DataEng S24: Data Integration In-class Assignment

This week you will gain hands-on experience with Data Integration by combining data from two distinct sources into a unified DataFrame for analysis.

Submit: Make a copy of this document and use it to record your responses and results. Use colored highlighting for your responses and results. Store a PDF copy of the document in your git repository along with any needed code before submitting for this week.

Your job is to integrate **county-level COVID-19 data** with the **ACS Census Tract data for 2017** to build a model that allows you to relate COVID numbers with economic data such as population, **per capita income and poverty level. To do this you should build two pandas DataFrames as follows.**

County_Info: demographic summaries for each county. This table should have one row per county (there are more than 3000 counties in the USA), and each row must include the following columns:

Name - name of the county
State - name of the state in which the county resides
Population - population of this county
Poverty - % of people in poverty in this county
PerCapitaIncome - per capita personal income for this county
ID - a unique integer ID for the county. You may choose this ID any way you like, it just needs to be unique among all counties and needs to be referenced by the COVID_monthly rows (foreign key lookup).

COVID_monthly: data about COVID cases and deaths for each USA county.

ID - refers to the county found in the County_info table (i.e., this is a foreign key lookup)
Month - integer in range 1..12 indicating the month of the year
Cases - number of COVID cases recorded for the corresponding county during the corresponding month
Deaths - number of COVID-related deaths recorded for the corresponding county during the corresponding month

For this activity you should use whichever development environment is convenient for you to develop with python and pandas. Google Colab and Jupyter are good choices. You are not required to use GCP, but you can use it if you prefer.

Submit: by Friday at 10pm

A. [MUST] Discussion Question

Within your group, identify two or more sources of data that might be integrated to analyze a problem that could not be easily analyzed or solved otherwise. The data sources and problems do not need to be related to anything we have done in class and do not even need to be serious. Be imaginative; you do not need to describe how to integrate the data sets.
ans)

Satellite Imagery and Agricultural Yield Data:
Problem: Predicting crop yields and identifying factors influencing agricultural productivity.
Data Sources:

Satellite imagery captures agricultural aspects like vegetation health, soil moisture, and land use patterns. Combining this with yield data from farms can reveal the impact of environmental factors on crop productivity. Analyzing vegetation and soil moisture alongside yield data can enhance models for predicting future yields. This integration offers insights to optimize agricultural practices.

For example, you might say something like, “Integrate historic weather/precipitation data with crop yield data to help develop a system for anticipating future prices of corn and wheat.”

<insert your own ideas here and discuss them with your work group>

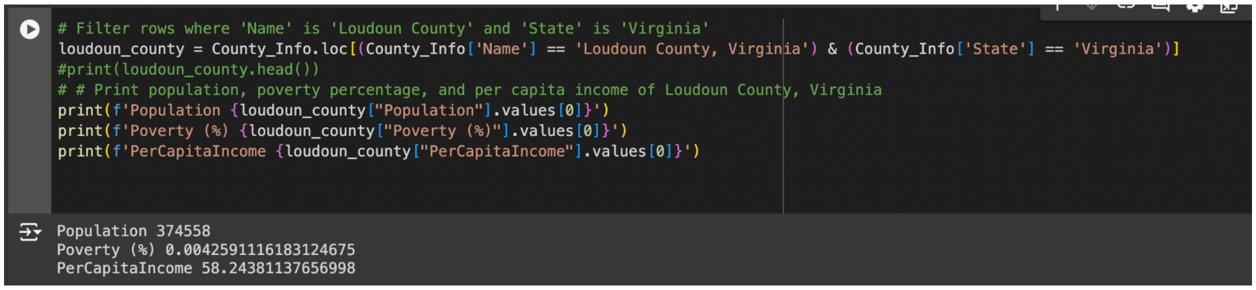
B. [MUST] Transform the ACS Data

The ACS data is separated into “Census Tracts” which are regions within counties that correspond to groups of approximately 4000 people. The Census Bureau defines these to help organize the actual job of collecting census data. I.e., they did it this way for their convenience, not yours, and this grouping makes your Data Engineering job more challenging. Your “dimension of overlap” is the county not the census tract, so you must transform the ACS data.

To properly integrate the ACS with the COVID data, aggregate the per-tract data to the county level of resolution.

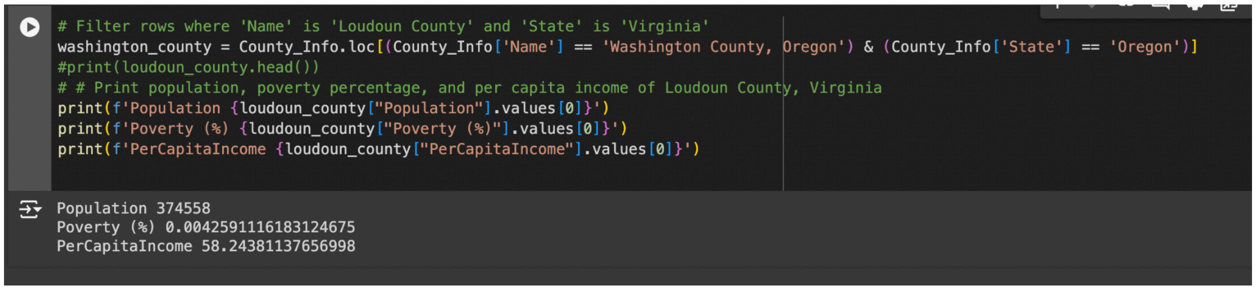
Create a python+pandas program that transforms the ACS data into a one-row-per-county ACS DataFrame called County_info. To do this you will need to think about how to properly aggregate Census Tract-level data into County-level summaries. Your transformation code should also eliminate unneeded columns from the ACS data.

Also, add an ID column to your County_info dataframe so that the county can be referred to by the COVID data.



