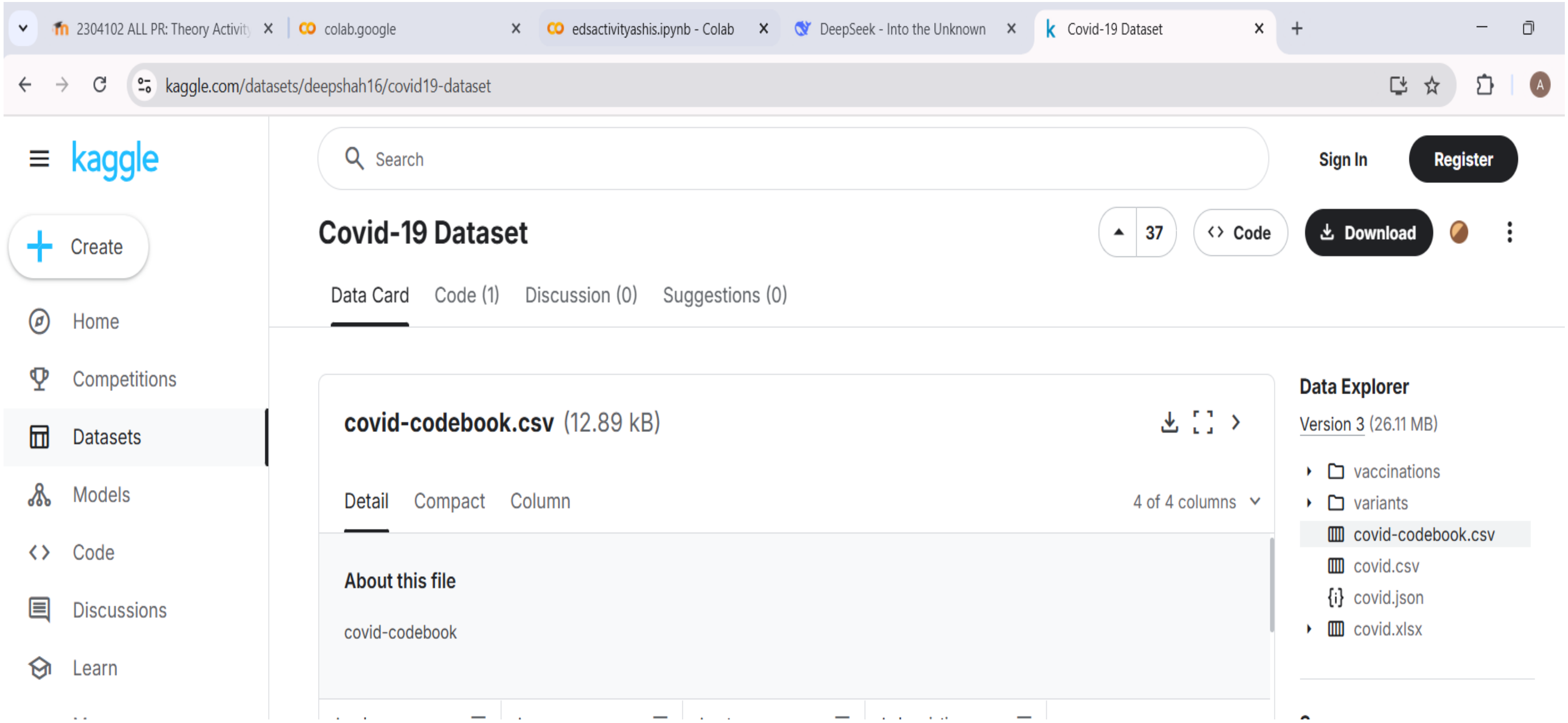
EDS ACTIVITY

NAME-ASHIS KUMAR LENKA
CLASS-CS7
ROLL NO -CS7-90
PRN-202401110068

COLLAB LINK:- https://colab.research.google.com/drive/1jhtE-CT-ij3x_Bj4AdIyM-RWyfbfJr3i?usp=sharing

DATASET:-**COVID 19** https://www.kaggle.com/datasets/deepshah16/covid19-dataset



Covid-19 Dataset

covid-codebook.csv (12.89 kB)

