COVID-19 Explorary Data Analysis - Human vs Disease

In [3]:

```
# imports
import math
import numpy as np
import pandas as pd
import plotly.express as ex
import plotly.graph_objects as go
import plotly.offline as pyo
from datetime import datetime
# helpful modules
import fuzzywuzzy
from fuzzywuzzy import process
import chardet

# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) will list the files

pyo.init_notebook_mode()
```

```
C:\Users\lianq\miniconda3\lib\site-packages\fuzzywuzzy\fuzz.py:11: UserWarning:

Using slow pure-python SequenceMatcher. Install python-Levenshtein to remove this warning
```

1. load the data

In [4]:

```
# load data
vacc_df = pd.read_csv("COVID-19 World Vaccination Progress/country_vaccinations.csv")
summary_df = pd.read_csv("Covid-19 Global Dataset/worldometer_coronavirus_summary_data.csv
daily_df = pd.read_csv("Covid-19 Global Dataset/worldometer_coronavirus_daily_data.csv")
vacc_manu = pd.read_csv("COVID-19 World Vaccination Progress/country_vaccinations_by_manuf
```

In [5]:

```
summary_df.shape
```

Out[5]:

```
(221, 12)
```

In [6]:

```
daily_df.shape
```

Out[6]:

```
(152735, 7)
```

In [7]:

```
vacc_manu.shape
```

Out[7]:

```
(26172, 4)
```

In [8]:

```
vacc_df.shape
```

Out[8]:

```
(73009, 15)
```

In [9]:

```
vacc_df.tail()
```

Out[9]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_r
73004	Zimbabwe	ZWE	2022-01-21	7496882.0	4234640.0	3262242.0	1365
73005	Zimbabwe	ZWE	2022-01-22	7506786.0	4239537.0	3267249.0	990
73006	Zimbabwe	ZWE	2022-01-23	7512903.0	4242647.0	3270256.0	611
73007	Zimbabwe	ZWE	2022-01-24	7517985.0	4245063.0	3272922.0	508
73008	Zimbabwe	ZWE	2022-01-25	7525574.0	4248576.0	3276998.0	758

In [10]:

```
summary_df.head()
```

Out[10]:

	country	continent	total_confirmed	total_deaths	total_recovered	active_cases	serious_or_critical	total_cases_
0	Afghanistan	Asia	158275	7367.0	145750.0	5158.0	1124.0	
1	Albania	Europe	213257	3228.0	202077.0	7952.0	23.0	
2	Algeria	Africa	220415	6310.0	151347.0	62758.0	34.0	
3	Andorra	Europe	25289	141.0	21511.0	3637.0	31.0	
4	Angola	Africa	86636	1789.0	67477.0	17370.0	7.0	

In [11]:

```
daily_df.head()
```

Out[11]:

	date	country	cumulative_total_cases	daily_new_cases	active_cases	cumulative_total_deaths	daily_new_deaths
0	2020-2-15	Afghanistan	0.0	NaN	0.0	0.0	NaN
1	2020-2-16	Afghanistan	0.0	NaN	0.0	0.0	NaN
2	2020-2-17	Afghanistan	0.0	NaN	0.0	0.0	NaN
3	2020-2-18	Afghanistan	0.0	NaN	0.0	0.0	NaN
4	2020-2-19	Afghanistan	0.0	NaN	0.0	0.0	NaN

In [12]:

```
vacc_manu.head()
```

Out[12]:

	location	date	vaccine	total_vaccinations
0	Austria	2021-01-08	Johnson&Johnson	0

	location	date	vaccine	total_vaccinations
1	Austria	2021-01-08	Moderna	0
2	Austria	2021-01-08	Oxford/AstraZeneca	0
3	Austria	2021-01-08	Pfizer/BioNTech	31530
4	Austria	2021-01-15	Johnson&Johnson	0

2.data cleaning

deal with nan

missing values: From the first rows I can see there are some NaN values.

In [13]:

```
missing_values_count_vacc = vacc_df.isnull().sum()
missing_values_count_vacc
# percent of data that is missing
total_cells_vacc = np.product(vacc_df.shape)
total_missing_vacc = missing_values_count_vacc.sum()
(total_missing_vacc/total_cells_vacc) * 100
```

Out[13]:

```
24.168070603167646
```

In [14]:

```
missing_values_count_summary= summary_df.isnull().sum()
missing_values_count_summary
# percent of data that is missing
total_cells_summary = np.product(summary_df.shape)
total_missing_summary = missing_values_count_summary.sum()
(total_missing_summary/total_cells_summary) * 100
```

Out[14]:

```
4.675716440422323
```

In [15]:

```
missing_values_count_daily= daily_df.isnull().sum()
missing_values_count_daily
# percent of data that is missing
total_cells_daily= np.product(daily_df.shape)
total_missing_daily= missing_values_count_daily.sum()
(total_missing_daily/total_cells_daily) * 100
```

Out[15]:

```
3.9756066763628883
```

In [16]:

```
missing_values_count_vacc_manu=vacc_manu.isnull().sum()
missing_values_count_vacc_manu
```

Out[16]:

```
location              0
date                  0
vaccine               0
total_vaccinations    0
dtype: int64
```

In [17]:

```
missing_values_count_vacc
```

Out[17]:

```
country                                0
iso_code                               0
date                                   0
```

```
total_vaccinations                     34800
people_vaccinated                      36755
people_fully_vaccinated                39521
daily_vaccinations_raw                 41795
daily_vaccinations                       363
total_vaccinations_per_hundred         34800
people_vaccinated_per_hundred          36755
people_fully_vaccinated_per_hundred    39521
daily_vaccinations_per_million           363
vaccines                                   0
source_name                                0
source_website                             0
dtype: int64
```

There are almost a quarter of the cells in **vaccination dataset** are empty(unavailable). total_vaccinations, people_vaccinated, people_fully_vaccinated, total_vaccinations_per_hundred , people_vaccinated_per_hundred, people_fully_vaccinated_per_hundred are columns that have a lot of unavailable values. ("for the data entry; for some of the dates we have only the daily vaccinations, for others, only the (cumulative) total") However, the reason is the number is not available, and there's may be still some information in the same row/column. **If the row has no not-null numeric values, I would drop the row. We have daily_vaccinations, so I would also drop the daily_vaccinations_raw column.**

I will keep all other rows and columns.

In [18]:

```
vacc_df = vacc_df.dropna(subset=['total_vaccinations','people_vaccinated','people_fully_va
                                'people_vaccinated_per_hundred','people_fully_vaccinated_p
```

In [19]:

```
vacc_df= vacc_df.drop('daily_vaccinations_raw', 1)
```

```
C:\Users\lianq\AppData\Local\Temp/ipykernel_23688/415337386.py:1: FutureWarning:

In a future version of pandas all arguments of DataFrame.drop except for the argument 'lab
els' will be keyword-only
```

In [20]:

```
vacc_df.shape #removed some rows (73009 before), removed 1 column
```

Out[20]:

```
(72887, 14)
```

date parsing

In [21]:

```
daily_df.date.dtype
```

Out[21]:

```
dtype('O')
```

In [22]:

```
vacc_df.date.dtype
```

Out[22]:

```
dtype('O')
```

Parse the two date columns into date format.

In [23]:

```
daily_df['date_parsed'] = pd.to_datetime(daily_df['date'], format = "%Y-%m-%d")
daily_df['date_parsed'].head()
```

```
0    2020-02-15
```

```
Out[23]: 1   2020-02-16
         2   2020-02-17
         3   2020-02-18
         4   2020-02-19
         Name: date_parsed, dtype: datetime64[ns]
```

```
In [24]:
vacc_df['date_parsed'] = pd.to_datetime(vacc_df['date'], format = "%Y-%m-%d")
vacc_df['date_parsed'].head()
```

```
Out[24]: 0   2021-02-22
         1   2021-02-23
         2   2021-02-24
         3   2021-02-25
         4   2021-02-26
         Name: date_parsed, dtype: datetime64[ns]
```

inconsistent-data-entry

Country Names - vacc df and the global covid data df

The country column is the only mutual column the two datasets have.

```
In [25]:
# get all the unique values in the 'country' column of the vacc df
vacc_country = vacc_df['country'].unique()
# sort them alphabetically and then take a closer look
vacc_country.sort()
vacc_country
```

```
Out[25]: array(['Afghanistan', 'Albania', 'Algeria', 'Andorra', 'Angola',
                'Anguilla', 'Antigua and Barbuda', 'Argentina', 'Armenia', 'Aruba',
                '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', 'Ecuador', 'Egypt',
                'El Salvador', 'England', 'Equatorial Guinea', 'Estonia',
                'Eswatini', 'Ethiopia', 'Faeroe Islands', 'Falkland Islands',
                'Fiji', 'Finland', 'France', 'French Polynesia', 'Gabon', 'Gambia',
                'Georgia', 'Germany', 'Ghana', 'Gibraltar', 'Greece', 'Greenland',
                'Grenada', 'Guatemala', 'Guernsey', 'Guinea', 'Guinea-Bissau',
                'Guyana', 'Haiti', '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', 'Luxembourg', 'Macao',
                'Madagascar', 'Malawi', 'Malaysia', 'Maldives', 'Mali', 'Malta',
                'Mauritania', 'Mauritius', 'Mexico', 'Moldova', 'Monaco',
                'Mongolia', 'Montenegro', 'Montserrat', 'Morocco', 'Mozambique',
                'Myanmar', 'Namibia', 'Nauru', 'Nepal', 'Netherlands',
                'New Caledonia', 'New Zealand', 'Nicaragua', 'Niger', 'Nigeria',
                'Niue', 'North Macedonia', 'Northern Cyprus', 'Northern Ireland',
                'Norway', 'Oman', 'Pakistan', 'Palestine', 'Panama',
                'Papua New Guinea', 'Paraguay', 'Peru', 'Philippines', 'Pitcairn',
                'Poland', 'Portugal', 'Qatar', 'Romania', 'Russia', 'Rwanda',
                'Saint Helena', 'Saint Kitts and Nevis', 'Saint Lucia',
                '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 Korea',
                'South Sudan', 'Spain', 'Sri Lanka', 'Sudan', 'Suriname', 'Sweden',
                'Switzerland', 'Syria', 'Taiwan', 'Tajikistan', 'Tanzania',
                'Thailand', 'Timor', 'Togo', 'Tokelau', 'Tonga',
                'Trinidad and Tobago', 'Tunisia', 'Turkey', 'Turkmenistan',
                'Turks and Caicos Islands', 'Tuvalu', 'Uganda', 'Ukraine',
                'United Arab Emirates', 'United Kingdom', 'United States',
                'Uruguay', 'Uzbekistan', 'Vanuatu', 'Venezuela', 'Vietnam',
                'Wales', 'Wallis and Futuna', 'Yemen', 'Zambia', 'Zimbabwe'],
               dtype=object)
```

In [26]:

```
# get all the unique values in the 'country' column of the vacc df
daily_country = daily_df['country'].unique()
# sort them alphabetically and then take a closer look
daily_country.sort()
daily_country
```