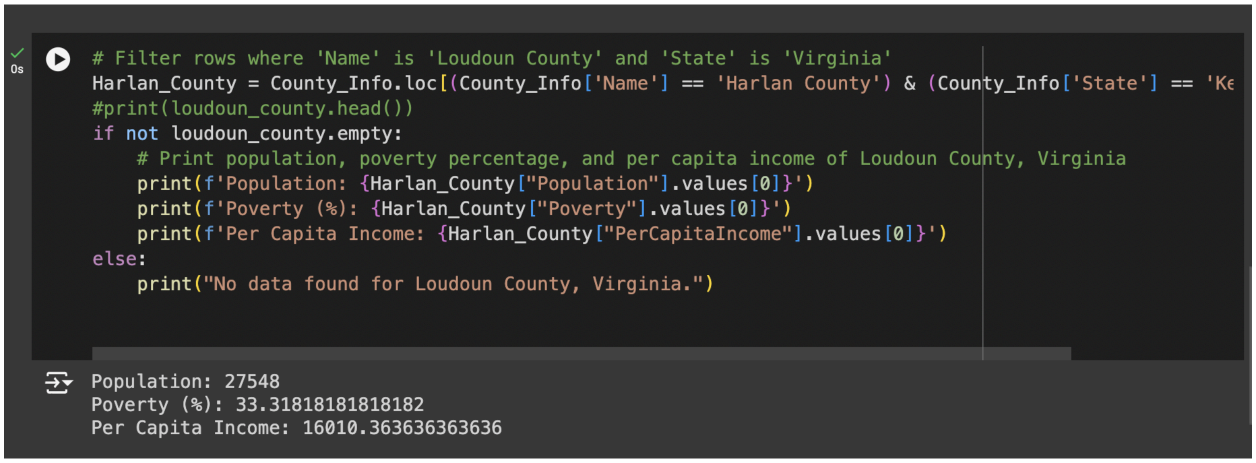
```
# Filter rows where 'Name' is 'Loudoun County' and 'State' is 'Virginia'
loudoun_county = County_Info.loc[(County_Info['Name'] == 'Loudoun County, Virginia') & (County_Info['State'] == 'Virginia')]
#print(loudoun_county.head())
# # Print population, poverty percentage, and per capita income of Loudoun County, Virginia
print(f'Population {loudoun_county["Population"].values[0]}')
print(f'Poverty (%) {loudoun_county["Poverty (%)"].values[0]}')
print(f'PerCapitaIncome {loudoun_county["PerCapitaIncome"].values[0]}')
```

```
Population 374558
Poverty (%) 0.0042591116183124675
PerCapitaIncome 58.24381137656998
```



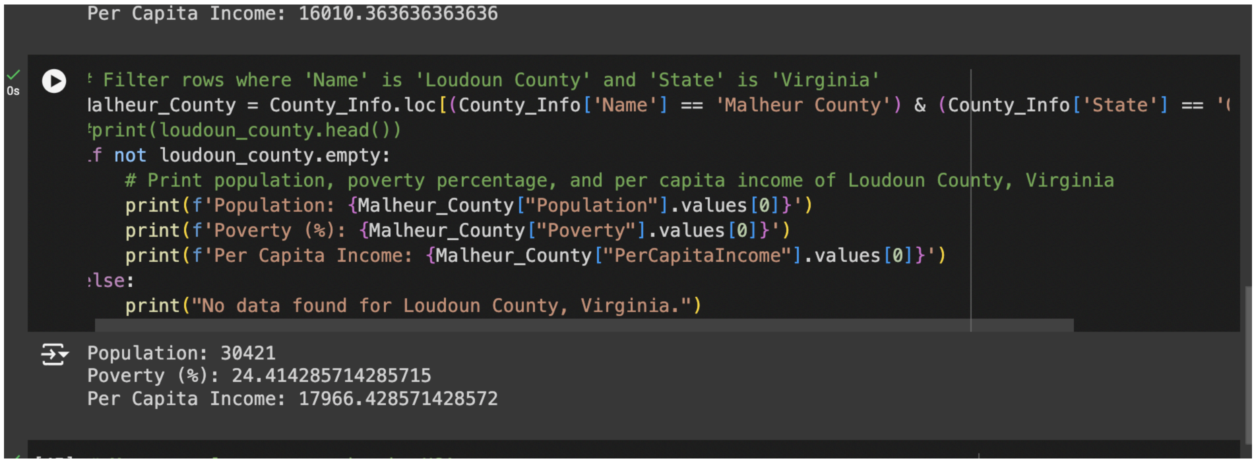
```
# Filter rows where 'Name' is 'Loudoun County' and 'State' is 'Virginia'
washington_county = County_Info.loc[(County_Info['Name'] == 'Washington County, Oregon') & (County_Info['State'] == 'Oregon')]
#print(loudoun_county.head())
# # Print population, poverty percentage, and per capita income of Loudoun County, Virginia
print(f'Population {loudoun_county["Population"].values[0]}')
print(f'Poverty (%) {loudoun_county["Poverty (%)"].values[0]}')
print(f'PerCapitaIncome {loudoun_county["PerCapitaIncome"].values[0]}')
```

```
Population 374558
Poverty (%) 0.0042591116183124675
PerCapitaIncome 58.24381137656998
```



```
# Filter rows where 'Name' is 'Loudoun County' and 'State' is 'Virginia'
Harlan_County = County_Info.loc[(County_Info['Name'] == 'Harlan County') & (County_Info['State'] == 'Ke
#print(loudoun_county.head())
if not loudoun_county.empty:
    # Print population, poverty percentage, and per capita income of Loudoun County, Virginia
    print(f'Population: {Harlan_County["Population"].values[0]}')
    print(f'Poverty (%): {Harlan_County["Poverty"].values[0]}')
    print(f'Per Capita Income: {Harlan_County["PerCapitaIncome"].values[0]}')
else:
    print("No data found for Loudoun County, Virginia.")
```

```
Population: 27548
Poverty (%): 33.31818181818182
Per Capita Income: 16010.363636363636
```