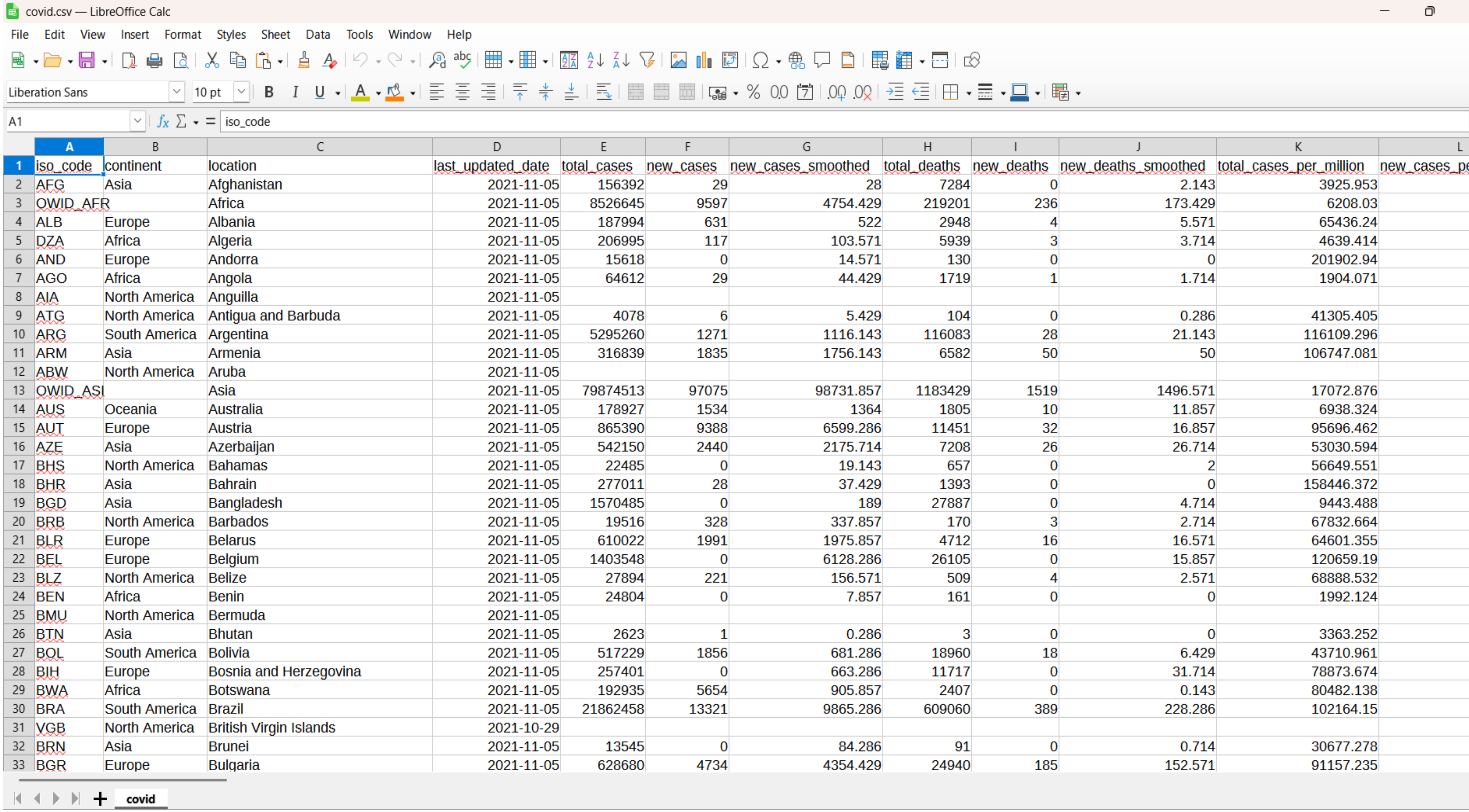
About this file

covid-codebook

Data Explorer

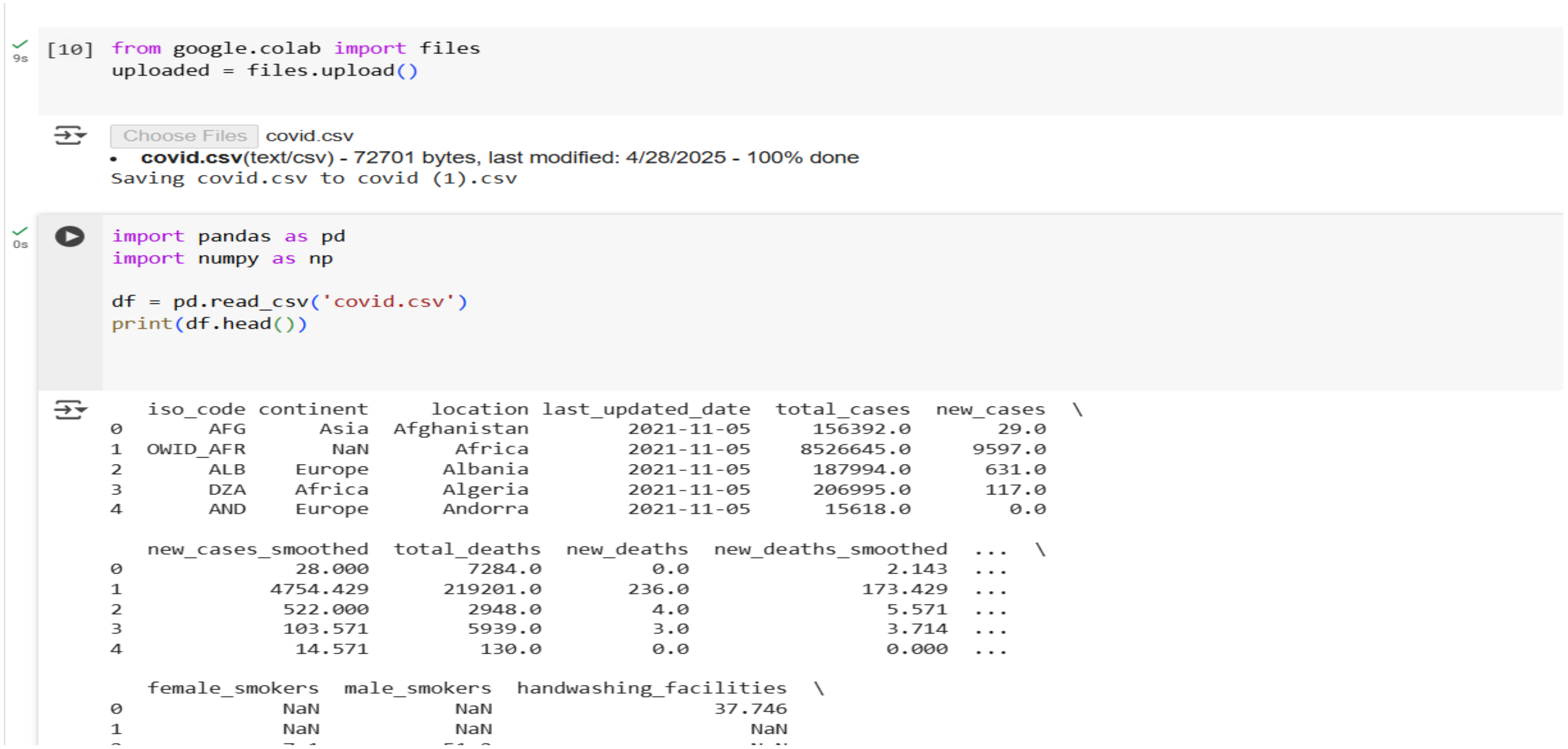
Version 3 (26.11 MB)

- vaccinations
- variants
- covid-codebook.csv
- covid.csv
- covid.json
- covid.xlsx


covid.csv — LibreOffice Calc
File Edit View Insert Format Styles Sheet Data Tools Window Help
Liberation Sans
10 pt
A1
iso_code
covid