Out[26]:

```
array(['Afghanistan', 'Albania', 'Algeria', 'Andorra', 'Angola',
       'Anguilla', 'Antigua And Barbuda', 'Argentina', 'Armenia', 'Aruba',
       'Australia', 'Austria', 'Azerbaijan', 'Bahamas', 'Bahrain',
       'Bangladesh', 'Barbados', 'Belarus', 'Belgium', 'Belize', 'Benin',
       'Bermuda', 'Bhutan', 'Bolivia', 'Bosnia And Herzegovina',
       'Botswana', 'Brazil', 'British Virgin Islands',
       'Brunei Darussalam', 'Bulgaria', 'Burkina Faso', 'Burundi',
       'Cabo Verde', 'Cambodia', 'Cameroon', 'Canada',
       'Caribbean Netherlands', 'Cayman Islands',
       'Central African Republic', 'Chad', 'Channel Islands', 'Chile',
       'China', 'China Hong Kong Sar', 'China Macao Sar', 'Colombia',
       'Comoros', 'Congo', 'Costa Rica', 'Cote D Ivoire', 'Croatia',
       'Cuba', 'Curacao', 'Cyprus', 'Czech Republic',
       'Democratic Republic Of The Congo', 'Denmark', 'Djibouti',
       'Dominica', 'Dominican Republic', 'Ecuador', 'Egypt',
       'El Salvador', 'Equatorial Guinea', 'Eritrea', 'Estonia',
       'Ethiopia', 'Faeroe Islands', 'Falkland Islands Malvinas', 'Fiji',
       'Finland', 'France', 'French Guiana', 'French Polynesia', 'Gabon',
       'Gambia', 'Georgia', 'Germany', 'Ghana', 'Gibraltar', 'Greece',
       'Greenland', 'Grenada', 'Guadeloupe', 'Guatemala', 'Guinea',
       'Guinea Bissau', 'Guyana', 'Haiti', 'Holy See', 'Honduras',
       'Hungary', 'Iceland', 'India', 'Indonesia', 'Iran', 'Iraq',
       'Ireland', 'Isle Of Man', 'Israel', 'Italy', 'Jamaica', 'Japan',
       'Jordan', 'Kazakhstan', 'Kenya', 'Kuwait', 'Kyrgyzstan', 'Laos',
       'Latvia', 'Lebanon', 'Lesotho', 'Liberia', 'Libya',
       'Liechtenstein', 'Lithuania', 'Luxembourg', 'Macedonia',
       'Madagascar', 'Malawi', 'Malaysia', 'Maldives', 'Mali', 'Malta',
       'Marshall Islands', 'Martinique', 'Mauritania', 'Mauritius',
       'Mayotte', 'Mexico', 'Micronesia', 'Moldova', 'Monaco', 'Mongolia',
       'Montenegro', 'Montserrat', 'Morocco', 'Mozambique', 'Myanmar',
       'Namibia', 'Nepal', 'Netherlands', 'New Caledonia', 'New Zealand',
       'Nicaragua', 'Niger', 'Nigeria', 'Norway', 'Oman', 'Pakistan',
       'Panama', 'Papua New Guinea', 'Paraguay', 'Peru', 'Philippines',
       'Poland', 'Portugal', 'Qatar', 'Reunion', 'Romania', 'Russia',
       'Rwanda', 'Saint Barthelemy', 'Saint Helena',
       'Saint Kitts And Nevis', 'Saint Lucia', 'Saint Martin',
       'Saint Pierre And Miquelon', 'Saint Vincent And The Grenadines',
       'Samoa', 'San Marino', 'Sao Tome And Principe', 'Saudi Arabia',
       'Senegal', 'Serbia', 'Seychelles', 'Sierra Leone', 'Singapore',
       'Sint Maarten', 'Slovakia', 'Slovenia', 'Solomon Islands',
       'Somalia', 'South Africa', 'South Korea', 'South Sudan', 'Spain',
       'Sri Lanka', 'State Of Palestine', 'Sudan', 'Suriname',
       'Swaziland', 'Sweden', 'Switzerland', 'Syria', 'Taiwan',
       'Tajikistan', 'Tanzania', 'Thailand', 'Timor Leste', 'Togo',
       'Tonga', 'Trinidad And Tobago', 'Tunisia', 'Turkey',
       'Turks And Caicos Islands', 'UK', 'USA', 'Uganda', 'Ukraine',
       'United Arab Emirates', 'Uruguay', 'Uzbekistan', 'Vanuatu',
       'Venezuela', 'Viet Nam', 'Wallis And Futuna Islands',
       'Western Sahara', 'Yemen', 'Zambia', 'Zimbabwe'], dtype=object)
```

In [27]:

```
# Identify the differences
print("Countries in Vaccination Data not in Covid Data")
print([country for country in vacc_df.country.unique() if country not in daily_df.country.
print("Countries in Covid Data not in Vaccination Data ")
print([country for country in daily_df.country.unique() if country not in vacc_df.country.
```

```
Countries in Vaccination Data not in Covid Data
['Antigua and Barbuda', 'Bonaire Sint Eustatius and Saba', 'Bosnia and Herzegovina', 'Brun
ei', 'Cape Verde', 'Cook Islands', "Cote d'Ivoire", 'Czechia', 'Democratic Republic of Con
go', 'England', 'Eswatini', 'Falkland Islands', 'Guernsey', 'Guinea-Bissau', 'Hong Kong',
'Isle of Man', 'Jersey', 'Kiribati', 'Kosovo', 'Macao', 'Nauru', 'Niue', 'North Macedoni
a', 'Northern Cyprus', 'Northern Ireland', 'Palestine', 'Pitcairn', 'Saint Kitts and Nevi
s', 'Saint Vincent and the Grenadines', 'Sao Tome and Principe', 'Scotland', 'Sint Maarten
(Dutch part)', 'Timor', 'Tokelau', 'Trinidad and Tobago', 'Turkmenistan', 'Turks and Caico
s Islands', 'Tuvalu', 'United Kingdom', 'United States', 'Vietnam', 'Wales', 'Wallis and F
utuna']
Countries in Covid Data not in Vaccination Data
['Antigua And Barbuda', 'Bosnia And Herzegovina', 'Brunei Darussalam', 'Cabo Verde', 'Cari
bbean Netherlands', 'Channel Islands', 'China Hong Kong Sar', 'China Macao Sar', 'Cote D I
voire', 'Czech Republic', 'Democratic Republic Of The Congo', 'Eritrea', 'Falkland Islands
Malvinas', 'French Guiana', 'Guadeloupe', 'Guinea Bissau', 'Holy See', 'Isle Of Man', 'Mac
edonia', 'Marshall Islands', 'Martinique', 'Mayotte', 'Micronesia', 'Reunion', 'Saint Bart
helemy', 'Saint Kitts And Nevis', 'Saint Martin', 'Saint Pierre And Miquelon', 'Saint Vinc
ent And The Grenadines', 'Sao Tome And Principe', 'Sint Maarten', 'State Of Palestine', 'S
waziland', 'Timor Leste', 'Trinidad And Tobago', 'Turks And Caicos Islands', 'UK', 'USA',
'Viet Nam', 'Wallis And Futuna Islands', 'Western Sahara']
```

- First, there are some inconsistencies related to spelling and upper/lower cases: e.g. 'Bosnia and Herzegovina' and 'Bosnia And Herzegovina' should probably be the same. I used Fuzzy matching to deal with this problem.

In [28]:

```
def replace_matches_in_column(df, column, string_to_match, min_ratio = 90):
    # get a list of unique strings
    strings = df[column].unique()
    # get the top 10 closest matches to our input string
    matches = fuzzywuzzy.process.extract(string_to_match, strings,
                                         limit=10, scorer=fuzzywuzzy.fuzz.token_sort_ratio
    # only get matches with a ratio > 90
    close_matches = [matches[0] for matches in matches if matches[1] >= min_ratio]
    # get the rows of all the close matches in our dataframe
    rows_with_matches = df[column].isin(close_matches)
    #print(rows_with_matches)#true/false
    # replace all rows with close matches with the input matches
    df.loc[rows_with_matches, column] = string_to_match
    print("All done!")
```

In [29]:

```
for countryname in [country for country in daily_df.country.unique() if country not in vac
    replace_matches_in_column(df=vacc_df, column='country', string_to_match=countryname)
```

```
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
All done!
```

In [30]:

```
print("Countries in Vaccination Data not in Covid Data")
print([country for country in vacc_df.country.unique() if country not in daily_df.country.
print("Countries in Covid Data not in Vaccination Data ")
print([country for country in daily_df.country.unique() if country not in vacc_df.country.
```

```
Countries in Vaccination Data not in Covid Data
['Bonaire Sint Eustatius and Saba', 'Brunei', 'Cape Verde', 'Cook Islands', 'Czechia', 'En
gland', 'Eswatini', 'Falkland Islands', 'Guernsey', 'Hong Kong', 'Jersey', 'Kiribati', 'Ko
sovo', 'Macao', 'Nauru', 'Niue', 'North Macedonia', 'Northern Cyprus', 'Northern Ireland',
'Palestine', 'Pitcairn', 'Scotland', 'Sint Maarten (Dutch part)', 'Timor', 'Tokelau', 'Tur
kmenistan', 'Tuvalu', 'United Kingdom', 'United States', 'Vietnam', 'Wales', 'Wallis and F
utuna']
Countries in Covid Data not in Vaccination Data
['Brunei Darussalam', 'Cabo Verde', 'Caribbean Netherlands', 'Channel Islands', 'China Hon
g Kong Sar', 'China Macao Sar', 'Czech Republic', 'Eritrea', 'Falkland Islands Malvinas',
'French Guiana', 'Guadeloupe', 'Holy See', 'Macedonia', 'Marshall Islands', 'Martinique',
'Mayotte', 'Micronesia', 'Reunion', 'Saint Barthelemy', 'Saint Martin', 'Saint Pierre And
Miquelon', 'Sint Maarten', 'State Of Palestine', 'Swaziland', 'Timor Leste', 'UK', 'USA',
'Viet Nam', 'Wallis And Futuna Islands', 'Western Sahara']
```

- There are still some remaining inconsistencies needs to be corrected manually based on human-knowledge.

'Brunei'=='Brunei Darussalam'

'Cape Verde' == 'Cabo Verde'

'Czechia' == "Czech Republic"

'Hong Kong'=='China Hong Kong Sar'

'Macao'=='China Macao Sar'

'United Kingdom' == "UK"
'United States' == "USA"
'Eswatini'=='Swaziland'(renamed)
'Falkland Islands'=='Falkland Islands Malvinas'
'North Macedonia'=='Macedonia'(renamed)
'Palestine'=='State Of Palestine'
'Sint Maarten (Dutch part)'=='Saint Martin'
'Timor'== 'Timor Leste'
'Vietnam'=='Viet Nam'
'Wallis and Futuna'=='Wallis And Futuna Islands'

'Jersey'=='Channel Islands'
'Guernsey'=='Channel Islands'
(since they are parts of the Channel Islands)

'Bonaire Sint Eustatius and Saba'=='Caribbean Netherlands'
(since it is a part of the Caribbean Netherlands)

'England'== "UK"
'Wales'== "UK"
'Scotland'== "UK"
'Northern Ireland'== "UK"
(since they are parts of the UK)

In [31]:

```
vacc_df.country = vacc_df.country.replace().replace({
    'Brunei':'Brunei Darussalam',
    'Cape Verde':'Cabo Verde',
    'Czechia':"Czech Republic",
    'Hong Kong':'China Hong Kong Sar',
    'Macao':'China Macao Sar',
    'United Kingdom':"UK",
    'United States':"USA",
    'Eswatini':'Swaziland',
    'Falkland Islands':'Falkland Islands Malvinas',
    'North Macedonia':'Macedonia',
    'Palestine':'State Of Palestine',
    'Sint Maarten (Dutch part)':'Saint Martin',
    'Timor':'Timor Leste',
    'Vietnam':'Viet Nam',
    'Wallis and Futuna':'Wallis And Futuna Islands',
    'Jersey':'Channel Islands',
    'Guernsey':'Channel Islands',
    'Bonaire Sint Eustatius and Saba':'Caribbean Netherlands',
    'England':"UK",
    'Wales':"UK",
    'Scotland':"UK",
    'Northern Ireland':"UK"
})
```

3. Explore the dataset

In [32]:

```
vacc_df.tail()
```

Out[32]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations
73004	Zimbabwe	ZWE	2022-01-21	7496882.0	4234640.0	3262242.0	10405.0
73005	Zimbabwe	ZWE	2022-01-22	7506786.0	4239537.0	3267249.0	10567.0
73006	Zimbabwe	ZWE	2022-01-23	7512903.0	4242647.0	3270256.0	10631.0
73007	Zimbabwe	ZWE	2022-01-24	7517985.0	4245063.0	3272922.0	10273.0
73008	Zimbabwe	ZWE	2022-01-25	7525574.0	4248576.0	3276998.0	9579.0

In [33]:

```
vacc_df.dtypes
```

Out[33]:

```
country                                           object
iso_code                                          object
date                                              object
total_vaccinations                               float64
people_vaccinated                                float64
people_fully_vaccinated                          float64
daily_vaccinations                               float64
total_vaccinations_per_hundred                   float64
people_vaccinated_per_hundred                    float64
people_fully_vaccinated_per_hundred              float64
daily_vaccinations_per_million                   float64
vaccines                                          object
source_name                                       object
source_website                                    object
date_parsed                               datetime64[ns]
dtype: object
```

In [34]:

```
vacc_df.date.min()
```

Out[34]:

```
'2020-12-01'
```

In [35]:

```
vacc_df.date.max()
```

Out[35]:

```
'2022-01-25'
```

In [36]:

```
vacc_df.shape
```

Out[36]:

```
(72887, 15)
```

In [37]:

```
daily_df.tail()
```

Out[37]:

	date	country	cumulative_total_cases	daily_new_cases	active_cases	cumulative_total_deaths	daily_new_d
152730	2022-1-01	Zimbabwe	214214.0	956.0	26786.0	5017.0	

	date	country	cumulative_total_cases	daily_new_cases	active_cases	cumulative_total_deaths	daily_new_d
152731	2022-1-02	Zimbabwe	214878.0	664.0	26585.0	5032.0	
152732	2022-1-03	Zimbabwe	216087.0	1209.0	25446.0	5047.0	
152733	2022-1-04	Zimbabwe	217678.0	1591.0	24620.0	5078.0	
152734	2022-1-05	Zimbabwe	219057.0	1379.0	24252.0	5092.0	

