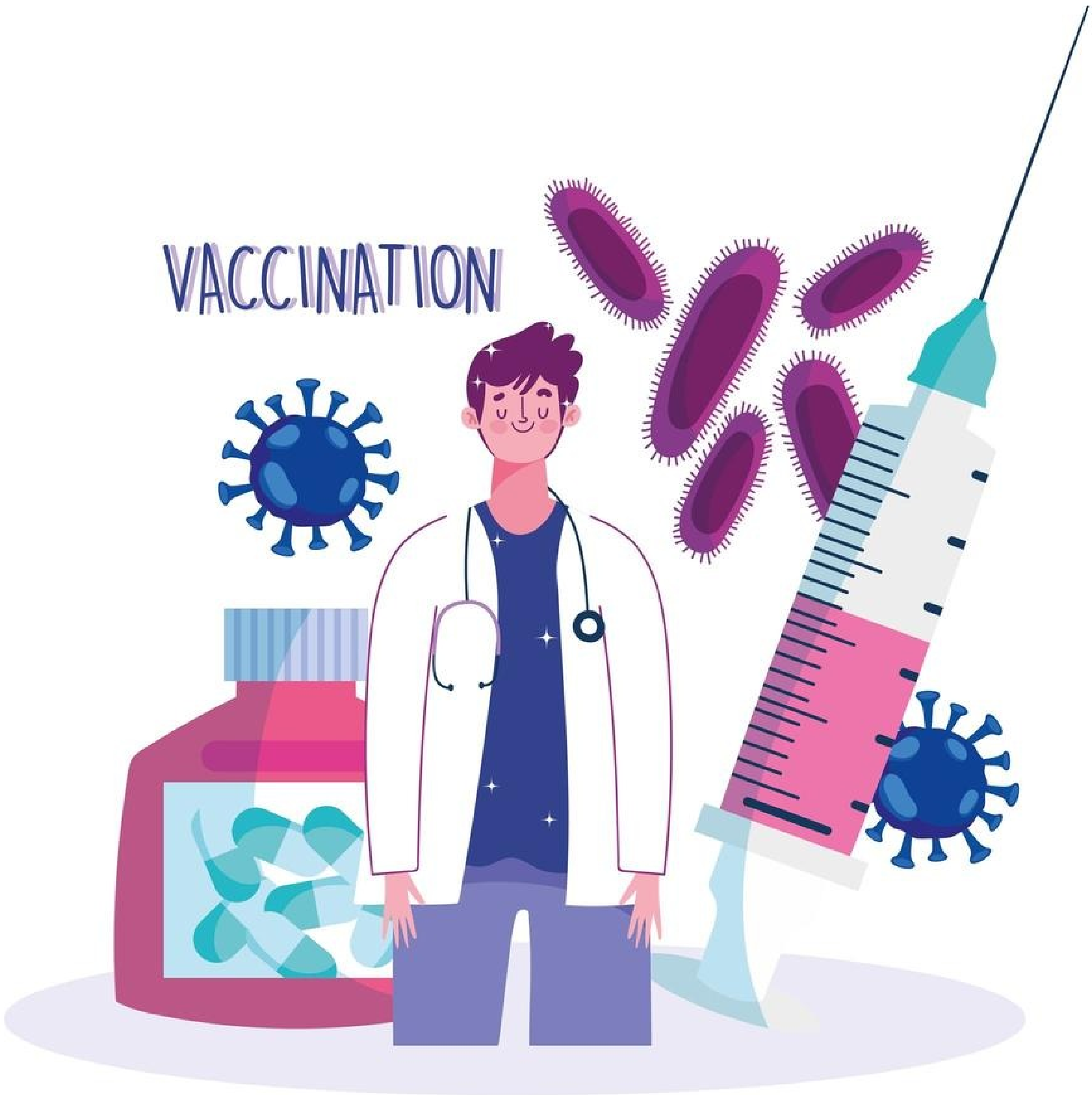
COVID-19 World Vaccination Progress

Daniil Valyano 03/10/2021



Acknowledgement

This notebook is a submission for a task of a [COVID-19 World Vaccination Progress](#) dataset, offered by Gabriel Preda [@gpreda](#)

If you like my job or the methods I used to do it, please upvote and leave your suggestions in comments!

Information

The initial dataset contains the following features:

- **Country- this is the country for which the vaccination information is provided;**
- **Country ISO Code - ISO code for the country;**
- **Date - date for the data entry; for some of the dates we have only the daily vaccinations, for others, only the (cumulative) total;**
- **Total number of vaccinations - this is the absolute number of total immunizations in the country;**
- **Total number of people vaccinated - a person, depending on the immunization scheme, will receive one or more (typically 2) vaccines; at a certain moment, the number of vaccination might be larger than the number of people;**
- **Total number of people fully vaccinated - this is the number of people that received the entire set of immunization according to the immunization scheme (typically 2); at a certain moment in time, there might be a certain number of people that received one vaccine and another number (smaller) of people that received all vaccines in the scheme;**
- **Daily vaccinations (raw) - for a certain data entry, the number of vaccination for that date/country;**
- **Daily vaccinations - for a certain data entry, the number of vaccination for that date/country;**
- **Total vaccinations per hundred - ratio (in percent) between vaccination number and total population up to the date in the country;**
- **Total number of people vaccinated per hundred - ratio (in percent) between population immunized and total population up to the date in the country;**
- **Total number of people fully vaccinated per hundred - ratio (in percent) between population fully immunized and total population up to the date in the country;**
- **Number of vaccinations per day - number of daily vaccination for that day and country;**
- **Daily vaccinations per million - ratio (in ppm) between vaccination number and total population for the current date in the country;**
- **Vaccines used in the country - total number of vaccines used in the country (up to date);**
- **Source name - source of the information (national authority, international organization, local organization etc.);**
- **Source website - website of the source of information;**

Packages used

In [1]:

```
# Data manipulation
import pandas as pd
import numpy as np

# Data visualization
import plotly.express as px
import matplotlib.pyplot as plt
import seaborn as sns
import folium

# Statistics
import scipy
import statsmodels as sms
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_percentage_error
```

Loading data

Read the initial dataset with the *Pandas* package, and display the first five rows of it.

In [2]:

```
vaccinations = pd.read_csv('../input/covid-world-vaccination-progress/country_vaccination
s.csv')
vaccinations.head()
```

Out[2]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_v
0	Afghanistan	AFG	2021-02-22	0.0	0.0	NaN	NaN	
1	Afghanistan	AFG	2021-02-23	NaN	NaN	NaN	NaN	
2	Afghanistan	AFG	2021-02-24	NaN	NaN	NaN	NaN	
3	Afghanistan	AFG	2021-02-25	NaN	NaN	NaN	NaN	
4	Afghanistan	AFG	2021-02-26	NaN	NaN	NaN	NaN	

Data Cleaning

There are few steps in this section:

- **As shown by *.head()* method, our data obviously contains missing values, so we'll need to dela with it somehow**
- **Some of the country names appeared in the dataset, does not represent a single country (constituent countries)**
- ***date* feature refers to a unique moment in time, so needs to be converted into a *Pandas date format***

In [3]:

```
vaccinations["date"] = pd.to_datetime(vaccinations["date"], format = '%Y-%m-%d')
vaccinations = vaccinations.replace([np.inf, -np.inf], np.nan)
vaccinations = vaccinations.fillna(0)
```

Normalizing country names

There is still a reason not to start the analysis: unused country names. The United Kingdom of Great Britain and Northern Ireland (UK), since 1922, comprises four constituent countries: England, Scotland, and Wales (which collectively make up Great Britain), as well as Northern Ireland. To fix this, we won´t include England, Scotland, Wales, and Nothern Ireland in our analysis, their data is already stored in the United Kingdom observation.

I also consider to eliminate Gibraltar, Isle of Man, Cayman Islands, Falkland Islands, Guernsey, Saint Helena, Turks and Caicos Islands (UK pertinence), Faeroe Islands (Denmark pertinence).

In [4]:

```
vaccinations.country.unique()
```

Out[4]:

```
array(['Afghanistan', 'Albania', 'Algeria', 'Andorra', 'Angola',
       'Anguilla', 'Antigua and Barbuda', 'Argentina', 'Australia',
       'Austria', 'Azerbaijan', 'Bahrain', 'Bangladesh', 'Barbados',
       'Belarus', 'Belgium', 'Belize', 'Bermuda', 'Bolivia', 'Brazil',
       'Bulgaria', 'Cambodia', 'Canada', 'Cayman Islands', 'Chile',
       'China', 'Colombia', 'Costa Rica', "Cote d'Ivoire", 'Croatia',
       'Cyprus', 'Czechia', 'Denmark', 'Dominica', 'Dominican Republic',
       'Ecuador', 'Egypt', 'El Salvador', 'England', 'Estonia',
       'Faeroe Islands', 'Falkland Islands', 'Finland', 'France',
       'Germany', 'Ghana', 'Gibraltar', 'Greece', 'Greenland', 'Grenada',
       'Guatemala', 'Guernsey', 'Guyana', 'Honduras', 'Hong Kong',
       'Hungary', 'Iceland', 'India', 'Indonesia', 'Iran', 'Ireland',
       'Isle of Man', 'Israel', 'Italy', 'Jamaica', 'Japan', 'Jersey',
```

```
       'Jordan', 'Kazakhstan', 'Kenya', 'Kuwait', 'Laos', 'Latvia',
       'Lebanon', 'Liechtenstein', 'Lithuania', 'Luxembourg', 'Macao',
       'Malawi', 'Malaysia', 'Maldives', 'Malta', 'Mauritius', 'Mexico',
       'Moldova', 'Monaco', 'Mongolia', 'Montenegro', 'Montserrat',
       'Morocco', 'Myanmar', 'Nepal', 'Netherlands', 'New Zealand',
       'Nigeria', 'Northern Cyprus', 'Northern Ireland', 'Norway', 'Oman',
       'Pakistan', 'Panama', 'Paraguay', 'Peru', 'Philippines', 'Poland',
       'Portugal', 'Qatar', 'Romania', 'Russia', 'Rwanda', 'Saint Helena',
       'Saint Kitts and Nevis', 'Saint Lucia', 'San Marino',
       'Saudi Arabia', 'Scotland', 'Senegal', 'Serbia', 'Seychelles',
       'Singapore', 'Slovakia', 'Slovenia', 'South Africa', 'South Korea',
       'Spain', 'Sri Lanka', 'Sweden', 'Switzerland', 'Thailand',
       'Trinidad and Tobago', 'Tunisia', 'Turkey',
       'Turks and Caicos Islands', 'Uganda', 'Ukraine',
       'United Arab Emirates', 'United Kingdom', 'United States',
       'Uruguay', 'Venezuela', 'Vietnam', 'Wales', 'Zimbabwe'],
      dtype=object)
```

In [5]:

```
vaccinations = vaccinations[vaccinations.country.apply(lambda x : x not in ['Scotland',
'Wales', 'Northern Ireland', 'England', 'Isle of Man', 'Cayman Islands', 'Falkland Island
s', 'Guernsey', 'Saint Helena', 'Turks and Caicos Islands', 'Faeroe Islands', 'Gibraltar
'])]
vaccinations.country.unique()
```

Out[5]:

```
array(['Afghanistan', 'Albania', 'Algeria', 'Andorra', 'Angola',
       'Anguilla', 'Antigua and Barbuda', 'Argentina', 'Australia',
       'Austria', 'Azerbaijan', 'Bahrain', 'Bangladesh', 'Barbados',
       'Belarus', 'Belgium', 'Belize', 'Bermuda', 'Bolivia', 'Brazil',
       'Bulgaria', 'Cambodia', 'Canada', 'Chile', 'China', 'Colombia',
       'Costa Rica', "Cote d'Ivoire", 'Croatia', 'Cyprus', 'Czechia',
       'Denmark', 'Dominica', 'Dominican Republic', 'Ecuador', 'Egypt',
       'El Salvador', 'Estonia', 'Finland', 'France', 'Germany', 'Ghana',
       'Greece', 'Greenland', 'Grenada', 'Guatemala', 'Guyana',
       'Honduras', 'Hong Kong', 'Hungary', 'Iceland', 'India',
       'Indonesia', 'Iran', 'Ireland', 'Israel', 'Italy', 'Jamaica',
       'Japan', 'Jersey', 'Jordan', 'Kazakhstan', 'Kenya', 'Kuwait',
       'Laos', 'Latvia', 'Lebanon', 'Liechtenstein', 'Lithuania',
       'Luxembourg', 'Macao', 'Malawi', 'Malaysia', 'Maldives', 'Malta',
       'Mauritius', 'Mexico', 'Moldova', 'Monaco', 'Mongolia',
       'Montenegro', 'Montserrat', 'Morocco', 'Myanmar', 'Nepal',
       'Netherlands', 'New Zealand', 'Nigeria', 'Northern Cyprus',
       'Norway', 'Oman', 'Pakistan', 'Panama', 'Paraguay', 'Peru',
       'Philippines', 'Poland', 'Portugal', 'Qatar', 'Romania', 'Russia',
       'Rwanda', 'Saint Kitts and Nevis', 'Saint Lucia', 'San Marino',
       'Saudi Arabia', 'Senegal', 'Serbia', 'Seychelles', 'Singapore',
       'Slovakia', 'Slovenia', 'South Africa', 'South Korea', 'Spain',
       'Sri Lanka', 'Sweden', 'Switzerland', 'Thailand',
       'Trinidad and Tobago', 'Tunisia', 'Turkey', 'Uganda', 'Ukraine',
       'United Arab Emirates', 'United Kingdom', 'United States',
       'Uruguay', 'Venezuela', 'Vietnam', 'Zimbabwe'], dtype=object)
```

EDA

Vaccination progress track. What type of vaccines is used around the globe?

