Importing Necessary Modules

In [1]:
```
import numpy as np
import pandas as pd
import pprint
import warnings
warnings.filterwarnings('ignore')
```

A. Loading The Dataset

In [2]:
```
data = pd.read_csv("https://covid.ourworldindata.org/data/owid-covid-data.csv")
data.head()
```

Out[2]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoothed	...	male_smok
0	AFG	Asia	Afghanistan	2020-01-05	0.0	0.0	NaN	0.0	0.0	NaN	...	N
1	AFG	Asia	Afghanistan	2020-01-06	0.0	0.0	NaN	0.0	0.0	NaN	...	N
2	AFG	Asia	Afghanistan	2020-01-07	0.0	0.0	NaN	0.0	0.0	NaN	...	N
3	AFG	Asia	Afghanistan	2020-01-08	0.0	0.0	NaN	0.0	0.0	NaN	...	N
4	AFG	Asia	Afghanistan	2020-01-09	0.0	0.0	NaN	0.0	0.0	NaN	...	N

5 rows × 67 columns

Information about dataset

The variables represent all of our main data related to confirmed cases, deaths, hospitalizations, and testing, as well as other variables of potential interest.

Confirmed cases

Variable	Description
`total_cases`	Total confirmed cases of COVID-19. Counts can include probable cases, where reported.
`new_cases`	New confirmed cases of COVID-19. Counts can include probable cases, where reported. In rare cases where our source reports a negative daily change due to a data correction, we set this metric to NA.
`new_cases_smoothed`	New confirmed cases of COVID-19 (7-day smoothed). Counts can include probable cases, where reported.
`total_cases_per_million`	Total confirmed cases of COVID-19 per 1,000,000 people. Counts can include probable cases, where reported.
`new_cases_per_million`	New confirmed cases of COVID-19 per 1,000,000 people. Counts can include probable cases, where reported.
`new_cases_smoothed_per_million`	New confirmed cases of COVID-19 (7-day smoothed) per 1,000,000 people. Counts can include probable cases, where reported.

Confirmed deaths

Variable	Description
`total_deaths`	Total deaths attributed to COVID-19. Counts can include probable deaths, where reported.
`new_deaths`	New deaths attributed to COVID-19. Counts can include probable deaths, where reported. In rare cases where our source reports a negative daily change due to a data correction, we set this metric to NA.
`new_deaths_smoothed`	New deaths attributed to COVID-19 (7-day smoothed). Counts can include probable deaths, where reported.
`total_deaths_per_million`	Total deaths attributed to COVID-19 per 1,000,000 people. Counts can include probable deaths, where reported.
`new_deaths_per_million`	New deaths attributed to COVID-19 per 1,000,000 people. Counts can include probable deaths, where reported.
`new_deaths_smoothed_per_million`	New deaths attributed to COVID-19 (7-day smoothed) per 1,000,000 people. Counts can include probable deaths, where reported.

Notes:

- Due to varying protocols and challenges in the attribution of the cause of death, the number of confirmed deaths may not accurately represent the true number of deaths caused by COVID-19.

Excess mortality

Variable	Description
excess_mortality	Percentage difference between the reported number of weekly or monthly deaths in 2020–2021 and the projected number of deaths for the same period based on previous years. For more information, see https://github.com/owid/covid-19-data/tree/master/public/data/excess_mortality
excess_mortality_cumulative	Percentage difference between the cumulative number of deaths since 1 January 2020 and the cumulative projected deaths for the same period based on previous years. For more information, see https://github.com/owid/covid-19-data/tree/master/public/data/excess_mortality
excess_mortality_cumulative_absolute	Cumulative difference between the reported number of deaths since 1 January 2020 and the projected number of deaths for the same period based on previous years. For more information, see https://github.com/owid/covid-19-data/tree/master/public/data/excess_mortality
excess_mortality_cumulative_per_million	Cumulative difference between the reported number of deaths since 1 January 2020 and the projected number of deaths for the same period based on previous years, per million people. For more information, see https://github.com/owid/covid-19-data/tree/master/public/data/excess_mortality

Hospital & ICU

Variable	Description
icu_patients	Number of COVID-19 patients in intensive care units (ICUs) on a given day
icu_patients_per_million	Number of COVID-19 patients in intensive care units (ICUs) on a given day per 1,000,000 people
hosp_patients	Number of COVID-19 patients in hospital on a given day
hosp_patients_per_million	Number of COVID-19 patients in hospital on a given day per 1,000,000 people
weekly_icu_admissions	Number of COVID-19 patients newly admitted to intensive care units (ICUs) in a given week (reporting date and the preceeding 6 days)
weekly_icu_admissions_per_million	Number of COVID-19 patients newly admitted to intensive care units (ICUs) in a given week per 1,000,000 people (reporting date and the preceeding 6 days)
weekly_hosp_admissions	Number of COVID-19 patients newly admitted to hospitals in a given week (reporting date and the preceeding 6 days)
weekly_hosp_admissions_per_million	Number of COVID-19 patients newly admitted to hospitals in a given week per 1,000,000 people (reporting date and the preceeding 6 days)

Policy responses

Variable	Description
stringency_index	Government Response Stringency Index: composite measure based on 9 response indicators including school closures, workplace closures, and travel bans, rescaled to a value from 0 to 100 (100 = strictest response)

Reproduction rate

Variable	Description
reproduction_rate	Real-time estimate of the effective reproduction rate (R) of COVID-19. See https://github.com/crondonm/TrackingR/tree/main/Estimates-Database

Tests & positivity

On 23 June 2022, we stopped adding new datapoints to our COVID-19 testing dataset. You can read more at https://github.com/owid/covid-19-data/discussions/2667.

Variable	Description
total_tests	Total tests for COVID-19
new_tests	New tests for COVID-19 (only calculated for consecutive days)
total_tests_per_thousand	Total tests for COVID-19 per 1,000 people
new_tests_per_thousand	New tests for COVID-19 per 1,000 people
new_tests_smoothed	New tests for COVID-19 (7-day smoothed). For countries that don't report testing data on a daily basis, we assume that testing changed equally on a daily basis over any periods in which no data was reported. This produces a complete series of daily figures, which is then averaged over a rolling 7-day window
new_tests_smoothed_per_thousand	New tests for COVID-19 (7-day smoothed) per 1,000 people
positive_rate	The share of COVID-19 tests that are positive, given as a rolling 7-day average (this is the inverse of tests_per_case)
tests_per_case	Tests conducted per new confirmed case of COVID-19, given as a rolling 7-day average (this is the inverse of positive_rate)
tests_units	Units used by the location to report its testing data. A country file can't contain mixed units. All metrics concerning testing data use the specified test unit. Valid units are 'people tested' (number of people tested), 'tests performed' (number of tests performed. a single person can be tested more than once in a given day) and 'samples tested' (number of samples tested. In some cases, more than one sample may be required to perform a given test.)

Vaccinations

Variable	Description
total_vaccinations	Total number of COVID-19 vaccination doses administered

Variable	Description
people_vaccinated	Total number of people who received at least one vaccine dose
people_fully_vaccinated	Total number of people who received all doses prescribed by the initial vaccination protocol
total_boosters	Total number of COVID-19 vaccination booster doses administered (doses administered beyond the number prescribed by the vaccination protocol)
new_vaccinations	New COVID-19 vaccination doses administered (only calculated for consecutive days)
new_vaccinations_smoothed	New COVID-19 vaccination doses administered (7-day smoothed). For countries that don't report vaccination data on a daily basis, we assume that vaccination changed equally on a daily basis over any periods in which no data was reported. This produces a complete series of daily figures, which is then averaged over a rolling 7-day window
total_vaccinations_per_hundred	Total number of COVID-19 vaccination doses administered per 100 people in the total population
people_vaccinated_per_hundred	Total number of people who received at least one vaccine dose per 100 people in the total population
people_fully_vaccinated_per_hundred	Total number of people who received all doses prescribed by the initial vaccination protocol per 100 people in the total population
total_boosters_per_hundred	Total number of COVID-19 vaccination booster doses administered per 100 people in the total population
new_vaccinations_smoothed_per_million	New COVID-19 vaccination doses administered (7-day smoothed) per 1,000,000 people in the total population
new_people_vaccinated_smoothed	Daily number of people receiving their first vaccine dose (7-day smoothed)
new_people_vaccinated_smoothed_per_hundred	Daily number of people receiving their first vaccine dose (7-day smoothed) per 100 people in the total population

Others

Variable	Description
iso_code	ISO 3166-1 alpha-3 – three-letter country codes. Note that OWID-defined regions (e.g. continents like 'Europe') contain prefix 'OWID_'.
continent	Continent of the geographical location
location	Geographical location
date	Date of observation
population	Population (latest available values). See https://github.com/owid/covid-19-data/blob/master/scripts/input/un/population_latest.csv for full list of sources
population_density	Number of people divided by land area, measured in square kilometers, most recent year available
median_age	Median age of the population, UN projection for 2020
aged_65_older	Share of the population that is 65 years and older, most recent year available
aged_70_older	Share of the population that is 70 years and older in 2015
gdp_per_capita	Gross domestic product at purchasing power parity (constant 2011 international dollars), most recent year available
extreme_poverty	Share of the population living in extreme poverty, most recent year available since 2010
cardiovasc_death_rate	Death rate from cardiovascular disease in 2017 (annual number of deaths per 100,000 people)
diabetes_prevalence	Diabetes prevalence (% of population aged 20 to 79) in 2017
female_smokers	Share of women who smoke, most recent year available
male_smokers	Share of men who smoke, most recent year available
handwashing_facilities	Share of the population with basic handwashing facilities on premises, most recent year available
hospital_beds_per_thousand	Hospital beds per 1,000 people, most recent year available since 2010
life_expectancy	Life expectancy at birth in 2019
human_development_index	A composite index measuring average achievement in three basic dimensions of human development—a long and healthy life, knowledge and a decent standard of living. Values for 2019, imported from http://hdr.undp.org/en/indicators/137506

B. Analyzing the Covid Data

i) overall meta data for the Covid data

In [3]:
```
# Column Type and datapoints summary
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 429435 entries, 0 to 429434
Data columns (total 67 columns):
 #   Column                                    Non-Null Count   Dtype
---  ------                                    --------------   -----
 0   iso_code                                  429435 non-null  object
 1   continent                                 402910 non-null  object
 2   location                                  429435 non-null  object
 3   date                                      429435 non-null  object
 4   total_cases                               411804 non-null  float64
 5   new_cases                                 410159 non-null  float64
 6   new_cases_smoothed                        408929 non-null  float64
 7   total_deaths                              411804 non-null  float64
 8   new_deaths                                410608 non-null  float64
 9   new_deaths_smoothed                       409378 non-null  float64
 10  total_cases_per_million                   411804 non-null  float64
 11  new_cases_per_million                     410159 non-null  float64
 12  new_cases_smoothed_per_million            408929 non-null  float64
 13  total_deaths_per_million                  411804 non-null  float64
 14  new_deaths_per_million                    410608 non-null  float64
 15  new_deaths_smoothed_per_million           409378 non-null  float64
 16  reproduction_rate                         184817 non-null  float64
 17  icu_patients                              39116 non-null   float64
 18  icu_patients_per_million                  39116 non-null   float64
 19  hosp_patients                             40656 non-null   float64
 20  hosp_patients_per_million                 40656 non-null   float64
 21  weekly_icu_admissions                     10993 non-null   float64
 22  weekly_icu_admissions_per_million         10993 non-null   float64
 23  weekly_hosp_admissions                    24497 non-null   float64
 24  weekly_hosp_admissions_per_million        24497 non-null   float64
 25  total_tests                               79387 non-null   float64
 26  new_tests                                 75403 non-null   float64
 27  total_tests_per_thousand                  79387 non-null   float64
 28  new_tests_per_thousand                    75403 non-null   float64
 29  new_tests_smoothed                        103965 non-null  float64
 30  new_tests_smoothed_per_thousand           103965 non-null  float64
 31  positive_rate                             95927 non-null   float64
 32  tests_per_case                            94348 non-null   float64
 33  tests_units                               106788 non-null  object
 34  total_vaccinations                        85417 non-null   float64
 35  people_vaccinated                         81132 non-null   float64
 36  people_fully_vaccinated                   78061 non-null   float64
 37  total_boosters                            53600 non-null   float64
 38  new_vaccinations                          70971 non-null   float64
 39  new_vaccinations_smoothed                 195029 non-null  float64
 40  total_vaccinations_per_hundred            85417 non-null   float64
 41  people_vaccinated_per_hundred             81132 non-null   float64
 42  people_fully_vaccinated_per_hundred       78061 non-null   float64
 43  total_boosters_per_hundred                53600 non-null   float64
 44  new_vaccinations_smoothed_per_million     195029 non-null  float64
 45  new_people_vaccinated_smoothed            192177 non-null  float64
 46  new_people_vaccinated_smoothed_per_hundred  192177 non-null  float64
 47  stringency_index                          196190 non-null  float64
 48  population_density                        360492 non-null  float64
 49  median_age                                334663 non-null  float64
 50  aged_65_older                             323270 non-null  float64
 51  aged_70_older                             331315 non-null  float64
 52  gdp_per_capita                            328292 non-null  float64
 53  extreme_poverty                           211996 non-null  float64
 54  cardiovasc_death_rate                     328865 non-null  float64
 55  diabetes_prevalence                       345911 non-null  float64
 56  female_smokers                            247165 non-null  float64
 57  male_smokers                              243817 non-null  float64
 58  handwashing_facilities                    161741 non-null  float64
 59  hospital_beds_per_thousand                290689 non-null  float64
 60  life_expectancy                           390299 non-null  float64
 61  human_development_index                   319127 non-null  float64
 62  population                                429435 non-null  int64
 63  excess_mortality_cumulative_absolute      13411 non-null   float64
 64  excess_mortality_cumulative               13411 non-null   float64
 65  excess_mortality                          13411 non-null   float64
 66  excess_mortality_cumulative_per_million   13411 non-null   float64
dtypes: float64(61), int64(1), object(5)
memory usage: 219.5+ MB
```

Observation

- Based on `data.info()` it is evident that there are some null values in the data which needs to be handled going forward.
- Based on `date` column datatype which is of type `object` needs to be converted to date-time going forward.

