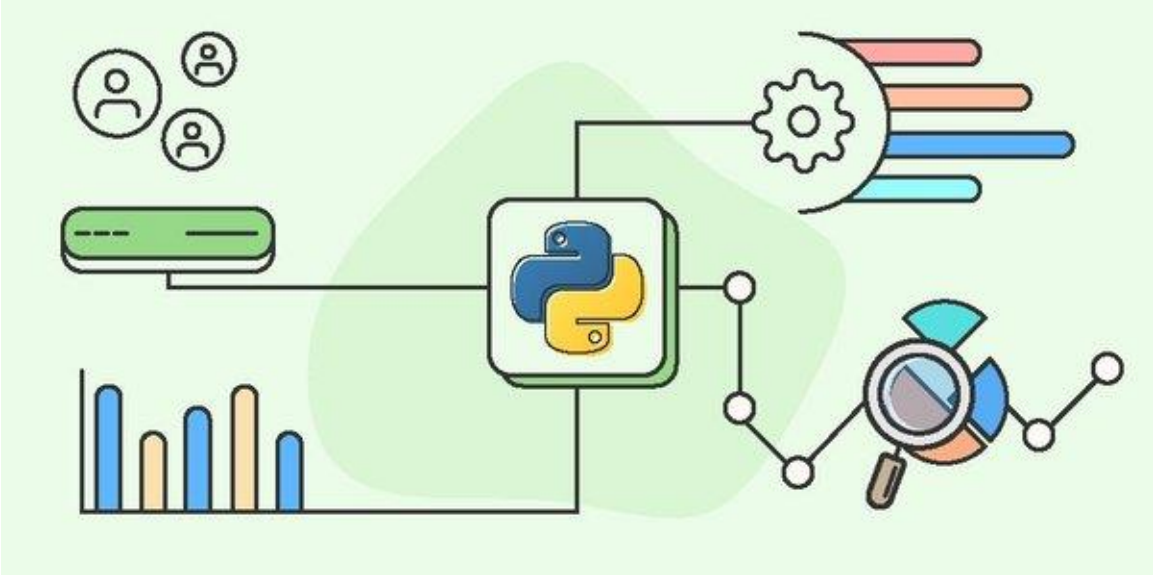
Covid-19 Vaccination Data Analysis & Visualization

PFDS PROJECT



J Vishwanath 20csu296
Mridul Gupta 20csu292
The Northcap University
Nov. 26, 2021

Contents

Abstract 3

Data Sciences 4

Software Used 5

- The Jupyter Notebook IDE 5
- ExcelSheet 5

Python Modules Imported 6

- Numpy 6
- pandas 6
- Matplotlib 7
- Sraborn 7
- Plotly 7

Introduction 8

Problem Statement 9

Approaching the Problem 10

Graphs used 11

Staring with Data Preparation and Cleaning 12

Data analysis and visualization 15

- India 15
- China 18
- USA 19
- Russia 22

Takeaways from the Dataset 23

- Which country devloped the vaccine the fastest? 23
- Which country has the highest vaccinated citizens? 24
- Different kinds of Vaccines given to people around the world 25
- Which country is using what kind of vaccine 26
- Percentage of total vaccinations in each country 26

Inferences and Conclusion 28

References 29

Abstract

As each and every sector across the world is re-emerging, new data is building up day by day with , we need to keep the record of the new data which can be helpful for the analytics and evaluation. Now we don't have data in gigabyte or terabyte but in zetta byte and petabyte and this data can not be handled with the day to day software such as Excel or Matlab. Therefore in this report we will be dealing with large data sets with the high-level programming language 'Python'.

The main goal of this project is to aggregate and analyze the data collected from the different data sources available on Vaccines. This project mainly focuses on the usage of the python programming language, more specifically pandas. This module has not only it's application in the field of just analyzing the data but also for the prediction of the upcoming scenarios.

The purpose of using this specific language is due to its versatility, vast libraries (Pandas, Numpy, Matplotlib, etc.), speed limitations, and ease of learning. We will be analyzing large datasets in this project which cannot be easily analyzed in other tools as compared to python. Python does not have it's limitation to only data analytics but also in many other fields such as Artificial intelligence, Machine learning, and many more.

Data Sciences

Data science is the field of data analytics and data visualization in which raw data or the unstructured data is cleaned and made ready for the analysis purpose. Data scientists use this data to get the required information for the future purpose.[1] ”Data science uses many processes and methods on the big data, the data may be structured or unstructured”. Data frames avail-able on the internet is the raw data we get. It may be either in unstructured or semi structured format. This data is further filtered, cleaned and then number of required task are performed for the analysis with the use of the high programming language. This data is further analyzed and then presented for our better understanding and evaluation.

Software Used

THE JUPYTER NOTEBOOK IDE

Formerly known as the IPython Notebook The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text.

Uses include

- data cleaning and transformation
- numerical simulation
- statistical modeling
- data visualization
- machine learning

EXCELSHEET

Microsoft Excel is a spreadsheet developed by Microsoft for Windows, macOS, Android and iOS. It features calculation, graphing tools, pivot tables, and a macro programming language called Visual Basic for Applications (VBA). It has been a very widely applied spreadsheet for these platforms, especially since version 5 in 1993, and it has replaced Lotus 1-2-3 as the industry standard for spreadsheets. Excel forms part of the Microsoft Office suite of software.

Python Modules Imported

A Python module is a file containing Python definitions and statements. A module can define functions, classes, and variables. A module can also include runnable code. Grouping related code into a module makes the code easier to understand and use. It also makes the code logically organized.

NUMPY

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed.

PANDAS

Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy. As one of the most popular data wrangling packages, Pandas works well with many other data science modules inside the Python ecosystem, and is typically included in every Python distribution

MATPLOTLIB

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, etc.

SRABORN

Seaborn is a library in Python predominantly used for making statistical graphics. Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data.

PLOTLY

The plotly Python library is an interactive, open-source plotting library that supports over 40 unique chart types covering a wide range of statistical, financial, geographic, scientific, and 3-dimensional use-cases. Built on top of the Plotly JavaScript library (plotly).

Introduction

As the world shifts its focus from Covid-19 restrictions to Vaccination, each country is running on its on pace in terms of vaccination of all its people. With every opening of a vaccine box to injection of vaccine into the bodies, everything is being recorded. This means that there is data being updated and entered every time.

This report takes you through a deep dive analysis of every country's covid vaccinations and the type of vaccination used in the month of JAN-FEB 2021 and OCT-NOV 2021. By comparing and contrasting both these time periods, we will take a look at the progress each country made w.r.t to itself and others.

Problem Statement

What we failed to understand is that even after being a developing nation, India tried to keep their citizen immune against Covid-19.

Through the data sheet we got on the Covid-19 vaccinations, we tried to understand the trend of how vaccinations are being done throughout the world and compare India with other developed countries.

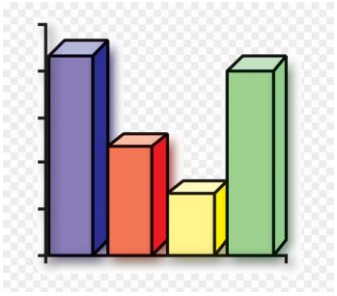
Approaching the Problem

We started with educating with the data sheet we found on Kaggle. Kaggle is an online community of data scientists and machine learning practitioners. Then we starting to list out the useful data we can compare. We then started to list out the best suitable graphs which was one of the crucial steps in order to represent the data in more understandable manner to the user.

