Welcome to the Covid-19 Data Analysis Notebook

Let's import the modules

In [402…

```
# Importing necessary libraries for data analysis and visualization
import pandas as pd          # For data manipulation and analysis
import numpy as np           # For numerical operations
import seaborn as sns        # For statistical data visualization
import matplotlib.pyplot as plt  # For plotting graphs and figures

print('Modules are imported.')
```

```
Modules are imported.
```

Task 2

Task 2.1: Importing the Covid-19 dataset

Importing "Covid19_Confirmed_dataset.csv" from the "./Dataset" folder.

In [403…

```
# Import dataset.
data = pd.read_csv('Datasets/covid19_Confirmed_dataset.csv')
data.head(10)
```

Out[403...

	Province/ State	Country/ Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/2
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	
1	NaN	Albania	41.1533	20.1683	0	0	0	0	
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	
5	NaN	Antigua and Barbuda	17.0608	-61.7964	0	0	0	0	
6	NaN	Argentina	-38.4161	-63.6167	0	0	0	0	
7	NaN	Armenia	40.0691	45.0382	0	0	0	0	
8	Australian Capital Territory	Australia	-35.4735	149.0124	0	0	0	0	
9	New South Wales	Australia	-33.8688	151.2093	0	0	0	0	

10 rows × 104 columns

Let's check the shape of the DataFrame

In [404...
```
data.shape
```

Out[404...
```
(266, 104)
```

Task 2.2: Delete the unnecessary columns

In [405...
```
data.drop(['Lat', 'Long'], axis=1, inplace=True)  # Dropping unnecessary columns
```

In [406...
```
data.head(15)
```

Out[406...

	Province/ State	Country/ Region	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/
0	NaN	Afghanistan	0	0	0	0	0	0	
1	NaN	Albania	0	0	0	0	0	0	
2	NaN	Algeria	0	0	0	0	0	0	
3	NaN	Andorra	0	0	0	0	0	0	
4	NaN	Angola	0	0	0	0	0	0	
5	NaN	Antigua and Barbuda	0	0	0	0	0	0	
6	NaN	Argentina	0	0	0	0	0	0	
7	NaN	Armenia	0	0	0	0	0	0	
8	Australian Capital Territory	Australia	0	0	0	0	0	0	
9	New South Wales	Australia	0	0	0	0	3	4	
10	Northern Territory	Australia	0	0	0	0	0	0	
11	Queensland	Australia	0	0	0	0	0	0	
12	South Australia	Australia	0	0	0	0	0	0	
13	Tasmania	Australia	0	0	0	0	0	0	
14	Victoria	Australia	0	0	0	0	1	1	

15 rows × 102 columns

Task 2.3: Aggregating the rows by country

In [407...
```
data_aggregated = data.groupby('Country/Region').sum()
```

In [408...
```
data_aggregated.head(10)
```

Out[408...

Country/ Region	Province/ State	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20
Afghanistan		0	0	0	0	0	0	0
Albania		0	0	0	0	0	0	0
Algeria		0	0	0	0	0	0	0
Andorra		0	0	0	0	0	0	0
Angola		0	0	0	0	0	0	0
Antigua and Barbuda		0	0	0	0	0	0	0
Argentina		0	0	0	0	0	0	0
Armenia		0	0	0	0	0	0	0
Australia	Australian Capital TerritoryNew South WalesNor...	0	0	0	0	4	5	5
Austria		0	0	0	0	0	0	0

10 rows × 101 columns

In [409...
```
data_aggregated.shape
```

Out[409... `(187, 101)`

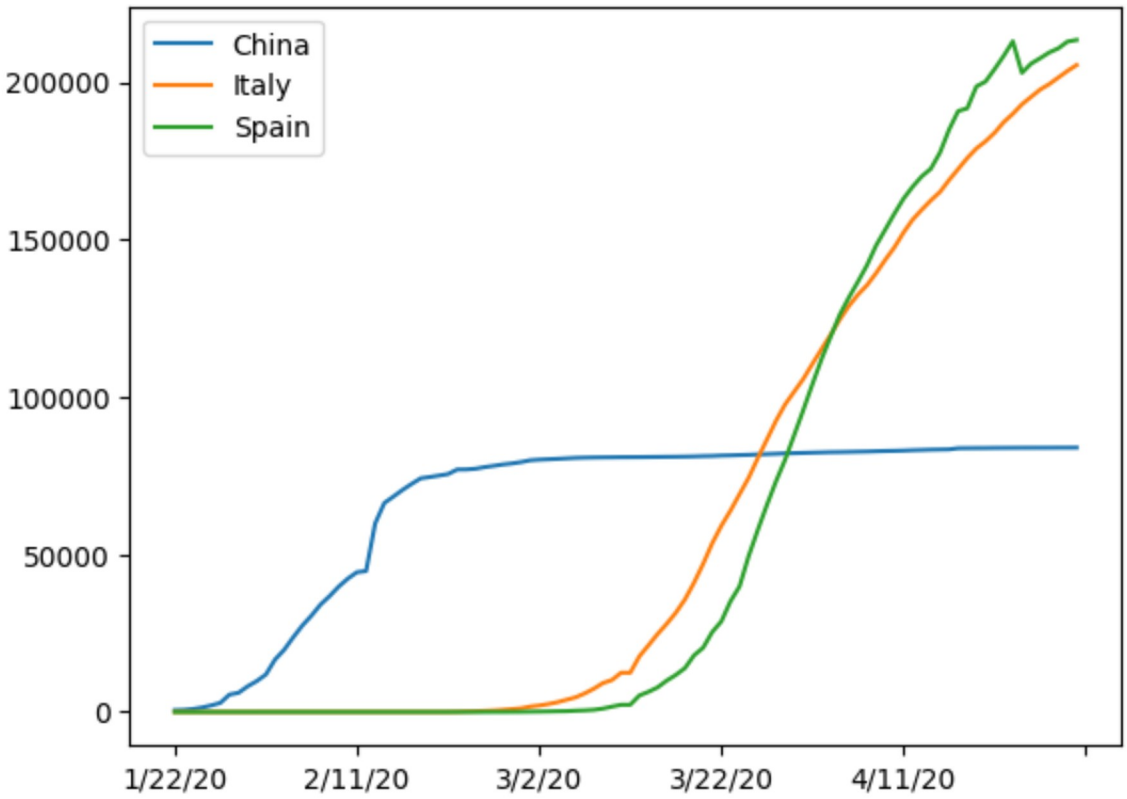
Task 2.4: Visualizing data related to a country, for example, China

Visualization always helps us better understand our data.

In [410...
```
# Select only the numeric (date) columns for China and plot
data_aggregated.loc['China'].drop('Province/State', errors='ignore').plot()
data_aggregated.loc['Italy'].drop('Province/State', errors='ignore').plot()
data_aggregated.loc['Spain'].drop('Province/State', errors='ignore').plot()

plt.legend()
```