	A	B	C	D	E	F	G	H	I	J	K	L
1	iso_code	continent	location	last_updated_date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoothed	total_cases_per_million	new_cases_pe
2	AFG	Asia	Afghanistan	2021-11-05	156392	29	28	7284	0	2.143	3925.953	
3	OWID_AFR		Africa	2021-11-05	8526645	9597	4754.429	219201	236	173.429	6208.03	
4	ALB	Europe	Albania	2021-11-05	187994	631	522	2948	4	5.571	65436.24	
5	DZA	Africa	Algeria	2021-11-05	206995	117	103.571	5939	3	3.714	4639.414	
6	AND	Europe	Andorra	2021-11-05	15618	0	14.571	130	0	0	201902.94	
7	AGO	Africa	Angola	2021-11-05	64612	29	44.429	1719	1	1.714	1904.071	
8	AIA	North America	Anguilla	2021-11-05								
9	ATG	North America	Antigua and Barbuda	2021-11-05	4078	6	5.429	104	0	0.286	41305.405	
10	ARG	South America	Argentina	2021-11-05	5295260	1271	1116.143	116083	28	21.143	116109.296	
11	ARM	Asia	Armenia	2021-11-05	316839	1835	1756.143	6582	50	50	106747.081	
12	ABW	North America	Aruba	2021-11-05								
13	OWID_ASI		Asia	2021-11-05	79874513	97075	98731.857	1183429	1519	1496.571	17072.876	
14	AUS	Oceania	Australia	2021-11-05	178927	1534	1364	1805	10	11.857	6938.324	
15	AUT	Europe	Austria	2021-11-05	865390	9388	6599.286	11451	32	16.857	95696.462	
16	AZE	Asia	Azerbaijan	2021-11-05	542150	2440	2175.714	7208	26	26.714	53030.594	
17	BHS	North America	Bahamas	2021-11-05	22485	0	19.143	657	0	2	56649.551	
18	BHR	Asia	Bahrain	2021-11-05	277011	28	37.429	1393	0	0	158446.372	
19	BGD	Asia	Bangladesh	2021-11-05	1570485	0	189	27887	0	4.714	9443.488	
20	BRB	North America	Barbados	2021-11-05	19516	328	337.857	170	3	2.714	67832.664	
21	BLR	Europe	Belarus	2021-11-05	610022	1991	1975.857	4712	16	16.571	64601.355	
22	BEL	Europe	Belgium	2021-11-05	1403548	0	6128.286	26105	0	15.857	120659.19	
23	BLZ	North America	Belize	2021-11-05	27894	221	156.571	509	4	2.571	68888.532	
24	BEN	Africa	Benin	2021-11-05	24804	0	7.857	161	0	0	1992.124	
25	BMU	North America	Bermuda	2021-11-05								
26	BTN	Asia	Bhutan	2021-11-05	2623	1	0.286	3	0	0	3363.252	
27	BOL	South America	Bolivia	2021-11-05	517229	1856	681.286	18960	18	6.429	43710.961	
28	BIH	Europe	Bosnia and Herzegovina	2021-11-05	257401	0	663.286	11717	0	31.714	78873.674	
29	BWA	Africa	Botswana	2021-11-05	192935	5654	905.857	2407	0	0.143	80482.138	
30	BRA	South America	Brazil	2021-11-05	21862458	13321	9865.286	609060	389	228.286	102164.15	
31	VGB	North America	British Virgin Islands	2021-10-29								
32	BRN	Asia	Brunei	2021-11-05	13545	0	84.286	91	0	0.714	30677.278	
33	BGR	Europe	Bulgaria	2021-11-05	628680	4734	4354.429	24940	185	152.571	91157.235	

1. Load the dataset and display the first 5 rows.



```
[10] from google.colab import files
     uploaded = files.upload()
```

Choose Files covid.csv

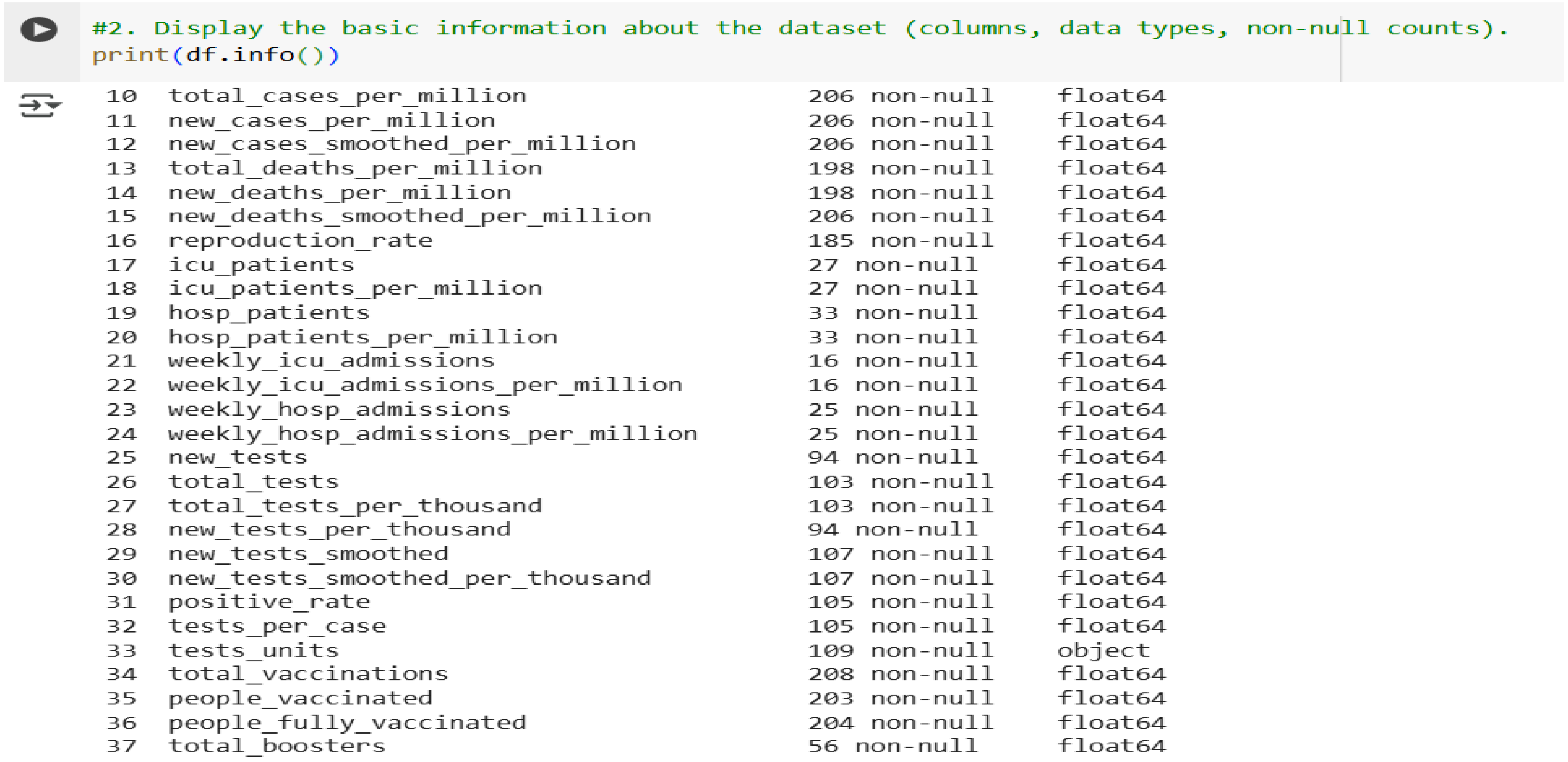
- **covid.csv**(text/csv) - 72701 bytes, last modified: 4/28/2025 - 100% done

```
Saving covid.csv to covid (1).csv
```

```
import pandas as pd
import numpy as np

df = pd.read_csv('covid.csv')
print(df.head())
```

```
   iso_code continent     location last_updated_date  total_cases  new_cases  \
0       AFG      Asia  Afghanistan        2021-11-05     156392.0       29.0
1  OWID_AFR       NaN       Africa        2021-11-05    8526645.0     9597.0
2       ALB    Europe      Albania        2021-11-05     187994.0      631.0
3       DZA    Africa      Algeria        2021-11-05     206995.0      117.0
4       AND    Europe      Andorra        2021-11-05      15618.0        0.0

   new_cases_smoothed  total_deaths  new_deaths  new_deaths_smoothed  ...  \
0              28.000        7284.0         0.0                2.143  ...
1            4754.429      219201.0       236.0              173.429  ...
2             522.000        2948.0         4.0                5.571  ...
3             103.571        5939.0         3.0                3.714  ...
4              14.571         130.0         0.0                0.000  ...

   female_smokers  male_smokers  handwashing_facilities  \
0             NaN           NaN                  37.746
1             NaN           NaN                     NaN
```

