3031_A5_Complete_Solutions_v0

November 4, 2021

1 Assignment 5

Instructions:

a. You may talk to a friend, discuss the questions and potential directions for solving them. However, you need to write your own solutions and code separately, and not as a group activity.

b. Do not write your name on the assignment. (1 point)

c. Please include each question (or question number) followed by code and your answer (if applicable). Write your code in the 'Code' cells and your answer in the 'Markdown' cells of the Jupyter notebook. Ensure that the solution is written neatly enough to understand and grade. (1/2 point value of each question)

d. Export your Jupyter notebook as a PDF file. If you get an error, make sure you have downloaded the MikTex software (for windows) or MacTex (for mac). Note that after installing MikTex/MacTex, you will need to check for updates, install the updates if needed, and re-start your system. Submit the PDF file. (1 point)

This assignment is due at 11:59pm on Wednesday, November 10th. Good luck! *(30 points overall – 28 points for code & answers, 2 points for anonymity and proper formatting)*

1.1 Part 1

(5 points total)

Read FIFA world cup attendance data from the page: https://en.wikipedia.org/wiki/FIFA_World_Cup . Use 'attendance' as the matching string to find the table.

(a) Find the number of levels of column labels and row labels in the data *(2 points for code)*

(b) Reduce the multiple levels of column labels to a single level as follows. If the column names at all the levels are different, then concatenate the names together. Otherwise, keep the name at the highest level. For example, if the column name is ('Hosts','Hosts'), it should change to 'Host'. If the column name is ('Highest attendances †','Number'), it should change to 'Highest attendances †Number'. Do not rename each column manually. Use a method that will work efficiently if there were a large number of columns, say 10,000 columns. *(3 points for code)*

[1]:
```
import numpy as np
import pandas as pd
```

[2]:
```
dfs = pd.read_html('https://en.wikipedia.org/wiki/
↪FIFA_World_Cup',match='attendance')
```

[3]:
```
print(len(dfs))
data = dfs[0]
```

```
1
```

[4]:
```
print(data.columns.nlevels)
print(data.index.nlevels)
```

```
2
1
```

[5]:
```
combine_levels = ~(data.columns.get_level_values(0)==data.columns.
↪get_level_values(1))
```

[6]:
```
col_names = (data.columns.get_level_values(0)).to_list()
```

[7]:
```
for i in np.where(combine_levels)[0]:
    col_names[i] = (data.columns.get_level_values(0)[i]) + (data.columns.
↪get_level_values(1)[i])
```

[8]:
```
data.columns = col_names
```

[9]:
```
data.head()
```

[9]:

	Year	Hosts	Venues/Cities	Totalattendance	Matches	Avg.attendance	\
0	1930	Uruguay	3/1	590549	18	32808	
1	1934	Italy	8/8	363000	17	21353	
2	1938	France	10/9	375700	18	20872	
3	1950	Brazil	6/6	1045246	22	47511	
4	1954	Switzerland	6/6	768607	26	29562	

	Highest attendances †Number	Highest attendances †Venue	\
0	93000	Estadio Centenario, Montevideo	
1	55000	Stadio Nazionale PNF, Rome	
2	58455	Olympique de Colombes, Paris	
3	173,850[84]	Maracanã Stadium, Rio de Janeiro	
4	63000	Wankdorf Stadium, Bern	

	Highest attendances †Game(s)
0	Uruguay 6-1 Yugoslavia, Semi-final
1	Italy 2-1 Czechoslovakia, Final
2	France 1-3 Italy, Quarter-final
3	Brazil 1-2 Uruguay, Deciding match
4	West Germany 3-2 Hungary, Final

1.2 Part 2

(7 points total)

(a): Read the GDP per capita data from https://en.wikipedia.org/wiki/List_of_countries_by_GDP_(nominal)_

Perform the following operations on this datatable:

(i) Drop all the columns except Country and GDP per capita estimate by IMF. *(1 point for code)*

(ii) The country names contain some special characters (characters other than letters) and need to be cleaned. The following code can help clean country names:

```
import re

f = lambda x: re.sub(r'[^A-Za-z]', '', x)
```

Apply the above lambda function on the country column to clean country names. *(1 point for code)*

(b) Read the population data from https://en.wikipedia.org/wiki/List_of_countries_by_population_(United_Na
Drop all columns except Country and Population (1 July 2019). *(1 point for code)*