First of all, we will track the vaccination process by each vaccine, grouped by countries this combination of vaccines is used in, and sorted by *total_vaccinations*. For this, we'll need to know all the unique values the feature *vaccines* may take.

In [6]:

```
vaccinations.vaccines.unique()
```

Out[6]:

```
array(['Oxford/AstraZeneca', 'Pfizer/BioNTech', 'Sputnik V',
       'Oxford/AstraZeneca, Sinopharm/Beijing, Sputnik V',
       'Oxford/AstraZeneca, Pfizer/BioNTech',
       'Moderna, Oxford/AstraZeneca, Pfizer/BioNTech', 'Sinovac',
       'Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V',
       'Oxford/AstraZeneca, Sinovac', 'Sinopharm/Beijing',
       'Pfizer/BioNTech, Sinovac',
       'Sinopharm/Beijing, Sinopharm/Wuhan, Sinovac', 'Moderna',
       'Moderna, Oxford/AstraZeneca',
       'Moderna, Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V',
       'Covaxin, Oxford/AstraZeneca', 'Moderna, Pfizer/BioNTech',
       'Pfizer/BioNTech, Sinopharm/Beijing',
       'Pfizer/ BioNTech, Sinopharm/Beijing',
       'Oxford/AstraZeneca, Pfizer/BioNTech, Sputnik V',
       'Sinopharm/Beijing, Sputnik V',
       'Oxford/AstraZeneca, Sinopharm/Beijing', 'EpiVacCorona, Sputnik V',
       'Johnson&Johnson',
       'Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sinopharm/Wuhan, Sputnik 
V',
       'Johnson&Johnson, Moderna, Pfizer/BioNTech'], dtype=object)
```

Define *latest_data* as a DataFrame, which contains the same features, that the initial dataset, but using the latest vaccination data in each country.

In [7]:

```
columns = ['country', 'date', 'total_vaccinations', 'people_vaccinated', 'people_fully_v
accinated','people_fully_vaccinated_per_hundred','vaccines','iso_code']
latest_data = vaccinations[columns].groupby('country', as_index = True).max().sort_value
s(by='total_vaccinations', ascending = False)
latest_data.head()
```

Out[7]:

country	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	people_fully_vaccinated_per_hundred	
United States	2021-03-17	113037627.0	73669956.0	39989196.0	11.96	Johnso Pfize
China	2021-03-14	64980000.0	0.0	0.0	0.00	Sinoph Sinoph
India	2021-03-17	37143255.0	30600787.0	6542468.0	0.47	Oxford/A
United Kingdom	2021-03-16	27032671.0	25273226.0	1759445.0	2.59	Oxford/A Pfize
Brazil	2021-03-17	12682290.0	9451188.0	3231102.0	1.52	Oxford/A

Which combination of vaccines is used the most?

The initial dataset *vaccines* feature only contains a combination of many types of vaccines, so that, it's impossible to determine the number of vaccinations by each vaccine individually.

In [8]:

```
total_vaccines = pd.DataFrame()
vaccines = vaccinations.groupby('vaccines')
for col, group in vaccines:
    total_vaccines.loc[col,"total_vaccinations"] = group["daily_vaccinations"].sum()
total_vaccines = total_vaccines.sort_values(by='total_vaccinations', ascending = False)
```

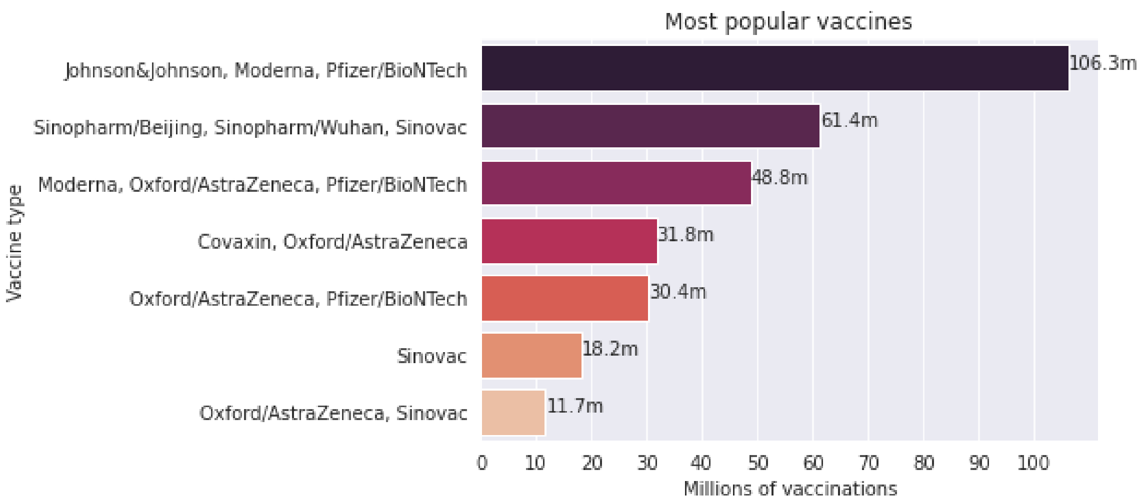
In [9]:

```
def show_values_on_bars(axs, h_v="v", space=0.7):
    def _show_on_single_plot(ax):
        if h_v == "v":
            for p in ax.patches:
                _x = p.get_x() + p.get_width() / 2
                _y = p.get_y() + p.get_height()
                value = '{:.0f}%'.format(int(p.get_height()))
                ax.text(_x, _y, value, ha="center")
        elif h_v == "h":
            for p in ax.patches:
                _x = p.get_x() + p.get_width() + float(space)
                _y = p.get_y() + p.get_height() / 2
                value = '{:.1f}m'.format(int(p.get_width()) / 1000000)
                ax.text(_x, _y, value, ha="left")

    if isinstance(axs, np.ndarray):
        for idx, ax in np.ndenumerate(axs):
            _show_on_single_plot(ax)
    else:
        _show_on_single_plot(axs)
```

In [10]:

```
sns.set_style('darkgrid')
ax = sns.barplot(x = total_vaccines.iloc[:7,:]['total_vaccinations'] , y =total_vaccines
.iloc[:7,:].index, palette="rocket")
ax.set_xlabel('Millions of vaccinations')
ax.set_ylabel('Vaccine type')
ax.set_title('Most popular vaccines')
plt.xticks(ticks = [0,10000000,20000000,30000000,40000000,50000000,60000000,70000000,8000
0000,90000000,100000000],labels= [0,10,20,30,40,50,60,70,80,90,100])
show_values_on_bars(ax,h_v = 'h', space = 0.7)
```



The top-3 vaccine groups by total_vaccinations are:

- **Johnson&Johnson, Moderna, Pfizer/BioNTech - 106.3 M vaccines**
- **Sinopharm/Beijing, Sinopharm/Wuhan, Sinovac - 61.4 M vaccines**
- **Moderna, Oxford/AstraZeneca, Pfizer/BioNTech - 48.8 M vaccines**

Pfizer/BioNTech

In [11]:

```
latest_data[latest_data['vaccines'] == 'Pfizer/BioNTech'][['vaccines','total_vaccination
s', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_values(by='total_vaccin
ations', ascending = False).head(15).style.background_gradient(cmap='Blues')
```

Out[11]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Singapore	Pfizer/BioNTech	792423.000000	549254.000000	243169.000000	2021-03-15 00:00:00
Slovakia	Pfizer/BioNTech	678316.000000	454974.000000	223342.000000	2021-03-16 00:00:00
Qatar	Pfizer/BioNTech	510000.000000	100000.000000	0.000000	2021-03-16 00:00:00
Japan	Pfizer/BioNTech	437485.000000	423196.000000	14289.000000	2021-03-17 00:00:00
Malaysia	Pfizer/BioNTech	367213.000000	367213.000000	0.000000	2021-03-17 00:00:00
Kuwait	Pfizer/BioNTech	360000.000000	322000.000000	38000.000000	2021-03-08 00:00:00
Croatia	Pfizer/BioNTech	300956.000000	233423.000000	67533.000000	2021-03-12 00:00:00
Panama	Pfizer/BioNTech	266298.000000	0.000000	0.000000	2021-03-16 00:00:00
Costa Rica	Pfizer/BioNTech	248082.000000	190088.000000	57994.000000	2021-03-15 00:00:00
Cyprus	Pfizer/BioNTech	148299.000000	116331.000000	31968.000000	2021-03-12 00:00:00
Ecuador	Pfizer/BioNTech	141191.000000	121054.000000	20137.000000	2021-03-15 00:00:00
Malta	Pfizer/BioNTech	130861.000000	90002.000000	40859.000000	2021-03-16 00:00:00
Lebanon	Pfizer/BioNTech	120890.000000	88219.000000	32671.000000	2021-03-17 00:00:00
Albania	Pfizer/BioNTech	33369.000000	6073.000000	655.000000	2021-03-17 00:00:00
Bermuda	Pfizer/BioNTech	30481.000000	18807.000000	11674.000000	2021-03-15 00:00:00

Sputnik V

In [12]:

```
latest_data[latest_data['vaccines'] == 'Sputnik V'][['vaccines','total_vaccinations', 'pe
ople_vaccinated', 'people_fully_vaccinated','date']].sort_values(by='total_vaccinations'
, ascending = False).head(15).style.background_gradient(cmap='Blues')
```

Out[12]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Bolivia	Sputnik V	150593.000000	140532.000000	10061.000000	2021-03-17 00:00:00
Kazakhstan	Sputnik V	87902.000000	69095.000000	18807.000000	2021-03-15 00:00:00
Algeria	Sputnik V	75000.000000	0.000000	0.000000	2021-02-19 00:00:00
Belarus	Sputnik V	30000.000000	20944.000000	10000.000000	2021-03-12 00:00:00
Paraguay	Sputnik V	12820.000000	12223.000000	0.000000	2021-03-17 00:00:00
Venezuela	Sputnik V	12194.000000	12194.000000	0.000000	2021-03-04 00:00:00
Iran	Sputnik V	10000.000000	10000.000000	0.000000	2021-02-17 00:00:00
San Marino	Sputnik V	6087.000000	6087.000000	0.000000	2021-03-16 00:00:00
Tunisia	Sputnik V	2555.000000	2555.000000	0.000000	2021-03-15 00:00:00

Oxford/AstraZeneca

In [13]:

```
latest_data[latest_data['vaccines'] == 'Oxford/AstraZeneca'][['vaccines','total_vaccinat
ions', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_values(by='total_vac
cinations', ascending = False).head(15).style.background_gradient(cmap='Blues')
```

Out[13]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Bangladesh	Oxford/AstraZeneca	4580391.000000	4580391.000000	0.000000	2021-03-16 00:00:00
Sri Lanka	Oxford/AstraZeneca	806449.000000	0.000000	0.000000	2021-03-17 00:00:00
Dominican Republic	Oxford/AstraZeneca	675000.000000	675000.000000	0.000000	2021-03-15 00:00:00
Nepal	Oxford/AstraZeneca	402264.000000	0.000000	0.000000	2021-02-20 00:00:00
Myanmar	Oxford/AstraZeneca	380000.000000	380000.000000	0.000000	2021-02-05 00:00:00
Ghana	Oxford/AstraZeneca	300000.000000	300000.000000	0.000000	2021-03-10 00:00:00
Maldives	Oxford/AstraZeneca	212711.000000	0.000000	0.000000	2021-03-15 00:00:00
Mongolia	Oxford/AstraZeneca	139636.000000	0.000000	0.000000	2021-03-17 00:00:00
Ukraine	Oxford/AstraZeneca	81756.000000	81755.000000	1.000000	2021-03-17 00:00:00
Barbados	Oxford/AstraZeneca	54631.000000	54631.000000	0.000000	2021-03-16 00:00:00
El Salvador	Oxford/AstraZeneca	36000.000000	36000.000000	0.000000	2021-03-17 00:00:00
Antigua and Barbuda	Oxford/AstraZeneca	24164.000000	0.000000	0.000000	2021-03-15 00:00:00
Vietnam	Oxford/AstraZeneca	24054.000000	24054.000000	0.000000	2021-03-17 00:00:00
Kenya	Oxford/AstraZeneca	20000.000000	20000.000000	0.000000	2021-03-17 00:00:00
Cote d'Ivoire	Oxford/AstraZeneca	19113.000000	19113.000000	0.000000	2021-03-17 00:00:00