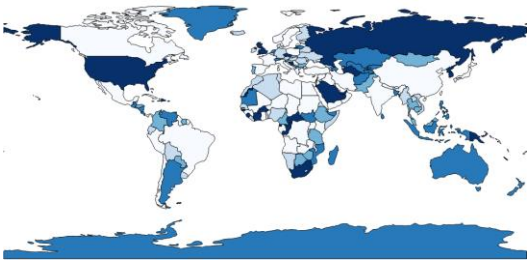
Graphs used

Stem Plot and **Line Plot** to show the number of daily vaccinations done in a country on a particular date.

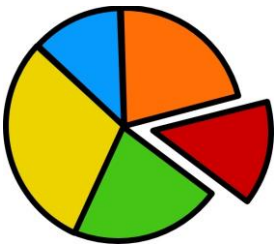
Bar Graph to show what vaccine is being given to people and the total number of vaccinations done till date. Bar Graph can be used from matplotlib and seaborn, we have used both depending on the complexity of data.



Choropleth Maps to show what all vaccines were given to people in different part of the world.



Pie Chart to show the sum of total vaccination done by the country in the starting month of the vaccination process and the latest months.



Staring with Data Preparation and Cleaning

We started by importing packages – **numpy and pandas**

```
In [1]: import numpy as np
        import pandas as pd
```

Then we imported the csv file as data set in df as shown in the code below

```
In [2]: df = pd.read_csv(r"C:\Users\gupta\Music\COVID 19 DATA SET ANALYSIS\COVID-19-(13.12.2020-20.02.2021)\Data\count
```

We got the basic synopsis of the data which was in the csv file and description of it

```
In [6]: df.head()
```

Out[6]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccina
0	Albania	ALB	2021-01-10	0.0	0.0	NaN	NaN	NaN	0.00
1	Albania	ALB	2021-01-11	NaN	NaN	NaN	NaN	64.0	NaN
2	Albania	ALB	2021-01-12	128.0	128.0	NaN	NaN	64.0	0.00
3	Albania	ALB	2021-01-13	188.0	188.0	NaN	60.0	63.0	0.01
4	Albania	ALB	2021-01-14	266.0	266.0	NaN	78.0	66.0	0.01

```
In [7]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3619 entries, 0 to 3618
Data columns (total 15 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   country                              3619 non-null   object
 1   iso_code                             3343 non-null   object
 2   date                                 3619 non-null   object
 3   total_vaccinations                   2393 non-null   float64
 4   people_vaccinated                    1981 non-null   float64
 5   people_fully_vaccinated              1324 non-null   float64
 6   daily_vaccinations_raw               2019 non-null   float64
 7   daily_vaccinations                   3483 non-null   float64
 8   total_vaccinations_per_hundred       2393 non-null   float64
 9   people_vaccinated_per_hundred        1981 non-null   float64
 10  people_fully_vaccinated_per_hundred  1324 non-null   float64
 11  daily_vaccinations_per_million       3483 non-null   float64
 12  vaccines                             3619 non-null   object
 13  source_name                          3619 non-null   object
 14  source_website                       3619 non-null   object
dtypes: float64(9), object(6)
memory usage: 424.2+ KB
```

```
In [8]: df.describe()
```

Out[8]:

	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_per_hundred	peo
count	2.393000e+03	1.981000e+03	1.324000e+03	2.019000e+03	3.483000e+03	2393.000000	198
mean	1.521028e+06	1.269657e+06	3.888861e+05	7.453608e+04	5.754796e+04	6.174785	5.24
std	5.038410e+06	4.148487e+06	1.476223e+06	2.065813e+05	1.784900e+05	11.530328	8.59
min	0.000000e+00	0.000000e+00	1.000000e+00	-5.001200e+04	1.000000e+00	0.000000	0.00
25%	2.989300e+04	2.702000e+04	8.366000e+03	2.021000e+03	1.207500e+03	0.590000	0.61
50%	1.917820e+05	1.694400e+05	3.395450e+04	1.164200e+04	6.081000e+03	2.420000	2.37
75%	7.689500e+05	6.324390e+05	1.947678e+05	5.658950e+04	2.922500e+04	5.570000	4.43
max	6.128950e+07	4.280960e+07	1.789567e+07	2.242472e+06	1.916190e+06	87.070000	49.7

```
In [9]: df.columns
```

```
Out[9]: Index(['country', 'iso_code', 'date', 'total_vaccinations',
               'people_vaccinated', 'people_fully_vaccinated',
               'daily_vaccinations_raw', 'daily_vaccinations',
               'total_vaccinations_per_hundred', 'people_vaccinated_per_hundred',
               'people_fully_vaccinated_per_hundred', 'daily_vaccinations_per_million',
               'vaccines', 'source_name', 'source_website'],
              dtype='object')
```

We then filled the null values with 0 and dropped the data where iso code were 0

```
In [10]: df.fillna(0, inplace = True)
         df['iso_code'].fillna('GBR', inplace=True)
         df.drop(df.index[df['iso_code'] == 0], inplace = True)
```

We then dropped the following data which were not required for us.

1. **source_name**
2. **source_website**
3. **people_fully_vaccinated**
4. **daily_vaccinations_raw**
5. **people_fully_vaccinated_per_hundred**
6. **daily_vaccinations_per_million**
7. **people_vaccinated_per_hundred**

```
In [11]: df.drop(["source_name","source_website","people_fully_vaccinated","daily_vaccinations_raw","people_fully_vacci
```

This marked the end of our data cleaning process, and we printed the data set just for making sure that the data was ready for visualization.

```
In [12]: df
```

Out[12]:

	country	iso_code	date	total_vaccinations	people_vaccinated	daily_vaccinations	total_vaccinations_per_hundred	vaccines
0	Albania	ALB	2021-01-10	0.0	0.0	0.0	0.00	Pfizer/BioNTech
1	Albania	ALB	2021-01-11	0.0	0.0	64.0	0.00	Pfizer/BioNTech
2	Albania	ALB	2021-01-12	128.0	128.0	64.0	0.00	Pfizer/BioNTech
3	Albania	ALB	2021-01-13	188.0	188.0	63.0	0.01	Pfizer/BioNTech
4	Albania	ALB	2021-01-14	266.0	266.0	66.0	0.01	Pfizer/BioNTech
...	...	...	...	...	...	...	...	...
3545	United States	USA	2021-02-16	55220364.0	39670551.0	1716311.0	16.51	Moderna, Pfizer/BioNTech
3546	United States	USA	2021-02-17	56281827.0	40268009.0	1644551.0	16.83	Moderna, Pfizer/BioNTech
3547	United States	USA	2021-02-18	57737767.0	41021049.0	1621071.0	17.26	Moderna, Pfizer/BioNTech
3548	United States	USA	2021-02-19	59585043.0	41977401.0	1596355.0	17.82	Moderna, Pfizer/BioNTech
3549	United States	USA	2021-02-20	61289500.0	42809595.0	1521088.0	18.33	Moderna, Pfizer/BioNTech