(c) Merge the datasets obtained in (a) and (b) such that the merged dataset contains each observation of the GDP per capita data (dataset obtained in (a)), but not necessarily each observation of the population data (dataset obtained in (b)). *(2 points for code)*

(d) For how many countries in the GDP per capita data was the population not available in the population data? Note that you don't need to clean country names in the population table. *(1 point for code, 1 point for answer)*

[10]:
```
# Read GDP per capita data from the webpage: https://en.wikipedia.org/wiki/
 ↪List_of_countries_by_GDP_(nominal)_per_capita
dfs = pd.read_html('https://en.wikipedia.org/wiki/
 ↪List_of_countries_by_GDP_(nominal)_per_capita', match = 'Country')

#How many tables did you read?
print(len(dfs))
```

```
1
```

[11]:
```
data = dfs[0]
```

[12]:
```
data = data.iloc[:,[0,3]]
```

[13]:
```
data.columns = ['Country','GDP_per_capita']
```

[14]:
```
data.head()
```

[14]:

	Country	GDP_per_capita
0	Monaco *	NaN
1	Liechtenstein *	NaN
2	Luxembourg *	131782
3	Bermuda *	NaN

```
4    Switzerland *          94696
```

[15]:
```
import re
f = lambda x: re.sub(r'[^A-Za-z]', '', x)
```

[16]:
```
data.loc[:,'Country'] = data.loc[:,"Country"].apply(f)
```

```
/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py:1843:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  self.obj[item_labels[indexer[info_axis]]] = value
```

[17]:
```
data.head()
```

[17]:

	Country	GDP_per_capita
0	Monaco	NaN
1	Liechtenstein	NaN
2	Luxembourg	131782
3	Bermuda	NaN
4	Switzerland	94696

[18]:
```
# Read GDP per capita data from the webpage: https://en.wikipedia.org/wiki/
 ↪List_of_countries_by_GDP_(nominal)_per_capita
dfs = pd.read_html('https://en.wikipedia.org/wiki/
 ↪List_of_countries_by_population_(United_Nations)', match = 'Country')

#How many tables did you read?
print(len(dfs))
```

```
1
```

[19]:
```
dp=dfs[0]
```

[20]:
```
dp.head()
```

[20]:

	Country/Area	UN continentalregion[4]	UN statisticalsubregion[4]	\
0	China[a]	Asia	Eastern Asia	
1	India	Asia	Southern Asia	
2	United States	Americas	Northern America	
3	Indonesia	Asia	South-eastern Asia	
4	Pakistan	Asia	Southern Asia	

	Population(1 July 2018)	Population(1 July 2019)	Change
0	1427647786	1433783686	+0.43%

```
1              1352642280              1366417754  +1.02%
2               327096265               329064917  +0.60%
3               267670543               270625568  +1.10%
4               212228286               216565318  +2.04%
```

[21]:
```
dp = dp.iloc[:,[0,4]]
```

[22]:
```
dp.rename(columns={'Country/Area':'Country'},inplace = True)
```

```
/opt/anaconda3/lib/python3.8/site-packages/pandas/core/frame.py:4441:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  return super().rename(
```

[23]:
```
data_all = pd.merge(data,dp,how = 'outer')
```

[24]:
```
data_all.isnull().sum()
```

[24]:
```
Country                     0
GDP_per_capita            113
Population(1 July 2019)    74
dtype: int64
```

[25]:
```
data_all
```

[25]:
```
                            Country GDP_per_capita  Population(1 July 2019)
0                            Monaco            NaN                  38964.0
1                     Liechtenstein            NaN                  38019.0
2                        Luxembourg         131782                 615729.0
3                           Bermuda            NaN                      NaN
4                       Switzerland          94696                8591365.0
..                              …              …                        …
303       Montserrat (United Kingdom)            NaN                  4989.0
304  Falkland Islands (United Kingdom)            NaN                  3377.0
305                Niue (New Zealand)            NaN                  1615.0
306             Tokelau (New Zealand)            NaN                  1340.0
307                   Vatican City[z]            NaN                   799.0

[308 rows x 3 columns]
```

113 countries had GDP data missing.

1.3 Part 3

(16 points total)

The dataset *Real GDP.csv* contains the GDP of each US State for all years starting from 1997 until 2020. The data is at State level, i.e., each observation corresponds to a unique State.

The dataset *Surplus.csv* contains the surplus of each US State for all years starting from 1997 until 2020. The data is at year level, i.e., each observation corresponds to a unique year.

