In [1]:
```
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import plotly.express as px
```

In [2]:
```
country_df = pd.read_csv('covid_19_india.csv')
```

In [3]:
```
vaccine_df = pd.read_csv('covid_vaccine_statewise.csv')
```

In [4]:
```
country_df.shape
```

Out[4]:
```
(18110, 9)
```

Data information

In [5]:
```
country_df.columns
```

Out[5]:
```
Index(['Sno', 'Date', 'Time', 'State/UnionTerritory',
       'ConfirmedIndianNational', 'ConfirmedForeignNational', 'Cured',
       'Deaths', 'Confirmed'],
      dtype='object')
```

In [6]:
```
country_df.info()
```

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

Data cleaning

In [7]:
```
country_df.describe()
```

Out[7]:

	Sno	Cured	Deaths	Confirmed
count	18110.000000	1.811000e+04	18110.000000	1.811000e+04
mean	9055.500000	2.786375e+05	4052.402264	3.010314e+05
std	5228.051023	6.148909e+05	10919.076411	6.561489e+05
min	1.000000	0.000000e+00	0.000000	0.000000e+00
25%	4528.250000	3.360250e+03	32.000000	4.376750e+03
50%	9055.500000	3.336400e+04	588.000000	3.977350e+04
75%	13582.750000	2.788698e+05	3643.750000	3.001498e+05
max	18110.000000	6.159676e+06	134201.000000	6.363442e+06

In [8]:
```
country_df[((country_df['Date']=='2020-03-30')&(country_df['State/UnionTerritory']=='Ta
```

Out[8]:

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cure
495	496	2020-03-30	9:30 PM	Tamil Nadu	-	-	

In [9]:
```
df=country_df.copy()
```

Data cleaning

In [10]:
```
def drop_star(df):
    for i in df['State/UnionTerritory'].iteritems():
        if i[1][-3:] == "***":
            df.drop(i[0],inplace=True)

drop_star(df)
df['State/UnionTerritory'].unique()
```

Out[10]:
```
array(['Kerala', 'Telengana', 'Delhi', 'Rajasthan', 'Uttar Pradesh',
       'Haryana', 'Ladakh', 'Tamil Nadu', 'Karnataka', 'Maharashtra',
       'Punjab', 'Jammu and Kashmir', 'Andhra Pradesh', 'Uttarakhand',
       'Odisha', 'Puducherry', 'West Bengal', 'Chhattisgarh',
       'Chandigarh', 'Gujarat', 'Himachal Pradesh', 'Madhya Pradesh',
       'Bihar', 'Manipur', 'Mizoram', 'Andaman and Nicobar Islands',
       'Goa', 'Unassigned', 'Assam', 'Jharkhand', 'Arunachal Pradesh',
       'Tripura', 'Nagaland', 'Meghalaya',
       'Dadra and Nagar Haveli and Daman and Diu',
       'Cases being reassigned to states', 'Sikkim', 'Daman & Diu',
       'Lakshadweep', 'Telangana', 'Dadra and Nagar Haveli',
       'Himanchal Pradesh', 'Karanataka'], dtype=object)
```

In [11]:
```
df['Date']=pd.to_datetime(df['Date'],format='%Y-%m-%d')

df.drop(['Time'],axis=1, inplace=True)
```

```
df.rename(columns={'State/UnionTerritory':'States'}, inplace=True)
```

In [12]:

```
df['Active_cases']=df['Confirmed']-(df['Cured']+df['Deaths'])
df['Discharge_Rate'] = np.round((df['Cured']/df['Confirmed'])*100, decimals = 4)      #
df['Death_Rate'] = np.round((df['Deaths']/df['Confirmed'])*100, decimals = 4)
df.head()
```

Out[12]:

	Sno	Date	States	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed
0	1	2020-01-30	Kerala	1	0	0	0	1
1	2	2020-01-31	Kerala	1	0	0	0	1
2	3	2020-02-01	Kerala	2	0	0	0	2
3	4	2020-02-02	Kerala	3	0	0	0	3
4	5	2020-02-03	Kerala	3	0	0	0	3

In [13]:

```
import matplotlib.pyplot as plt
import matplotlib.dates as mtd
import seaborn as sns
from matplotlib.ticker import ScalarFormatter
colors=['#0C68C7','#3A6794','#00FAF3','#FA643C','#C71D12']
sns.set(palette=colors, style='white')

sns.palplot(colors)
```

state wise list on cured deaths and active cases on a particular day

In [14]:

```
current = df[df.Date == '2021-08-08']
```

In [15]:

```
max_confirmed_cases = current.sort_values(by='Confirmed', ascending = False)
max_confirmed_cases.head()
```

Out[15]:

	Sno	Date	States	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths
17986	17987	2021-08-08	Maharashtra	-	-	6139493	133845

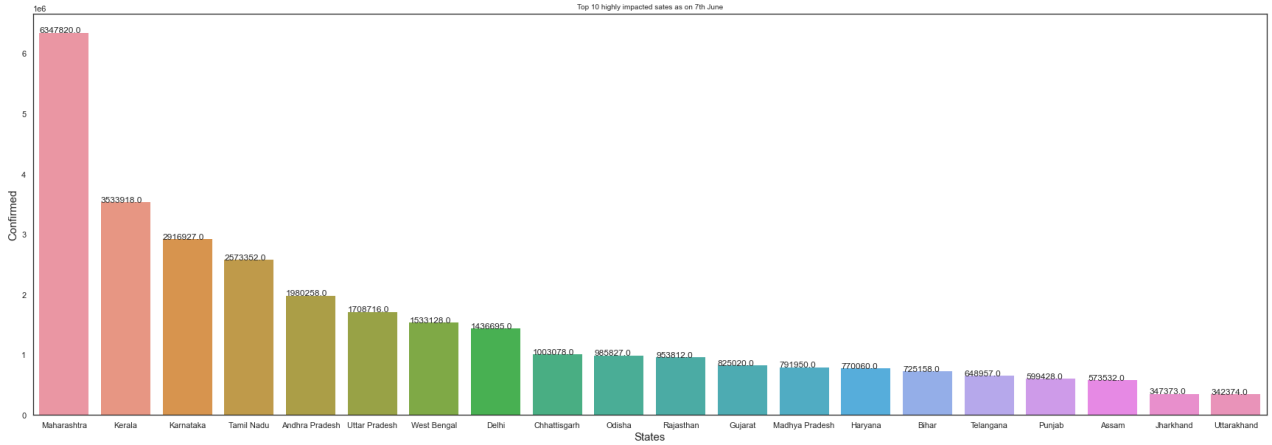
	Sno	Date	States	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths
17982	17983	2021-08-08	Kerala	-	-	3337579	17654
17981	17982	2021-08-08	Karnataka	-	-	2855862	36773
17996	17997	2021-08-08	Tamil Nadu	-	-	2518777	34289
17967	17968	2021-08-08	Andhra Pradesh	-	-	1946370	13513

