Name:manmay mahurtale
class:SY A
batch:A1
Roll no:231035
Prn no:22010225

Name:utkarsh kamble
class:SY A
batch :A2
roll no:231030
prn no:22010651

Report for mini project

Topic:analyses of covid-19 data

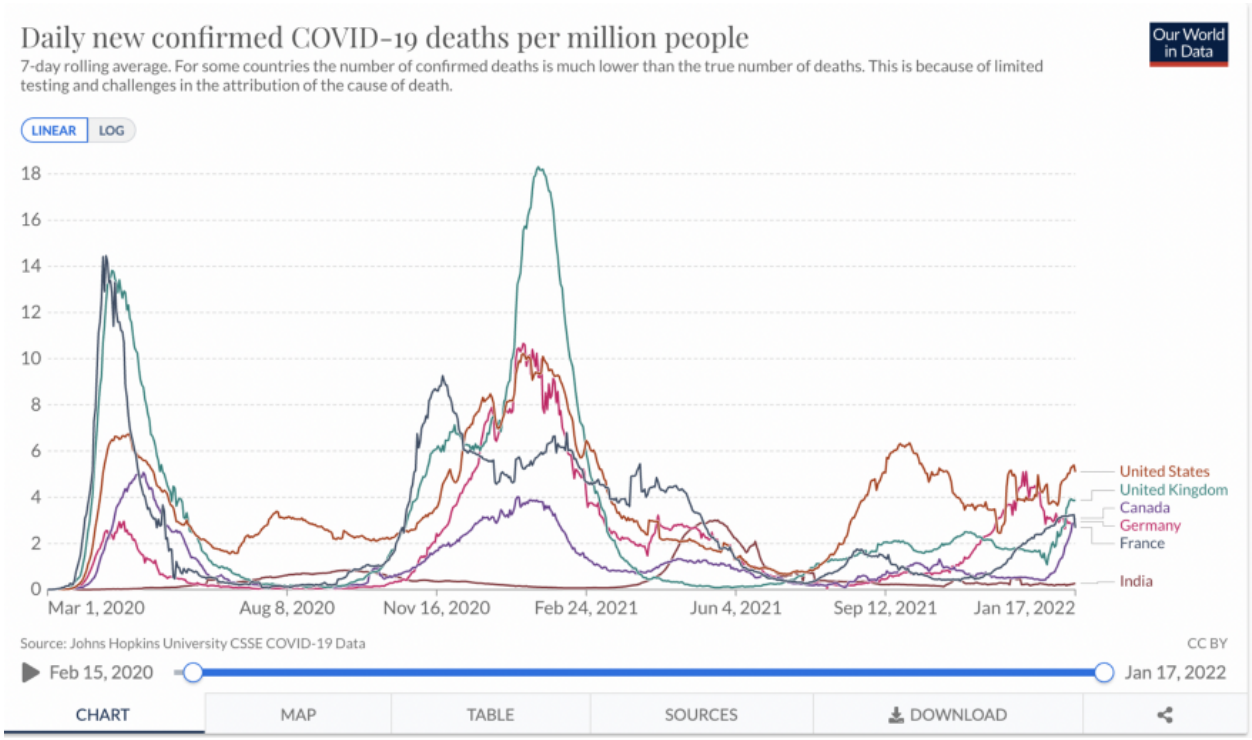
Introduction:

what we had done in this project is to collect the raw data and analyze it to make some conclusions which will help authorities to formulates plans and schemes.Remember, the dataset is not perfect but we can improve it by analyzing and processing it to get our answers.

Data set used:

https://ourworldindata.org/covid-deaths we had taken data from this website in the form of csv file.

implementation:



Import a file to Goggle Colab

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

Choose Files No file chosen Cancel upload

Photo by Author via Google Colab

Write this command in your Goggle Colab and you will get an option to upload a file. Click on Choose Files and browse your CSV dataset file and upload it.

We will import libraries:

```
import matplotlib.pyplot as plt
```

```
import pandas as pd
```

```
import numpy as np
```

```
from sklearn.impute import SimpleImputer
```

Read the data:

```
df = pd.read_csv('owid-covid-data.csv')
```

`owid-covid-data.csv` is the name of our dataset that we uploaded in Google Colab.

