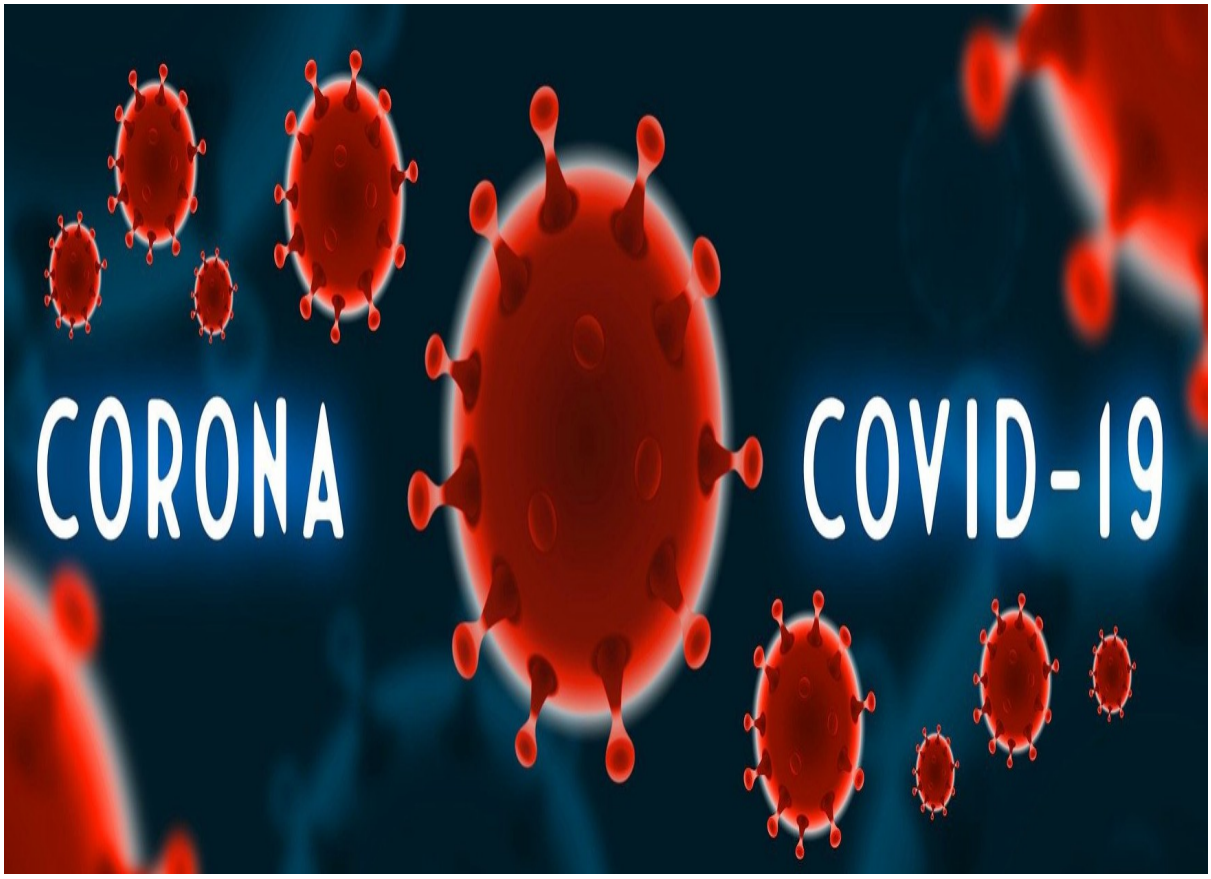
Project: Exploratory Data Analysis on Covid Dataset

Submitted By

ADAIKKAPPAN A (EBEON0522601032)



ABSTRACT

Coronavirus disease (COVID-19) is an infectious disease caused by the SARS-CoV-2 virus.

Most people infected with the virus will experience mild to moderate respiratory illness and recover without requiring special treatment. However, some will become seriously ill and require medical attention. Older people and those with underlying medical conditions like cardiovascular disease, diabetes, chronic respiratory disease, or cancer are more likely to develop serious illness. Anyone can get sick with COVID-19 and become seriously ill or die at any age.

The best way to prevent and slow down transmission is to be well informed about the disease and how the virus spreads. Protect yourself and others from infection by staying at least 1 metre apart from others, wearing a properly fitted mask, and washing your hands or using an alcohol-based rub frequently. Get vaccinated when it's your turn and follow local guidance.

The virus can spread from an infected person's mouth or nose in small liquid particles when they cough, sneeze, speak, sing or breathe. These particles range from larger respiratory droplets to smaller aerosols. It is important to practice respiratory etiquette, for example by coughing into a flexed elbow, and to stay home and self-isolate until you recover if you feel unwell.

Now, retailers need a 360-degree view of their consumers, without which, they can miss competitive edge of the market. Retailers have to create effective promotions and offers to meet its sales and marketing goals, otherwise they will forgo the major opportunities that the current market offers. Many times it is hard for the retailers to comprehend the market condition since their retail stores are at various geographical locations. Big Data application enables these retail organizations to use prior year's data to better forecast and predict the coming year's sales. It also enables retailers with valuable and analytical insights, especially determining customers with desired products at desired time in a particular store at different geographical locations.

Now, retailers need a 360-degree view of their consumers, without which, they can miss competitive edge of the market. Retailers have to create effective promotions and offers to meet its sales and marketing goals, otherwise they will forgo the major opportunities that the current market offers. Many times it is hard for the retailers to comprehend the market condition since their retail stores are at various geographical locations. Big Data application enables these retail organizations to use prior year's data to better forecast and predict the coming year's sales. It also enables retailers with valuable and analytical insights, especially determining customers with desired products at desired time in a particular store at different geographical locations.

Now, retailers need a 360-degree view of their consumers, without which, they can miss competitive edge of the market. Retailers have to create effective promotions and offers to meet its sales and marketing goals, otherwise they will forgo the major opportunities that the current market offers. Many times it is hard for the retailers to comprehend the market condition since their retail stores are at various geographical locations. Big Data application enables these retail organizations to use prior year's data to better forecast and predict the coming year's sales. It also enables retailers with valuable and analytical insights, especially determining customers with desired products at desired time in a particular store at different geographical locations.

Now, retailers need a 360-degree view of their consumers, without which, they can miss competitive edge of the market. Retailers have to create effective promotions and offers to meet its sales and marketing goals, otherwise they will forgo the major opportunities that the current market offers. Many times it is hard for the retailers to comprehend the market condition since their retail stores are at various geographical locations. Big Data application enables these retail organizations to use prior year's data to better forecast and predict the coming year's sales. It also enables retailers with valuable and analytical insights, especially determining customers with desired products at desired time in a particular store at different geographical locations.

Now, retailers need a 360-degree view of their

consumers, without which, they can miss competitive edge of the market. Retailers have to create effective promotions and offers to meet its sales and marketing goals, otherwise they will forgo the major opportunities that the current market offers. Many times it is hard for the retailers to comprehend the market condition since their retail stores are at various geographical locations. Big Data application enables these retail organizations to use prior year's data to better forecast and predict the coming year's sales. It also enables retailers with valuable and analytical insights, especially determining customers with desired products at desired time in a particular store at different geographical locationsNow, retailers need a 360-degree view of their consumers, without which, they can miss competitive edge of the market. Retailers have to create effective promotions and offers to meet its sales and marketing goals, otherwise they will forgo the major opportunities that the current market offers. Many times it is hard for the retailers to comprehend the market condition since their retail stores are at various geographical locations. Big Data application enables these retail organizations to use prior year's data to better forecast and predict the coming year's sales. It also enables retailers with valuable and analytical insights, especially determining customers with desired products at desired time in a particular store at different geographical locations. In this paper, we analysed the data sets of world's largest retailers, Walmart Store to determine the business drivers and predict which departments are affected by the different scenarios (such as temperature, fuel price and holidays) and their impact on sales at stores' of different locations. We have made use of Scala and Python API of the Spark framework to gain new insights into the consumer behaviours and comprehend Walmart's marketing efforts and their data-driven strategies through visual representation of the analyzed data.

CONTENTS

TABLE OF CONTENTS

Chapter 1

Introduction

Since its emergence in December 2019, corona virus disease 2019 (COVID-19) has impacted several countries, affected more than 90 thousand patients and made it a global public threat. The routes of transmission are direct contact, and droplet and possible aerosol transmissions. Due to the unique nature of dentistry, most dental procedures generate significant amounts of droplets and aerosols, posing potential risks of infection transmission. Understanding the significance of aerosol transmission and its implications in dentistry can facilitate the identification and correction of negligence in daily dental practice. some special precautions that need to be implemented.

Scenario:

you have just been hired as a Data Scientist at a government health department to do the EDA that enables you to use prior year's data to better forecast and predict the coming year's pandemic precautions and what are the safety measures need to take care in advance.

Goal:

- Examine which year has the highest confirmed cases and death cases.
- Examine which states having the highest confirmed, death and cured cases.
- Examine which month has the highest confirmed cases and death cases.
- Examine why the death cases are increasing?
- Predict the coming year's pandemic precautions.

Chapter 2

Features & Predictor

StatewiseTestingDetails.csv Dataset

- Date - object
- State - object
- TotalSamples - float64
- Negative - object
- Positive - float64

Note

- Total - 5 columns
- Decimal – 3 - Continuous: Which is quantitative data that can be measured.
- String – 1 - Ordinal Data: Categorical data that has an00 order to it.
- Date Time – 1 - Continuous: Which is quantitative data that can be measured.

Features & Predictor

Covid_19_india.csv Dataset

- Sno – int64
- Date – object
- Time – object
- State/UnionTerritory – object
- ConfirmedIndianNational – object
- ConfirmedForeignNational – object
- Cured - int64
- Deaths – int64
- Confirmed – int64

Note

- Total – 9 columns
- Decimal – 4 – Continuous: Which is quantitative data that can be measured.
- String – 3 – Ordinal Data: Categorical data that has an order to it.
- Date Time – 2 – Continuous: Which is quantitative data that can be measured.

Features & Predictor

covid_vaccine_statewise.csv Dataset

- Updated - object
- State - object
- Total Doses Administered - float64
- Sessions - float64
- Sites - float64
- First Dose Administered - float64
- Second Dose Administered - float64
- Male (Doses Administered) - float64
- Female (Doses Administered) - float64
- Transgender (Doses Administered) - float64
- Covaxin (Doses Administered) - float64
- CoviShield (Doses Administered) - float64
- Sputnik V (Doses Administered) - float64
- AEFI - float64
- 18-44 Years (Doses Administered) - float64
- 45-60 Years (Doses Administered) - float64
- 60+ Years (Doses Administered) - float64
- 18-44 Years(Individuals Vaccinated) - float64
- 45-60 Years(Individuals Vaccinated) - float64
- 60+ Years(Individuals Vaccinated) - float64
- Male(Individuals Vaccinated) - float64
- Female(Individuals Vaccinated) - float64
- Transgender(Individuals Vaccinated) - float64
- Total Individuals Vaccinated - float64

Note

- Total - 24 columns
- Decimal – 22 - Continuous: Which is quantitative data that can be measured.
- String – 1 - Ordinal Data: Categorical data that has a order to it.
- Date Time – 1 - Continuous: Which is quantitative data that can be measured.

Chapter 3
Methodology

Data Cleaning and Pre-processing:

The datasets which were collected from COVID-19 in India Dataset from Kaggle website contain unfiltered data which must be filtered before the final data set can be used to do analysis. Also, data has some categorical variables which must be modified into numerical values for which we used Panda's library of Python. In data cleaning step, first we checked whether there are any missing or junk values in the dataset for which we used the is null () function.

Implementation Steps:

As we already discussed in the methodology section about some of the implementation details. So, the language used in this project is Python programming. We're running python code in anaconda navigator's Jupyter notebook. Jupyter notebook is much faster than Python IDE tools like PyCharm or Visual studio for implementing ML algorithms. The advantage of Jupyter notebook is that while writing code, it's really helpful for Data visualization and plotting some graphs like histogram and heatmap of correlated matrices. Let's revise implementation steps: a) Dataset collection. b) Importing Libraries: NumPy, Pandas, Matplotlib and Seaborn libraries were used. c) Exploratory data analysis: For getting more insights about data. d) Data cleaning and pre-processing: Checked for null and junk values using isnull() and isna().sum() functions of python. In Pre-processing phase, we did feature engineering on our dataset. As we converted categorical variables into numerical variables using function of Pandas library. All our datasets contains some categorical variables.

LIBRARIES:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px


import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

Data Wrangling:

```
cvd_ind=pd.read_csv("covid_19_india.csv")
vacc_st_wise=pd.read_csv("covid_vaccine_statewise.csv")
st_test_dt=pd.read_csv("StatewiseTestingDetails.csv")
```

Display the contents of covid_19_india.csv file:

```
In [4]: 1 cvd_ind
```

Out[4]:

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed
0	1	2020-01-30	6:00 PM	Kerala	1	0	0	0	1
1	2	2020-01-31	6:00 PM	Kerala	1	0	0	0	1
2	3	2020-02-01	6:00 PM	Kerala	2	0	0	0	2
3	4	2020-02-02	6:00 PM	Kerala	3	0	0	0	3
4	5	2020-02-03	6:00 PM	Kerala	3	0	0	0	3
...	...	...	...	...	...	...	...	...	...
18105	18106	2021-08-11	8:00 AM	Telangana	-	-	638410	3831	650353
18106	18107	2021-08-11	8:00 AM	Tripura	-	-	77811	773	80660
18107	18108	2021-08-11	8:00 AM	Uttarakhand	-	-	334650	7368	342462
18108	18109	2021-08-11	8:00 AM	Uttar Pradesh	-	-	1685492	22775	1708812
18109	18110	2021-08-11	8:00 AM	West Bengal	-	-	1506532	18252	1534999

18110 rows × 9 columns

Find out how many rows and columns:

```
In [5]: 1 cvd_ind.shape
```

```
Out[5]: (18110, 9)
```

Find out all the column Names:

```
In [6]: 1 cvd_ind.columns
```

```
Out[6]: Index(['Sno', 'Date', 'Time', 'State/UnionTerritory',
       'ConfirmedIndianNational', 'ConfirmedForeignNational', 'Cured',
       'Deaths', 'Confirmed'],
      dtype='object')
```

Find out all the column Names and its corresponding datatypes:

```
In [7]: 1 cvd_ind.dtypes
```

```
Out[7]: Sno                       int64
        Date                     object
        Time                     object
        State/UnionTerritory     object
        ConfirmedIndianNational  object
        ConfirmedForeignNational object
        Cured                     int64
        Deaths                    int64
        Confirmed                 int64
        dtype: object
```

To check any null values is available or not:

```
In [8]: 1 cvd_ind.isnull().sum()
```

```
Out[8]: Sno                         0
        Date                        0
        Time                        0
        State/UnionTerritory        0
        ConfirmedIndianNational     0
        ConfirmedForeignNational    0
        Cured                       0
        Deaths                      0
        Confirmed                   0
        dtype: int64
```

To check the Duplicate records:

```
In [9]: 1 cvd_ind.duplicated().sum()
```

```
Out[9]: 0
```

Display the contents of covid_vaccine_statewise.csv file:

```
1 vacc_st_wise
```

	Updated On	State	Total Doses Administered	Sessions	Sites	First Dose Administered	Second Dose Administered	Male (Doses Administered)	Female (Doses Administered)	Transgender (Doses Administered)	...	18-44 Years (Doses Administered)	45-60 (Adminis
0	16/01/2021	India	48276.0	3455.0	2957.0	48276.0	0.0	NaN	NaN	NaN	...	NaN	
1	17/01/2021	India	58604.0	8532.0	4954.0	58604.0	0.0	NaN	NaN	NaN	...	NaN	
2	18/01/2021	India	99449.0	13611.0	6583.0	99449.0	0.0	NaN	NaN	NaN	...	NaN	
3	19/01/2021	India	195525.0	17855.0	7951.0	195525.0	0.0	NaN	NaN	NaN	...	NaN	
4	20/01/2021	India	251280.0	25472.0	10504.0	251280.0	0.0	NaN	NaN	NaN	...	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	
7840	11/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
7841	12/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
7842	13/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
7843	14/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
7844	15/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	

