

## SPRAWOZDANIE

Zajęcia: Nauka o danych I

Prowadzący: prof. dr hab. Vasyl Martsenyuk

Laboratorium Nr 1 Data 28.09.2024 Temat: Wprowadzenie do narzędzi i środowiska pracy w analizie danych Wariant drugi (2)	Bartosz Bieniek Informatyka II stopień, stacjonarne, 1 semestr, gr.A
---	---

### 1. Polecenie: (wariant drugi zadania)

Gross Domestic Product Per Capita 1960-2050

<http://ghdx.healthdata.org/record/ihme-data/global-gdp-per-capita-1960-2050>

Lista zadań:

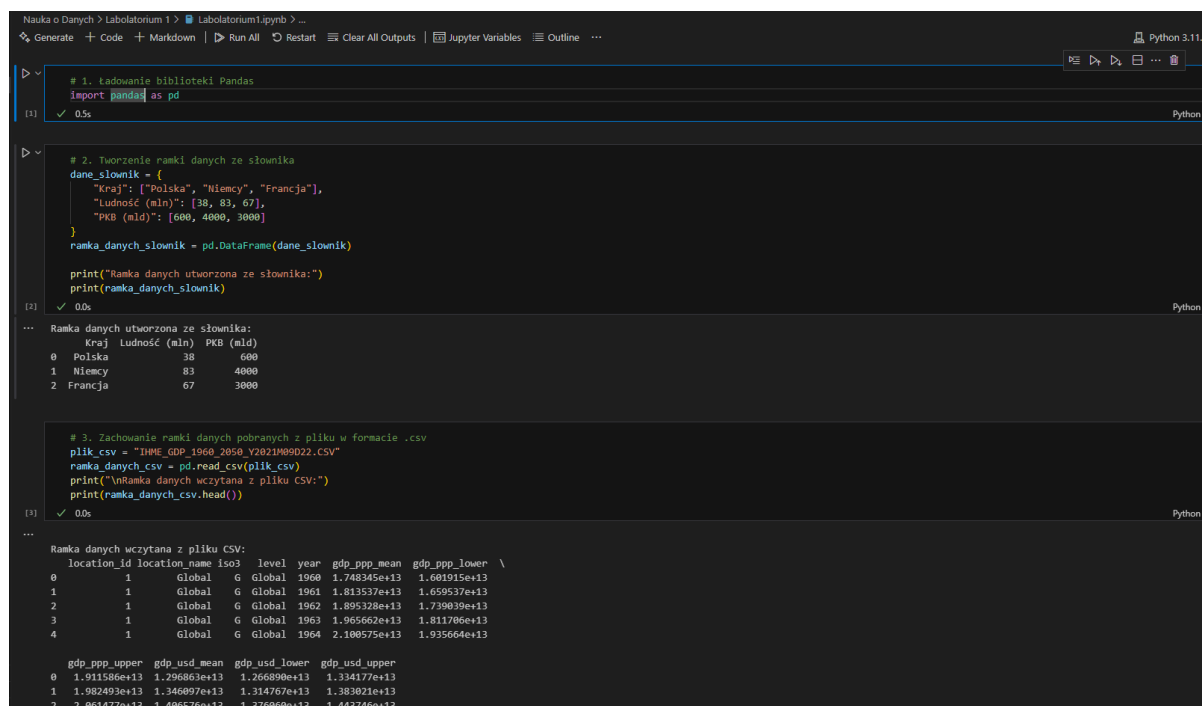
- ☐ Ładowanie biblioteki Pandas
- ☐ Tworzenie ramki danych ze słownika
- ☐ Zachowanie ramki danych pobranych z pliku w formacie .csv
- ☐ Tworzenie ramki danych z listy list
- ☐ Transponowanie (wymiana kolumny a wierszy)
- ☐ Wyświetlić pierwsze 10 wierszy ramki danych
- ☐ Wyświetlić ostatnie 10 wierszy ramki danych
- ☐ Wyświetlić informacje o ramce danych
- ☐ Wyświetlić ile wierszy i kolumn znajduje się w ramce danych
- ☐ Wyświetlić informacje statystyczną o kolumnach liczbowych (wartości niepowtarzalne, średnia, odchylenie standardowe, minimum, kwartyle, maksimum)
- ☐ Wyświetlić informacje statystyczną o kolumnach kategorizowanych (ile unikalnych wartości, top - jaka jest najpopularniejsza wartość, freq - jak często najpopularniejsza)
- ☐ Usunąć brakujące wartości w ramce danych
- ☐ Przedstawić wybór wierszy i kolumny używając nazw oraz indeksów na różne sposoby
- ☐ Przedstawić wybór wierszy z ramki danych pod warunkiem osnośnie określonej wartości kolumny
- ☐ Przedstawić wybór wierszy z ramki danych pod warunkiem spełnienia kilku warunków jednocześnie
- ☐ Wybrać wiersze które zawierają w kolumnie skategoryzowanej określone słowo

- ☐ Wybrać wiersze które nie zawierają w kolumnie skategoryzowanej określonego słowa
- ☐ Utwórz kolumnę na podstawie istniejącej
- ☐ Usuń kolumnę
- ☐ Zmień nazwę kolumny
- ☐ Zachowaj ramkę danych jako plik csv na komputerze
- ☐ Wyświetlić liczbę wierszy
- ☐ Wyświetlić wartości unikatowe w kolumnie
- ☐ Wyświetlić liczby rekordów odpowiadających do wartości
- ☐ Sortowanie wierszy ramki danych według wartości określonej kolumny
- ☐ (malejąco, rosnąco)
- ☐ Wyświetlić wierszy dla 10 największych (najmniejszych) wartości określonej kolumny
- ☐ Wyświetlić wierszy dla 10 największych wartości określonej kolumny
- ☐ pod warunkiem określonych wartości innej kolumny
- ☐ Grupowanie wierszy według wartości kolumny skategoryzowanej, potem uśrednienie wartości wszystkich kolumn w grupie - MultiIndex
- ☐ Grupowanie wierszy według wartości kolumny skategoryzowanej, potem uśrednienie wartości dla pewnych kolumn, liczba wartości i mediana dla pozostałych kolumn w grupach
- ☐ Wyświetlić nazwy kolumn indeksu złożonego
- ☐ Posortować kolumnę, indeksu złożonego
- ☐ Stworzyć tabelę przestawną (pivot table) na podstawie ramki danych
- ☐ Wyświetlić indeksy i kolumny tabeli przestawnej
- ☐ Utwórz indeks złożony tabeli przestawnej i wyświetl go
- ☐ Zaimportuj moduł pyplot z biblioteki matplotlib
- ☐ Wskazać, że wykresy należy rysować bezpośrednio w zeszycie, a nie w osobnej zakładce
- ☐ Wyświetlić wykres na podstawie tabeli przestawnej
- ☐ Narysować histogram na podstawie wartości kolumny
- ☐ Przedstawić sposoby łączenia ramek danych za pomocą metod merge i concat
- ☐ Pokazać dodawanie nowych kolumn za pomocą operacji matematycznych
- ☐ Przedstawić na przykładzie dodawanie nowych kolumn z pomocą funkcji lambda
- ☐ Przedstawić możliwości pracy z dużymi plikami przy użyciu argumentu chunksize