View the Data:

`df.shape`: will show you several rows and columns in the dataset

```
df.shape
```

```
(155139, 67)
```

`df`: will show you the whole dataset

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoothed
0	AFG	Asia	Afghanistan	2020-02-24	5.0	5.0	NaN	NaN	NaN	NaN
1	AFG	Asia	Afghanistan	2020-02-25	5.0	0.0	NaN	NaN	NaN	NaN
2	AFG	Asia	Afghanistan	2020-02-26	5.0	0.0	NaN	NaN	NaN	NaN
3	AFG	Asia	Afghanistan	2020-02-27	5.0	0.0	NaN	NaN	NaN	NaN
4	AFG	Asia	Afghanistan	2020-02-28	5.0	0.0	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...
155134	ZWE	Africa	Zimbabwe	2022-01-12	224433.0	1433.0	768.000	5215.0	35.0	17.571
155135	ZWE	Africa	Zimbabwe	2022-01-13	225084.0	651.0	700.857	5222.0	7.0	16.286
155136	ZWE	Africa	Zimbabwe	2022-01-14	225637.0	553.0	622.143	5238.0	16.0	14.571
155137	ZWE	Africa	Zimbabwe	2022-01-15	225637.0	0.0	531.286	5238.0	0.0	12.857
155138	ZWE	Africa	Zimbabwe	2022-01-16	226078.0	441.0	594.286	5247.0	9.0	14.143

155139 rows × 67 columns

df: The image shows only a part of the dataset but overall there are 155139 rows and 67 columns

`df.head()`: It will show the first 5 rows as default. If you want to return the first 10 rows then insert a value in parenthesis — `df.head(10)`

```
df.head()
```

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoothed
0	AFG	Asia	Afghanistan	2020-02-24	5.0	5.0	NaN	NaN	NaN	NaN
1	AFG	Asia	Afghanistan	2020-02-25	5.0	0.0	NaN	NaN	NaN	NaN
2	AFG	Asia	Afghanistan	2020-02-26	5.0	0.0	NaN	NaN	NaN	NaN
3	AFG	Asia	Afghanistan	2020-02-27	5.0	0.0	NaN	NaN	NaN	NaN
4	AFG	Asia	Afghanistan	2020-02-28	5.0	0.0	NaN	NaN	NaN	NaN

df.head()

`df.describe()`: It will show the basic summary of data such as count, min, max, unique values, etc.

```
df.describe(include="all")
```

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths
count	155139	145807	155139	155139	1.523460e+05	1.522390e+05	1.510880e+05	1.349990e+05	135153.000000
unique	238	6	238	747	NaN	NaN	NaN	NaN	NaN
top	PER	Africa	Mexico	2021-08-27	NaN	NaN	NaN	NaN	NaN
freq	747	36777	747	237	NaN	NaN	NaN	NaN	NaN
mean	NaN	NaN	NaN	NaN	2.185169e+06	9.063814e+03	8.895071e+03	5.354964e+04	170.072851
std	NaN	NaN	NaN	NaN	1.296057e+07	5.943441e+04	5.568795e+04	2.813624e+05	829.028606
min	NaN	NaN	NaN	NaN	1.000000e+00	-7.434700e+04	-6.223000e+03	1.000000e+00	-1918.000000
25%	NaN	NaN	NaN	NaN	1.629000e+03	1.000000e+00	6.000000e+00	6.900000e+01	0.000000
50%	NaN	NaN	NaN	NaN	2.147650e+04	7.200000e+01	9.528600e+01	6.940000e+02	2.000000
75%	NaN	NaN	NaN	NaN	2.619462e+05	9.470000e+02	9.974645e+02	6.583500e+03	19.000000
max	NaN	NaN	NaN	NaN	3.280769e+08	3.701643e+06	2.946282e+06	5.539572e+06	18062.000000

Dropping the column:

`df.drop()`: It drops the specific column or row of a dataset. We will look at how to drop a column

```
Syntax: df.drop('column_name', axis=1,inplace=True)
```

```
df.drop(['new_cases_smoothed'],axis=1,inplace=True)
```

```
#For multiple deletion of column
```

```
df.drop(['new_deaths_smoothed','new_cases_per_million','total_cases_per_million'],axis=1,inplace=True)
```

axis =1 specifies column

axis=0 specifies row

inplace=True: It specifies that drop should happen in the same data frame.

