In [1]:
```python
import pandas as pd
import numpy as np
from datetime import datetime
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
sns.set(style='whitegrid',color_codes=True)
```

In [2]:
```python
df = pd.read_csv(r'C:\Users\Admin\Downloads\covid_19_india.csv')
```

In [3]:
```python
df
```

Out[3]:
```
         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
  ...    ...         ...      ...                   ...                      ...                       ...      ...     ...        ...
16845  16846  2021-07-07  8:00 AM             Telangana                        -                         -   613124    3703     628282
16846  16847  2021-07-07  8:00 AM               Tripura                        -                         -    63964     701      68612
16847  16848  2021-07-07  8:00 AM           Uttarakhand                        -                         -   332006    7338     340882
16848  16849  2021-07-07  8:00 AM         Uttar Pradesh                        -                         -  1682130   22656    1706818
16849  16850  2021-07-07  8:00 AM           West Bengal                        -                         -  1472132   17834    1507241

16850 rows × 9 columns
```

Data cleaning

In [4]:
```
df.replace('-',0)
```

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
...	...	...	...	...	...	...	...	...	...
16845	16846	2021-07-07	8:00 AM	Telangana	0	0	613124	3703	628282
16846	16847	2021-07-07	8:00 AM	Tripura	0	0	63964	701	68612
16847	16848	2021-07-07	8:00 AM	Uttarakhand	0	0	332006	7338	340882
16848	16849	2021-07-07	8:00 AM	Uttar Pradesh	0	0	1682130	22656	1706818
16849	16850	2021-07-07	8:00 AM	West Bengal	0	0	1472132	17834	1507241

16850 rows × 9 columns

In [5]:
```
df.drop(["ConfirmedIndianNational","ConfirmedForeignNational"],inplace=True,axis=1)
```

In [6]:
```
df.drop(['Sno'],inplace=True,axis=1)
```

In [7]:
```
df
```

Out[7]:

	Date	Time	State/UnionTerritory	Cured	Deaths	Confirmed
0	2020-01-30	6:00 PM	Kerala	0	0	1
1	2020-01-31	6:00 PM	Kerala	0	0	1
2	2020-02-01	6:00 PM	Kerala	0	0	2
3	2020-02-02	6:00 PM	Kerala	0	0	3
4	2020-02-03	6:00 PM	Kerala	0	0	3
...	...	...	...	...	...	...
16845	2021-07-07	8:00 AM	Telangana	613124	3703	628282
16846	2021-07-07	8:00 AM	Tripura	63964	701	68612
16847	2021-07-07	8:00 AM	Uttarakhand	332006	7338	340882
16848	2021-07-07	8:00 AM	Uttar Pradesh	1682130	22656	1706818
16849	2021-07-07	8:00 AM	West Bengal	1472132	17834	1507241

16850 rows × 6 columns

In [8]:
```
ex=np.unique(df['State/UnionTerritory'])
```

In [9]:
```
ex
```

Out[9]:
```
array(['Andaman and Nicobar Islands', 'Andhra Pradesh',
       'Arunachal Pradesh', 'Assam', 'Bihar', 'Bihar****',
       'Cases being reassigned to states', 'Chandigarh', 'Chhattisgarh',
       'Dadra and Nagar Haveli',
       'Dadra and Nagar Haveli and Daman and Diu', 'Daman & Diu', 'Delhi',
       'Goa', 'Gujarat', 'Haryana', 'Himachal Pradesh',
       'Jammu and Kashmir', 'Jharkhand', 'Karnataka', 'Kerala', 'Ladakh',
       'Lakshadweep', 'Madhya Pradesh', 'Maharashtra', 'Manipur',
       'Meghalaya', 'Mizoram', 'Nagaland', 'Odisha', 'Puducherry',
       'Punjab', 'Rajasthan', 'Sikkim', 'Tamil Nadu', 'Telangana',
       'Telengana', 'Tripura', 'Unassigned', 'Uttar Pradesh',
       'Uttarakhand', 'West Bengal'], dtype=object)
```

In [10]:
```
def clean_stateName(stateName):
    if stateName == 'Telangana':
        stateName = 'Telangana'
    elif stateName == 'Telengana':
        stateName = 'Telangana'
    elif stateName == 'Bihar****':
        stateName = 'Bihar'
    elif stateName == 'Himanchal Pradesh':
        stateName = 'Himachal Pradesh'
    elif stateName == 'Karanataka':
        stateName = 'Karnataka'
    elif stateName == 'Madhya Pradesh***':
        stateName = 'Madhya Pradesh'
    elif stateName == 'Maharashtra***':
        stateName = 'Maharashtra'
    elif stateName == 'Daman & Diu':
        stateName = 'Dadra and Nagar Haveli and Daman and Diu'
    elif stateName == 'Dadra and Nagar Haveli':
        stateName = 'Dadra and Nagar Haveli and Daman and Diu'
    return stateName
```

In [11]:
```
df['State/UnionTerritory']=df['State/UnionTerritory'].apply(lambda x
                                                            : clean_stateName(x))
np.unique(df['State/UnionTerritory'])
```

Out[11]:
```
array(['Andaman and Nicobar Islands', 'Andhra Pradesh',
       'Arunachal Pradesh', 'Assam', 'Bihar',
       'Cases being reassigned to states', 'Chandigarh', 'Chhattisgarh',
       'Dadra and Nagar Haveli and Daman and Diu', 'Delhi', 'Goa',
       'Gujarat', 'Haryana', 'Himachal Pradesh', 'Jammu and Kashmir',
       'Jharkhand', 'Karnataka', 'Kerala', 'Ladakh', 'Lakshadweep',
       'Madhya Pradesh', 'Maharashtra', 'Manipur', 'Meghalaya', 'Mizoram',
       'Nagaland', 'Odisha', 'Puducherry', 'Punjab', 'Rajasthan',
       'Sikkim', 'Tamil Nadu', 'Telangana', 'Tripura', 'Unassigned',
       'Uttar Pradesh', 'Uttarakhand', 'West Bengal'], dtype=object)
```

In [12]:
```
df.drop(df[df['State/UnionTerritory']=='Unassigned'].index, inplace=True)
```

In [13]:
```
df
```

Out[13]:

	Date	Time	State/UnionTerritory	Cured	Deaths	Confirmed
0	2020-01-30	6:00 PM	Kerala	0	0	1
1	2020-01-31	6:00 PM	Kerala	0	0	1
2	2020-02-01	6:00 PM	Kerala	0	0	2
3	2020-02-02	6:00 PM	Kerala	0	0	3
4	2020-02-03	6:00 PM	Kerala	0	0	3
...	...	...	...	...	...	...
16845	2021-07-07	8:00 AM	Telangana	613124	3703	628282
16846	2021-07-07	8:00 AM	Tripura	63964	701	68612
16847	2021-07-07	8:00 AM	Uttarakhand	332006	7338	340882
16848	2021-07-07	8:00 AM	Uttar Pradesh	1682130	22656	1706818
16849	2021-07-07	8:00 AM	West Bengal	1472132	17834	1507241

16847 rows × 6 columns

In [14]:
```
df.drop(df[df['State/UnionTerritory']=='Cases being reassigned to states'].index, inplace=True)
```

```
In [15]: df
```

Out[15]:

	Date	Time	State/UnionTerritory	Cured	Deaths	Confirmed
0	2020-01-30	6:00 PM	Kerala	0	0	1
1	2020-01-31	6:00 PM	Kerala	0	0	1
2	2020-02-01	6:00 PM	Kerala	0	0	2
3	2020-02-02	6:00 PM	Kerala	0	0	3
4	2020-02-03	6:00 PM	Kerala	0	0	3
...	...	...	...	...	...	...
16845	2021-07-07	8:00 AM	Telangana	613124	3703	628282
16846	2021-07-07	8:00 AM	Tripura	63964	701	68612
16847	2021-07-07	8:00 AM	Uttarakhand	332006	7338	340882
16848	2021-07-07	8:00 AM	Uttar Pradesh	1682130	22656	1706818
16849	2021-07-07	8:00 AM	West Bengal	1472132	17834	1507241