In [38]:

```
daily_df.shape
```

Out[38]:

```
(152735, 8)
```

In [39]:

```
daily_df.dtypes
```

Out[39]:

```
date                                object
country                             object
cumulative_total_cases             float64
daily_new_cases                    float64
active_cases                       float64
cumulative_total_deaths            float64
daily_new_deaths                   float64
date_parsed                 datetime64[ns]
dtype: object
```

In [40]:

```
daily_df.date.min()
```

Out[40]:

```
'2020-1-22'
```

In [41]:

```
daily_df.date.max()
```

Out[41]:

```
'2022-1-05'
```

In [42]:

```
summary_df.tail()
```

Out[42]:

	country	continent	total_confirmed	total_deaths	total_recovered	active_cases	serious_or_critical	tota
216	Wallis And Futuna Islands	Australia/Oceania	454	7.0	438.0	9.0	NaN	
217	Western Sahara	Africa	10	1.0	8.0	1.0	NaN	
218	Yemen	Asia	10152	1986.0	7043.0	1123.0	23.0	
219	Zambia	Africa	274087	3782.0	236878.0	33427.0	317.0	
220	Zimbabwe	Africa	219057	5092.0	189713.0	24252.0	12.0	

In [43]:

```
summary_df.shape
```

Out[43]:

```
(221, 12)
```

In [44]:

```
summary_df.dtypes
```

Out[44]:

```
country                             object
continent                           object
total_confirmed                      int64
total_deaths                       float64
total_recovered                    float64
active_cases                       float64
serious_or_critical                float64
total_cases_per_1m_population        int64
total_deaths_per_1m_population     float64
total_tests                        float64
total_tests_per_1m_population      float64
population                           int64
dtype: object
```

In [45]:

```
vacc_manu.tail()
```

Out[45]:

	location	date	vaccine	total_vaccinations
26167	European Union	2022-01-25	Oxford/AstraZeneca	67354287
26168	European Union	2022-01-25	Pfizer/BioNTech	553562496
26169	European Union	2022-01-25	Sinopharm/Beijing	2264826
26170	European Union	2022-01-25	Sinovac	9
26171	European Union	2022-01-25	Sputnik V	1845079

In [46]:

```
vacc_manu.shape
```

Out[46]:

```
(26172, 4)
```

In [47]:

```
vacc_manu.dtypes
```

Out[47]:

```
location                object
date                    object
vaccine                 object
total_vaccinations       int64
dtype: object
```

(1) vaccination dataset

1. There are 72887 daily data in this dataset, ranges from 2020-12-01 to 2022-01-25.
2. 8 numerical columns:

- total_vaccinations - total immunizations in the country.
- people_vaccinated - total number of people who received at least one vaccine dose.
- people_fully_vaccinated - the number of people that received the entire set of immunization according to the immunization scheme (typically 2).
- daily_vaccinations - the number of vaccination for that date/country on the day.
- total_vaccinations_per_hundred - ratio (in percent) between vaccination number and total population up to the date in the country.
- people_vaccinated_per_hundred - ratio (in percent) between population immunized and total population up to the date in the country.

- people_fully_vaccinated_per_hundred - ratio (in percent) between population fully immunized and total population up to the date in the country.
- daily_vaccinations_per_million - ratio (in ppm) between vaccination number and total population for the current date in the country.

1. 6 categorical columns.

- country- this is the country for which the vaccination information is provided.
- iso_code- ISO code for the country.
- date - date for the data entry.
- vaccines - vaccines used in the country
- source_website - source of the information
- source_name - website of the source of information.

1. 1 date column: date_parsed: the parsed date.

(2) daily covid dataset

1. There are 152735 daily data in this dataset, ranges from 2020-1-22 to 2022-1-05.
2. 5 numerical columns:

- cumulative_total_cases - designates the cumulative number of confirmed cases as of the row's date, for the row's country.
- daily_new_cases - designates the daily new number of confirmed cases on the row's date, for the row's country.
- active_cases - designates the number of active cases (i.e., confirmed cases that still didn't recover nor die) on the row's date, for the row's country.
- cumulative_total_deaths - designates the cumulative number of confirmed deaths as of the row's date, for the row's country.
- daily_new_deaths - designates the daily new number of confirmed deaths on the row's date, for the row's country.

1. 2 categorical columns:

- date - the date of observation of the row's data in YYYY-MM-DD format.
- country - designates the Country in which the the row's data was observed.

1. 1 date column: date_parsed: the parsed date.

(3) covid summary dataset

1. There are 221 summary rows in this dataset.
2. 10 numerical columns:

- total_confirmed - The total number of confirmed cases in the observed country.
- total_deaths - The total number of confirmed deaths in the observed country.
- total_recovered - The total number of confirmed recoveries in the observed country.
- active_cases - The number of active cases in the observed country.
- serious_or_critical - The estimated number of cases in serious or critical conditions in the observed country.

- total_cases_per_1m_population - The number of total cases per 1 million population in the observed country.
- total_deaths_per_1m_population - The number of total deaths per 1 million population in the observed country.
- total_tests - The number of total tests done in the observed country.
- total_tests_per_1m_population - The number of total test done per 1 million population in the observed country.
- population - The population count in the observed country.

1. 2 categorical columns:

- country - designates the Country in which the the row's data was observed.
- continent - designates the Continent of the observed country.

(4) country vaccinations by manufacturer

1. There are 26172 records in this dataset.
2. 3 categorical columns:

- Location - country
- Date - date
- Vaccine - vaccine type

1. 1 numerical column:

- Total number of vaccinations - total number of vaccinations / current time and vaccine type.

4. data wrangling

(1) For Summary

Add the Numbers:

- Number of vaccine doses administered
- Total number of people vaccinated
- Total number of people fully vaccinated
- Number of vaccine doses administered per hundred population

Add the Categorical information:

- Vaccine combinations in use for each country

Calculate the Rates:

- Percentage of the total population fully vaccinated
- Percentage of the tested that result in positive
- Confirm rate: the percentage that a person would get covid
- Test rate: the percentage that a person was tested (test cover rate)
- Death rate: percentage of the confirmed that dead
- Recover rate: percentage of the confirmed that recovered
- Critical rate: percentage of the current active cases that are critical

In [48]:

```
#join the two tables using the index "country",#keep the vaccines column:combinations in u
summary = summary_df.set_index("country")
vaccines = vacc_df[['country', 'vaccines']].drop_duplicates().set_index('country')
summary = summary.join(vaccines)
```

In [49]:

```
#total number of vaccine doses administered
total_vaccinations=pd.DataFrame(vacc_df.groupby("country")['total_vaccinations'].max())
summary=summary.join(total_vaccinations)
```

In [50]:

```
#people vaccinated
people_vaccinated=pd.DataFrame(vacc_df.groupby("country")['people_vaccinated'].max())
summary=summary.join(people_vaccinated)
```

In [51]:

```
#people fully vaccinated
people_fully_vaccinated=pd.DataFrame(vacc_df.groupby("country")['people_fully_vaccinated']
summary=summary.join(people_fully_vaccinated)
```

In [52]:

```
#Number of vaccine doses administered per hundred population
total_vaccinations_per_hundred=pd.DataFrame(vacc_df.groupby("country")['total_vaccinations
summary=summary.join(total_vaccinations_per_hundred)
```

In [53]:

```
#Percentage of the total population fully vaccinated
summary['percentage_fully_vaccinated'] = summary.people_fully_vaccinated / summary.populat
summary['percentage_vaccinated'] = summary.people_vaccinated / summary.population * 100
```

In [54]:

```
#Percentage of the tested that result in positive
summary['tested_positive'] = summary.total_confirmed / summary.total_tests * 100
#Confirm rate: the percentage that a person would get covid
summary['confirm_rate'] = summary.total_confirmed / summary.population * 100
#Test rate: the rate that a person was tested (test cover rate)
#may greater than 100%
summary['test_rate'] = summary.total_tests / summary.population * 100
#Death rate: percentage of the confirmed that dead
summary['death_rate'] = summary.total_deaths / summary.total_confirmed * 100
#Recover rate: percentage of the confirmed that recovered
summary['recover_rate'] = summary.total_recovered / summary.total_confirmed * 100
#Critical rate: percentage of the current active cases that are critical
summary['critical_rate'] = summary.serious_or_critical/summary.active_cases * 100
```

In [55]:

```
#drop the columns that are duplicate in meaning(not for use)
summary=summary.drop(columns=['total_tests_per_1m_population', 'total_deaths_per_1m_popula
```

In [56]:

```
pd.set_option('display.max_columns', None)
summary.head(3)
```

Out[56]:

	continent	total_confirmed	total_deaths	total_recovered	active_cases	serious_or_critical	total_tests	p
country								
Afghanistan	Asia	158275	7367.0	145750.0	5158.0	1124.0	826810.0	

country	continent	total_confirmed	total_deaths	total_recovered	active_cases	serious_or_critical	total_tests	p
Albania	Europe	213257	3228.0	202077.0	7952.0	23.0	1495002.0	
Algeria	Africa	220415	6310.0	151347.0	62758.0	34.0	230861.0	

In [57]:

```python
# which country are using what kind of vacc
vaccine = vacc_df.vaccines.unique().tolist()
country = vacc_df.country.unique().tolist()
vaccine_country_df=pd.DataFrame(columns = ["vaccine"])
vaccine_country_df
for c in country:
    vaccines = "".join(sorted(list(set(list(vacc_df.loc[vacc_df.country==c,'vaccines'].val
    vaccine_country_df.loc[c,"vaccine"]=vaccines
vaccine_country_df=vaccine_country_df.reset_index()
vaccine_country_df.columns=["country","vaccine"]
vaccine_country_df.head(3)
```

Out[57]:

	country	vaccine
0	Afghanistan	Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
1	Albania	Oxford/AstraZeneca, Pfizer/BioNTech, Sinovac, ...
2	Algeria	Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...

In [58]:

```python
#Popularity of vaccination combinations
vaccine = vacc_df.vaccines.unique().tolist()
country = vacc_df.country.unique().tolist()
pop_count={}
for i in range(0,vaccine_country_df.shape[0]):
    v = vaccine_country_df.iloc[i].vaccine
    if v not in pop_count:
        pop_count[v]=1
    else:
        pop_count[v]+=1
pop_vacc_df= pd. DataFrame(list(pop_count.items()))
pop_vacc_df.columns=["vaccine","count"]
pop_vacc_df=pop_vacc_df.sort_values(by=['count'],ascending=False)
pop_vacc_df[pop_vacc_df['count']>=5]#11
head_pop_vacc_df=pop_vacc_df.head(11)
head_pop_vacc_df
```

Out[58]:

	vaccine	count
10	Johnson&Johnson, Moderna, Oxford/AstraZeneca, ...	24
4	Oxford/AstraZeneca	23
3	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	10
5	Oxford/AstraZeneca, Pfizer/BioNTech	10
18	Moderna, Pfizer/BioNTech	8
9	Pfizer/BioNTech	7

	vaccine	count
22	Johnson&Johnson, Oxford/AstraZeneca, Sinopharm...	7
12	Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm...	6
41	Oxford/AstraZeneca, Sinopharm/Beijing	5
14	Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm...	5
0	Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...	5

```
In [59]:
summary=summary.reset_index()
#summary.dtypes
summary
```

Out[59]:

	country	continent	total_confirmed	total_deaths	total_recovered	active_cases	serious_or_critical	to
0	Afghanistan	Asia	158275	7367.0	145750.0	5158.0	1124.0	8
1	Albania	Europe	213257	3228.0	202077.0	7952.0	23.0	14
2	Algeria	Africa	220415	6310.0	151347.0	62758.0	34.0	2
3	Andorra	Europe	25289	141.0	21511.0	3637.0	31.0	2
4	Angola	Africa	86636	1789.0	67477.0	17370.0	7.0	12
...	...	...	...	...	...	...	...	
216	Wallis And Futuna Islands	Australia/Oceania	454	7.0	438.0	9.0	NaN	
217	Western Sahara	Africa	10	1.0	8.0	1.0	NaN	
218	Yemen	Asia	10152	1986.0	7043.0	1123.0	23.0	2
219	Zambia	Africa	274087	3782.0	236878.0	33427.0	317.0	30
220	Zimbabwe	Africa	219057	5092.0	189713.0	24252.0	12.0	18

221 rows × 22 columns

Percentage Metrics:

- percentage_vaccinated
- percentage_fully_vaccinated
- tested_positive

- confirm_rate
- test_rate
- death_rate
- recover_rate
- critical_rate

In [60]:

```
rate_columns=["percentage_vaccinated","percentage_fully_vaccinated","tested_positive","cor
              "death_rate","recover_rate","critical_rate"]
rate_summary=summary[rate_columns]
```

(2) For Daily data

We human are fighting as a whole, so I formed a new dataframe to see the global trend.