3343 rows × 8 columns

Data analysis and visualization

Now we start with the data visualization process

Here we will be visualizing vaccination progress of 4 countries namely India, China, USA, Russia

We start by importing seaborn, matplotlib, matplotlib.pyplot, plotly.express

In [5]:
```
import seaborn as sns
import matplotlib
import matplotlib.pyplot as plt
import plotly.express as px
%matplotlib inline

sns.set_style('darkgrid')
matplotlib.rcParams['font.size'] = 14
matplotlib.rcParams['figure.figsize'] = (9,6)
matplotlib.rcParams['figure.facecolor'] = '#00000000'
plt.rc('font', size=12)
```

INDIA

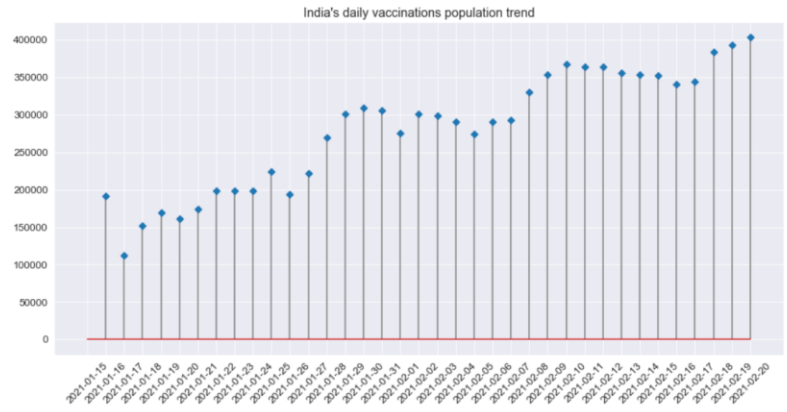
In [8]:
```
df_India = df[df["iso_code"] == 'IND'].copy()
df_India.drop(['people_vaccinated'], axis = 1, inplace = True)
df_India
```

Out[8]:

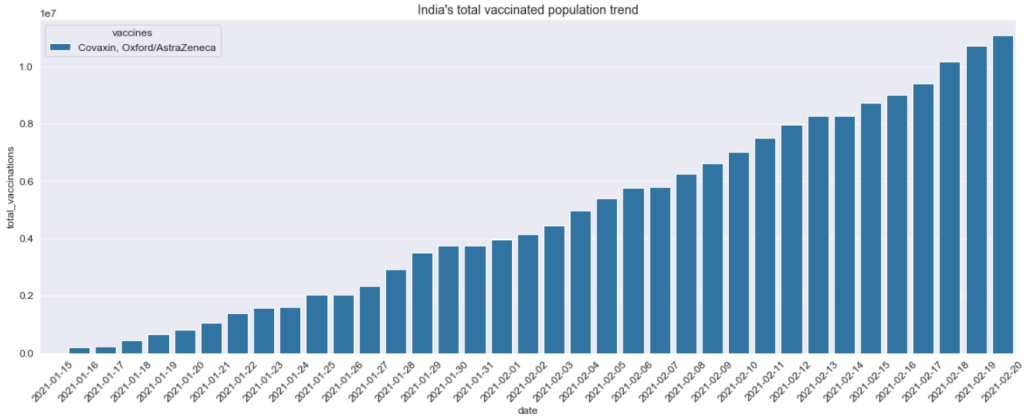
	country	iso_code	date	total_vaccinations	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_per_hundre
1554	India	IND	2021-01-15	0.0	NaN	NaN	NaN	0.00
1555	India	IND	2021-01-16	191181.0	NaN	191181.0	191181.0	0.01
1556	India	IND	2021-01-17	224301.0	NaN	33120.0	112150.0	0.02
1557	India	IND	2021-01-18	454049.0	NaN	229748.0	151350.0	0.03
1558	India	IND	2021-01-19	674835.0	NaN	220786.0	168709.0	0.05
1559	India	IND	2021-01-20	806484.0	NaN	131649.0	161297.0	0.06
1560	India	IND	2021-01-21	1043534.0	NaN	237050.0	173922.0	0.08
1561	India	IND	2021-01-22	1390592.0	NaN	347058.0	198656.0	0.10
1562	India	IND	2021-01-23	1582201.0	NaN	191609.0	198717.0	0.11
1563	India	IND	2021-01-24	1615504.0	NaN	33303.0	198743.0	0.12
1564	India	IND	2021-01-25	2023809.0	NaN	408305.0	224251.0	0.15
1565	India	IND	2021-01-26	2029480.0	NaN	5671.0	193521.0	0.15
1566	India	IND	2021-01-27	2355979.0	NaN	326499.0	221356.0	0.17
1567	India	IND	2021-01-28	2928053.0	NaN	572074.0	269217.0	0.21

Jan-Feb

In [21]:
```
plt.figure(figsize=(15,7))
x = df_India["date"]
y = df_India["daily_vaccinations"]

plt.stem(x,y,linefmt='grey', markerfmt='D')

plt.title("India's daily vaccinations population trend")
plt.xticks(rotation=45)
plt.show();
```

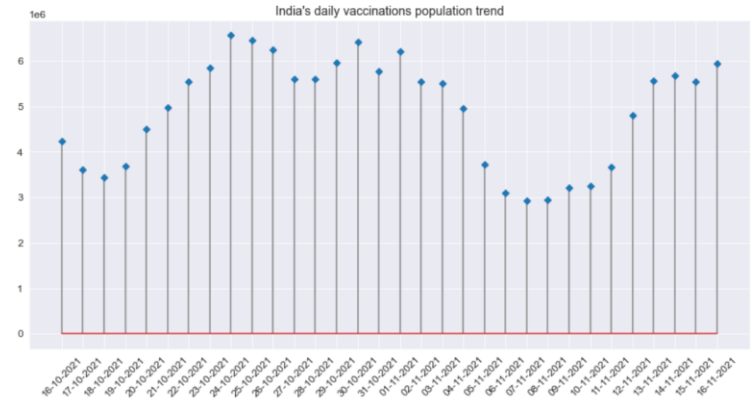


In [16]:
```
plt.figure(figsize=(20,7))
sns.barplot(data=df_India, y="total_vaccinations",x="date",hue = 'vaccines')

plt.title("India's total vaccinated population trend")
plt.xticks(rotation=45);
```

