SPRAWOZDANIE

Zajęcia: Nauka o danych I

Prowadzący: prof. dr hab. Vasyl Martsenyuk

Laboratorium Nr 1 Data 28.09.2024 Temat: " Ustalenia platformu Jupyter. Użycie biblioteki pandas" Wariant 11	Szymon Nycz Informatyka II stopień, stacjonarne, 1 semestr, gr. 1b

Link do Githuba:

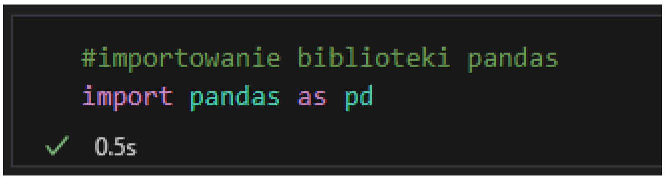
https://github.com/Maciek332/NoD

1. Polecenie: wariant 11 zadania

Premise General Population COVID-19 Health Services Disruption Survey 2020 http://ghdx.healthdata.org/record/ihme-data/premise-general-population-covid-19-health-services-disruption-survey-2020

2. Opis programu opracowanego

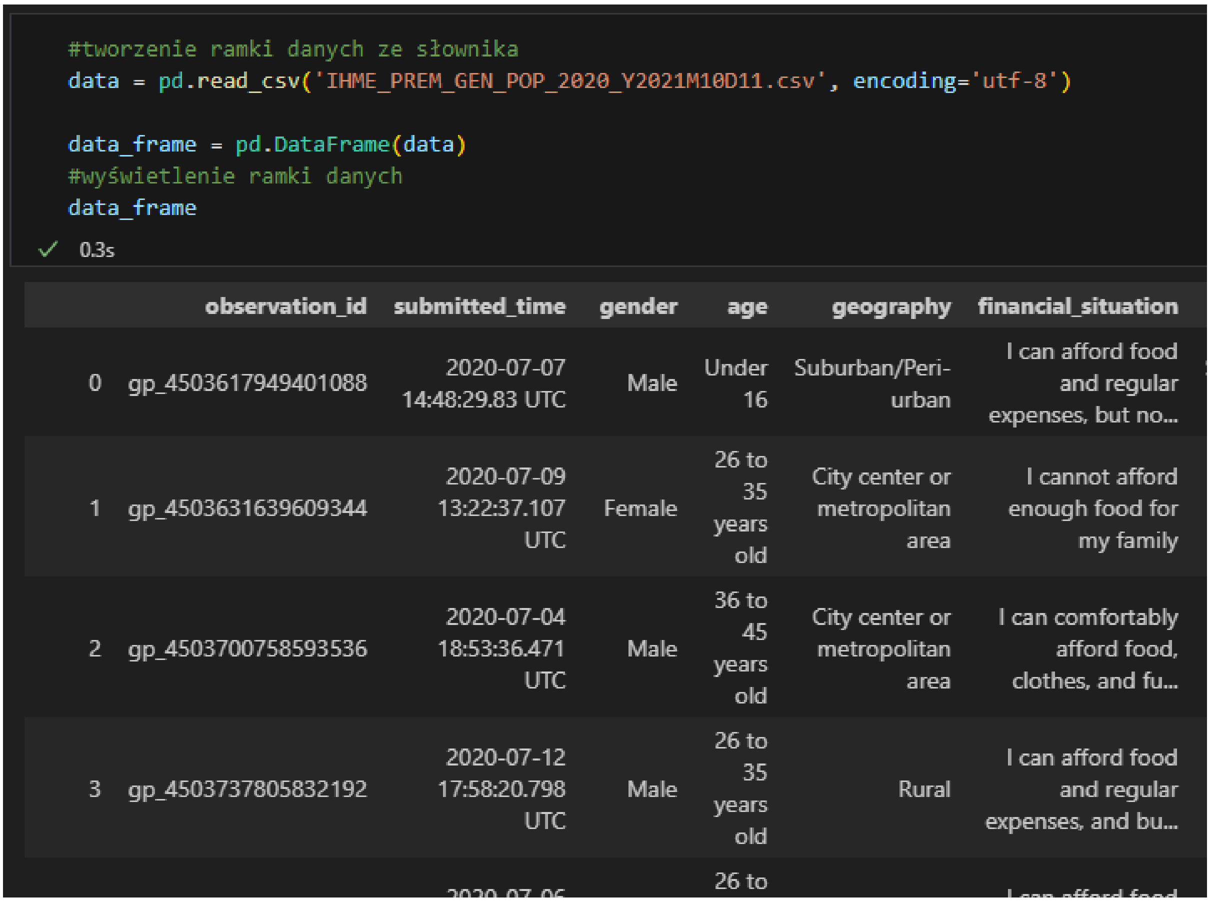
- ładowanie biblioteki Pandas



```
#importowanie biblioteki pandas
import pandas as pd
```

✓ 0.5s

- tworzenie ramki danych ze słownika

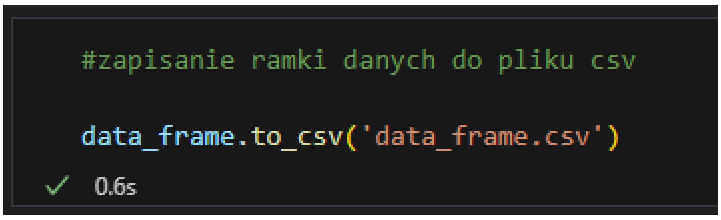


```
#tworzenie ramki danych ze słownika
data = pd.read_csv('IHME_PREM_GEN_POP_2020_Y2021M10D11.csv', encoding='utf-8')

data_frame = pd.DataFrame(data)
#wyświetlenie ramki danych
data_frame
```

✓ 0.3s

	observation_id	submitted_time	gender	age	geography	financial_situation
0	gp_4503617949401088	2020-07-07 14:48:29.83 UTC	Male	Under 16	Suburban/Peri-urban	I can afford food and regular expenses, but no...
1	gp_4503631639609344	2020-07-09 13:22:37.107 UTC	Female	26 to 35 years old	City center or metropolitan area	I cannot afford enough food for my family
2	gp_4503700758593536	2020-07-04 18:53:36.471 UTC	Male	36 to 45 years old	City center or metropolitan area	I can comfortably afford food, clothes, and fu...
3	gp_4503737805832192	2020-07-12 17:58:20.798 UTC	Male	26 to 35 years old	Rural	I can afford food and regular expenses, and bu...
		2020-07-06		26 to		I can afford food

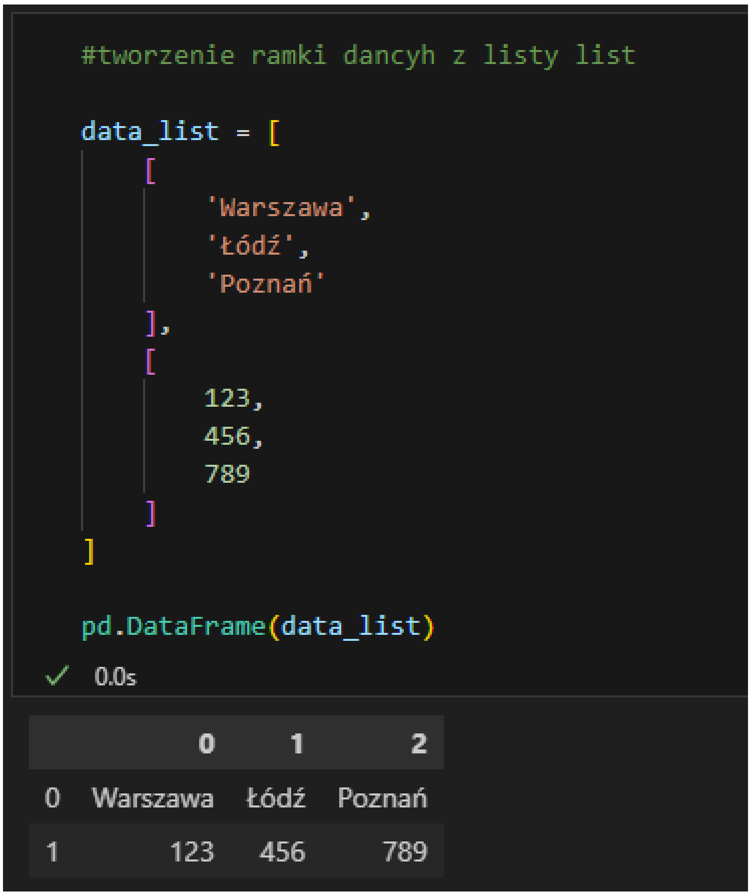
- zachowanie ramki danych pobranych z pliku w formacie csv (xlsx)



```
#zapisanie ramki danych do pliku csv

data_frame.to_csv('data_frame.csv')
```

✓ 0.6s

- tworzenie ramki danych z listy list

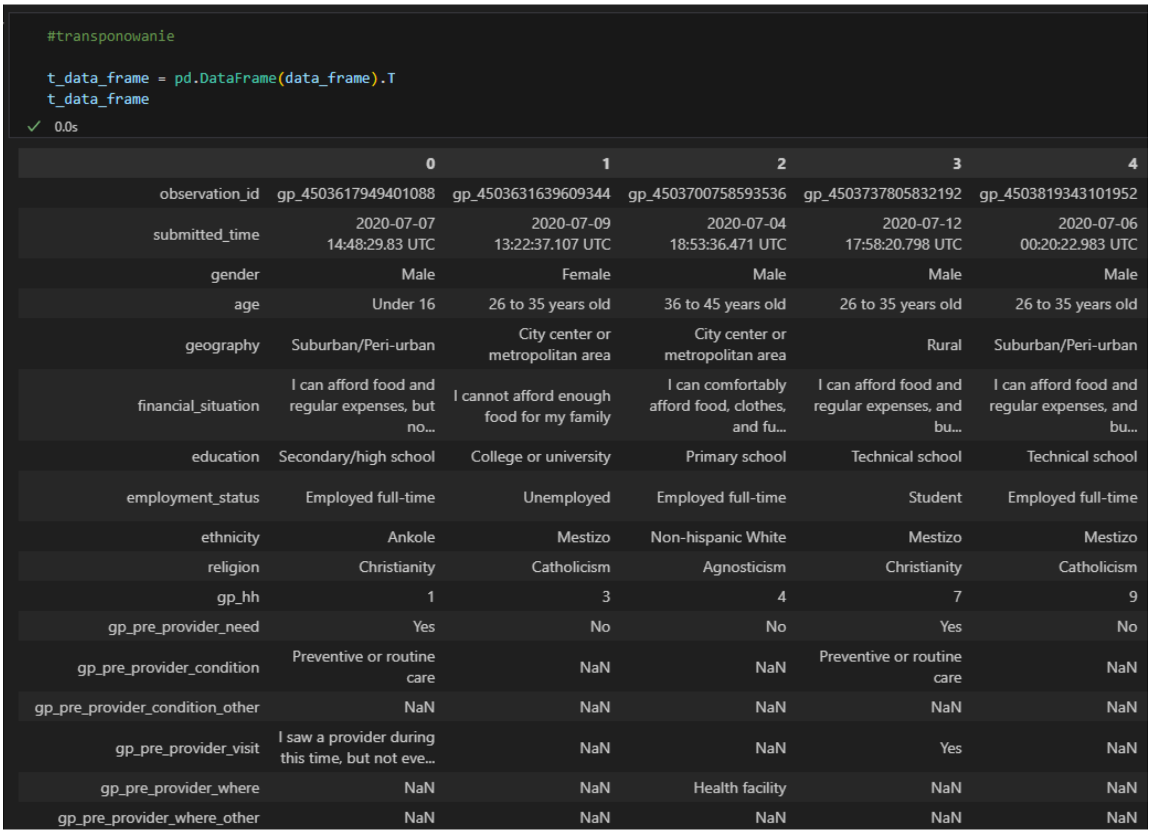


```python
#tworzenie ramki dancyh z listy list

data_list = [
    [
        'Warszawa',
        'Łódź',
        'Poznań'
    ],
    [
        123,
        456,
        789
    ]
]

pd.DataFrame(data_list)
```

✓ 0.0s

```
          0     1       2
0  Warszawa  Łódź  Poznań
1       123   456     789
```

- transponowanie (wymieniamy kolumny a wierszy)

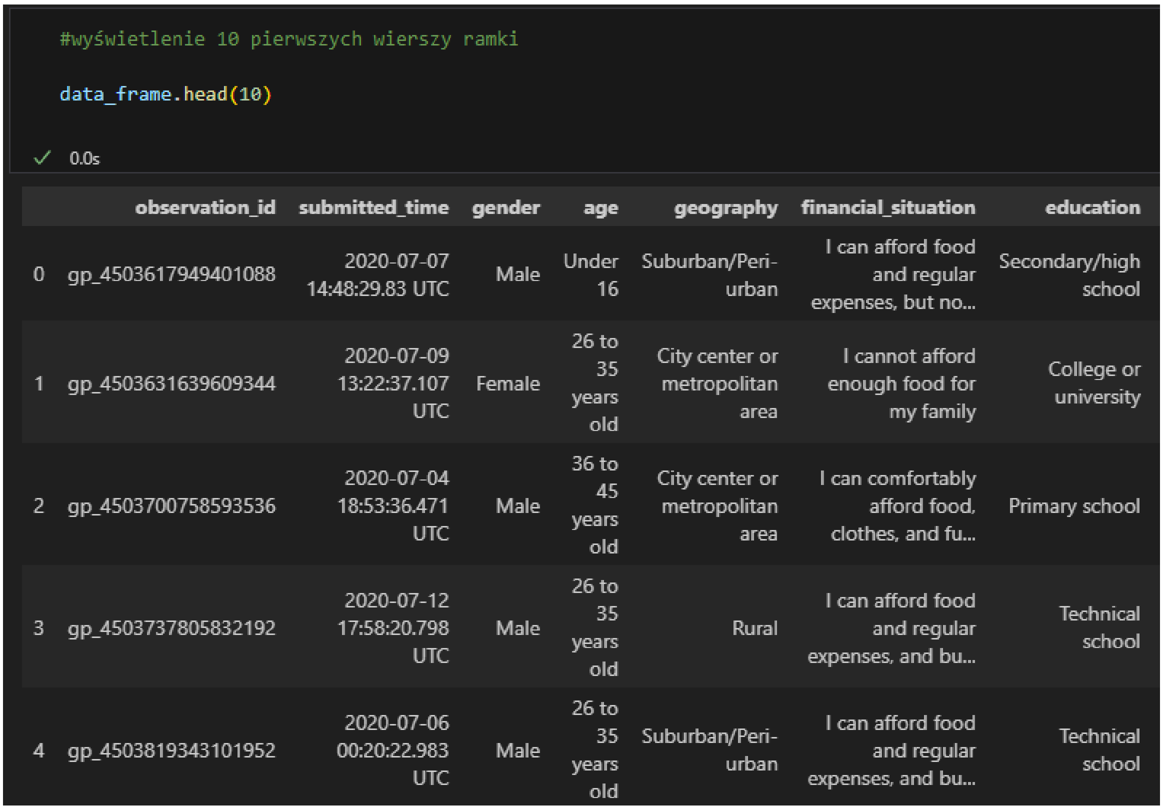


```
#transponowanie

t_data_frame = pd.DataFrame(data_frame).T
t_data_frame
```