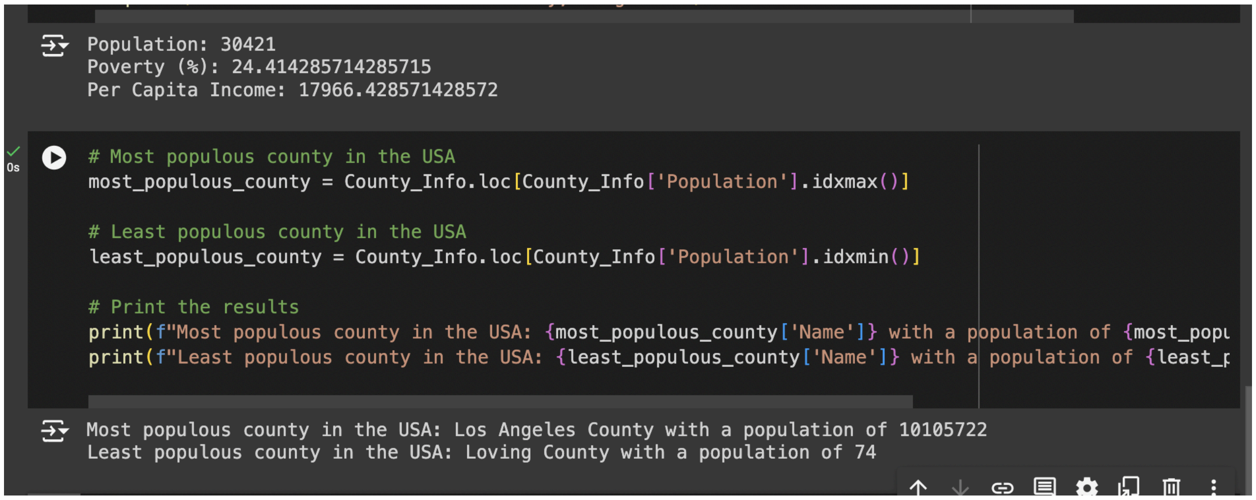


```
Per Capita Income: 16010.363636363636
```

```
: Filter rows where 'Name' is 'Loudoun County' and 'State' is 'Virginia'
lalheur_County = County_Info.loc[(County_Info['Name'] == 'Malheur County') & (County_Info['State'] == '(
:print(loudoun_county.head())
.f not loudoun_county.empty:
    # Print population, poverty percentage, and per capita income of Loudoun County, Virginia
    print(f'Population: {Malheur_County["Population"].values[0]}')
    print(f'Poverty (%): {Malheur_County["Poverty"].values[0]}')
    print(f'Per Capita Income: {Malheur_County["PerCapitaIncome"].values[0]}')
:lse:
    print("No data found for Loudoun County, Virginia.")
```

```
Population: 30421
Poverty (%): 24.414285714285715
Per Capita Income: 17966.428571428572
```

\



```
Population: 30421
Poverty (%): 24.414285714285715
Per Capita Income: 17966.428571428572
```

```
# Most populous county in the USA
most_populous_county = County_Info.loc[County_Info['Population'].idxmax()]

# Least populous county in the USA
least_populous_county = County_Info.loc[County_Info['Population'].idxmin()]

# Print the results
print(f"Most populous county in the USA: {most_populous_county['Name']} with a population of {most_popu
print(f"Least populous county in the USA: {least_populous_county['Name']} with a population of {least_p
```

```
Most populous county in the USA: Los Angeles County with a population of 10105722
Least populous county in the USA: Loving County with a population of 74
```

Fill the following table using data from your County_info DataFrame:

County	Population	%poverty	PerCapitaIncome
Loudoun County, Virginia	374558	3.884375	50391.015625
Washington County, Oregon	572071	10.446153846153846	34970.817307692305
Harlan County, Kentucky	27548	33.31818181818182	16010.363636363636
Malheur County, Oregon	30421	24.414285714285715	17966.428571428572

Answer the following questions:

- Most populous county in the USA?
- Least populous county in the USA?

```
1. Most populous county in the USA: Los Angeles County with a
   population of 10105722
2. Least populous county in the USA: Loving County with a
   population of 74
```

C. [MUST] Transform the COVID Data

Simplify the COVID data along the time dimension. The COVID data set contains day-level resolution data from (approximately) January of 2020 through February of 2021. From this you should produce a **COVID_monthly** Data Frame that has just one row per month per county.

Also, add a county ID column that is a foreign key lookup to the corresponding ID column in the County_info DataFrame.

Fill the following table using data from your COVID_monthly DataFrame:

County	Month	# cases	# deaths
Malheur County, Oregon	August 2020	28163	459.0
Malheur County,	January 2021	96297	1627.0

Oregon			
Malheur County, Oregon	February 2021	`65951`	`1137.0`

D. [MUST] Integrate COVID Data with ACS Data

Create a new single pandas DataFrame called COVID_summary containing one row per county. It should have these columns:
ID - integer identifier for the county
Population, Poverty, PerCapitaIncome - these should all be the same as from the County_info DataFrame
TotalCases, TotalDeaths - these two values should come from the COVID_monthly data and summed over all months
TotalCasesPer100K, TotalDeathsPer100K - these two values should be computed by dividing the the TotalCases and TotalDeaths (respectively) by (Population / 100000)

Fill the following table using data from your COVID_summary DataFrame:

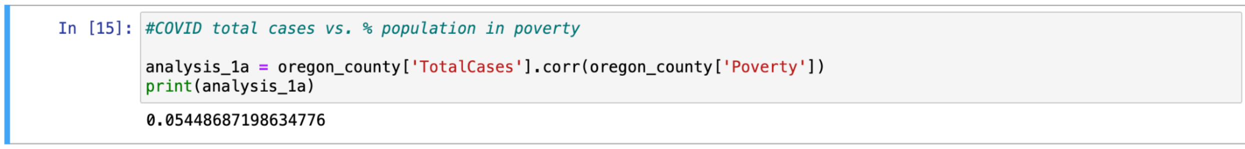
County	Poverty %	TotalCasesPer100K
Washington County, Oregon	**10.446154**	**61741.46**
Malheur County, Oregon	**24.414286**	**42792.81**
Loudoun County, Virginia	**3.884375**	**3086.84**
Harlan County, Kentucky	**33.318182**	**71979.82**

E. [SHOULD] Analysis

For each of the following, determine the strength of the correlation between each pair of variables. Compute the correlation strength by calculating the Pearson correlation coefficient R for pairs of columns in your DataFrame. For example, if you have a DataFrame df with each row representing a distinct county, and columns named ‘TotalCasesPer100K’ and ‘Poverty’, then you can compute R like this:

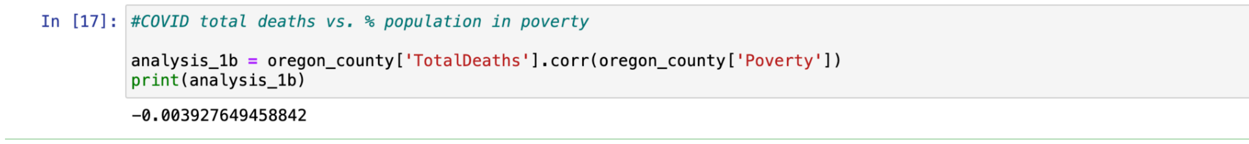
```
R = df['TotalCasesPer100K'].corr(df['Poverty'])
```

1. Compute the correlation coefficient for the following relationships for all Oregon counties
   a. COVID total cases vs. % population in poverty
   => 0.05448687198634776



```
In [15]: #COVID total cases vs. % population in poverty

         analysis_1a = oregon_county['TotalCases'].corr(oregon_county['Poverty'])
         print(analysis_1a)

         0.05448687198634776
```

   b. COVID total deaths vs. % population in poverty
   => -0.003927649458842



```
In [17]: #COVID total deaths vs. % population in poverty

         analysis_1b = oregon_county['TotalDeaths'].corr(oregon_county['Poverty'])
         print(analysis_1b)

         -0.003927649458842
```

   c. COVID total cases vs. Per Capita Income level
   => -0.2557187510018219

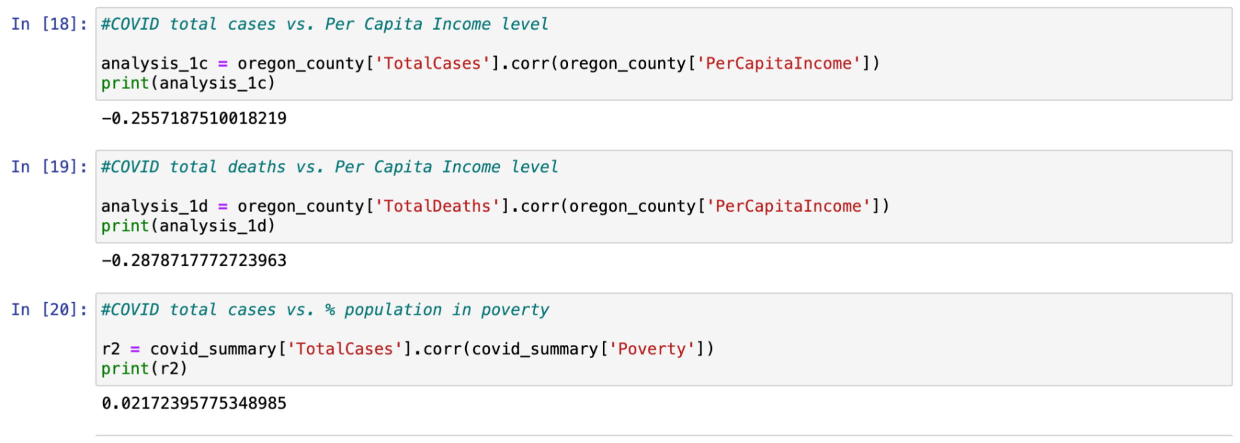
   d. COVID total deaths vs. Per Capita Income level
   => -0.2878717772723963

2. Across all of the counties in the entire USA
   a. COVID total cases vs. % population in poverty
   => 0.02172395775348985

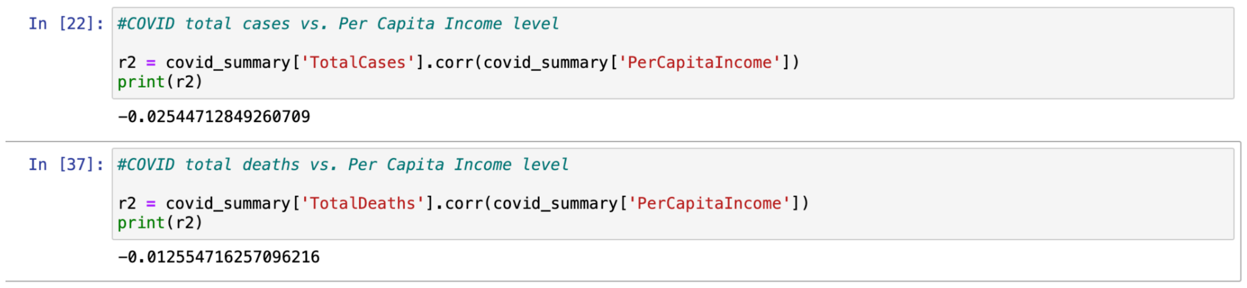
   b. COVID total deaths vs. % population in poverty
   => 0.00988447825942164

   c. COVID total cases vs. Per Capita Income level
   => 0.00988447825942164

   d. COVID total deaths vs. Per Capita Income level
   => -0.012554716257096216



```
In [18]: #COVID total cases vs. Per Capita Income level

         analysis_1c = oregon_county['TotalCases'].corr(oregon_county['PerCapitaIncome'])
         print(analysis_1c)

         -0.2557187510018219

In [19]: #COVID total deaths vs. Per Capita Income level

         analysis_1d = oregon_county['TotalDeaths'].corr(oregon_county['PerCapitaIncome'])
         print(analysis_1d)

         -0.2878717772723963

In [20]: #COVID total cases vs. % population in poverty

         r2 = covid_summary['TotalCases'].corr(covid_summary['Poverty'])
         print(r2)

         0.02172395775348985
```



```
In [22]: #COVID total cases vs. Per Capita Income level

         r2 = covid_summary['TotalCases'].corr(covid_summary['PerCapitaIncome'])
         print(r2)

         -0.02544712849260709

In [37]: #COVID total deaths vs. Per Capita Income level

         r2 = covid_summary['TotalDeaths'].corr(covid_summary['PerCapitaIncome'])
         print(r2)

         -0.012554716257096216
```