## 2. Opis programu opracowanego (kody Źródłowe, zrzuty ekranu)

Kod Źródłowy:

<https://github.com/mindgoner/Studia/tree/master/Nauka%20o%20Danych/Laboratorium%201>



```
# 1. Ładowanie biblioteki Pandas
import pandas as pd

# 2. Tworzenie ramki danych ze słownika
dane_sownik = {
    "Kraj": ["Polska", "Niemcy", "Francja"],
    "Ludność (mln)": [38, 83, 67],
    "PKB (mld)": [600, 4000, 3000]
}
ramka_danych_sownik = pd.DataFrame(dane_sownik)

print("Ramka danych utworzona ze słownika:")
print(ramka_danych_sownik)

# 3. Zachowanie ramki danych pobranych z pliku w formacie .csv
plik_csv = "TME_cdp_1960_2019_V202108022.csv"
ramka_danych_csv = pd.read_csv(plik_csv)
print("\nRamka danych wczytana z pliku CSV:")
print(ramka_danych_csv.head())
```

Ramka danych utworzona ze słownika:

	Kraj	Ludność (mln)	PKB (mld)
0	Polska	38	600
1	Niemcy	83	4000
2	Francja	67	3000

Ramka danych wczytana z pliku CSV:

	location_id	location_name	iso3	level	year	gdp_ppp_mean	gdp_ppp_lower	gdp_ppp_upper	gdp_usd_mean	gdp_usd_lower	gdp_usd_upper
0	1	Global	G	Global	1960	1.748345e+13	1.601915e+13	1.911506e+13	1.296863e+13	1.266090e+13	1.334177e+13
1	1	Global	G	Global	1961	1.813537e+13	1.659537e+13	1.982493e+13	1.346097e+13	1.314767e+13	1.383821e+13
2	1	Global	G	Global	1962	1.895328e+13	1.739039e+13	2.061477e+13	1.406576e+13	1.376060e+13	1.443746e+13
3	1	Global	G	Global	1963	1.965662e+13	1.811706e+13				
4	1	Global	G	Global	1964	2.100575e+13	1.935664e+13				

Rys. 1. Fragment kodu Źródłowego.

Pandas jest jedną z najpopularniejszych bibliotek w Pythonie, używaną do manipulacji i analizy danych w strukturze tabelarycznej, takich jak ramki danych. Tworzenie ramki danych za pomocą słownika umożliwia szybkie zorganizowanie danych w strukturze, gdzie klucze stanowią nazwy kolumn, a wartości to dane w tych kolumnach. Ładowanie danych z pliku CSV to powszechnie używana metoda do importowania danych zewnętrznych, która pozwala na łatwe przetwarzanie w Pythonie.

```
Nauka o Danych > Laboratorium 1 > Laboratorium1.py:nb > ...
Generate + Code + Markdown Run All Restart Clear All Outputs Jupyter Variables Outline Python 3.11.9

# 4. Tworzenie ramki danych z listy list
dane_lista = [
    ["Polska", 38, 600],
    ["Niemcy", 83, 4000],
    ["Francja", 67, 3000]
]
ramka_danych_lista = pd.DataFrame(dane_lista, columns=["Kraj", "Ludność (mln)", "PKB (mld)"])

print("\nRamka danych utworzona z listy list:")
print(ramka_danych_lista)

[4] ✓ 0.0s Python

...

Ramka danych utworzona z listy list:
   Kraj  Ludność (mln)  PKB (mld)
0  Polska           38        600
1  Niemcy           83       4000
2  Francja           67       3000

# 5. Transponowanie (wymiana kolumny a wierszy)
ramka_danych_sownik_transponowany = ramka_danych_sownik.T
print(ramka_danych_sownik_transponowany)

[5] ✓ 0.0s Python

...

   Kraj      Polska  Niemcy  Francja
Ludność (mln)    38     83     67
PKB (mld)       600   4000   3000

# 6. Wyświetlić pierwsze 10 wierszy ramki danych
print(ramka_danych_csv.head(10))

[6] ✓ 0.0s Python

...

location_id location_name iso3 level year gdp_ppp_mean gdp_ppp_lower \
0          1      Global    G   Global  1960  1.748365e+13  1.601915e+13
1          1      Global    G   Global  1961  1.813537e+13  1.659537e+13
2          1      Global    G   Global  1962  1.895328e+13  1.739039e+13
3          1      Global    G   Global  1963  1.965662e+13  1.811706e+13
4          1      Global    G   Global  1964  2.100575e+13  1.935664e+13
5          1      Global    G   Global  1965  2.202459e+13  2.034585e+13
6          1      Global    G   Global  1966  2.306193e+13  2.136085e+13
7          1      Global    G   Global  1967  2.391268e+13  2.217842e+13
8          1      Global    G   Global  1968  2.515733e+13  2.319730e+13
9          1      Global    G   Global  1969  2.640200e+13  2.444200e+13
```

Rys. 2. Fragment kodu Źródłowego.