The dataset *Compensation.csv* contains 'Compensation' and 'Chain-type quantity indexes for real GDP' for each US State and year starting from 1997 to 2020. The dataset is at Year-State-Description level, where 'Description' refers to either 'Compensation' or 'Chain-type quantity indexes for real GDP'.

Q1) Combine all these datasets to obtain a dataset at State-Year level, that contains the GDP, surplus, 'Compensation', and 'Chain-type quantity indexes for real GDP' for each US State and all years starting from 1997 until 2020. *Note that each observation must contain the name of the US State, year, and the four values (GDP, surplus, compensation, and Chain-type quantity indexes for real GDP).*

Hint: Here is one way to do it: 1) Melt the GDP dataset to year-State level
2) Melt the Surplus dataset to year-State level
3) Pivot the compensation dataset to year-State level

Now that all the datasets are at the year-State level, merge them!

(4 points for code)

[26]:
```
compensation = pd.read_csv('Compensation.csv')
surplus = pd.read_csv('Surplus.csv')
GDP = pd.read_csv('Real GDP.csv')
```

[27]:
```
result1= compensation.pivot(['Year','GeoName'],'Description','value')
result1
```

[27]:
```
Description         Chain-type quantity indexes for real GDP  \
Year GeoName
1997 Alabama                                           76.356
     Alaska                                            72.424
     Arizona                                           62.043
     Arkansas                                          76.108
     California                                        65.225
…                                                        …
2020 Virginia                                         106.243
     Washington                                       133.039
     West Virginia                                     99.132
     Wisconsin                                        105.272
     Wyoming                                           93.312

Description         Compensation (millions of dollars)
Year GeoName
1997 Alabama                                   61083.8
     Alaska                                    12347.8
```

```
           Arizona                                  69876.9
           Arkansas                                 32715.5
           California                              574432.9
...                                                    ...
2020       Virginia                                327867.2
           Washington                              326111.4
           West Virginia                            40948.0
           Wisconsin                               197578.0
           Wyoming                                  18623.2

[1224 rows x 2 columns]
```

[28]:
```
GDP.drop(columns = 'Description',inplace = True)
```

[29]:
```
GDP.columns.name = 'Year'
#result2 = GDP.unstack("GeoName").stack("Year").unstack("Description").
 ↪stack("GeoName")
result2 = pd.melt(GDP,id_vars = ['GeoName'])
```

[30]:
```
result2
```

[30]:
```
             GeoName  Year      value
0            Alabama  1997   144501.2
1             Alaska  1997    42211.3
2            Arizona  1997   168408.8
3           Arkansas  1997    82571.3
4         California  1997  1378276.5
...              ...   ...        ...
1219        Virginia  2020   473817.5
1220      Washington  2020   532861.9
1221  West Virginia  2020    69711.6
1222       Wisconsin  2020   291715.8
1223         Wyoming  2020    36256.7

[1224 rows x 3 columns]
```

[31]:
```
surplus_melted = surplus.melt('Year')
```

[32]:
```
surplus_rename = surplus_melted.rename({'variable': 'GeoName', 'value': 'Gross␣
 ↪operating surplus (millions of dollars)'}, axis=1)
```

[33]:
```
result3 = surplus_rename.sort_values(['Year','GeoName']).
 ↪set_index(['Year','GeoName'])
```

[34]:
```
result3
```

[34]:
```
                    Gross operating surplus (millions of dollars)
Year GeoName
1997 Alabama                                                37247.9
     Alaska                                                 11061.3
     Arizona                                                53776.0
     Arkansas                                               23316.0
     California                                            431069.4
…                                                              …
2020 Virginia                                              200788.1
     Washington                                            248091.9
     West Virginia                                          31157.2
     Wisconsin                                             128759.4
     Wyoming                                                15469.6

[1224 rows x 1 columns]
```

[35]:
```
result2.index = result3.index #result2 index 'years' is not numeric
```

[36]:
```
result2
```

[36]:
```
                          GeoName  Year      value
Year GeoName
1997 Alabama              Alabama  1997   144501.2
     Alaska                Alaska  1997    42211.3
     Arizona              Arizona  1997   168408.8
     Arkansas            Arkansas  1997    82571.3
     California        California  1997  1378276.5
…                             …    …         …
2020 Virginia            Virginia  2020   473817.5
     Washington        Washington  2020   532861.9
     West Virginia  West Virginia  2020    69711.6
     Wisconsin          Wisconsin  2020   291715.8
     Wyoming              Wyoming  2020    36256.7

[1224 rows x 3 columns]
```