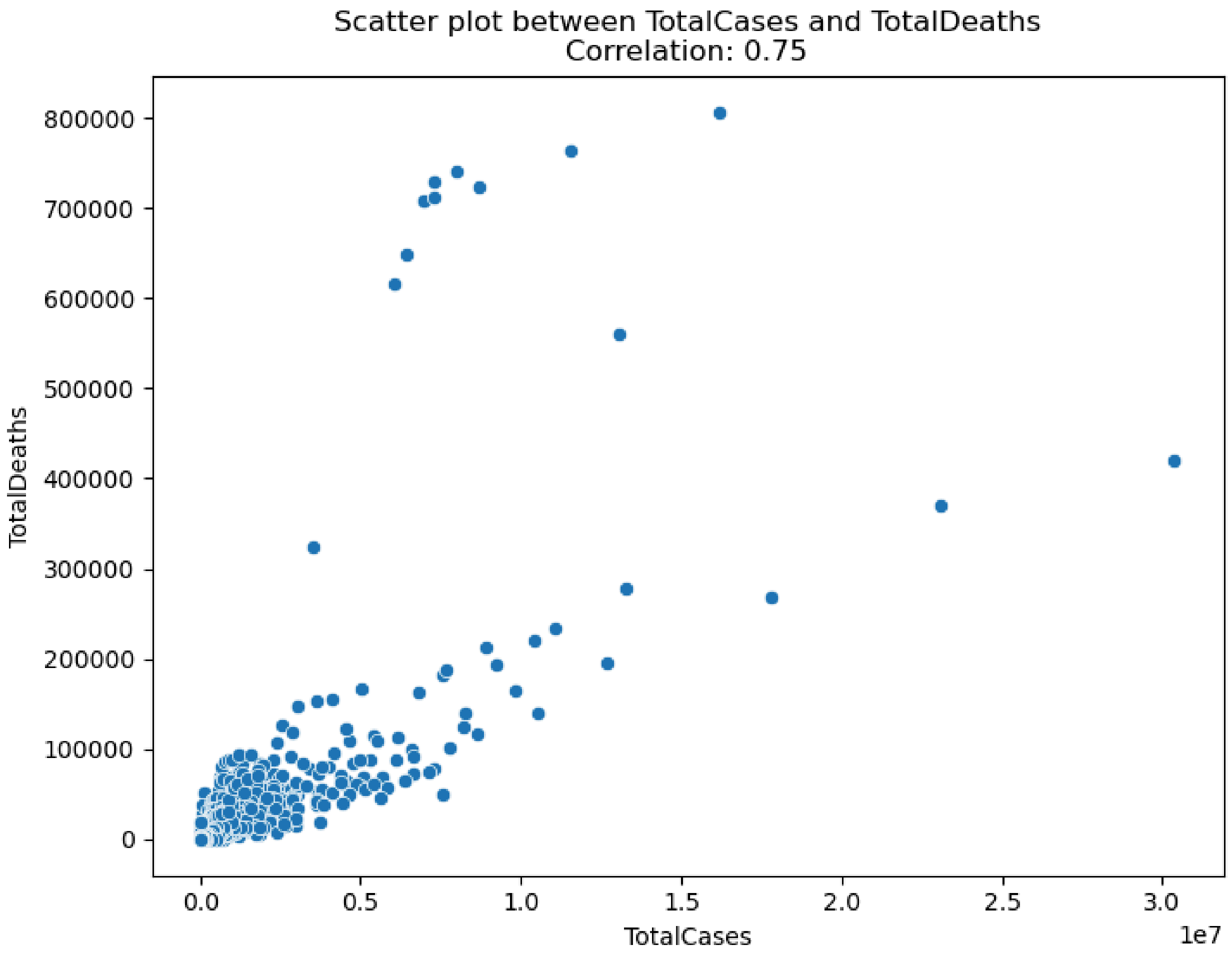
Fill the following table using data from your analysis. Add more rows to the table if you find additional interesting information:

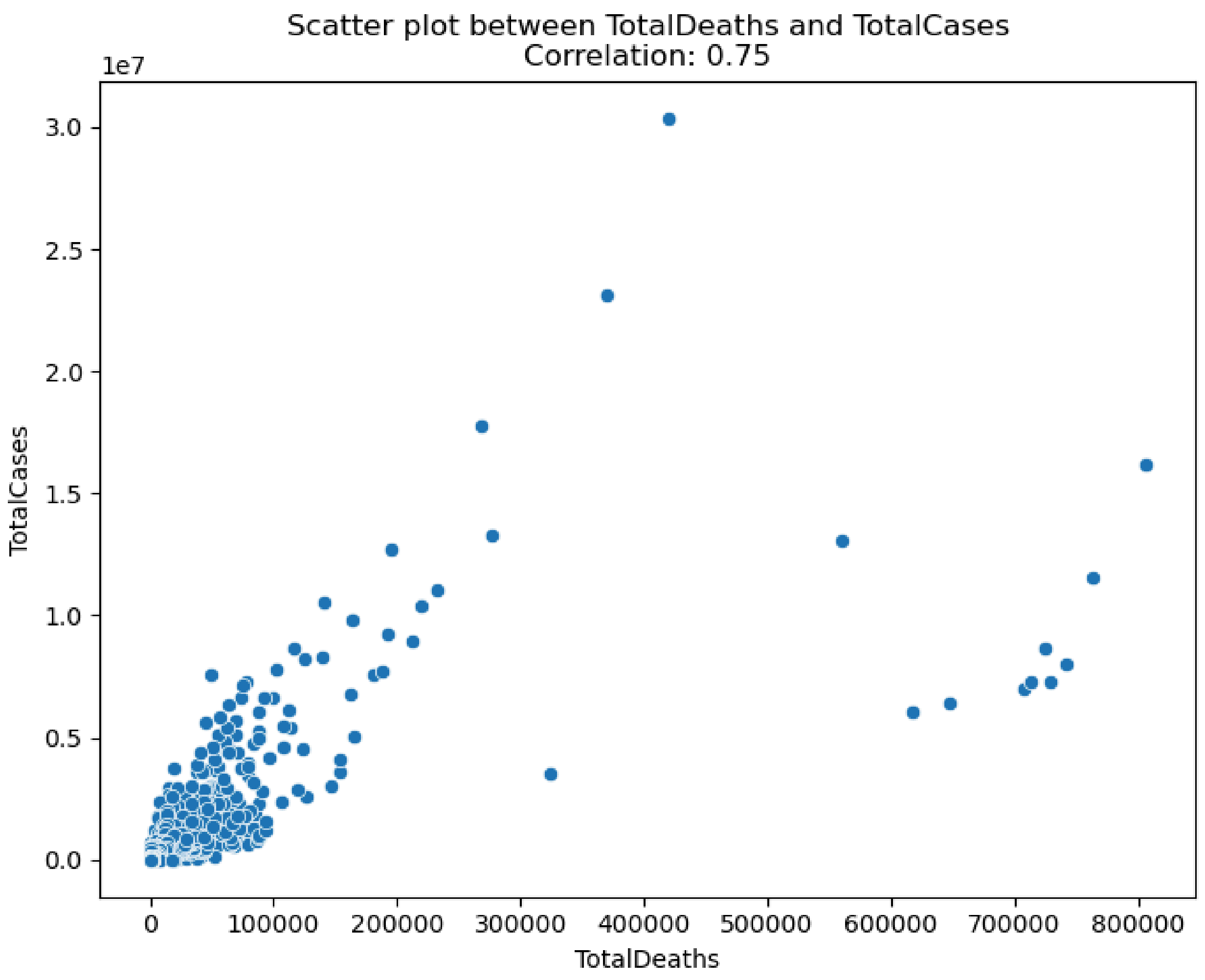
County	**R-value**
For Oregon Counties only: correlation between % poverty and COVID cases	0.05448687198634776
For all counties: correlation between population and COVID cases	0.02172395775348985
For Oregon counties only: correlation between PerCapitaIncome and COVID deaths	-0.2878717772723963
For all USA counties: correlation between PerCapitaIncome and COVID cases	-0.2557187510018219
<explore at least one other correlation computation that is made possible by this integrated data set>	
Calculate correlation between poverty percentage and per capita income	-0.4126570053840566