Czwarty punkt umożliwia tworzenie ramki danych, gdzie każda podlista reprezentuje wiersz danych, co jest przydatne przy pracy z danymi o prostym układzie. Transponowanie ramki danych pozwala na zamianę wierszy z kolumnami, co może być użyteczne przy analizie danych w różnych układach. Wyświetlenie pierwszych kilku wierszy umożliwia szybkie zapoznanie się z danymi i ich strukturą, a także weryfikację poprawności importu.

```
Nauka o Danych > Laboratorium 1 > Laboratorium1.py:nb > ...
Generate + Code + Markdown Run All Restart Clear All Outputs Jupyter Variables Outline Python 3.11.9

# 7. Wyświetlić ostatnie 10 wierszy ramki danych
print(ramka_danych_csv.tail(10))

[7] ✓ 0.0s Python

...

19828 44578 Low income NaN World Bank Income Group 2041
19829 44578 Low income NaN World Bank Income Group 2042
19830 44578 Low income NaN World Bank Income Group 2043
19831 44578 Low income NaN World Bank Income Group 2044
19832 44578 Low income NaN World Bank Income Group 2045
19833 44578 Low income NaN World Bank Income Group 2046
19834 44578 Low income NaN World Bank Income Group 2047
19835 44578 Low income NaN World Bank Income Group 2048
19836 44578 Low income NaN World Bank Income Group 2049
19837 44578 Low income NaN World Bank Income Group 2050

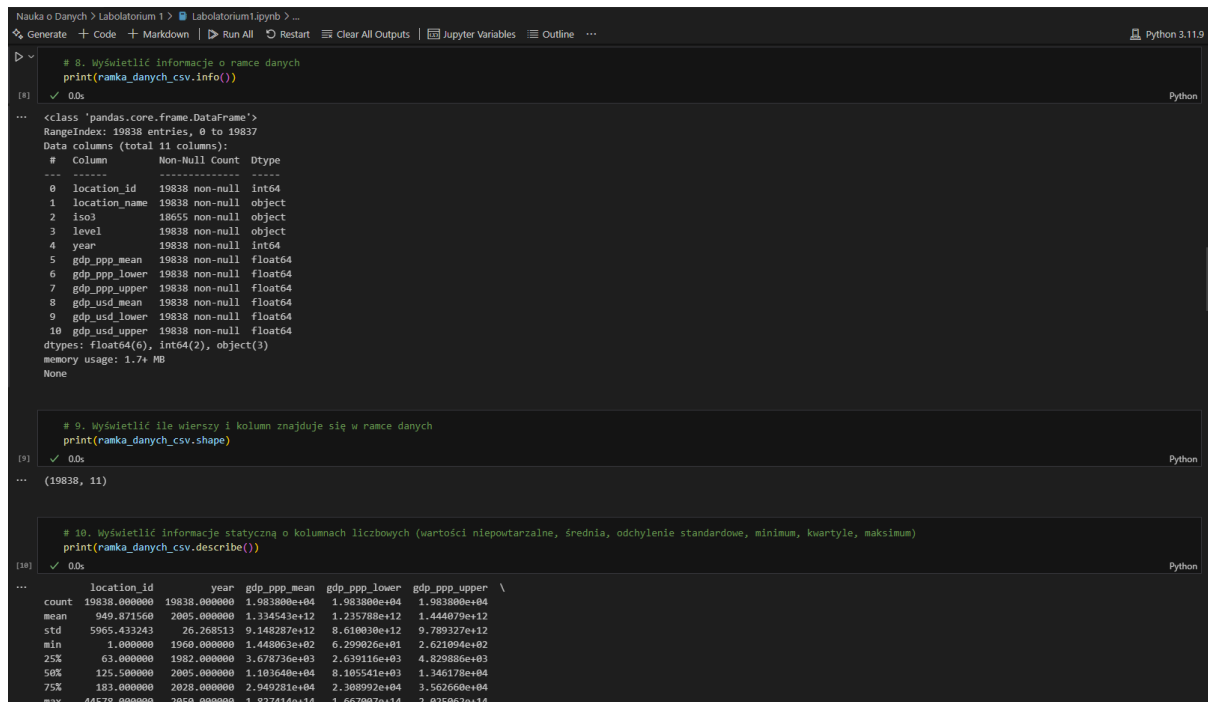
gdp_ppp_mean gdp_ppp_lower gdp_ppp_upper gdp_usd_mean \
19828 3.120963e+12 2.724077e+12 3.582807e+12 9.752426e+11
19829 3.216988e+12 2.801335e+12 3.686394e+12 1.008813e+12
19830 3.314031e+12 2.886768e+12 3.815672e+12 1.042881e+12
19831 3.413020e+12 2.968361e+12 3.933135e+12 1.077714e+12
19832 3.514244e+12 3.055623e+12 4.049325e+12 1.113207e+12
19833 3.617318e+12 3.148035e+12 4.166469e+12 1.149318e+12
19834 3.724063e+12 3.225840e+12 4.292403e+12 1.186597e+12
19835 3.831942e+12 3.307609e+12 4.424674e+12 1.224062e+12
19836 3.941856e+12 3.398884e+12 4.560961e+12 1.262129e+12
19837 4.053883e+12 3.482933e+12 4.713596e+12 1.300764e+12

gdp_usd_lower gdp_usd_upper
...
19834 1.061313e+12 1.318836e+12
19835 1.092874e+12 1.365610e+12
19836 1.122095e+12 1.413091e+12
19837 1.151548e+12 1.457362e+12

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

Rys. 3. Fragment kodu Źródłowego.

Sprawdzanie ostatnich kilku wierszy danych jest równie ważne, ponieważ pozwala zobaczyć, czy dane zostały poprawnie załadowane i czy nie zawierają błędów.



```
# 8. Wyświetlić informacje o ramce danych
print(ramka_danych_csv.info())

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19838 entries, 0 to 19837
Data columns (total 11 columns):
 #   Column              Non-Null Count  Dtype  
---  --
 0   location_id         19838 non-null  int64  
 1   location_name       19838 non-null  object  
 2   iso3                18655 non-null  object  
 3   level              19838 non-null  object  
 4   year               19838 non-null  int64  
 5   gdp_ppp_mean       19838 non-null  float64 
 6   gdp_ppp_lower      19838 non-null  float64 
 7   gdp_ppp_upper      19838 non-null  float64 
 8   gdp_usd_mean       19838 non-null  float64 
 9   gdp_usd_lower      19838 non-null  float64 
10   gdp_usd_upper      19838 non-null  float64 
dtypes: float64(6), int64(2), object(3)
memory usage: 1.7+ MB
None

# 9. Wyświetlić ile wierszy i kolumn znajduje się w ramce danych
print(ramka_danych_csv.shape)

(19838, 11)

# 10. Wyświetlić informacje statystyczną o kolumnach liczbowych (wartości niepowtarzalne, średnia, odchylenie standardowe, minimum, kwartyle, maksimum)
print(ramka_danych_csv.describe())

count    location_id         year    gdp_ppp_mean    gdp_ppp_lower    gdp_ppp_upper  \
mean      949.871560    2005.000000    1.334543e+12    1.235788e+12    1.444079e+12
std      5965.433243      26.268513    9.148287e+12    8.610030e+12    9.789327e+12
min         1.000000    1960.000000    1.448063e+02    6.299026e+01    2.621094e+02
25%         63.000000    1982.000000    3.678736e+03    2.639116e+03    4.829886e+03
50%        125.500000    2005.000000    1.183640e+04    8.185541e+03    1.346178e+04
75%        183.000000    2028.000000    2.949281e+04    2.388992e+04    3.562668e+04
max       44578.000000    2050.000000    1.827414e+14    1.667007e+14    2.825062e+14
```

Rys. 4. Fragment kodu Źródłowego.

Metoda `info()` pozwala uzyskać przegląd ramki danych, w tym typy danych w kolumnach, liczbę niepustych wartości i inne istotne informacje. Zrozumienie rozmiaru ramki danych jest kluczowe przy pracy z dużymi zbiorami danych, pozwala na oszacowanie wymaganych zasobów. Metoda `describe()` generuje szybki przegląd statystyki liczbowe w ramce danych, takie jak średnia, mediana czy kwartyle, co jest pomocne w analizie rozkładu danych.