✓ 0.0s

	0	1	2	3	4
observation_id	gp_4503617949401088	gp_4503631639609344	gp_4503700758593536	gp_4503737805832192	gp_4503819343101952
submitted_time	2020-07-07 14:48:29.83 UTC	2020-07-09 13:22:37.107 UTC	2020-07-04 18:53:36.471 UTC	2020-07-12 17:58:20.798 UTC	2020-07-06 00:20:22.983 UTC
gender	Male	Female	Male	Male	Male
age	Under 16	26 to 35 years old	36 to 45 years old	26 to 35 years old	26 to 35 years old
geography	Suburban/Peri-urban	City center or metropolitan area	City center or metropolitan area	Rural	Suburban/Peri-urban
financial_situation	I can afford food and regular expenses, but no...	I cannot afford enough food for my family	I can comfortably afford food, clothes, and fu...	I can afford food and regular expenses, and bu...	I can afford food and regular expenses, and bu...
education	Secondary/high school	College or university	Primary school	Technical school	Technical school
employment_status	Employed full-time	Unemployed	Employed full-time	Student	Employed full-time
ethnicity	Ankole	Mestizo	Non-hispanic White	Mestizo	Mestizo
religion	Christianity	Catholicism	Agnosticism	Christianity	Catholicism
gp_hh	1	3	4	7	9
gp_pre_provider_need	Yes	No	No	Yes	No
gp_pre_provider_condition	Preventive or routine care	NaN	NaN	Preventive or routine care	NaN
gp_pre_provider_condition_other	NaN	NaN	NaN	NaN	NaN
gp_pre_provider_visit	I saw a provider during this time, but not eve...	NaN	NaN	Yes	NaN
gp_pre_provider_where	NaN	NaN	Health facility	NaN	NaN
gp_pre_provider_where_other	NaN	NaN	NaN	NaN	NaN

- wyświetlić pierwsze 10 wierszy ramki danych

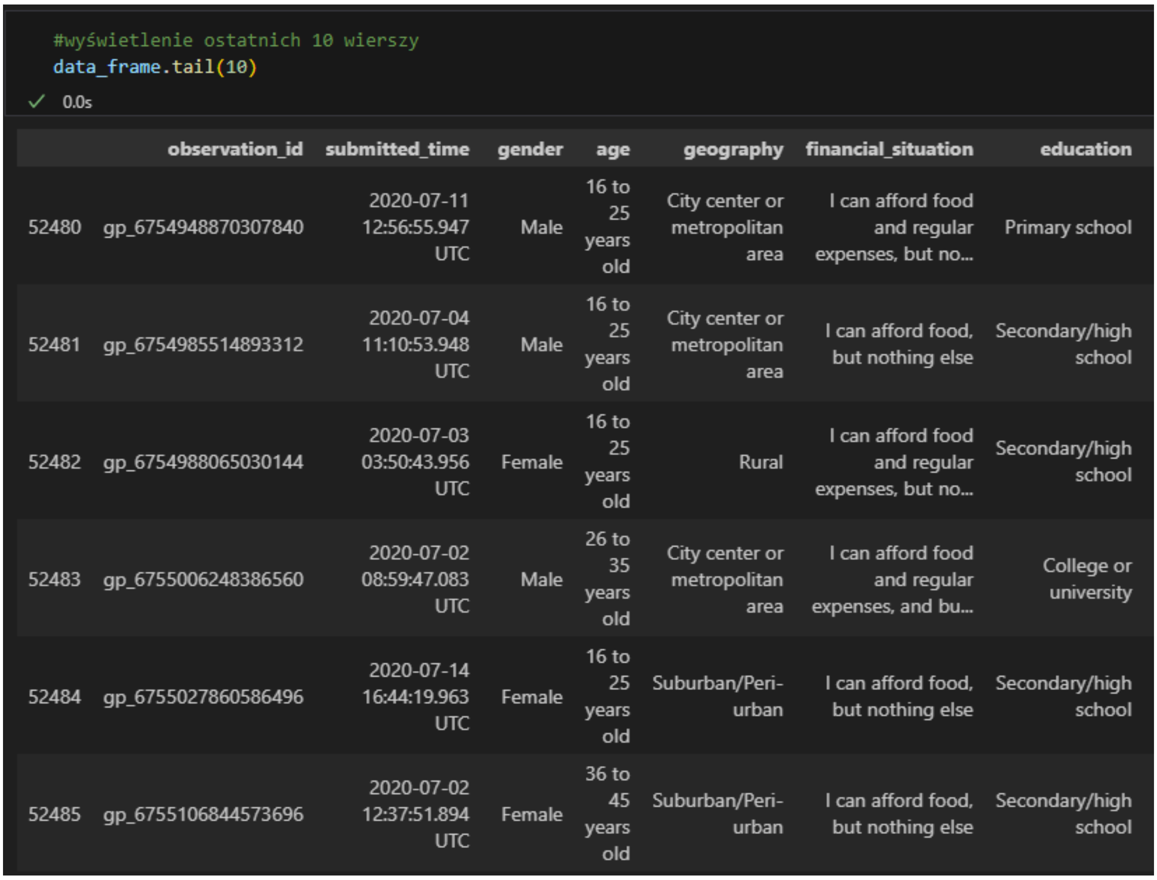


```
#wyświetlenie 10 pierwszych wierszy ramki

data_frame.head(10)
```

✓ 0.0s

	observation_id	submitted_time	gender	age	geography	financial_situation	education
0	gp_4503617949401088	2020-07-07 14:48:29.83 UTC	Male	Under 16	Suburban/Peri-urban	I can afford food and regular expenses, but no...	Secondary/high school
1	gp_4503631639609344	2020-07-09 13:22:37.107 UTC	Female	26 to 35 years old	City center or metropolitan area	I cannot afford enough food for my family	College or university
2	gp_4503700758593536	2020-07-04 18:53:36.471 UTC	Male	36 to 45 years old	City center or metropolitan area	I can comfortably afford food, clothes, and fu...	Primary school
3	gp_4503737805832192	2020-07-12 17:58:20.798 UTC	Male	26 to 35 years old	Rural	I can afford food and regular expenses, and bu...	Technical school
4	gp_4503819343101952	2020-07-06 00:20:22.983 UTC	Male	26 to 35 years old	Suburban/Peri-urban	I can afford food and regular expenses, and bu...	Technical school

- wyświetlić ostatnie 10 wierszy ramki danych

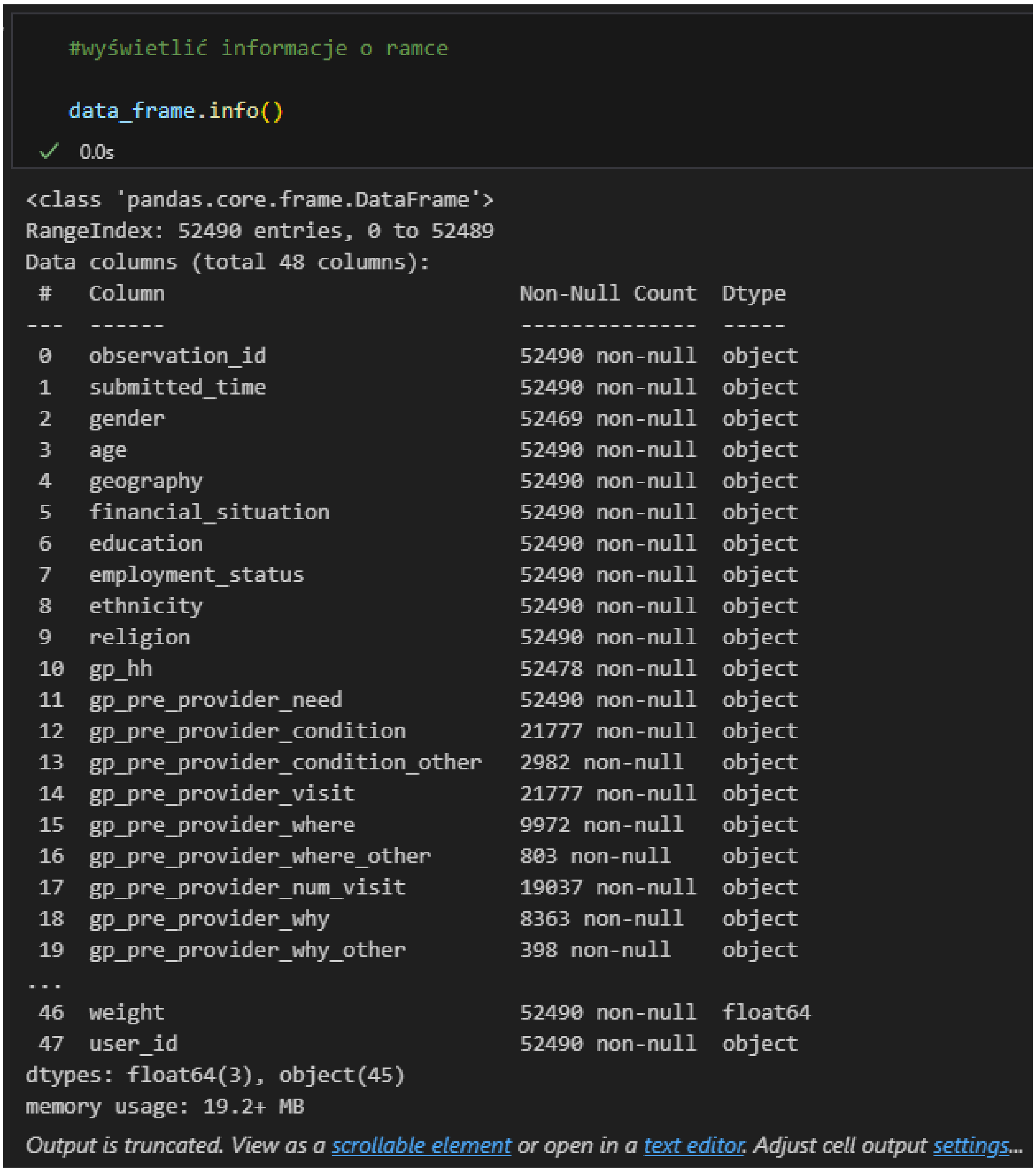


```
#wyświetlenie ostatnich 10 wierszy
data_frame.tail(10)
```

✓ 0.0s

	observation_id	submitted_time	gender	age	geography	financial_situation	education
52480	gp_6754948870307840	2020-07-11 12:56:55.947 UTC	Male	16 to 25 years old	City center or metropolitan area	I can afford food and regular expenses, but no...	Primary school
52481	gp_6754985514893312	2020-07-04 11:10:53.948 UTC	Male	16 to 25 years old	City center or metropolitan area	I can afford food, but nothing else	Secondary/high school
52482	gp_6754988065030144	2020-07-03 03:50:43.956 UTC	Female	16 to 25 years old	Rural	I can afford food and regular expenses, but no...	Secondary/high school
52483	gp_6755006248386560	2020-07-02 08:59:47.083 UTC	Male	26 to 35 years old	City center or metropolitan area	I can afford food and regular expenses, and bu...	College or university
52484	gp_6755027860586496	2020-07-14 16:44:19.963 UTC	Female	16 to 25 years old	Suburban/Peri-urban	I can afford food, but nothing else	Secondary/high school
52485	gp_6755106844573696	2020-07-02 12:37:51.894 UTC	Female	36 to 45 years old	Suburban/Peri-urban	I can afford food, but nothing else	Secondary/high school

- wyświetlić informacje¸ o ramce danych



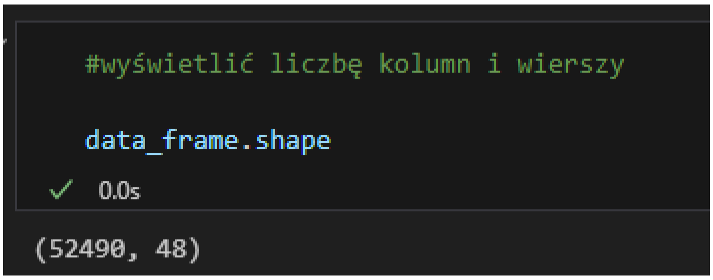
```
#wyświetlić informacje o ramce

data_frame.info()
```

✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 52490 entries, 0 to 52489
Data columns (total 48 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   observation_id                  52490 non-null  object
 1   submitted_time                  52490 non-null  object
 2   gender                          52469 non-null  object
 3   age                             52490 non-null  object
 4   geography                       52490 non-null  object
 5   financial_situation             52490 non-null  object
 6   education                       52490 non-null  object
 7   employment_status               52490 non-null  object
 8   ethnicity                       52490 non-null  object
 9   religion                        52490 non-null  object
 10  gp_hh                           52478 non-null  object
 11  gp_pre_provider_need            52490 non-null  object
 12  gp_pre_provider_condition       21777 non-null  object
 13  gp_pre_provider_condition_other 2982 non-null   object
 14  gp_pre_provider_visit           21777 non-null  object
 15  gp_pre_provider_where           9972 non-null   object
 16  gp_pre_provider_where_other     803 non-null    object
 17  gp_pre_provider_num_visit       19037 non-null  object
 18  gp_pre_provider_why             8363 non-null   object
 19  gp_pre_provider_why_other       398 non-null    object
...
 46  weight                          52490 non-null  float64
 47  user_id                         52490 non-null  object
dtypes: float64(3), object(45)
memory usage: 19.2+ MB
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

- wyświetlić, ile wierszy i kolumn znajduje się, w ramce danych

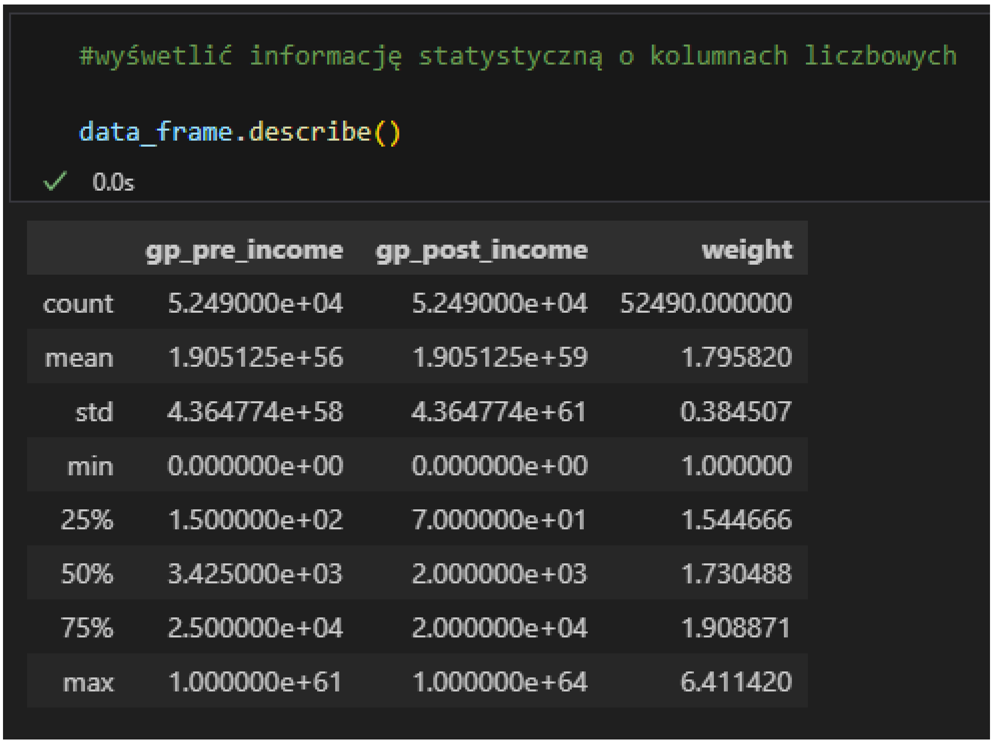


```
#wyświetlić liczbę kolumn i wierszy

data_frame.shape
```

✓ 0.0s

```
(52490, 48)
```

- wyświetlić informacje̜ statystyczna̜ o kolumnach liczbowych (wartości niepowtarzalne, średnia, odchylenie standardowe, minimum, kwartyle, maksimum)