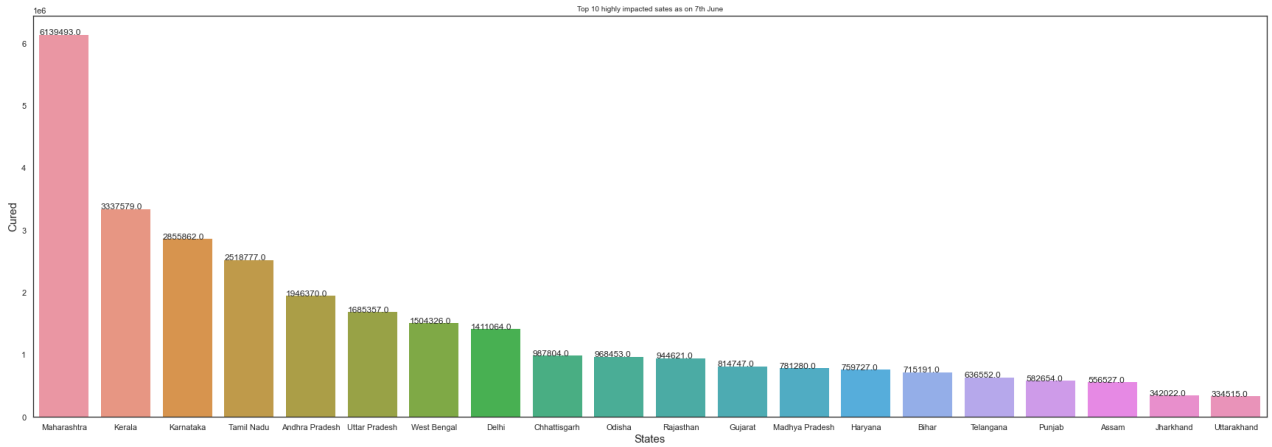
In [16]:
```
top_cases = max_confirmed_cases[:20]
top_cases.head()
```

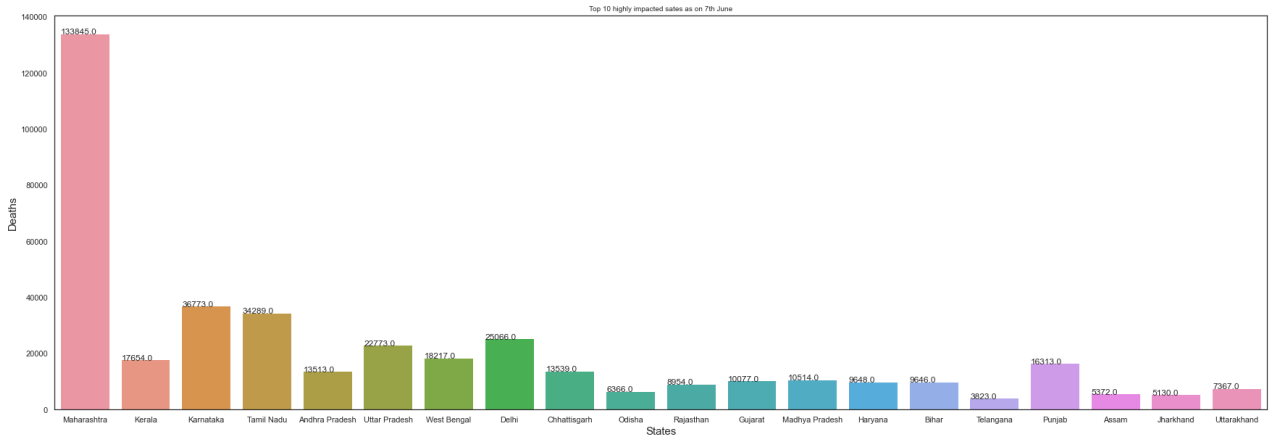
Out[16]:

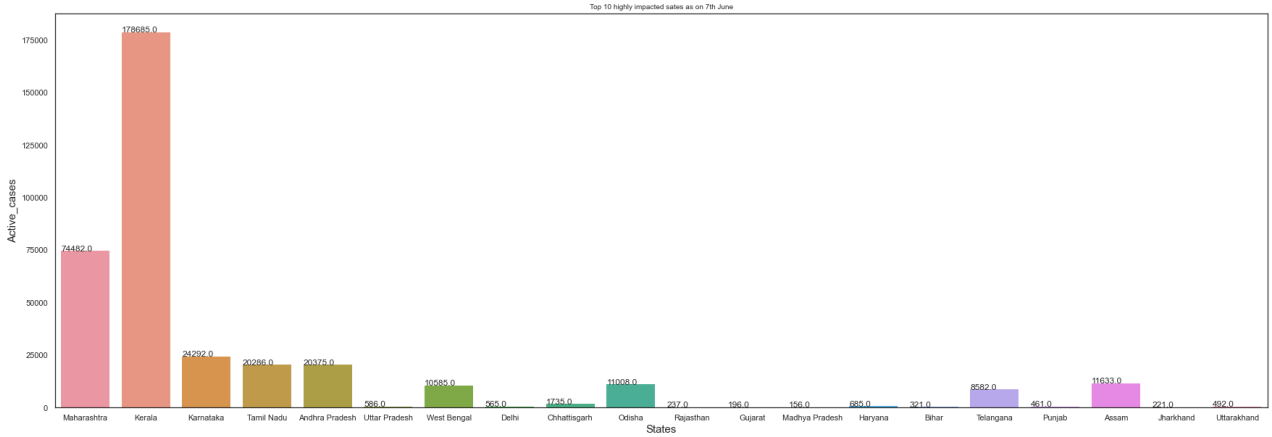
	Sno	Date	States	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths
17986	17987	2021-08-08	Maharashtra	-	-	6139493	133845
17982	17983	2021-08-08	Kerala	-	-	3337579	17654
17981	17982	2021-08-08	Karnataka	-	-	2855862	36773
17996	17997	2021-08-08	Tamil Nadu	-	-	2518777	34289
17967	17968	2021-08-08	Andhra Pradesh	-	-	1946370	13513

In [17]:
```
for feature in top_cases[['Confirmed','Cured','Deaths','Active_cases']]:
    fig=plt.figure(figsize=(30,10))
    plt.title("Top 10 highly impacted sates as on 7th June", size=10)
    ax=sns.barplot(data=top_cases,y=top_cases[feature],x='States', linewidth=0, edgecol
    plt.xlabel('States', size = 15)
    plt.ylabel(feature, size = 15)
    for i in ax.patches:
        ax.text(x=i.get_x(),y=i.get_height(),s=i.get_height())
plt.show()
```