In [4]:
```python
# Converting date column into datetime

data['date'] = pd.to_datetime(data['date'])
```

In [5]:
```python
# Set Pandas options to prevent truncation

pd.set_option('display.max_columns', None)  # Show all columns
pd.set_option('display.max_rows', None)     # Show all rows
pd.set_option('display.max_colwidth', None) # No column width limit
```

In [6]:
```
# Data Summary for every column
data.describe(include = "all").T
```

Out[6]:

	count	unique	top	freq	mean	min	25%	50%	75%
iso_code	429435	255	OWID_HIC	3026	NaN	NaN	NaN	NaN	NaN
continent	402910	6	Africa	95419	NaN	NaN	NaN	NaN	NaN
location	429435	255	High-income countries	3026	NaN	NaN	NaN	NaN	NaN
date	429435	NaN	NaN	NaN	2022-04-21 01:06:25.463691008	2020-01-01 00:00:00	2021-03-05 00:00:00	2022-04-20 00:00:00	2023-06-08 00:00:00
total_cases	411804.0	NaN	NaN	NaN	7365292.354484	0.0	6280.75	63653.0	758272.0
new_cases	410159.0	NaN	NaN	NaN	8017.359934	0.0	0.0	0.0	0.0
new_cases_smoothed	408929.0	NaN	NaN	NaN	8041.025775	0.0	0.0	12.0	313.286
total_deaths	411804.0	NaN	NaN	NaN	81259.574278	0.0	43.0	799.0	9574.0
new_deaths	410608.0	NaN	NaN	NaN	71.852139	0.0	0.0	0.0	0.0
new_deaths_smoothed	409378.0	NaN	NaN	NaN	72.060873	0.0	0.0	0.0	3.143
total_cases_per_million	411804.0	NaN	NaN	NaN	112096.199396	0.0	1916.1005	29145.475	156770.19
new_cases_per_million	410159.0	NaN	NaN	NaN	122.357074	0.0	0.0	0.0	0.0
new_cases_smoothed_per_million	408929.0	NaN	NaN	NaN	122.713844	0.0	0.0	2.794	56.253
total_deaths_per_million	411804.0	NaN	NaN	NaN	835.514313	0.0	24.568	295.089	1283.817
new_deaths_per_million	410608.0	NaN	NaN	NaN	0.762323	0.0	0.0	0.0	0.0
new_deaths_smoothed_per_million	409378.0	NaN	NaN	NaN	0.764555	0.0	0.0	0.0	0.357
reproduction_rate	184817.0	NaN	NaN	NaN	0.911495	-0.07	0.72	0.95	1.14
icu_patients	39116.0	NaN	NaN	NaN	660.971418	0.0	21.0	90.0	413.0
icu_patients_per_million	39116.0	NaN	NaN	NaN	15.65634	0.0	2.328	6.434	18.77925
hosp_patients	40656.0	NaN	NaN	NaN	3911.741563	0.0	186.0	776.0	3051.0
hosp_patients_per_million	40656.0	NaN	NaN	NaN	125.988007	0.0	30.997	74.236	159.75825
weekly_icu_admissions	10993.0	NaN	NaN	NaN	317.894114	0.0	17.0	92.0	353.0
weekly_icu_admissions_per_million	10993.0	NaN	NaN	NaN	9.671944	0.0	1.549	4.645	12.651
weekly_hosp_admissions	24497.0	NaN	NaN	NaN	4291.723313	0.0	223.0	864.0	3893.0
weekly_hosp_admissions_per_million	24497.0	NaN	NaN	NaN	82.61913	0.0	23.728	56.277	109.998
total_tests	79387.0	NaN	NaN	NaN	21104573.938013	0.0	364654.0	2067330.0	10248451.5
new_tests	75403.0	NaN	NaN	NaN	67285.412119	1.0	2244.0	8783.0	37229.0
total_tests_per_thousand	79387.0	NaN	NaN	NaN	924.254762	0.0	43.5855	234.141	894.3745
new_tests_per_thousand	75403.0	NaN	NaN	NaN	3.272466	0.0	0.286	0.971	2.914
new_tests_smoothed	103965.0	NaN	NaN	NaN	142178.363699	0.0	1486.0	6570.0	32205.0
new_tests_smoothed_per_thousand	103965.0	NaN	NaN	NaN	2.826309	0.0	0.203	0.851	2.584
positive_rate	95927.0	NaN	NaN	NaN	0.098163	0.0	0.017	0.055	0.1381
tests_per_case	94348.0	NaN	NaN	NaN	2403.632807	1.0	7.1	17.5	54.6
tests_units	106788	4	tests performed	80099	NaN	NaN	NaN	NaN	NaN
total_vaccinations	85417.0	NaN	NaN	NaN	561697983.425407	0.0	1970788.0	14394348.0	116197175.0
people_vaccinated	81132.0	NaN	NaN	NaN	248706410.740053	0.0	1050009.25	6901087.5	50932952.0
people_fully_vaccinated	78061.0	NaN	NaN	NaN	228663910.073391	1.0	964400.0	6191345.0	47731850.0
total_boosters	53600.0	NaN	NaN	NaN	150581058.901567	1.0	602282.0	5765440.0	40190716.25
new_vaccinations	70971.0	NaN	NaN	NaN	739864.026743	0.0	2010.0	20531.0	173611.5
new_vaccinations_smoothed	195029.0	NaN	NaN	NaN	283875.815135	0.0	279.0	3871.0	31803.0
total_vaccinations_per_hundred	85417.0	NaN	NaN	NaN	124.279558	0.0	44.77	130.55	194.99
people_vaccinated_per_hundred	81132.0	NaN	NaN	NaN	53.501409	0.0	27.88	64.3	77.78
people_fully_vaccinated_per_hundred	78061.0	NaN	NaN	NaN	48.680182	0.0	21.22	57.92	73.61
total_boosters_per_hundred	53600.0	NaN	NaN	NaN	36.301489	0.0	5.92	35.905	57.62
new_vaccinations_smoothed_per_million	195029.0	NaN	NaN	NaN	1851.477596	0.0	106.0	605.0	2402.0
new_people_vaccinated_smoothed	192177.0	NaN	NaN	NaN	106070.698866	0.0	43.0	771.0	9307.0
new_people_vaccinated_smoothed_per_hundred	192177.0	NaN	NaN	NaN	0.07498	0.0	0.001	0.014	0.073
stringency_index	196190.0	NaN	NaN	NaN	42.87756	0.0	22.22	42.85	62.04
population_density	360492.0	NaN	NaN	NaN	394.073095	0.137	37.728	88.125	222.873

	count	unique	top	freq	mean	min	25%	50%	75%
median_age	334663.0	NaN	NaN	NaN	30.456296	15.1	22.2	29.7	38.7
aged_65_older	323270.0	NaN	NaN	NaN	8.684103	1.144	3.526	6.293	13.928
aged_70_older	331315.0	NaN	NaN	NaN	5.486843	0.526	2.063	3.871	8.643
gdp_per_capita	328292.0	NaN	NaN	NaN	18904.182986	661.24	4227.63	12294.876	27216.445
extreme_poverty	211996.0	NaN	NaN	NaN	13.924729	0.1	0.6	2.5	21.4
cardiovasc_death_rate	328865.0	NaN	NaN	NaN	264.639387	79.37	175.695	245.465	333.436
diabetes_prevalence	345911.0	NaN	NaN	NaN	8.556055	0.99	5.35	7.2	10.79
female_smokers	247165.0	NaN	NaN	NaN	10.772465	0.1	1.9	6.3	19.3
male_smokers	243817.0	NaN	NaN	NaN	33.097723	7.7	22.6	33.1	41.5
handwashing_facilities	161741.0	NaN	NaN	NaN	50.649264	1.188	20.859	49.542	82.502
hospital_beds_per_thousand	290689.0	NaN	NaN	NaN	3.106912	0.1	1.3	2.5	4.21
life_expectancy	390299.0	NaN	NaN	NaN	73.702098	53.28	69.5	75.05	79.46
human_development_index	319127.0	NaN	NaN	NaN	0.722139	0.394	0.602	0.74	0.829
population	429435.0	NaN	NaN	NaN	152033640.396274	47.0	523798.0	6336393.0	32969520.0
excess_mortality_cumulative_absolute	13411.0	NaN	NaN	NaN	56047.6535	-37726.098	176.500005	6815.1987	39128.0425
excess_mortality_cumulative	13411.0	NaN	NaN	NaN	9.766431	-44.23	2.06	8.13	15.16
excess_mortality	13411.0	NaN	NaN	NaN	10.925353	-95.92	-1.5	5.66	15.575
excess_mortality_cumulative_per_million	13411.0	NaN	NaN	NaN	1772.6664	-2936.4531	116.872242	1270.8014	2883.02415

ii) Analyzing deaths based on countries

In [7]:
```
# Looking at the values of various location available in data

data['location'].unique()
```

Out[7]:
```
array(['Afghanistan', 'Africa', 'Albania', 'Algeria', 'American Samoa',
       'Andorra', 'Angola', 'Anguilla', 'Antigua and Barbuda',
       'Argentina', 'Armenia', 'Aruba', 'Asia', 'Australia', 'Austria',
       'Azerbaijan', 'Bahamas', 'Bahrain', 'Bangladesh', 'Barbados',
       'Belarus', 'Belgium', 'Belize', 'Benin', 'Bermuda', 'Bhutan',
       'Bolivia', 'Bonaire Sint Eustatius and Saba',
       'Bosnia and Herzegovina', 'Botswana', 'Brazil',
       'British Virgin Islands', 'Brunei', 'Bulgaria', 'Burkina Faso',
       'Burundi', 'Cambodia', 'Cameroon', 'Canada', 'Cape Verde',
       'Cayman Islands', 'Central African Republic', 'Chad', 'Chile',
       'China', 'Colombia', 'Comoros', 'Congo', 'Cook Islands',
       'Costa Rica', "Cote d'Ivoire", 'Croatia', 'Cuba', 'Curacao',
       'Cyprus', 'Czechia', 'Democratic Republic of Congo', 'Denmark',
       'Djibouti', 'Dominica', 'Dominican Republic', 'East Timor',
       'Ecuador', 'Egypt', 'El Salvador', 'England', 'Equatorial Guinea',
       'Eritrea', 'Estonia', 'Eswatini', 'Ethiopia', 'Europe',
       'European Union (27)', 'Faroe Islands', 'Falkland Islands', 'Fiji',
       'Finland', 'France', 'French Guiana', 'French Polynesia', 'Gabon',
       'Gambia', 'Georgia', 'Germany', 'Ghana', 'Gibraltar', 'Greece',
       'Greenland', 'Grenada', 'Guadeloupe', 'Guam', 'Guatemala',
       'Guernsey', 'Guinea', 'Guinea-Bissau', 'Guyana', 'Haiti',
       'High-income countries', 'Honduras', 'Hong Kong', 'Hungary',
       'Iceland', 'India', 'Indonesia', 'Iran', 'Iraq', 'Ireland',
       'Isle of Man', 'Israel', 'Italy', 'Jamaica', 'Japan', 'Jersey',
       'Jordan', 'Kazakhstan', 'Kenya', 'Kiribati', 'Kosovo', 'Kuwait',
       'Kyrgyzstan', 'Laos', 'Latvia', 'Lebanon', 'Lesotho', 'Liberia',
       'Libya', 'Liechtenstein', 'Lithuania', 'Low-income countries',
       'Lower-middle-income countries', 'Luxembourg', 'Macao',
       'Madagascar', 'Malawi', 'Malaysia', 'Maldives', 'Mali', 'Malta',
       'Marshall Islands', 'Martinique', 'Mauritania', 'Mauritius',
       'Mayotte', 'Mexico', 'Micronesia (country)', 'Moldova', 'Monaco',
       'Mongolia', 'Montenegro', 'Montserrat', 'Morocco', 'Mozambique',
       'Myanmar', 'Namibia', 'Nauru', 'Nepal', 'Netherlands',
       'New Caledonia', 'New Zealand', 'Nicaragua', 'Niger', 'Nigeria',
       'Niue', 'North America', 'North Korea', 'North Macedonia',
       'Northern Cyprus', 'Northern Ireland', 'Northern Mariana Islands',
       'Norway', 'Oceania', 'Oman', 'Pakistan', 'Palau', 'Palestine',
       'Panama', 'Papua New Guinea', 'Paraguay', 'Peru', 'Philippines',
       'Pitcairn', 'Poland', 'Portugal', 'Puerto Rico', 'Qatar',
       'Reunion', 'Romania', 'Russia', 'Rwanda', 'Saint Barthelemy',
       'Saint Helena', 'Saint Kitts and Nevis', 'Saint Lucia',
       'Saint Martin (French part)', 'Saint Pierre and Miquelon',
       'Saint Vincent and the Grenadines', 'Samoa', 'San Marino',
       'Sao Tome and Principe', 'Saudi Arabia', 'Scotland', 'Senegal',
       'Serbia', 'Seychelles', 'Sierra Leone', 'Singapore',
       'Sint Maarten (Dutch part)', 'Slovakia', 'Slovenia',
       'Solomon Islands', 'Somalia', 'South Africa', 'South America',
       'South Korea', 'South Sudan', 'Spain', 'Sri Lanka', 'Sudan',
       'Suriname', 'Sweden', 'Switzerland', 'Syria', 'Taiwan',
       'Tajikistan', 'Tanzania', 'Thailand', 'Togo', 'Tokelau', 'Tonga',
       'Trinidad and Tobago', 'Tunisia', 'Turkey', 'Turkmenistan',
       'Turks and Caicos Islands', 'Tuvalu', 'Uganda', 'Ukraine',
       'United Arab Emirates', 'United Kingdom', 'United States',
       'United States Virgin Islands', 'Upper-middle-income countries',
       'Uruguay', 'Uzbekistan', 'Vanuatu', 'Vatican', 'Venezuela',
       'Vietnam', 'Wales', 'Wallis and Futuna', 'Western Sahara', 'World',
       'Yemen', 'Zambia', 'Zimbabwe'], dtype=object)
```

Observation

- The above location not only contains countries but also grouped countries which must not be considered while working for individual countries.

In [8]:
```
# Filtering out the data of countries only by not considering grouped countries
grouped_locations = ['World',
                     'Lower-middle-income countries',
                     'Upper-middle-income countries',
                     'High-income countries',
                     'Low-income countries',
                     'Asia',
                     'Africa',
                     'Europe',
                     'European Union (27)',
                     'North America',
                     'South America']

# Filtered Countries data
countries_data = data[~data["location"].isin(grouped_locations)] #Select only those location which are not in grouped locations
```

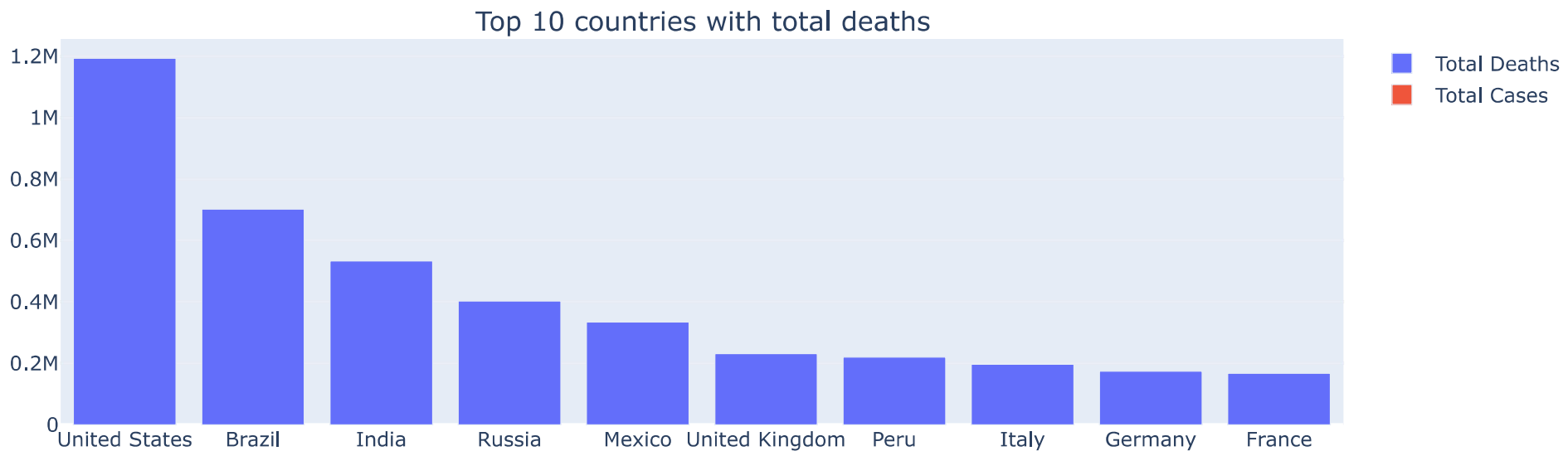
In [9]:
```
# filtering out the deaths and infection numbers for each country

# Grouping all the countries based on total_cases and total_deaths for COVID, followed by taking the largest value
infection_death_country = countries_data.groupby('location')[['total_cases','total_deaths']].max().reset_index()
# reset_index for better output

# Sorting values in descending order for better Interpretation
infection_death_country = infection_death_country.sort_values(by='total_deaths',ascending=False)

# Removing the null values
infection_death_country = infection_death_country.dropna()

infection_death_country.head()
```