Moderna, Oxford/AstraZeneca, Pfizer/BioNTech

In [14]:

```
latest_data[latest_data['vaccines'] == 'Moderna, Oxford/AstraZeneca, Pfizer/BioNTech'][[
'vaccines','total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated','date']].
sort_values(by='total_vaccinations', ascending = False).head(15).style.background_gradie
nt(cmap='Blues')
```

Out[14]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Germany	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	9853966.000000	6835216.000000	3018750.000000	2021-03-16 00:00:00
France	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	7552120.000000	5295735.000000	2256385.000000	2021-03-15 00:00:00
Italy	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	7204358.000000	4978706.000000	2225652.000000	2021-03-17 00:00:00
Spain	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	5857085.000000	4052470.000000	1804615.000000	2021-03-17 00:00:00
Poland	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	4605929.000000	2984642.000000	1621287.000000	2021-03-16 00:00:00

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Canada	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	3409996.000000	2799925.000000	610071.000000	2021-03-17 00:00:00
Romania	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	2270124.000000	1530483.000000	739641.000000	2021-03-16 00:00:00
Netherlands	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	1887726.000000	1394603.000000	493123.000000	2021-03-14 00:00:00
Greece	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	1345735.000000	915641.000000	430094.000000	2021-03-17 00:00:00
Belgium	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	1233120.000000	832140.000000	400980.000000	2021-03-16 00:00:00
Czechia	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	1168025.000000	849354.000000	318671.000000	2021-03-16 00:00:00
Austria	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	1109315.000000	823998.000000	285317.000000	2021-03-16 00:00:00
Denmark	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	873894.000000	595943.000000	277951.000000	2021-03-16 00:00:00
Finland	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	723154.000000	636349.000000	86805.000000	2021-03-17 00:00:00
Norway	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	709159.000000	451308.000000	257851.000000	2021-03-16 00:00:00

Oxford/AstraZeneca, Sputnik V

In [15]:

```
latest_data[latest_data['vaccines'] == 'Oxford/AstraZeneca, Sputnik V'][['vaccines','tota
l_vaccinations', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_values(by=
'total_vaccinations', ascending = False).head(15).style.background_gradient(cmap='Blues'
)
```

Out[15]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date

Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V

In [16]:

```
latest_data[latest_data['vaccines'] == 'Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Be
ijing, Sputnik V'][['vaccines','total_vaccinations', 'people_vaccinated', 'people_fully_v
accinated','date']].sort_values(by='total_vaccinations', ascending = False).head(15).sty
le.background_gradient(cmap='Blues')
```

Out[16]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Serbia	Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V	2077197.000000	1267822.000000	809375.000000	2021-03-17 00:00:00

	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Bahrain	Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V	597446.000000	374873.000000	222573.000000	2021-03-17 00:00:00

Oxford/AstraZeneca, Sinovac

In [17]:

```
latest_data[latest_data['vaccines'] == 'Oxford/AstraZeneca, Sinovac'][['vaccines','total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_values(by='total_vaccinations', ascending = False).head(15).style.background_gradient(cmap='Blues')
```

Out[17]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Brazil	Oxford/AstraZeneca, Sinovac	12682290.000000	9451188.000000	3231102.000000	2021-03-17 00:00:00
Thailand	Oxford/AstraZeneca, Sinovac	53842.000000	53842.000000	0.000000	2021-03-13 00:00:00

Sinopharm/Beijing

In [18]:

```
latest_data[latest_data['vaccines'] == 'Sinopharm/Beijing'][['vaccines','total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_values(by='total_vaccinations', ascending = False).head(15).style.background_gradient(cmap='Blues')
```

Out[18]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Peru	Sinopharm/Beijing	582457.000000	410062.000000	172395.000000	2021-03-17 00:00:00
Cambodia	Sinopharm/Beijing	170659.000000	170659.000000	0.000000	2021-03-15 00:00:00
Senegal	Sinopharm/Beijing	144207.000000	144207.000000	0.000000	2021-03-16 00:00:00
Zimbabwe	Sinopharm/Beijing	39607.000000	39607.000000	0.000000	2021-03-17 00:00:00
Egypt	Sinopharm/Beijing	1315.000000	0.000000	0.000000	2021-01-30 00:00:00

Moderna, Pfizer/BioNTech

In [19]:

```
latest_data[latest_data['vaccines'] == 'Moderna, Pfizer/BioNTech'][['vaccines','total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_values(by='total_vaccinations', ascending = False).head(15).style.background_gradient(cmap='Blues')
```

Out[19]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Israel	Moderna, Pfizer/BioNTech	9560280.000000	5153197.000000	4407083.000000	2021-03-17 00:00:00
Portugal	Moderna, Pfizer/BioNTech	1200691.000000	851022.000000	349669.000000	2021-03-17 00:00:00
Switzerland	Moderna, Pfizer/BioNTech	1097977.000000	703872.000000	394105.000000	2021-03-14 00:00:00
Liechtenstein	Moderna, Pfizer/BioNTech	3776.000000	0.000000	0.000000	2021-03-14 00:00:00

	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
	Pfizer/BioNTech				

Pfizer/BioNTech, Sinovac

In [20]:

```
latest_data[latest_data['vaccines'] == 'Pfizer/BioNTech, Sinovac'][['vaccines','total_va
ccinations', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_values(by='tot
al_vaccinations', ascending = False).head(15).style.background_gradient(cmap='Blues')
```

Out[20]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Chile	Pfizer/BioNTech, Sinovac	7779529.000000	5274603.000000	2504926.000000	2021-03-17 00:00:00
Colombia	Pfizer/BioNTech, Sinovac	976137.000000	928927.000000	47210.000000	2021-03-17 00:00:00
Hong Kong	Pfizer/BioNTech, Sinovac	253900.000000	253900.000000	0.000000	2021-03-17 00:00:00
Uruguay	Pfizer/BioNTech, Sinovac	223265.000000	223265.000000	0.000000	2021-03-17 00:00:00
Northern Cyprus	Pfizer/BioNTech, Sinovac	11000.000000	0.000000	0.000000	2021-01-22 00:00:00

Sinopharm/Beijing, Sinopharm/Wuhan, Sinovac

In [21]:

```
latest_data[latest_data['vaccines'] == 'Sinopharm/Beijing, Sinopharm/Wuhan, Sinovac'][['
vaccines','total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated','date']].s
ort_values(by='total_vaccinations', ascending = False).head(15).style.background_gradien
t(cmap='Blues')
```

Out[21]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
China	Sinopharm/Beijing, Sinopharm/Wuhan, Sinovac	64980000.000000	0.000000	0.000000	2021-03-14 00:00:00

Oxford/AstraZeneca, Pfizer/BioNTech

In [22]:

```
latest_data[latest_data['vaccines'] == 'Oxford/AstraZeneca, Pfizer/BioNTech'][['vaccines
','total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_valu
es(by='total_vaccinations', ascending = False).head(15).style.background_gradient(cmap='
Blues')
```

Out[22]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
United Kingdom	Oxford/AstraZeneca, Pfizer/BioNTech	27032671.000000	25273226.000000	1759445.000000	2021-03-16 00:00:00
Saudi Arabia	Oxford/AstraZeneca, Pfizer/BioNTech	2577630.000000	0.000000	0.000000	2021-03-17 00:00:00
Sweden	Oxford/AstraZeneca,	1223422.000000	861525.000000	361897.000000	2021-03-17

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
South Korea	Oxford/AstraZeneca, Pfizer/BioNTech	641331.000000	641331.000000	0.000000	2021-03-17 00:00:00
Slovenia	Oxford/AstraZeneca, Pfizer/BioNTech	262121.000000	176956.000000	85165.000000	2021-03-16 00:00:00
Australia	Oxford/AstraZeneca, Pfizer/BioNTech	203557.000000	159294.000000	0.000000	2021-03-17 00:00:00
Oman	Oxford/AstraZeneca, Pfizer/BioNTech	109844.000000	90825.000000	19019.000000	2021-03-17 00:00:00
Jersey	Oxford/AstraZeneca, Pfizer/BioNTech	42231.000000	38073.000000	4158.000000	2021-03-10 00:00:00

Sinovac

In [23]:

```
latest_data[latest_data['vaccines'] == 'Sinovac'][['vaccines','total_vaccinations', 'peo
ple_vaccinated', 'people_fully_vaccinated','date']].sort_values(by='total_vaccinations',
ascending = False).head(15).style.background_gradient(cmap='Blues')
```

Out[23]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Turkey	Sinovac	12142245.000000	7977222.000000	4165023.000000	2021-03-17 00:00:00
Indonesia	Sinovac	6581388.000000	4705248.000000	1876140.000000	2021-03-17 00:00:00
Azerbaijan	Sinovac	453586.000000	453586.000000	0.000000	2021-03-14 00:00:00
Philippines	Sinovac	215997.000000	215997.000000	0.000000	2021-03-15 00:00:00
Laos	Sinovac	40732.000000	40732.000000	0.000000	2021-03-17 00:00:00

Moderna

In [24]:

```
latest_data[latest_data['vaccines'] == 'Moderna'][['vaccines','total_vaccinations', 'peo
ple_vaccinated', 'people_fully_vaccinated','date']].sort_values(by='total_vaccinations',
ascending = False).head(15).style.background_gradient(cmap='Blues')
```

Out[24]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Guatemala	Moderna	48130.000000	48130.000000	0.000000	2021-03-16 00:00:00

Moderna, Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V

In [25]:

```
latest_data[latest_data['vaccines'] == 'Moderna, Oxford/AstraZeneca, Pfizer/BioNTech, Sin
opharm/Beijing, Sputnik V'][['vaccines','total_vaccinations', 'people_vaccinated', 'peopl
e_fully_vaccinated','date']].sort_values(by='total_vaccinations', ascending = False).hea
d(15).style.background_gradient(cmap='Blues')
```

Out[25]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Hungary	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V	1863621.000000	1441706.000000	421915.000000	2021-03-17 00:00:00

Covaxin, Oxford/AstraZeneca

In [26]:

```
latest_data[latest_data['vaccines'] == 'Covaxin, Oxford/AstraZeneca'][['vaccines','total
_vaccinations', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_values(by='
total_vaccinations', ascending = False).head(15).style.background_gradient(cmap='Blues')
```

Out[26]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
India	Covaxin, Oxford/AstraZeneca	37143255.000000	30600787.000000	6542468.000000	2021-03-17 00:00:00

Pfizer/BioNTech, Sinopharm/Beijing

In [27]:

```
latest_data[latest_data['vaccines'] == 'Pfizer/BioNTech, Sinopharm/Beijing'][['vaccines'
,'total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_value
s(by='total_vaccinations', ascending = False).head(15).style.background_gradient(cmap='B
lues')
```

Out[27]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Jordan	Pfizer/BioNTech, Sinopharm/Beijing	241868.000000	189456.000000	52412.000000	2021-03-17 00:00:00

Oxford/AstraZeneca, Pfizer/BioNTech, Sputnik V

In [28]:

```
latest_data[latest_data['vaccines'] == 'Oxford/AstraZeneca, Pfizer/BioNTech, Sputnik V'][
['vaccines','total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated','date']]
.sort_values(by='total_vaccinations', ascending = False).head(15).style.background_gradi
ent(cmap='Blues')
```

Out[28]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Mexico	Oxford/AstraZeneca, Pfizer/BioNTech, Sputnik V	4737622.000000	4110741.000000	626881.000000	2021-03-17 00:00:00

Sinopharm/Beijing, Sputnik V

In [29]:

```
latest_data[latest_data['vaccines'] == 'Sinopharm/Beijing, Sputnik V'][['vaccines','total
_vaccinations', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_values(by='
total_vaccinations', ascending = False).head(15).style.background_gradient(cmap='Blues')
```