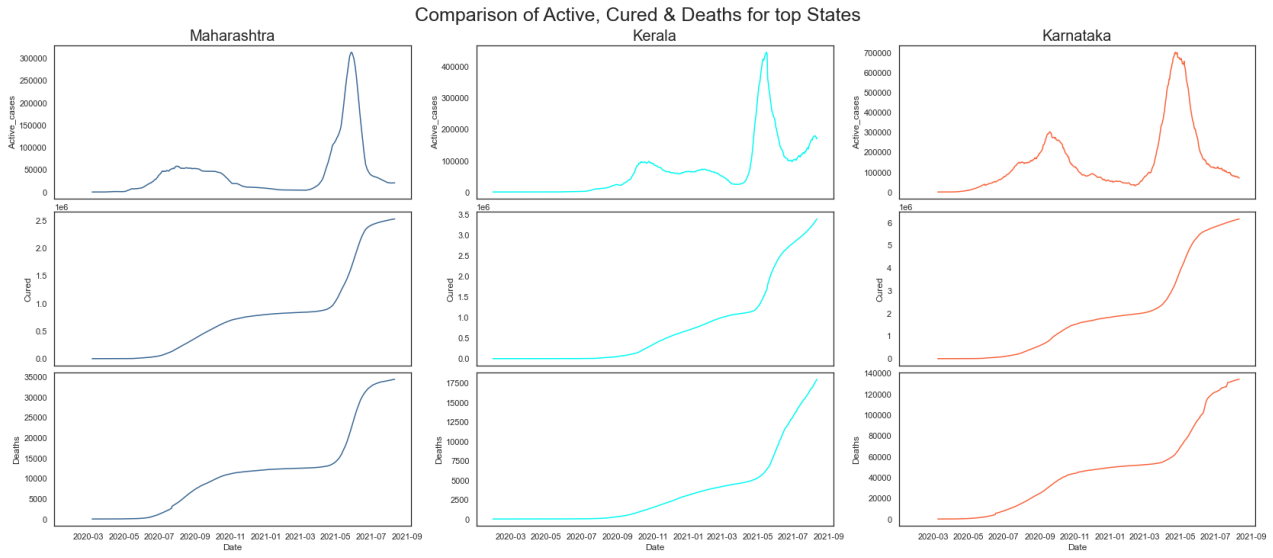


comparsion Cured, deaths, active cases

between top states

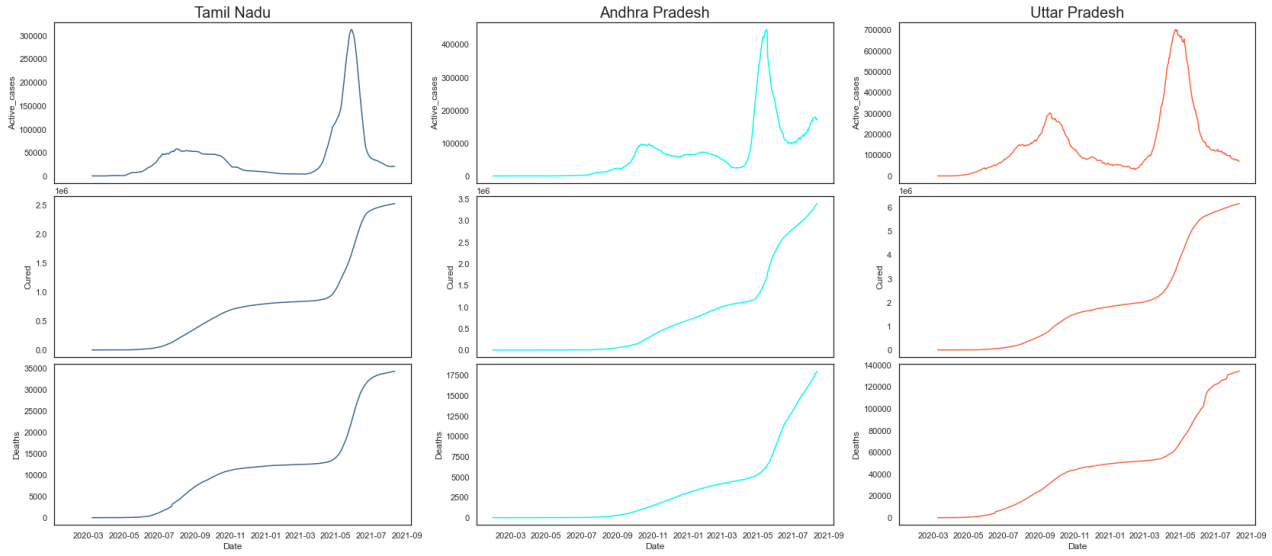
In [18]:
```
states=['Kerala', 'Tamil Nadu', 'Maharashtra', 'Tamil Nadu', 'Andhra Pradesh', 'Uttar P
mh=df[df['States']=='Maharashtra']
kl=df[df['States']=='Kerala']
ka=df[df['States']=='Karnataka']
tn=df[df['States']=='Tamil Nadu']
ap=df[df['States']=='Andhra Pradesh']
up=df[df['States']=='Uttar Pradesh']
```

In [19]:
```
fig, ax=plt.subplots(nrows=3, ncols=3, figsize=(23,10), squeeze=False, sharex=True, sha
plt.suptitle("Comparison of Active, Cured & Deaths for top States", size = 25)
sns.lineplot(data=tn, x='Date',y='Active_cases', ax=ax[0,0], color=colors[1])
ax[0,0].set_title("Maharashtra", size=20)
sns.lineplot(data=tn, x='Date',y='Cured', ax=ax[1,0], color=colors[1])
sns.lineplot(data=tn, x='Date',y='Deaths', ax=ax[2,0], color=colors[1])

sns.lineplot(data=kl, x='Date',y='Active_cases', ax=ax[0,1], color=colors[2])
ax[0,1].set_title("Kerala", size=20)
sns.lineplot(data=kl, x='Date',y='Cured', ax=ax[1,1], color=colors[2])
sns.lineplot(data=kl, x='Date',y='Deaths', ax=ax[2,1], color=colors[2])

sns.lineplot(data=mh, x='Date',y='Active_cases', ax=ax[0,2], color=colors[3])
ax[0,2].set_title("Karnataka", size=20)
sns.lineplot(data=mh, x='Date',y='Cured', ax=ax[1,2], color=colors[3])
sns.lineplot(data=mh, x='Date',y='Deaths', ax=ax[2,2], color=colors[3])
plt.show()
```



In [20]:
```
fig, ax=plt.subplots(nrows=3, ncols=3, figsize=(23,10), squeeze=False, sharex=True, sha
sns.lineplot(data=tn, x='Date',y='Active_cases', ax=ax[0,0], color=colors[1])
ax[0,0].set_title("Tamil Nadu", size=20)
sns.lineplot(data=tn, x='Date',y='Cured', ax=ax[1,0], color=colors[1])
sns.lineplot(data=tn, x='Date',y='Deaths', ax=ax[2,0], color=colors[1])

sns.lineplot(data=kl, x='Date',y='Active_cases', ax=ax[0,1], color=colors[2])
ax[0,1].set_title("Andhra Pradesh", size=20)
sns.lineplot(data=kl, x='Date',y='Cured', ax=ax[1,1], color=colors[2])
sns.lineplot(data=kl, x='Date',y='Deaths', ax=ax[2,1], color=colors[2])
```

```
sns.lineplot(data=mh, x='Date',y='Active_cases', ax=ax[0,2], color=colors[3])
ax[0,2].set_title("Uttar Pradesh", size=20)
sns.lineplot(data=mh, x='Date',y='Cured', ax=ax[1,2], color=colors[3])
sns.lineplot(data=mh, x='Date',y='Deaths', ax=ax[2,2], color=colors[3])
plt.show()
```



what is the Discharge rate per month in 2020 and 2021?