Out[410... `<matplotlib.legend.Legend at 0x2559cd4e8d0>`

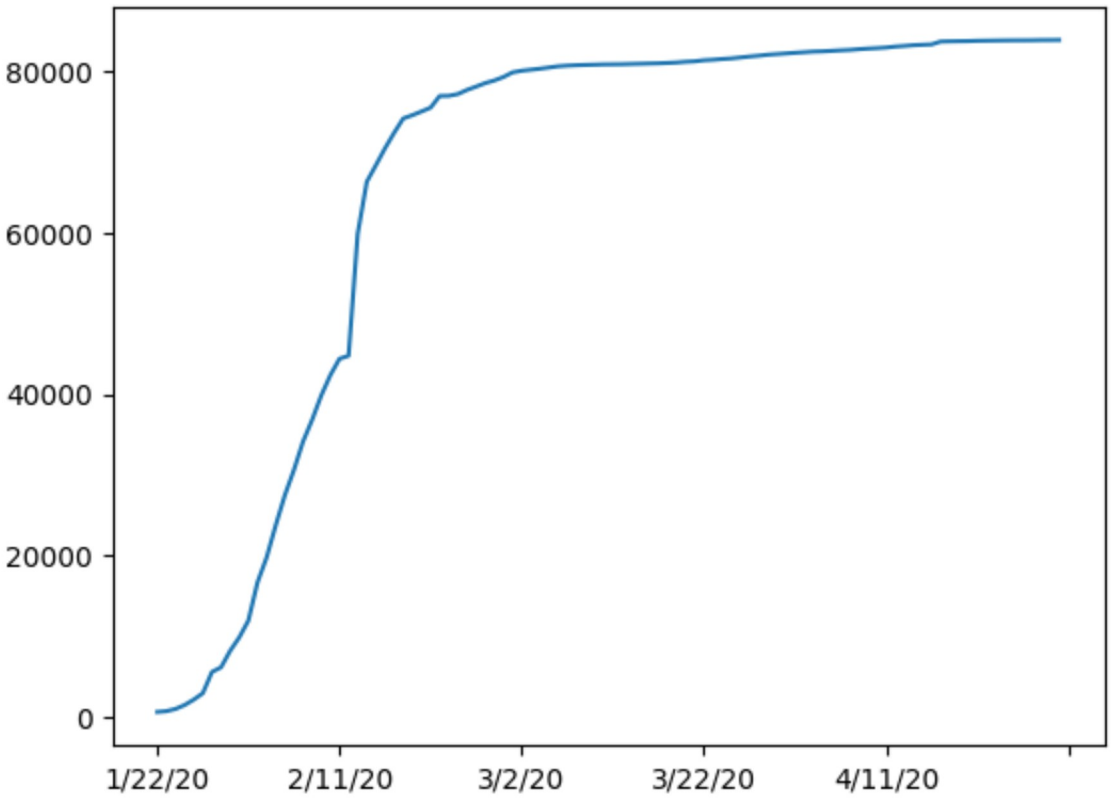


Task 3: Calculating a good measure

We need to find a good measure, represented as a number, describing the spread of the virus in a country.

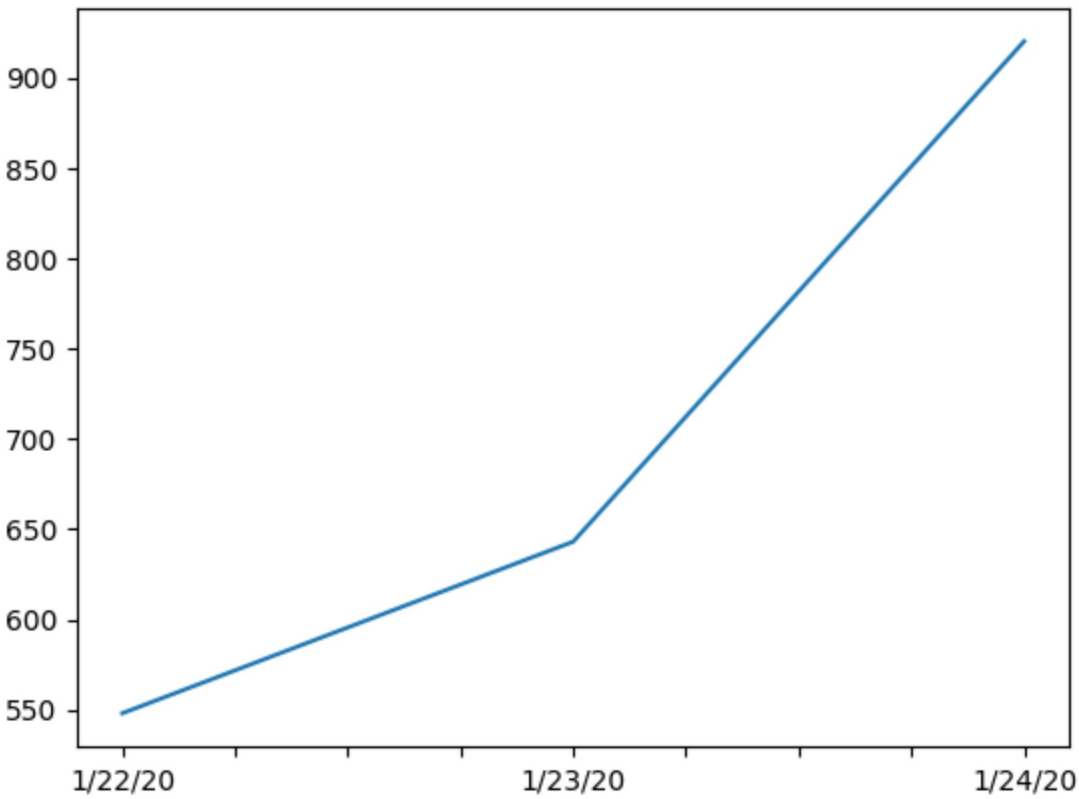
In [411…
```
data_aggregated.loc['China'].drop('Province/State', errors='ignore').plot()
```

Out[411… <Axes: >



In [ ]:
```
# Plot the first 3 date columns for China (excluding 'Province/State' if present)
data_aggregated.loc['China'].drop('Province/State', errors='ignore').iloc[:3].plot(
```

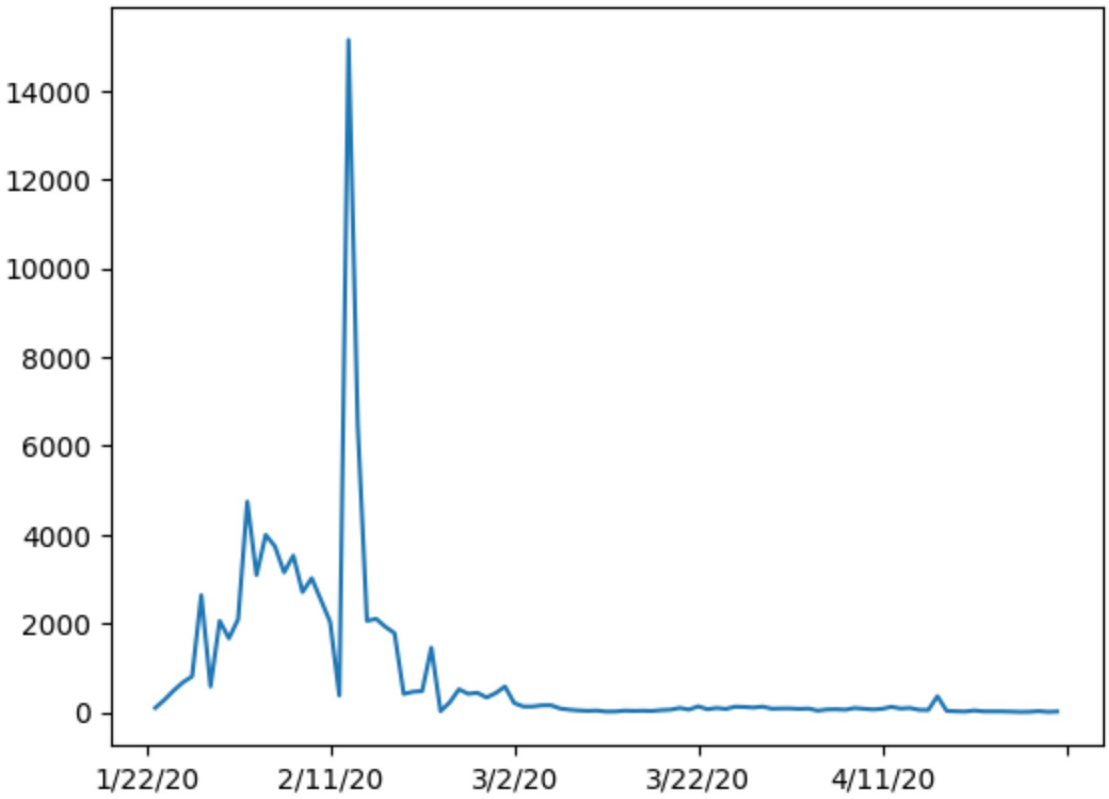
Out[ ]: <Axes: >



Task 3.1: Calculating the first derivative of the curve

In [ ]:
```
data_aggregated.loc['China'].drop('Province/State', errors='ignore').diff().plot()
```

Out[ ]: <Axes: >



Task 3.2: Find the maximum infection rate for China

In [414…
```
data_aggregated.loc['China'].drop('Province/State', errors='ignore').diff().max()
```

Out[414… 15136

In [415…
```
data_aggregated.loc['Italy'].drop('Province/State', errors='ignore').diff().max()
```

Out[415… 6557

In [416…
```
data_aggregated.loc['Spain'].drop('Province/State', errors='ignore').diff().max()
```

Out[416… 9630

In [417…
```
data_aggregated.loc['US'].drop('Province/State', errors='ignore').diff().max()
```

Out[417… 36188

Task 3.3: Find the maximum infection rate for all countries

In [418…
```
max_infection_rates = []
```

```
for c in countries:
    max_infection_rates.append(
        data_aggregated.loc[c].drop('Province/State', errors='ignore').diff().max()
    )
data_aggregated['max_infection_rate'] = max_infection_rates
```

In [419…
```
data_aggregated.head(10)
```

Out[419…

Country/ Region	Province/ State	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20
Afghanistan	0	0	0	0	0	0	0	0
Albania	0	0	0	0	0	0	0	0
Algeria	0	0	0	0	0	0	0	0
Andorra	0	0	0	0	0	0	0	0
Angola	0	0	0	0	0	0	0	0
Antigua and Barbuda	0	0	0	0	0	0	0	0
Argentina	0	0	0	0	0	0	0	0
Armenia	0	0	0	0	0	0	0	0
Australia	Australian Capital TerritoryNew South WalesNor...	0	0	0	0	4	5	5
Austria	0	0	0	0	0	0	0	0

10 rows × 102 columns

Task 3.4: Create a new DataFrame with only the needed column

In [420…
```
data_dropped = pd.DataFrame(data_aggregated['max_infection_rate'])
```

In [421…
```
data_dropped.head(10)
```

Out[421...