After dropping columns our dataset has now 63 columns instead of 67.

After dropping 4 columns dataset got reduced

Renaming the column name:

`df.rename()`: We can rename column name, row name, or index. In our dataset we will rename the columns:

```
df.rename(columns={'date':
'Date','location':'Country','continent':
'Continent','iso_code':'ISO_code'},inplace=True)
```

```
df
```

	ISO_code	Continent	Country	Date	total_cases	new_cases	total_deaths
0	AFG	Asia	Afghanistan	2020-02-24	5.0	5.0	NaN
1	AFG	Asia	Afghanistan	2020-02-25	5.0	0.0	NaN
2	AFG	Asia	Afghanistan	2020-02-26	5.0	0.0	NaN
3	AFG	Asia	Afghanistan	2020-02-27	5.0	0.0	NaN
4	AFG	Asia	Afghanistan	2020-02-28	5.0	0.0	NaN
...	...	...	...	...	...	...	...

Image after renaming the column

List the continent name:

```
continent_unique = list(df.Continent.unique())
```

```
continent_unique
```

```
['Asia', nan, 'Europe', 'Africa', 'North America', 'South America', 'Oceania']
```

Using simple imputer:

Simple imputer helps with missing values in a dataset. In the below code, a simple imputer will replace a missing value with a constant value.

```
imputer = SimpleImputer(strategy='constant')
```

```
df2 = pd.DataFrame(imputer.fit_transform(df),columns=df.columns)
```

Form a subset of data using Group by:

`df.groupby()`: The groupby command allows us to divide our data into different groups and perform our data analysis.

```
df3 =
df2.groupby(['Date','Country',])[['Date','Country','total_cases'
,'total_deaths','total_vaccinations']].sum().reset_index()
```

df3

	Date	Country	total_cases	total_deaths	total_vaccinations
0	2020-01-01	Argentina	missing_value	missing_value	missing_value
1	2020-01-01	Mexico	missing_value	missing_value	missing_value
2	2020-01-01	Peru	missing_value	missing_value	missing_value
3	2020-01-02	Argentina	missing_value	missing_value	missing_value
4	2020-01-02	Mexico	missing_value	missing_value	missing_value
...	...	...	...	...	...
155134	2022-01-16	Wallis and Futuna	454	7	missing_value
155135	2022-01-16	World	3.28077e+08	5.53957e+06	9.67623e+09
155136	2022-01-16	Yemen	10252	1990	missing_value
155137	2022-01-16	Zambia	296132	3860	missing_value
155138	2022-01-16	Zimbabwe	226078	5247	7.43848e+06

155139 rows × 5 columns

The output of the df.groupby()

You can see there are missing_value because we have used simple imputer to add constant value at all missing places. If you want you can even replace that missing_value with 0. The command below is to change missing_value to 0 in the total_cases column:

```
df3['total_cases'].replace({'missing_value': 0}, inplace=True)
```

```
df3
```

	Date	Country	total_cases	total_deaths	total_vaccinations
0	2020-01-01	Argentina	0.0	missing_value	missing_value
1	2020-01-01	Mexico	0.0	missing_value	missing_value
2	2020-01-01	Peru	0.0	missing_value	missing_value
3	2020-01-02	Argentina	0.0	missing_value	missing_value
4	2020-01-02	Mexico	0.0	missing_value	missing_value
...	...	...	...	...	...
155134	2022-01-16	Wallis and Futuna	454.0	7	missing_value
155135	2022-01-16	World	328076885.0	5.53957e+06	9.67623e+09
155136	2022-01-16	Yemen	10252.0	1990	missing_value
155137	2022-01-16	Zambia	296132.0	3860	missing_value
155138	2022-01-16	Zimbabwe	226078.0	5247	7.43848e+06

155139 rows × 5 columns

Similarly do it for, total_deaths and total_vaccinations columns.

```
df3['total_deaths'].replace({'missing_value':0},inplace=True)
```

```
df3['total_vaccinations'].replace({'missing_value':0},inplace=True)
```

```
df3
```