In [21]:

```
df['Date']= pd.to_datetime(df['Date'])
data_20 = df[df['Date'].dt.year==2020]
data_21 = df[df['Date'].dt.year==2021]

data_20['Month']=data_20['Date'].dt.month
data_21['Month']=data_21['Date'].dt.month

#Year 2020
data_confirm_20= data_20['Confirmed'].groupby(data_20['Month']).sum()
data_dis_20= data_20['Cured'].groupby(data_20['Month']).sum()      # creating instances
data_death_20= data_20['Deaths'].groupby(data_20['Month']).sum()

#Year 2021
data_confirm_21= data_21['Confirmed'].groupby(data_21['Month']).sum()
data_dis_21= data_21['Cured'].groupby(data_21['Month']).sum()      # creating instances
data_death_21= data_21['Deaths'].groupby(data_21['Month']).sum()

cols_20=[data_confirm_20,data_dis_20,data_death_20]
data_20=pd.concat(cols_20,axis=1)

cols_21=[data_confirm_21,data_dis_21,data_death_21]
data_21=pd.concat(cols_21,axis=1)

data_20['discharge_rate_20'] = np.round((data_20['Cured']/data_20['Confirmed'])*100, de
data_20['death_rate_20'] = np.round((data_20['Deaths']/data_20['Confirmed'])*100, decim

#Year 2020
```

```
data_21['discharge_rate_21'] = np.round((data_21['Cured']/data_21['Confirmed'])*100, de
data_21['death_rate_21'] = np.round((data_21['Deaths']/data_21['Confirmed'])*100, decim
```

```
C:\Users\SOWNDAR\AppData\Local\Temp/ipykernel_10960/2646291606.py:5: SettingWithCopyWarn
ing:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  data_20['Month']=data_20['Date'].dt.month
C:\Users\SOWNDAR\AppData\Local\Temp/ipykernel_10960/2646291606.py:6: SettingWithCopyWarn
ing:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  data_21['Month']=data_21['Date'].dt.month
```

In [22]:

```
data_20.reset_index(inplace=True)
data_20.head()
```

Out[22]:

	Month	Confirmed	Cured	Deaths	discharge_rate_20	death_rate_20
0	1	2	0	0	0.0000	0.0000
1	2	86	0	0	0.0000	0.0000
2	3	9687	808	202	8.3411	2.0853
3	4	422442	75443	13270	17.8588	3.1413
4	5	2938234	1133341	89834	38.5722	3.0574

In [23]:

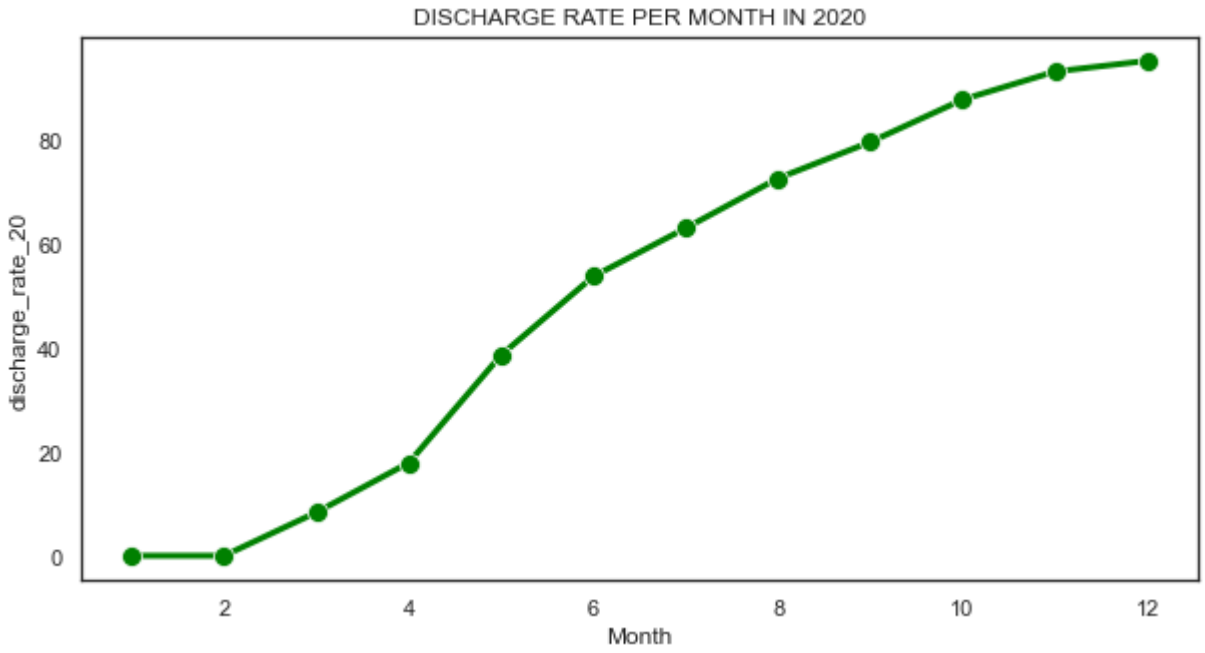
```
data_21.reset_index(inplace=True)
data_21.head()
```

Out[23]:

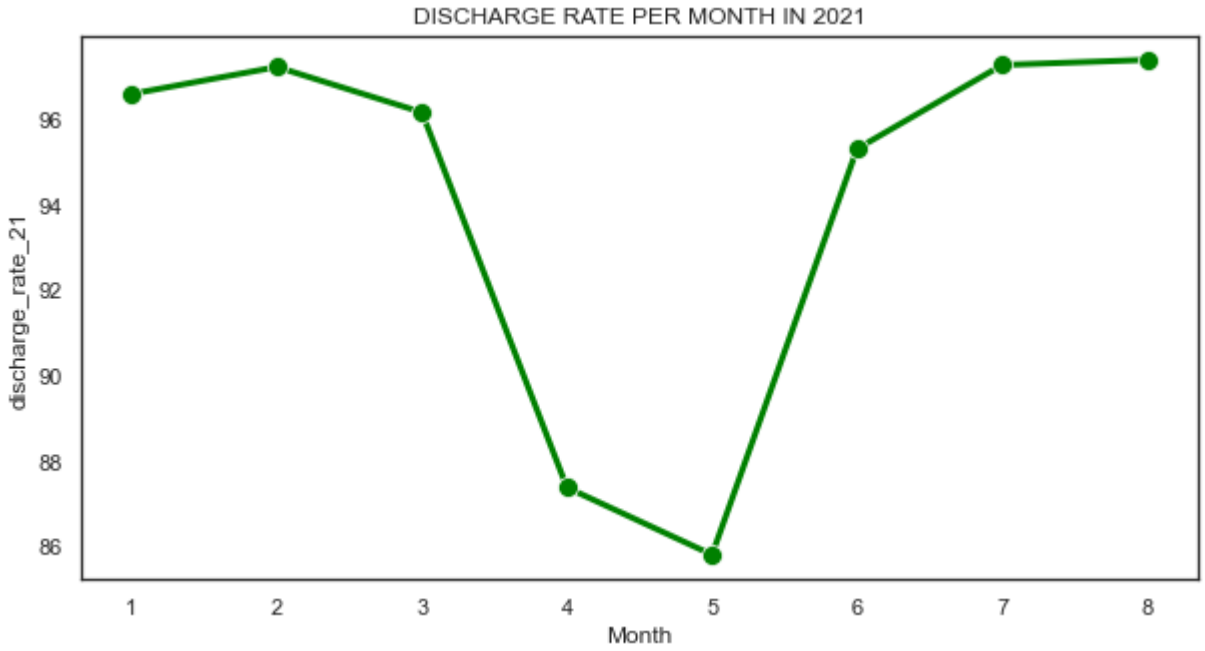
	Month	Confirmed	Cured	Deaths	discharge_rate_21	death_rate_21
0	1	326469747	315332019	4709167	96.5884	1.4425
1	2	305631803	297133802	4359434	97.2195	1.4264
2	3	356305616	342610397	4935253	96.1563	1.3851
3	4	440660671	384990190	5340298	87.3666	1.2119
4	5	751927486	645106765	8390917	85.7937	1.1159