In [61]:

```
daily_df.tail(3)
```

Out[61]:

	date	country	cumulative_total_cases	daily_new_cases	active_cases	cumulative_total_deaths	daily_new_d
152732	2022-1-03	Zimbabwe	216087.0	1209.0	25446.0	5047.0	
152733	2022-1-04	Zimbabwe	217678.0	1591.0	24620.0	5078.0	
152734	2022-1-05	Zimbabwe	219057.0	1379.0	24252.0	5092.0	

In [62]:

```
daily_new_cases_g=pd.DataFrame(daily_df.groupby("date")['daily_new_cases'].sum())
daily_new_deaths_g=pd.DataFrame(daily_df.groupby("date")['daily_new_deaths'].sum())
active_cases_g=pd.DataFrame(daily_df.groupby("date")['active_cases'].sum())
cumulative_total_cases_g=pd.DataFrame(daily_df.groupby("date")['cumulative_total_cases'].s
cumulative_total_deaths_g=pd.DataFrame(daily_df.groupby("date")['cumulative_total_deaths']
```

In [63]:

```
global_daily=daily_df[['date','date_parsed']].drop_duplicates().set_index('date')
global_daily=global_daily.join(daily_new_cases_g).join(daily_new_deaths_g).join(active_cas
```

In [64]:

```
global_daily=global_daily.sort_values(by=['date'])
global_daily.head(3)
```

Out[64]:

	date_parsed	daily_new_cases	daily_new_deaths	active_cases	cumulative_total_cases	cumulative_total_deaths
date						
2020-1-22	2020-01-22	0.0	0.0	554.0	571.0	17.0
2020-1-23	2020-01-23	259.0	8.0	771.0	830.0	25.0
2020-1-24	2020-01-24	457.0	16.0	1208.0	1287.0	41.0

(3) Add daily vaccination data to global_daily

In [65]:

```
vacc_df.tail(3)
```

Out[65]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations
73006	Zimbabwe	ZWE	2022-01-23	7512903.0	4242647.0	3270256.0	10631.0
73007	Zimbabwe	ZWE	2022-01-24	7517985.0	4245063.0	3272922.0	10273.0
73008	Zimbabwe	ZWE	2022-01-25	7525574.0	4248576.0	3276998.0	9579.0

In [66]:

```
global_dailyvacc=vacc_df[['date','date_parsed']].drop_duplicates().set_index('date')
#dailyvacc
```

In [67]:

```
daily_total_vaccinations=pd.DataFrame(vacc_df.groupby("date")['total_vaccinations'].sum())
daily_people_vaccinated=pd.DataFrame(vacc_df.groupby("date")['people_vaccinated'].sum())
daily_people_fully_vaccinated=pd.DataFrame(vacc_df.groupby("date")['people_fully_vaccinate
daily_vaccinations=pd.DataFrame(vacc_df.groupby("date")['daily_vaccinations'].sum())
global_dailyvacc=global_dailyvacc.join(daily_total_vaccinations).join(daily_people_vaccina
global_dailyvacc=global_dailyvacc.sort_values(by=['date'])
global_dailyvacc.tail(3)
```

Out[67]:

	date_parsed	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations
date					
2022-01-23	2022-01-23	8.966349e+09	2.882545e+09	2.252420e+09	27149305.0
2022-01-24	2022-01-24	8.773421e+09	2.891273e+09	2.262299e+09	23342577.0
2022-01-25	2022-01-25	7.635721e+09	2.382629e+09	1.795026e+09	20932767.0

In [68]:

```
#add to Daily summary data
global_dailyvacc=global_dailyvacc.drop('date_parsed',axis=1)
global_daily=global_daily.drop('date_parsed',axis=1)
global_dailyvacc
```

Out[68]:

	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations
date				
2020-12-01	1.000000e+00	1.000000e+00	0.000000e+00	0.0
2020-12-02	0.000000e+00	0.000000e+00	0.000000e+00	0.0
2020-12-03	0.000000e+00	0.000000e+00	0.000000e+00	0.0
2020-12-04	1.000000e+00	1.000000e+00	0.000000e+00	0.0
2020-12-05	0.000000e+00	0.000000e+00	0.000000e+00	0.0
...	...	...	...	...
2022-01-21	8.330316e+09	3.918040e+09	3.274116e+09	29814415.0
2022-01-22	8.371825e+09	2.660690e+09	2.048970e+09	28621976.0
2022-01-23	8.966349e+09	2.882545e+09	2.252420e+09	27149305.0

	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations
date				
2022-01-24	8.773421e+09	2.891273e+09	2.262299e+09	23342577.0
2022-01-25	7.635721e+09	2.382629e+09	1.795026e+09	20932767.0

421 rows × 4 columns

In [69]:

```
global_daily=global_daily.join(global_dailyvacc,how='outer')
global_daily
```

Out[69]:

	daily_new_cases	daily_new_deaths	active_cases	cumulative_total_cases	cumulative_total_deaths	total_vaccinat
date						
2020-1-22	0.0	0.0	554.0	571.0	17.0	
2020-1-23	259.0	8.0	771.0	830.0	25.0	
2020-1-24	457.0	16.0	1208.0	1287.0	41.0	
2020-1-25	688.0	15.0	1870.0	1975.0	56.0	
2020-1-26	769.0	24.0	2613.0	2744.0	80.0	
...	...	...	...	...	...	
2022-1-01	1775009.0	4592.0	30485704.0	290617792.0	5458720.0	
2022-1-02	1293087.0	3630.0	31334086.0	291910879.0	5462350.0	
2022-1-03	1468676.0	4712.0	32126017.0	293379555.0	5467062.0	
2022-1-04	2213067.0	7437.0	33573782.0	295592622.0	5474499.0	
2022-1-05	2576853.0	7551.0	35451328.0	298169475.0	5482050.0	

1013 rows × 9 columns

(4) For vaccinations by manufacturer

In [70]:

```
vacc_manu.tail(3)
```

Out[70]:

	location	date	vaccine	total_vaccinations
26169	European Union	2022-01-25	Sinopharm/Beijing	2264826
26170	European Union	2022-01-25	Sinovac	9
26171	European Union	2022-01-25	Sputnik V	1845079

In [71]:

```
global_vacc_manu=pd.DataFrame(vacc_manu.groupby(["date","vaccine"])['total_vaccinations'].
global_vacc_manu=global_vacc_manu.reset_index()
global_vacc_manu
```

Out[71]:

	date	vaccine	total_vaccinations
0	2020-12-04	Moderna	1
1	2020-12-07	Pfizer/BioNTech	1
2	2020-12-09	Pfizer/BioNTech	2
3	2020-12-15	Pfizer/BioNTech	3
4	2020-12-16	Pfizer/BioNTech	4
...	...	...	...
3022	2022-01-25	Oxford/AstraZeneca	96145780
3023	2022-01-25	Pfizer/BioNTech	1067361804
3024	2022-01-25	Sinopharm/Beijing	2264826
3025	2022-01-25	Sinovac	16949842
3026	2022-01-25	Sputnik V	1845079

3027 rows × 3 columns

5. Data Visualization and Analysis

In [ ]:

```

```

(1) With Summarized Data - What's the current stage?

Available data sets:

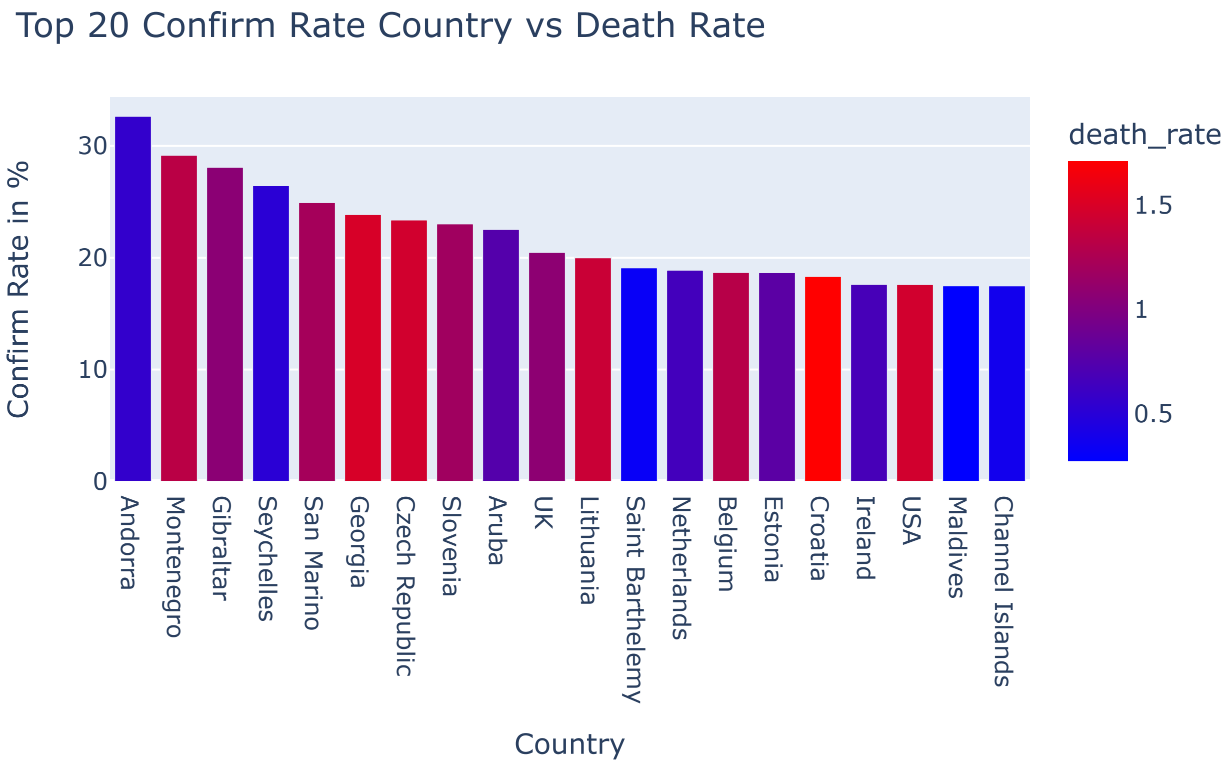
- summary
- rate_summary(subset of summary)

Top 20 confirm rate countries

In [220…

```
import plotly.express as px

fig = px.bar(summary.sort_values('confirm_rate', ascending=False).head(20),
             x='country', y='confirm_rate',
             hover_data=['test_rate','confirm_rate','critical_rate','death_rate','percenta
             color='death_rate',
             color_continuous_scale=px.colors.sequential.Bluered,
             labels={'pop':'population'}, height=400)
fig.update_layout(
    title="Top 20 Confirm Rate Country vs Death Rate",
    xaxis_title="Country",
    yaxis_title="Confirm Rate in %",
    legend_title="Death Rate in %"
)
fig.show()
```



- Top 20 confirm rate countries are: Andorra, Montenegro, Gibraltar, Seychelles, San Marino, Georgia, Czech Republic, Slovenia, Aruba, UK, Lithuania, Saint Barthelemy, Netherlands, Belgium, Estonia, Croatia, Ireland, USA, Maldives, Channel Islands.
- Generally, the death rate is high in the high confirm rate countries.

In [99]:

```
print("Average vaccination rate of top 20 death rate countries:{:.2%}".format(np.nanmean(s
print("Average vaccination rate of top 10 death rate countries:{:.2%}".format(np.nanmean(s
```

```
Average vaccination rate of top 20 death rate countries:0.98%
Average vaccination rate of top 10 death rate countries:1.06%
```

Top 20 death rate countries

In [ ]:

```
import plotly.express as px

fig = px.bar(summary.sort_values('death_rate', ascending=False).head(20),
             x='country', y='death_rate',
             hover_data=['test_rate','confirm_rate','critical_rate','death_rate','percenta
             color='percentage_vaccinated',
             color_continuous_scale=px.colors.sequential.speed,
             labels={'pop':'population'}, height=400)
fig.update_layout(
    title="Top 20 Death Rate Country vs Vaccination Status",
    xaxis_title="Country",
    yaxis_title="Death Rate",
    legend_title="Vaccinated Percentage"
)
fig.update_xaxes(tickangle=45)
fig.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_23688/122969398.py in <module>
      3              hover_data=['test_rate','confirm_rate','critical_rate','death_rate',
'percentage_fully_vaccinated','percentage_vaccinated'],
      4                    color='percentage_vaccinated',
----> 5                    color_continuous_scale=px.colors.sequential.speed,
      6                    labels={'pop':'population'}, height=400)
      7 fig.update_layout(

NameError: name 'px' is not defined
```

- Top 20 death rate countries are: Yemen, Vanuatu, Western Sahara, Peru, Mexico, Sudan, Eduador, Syria, Egypt, Somalia, Taiwan, Afghanistan, Bosnia And Herzegovina, China, Liberia, Bulgaria, Niger, Myanmar, Paraguay, Macedonia.
- The percentage of vaccinated varies - some of them have over an over 80% vaccination rate. But for most of the high death rate countries, the overall vaccination rate is low.