[37]:
```
final = pd.concat([result1, result2, result3], axis=1)
```

[38]:
```
final
```

[38]:
```
                    Chain-type quantity indexes for real GDP  \
Year GeoName
1997 Alabama                                           76.356
     Alaska                                            72.424
     Arizona                                           62.043
     Arkansas                                          76.108
     California                                        65.225
```

```
…                                                    …
2020 Virginia                                    106.243
     Washington                                  133.039
     West Virginia                                99.132
     Wisconsin                                   105.272
     Wyoming                                      93.312

                    Compensation (millions of dollars)        GeoName  Year  \
Year GeoName
1997 Alabama                                     61083.8        Alabama  1997
     Alaska                                      12347.8         Alaska  1997
     Arizona                                     69876.9        Arizona  1997
     Arkansas                                    32715.5       Arkansas  1997
     California                                 574432.9     California  1997
…                                                  …            …    …
2020 Virginia                                   327867.2       Virginia  2020
     Washington                                 326111.4     Washington  2020
     West Virginia                               40948.0  West Virginia  2020
     Wisconsin                                  197578.0      Wisconsin  2020
     Wyoming                                     18623.2        Wyoming  2020

                         value  Gross operating surplus (millions of dollars)
Year GeoName
1997 Alabama         144501.2                                        37247.9
     Alaska           42211.3                                        11061.3
     Arizona         168408.8                                        53776.0
     Arkansas         82571.3                                        23316.0
     California     1378276.5                                       431069.4
…                         …                                            …
2020 Virginia        473817.5                                       200788.1
     Washington      532861.9                                       248091.9
     West Virginia    69711.6                                        31157.2
     Wisconsin       291715.8                                       128759.4
     Wyoming          36256.7                                        15469.6

[1224 rows x 6 columns]
```

[39]:
```
final.drop(columns = ['Year','GeoName'],inplace=True)
final.reset_index(inplace = True)
```

[40]:
```
final.rename(columns = {'value':'GDP'},inplace = True)
```

[41]:
```
final.head()
```

[41]:
```
   Year     GeoName  Chain-type quantity indexes for real GDP  \
0  1997     Alabama                                    76.356
1  1997      Alaska                                    72.424
```

```
2  1997     Arizona                                        62.043
3  1997    Arkansas                                        76.108
4  1997  California                                        65.225

   Compensation (millions of dollars)        GDP  \
0                             61083.8   144501.2
1                             12347.8    42211.3
2                             69876.9   168408.8
3                             32715.5    82571.3
4                            574432.9  1378276.5

   Gross operating surplus (millions of dollars)
0                                        37247.9
1                                        11061.3
2                                        53776.0
3                                        23316.0
4                                       431069.4
```

Q2) Use a single plot to answer all three questions below by visualizing:
(a) How does the mean GDP (mean over all States) change with year? *(1 point for visualization)*
(b) How does the mean compensation (mean over all States) change with year? *(1 point for visualization)*
(c) How does the mean surplus (mean over all States) change with year? *(1 point for visualization)*