In [24]:

```
plt.figure(figsize=(10,5))
sns.lineplot(x="Month",y="discharge_rate_20",data=data_20,color="g",lw=3,marker='o',mar
plt.title('DISCHARGE RATE PER MONTH IN 2020')
plt.show()
```

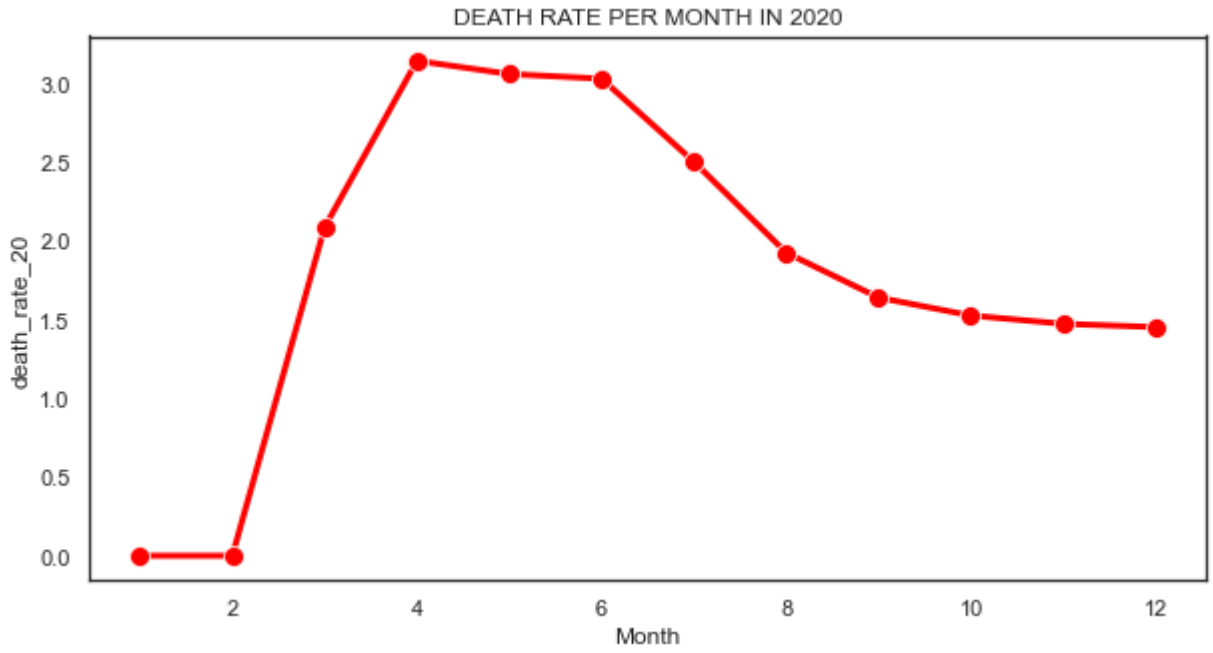


In [25]:
```
plt.figure(figsize=(10,5))
sns.lineplot(x="Month",y="discharge_rate_21",data=data_21,color="g",lw=3,marker='o',mar
plt.title('DISCHARGE RATE PER MONTH IN 2021')
plt.show()
```

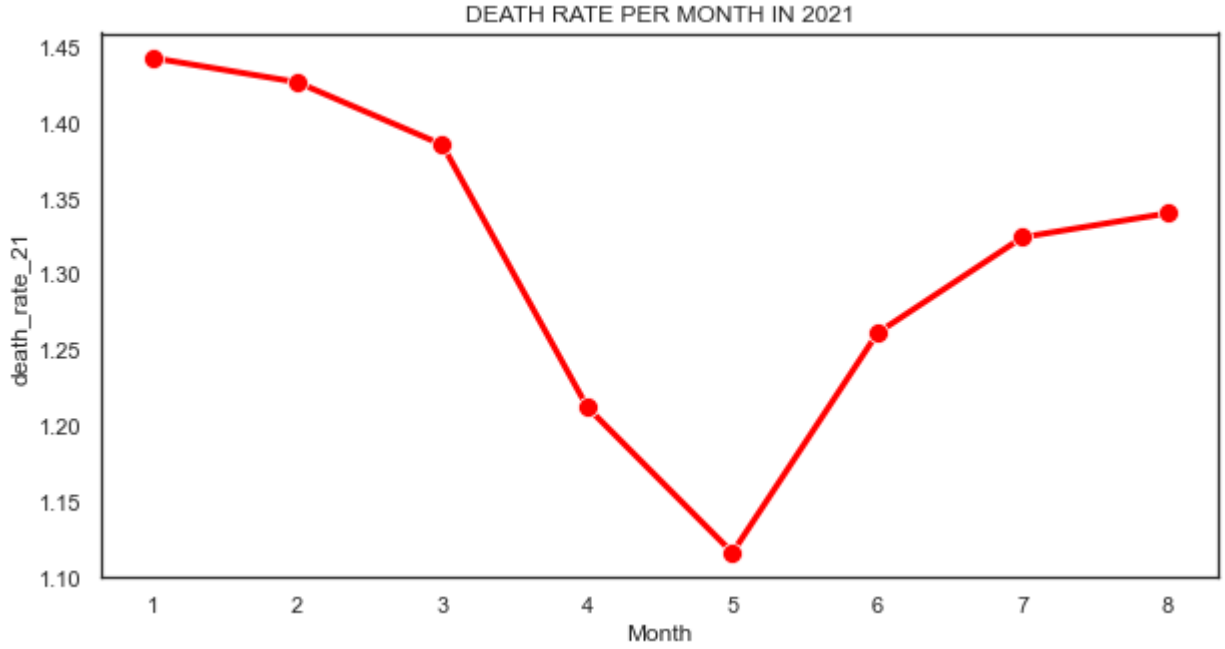


what is Death Rate in 2020 and 2021?