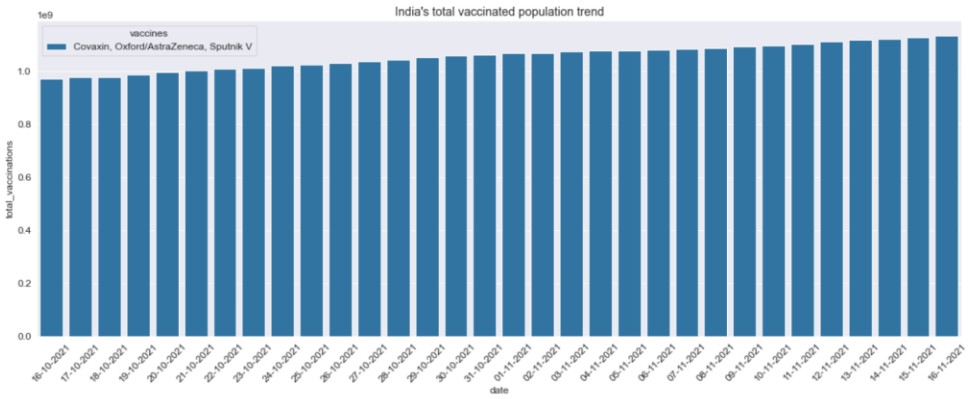


Oct-Nov

In [12]:
```
plt.figure(figsize=(15,7))
x = df_India["date"]
y = df_India["daily_vaccinations"]

plt.stem(x,y,linefmt='grey', markerfmt='D')

plt.title("India's daily vaccinations population trend")
plt.xticks(rotation=45)
plt.show();
```



In [26]:
```
plt.figure(figsize=(20,7))
sns.barplot(data=df_India, y="total_vaccinations",x="date",hue = 'vaccines')

plt.title("India's total vaccinated population trend")
plt.xticks(rotation=45);
```



CHINA

```
In [23]: df_China = df[df["iso_code"] == 'CHN'].copy()
```

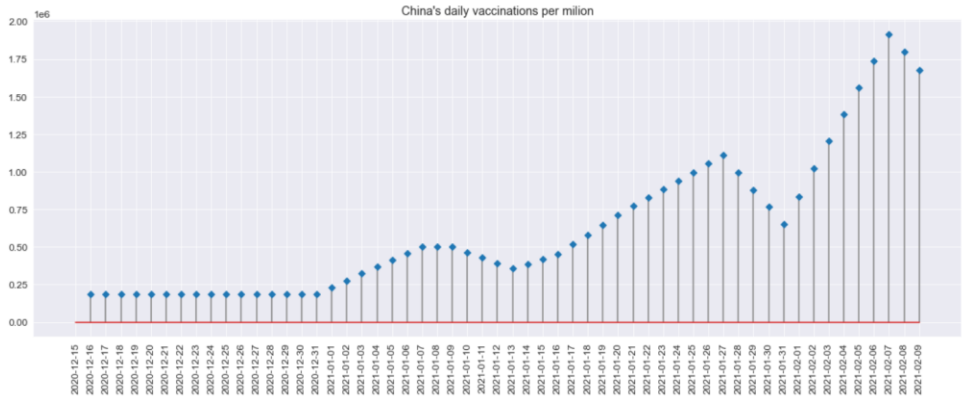
```
In [24]: df_China.drop(['people_vaccinated'], axis = 1, inplace = True)
         df_China
```

Out[24]:

	country	iso_code	date	total_vaccinations	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_per_hundred
660	China	CHN	2020-12-15	1500000.0	NaN	NaN	NaN	0.10
661	China	CHN	2020-12-16	NaN	NaN	NaN	187500.0	NaN
662	China	CHN	2020-12-17	NaN	NaN	NaN	187500.0	NaN
663	China	CHN	2020-12-18	NaN	NaN	NaN	187500.0	NaN
664	China	CHN	2020-12-19	NaN	NaN	NaN	187500.0	NaN
665	China	CHN	2020-12-20	NaN	NaN	NaN	187500.0	NaN
666	China	CHN	2020-12-21	NaN	NaN	NaN	187500.0	NaN

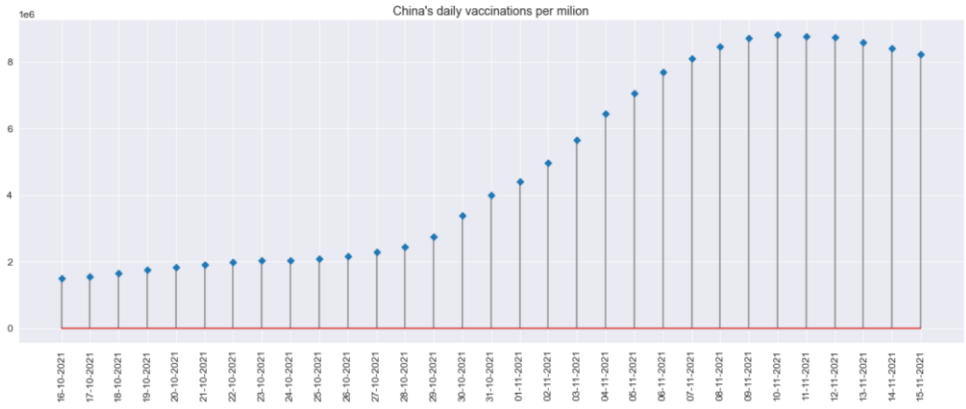
Jan-Feb

```
In [29]: plt.figure(figsize=(20,7))
         x = df_China["date"]
         y = df_China["daily_vaccinations"]

         plt.stem(x,y,linefmt='grey', markerfmt='D')
         plt.xticks(rotation=90);
         plt.title("China's daily vaccinations per milion");
```



Oct-Nov

In [16]:
```
plt.figure(figsize=(20,7))
x = df_China["date"]
y = df_China["daily_vaccinations"]

plt.stem(x,y,linefmt='grey', markerfmt='D')
plt.xticks(rotation=90);
plt.title("China's daily vaccinations per milion");
```



USA

In [18]:
```
df_USA = df[df["iso_code"] == 'USA'].copy()
df_USA
```

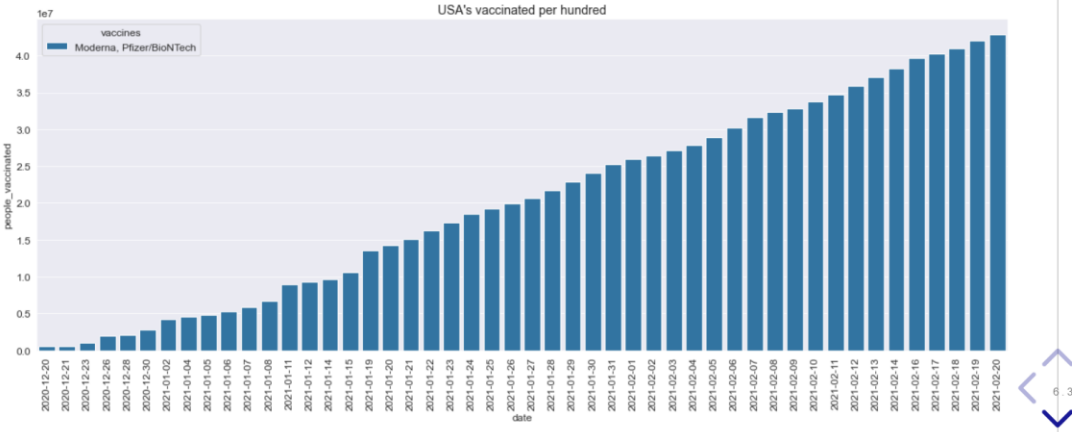
Out[18]:

	country	iso_code	date	total_vaccinations	people_vaccinated	daily_vaccinations	total_vaccinations_per_hundred	vaccines
3487	United States	USA	2020-12-20	556208.0	556208.0	0.0	0.17	Moderna, Pfizer/BioNTech
3488	United States	USA	2020-12-21	614117.0	614117.0	57909.0	0.18	Moderna, Pfizer/BioNTech
3489	United States	USA	2020-12-22	0.0	0.0	127432.0	0.00	Moderna, Pfizer/BioNTech
3490	United States	USA	2020-12-23	1008025.0	1008025.0	150606.0	0.30	Moderna, Pfizer/BioNTech
3491	United States	USA	2020-12-24	0.0	0.0	191001.0	0.00	Moderna, Pfizer/BioNTech
...	...	...	...	...	...	...	...	...
3545	United States	USA	2021-02-16	55220364.0	39670551.0	1716311.0	16.51	Moderna, Pfizer/BioNTech
3546	United States	USA	2021-02-17	56281827.0	40268009.0	1644551.0	16.83	Moderna, Pfizer/BioNTech
3547	United States	USA	2021-02-18	57737767.0	41021049.0	1621071.0	17.26	Moderna, Pfizer/BioNTech
3548	United States	USA	2021-02-19	59585043.0	41977401.0	1596355.0	17.82	Moderna, Pfizer/BioNTech
3549	United States	USA	2021-02-20	61289500.0	42809595.0	1521088.0	18.33	Moderna, Pfizer/BioNTech

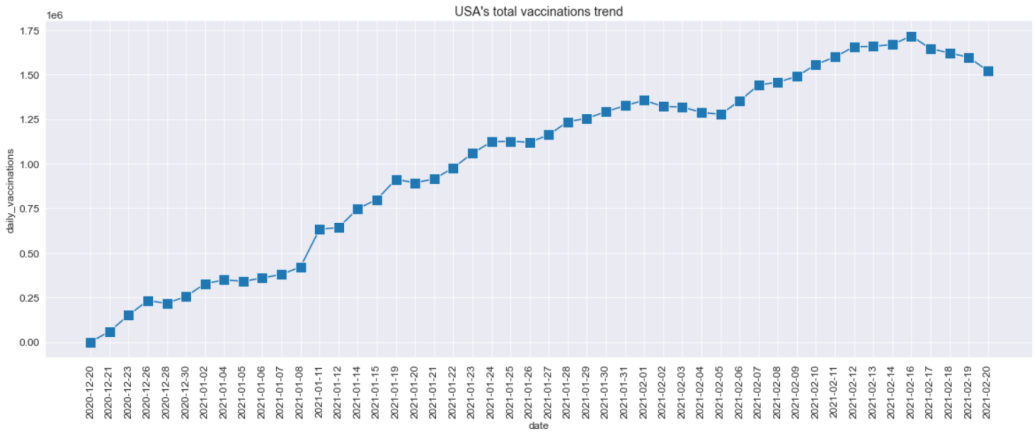
63 rows × 8 columns

Jan-Feb

In [19]:
```
plt.figure(figsize=(20,7))

df_USA.drop(df_USA.index[df_USA['people_vaccinated'] == 0], inplace = True)

sns.barplot(data=df_USA,x="date",y="people_vaccinated", hue = 'vaccines')
plt.title("USA's vaccinated per hundred")

plt.xticks(rotation=90);

plt.show();
```

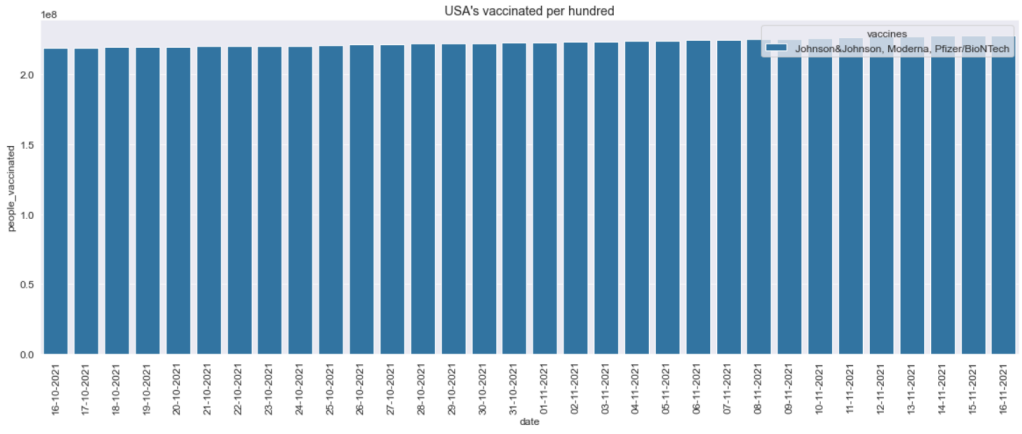


In [20]:
```
plt.figure(figsize=(20,7))
sns.lineplot(data=df_USA,x="date",y="daily_vaccinations",marker='s', markersize = 12);

plt.xticks(rotation=90);
plt.title("USA's total vaccinations trend");
```

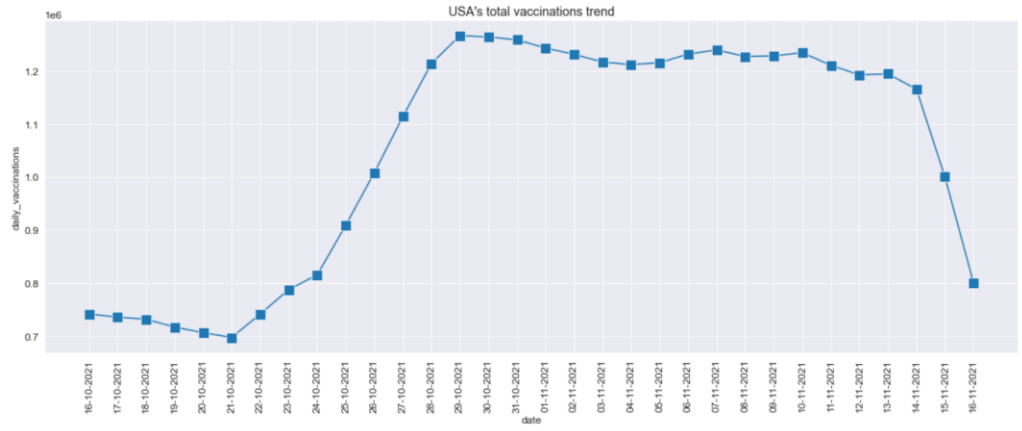


Oct-Nov

In [31]:
```
plt.figure(figsize=(20,7))

df_USA.drop(df_USA.index[df_USA['people_vaccinated'] == 0], inplace = True)

sns.barplot(data=df_USA,x="date",y="people_vaccinated", hue = 'vaccines')
plt.title("USA's vaccinated per hundred")

plt.xticks(rotation=90);

plt.show();
```



In [32]:
```
plt.figure(figsize=(20,7))
sns.lineplot(data=df_USA,x="date",y="daily_vaccinations",marker='s', markersize = 12);

plt.xticks(rotation=90);
plt.title("USA's total vaccinations trend");
```