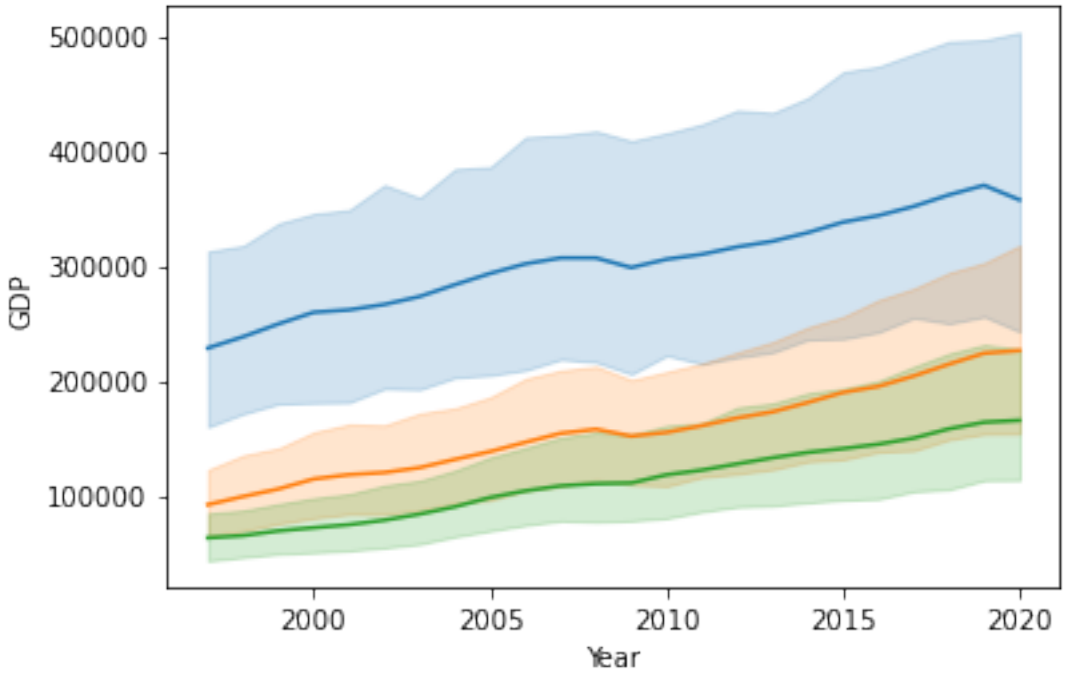
Also show the 95% confidence interval for the mean GDP, mean compensation, and mean surplus in the plot.

Hint: Use the *seaborn* function *lineplot()* . No calculations are needed. Just use *lineplot()* three times.

[42]:
```
import seaborn as sns
```

[43]:
```
sns.lineplot(data = final, x = 'Year', y = 'GDP')
sns.lineplot(data = final, x = 'Year', y = 'Compensation (millions of dollars)')
sns.lineplot(data = final, x = 'Year', y = 'Gross operating surplus (millions␣
 ↪of dollars)')
```

[43]:
```
<AxesSubplot:xlabel='Year', ylabel='GDP'>
```



Q3) The mean GDP (over all States) seems to have decreased in 2020 as compared to 2019 *(you know why!)*. How many States observed a decrease in GDP in 2020 (as compared to 2019)? For which States did the GDP increase in 2020 (as compared to 2019)? *(2 points for code, 2 points for answers)*

[44]:
```
(GDP[(GDP['2020']-GDP['2019'])<0]).shape
```

[44]:
```
(49, 25)
```

[45]:
```
GDP[(GDP['2020']-GDP['2019'])>0].GeoName
```

[45]:
```
41    South Dakota
44            Utah
Name: GeoName, dtype: object
```

South Dakota and Utah observed increases in GDP.

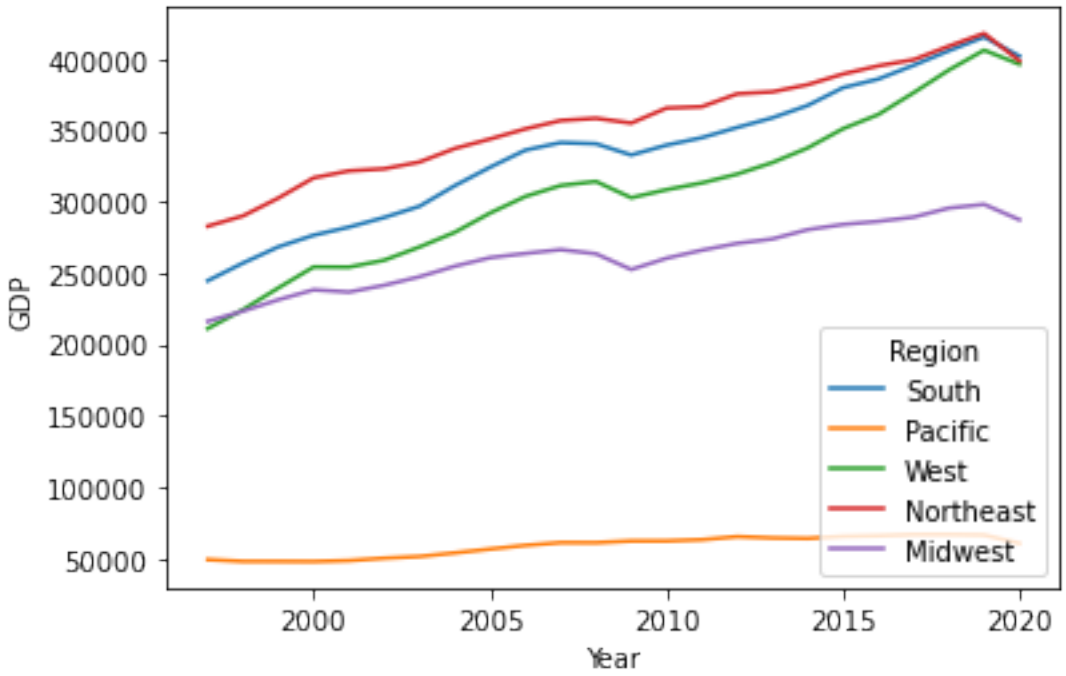
Q4) Merge the file *State_region_mapping.csv* with the dataset obtained in Q1. Make a lineplot showing the mean GDP for each of the five regions with year. Do not display the confidence interval. Which two regions seems to have the least growth in GDP over the past 24 years? *(2 points for code, 1 point for answer)*