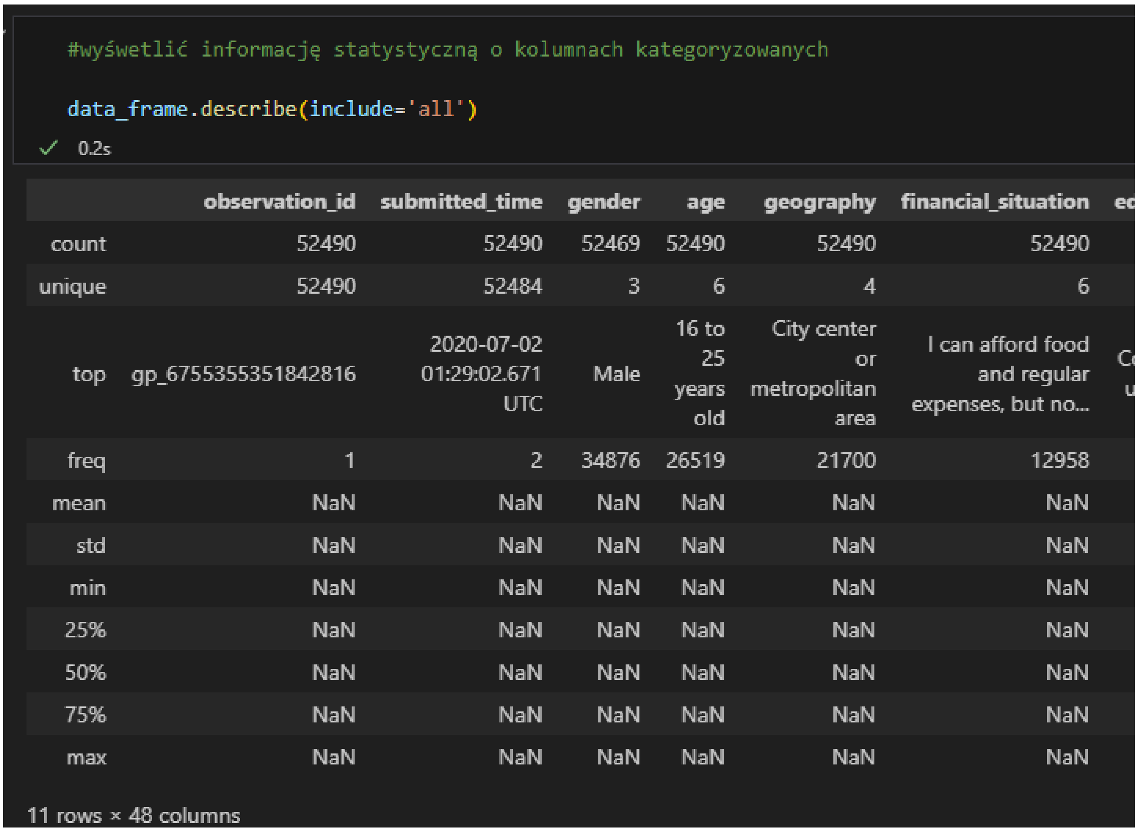


```python
#wyświetlić informację statystyczną o kolumnach liczbowych

data_frame.describe()
```

✓ 0.0s

	gp_pre_income	gp_post_income	weight
count	5.249000e+04	5.249000e+04	52490.000000
mean	1.905125e+56	1.905125e+59	1.795820
std	4.364774e+58	4.364774e+61	0.384507
min	0.000000e+00	0.000000e+00	1.000000
25%	1.500000e+02	7.000000e+01	1.544666
50%	3.425000e+03	2.000000e+03	1.730488
75%	2.500000e+04	2.000000e+04	1.908871
max	1.000000e+61	1.000000e+64	6.411420

- wyświetlić informacje̜ statystyczna̜ o kolumnach kategoryzowanych (ile unikalnych wartości, top - jaka jest najpopularniejsza wartość, freq - jak często najpopularniejsza)



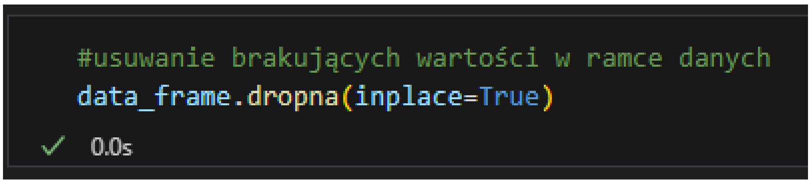
```python
#wyświetlić informację statystyczną o kolumnach kategoryzowanych

data_frame.describe(include='all')
```

✓ 0.2s

	observation_id	submitted_time	gender	age	geography	financial_situation	ec
count	52490	52490	52469	52490	52490	52490	
unique	52490	52484	3	6	4	6	
top	gp_6755355351842816	2020-07-02 01:29:02.671 UTC	Male	16 to 25 years old	City center or metropolitan area	I can afford food and regular expenses, but no...	C u
freq	1	2	34876	26519	21700	12958	
mean	NaN	NaN	NaN	NaN	NaN	NaN	
std	NaN	NaN	NaN	NaN	NaN	NaN	
min	NaN	NaN	NaN	NaN	NaN	NaN	
25%	NaN	NaN	NaN	NaN	NaN	NaN	
50%	NaN	NaN	NaN	NaN	NaN	NaN	
75%	NaN	NaN	NaN	NaN	NaN	NaN	
max	NaN	NaN	NaN	NaN	NaN	NaN	

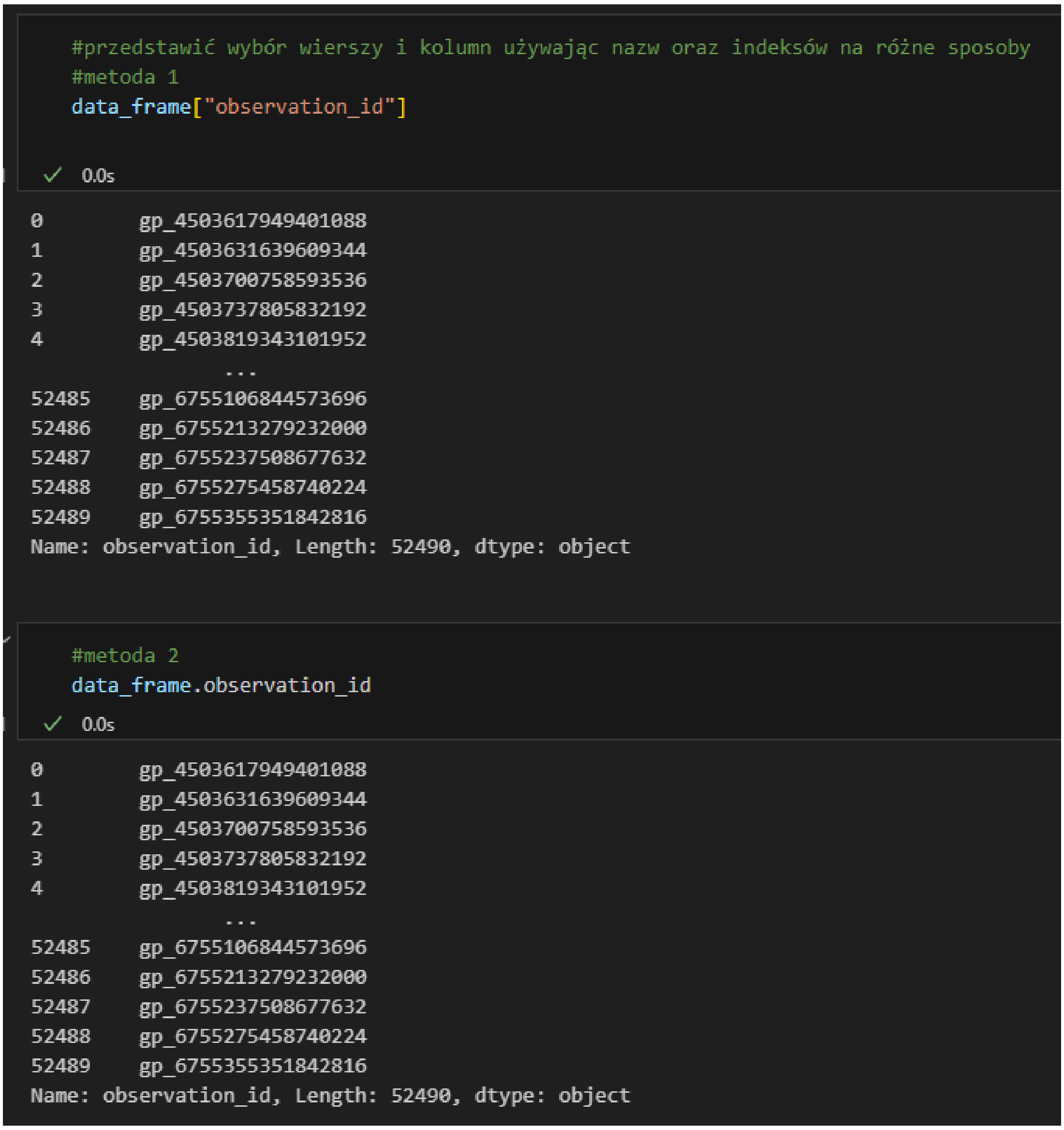
11 rows × 48 columns

- usunąć brakujące wartości w ramce danych



```
#usuwanie brakujących wartości w ramce danych
data_frame.dropna(inplace=True)
```

- przedstawić wybór wierszy i kolumny używając nazw oraz indeksów na różne sposoby



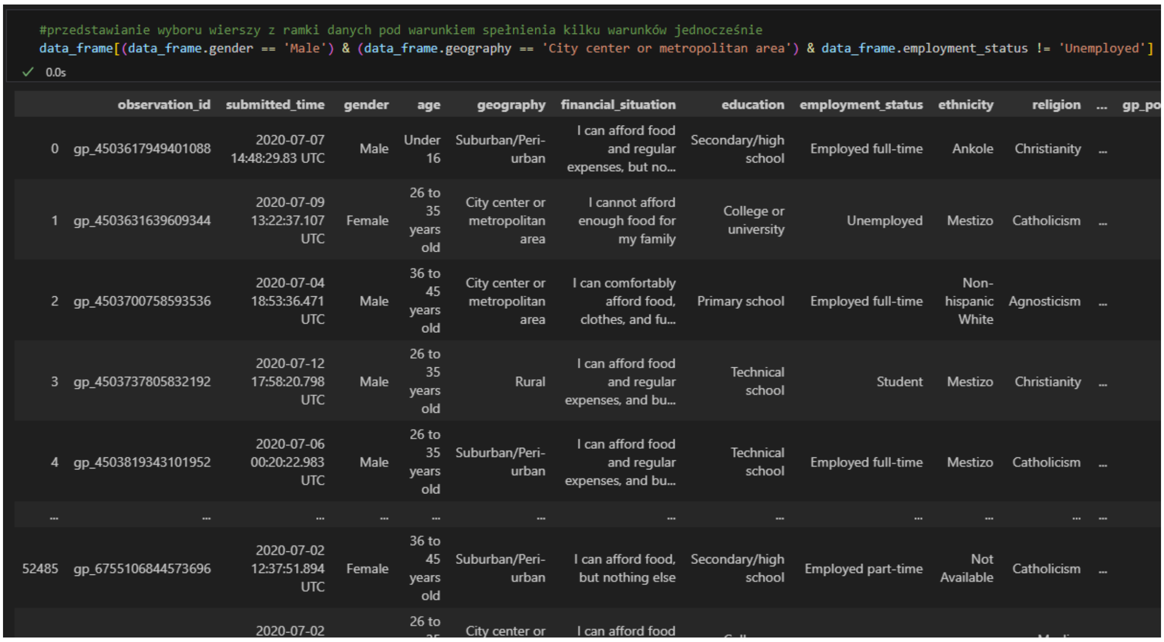
```
#przedstawić wybór wierszy i kolumn używając nazw oraz indeksów na różne sposoby
#metoda 1
data_frame["observation_id"]
```

```
0        gp_4503617949401088
1        gp_4503631639609344
2        gp_4503700758593536
3        gp_4503737805832192
4        gp_4503819343101952
                ...
52485    gp_6755106844573696
52486    gp_6755213279232000
52487    gp_6755237508677632
52488    gp_6755275458740224
52489    gp_6755355351842816
Name: observation_id, Length: 52490, dtype: object
```

```
#metoda 2
data_frame.observation_id
```

```
0        gp_4503617949401088
1        gp_4503631639609344
2        gp_4503700758593536
3        gp_4503737805832192
4        gp_4503819343101952
                ...
52485    gp_6755106844573696
52486    gp_6755213279232000
52487    gp_6755237508677632
52488    gp_6755275458740224
52489    gp_6755355351842816
Name: observation_id, Length: 52490, dtype: object
```

- przedstawić wybór wierszy z ramki danych pod warunkiem odnośnie określonej wartości kolumny, przedstawić wybór wierszy z ramki danych pod warunkiem spełnienia, kilku warunków jednocześnie

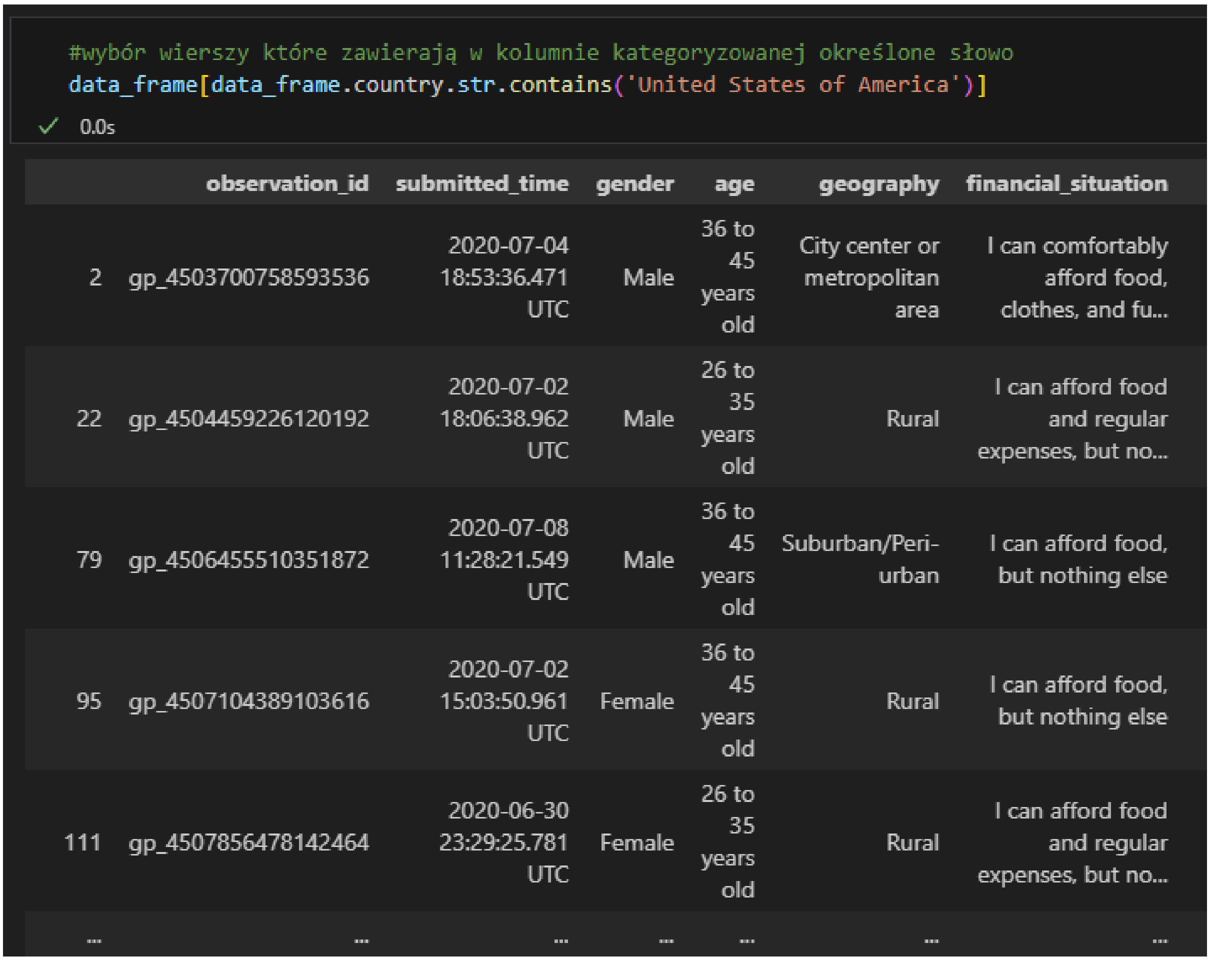


```
#przedstawianie wyboru wierszy z ramki danych pod warunkiem spełnienia kilku warunków jednocześnie
data_frame[(data_frame.gender == 'Male') & (data_frame.geography == 'City center or metropolitan area') & data_frame.employment_status != 'Unemployed']
```