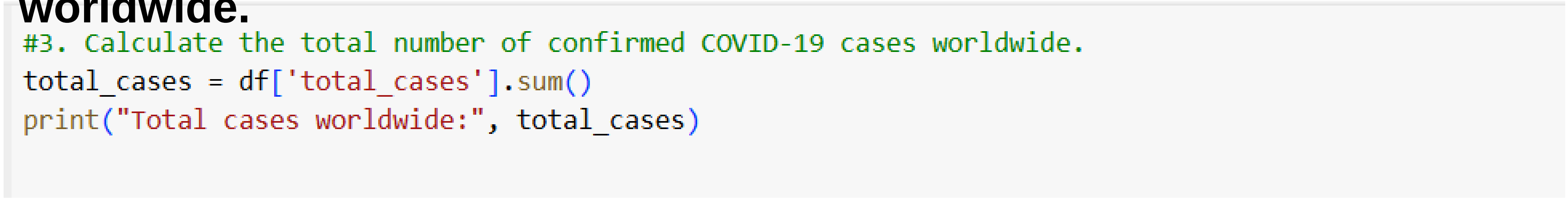
2. Display the basic information about the dataset (columns, data types, non-null counts)



```
#2. Display the basic information about the dataset (columns, data types, non-null counts).
print(df.info())
```

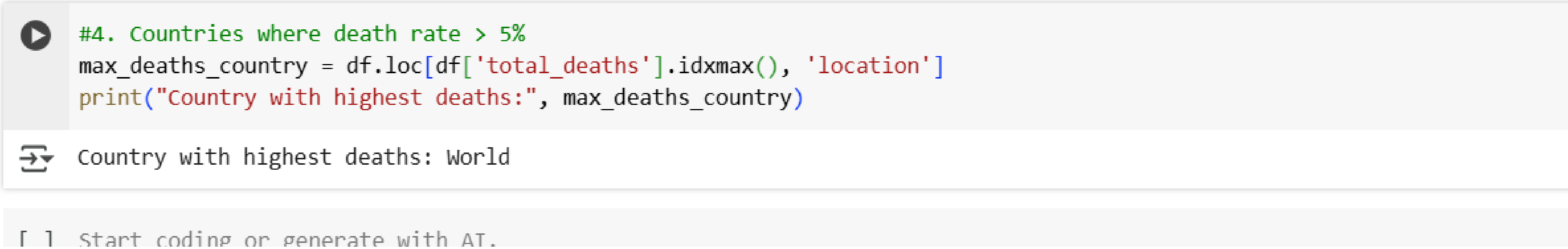
```
 10  total_cases_per_million                    206 non-null    float64
 11  new_cases_per_million                      206 non-null    float64
 12  new_cases_smoothed_per_million             206 non-null    float64
 13  total_deaths_per_million                   198 non-null    float64
 14  new_deaths_per_million                     198 non-null    float64
 15  new_deaths_smoothed_per_million            206 non-null    float64
 16  reproduction_rate                          185 non-null    float64
 17  icu_patients                               27 non-null     float64
 18  icu_patients_per_million                   27 non-null     float64
 19  hosp_patients                              33 non-null     float64
 20  hosp_patients_per_million                  33 non-null     float64
 21  weekly_icu_admissions                      16 non-null     float64
 22  weekly_icu_admissions_per_million          16 non-null     float64
 23  weekly_hosp_admissions                     25 non-null     float64
 24  weekly_hosp_admissions_per_million         25 non-null     float64
 25  new_tests                                  94 non-null     float64
 26  total_tests                                103 non-null    float64
 27  total_tests_per_thousand                   103 non-null    float64
 28  new_tests_per_thousand                     94 non-null     float64
 29  new_tests_smoothed                         107 non-null    float64
 30  new_tests_smoothed_per_thousand            107 non-null    float64
 31  positive_rate                              105 non-null    float64
 32  tests_per_case                             105 non-null    float64
 33  tests_units                                109 non-null    object
 34  total_vaccinations                         208 non-null    float64
 35  people_vaccinated                          203 non-null    float64
 36  people_fully_vaccinated                    204 non-null    float64
 37  total_boosters                             56 non-null     float64
```

3. Calculate the total number of confirmed COVID-19 cases worldwide.



```python
#3. Calculate the total number of confirmed COVID-19 cases worldwide.
total_cases = df['total_cases'].sum()
print("Total cases worldwide:", total_cases)
```