```
Nauka o Danych > Laboratorium 1 > Laboratorium1.py:nb > ...
Generate + Code + Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline ... Python 3.11.9

# 11. Wyświetlić informacje statystyczną o kolumnach kategorizowanych (ile unikalnych wartości, top - jaka jest najpopularniejsza wartość, freq - jak często najpopularniejsza)
print(ramka_danych_csv.describe(include=['object']))

[11] ✓ 0.0s Python

...
count      19838      18655      19838
unique         216         285         4
top      South Asia         6  Country
freq         182         91      18564

# 12. Usunąć brakujące wartości w ramce danych
ramka_danych_csv = ramka_danych_csv.dropna()

[12] ✓ 0.0s Python

# 13. Przedstawić wybór wierszy i kolumny używając nazw oraz indeksów na różne sposoby
print(ramka_danych_csv.loc[0:5, ["location_name", "gdp_ppp_mean"]]) # Nazwy kolumn
print(ramka_danych_csv.iloc[0:5, 1:3]) # Indeksy kolumn

[13] ✓ 0.0s Python

...
location_name  gdp_ppp_mean
0      Global  1.748345e+13
1      Global  1.813537e+13
2      Global  1.895328e+13
3      Global  1.965662e+13
4      Global  2.100575e+13
5      Global  2.202459e+13
location_name  iso3
0      Global    G
1      Global    G
2      Global    G
3      Global    G
4      Global    G

# 14. Przedstawić wybór wierszy z ramki danych pod warunkiem określonej wartości kolumny
print(ramka_danych_csv[ramka_danych_csv['location_name'] == 'Poland'])

[14] ✓ 0.0s Python

...
location_id location_name iso3  level  year  gdp_ppp_mean \
3913         51      Poland  POL  Country  1960  64771.852541
3914         51      Poland  POL  Country  1961  6854.160388
3915         51      Poland  POL  Country  1962  6696.268564
```

*Rys. 5. Fragment kodu Źródłowego.*

Funkcja `describe()` jest również użyteczna do analizy danych kategorizowanych, pozwala na uzyskanie informacji o unikalnych wartościach oraz najczęściej występującej wartości. Usuwanie brakujących danych jest jednym z podstawowych etapów czyszczenia danych, aby uniknąć błędów podczas analizy lub pozbyć się niepotrzebnych kolumn. Wybór danych za pomocą indeksów i nazw kolumn pozwala na elastyczne filtrowanie danych w zależności od potrzeb, co również przekłada się na ograniczenie potencjalnych błędów przy analizie. Filtrowanie wierszy na podstawie wartości w kolumnach jest jednym z podstawowych sposobów analizy danych, pomagając w izolacji interesujących nas rekordów.

```
Nauka o Danych > Laboratorium 1 > Laboratorium1.py:nb > ...
Generate + Code + Markdown Run All Restart Clear All Outputs Jupyter Variables Outline Python 3.11.9

# 15. Przedstawić wybór wierszy z ramki danych pod warunkiem spełnienia kilku warunków jednocześnie
print(ramka_danych_csv[ramka_danych_csv['location_name'] == 'Poland'] & (ramka_danych_csv['year'] > 2024)])

[15] ✓ 0.0s Python

...
location_id location_name iso3 level year gdp_ppp_mean \
3978 51 Poland POL Country 2025 32746.999437
3979 51 Poland POL Country 2026 32914.035363
3980 51 Poland POL Country 2027 33150.861860
3981 51 Poland POL Country 2028 33466.534617
3982 51 Poland POL Country 2029 33845.943995
3983 51 Poland POL Country 2030 34262.156546
3984 51 Poland POL Country 2031 34684.507200
3985 51 Poland POL Country 2032 35094.834819
3986 51 Poland POL Country 2033 35489.219953
3987 51 Poland POL Country 2034 35879.588365
3988 51 Poland POL Country 2035 36263.317214
3989 51 Poland POL Country 2036 36623.408529
3990 51 Poland POL Country 2037 36958.569309
3991 51 Poland POL Country 2038 37248.918811
3992 51 Poland POL Country 2039 37497.039670
3993 51 Poland POL Country 2040 37703.694057
3994 51 Poland POL Country 2041 37879.536633
3995 51 Poland POL Country 2042 38038.938282
3996 51 Poland POL Country 2043 38167.733724
3997 51 Poland POL Country 2044 38255.709502
3998 51 Poland POL Country 2045 38310.991457
3999 51 Poland POL Country 2046 38341.602497
4000 51 Poland POL Country 2047 38361.938630
4001 51 Poland POL Country 2048 38342.127372
...
4000 27796.090370 52186.079877 17401.902620 12665.251631 23581.749549
4001 27466.932992 52654.694631 17393.027956 12498.169215 23783.922827
4002 27105.252343 53056.023696 17369.059621 12305.183049 24051.240960
4003 26728.543779 53156.930801 17346.349035 12139.994778 24206.609198
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

# 16. Wybrać wiersze które zawierają w kolumnie skategoryzowanej określone słowo
print(ramka_danych_csv[ramka_danych_csv['location_name'].str.contains('United', na=False)])

[16] ✓ 0.0s Python

...
location_id location_name iso3 level year \
7553 95 United Kingdom GBR Country 1960
7554 95 United Kingdom GBR Country 1961
7555 95 United Kingdom GBR Country 1962
```

Rys. 6. Fragment kodu Źródłowego.

Używanie operatorów logicznych do łączenia warunków pozwala na bardziej precyzyjne filtrowanie danych. Filtrowanie danych w zależności od wartości tekstowych w kolumnach jest przydatne w analizie danych jakościowych.