✓ 0.0s

	observation_id	submitted_time	gender	age	geography	financial_situation	education	employment_status	ethnicity	religion	...	gp_po
0	gp_4503617949401088	2020-07-07 14:48:29.83 UTC	Male	Under 16	Suburban/Peri-urban	I can afford food and regular expenses, but no...	Secondary/high school	Employed full-time	Ankole	Christianity	...	
1	gp_4503631639609344	2020-07-09 13:22:37.107 UTC	Female	26 to 35 years old	City center or metropolitan area	I cannot afford enough food for my family	College or university	Unemployed	Mestizo	Catholicism	...	
2	gp_4503700758593536	2020-07-04 18:53:36.471 UTC	Male	36 to 45 years old	City center or metropolitan area	I can comfortably afford food, clothes, and fu...	Primary school	Employed full-time	Non-hispanic White	Agnosticism	...	
3	gp_4503737805832192	2020-07-12 17:58:20.798 UTC	Male	26 to 35 years old	Rural	I can afford food and regular expenses, and bu...	Technical school	Student	Mestizo	Christianity	...	
4	gp_4503819343101952	2020-07-06 00:20:22.983 UTC	Male	26 to 35 years old	Suburban/Peri-urban	I can afford food and regular expenses, and bu...	Technical school	Employed full-time	Mestizo	Catholicism	...	
...	...	...	...	...	...	...	...	...	...	...	...	
52485	gp_6755106844573696	2020-07-02 12:37:51.894 UTC	Female	36 to 45 years old	Suburban/Peri-urban	I can afford food, but nothing else	Secondary/high school	Employed part-time	Not Available	Catholicism	...	
		2020-07-02		26 to	City center or	I can afford food						

- wybrać wiersze które zawierają w kolumnie kategoryzowanej określone słowo

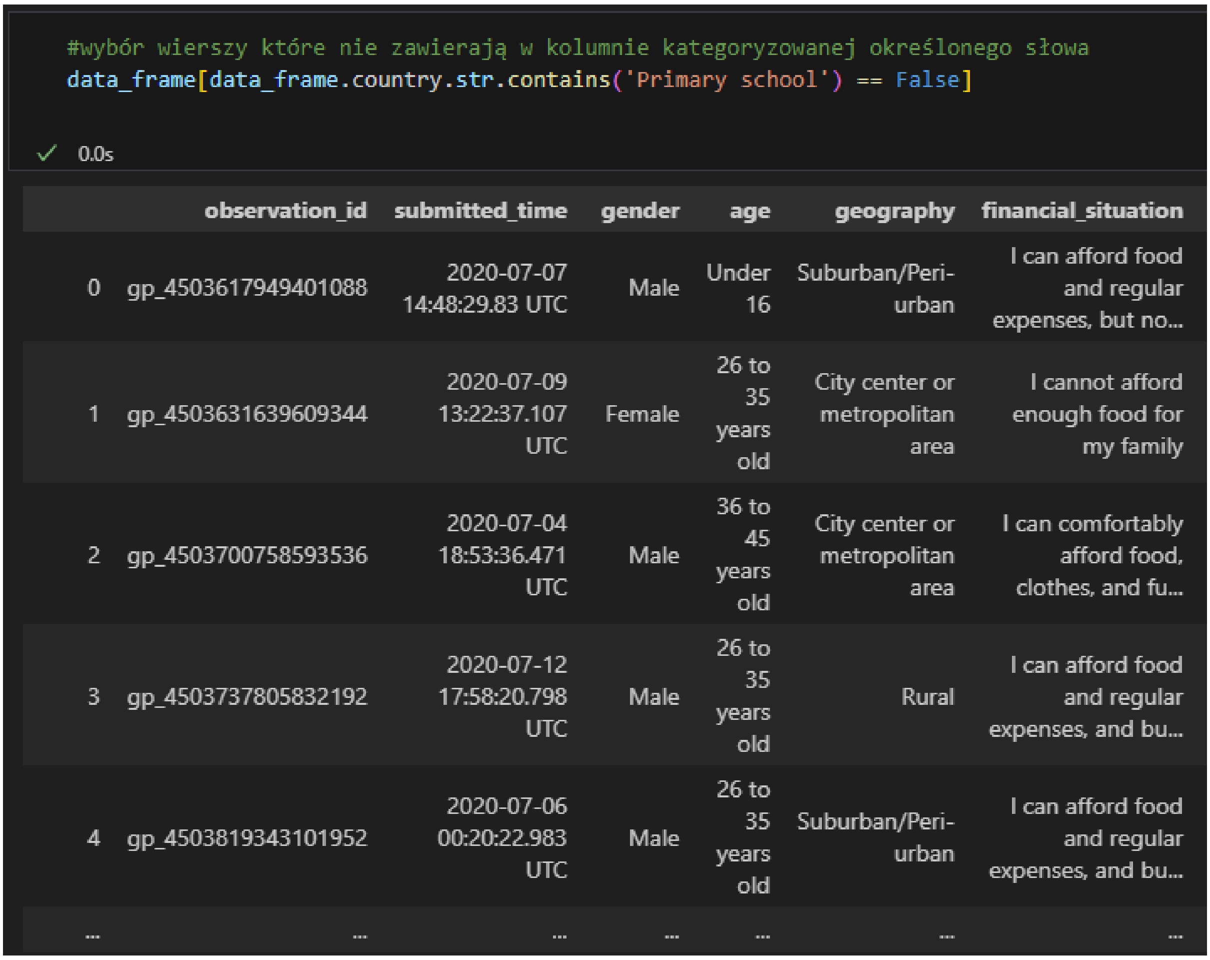


```
#wybór wierszy które zawierają w kolumnie kategoryzowanej określone słowo
data_frame[data_frame.country.str.contains('United States of America')]
```

✓ 0.0s

	observation_id	submitted_time	gender	age	geography	financial_situation
2	gp_4503700758593536	2020-07-04 18:53:36.471 UTC	Male	36 to 45 years old	City center or metropolitan area	I can comfortably afford food, clothes, and fu...
22	gp_4504459226120192	2020-07-02 18:06:38.962 UTC	Male	26 to 35 years old	Rural	I can afford food and regular expenses, but no...
79	gp_4506455510351872	2020-07-08 11:28:21.549 UTC	Male	36 to 45 years old	Suburban/Peri-urban	I can afford food, but nothing else
95	gp_4507104389103616	2020-07-02 15:03:50.961 UTC	Female	36 to 45 years old	Rural	I can afford food, but nothing else
111	gp_4507856478142464	2020-06-30 23:29:25.781 UTC	Female	26 to 35 years old	Rural	I can afford food and regular expenses, but no...
...	...	...	...	...	...	...

- wybrać wiersze które nie zawierają w kolumnie kategoryzowanej określone słowo

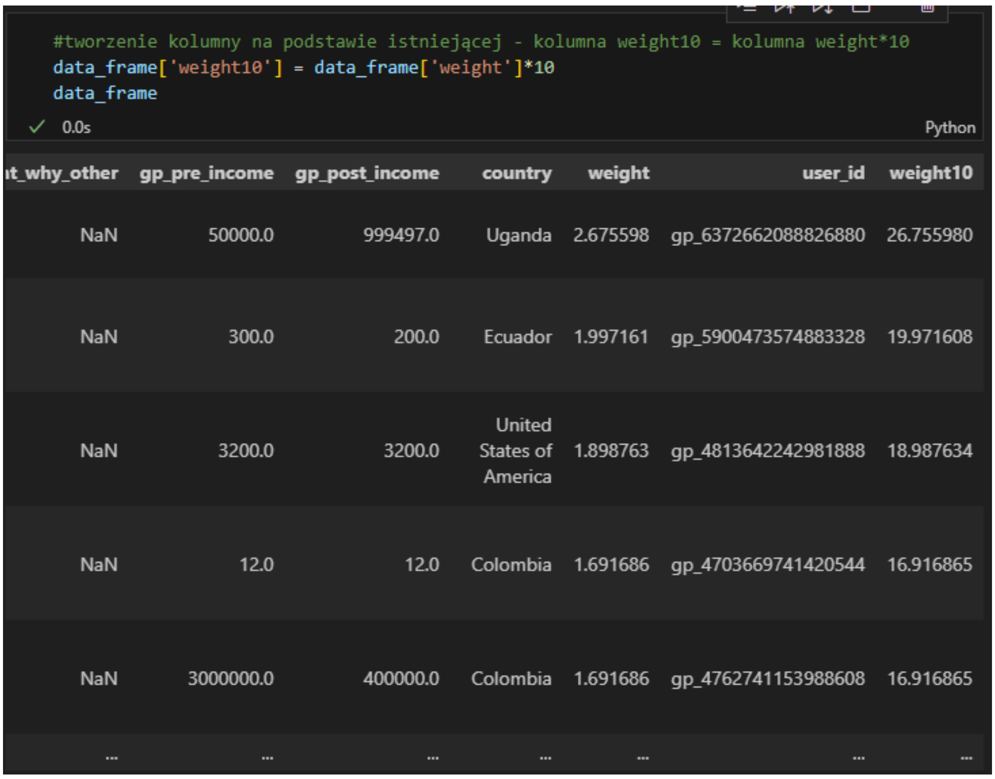


```
#wybór wierszy które nie zawierają w kolumnie kategoryzowanej określonego słowa
data_frame[data_frame.country.str.contains('Primary school') == False]
```

✓ 0.0s

	observation_id	submitted_time	gender	age	geography	financial_situation
0	gp_4503617949401088	2020-07-07 14:48:29.83 UTC	Male	Under 16	Suburban/Peri-urban	I can afford food and regular expenses, but no...
1	gp_4503631639609344	2020-07-09 13:22:37.107 UTC	Female	26 to 35 years old	City center or metropolitan area	I cannot afford enough food for my family
2	gp_4503700758593536	2020-07-04 18:53:36.471 UTC	Male	36 to 45 years old	City center or metropolitan area	I can comfortably afford food, clothes, and fu...
3	gp_4503737805832192	2020-07-12 17:58:20.798 UTC	Male	26 to 35 years old	Rural	I can afford food and regular expenses, and bu...
4	gp_4503819343101952	2020-07-06 00:20:22.983 UTC	Male	26 to 35 years old	Suburban/Peri-urban	I can afford food and regular expenses, and bu...
...	...	...	...	...	...	...

- utwórz kolumnę na podstawie istniejącym

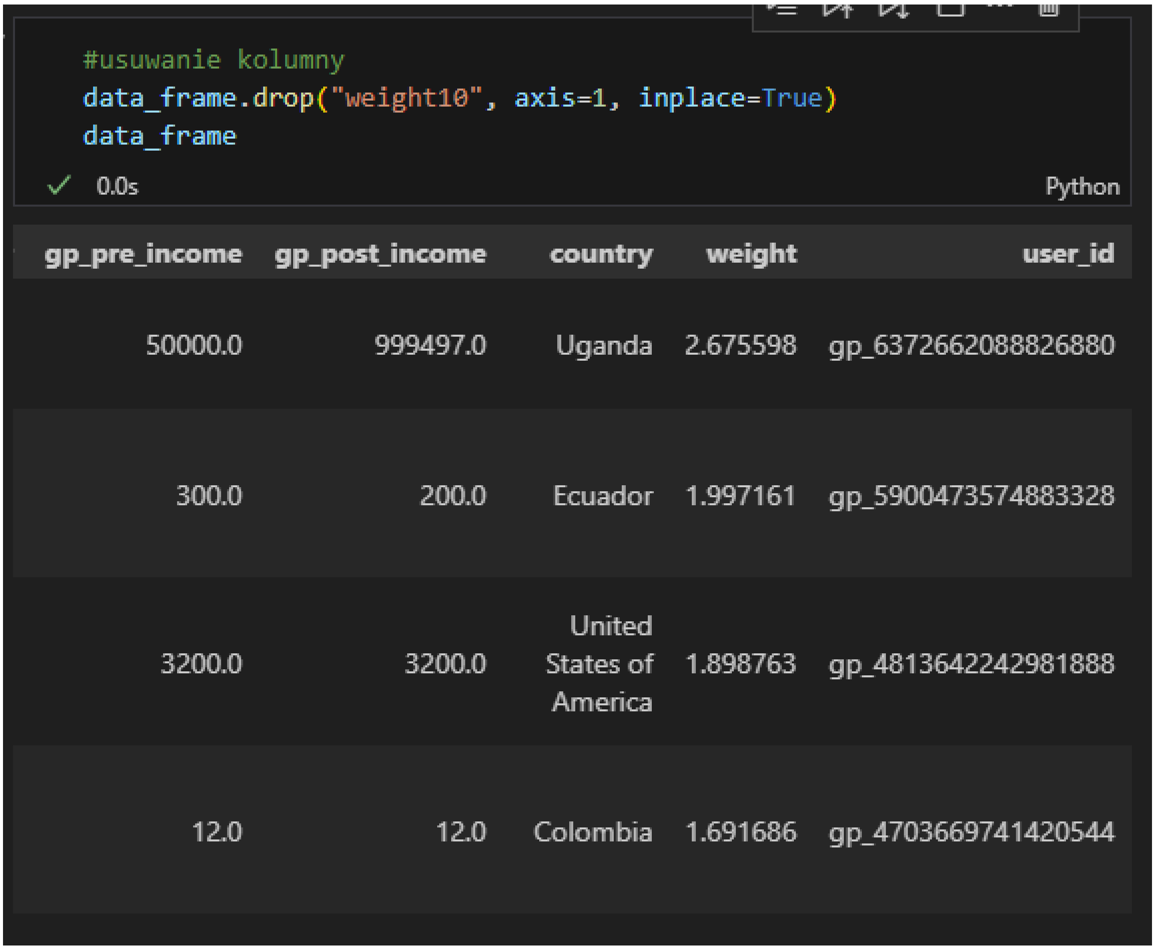


```
#tworzenie kolumny na podstawie istniejącej - kolumna weight10 = kolumna weight*10
data_frame['weight10'] = data_frame['weight']*10
data_frame
```

✓ 0.0s Python

t_why_other	gp_pre_income	gp_post_income	country	weight	user_id	weight10
NaN	50000.0	999497.0	Uganda	2.675598	gp_6372662088826880	26.755980
NaN	300.0	200.0	Ecuador	1.997161	gp_5900473574883328	19.971608
NaN	3200.0	3200.0	United States of America	1.898763	gp_4813642242981888	18.987634
NaN	12.0	12.0	Colombia	1.691686	gp_4703669741420544	16.916865
NaN	3000000.0	400000.0	Colombia	1.691686	gp_4762741153988608	16.916865
...	...	...	...	...	...	...