	Date	Country	total_cases	total_deaths	total_vaccinations
0	2020-01-01	Argentina	0.0	0.0	0.000000e+00
1	2020-01-01	Mexico	0.0	0.0	0.000000e+00
2	2020-01-01	Peru	0.0	0.0	0.000000e+00
3	2020-01-02	Argentina	0.0	0.0	0.000000e+00
4	2020-01-02	Mexico	0.0	0.0	0.000000e+00
...	...	...	...	...	...
155134	2022-01-16	Wallis and Futuna	454.0	7.0	0.000000e+00
155135	2022-01-16	World	328076885.0	5539572.0	9.676232e+09
155136	2022-01-16	Yemen	10252.0	1990.0	0.000000e+00
155137	2022-01-16	Zambia	296132.0	3860.0	0.000000e+00
155138	2022-01-16	Zimbabwe	226078.0	5247.0	7.438485e+06

155139 rows × 5 columns

Will plot subset of specific data:

We will find total countries where total_deaths is greater than 1000000.

```
df4 = df3[df3['total_deaths']>1000000]
```

```
countries = df4['Country'].unique()
len(countries)
```

```
8
```

There are 8 countries where total_deaths is greater than 1000000. We will find out which are those countries in a table.

```
country_deaths_greaterthan1000000 = list(df4.Country.unique())
```

```
country_deaths_greaterthan1000000
```

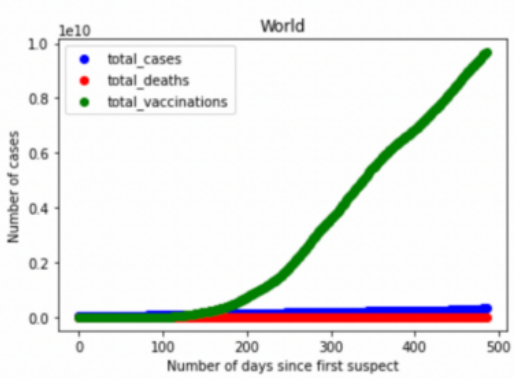
```
['World',
 'High income',
 'Upper middle income',
 'Europe',
 'South America',
 'Asia',
 'Lower middle income',
 'North America']
```

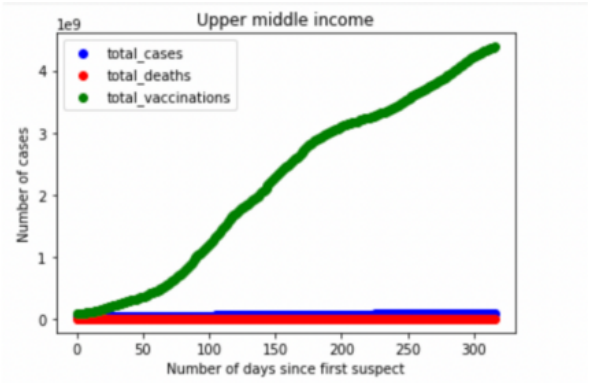
We will show the trend of `total_cases, total_deaths, total_vaccinations` of these 8 countries using a scatter plot. We will use a for loop to get all those 8 countries' graphs.

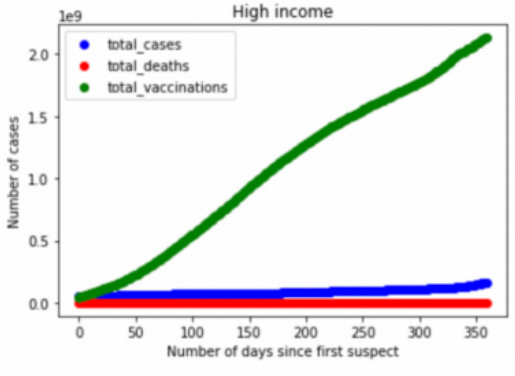
+ Code + Text

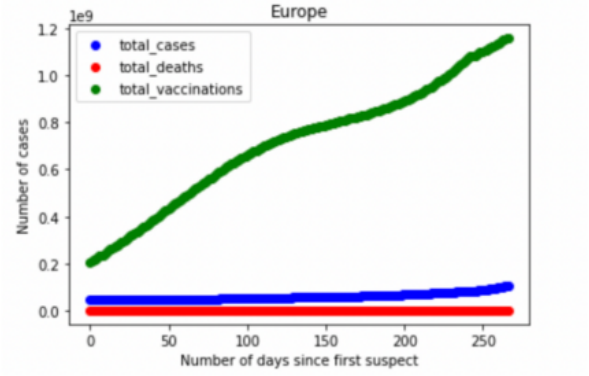
```
for idx in range(0,len(countries)):
  C = df4[df4['Country']==countries[idx]].reset_index()
  plt.scatter(np.arange(0,len(C)),C['total_cases'],color="blue",label="total_cases")
  plt.scatter(np.arange(0,len(C)),C['total_deaths'],color="red",label="total_deaths")
  plt.scatter(np.arange(0,len(C)),C['total_vaccinations'],color="green",label="total_vaccinations")
  plt.title(countries[idx])
  plt.xlabel("Number of days since first suspect")
  plt.ylabel("Number of cases")
  plt.legend()
  plt.show()
```