```
Nauka o Danych > Laboratorium 1 > Laboratorium1.py:nb > ...
Generate + Code + Markdown Run All Restart Clear All Outputs Jupyter Variables Outline Python 3.11.9

# 17. Wybrać wiersze które nie zawierają w kolumnie skategoryzowanej określonego słowa
print(ramka_danych_csv[~ramka_danych_csv['location_name'].str.contains('United', na=False)])

[17] ✓ 0.0s Python

...
location_id location_name iso3 level year gdp_ppp_mean \
0 1 Global G Global 1960 1.748345e+13
1 1 Global G Global 1961 1.813537e+13
2 1 Global G Global 1962 1.895328e+13
3 1 Global G Global 1963 1.965662e+13
4 1 Global G Global 1964 2.108575e+13
...
...
19469 522 Sudan SDN Country 2046 6.656899e+03
19470 522 Sudan SDN Country 2047 6.729027e+03
19471 522 Sudan SDN Country 2048 6.796123e+03
19472 522 Sudan SDN Country 2049 6.866343e+03
19473 522 Sudan SDN Country 2050 6.935555e+03
...
gdp_ppp_lower gdp_ppp_upper gdp_usd_mean gdp_usd_lower \
0 1.601915e+13 1.911586e+13 1.296862e+13 1.266000e+13
1 1.659537e+13 1.982493e+13 1.346097e+13 1.314767e+13
2 1.739039e+13 2.061477e+13 1.406576e+13 1.376060e+13
3 1.811706e+13 2.134993e+13 1.461831e+13 1.432132e+13
4 1.935664e+13 2.276791e+13 1.552986e+13 1.523498e+13
...
...
19469 3.356042e+03 1.155051e+04 1.459547e+03 9.801683e+02
19470 3.374504e+03 1.171206e+04 1.475378e+03 9.886902e+02
19471 3.390609e+03 1.184106e+04 1.490021e+03 9.935208e+02
19472 3.417444e+03 1.196204e+04 1.505368e+03 1.002889e+03
19473 3.429198e+03 1.208179e+04 1.520564e+03 1.002953e+03
...
19472 2.362591e+03
19473 2.408108e+03

[18200 rows x 11 columns]
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

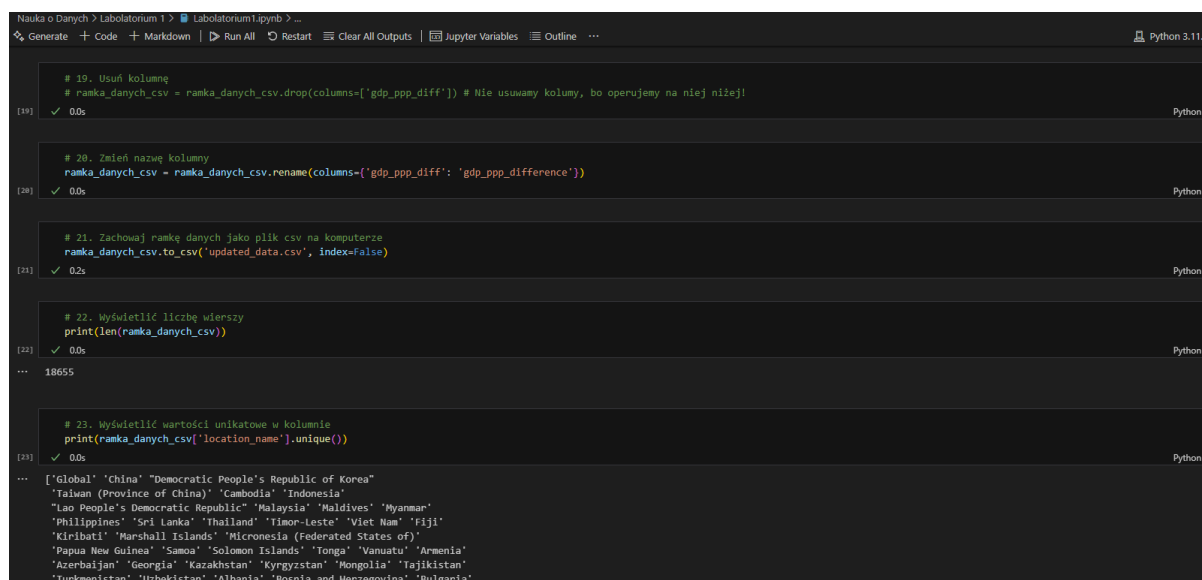
# 18. Utwórz kolumnę na podstawie istniejącej
ramka_danych_csv['gdp_ppp_diff'] = ramka_danych_csv['gdp_ppp_upper'] - ramka_danych_csv['gdp_ppp_lower'] # Utworzenie kolumny z różnicą między gdp_ppp_upper i gdp_ppp_lower
print(ramka_danych_csv['gdp_ppp_diff'].head(10))

[18] ✓ 0.0s Python

...
0 3.096716e+12
1 3.229556e+12
2 3.224381e+12
3 3.220877e+12
```

Rys. 7. Fragment kodu Źródłowego.

Analogiczne do poprzedniego zadania, ale tym razem umożliwiające eliminowanie danych zawierających określone słowo przy użyciu znaku tyldy jako negacji. Dodawanie nowych kolumn na podstawie danych z innych kolumn pozwala na tworzenie dodatkowych zmiennych, które mogą być istotne w analizie.



```
# 19. Usuń kolumnę
# ramka_danych_csv = ramka_danych_csv.drop(columns=['gdp_ppp_diff']) # Nie usuwamy kolumny, bo operujemy na niej niżej

# 20. Zmień nazwę kolumny
ramka_danych_csv = ramka_danych_csv.rename(columns={'gdp_ppp_diff': 'gdp_ppp_difference'})

# 21. Zachowaj ramkę danych jako plik csv na komputerze
ramka_danych_csv.to_csv('updated_data.csv', index=False)

# 22. Wyświetl liczbę wierszy
print(len(ramka_danych_csv))

# 23. Wyświetl wartości unikatowe w kolumnie
print(ramka_danych_csv['location_name'].unique())
```

Output for cell 22: 18655

Output for cell 23:

```
['Global' 'China' 'Democratic People's Republic of Korea'
 'Taiwan (Province of China)' 'Cambodia' 'Indonesia'
 'Lao People's Democratic Republic' 'Malaysia' 'Maldives' 'Myanmar'
 'Philippines' 'Sri Lanka' 'Thailand' 'Timor-Leste' 'Viet Nam' 'Fiji'
 'Kiribati' 'Marshall Islands' 'Micronesia (Federated States of)'
 'Papua New Guinea' 'Samoa' 'Solomon Islands' 'Tonga' 'Vanuatu' 'Armenia'
 'Azerbaijan' 'Georgia' 'Kazakhstan' 'Kyrgyzstan' 'Mongolia' 'Tajikistan'
 'Turkmenistan' 'Uzbekistan' 'Albania' 'Bosnia and Herzegovina' 'Bulgaria']
```

Rys. 8. Fragment kodu Źródłowego.

Usuwanie zbędnych kolumn (jak wspomniano wcześniej) jest częścią procesu czyszczenia danych, eliminując dane, które nie są potrzebne do analizy. Zmiana nazw kolumn poprawia czytelność danych i dostosowuje je do konwencji lub wymagań analitycznych. Zapisanie danych w pliku CSV jest jedną z metod przechowywania wyników analiz, co pozwala na ich późniejsze wykorzystanie, wstawienie punktu kontrolnego przy analizie dużych danych lub zapisanie danych jako wyjściowych. Szybkie sprawdzenie liczby wierszy w ramce danych pomaga w ocenie rozmiaru danych i ich przydatności do analizy. Głównym powodem tej operacji jest dobranie odpowiedniego sprzętu w przypadku dużych danych. Sprawdzanie unikalnych wartości w kolumnie jest pomocne w identyfikacji rozkładu i różnorodności danych w tej kolumnie. Na kolejnym przykładzie przedstawiono jak dodać ilość występowania w zależności od unikalnych wartości.