Country/Region	max_infection_rate
Afghanistan	232
Albania	34
Algeria	199
Andorra	43
Angola	5
Antigua and Barbuda	6
Argentina	291
Armenia	134
Australia	497
Austria	1321

In [422...
```
corona_data = data_dropped
```

Task4:

- Importing the WorldHappinessReport.csv dataset
- selecting needed columns for our analysis
- join the datasets
- calculate the correlations as the result of our analysis

Task 4.1 : importing the dataset

In [423...
```
happiness_report = pd.read_csv('Datasets/worldwide_happiness_report.csv')
```

In [424...
```
happiness_report.head(10)
```

Out[424...

	Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Percepti corrup
0	1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0
1	2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0
2	3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0
3	4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0
4	5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0
5	6	Switzerland	7.480	1.452	1.526	1.052	0.572	0.263	0
6	7	Sweden	7.343	1.387	1.487	1.009	0.574	0.267	0
7	8	New Zealand	7.307	1.303	1.557	1.026	0.585	0.330	0
8	9	Canada	7.278	1.365	1.505	1.039	0.584	0.285	0
9	10	Austria	7.246	1.376	1.475	1.016	0.532	0.244	0

Task 4.2: let's drop the useless columns

In [425...
```
columns_to_drop = ['Overall rank', 'Score', 'Generosity', 'Perceptions of corruptio
```

In [426...
```
happiness_report.drop(columns=columns_to_drop, axis=1,inplace=True)
happiness_report.head(10)
```

Out[426...

	Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
0	Finland	1.340	1.587	0.986	0.596
1	Denmark	1.383	1.573	0.996	0.592
2	Norway	1.488	1.582	1.028	0.603
3	Iceland	1.380	1.624	1.026	0.591
4	Netherlands	1.396	1.522	0.999	0.557
5	Switzerland	1.452	1.526	1.052	0.572
6	Sweden	1.387	1.487	1.009	0.574
7	New Zealand	1.303	1.557	1.026	0.585
8	Canada	1.365	1.505	1.039	0.584
9	Austria	1.376	1.475	1.016	0.532

Task 4.3: changing the indices of the dataframe

In [427…
```
happiness_report.set_index('Country or region', inplace=True)  # Set 'Country name'
happiness_report.head(10)
```

Out[427…

Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Finland	1.340	1.587	0.986	0.596
Denmark	1.383	1.573	0.996	0.592
Norway	1.488	1.582	1.028	0.603
Iceland	1.380	1.624	1.026	0.591
Netherlands	1.396	1.522	0.999	0.557
Switzerland	1.452	1.526	1.052	0.572
Sweden	1.387	1.487	1.009	0.574
New Zealand	1.303	1.557	1.026	0.585
Canada	1.365	1.505	1.039	0.584
Austria	1.376	1.475	1.016	0.532

Task4.4: now let's join two dataset we have prepared

Corona Dataset:

In [428…
```
corona_data.head(10)
```

Out[428...

Country/Region	max_infection_rate
Afghanistan	232
Albania	34
Algeria	199
Andorra	43
Angola	5
Antigua and Barbuda	6
Argentina	291
Armenia	134
Australia	497
Austria	1321

In [429...
```
corona_data.shape
```

Out[429... (187, 1)

World Happiness Report Dataset:

In [430...
```
happiness_report.head(10)
```

Out[430...

Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Finland	1.340	1.587	0.986	0.596
Denmark	1.383	1.573	0.996	0.592
Norway	1.488	1.582	1.028	0.603
Iceland	1.380	1.624	1.026	0.591
Netherlands	1.396	1.522	0.999	0.557
Switzerland	1.452	1.526	1.052	0.572
Sweden	1.387	1.487	1.009	0.574
New Zealand	1.303	1.557	1.026	0.585
Canada	1.365	1.505	1.039	0.584
Austria	1.376	1.475	1.016	0.532

In [431…
```
happiness_report.shape
```

Out[431… (156, 4)

Task 4.5: correlation matrix

In [432…
```
combined_data = corona_data.join(happiness_report, how='inner')
combined_data.head(10)
```

Out[432…

	max_infection_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Afghanistan	232	0.350	0.517	0.361	0.000
Albania	34	0.947	0.848	0.874	0.383
Algeria	199	1.002	1.160	0.785	0.086
Argentina	291	1.092	1.432	0.881	0.471
Armenia	134	0.850	1.055	0.815	0.283
Australia	497	1.372	1.548	1.036	0.557
Austria	1321	1.376	1.475	1.016	0.532
Azerbaijan	105	1.043	1.147	0.769	0.351
Bahrain	301	1.362	1.368	0.871	0.536
Bangladesh	641	0.562	0.928	0.723	0.527

Task 5: Visualization of the results

our Analysis is not finished unless we visualize the results in terms figures and graphs so that everyone can understand what you get out of our analysis

In [433…
```
combined_data.corr().style.background_gradient(cmap='coolwarm')
```

Out[433…

	max_infection_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
max_infection_rate	1.000000	0.250118	0.191958	0.289263	0.078196
GDP per capita	0.250118	1.000000	0.759468	0.863062	0.394603
Social support	0.191958	0.759468	1.000000	0.765286	0.456246
Healthy life expectancy	0.289263	0.863062	0.765286	1.000000	0.427892
Freedom to make life choices	0.078196	0.394603	0.456246	0.427892	1.000000