16787 rows × 6 columns

```
In [16]: np.unique(df['State/UnionTerritory'])
```

```
Out[16]: array(['Andaman and Nicobar Islands', 'Andhra Pradesh',
               'Arunachal Pradesh', 'Assam', 'Bihar', 'Chandigarh',
               'Chhattisgarh', 'Dadra and Nagar Haveli and Daman and Diu',
               'Delhi', 'Goa', 'Gujarat', 'Haryana', 'Himachal Pradesh',
               'Jammu and Kashmir', 'Jharkhand', 'Karnataka', 'Kerala', 'Ladakh',
               'Lakshadweep', 'Madhya Pradesh', 'Maharashtra', 'Manipur',
               'Meghalaya', 'Mizoram', 'Nagaland', 'Odisha', 'Puducherry',
               'Punjab', 'Rajasthan', 'Sikkim', 'Tamil Nadu', 'Telangana',
               'Tripura', 'Uttar Pradesh', 'Uttarakhand', 'West Bengal'],
              dtype=object)
```

In [17]:
```
df.groupby(['Date'])['Confirmed','Cured','Deaths','State/UnionTerritory'].max()
```

```
C:\Users\Admin\AppData\Local\Temp/ipykernel_6380/3733384939.py:1: FutureWarning: Indexing with multiple keys (implicitl
y converted to a tuple of keys) will be deprecated, use a list instead.
  df.groupby(['Date'])['Confirmed','Cured','Deaths','State/UnionTerritory'].max()
```

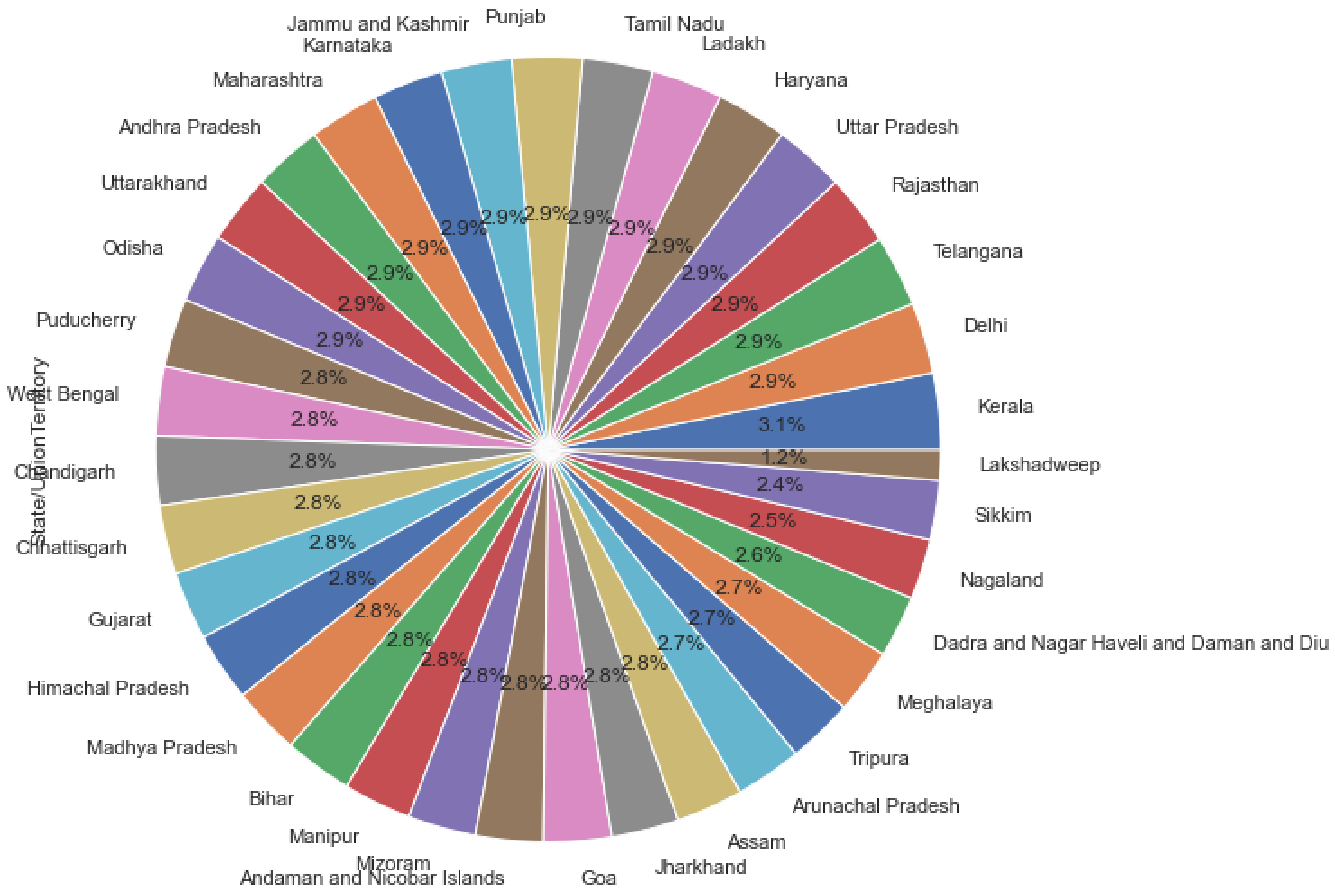
Out[17]:

	Confirmed	Cured	Deaths	State/UnionTerritory
Date				
2020-01-30	1	0	0	Kerala
2020-01-31	1	0	0	Kerala
2020-02-01	2	0	0	Kerala
2020-02-02	3	0	0	Kerala
2020-02-03	3	0	0	Kerala
...	...	...	...	...
2021-07-03	6079352	5836920	122353	West Bengal
2021-07-04	6088841	5845315	122724	West Bengal
2021-07-05	6098177	5848693	123030	West Bengal
2021-07-06	6104917	5861720	123136	West Bengal
2021-07-07	6113335	5872268	123531	West Bengal

525 rows × 4 columns

```
In [18]: plt.figure(figsize=(20,10))
         df['State/UnionTerritory'].value_counts().plot.pie(autopct='%1.1f%%')
```

```
Out[18]: <AxesSubplot:ylabel='State/UnionTerritory'>
```



In [19]:
```
df['Date']= pd.to_datetime(df['Date'])
df['Date']
```

Out[19]:
```
0        2020-01-30
1        2020-01-31
2        2020-02-01
3        2020-02-02
4        2020-02-03
            ...
16845    2021-07-07
16846    2021-07-07
16847    2021-07-07
16848    2021-07-07
16849    2021-07-07
Name: Date, Length: 16787, dtype: datetime64[ns]
```

In [20]:
```
df['Day'] = df['Date'].dt.day
df['Month']= df['Date'].dt.month
df['Year'] = df['Date'].dt.year
```

In [21]:
```
Monthly_data= df.groupby(['Month','State/UnionTerritory'])[['Date',"Cured"]].sum().sort_values(by=['Month']).reset_index
Monthly_data
```

Out[21]:

	Month	State/UnionTerritory	Cured
0	1	Andaman and Nicobar Islands	151473
1	1	Maharashtra	58313365
2	1	Manipur	865323
3	1	Meghalaya	414228
4	1	Mizoram	130882
...	...	...	...
423	12	Himachal Pradesh	1309890
424	12	Jammu and Kashmir	3414908
425	12	Jharkhand	3388287
426	12	Kerala	19089246
427	12	West Bengal	15314155