RUSSIA

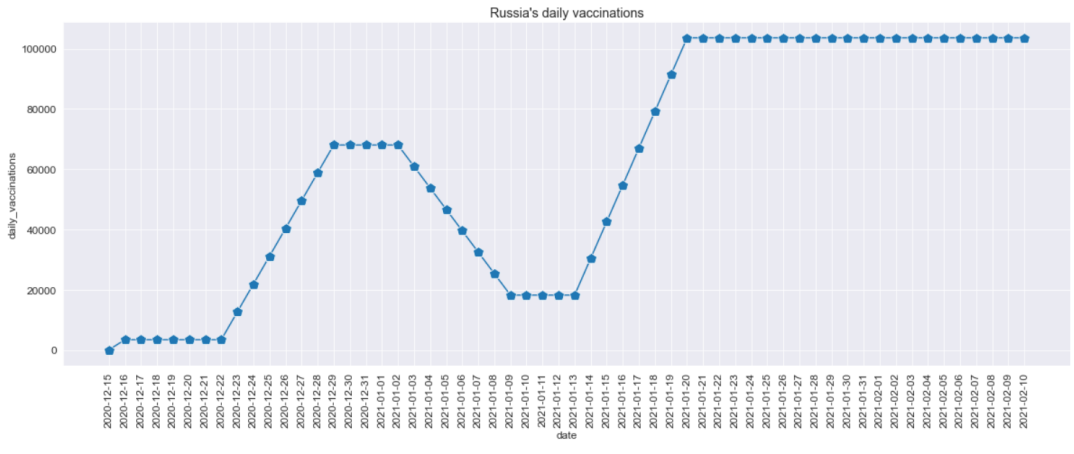
In [21]:
```
df_Russia = df[df["iso_code"] == 'RUS'].copy()
df_Russia
```

Out[21]:

	country	iso_code	date	total_vaccinations	people_vaccinated	daily_vaccinations	total_vaccinations_per_hundred	vaccines
2764	Russia	RUS	2020-12-15	28500.0	28500.0	0.0	0.02	Sputnik V
2765	Russia	RUS	2020-12-16	0.0	0.0	3357.0	0.00	Sputnik V
2766	Russia	RUS	2020-12-17	0.0	0.0	3357.0	0.00	Sputnik V
2767	Russia	RUS	2020-12-18	0.0	0.0	3357.0	0.00	Sputnik V
2768	Russia	RUS	2020-12-19	0.0	0.0	3357.0	0.00	Sputnik V
2769	Russia	RUS	2020-12-20	0.0	0.0	3357.0	0.00	Sputnik V
2770	Russia	RUS	2020-12-21	0.0	0.0	3357.0	0.00	Sputnik V
2771	Russia	RUS	2020-12-22	52000.0	52000.0	3357.0	0.04	Sputnik V
2772	Russia	RUS	2020-12-23	0.0	0.0	12592.0	0.00	Sputnik V
2773	Russia	RUS	2020-12-24	0.0	0.0	21827.0	0.00	Sputnik V
2774	Russia	RUS	2020-12-25	0.0	0.0	31061.0	0.00	Sputnik V
2775	Russia	RUS	2020-12-26	0.0	0.0	40296.0	0.00	Sputnik V
2776	Russia	RUS	2020-12-27	0.0	0.0	49531.0	0.00	Sputnik V
2777	Russia	RUS	2020-12-28	0.0	0.0	58765.0	0.00	Sputnik V
2778	Russia	RUS	2020-12-29	0.0	0.0	68000.0	0.00	Sputnik V
2779	Russia	RUS	2020-12-30	0.0	0.0	68000.0	0.00	Sputnik V
2780	Russia	RUS	2020-12-31	0.0	0.0	68000.0	0.00	Sputnik V
2781	Russia	RUS	2021-01-01	0.0	0.0	68000.0	0.00	Sputnik V
2782	Russia	RUS	2021-01-02	800000.0	800000.0	68000.0	0.55	Sputnik V
2783	Russia	RUS	2021-01-03	0.0	0.0	60883.0	0.00	Sputnik V
2784	Russia	RUS	2021-01-04	0.0	0.0	53766.0	0.00	Sputnik V
2785	Russia	RUS	2021-01-05	0.0	0.0	46649.0	0.00	Sputnik V
2786	Russia	RUS	2021-01-06	0.0	0.0	39532.0	0.00	Sputnik V
2787	Russia	RUS	2021-01-07	0.0	0.0	32416.0	0.00	Sputnik V
2788	Russia	RUS	2021-01-08	0.0	0.0	25299.0	0.00	Sputnik V

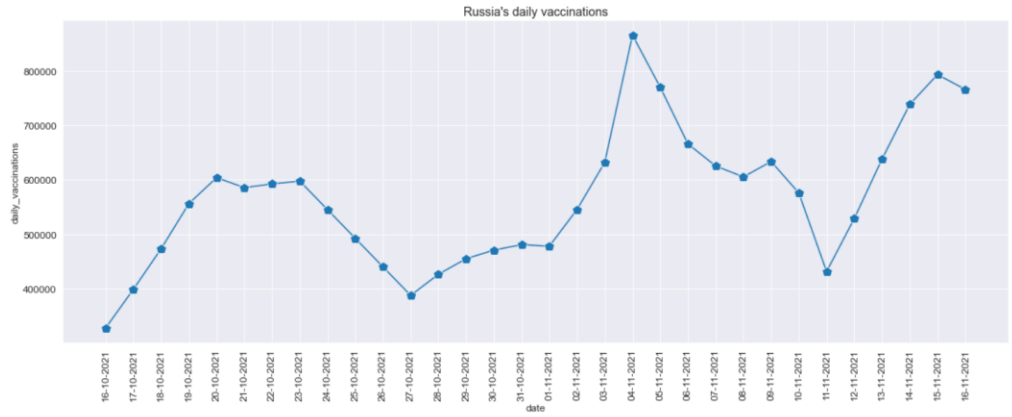
Jan-Feb

In [22]:
```
plt.figure(figsize=(20,7))
sns.lineplot(data=df_Russia,x="date",y="daily_vaccinations",marker='p', markersize = 12);

plt.xticks(rotation=90)
plt.title("Russia's daily vaccinations");
```



Oct-Nov

```
In [34]: plt.figure(figsize=(20,7))
         sns.lineplot(data=df_Russia,x="date",y="daily_vaccinations",marker='p', markersize = 12);

         plt.xticks(rotation=90)
         plt.title("Russia's daily vaccinations");
```



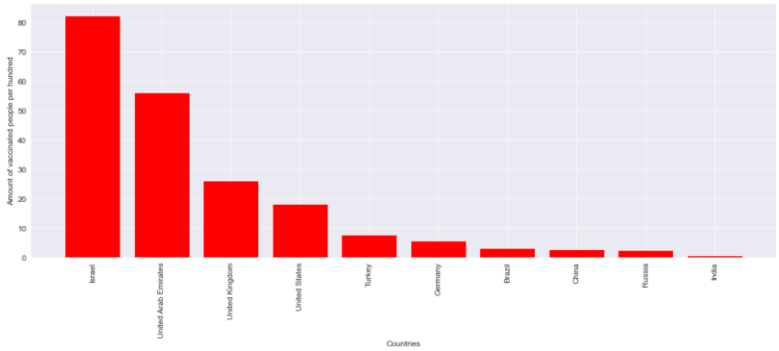
Takeaways from the Dataset

Which country devloped the vaccine the fastest?

```
In [23]: cols = ['country', 'total_vaccinations', 'iso_code', 'vaccines','total_vaccinations_per_hundred']
         vacc_amount = df[cols].groupby('country').max().sort_values('total_vaccinations', ascending=False).dropna(subs
         vacc_amount = vacc_amount.iloc[:10]
         vacc_amount = vacc_amount.sort_values('total_vaccinations_per_hundred', ascending=False)
```