Task 5.1: Plotting GDP per capita vs. maximum infection rate

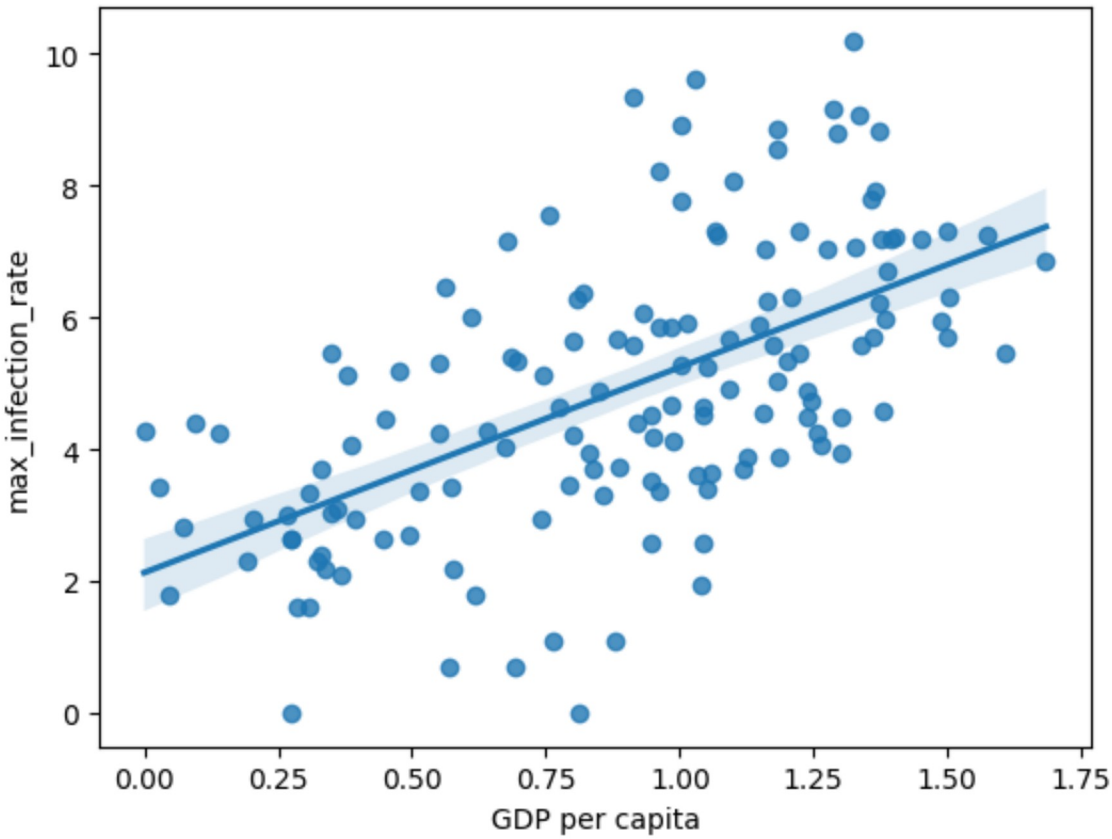
In [434…
```
combined_data.head(10)
```

Out[434…

	max_infection_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Afghanistan	232	0.350	0.517	0.361	0.000
Albania	34	0.947	0.848	0.874	0.383
Algeria	199	1.002	1.160	0.785	0.086
Argentina	291	1.092	1.432	0.881	0.471
Armenia	134	0.850	1.055	0.815	0.283
Australia	497	1.372	1.548	1.036	0.557
Austria	1321	1.376	1.475	1.016	0.532
Azerbaijan	105	1.043	1.147	0.769	0.351
Bahrain	301	1.362	1.368	0.871	0.536
Bangladesh	641	0.562	0.928	0.723	0.527

In [435…
```
x = combined_data['GDP per capita']
y = combined_data['max_infection_rate']
sns.regplot(x=x, y=np.log(y))
```

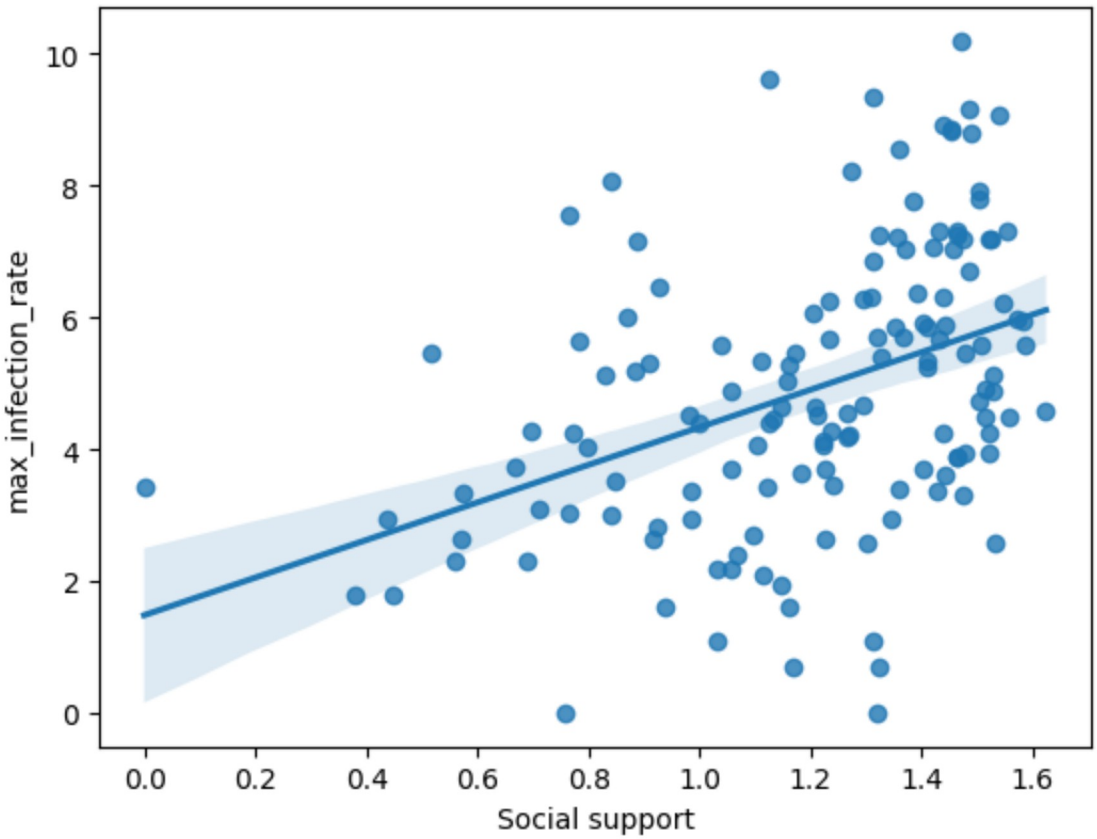
Out[435…
```
<Axes: xlabel='GDP per capita', ylabel='max_infection_rate'>
```



Task 5.2: Plotting Social Support vs. Maximum Infection Rate

In [436…
```
x = combined_data['Social support']
y = combined_data['max_infection_rate']
sns.regplot(x=x, y=np.log(y))
```

Out[436…
```
<Axes: xlabel='Social support', ylabel='max_infection_rate'>
```



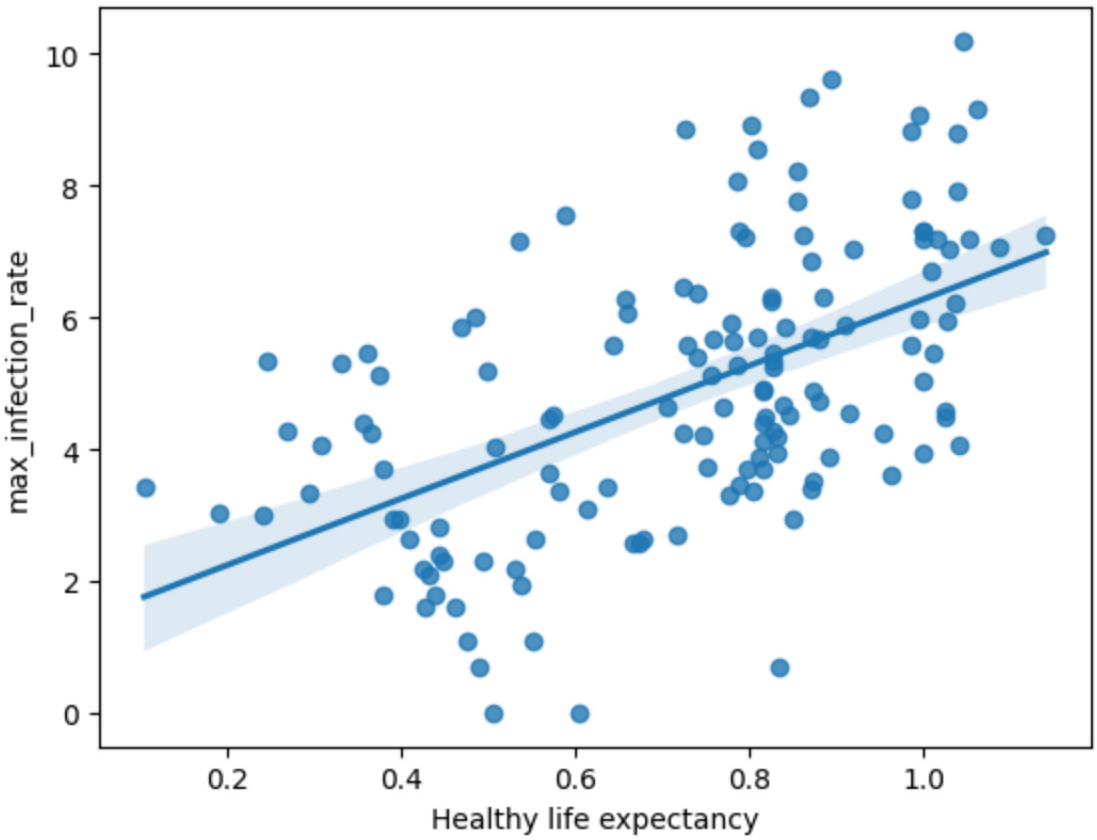
Task 5.3: Plotting Healthy Life Expectancy vs. Maximum Infection Rate

In [437…

```
x = combined_data['Healthy life expectancy']
y = combined_data['max_infection_rate']
sns.regplot(x=x, y=np.log(y))
```

Out[437…

```
<Axes: xlabel='Healthy life expectancy', ylabel='max_infection_rate'>
```