7845 rows × 24 columns

Find out how many rows and columns:

```
In [11]: 1 vacc_st_wise.shape
```

```
Out[11]: (7845, 24)
```

Find out all the column names:

```
In [12]: 1 vacc_st_wise.columns
```

```
Out[12]: Index(['Updated On', 'State', 'Total Doses Administered', 'Sessions',
                ' Sites ', 'First Dose Administered', 'Second Dose Administered',
                'Male (Doses Administered)', 'Female (Doses Administered)',
                'Transgender (Doses Administered)', ' Covaxin (Doses Administered)',
                'CoviShield (Doses Administered)', 'Sputnik V (Doses Administered)',
                'AEFI', '18-44 Years (Doses Administered)',
                '45-60 Years (Doses Administered)', '60+ Years (Doses Administered)',
                '18-44 Years(Individuals Vaccinated)',
                '45-60 Years(Individuals Vaccinated)',
                '60+ Years(Individuals Vaccinated)', 'Male(Individuals Vaccinated)',
                'Female(Individuals Vaccinated)', 'Transgender(Individuals Vaccinated)',
                'Total Individuals Vaccinated'],
               dtype='object')
```

Find out all the column Names and its corresponding datatypes:

```
In [13]: 1 vacc_st_wise.dtypes
```

```
Out[13]: Updated On                          object
         State                               object
         Total Doses Administered           float64
         Sessions                           float64
          Sites                             float64
         First Dose Administered            float64
         Second Dose Administered           float64
         Male (Doses Administered)          float64
         Female (Doses Administered)        float64
         Transgender (Doses Administered)   float64
          Covaxin (Doses Administered)      float64
         CoviShield (Doses Administered)    float64
         Sputnik V (Doses Administered)     float64
         AEFI                               float64
         18-44 Years (Doses Administered)   float64
         45-60 Years (Doses Administered)   float64
         60+ Years (Doses Administered)     float64
         18-44 Years(Individuals Vaccinated) float64
         45-60 Years(Individuals Vaccinated) float64
         60+ Years(Individuals Vaccinated)  float64
         Male(Individuals Vaccinated)       float64
         Female(Individuals Vaccinated)     float64
         Transgender(Individuals Vaccinated) float64
         Total Individuals Vaccinated       float64
         dtype: object
```

To check any null values is available or not:

```
In [14]: 1 vacc_st_wise.isnull().sum()
```

```
Out[14]: Updated On                              0
         State                                   0
         Total Doses Administered              224
         Sessions                              224
          Sites                                224
         First Dose Administered               224
         Second Dose Administered              224
         Male (Doses Administered)             384
         Female (Doses Administered)           384
         Transgender (Doses Administered)      384
          Covaxin (Doses Administered)         224
         CoviShield (Doses Administered)       224
         Sputnik V (Doses Administered)       4850
         AEFI                                 2407
         18-44 Years (Doses Administered)     6143
         45-60 Years (Doses Administered)     6143
         60+ Years (Doses Administered)       6143
         18-44 Years(Individuals Vaccinated)  4112
         45-60 Years(Individuals Vaccinated)  4111
         60+ Years(Individuals Vaccinated)    4111
         Male(Individuals Vaccinated)         7685
         Female(Individuals Vaccinated)       7685
         Transgender(Individuals Vaccinated)  7685
         Total Individuals Vaccinated         1926
         dtype: int64
```

To check the Duplicate records:

```
In [107]: 1 vacc_st_wise.duplicated().sum()
```

```
Out[107]: 0
```

Display the contents of StatewiseTestingDetails.csv file:

```
In [16]: 1 st_test_dt
```

Out[16]:

	Date	State	TotalSamples	Negative	Positive
0	2020-04-17	Andaman and Nicobar Islands	1403.0	1210	12.0
1	2020-04-24	Andaman and Nicobar Islands	2679.0	NaN	27.0
2	2020-04-27	Andaman and Nicobar Islands	2848.0	NaN	33.0
3	2020-05-01	Andaman and Nicobar Islands	3754.0	NaN	33.0
4	2020-05-16	Andaman and Nicobar Islands	6677.0	NaN	33.0
...	...	...	...	...	...
16331	2021-08-06	West Bengal	15999961.0	NaN	NaN
16332	2021-08-07	West Bengal	16045662.0	NaN	NaN
16333	2021-08-08	West Bengal	16092192.0	NaN	NaN
16334	2021-08-09	West Bengal	16122345.0	NaN	NaN
16335	2021-08-10	West Bengal	16162814.0	NaN	NaN

16336 rows × 5 columns

Find out how many rows and columns:

```
In [17]: 1 st_test_dt.shape
```

```
Out[17]: (16336, 5)
```

Find out all the column Names:

```
In [18]: 1 st_test_dt.columns
```

```
Out[18]: Index(['Date', 'State', 'TotalSamples', 'Negative', 'Positive'], dtype='object')
```

Find out all the column Names and its corresponding datatypes:

```
In [19]: 1 st_test_dt.dtypes
```

```
Out[19]: Date            object
         State           object
         TotalSamples   float64
         Negative        object
         Positive       float64
         dtype: object
```

To check any null values is available or not:

```
In [11]: 1 st_test_dt.isnull().sum()
```

```
Out[11]: Date              0
         State             0
         TotalSamples      0
         Negative       9367
         Positive      10674
         dtype: int64
```

To check the Duplicate records:

```
In [21]: 1 st_test_dt.duplicated().sum()
```

```
Out[21]: 1
```

Drop the Duplicate records:

```
In [22]: 1 st_test_dt=st_test_dt.drop_duplicates()
```

Exploratory Data Analysis:

Display the contents of covid_19_india.csv file:

```
In [23]: 1 cvd_ind
```

Out[23]:

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed
0	1	2020-01-30	6:00 PM	Kerala	1	0	0	0	1
1	2	2020-01-31	6:00 PM	Kerala	1	0	0	0	1
2	3	2020-02-01	6:00 PM	Kerala	2	0	0	0	2
3	4	2020-02-02	6:00 PM	Kerala	3	0	0	0	3
4	5	2020-02-03	6:00 PM	Kerala	3	0	0	0	3
...	...	...	...	...	...	...	...	...	...
18105	18106	2021-08-11	8:00 AM	Telangana	-	-	638410	3831	650353
18106	18107	2021-08-11	8:00 AM	Tripura	-	-	77811	773	80660
18107	18108	2021-08-11	8:00 AM	Uttarakhand	-	-	334650	7368	342462
18108	18109	2021-08-11	8:00 AM	Uttar Pradesh	-	-	1685492	22775	1708812
18109	18110	2021-08-11	8:00 AM	West Bengal	-	-	1506532	18252	1534999

18110 rows × 9 columns

Drop the unnecessary columns from the original dataset:

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

```
In [25]: 1 cvd_ind.drop(['Time'],axis=1,inplace=True)
```

Create two new columns month and year from the Date column:

```
In [26]: 1 cvd_ind['month']=pd.to_datetime(cvd_ind['Date']).dt.month
```

```
In [27]: 1 cvd_ind['year']=pd.to_datetime(cvd_ind['Date']).dt.year
```

Read first five lines of covid_19_india.csv file:

```
In [28]: 1 cvd_ind.head()
```

Out[28]:

	Date	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed	month	year
0	2020-01-30	Kerala	1	0	0	0	1	1	2020
1	2020-01-31	Kerala	1	0	0	0	1	1	2020
2	2020-02-01	Kerala	2	0	0	0	2	2	2020
3	2020-02-02	Kerala	3	0	0	0	3	2	2020
4	2020-02-03	Kerala	3	0	0	0	3	2	2020

Read last five lines of covid_19_india.csv file:

```
In [29]: 1 cvd_ind.tail()
```

Out[29]:

	Date	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed	month	year
18105	2021-08-11	Telangana	-	-	638410	3831	650353	8	2021
18106	2021-08-11	Tripura	-	-	77811	773	80660	8	2021
18107	2021-08-11	Uttarakhand	-	-	334650	7368	342462	8	2021
18108	2021-08-11	Uttar Pradesh	-	-	1685492	22775	1708812	8	2021
18109	2021-08-11	West Bengal	-	-	1506532	18252	1534999	8	2021

Satistical Information about the covid_19_india.csv data:

```
In [30]: 1 cvd_ind.describe()
```

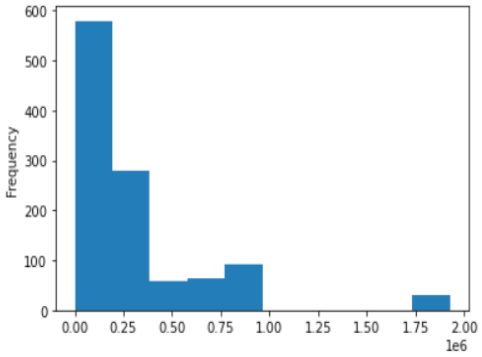
Out[30]:

	Cured	Deaths	Confirmed	month	year
count	1.811000e+04	18110.000000	1.811000e+04	18110.000000	18110.000000
mean	2.786375e+05	4052.402264	3.010314e+05	6.207565	2020.443291
std	6.148909e+05	10919.076411	6.561489e+05	3.060156	0.496787
min	0.000000e+00	0.000000	0.000000e+00	1.000000	2020.000000
25%	3.360250e+03	32.000000	4.376750e+03	4.000000	2020.000000
50%	3.336400e+04	588.000000	3.977350e+04	6.000000	2020.000000
75%	2.788698e+05	3643.750000	3.001498e+05	8.000000	2021.000000
max	6.159676e+06	134201.000000	6.363442e+06	12.000000	2021.000000

Plot the graph for confirmed cases in the month of 12(December):

```
In [32]: 1 cvd_ind[cvd_ind['month']==12]['Confirmed'].plot(kind="hist")
```

```
Out[32]: <AxesSubplot:ylabel='Frequency'>
```



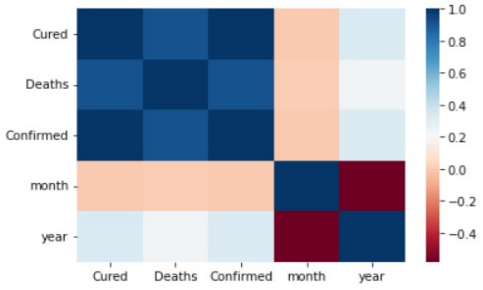
Find the Correlation for covid_19_india Dataset:

```
In [33]: 1 cvd_ind.corr()
```

Out[33]:

	Cured	Deaths	Confirmed	month	year
Cured	1.000000	0.917529	0.997751	-0.010752	0.337670
Deaths	0.917529	1.000000	0.918346	0.005348	0.242629
Confirmed	0.997751	0.918346	1.000000	-0.008707	0.332204
month	-0.010752	0.005348	-0.008707	1.000000	-0.580975
year	0.337670	0.242629	0.332204	-0.580975	1.000000

```
In [34]: 1 sns.heatmap(cvd_ind.corr(),cmap='RdBu')
         2 plt.show()
```



Observation:

- If you see the above correlation, Deaths is highly correlated with Cured and Confirmed
- Confirmed is highly Cured and Deaths
- Cured is highly Deaths and Confirmed
- Month is not correlated with cured

To Find the Covariance:

```
In [35]: 1 cvd_ind.cov()
```

Out[35]:

	Cured	Deaths	Confirmed	month	year
Cured	3.780908e+11	6.160329e+09	4.025525e+11	-20231.387176	103148.117623
Deaths	6.160329e+09	1.192262e+08	6.579523e+09	178.690093	1316.130743
Confirmed	4.025525e+11	6.579523e+09	4.305313e+11	-17482.957670	108287.360548
month	-2.023139e+04	1.786901e+02	-1.748296e+04	9.364557	-0.883226
year	1.031481e+05	1.316131e+03	1.082874e+05	-0.883226	0.246798

To check how many records are available for each State/UnionTerritory:

```
In [36]: 1 cvd_ind['State/UnionTerritory'].value_counts()
```

Out[36]:

```
Kerala                  560
Delhi                   528
Rajasthan               527
Uttar Pradesh           526
Haryana                 526
Ladakh                  523
Tamil Nadu              523
Punjab                  521
Jammu and Kashmir       521
Maharashtra             520
Karnataka               520
Andhra Pradesh          518
Uttarakhand             515
Odisha                  514
Puducherry              512
```

Create a copy of duplicate data set from the original data set:

```
In [37]: 1 cvd_ind1=cvd_ind.copy()
         2 cvd_ind1
```

Out[37]:

	Date	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed	month	year
0	2020-01-30	Kerala	1	0	0	0	1	1	2020
1	2020-01-31	Kerala	1	0	0	0	1	1	2020
2	2020-02-01	Kerala	2	0	0	0	2	2	2020
3	2020-02-02	Kerala	3	0	0	0	3	2	2020
4	2020-02-03	Kerala	3	0	0	0	3	2	2020
...	...	...	...	...	...	...	...	...	...
18105	2021-08-11	Telangana	-	-	638410	3831	650353	8	2021
18106	2021-08-11	Tripura	-	-	77811	773	80660	8	2021
18107	2021-08-11	Uttarakhand	-	-	334650	7368	342462	8	2021
18108	2021-08-11	Uttar Pradesh	-	-	1685492	22775	1708812	8	2021
18109	2021-08-11	West Bengal	-	-	1506532	18252	1534999	8	2021

18110 rows × 9 columns

Replace the hyphen symbol to 0 for the ConfirmedIndianNational and ConfirmedForeignNational columns:

In [38]:
```
cvd_ind1['ConfirmedIndianNational']=cvd_ind1.ConfirmedIndianNational.mask(cvd_ind1.ConfirmedIndianNational=='-',0)
cvd_ind1['ConfirmedForeignNational']=cvd_ind1.ConfirmedForeignNational.mask(cvd_ind1.ConfirmedForeignNational=='-',0)
cvd_ind1
```

Out[38]:

	Date	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed	month	year
0	2020-01-30	Kerala	1	0	0	0	1	1	2020
1	2020-01-31	Kerala	1	0	0	0	1	1	2020
2	2020-02-01	Kerala	2	0	0	0	2	2	2020
3	2020-02-02	Kerala	3	0	0	0	3	2	2020
4	2020-02-03	Kerala	3	0	0	0	3	2	2020
...	...	...	...	...	...	...	...	...	...
18105	2021-08-11	Telangana	0	0	638410	3831	650353	8	2021
18106	2021-08-11	Tripura	0	0	77811	773	80660	8	2021
18107	2021-08-11	Uttarakhand	0	0	334650	7368	342462	8	2021
18108	2021-08-11	Uttar Pradesh	0	0	1685492	22775	1708812	8	2021
18109	2021-08-11	West Bengal	0	0	1506532	18252	1534999	8	2021

18110 rows × 9 columns

Replace the hyphen symbol to 0 for the Cured, Deaths and Confirmed columns:

In [39]:
```
cvd_ind1['Cured']=cvd_ind1.Cured.mask(cvd_ind1.Cured=='-',0)
cvd_ind1['Deaths']=cvd_ind1.Deaths.mask(cvd_ind1.Deaths=='-',0)
cvd_ind1['Confirmed']=cvd_ind1.Confirmed.mask(cvd_ind1.Confirmed=='-',0)
cvd_ind1
```

Out[39]:

	Date	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed	month	year
0	2020-01-30	Kerala	1	0	0	0	1	1	2020
1	2020-01-31	Kerala	1	0	0	0	1	1	2020
2	2020-02-01	Kerala	2	0	0	0	2	2	2020
3	2020-02-02	Kerala	3	0	0	0	3	2	2020
4	2020-02-03	Kerala	3	0	0	0	3	2	2020
...	...	...	...	...	...	...	...	...	...
18105	2021-08-11	Telangana	0	0	638410	3831	650353	8	2021
18106	2021-08-11	Tripura	0	0	77811	773	80660	8	2021
18107	2021-08-11	Uttarakhand	0	0	334650	7368	342462	8	2021
18108	2021-08-11	Uttar Pradesh	0	0	1685492	22775	1708812	8	2021
18109	2021-08-11	West Bengal	0	0	1506532	18252	1534999	8	2021

18110 rows × 9 columns

Information about the Covid data set:

In [40]:
```
cvd_ind1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18110 entries, 0 to 18109
Data columns (total 9 columns):
 #   Column                   Non-Null Count  Dtype 
---  ------                   --------------  ----- 
 0   Date                     18110 non-null  object
 1   State/UnionTerritory     18110 non-null  object
 2   ConfirmedIndianNational  18110 non-null  object
 3   ConfirmedForeignNational  18110 non-null  object
 4   Cured                    18110 non-null  int64 
 5   Deaths                   18110 non-null  int64 
 6   Confirmed                18110 non-null  int64 
 7   month                    18110 non-null  int64 
 8   year                     18110 non-null  int64 
dtypes: int64(5), object(4)
memory usage: 1.2+ MB
```

Convert the object datatype to int datatype:

In [41]:
```
cvd_ind1.ConfirmedIndianNational = cvd_ind1.ConfirmedIndianNational.astype('int64')
cvd_ind1.ConfirmedForeignNational = cvd_ind1.ConfirmedForeignNational.astype('int64')
```

State/UnionTerritory wise ConfirmedIndianNational Report:

In [42]:
```
st_wise_confirmed_ind_nt=cvd_ind1.groupby(['State/UnionTerritory'])['ConfirmedIndianNational'].sum().sort_values(ascending=F
st_wise_confirmed_ind_nt
```

Out[42]:
```
State/UnionTerritory
Maharashtra                 1111
Kerala                      1091
Uttar Pradesh                462
Karnataka                    405
Delhi                        352
Rajasthan                    296
Gujarat                      256
Telengana                    246
Punjab                       231
Ladakh                       162
Tamil Nadu                   138
Haryana                      111
Madhya Pradesh               105
Jammu and Kashmir             95
Andhra Pradesh                81
West Bengal                   71
Chandigarh                    50
Uttarakhand                   35
Bihar                         32
Chhattisgarh                  25
Odisha                        23
Himachal Pradesh              21
Puducherry                    11
Goa                            9
Andaman and Nicobar Islands    8
```

Observation:

- Maharashtra state was having the highest no. of ConfirmedIndianNationals 1111 followed by Kerala,Uttar Pradesh,Karnataka,Delhi
- These are first five states having highest no. of ConfirmedIndianNationals

State/UnionTerritory wise ConfirmedForeignNational Report:

```
In [43]: st_wise_confirmed_For_nt=cvd_ind1.groupby(['State/UnionTerritory'])['ConfirmedForeignNational'].sum().sort_values(ascending=
         st_wise_confirmed_For_nt
```

Out[43]:

State/UnionTerritory	
Haryana	266
Rajasthan	135
Telengana	97
Kerala	69
Maharashtra	36
Tamil Nadu	26
Uttar Pradesh	17
Delhi	11
Uttarakhand	5
Gujarat	5
Odisha	0
Maharashtra***	0

Observation:

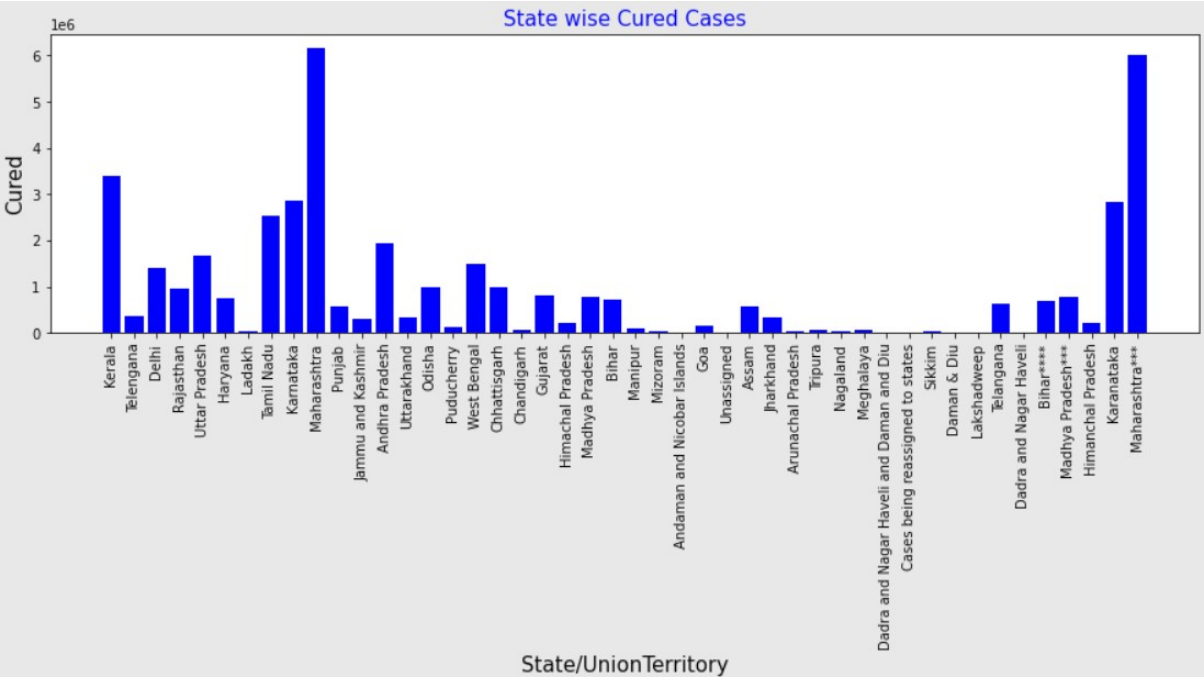
- Haryana was having the highest no. of ConfirmedForeignNationals 266 followed by Rajasthan,Telengana,Kerala,Maharashtra.
- These are first five states having highest no. of ConfirmedForeignNationals.

State/UnionTerritory wise Cured Cases Report:

In [44]:
```
st_wise_cured=cvd_ind1.groupby(['State/UnionTerritory'])['Cured'].sum().sort_values(ascending=False)
st_wise_cured
```

Out[44]:
```
State/UnionTerritory
Maharashtra                    1018765039
Karnataka                       441844360
Kerala                          420174235
Tamil Nadu                      404095807
Andhra Pradesh                  370426530
Uttar Pradesh                   291479351
Delhi                           273419887
West Bengal                     247515102
Chhattisgarh                    151609364
Odisha                          150923455
Rajasthan                       150356820
Gujarat                         132487127
Madhya Pradesh                  126724997
Haryana                         126585342
Bihar                           125122902
Assam                            92678680
Punjab                           91458159
Telengana                        64666267
Jharkhand                        58034506
Telangana                        57488245
Jammu and Kashmir                53297341
Uttarakhand                      48362741
Himachal Pradesh                 27501110
Goa                              26027201
Puducherry                       18483117
Tripura                          12976846
Manipur                          11230568
Chandigarh                       10117035
Arunachal Pradesh                 6588149
Meghalaya                         6537909
Maharashtra***                    6000911
```

In [108]:
```
plt.figure(figsize=(15,4))
plt.bar(cvd_ind1['State/UnionTerritory'],cvd_ind1['Cured'],color='blue')
plt.title("State wise Cured Cases",fontsize=15,color='blue')
plt.xlabel("State/UnionTerritory",fontsize=15)
plt.ylabel("Cured",fontsize=15)
plt.xticks(rotation=90)
plt.grid(False)
plt.show()
```



Observation:

- Maharashtra was having the highest no. of Cured Cases 1018765039 followed by Karnataka,Kerala,Tamil Nadu,Andhra Pradesh.
- These are first five states having highest no. of Cured Cases.

State/UnionTerritory wise Death Cases Report:

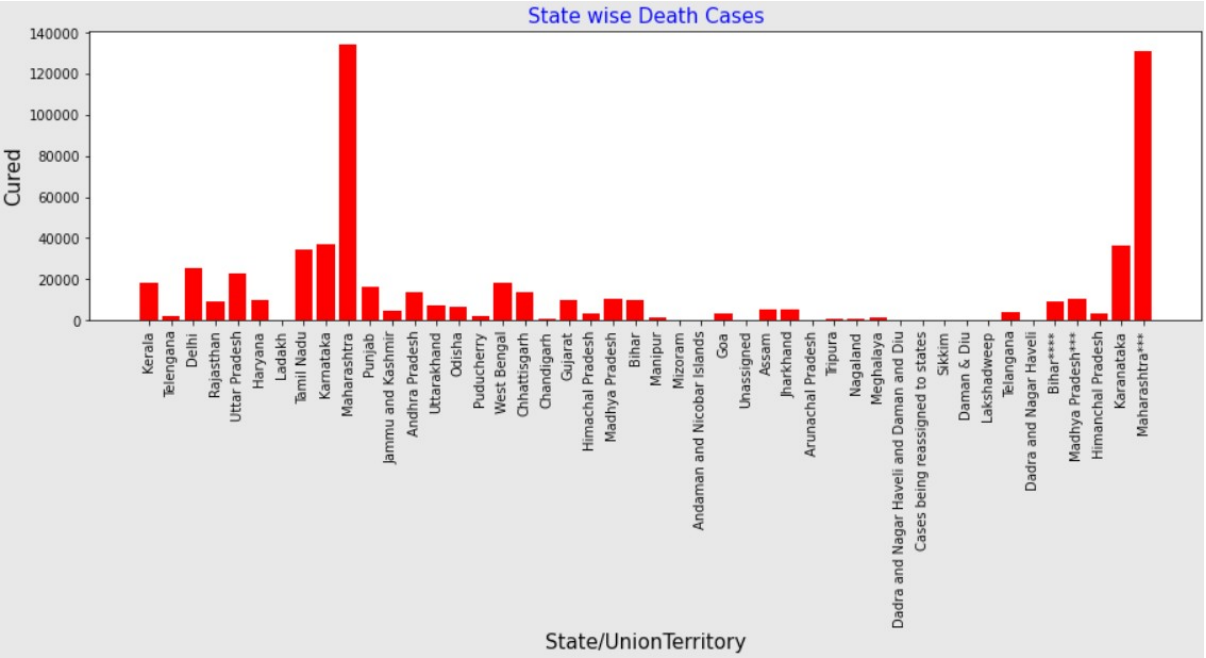
In [46]:
```
st_wise_deaths=cvd_ind1.groupby(['State/UnionTerritory'])['Deaths'].sum().sort_values(ascending=False)
st_wise_deaths
```

Out[46]:

State/UnionTerritory	
Maharashtra	23737432
Karnataka	6053762
Tamil Nadu	5916658
Delhi	4943294
Uttar Pradesh	4143450
West Bengal	3846989
Andhra Pradesh	2939367
Punjab	2785594
Gujarat	2219448
Chhattisgarh	2063920
Kerala	1888177
Madhya Pradesh	1777752
Haryana	1502799
Rajasthan	1473089
Bihar	1093466
Uttarakhand	986001
Jammu and Kashmir	839694

In [109]:

```
plt.figure(figsize=(15,4))
plt.bar(cvd_ind1['State/UnionTerritory'],cvd_ind1['Deaths'],color='red')
plt.title("State wise Death Cases",fontsize=15,color='blue')
plt.xlabel("State/UnionTerritory",fontsize=15)
plt.ylabel("Cured",fontsize=15)
plt.xticks(rotation=90)
plt.grid(False)
plt.show()
```



Observation:

- Maharashtra was having the highest no. of Cured Cases 23737432 followed by Karnataka,Tamil Nadu,Delhi,Uttar Pradesh.
- These are first five states having highest no. of Death Cases.

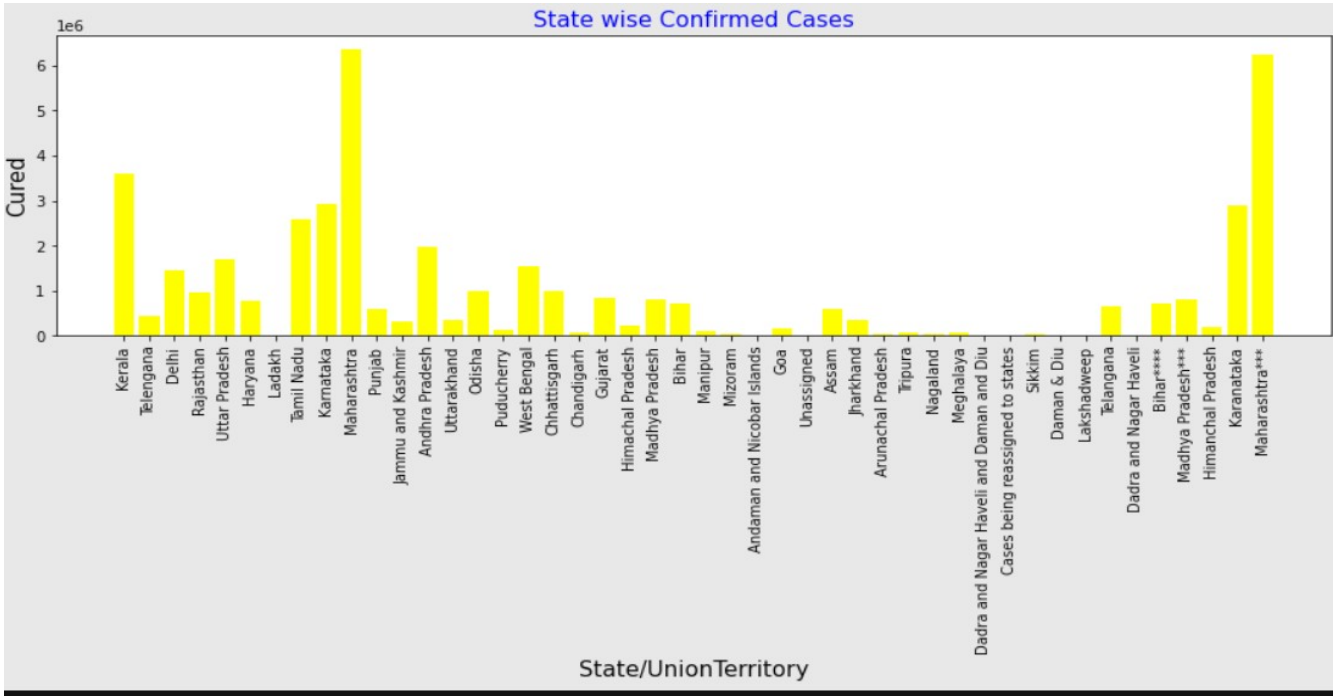
State/UnionTerritory wise Confirmed Cases Report:

In [48]:
```
st_wise_cnfm=cvd_ind1.groupby(['State/UnionTerritory'])['Confirmed'].sum().sort_values(ascending=False)
st_wise_cnfm
```

Out[48]:

State/UnionTerritory	
Maharashtra	1121491467
Karnataka	485970693
Kerala	458906023
Tamil Nadu	431928644
Andhra Pradesh	392432753
Uttar Pradesh	312625843
Delhi	287227765
West Bengal	263107876
Chhattisgarh	163776262
Rajasthan	162369656
Odisha	160130533
Gujarat	143420082
Madhya Pradesh	135625265
Haryana	134347285
Bihar	132231166
Punjab	99949702
Assam	99837011
Telengana	69990668
Jharkhand	62111994
Telangana	60571979
Jammu and Kashmir	58117726
Uttarakhand	53140414
Himachal Pradesh	30033289
Goa	28240159
Puducherry	20065891
Tripura	14050250

In [110]:
```
plt.figure(figsize=(15,4))
plt.bar(cvd_ind1['State/UnionTerritory'],cvd_ind1['Confirmed'],color='yellow')
plt.title("State wise Confirmed Cases",fontsize=15,color='blue')
plt.xlabel("State/UnionTerritory",fontsize=15)
plt.ylabel("Cured",fontsize=15)
plt.xticks(rotation=90)
plt.grid(False)
plt.show()
```



Observation:

- Maharashtra was having the highest no. of Cured Cases 1121491467 followed by Karnataka,Kerala,Tamil Nadu,Andhra Pradesh.
- These are first five states having highest no. of Confirmed Cases.