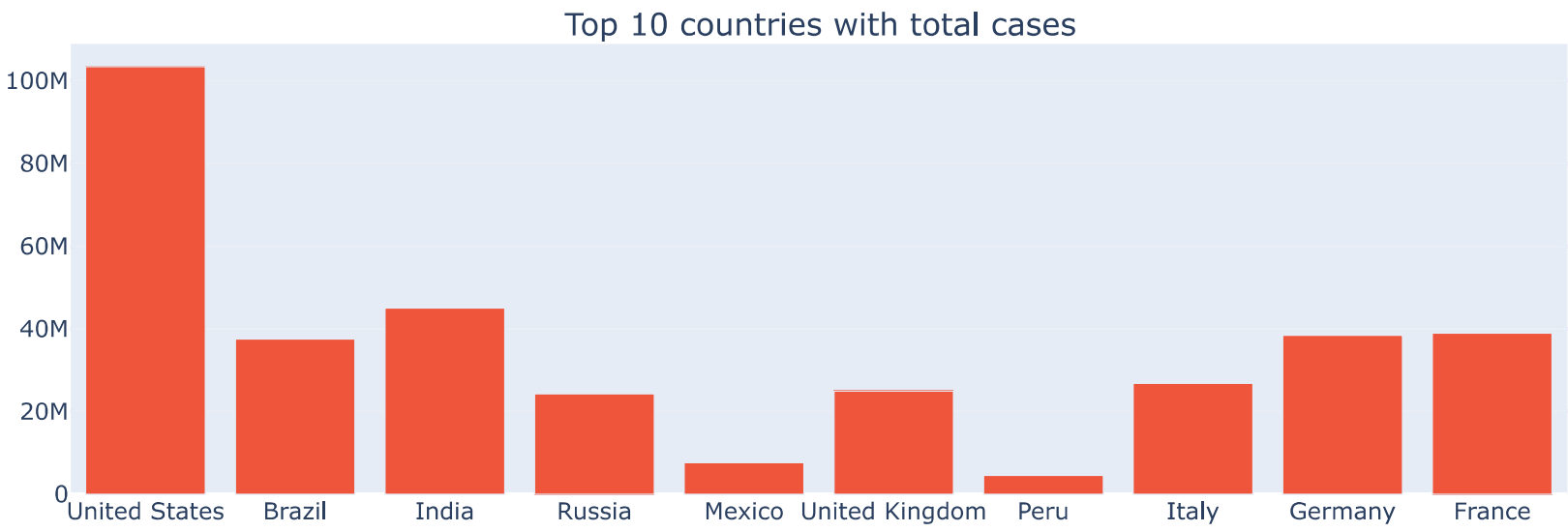
Out[9]:

	location	total_cases	total_deaths
230	United States	103436829.0	1193165.0
28	Brazil	37511921.0	702116.0
97	India	45041748.0	533623.0
179	Russia	24268728.0	403188.0
136	Mexico	7619458.0	334551.0

In [10]:
```
# Loading modules for plotting and visualizations
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
```

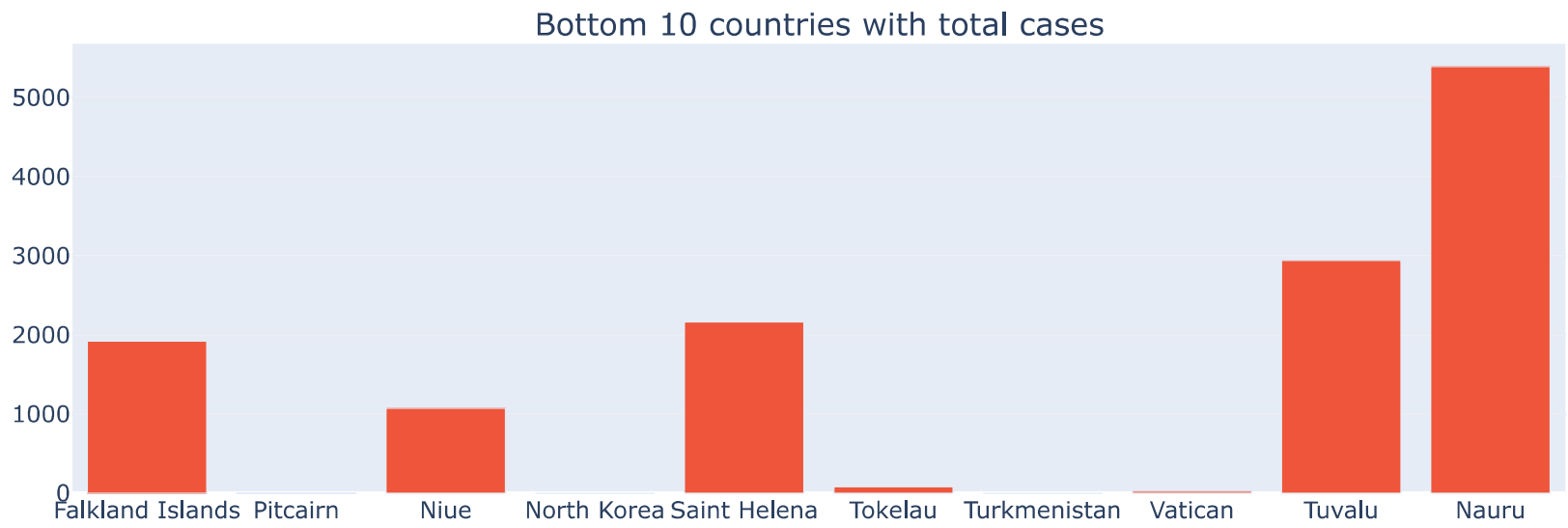
In [11]:
```
# Plotting top 10 deaths

# Defining a canvas for 2 subplots
fig = make_subplots(rows = 2,
                    cols = 1,
                    subplot_titles = ('Top 10 countries with total deaths',
                                      'Top 10 countries with total cases'))

# Filtering out top 10 datapoints
top10 = infection_death_country.nlargest(10,'total_deaths')

# Adding plots in canvas
fig.add_trace(go.Bar(x = top10['location'],
                     y = top10['total_deaths'],
                     name = 'Total Deaths'),
              row = 1,
              col = 1)

fig.add_trace(go.Bar(x = top10['location'],
                     y = top10['total_cases'],
                     name = 'Total Cases'),
              row = 2,
              col = 1)

fig.update_layout(width = 1000, height = 800)

fig.show()
```





In [12]:

```
# Plotting Countries with least deaths (top 10)

# Defining a canvas for 2 subplots
fig = make_subplots(rows = 2,
                    cols = 1,
                    subplot_titles = ('Bottom 10 countries with total deaths',
                                      'Bottom 10 countries with total cases'))

# Filtering out top 10 datapoints
top10 = infection_death_country.nsmallest(10,'total_deaths')


# Adding plots in canvas
fig.add_trace(go.Bar(x = top10['location'],
                     y = top10['total_deaths'],
                     name = 'Total Deaths'),
              row = 1,
              col = 1)

fig.add_trace(go.Bar(x = top10['location'],
                     y = top10['total_cases'],
                     name = 'Total Cases'),
              row = 2,
              col = 1)

fig.update_layout(width = 1000, height = 800)

fig.show()
```



Observation

- Based on the above plot it becomes easier to interpret the numbers in `infection_death_country` much easier.
- Countries with highest total covid deaths have a common factor, i.e. All of them high tourist attraction leading to widespread of covid infections.
- Countries with least total covid deaths have either very strict tourist policy or little to no tourism which would explain so many less cases.
- The reason for 0 to 1 deaths in countries like Niue, North Korea, etc. can be atrributed to multiple factors like:
  - Quick and effective vaccine drives
  - Best Quarantine Practices

iii) Vaccination for Old People

- While working with this problem, I have assumed that the question is asking based on country wise old people and will be considering the mean age in my calculations and plots.

In [13]:
```
# Taking the countries dataframe created above for countries and making a copy of locations
old_people_country = countries_data[['location']].copy()

# taking the mean age of of all people above age 65 and 70
old_people_country['mean_old_age'] = data[['aged_65_older', 'aged_70_older']].mean(axis=1)

# Grouping based on location with mean old ages  and sorting in descending order
old_people_country = old_people_country.groupby('location', as_index=False)['mean_old_age'].mean()
old_people_country.sort_values(by='mean_old_age', ascending=False, inplace = True)

# Reseting the index values
old_people_country = old_people_country.reset_index(drop=True)

old_people_country
```

Out[13]:

	location	mean_old_age
0	Japan	22.7710
1	Italy	19.6305
2	Germany	18.7050
3	Portugal	18.2130
4	Greece	17.4600
5	Serbia	17.3660
6	Finland	17.2460
7	Bulgaria	17.0365
8	Latvia	16.9450
9	Sweden	16.7090
10	Spain	16.6175
11	Austria	16.4750
12	Estonia	16.4715
13	France	16.3985
14	Lithuania	16.3900
15	Croatia	16.3885
16	Denmark	16.0010
17	Slovenia	15.9960
18	Belgium	15.7100
19	Switzerland	15.5400
20	United Kingdom	15.5220
21	Malta	15.3750
22	Netherlands	15.3300
23	Czechia	15.3035
24	Hungary	15.2765
25	Romania	14.7700
26	United States Virgin Islands	14.7000
27	Canada	13.8905
28	Norway	13.8170
29	Ukraine	13.7975
30	Bosnia and Herzegovina	13.6400
31	Poland	13.4825
32	Hong Kong	13.2305
33	Curacao	13.2175
34	Australia	12.8165
35	United States	12.5725
36	Georgia	12.5540
37	Martinique	12.5430
38	New Zealand	12.5210
39	Uruguay	12.5080
40	Puerto Rico	12.4985
41	Belarus	12.2935
42	Cuba	12.2285
43	Barbados	12.2125
44	Slovakia	12.1185
45	Montenegro	12.0785
46	Luxembourg	12.0770
47	Iceland	11.8190
48	Russia	11.7855
49	Ireland	11.3030
50	South Korea	11.2680

	location	mean_old_age
51	Cyprus	10.9895
52	Albania	10.9155
53	North Macedonia	10.7100
54	Aruba	10.2685
55	Singapore	9.9855
56	Israel	9.5460
57	Armenia	9.4015
58	Argentina	9.3195
59	Thailand	9.1315
60	Chile	9.0125
61	Moldova	8.9095
62	Mauritius	8.4145
63	Taiwan	8.3530
64	China	8.2850
65	New Caledonia	8.2215
66	Saint Lucia	8.0630
67	Jamaica	8.0370
68	Trinidad and Tobago	7.9165
69	North Korea	7.8150
70	Sri Lanka	7.7000
71	Costa Rica	7.5810
72	Guam	7.5220
73	Macao	7.3945
74	Bahamas	7.0980
75	Seychelles	7.0960
76	Lebanon	6.9720
77	El Salvador	6.8450
78	Reunion	6.8410
79	Brazil	6.8060
80	Turkey	6.6070
81	Tunisia	6.5380
82	Panama	6.4740
83	Saint Vincent and the Grenadines	6.2780
84	French Polynesia	6.1840
85	Grenada	6.1625
86	Colombia	5.9790
87	Vietnam	5.9340
88	Kazakhstan	5.8080
89	Peru	5.8030
90	Antigua and Barbuda	5.7820
91	Ecuador	5.7810
92	Dominican Republic	5.7000
93	Mexico	5.5890
94	Suriname	5.5810
95	Bolivia	5.5485
96	Morocco	5.4890
97	Venezuela	5.2645
98	Paraguay	5.1055
99	Algeria	5.0340
100	Azerbaijan	4.9445
101	Tonga	4.9260

	location	mean_old_age
102	Malaysia	4.8500
103	Fiji	4.7540
104	India	4.7015
105	Samoa	4.5850
106	Nepal	4.5105
107	Nicaragua	4.4820
108	Myanmar	4.4260
109	Iran	4.3110
110	South Africa	4.1985
111	Indonesia	4.1860
112	Bangladesh	4.1800
113	Guyana	4.0710
114	Egypt	4.0250
115	Cape Verde	3.9485
116	Bhutan	3.9310
117	Haiti	3.8770
118	Guatemala	3.8550
119	Honduras	3.7675
120	Philippines	3.7320
121	Gabon	3.7130
122	Kyrgyzstan	3.6855
123	Uzbekistan	3.6710
124	Pakistan	3.6375
125	Libya	3.6200
126	Micronesia (country)	3.6010
127	Lesotho	3.5765
128	Vanuatu	3.5070
129	Maldives	3.4975
130	Brunei	3.4865
131	Turkmenistan	3.4090
132	Cambodia	3.3985
133	Djibouti	3.2965
134	Mongolia	3.2260
135	Laos	3.1755
136	Botswana	3.0915
137	Jordan	3.0855
138	Belize	3.0660
139	Kiribati	3.0525
140	Papua New Guinea	2.9750
141	Central African Republic	2.9530
142	French Guiana	2.8970
143	Eritrea	2.8890
144	Namibia	2.8185
145	Tajikistan	2.8105
146	Ethiopia	2.7945
147	Sudan	2.7910
148	Solomon Islands	2.7750
149	South Sudan	2.7365
150	Congo	2.7325
151	East Timor	2.7265
152	Ghana	2.6665

	location	mean_old_age
153	Benin	2.5930
154	Syria	2.5770
155	Iraq	2.5715
156	Saudi Arabia	2.5700
157	Cameroon	2.5420
158	Sao Tome and Principe	2.5240
159	Mozambique	2.5140
160	Eswatini	2.5040
161	Tanzania	2.4910
162	Mauritania	2.4650
163	Guinea	2.4340
164	Mayotte	2.4110
165	Liberia	2.4065
166	Senegal	2.4020
167	Palestine	2.3845
168	Democratic Republic of Congo	2.3825
169	Malawi	2.3810
170	Zimbabwe	2.3520
171	Comoros	2.3445
172	Rwanda	2.3080
173	Madagascar	2.3075
174	Equatorial Guinea	2.2990
175	Guinea-Bissau	2.2835
176	Cote d'Ivoire	2.2575
177	Yemen	2.2525
178	Togo	2.1820
179	Somalia	2.1135
180	Kenya	2.1070
181	Nigeria	2.0990
182	Burundi	2.0330
183	Zambia	2.0110
184	Mali	2.0025
185	Chad	1.9660
186	Niger	1.9655
187	Afghanistan	1.9590
188	Oman	1.9425
189	Sierra Leone	1.9115
190	Angola	1.8835
191	Burkina Faso	1.8835
192	Bahrain	1.8795
193	Gambia	1.8780
194	Uganda	1.7380
195	Kuwait	1.7295
196	Western Sahara	1.3800
197	Qatar	0.9620
198	United Arab Emirates	0.8350
199	American Samoa	NaN
200	Andorra	NaN
201	Anguilla	NaN
202	Bermuda	NaN
203	Bonaire Sint Eustatius and Saba	NaN

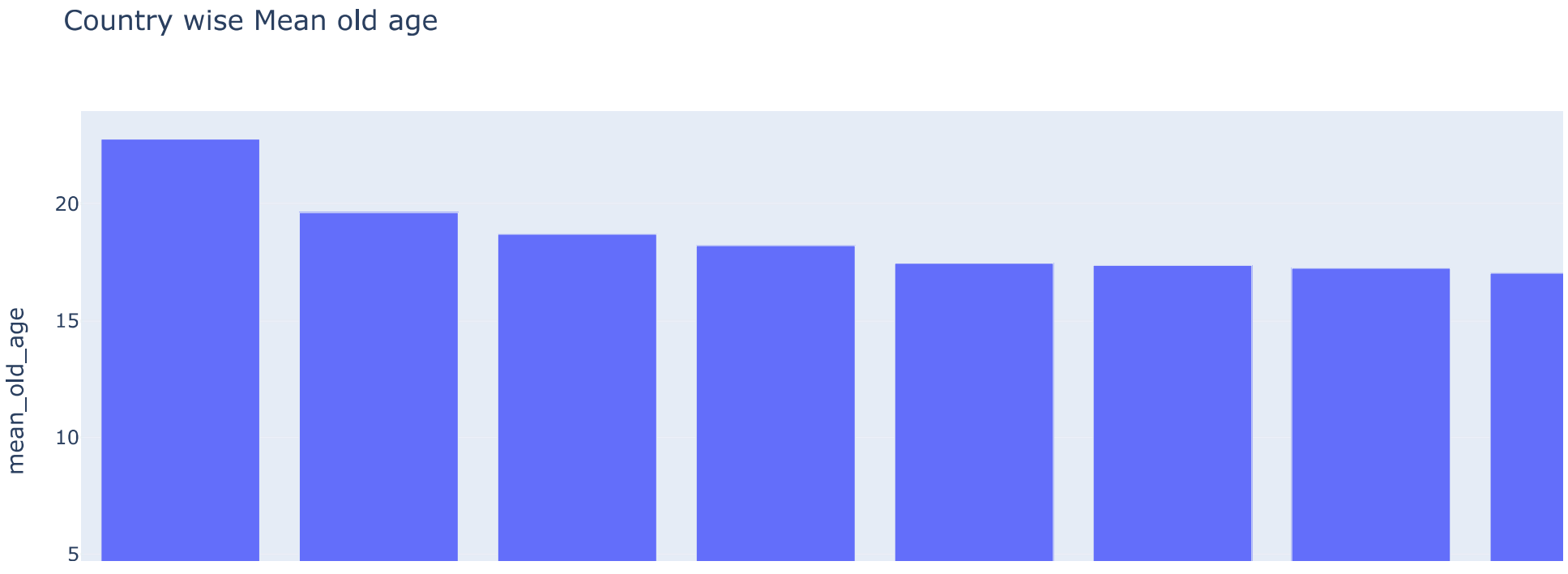
	location	mean_old_age
204	British Virgin Islands	NaN
205	Cayman Islands	NaN
206	Cook Islands	NaN
207	Dominica	NaN
208	England	NaN
209	Falkland Islands	NaN
210	Faroe Islands	NaN
211	Gibraltar	NaN
212	Greenland	NaN
213	Guadeloupe	NaN
214	Guernsey	NaN
215	Isle of Man	NaN
216	Jersey	NaN
217	Kosovo	NaN
218	Liechtenstein	NaN
219	Marshall Islands	NaN
220	Monaco	NaN
221	Montserrat	NaN
222	Nauru	NaN
223	Niue	NaN
224	Northern Cyprus	NaN
225	Northern Ireland	NaN
226	Northern Mariana Islands	NaN
227	Oceania	NaN
228	Palau	NaN
229	Pitcairn	NaN
230	Saint Barthelemy	NaN
231	Saint Helena	NaN
232	Saint Kitts and Nevis	NaN
233	Saint Martin (French part)	NaN
234	Saint Pierre and Miquelon	NaN
235	San Marino	NaN
236	Scotland	NaN
237	Sint Maarten (Dutch part)	NaN
238	Tokelau	NaN
239	Turks and Caicos Islands	NaN
240	Tuvalu	NaN
241	Vatican	NaN
242	Wales	NaN
243	Wallis and Futuna	NaN