Out[29]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Montenegro	Sinopharm/Beijing, Sputnik V	5730.000000	5730.000000	0.000000	2021-03-17 00:00:00

Oxford/AstraZeneca, Sinopharm/Beijing

In [30]:

```
latest_data[latest_data['vaccines'] == 'Oxford/AstraZeneca, Sinopharm/Beijing'][['vaccine
s','total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_val
ues(by='total_vaccinations', ascending = False).head(15).style.background_gradient(cmap=
'Blues')
```

Out[30]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Morocco	Oxford/AstraZeneca, Sinopharm/Beijing	6360732.000000	4244651.000000	2116081.000000	2021-03-17 00:00:00
Seychelles	Oxford/AstraZeneca, Sinopharm/Beijing	89509.000000	62067.000000	27442.000000	2021-03-15 00:00:00

Oxford/AstraZeneca, Sinopharm/Beijing, Sputnik V

In [31]:

```
latest_data[latest_data['vaccines'] == 'Oxford/AstraZeneca, Sinopharm/Beijing, Sputnik V'
][['vaccines','total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated','date'
]].sort_values(by='total_vaccinations', ascending = False).head(15).style.background_gra
dient(cmap='Blues')
```

Out[31]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Argentina	Oxford/AstraZeneca, Sinopharm/Beijing, Sputnik V	2668103.000000	2170116.000000	497987.000000	2021-03-17 00:00:00
Pakistan	Oxford/AstraZeneca, Sinopharm/Beijing, Sputnik V	350000.000000	72882.000000	0.000000	2021-03-14 00:00:00

EpiVacCorona, Sputnik V

In [32]:

```
latest_data[latest_data['vaccines'] == 'EpiVacCorona, Sputnik V'][['vaccines','total_vacc
inations', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_values(by='total
_vaccinations', ascending = False).head(15).style.background_gradient(cmap='Blues')
```

Out[32]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
Russia	EpiVacCorona, Sputnik V	7818009.000000	5545133.000000	2438650.000000	2021-03-17 00:00:00

Johnson&Johnson

In [33]:

```
latest_data[latest_data['vaccines'] == 'Johnson&Johnson'][['vaccines','total_vaccination
s', 'people_vaccinated', 'people_fully_vaccinated','date']].sort_values(by='total_vaccin
ations', ascending = False).head(15).style.background_gradient(cmap='Blues')
```

Out[33]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
South Africa	Johnson&Johnson	168413.000000	168413.000000	168413.000000	2021-03-17 00:00:00

Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sinopharm/Wuhan, Sputnik V

In [34]:

```
latest_data[latest_data['vaccines'] == 'Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Be
ijing, Sinopharm/Wuhan, Sputnik V'][['vaccines','total_vaccinations', 'people_vaccinated'
, 'people_fully_vaccinated','date']].sort_values(by='total_vaccinations', ascending = Fa
lse).head(15).style.background_gradient(cmap='Blues')
```

Out[34]:

country	vaccines	total_vaccinations	people_vaccinated	people_fully_vaccinated	date
United Arab Emirates	Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sinopharm/Wuhan, Sputnik V	6830369.000000	3480415.000000	2187849.000000	2021-03-17 00:00:00

Vaccine type conclusions

The initial dataset *vaccines* feature only contains a combination of many types of vaccines, so that, it's impossible to determine the number of vaccinations by each vaccine individually. However, we could see, that all of the countries use a different combination of vaccines. We can see common patterns in vaccine type use only in Asiatic sector (Philippines, Indonesia, Thailand) and European sector (see *Moderna, Oxford/AstraZeneca, Pfizer/BioNTech* section). This can be explained by foreign policy, overall financial state of a country (e.g. GPD per capita).

In [35]:

```
# Sns settings
c = '#386B7F'
sns.set_style('darkgrid')
```

In which country the vaccination programme is more advanced?

This can be shown from different perspectives. One of it is by viewing people fully vaccinated per hundred.

Countries by most fully vaccinated people per hundred

In [36]:

```
latest_data[['people_fully_vaccinated_per_hundred','people_fully_vaccinated', 'date' ]].s
ort_values(by='people_fully_vaccinated_per_hundred', ascending = False).head(30).style.b
ackground_gradient(cmap = 'Blues')
```

Out[36]:

country	people_fully_vaccinated_per_hundred	people_fully_vaccinated	date
Israel	50.920000	4407083.000000	2021-03-17 00:00:00
Seychelles	27.910000	27442.000000	2021-03-15 00:00:00
United Arab Emirates	22.120000	2187849.000000	2021-03-17 00:00:00
Bermuda	18.750000	11674.000000	2021-03-15 00:00:00
Monaco	18.250000	7163.000000	2021-03-04 00:00:00
Chile	13.100000	2504926.000000	2021-03-17 00:00:00
Bahrain	13.080000	222573.000000	2021-03-17 00:00:00
United States	11.960000	39989196.000000	2021-03-17 00:00:00
Serbia	11.890000	809375.000000	2021-03-17 00:00:00
Malta	9.250000	40859.000000	2021-03-16 00:00:00
Morocco	5.730000	2116081.000000	2021-03-17 00:00:00
Turkey	4.940000	4165023.000000	2021-03-17 00:00:00
Denmark	4.800000	277951.000000	2021-03-16 00:00:00
Norway	4.760000	257851.000000	2021-03-16 00:00:00
Switzerland	4.550000	394105.000000	2021-03-14 00:00:00
Hungary	4.370000	421915.000000	2021-03-17 00:00:00
Poland	4.280000	1621287.000000	2021-03-16 00:00:00
Singapore	4.160000	243169.000000	2021-03-15 00:00:00
Greece	4.130000	430094.000000	2021-03-17 00:00:00
Jersey	4.110000	4158.000000	2021-03-10 00:00:00
Slovenia	4.100000	85165.000000	2021-03-16 00:00:00
Slovakia	4.090000	223342.000000	2021-03-16 00:00:00
Estonia	4.050000	53729.000000	2021-03-17 00:00:00
Iceland	3.960000	13522.000000	2021-03-17 00:00:00
Spain	3.860000	1804615.000000	2021-03-17 00:00:00
Lithuania	3.850000	104771.000000	2021-03-16 00:00:00
Romania	3.840000	739641.000000	2021-03-16 00:00:00
Italy	3.680000	2225652.000000	2021-03-17 00:00:00
Cyprus	3.650000	31968.000000	2021-03-12 00:00:00
Germany	3.600000	3018750.000000	2021-03-16 00:00:00

Top-15 countries barplot by fully vaccinated people per hundred

In [37]:

```
latest_data[['people_fully_vaccinated_per_hundred', 'people_fully_vaccinated']].sort_valu
es(by='people_fully_vaccinated', ascending = False).head(10).style.background_gradient(c
map = 'Blues')
```

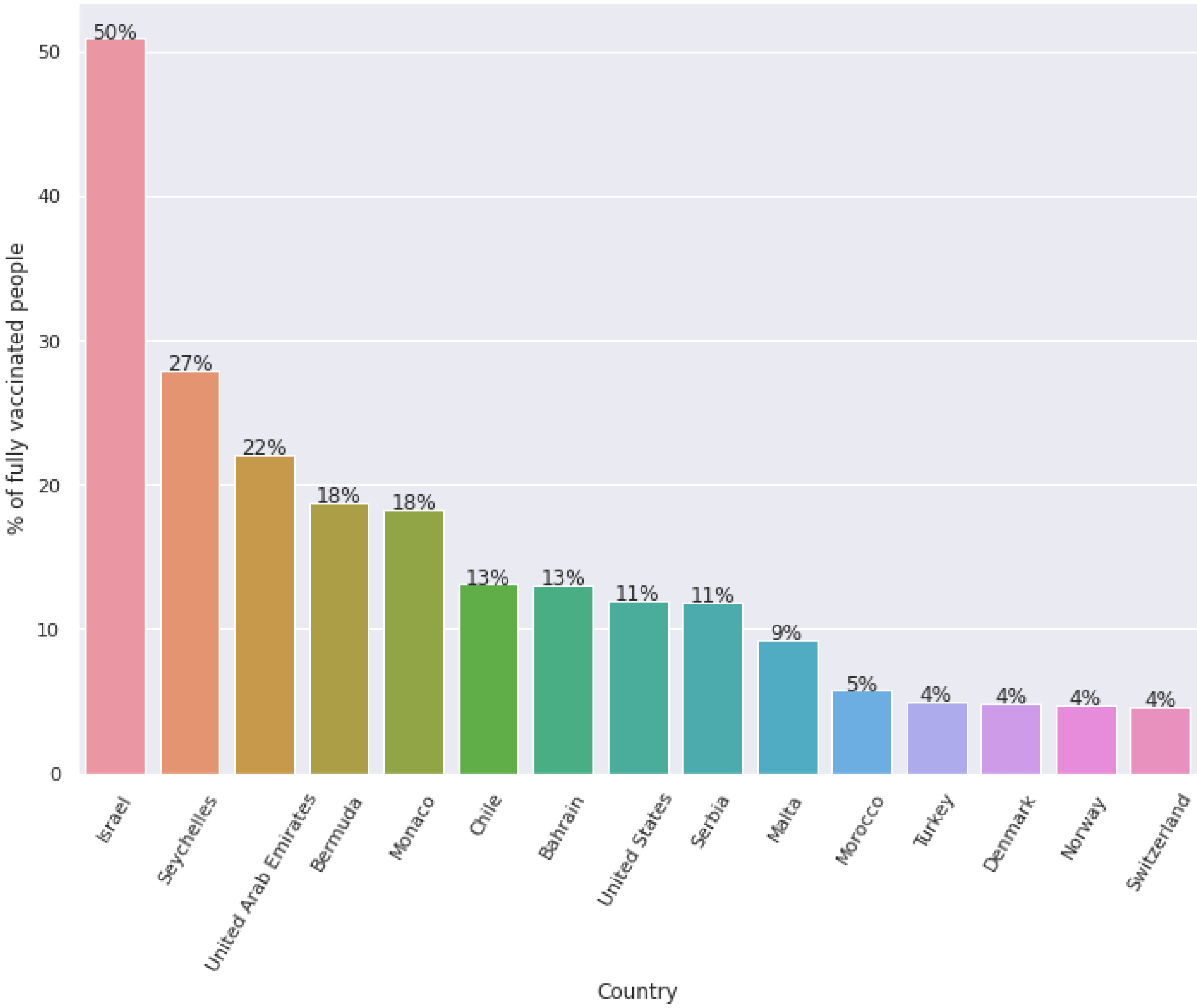
Out[37]:

country	people_fully_vaccinated_per_hundred	people_fully_vaccinated
United States	11.960000	39989196.000000
India	0.470000	6542468.000000
Israel	50.920000	4407083.000000

country	people_fully_vaccinated_per_hundred	people_fully_vaccinated
Turkey	4.040000	4165023.000000
Brazil	1.520000	3231102.000000
Germany	3.600000	3018750.000000
Chile	13.100000	2504926.000000
Russia	1.670000	2438650.000000
France	3.310000	2256385.000000
Italy	3.680000	2225652.000000

In [38]:

```
sns.set(rc={'figure.figsize':(11.7,8.27)}, style = 'darkgrid')
ax3 = sns.barplot(x = latest_data['people_fully_vaccinated_per_hundred'].sort_values(asce
nding = False).head(15).index,
                  y = latest_data['people_fully_vaccinated_per_hundred'].sort_values(asce
nding = False).head(15))
ax3.set_xlabel('Country')
ax3.set_ylabel('% of fully vaccinated people')
plt.xticks(rotation = 60)
show_values_on_bars(ax3, h_v = 'v')
```



Israel has the highest COVID-19 vaccination rate among the countries in the dataset, having administered 50 fully vaccinated per 100 people in the country. Comparing this statement with the total number of fully vaccinated people, it's clear that Israel is relatively small, achieving 50% vaccination of all its' population by vaccinating 4.4 million people. Notice that USA is the most populated country (328,2 millions of habitants in 2019) and it's still ranked in top-15 countries with the highest full vaccination rate.