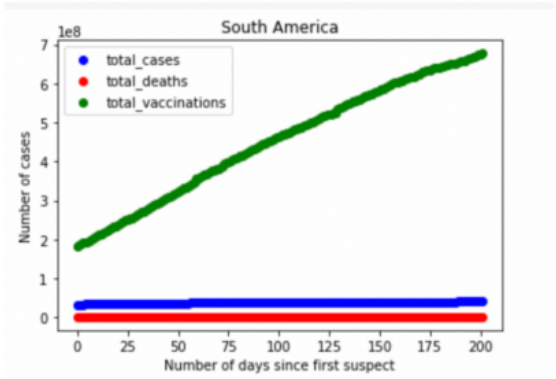
The code for showing trends

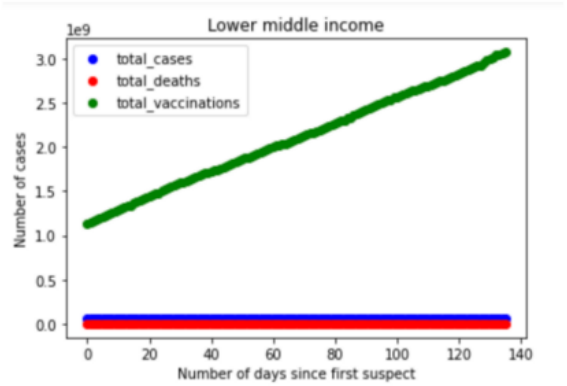

World
1e10
total_cases
total_deaths
total_vaccinations
Number of cases
Number of days since first suspect

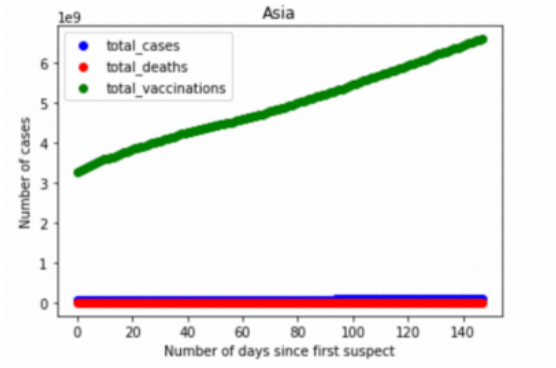

Upper middle income
1e9
total_cases
total_deaths
total_vaccinations
Number of cases
Number of days since first suspect

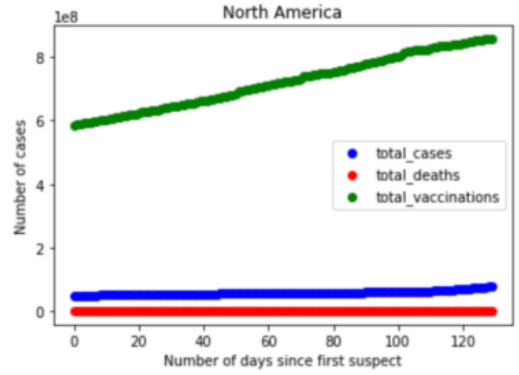

High income
1e9
total_cases
total_deaths
total_vaccinations
Number of cases
Number of days since first suspect


Europe
1e9
total_cases
total_deaths
total_vaccinations
Number of cases
Number of days since first suspect


South America
1e8
total_cases
total_deaths
total_vaccinations
Number of cases
Number of days since first suspect


Lower middle income
1e9
total_cases
total_deaths
total_vaccinations
Number of cases
Number of days since first suspect


Asia
1e9
total_cases
total_deaths
total_vaccinations
Number of cases
Number of days since first suspect