In [14]:
```
# There are some null values in the old_people country dataframe which need to be dropped as they are of no use to us
print("Shape of old people data before removing null values:",old_people_country.shape)
old_people_country.dropna(inplace = True)
print("Shape of old people data after removing null values:",old_people_country.shape)
```

```
Shape of old people data before removing null values: (244, 2)
Shape of old people data after removing null values: (199, 2)
```

- The removal of null values was inevitable as replacing with mean or median without context is not a wise choice, also no other source to replace the value is available so the null values must be dropped

In [15]:
```
# Plotting the above dataframe for better understanding
figure = px.bar(old_people_country.nlargest(10,'mean_old_age'),
                x = 'location',
               y = 'mean_old_age',
               title = 'Country wise Mean old age')
figure.show()
```



Observation

- From the above plot, we see the top countries with highest mean old age, i.e. these countries should prioritizing vaccinatig old people first.

iv) Analyzing the trends of Covid for my Neighbourhood

- For analyzing the trend, I will be considering my neighbourhood as `United States` in `location` column of the given data

In [16]:
```
# filtering out all the data points where the location is United States
USA_neighbors = data[data['location'] == "United States"]
USA_neighbors.head()
```

Out[16]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoothed	total_cases_
403451	USA	North America	United States	2020-01-05	0.0	0.0	NaN	0.0	0.0	NaN	
403452	USA	North America	United States	2020-01-06	0.0	0.0	NaN	0.0	0.0	NaN	
403453	USA	North America	United States	2020-01-07	0.0	0.0	NaN	0.0	0.0	NaN	
403454	USA	North America	United States	2020-01-08	0.0	0.0	NaN	0.0	0.0	NaN	
403455	USA	North America	United States	2020-01-09	0.0	0.0	NaN	0.0	0.0	NaN	

In [17]:
```
# Checking the shape of data
USA_neighbors.shape
```

Out[17]: (1674, 67)

In [18]:
```
# lets look at a general description of the data
USA_neighbors.describe(include = "all").T
```

Out[18]:

	count	unique	top	freq	mean	min	25%	50%	75%
iso_code	1674	1	USA	1674	NaN	NaN	NaN	NaN	NaN
continent	1674	1	North America	1674	NaN	NaN	NaN	NaN	NaN
location	1674	1	United States	1674	NaN	NaN	NaN	NaN	NaN
date	1674	NaN	NaN	NaN	2022-04-20 12:00:00	2020-01-05 00:00:00	2021-02-26 06:00:00	2022-04-20 12:00:00	2023-06-12 18:00:00
total_cases	1674.0	NaN	NaN	NaN	63270300.750896	0.0	27864340.0	79946773.0	103436829.0
new_cases	1232.0	NaN	NaN	NaN	83958.465097	0.0	0.0	0.0	0.0
new_cases_smoothed	1227.0	NaN	NaN	NaN	84300.594131	0.0	29820.0	54857.143	101376.714
total_deaths	1674.0	NaN	NaN	NaN	777909.996416	0.0	506493.0	984444.0	1129984.0
new_deaths	1674.0	NaN	NaN	NaN	712.762843	0.0	0.0	0.0	0.0
new_deaths_smoothed	1669.0	NaN	NaN	NaN	714.580235	0.0	198.857	394.857	972.857
total_cases_per_million	1674.0	NaN	NaN	NaN	185253.278635	0.0	81585.83	234081.42	302859.5
new_cases_per_million	1232.0	NaN	NaN	NaN	245.827515	0.0	0.0	0.0	0.0
new_cases_smoothed_per_million	1227.0	NaN	NaN	NaN	246.829293	0.0	87.312	160.62	296.828
total_deaths_per_million	1674.0	NaN	NaN	NaN	2277.693871	0.0	1482.994	2882.418	3308.554
new_deaths_per_million	1674.0	NaN	NaN	NaN	2.086938	0.0	0.0	0.0	0.0
new_deaths_smoothed_per_million	1669.0	NaN	NaN	NaN	2.092237	0.0	0.582	1.156	2.848
reproduction_rate	1034.0	NaN	NaN	NaN	1.080938	0.52	0.91	1.02	1.13
icu_patients	1381.0	NaN	NaN	NaN	7703.354815	566.0	2044.0	3865.0	11534.0
icu_patients_per_million	1381.0	NaN	NaN	NaN	22.771458	1.673	6.042	11.425	34.095
hosp_patients	1381.0	NaN	NaN	NaN	36706.537292	4633.0	15321.0	26484.0	42200.0
hosp_patients_per_million	1381.0	NaN	NaN	NaN	108.50616	13.695	45.29	78.288	124.745
weekly_icu_admissions	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
weekly_icu_admissions_per_million	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
weekly_hosp_admissions	1375.0	NaN	NaN	NaN	36406.191273	5396.0	17456.0	28510.0	42353.5
weekly_hosp_admissions_per_million	1375.0	NaN	NaN	NaN	107.618337	15.951	51.601	84.277	125.199
total_tests	840.0	NaN	NaN	NaN	410386446.70119	348.0	116097254.75	416340619.5	660354848.25
new_tests	840.0	NaN	NaN	NaN	1086629.909524	348.0	661345.5	1011313.0	1496002.75
total_tests_per_thousand	840.0	NaN	NaN	NaN	1217.772511	0.001	344.5045	1235.4405	1959.524
new_tests_per_thousand	840.0	NaN	NaN	NaN	3.224456	0.001	1.96225	3.001	4.4395
new_tests_smoothed	833.0	NaN	NaN	NaN	1094185.831933	1165.0	773383.0	1080551.0	1479284.0
new_tests_smoothed_per_thousand	833.0	NaN	NaN	NaN	3.246857	0.003	2.295	3.206	4.39
positive_rate	834.0	NaN	NaN	NaN	0.085315	0.017	0.05	0.072	0.105
tests_per_case	834.0	NaN	NaN	NaN	16.776499	3.4	9.5	13.9	20.0
tests_units	840	1	tests performed	840	NaN	NaN	NaN	NaN	NaN
total_vaccinations	878.0	NaN	NaN	NaN	471846074.571754	45620.0	349277487.5	560841574.0	627458474.75
people_vaccinated	878.0	NaN	NaN	NaN	214009733.722096	36817.0	188706163.5	253955428.5	264498443.25
people_fully_vaccinated	878.0	NaN	NaN	NaN	181615607.96697	9669.0	168845486.75	218097651.5	226547731.5
total_boosters	575.0	NaN	NaN	NaN	53957174.561739	1.0	308.5	39811513.0	106605615.5
new_vaccinations	877.0	NaN	NaN	NaN	771588.554162	2556.0	198058.0	475619.0	985889.0
new_vaccinations_smoothed	877.0	NaN	NaN	NaN	771667.657925	4848.0	228745.0	503599.0	1025590.0
total_vaccinations_per_hundred	878.0	NaN	NaN	NaN	142.118542	0.01	105.1975	168.92	188.985
people_vaccinated_per_hundred	878.0	NaN	NaN	NaN	64.459021	0.01	56.84	76.49	79.6675
people_fully_vaccinated_per_hundred	878.0	NaN	NaN	NaN	54.702141	0.0	50.86	65.69	68.2375
total_boosters_per_hundred	575.0	NaN	NaN	NaN	16.251757	0.0	0.0	11.99	32.11
new_vaccinations_smoothed_per_million	877.0	NaN	NaN	NaN	2324.238312	15.0	689.0	1517.0	3089.0
new_people_vaccinated_smoothed	877.0	NaN	NaN	NaN	308281.36488	4648.0	41923.0	75895.0	372643.0
new_people_vaccinated_smoothed_per_hundred	877.0	NaN	NaN	NaN	0.092851	0.001	0.013	0.023	0.112
stringency_index	1092.0	NaN	NaN	NaN	48.580549	0.0	30.57	52.19	68.06
population_density	1674.0	NaN	NaN	NaN	35.608	35.608	35.608	35.608	35.608

	count	unique	top	freq	mean	min	25%	50%	75%
median_age	1674.0	NaN	NaN	NaN	38.3	38.3	38.3	38.3	38.3
aged_65_older	1674.0	NaN	NaN	NaN	15.413	15.413	15.413	15.413	15.413
aged_70_older	1674.0	NaN	NaN	NaN	9.732	9.732	9.732	9.732	9.732
gdp_per_capita	1674.0	NaN	NaN	NaN	54225.446	54225.446	54225.446	54225.446	54225.446
extreme_poverty	1674.0	NaN	NaN	NaN	1.2	1.2	1.2	1.2	1.2
cardiovasc_death_rate	1674.0	NaN	NaN	NaN	151.089	151.089	151.089	151.089	151.089
diabetes_prevalence	1674.0	NaN	NaN	NaN	10.79	10.79	10.79	10.79	10.79
female_smokers	1674.0	NaN	NaN	NaN	19.1	19.1	19.1	19.1	19.1
male_smokers	1674.0	NaN	NaN	NaN	24.6	24.6	24.6	24.6	24.6
handwashing_facilities	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
hospital_beds_per_thousand	1674.0	NaN	NaN	NaN	2.77	2.77	2.77	2.77	2.77
life_expectancy	1674.0	NaN	NaN	NaN	78.86	78.86	78.86	78.86	78.86
human_development_index	1674.0	NaN	NaN	NaN	0.926	0.926	0.926	0.926	0.926
population	1674.0	NaN	NaN	NaN	338289856.0	338289856.0	338289856.0	338289856.0	338289856.0
excess_mortality_cumulative_absolute	209.0	NaN	NaN	NaN	832382.38837	-13459.8	475144.72	987298.8	1280796.0
excess_mortality_cumulative	209.0	NaN	NaN	NaN	12.928421	-3.84	12.03	14.43	15.8
excess_mortality	209.0	NaN	NaN	NaN	11.40311	-4.44	2.55	7.3	19.34
excess_mortality_cumulative_per_million	209.0	NaN	NaN	NaN	2458.452738	-40.06584	1409.9349	2918.4995	3767.085

Observation

- There are some null values in certain columns which needs to be dealt with

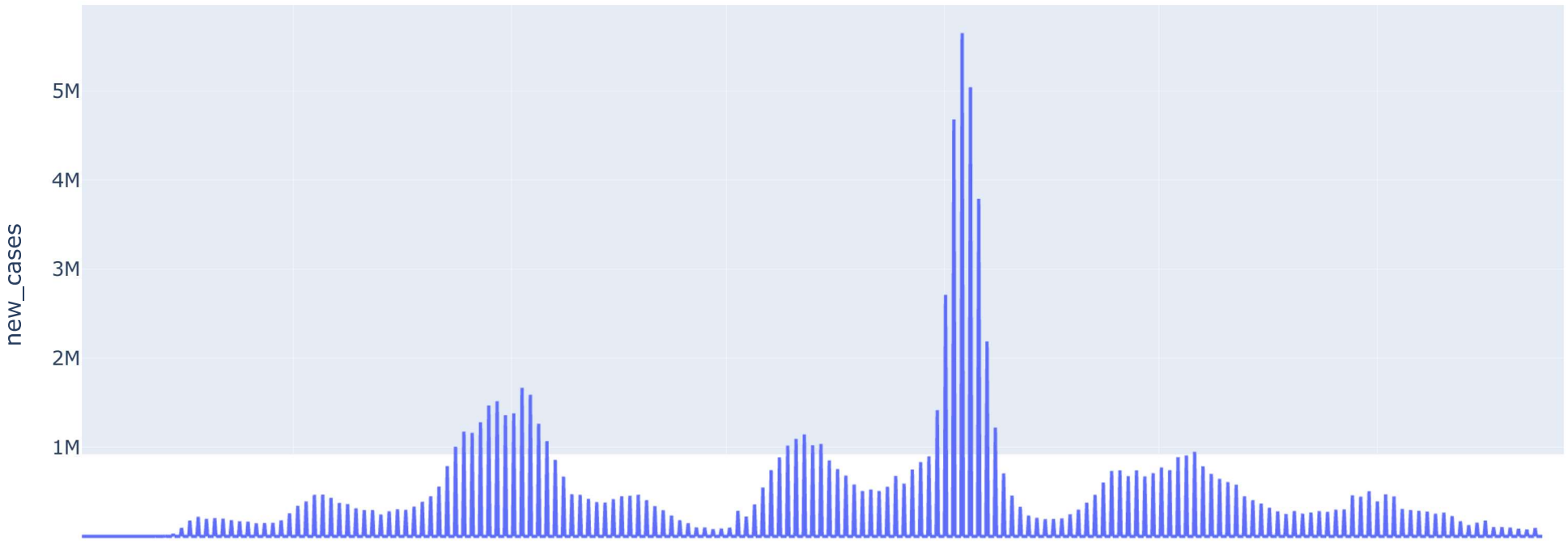
In [19]:
```
# Checking for null values
percent = (USA_neighbors.isnull().sum() * 100)/USA_neighbors.shape[0]
percent
```

Out[19]:
```
iso_code                                    0.000000
continent                                   0.000000
location                                    0.000000
date                                        0.000000
total_cases                                 0.000000
new_cases                                  26.403823
new_cases_smoothed                         26.702509
total_deaths                                0.000000
new_deaths                                  0.000000
new_deaths_smoothed                         0.298686
total_cases_per_million                     0.000000
new_cases_per_million                      26.403823
new_cases_smoothed_per_million             26.702509
total_deaths_per_million                    0.000000
new_deaths_per_million                      0.000000
new_deaths_smoothed_per_million             0.298686
reproduction_rate                          38.231780
icu_patients                               17.502987
icu_patients_per_million                   17.502987
hosp_patients                              17.502987
hosp_patients_per_million                  17.502987
weekly_icu_admissions                     100.000000
weekly_icu_admissions_per_million         100.000000
weekly_hosp_admissions                     17.861410
weekly_hosp_admissions_per_million         17.861410
total_tests                                49.820789
new_tests                                  49.820789
total_tests_per_thousand                   49.820789
new_tests_per_thousand                     49.820789
new_tests_smoothed                         50.238949
new_tests_smoothed_per_thousand            50.238949
positive_rate                              50.179211
tests_per_case                             50.179211
tests_units                                49.820789
total_vaccinations                         47.550777
people_vaccinated                          47.550777
people_fully_vaccinated                    47.550777
total_boosters                             65.651135
new_vaccinations                           47.610514
new_vaccinations_smoothed                  47.610514
total_vaccinations_per_hundred             47.550777
people_vaccinated_per_hundred              47.550777
people_fully_vaccinated_per_hundred        47.550777
total_boosters_per_hundred                 65.651135
new_vaccinations_smoothed_per_million      47.610514
new_people_vaccinated_smoothed             47.610514
new_people_vaccinated_smoothed_per_hundred 47.610514
stringency_index                           34.767025
population_density                          0.000000
median_age                                  0.000000
aged_65_older                               0.000000
aged_70_older                               0.000000
gdp_per_capita                              0.000000
extreme_poverty                             0.000000
cardiovasc_death_rate                       0.000000
diabetes_prevalence                         0.000000
female_smokers                              0.000000
male_smokers                                0.000000
handwashing_facilities                    100.000000
hospital_beds_per_thousand                  0.000000
life_expectancy                             0.000000
human_development_index                     0.000000
population                                  0.000000
excess_mortality_cumulative_absolute       87.514934
excess_mortality_cumulative                87.514934
excess_mortality                           87.514934
excess_mortality_cumulative_per_million    87.514934
dtype: float64
```