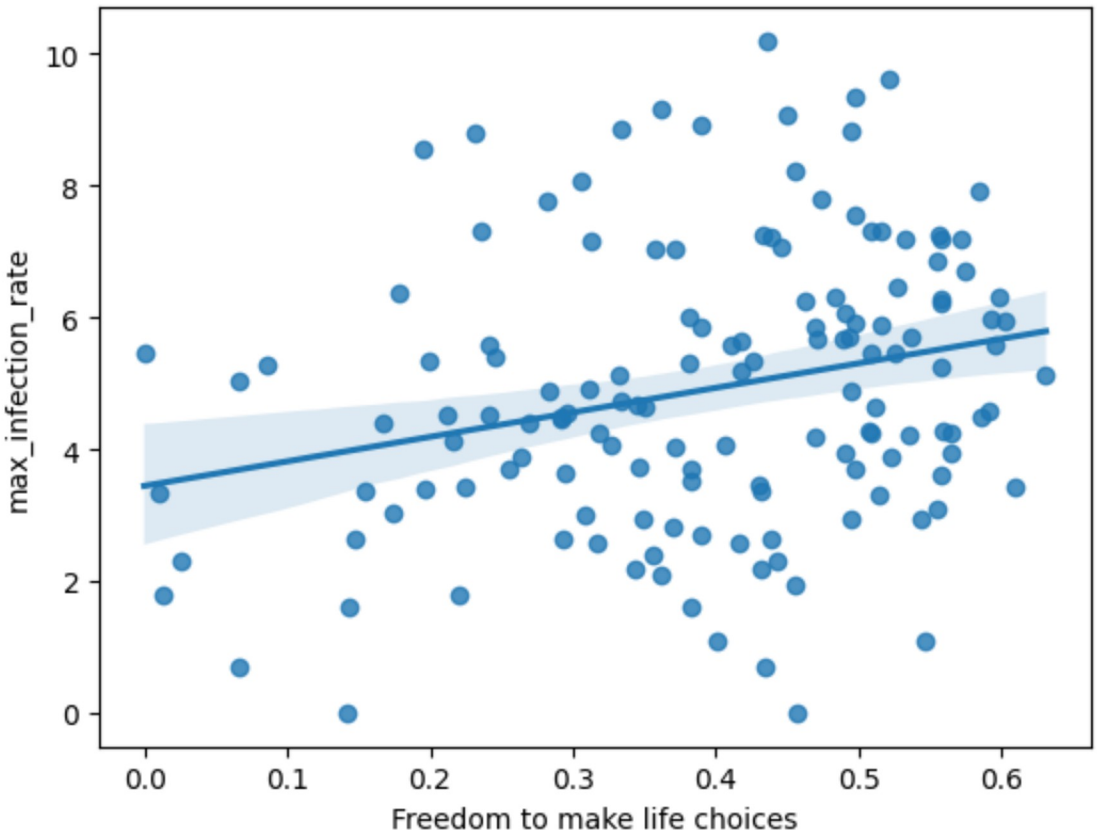
Task 5.4: Plotting Freedom to Make Life Choices vs. Maximum Infection Rate

In [438…

```
x = combined_data['Freedom to make life choices']
y = combined_data['max_infection_rate']
sns.regplot(x=x, y=np.log(y))
```

Out[438…

```
<Axes: xlabel='Freedom to make life choices', ylabel='max_infection_rate'>
```



Task 6:

- Import the covid19_deaths_dataset.csv dataset
- Select the needed columns for our analysis
- Join the datasets
- Calculate the correlations as the result of our analysis

Task 6.1: Importing the dataset - Covid-19 Deaths

In [439…
```
covid19_deaths = pd.read_csv('Dataset for practice/covid19_deaths_dataset.csv')
```

In [440…
```
covid19_deaths.head(10)
```

Out[440…

	Province/ State	Country/ Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/2
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	
1	NaN	Albania	41.1533	20.1683	0	0	0	0	
2	NaN	Algeria	28.0339	1.6596	0	0	0	0	
3	NaN	Andorra	42.5063	1.5218	0	0	0	0	
4	NaN	Angola	-11.2027	17.8739	0	0	0	0	
5	NaN	Antigua and Barbuda	17.0608	-61.7964	0	0	0	0	
6	NaN	Argentina	-38.4161	-63.6167	0	0	0	0	
7	NaN	Armenia	40.0691	45.0382	0	0	0	0	
8	Australian Capital Territory	Australia	-35.4735	149.0124	0	0	0	0	
9	New South Wales	Australia	-33.8688	151.2093	0	0	0	0	

10 rows × 104 columns

Task 6.2: Delete the unnecessary columns

In [441…
```
covid19_deaths.drop(['Lat', 'Long'], axis=1, inplace=True)  # Dropping unnecessary
```

In [442…
```
covid19_deaths.head(10)
```

Out[442...

	Province/ State	Country/ Region	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20
0	NaN	Afghanistan	0	0	0	0	0	0	0
1	NaN	Albania	0	0	0	0	0	0	0
2	NaN	Algeria	0	0	0	0	0	0	0
3	NaN	Andorra	0	0	0	0	0	0	0
4	NaN	Angola	0	0	0	0	0	0	0
5	NaN	Antigua and Barbuda	0	0	0	0	0	0	0
6	NaN	Argentina	0	0	0	0	0	0	0
7	NaN	Armenia	0	0	0	0	0	0	0
8	Australian Capital Territory	Australia	0	0	0	0	0	0	0
9	New South Wales	Australia	0	0	0	0	0	0	0

10 rows × 102 columns

In [443...
```
covid19_deaths_aggregated = covid19_deaths.groupby('Country/Region').sum()
```

In [444...
```
covid19_deaths_aggregated.head(10)
```

Out[444...

Country/ Region	Province/ State	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20
Afghanistan	0	0	0	0	0	0	0	0
Albania	0	0	0	0	0	0	0	0
Algeria	0	0	0	0	0	0	0	0
Andorra	0	0	0	0	0	0	0	0
Angola	0	0	0	0	0	0	0	0
Antigua and Barbuda	0	0	0	0	0	0	0	0
Argentina	0	0	0	0	0	0	0	0
Armenia	0	0	0	0	0	0	0	0
Australia	Australian Capital TerritoryNew South WalesNor...	0	0	0	0	0	0	0
Austria	0	0	0	0	0	0	0	0

10 rows × 101 columns

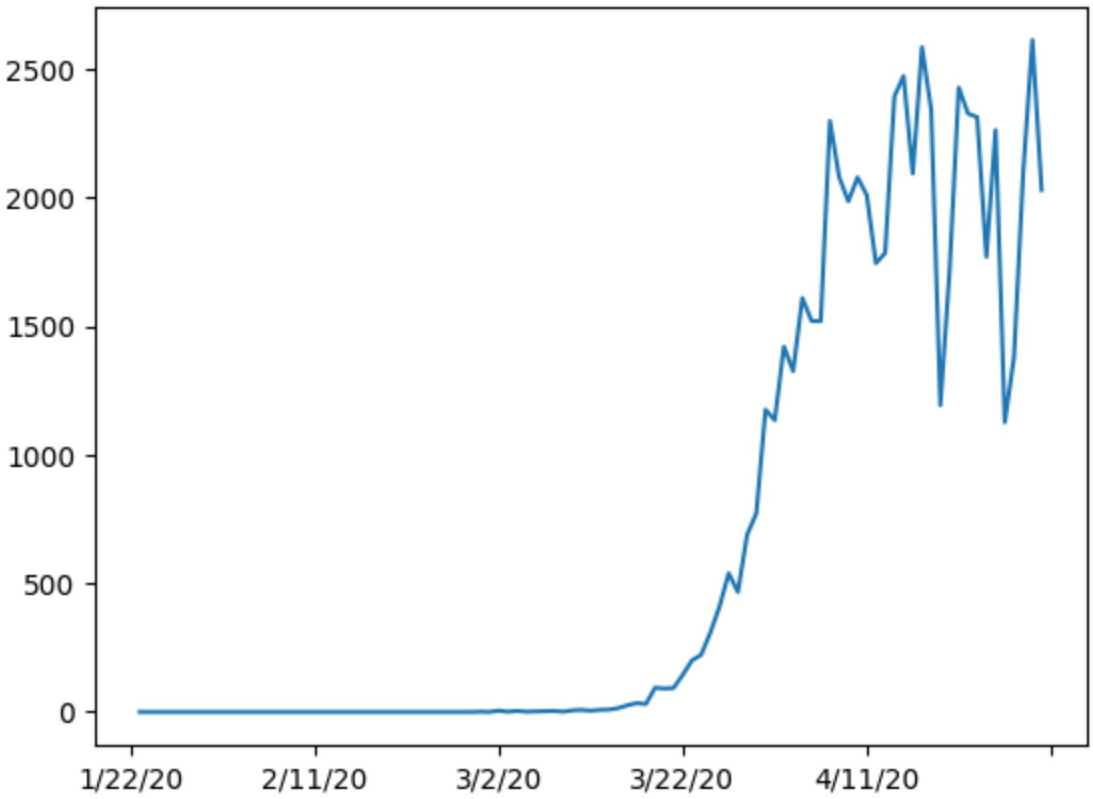
In [445...
```
covid19_deaths_aggregated.shape
```