North America
1e8
total_cases
total_deaths
total_vaccinations
Number of cases
Number of days since first suspect

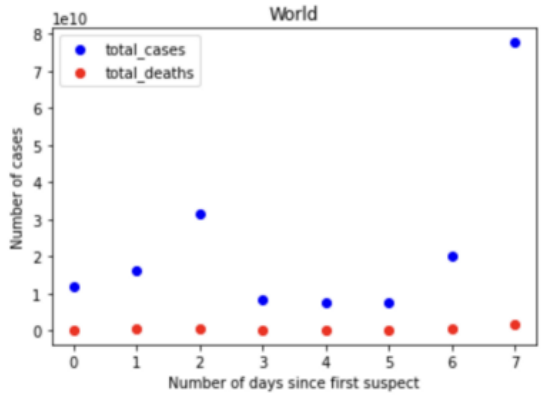
Output

You will see the above 8 graph trends in your Goggle Colab. I have merged those and taken a picture of them.

We will do another analysis where we will group all countries.

```
df5 = df4.groupby(['Country'])[['Country','total_cases','total_deaths']].sum().reset_index()
```

```
C = df5
plt.scatter(np.arange(0,len(C)),C['total_cases'],color="blue",label="total_cases")
plt.scatter(np.arange(0,len(C)),C['total_deaths'],color="red",label="total_deaths")
plt.title("World")
plt.xlabel("Number of days since first suspect")
plt.ylabel("Number of cases")
plt.legend()
plt.show()
```



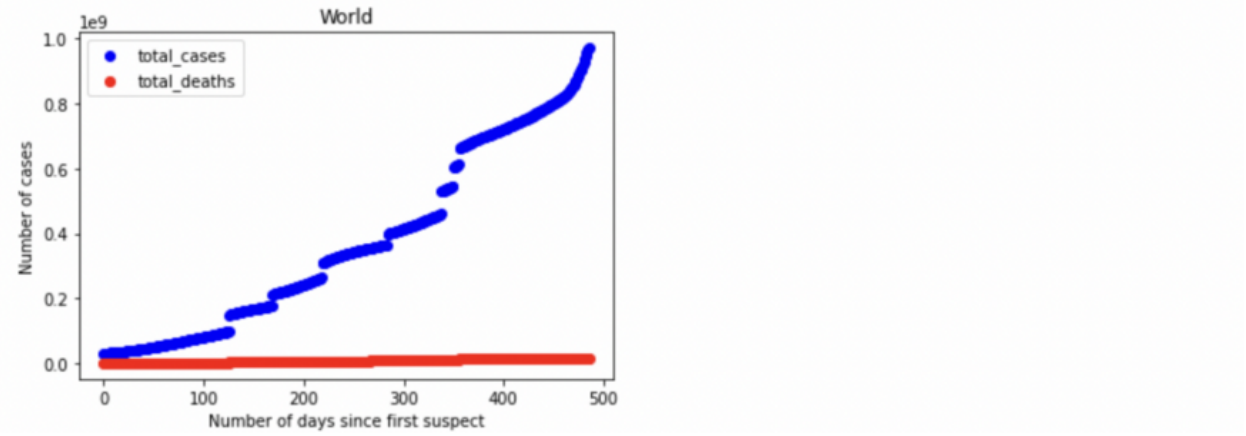
We will do another analysis by date. First, we will look at how many dates we have in total where total_deaths is greater than 1000000

```
date = df4['Date'].unique()
len(date)
```

```
486
```

```
df6 = df4.groupby(['Date'])[['Date','total_cases','total_deaths']].sum().reset_index()
```

```
C = df6
plt.scatter(np.arange(0,len(C)),C['total_cases'],color="blue",label="total_cases")
plt.scatter(np.arange(0,len(C)),C['total_deaths'],color="red",label="total_deaths")
plt.title("World")
plt.xlabel("Number of days since first suspect")
plt.ylabel("Number of cases")
plt.legend()
plt.show()
```



Conclusion: This way you can do data analysis of the dataset available with you. I have shown just the basics, you can explore it and have fun exploring different columns of the dataset.

Future scope:we can make use of probability and statistics to make some more conclusions .

References: used https://ourworldindata.org/covid-deaths this url for data set.Along with this we had taken help from the documents of the libraries involved.