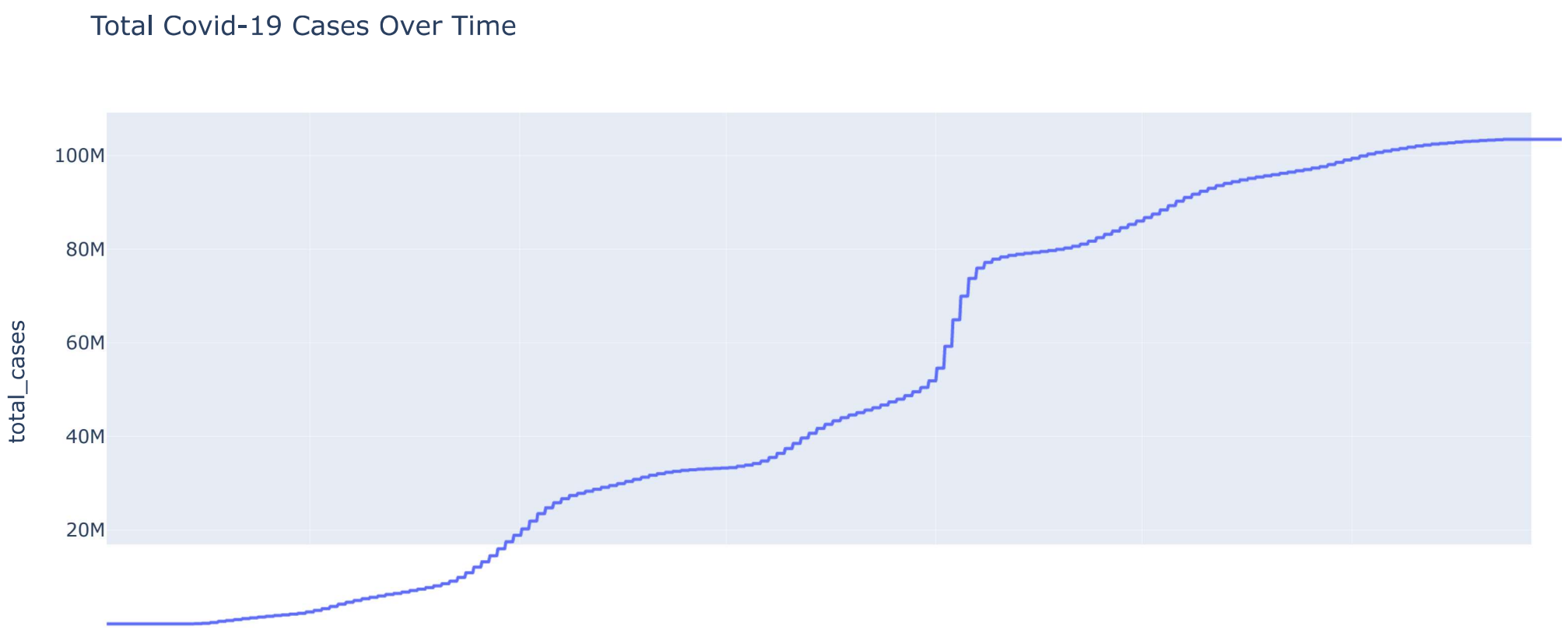
Observation

- There are multiple columns with more than **60% Null values** and some of them are also going upto all the way **100% null values**.
- Since we will not be dealing with all the columns for trend, I will b dealing with null values based on columns as and when they appear.

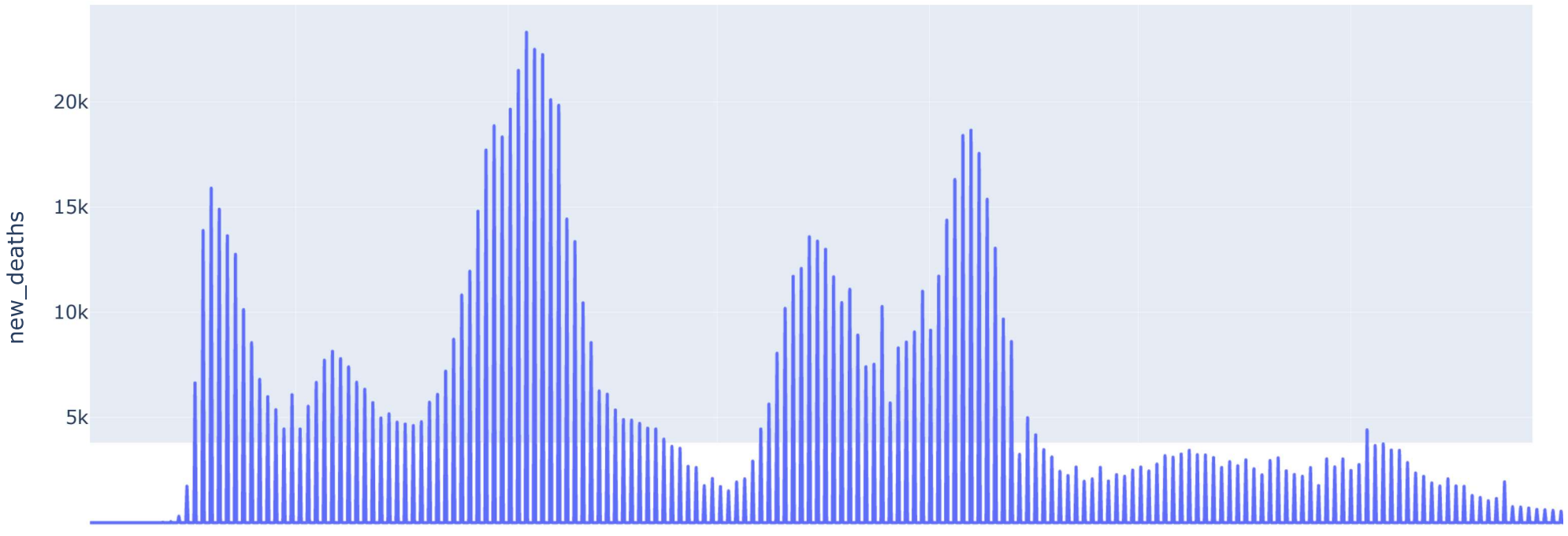
In [20]:
```
# Looking at the trend of daily new cases in the given data for USA
figure = px.line(USA_neighbors,
                 x='date',
                 y='new_cases',
                 title='New COVID-19 Cases Over Time')
figure.show()
```



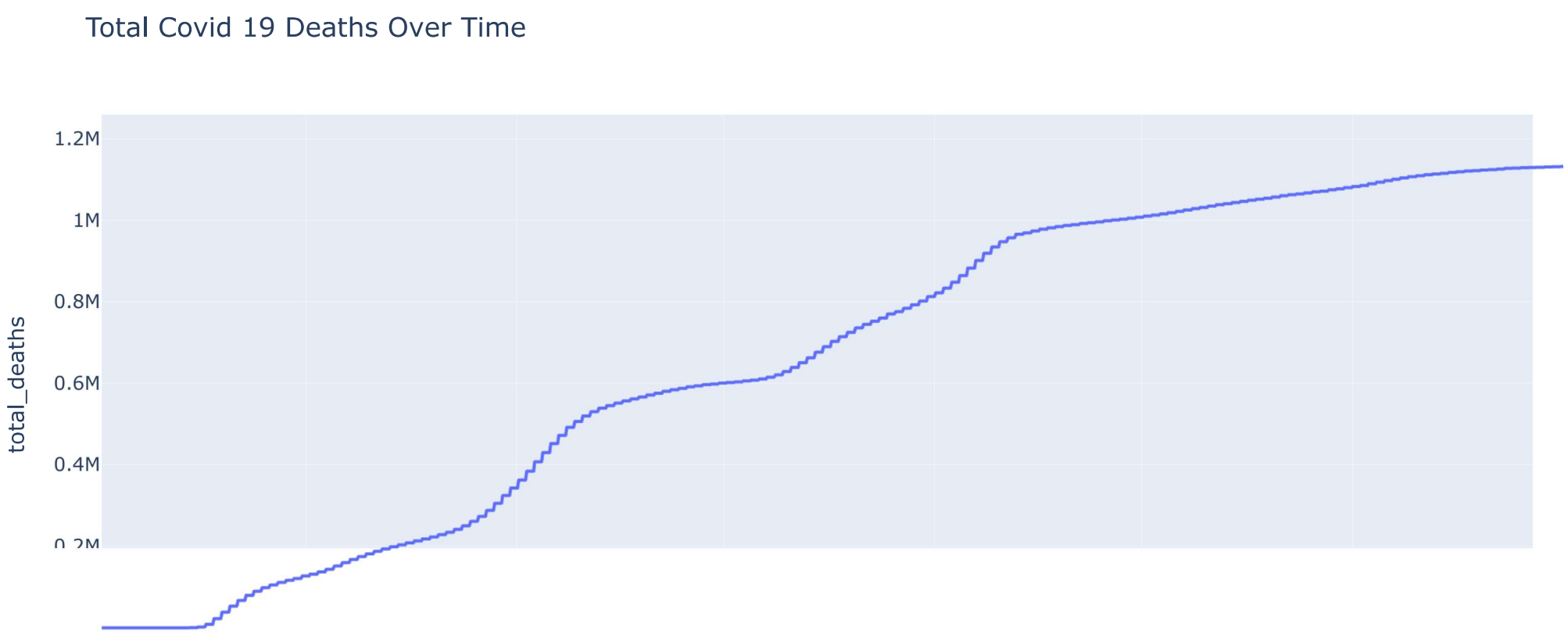
In [21]:
```
# Looking at the trend of total covid cases over time
figure = px.line(USA_neighbors,
                 x='date',
                 y='total_cases',
                 title='Total Covid-19 Cases Over Time')
figure.show()
```



In [22]:
```
# Looking at the trend of New Deaths due to Covid-19
figure = px.line(USA_neighbors,
                 x='date',
                 y='new_deaths',
                 title='New Deaths Over Time')
figure.show()
```



In [23]:
```
# Looking at the trend of Total Deaths over time
figure = px.line(USA_neighbors,
                 x='date',
                 y='total_deaths',
                 title='Total Covid 19 Deaths Over Time')
figure.show()
```

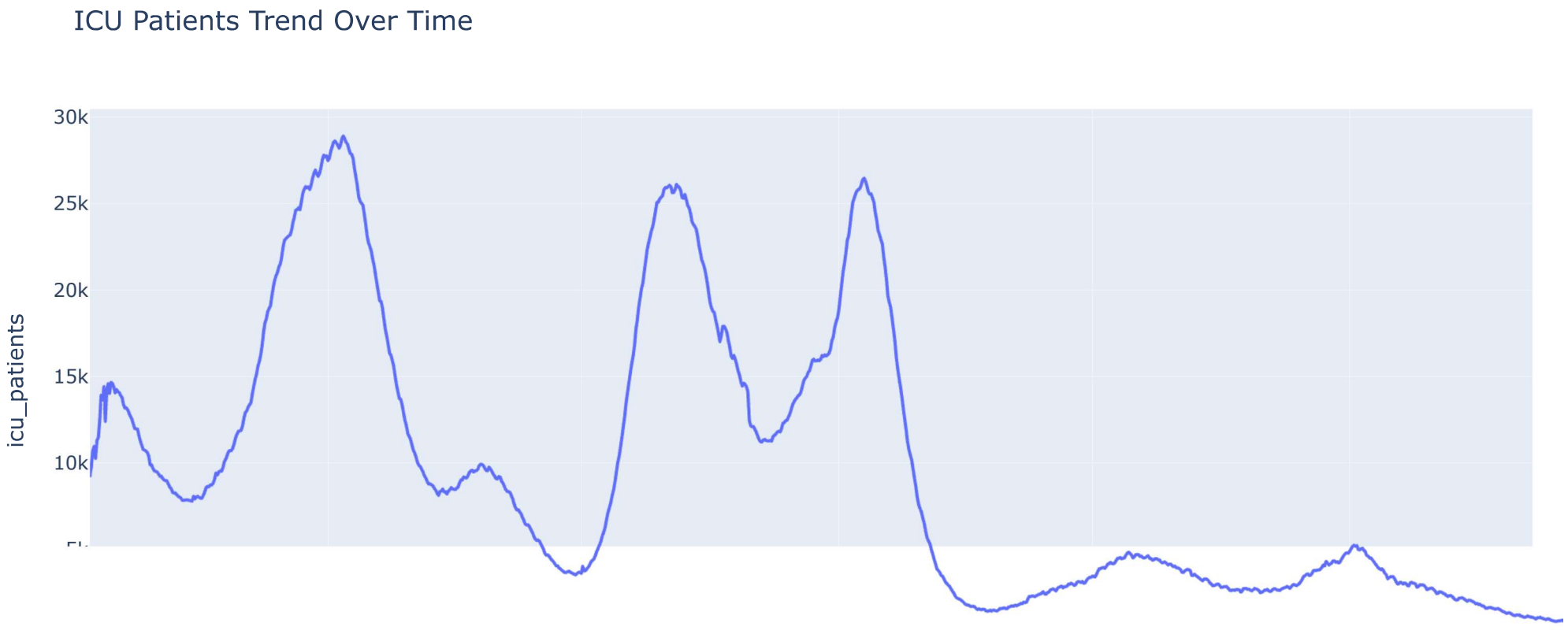


In [24]:
```
# Looking at the trend of covid positve rate over time

# Since more than 50% of the data is missing for positive rate, those rows are removed for visualizations
temp =  USA_neighbors.dropna(subset = ['positive_rate'])

figure = px.line(temp,
                 x='date',
                 y='positive_rate',
                 title='Covid-19 Positive Rate Over Time')
figure.show()
```

In [25]:
```python
# Looking at the trend of Stringency index over time

# Stringency Index: Government Response Stringency Index: composite measure based on 9 response indicators
#                   including school closures, workplace closures, and travel bans, rescaled to a value from 0 to 100
#                   (100 = strictest response)

# Stringency Index column contains almost 35% null values and
# since its relatively stable from outlier point of view as mean is close to median
# I will be replacing all the null values of Stringency index with Median value

temp = USA_neighbors.copy()
temp['stringency_index'].fillna(USA_neighbors['stringency_index'].median(),inplace=True)

# Plotting the trend
figure = px.line(temp,
                 x='date',
                 y='stringency_index',
                 title='Stringency Index Over Time')
figure.show()
```

In [26]:
```python
# Finally lets see the trend of ICU patients over time

# Since the null values are only 17.5% for ICU patients,
# I will be dropping them, as the difference between mean and median is too big for imputation

temp =  USA_neighbors.dropna(subset = ['icu_patients'])

figure = px.line(temp,
```

```
                  x='date',
                  y='icu_patients',
                  title='ICU Patients Trend Over Time')
figure.show()
```



Observations

- `New COVID-19` cases did **spike in Jan 2021 and Jan 2022** when the COVID waves 1 and 2 hit, but by the time we arrive in **2024**, there are **no new cases**. The reasons for the same can be attributed to, good quarantine, vaccinations and human immunity adaptation to COVID,etc.
- The `total COVID-19 cases` over time becomes more or less ``stable as we enter 2024`, in lieu with our previous observation, as there are little to no new cases, so the total doesn't go up in my neighbourhood, i.e. USA.
- Commenting on the `positive rate` might be a bit difficult based on recent time as the data is missing, but upon looking at the overall trend, it is `oscillating`; which means there are chances `people might turn up covid positive`, but `based on the first 2 plots its safe to say with proper doctor prescription, full recovery is possible`.
- Commenting on `stringency index` directly is not very appropriate as we did replace a significant amount of data with median value, but when viewed `in lieu with the ICU patients trend` we can come to the following conclusions:
  - `Stringency index` being `constant` in the `year 2024` can be interpreted as `No more closures and bans`, becuase the number of ICU patients dropped and as well as based on previous plots, its safe to say in the year 2024, `USA returned closed to normality`.
  - ICU patients number dropping over-time solidifies the fact that practices like, vaccination, quarantine and human immunity adaptation to covid helped us to beat COVID-19.

Conclusion

- For my neighbourhood, USA, COVID-19 waves are gone, but there are still chances of an individual being diagnosed with COVID-19 but the chances of full recovery is very high.