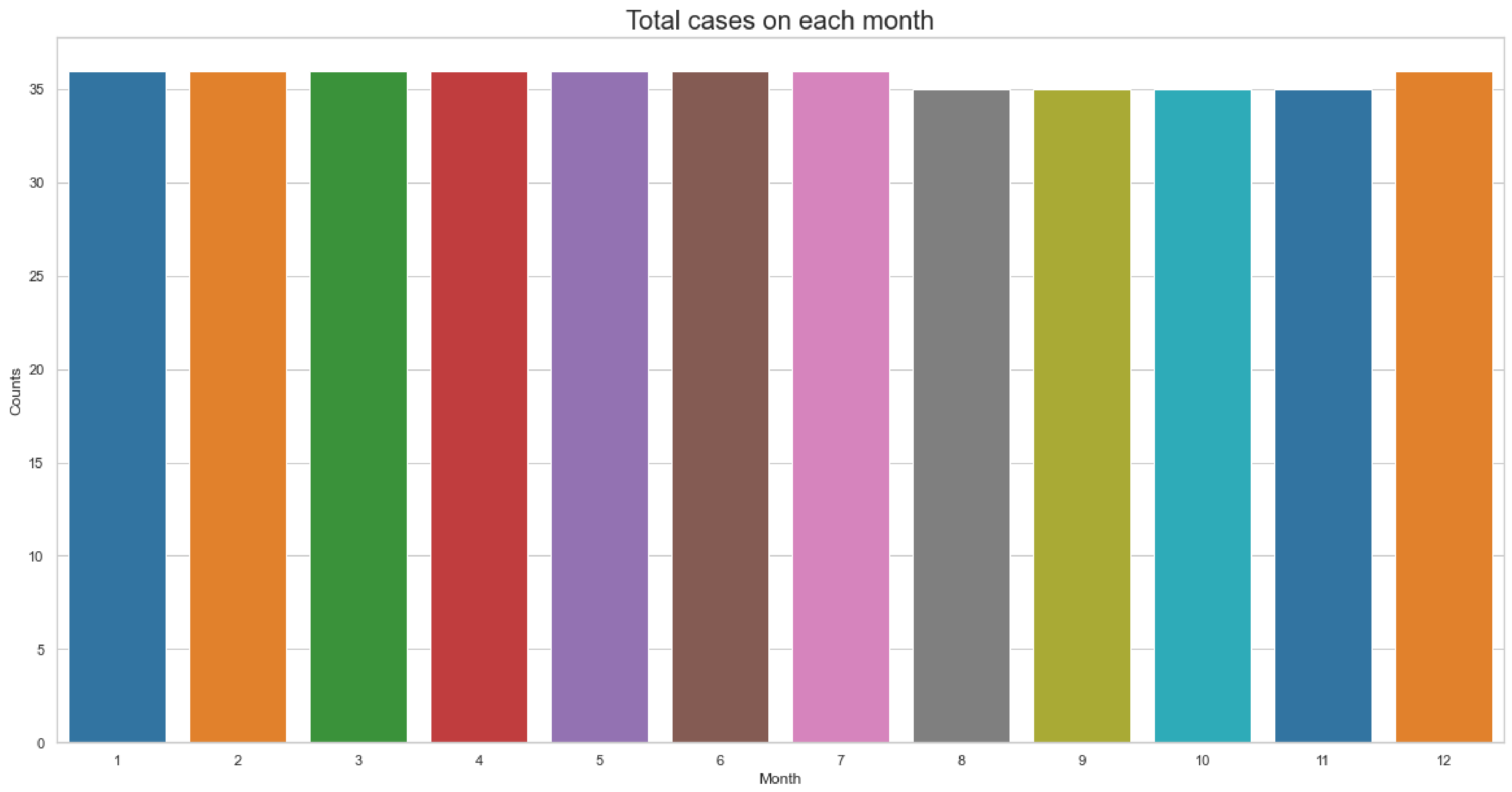
428 rows × 3 columns

In [22]:
```
Month_count =Monthly_data['Month'].value_counts()
Month_count =Month_count.rename_axis('Month').reset_index(name='Counts')
Month_count
```

Out[22]:

	Month	Counts
0	1	36
1	2	36
2	3	36
3	4	36
4	5	36
5	6	36
6	7	36
7	12	36
8	8	35
9	9	35
10	10	35
11	11	35

In [23]:
```
plt.figure(figsize=(20,10))
plt.title('Total cases on each month',size=20)

sns.barplot(data=Month_count, x= 'Month', y='Counts', palette='tab10')
sns.set()
plt.show()
```

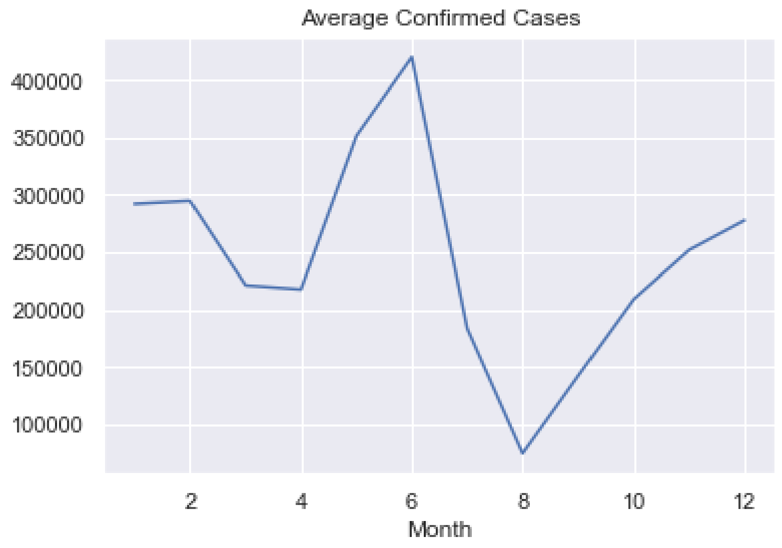


```
In [24]: df.groupby('Month')["Confirmed"].mean()
```

```
Out[24]: Month
         1     292012.297853
         2     294726.990357
         3     221039.217742
         4     217496.566075
         5     351409.776536
         6     419986.480526
         7     183579.364248
         8      74423.612903
         9     142013.102857
         10    209004.895853
         11    251958.487619
         12    277737.208861
         Name: Confirmed, dtype: float64
```

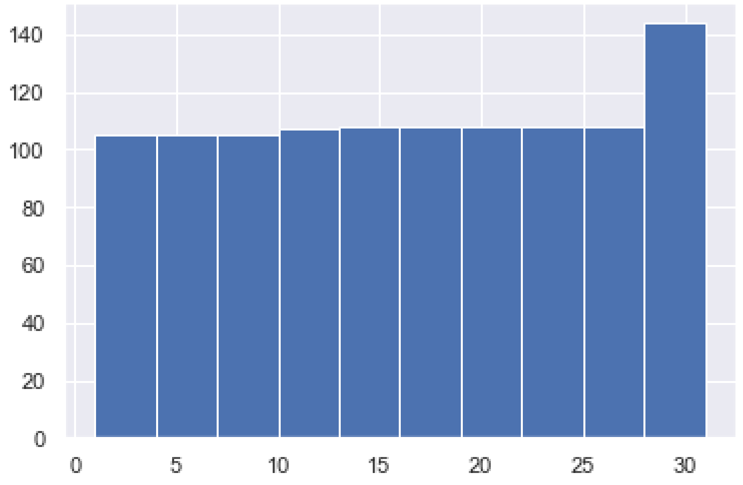
```
In [25]: df.groupby('Month')["Confirmed"].mean().plot(title = 'Average Confirmed Cases')
```

```
Out[25]: <AxesSubplot:title={'center':'Average Confirmed Cases'}, xlabel='Month'>
```



In [26]:
```
df[df.Month == 12].Day.hist()
```

Out[26]:
```
<AxesSubplot:>
```



In [27]:
```
E_year=df[df.Year == 2021]
E_year
```

Out[27]:

	Date	Time	State/UnionTerritory	Cured	Deaths	Confirmed	Day	Month	Year
10082	2021-01-01	8:00 AM	Andhra Pradesh	871916	7108	882286	1	1	2021
10083	2021-01-01	8:00 AM	Andaman and Nicobar Islands	4826	62	4945	1	1	2021
10084	2021-01-01	8:00 AM	Arunachal Pradesh	16564	56	16719	1	1	2021
10085	2021-01-01	8:00 AM	Assam	211910	1045	216211	1	1	2021
10086	2021-01-01	8:00 AM	Bihar	245476	1397	251743	1	1	2021
...	...	...	...	...	...	...	...	...	...
16845	2021-07-07	8:00 AM	Telangana	613124	3703	628282	7	7	2021
16846	2021-07-07	8:00 AM	Tripura	63964	701	68612	7	7	2021
16847	2021-07-07	8:00 AM	Uttarakhand	332006	7338	340882	7	7	2021
16848	2021-07-07	8:00 AM	Uttar Pradesh	1682130	22656	1706818	7	7	2021
16849	2021-07-07	8:00 AM	West Bengal	1472132	17834	1507241	7	7	2021

6768 rows × 9 columns

In [ ]:

In [28]:
```
df["Day"] = df['Date'].dt.day
df["Month"] = df['Date'].dt.month
df["Year"] = df['Date'].dt.year
```

In [29]:
```
Yearly_data= df.groupby(['Year','State/UnionTerritory'])[['Deaths', 'Confirmed',"Cured"]].sum().sort_values(by=['Year','
```

In [30]:
```
Yearly_data
```

Out[30]:

	Year	State/UnionTerritory	Deaths	Confirmed	Cured
0	2020	Lakshadweep	0	0	0
1	2020	Mizoram	319	375091	314163
2	2020	Dadra and Nagar Haveli and Daman and Diu	340	458806	426214
3	2020	Sikkim	8689	521693	444818
4	2020	Andaman and Nicobar Islands	7772	590838	534731
...	...	...	...	...	...
67	2021	Andhra Pradesh	1604638	220012717	208333131
68	2021	Tamil Nadu	3177112	242307447	225565784
69	2021	Karnataka	3410087	288259930	258950406
70	2021	Kerala	1134378	292464927	268176209
71	2021	Maharashtra	13129594	685991838	626754637

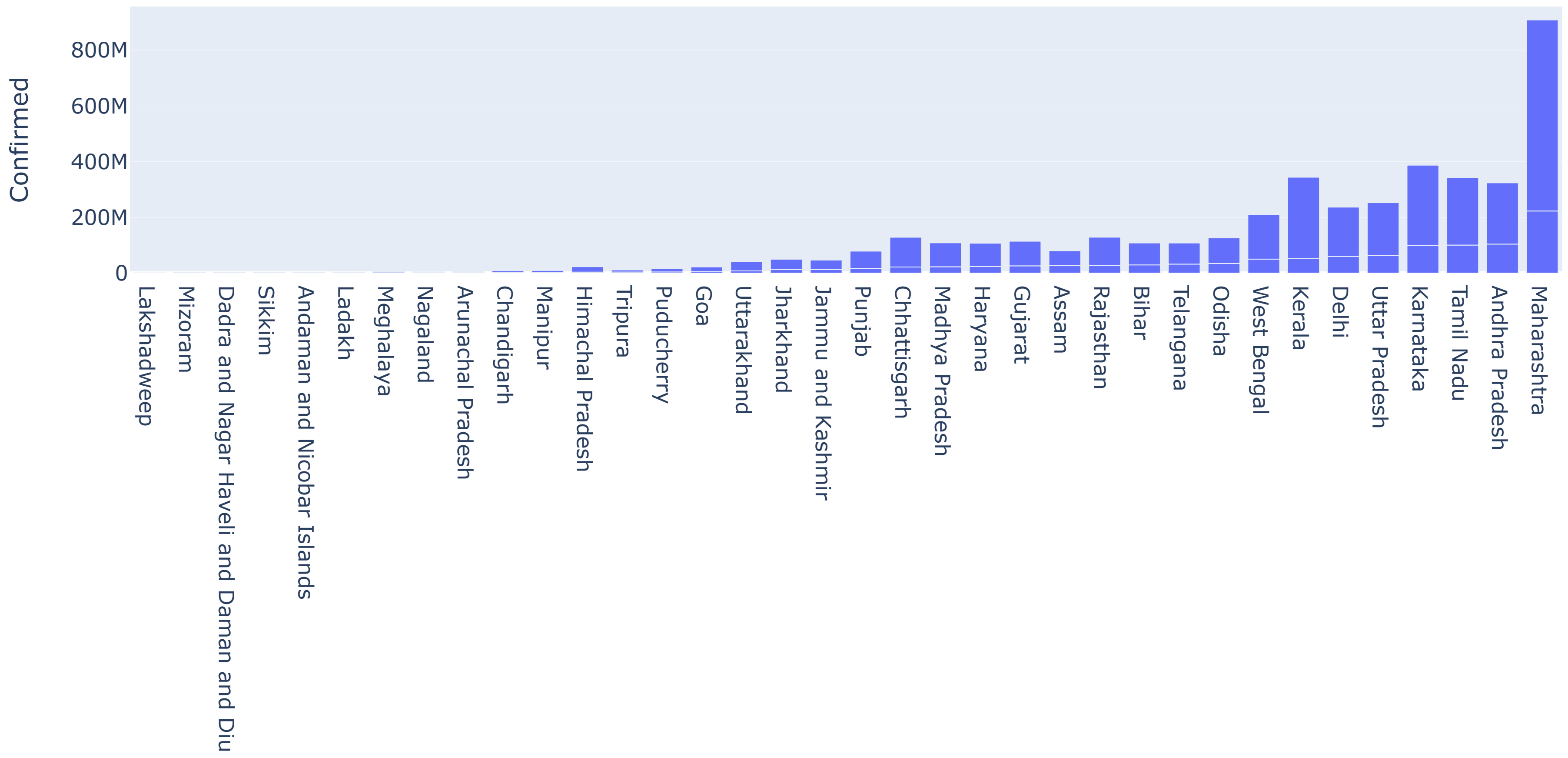
72 rows × 5 columns

In [31]:
```
yearly=Yearly_data.sample(10)
yearly
```

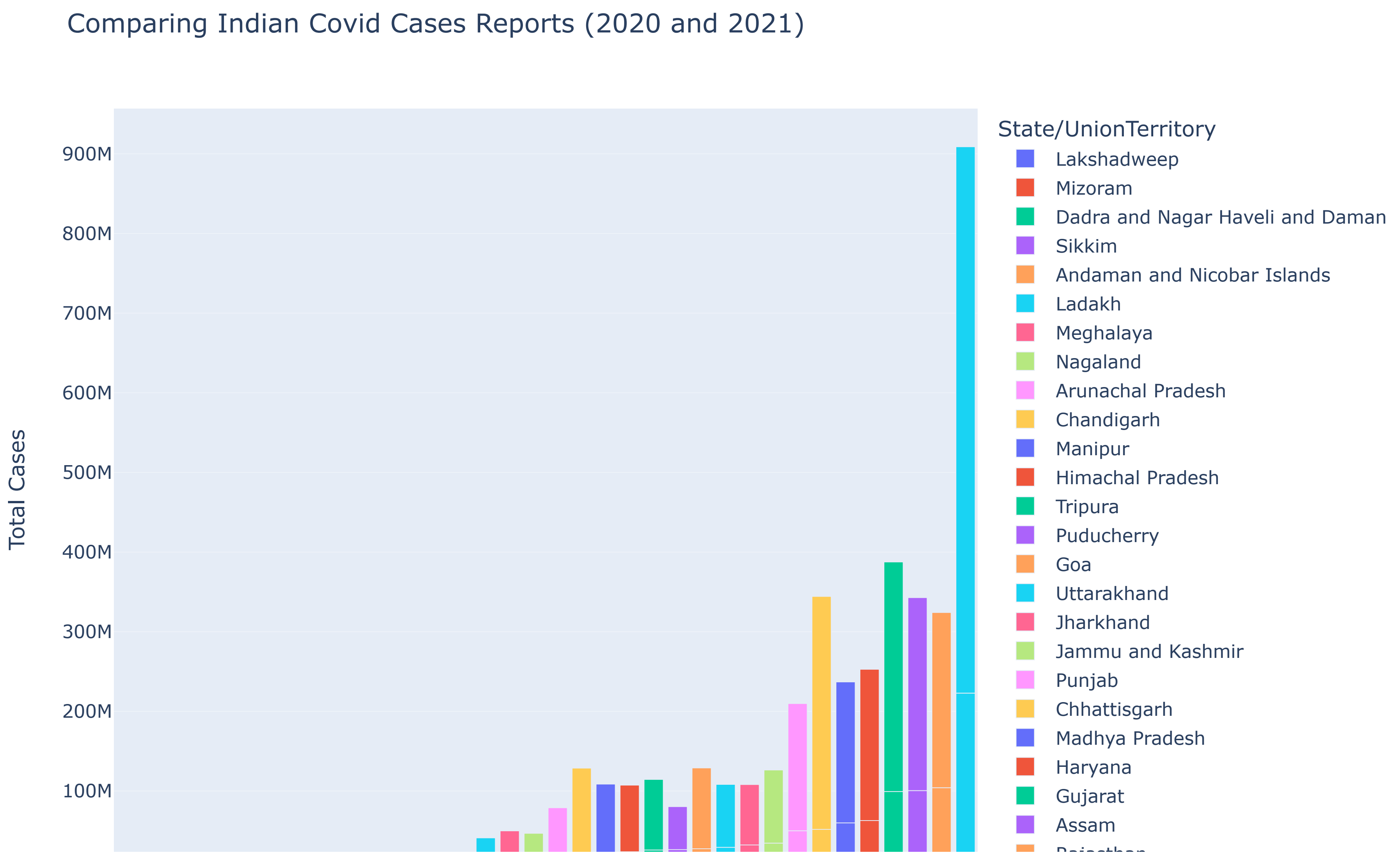
Out[31]:

	Year	State/UnionTerritory	Deaths	Confirmed	Cured
2	2020	Dadra and Nagar Haveli and Daman and Diu	340	458806	426214
68	2021	Tamil Nadu	3177112	242307447	225565784
45	2021	Chandigarh	91886	6869525	6384091
65	2021	Delhi	2910463	177001792	170262449
33	2020	Tamil Nadu	1554515	100522250	91501715
69	2021	Karnataka	3410087	288259930	258950406
64	2021	West Bengal	2273362	159727639	150788352
29	2020	Kerala	193376	51854118	42951434
38	2021	Dadra and Nagar Haveli and Daman and Diu	542	1128764	1065124
24	2020	Rajasthan	287978	27496951	24158911

In [32]:
```
px.bar(data_frame= Yearly_data,x='State/UnionTerritory',hover_name='Year', y ='Confirmed')
```



```
In [33]: px.bar(Yearly_data, x='State/UnionTerritory', y='Confirmed',
               color='State/UnionTerritory',
               hover_name='Year', height=912,
               labels={'Confirmed':'Total Cases'},
               title="Comparing Indian Covid Cases Reports (2020 and 2021) "
               )
```



0
Lakshadweep
Mizoram
Dadra and Nagar Haveli and Daman and
Sikkim
Andaman and Nicobar Islands
Ladakh
Meghalaya
Nagaland
Arunachal Pradesh
Chandigarh
Manipur
Himachal Pradesh
Tripura
Puducherry
Goa
Uttarakhand
Jharkhand
Jammu and Kashmir
Punjab
Chhattisgarh
Madhya Pradesh
Haryana
Gujarat
Assam
Rajasthan
Bihar
Telangana
Odisha
West Bengal
Kerala
Delhi
Uttar Pradesh
Karnataka
Tamil Nadu
Andhra Pradesh
Maharashtra
Rajasthan

```
In [34]: Jan_2021=df[(df['Year'] ==2021) & (df['Month']==1)].groupby('State/UnionTerritory')[['Confirmed','Cured', 'Deaths',]].su
```

In [35]:
```
Jan_2021
```

Out[35]:

	State/UnionTerritory	Confirmed	Cured	Deaths
0	Andaman and Nicobar Islands	154187	151473	1922
1	Andhra Pradesh	27448884	27160550	221186
2	Arunachal Pradesh	520415	516950	1736
3	Assam	6718515	6600118	33089
4	Bihar	7952001	7800177	44922
5	Chandigarh	632798	615729	10173
6	Chhattisgarh	9047658	8712058	109600
7	Dadra and Nagar Haveli and Daman and Diu	104994	104703	62
8	Delhi	19566622	19139813	332233
9	Goa	1619005	1569499	23325
10	Gujarat	7880572	7537895	134971
11	Haryana	8233403	8074769	92110
12	Himachal Pradesh	1756015	1696881	29793
13	Jammu and Kashmir	3813238	3701427	59336
14	Jharkhand	3633956	3564069	32574
15	Karnataka	28833752	28192578	376925
16	Kerala	26008754	23797725	105654
17	Ladakh	298640	290444	3969
18	Lakshadweep	702	44	0
19	Madhya Pradesh	7747005	7434380	115640
20	Maharashtra	61433195	58313365	1559536
21	Manipur	890695	865323	11322
22	Meghalaya	422833	414228	4445
23	Mizoram	133505	130882	269

	State/UnionTerritory	Confirmed	Cured	Deaths
24	Nagaland	372584	366315	2635
25	Odisha	10312437	10199711	58761
26	Puducherry	1196248	1166654	19844
27	Punjab	5263530	5015847	169959
28	Rajasthan	9733025	9476841	84943
29	Sikkim	186334	173952	4040
30	Tamil Nadu	25697977	25122063	379745
31	Telangana	9024439	8847617	48803
32	Tripura	1032922	1019367	12063
33	Uttar Pradesh	18433015	17870678	264524
34	Uttarakhand	2918219	2787733	49344
35	West Bengal	17447673	16900161	309714

```
In [36]: feb_2021 = df[(df['Year'] == 2021) & (df['Month'] ==2)].groupby('State/UnionTerritory')[['Cured', 'Deaths', 'Confirmed']
         feb_2021
```

Out[36]:

	State/UnionTerritory	Cured	Deaths	Confirmed
0	Andaman and Nicobar Islands	138309	1736	140209
1	Andhra Pradesh	24663119	200548	24886770
2	Arunachal Pradesh	469590	1568	471312
3	Assam	6007488	30450	6084562
4	Bihar	7257708	42664	7317092
5	Chandigarh	580561	9637	595366
6	Chhattisgarh	8449346	105517	8653333
7	Dadra and Nagar Haveli and Daman and Diu	95075	56	95209
8	Delhi	17497348	304835	17833650
9	Goa	1478330	21827	1517445
10	Gujarat	7242705	123185	7425825
11	Haryana	7423157	84987	7533997
12	Himachal Pradesh	1589217	27703	1626357
13	Jammu and Kashmir	3436195	54554	3509838
14	Jharkhand	3297004	30279	3340227
15	Karnataka	25951947	343417	26460582
16	Kerala	26076321	111157	27930632
17	Ladakh	268325	3640	273521
18	Lakshadweep	3925	3	6125
19	Madhya Pradesh	7051806	107390	7215130
20	Maharashtra	55303793	1442941	57992941
21	Manipur	804262	10434	816952
22	Meghalaya	382844	4134	389242

	State/UnionTerritory	Cured	Deaths	Confirmed
23	Mizoram	122141	265	123070
24	Nagaland	336793	2498	340686
25	Odisha	9336655	53504	9410919
26	Puducherry	1078459	18426	1103823
27	Punjab	4709964	159678	4937795
28	Rajasthan	8809340	77796	8927168
29	Sikkim	165763	3779	171269
30	Tamil Nadu	23187080	347779	23654321
31	Telangana	8209677	45290	8318653
32	Tripura	922872	10948	934322
33	Uttar Pradesh	16520422	243563	16856451
34	Uttarakhand	2641711	46912	2708370
35	West Bengal	15624550	286334	16028639

In [37]:
```
All_months = df[(df['Year'] == 2021)].groupby('State/UnionTerritory')[['Cured', 'Deaths', 'Confirmed']].sum().reset_inde
All_months
```

Out[37]:

	State/UnionTerritory	Cured	Deaths	Confirmed
0	Andaman and Nicobar Islands	1055204	14852	1084410
1	Andhra Pradesh	208333131	1604638	220012717
2	Arunachal Pradesh	3707750	14815	3918816
3	Assam	50452531	351525	53978391
4	Bihar	74580328	623485	79013525
5	Chandigarh	6384091	91886	6869525
6	Chhattisgarh	98637274	1350909	106849807
7	Dadra and Nagar Haveli and Daman and Diu	1065124	542	1128764
8	Delhi	170262449	2910463	177001792
9	Goa	15446939	267330	16931291
10	Gujarat	81619153	1198983	88613319
11	Haryana	78777999	910098	83703959
12	Himachal Pradesh	17862401	320714	19558473
13	Jammu and Kashmir	31613216	488275	34435422
14	Jharkhand	34995679	459343	37576688
15	Karnataka	258950406	3410087	288259930
16	Kerala	268176209	1134378	292464927
17	Ladakh	2319647	28092	2459725
18	Lakshadweep	471712	2178	561459
19	Madhya Pradesh	80352580	1012730	86099411
20	Maharashtra	626754637	13129594	685991838
21	Manipur	6457463	100291	7046618
22	Meghalaya	3667844	55977	4082500

	State/UnionTerritory	Cured	Deaths	Confirmed
23	Mizoram	1220467	4754	1447099
24	Nagaland	2650666	34280	2911123
25	Odisha	86348435	432006	91841855
26	Puducherry	10733374	178371	11694287
27	Punjab	56231059	1694819	61942495
28	Rajasthan	93153861	871845	101501150
29	Sikkim	1539081	32841	1793826
30	Tamil Nadu	225565784	3177112	242307447
31	Telangana	71676818	420988	75887876
32	Tripura	7158306	83818	7583078
33	Uttar Pradesh	177050153	2411300	189954972
34	Uttarakhand	29819818	606811	33219139
35	West Bengal	150788352	2273362	159727639

In [38]:
```
All_months['Cure-Percentage']=All_months['Cured']/All_months['Confirmed']*100
```

In [39]:
```
All_months['Death-Percentage']=All_months['Deaths']/All_months['Confirmed']*100
```