In [93]:

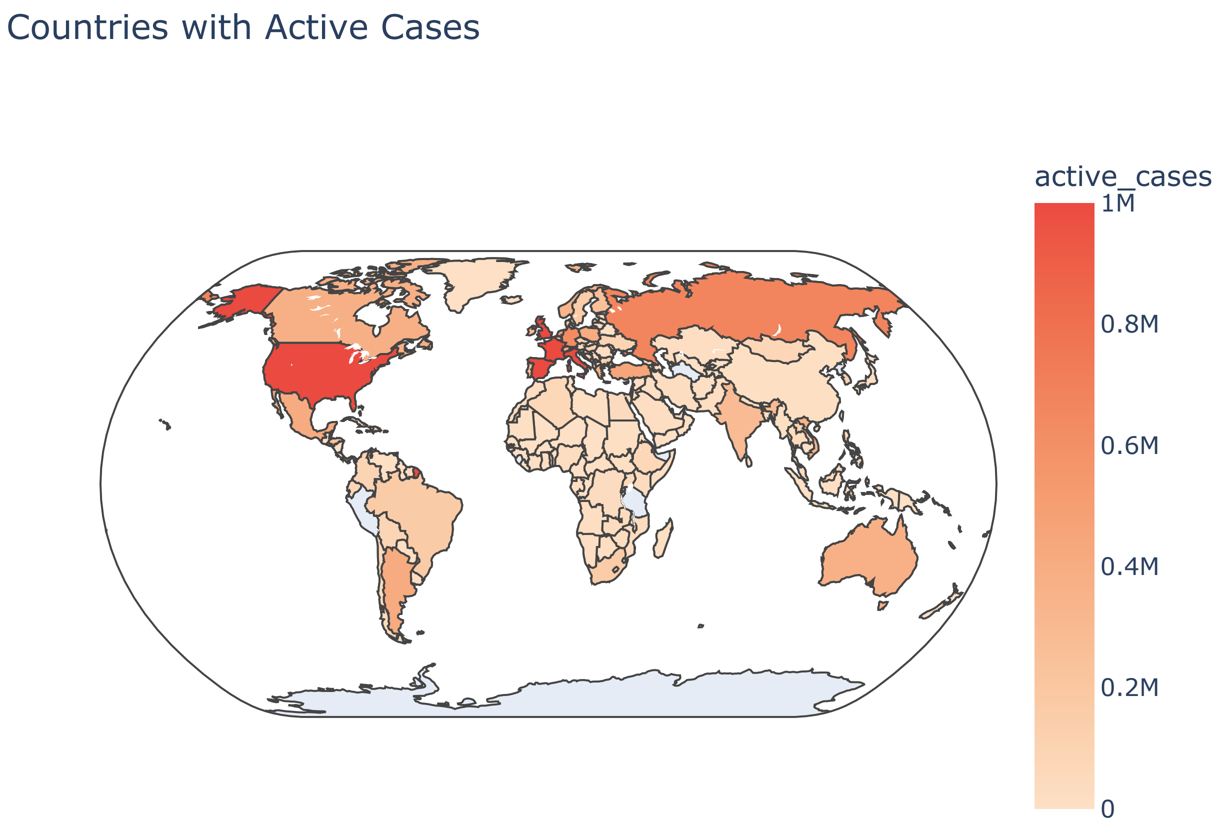
```
print("Average vaccination rate of top 20 death rate countries:{:.2%}".format(np.nanmean(s
```

```
Average vaccination rate of top 20 death rate countries:37.08%
```

Countries with Active Cases

In [144…

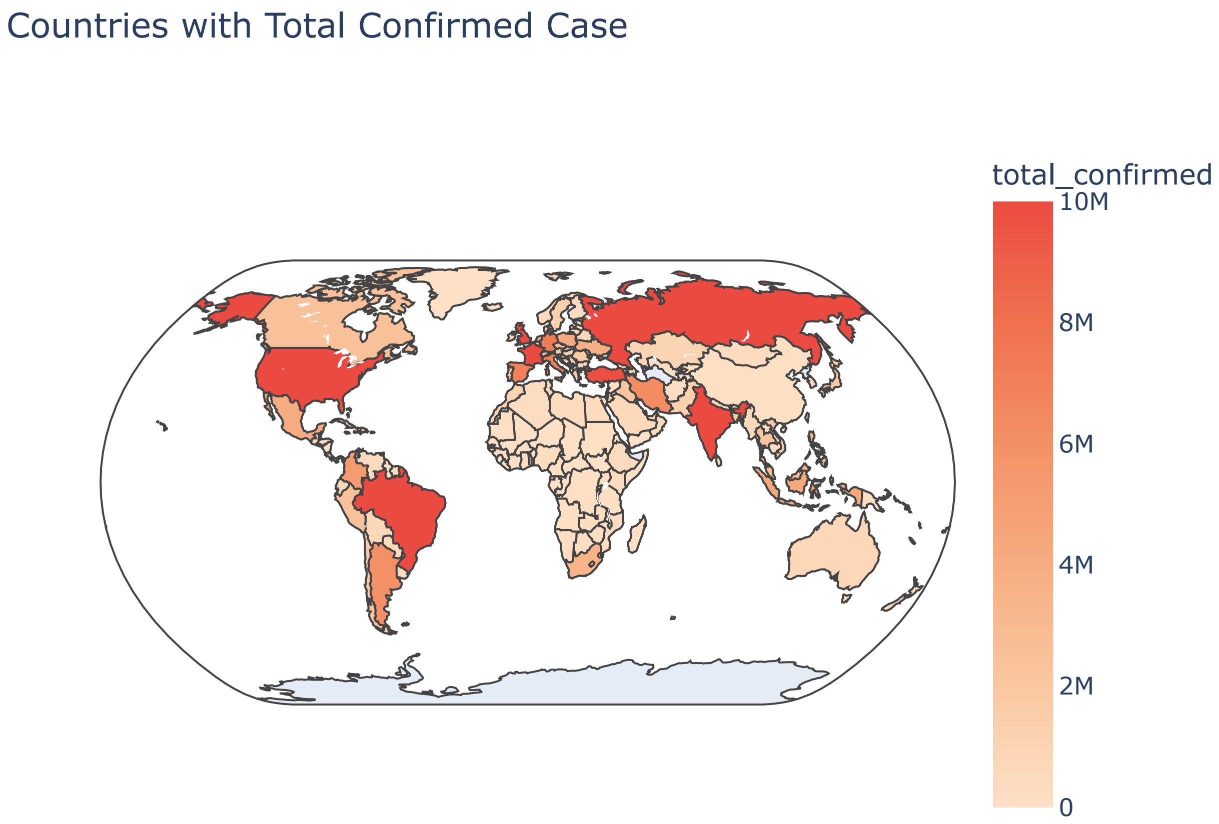
```
figure = px.choropleth(summary, locations="country",
                    locationmode='country names', color="active_cases",
                    hover_name="country", range_color=[1,1000000],
                    color_continuous_scale="Peach", title ="Countries with Active Cases",
                    projection="natural earth")
fig.update_layout(
    autosize=False,
    width=800,
    height=600
)
figure.show()
```



Countries with Total Confirmed Case

In [159…

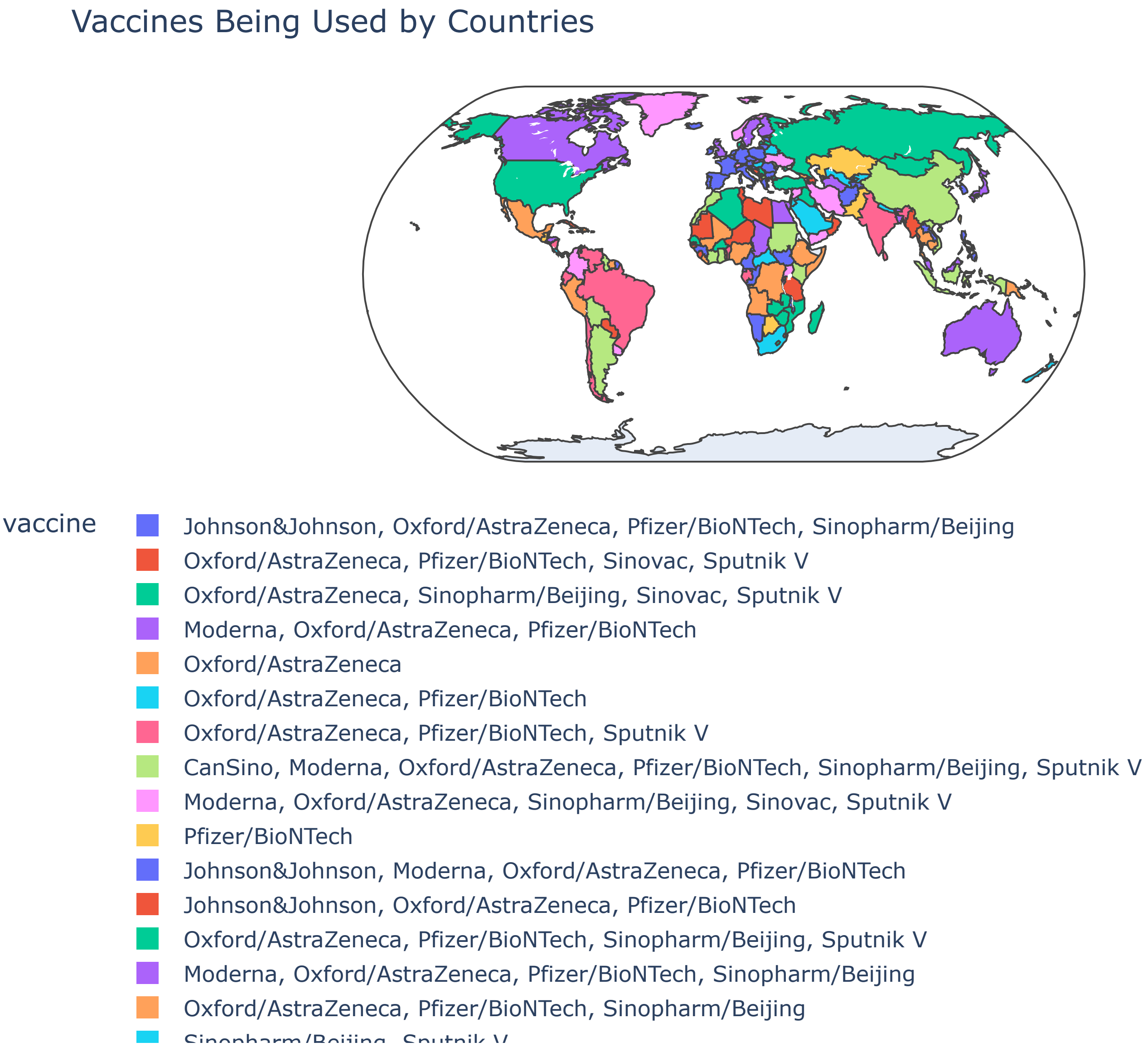
```
figure = px.choropleth(summary, locations="country",
                    locationmode='country names', color="total_confirmed",
                    hover_name="country", range_color=[1,10000000],
                    color_continuous_scale="Peach", title ="Countries with Total Confirmed
                    projection="natural earth")
fig.update_layout(
    autosize=False,
    width=800,
    height=600
)
figure.show()
```