In [26]:
```
plt.figure(figsize=(10,5))
sns.lineplot(x="Month",y="death_rate_20",data=data_20,color="r",lw=3,marker='o',markers
plt.title('DEATH RATE PER MONTH IN 2020')
plt.show()
```

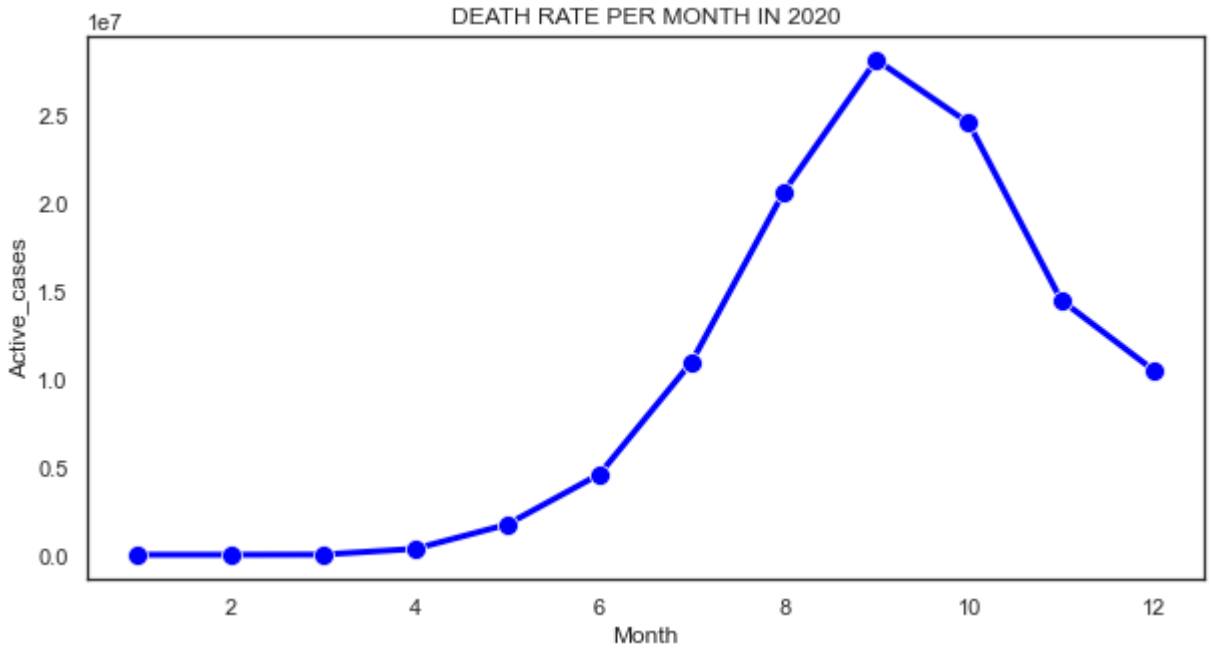


In [27]:
```
plt.figure(figsize=(10,5))
sns.lineplot(x="Month",y="death_rate_21",data=data_21,color="r",lw=3,marker='o',markers
plt.title('DEATH RATE PER MONTH IN 2021')
plt.show()
```

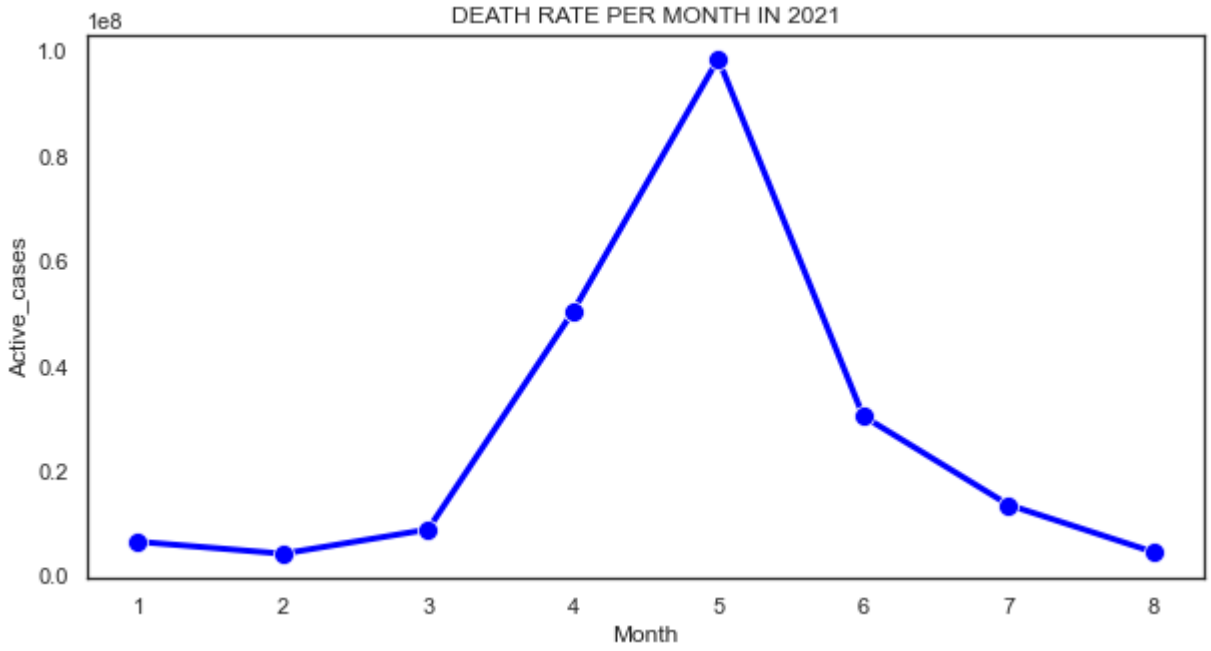


what is death rate per month in 2020 and 2021?

In [28]:
```
data_20['Active_cases']=data_20['Confirmed']-(data_20['Cured']+data_20['Deaths'])
plt.figure(figsize=(10,5))
sns.lineplot(x="Month",y="Active_cases",data=data_20,color="b",lw=3,marker='o',markersi
plt.title('DEATH RATE PER MONTH IN 2020')
plt.show()
```



In [29]:
```
data_21['Active_cases']=data_21['Confirmed']-(data_21['Cured']+data_21['Deaths'])
plt.figure(figsize=(10,5))
sns.lineplot(x="Month",y="Active_cases",data=data_21,color="b",lw=3,marker='o',markersi
plt.title('DEATH RATE PER MONTH IN 2021')
plt.show()
```



What Male, Female and Transgender Vaccinated ratio for Covid19

In [30]:
```
vacc_df=vaccine_df.copy()
vaccine_df = vaccine_df[['Updated On','State','Total Doses Administered','First Dose Ad
                         'Male (Doses Administered)','Female (Doses Administered)','Tra
```

```
                          '18-44 Years (Doses Administered)','45-60 Years (Doses Adminis
                          '18-44 Years(Individuals Vaccinated)','45-60 Years(Individuals
                          'Male(Individuals Vaccinated)','Female(Individuals Vaccinated)
                          'Total Individuals Vaccinated']]
vaccine_df.columns = ['Date','States','Total Doses Administered','First Dose Administer
                      'Male (Doses Administered)','Female (Doses Administered)','Transg
                      '18-44 Years (Doses Administered)','45-60 Years (Doses Administer
                      '18-44 Years(Individuals Vaccinated)','45-60 Years(Individuals Va
                      'Male Vaccinated','Female Vaccinated','Transgender Vaccinated','T
vaccine_df.head()
```