Create a New dataset for the states Maharashtra,Karnataka,Kerala with having highest no. of confirmed cases:

In [51]:
```
covid19India_df_Maharashtra = cvd_ind1[cvd_ind1['State/UnionTerritory']=='Maharashtra']
covid19India_df_Maharashtra
covid19India_df_Karnataka = cvd_ind1[cvd_ind1['State/UnionTerritory']=='Karnataka']
covid19India_df_Karnataka
covid19India_df_Kerala  = cvd_ind1[cvd_ind1['State/UnionTerritory']=='Kerala']
covid19India_df_Kerala
```

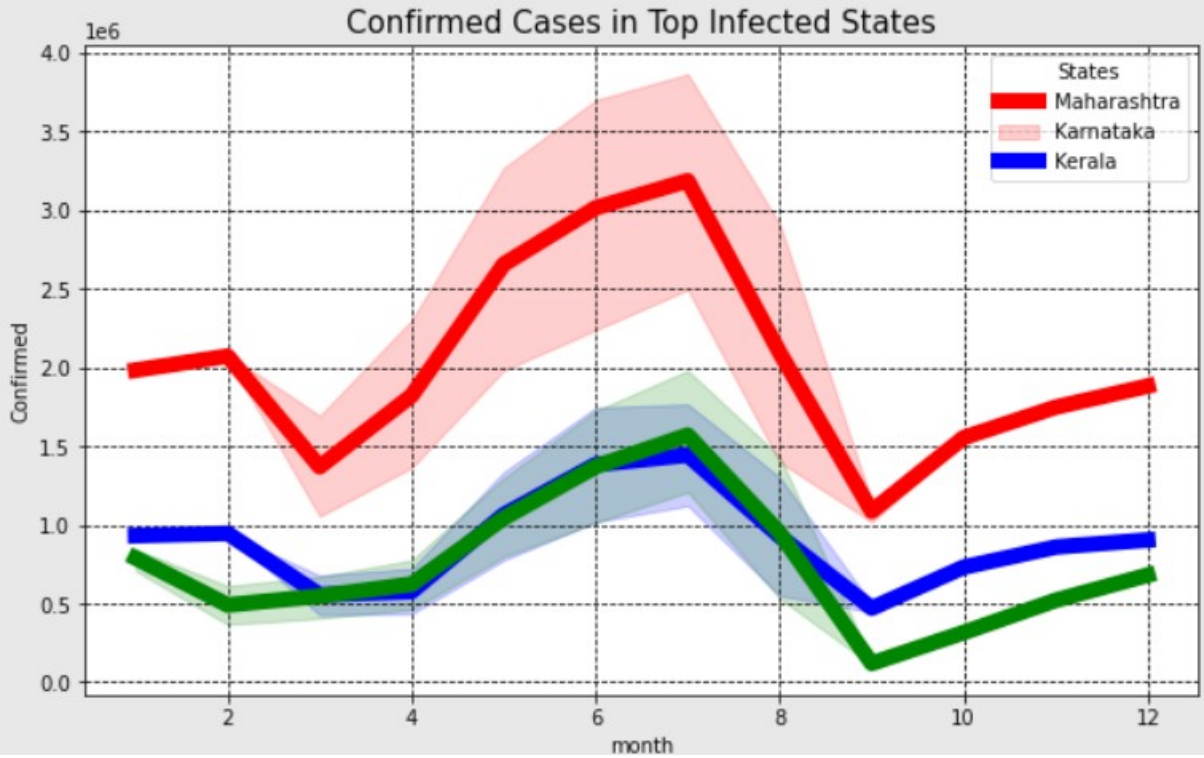
Out[51]:

	Date	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed	month	year
0	2020-01-30	Kerala	1	0	0	0	1	1	2020
1	2020-01-31	Kerala	1	0	0	0	1	1	2020
2	2020-02-01	Kerala	2	0	0	0	2	2	2020
3	2020-02-02	Kerala	3	0	0	0	3	2	2020
4	2020-02-03	Kerala	3	0	0	0	3	2	2020
...	...	...	...	...	...	...	...	...	...
17946	2021-08-07	Kerala	0	0	3317314	17515	3513551	8	2021
17982	2021-08-08	Kerala	0	0	3337579	17654	3533918	8	2021
18018	2021-08-09	Kerala	0	0	3357687	17747	3552525	8	2021
18054	2021-08-10	Kerala	0	0	3377691	17852	3565574	8	2021
18090	2021-08-11	Kerala	0	0	3396184	18004	3586693	8	2021

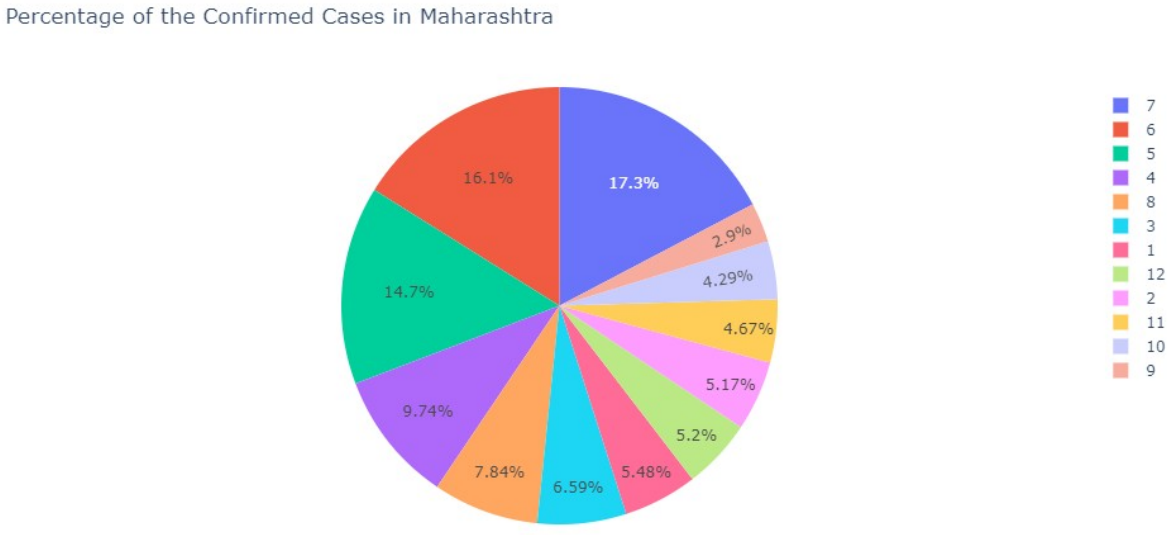
560 rows × 9 columns

In [112]:
```
plt.figure(figsize=(10,6))

#sns.lineplot(data=covid19India_df_Maharashtra)
sns.lineplot(x="month",y="Confirmed",data=covid19India_df_Maharashtra,color="red",linewidth=8)
sns.lineplot(x="month",y="Confirmed",data=covid19India_df_Karnataka,color="blue",linewidth=8)
sns.lineplot(x="month",y="Confirmed",data=covid19India_df_Kerala,color="green",linewidth=8)

plt.legend(["Maharashtra","Karnataka","Kerala"],title="States")
plt.title("Confirmed Cases in Top Infected States",size=15)
plt.grid(color="black",linestyle="--")


plt.show()
```

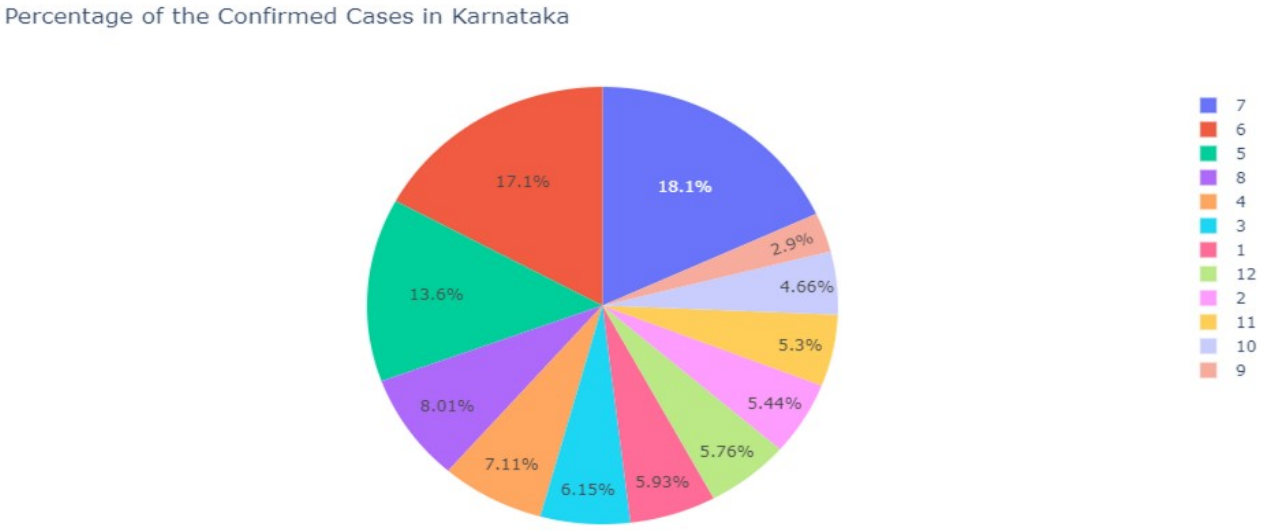
Confirmed Cases in Top Infected States:



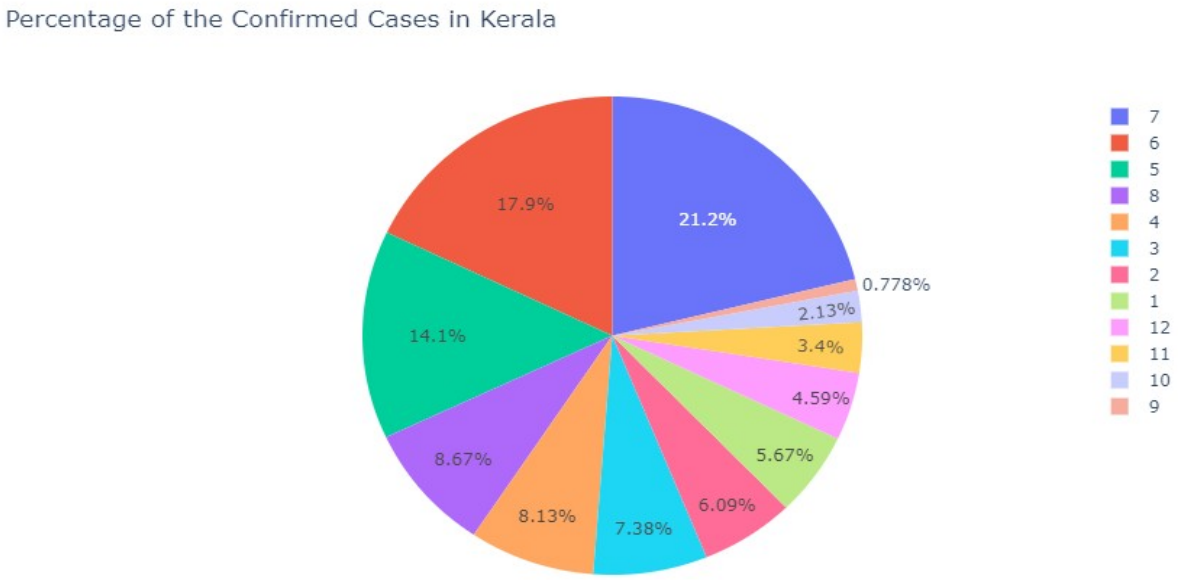

```
In [63]: 1 fig = px.pie(covid19India_df_Maharashtra, title = "Percentage of the Confirmed Cases in Maharashtra", values='Confirmed', na
         2 fig.show()
```



In [64]:
```
fig = px.pie(covid19India_df_Karnataka, title = "Percentage of the Confirmed Cases in Karnataka", values='Confirmed', names=
fig.show()
```



In [65]:
```
fig = px.pie(covid19India_df_Kerala, title = "Percentage of the Confirmed Cases in Kerala", values='Confirmed', names='month
fig.show()
```

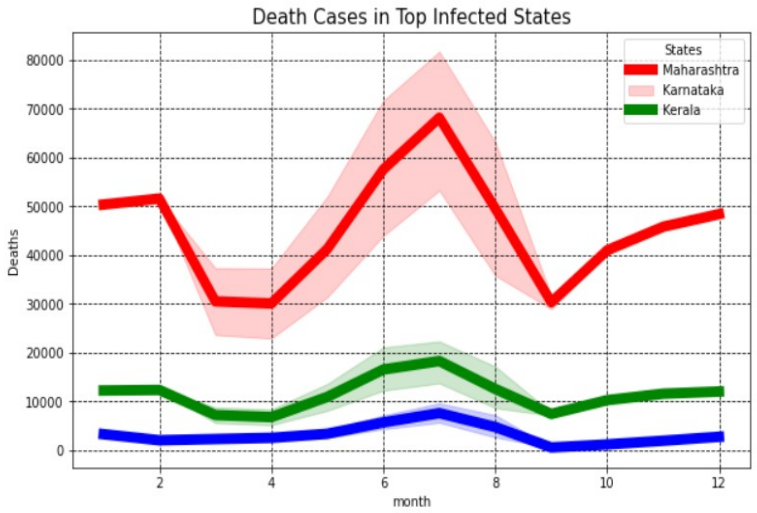


Observation:

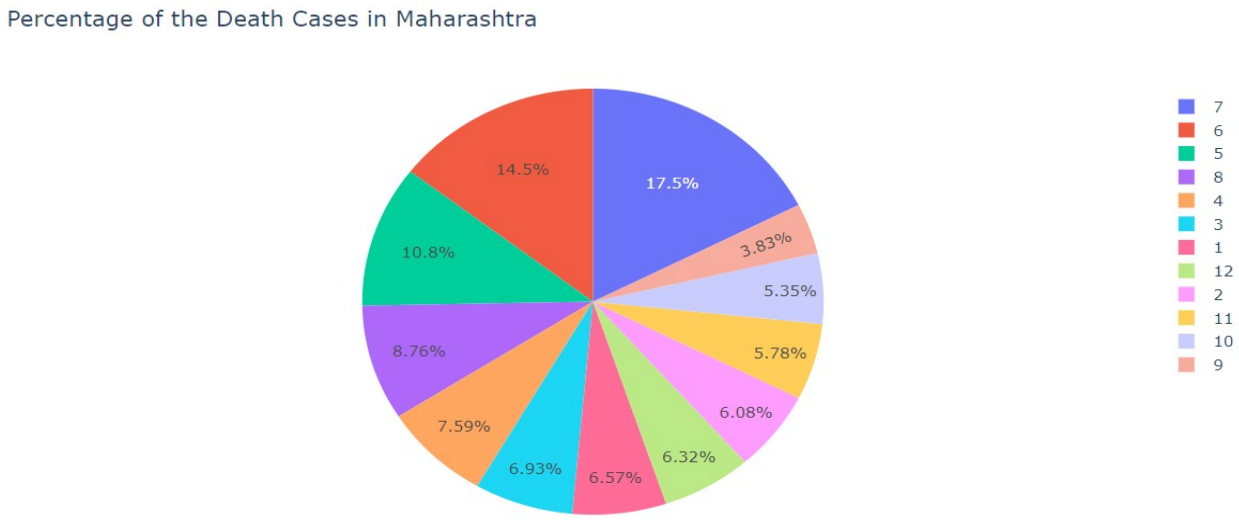
- If you see the above pie graph, in the month of 7,6 and 5 of all the three states Maharashtra,Karnataka and Kerala having high confirmed cases.
- Maharashtra state having the high confirmed cases.

