.15263, -66.69375 18.215...	NaN	NaN
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	POLYGON ((-66.95752 18.40917, -66.92202 18.393...	NaN	NaN
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	POLYGON ((-67.18285 18.31303, -67.23868 18.375...	NaN	NaN
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	POLYGON ((-66.23145 18.09521, -66.22953 18.095...	NaN	NaN

78 rows × 9 columns

```
# The total number of data w/o fip is 78, which is correct because 3221-3143=78.
```

In [105…

```
# Counting every single column:
cntyDatMerged.groupby('NAME').count()
```

Out[105…

1909 rows × 8 columns

In [106…

```
# Top5 most common counties:
cntyDatMerged.groupby('NAME').count().sort_values(by=['stName'], ascending = False).hea
```

Out[106…

In [115…

```
cntyDatMerged[cntyDatMerged['NAME'] == 'Washington']
cntyDatMerged[cntyDatMerged['NAME'].isin(['Washington'])]
cntyDatMerged.loc[cntyDatMerged['NAME'] == 'Washington']
```

Out[115…

	GEO_ID	STATE	COUNTY	NAME	LSAD	CENSUSAREA	geometry	fip	stl
[illegible]	[illegible]	[illegible]	[illegible]	Washington	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	Washington	[illegible]	[illegible]	[illegible]	[illegible]	G
196	0500000US21229	21	229	Washington	County	297.266	POLYGON ((-85.40178 37.73066, -85.29904 37.831...	21	Ker
271	0500000US23029	23	029	Washington	County	2562.660	MULTIPOLYGON (((-67.61976 44.51975, -67.61541 ...	23	I
[illegible]	[illegible]	[illegible]	[illegible]	Washington	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	Washington	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	Washington	[illegible]	[illegible]	[illegible]	[illegible]	enn
[illegible]	[illegible]	[illegible]	[illegible]	Washington	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	Washington	[illegible]	[illegible]	[illegible]	[illegible]	Ark

	GEO_ID	STATE	COUNTY	NAME	LSAD	CENSUSAREA	geometry	fip	stl
823	0500000US08121	08	121	Washington	County	2518.031	POLYGON ((-103.70570 40.00137, -103.47199 40.0...	08	Col
895	0500000US18175	18	175	Washington	County	513.725	POLYGON ((-85.99462 38.41835, -86.01414 38.419...	18	In
[illegible]	[illegible]	[illegible]	[illegible]	Washington	County	[illegible]	POLYGON ((-91.48572 41.16240, -91.71582 41.162...	[illegible]	
[illegible]	[illegible]	[illegible]	[illegible]	Washington	County	[illegible]	POLYGON ((-92.74614 45.29605, -92.78174 45.292...	[illegible]	Minr
1120	0500000US24043	24	043	Washington	County	457.780	POLYGON ((-77.82541 39.49404, -77.84511 39.498...	24	Ma
[illegible]	[illegible]	[illegible]	[illegible]	Washington	County	[illegible]	POLYGON ((-91.37127 39.34206, -91.37936 39.341...	[illegible]	
[illegible]	[illegible]	[illegible]	[illegible]	Washington	County	[illegible]	POLYGON ((-96.64068 39.29679, -96.38618 39.285...	[illegible]	
[illegible]	[illegible]	[illegible]	[illegible]	Washington	County	[illegible]	POLYGON ((-80.51922 39.96220, -80.51922 39.962...	[illegible]	Pennsy
[illegible]	[illegible]	[illegible]	[illegible]	Washington	Parish	[illegible]	POLYGON ((-90.34724 31.89825, -90.34724 31.898...	[illegible]	Lou
[illegible]	[illegible]	[illegible]	[illegible]	Washington	County	[illegible]	POLYGON ((-76.69238 35.94534, -76.89179 35.944...	[illegible]	Ca

	GEO_ID	STATE	COUNTY	NAME	LSAD	CENSUSAREA	geometry	fip	stl
1916	0500000US01129	01	129	Washington	County	1080.207	POLYGON ((-88.43201 31.11430, -88.43810 31.230...	01	Ala
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	Neb
2733	0500000US44009	44	009	Washington	County	329.235	MULTIPOLYGON (((-71.58955 41.19656, -71.58023 ...	44	R
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	Nev
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	Wisc
3162	0500000US20201	20	201	Washington	County	894.756	POLYGON ((-97.36920 40.00206, -97.36920 40.002...	20	K
3186	0500000US49053	49	053	Washington	County	2426.358	POLYGON ((-112.89937	49	

	GEO_ID	STATE	COUNTY	NAME	LSAD	CENSUSAREA	geometry	fip	stl
3190	0500000US50023	50	023	Washington	County	687.233	POLYGON ((-72.82611 44.35919, -72.80500 44.451...	50	Ver

In [114...]

```
# For some reason this code did not work well and I could not get the Top5 counties in
type(cntyDatMerged.groupby('NAME').size())
cntyDatMerged.groupby('NAME')size().sort_values(ascending=False).head(5)
```

```
  File "C:\Users\njanjic\AppData\Local\Temp/ipykernel_11580/390077300.py", line 2
    cntyDatMerged.groupby('NAME')size().sort_values(ascending=False).head(5)
                                 ^
SyntaxError: invalid syntax
```

In [113...]

```
cntyDatMerged.groupby('NAME').size().nlargest(5)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_11580/2510764956.py in <module>
----> 1 cntyDatMerged.groupby('NAME').size().nlargest(5)

C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\groupby\groupby.py in size(self)
   1826
   1827         # GH28330 preserve subclassed Series/DataFrames through calls
-> 1828         if issubclass(self.obj._constructor, Series):
   1829             result = self._obj_1d_constructor(result, name=self.obj.name)
   1830         else:

TypeError: issubclass() arg 1 must be a class
```

In [116...]

```
import plotly.express as px
import plotly.io as pio
```

In [ ]:

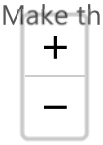
```
cntyDatMerged = px.pie(file, values="")
```

In [119...]

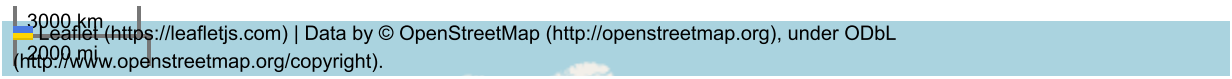
```
mobj = cntyDatMerged.explore()
washingtonCnties.explore(m = mobj, style_kwds = {'color': 'yellow'})
```

Out[119...]

Make this Notebook Trusted to load map: File -> Trust Notebook



In [ ]:


3000 km
2000 mi
Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).