Jan-Feb

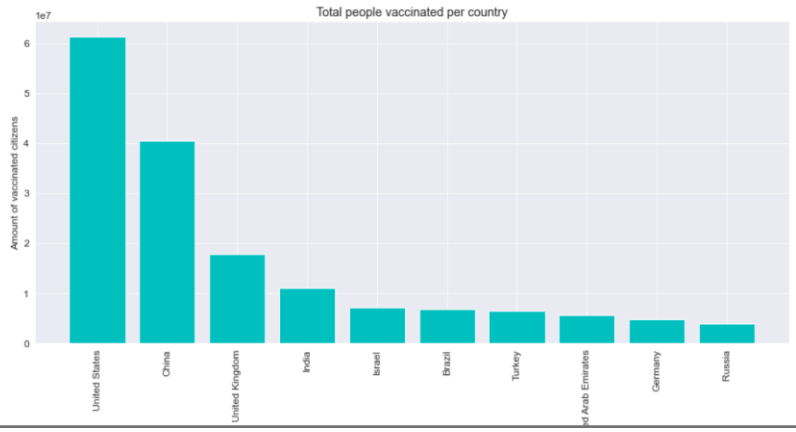
```
In [24]: plt.figure(figsize=(20, 7))
         plt.bar(vacc_amount.index, vacc_amount.total_vaccinations_per_hundred, color = 'r')

         plt.xticks(rotation = 90)
         plt.ylabel('Amount of vaccinated people per hundred')
         plt.xlabel('Countries')
         plt.show()
```



WHICH COUNTRY HAS THE HIGHEST VACCINATED CITIZENS?

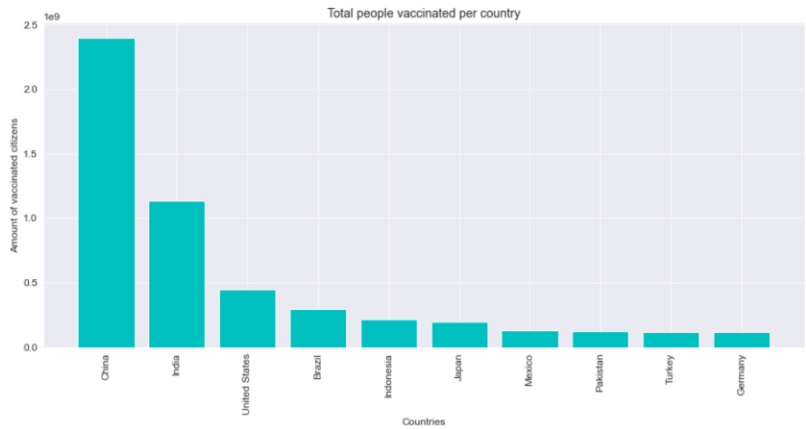
Jan-Feb

In [25]:
```
cols = ['country', 'total_vaccinations', 'iso_code', 'vaccines']
vacc_amount = df[cols].groupby('country').max().sort_values('total_vaccinations', ascending=False).dropna(subs
vacc_amount = vacc_amount.iloc[:10]
plt.figure(figsize=(16, 7))
plt.bar(vacc_amount.index, vacc_amount.total_vaccinations, color = 'c')
plt.title('Total people vaccinated per country')
plt.xticks(rotation = 90)
plt.ylabel('Amount of vaccinated citizens')
plt.xlabel('Countries')
plt.show();
```



Oct-Nov

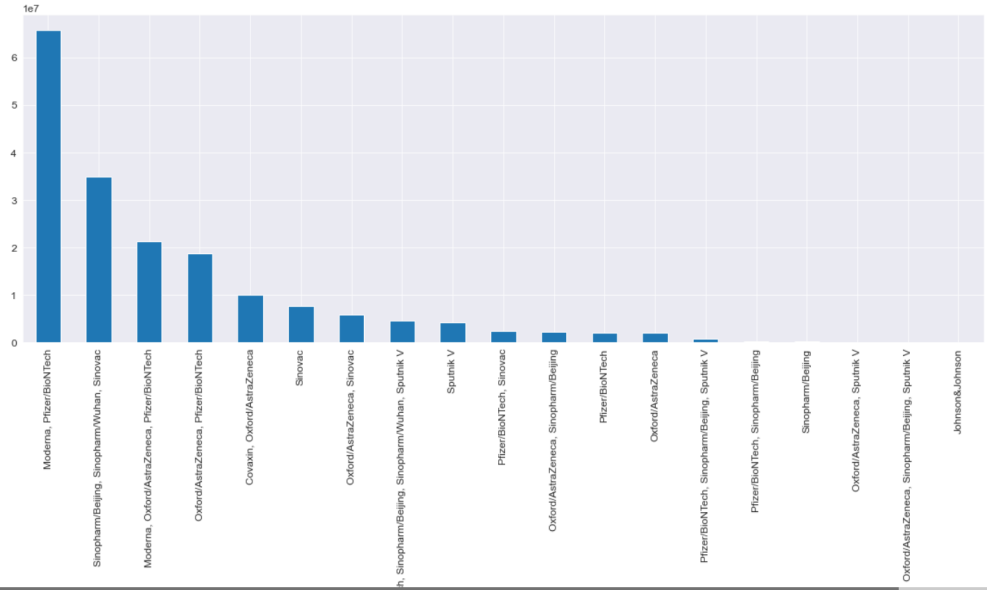
In [37]:
```
cols = ['country', 'total_vaccinations', 'iso_code', 'vaccines']
vacc_amount = df[cols].groupby('country').max().sort_values('total_vaccinations', ascending=False).dropna(subs
vacc_amount = vacc_amount.iloc[:10]
plt.figure(figsize=(16, 7))
plt.bar(vacc_amount.index, vacc_amount.total_vaccinations, color = 'c')
plt.title('Total people vaccinated per country')
plt.xticks(rotation = 90)
plt.ylabel('Amount of vaccinated citizens')
plt.xlabel('Countries')
plt.show();
```



DIFFERENT KINDS OF VACCINES GIVEN TO PEOPLE AROUND THE WORLD?