```
Nauka o Danych > Laboratorium 1 > Laboratorium1.py:nb > ...
Generate + Code + Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline ... Python 3.11.9

# 24. Wyświetlić liczbę rekordów odpowiadających do wartości
print(ramka_danych_csv['location_name'].value_counts())

[24] ✓ 0.0s Python

... location_name
Global 91
China 91
Democratic People's Republic of Korea 91
Taiwan (Province of China) 91
Cambodia 91
..
Tokelau 91
Tuvalu 91
United States Virgin Islands 91
South Sudan 91
Sudan 91
Name: count, Length: 205, dtype: int64

# 25. Sortowanie wierszy ramki danych według wartości określonej kolumny (malejąco, rosnąco)
ramka_danych_csv = ramka_danych_csv.sort_values(by='location_name', ascending=False)

[25] ✓ 0.0s Python

# 26. Wyświetlić wiersze dla 10 największych (najmniejszych) wartości określonej kolumny
print(ramka_danych_csv.nlargest(10, 'gdp_ppp_difference'))
print(ramka_danych_csv.nsmallest(10, 'gdp_ppp_difference'))

[26] ✓ 0.0s Python

... location_id location_name iso3 level year gdp_ppp_mean gdp_ppp_lower \
90 1 Global G Global 2050 1.827414e+14 1.667007e+14
89 1 Global G Global 2049 1.811701e+14 1.657675e+14
88 1 Global G Global 2048 1.795422e+14 1.647031e+14
87 1 Global G Global 2047 1.778053e+14 1.635681e+14
86 1 Global G Global 2046 1.759560e+14 1.622744e+14
85 1 Global G Global 2045 1.740498e+14 1.608327e+14
84 1 Global G Global 2044 1.720934e+14 1.594056e+14
83 1 Global G Global 2043 1.701152e+14 1.579438e+14
82 1 Global G Global 2042 1.681175e+14 1.566207e+14
81 1 Global G Global 2041 1.661209e+14 1.552230e+14

gdp_ppp_upper gdp_usd_mean gdp_usd_lower gdp_usd_upper \
90 2.025062e+14 1.110460e+14 1.017185e+14 1.230700e+14
```

Rys. 9. Fragment kodu Źródłowego.

Wyświetlenie liczby rekordów odpowiadających do wartości pozwala na zrozumienie, jak często dana wartość występuje w kolumnie, co może być pomocne przy analizie częstości. Sortowanie danych według określonej kolumny jest kluczowe przy analizie danych, zwłaszcza gdy chcemy znaleźć największe lub najmniejsze wartości. Zadanie 26. umożliwia wybór rekordów z najwyższymi lub najniższymi wartościami w kolumnie, co jest przydatne do analizy danych skrajnych.

```
Nauka o Danych > Laboratorium 1 > Laboratorium1.py:nb > ...
Generate + Code + Markdown | Run All | Restart | Clear All Outputs | Jupyter Variables | Outline ... Python 3.11.9

# 27. Wyświetlić wiersze dla 10 największych wartości określonej kolumny pod warunkiem określonych wartości innej kolumny
filtered_data = ramka_danych_csv[ramka_danych_csv['location_name'] == 'Global']
print(filtered_data.nlargest(10, 'gdp_ppp_mean'))

[27] ✓ 0.0s Python

... location_id location_name iso3 level year gdp_ppp_mean gdp_ppp_lower \
90 1 Global G Global 2050 1.827414e+14 1.667007e+14
89 1 Global G Global 2049 1.811701e+14 1.657675e+14
88 1 Global G Global 2048 1.795422e+14 1.647031e+14
87 1 Global G Global 2047 1.778053e+14 1.635681e+14
86 1 Global G Global 2046 1.759560e+14 1.622744e+14
85 1 Global G Global 2045 1.740498e+14 1.608327e+14
84 1 Global G Global 2044 1.720934e+14 1.594056e+14
83 1 Global G Global 2043 1.701152e+14 1.579438e+14
82 1 Global G Global 2042 1.681175e+14 1.566207e+14
81 1 Global G Global 2041 1.661209e+14 1.552230e+14

gdp_ppp_upper gdp_usd_mean gdp_usd_lower gdp_usd_upper \
90 2.025062e+14 1.110460e+14 1.017185e+14 1.230700e+14
89 2.003282e+14 1.110748e+14 1.012670e+14 1.226294e+14
88 1.978349e+14 1.101656e+14 1.008704e+14 1.212579e+14
87 1.952850e+14 1.091923e+14 1.003097e+14 1.192614e+14
```

Rys. 10. Fragment kodu Źródłowego.

Filtrowanie na podstawie warunków w różnych kolumnach pozwala na dokładniejszą selekcję interesujących nas danych, jak przedstawiono w zadaniu 27.

```

# 28. Grupowanie wierszy według wartości kolumny skategoryzowanej, potem uśrednienie wartości wszystkich kolumn w grupie - MultiIndex
grouped_mean = ramka_danych_csv.groupby(['location_name', 'year']).mean(numeric_only=True)
print(grouped_mean)

```

location_name	year	location_id	gdp_ppp_mean	gdp_ppp_lower	gdp_ppp_upper
Afghanistan	1960	160.0	2221.335586	1353.292858	3882.415995
	1961	160.0	2192.653614	1336.349082	3812.241182
	1962	160.0	2178.869688	1327.326777	2963.818432
	1963	160.0	2169.572283	1322.083248	2940.315793
Zimbabwe	2046	198.0	3086.223490	1856.652268	4531.850929
	2047	198.0	3116.294363	1866.152043	4625.577546
	2048	198.0	3145.941464	1857.642354	4693.442459
	2049	198.0	3175.312716	1861.027391	4766.287444
Zimbabwe	2050	198.0	3284.717710	1852.077740	4849.920213

Rys. 11. Fragment kodu źródłowego.

Grupowanie danych pozwala na agregację i analizowanie danych w kontekście różnych kategorii, co może ujawnić ewentualne wzorce w danych.