Death Cases in Top Infected States:

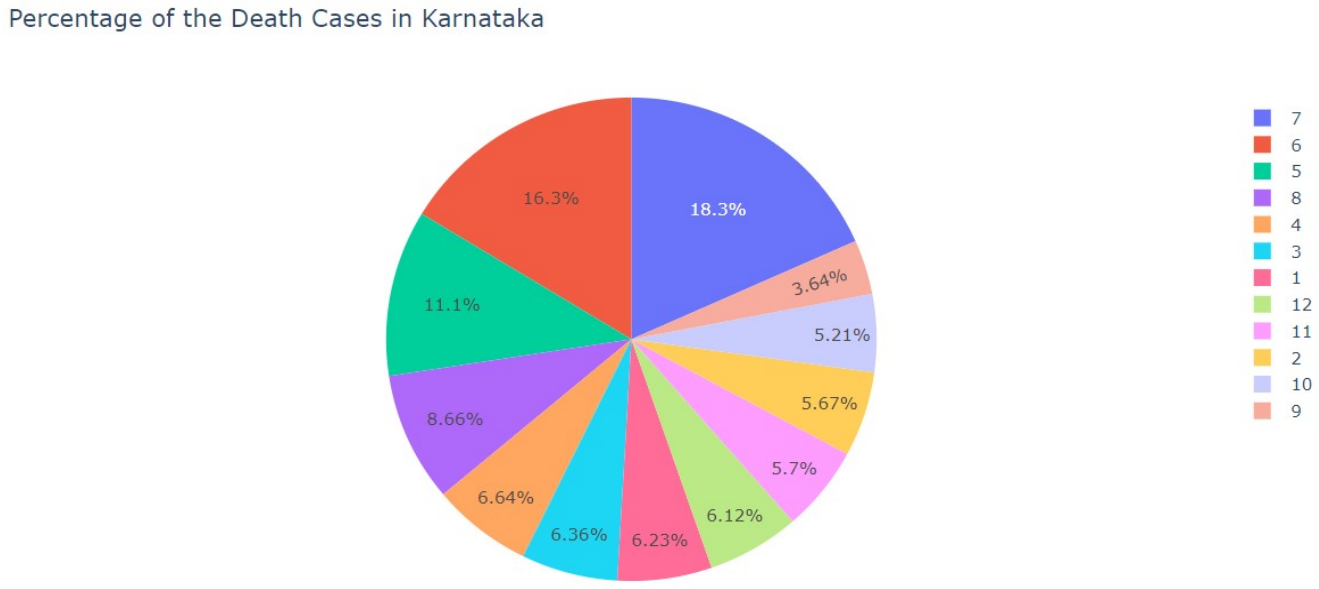
In [114]:
```
plt.figure(figsize=(10,6))

sns.lineplot(x="month",y="Deaths",data=covid19India_df_Maharashtra,color="red",linewidth=8)
sns.lineplot(x="month",y="Deaths",data=covid19India_df_Karnataka,color="green",linewidth=8)
sns.lineplot(x="month",y="Deaths",data=covid19India_df_Kerala,color="blue",linewidth=8)

plt.legend(["Maharashtra","Karnataka","Kerala"],title="States")
plt.title("Death Cases in Top Infected States",size=15)
plt.grid(color="black",linestyle="--")


plt.show()
```



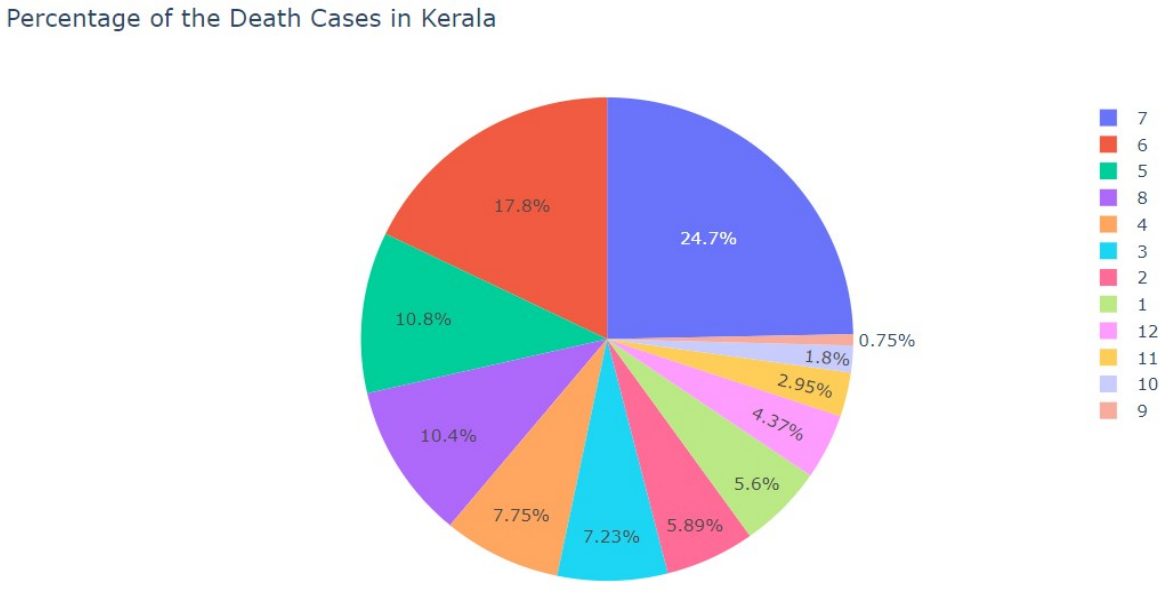
In [66]:
```
fig = px.pie(covid19India_df_Maharashtra, title = "Percentage of the Death Cases in Maharashtra", values='Deaths', names='mo
fig.show()
```



In [67]:
```
fig = px.pie(covid19India_df_Karnataka, title = "Percentage of the Death Cases in Karnataka", values='Deaths', names='month')
fig.show()
```



In [68]:
```
fig = px.pie(covid19India_df_Kerala, title = "Percentage of the Death Cases in Kerala", values='Deaths', names='month')
fig.show()
```



Observation:

- If you see the above pie graph, in the month of 7,6 and 5 of all the three states Maharashtra,Karnataka and Kerala having high Death cases.
- Maharashtra state having the high Death cases.

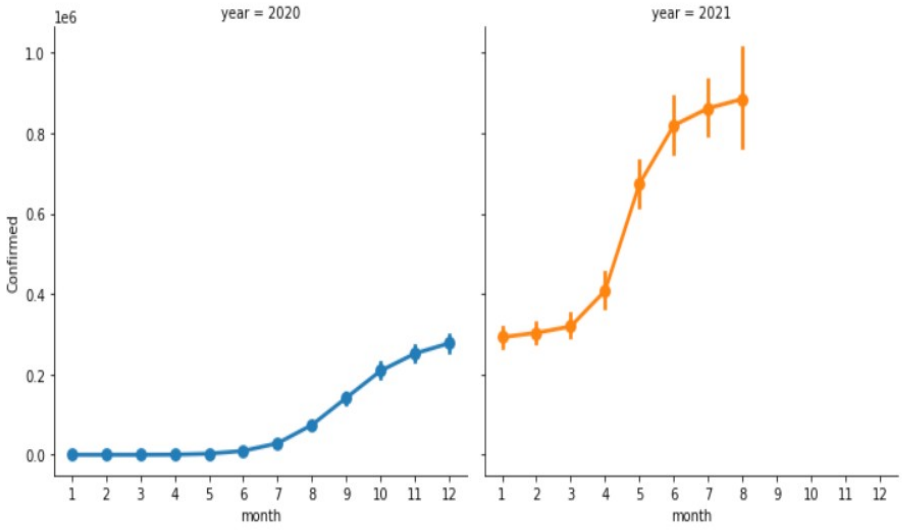
Month and Year wise Confirmed Cases Report:

```
In [69]: 1 month_year_wise_confirmed=cvd_ind1.groupby(['month','year'])['Confirmed'].sum().sort_values(ascending=False)
         2 month_year_wise_confirmed
```

Out[69]:

month	year	
7	2021	961636364
6	2021	884673464
5	2021	751927486
4	2021	440660671
3	2021	356305616
8	2021	350350755
1	2021	326469747
12	2020	307177353
2	2021	305631803
11	2020	264556412
10	2020	226770312
9	2020	149113758
8	2020	80749620
7	2020	31726501
6	2020	10558374
5	2020	2938234
4	2020	422442
3	2020	9687

```
In [70]: 1 sns.catplot(x='month',y='Confirmed',col='year',hue='year',kind='point',data=cvd_ind1)
         2 plt.show()
```

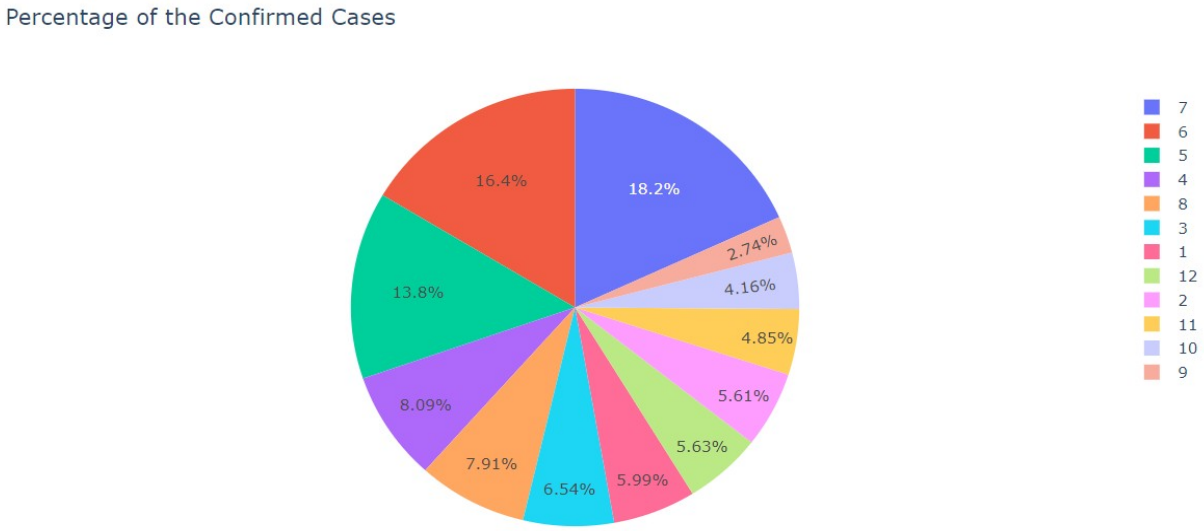


Observation:

- If you see the above plotted graph, in the year 2020 after 8th month, the confirmed cases gradually increasing.
- In the year 2021, the confirmed cases gradually increasing.

Percentage of the Confirmed Cases:

```
In [71]: 1 fig = px.pie(cvd_ind1, title = "Percentage of the Confirmed Cases", values='Confirmed', names='month')
         2 fig.show()
```

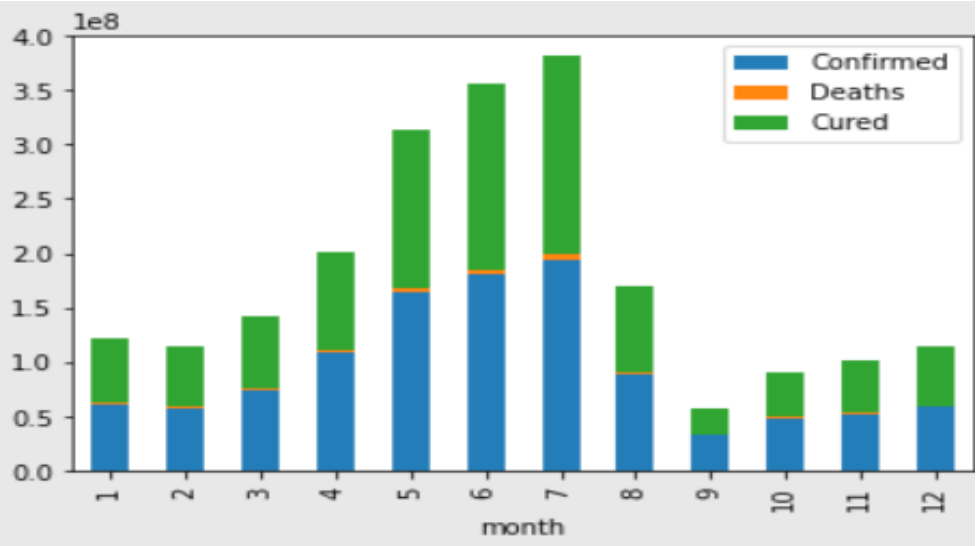


Observation:

- If you see the above pie graph, in the month of 7,6 and 5 having high Confirmed cases.
- In the month of 7 having 18.2% Confirmed cases.

In the maharashtra state, check the month wise confirmed , deaths and cured cases:

```
In [72]:
cv_mh=covid19India_df_Maharashtra.groupby('month')[['Confirmed','Deaths','Cured']].sum()
cv_mh.plot(kind="bar",stacked=True)
plt.show()
```



<u>Observation:</u>

- In the 7th month, the no. of deadth cases are higher when compared to Confirmed cases whereas the cured cases are also high.
- In the 6th month, the no. of deadth cases are higher when compared to Confirmed cases.
- In the 5th month, the no. of deadth cases are higher when compared to Confirmed cases.

Displaying the contents of Karnataka State:

In [73]:
```
1 covid19India_df_Karnataka
```

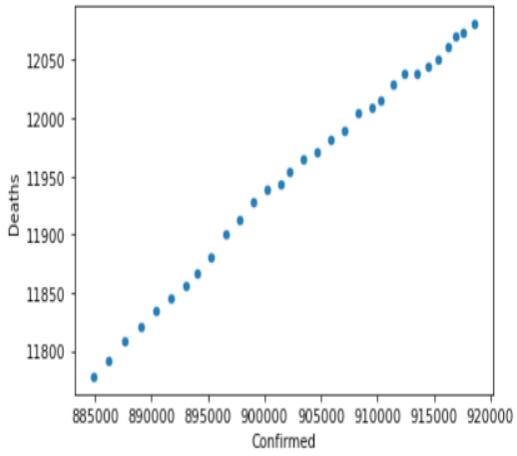
Out[73]:

	Date	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed	month	year
74	2020-03-09	Karnataka	1	0	0	0	1	3	2020
89	2020-03-10	Karnataka	4	0	0	0	4	3	2020
108	2020-03-11	Karnataka	4	0	0	0	4	3	2020
119	2020-03-12	Karnataka	4	0	0	0	4	3	2020
132	2020-03-13	Karnataka	6	0	0	1	6	3	2020
...	...	...	...	...	...	...	...	...	...
17945	2021-08-07	Karnataka	0	0	2854222	36741	2915317	8	2021
17981	2021-08-08	Karnataka	0	0	2855862	36773	2916927	8	2021
18017	2021-08-09	Karnataka	0	0	2857776	36793	2918525	8	2021
18053	2021-08-10	Karnataka	0	0	2859552	36817	2919711	8	2021
18089	2021-08-11	Karnataka	0	0	2861499	36848	2921049	8	2021

520 rows × 9 columns

In [74]:
```
1 cv_kar=covid19India_df_Karnataka[covid19India_df_Karnataka['month']==12][['Confirmed','Deaths']]
2 cv_kar.plot(kind='scatter',x='Confirmed',y='Deaths')
```

Out[74]: <AxesSubplot:xlabel='Confirmed', ylabel='Deaths'>

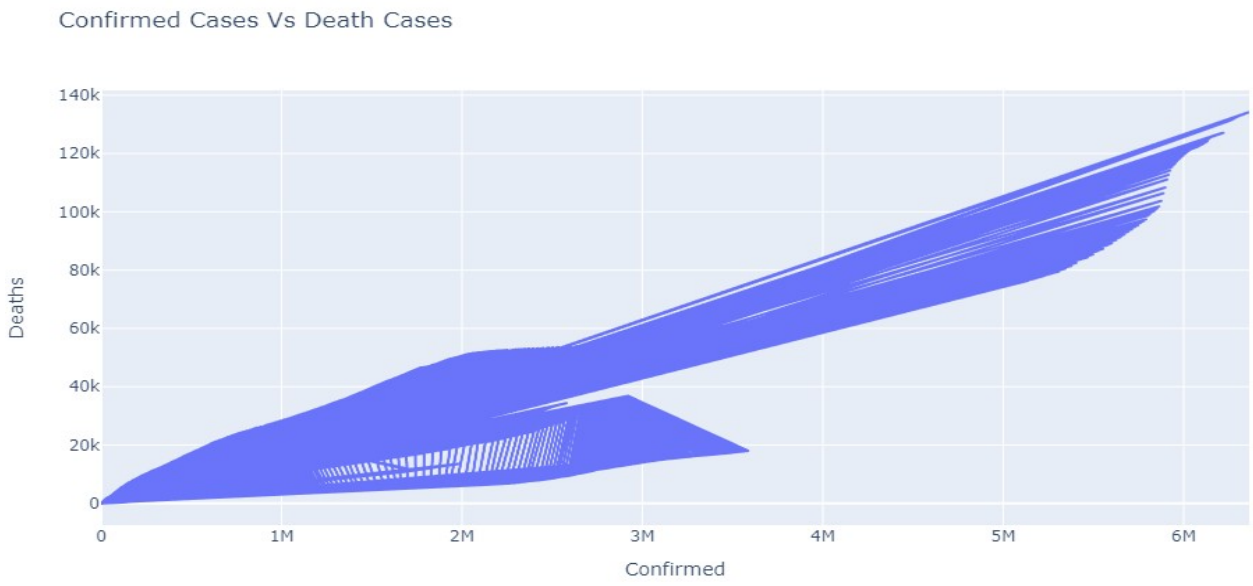


Observation:

- If you see the above graph, in the Karnataka state 12th month the no. of confirmed cases are increase and the no. of death cases are also increasing.
- you can see the line of regression.