[46]:
```
state_region = pd.read_csv('State_region_mapping.csv')
```

[47]:
```
final2=pd.merge(final,state_region,left_on = 'GeoName',right_on = 'State')
```

[48]:
```
sns.lineplot(data = final2, x = 'Year', y = 'GDP',hue = 'Region',ci=None)
```

[48]:
```
<AxesSubplot:xlabel='Year', ylabel='GDP'>
```



The Pacific and West regions seem to have the least growth in GDP over the past 24 years.

Q5) Identify the States contributing the most to the total GDP of their region, in 2020. Also, find the percentage contribution of these States to the total GDP of their region. *(2 points for code)*

Hint: You may use *DataFrameGroupBy.idxmax()* for the first part of this question.

[49]:
```
final2.head()
```

[49]:

	Year	GeoName	Chain-type quantity indexes for real GDP	Compensation (millions of dollars)	GDP
0	1997	Alabama	76.356	61083.8	144501.2
1	1998	Alabama	79.034	64168.6	149568.2
2	1999	Alabama	81.851	67225.1	154900.2
3	2000	Alabama	83.078	69764.4	157221.3
4	2001	Alabama	82.883	72038.4	156853.2

```
   Gross operating surplus (millions of dollars)    State Region
0                                        37247.9  Alabama  South
1                                        39368.1  Alabama  South
2                                        41513.7  Alabama  South
3                                        42583.4  Alabama  South
4                                        43348.6  Alabama  South
```

[50]:
```
dy = final2[final2.Year==2000]
ids_max = dy[['Region','GDP']].groupby('Region').idxmax()
```

[51]:
```
dy.head()
```

[51]:
```
    Year     GeoName  Chain-type quantity indexes for real GDP  \
3   2000     Alabama                                     83.078
27  2000      Alaska                                     67.612
51  2000     Arizona                                     76.790
75  2000    Arkansas                                     82.837
99  2000  California                                     80.270

    Compensation (millions of dollars)        GDP  \
3                                69764.4   157221.3
27                               13893.0    39406.6
51                               91205.4   208439.5
75                               38766.8    89871.7
99                              768199.0  1696172.4

    Gross operating surplus (millions of dollars)       State   Region
3                                         42583.4     Alabama    South
27                                        10455.7      Alaska  Pacific
51                                        62781.3     Arizona     West
75                                        25974.1    Arkansas     West
99                                       509912.0  California     West
```

[52]:
```
dy.loc[ids_max.iloc[:,0],:]
```

[52]:
```
      Year     GeoName  Chain-type quantity indexes for real GDP  \
315   2000    Illinois                                     88.205
771   2000    New York                                     82.229
267   2000      Hawaii                                     77.729
1035  2000       Texas                                     70.059
99    2000  California                                     80.270

      Compensation (millions of dollars)        GDP  \
315                              285572.2   640723.4
771                              478847.4  1092188.2
267                               23456.2    55891.1
```

```
1035                            405918.7   995661.2
99                              768199.0  1696172.4

      Gross operating surplus (millions of dollars)       State     Region
315                                          170702.3    Illinois    Midwest
771                                          309367.0    New York  Northeast
267                                           14200.8      Hawaii    Pacific
1035                                         274635.9       Texas      South
99                                           509912.0  California       West
```

[53]:
```
dy[['Region','GDP']].groupby('Region').max()['GDP']/dy[['Region','GDP']].
↪groupby('Region').sum()['GDP']
```

[53]:
```
Region
Midwest      0.206946
Northeast    0.383204
Pacific      0.586489
South        0.240102
West         0.556042
Name: GDP, dtype: float64
```