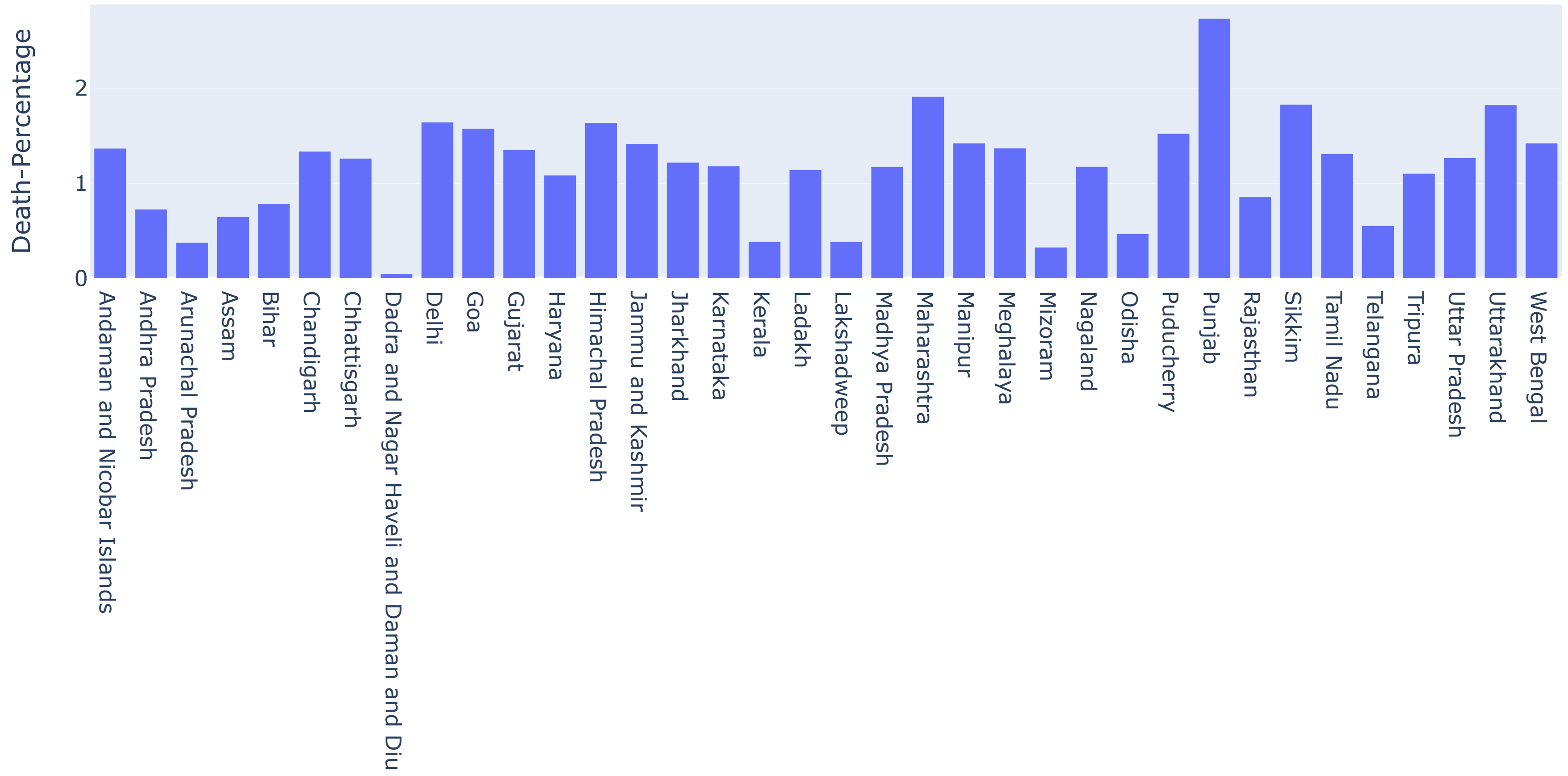
In [40]:
```
All_months
```

Out[40]:

	State/UnionTerritory	Cured	Deaths	Confirmed	Cure-Percentage	Death-Percentage
0	Andaman and Nicobar Islands	1055204	14852	1084410	97.306738	1.369593
1	Andhra Pradesh	208333131	1604638	220012717	94.691404	0.729339
2	Arunachal Pradesh	3707750	14815	3918816	94.614036	0.378048
3	Assam	50452531	351525	53978391	93.468016	0.651233
4	Bihar	74580328	623485	79013525	94.389319	0.789086
5	Chandigarh	6384091	91886	6869525	92.933514	1.337589
6	Chhattisgarh	98637274	1350909	106849807	92.313947	1.264306
7	Dadra and Nagar Haveli and Daman and Diu	1065124	542	1128764	94.361975	0.048017
8	Delhi	170262449	2910463	177001792	96.192500	1.644313
9	Goa	15446939	267330	16931291	91.233084	1.578911
10	Gujarat	81619153	1198983	88613319	92.107094	1.353051
11	Haryana	78777999	910098	83703959	94.115021	1.087282
12	Himachal Pradesh	17862401	320714	19558473	91.328198	1.639770
13	Jammu and Kashmir	31613216	488275	34435422	91.804352	1.417944
14	Jharkhand	34995679	459343	37576688	93.131356	1.222415
15	Karnataka	258950406	3410087	288259930	89.832259	1.182990
16	Kerala	268176209	1134378	292464927	91.695169	0.387868
17	Ladakh	2319647	28092	2459725	94.305136	1.142079
18	Lakshadweep	471712	2178	561459	84.015396	0.387918
19	Madhya Pradesh	80352580	1012730	86099411	93.325354	1.176233
20	Maharashtra	626754637	13129594	685991838	91.364737	1.913958
21	Manipur	6457463	100291	7046618	91.639181	1.423250
22	Meghalaya	3667844	55977	4082500	89.843086	1.371145
23	Mizoram	1220467	4754	1447099	84.338874	0.328519

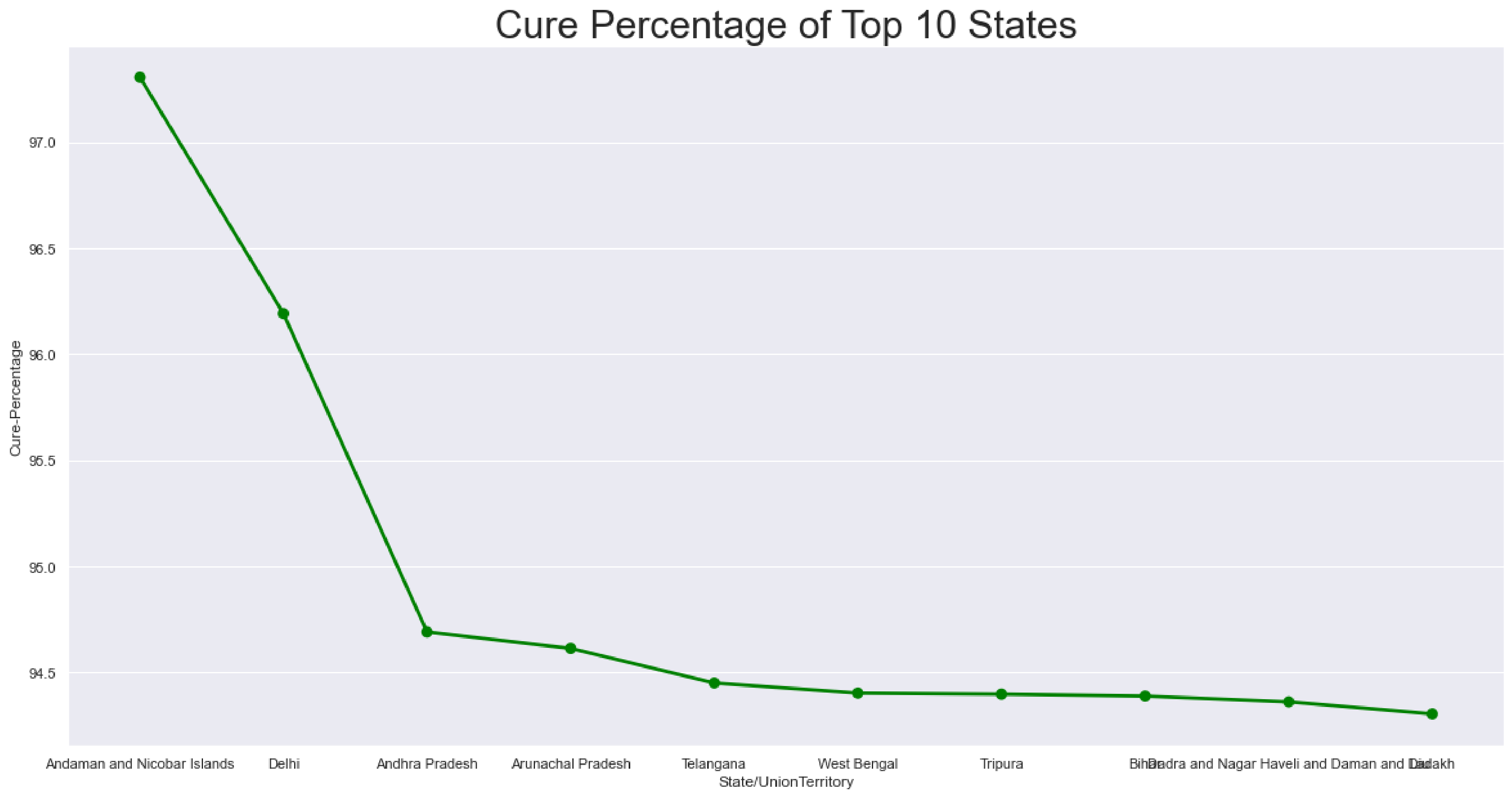
	State/UnionTerritory	Cured	Deaths	Confirmed	Cure-Percentage	Death-Percentage
24	Nagaland	2650666	34280	2911123	91.053040	1.177552
25	Odisha	86348435	432006	91841855	94.018609	0.470380
26	Puducherry	10733374	178371	11694287	91.783056	1.525283
27	Punjab	56231059	1694819	61942495	90.779454	2.736117
28	Rajasthan	93153861	871845	101501150	91.776163	0.858951
29	Sikkim	1539081	32841	1793826	85.798790	1.830780
30	Tamil Nadu	225565784	3177112	242307447	93.090735	1.311190
31	Telangana	71676818	420988	75887876	94.450948	0.554750
32	Tripura	7158306	83818	7583078	94.398422	1.105330
33	Uttar Pradesh	177050153	2411300	189954972	93.206380	1.269406
34	Uttarakhand	29819818	606811	33219139	89.766980	1.826691
35	West Bengal	150788352	2273362	159727639	94.403419	1.423274