Confirmed Cases Vs Death Cases:

In [75]:
```
fig = px.line(cvd_ind1, x="Confirmed", y="Deaths", title='Confirmed Cases Vs Death Cases')
fig.show()
```



Observation:

- If you see the above graph, the no. of confirmed cases are increasing and the no. of death cases are also increasing.
- you can see the line of regression.

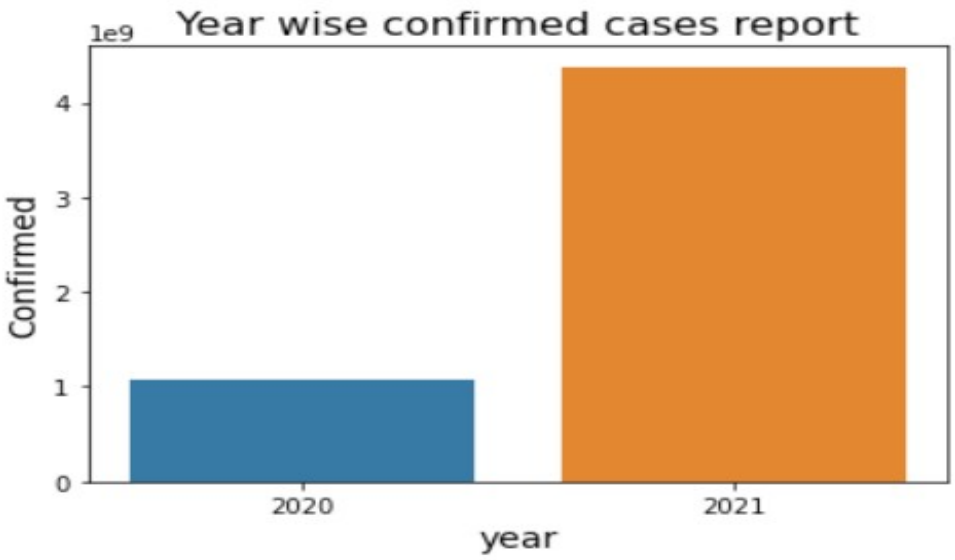
Year wise confirmed cases report:

In [76]:
```
year_wise_confirmed=cvd_ind1.groupby(['year'])['Confirmed'].sum().sort_values(ascending=False)
year_wise_confirmed = year_wise_confirmed.reset_index()
year_wise_confirmed
```

Out[76]:

	year	Confirmed
0	2021	4377655906
1	2020	1074022781

In [77]:
```
sns.barplot(x=year_wise_confirmed['year'],y=year_wise_confirmed['Confirmed'])
plt.title('Year wise confirmed cases report',size=16)
plt.xlabel('year',size=14)
plt.ylabel('Confirmed',size=14)
plt.show()
```



Observation:

- If you see the above graph, 2021 having the highest confirmed cases than the 2020.
-

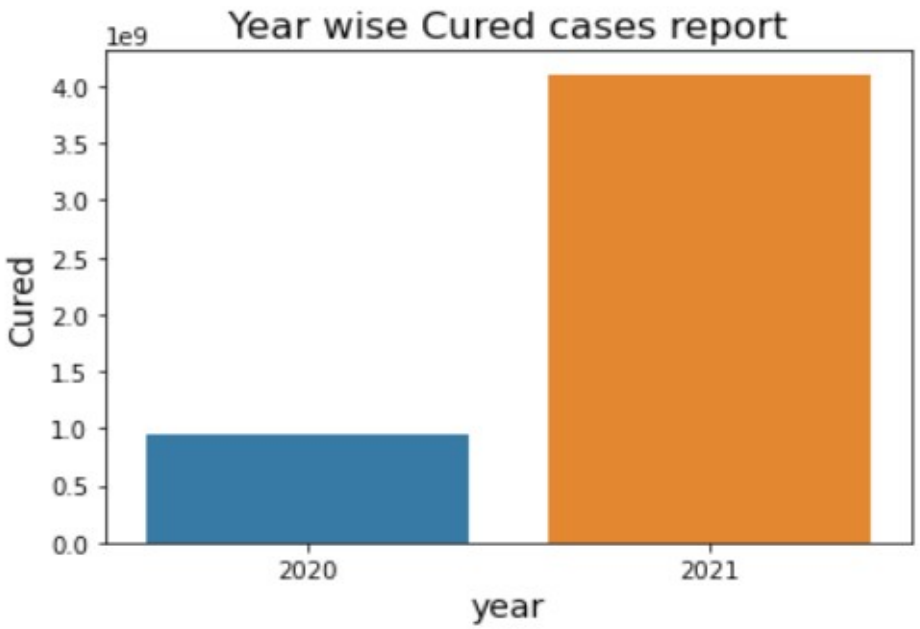
Year wise Cured cases report:

In [78]:
```
year_wise_cured=cvd_ind1.groupby(['year'])['Cured'].sum().sort_values(ascending=False)
year_wise_cured = year_wise_cured.reset_index()
year_wise_cured
```

Out[78]:

	year	Cured
0	2021	4104811257
1	2020	941314195

In [79]:
```
sns.barplot(x=year_wise_cured['year'],y=year_wise_cured['Cured'])
plt.title('Year wise Cured cases report',size=16)
plt.xlabel('year',size=14)
plt.ylabel('Cured',size=14)
plt.show()
```



Observation:

- If you see the above graph, 2021 having the highest cured cases than the 2020.

Displaying the contents of state wise records:

In [80]:
```
1 st_test_dt
```

Out[80]:

	Date	State	TotalSamples	Negative	Positive
0	2020-04-17	Andaman and Nicobar Islands	1403.0	1210	12.0
1	2020-04-24	Andaman and Nicobar Islands	2679.0	NaN	27.0
2	2020-04-27	Andaman and Nicobar Islands	2848.0	NaN	33.0
3	2020-05-01	Andaman and Nicobar Islands	3754.0	NaN	33.0
4	2020-05-16	Andaman and Nicobar Islands	6677.0	NaN	33.0
...	...	...	...	...	...
16331	2021-08-06	West Bengal	15999961.0	NaN	NaN
16332	2021-08-07	West Bengal	16045662.0	NaN	NaN
16333	2021-08-08	West Bengal	16092192.0	NaN	NaN
16334	2021-08-09	West Bengal	16122345.0	NaN	NaN
16335	2021-08-10	West Bengal	16162814.0	NaN	NaN

16335 rows × 5 columns

Create a copy of the datatset from the original dataset:

In [81]:
```
1 st_test_dt1=st_test_dt.copy()
2 st_test_dt1
```

Out[81]:

	Date	State	TotalSamples	Negative	Positive
0	2020-04-17	Andaman and Nicobar Islands	1403.0	1210	12.0
1	2020-04-24	Andaman and Nicobar Islands	2679.0	NaN	27.0
2	2020-04-27	Andaman and Nicobar Islands	2848.0	NaN	33.0
3	2020-05-01	Andaman and Nicobar Islands	3754.0	NaN	33.0
4	2020-05-16	Andaman and Nicobar Islands	6677.0	NaN	33.0
...	...	...	...	...	...
16331	2021-08-06	West Bengal	15999961.0	NaN	NaN
16332	2021-08-07	West Bengal	16045662.0	NaN	NaN
16333	2021-08-08	West Bengal	16092192.0	NaN	NaN
16334	2021-08-09	West Bengal	16122345.0	NaN	NaN
16335	2021-08-10	West Bengal	16162814.0	NaN	NaN

16335 rows × 5 columns

Fill nan values with zero:

In [82]:
```
st_test_dt1=st_test_dt1.fillna(0)
st_test_dt1
```

Out[82]:

	Date	State	TotalSamples	Negative	Positive
0	2020-04-17	Andaman and Nicobar Islands	1403.0	1210	12.0
1	2020-04-24	Andaman and Nicobar Islands	2679.0	0	27.0
2	2020-04-27	Andaman and Nicobar Islands	2848.0	0	33.0
3	2020-05-01	Andaman and Nicobar Islands	3754.0	0	33.0
4	2020-05-16	Andaman and Nicobar Islands	6677.0	0	33.0
...	...	...	...	...	...
16331	2021-08-06	West Bengal	15999961.0	0	0.0
16332	2021-08-07	West Bengal	16045662.0	0	0.0
16333	2021-08-08	West Bengal	16092192.0	0	0.0
16334	2021-08-09	West Bengal	16122345.0	0	0.0
16335	2021-08-10	West Bengal	16162814.0	0	0.0

16335 rows × 5 columns

To view the Satistical Information about the data:

In [83]:
```
st_test_dt1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 16335 entries, 0 to 16335
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Date          16335 non-null  object 
 1   State         16335 non-null  object 
 2   TotalSamples  16335 non-null  float64
 3   Negative      16335 non-null  object 
 4   Positive      16335 non-null  float64
dtypes: float64(2), object(3)
memory usage: 765.7+ KB
```

Fill the space with zero:

In [84]:
```
st_test_dt1['Negative']=st_test_dt1.Negative.mask(cvd_ind1.Cured==' ',0)
st_test_dt1
```

Out[84]:

	Date	State	TotalSamples	Negative	Positive
0	2020-04-17	Andaman and Nicobar Islands	1403.0	1210	12.0
1	2020-04-24	Andaman and Nicobar Islands	2679.0	0	27.0
2	2020-04-27	Andaman and Nicobar Islands	2848.0	0	33.0
3	2020-05-01	Andaman and Nicobar Islands	3754.0	0	33.0
4	2020-05-16	Andaman and Nicobar Islands	6677.0	0	33.0
...	...	...	...	...	...
16331	2021-08-06	West Bengal	15999961.0	0	0.0
16332	2021-08-07	West Bengal	16045662.0	0	0.0
16333	2021-08-08	West Bengal	16092192.0	0	0.0
16334	2021-08-09	West Bengal	16122345.0	0	0.0
16335	2021-08-10	West Bengal	16162814.0	0	0.0

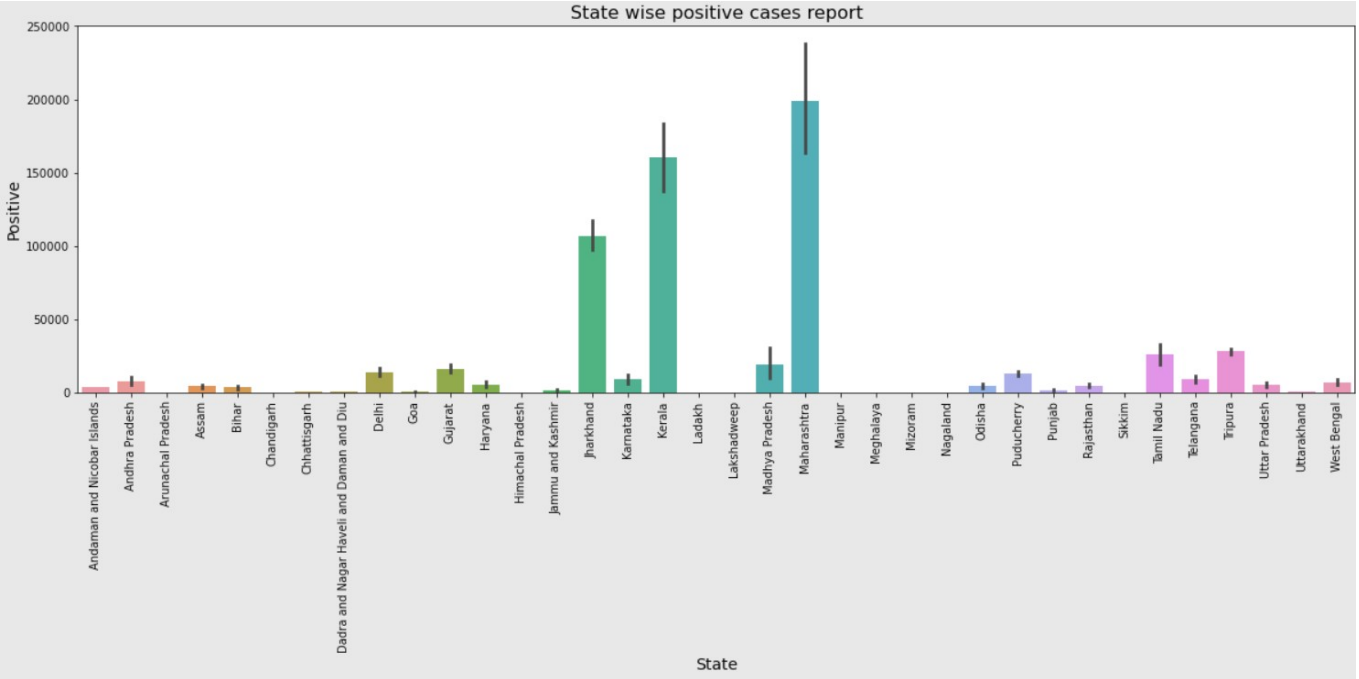
16335 rows × 5 columns

State wise positive cases report:

In [85]:
```
st_test_dt1.groupby(['State'])['Positive'].sum().sort_values(ascending=False)
```

Out[85]:
```
State
Maharashtra                                96901583.0
Kerala                                     79723175.0
Jharkhand                                  51696495.0
Tamil Nadu                                 12772604.0
Tripura                                    12630766.0
Madhya Pradesh                              9584080.0
Gujarat                                     8009517.0
Delhi                                       6848173.0
Puducherry                                  6287323.0
Karnataka                                   4701197.0
Andhra Pradesh                              3859260.0
Telangana                                   3855373.0
West Bengal                                 3487431.0
Haryana                                     2830153.0
Uttar Pradesh                               2743693.0
Rajasthan                                   2445076.0
Odisha                                      2214458.0
Assam                                       2065991.0
Bihar                                       1859345.0
Andaman and Nicobar Islands                 1763591.0
Jammu and Kashmir                            977615.0
Punjab                                       960287.0
Chhattisgarh                                 467857.0
Uttarakhand                                  350257.0
Goa                                          266181.0
Dadra and Nagar Haveli and Daman and Diu     169010.0
Himachal Pradesh                             119494.0
Manipur                                      101501.0
```

In [86]:
```
plt.figure(figsize=(20,6))
sns.barplot(x=st_test_dt1['State'],y=st_test_dt1['Positive'])
plt.title('State wise positive cases report',size=16)
plt.xticks(rotation=90)
plt.xlabel('State',size=14)
plt.ylabel('Positive',size=14)
plt.show()
```



Observation:

Highest no. of positive cases found in the following states

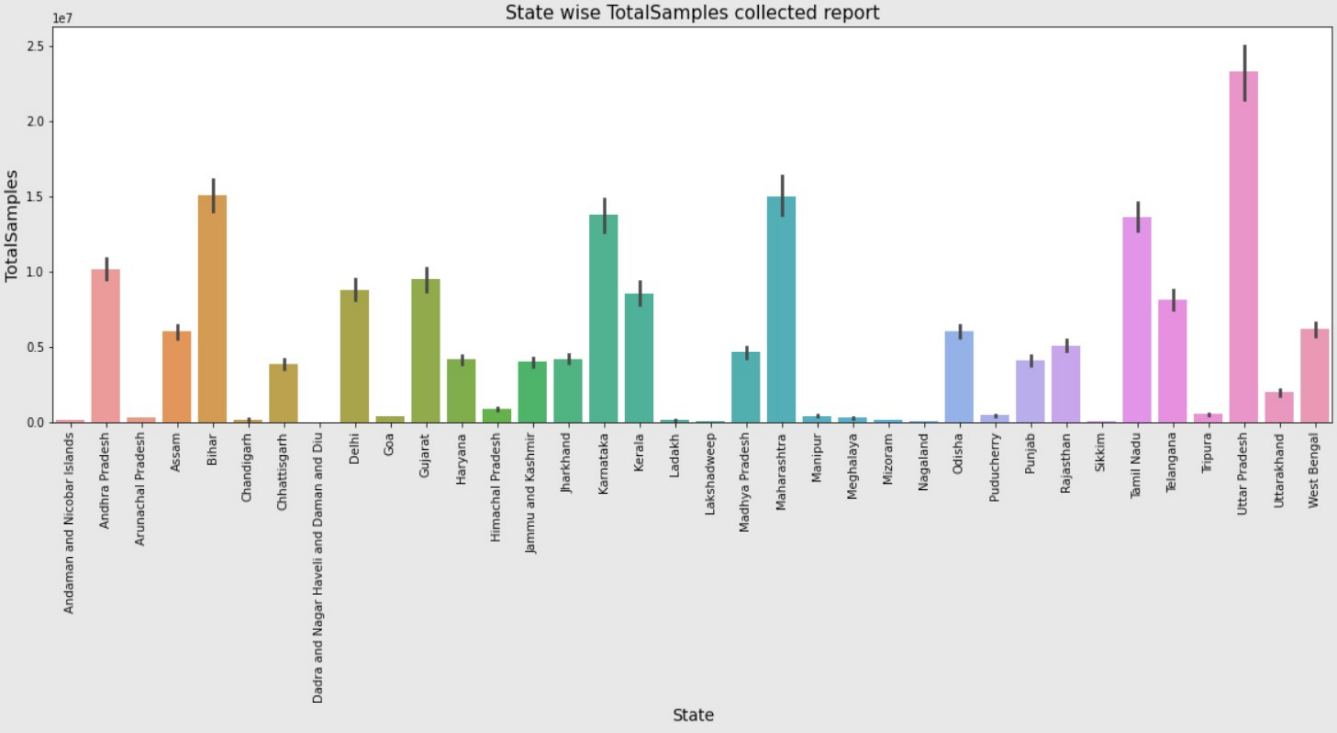
- Maharashtra - 96901583.0
- Kerala - 79723175.0
- Jharkhand - 51696495.0

State wise TotalSamples collected report:

```
In [87]: 1 st_test_dt.groupby(['State'])['TotalSamples'].sum().sort_values(ascending=False)

Out[87]: State
         Uttar Pradesh                1.138818e+10
         Bihar                        7.392796e+09
         Maharashtra                  7.334574e+09
         Karnataka                    6.773248e+09
         Tamil Nadu                   6.711189e+09
         Andhra Pradesh               4.967773e+09
         Gujarat                      4.623914e+09
         Delhi                        4.310596e+09
         Kerala                       4.269006e+09
         Telangana                    3.422977e+09
         West Bengal                  3.051636e+09
         Odisha                       2.965651e+09
         Assam                        2.853509e+09
         Rajasthan                    2.520540e+09
         Madhya Pradesh               2.298458e+09
         Haryana                      2.056736e+09
         Jharkhand                    2.053512e+09
         Punjab                       2.026576e+09
         Jammu and Kashmir            1.960284e+09
         Chhattisgarh                 1.863129e+09
         Uttarakhand                  9.792698e+08
         Himachal Pradesh             4.271429e+08
         Tripura                      2.467333e+08
```

In [92]:

```
plt.figure(figsize=(20,6))
sns.barplot(x=st_test_dt1['State'],y=st_test_dt1['TotalSamples'])
plt.title('State wise TotalSamples collected report',size=16)
plt.xticks(rotation=90)
plt.xlabel('State',size=14)
plt.ylabel('TotalSamples',size=14)
plt.show()
```



Observation:

Highest no. of samples collected in the following states

- Uttar Pradesh - 1.138818
- Bihar - 7.392796
- Maharashtra - 7.334574

Create a copy of the datatset from the original dataset:

In [93]:
```
vacc_st_wise1=vacc_st_wise.copy()
vacc_st_wise1
```

Out[93]:

	Updated On	State	Total Doses Administered	Sessions	Sites	First Dose Administered	Second Dose Administered	Male (Doses Administered)	Female (Doses Administered)	Transgender (Doses Administered)	...	18-44 Years (Doses Administered)	45-60 (Adminis
0	16/01/2021	India	48276.0	3455.0	2957.0	48276.0	0.0	NaN	NaN	NaN	...	NaN	
1	17/01/2021	India	58604.0	8532.0	4954.0	58604.0	0.0	NaN	NaN	NaN	...	NaN	
2	18/01/2021	India	99449.0	13611.0	6583.0	99449.0	0.0	NaN	NaN	NaN	...	NaN	
3	19/01/2021	India	195525.0	17855.0	7951.0	195525.0	0.0	NaN	NaN	NaN	...	NaN	
4	20/01/2021	India	251280.0	25472.0	10504.0	251280.0	0.0	NaN	NaN	NaN	...	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	
7840	11/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
7841	12/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
7842	13/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
7843	14/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
7844	15/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	

7845 rows × 24 columns

Fill nan values with zero:

In [94]:
```
vacc_st_wise1.fillna(0)
```

Out[94]:

	Updated On	State	Total Doses Administered	Sessions	Sites	First Dose Administered	Second Dose Administered	Male (Doses Administered)	Female (Doses Administered)	Transgender (Doses Administered)	...	18-44 Years (Doses Administered)	45-60 (Adminis
0	16/01/2021	India	48276.0	3455.0	2957.0	48276.0	0.0	0.0	0.0	0.0	...	0.0	
1	17/01/2021	India	58604.0	8532.0	4954.0	58604.0	0.0	0.0	0.0	0.0	...	0.0	
2	18/01/2021	India	99449.0	13611.0	6583.0	99449.0	0.0	0.0	0.0	0.0	...	0.0	
3	19/01/2021	India	195525.0	17855.0	7951.0	195525.0	0.0	0.0	0.0	0.0	...	0.0	
4	20/01/2021	India	251280.0	25472.0	10504.0	251280.0	0.0	0.0	0.0	0.0	...	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
7840	11/08/2021	West Bengal	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	
7841	12/08/2021	West Bengal	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	
7842	13/08/2021	West Bengal	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	
7843	14/08/2021	West Bengal	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	
7844	15/08/2021	West Bengal	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	

7845 rows × 24 columns

State wise Male Individuals Vaccinated:

```
In [95]: 1 vacc_st_wise1.groupby(['State'])['Male(Individuals Vaccinated)'].sum().sort_values(ascending=False)
```

Out[95]:

State	
India	7.138699e+09
Andaman and Nicobar Islands	0.000000e+00
Puducherry	0.000000e+00
Maharashtra	0.000000e+00
Manipur	0.000000e+00
Meghalaya	0.000000e+00
Mizoram	0.000000e+00
Nagaland	0.000000e+00
Odisha	0.000000e+00
Punjab	0.000000e+00
Lakshadweep	0.000000e+00
Rajasthan	0.000000e+00
Sikkim	0.000000e+00
Tamil Nadu	0.000000e+00
Telangana	0.000000e+00
Tripura	0.000000e+00
Uttar Pradesh	0.000000e+00
Uttarakhand	0.000000e+00
Madhya Pradesh	0.000000e+00
Ladakh	0.000000e+00
Andhra Pradesh	0.000000e+00
Delhi	0.000000e+00
Arunachal Pradesh	0.000000e+00
Assam	0.000000e+00
Bihar	0.000000e+00
Chandigarh	0.000000e+00
Chhattisgarh	0.000000e+00
Dadra and Nagar Haveli and Daman and Diu	0.000000e+00

State wise Female Individuals Vaccinated:

```
In [96]: 1 vacc_st_wise1.groupby(['State'])['Female(Individuals Vaccinated)'].sum().sort_values(ascending=False)
```

Out[96]:

State	
India	6.321629e+09
Andaman and Nicobar Islands	0.000000e+00
Puducherry	0.000000e+00
Maharashtra	0.000000e+00
Manipur	0.000000e+00
Meghalaya	0.000000e+00
Mizoram	0.000000e+00
Nagaland	0.000000e+00
Odisha	0.000000e+00
Punjab	0.000000e+00
Lakshadweep	0.000000e+00
Rajasthan	0.000000e+00
Sikkim	0.000000e+00
Tamil Nadu	0.000000e+00
Telangana	0.000000e+00
Tripura	0.000000e+00
Uttar Pradesh	0.000000e+00
Uttarakhand	0.000000e+00
Madhya Pradesh	0.000000e+00
Ladakh	0.000000e+00
Andhra Pradesh	0.000000e+00
Delhi	0.000000e+00
Arunachal Pradesh	0.000000e+00
Assam	0.000000e+00
Bihar	0.000000e+00
Chandigarh	0.000000e+00
Chhattisgarh	0.000000e+00
Dadra and Nagar Haveli and Daman and Diu	0.000000e+00
Goa	0.000000e+00
Kerala	0.000000e+00
Gujarat	0.000000e+00
Haryana	0.000000e+00

State wise Total Individuals Vaccinated:

In [97]:
```
tot_ind_vacc=vacc_st_wise1.groupby(['State'])['Total Individuals Vaccinated'].sum().sort_values(ascending=False)
tot_ind_vacc
```

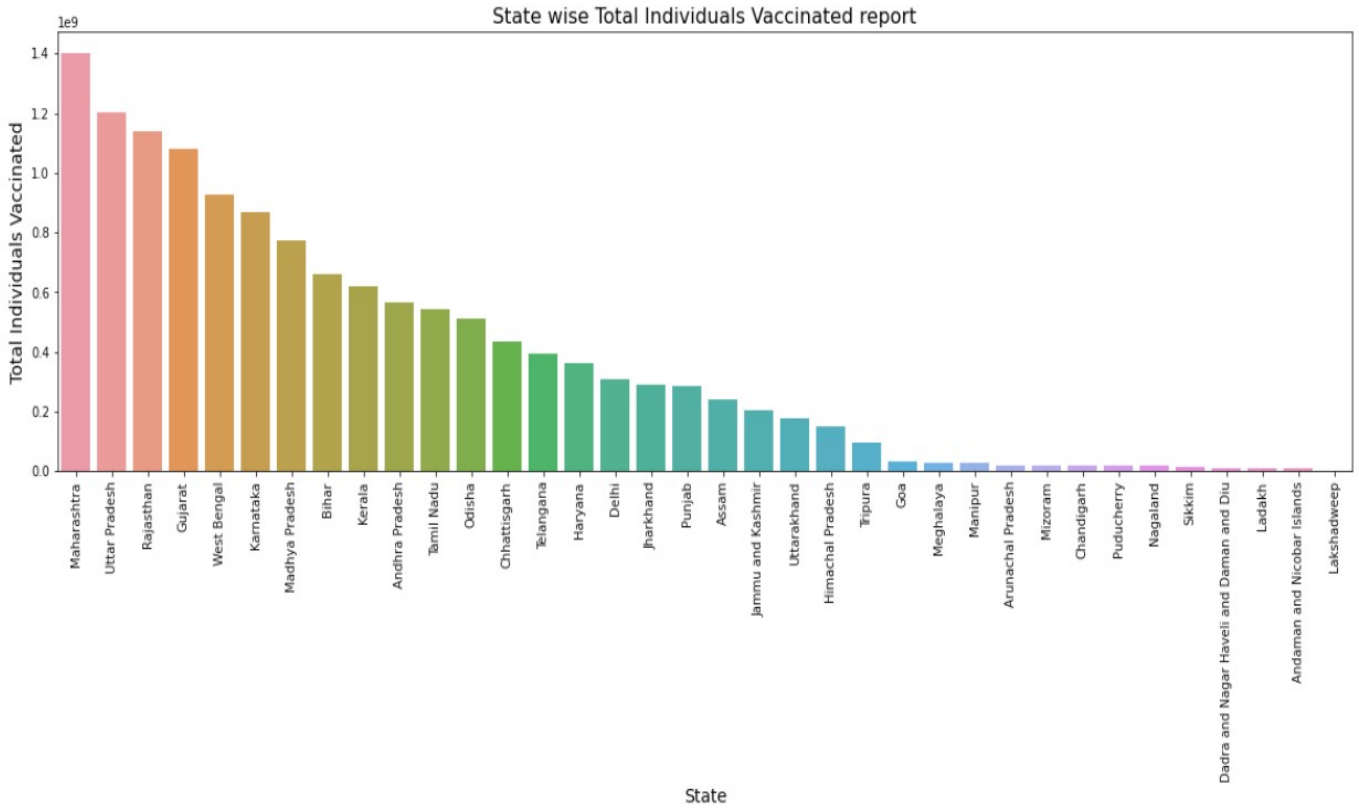
Out[97]:
```
State
India                                1.346231e+10
Maharashtra                          1.403075e+09
Uttar Pradesh                        1.200575e+09
Rajasthan                            1.141163e+09
Gujarat                              1.078261e+09
West Bengal                          9.250227e+08
Karnataka                            8.685235e+08
Madhya Pradesh                       7.718640e+08
Bihar                                6.608479e+08
Kerala                               6.208252e+08
Andhra Pradesh                       5.645911e+08
Tamil Nadu                           5.437461e+08
Odisha                               5.105198e+08
Chhattisgarh                         4.353092e+08
Telangana                            3.933718e+08
Haryana                              3.637547e+08
Delhi                                3.057372e+08
Jharkhand                            2.891507e+08
Punjab                               2.875444e+08
Assam                                2.397691e+08
Jammu and Kashmir                    2.037598e+08
Uttarakhand                          1.747382e+08
Himachal Pradesh                     1.504916e+08
Tripura                              9.379244e+07
Goa                                  3.211478e+07
Meghalaya                            2.720527e+07
Manipur                              2.665426e+07
Arunachal Pradesh                    2.108156e+07
Mizoram                              2.057245e+07
Chandigarh                           1.973150e+07
Puducherry                           1.776065e+07
```

In [98]:
```
tot_ind_vacc.drop('India',inplace=True)
tot_ind_vacc=tot_ind_vacc.reset_index()
tot_ind_vacc
```

Out[98]:

	State	Total Individuals Vaccinated
0	Maharashtra	1.403075e+09
1	Uttar Pradesh	1.200575e+09
2	Rajasthan	1.141163e+09
3	Gujarat	1.078261e+09
4	West Bengal	9.250227e+08
5	Karnataka	8.685235e+08
6	Madhya Pradesh	7.718640e+08
7	Bihar	6.608479e+08
8	Kerala	6.208252e+08
9	Andhra Pradesh	5.645911e+08
10	Tamil Nadu	5.437461e+08
11	Odisha	5.105198e+08
12	Chhattisgarh	4.353092e+08
13	Telangana	3.933718e+08
14	Haryana	3.637547e+08
15	Delhi	3.057372e+08
16	Jharkhand	2.891507e+08

In [99]:
```
plt.figure(figsize=(20,6))
sns.barplot(x=tot_ind_vacc['State'],y=tot_ind_vacc['Total Individuals Vaccinated'])
plt.title('State wise Total Individuals Vaccinated report',size=16)
plt.xticks(rotation=90)
plt.xlabel('State',size=14)
plt.ylabel('Total Individuals Vaccinated',size=14)
plt.show()
```



Observation:

State wise Total Individuals Vaccinated report, Following are the first five states having the high number of Individuals Vaccinated

- Maharashtra - 1.403075e+09
- Uttar Pradesh - 1.200575e+09
- Rajasthan - 1.141163e+09
- Gujarat - 1.078261e+09
- West Bengal - 9.250227e+08
- Karnataka - 8.685235e+08