- usuń kolumnę

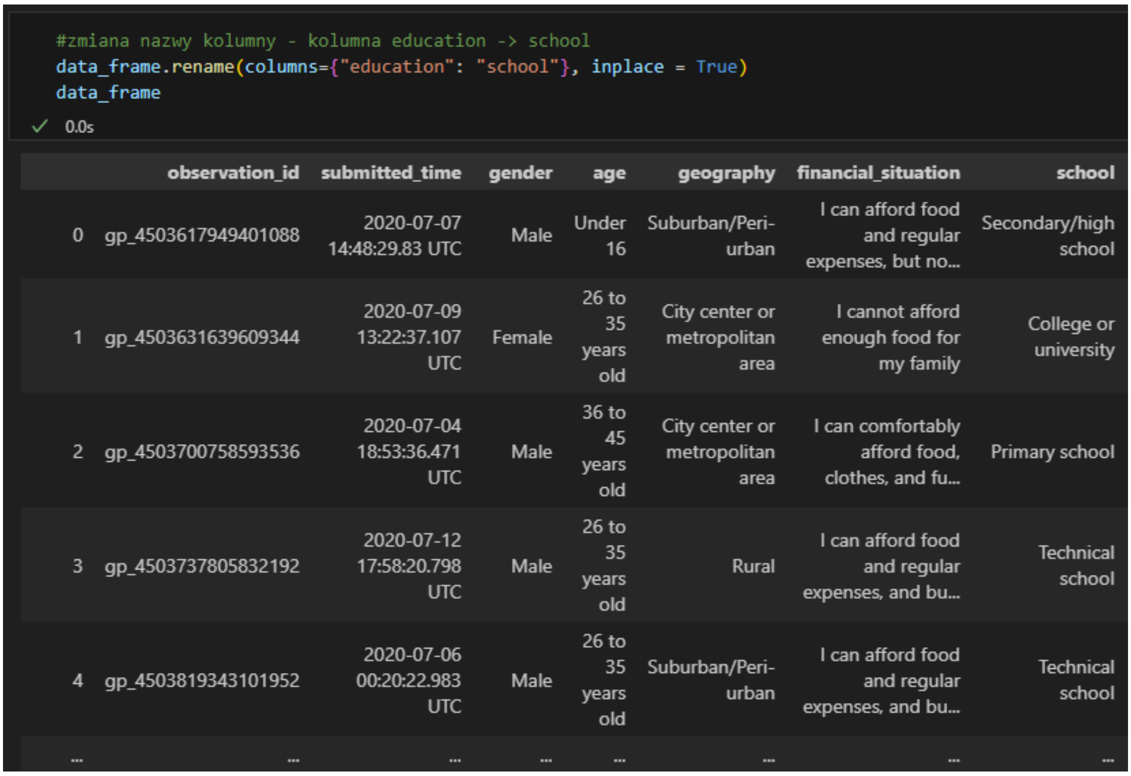


```
#usuwanie kolumny
data_frame.drop("weight10", axis=1, inplace=True)
data_frame
```

✓ 0.0s Python

gp_pre_income	gp_post_income	country	weight	user_id
50000.0	999497.0	Uganda	2.675598	gp_6372662088826880
300.0	200.0	Ecuador	1.997161	gp_5900473574883328
3200.0	3200.0	United States of America	1.898763	gp_4813642242981888
12.0	12.0	Colombia	1.691686	gp_4703669741420544

- zmień nazwę¸ kolumny

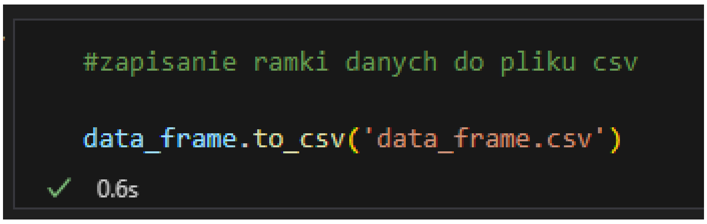


```
#zmiana nazwy kolumny - kolumna education -> school
data_frame.rename(columns={"education": "school"}, inplace = True)
data_frame
```

✓ 0.0s

	observation_id	submitted_time	gender	age	geography	financial_situation	school
0	gp_4503617949401088	2020-07-07 14:48:29.83 UTC	Male	Under 16	Suburban/Peri-urban	I can afford food and regular expenses, but no...	Secondary/high school
1	gp_4503631639609344	2020-07-09 13:22:37.107 UTC	Female	26 to 35 years old	City center or metropolitan area	I cannot afford enough food for my family	College or university
2	gp_4503700758593536	2020-07-04 18:53:36.471 UTC	Male	36 to 45 years old	City center or metropolitan area	I can comfortably afford food, clothes, and fu...	Primary school
3	gp_4503737805832192	2020-07-12 17:58:20.798 UTC	Male	26 to 35 years old	Rural	I can afford food and regular expenses, and bu...	Technical school
4	gp_4503819343101952	2020-07-06 00:20:22.983 UTC	Male	26 to 35 years old	Suburban/Peri-urban	I can afford food and regular expenses, and bu...	Technical school
...	...	...	...	...	...	...	...

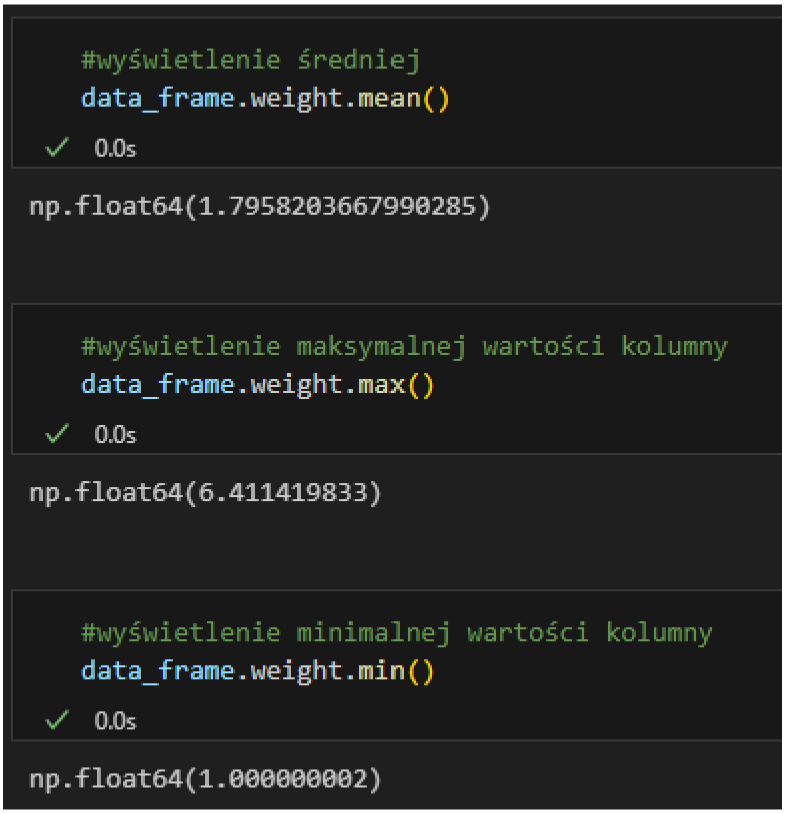
- zachowaj ramkę danych jako plik csv na komputerze



```
#zapisanie ramki danych do pliku csv

data_frame.to_csv('data_frame.csv')
```

✓ 0.6s

- wyświetlić średnia (maksymalną, minimalną) wartość z jednej kolumny



```
#wyświetlenie średniej
data_frame.weight.mean()
```

✓ 0.0s

```
np.float64(1.7958203667990285)
```

```
#wyświetlenie maksymalnej wartości kolumny
data_frame.weight.max()
```

✓ 0.0s

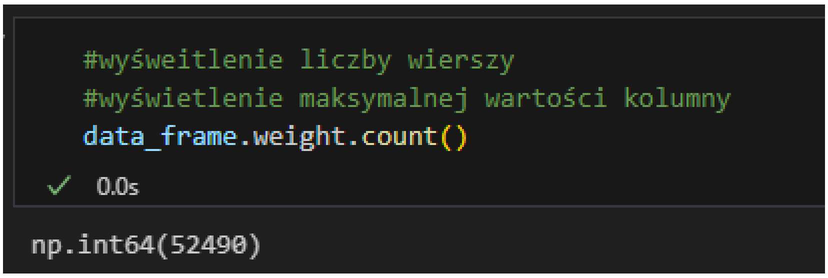
```
np.float64(6.411419833)
```

```
#wyświetlenie minimalnej wartości kolumny
data_frame.weight.min()
```

✓ 0.0s

```
np.float64(1.000000002)
```

- wyświetlić liczbę wierszy

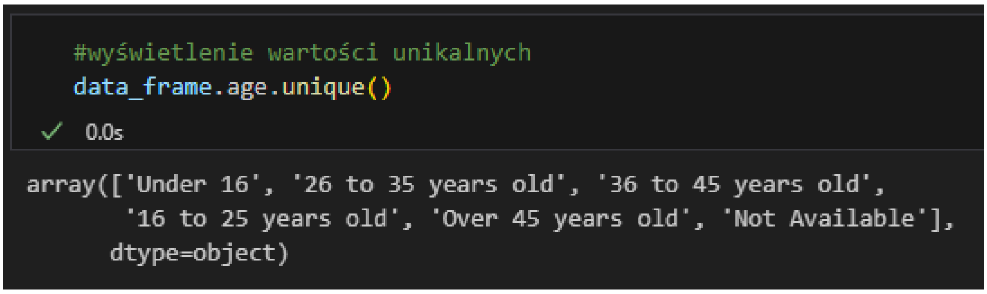


```
#wyświeitlenie liczby wierszy
#wyświetlenie maksymalnej wartości kolumny
data_frame.weight.count()
```

✓ 0.0s

```
np.int64(52490)
```

- wyświetlić wartości unikatowe w kolumnie

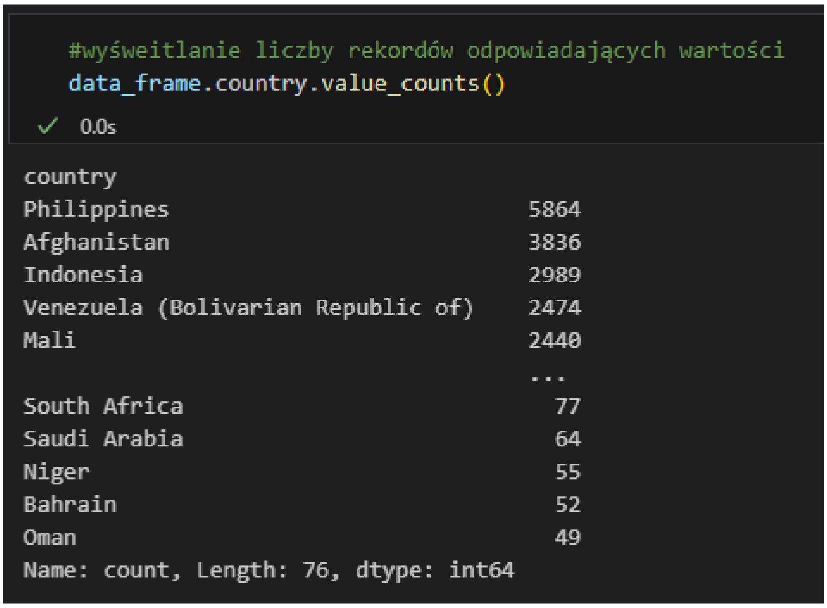


```
#wyświetlenie wartości unikalnych
data_frame.age.unique()
```

✓ 0.0s

```
array(['Under 16', '26 to 35 years old', '36 to 45 years old',
       '16 to 25 years old', 'Over 45 years old', 'Not Available'],
      dtype=object)
```

- wyświetlić liczby rekordów odpowiadających do wartości

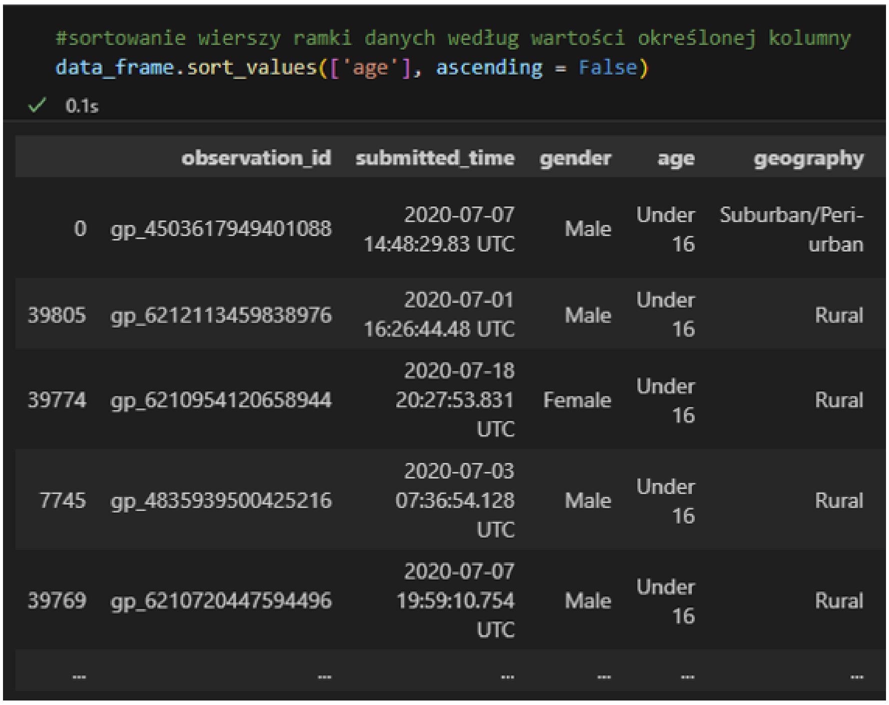


```
#wyśweitlanie liczby rekordów odpowiadających wartości
data_frame.country.value_counts()
```

✓ 0.0s

```
country
Philippines                      5864
Afghanistan                      3836
Indonesia                        2989
Venezuela (Bolivarian Republic of)    2474
Mali                             2440
                                 ...
South Africa                       77
Saudi Arabia                       64
Niger                              55
Bahrain                            52
Oman                               49
Name: count, Length: 76, dtype: int64
```

- sortowanie wierszy ramki danych według wartości określonej kolumny (malejąco, rosnąco)

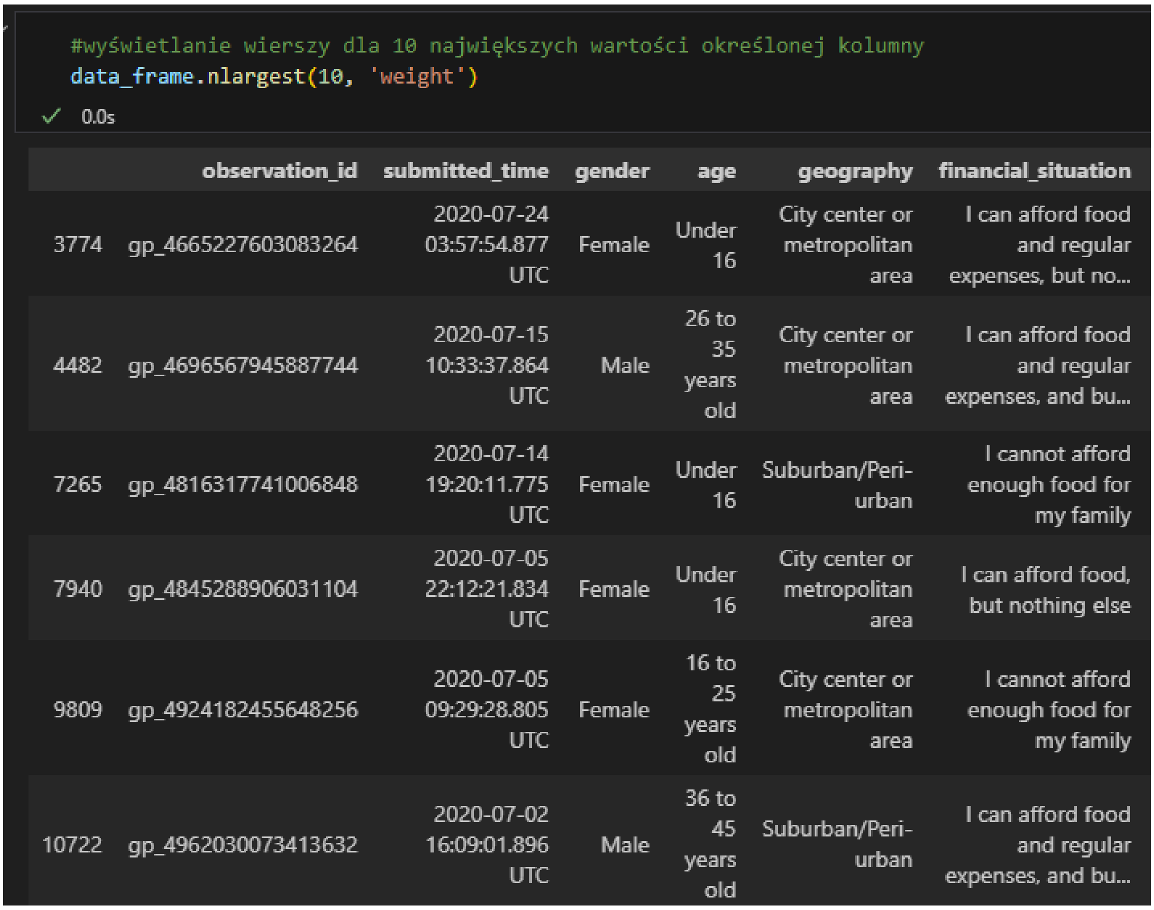


```
#sortowanie wierszy ramki danych według wartości określonej kolumny
data_frame.sort_values(['age'], ascending = False)
```

✓ 0.1s

	observation_id	submitted_time	gender	age	geography
0	gp_4503617949401088	2020-07-07 14:48:29.83 UTC	Male	Under 16	Suburban/Peri-urban
39805	gp_6212113459838976	2020-07-01 16:26:44.48 UTC	Male	Under 16	Rural
39774	gp_6210954120658944	2020-07-18 20:27:53.831 UTC	Female	Under 16	Rural
7745	gp_4835939500425216	2020-07-03 07:36:54.128 UTC	Male	Under 16	Rural
39769	gp_6210720447594496	2020-07-07 19:59:10.754 UTC	Male	Under 16	Rural
...	...	...	...	...	...