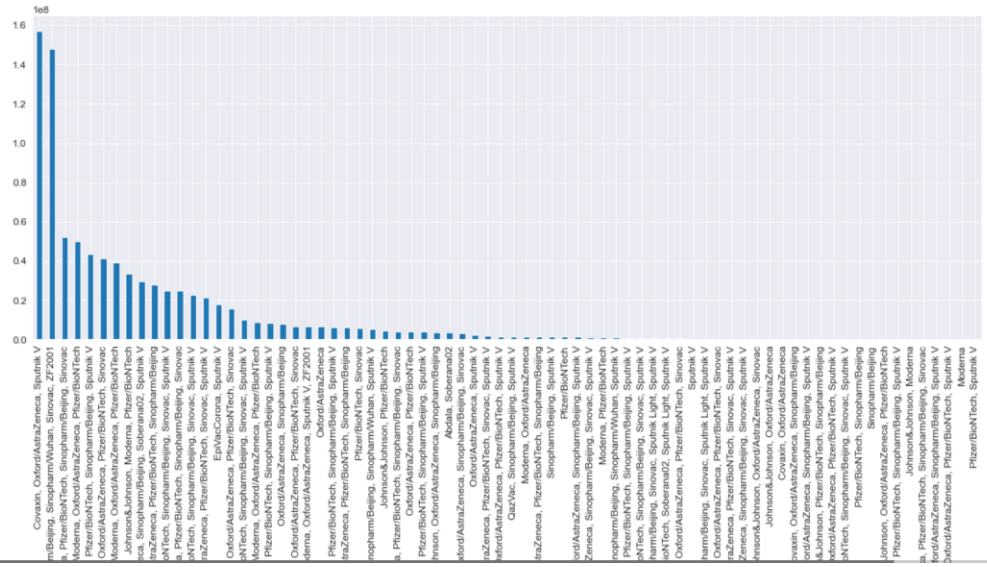
Jan-Feb

In [26]:
```
plt.figure(figsize=(20,7))
df_vaccines = df[["country","daily_vaccinations","vaccines"]].copy()
df_group = df_vaccines.groupby("vaccines")["daily_vaccinations"].sum()
df_group = df_group.sort_values(ascending=False).plot.bar()
```



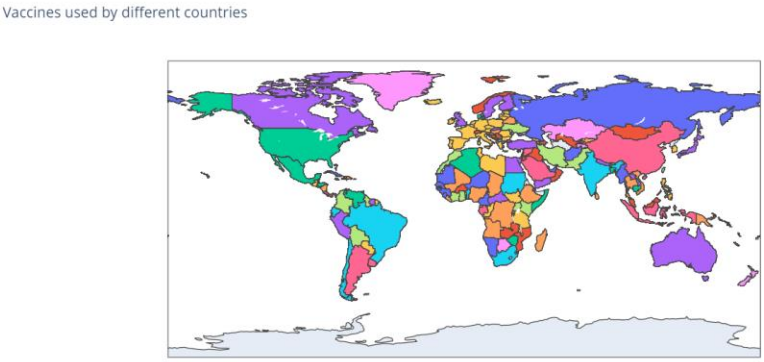
Oct-Nov

In [38]:
```
plt.figure(figsize=(20,7))
df_vaccines = df[["country","daily_vaccinations","vaccines"]].copy()
df_group = df_vaccines.groupby("vaccines")["daily_vaccinations"].sum()
df_group = df_group.sort_values(ascending=False).plot.bar()
```



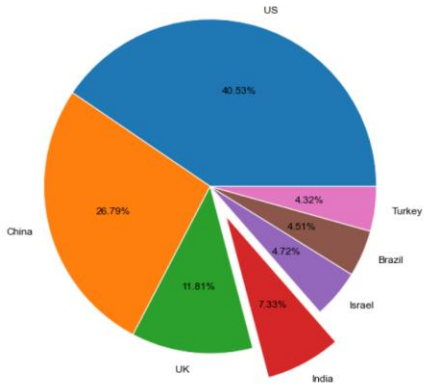
WHICH COUNTRY IS USING WHAT KIND OF VACCINE?

```
In [17]: fig = px.choropleth(df, locations="iso_code",
                            color="vaccines",
                            hover_name="country",
                            color_continuous_scale=px.colors.sequential.Plasma,
                           title= "Vaccines used by different countries")
         fig.update_layout(showlegend=False)
         fig.show()
```

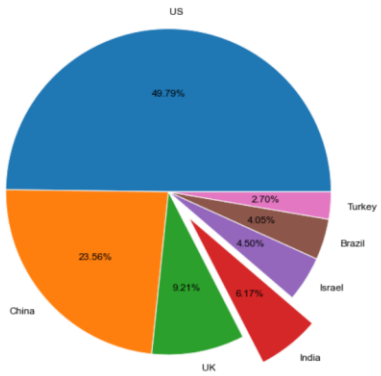


PERCENTAGE OF TOTAL VACCINATIONS IN EACH COUNTRY?

```
In [35]: vacc_amount1 = df[cols].groupby('country').max().sort_values('total_vaccinations', ascending=False).dropna(sub
         vacc_amount1.drop(columns= ["iso_code","vaccines"] , inplace= True)
         df_pie = vacc_amount1.head(7)
         myexplode = [0, 0, 0, 0.3,0,0,0]
         plt.pie(df_pie['total_vaccinations'], labels =  ["US", "China" , "UK" , "India" , "Israel" , "Brazil" , "Turke
         plt.show()
```



In [41]:
```
vacc_amount1 = df[cols].groupby('country').max().sort_values('total_vaccinations', ascending=False).dropna(sub
vacc_amount1.drop(columns= ["iso_code","vaccines"] , inplace= True)
df_pie = vacc_amount1.head(7)
myexplode = [0, 0, 0, 0.3,0,0,0]
plt.pie(df_pie['total_vaccinations'], labels =  ["US", "China" , "UK" , "India" , "Israel" , "Brazil" , "Turke
plt.show()
```



Inferences and Conclusion

After Analyzing and visualizing each of the datasets of Jan-Feb and Oct-Nov.

1) Many countries stated applying vaccines to their people by the end of 2020, whereas India started its vaccination program from 15th Jan 2021.

2) The rate of applying vaccines to the patients is highest in UK, whereas it is lowest in India. One of the reasons the rate of vaccine reaching people is low could be because India is a developing country.

3) Moderna, Pfizer/BioNTech is the most popular vaccine used worldwide, since it has almost negligible side effects (known till date). Also, India uses Covaxin, Covishield for vaccinating its subjects.

4) Maximum number of people vaccinated is highest in USA since USA is a developed nation as well as it has better health facilities than other developing nations.

5) India has seen an exponential rise in the amount of vaccinations in the span of these 6 months as it is quite evident as well. This is an remarkable achievement in itself.

6) Covaxin, a vaccine of Indian origin is at the top of the list of kind of vaccines used by the end of Oct-Nov.

From the above inferences it can be concluded that people from all the parts of the world are educating themselves and willingly taking the vaccines in most parts of the world. Also, these vaccines have been proved effective against COVID-19 (till now).

If the rate of people taking the vaccine continues to grow, then all the countries can vaccinate their people before the end of this year.

References

Dataset : https://www.kaggle.com/gpreda/covid-world-vaccination-progress

DateTime library documentation : https://docs.python.org/3/library/datetime.html

Matplotlib documentation : https://matplotlib.org/3.1.1/contents.html

Tutorialspoint : https://www.tutorialspoint.com/matplotlib/matplotlib_bar_plot.htm

Seaborn documentation : https://seaborn.pydata.org/introduction.html

Pandas documentation : https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.items.html

pie charts in matplotlib (w3schools) : https://www.w3schools.com/python/matplotlib_pie_charts.asp