4. Find the country with the highest number of deaths

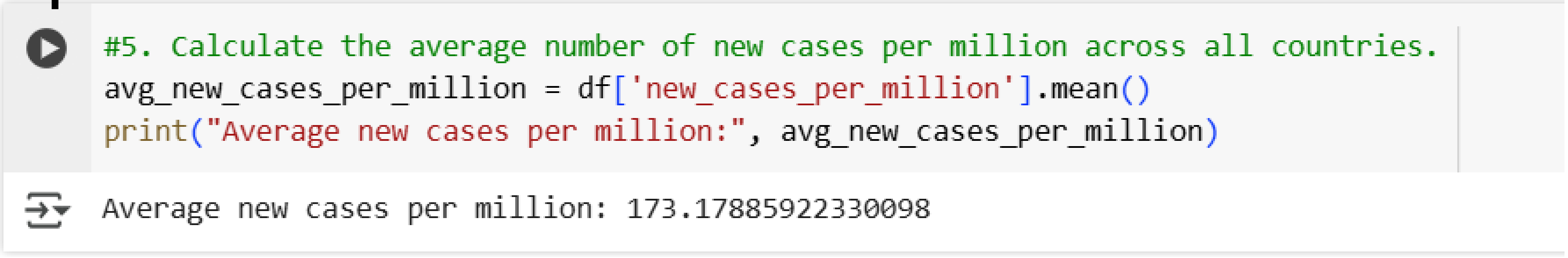


```python
#4. Countries where death rate > 5%
max_deaths_country = df.loc[df['total_deaths'].idxmax(), 'location']
print("Country with highest deaths:", max_deaths_country)
```

```
Country with highest deaths: World
```

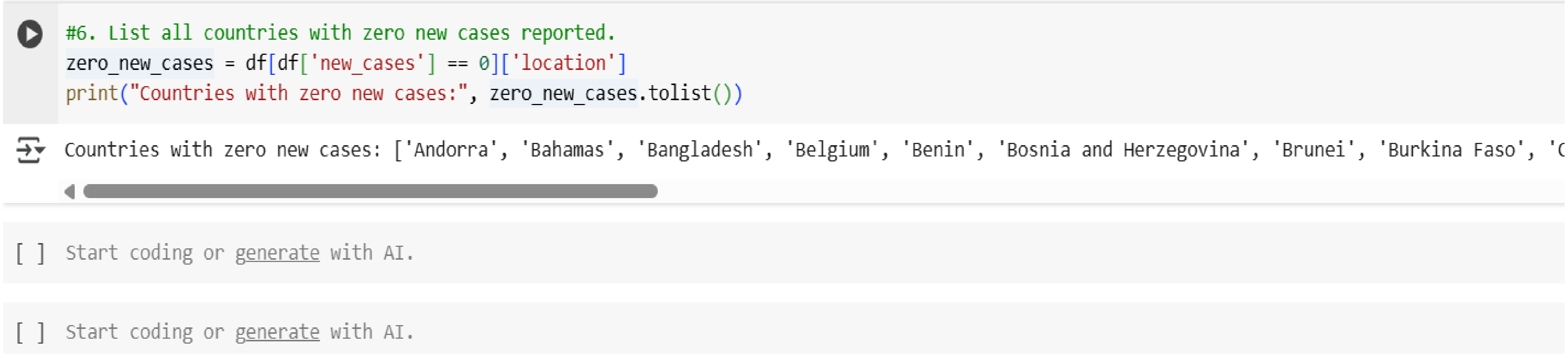
```
[ ] Start coding or generate with AI.
```

5. Calculate the average number of new cases per million across all countries.



```python
#5. Calculate the average number of new cases per million across all countries.
avg_new_cases_per_million = df['new_cases_per_million'].mean()
print("Average new cases per million:", avg_new_cases_per_million)
```

```
Average new cases per million: 173.17885922330098
```

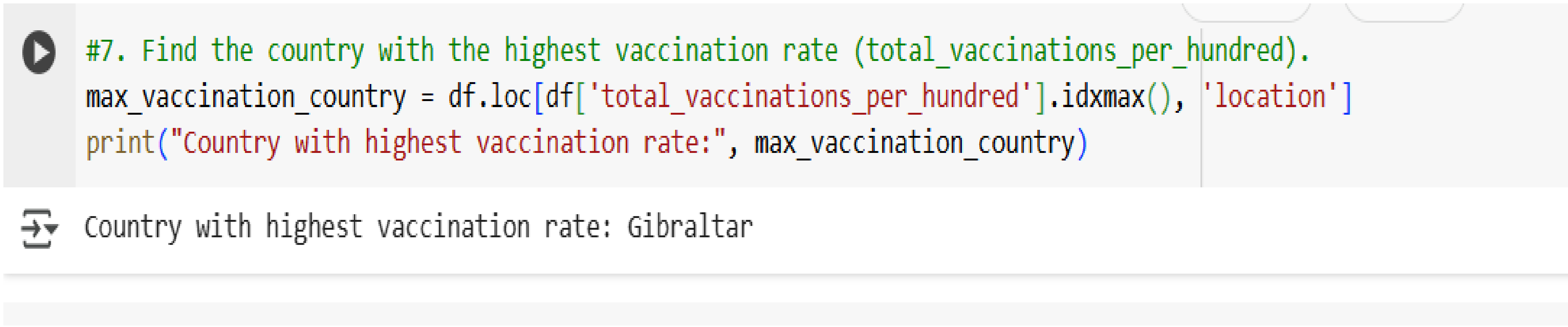


```
#6. List all countries with zero new cases reported.
zero_new_cases = df[df['new_cases'] == 0]['location']
print("Countries with zero new cases:", zero_new_cases.tolist())
```

```
Countries with zero new cases: ['Andorra', 'Bahamas', 'Bangladesh', 'Belgium', 'Benin', 'Bosnia and Herzegovina', 'Brunei', 'Burkina Faso', '
```