- wyświetlić wierszy dla 10 największych (najmniejszych) wartości określonej kolumny

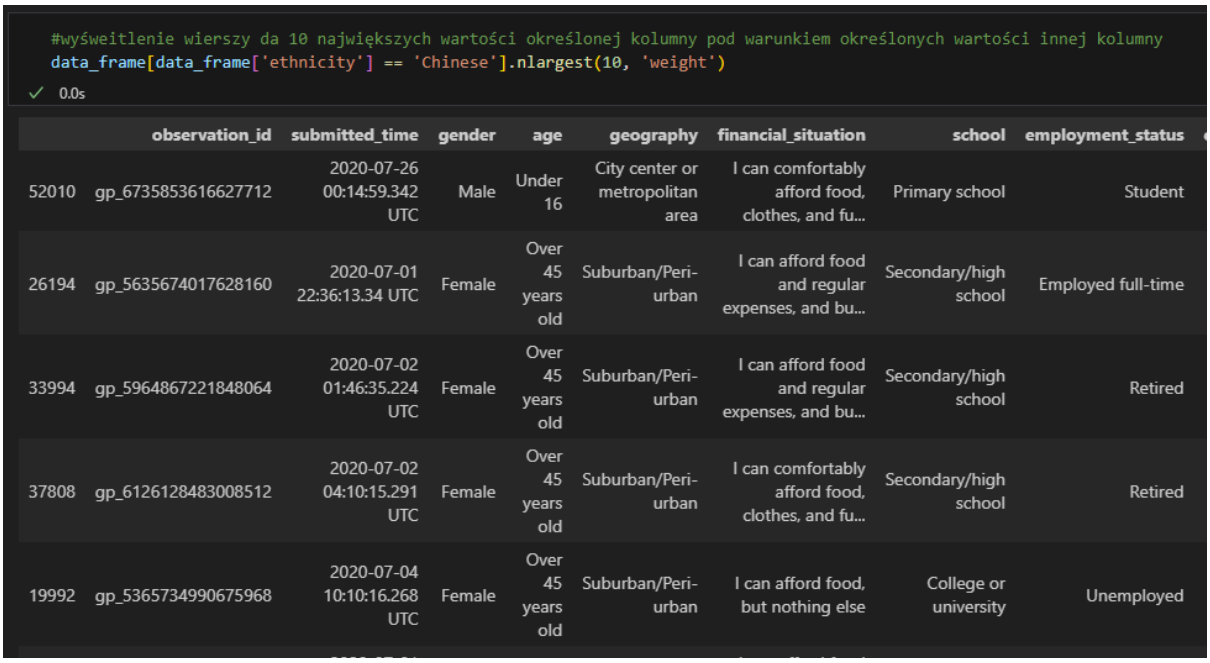


```
#wyświetlanie wierszy dla 10 największych wartości określonej kolumny
data_frame.nlargest(10, 'weight')
```

✓ 0.0s

	observation_id	submitted_time	gender	age	geography	financial_situation
3774	gp_4665227603083264	2020-07-24 03:57:54.877 UTC	Female	Under 16	City center or metropolitan area	I can afford food and regular expenses, but no...
4482	gp_4696567945887744	2020-07-15 10:33:37.864 UTC	Male	26 to 35 years old	City center or metropolitan area	I can afford food and regular expenses, and bu...
7265	gp_4816317741006848	2020-07-14 19:20:11.775 UTC	Female	Under 16	Suburban/Peri-urban	I cannot afford enough food for my family
7940	gp_4845288906031104	2020-07-05 22:12:21.834 UTC	Female	Under 16	City center or metropolitan area	I can afford food, but nothing else
9809	gp_4924182455648256	2020-07-05 09:29:28.805 UTC	Female	16 to 25 years old	City center or metropolitan area	I cannot afford enough food for my family
10722	gp_4962030073413632	2020-07-02 16:09:01.896 UTC	Male	36 to 45 years old	Suburban/Peri-urban	I can afford food and regular expenses, and bu...

- wyświetlić wierszy dla 10 największych wartości określonej kolumny pod warunkiem określonych wartości innej kolumny

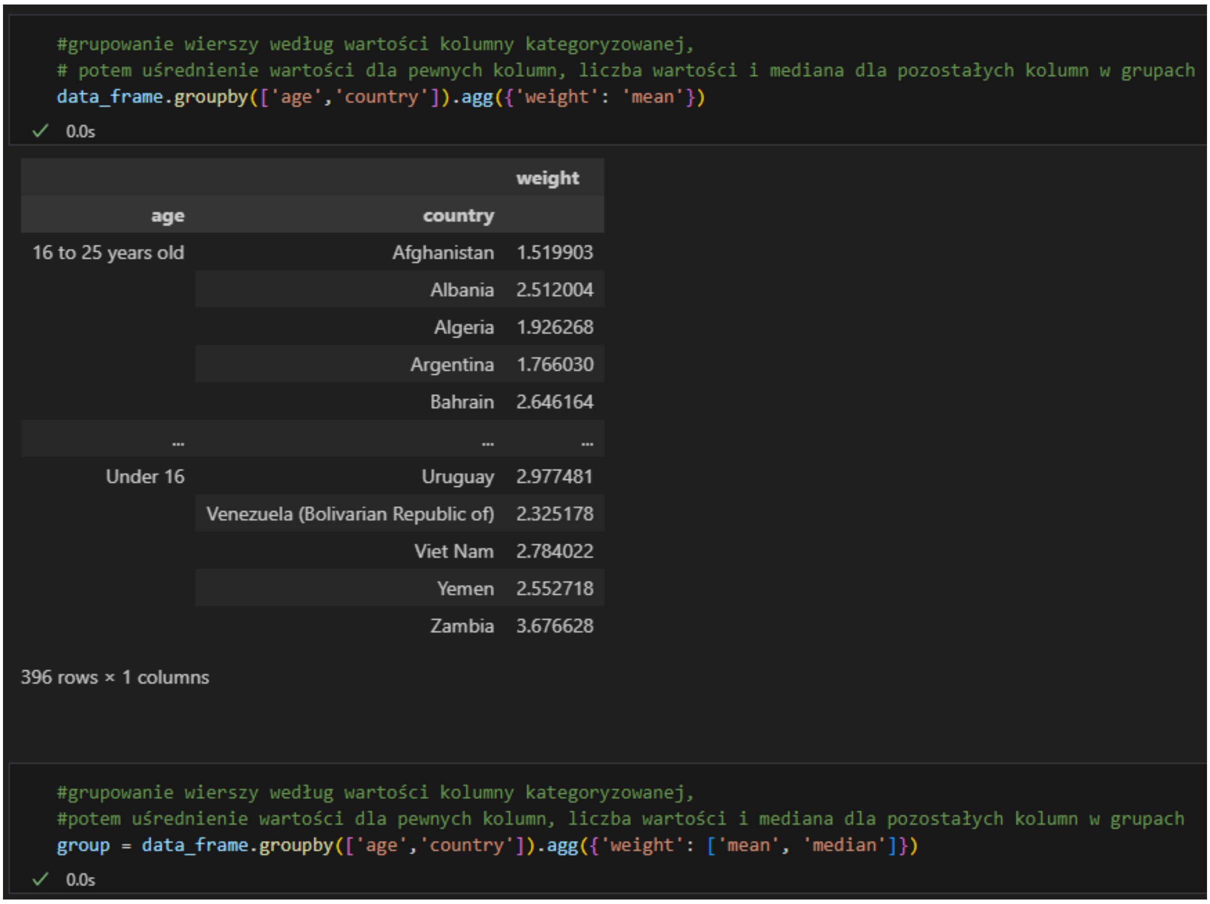


```
#wyśweitlenie wierszy da 10 największych wartości określonej kolumny pod warunkiem określonych wartości innej kolumny
data_frame[data_frame['ethnicity'] == 'Chinese'].nlargest(10, 'weight')
```

✓ 0.0s

	observation_id	submitted_time	gender	age	geography	financial_situation	school	employment_status
52010	gp_6735853616627712	2020-07-26 00:14:59.342 UTC	Male	Under 16	City center or metropolitan area	I can comfortably afford food, clothes, and fu...	Primary school	Student
26194	gp_5635674017628160	2020-07-01 22:36:13.34 UTC	Female	Over 45 years old	Suburban/Peri-urban	I can afford food and regular expenses, and bu...	Secondary/high school	Employed full-time
33994	gp_5964867221848064	2020-07-02 01:46:35.224 UTC	Female	Over 45 years old	Suburban/Peri-urban	I can afford food and regular expenses, and bu...	Secondary/high school	Retired
37808	gp_6126128483008512	2020-07-02 04:10:15.291 UTC	Female	Over 45 years old	Suburban/Peri-urban	I can comfortably afford food, clothes, and fu...	Secondary/high school	Retired
19992	gp_5365734990675968	2020-07-04 10:10:16.268 UTC	Female	Over 45 years old	Suburban/Peri-urban	I can afford food, but nothing else	College or university	Unemployed

- grupowanie wierszy według wartości kolumny kategoryzowanej, potem - uśrednienie wartości dla pewnych kolumn, liczba wartości i mediana dla pozostałych kolumn w grupach



```
#grupowanie wierszy według wartości kolumny kategoryzowanej,
# potem uśrednienie wartości dla pewnych kolumn, liczba wartości i mediana dla pozostałych kolumn w grupach
data_frame.groupby(['age','country']).agg({'weight': 'mean'})
```

✓ 0.0s

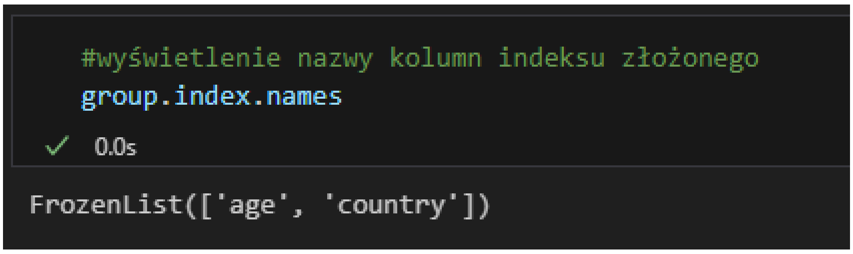
		weight
age	country	
16 to 25 years old	Afghanistan	1.519903
	Albania	2.512004
	Algeria	1.926268
	Argentina	1.766030
	Bahrain	2.646164
...	...	...
Under 16	Uruguay	2.977481
	Venezuela (Bolivarian Republic of)	2.325178
	Viet Nam	2.784022
	Yemen	2.552718
	Zambia	3.676628

396 rows × 1 columns

```
#grupowanie wierszy według wartości kolumny kategoryzowanej,
#potem uśrednienie wartości dla pewnych kolumn, liczba wartości i mediana dla pozostałych kolumn w grupach
group = data_frame.groupby(['age','country']).agg({'weight': ['mean', 'median']})
```

✓ 0.0s

- wyświetlić nazwy kolumn indeksu złożonego

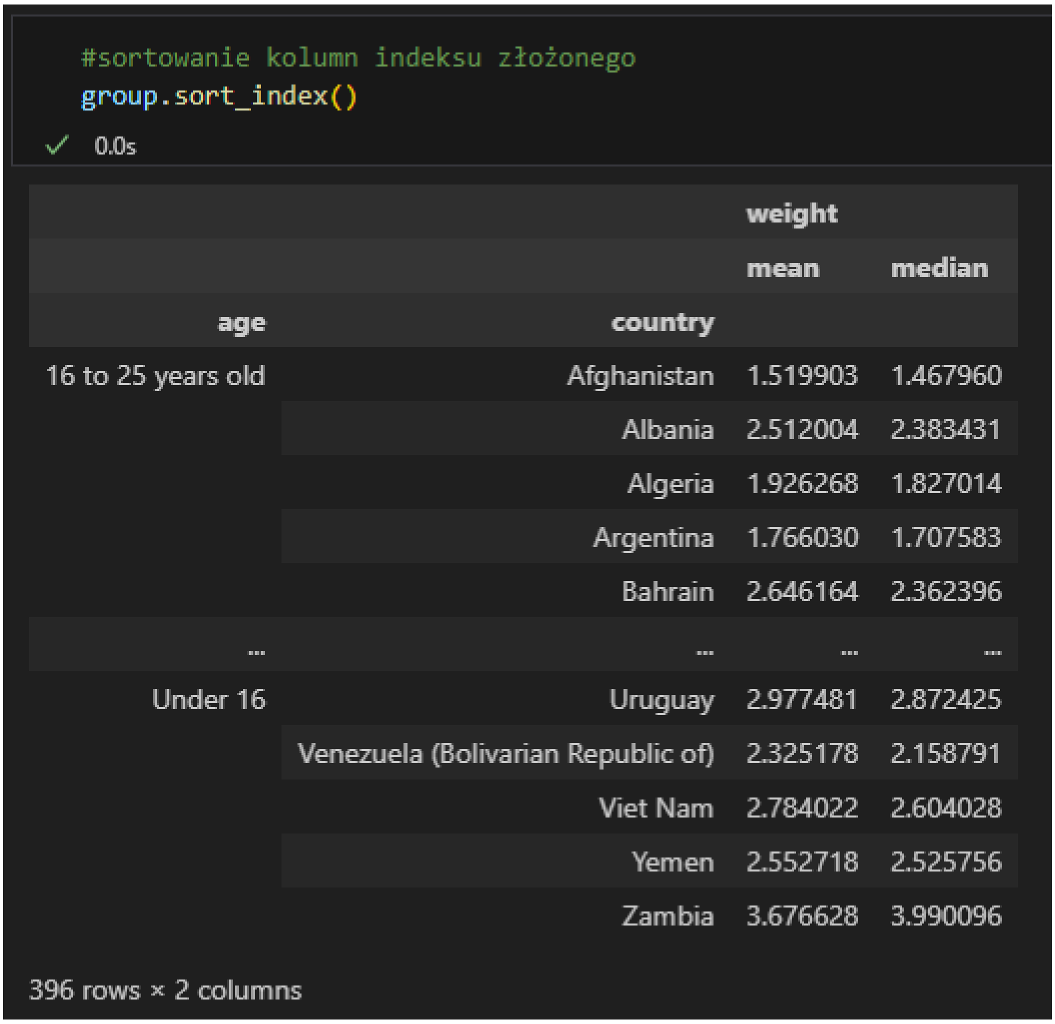


```
#wyświetlenie nazwy kolumn indeksu złożonego
group.index.names
```

✓ 0.0s

```
FrozenList(['age', 'country'])
```

- sortować kolumnę indeksu złożonego



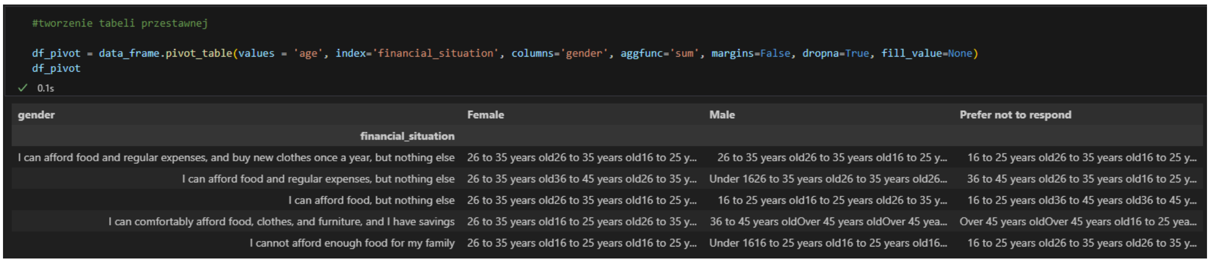
```
#sortowanie kolumn indeksu złożonego
group.sort_index()
```