Added two columns from the Updated On column of the original dataset:

```
In [100]: 1 vacc_st_wise1['st_month']=pd.to_datetime(vacc_st_wise1['Updated On']).dt.month
          2 vacc_st_wise1['st_year']=pd.to_datetime(vacc_st_wise1['Updated On']).dt.year
```

Displaying the contents of statewise record:

```
In [101]: 1 vacc_st_wise1
```

Out[101]:

	Updated On	State	Total Doses Administered	Sessions	Sites	First Dose Administered	Second Dose Administered	Male (Doses Administered)	Female (Doses Administered)	Transgender (Doses Administered)	...	60+ Years (Doses Administered)	Years(I V
0	16/01/2021	India	48276.0	3455.0	2957.0	48276.0	0.0	NaN	NaN	NaN	...	NaN	
1	17/01/2021	India	58604.0	8532.0	4954.0	58604.0	0.0	NaN	NaN	NaN	...	NaN	
2	18/01/2021	India	99449.0	13611.0	6583.0	99449.0	0.0	NaN	NaN	NaN	...	NaN	
3	19/01/2021	India	195525.0	17855.0	7951.0	195525.0	0.0	NaN	NaN	NaN	...	NaN	
4	20/01/2021	India	251280.0	25472.0	10504.0	251280.0	0.0	NaN	NaN	NaN	...	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	
7840	11/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
7841	12/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
7842	13/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
7843	14/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
7844	15/08/2021	West Bengal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	

7845 rows × 26 columns

Month wise Total Individuals Vaccinated:

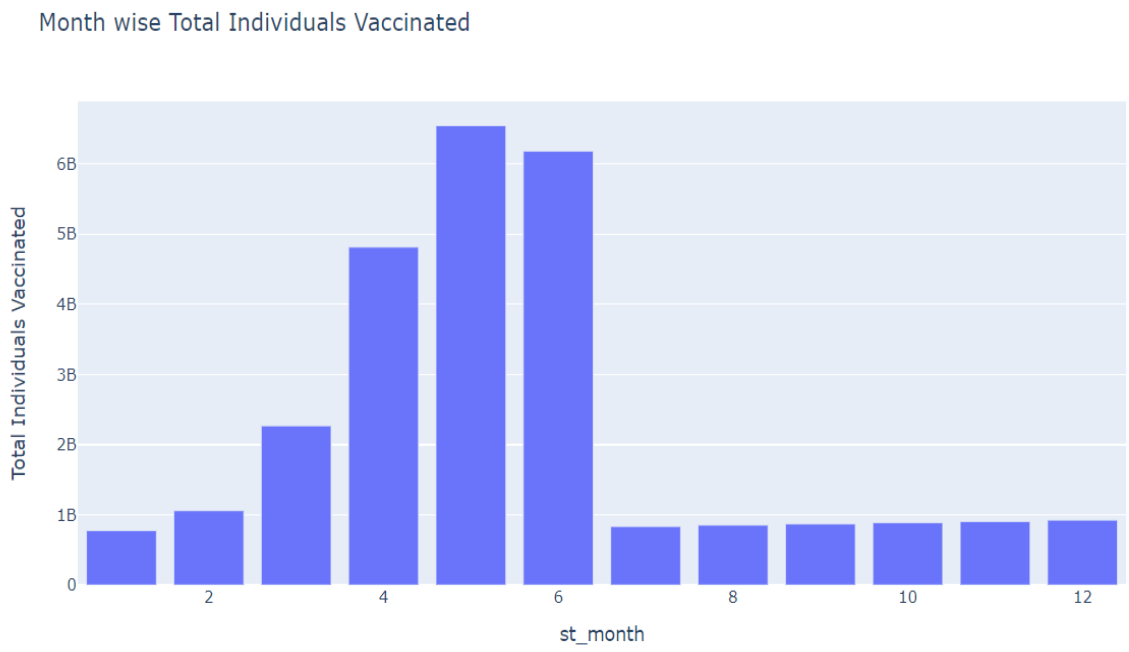
In [102]:
```
monthwise_vacc=vacc_st_wise1.groupby(['st_month'])['Total Individuals Vaccinated'].sum().sort_values(ascending=False)
```

In [103]:
```
monthwise_vacc=monthwise_vacc.reset_index()
monthwise_vacc
```

Out[103]:

	st_month	Total Individuals Vaccinated
0	5	6.544899e+09
1	6	6.182028e+09
2	4	4.814328e+09
3	3	2.268107e+09
4	2	1.059021e+09
5	12	9.254012e+08
6	11	9.049878e+08
7	10	8.885708e+08
8	9	8.702361e+08
9	8	8.527892e+08
10	7	8.330908e+08
11	1	7.752146e+08

In [104]:
```
fig = px.bar(monthwise_vacc, title= "Month wise Total Individuals Vaccinated", x='st_month', y='Total Individuals Vaccinated
fig.show()
```



Observation:

- In the month of 5 having highest Total Individuals Vaccinated.
- Followed by, In the month of 6 having second highest Total Individuals Vaccinated.

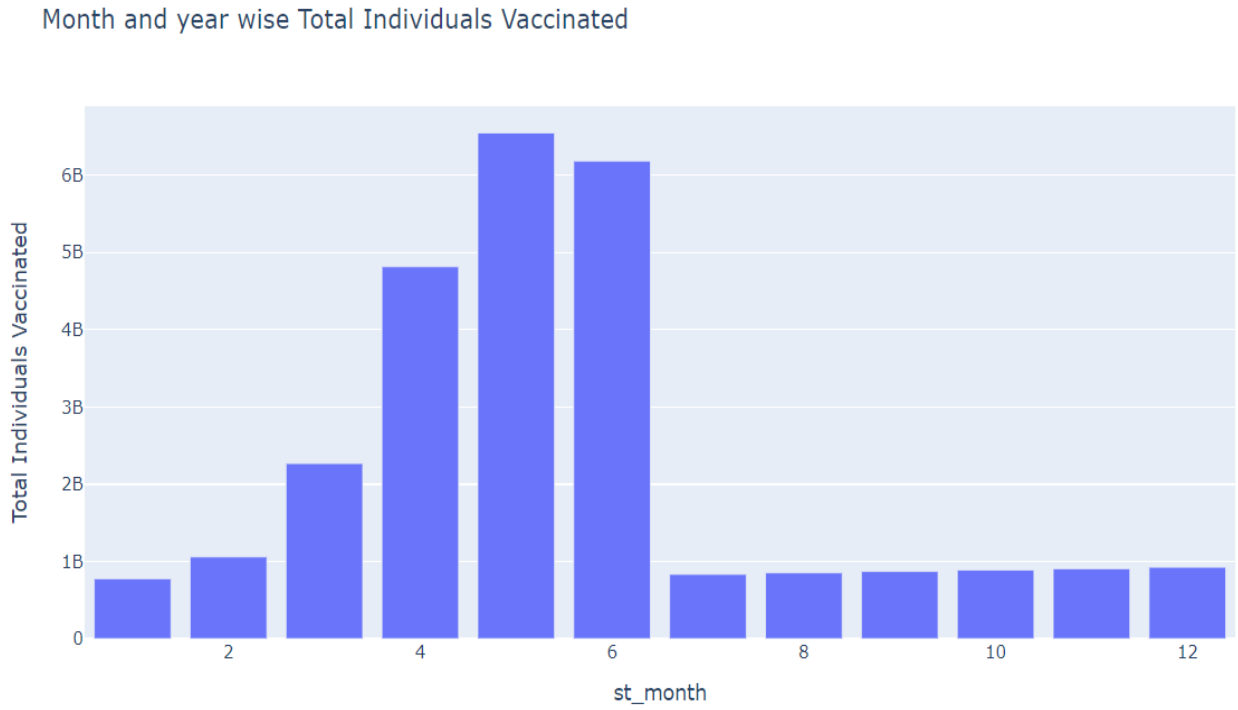
Month and year wise Total Individuals Vaccinated:

In [105]:
```
month_year_wise_confirmed=vacc_st_wise1.groupby(['st_month','st_year'])['Total Individuals Vaccinated'].sum().sort_values(as
month_year_wise_confirmed=month_year_wise_confirmed.reset_index()
month_year_wise_confirmed
```

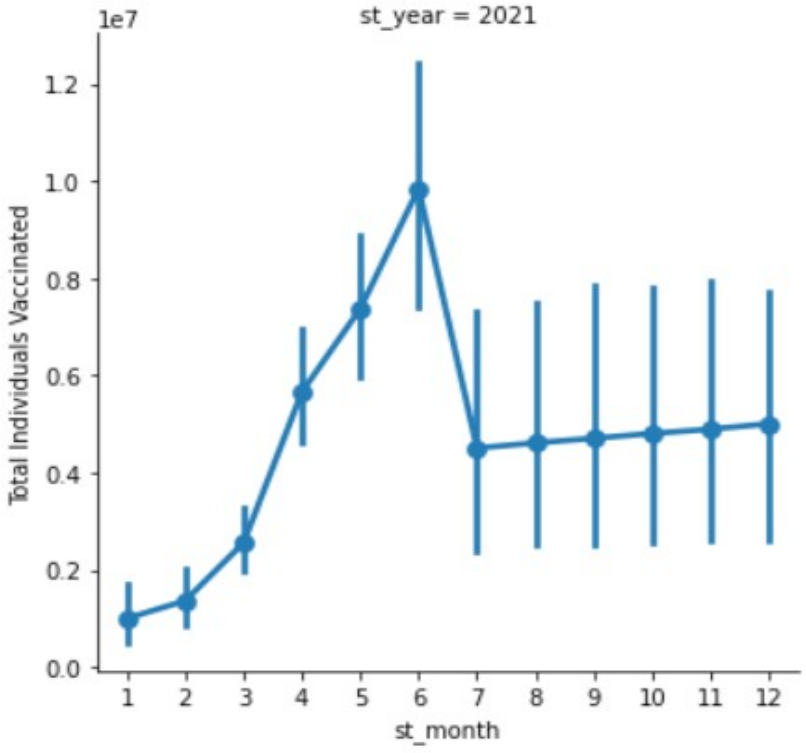
Out[105]:

	st_month	st_year	Total Individuals Vaccinated
0	5	2021	6.544899e+09
1	6	2021	6.182028e+09
2	4	2021	4.814328e+09
3	3	2021	2.268107e+09
4	2	2021	1.059021e+09
5	12	2021	9.254012e+08
6	11	2021	9.049878e+08
7	10	2021	8.885708e+08
8	9	2021	8.702361e+08
9	8	2021	8.527892e+08
10	7	2021	8.330908e+08
11	1	2021	7.752146e+08

In [106]:
```
fig = px.bar(month_year_wise_confirmed, title= "Month and year wise Total Individuals Vaccinated", x='st_month', y='Total In
fig.show()
```



In [107]:
```
1 sns.catplot(x='st_month',y='Total Individuals Vaccinated',col='st_year',hue='st_year',kind='point',data=vacc_st_wise1)
2 plt.show()
```



Observation:

- In the month of 5 having highest Total Individuals Vaccinated in the year 2021.
- Followed by, In the month of 6 having second highest Total Individuals Vaccinated in the year 2021.

Chapter 4

Analysis of the Result

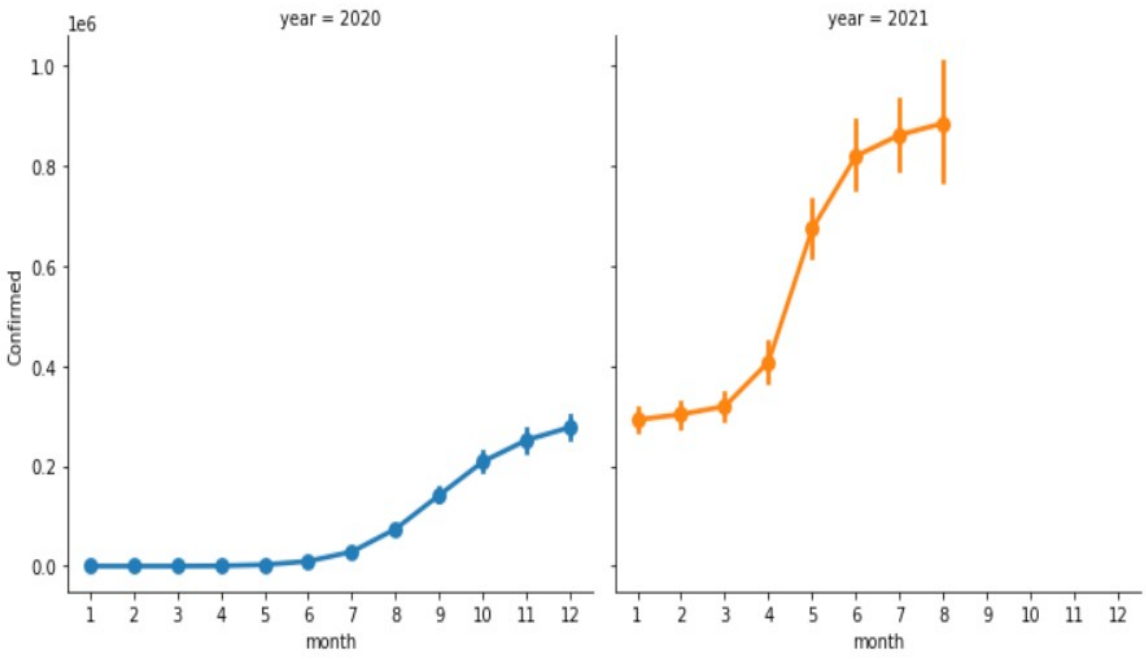
Month and year wise Total Individuals Vaccinated:

In [111]:
```
1 month_year_wise_confirmed
```

Out[111]:

	st_month	st_year	Total Individuals Vaccinated
0	5	2021	6.544899e+09
1	6	2021	6.182028e+09
2	4	2021	4.814328e+09
3	3	2021	2.268107e+09
4	2	2021	1.059021e+09
5	12	2021	9.254012e+08
6	11	2021	9.049878e+08
7	10	2021	8.885708e+08
8	9	2021	8.702361e+08
9	8	2021	8.527892e+08
10	7	2021	8.330908e+08
11	1	2021	7.752146e+08

In [109]:
```
1 sns.catplot(x='month',y='Confirmed',col='year',hue='year',kind='point',data=cvd_ind1) # from previous data set
2 plt.show()
```



State wise high no. of Confirmed cases:

- Maharashtra - 1121491467
- Karnataka - 485970693
- Kerala - 458906023
- Tamil Nadu - 431928644
- Andhra Pradesh - 392432753
- Uttar Pradesh - 312625843

State wise high no. of death cases:

- Maharashtra - 23737432
- Karnataka - 6053762
- Tamil Nadu - 5916658
- Delhi - 4943294
- Uttar Pradesh - 4143450

State wise high no. of cured cases:

- Maharashtra - 1018765039
- Karnataka - 441844360
- Kerala - 420174235
- Tamil Nadu - 404095807
- Andhra Pradesh - 370426530

Chapter 5

Conclusion

- Highest no. of confirmed cases is in Maharashtra and Highest no. of death cases is in Maharashtra and Highest no. of cured cases is in Maharashtra whereas total individuals vaccinated is also high in Maharashtra
- By this we can conclude, If we vaccinated and take the necessary precautions like wearing mask and social distance, we can able overcome this pandemic.
- If you see, Tamil Nadu have been placed third position in the highest no. of death cases because the total individuals vaccinated is low whereas confirmed cases are also so high.
- Over 70% of people use a mask only while stepping out of home to visit crowded marketplaces or at work.
- Total Vaccination count: 2,13,91,49,934 (As on 07-Sep-2022)
- As a Individuals, we also need to cooperate to the government, to take the necessary precautions like vaccination, wearing mask and social distance.
- These necessary actions will help us to lead our live.

Here is access to the data set & code from my GitHub page:

https://github.com/Adaikkkappan/Covid-Dataset-EDA

References

1. COVID-19 in India from Kaggle Created by **SRK AND 1 Devakumar K. P.**

2. COVID-19 in India from Kaggle Created by **SONIA SHARMA.**

3. **WORLD HEALTH ORGANIZATION**

https://www.who.int/emergencies/diseases/novel-coronavirus-2019/global-research-on-novel-coronavirus-2019-ncov

4. Ministry of Health and Family Welfare - https://www.mohfw.gov.in/