C) Analyzing The Effectivenss of Vaccination

In order to analyze the effectiveness of vaccination:

- Plot the percentage of people vaccinated with time.
- Influence of vaccination with respect to new covid cases and total covid cases.
- Change in new covid cases and total covid cases post vaccination
- As per the information mentioned about the dataset above, we will need to work with information of total death columns, total cases columns, vaccination columns, population and date columns to properly analyze the situtation.

Assumed Hypothesis

```
VACCINATION HELPED IN REDUCTION OF COVID CASES AND HAS BEEN BENEFICIAL
```

- In the subsequent code blocks we will try to validate this hypothesis, if what has been assumed is correct or not.

In [27]:
```
# Since the effectiveness will be analyzed at world level, we will take location as world

covid_world = data[data['location']=='World']

# Selecting only relevant columns
```

```
covid_world = covid_world[['date',
                   'population',
                    'total_cases',
                    'new_cases',
                    'total_deaths',
                    'new_deaths',
                    'total_vaccinations',
                    'new_vaccinations',
                    'people_fully_vaccinated',
                    'people_vaccinated',
                    'total_boosters']]
covid_world.head()
```

Out[27]:

	date	population	total_cases	new_cases	total_deaths	new_deaths	total_vaccinations	new_vaccinations	people_fully_vaccinated	people_vaccir
422729	2020-01-05	7975105024	2.0	2.0	3.0	3.0	NaN	NaN	NaN	
422730	2020-01-06	7975105024	2.0	0.0	3.0	0.0	NaN	NaN	NaN	
422731	2020-01-07	7975105024	2.0	0.0	3.0	0.0	NaN	NaN	NaN	
422732	2020-01-08	7975105024	2.0	0.0	3.0	0.0	NaN	NaN	NaN	
422733	2020-01-09	7975105024	2.0	0.0	3.0	0.0	NaN	NaN	NaN	

In [28]:
```
# Analyzing the summary information
covid_world.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1684 entries, 422729 to 424412
Data columns (total 11 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   date                     1684 non-null   datetime64[ns]
 1   population               1684 non-null   int64
 2   total_cases              1674 non-null   float64
 3   new_cases                1674 non-null   float64
 4   total_deaths             1674 non-null   float64
 5   new_deaths               1674 non-null   float64
 6   total_vaccinations       1352 non-null   float64
 7   new_vaccinations         1346 non-null   float64
 8   people_fully_vaccinated  1341 non-null   float64
 9   people_vaccinated        1352 non-null   float64
 10  total_boosters           1325 non-null   float64
dtypes: datetime64[ns](1), float64(9), int64(1)
memory usage: 157.9 KB
```

Observation

- Superficially glancing at the above info, we can concur that there are null values which will later trouble us in mathematical calculations, so they needed to be handled appropriately.

In [29]:
```
# Handling Null Values
covid_world.describe().T
```

Out[29]:

	count	mean	min	25%	50%	75%	max	std
date	1684	2022-04-25 12:00:00.000000256	2020-01-05 00:00:00	2021-02-28 18:00:00	2022-04-25 12:00:00	2023-06-20 06:00:00	2024-08-14 00:00:00	NaN
population	1684.0	7975105024.0	7975105024.0	7975105024.0	7975105024.0	7975105024.0	7975105024.0	0.0
total_cases	1674.0	427537145.818996	2.0	110747235.0	503419024.0	766875788.0	775866783.0	308144071.644467
new_cases	1674.0	463521.539427	0.0	0.0	0.0	0.0	44236227.0	2189362.375056
total_deaths	1674.0	4784922.930108	3.0	2620744.0	6241478.0	6948012.0	7057132.0	2523699.408514
new_deaths	1674.0	4218.033453	0.0	0.0	0.0	0.0	103719.0	15119.676972
total_vaccinations	1352.0	10092511846.466717	0.0	7246320823.25	12804674096.5	13536930110.75	13578774356.0	4722349564.887587
new_vaccinations	1346.0	10079698.562407	0.0	273110.75	4193852.5	16312893.5	49673198.0	12971208.233496
people_fully_vaccinated	1341.0	3917015681.369128	9669.0	3179289407.0	4965268861.0	5170566932.0	5177942957.0	1843131085.372163
people_vaccinated	1352.0	4386874584.881657	0.0	3944208626.5	5386799865.0	5620701222.0	5631263739.0	1884966060.451264
total_boosters	1325.0	1820061464.669434	1.0	238849578.0	2563574171.0	2798123643.0	2817381093.0	1177024209.56291

In [30]:
```
# Analyzing the percentage of null values in covid world data
percent = (covid_world.isnull().sum() * 100)/covid_world.shape[0]
percent
```

Out[30]:
```
date                        0.000000
population                  0.000000
total_cases                 0.593824
new_cases                   0.593824
total_deaths                0.593824
new_deaths                  0.593824
total_vaccinations         19.714964
new_vaccinations           20.071259
people_fully_vaccinated    20.368171
people_vaccinated          19.714964
total_boosters             21.318290
dtype: float64
```

Observation

- Since the percentage of null values is very small, we have 2 options:
  - Drop all the null values
  - Replace the null values with median values

Conclusion

- Since the number of null values is very small, I will proceed by `replacing with median values`.

In [31]:
```
# Replacing Null values of each column with its corresponding median value
covid_world.fillna(covid_world.median(), inplace = True)

# Checking the overall info after null value treatment
covid_world.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1684 entries, 422729 to 424412
Data columns (total 11 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   date                     1684 non-null   datetime64[ns]
 1   population               1684 non-null   int64
 2   total_cases              1684 non-null   float64
 3   new_cases                1684 non-null   float64
 4   total_deaths             1684 non-null   float64
 5   new_deaths               1684 non-null   float64
 6   total_vaccinations       1684 non-null   float64
 7   new_vaccinations         1684 non-null   float64
 8   people_fully_vaccinated  1684 non-null   float64
 9   people_vaccinated        1684 non-null   float64
 10  total_boosters           1684 non-null   float64
dtypes: datetime64[ns](1), float64(9), int64(1)
memory usage: 157.9 KB
```

In [32]:
```
covid_world.describe().T
```

Out[32]:

	count	mean	min	25%	50%	75%	max	std
date	1684	2022-04-25 12:00:00.000000256	2020-01-05 00:00:00	2021-02-28 18:00:00	2022-04-25 12:00:00	2023-06-20 06:00:00	2024-08-14 00:00:00	NaN
population	1684.0	7975105024.0	7975105024.0	7975105024.0	7975105024.0	7975105024.0	7975105024.0	0.0
total_cases	1684.0	427987750.796318	2.0	113408711.0	503419024.0	766665346.25	775866783.0	307282591.790659
new_cases	1684.0	460769.036223	0.0	0.0	0.0	0.0	44236227.0	2183139.000768
total_deaths	1684.0	4793572.307007	3.0	2684750.0	6241478.0	6946220.25	7057132.0	2518679.444012
new_deaths	1684.0	4192.985748	0.0	0.0	0.0	0.0	103719.0	15078.176366
total_vaccinations	1684.0	10627213667.732185	0.0	9991710242.0	12804674096.5	13497789019.5	13578774356.0	4366509414.439718
new_vaccinations	1684.0	8898335.160333	0.0	439382.25	4193852.5	9957669.75	49673198.0	11833124.695385
people_fully_vaccinated	1684.0	4130525681.733373	9669.0	4205513777.5	4965268861.0	5153867245.75	5177942957.0	1697975054.342641
people_vaccinated	1684.0	4584009497.589073	0.0	4784461186.5	5386799865.0	5604425460.25	5631263739.0	1735091080.574934
total_boosters	1684.0	1978565658.002375	1.0	1294502086.75	2563574171.0	2786295236.25	2817381093.0	1087497725.786207

Observation

- The columns `new_cases` and `new _deaths` have 0 as their minimum value which can cause `ZeroDivisionError` down the line, so while selecting datapoints, the zero values must be filtered out.

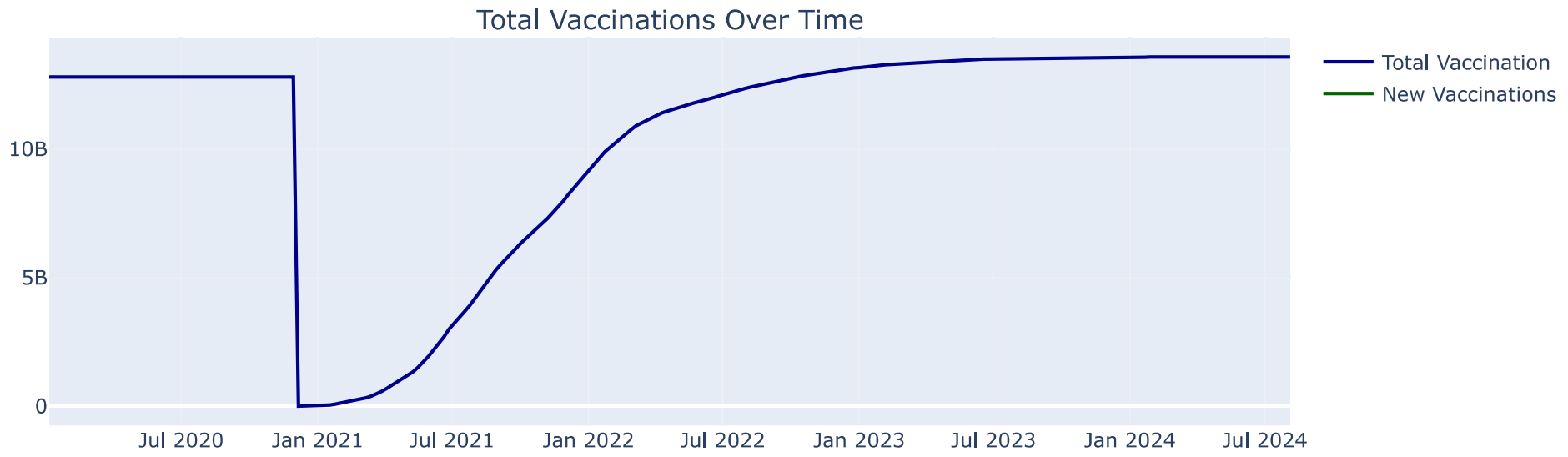
In [33]:
```
# Filtering Datapoints
covid_world_filtered = covid_world.query('new_cases!=0 and new_deaths!=0')

# General information on filtered data
covid_world_filtered.info()
```

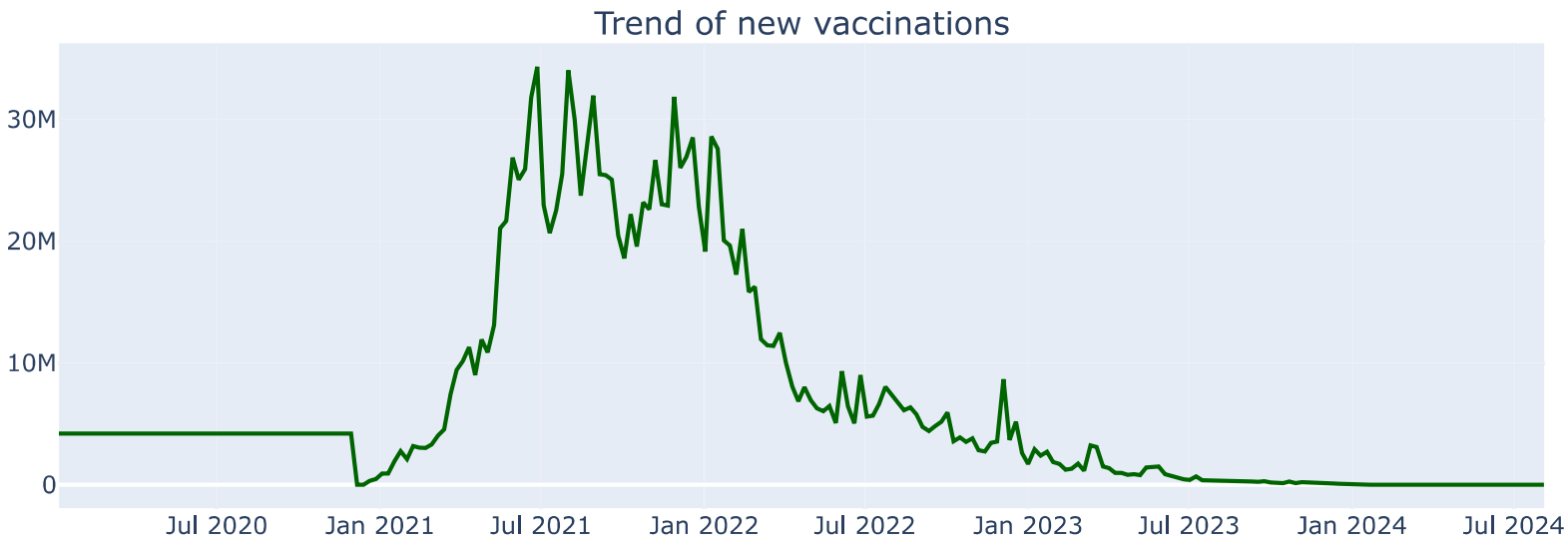
```
<class 'pandas.core.frame.DataFrame'>
Index: 240 entries, 422729 to 424402
Data columns (total 11 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   date                     240 non-null    datetime64[ns]
 1   population               240 non-null    int64
 2   total_cases              240 non-null    float64
 3   new_cases                240 non-null    float64
 4   total_deaths             240 non-null    float64
 5   new_deaths               240 non-null    float64
 6   total_vaccinations       240 non-null    float64
 7   new_vaccinations         240 non-null    float64
 8   people_fully_vaccinated  240 non-null    float64
 9   people_vaccinated        240 non-null    float64
 10  total_boosters           240 non-null    float64
dtypes: datetime64[ns](1), float64(9), int64(1)
memory usage: 22.5 KB
```

Making plots to validate hypotheis

In [34]:

```
# Plot the trend of total vaccinations over time as well as comparing the trend of new vaccinations and fully vaccinated people

# Defining a canvas for 2 subplots
fig = make_subplots(rows = 2,
                    cols = 1,
                    subplot_titles = ('Total Vaccinations Over Time',
                                      'Trend of new vaccinations'))



# Adding plots in canvas
fig.add_trace(go.Scatter(x = covid_world_filtered["date"],
                         y = covid_world_filtered['total_vaccinations'],
                         mode = 'lines',
                         name = 'Total Vaccination',
                         line = dict(color='darkblue')),
              row = 1,
             col = 1)

fig.add_trace(go.Scatter(x = covid_world_filtered["date"],
                         y = covid_world_filtered['new_vaccinations'],
                         mode = 'lines',
                         name = 'New Vaccinations',
                         line = dict(color='darkgreen')),
              row = 2,
              col = 1)

fig.update_layout(width=1000, height=800)

fig.show()
```





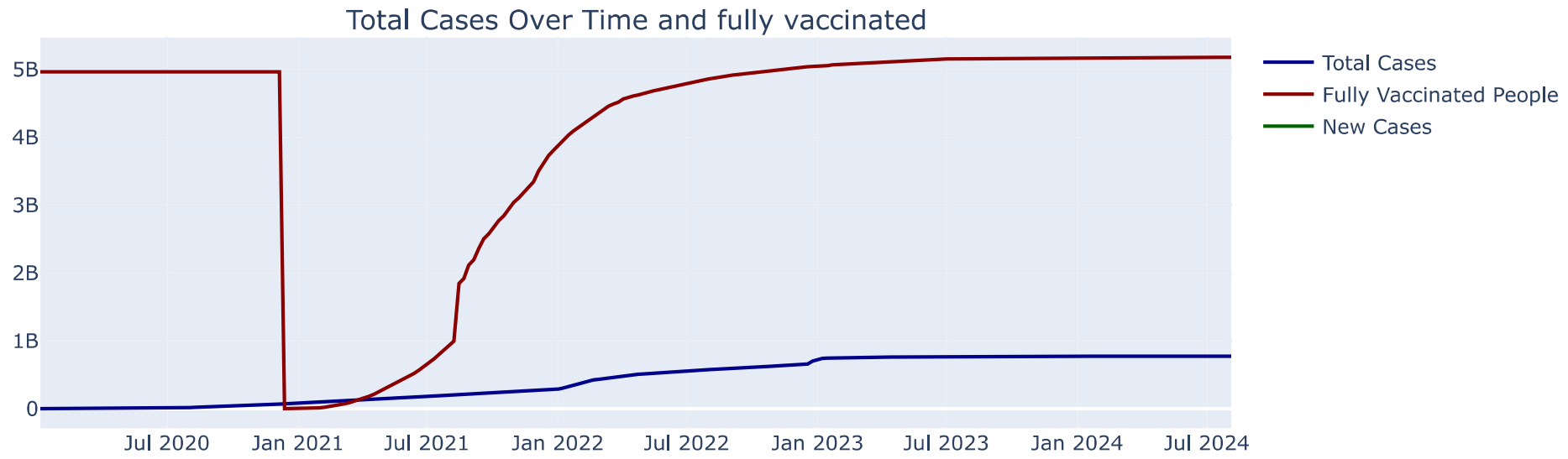
In [35]:
```
# Analyzing the trend of people fully vaccinated

figure = px.line(covid_world_filtered,
                 x = 'date',
                 y = 'people_fully_vaccinated',
                 title = "Fully Vaccinated Trend")

figure.show()
```