Vaccines Being Used by Countries

In [147…

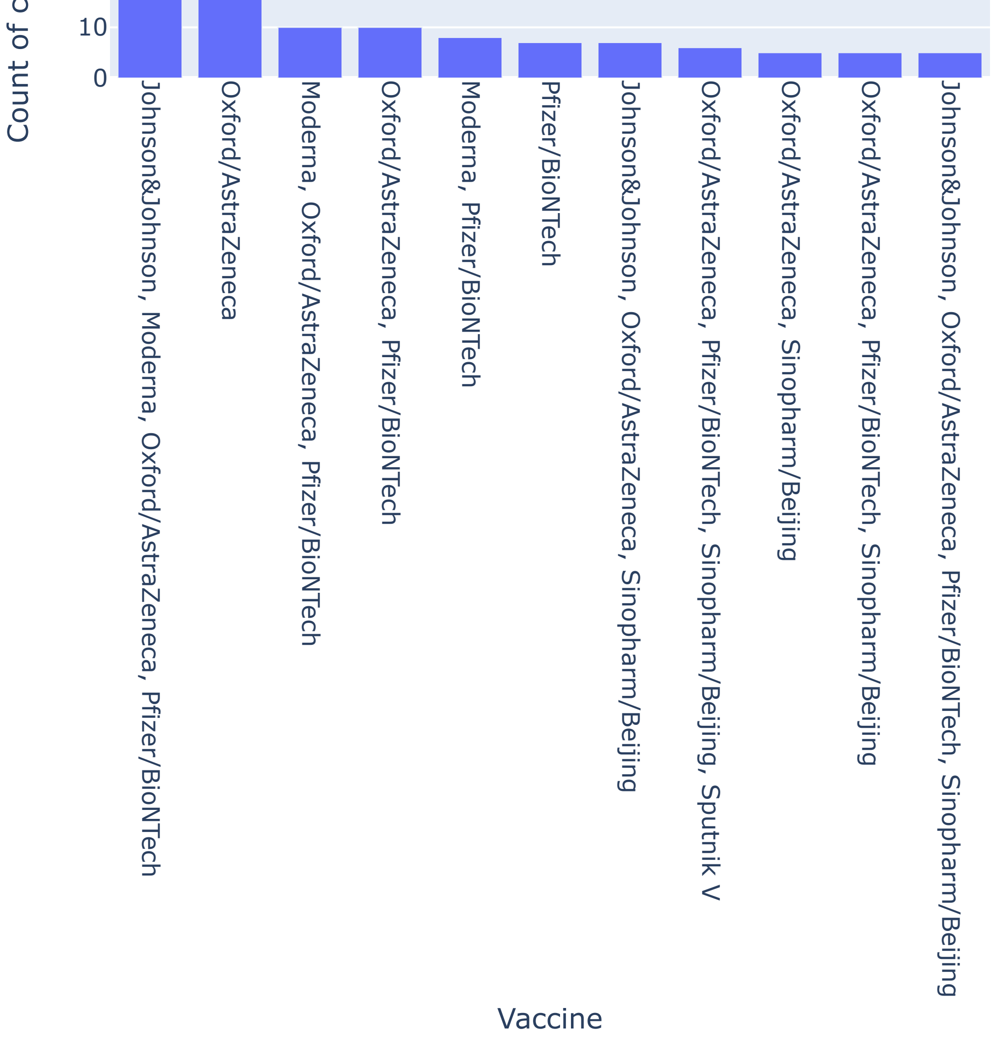
```
fig = ex.choropleth(vaccine_country_df, locations="country",
                    locationmode='country names',
                    color="vaccine",
                    hover_name="country",
                   projection="natural earth")
fig.update_layout(
    title="Vaccines Being Used by Countries",
    autosize=False,
    width=800,
    height=600,
    legend_orientation = 'h'
)
fig.show()
```


Vaccines Being Used by Countries
vaccine
Johnson&Johnson, Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing
Oxford/AstraZeneca, Pfizer/BioNTech, Sinovac, Sputnik V
Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac, Sputnik V
Moderna, Oxford/AstraZeneca, Pfizer/BioNTech
Oxford/AstraZeneca
Oxford/AstraZeneca, Pfizer/BioNTech
Oxford/AstraZeneca, Pfizer/BioNTech, Sputnik V
CanSino, Moderna, Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V
Moderna, Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac, Sputnik V
Pfizer/BioNTech
Johnson&Johnson, Moderna, Oxford/AstraZeneca, Pfizer/BioNTech
Johnson&Johnson, Oxford/AstraZeneca, Pfizer/BioNTech
Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V
Moderna, Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing
Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing
Sinopharm/Beijing, Sputnik V

Popularity of vaccination combinations

In [256…

```
fig = px.bar(head_pop_vacc_df, x='vaccine', y='count')
fig.update_layout(
    title="Popular Vaccination Combinations by Countries",
    xaxis_title="Vaccine",
    yaxis_title="Count of countries",
    autosize=False,
    width=600,
    height=600
)
fig.update_xaxes(tickangle=90)
fig.show()
```

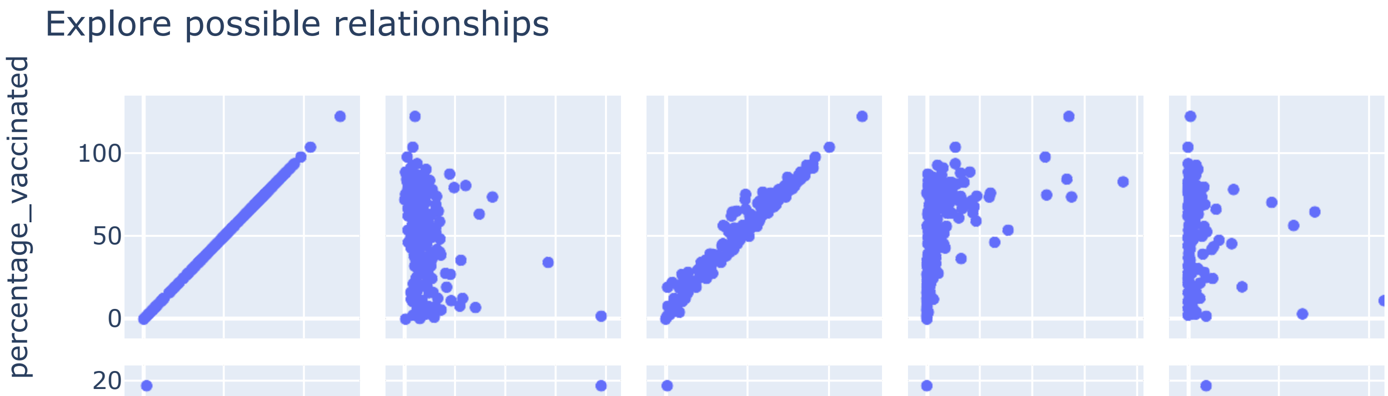


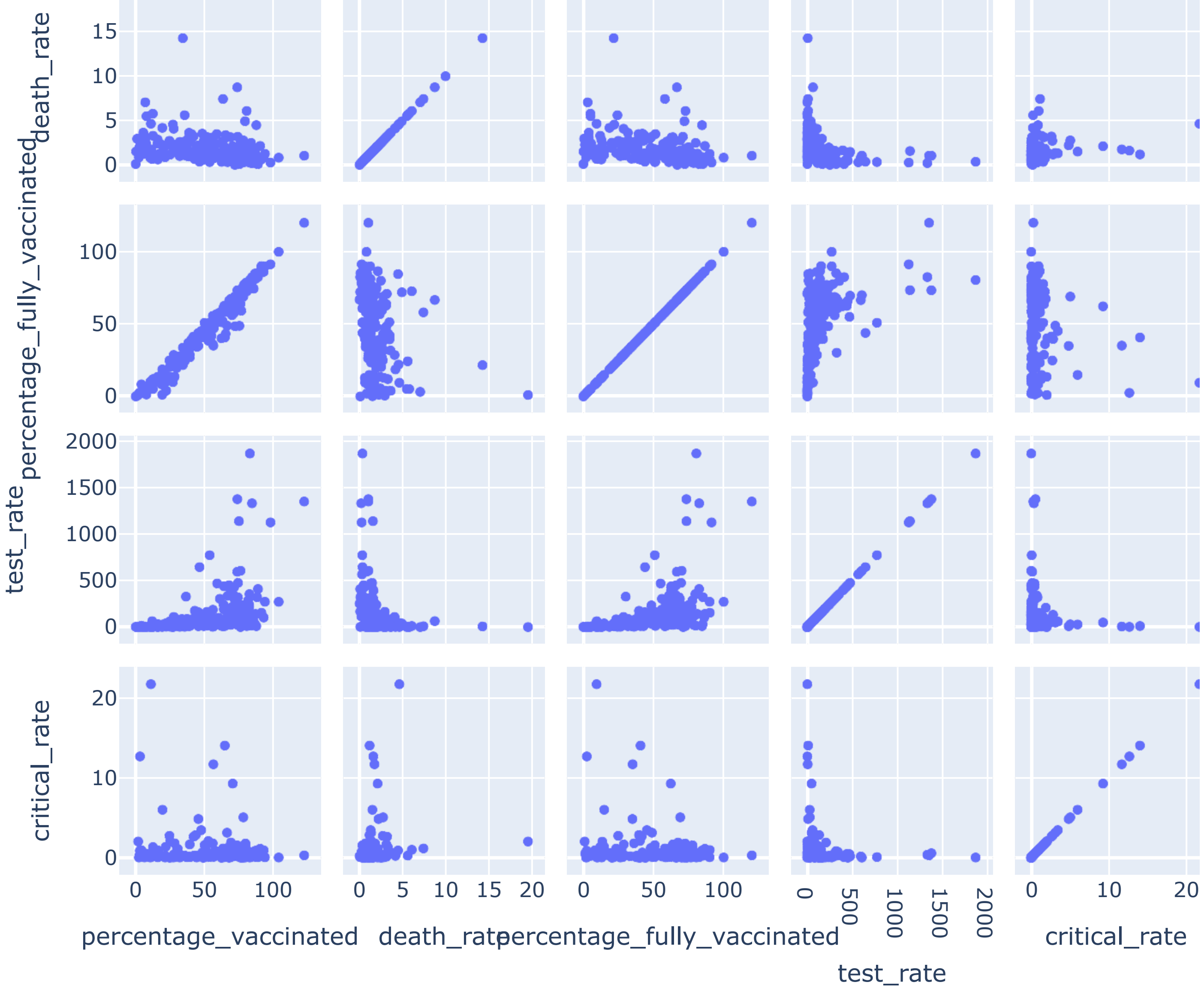
- The two most popular vaccination combinations which has a significant advantage are: *Johnson&Johnson, Moderna, Oxford/AstraZeneca, Pfizer/BioNTech*, and *Oxford/AstraZeneca*. Both of them are used by more than 20 countries.

Explore possible relationships

In [ ]:

```
import plotly.express as px
fig = px.scatter_matrix(summary,dimensions=["percentage_vaccinated", "death_rate", "percen
fig.update_layout(
    autosize=False,
    width=800,
    height=800,
    title="Explore possible relationships"
)
fig.show()
```





- The vaccination rate is highly positively correlated with fully vaccination rate.
- There's no **high correlations** between other variable pairs, however:
  - 1. percentage of vaccination vs. test rate: There seems to exist a positive relationship: if the vaccination rate is higher, the test rate tends to be higher as well. **This means if a country has a more positive and cautious attitude towards the epidemic, both the vaccination rate and the test rate will get higher.**
  - 1. test rate vs. critical rate: There seems to exist a negative relationship: if the test rate is higher, the critical rate tends to be lower. There's also a few extreme cases with very high test rate and a critical rate nears 0, and a few cases with a test rate that is close to 0 and a very high critical rate. **This means if a country test very often, it can decrease the probability that cases become severe and critical.**
  - 1. fully vaccination rate vs. death rate: Although it's not that obvious due to the variation in death rate, there seems to exist a negative relationship: if the fully vaccination rate is higher, the death rate tends to be lower. **This means fully vaccination may decrease the probablity of death caused by the virus.**

(2) With Daily Data

- What kind of trends and patterns has happened?

- Any critical points related to Omicron?