Out[445...
```
(187, 101)
```

Task 6.3: Calculating the first derivative of the curve

In [446...
```
covid19_deaths_aggregated.loc['US'].drop('Province/State', errors='ignore').diff().
```

Out[446...
```
<Axes: >
```



Task 6.4: Find the maximum death rate for all countries

In [447…
```
max_death_rates = []
for c in countries:
    max_death_rates.append(
        covid19_deaths_aggregated.loc[c].drop('Province/State', errors='ignore').di
    )
covid19_deaths_aggregated['max_death_rate'] = max_death_rates
```

In [448…
```
covid19_deaths_aggregated.head(10)
```

Out[448...

Country/ Region	Province/ State	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20
Afghanistan	0	0	0	0	0	0	0	0
Albania	0	0	0	0	0	0	0	0
Algeria	0	0	0	0	0	0	0	0
Andorra	0	0	0	0	0	0	0	0
Angola	0	0	0	0	0	0	0	0
Antigua and Barbuda	0	0	0	0	0	0	0	0
Argentina	0	0	0	0	0	0	0	0
Armenia	0	0	0	0	0	0	0	0
Australia	Australian Capital TerritoryNew South WalesNor...	0	0	0	0	0	0	0
Austria	0	0	0	0	0	0	0	0

10 rows × 102 columns

Task 6.5: Create a new DataFrame with only the needed column

In [449...
```
covid19_deaths_dropped = pd.DataFrame(covid19_deaths_aggregated['max_death_rate'])
```

In [450...
```
covid19_deaths_dropped.head(10)
```

Out[450…

Country/Region	max_death_rate
Afghanistan	7
Albania	4
Algeria	30
Andorra	4
Angola	2
Antigua and Barbuda	1
Argentina	13
Armenia	3
Australia	8
Austria	30

In [451…
```
covid19_death_data = covid19_deaths_dropped
```

Task 7.1: Now let's join the three datasets we have prepared

Corona Dataset:

In [452…
```
corona_data.head(10)
```

Out[452…

Country/Region	max_infection_rate
Afghanistan	232
Albania	34
Algeria	199
Andorra	43
Angola	5
Antigua and Barbuda	6
Argentina	291
Armenia	134
Australia	497
Austria	1321

In [453...
```
corona_data.shape
```

Out[453... (187, 1)

World Happiness Report Dataset:

In [454...
```
happiness_report.head(10)
```

Out[454...

Country or region	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices
Finland	1.340	1.587	0.986	0.596
Denmark	1.383	1.573	0.996	0.592
Norway	1.488	1.582	1.028	0.603
Iceland	1.380	1.624	1.026	0.591
Netherlands	1.396	1.522	0.999	0.557
Switzerland	1.452	1.526	1.052	0.572
Sweden	1.387	1.487	1.009	0.574
New Zealand	1.303	1.557	1.026	0.585
Canada	1.365	1.505	1.039	0.584
Austria	1.376	1.475	1.016	0.532

In [455...
```
happiness_report.shape
```

Out[455... (156, 4)

Covid19 Deaths Dataset:

In [456...
```
covid19_death_data.head(10)
```

Out[456...

Country/Region	max_death_rate
Afghanistan	7
Albania	4
Algeria	30
Andorra	4
Angola	2
Antigua and Barbuda	1
Argentina	13
Armenia	3
Australia	8
Austria	30

In [457...
```
covid19_death_data.shape
```

Out[457... (187, 1)

In [458...
```
combined_data = combined_data.join(covid19_death_data, how='inner')
combined_data.head(50)
```

Out[458...

	max_infection_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	max_death_rate
Afghanistan	232	0.350	0.517	0.361	0.000	7
Albania	34	0.947	0.848	0.874	0.383	4
Algeria	199	1.002	1.160	0.785	0.086	30
Argentina	291	1.092	1.432	0.881	0.471	13
Armenia	134	0.850	1.055	0.815	0.283	3
Australia	497	1.372	1.548	1.036	0.557	8
Austria	1321	1.376	1.475	1.016	0.532	30
Azerbaijan	105	1.043	1.147	0.769	0.351	3
Bahrain	301	1.362	1.368	0.871	0.536	1
Bangladesh	641	0.562	0.928	0.723	0.527	15
Belarus	1485	1.067	1.465	0.789	0.235	5
Belgium	2454	1.356	1.504	0.986	0.473	496
Benin	19	0.393	0.437	0.397	0.349	1
Bhutan	1	0.813	1.321	0.604	0.457	0
Bolivia	104	0.776	1.209	0.706	0.511	6
Bosnia and Herzegovina	92	0.945	1.212	0.845	0.212	6
Botswana	7	1.041	1.145	0.538	0.455	1
Brazil	7502	1.004	1.439	0.802	0.390	493
Bulgaria	137	1.092	1.513	0.815	0.311	6
Burkina Faso	41	0.331	1.056	0.380	0.255	4
Burundi	6	0.046	0.447	0.380	0.220	1
Cambodia	31	0.574	1.122	0.637	0.609	0
Cameroon	203	0.549	0.910	0.331	0.381	20
Canada	2778	1.365	1.505	1.039	0.584	251
Central African Republic	31	0.026	0.000	0.105	0.225	0
Chad	21	0.350	0.766	0.192	0.174	3

	max_infection_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	max_death_rate
Chile	1138	1.159	1.369	0.920	0.357	13
China	15136	1.029	1.125	0.893	0.521	1290
Colombia	353	0.985	1.410	0.841	0.470	26
Comoros	1	0.274	0.757	0.505	0.142	0
Congo (Brazzaville)	57	0.673	0.799	0.508	0.372	3
Congo (Kinshasa)	81	0.094	1.125	0.357	0.269	5
Costa Rica	37	1.034	1.441	0.963	0.558	1
Croatia	96	1.155	1.266	0.914	0.296	8
Cyprus	58	1.263	1.223	1.042	0.406	2
Denmark	391	1.383	1.573	0.996	0.592	22
Dominican Republic	371	1.015	1.401	0.779	0.497	38
Ecuador	11536	0.912	1.312	0.868	0.498	208
Egypt	269	0.913	1.039	0.644	0.241	22
El Salvador	32	0.794	1.242	0.789	0.430	1
Estonia	134	1.237	1.528	0.874	0.495	6
Ethiopia	9	0.336	1.033	0.532	0.344	2
Finland	267	1.340	1.587	0.986	0.596	43
France	26849	1.324	1.472	1.045	0.436	1440
Gabon	38	1.057	1.183	0.571	0.295	1
Gambia	5	0.308	0.939	0.428	0.382	1
Georgia	42	0.886	0.666	0.752	0.346	1
Germany	6933	1.373	1.454	0.987	0.495	510
Ghana	403	0.611	0.868	0.486	0.381	5
Greece	156	1.181	1.156	0.999	0.067	10

Task 8: Visualization of the results

Our analysis is not complete unless we visualize the results with figures and graphs so that

everyone can understand the insights from our analysis.