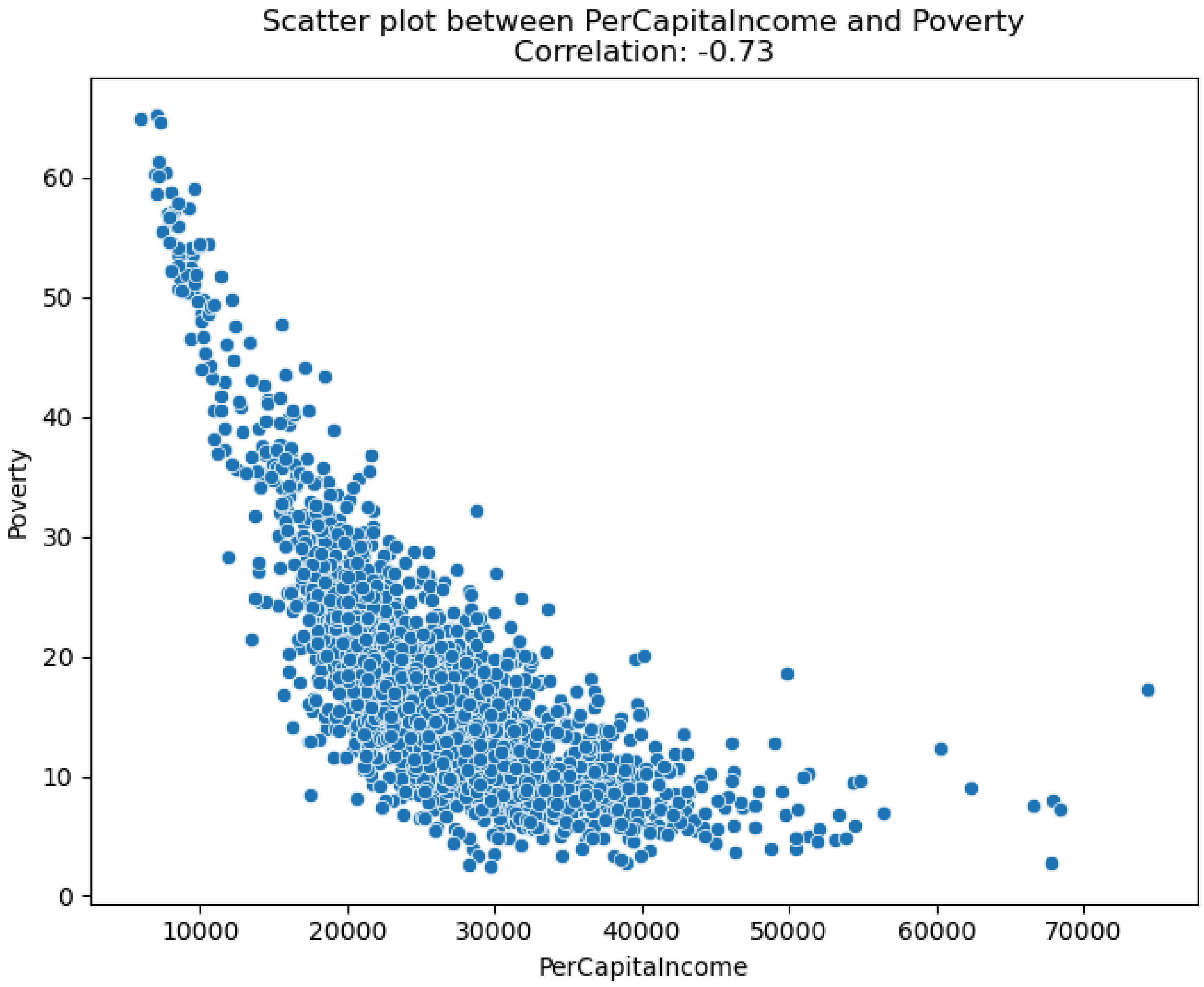
F. [ASPIRE] Charting

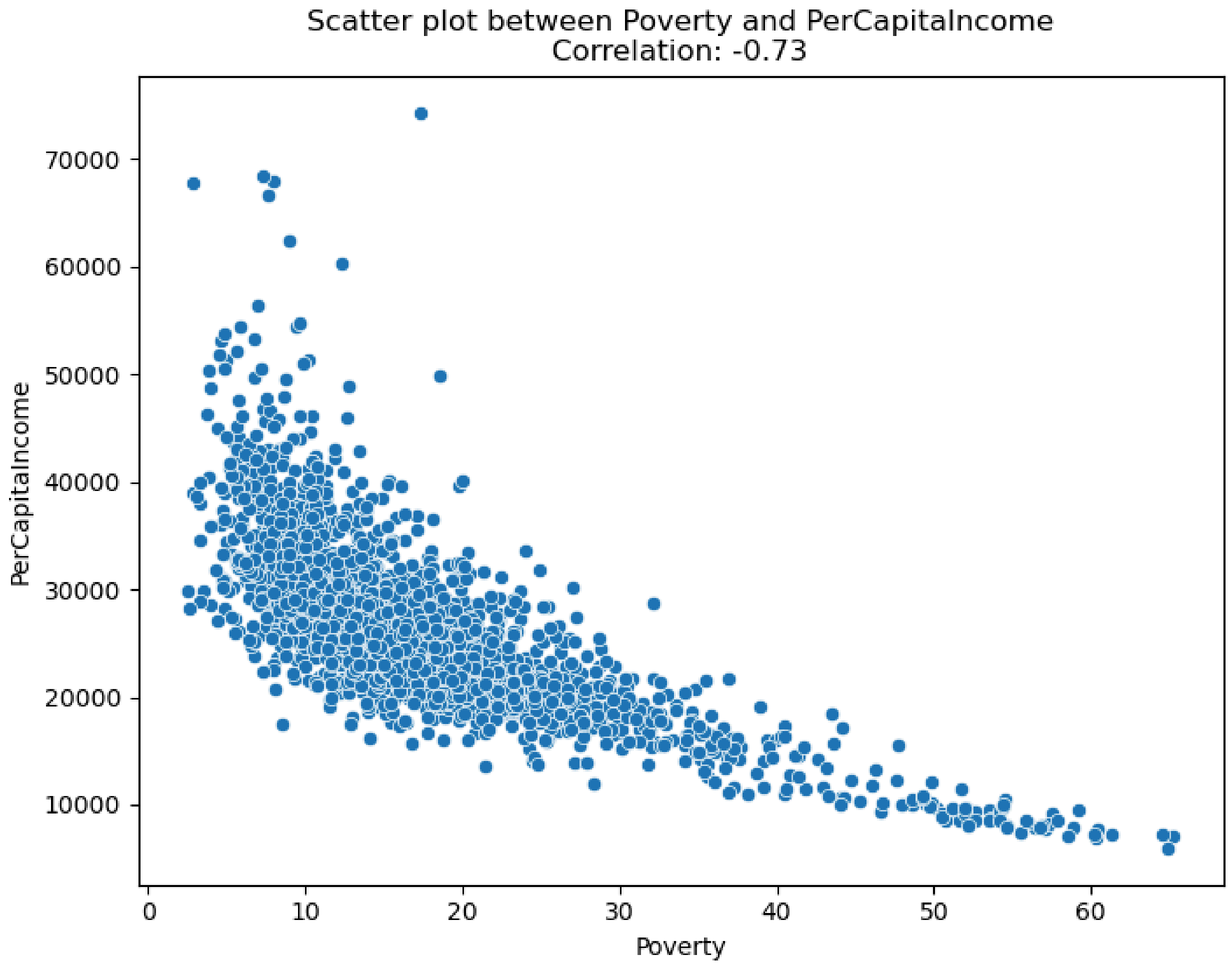
For any row (in the table above) with R > 0.5 or < -0.5 display a scatter plot (see pandas scatterplot and seaborn documentation for information about how to display scatter plots from DataFrame data).

Note that this assignment does not constitute a competent, thorough statistical analysis of the relationships between immunological data and demographic data. It is just an example of the type of work that is often required to integrate disparate data sets.