```

# 29. Grupowanie wierszy według wartości kolumny skategoryzowanej, potem uśrednienie wartości dla pewnych kolumn, liczba wartości i mediana dla pozostałych kolumn w grupach
grouped_stats = ramka_danych_csv.groupby('location_name').agg({
    'gdp_ppp_mean': ['mean', 'count'],
    'gdp_usd_mean': ['median', 'count'],
    'year': 'max'
})
print(grouped_stats)

```

location_name	gdp_ppp_mean	mean	count	gdp_usd_mean	year
Afghanistan	1941.160286	91	515.274036	2050	
Albania	9092.515182	91	3898.516205	2050	
Algeria	8820.271149	91	3163.885729	2050	
American Samoa	15340.365197	91	13620.772462	2050	
Andorra	25139.562251	91	38178.372791	2050	
Venezuela (Bolivarian Republic of)	18584.142490	91	5823.785745	2050	
Viet Nam	5737.873614	91	1437.919432	2050	
Yemen	2637.237249	91	828.086983	2050	
Zambia	3107.029470	91	1878.009951	2050	
Zimbabwe	2925.918096	91	1869.856772	2050	

```

# 30. Wyświetlić nazwy kolumn indeksu złożonego
print(grouped_mean.index.names)

```

```

# 31. Posortować kolumnę indeksu złożonego
sorted_index = grouped_mean.sort_index()
print(sorted_index)

```

location_name	year	location_id	gdp_ppp_mean	gdp_ppp_lower	gdp_ppp_upper
Afghanistan	1960	160.0	2221.335586	1353.292858	3882.415995
	1961	160.0	2192.653614	1336.349082	3812.241182
	1962	160.0	2178.869688	1327.326777	2963.818432
	1963	160.0	2169.572283	1322.083248	2940.315793

Rys. 12. Fragment kodu źródłowego.

Agregacja danych z wykorzystaniem różnych funkcji agregujących (średnia, liczba, mediana) pozwala na szczegółową analizę danych pogrupowanych według różnych cech. Zadanie 30 umożliwia identyfikację poziomów w złożonym indeksie, co jest istotne przy pracy z wieloma poziomami indeksów. Sortowanie indeksów jest przydatne w przypadku pracy z dużymi zbiorami danych, gdzie sortowanie umożliwia szybsze przetwarzanie i analizę.

```

Nauka o Danych > Laboratorium 1 > Laboratorium1.ipynb > ...
Generate + Code + Markdown Run All Restart Clear All Outputs Jupyter Variables Outline Python 3.11.5

# 32. Stwórzyc tabelę przestawną (pivot table) na podstawie ramki danych
pivot_table = pd.pivot_table(ramka_danych_csv, values='gdp_ppp_mean', index=['location_name'], columns=['year'], aggfunc='mean')
print(pivot_table)

[32] ✓ 0.0s Python

... year
location_name
Afghanistan 2221.335586 2192.653614 2178.869688
Albania 3158.241995 3179.711325 3272.046583
Algeria 5006.438360 4453.400566 3658.175321
American Samoa 22109.859273 21946.482938 21884.955783
Andorra 15636.337352 15796.419665 16031.522138
...
Venezuela (Bolivarian Republic of) 10432.796342 10430.234336 10804.656564
Viet Nam 1301.479002 1319.501206 1427.159417
Yemen 1207.599807 1213.637071 1220.327712
Zambia 2685.864457 2613.932835 2518.024702
Zimbabwe 2488.747887 2460.565228 2405.779753

year
location_name
Afghanistan 2169.572283 2155.861769 2151.064936
Albania 3370.096538 3471.383451 3583.125647
Algeria 4362.291504 4467.247602 4635.281973
American Samoa 21746.150959 22039.103093 21967.292788
Andorra 16212.717854 16721.236248 16962.508876
...
Venezuela (Bolivarian Republic of) 11116.532476 11810.328016 11935.494850
Viet Nam 1421.953513 1440.696855 1413.406360
Yemen 1230.605680 1236.624591 1245.455329
...
Zambia 4989.723480 5050.614585 5113.775149
Zimbabwe 3145.941464 3175.312716 3204.717710

[205 rows x 91 columns]
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings.

```

Rys. 13. Fragment kodu Źródłowego.

Tabele przestawne umożliwiają agregowanie i podsumowywanie danych w łatwy do zrozumienia sposób, zmieniając układ danych.

```

Nauka o Danych > Laboratorium 1 > Laboratorium1.ipynb > ...
Generate + Code + Markdown Run All Restart Clear All Outputs Jupyter Variables Outline Python 3.11.9

# 33. Wyświetlić indeks i kolumny tabeli przestawnej
print(pivot_table.index)
print(pivot_table.columns)

[33] ✓ 0.0s Python

... Index(['Afghanistan', 'Albania', 'Algeria', 'American Samoa', 'Andorra',
        'Angola', 'Antigua and Barbuda', 'Argentina', 'Armenia', 'Australia',
        ...,
        'United States Virgin Islands', 'United States of America', 'Uruguay',
        'Uzbekistan', 'Vanuatu', 'Venezuela (Bolivarian Republic of)',
        'Viet Nam', 'Yemen', 'Zambia', 'Zimbabwe'],
        dtype='object', name='location_name', length=205)
Index([1960, 1961, 1962, 1963, 1964, 1965, 1966, 1967, 1968, 1969, 1970, 1971,
        1972, 1973, 1974, 1975, 1976, 1977, 1978, 1979, 1980, 1981, 1982, 1983,
        1984, 1985, 1986, 1987, 1988, 1989, 1990, 1991, 1992, 1993, 1994, 1995,
        1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007,
        2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019,
        2020, 2021, 2022, 2023, 2024, 2025, 2026, 2027, 2028, 2029, 2030, 2031,
        2032, 2033, 2034, 2035, 2036, 2037, 2038, 2039, 2040, 2041, 2042, 2043,
        2044, 2045, 2046, 2047, 2048, 2049, 2050],
        dtype='int64', name='year')

# 34. Utwórz indeks złożony tabeli przestawnej i wyświetl go
pivot_table = pivot_table.reset_index()
pivot_table.set_index(['location_name'] + list(pivot_table.columns[1:]), inplace=True)
print(pivot_table.index)

[34] ✓ 0.0s Python

... MultiIndex([(
        ('Afghanistan', 2221.33558642915, ...),
        ('Albania', 3158.24199462837, ...),
        ('Algeria', 5006.43835997237, ...),
        ('American Samoa', 22109.859272757, ...),
        ('Andorra', 15636.33735217181, ...),
        ('Angola', 4424.22378093641, ...),
        ('Antigua and Barbuda', 3939.06456079785, ...),
        ('Argentina', 9947.85853860087, ...),
        ('Armenia', 4717.49893852611, ...),
        ('Australia', 17521.8499818484, ...),
        ...,
        ('United States Virgin Islands', 11482.6973448973, ...),
        ('United States of America', 21498.7249723808, ...),
        ('Uruguay', 8032.293102246791, ...),
        ('Uzbekistan', 2971.75653004325, ...),
        ('Vanuatu', 1761.93203871854, ...),
        ('Venezuela (Bolivarian Republic of)', 10432.796342326, ...),

```

Rys. 14. Fragment kodu Źródłowego.

Analizowanie struktury tabeli przestawnej pomaga w zrozumieniu, jak dane są zorganizowane i które informacje są dostępne. Tworzenie indeksu złożonego umożliwia jeszcze dokładniejsze przetwarzanie danych, szczególnie w przypadku dużych i złożonych zbiorów.



Rys. 15. Fragment kodu Źródłowego i wykres.

Moduł pyplot z Matplotlib pozwala na tworzenie wykresów i wizualizacji, które są ważnym narzędziem analitycznym. Rysowanie wykresów w bezpośredniej interakcji z kodem umożliwia ich natychmiastowe wykorzystanie i analizę wyników. Wizualizacja danych w postaci wykresów pozwala na łatwiejsze zrozumienie wyników analizy oraz identyfikację wzorców.



Rys. 16. Fragment kodu Źródłowego i wykres.

Histogramy są pomocne do analizy rozkładu danych numerycznych i wykrywania ewentualnych anomalii.