In [41]:
```
px.bar(data_frame= All_months,x='State/UnionTerritory',hover_name='Cure-Percentage', y ='Death-Percentage')
```

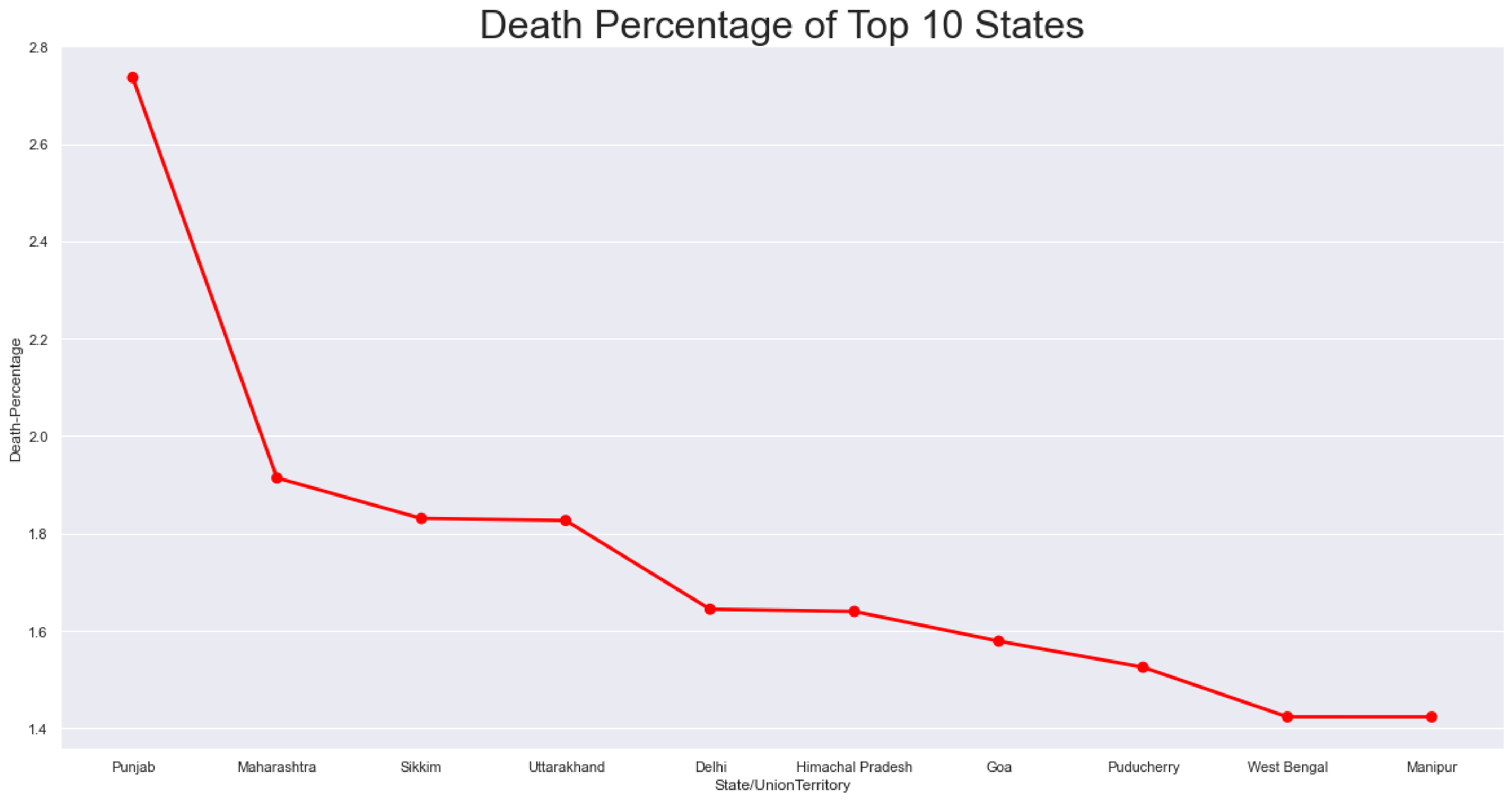


In [42]:
```
max_c=All_months.sort_values(by = "Cure-Percentage" , ascending = False).head(10)
max_d=All_months.sort_values(by = "Death-Percentage" , ascending = False).head(10)
```

In [43]:
```
plt.figure(figsize=(20,10))
plt.title('Cure Percentage of Top 10 States',size=30)

sns.pointplot(data=max_c, x='State/UnionTerritory', y='Cure-Percentage', color="Green")
sns.set()
plt.show()
```