✓ 0.0s

		weight	
		mean	median
age	country		
16 to 25 years old	Afghanistan	1.519903	1.467960
	Albania	2.512004	2.383431
	Algeria	1.926268	1.827014
	Argentina	1.766030	1.707583
	Bahrain	2.646164	2.362396
...	...	...	...
Under 16	Uruguay	2.977481	2.872425
	Venezuela (Bolivarian Republic of)	2.325178	2.158791
	Viet Nam	2.784022	2.604028
	Yemen	2.552718	2.525756
	Zambia	3.676628	3.990096

396 rows × 2 columns

- stworzyć tabelę przystawna̧ (pivot table) na podstawie ramki danych

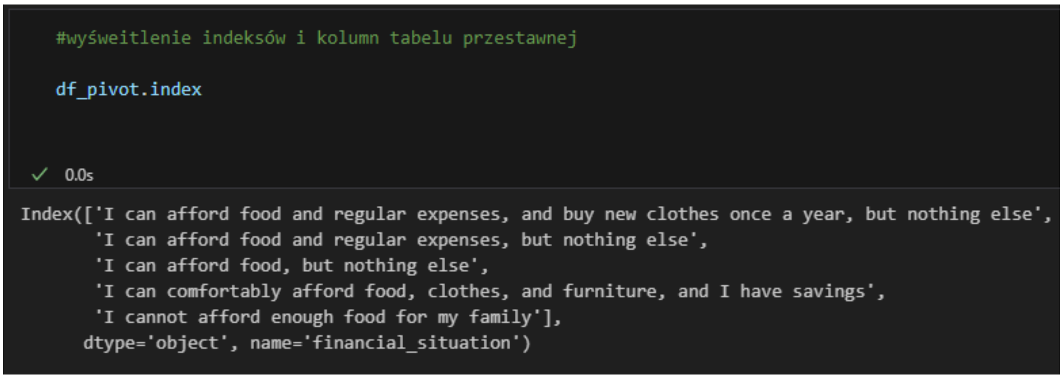


```
#tworzenie tabeli przestawnej

df_pivot = data_frame.pivot_table(values = 'age', index='financial_situation', columns='gender', aggfunc='sum', margins=False, dropna=True, fill_value=None)
df_pivot
```

✓ 0.1s

gender	Female	Male	Prefer not to respond
financial_situation			
I can afford food and regular expenses, and buy new clothes once a year, but nothing else	26 to 35 years old26 to 35 years old16 to 25 y...	26 to 35 years old26 to 35 years old16 to 25 y...	16 to 25 years old26 to 35 years old16 to 25 y...
I can afford food and regular expenses, but nothing else	26 to 35 years old36 to 45 years old26 to 35 y...	Under 1626 to 35 years old26 to 35 years old26...	36 to 45 years old26 to 35 years old16 to 25 y...
I can afford food, but nothing else	26 to 35 years old26 to 35 years old16 to 25 y...	16 to 25 years old16 to 25 years old26 to 35 y...	16 to 25 years old36 to 45 years old36 to 45 y...
I can comfortably afford food, clothes, and furniture, and I have savings	26 to 35 years old16 to 25 years old26 to 35 y...	36 to 45 years oldOver 45 years oldOver 45 yea...	Over 45 years oldOver 45 years old16 to 25 yea...
I cannot afford enough food for my family	26 to 35 years old16 to 25 years old16 to 25 y...	Under 1616 to 25 years old16 to 25 years old16...	16 to 25 years old26 to 35 years old26 to 35 y...

- wyświetlić indeksy i kolumny tabeli przystawnej

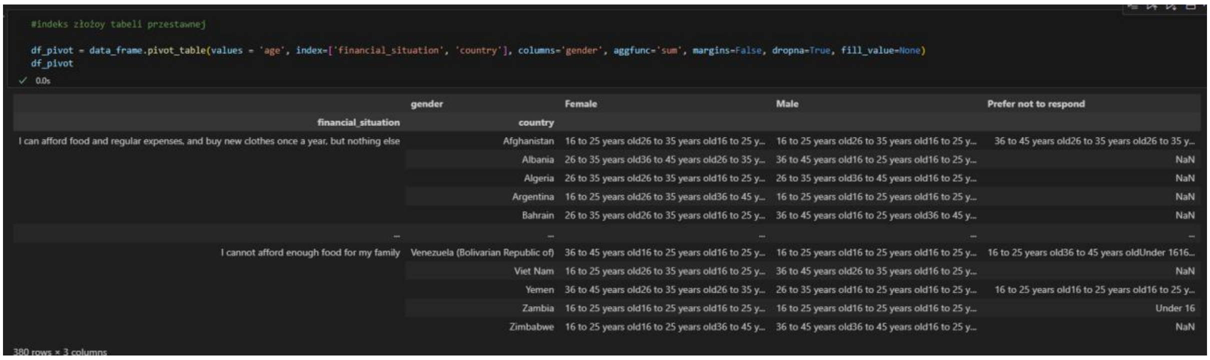


```
#wyśweitlenie indeksów i kolumn tabelu przestawnej

df_pivot.index
```

✓ 0.0s

```
Index(['I can afford food and regular expenses, and buy new clothes once a year, but nothing else',
       'I can afford food and regular expenses, but nothing else',
       'I can afford food, but nothing else',
       'I can comfortably afford food, clothes, and furniture, and I have savings',
       'I cannot afford enough food for my family'],
      dtype='object', name='financial_situation')
```

- utwórz indeks złożony tabeli przystawnej i wyświetl go



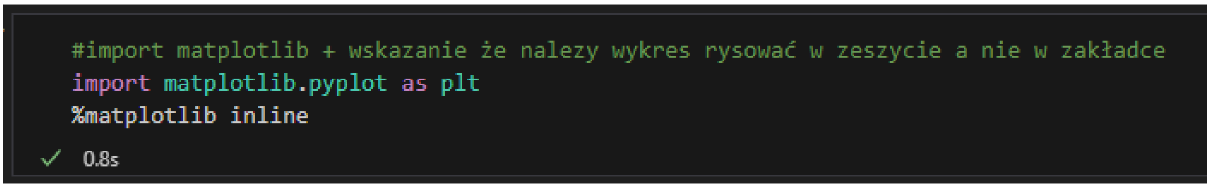
```
#indeks złożoy tabeli przestawnej

df_pivot = data_frame.pivot_table(values = 'age', index=['financial_situation', 'country'], columns='gender', aggfunc='sum', margins=False, dropna=True, fill_value=None)
df_pivot
```

✓ 0.0s

	gender	Female	Male	Prefer not to respond
financial_situation	country			
I can afford food and regular expenses, and buy new clothes once a year, but nothing else	Afghanistan	16 to 25 years old26 to 35 years old16 to 25 y...	16 to 25 years old26 to 35 years old16 to 25 y...	36 to 45 years old26 to 35 years old26 to 35 y...
	Albania	26 to 35 years old36 to 45 years old26 to 35 y...	36 to 45 years old16 to 25 years old16 to 25 y...	NaN
	Algeria	26 to 35 years old26 to 35 years old16 to 25 y...	26 to 35 years old36 to 45 years old16 to 25 y...	NaN
	Argentina	16 to 25 years old26 to 35 years old36 to 45 y...	16 to 25 years old16 to 25 years old16 to 25 y...	NaN
	Bahrain	26 to 35 years old26 to 35 years old16 to 25 y...	36 to 45 years old16 to 25 years old36 to 45 y...	NaN
...	...	...	...	...
I cannot afford enough food for my family	Venezuela (Bolivarian Republic of)	36 to 45 years old16 to 25 years old16 to 25 y...	16 to 25 years old16 to 25 years old16 to 25 y...	16 to 25 years old36 to 45 years oldUnder 1616...
	Viet Nam	16 to 25 years old26 to 35 years old16 to 25 y...	36 to 45 years old26 to 35 years old16 to 25 y...	NaN
	Yemen	36 to 45 years old26 to 35 years old26 to 35 y...	26 to 35 years old16 to 25 years old16 to 25 y...	16 to 25 years old16 to 25 years old16 to 25 y...
	Zambia	16 to 25 years old16 to 25 years old16 to 25 y...	16 to 25 years old16 to 25 years old16 to 25 y...	Under 16
	Zimbabwe	16 to 25 years old16 to 25 years old36 to 45 y...	36 to 45 years old36 to 45 years old16 to 25 y...	NaN

380 rows × 3 columns

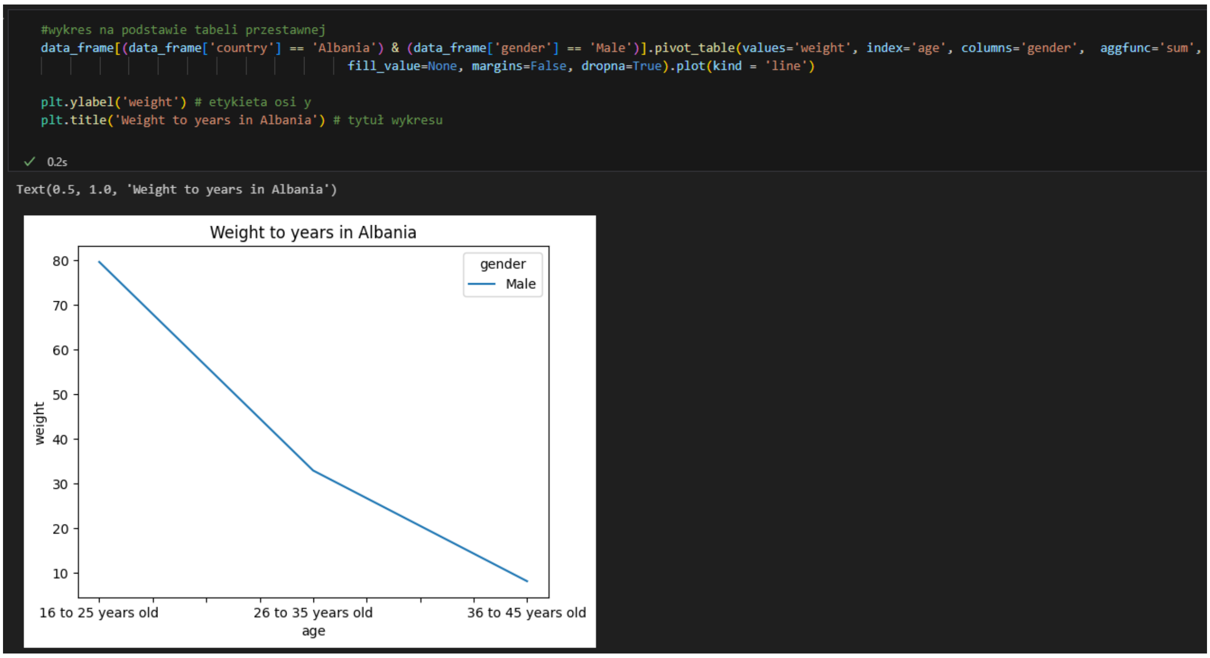
- zaimportuj moduł pyplot z biblioteki matplotlib, wskazać, że wykresy należy rysować bezpośrednio w zeszycie, a nie w osobnej zakładce



```
#import matplotlib + wskazanie że nalezy wykres rysować w zeszycie a nie w zakładce
import matplotlib.pyplot as plt
%matplotlib inline
```

✓ 0.8s

- wyświetlić wykres na podstawie tabeli przystawnej

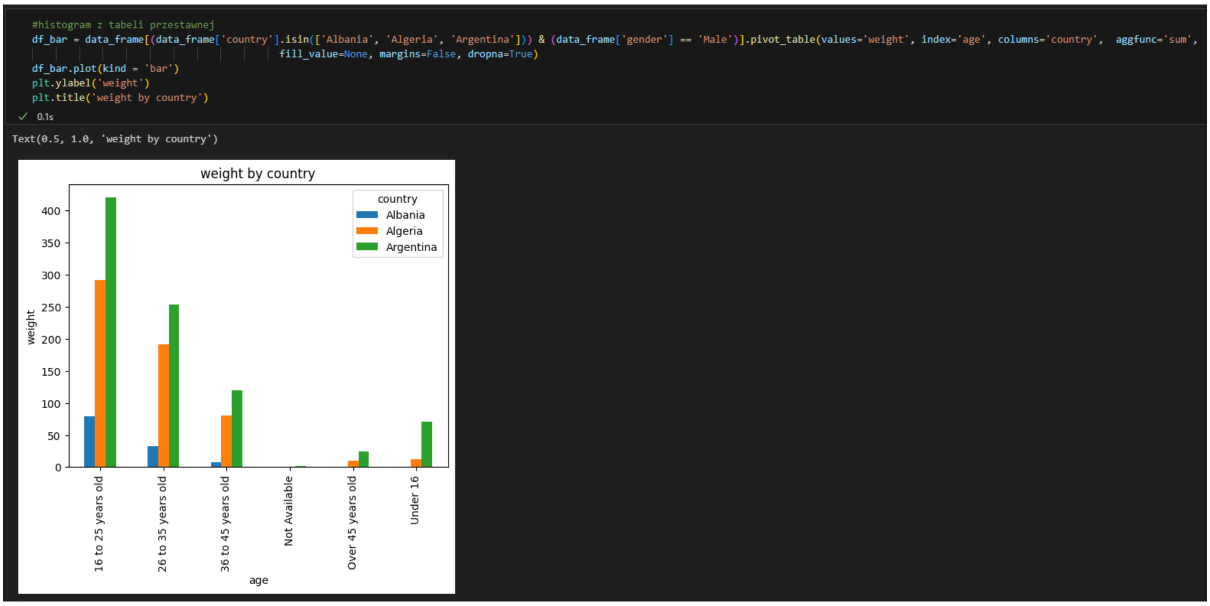


```
#wykres na podstawie tabeli przestawnej
data_frame[(data_frame['country'] == 'Albania') & (data_frame['gender'] == 'Male')].pivot_table(values='weight', index='age', columns='gender',  aggfunc='sum',
                                                                                            fill_value=None, margins=False, dropna=True).plot(kind = 'line')

plt.ylabel('weight') # etykieta osi y
plt.title('Weight to years in Albania') # tytuł wykresu
```