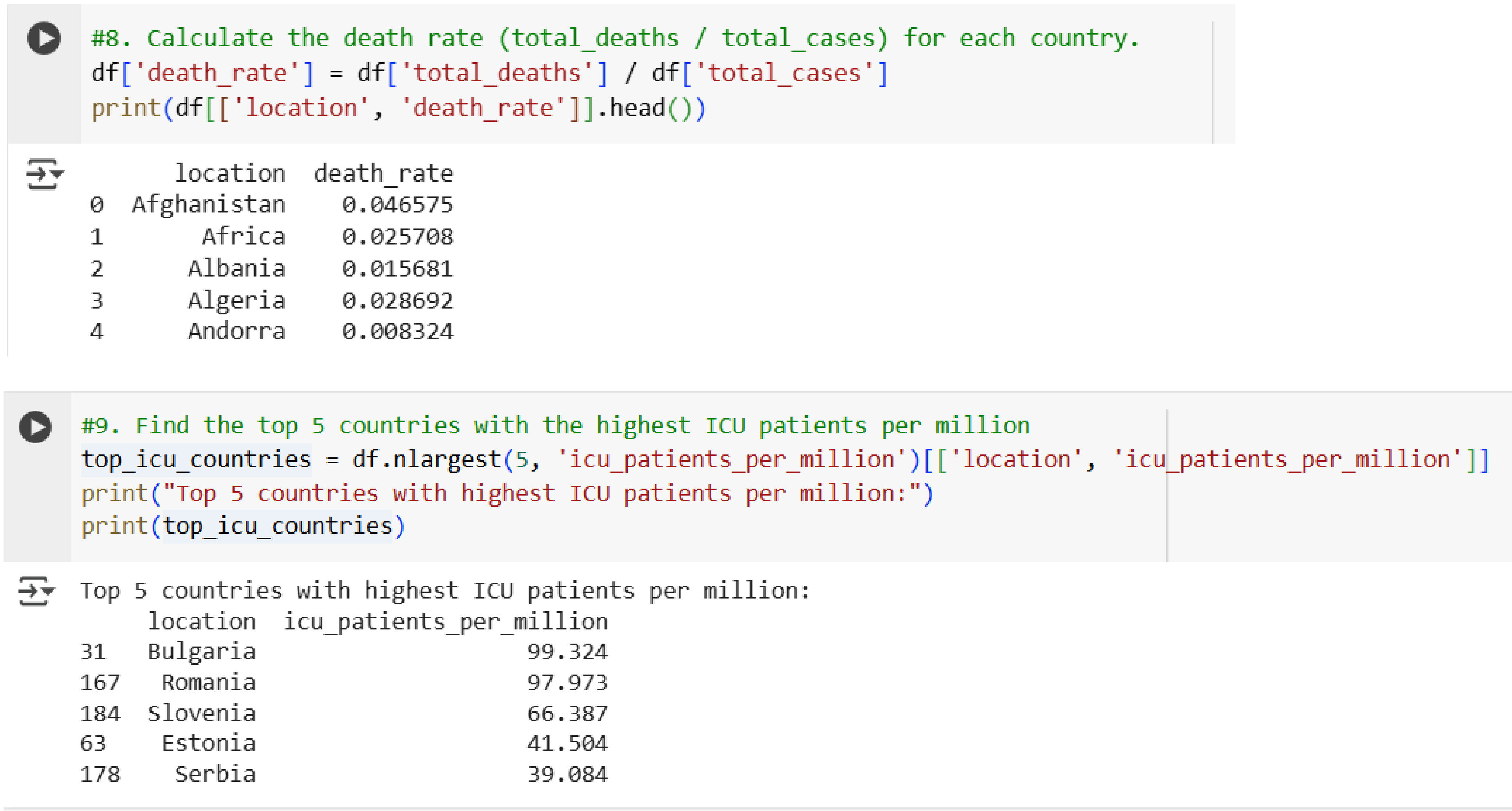
```
[ ] Start coding or generate with AI.
```

```
[ ] Start coding or generate with AI.
```



```
#7. Find the country with the highest vaccination rate (total_vaccinations_per_hundred).
max_vaccination_country = df.loc[df['total_vaccinations_per_hundred'].idxmax(), 'location']
print("Country with highest vaccination rate:", max_vaccination_country)
```

```
Country with highest vaccination rate: Gibraltar
```

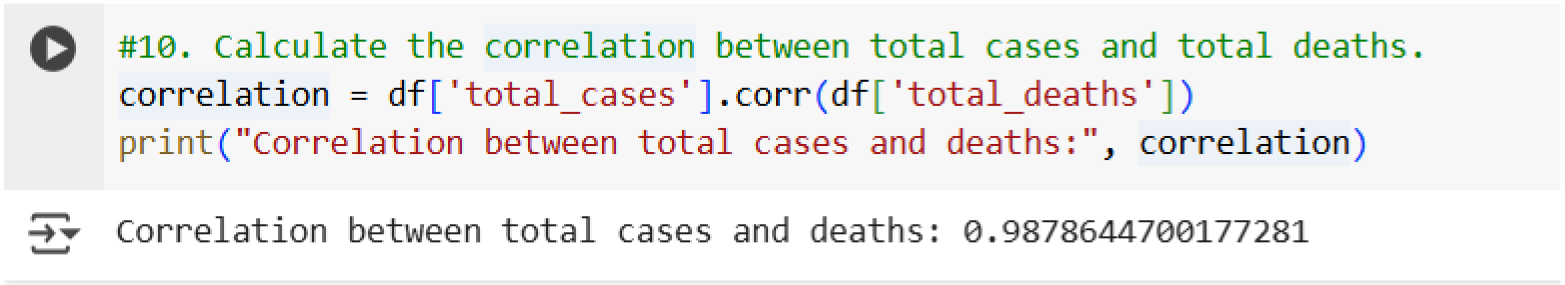


```python
#8. Calculate the death rate (total_deaths / total_cases) for each country.
df['death_rate'] = df['total_deaths'] / df['total_cases']
print(df[['location', 'death_rate']].head())
```

```
      location  death_rate
0  Afghanistan    0.046575
1       Africa    0.025708
2      Albania    0.015681
3      Algeria    0.028692
4      Andorra    0.008324
```