In [459…
```
combined_data.corr().style.background_gradient(cmap='coolwarm')
```

Out[459…

	max_infection_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	max_deat
max_infection_rate	1.000000	0.250118	0.191958	0.289263	0.078196	0.8
GDP per capita	0.250118	1.000000	0.759468	0.863062	0.394603	0.2
Social support	0.191958	0.759468	1.000000	0.765286	0.456246	0.2
Healthy life expectancy	0.289263	0.863062	0.765286	1.000000	0.427892	0.3
Freedom to make life choices	0.078196	0.394603	0.456246	0.427892	1.000000	0.0
max_death_rate	0.880597	0.259893	0.204148	0.309666	0.080166	1.0

Task 8.1: Plotting GDP per capita vs. maximum death rate

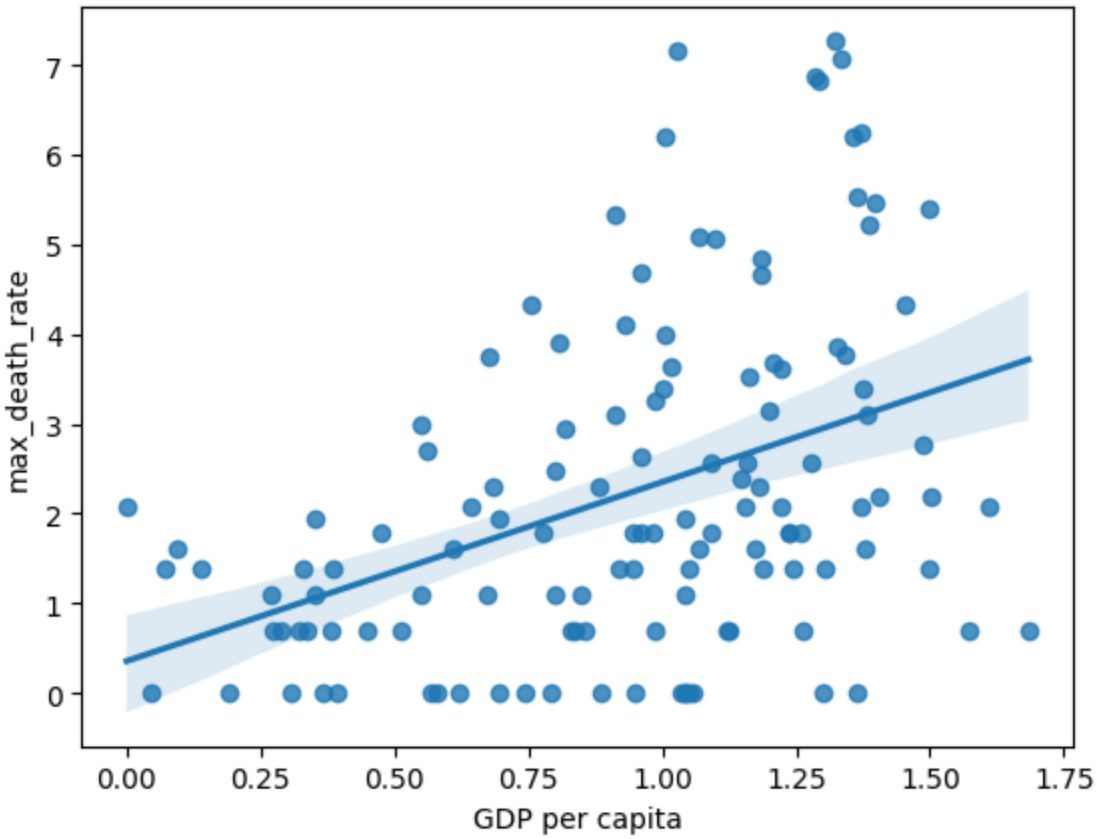
In [460…
```
combined_data.head(10)
```

Out[460…

	max_infection_rate	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	max_death_rate
Afghanistan	232	0.350	0.517	0.361	0.000	7
Albania	34	0.947	0.848	0.874	0.383	4
Algeria	199	1.002	1.160	0.785	0.086	30
Argentina	291	1.092	1.432	0.881	0.471	13
Armenia	134	0.850	1.055	0.815	0.283	3
Australia	497	1.372	1.548	1.036	0.557	8
Austria	1321	1.376	1.475	1.016	0.532	30
Azerbaijan	105	1.043	1.147	0.769	0.351	3
Bahrain	301	1.362	1.368	0.871	0.536	1
Bangladesh	641	0.562	0.928	0.723	0.527	15

In [464…
```
mask = y > 0
sns.regplot(x=x[mask], y=np.log(y[mask]))
```

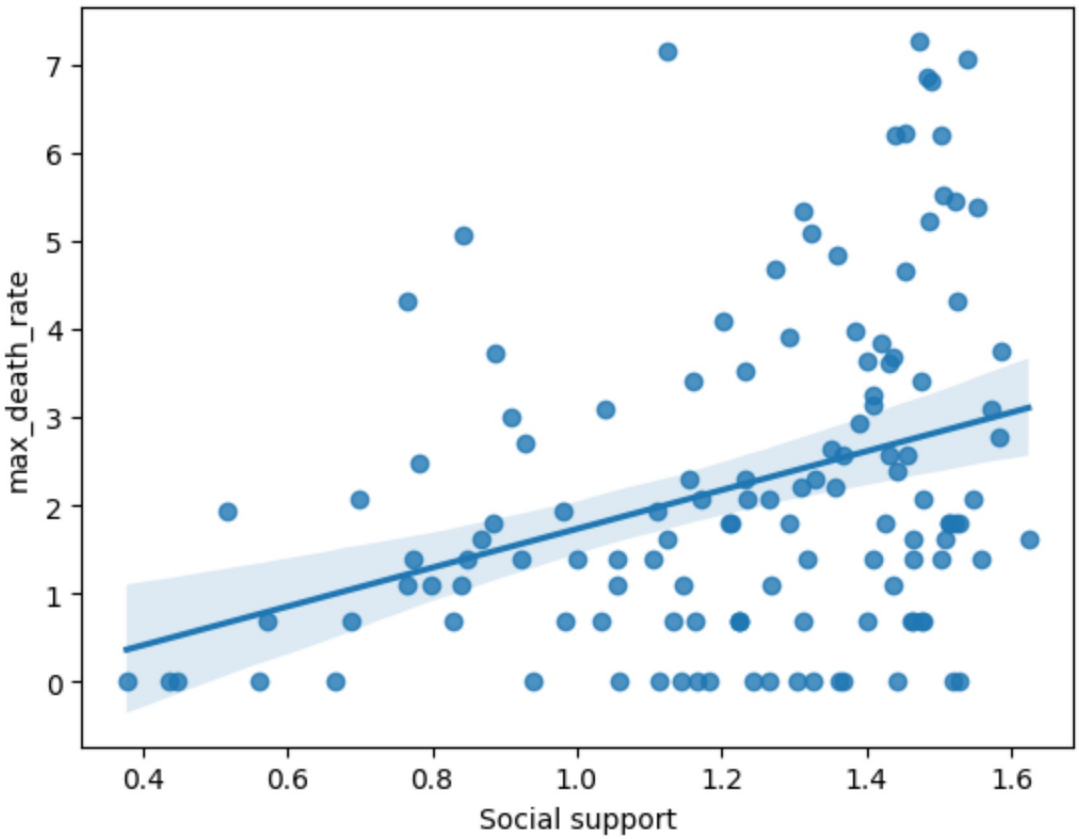
Out[464… <Axes: xlabel='GDP per capita', ylabel='max_death_rate'>



Task 8.2: Plotting Social Support vs. Maximum Death Rate

In [466…
```
x = combined_data['Social support']
y = combined_data['max_death_rate']
sns.regplot(x=x[mask], y=np.log(y[mask]))
```