Available data sets:

- daily_df
- global_daily
- vacc_df
- global_dailyvacc

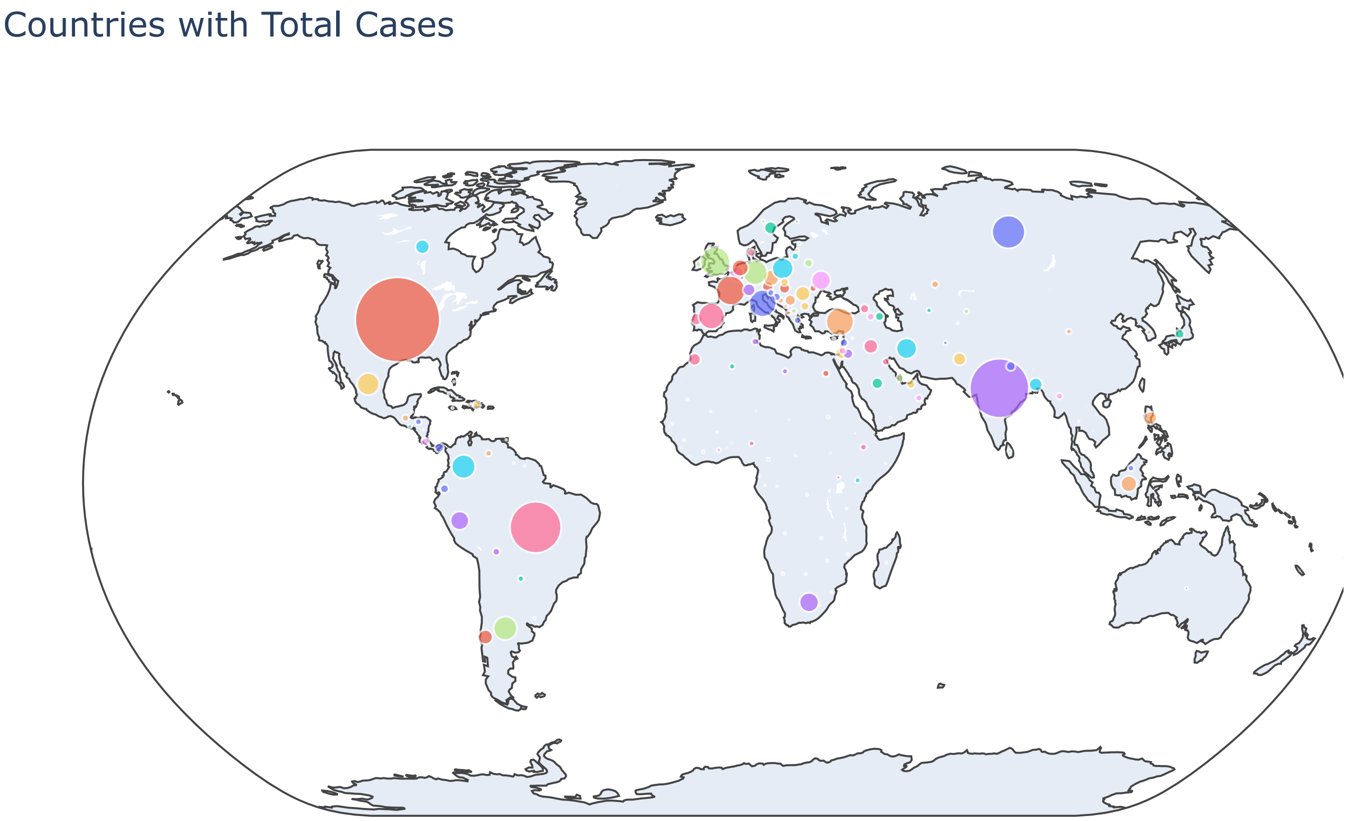
Geographic distribution of total cases

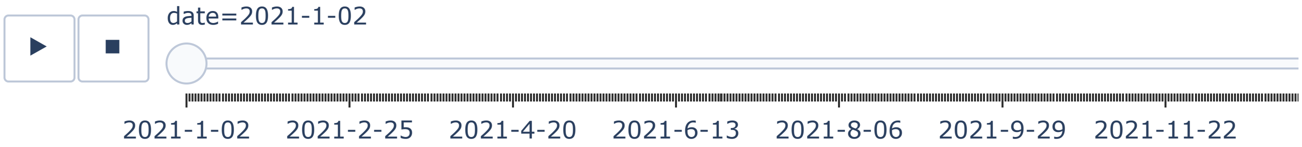
In [ ]:

```
import pandas as pd
import chart_studio.plotly as py
import plotly.offline as po
import plotly.graph_objs as pg
import matplotlib.pyplot as plt
%matplotlib inline
po.init_notebook_mode(connected = True)
```

In [142…

```
import plotly.express as px
fig = px.scatter_geo(daily_df[(daily_df.date>"2021-1-01")&(daily_df.cumulative_total_cases
                     projection="natural earth",locationmode="country names",animation_fra
fig.update_layout(
    title ="Countries with Total Cases",
    autosize=False,
    width=800,
    height=600,
    showlegend=False)
fig.update_layout(transition_duration=3000)
fig.show()
```

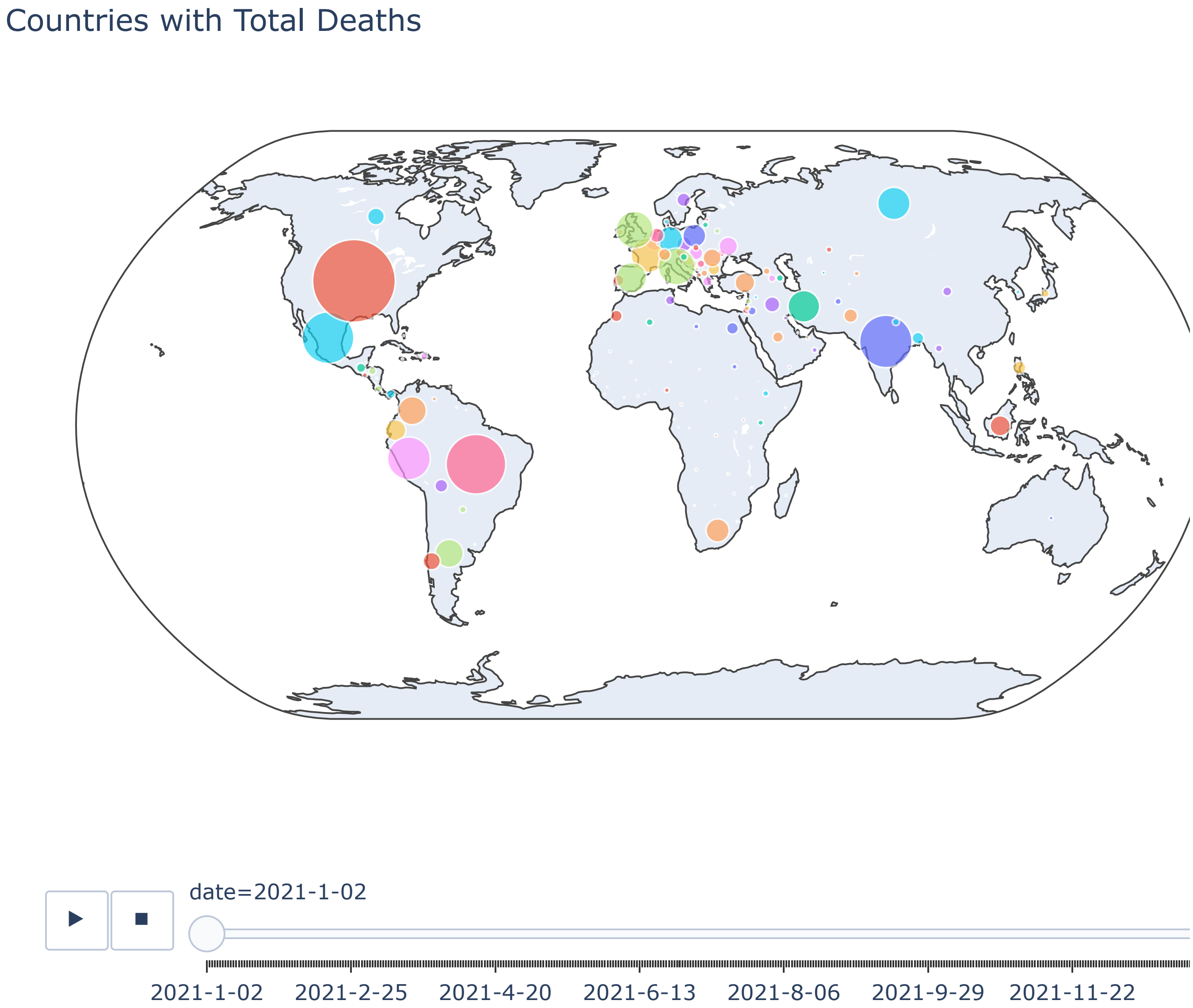




Geographic distribution of total deaths

In [143…

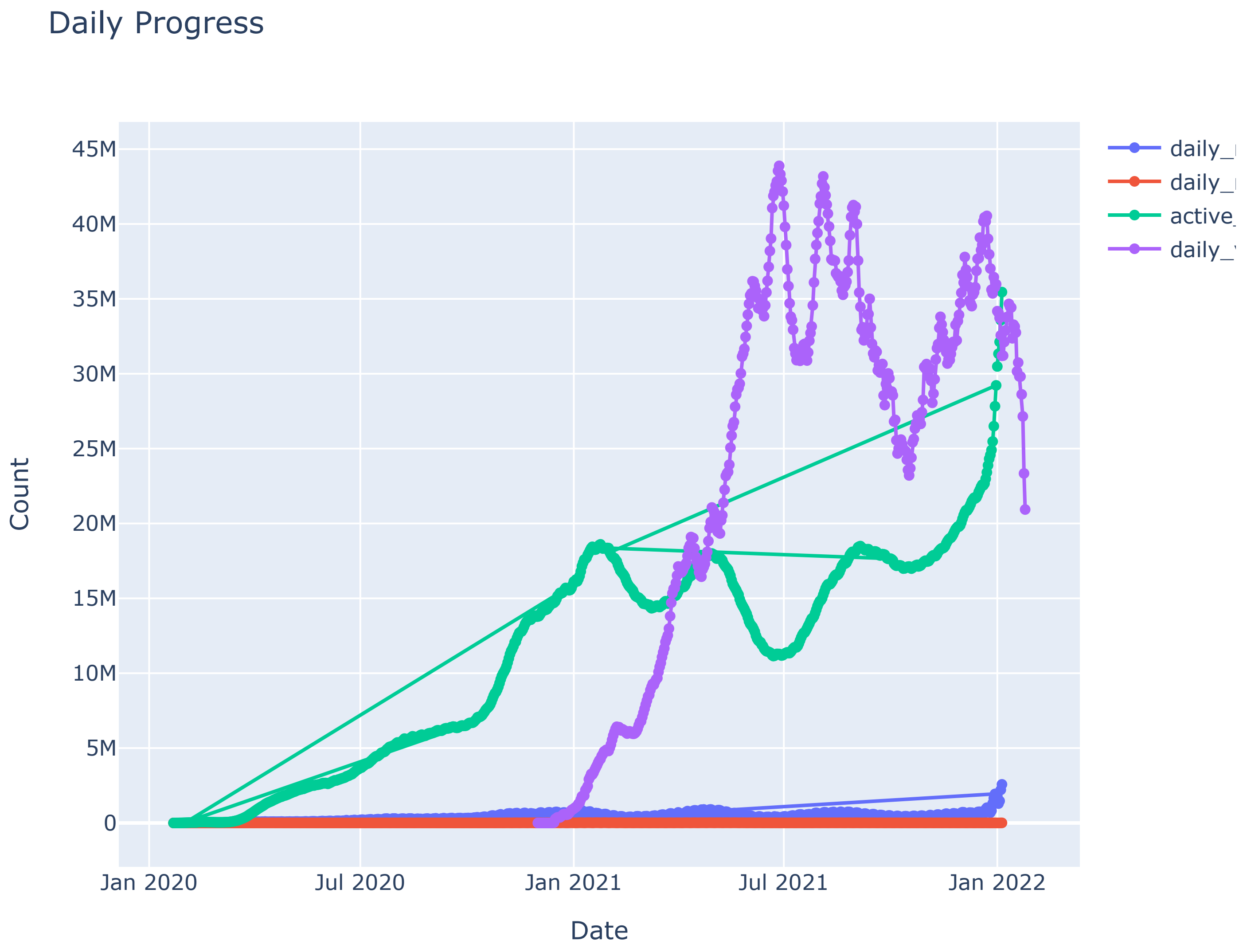
```
import plotly.express as px
fig = px.scatter_geo(daily_df[(daily_df.date>"2021-1-01")&(daily_df.cumulative_total_death
                     projection="natural earth",locationmode="country names",animation_fra
fig.update_layout(
    title ="Countries with Total Deaths",
    autosize=False,
    width=800,
    height=600,
    showlegend=False)
fig.update_layout(transition_duration=3000)
fig.show()
```



Global daily progress

In [225…

```
line_plots = []
variable_list=["daily_new_cases","daily_new_deaths","active_cases","daily_vaccinations"]
for variable in variable_list:
    line_plots.append(
        go.Scatter(
            name=variable,
            x = global_daily.index,
            mode = "lines+markers",
            y=global_daily[variable],
        )
    )
fig = go.Figure(line_plots)
fig.update_layout(
    title ="Daily Progress",
    xaxis_title="Date",
    yaxis_title="Count",
    hovermode='x',
    legend_orientation = 'v',
    autosize=False,
    width=800,
    height=600
)
fig.show()
```