In [39]:

```
def show_values_on_bars(axs, h_v="v", space=0.7):
```

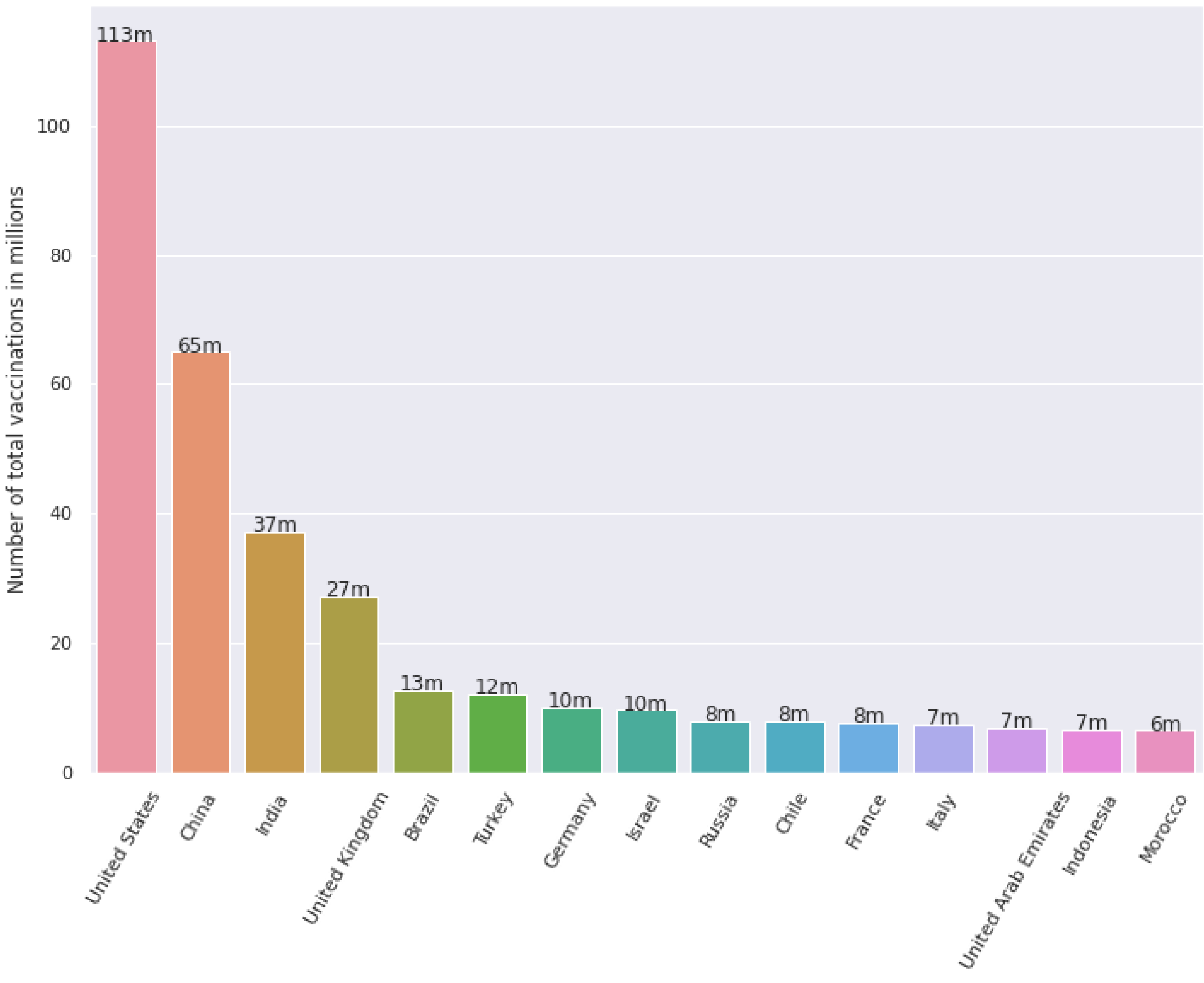
```
    def _show_on_single_plot(ax):
        if h_v == "v":
            for p in ax.patches:
                _x = p.get_x() + p.get_width() / 2
                _y = p.get_y() + p.get_height()
                value = '{:.0f}m'.format(int(p.get_height()) / 1000000)
                ax.text(_x, _y, value, ha="center")
        elif h_v == "h":
            for p in ax.patches:
                _x = p.get_x() + p.get_width() + float(space)
                _y = p.get_y() + p.get_height() / 2
                value = '{:.1f}m'.format(int(p.get_width()) / 1000000)
                ax.text(_x, _y, value, ha="left")

    if isinstance(axs, np.ndarray):
        for idx, ax in np.ndenumerate(axs):
            _show_on_single_plot(ax)
    else:
        _show_on_single_plot(axs)
```

Top-15 countries barplot by total vaccinations

In [40]:

```
sns.set(rc={'figure.figsize':(11.7,8.27)}, style = 'darkgrid')
ax2 = sns.barplot(x = latest_data['total_vaccinations'].sort_values(ascending = False).h
ead(15).index,
                  y = latest_data['total_vaccinations'].sort_values(ascending = False).he
ad(15))
ax2.set_xlabel('Country')
ax2.set_ylabel('Number of total vaccinations in millions')
plt.yticks(ticks = [0,20000000,40000000,60000000,80000000,100000000], labels = [0,20,40,
60,80,100])
plt.xticks(rotation = 60)
show_values_on_bars(ax2, h_v = 'v')
```



Country

Vaccines distribution worldmap

To draw a choropleth map of the vaccination percentage in each country, we will define 5 labels of vaccination progress:

- **very low (0-2.5%)**
- **low (2.5-5%)**
- **moderated (5-7.5%)**
- **medium(7.5-10%)**
- **high (10%+)**

In [41]:

```
conditions = [(latest_data['people_fully_vaccinated_per_hundred'] <= 2.5),
              (latest_data['people_fully_vaccinated_per_hundred'] > 2.5) & (latest_data[
'people_fully_vaccinated_per_hundred'] <= 5),
              (latest_data['people_fully_vaccinated_per_hundred'] > 5) & (latest_data['p
eople_fully_vaccinated_per_hundred'] <= 7.5),
              (latest_data['people_fully_vaccinated_per_hundred'] > 7.5) & (latest_data[
'people_fully_vaccinated_per_hundred'] <= 10),
             (latest_data['people_fully_vaccinated_per_hundred'] > 10)]
values = ['very low','low', 'moderated', 'medium', 'high']
new_data = latest_data
new_data['vaccination_level'] = np.select(conditions, values)
new_data.head()
```

Out[41]:

	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	people_fully_vaccinated_per_hundred	
country						
United States	2021-03-17	113037627.0	73669956.0	39989196.0	11.96	Johnso Pfize
China	2021-03-14	64980000.0	0.0	0.0	0.00	Sinoph Sinoph
India	2021-03-17	37143255.0	30600787.0	6542468.0	0.47	Oxford/
United Kingdom	2021-03-16	27032671.0	25273226.0	1759445.0	2.59	Oxford/A Pfize
Brazil	2021-03-17	12682290.0	9451188.0	3231102.0	1.52	Oxford/A

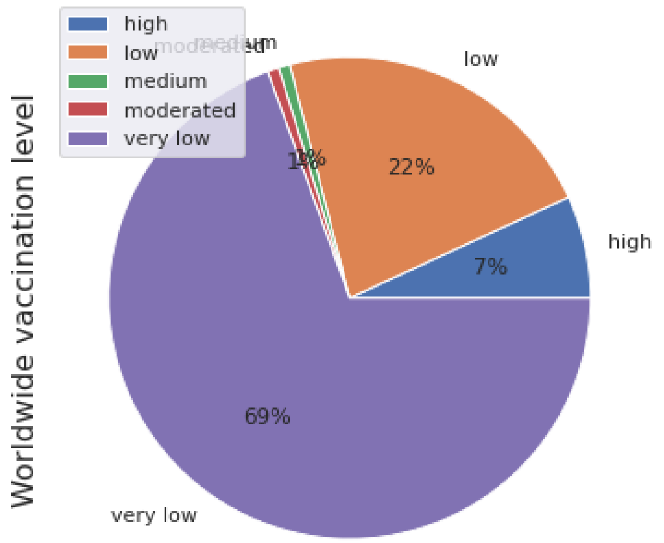
In [42]:

```
fig = px.choropleth(new_data, locations = 'iso_code', color = 'vaccination_level', featu
reidkey = 'properties.name', hover_name = new_data.index, hover_data = ['people_fully_va
ccinated_per_hundred', 'total_vaccinations'], title = 'Global vaccination progress')
fig.update_layout(title_x = 0.5)
fig.show()
```

COVID-19 Vaccination level pie-chart

In [43]:

```
fig, ax = plt.subplots(figsize=(8, 6))
new_data.groupby('vaccination_level').size().plot(kind='pie', autopct='%1.0f%%', ax = ax
)
ax.set_ylabel('Worldwide vaccination level', fontsize = 16)
plt.legend()
plt.show()
```



As we can see, there are still not so many countries with high vaccination rates. This means it will take some time to vaccinate more people. We will try to know how much time will approximatelly take.

Vaccination time series analysis

As object of research I've choosen the daily worldwide vaccinations. To operate with this feature, first, we'll need to create it.

In [44]:

```
daily_data = vaccinations[['daily_vaccinations','date']].groupby('date',as_index = False
).sum()
```

```
daily_data['vac_test'] = daily_data['daily_vaccinations']
daily_data.head()
```

Out[44]:

	date	daily_vaccinations	vac_test
0	2020-12-13	0.0	0.0
1	2020-12-14	84117.0	84117.0
2	2020-12-15	84835.0	84835.0
3	2020-12-16	276483.0	276483.0
4	2020-12-17	277373.0	277373.0

We are dealing now with time series data, so it's probably usefull to add new date features to the existing dataset.

In [45]:

```
daily_data['year'] = daily_data['date'].dt.year
daily_data['month'] = daily_data['date'].dt.month
daily_data['day'] = daily_data['date'].dt.day
daily_data.tail()
```

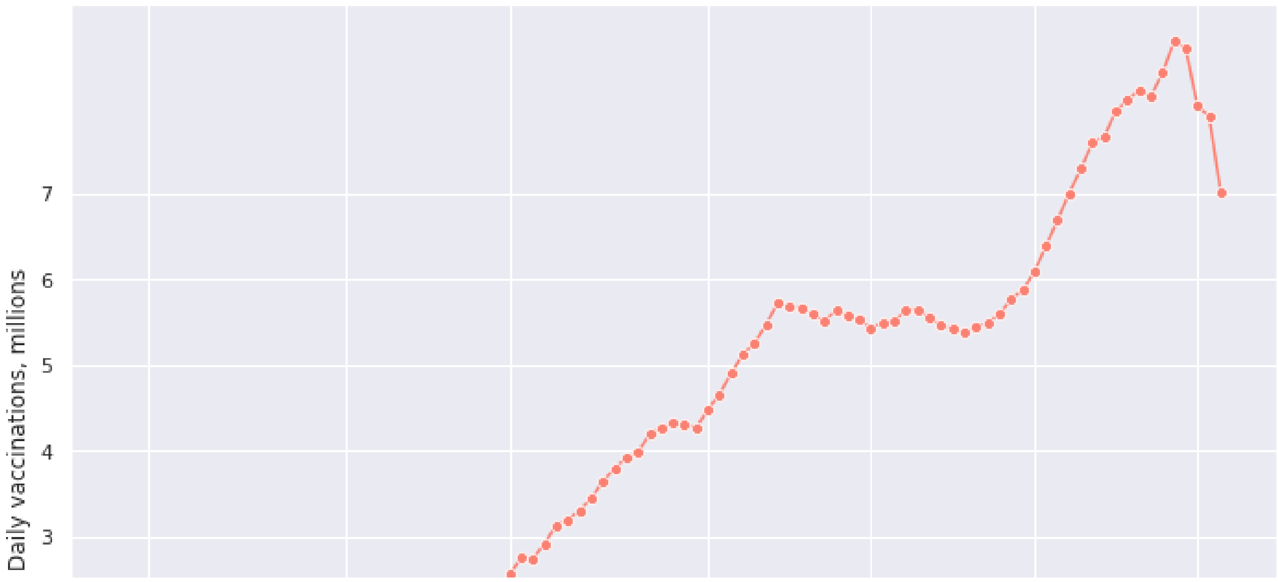
Out[45]:

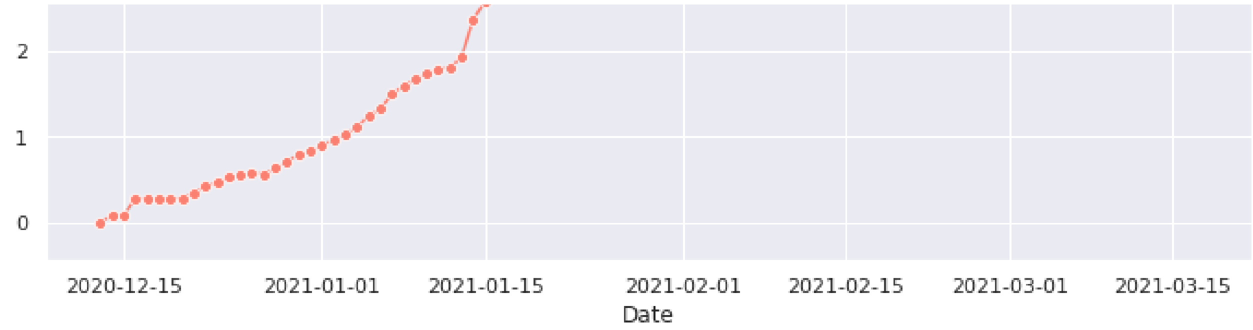
	date	daily_vaccinations	vac_test	year	month	day
90	2021-03-13	8766219.0	8766219.0	2021	3	13
91	2021-03-14	8674863.0	8674863.0	2021	3	14
92	2021-03-15	8011969.0	8011969.0	2021	3	15
93	2021-03-16	7890040.0	7890040.0	2021	3	16
94	2021-03-17	7003644.0	7003644.0	2021	3	17

Global trend

In [46]:

```
ax = sns.lineplot(x = daily_data['date'], y = daily_data['daily_vaccinations'], color='s
almon', marker='o')
ax.set_xlabel('Date')
plt.yticks(ticks = [0, 1000000, 2000000, 3000000, 4000000, 5000000, 6000000, 7000000], l
abels = [0, 1, 2, 3, 4, 5, 6, 7])
ax.set_ylabel('Daily vaccinations, millions')
plt.show()
```





Monthly trend

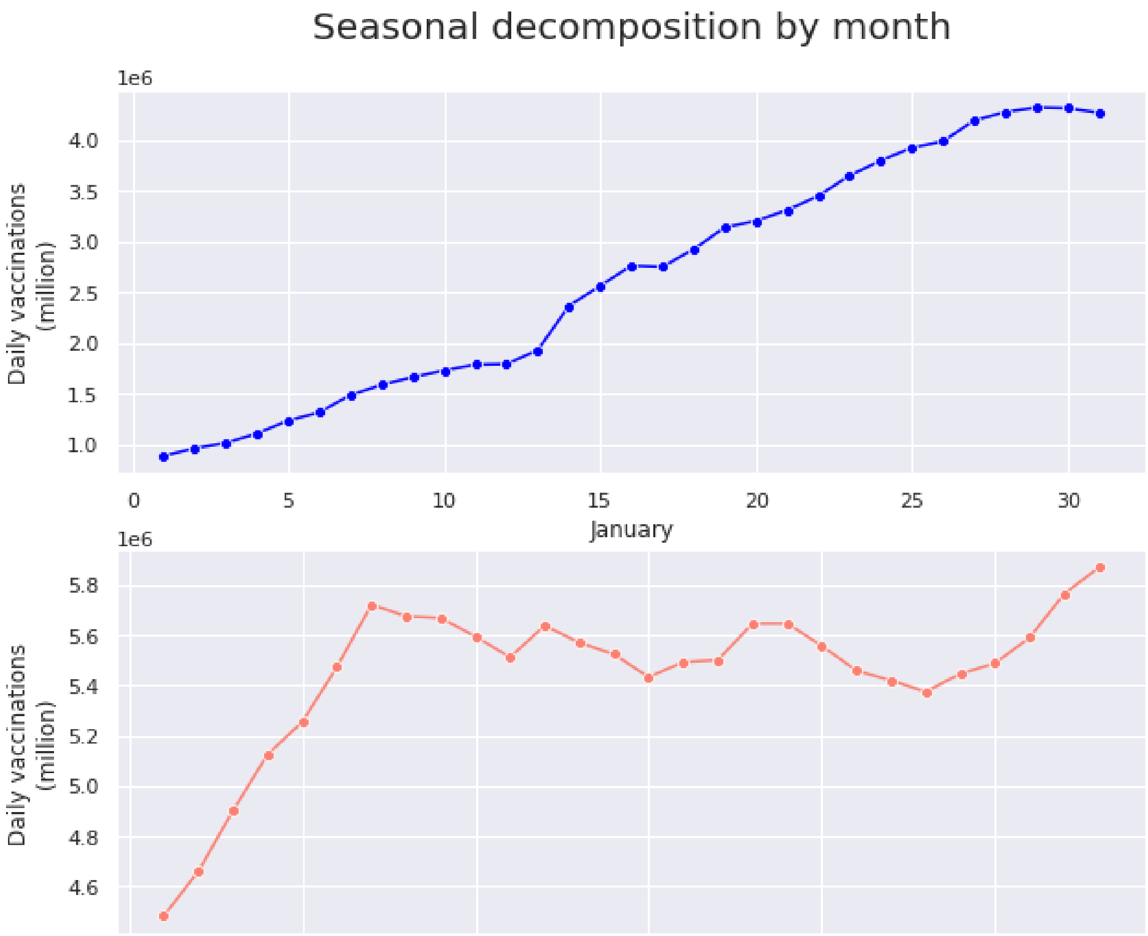
In [47]:

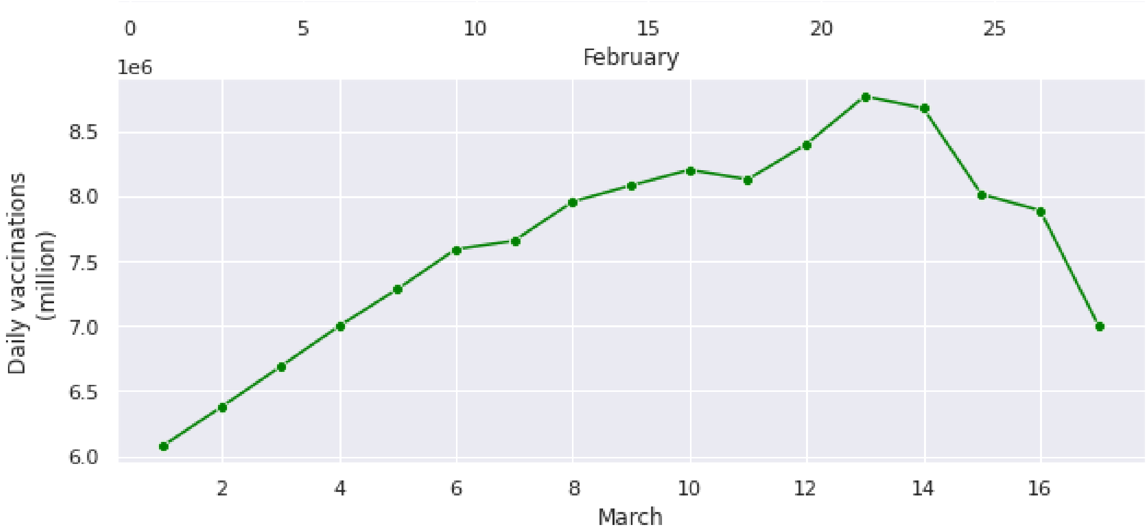
```
jan = daily_data[daily_data['month'] == 1]
feb = daily_data[daily_data['month'] == 2]
mar = daily_data[daily_data['month'] == 3]
```

In [48]:

```
# define figure and subplots
fig, (ax1, ax2, ax3) = plt.subplots(3, figsize = (10,13))
# creating subplot 1
sns.lineplot(x = jan['day'], y = jan['daily_vaccinations'], color='blue', marker='o', ax
=ax1)
ax1.set(xlabel = 'January', ylabel = 'Daily vaccinations\n(million)')
ax1.set_title('Seasonal decomposition by month\n', fontsize = 20)
# creating subplot 2
sns.lineplot(x = feb['day'], y = feb['daily_vaccinations'], color='salmon', marker='o',
ax=ax2)
ax2.set(xlabel = 'February', ylabel = 'Daily vaccinations\n(million)')
# creating subplot 3
sns.lineplot(x = mar['day'], y = mar['daily_vaccinations'], color='green', marker='o', a
x=ax3)
ax3.set(xlabel = 'March', ylabel = 'Daily vaccinations\n(million)')

plt.show()
```

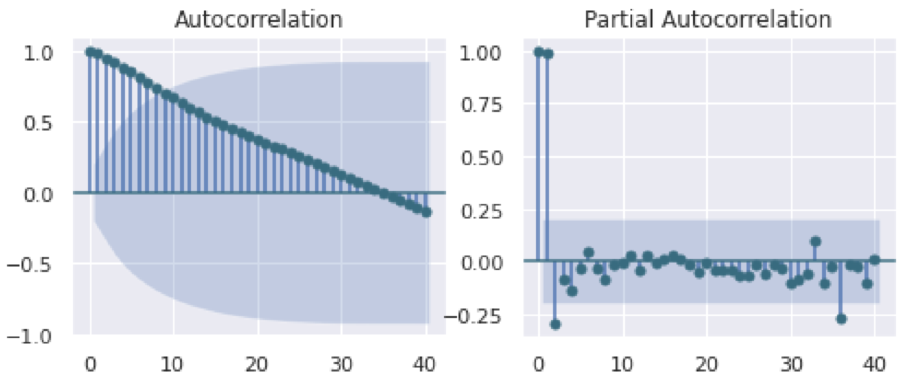




Stationarity analysis

In [49]:

```
plt.figure(figsize = (8,3))
plt.subplot(121); plot_acf(daily_data['daily_vaccinations'], lags = 40, ax = plt.gca(),
color = c)
plt.subplot(122); plot_pacf(daily_data['daily_vaccinations'], lags = 40,ax = plt.gca(),
color = c)
plt.show()
```



When analyzing the ACF, we can see the values to slowly decrease to zero. This could be the first signal of a series to be non-stationary on mean. Following the general lineplot data, the series reflect an increasing trend. Let's check it with the ADF test.

Dickey-Fuler test

It's used to check if the total vaccinations feature represents a stationary of a time series.

$$H_0 = \phi = 0$$

$$H_1 = \phi \neq 0$$

$\phi = 0$ **means our time series is a random walk process, while if** $\phi \neq 0$ **(**$-1 < 1 + \phi < 1$**) we get a stationary process. This way we need the p-value to be less than 0,05 to proceed with ACF and PACF.**

In [50]:

```
adfuller(daily_data['daily_vaccinations'])
```

Out[50]:

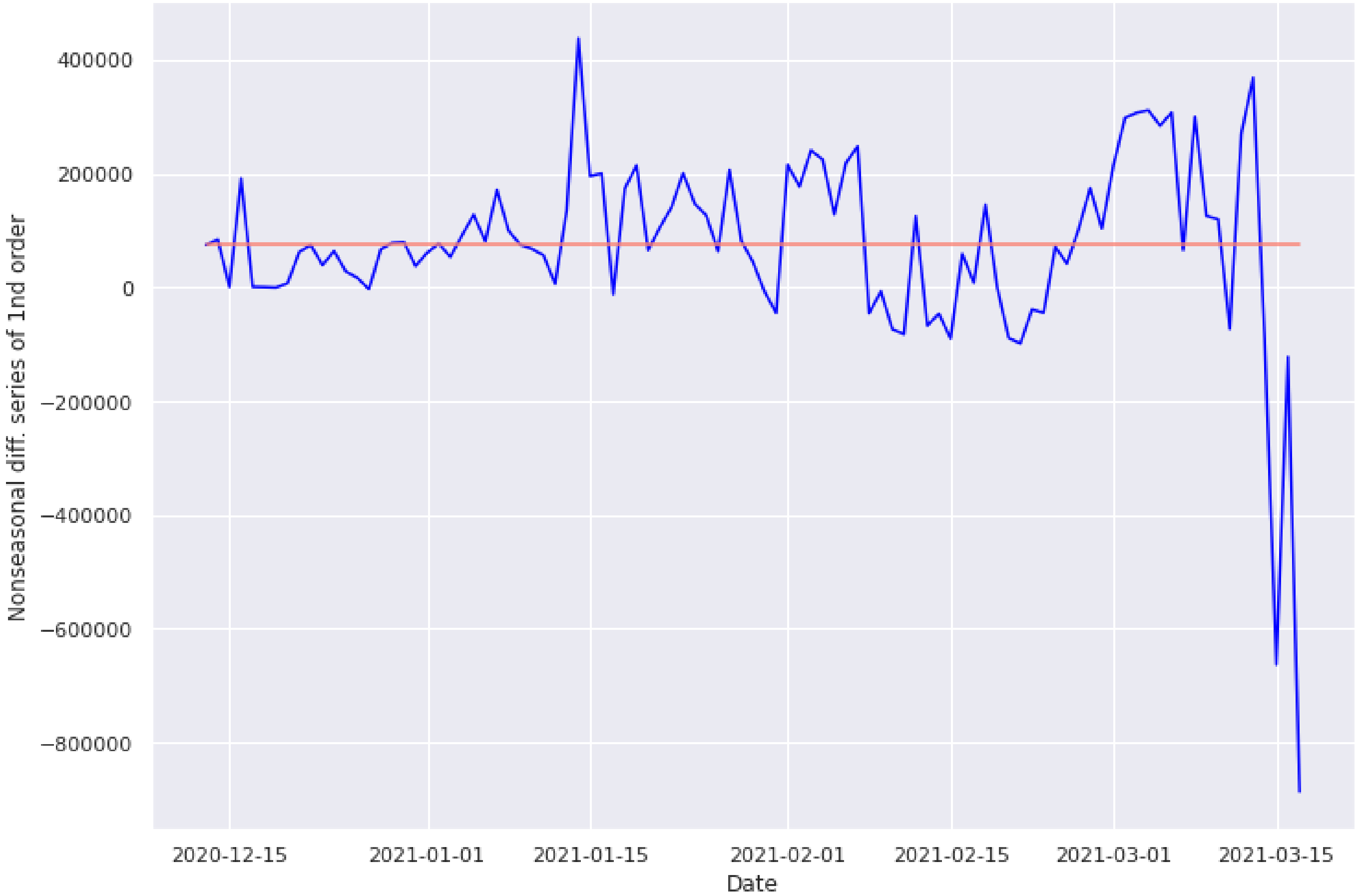
```
(-1.6928692312921958,
 0.434790157837617 93,
 2,
```

```
 2,
 92,
 {'1%': -3.503514579651927,
  '5%': -2.893507960466837,
  '10%': -2.583823615311909},
 2197.3988449405333)
```

$p - value$,=> we proceed to do different transformations of our series to make it stationary, which allowsus to
$= 0.3964$
> 0.05
buil an ARIMA model.