Out[466… <Axes: xlabel='Social support', ylabel='max_death_rate'>



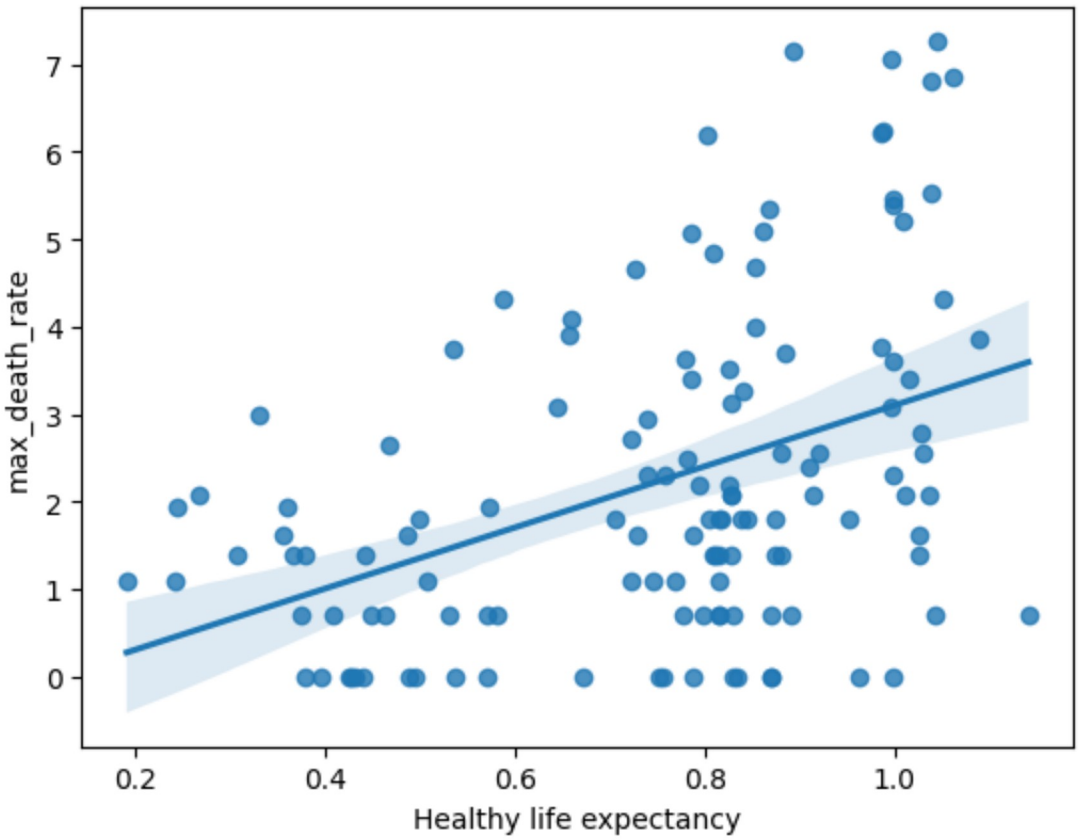
Task 8.3: Plotting Healthy Life Expectancy vs. Maximum Death Rate

In [467…

```
x = combined_data['Healthy life expectancy']
y = combined_data['max_death_rate']
sns.regplot(x=x[mask], y=np.log(y[mask]))
```

Out[467…

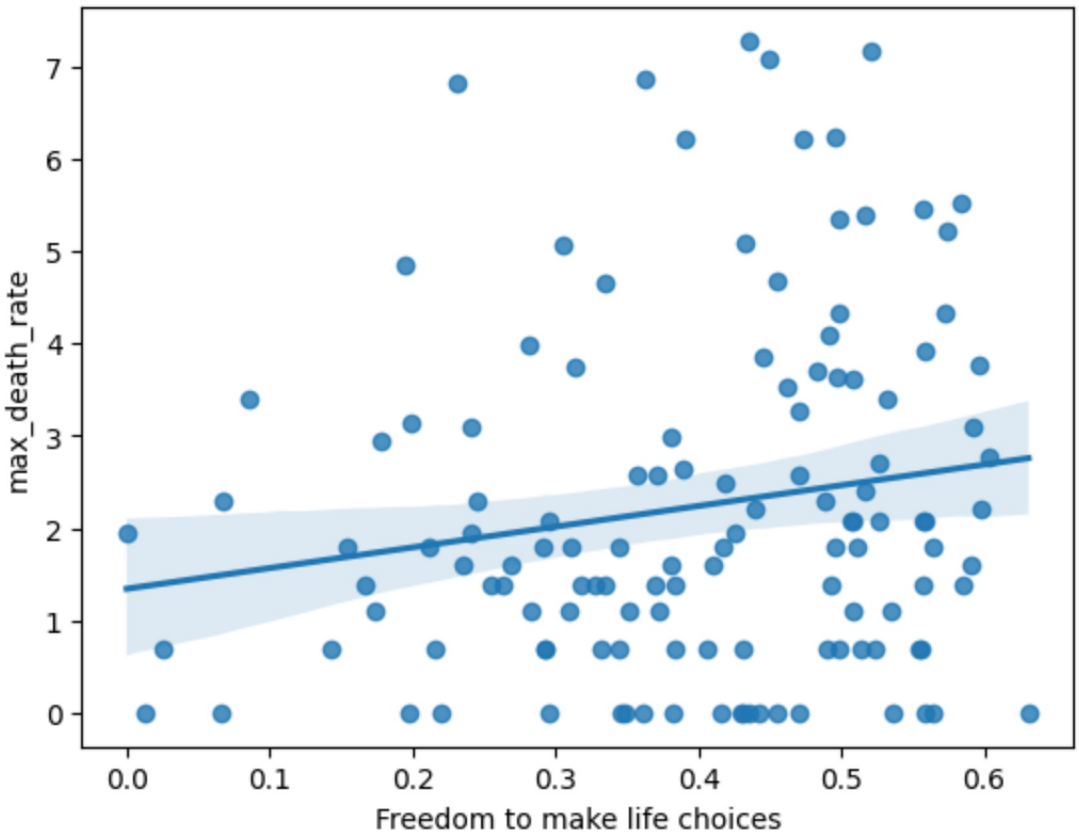
```
<Axes: xlabel='Healthy life expectancy', ylabel='max_death_rate'>
```



Task 8.4: Plotting Freedom to Make Life Choices vs. Maximum Death Rate

In [469…
```
x = combined_data['Freedom to make life choices']
y = combined_data['max_death_rate']
sns.regplot(x=x[mask], y=np.log(y[mask]))
```

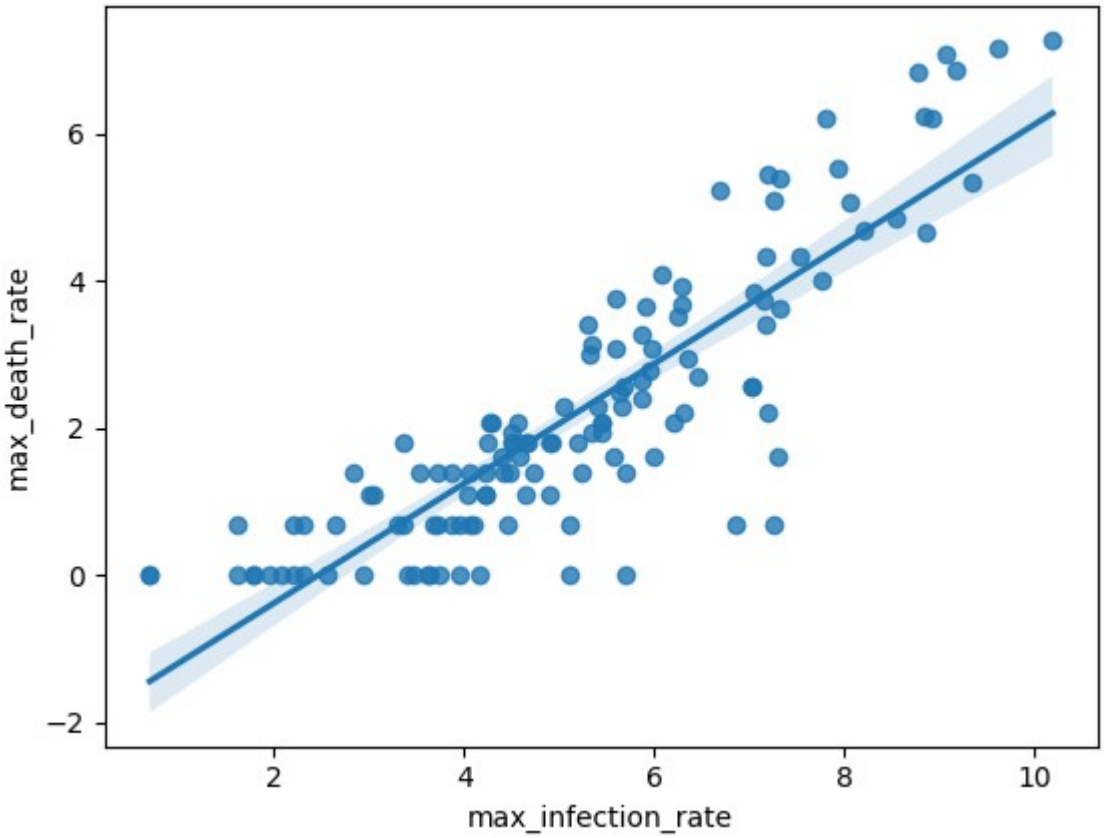
Out[469… `<Axes: xlabel='Freedom to make life choices', ylabel='max_death_rate'>`



Task 8.5: Plotting Maximum Infection Rate vs. Maximum Death Rate

In [475…
```
x = combined_data['max_infection_rate']
y = combined_data['max_death_rate']
sns.regplot(x=np.log(x[mask]), y=np.log(y[mask]))
```

Out[475…
```
<Axes: xlabel='max_infection_rate', ylabel='max_death_rate'>
```


max_death_rate
6
4
2
0
−2
2
4
6
8
10
max_infection_rate