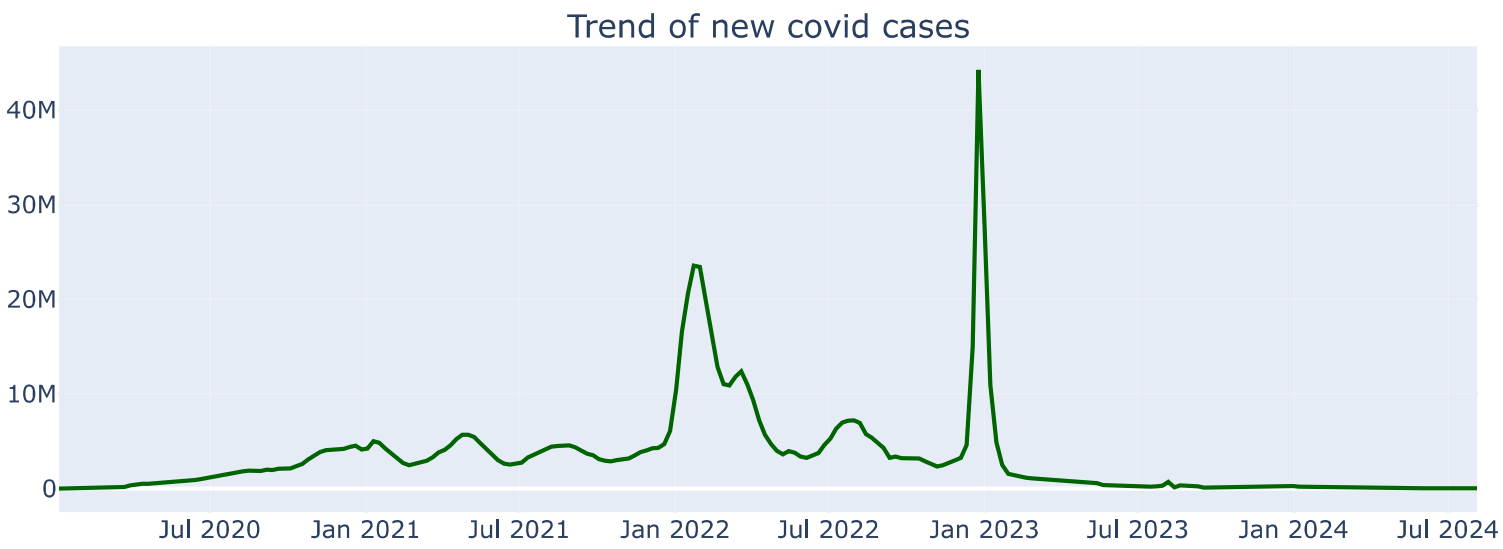
Observation

From the above plots we can infer that:

- Total Vaccinations have increased over time as more people got aware around the world.
- New Vaccination does see a spike during the covid waves but then dies down as the waves pass as the danger of covid reduced.
- After the arrival COVID-10, with the passage of time the number of fully vaccinated people has increased steadily.
- The Straight lines which are visible in the above plots is due to the fact that all those values have been replaced with median values, so thos eportions of graph are ignored while taking above inferences

In [36]:
```
# The trend of vaccination is good but now lets see how it affects covid cases

# Defining a canvas for 2 subplots
fig = make_subplots(rows = 2,
                    cols = 1,
                    subplot_titles = ('Total Cases Over Time and fully vaccinated',
                                      'Trend of new covid cases'))


# Adding plots in canvas
fig.add_trace(go.Scatter(x = covid_world_filtered["date"],
                         y = covid_world_filtered['total_cases'],
                         mode = 'lines',
                         name = 'Total Cases',
                         line = dict(color='darkblue')),
              row = 1,
             col = 1)

fig.add_trace(go.Scatter(x = covid_world_filtered["date"],
                         y = covid_world_filtered['people_fully_vaccinated'],
                         mode = 'lines',
                         name = 'Fully Vaccinated People',
                         line = dict(color='darkred')),
              row = 1,
             col = 1)

fig.add_trace(go.Scatter(x = covid_world_filtered["date"],
                         y = covid_world_filtered['new_cases'],
                         mode = 'lines',
                         name = 'New Cases',
                         line = dict(color='darkgreen')),
              row = 2,
              col = 1)

fig.update_layout(width=1000, height=800)

fig.show()
```





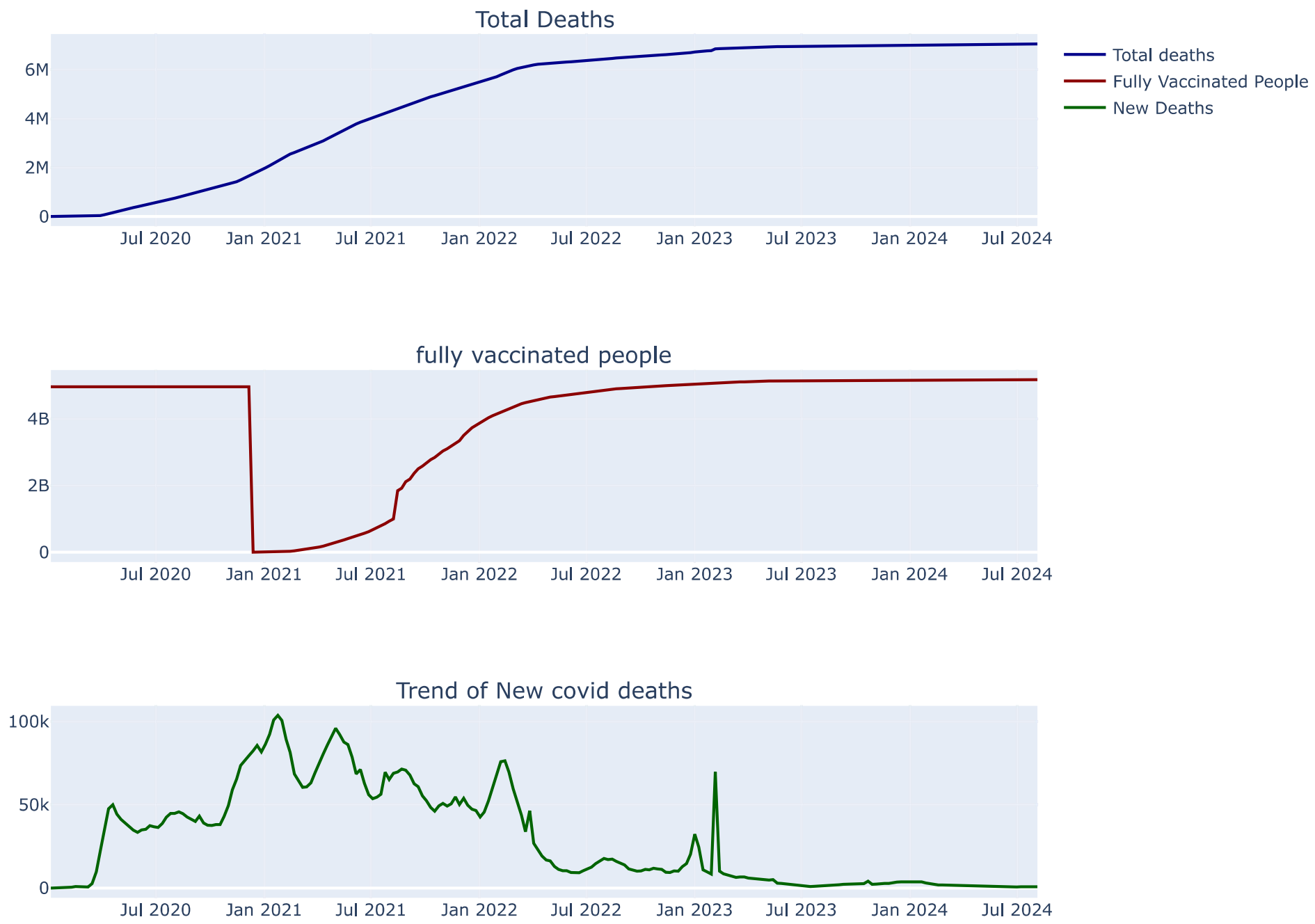
Observation

From the above plots we can infer the following:

- As the total number of fully vaccinated people began increasing it caused the total number of cases to stop increasing and become stagnant.
- Though the number of new covid cases do spike up during the first 2 waves, but die down eventually, and upon careful observation we also see that though the second wave spike was much bigger, it also died very quickly which can be attributed to the increase in the number of people who got fully vaccinated against covid 19.
- In the above plots the parallel line indicates the median values which were used to fill the null values.

In [37]:
```
# Analyzing the trend of deaths with regards to time and vaccinations

# Defining a canvas for 2 subplots
fig = make_subplots(rows = 3,
                    cols = 1,
                    subplot_titles = ('Total Deaths',
                                      'fully vaccinated people',
                                     'Trend of New covid deaths'))


# Adding plots in canvas
fig.add_trace(go.Scatter(x = covid_world_filtered["date"],
                         y = covid_world_filtered['total_deaths'],
                         mode = 'lines',
                         name = 'Total deaths',
                         line = dict(color='darkblue')),
              row = 1,
             col = 1)

fig.add_trace(go.Scatter(x = covid_world_filtered["date"],
                         y = covid_world_filtered['people_fully_vaccinated'],
                         mode = 'lines',
                         name = 'Fully Vaccinated People',
                         line = dict(color='darkred')),
              row = 2,
             col = 1)

fig.add_trace(go.Scatter(x = covid_world_filtered["date"],
                         y = covid_world_filtered['new_deaths'],
                         mode = 'lines',
                         name = 'New Deaths',
                         line = dict(color='darkgreen')),
```

```
                row = 3,
                col = 1)

fig.update_layout(width=1000, height=800)

fig.show()
```



Observation

From the above plots we can infer the following:

- The number of new covid deaths is higher during the initial covid waves, but post the second wave when majority of the people got vaccinated against covid-19 the number of new covid deaths has gone down.
- Also as the number of fully vaccinated people increases, the total death becomes stagnant at a fixed value

Hypothesis Conclusion

- **Based on the above plots and observations, we can confidently say that our hypothesis is supported by data and is valid.**

D) Analyzing the time taken by the covid 19 virus to kill a person

The above task is not a straight forward and needs to be performed as a series of steps

1. The columns needed to analyze the problem statement are: `date`, `location` and `total_deaths`.
2. Cleaning the data for null values.
3. Take the difference of first death day and last death day to approximate the time taken by the virus to kill.
4. Now since the data is entered agrregately over countries, we can take the aggregate difference over all the countries to find out how long does the virus take to kill a person.

In [38]:
```
# Filtering out Location, total_deaths and date columns
covid_death_data = countries_data[['location','total_deaths','date']].copy()

# Converting the date column to proper date time format
covid_death_data['date'] = pd.to_datetime(covid_death_data['date'])

# Dropping Null Values
covid_death_data=covid_death_data.dropna()
```

In [39]:
```
# Filtering out first and last death
first_death_record = covid_death_data.groupby('location').first().reset_index()
last_death_record = covid_death_data.groupby('location').last().reset_index()

# Renaming columns based on starting and end dates
first_death_record.rename(columns={'total_deaths': 'first_total_deaths', 'date': 'first_date'}, inplace=True)
last_death_record.rename(columns={'total_deaths': 'last_total_deaths', 'date': 'last_date'}, inplace=True)

# Merging the data to create a death_record dataframe
death_record = pd.merge(first_death_record, last_death_record, on='location')

# Calculating the difference in date and getting days
death_record['death_difference'] = death_record['last_total_deaths'] - death_record['first_total_deaths']
death_record['date_difference'] = (death_record['last_date'] - death_record['first_date']).dt.days
```

In [40]:
```
# Total days to required by covid to kill a patient based on country and overall average

# Selecting all the non-zero values
death_record['days_per_death'] = np.where(
    death_record['death_difference'] != 0,
    death_record['date_difference'] / death_record['death_difference'],
    0)

# Taking the average time
avg = death_record['days_per_death'].sum() / len(death_record['days_per_death'])

print(f"Avg time for virus to kill a person: {avg:.2f} days")
```

```
Avg time for virus to kill a person: 33.22 days
```

Observation

- Based on the data given, on an average if a person does not recover from COVID-19, they will die on an average between 33 to 34 days.

E) Misceallaneous Findings

Problem Statement: Analyze the mortality rate of countries

- To solve the above problem, first mortality rate needs to be defined.
- **Mortality Rate** : `It is calculated as total number of deaths divided by total number of covid cases`

In [41]:
```
# Filtering out unique dates from the data
dates_frame = data['date'].drop_duplicates().sort_values().reset_index(drop=True)

# Convert the dates to a list of strings
dates_list = dates_frame.astype(str).tolist()
```

In [42]:
```
# Creating the mortality dataframe
df_mortality = data[data['total_cases'] != 0.0].copy()
df_mortality['mortality'] = df_mortality['total_deaths'] / df_mortality['total_cases']

df_mortality.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 400131 entries, 56 to 429434
Data columns (total 68 columns):
 #   Column                                    Non-Null Count   Dtype
---  ------                                    --------------   -----
 0   iso_code                                  400131 non-null  object
 1   continent                                 373823 non-null  object
 2   location                                  400131 non-null  object
 3   date                                      400131 non-null  datetime64[ns]
 4   total_cases                               382500 non-null  float64
 5   new_cases                                 380855 non-null  float64
 6   new_cases_smoothed                        380815 non-null  float64
 7   total_deaths                              382500 non-null  float64
 8   new_deaths                                381304 non-null  float64
 9   new_deaths_smoothed                       381264 non-null  float64
 10  total_cases_per_million                   382500 non-null  float64
 11  new_cases_per_million                     380855 non-null  float64
 12  new_cases_smoothed_per_million            380815 non-null  float64
 13  total_deaths_per_million                  382500 non-null  float64
 14  new_deaths_per_million                    381304 non-null  float64
 15  new_deaths_smoothed_per_million           381264 non-null  float64
 16  reproduction_rate                         184137 non-null  float64
 17  icu_patients                              39083 non-null   float64
 18  icu_patients_per_million                  39083 non-null   float64
 19  hosp_patients                             40632 non-null   float64
 20  hosp_patients_per_million                 40632 non-null   float64
 21  weekly_icu_admissions                     10983 non-null   float64
 22  weekly_icu_admissions_per_million         10983 non-null   float64
 23  weekly_hosp_admissions                    24487 non-null   float64
 24  weekly_hosp_admissions_per_million        24487 non-null   float64
 25  total_tests                               78956 non-null   float64
 26  new_tests                                 75063 non-null   float64
 27  total_tests_per_thousand                  78956 non-null   float64
 28  new_tests_per_thousand                    75063 non-null   float64
 29  new_tests_smoothed                        102721 non-null  float64
 30  new_tests_smoothed_per_thousand           102721 non-null  float64
 31  positive_rate                             95789 non-null   float64
 32  tests_per_case                            94286 non-null   float64
 33  tests_units                               105326 non-null  object
 34  total_vaccinations                        85314 non-null   float64
 35  people_vaccinated                         81032 non-null   float64
 36  people_fully_vaccinated                   77990 non-null   float64
 37  total_boosters                            53594 non-null   float64
 38  new_vaccinations                          70969 non-null   float64
 39  new_vaccinations_smoothed                 191314 non-null  float64
 40  total_vaccinations_per_hundred            85314 non-null   float64
 41  people_vaccinated_per_hundred             81032 non-null   float64
 42  people_fully_vaccinated_per_hundred       77990 non-null   float64
 43  total_boosters_per_hundred                53594 non-null   float64
 44  new_vaccinations_smoothed_per_million     191314 non-null  float64
 45  new_people_vaccinated_smoothed            188462 non-null  float64
 46  new_people_vaccinated_smoothed_per_hundred  188462 non-null  float64
 47  stringency_index                          181856 non-null  float64
 48  population_density                        337453 non-null  float64
 49  median_age                                315565 non-null  float64
 50  aged_65_older                             304592 non-null  float64
 51  aged_70_older                             312350 non-null  float64
 52  gdp_per_capita                            308705 non-null  float64
 53  extreme_poverty                           201615 non-null  float64
 54  cardiovasc_death_rate                     309081 non-null  float64
 55  diabetes_prevalence                       323985 non-null  float64
 56  female_smokers                            234670 non-null  float64
 57  male_smokers                              231455 non-null  float64
 58  handwashing_facilities                    152381 non-null  float64
 59  hospital_beds_per_thousand                273243 non-null  float64
 60  life_expectancy                           362409 non-null  float64
 61  human_development_index                   301444 non-null  float64
 62  population                                400131 non-null  int64
 63  excess_mortality_cumulative_absolute      12872 non-null   float64
 64  excess_mortality_cumulative               12872 non-null   float64
 65  excess_mortality                          12872 non-null   float64
 66  excess_mortality_cumulative_per_million   12872 non-null   float64
 67  mortality                                 382500 non-null  float64
dtypes: datetime64[ns](1), float64(62), int64(1), object(4)
memory usage: 210.6+ MB
```