In [51]:

```
daily_data['diff1'] = daily_data['daily_vaccinations'].diff()
daily_data['diff1'] = daily_data['diff1'].fillna(daily_data['diff1'].mean())
ax = sns.lineplot(x = daily_data['date'], y = daily_data['diff1'], color='blue')
sns.lineplot(x = daily_data['date'], y = daily_data['diff1'].mean(), color='salmon')
plt.xlabel('Date')
plt.ylabel('Nonseasonal diff. series of 1nd order')
plt.show()
adfuller(daily_data.diff1)
```



Out[51]:

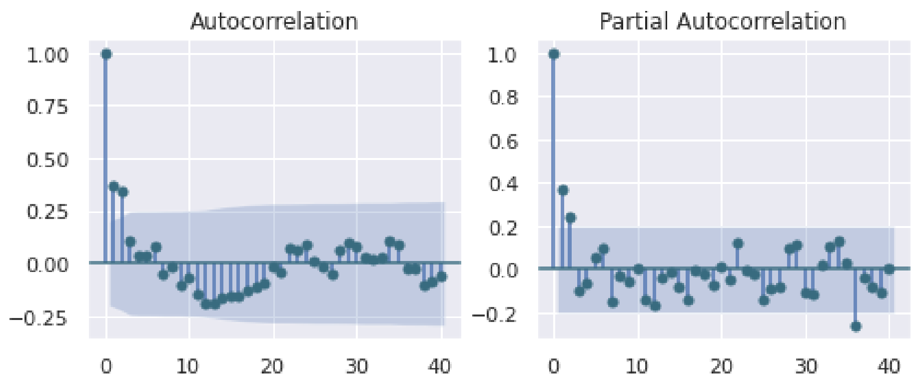
```
(-1.9334700728627532,
 0.3164831922715083,
 1,
 93,
 {'1%': -3.502704609582561,
  '5%': -2.8931578098779522,
  '10%': -2.583636712914788},
 2198.8290151881347)
```

The p-value returned from ADF Test is much lower than 0.05, however, we can still suspect some increasing trend and variance fluctuations, so we proceed to nonseasonal differencing of 2nd order.

In [52]:

```
plt.figure(figsize = (8,3))
```

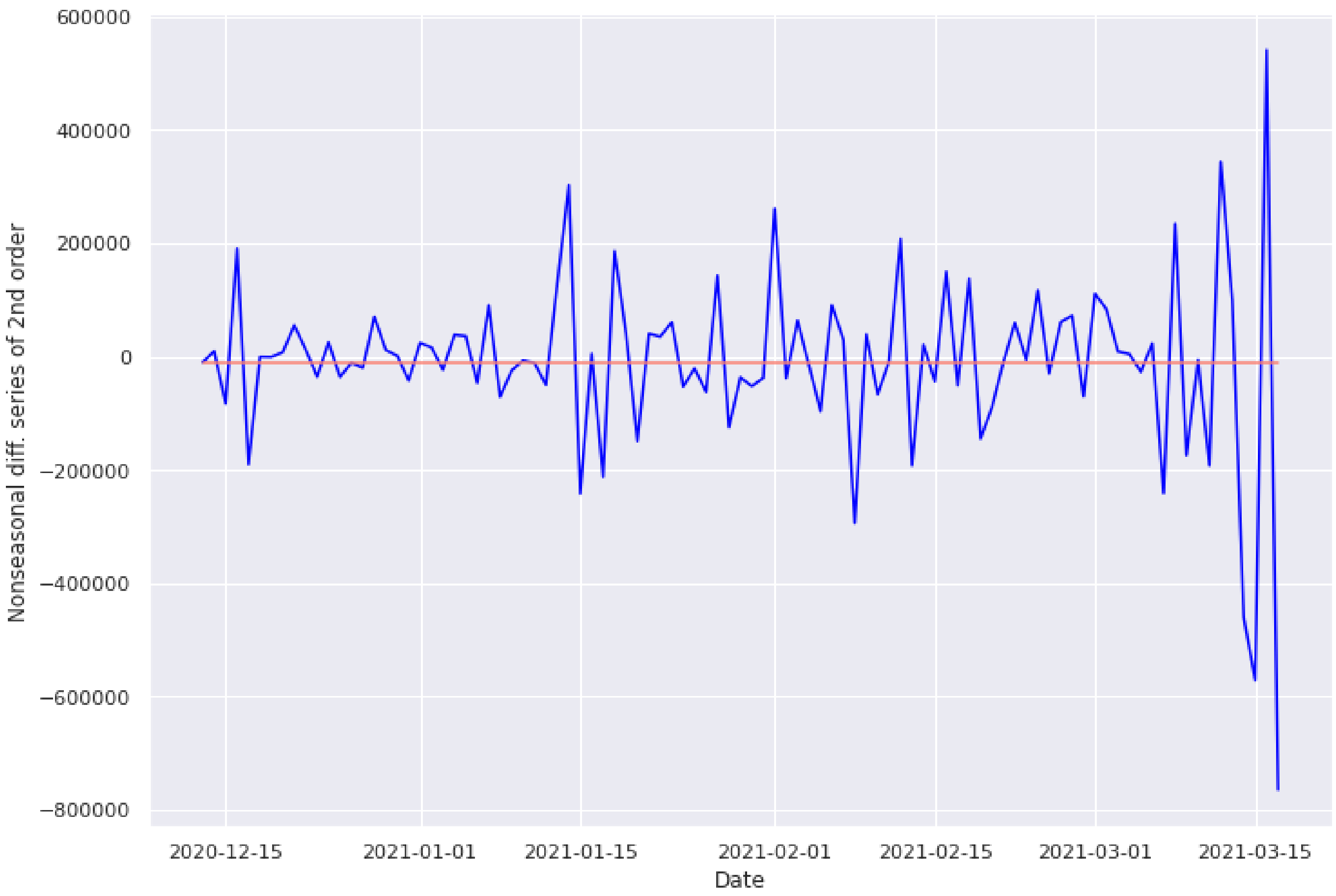
```
plt.subplot(121); plot_acf(daily_data['diff1'], lags = 40, ax = plt.gca(), color = c)
plt.subplot(122); plot_pacf(daily_data['diff1'], lags = 40,ax = plt.gca(), color = c)
plt.show()
```



After the nonseasonal differencing of order 1, the ACF still seems to decrease slowly to zero, however, the differencing helped to fix it to a certain extent. Let's try to make the 2nd order differencing to see if that would give better results.

In [53]:

```
daily_data['diff2'] = daily_data['diff1'].diff()
daily_data['diff2'] = daily_data['diff2'].fillna(daily_data['diff2'].mean())
ax = sns.lineplot(x = daily_data['date'], y = daily_data['diff2'], color='blue')
sns.lineplot(x = daily_data['date'], y = daily_data['diff2'].mean(), color='salmon')
plt.xlabel('Date')
plt.ylabel('Nonseasonal diff. series of 2nd order')
plt.show()
adfuller(daily_data.diff2)
```



Out[53]:

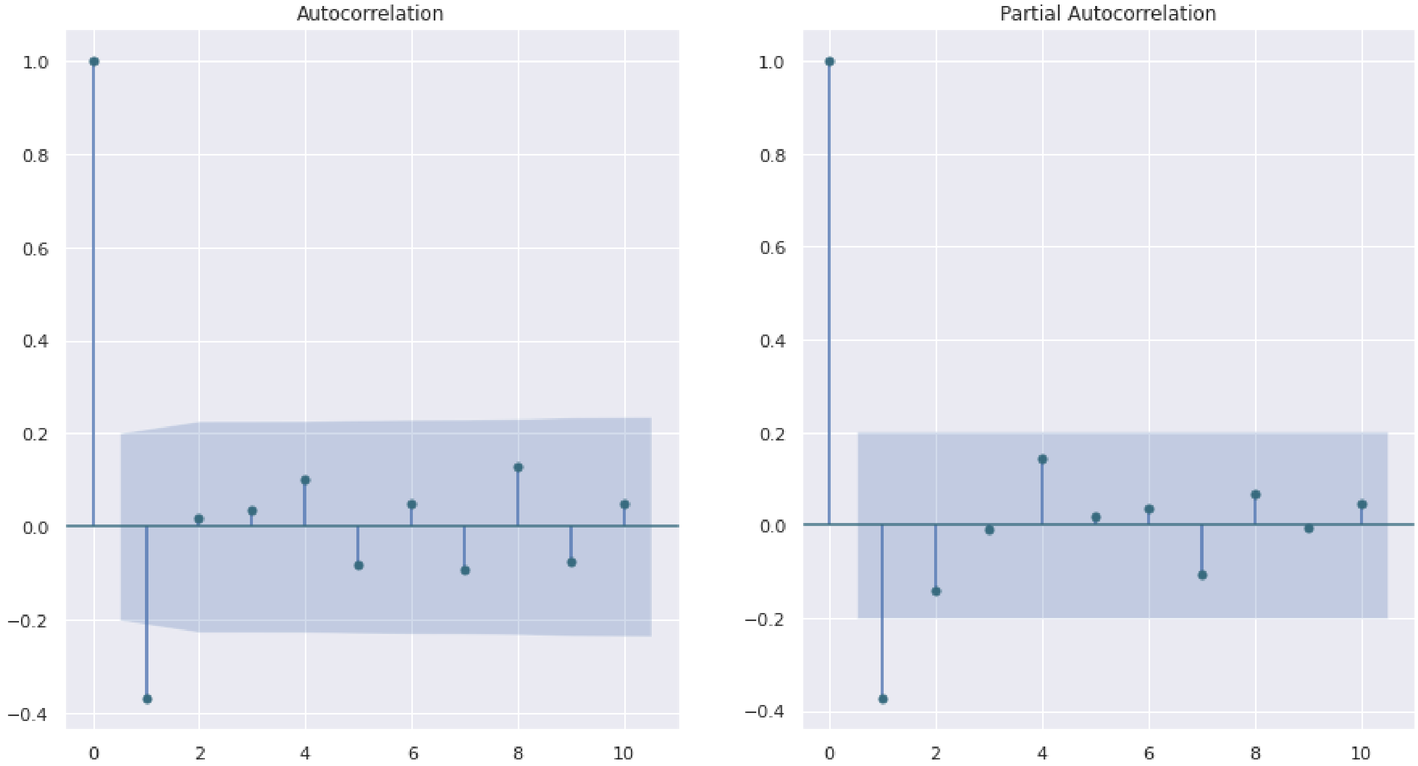
```
(-13.707634187084397,
 1.2550254333148225e-25,
 0,
 94,
 {'1%': -3.5019123847798657,
  '5%': -2.892815255482889,
```

```
 '10%': -2.583453861475781},
 2199.9361436646905)
```

This is definitelly the result we were waiting for. No evident trend detected on the plot, the mean, represented as a red line, is constant. Applying ADF test on *diff2* we get as a response a very small p-value, which indicates us that the series is stationary.