✓ 0.2s

```
Text(0.5, 1.0, 'Weight to years in Albania')
```

- narysować histogram na podstawie wartości kolumny

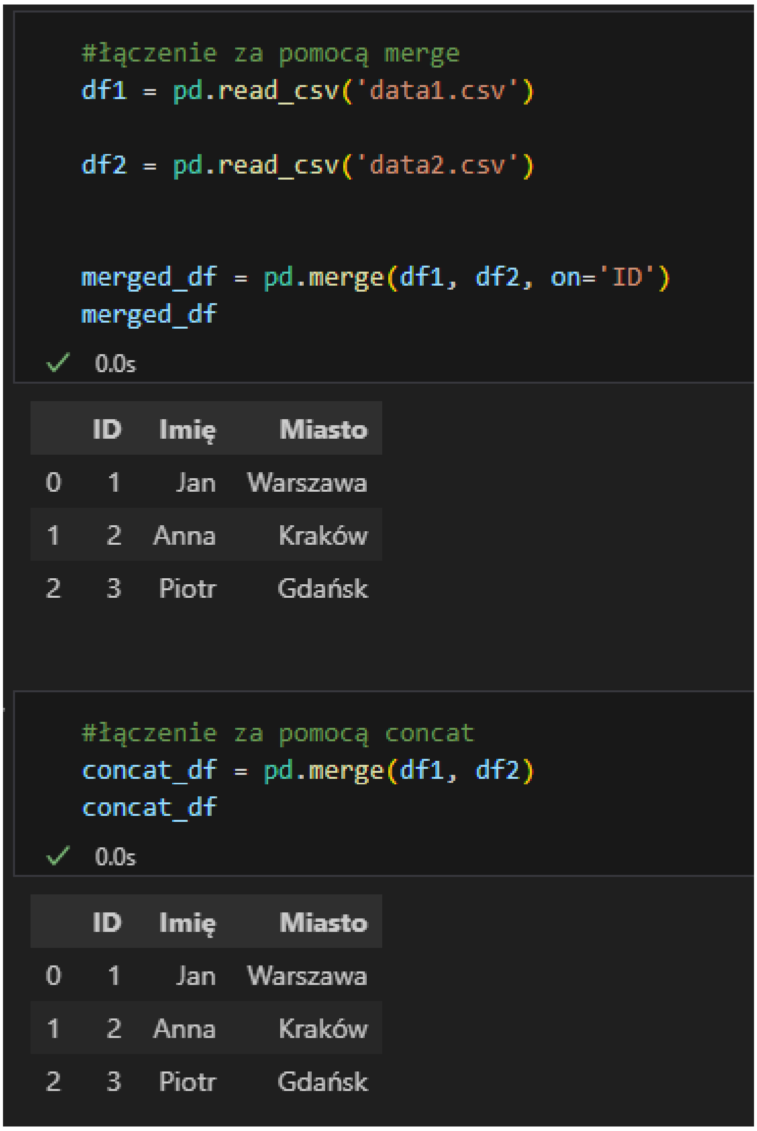


```
#histogram z tabeli przestawnej
df_bar = data_frame[(data_frame['country'].isin(['Albania', 'Algeria', 'Argentina'])) & (data_frame['gender'] == 'Male')].pivot_table(values='weight', index='age', columns='country',  aggfunc='sum',
                                         fill_value=None, margins=False, dropna=True)
df_bar.plot(kind = 'bar')
plt.ylabel('weight')
plt.title('weight by country')
```

✓ 0.1s

```
Text(0.5, 1.0, 'weight by country')
```

- przedstawić sposoby łączenia ramek danych za pomocą metod merge i concat



```
#łączenie za pomocą merge
df1 = pd.read_csv('data1.csv')

df2 = pd.read_csv('data2.csv')


merged_df = pd.merge(df1, df2, on='ID')
merged_df
```

✓ 0.0s

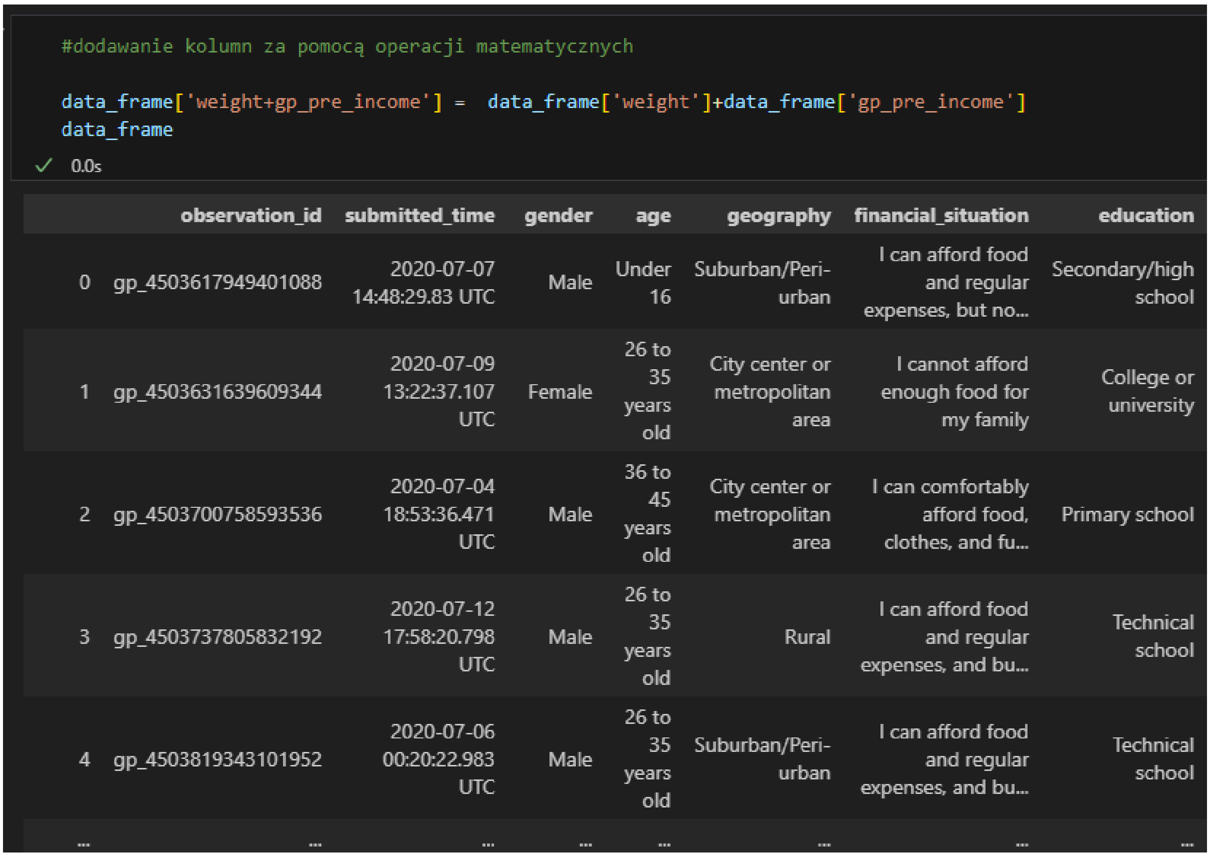
	ID	Imię	Miasto
0	1	Jan	Warszawa
1	2	Anna	Kraków
2	3	Piotr	Gdańsk

```
#łączenie za pomocą concat
concat_df = pd.merge(df1, df2)
concat_df
```

✓ 0.0s

	ID	Imię	Miasto
0	1	Jan	Warszawa
1	2	Anna	Kraków
2	3	Piotr	Gdańsk

- pokazać dodawanie nowych kolumn za pomocą operacji matematycznych

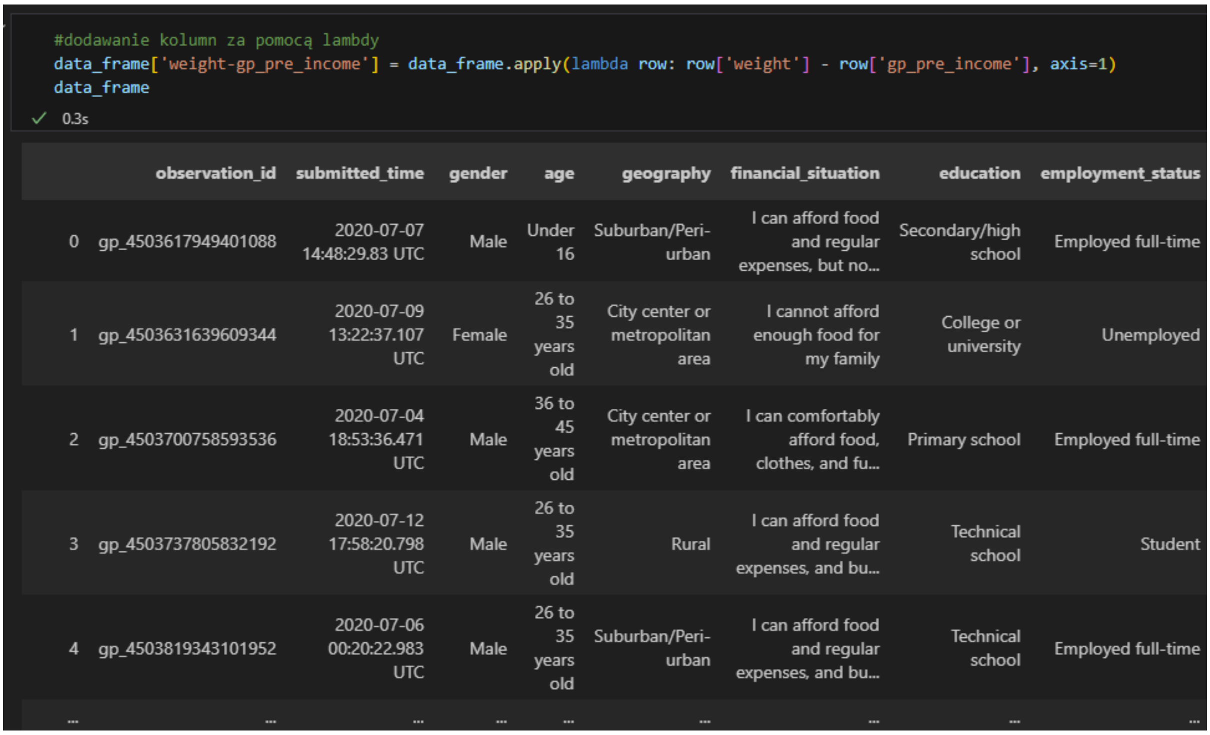


```
#dodawanie kolumn za pomocą operacji matematycznych

data_frame['weight+gp_pre_income'] =  data_frame['weight']+data_frame['gp_pre_income']
data_frame
```

✓ 0.0s

	observation_id	submitted_time	gender	age	geography	financial_situation	education
0	gp_4503617949401088	2020-07-07 14:48:29.83 UTC	Male	Under 16	Suburban/Peri-urban	I can afford food and regular expenses, but no...	Secondary/high school
1	gp_4503631639609344	2020-07-09 13:22:37.107 UTC	Female	26 to 35 years old	City center or metropolitan area	I cannot afford enough food for my family	College or university
2	gp_4503700758593536	2020-07-04 18:53:36.471 UTC	Male	36 to 45 years old	City center or metropolitan area	I can comfortably afford food, clothes, and fu...	Primary school
3	gp_4503737805832192	2020-07-12 17:58:20.798 UTC	Male	26 to 35 years old	Rural	I can afford food and regular expenses, and bu...	Technical school
4	gp_4503819343101952	2020-07-06 00:20:22.983 UTC	Male	26 to 35 years old	Suburban/Peri-urban	I can afford food and regular expenses, and bu...	Technical school
...	...	...	...	...	...	...	...

- przedstawić na przykładzie dodawanie nowych kolumn z pomocą funkcji lambda

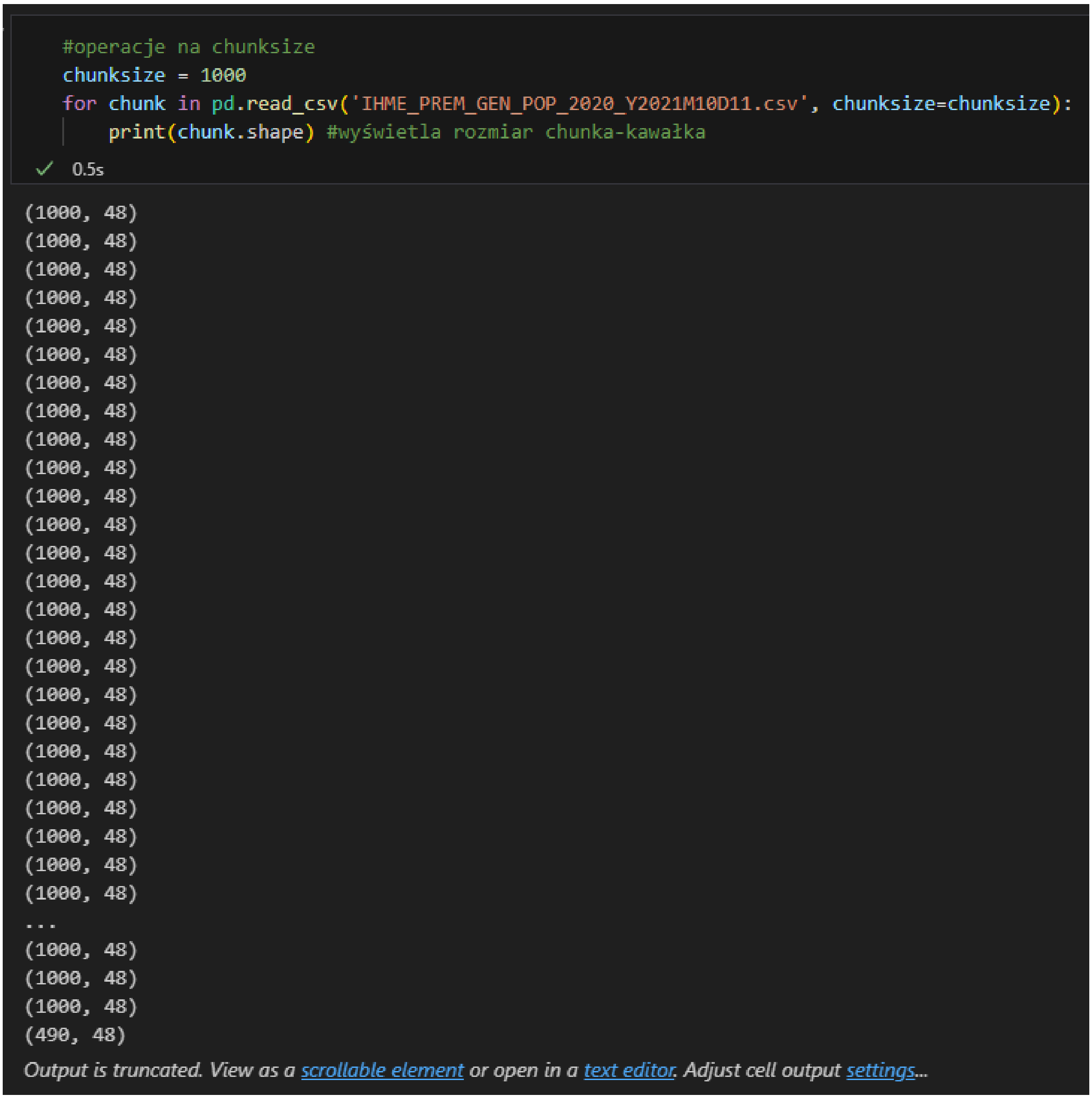


```
#dodawanie kolumn za pomocą lambdy
data_frame['weight-gp_pre_income'] = data_frame.apply(lambda row: row['weight'] - row['gp_pre_income'], axis=1)
data_frame
```

✓ 0.3s

	observation_id	submitted_time	gender	age	geography	financial_situation	education	employment_status
0	gp_4503617949401088	2020-07-07 14:48:29.83 UTC	Male	Under 16	Suburban/Peri-urban	I can afford food and regular expenses, but no...	Secondary/high school	Employed full-time
1	gp_4503631639609344	2020-07-09 13:22:37.107 UTC	Female	26 to 35 years old	City center or metropolitan area	I cannot afford enough food for my family	College or university	Unemployed
2	gp_4503700758593536	2020-07-04 18:53:36.471 UTC	Male	36 to 45 years old	City center or metropolitan area	I can comfortably afford food, clothes, and fu...	Primary school	Employed full-time
3	gp_4503737805832192	2020-07-12 17:58:20.798 UTC	Male	26 to 35 years old	Rural	I can afford food and regular expenses, and bu...	Technical school	Student
4	gp_4503819343101952	2020-07-06 00:20:22.983 UTC	Male	26 to 35 years old	Suburban/Peri-urban	I can afford food and regular expenses, and bu...	Technical school	Employed full-time
...	...	...	...	...	...	...	...	...

- przedstawić możliwości pracy z dużymi plikami przy użyciu argumentu chunksize



```
#operacje na chunksize
chunksize = 1000
for chunk in pd.read_csv('IHME_PREM_GEN_POP_2020_Y2021M10D11.csv', chunksize=chunksize):
    print(chunk.shape) #wyświetla rozmiar chunka-kawałka
```

✓ 0.5s

```
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
(1000, 48)
...
(1000, 48)
(1000, 48)
(1000, 48)
(490, 48)
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

3. Wnioski

Celem zadania było zdobycie podstawowej wiedzy na temat języka Python poprzez tworzenie i wyświetlanie ramki danych. Ważnym elementem było także opanowanie biblioteki pandas. Uzyskanie tych umiejętności umożliwiło realizację wszystkich poleceń, co pozwoliło na łatwe manipulowanie danymi oraz ich wyświetlanie w różnorodny sposób.