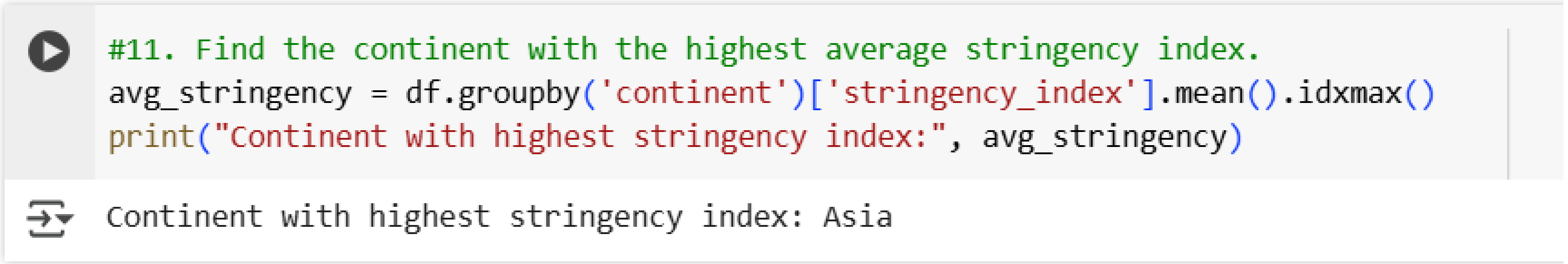
```python
#9. Find the top 5 countries with the highest ICU patients per million
top_icu_countries = df.nlargest(5, 'icu_patients_per_million')[['location', 'icu_patients_per_million']]
print("Top 5 countries with highest ICU patients per million:")
print(top_icu_countries)
```

```
Top 5 countries with highest ICU patients per million:
     location  icu_patients_per_million
31   Bulgaria                    99.324
167   Romania                    97.973
184  Slovenia                    66.387
63    Estonia                    41.504
178    Serbia                    39.084
```



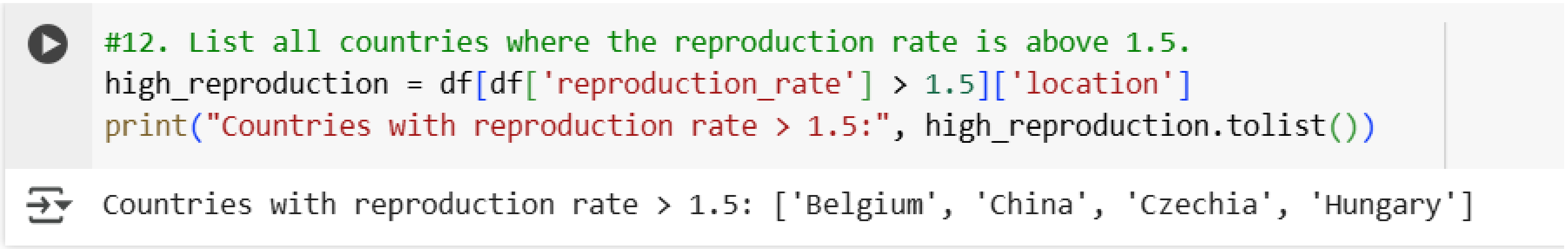
```python
#10. Calculate the correlation between total cases and total deaths.
correlation = df['total_cases'].corr(df['total_deaths'])
print("Correlation between total cases and deaths:", correlation)
```

```
Correlation between total cases and deaths: 0.9878644700177281
```



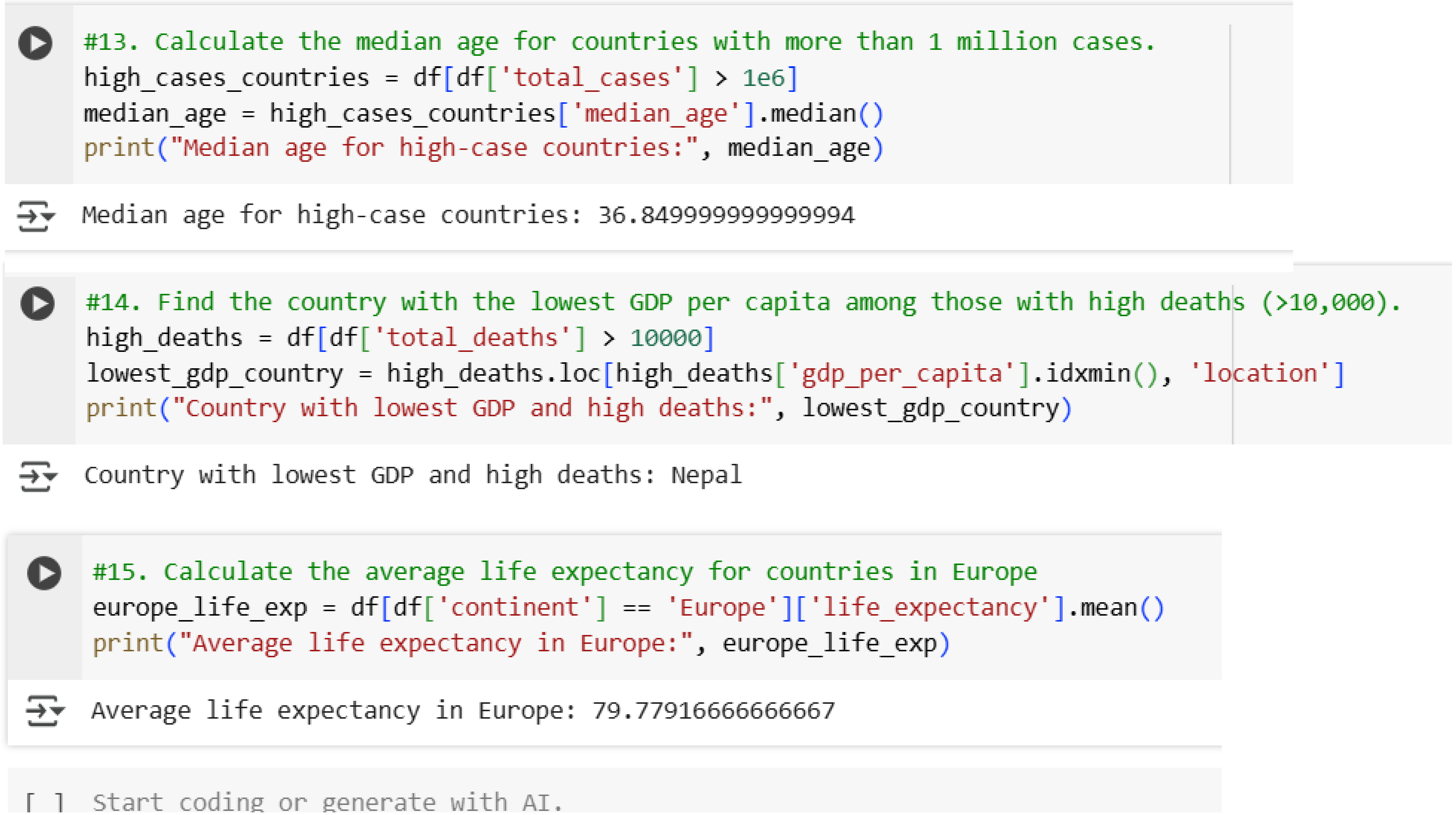
```python
#11. Find the continent with the highest average stringency index.
avg_stringency = df.groupby('continent')['stringency_index'].mean().idxmax()
print("Continent with highest stringency index:", avg_stringency)
```

```
Continent with highest stringency index: Asia
```



```python
#12. List all countries where the reproduction rate is above 1.5.
high_reproduction = df[df['reproduction_rate'] > 1.5]['location']
print("Countries with reproduction rate > 1.5:", high_reproduction.tolist())
```

```
Countries with reproduction rate > 1.5: ['Belgium', 'China', 'Czechia', 'Hungary']
```



```python
#13. Calculate the median age for countries with more than 1 million cases.
high_cases_countries = df[df['total_cases'] > 1e6]
median_age = high_cases_countries['median_age'].median()
print("Median age for high-case countries:", median_age)
```

```
Median age for high-case countries: 36.849999999999994
```

```python
#14. Find the country with the lowest GDP per capita among those with high deaths (>10,000).
high_deaths = df[df['total_deaths'] > 10000]
lowest_gdp_country = high_deaths.loc[high_deaths['gdp_per_capita'].idxmin(), 'location']
print("Country with lowest GDP and high deaths:", lowest_gdp_country)
```

```
Country with lowest GDP and high deaths: Nepal
```

```python
#15. Calculate the average life expectancy for countries in Europe
europe_life_exp = df[df['continent'] == 'Europe']['life_expectancy'].mean()
print("Average life expectancy in Europe:", europe_life_exp)
```

```
Average life expectancy in Europe: 79.77916666666667
```

```
[ ]  Start coding or generate with AI.
```

```python
#16. Find the country with the highest number of hospital beds per thousand.
max_beds_country = df.loc[df['hospital_beds_per_thousand'].idxmax(), 'location']
print("Country with most hospital beds per thousand:", max_beds_country)
```