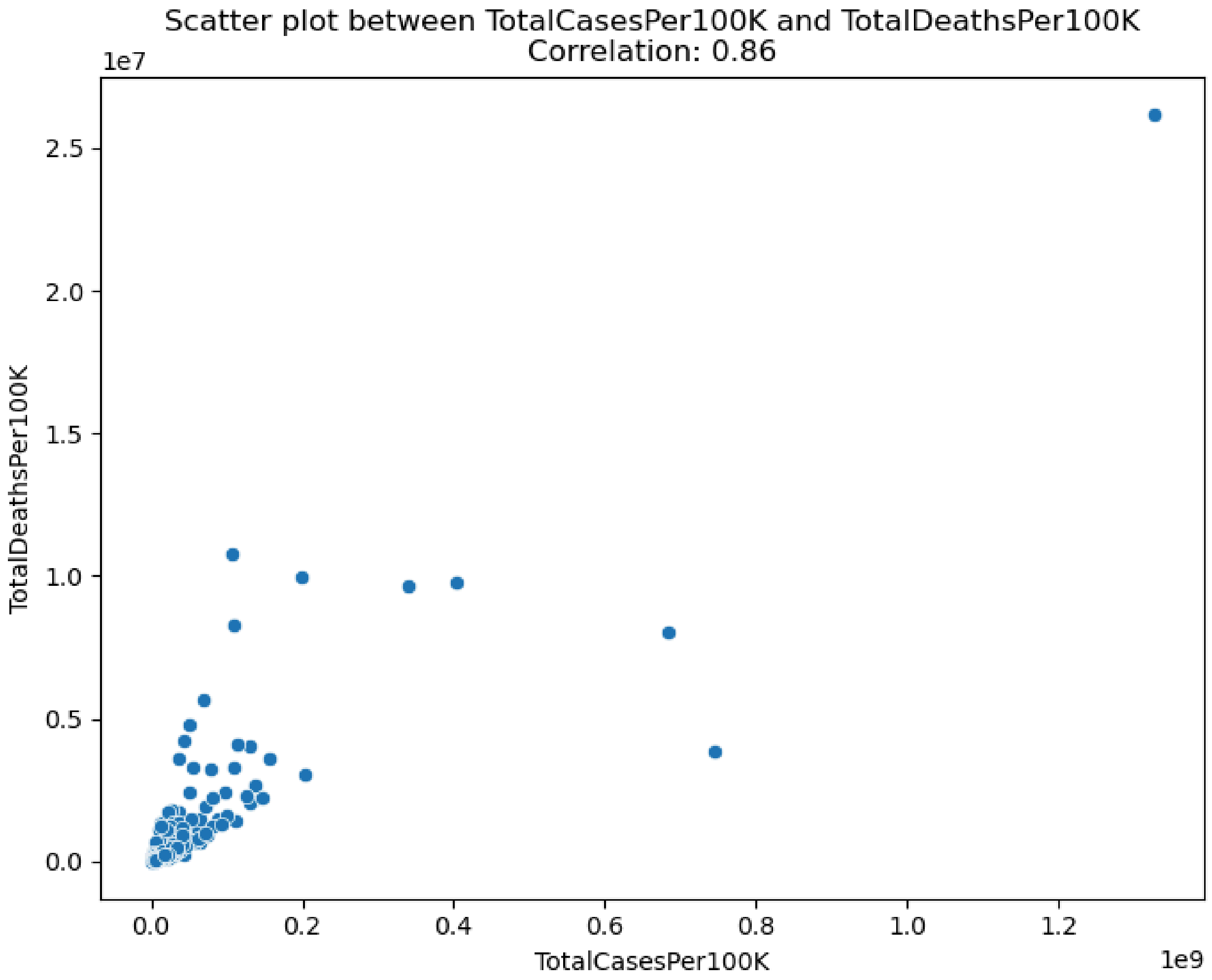



Scatter plot between TotalDeaths and TotalCases
Correlation: 0.75
1e7
3.0
2.5
2.0
1.5
1.0
0.5
0.0
TotalCases
0
100000
200000
300000
400000
500000
600000
700000
800000
TotalDeaths

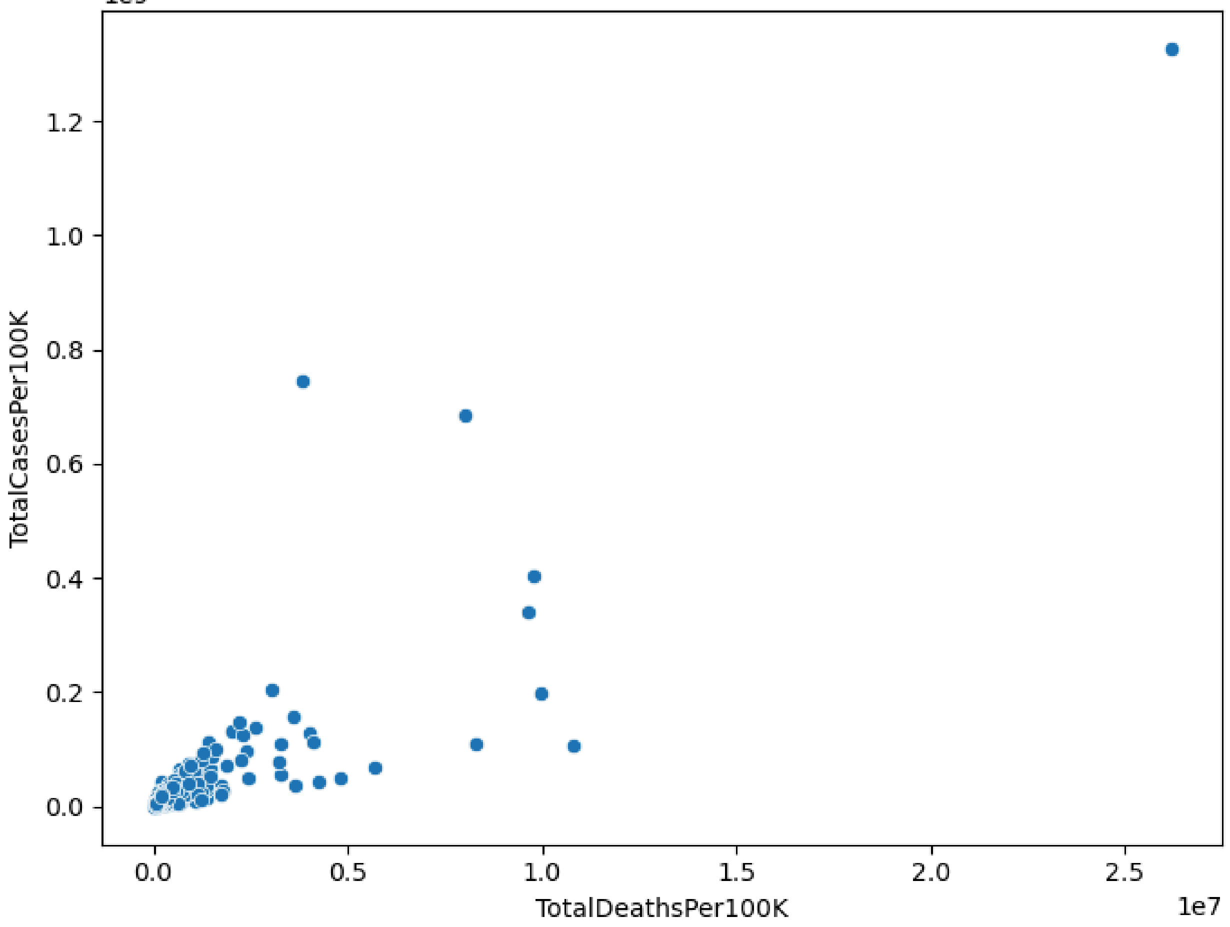

Scatter plot between PerCapitaIncome and Poverty
Correlation: -0.73
Poverty
PerCapitaIncome
0
10
20
30
40
50
60
10000
20000
30000
40000
50000
60000
70000



Scatter plot between Poverty and PerCapitaIncome
Correlation: -0.73
PerCapitaIncome
Poverty
70000
60000
50000
40000
30000
20000
10000
0
10
20
30
40
50
60


Scatter plot between TotalCasesPer100K and TotalDeathsPer100K
Correlation: 0.86
1e7
2.5
2.0
1.5
1.0
0.5
0.0
TotalDeathsPer100K
0.0
0.2
0.4
0.6
0.8
1.0
1.2
TotalCasesPer100K
1e9

Scatter plot between TotalDeathsPer100K and TotalCasesPer100K
Correlation: 0.86


1e9
TotalCasesPer100K
1e7
TotalDeathsPer100K