```
Naucz o Danych > Laboratorium 1 > Laboratorium1.ipynb > # 38. Narysować histogram na podstawie wartości kolumny
Generate + Code + Markdown Run All Restart Clear All Outputs Jupyter Variables Outline Python 3.11.9

# 39. Przedstawić sposoby łączenia ramek danych za pomocą metod merge i concat
# Metoda merge:
df1 = pd.DataFrame({'key': ['A', 'B', 'C'], 'value1': [1, 2, 3]})
df2 = pd.DataFrame({'key': ['A', 'B', 'D'], 'value2': [4, 5, 6]})
merged_df = pd.merge(df1, df2, on='key', how='inner')
print(merged_df)

# Metoda concatenate:
df3 = pd.DataFrame({'key': ['A', 'B'], 'value1': [7, 8]})
df4 = pd.DataFrame({'key': ['C', 'D'], 'value1': [9, 10]})
concatenated_df = pd.concat([df3, df4], ignore_index=True)
print(concatenated_df)

[108] ✓ 0.0s Python

key value1 value2
0 A 1 4
1 B 2 5
key value1
0 A 7
1 B 8
2 C 9
3 D 10

# 40. Pokazać dodawanie nowych kolumn za pomocą operacji matematycznych
ramka_danych_csv['gdp_growth'] = ramka_danych_csv['gdp_ppp_mean'] / ramka_danych_csv['gdp_ppp_mean'].shift(1) - 1
print(ramka_danych_csv[['location_name', 'year', 'gdp_growth']].head())

[109] ✓ 0.0s Python

location_name year gdp_growth
16017 Zimbabwe 1961 3.002170
16084 Zimbabwe 2028 0.033804
16082 Zimbabwe 2026 -0.019321
16081 Zimbabwe 2015 -0.000344
16080 Zimbabwe 2024 -0.007727

# 41. Przedstawić na przykładzie dodawanie nowych kolumn z pomocą funkcji lambda
ramka_danych_csv['gdp_per_capita'] = ramka_danych_csv.apply(lambda row: row['gdp_ppp_mean'] / row['gdp_usd_mean'], axis=1)
print(ramka_danych_csv[['location_name', 'year', 'gdp_per_capita']].head())

[109] ✓ 0.1s Python

location_name year gdp_per_capita
16017 Zimbabwe 1961 3.002170
16084 Zimbabwe 2028 2.404670
```

Rys. 17. Fragment kodu Źródłowego.

Łączenie ramek danych jest kluczową operacją przy pracy z wieloma Źródłami danych, umożliwiając ich scalanie w jedną spójną ramkę danych. Dodawanie nowych kolumn na podstawie istniejących danych matematycznych pozwala na wzbogacenie analizy o nowe informacje. Użycie funkcji lambda umożliwia tworzenie prostych, dynamicznych operacji na kolumnach, co pozwala na elastyczne modyfikowanie danych.

```
Naucz o Danych > Laboratorium 1 > Laboratorium1.ipynb > # 38. Narysować histogram na podstawie wartości kolumny
Generate + Code + Markdown Run All Restart Clear All Outputs Jupyter Variables Outline Python 3.11.9

# 42. Przedstawić możliwości pracy z dużymi plikami przy użyciu argumentu chunksize
chunksize = 10000
for chunk in pd.read_csv('DME_GDP_1960_2050_Y2021M05D22.CSV', chunksize=chunksize):
    print(chunk.head(4))

[109] ✓ 0.0s Python

location_id location_name iso3 level year gdp_ppp_mean gdp_ppp_lower \
0 1 Global G Global 1960 1.748345e+13 1.601915e+13
1 1 Global G Global 1961 1.813537e+13 1.659537e+13
2 1 Global G Global 1962 1.895328e+13 1.739039e+13
3 1 Global G Global 1963 1.965662e+13 1.811706e+13

gdp_ppp_upper gdp_usd_mean gdp_usd_lower gdp_usd_upper
0 1.911586e+13 1.296863e+13 1.266890e+13 1.334177e+13
1 1.982493e+13 1.346097e+13 1.314767e+13 1.383021e+13
2 2.404477e+13 1.406537e+13 1.376866e+13 1.442746e+13
3 2.134993e+13 1.461831e+13 1.432132e+13 1.497693e+13

location_id location_name iso3 level year gdp_ppp_mean \
10000 126 Costa Rica CRI Country 2041 21222.968139
10001 126 Costa Rica CRI Country 2042 21304.771030
10002 126 Costa Rica CRI Country 2043 21531.353078
10003 126 Costa Rica CRI Country 2044 21654.766957

gdp_ppp_lower gdp_ppp_upper gdp_usd_mean gdp_usd_lower
10000 16351.045082 26608.622004 12731.983537 10072.290008
10001 16373.548021 27237.275872 12828.873351 10014.947690
10002 16295.954034 27650.146027 12916.696284 9925.971971
10003 16183.900733 28059.557778 12990.534044 9872.441363

gdp_usd_upper
10000 15839.573177
10001 16004.157051
10002 16304.105693
10003 16584.745555
```

Rys. 18. Fragment kodu Źródłowego.

Użycie chunksize pozwala na przetwarzanie dużych plików w częściach, co oszczędza pamięć i przyspiesza analizę dużych zbiorów danych.

### 3. Wnioski

Analizując każdą liniijkę kodu można zauważyć, że większość zadań można wykonać w jednej linijce kody python - ramkowanie danych to wygodny sposób na przetwarzanie danych.

Większość komend (poleceń) nie tylko jest łatwe ze względu na uniwersalny angielski język. Duża część metod została zapożyczona z systemu Linux. Takie metody jak `head()` czy `tail()` pozwalają w systemie linuksowym podejrzeć pierwsze/ostatnie kilka linijek pliku.

Duże liczby zapisywane są w postaci notacji, aby zaoszczędzić miejsce na ekranie podczas wyświetlania danych.