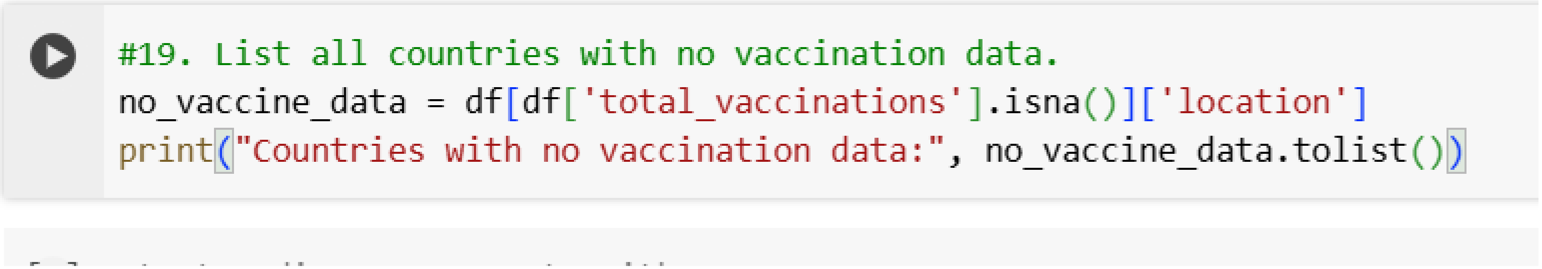
```
Country with most hospital beds per thousand: Monaco
```

```python
#17. Calculate the percentage of fully vaccinated people for each country.
df['fully_vaccinated_pct'] = (df['people_fully_vaccinated'] / df['population']) * 100
print(df[['location', 'fully_vaccinated_pct']].head())
```

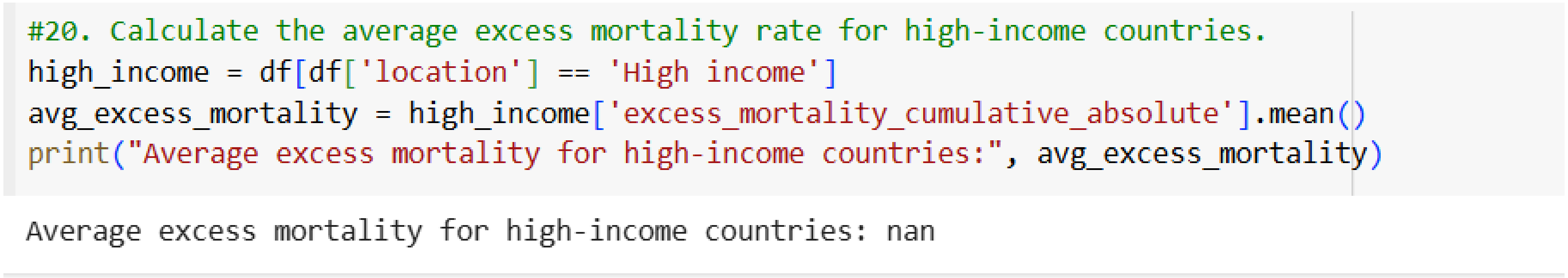
```
      location  fully_vaccinated_pct
0  Afghanistan              6.386835
1       Africa              6.167671
2      Albania             31.278999
3      Algeria             10.855816
4      Andorra                   NaN
```

```python
#18. Find the country with the highest positive test rate (positive_rate).
max_positive_rate_country = df.loc[df['positive_rate'].idxmax(), 'location']
print("Country with highest positive test rate:", max_positive_rate_country)
```

```
Country with highest positive test rate: Slovenia
```



```
#19. List all countries with no vaccination data.
no_vaccine_data = df[df['total_vaccinations'].isna()]['location']
print("Countries with no vaccination data:", no_vaccine_data.tolist())
```



```
#20. Calculate the average excess mortality rate for high-income countries.
high_income = df[df['location'] == 'High income']
avg_excess_mortality = high_income['excess_mortality_cumulative_absolute'].mean()
print("Average excess mortality for high-income countries:", avg_excess_mortality)
```

```
Average excess mortality for high-income countries: nan
```