In [43]:
```
# Removing null values from mortality as they are of no use to us
df_mortality.dropna(subset = ['mortality'], inplace=True)
```

In [44]:
```
# Removing all grouped countries from the data and the remaining null values
grouped_locations = ['World',
                     'Lower-middle-income countries',
                     'Upper-middle-income countries',
                     'High-income countries',
                     'Low-income countries',
                     'Asia',
                     'Africa',
                     'Europe',
                     'European Union (27)',
                     'North America',
                     'South America']

df_mortality = df_mortality[~df_mortality["location"].isin(grouped_locations)]
```

```
df_mortality.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 364282 entries, 56 to 429434
Data columns (total 68 columns):
 #   Column                                    Non-Null Count   Dtype
---  ------                                    --------------   -----
 0   iso_code                                  364282 non-null  object
 1   continent                                 362629 non-null  object
 2   location                                  364282 non-null  object
 3   date                                      364282 non-null  datetime64[ns]
 4   total_cases                               364282 non-null  float64
 5   new_cases                                 362637 non-null  float64
 6   new_cases_smoothed                        362627 non-null  float64
 7   total_deaths                              364282 non-null  float64
 8   new_deaths                                363086 non-null  float64
 9   new_deaths_smoothed                       363076 non-null  float64
 10  total_cases_per_million                   364282 non-null  float64
 11  new_cases_per_million                     362637 non-null  float64
 12  new_cases_smoothed_per_million            362627 non-null  float64
 13  total_deaths_per_million                  364282 non-null  float64
 14  new_deaths_per_million                    363086 non-null  float64
 15  new_deaths_smoothed_per_million           363076 non-null  float64
 16  reproduction_rate                         181351 non-null  float64
 17  icu_patients                              34775 non-null   float64
 18  icu_patients_per_million                  34775 non-null   float64
 19  hosp_patients                             35401 non-null   float64
 20  hosp_patients_per_million                 35401 non-null   float64
 21  weekly_icu_admissions                     10974 non-null   float64
 22  weekly_icu_admissions_per_million         10974 non-null   float64
 23  weekly_hosp_admissions                    19644 non-null   float64
 24  weekly_hosp_admissions_per_million        19644 non-null   float64
 25  total_tests                               77959 non-null   float64
 26  new_tests                                 74153 non-null   float64
 27  total_tests_per_thousand                  77959 non-null   float64
 28  new_tests_per_thousand                    74153 non-null   float64
 29  new_tests_smoothed                        100635 non-null  float64
 30  new_tests_smoothed_per_thousand           100635 non-null  float64
 31  positive_rate                             94519 non-null   float64
 32  tests_per_case                            93043 non-null   float64
 33  tests_units                               103202 non-null  object
 34  total_vaccinations                        65889 non-null   float64
 35  people_vaccinated                         61935 non-null   float64
 36  people_fully_vaccinated                   58955 non-null   float64
 37  total_boosters                            37248 non-null   float64
 38  new_vaccinations                          52140 non-null   float64
 39  new_vaccinations_smoothed                 169006 non-null  float64
 40  total_vaccinations_per_hundred            65889 non-null   float64
 41  people_vaccinated_per_hundred             61935 non-null   float64
 42  people_fully_vaccinated_per_hundred       58955 non-null   float64
 43  total_boosters_per_hundred                37248 non-null   float64
 44  new_vaccinations_smoothed_per_million     169006 non-null  float64
 45  new_people_vaccinated_smoothed            166368 non-null  float64
 46  new_people_vaccinated_smoothed_per_hundred  166368 non-null  float64
 47  stringency_index                          177926 non-null  float64
 48  population_density                        331849 non-null  float64
 49  median_age                                309006 non-null  float64
 50  aged_65_older                             299382 non-null  float64
 51  aged_70_older                             305791 non-null  float64
 52  gdp_per_capita                            303495 non-null  float64
 53  extreme_poverty                           198869 non-null  float64
 54  cardiovasc_death_rate                     304972 non-null  float64
 55  diabetes_prevalence                       319570 non-null  float64
 56  female_smokers                            231909 non-null  float64
 57  male_smokers                              228694 non-null  float64
 58  handwashing_facilities                    149670 non-null  float64
 59  hospital_beds_per_thousand                270482 non-null  float64
 60  life_expectancy                           355456 non-null  float64
 61  human_development_index                   297029 non-null  float64
 62  population                                364282 non-null  int64
 63  excess_mortality_cumulative_absolute      12754 non-null   float64
 64  excess_mortality_cumulative               12754 non-null   float64
 65  excess_mortality                          12754 non-null   float64
 66  excess_mortality_cumulative_per_million   12754 non-null   float64
 67  mortality                                 364282 non-null  float64
dtypes: datetime64[ns](1), float64(62), int64(1), object(4)
memory usage: 191.8+ MB
```

In [45]:
```
# Iterate over dates from 25:922 slice as before and after these dates the covid cases cease to exist so it gives empty results
this_day_top_10 = ""
for i, this_day in enumerate(dates_list[25:922]):

    # For each date selecting the top 10 high mortality country
    this_day_top_10 = df_mortality[df_mortality['date'] == this_day].sort_values(by='mortality', ascending=False).head(10)

    if i == 0:

        # List comprehension to selection location and mortality on the given date
        ct_list = [(row['location'], row['mortality']) for _, row in this_day_top_10.iterrows()]

        print(f"During {this_day}, the top 10 countries with the highest mortality rate were:")
```

```
        for country, instance in ct_list:
            print(f"{country}, with mortality rate {100 * instance:.2f}%.",end="\n\n")

        print(end="\n\n\n\n")

        # Tracks the current top 10 countries with highest mortality rate
        new_set = set(row['location'] for _, row in this_day_top_10.iterrows())

    elif i == len(dates_list[25:922]) - 1:

        ct_list = [(row['location'], row['mortality']) for _, row in this_day_top_10.iterrows()]

        print(f"During {this_day}, the top 10 countries with the highest mortality rate were:")

        for country, instance in ct_list:
            print(f"{country}, with mortality rate {100 * instance:.2f}%.",end="\n\n")

    else:
        # old set tracks the previous top 10 countries with highest mortality rate
        new_set = set(row['location'] for _, row in this_day_top_10.iterrows())

        # If the top 10 countries have changed with possage of time
        if new_set != old_set:

            # Countries Replaced
            left_out = old_set - new_set

            # # Set of countries which replaced the previous countries
            new_additions = new_set - old_set

            print(f"This was the top ten until {this_day}, when {', '.join(new_additions)} joined the list, replacing {', '.join(left_out)}",
                  end="\n\n")

        # Updating new set with empty set to take in fresh new top 10 and old_set is updated with current new_set as it will be old now
        new_set, old_set = set(), new_set
```

During 2020-01-26, the top 10 countries with the highest mortality rate were:
Germany, with mortality rate 300.00%.

China, with mortality rate 2.82%.

Canada, with mortality rate 0.00%.

Australia, with mortality rate 0.00%.

France, with mortality rate 0.00%.

Japan, with mortality rate 0.00%.

Malaysia, with mortality rate 0.00%.

Monaco, with mortality rate 0.00%.

Nepal, with mortality rate 0.00%.

Oceania, with mortality rate 0.00%.

This was the top ten until 2020-02-02, when Italy, Finland, Philippines, Cambodia, United Kingdom joined the list, replacing Monaco, Oceania, Japan, Nepal, Malaysia.

This was the top ten until 2020-02-09, when Iceland, India joined the list, replacing Italy, Australia.

This was the top ten until 2020-02-16, when Japan, Spain, Egypt joined the list, replacing Cambodia, Iceland, India.

This was the top ten until 2020-02-23, when Italy, South Korea, Iran, Australia joined the list, replacing France, Canada, Finland, Egypt.

This was the top ten until 2020-03-01, when Thailand, United States joined the list, replacing Australia, South Korea.

This was the top ten until 2020-03-08, when Argentina, Peru, Ireland, Oceania, Australia, Iraq joined the list, replacing Thailand, Spain, Iran, Germany, Japan, United Kingdom.

This was the top ten until 2020-03-15, when Cayman Islands, France, Guyana, San Marino, Sudan, Ukraine joined the list, replacing China, Peru, Ireland, Oceania, Australia, United States.

This was the top ten until 2020-03-22, when Puerto Rico, Curacao, Indonesia, Zimbabwe joined the list, replacing Ukraine, Iraq, Argentina, Philippines.

This was the top ten until 2020-03-29, when Peru, Nicaragua, Kenya joined the list, replacing San Marino, Puerto Rico, Indonesia.

This was the top ten until 2020-04-05, when Niger, Mauritania, Botswana, Gabon, United Kingdom joined the list, replacing Italy, Peru, Cayman Islands, Curacao, Kenya.

This was the top ten until 2020-04-12, when Sint Maarten (Dutch part), Bahamas, Belize, Northern Mariana Islands, Ethiopia joined the list, replacing Niger, Sudan, Nicaragua, Botswana, Gabon.

This was the top ten until 2020-04-19, when Sudan, Burundi, Democratic Republic of Congo joined the list, replacing Northern Mariana Islands, Guyana, Belize.

This was the top ten until 2020-04-26, when British Virgin Islands, Nicaragua joined the list, replacing Sudan, Burundi.

This was the top ten until 2020-05-03, when Montenegro, Sao Tome and Principe, Yemen joined the list, replacing Bahamas, British Virgin Islands, Sint Maarten (Dutch part).

This was the top ten until 2020-05-10, when Mexico, Sint Maarten (Dutch part) joined the list, replacing Ethiopia, Sao Tome and Principe.

This was the top ten until 2020-05-17, when Belgium joined the list, replacing Yemen.

This was the top ten until 2020-05-24, when Italy, Peru, Yemen joined the list, replacing Democratic Republic of Congo, Zimbabwe, Nicaragua.

This was the top ten until 2020-05-31, when Hungary joined the list, replacing Peru.

This was the top ten until 2020-06-21, when British Virgin Islands, Peru joined the list, replacing Mauritania, Montenegro.

This was the top ten until 2020-08-09, when Netherlands joined the list, replacing Sint Maarten (Dutch part).

This was the top ten until 2020-08-16, when Isle of Man joined the list, replacing Netherlands.

This was the top ten until 2020-08-23, when Netherlands joined the list, replacing British Virgin Islands.

This was the top ten until 2020-08-30, when Jersey joined the list, replacing Netherlands.

This was the top ten until 2020-09-06, when Netherlands joined the list, replacing Hungary.

This was the top ten until 2020-09-13, when Ecuador joined the list, replacing Netherlands.

This was the top ten until 2020-09-20, when Niger joined the list, replacing France.

This was the top ten until 2020-10-04, when Montserrat joined the list, replacing Belgium.

This was the top ten until 2020-10-11, when Chad joined the list, replacing Jersey.

This was the top ten until 2020-11-01, when Sudan, Bolivia joined the list, replacing Italy, United Kingdom.

This was the top ten until 2020-12-06, when Egypt joined the list, replacing Niger.

```
This was the top ten until 2020-12-20, when Syria joined the list, replacing Chad.

This was the top ten until 2021-01-24, when Guernsey joined the list, replacing Bolivia.

This was the top ten until 2021-01-31, when Bolivia joined the list, replacing Guernsey.

This was the top ten until 2021-02-07, when China joined the list, replacing Bolivia.

This was the top ten until 2021-03-14, when Bolivia joined the list, replacing Isle of Man.

This was the top ten until 2021-04-04, when Somalia joined the list, replacing Bolivia.

This was the top ten until 2021-05-30, when Bosnia and Herzegovina joined the list, replacing China.

This was the top ten until 2021-07-11, when China joined the list, replacing Ecuador.

This was the top ten until 2021-07-18, when Liberia joined the list, replacing China.

This was the top ten until 2021-07-25, when Ecuador joined the list, replacing Bosnia and Herzegovina.

This was the top ten until 2021-08-08, when Bosnia and Herzegovina joined the list, replacing Montserrat.

This was the top ten until 2021-08-29, when Afghanistan joined the list, replacing Bosnia and Herzegovina.

This was the top ten until 2021-12-26, when Bosnia and Herzegovina joined the list, replacing Liberia.

During 2022-07-10, the top 10 countries with the highest mortality rate were:
Yemen, with mortality rate 18.16%.

Sudan, with mortality rate 7.89%.

Peru, with mortality rate 5.83%.

Syria, with mortality rate 5.63%.

Mexico, with mortality rate 5.14%.

Somalia, with mortality rate 5.06%.

Egypt, with mortality rate 4.81%.

Afghanistan, with mortality rate 4.22%.

Bosnia and Herzegovina, with mortality rate 4.16%.

Liberia, with mortality rate 3.92%.
```

In [46]:
```
# Final top 10 countries based on mortality rate at the end of COVID-19 second wave
this_day_top_10
```

Out[46]:

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoothed	total_ca
425330	YEM	Asia	Yemen	2022-07-10	11832.0	8.0	1.143	2149.0	0.0	0.000	
366851	SDN	Africa	Sudan	2022-07-10	62813.0	109.0	15.571	4955.0	3.0	0.429	
301934	PER	South America	Peru	2022-07-10	3662685.0	32889.0	4698.429	213652.0	126.0	18.000	
373547	SYR	Asia	Syria	2022-07-10	55952.0	22.0	3.143	3150.0	0.0	0.000	
244615	MEX	North America	Mexico	2022-07-10	6365294.0	200105.0	28586.429	326875.0	315.0	45.000	
355133	SOM	Africa	Somalia	2022-07-10	26900.0	97.0	13.857	1361.0	0.0	0.000	
106401	EGY	Africa	Egypt	2022-07-10	514182.0	49.0	7.000	24725.0	1.0	0.143	
917	AFG	Asia	Afghanistan	2022-07-10	183219.0	576.0	82.286	7727.0	3.0	0.429	
47803	BIH	Europe	Bosnia and Herzegovina	2022-07-10	379674.0	829.0	118.429	15809.0	3.0	0.429	
211305	LBR	Africa	Liberia	2022-07-10	7504.0	3.0	0.429	294.0	0.0	0.000	

References and Resources

1. Cowid 19 Github: Used to understand data as well the readme file was used verbatim for the information on dataset.
2. Matplotlib module Documentation
3. Pandas Documentation
4. Numpy Documentation
5. Seaborn Documentation

6. Plotly Documentation

Credits

> Mathieu, E., Ritchie, H., Ortiz-Ospina, E. *et al.* A global database of COVID-19 vaccinations. *Nat Hum Behav* (2021). https://doi.org/10.1038/s41562-021-01122-8

The data produced by third parties and made available by *Our World in Data* is subject to the license terms from the original third-party authors. We will always indicate the original source of the data in our database, and you should always check the license of any such third-party data before use.

Authors

This data has been collected, aggregated, and documented by Edouard Mathieu, Hannah Ritchie, Lucas Rodés-Guirao, Cameron Appel, Daniel Gavrilov, Charlie Giattino, Joe Hasell, Bobbie Macdonald, Saloni Dattani, Diana Beltekian, Esteban Ortiz-Ospina, and Max Roser.

Our World in Data makes data and research on the world's largest problems understandable and accessible. Read more about our mission.