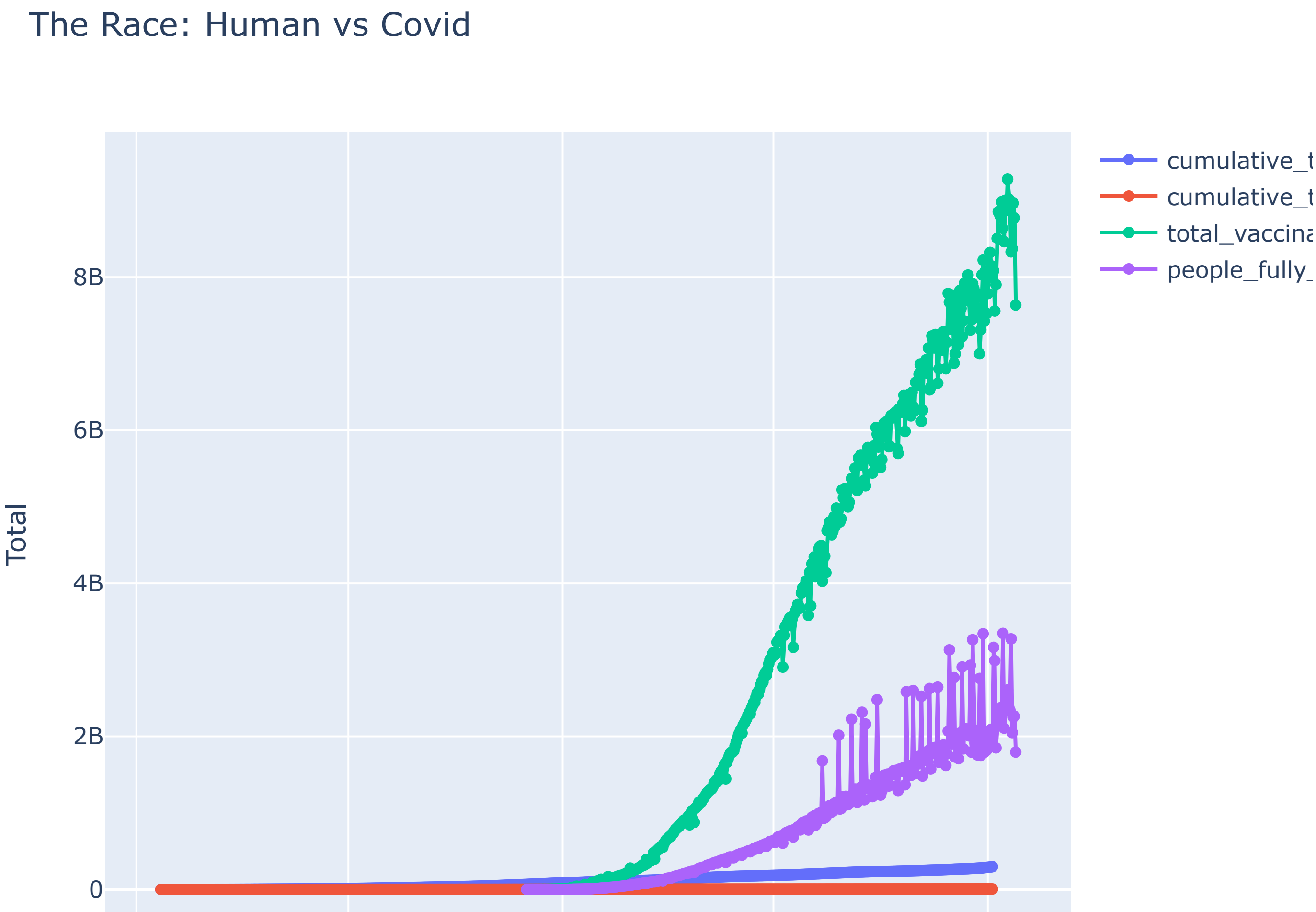
- There exists periodic fluctuation in daily new cases and active cases over time: It will go up for about 3 months, and then go down for about 3 months.

- There's a huge rapid growth in new cases and active cases in the end of 2021 and January 2022. Possible reasons would be more frequent travelling due to the holidays, and the high infectivity of Omicron. However, the number of vaccination goes down in this period.

Global total progress

In [105…

```
line_plots = []
variable_list=["cumulative_total_cases","cumulative_total_deaths","total_vaccinations","pe
for variable in variable_list:
    line_plots.append(
        go.Scatter(
            name=variable,
            x = global_daily.index,
            mode = "lines+markers",
            y=global_daily[variable],
        )
    )
fig = go.Figure(line_plots)
fig.update_layout(
    title ="The Race: Human vs Covid",
    xaxis_title="Date",
    yaxis_title="Total",
    hovermode='x',
    legend_orientation = 'v',
    autosize=False,
    width=800,
    height=600
)
fig.show()
```



The number of fully vaccinated people is many times of the number of cases. With the rapid growing trend of number of vaccinations, I'm confident that we human can beat the disease!

(3) With Vaccination by manufacture Data

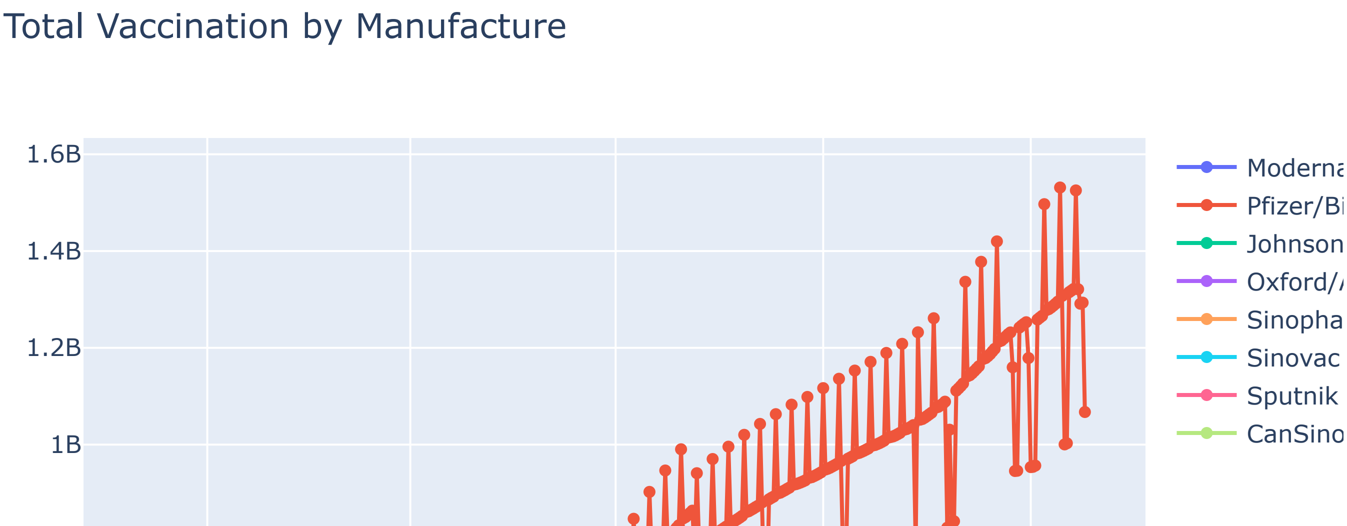
Available data sets:

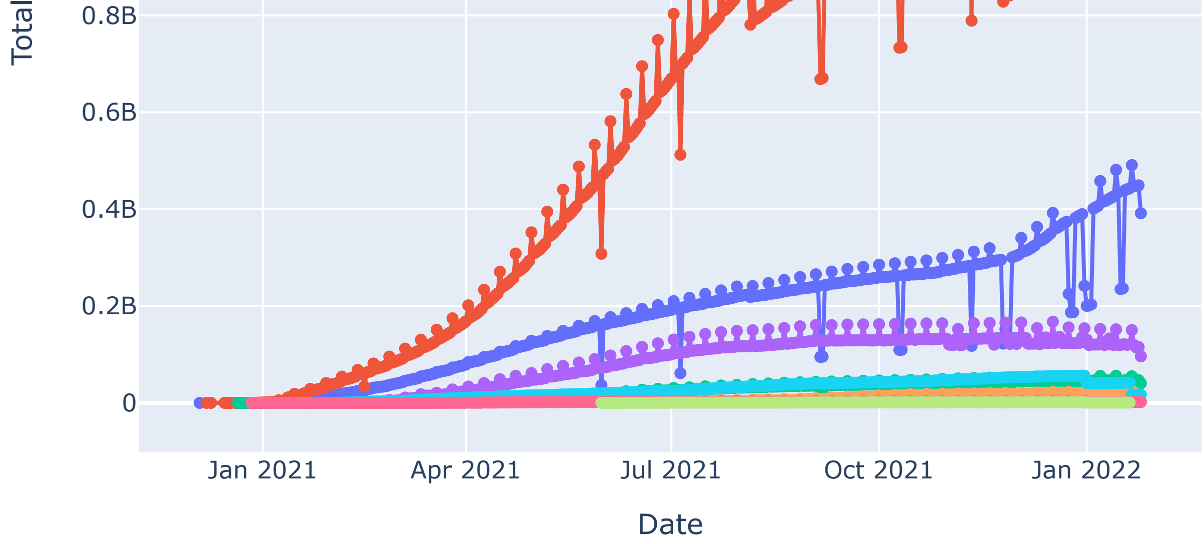
- vacc_manu
- global_vacc_manu

Total Vaccination by Manufacture

In [81]:

```
line_plots = []
vaccines = global_vacc_manu.vaccine.unique().tolist()
for v in vaccines:
    vacc_data = global_vacc_manu[global_vacc_manu.vaccine == v]
    line_plots.append(
        go.Scatter(
            name = v,
            x = vacc_data.date,
            mode = "lines+markers",
            y=vacc_data['total_vaccinations'],
        )
    )
fig = go.Figure(line_plots)
fig.update_layout(
    title ="Total Vaccination by Manufacture",
    xaxis_title="Date",
    yaxis_title="Total",
    hovermode='x',
    legend_orientation = 'v',
    autosize=False,
    width=800,
    height=600
)
fig.show()
```





- Pfizer has absolute dominance in vaccination, followed by Moderna.

6. Summary of the conclusions

Countries

- Top 20 confirm rate countries are: Andorra, Montenegro, Gibraltar, Seychelles, San Marino, Georgia, Czech Republic, Slovenia, Aruba, UK, Lithuania, Saint Barthelemy, Netherlands, Belgium, Estonia, Croatia, Ireland, USA, Maldives, Channel Islands. **Generally, the death rate is high in the high confirm rate countries.**
- Top 20 death rate countries are: Yemen, Vanuatu, Western Sahara, Peru, Mexico, Sudan, Eduador, Syria, Egypt, Somalia, Taiwan, Afghanistan, Bosnia And Herzegovina, China, Liberia, Bulgaria, Niger, Myanmar, Paraguay, Macedonia. The percentage of vaccinated varies - some of them have over an over 80% vaccination rate. **But for most of the high death rate countries, the overall vaccination rate is low. Possible strategies**
- Percentage of vaccination vs. test rate: There seems to exist a positive relationship: if the vaccination rate is higher, the test rate tends to be higher as well. **This means if a country has a more positive and cautious attitude towards the epidemic, both the vaccination rate and the test rate will get higher.**
- Test rate vs. critical rate: There seems to exist a negative relationship: if the test rate is higher, the critical rate tends to be lower. There's also a few extreme cases with very high test rate and a critical rate nears 0, and a few cases with a test rate that is close to 0 and a very high critical rate. **This means if a country test very often, it can decrease the probability that cases become severe and critical.**
- Fully vaccination rate vs. death rate: Although it's not that obvious due to the variation in death rate, there seems to exist a negative relationship: if the fully vaccination rate is higher, the death rate tends to be lower. **This means fully vaccination may decrease the probablity of death caused by the virus. Trends**
- There exists periodic fluctuation in daily new cases and active cases over time: It will go up for about 3 months, and then go down for about 3 months.
- There's a huge rapid growth in new cases and active cases in the end of 2021 and January 2022. Possible reasons would be more frequent travelling due to the holidays, and the high infectivity of Omicron. However, the number of vaccination goes down in this period. **Vaccination**
- The two most popular vaccination combinations which has a significant advantage are: *Johnson&Johnson, Moderna, Oxford/AstraZeneca, Pfizer/BioNTech*, and *Oxford/AstraZeneca*. Both of them are used by more than 20 countries.
- Pfizer has absolute dominance in vaccination, followed by Moderna.

- The number of fully vaccinated people is many times of the number of cases. With the rapid growing trend of number of vaccinations, I'm confident that we human can beat the disease!