In [54]:

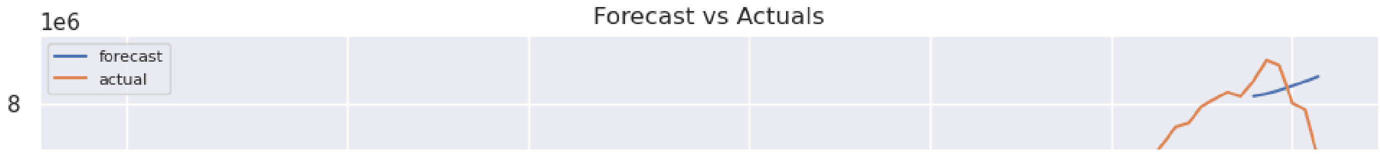
```
plt.figure(figsize = (15,8))
plt.subplot(121); plot_acf(daily_data['diff2'], lags = 10, ax = plt.gca(), color = c)
plt.subplot(122); plot_pacf(daily_data['diff2'], lags = 10,ax = plt.gca(), color = c)
plt.show()
```

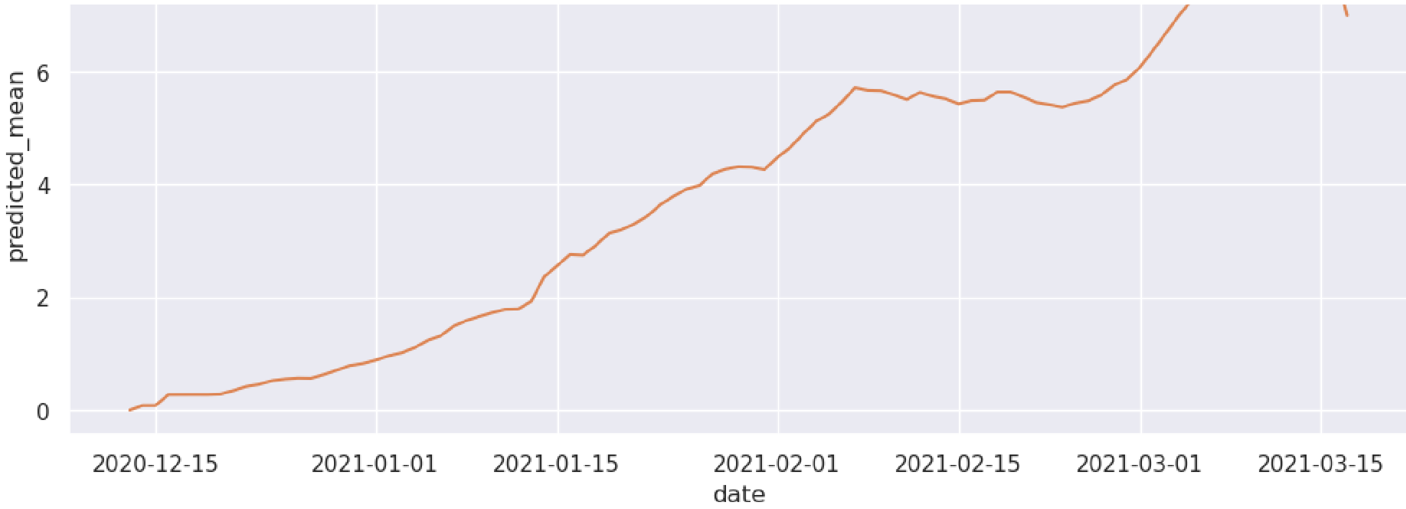


In [55]:

```
train = daily_data['daily_vaccinations'][:89]
test = daily_data['daily_vaccinations'][89:]
model = ARIMA(daily_data['daily_vaccinations'][:89], order = (1,2,1))
model = model.fit()
start = len(train)
end = len(train)+len(test)-1
pred = model.predict(start=start, end=end, typ = 'levels')
print(pred)
plt.figure(figsize=(12,5), dpi=100)
ax = sns.lineplot(x=daily_data['date'], y = pred, label='forecast')
sns.lineplot(x=daily_data['date'], y = daily_data['daily_vaccinations'], label='actual',
ax=ax)
plt.title('Forecast vs Actuals')
plt.legend(loc='upper left', fontsize=8)
plt.show()
```

```
89    8.129884e+06
90    8.172631e+06
91    8.236620e+06
92    8.311858e+06
93    8.393054e+06
94    8.477403e+06
Name: predicted_mean, dtype: float64
```





Training the ARIMA model of order $(1, 2, 1)$ on 89 observations of the daily_data DataFrame we get the predicted values marked as blue on the graph. Comparing these to the actual values, we see them be slightly different from each other. Later we will measure the goodness of fit of this model using the MAPE criterion and a simple comparison of the RMSE value to the test average.

In [56]:

```
mean_squared_error(pred,test, squared = False)
```

Out[56]:

722381.3704665307

In [57]:

```
test.mean()
```

Out[57]:

8124059.333333333

The test mean is around 8000000, while RMSE is much lower - 720000, this is not a really low value,but neither a signal of a big difference between the predicted and actual values.

In [58]:

```
mean_absolute_percentage_error(test,pred)
```

Out[58]:

0.07695415944616628

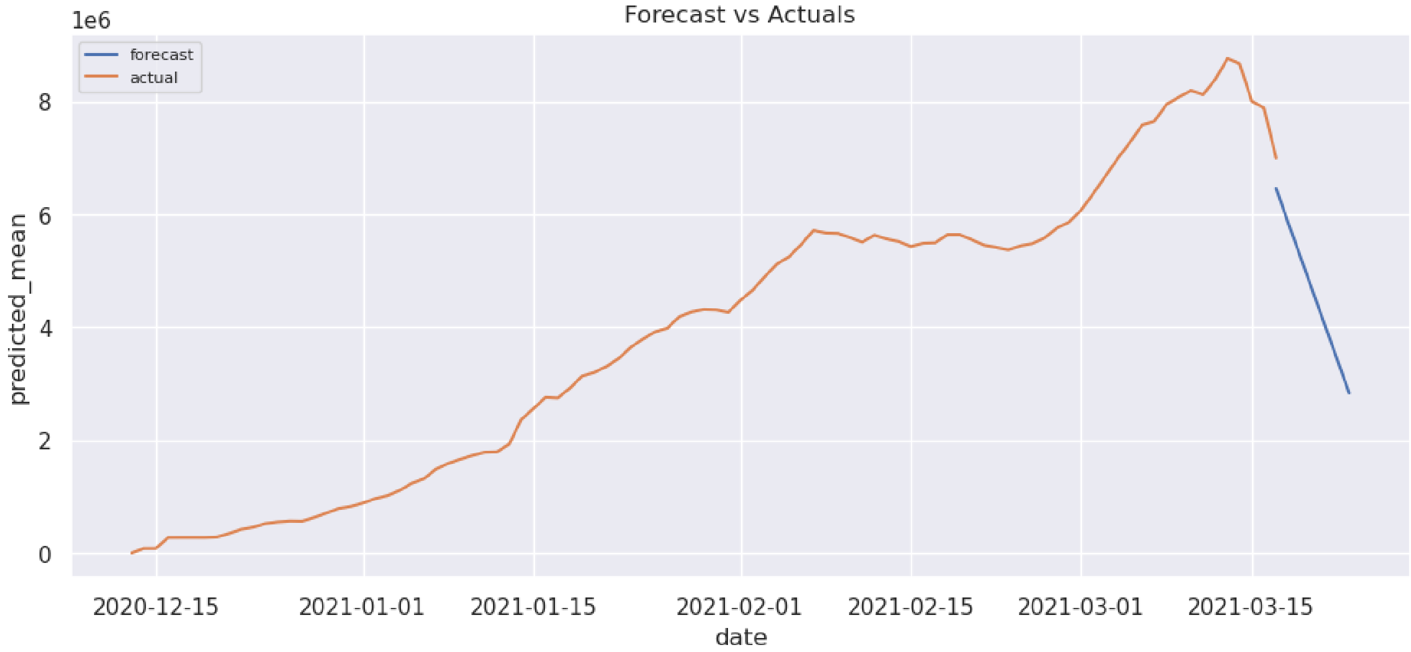
MAPE values is 0.07695. Notice that MAPE output is a non-negative float (where the best value is 0.00). MAPE represents statistical measure of how accurate a forecast system is. It measures this accuracy as a percentage. In our case the precentage of error is about 7.7%.

Let's now train our model on full data to make predictions in some future instances of time.

In [59]:

```
index_fut_dat = pd.date_range(start='2021-03-17', end='2021-03-23')
model_full = ARIMA(daily_data['daily_vaccinations'], order = (1,2,1))
model_full = model_full.fit()
pred2 = model_full.predict(start=len(daily_data), end=len(daily_data)+6, typ = 'levels')
pred2.index = index_fut_dat
plt.figure(figsize=(12,5), dpi=100)
ax = sns.lineplot(x=pred2.index, y = pred2, label='forecast')
sns.lineplot(x=daily_data['date'], y = daily_data['daily_vaccinations'], label='actual',
ax=ax)
plt.title('Forecast vs Actuals')
plt.legend(loc='upper left', fontsize=8)
```

```
plt.show()
```



In [60]:

```
print(pred2)
```

```
2021-03-17    6.465725e+06
2021-03-18    5.848048e+06
2021-03-19    5.248626e+06
2021-03-20    4.645025e+06
2021-03-21    4.042381e+06
2021-03-22    3.439518e+06
2021-03-23    2.836705e+06
Freq: D, Name: predicted_mean, dtype: float64
```

Inference

Assumptions based on previous analysis

- **We expect the mean value of daily vaccinations in February to be higher than the mean corresponding to January month, due to the vaccination progress.**

Hypothesis Testing

To check if the daily vaccinations mean in February is greater than in January, we will apply the Hypothesis testing technique. Confidence level was set to $\alpha = 0.05$ for further hypothesis and confidence intervals. This are both null and alternative hypothesis:

$$H_0 = \mu_1 - \mu_0 = 0$$

$$H_1 = \mu_1 - \mu_0 \neq 0$$

Where: μ_0 - January daily vaccinations average μ_1 - February daily vaccinations average

In [61]:

```
vaccinations['month'] = vaccinations['date'].dt.month
```

```
jan_month = vaccinations[vaccinations['month'] == 1]['daily_vaccinations']
feb_month = vaccinations[vaccinations['month'] == 2]['daily_vaccinations']
scipy.stats.ttest_ind(jan_month, feb_month)
```

Out[61]:

```
Ttest_indResult(statistic=-2.382783721402224, pvalue=0.01722614232724415)
```

Analysing the *p-value* of two-sample t-test, we can claim it's much lower (0.0172) than the previously set < 0.05 confidence level ($\alpha = 0,05$), thus, we have enough evidence to reject the null-hypothesis, and claim that the analysed means are not equal.

Confidence Interval

To make a confidence interval of 95% of difference in means (*Feb - Jan*). If we see the CI to take negative values, it means the average daily vaccination number in *January* was higher, than in *February*. Following the previous assumption, we expect the CI to take positive value => the bigger average daily vaccination number in *February, 2021*.

In [62]:

```
alpha = 0.05
n1, n2 = len(feb_month), len(jan_month)
s1, s2 = np.var(feb_month, ddof = 1), np.var(jan_month, ddof = 1)
s = np.sqrt(((n1 - 1) * s1 + (n2 - 1) * s2) / (n1 + n2 - 2))
df = n1 + n2 - 2
t = scipy.stats.t.ppf(1- alpha / 2, df)

lower = (np.mean(feb_month) - np.mean(jan_month)) - t * np.sqrt(1 / n1 + 1 / n2) * s
upper = (np.mean(feb_month) - np.mean(jan_month)) + t * np.sqrt(1 / n1 + 1 / n2) * s
lower, upper
```

Out[62]:

```
(2407.8428221785252, 24766.772571021083)
```

As expected, the 95% CI of mean difference (*Feb - Jan*) resulted positive and without containting a zero value in it. That means the average daily vaccination number in February has growm, when comparing to January. The difference between the means is enormous, in 95% of the cases it varies between 2407 and 24766 daily vaccinations. With a decent amount of confidence we can claim that the vaccination has become more effective in terms of ddaily vaccinations number all around the world.

Conclusions

Hypothesis testing gave us a sign about the incrementation of the total vaccinations all around the world in February, comparing to the January data. However, the predictions made by ARIMA (1,2,1) model indicate a decrease in the number of vaccinations. We don´t expect this value, to fall to 2.000.000 daily observations as predicted by the ARIMA forecast, but even the actual values show a decrease in the last few days. This could have happened due to manufacturing or any others reasons. The actual information is insuffecient to explain such changes in the daily vaccination process.