Out[30]:

	Date	States	Total Doses Administered	First Dose Administered	Second Dose Administered	Male (Doses Administered)	Female (Doses Administered)	Trans Admini
0	16/01/2021	India	48276.0	48276.0	0.0	NaN	NaN	
1	17/01/2021	India	58604.0	58604.0	0.0	NaN	NaN	
2	18/01/2021	India	99449.0	99449.0	0.0	NaN	NaN	
3	19/01/2021	India	195525.0	195525.0	0.0	NaN	NaN	
4	20/01/2021	India	251280.0	251280.0	0.0	NaN	NaN	

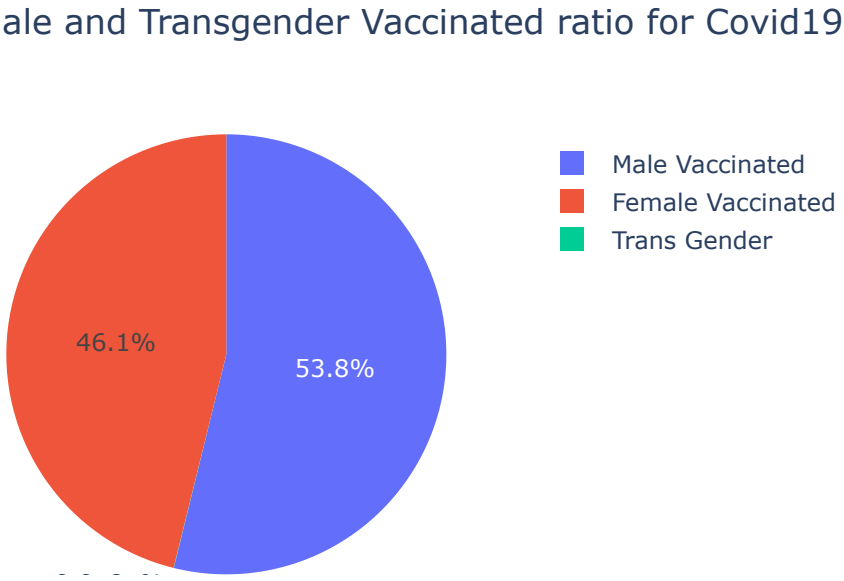
In [31]:

```
vaccine_df['Date'] = pd.to_datetime(vaccine_df['Date'], format = '%d/%m/%Y')
latest_date = max(vaccine_df['Date'])
print("Current Date : ", latest_date)
```

```
Current Date :  2021-08-16 00:00:00
```

In [32]:

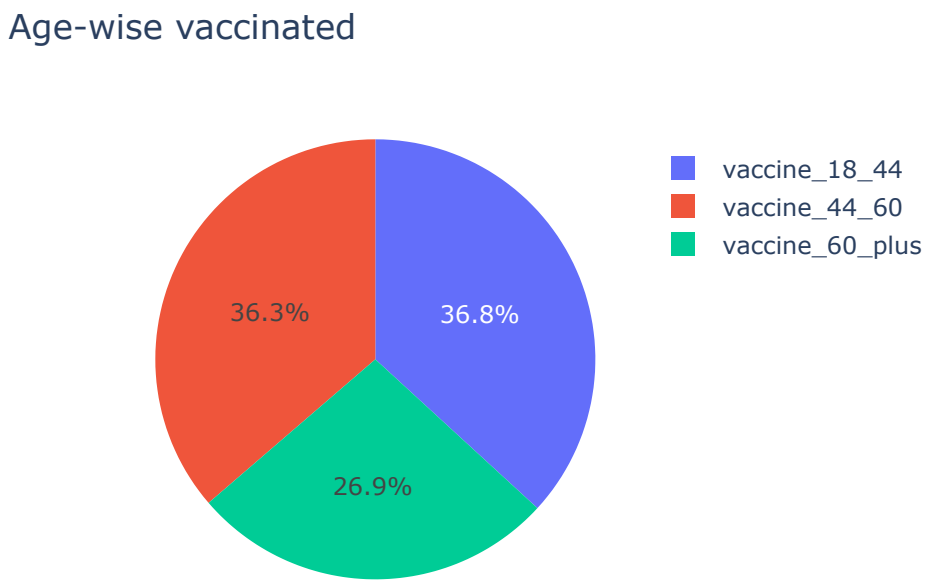
```
male = vaccine_df["Male Vaccinated"].max()
female = vaccine_df["Female Vaccinated"].max()
trans = vaccine_df["Transgender Vaccinated"].max()
px.pie(names=["Male Vaccinated","Female Vaccinated","Trans Gender"],values=[male,female
       title="Male, Female and Transgender Vaccinated ratio for Covid19",width=550,heig
```



0.0185%

Age wise vaccinated list

In [33]:
```
vaccine_18_44 = vaccine_df['18-44 Years(Individuals Vaccinated)'].max()
vaccine_44_60 = vaccine_df['45-60 Years(Individuals Vaccinated)'].max()
vaccine_60_plus = vaccine_df['60+ Years(Individuals Vaccinated)'].max()
px.pie(names=["vaccine_18_44","vaccine_44_60","vaccine_60_plus"],values=[vaccine_18_44,
```



Total state wise cure deaths and active cases list.

In [34]:
```
df_by_State = df.groupby('States')[['Cured','Deaths' ,'Confirmed']].sum().reset_index()
df_by_State
```

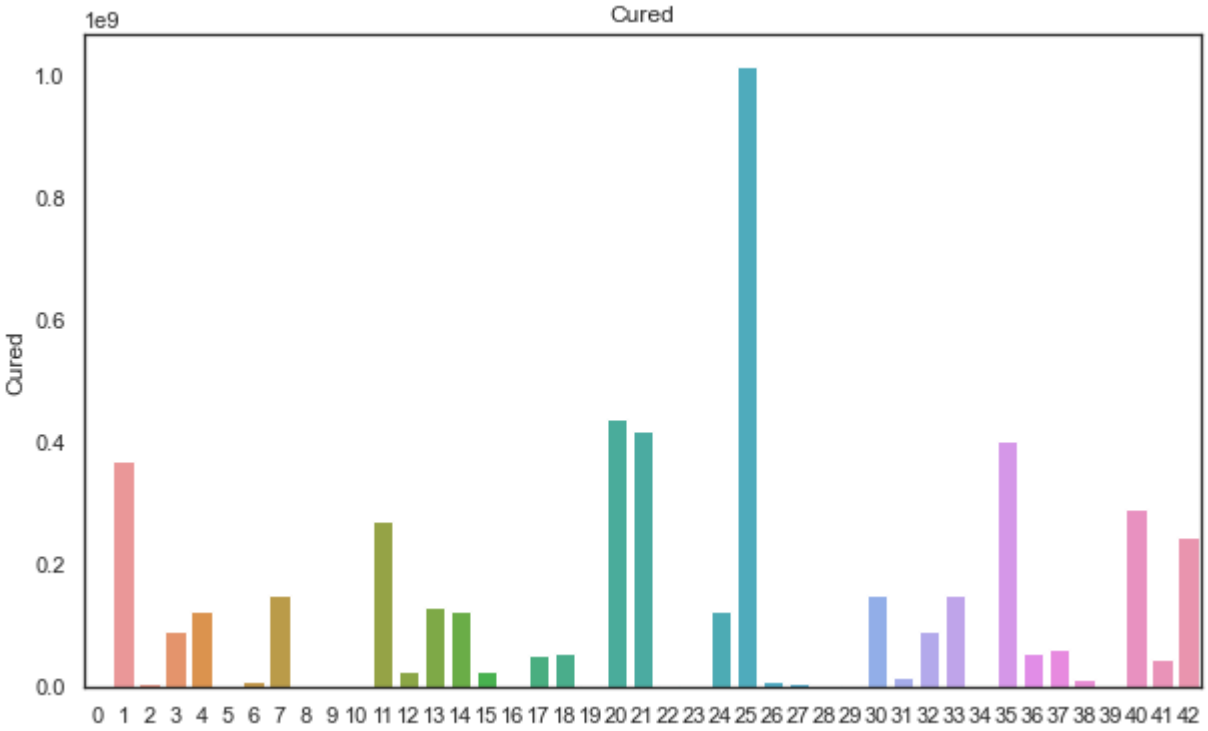
Out[34]:

	States	Cured	Deaths	Confirmed
0	Andaman and Nicobar Islands	1848286	27136	1938498
1	Andhra Pradesh	370426530	2939367	392432753
2	Arunachal Pradesh	6588149	26799	7176907
3	Assam	92678680	638323	99837011

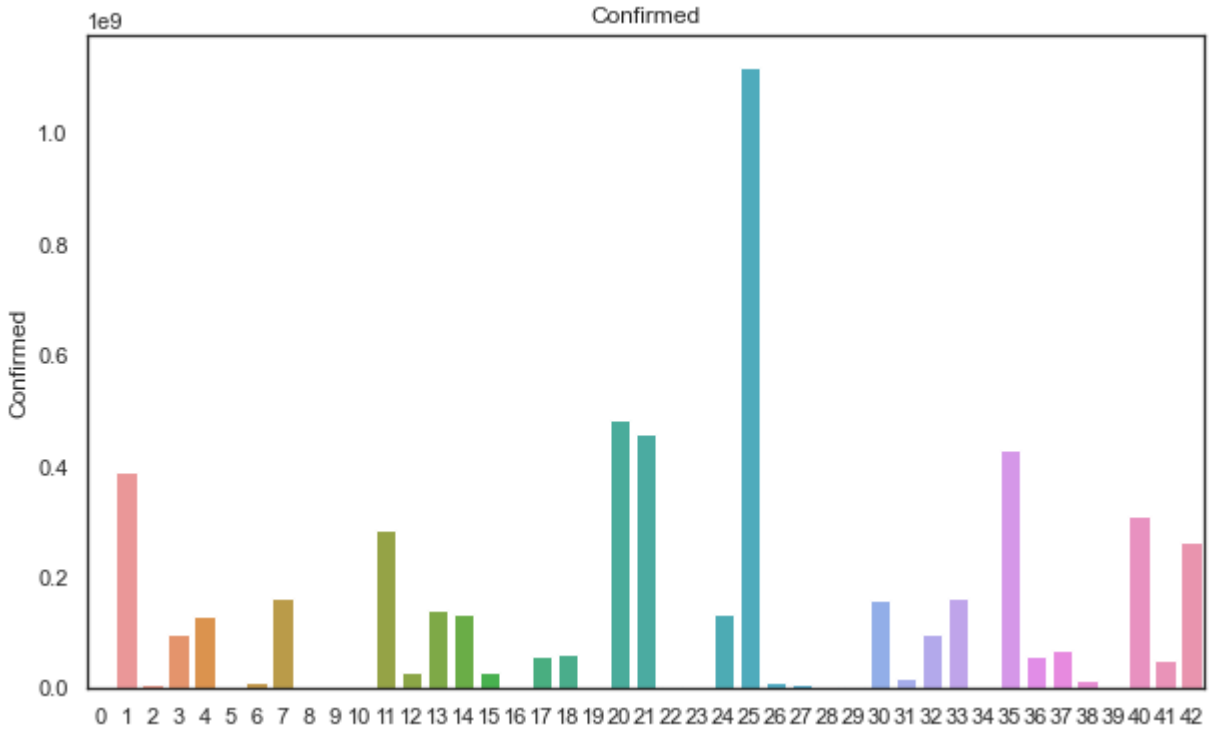
	States	Cured	Deaths	Confirmed
4	Bihar	125122902	1093466	132231166
5	Cases being reassigned to states	0	0	345565
6	Chandigarh	10117035	147694	10858627
7	Chhattisgarh	151609364	2063920	163776262
8	Dadra and Nagar Haveli	20352	8	20722
9	Dadra and Nagar Haveli and Daman and Diu	1841750	1014	1938632
10	Daman & Diu	0	0	2
11	Delhi	273419887	4943294	287227765
12	Goa	26027201	447801	28240159
13	Gujarat	132487127	2219448	143420082
14	Haryana	126585342	1502799	134347285
15	Himachal Pradesh	27501110	491348	30033289
16	Himanchal Pradesh	200040	3507	204516
17	Jammu and Kashmir	53297341	839694	58117726
18	Jharkhand	58034506	748641	62111994
19	Karanataka	2821491	36197	2885238
20	Karnataka	441844360	6053762	485970693
21	Kerala	420174235	1888177	458906023
22	Ladakh	3758960	45804	4054293
23	Lakshadweep	820925	3908	915784
24	Madhya Pradesh	126724997	1777752	135625265
25	Maharashtra	1018765039	23737432	1121491467
26	Manipur	11230568	173056	12617943
27	Meghalaya	6537909	101950	7355969
28	Mizoram	2384602	9791	2984732
29	Nagaland	4519526	58460	5041742
30	Odisha	150923455	790814	160130533
31	Puducherry	18483117	312155	20065891
32	Punjab	91458159	2785594	99949702
33	Rajasthan	150356820	1473089	162369656
34	Sikkim	2747214	53150	3186799
35	Tamil Nadu	404095807	5916658	431928644
36	Telangana	57488245	349648	60571979

	States	Cured	Deaths	Confirmed
37	Telengana	64666267	400427	69990668
38	Tripura	12976846	150342	14050250
39	Unassigned	0	0	161
40	Uttar Pradesh	291479351	4143450	312625843
41	Uttarakhand	48362741	986001	53140414
42	West Bengal	247515102	3846989	263107876

In [35]:
```
plt.figure(figsize=(10,6))

plt.title("Cured")

sns.barplot(x=df_by_State.index, y=df_by_State['Cured']);

plt.ylabel("Cured");
```



In [36]:
```
plt.figure(figsize=(10,6))

plt.title("Confirmed")

sns.barplot(x=df_by_State.index, y=df_by_State['Confirmed']);

plt.ylabel("Confirmed");
```



In [37]:

```
plt.figure(figsize=(10,6))

plt.title("Deaths")

sns.barplot(x=df_by_State.index, y=df_by_State['Deaths']);

plt.ylabel("Deaths");
```