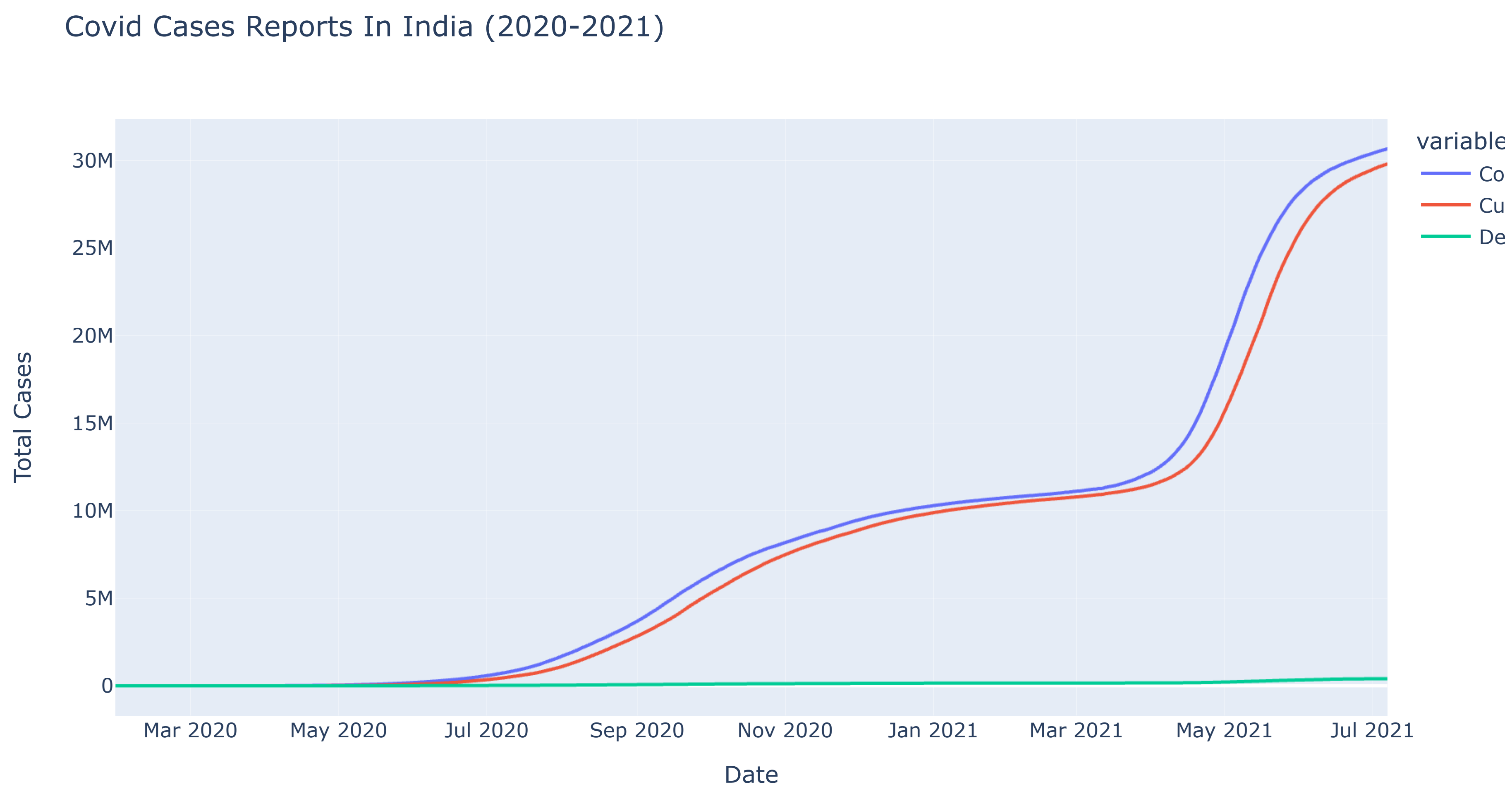


```
In [44]: plt.figure(figsize=(20,10))
         plt.title('Death Percentage of Top 10 States',size=30)

         sns.pointplot(data=max_d, x='State/UnionTerritory', y='Death-Percentage', color="Red")
         sns.set()
         plt.show()
```



In [45]:
```
px.line(df.groupby('Date')['Confirmed','Cured','Deaths'].sum().reset_index(),
        x='Date', y=['Confirmed','Cured','Deaths'],
        labels={'value':'Total Cases'},
        title='Covid Cases Reports In India (2020-2021)', height=540)
```

```
C:\Users\Admin\AppData\Local\Temp/ipykernel_6380/1869704013.py:1: FutureWarning:

Indexing with multiple keys (implicitly converted to a tuple of keys) will be deprecated, use a list instead.
```



In [46]:
```
df.columns
```

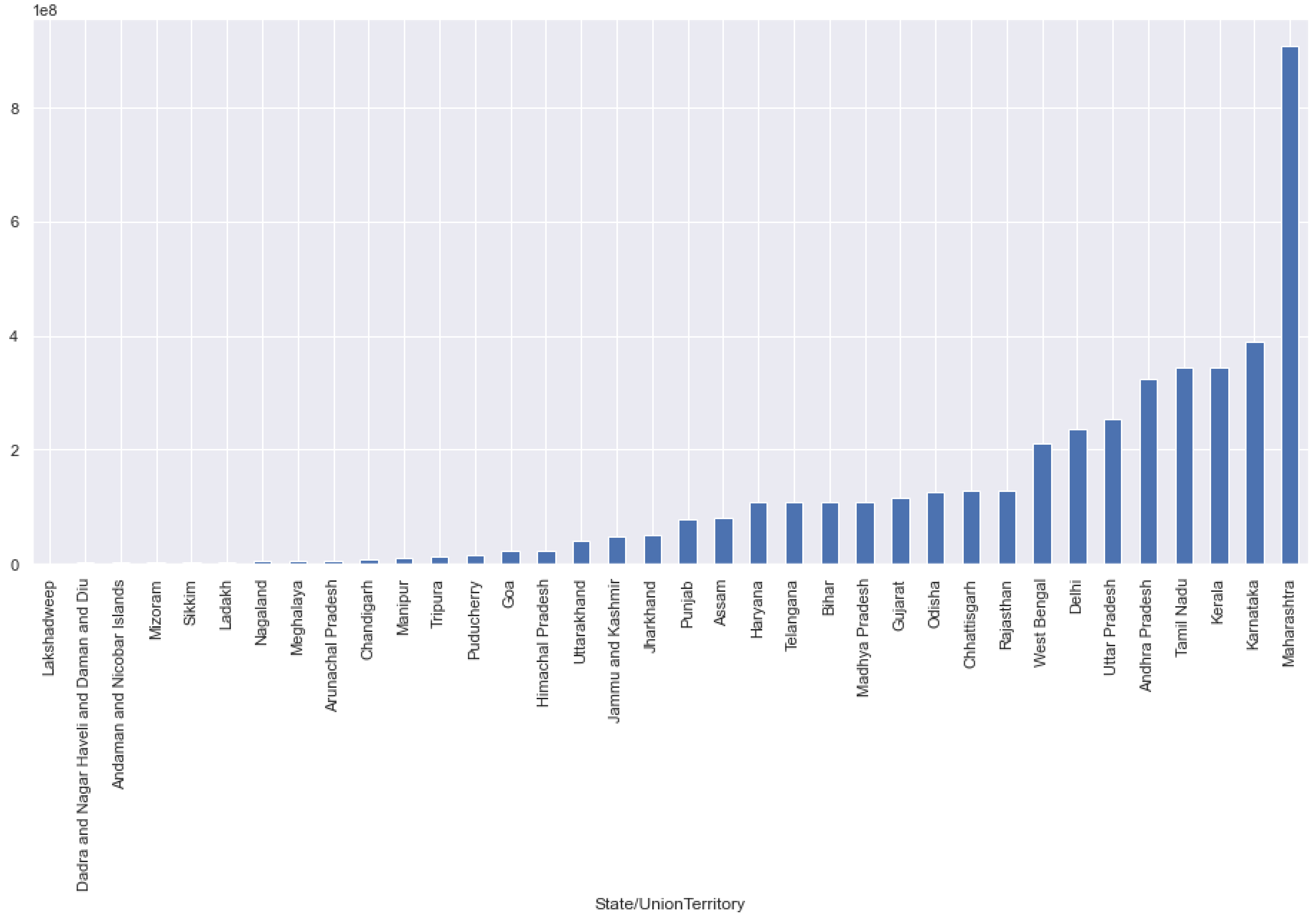
Out[46]:
```
Index(['Date', 'Time', 'State/UnionTerritory', 'Cured', 'Deaths', 'Confirmed',
       'Day', 'Month', 'Year'],
      dtype='object')
```

In [47]:
```
Statedata=df.groupby('State/UnionTerritory')['Confirmed'].sum().sort_values() #Check which state has maximum number of c
```

In [48]:
```
Statedata
```

Out[48]:
```
State/UnionTerritory
Lakshadweep                                    561459
Dadra and Nagar Haveli and Daman and Diu      1587570
Andaman and Nicobar Islands                   1675248
Mizoram                                       1822190
Sikkim                                        2315519
Ladakh                                        3344131
Nagaland                                      4089547
Meghalaya                                     5221064
Arunachal Pradesh                             5598324
Chandigarh                                    8691806
Manipur                                       9440912
Tripura                                      11397656
Puducherry                                   15858688
Goa                                          22280065
Himachal Pradesh                             23052151
Uttarakhand                                  41179396
Jammu and Kashmir                            46899925
Jharkhand                                    49971564
Punjab                                       78999515
Assam                                        80418492
Haryana                                     107408371
Telangana                                   108152726
Bihar                                       108312449
Madhya Pradesh                              108712983
Gujarat                                     114557615
Odisha                                      126408397
Chhattisgarh                                128751782
Rajasthan                                   128998101
West Bengal                                 209822848
Delhi                                       236972842
Uttar Pradesh                               252843682
Andhra Pradesh                              324146783
Tamil Nadu                                  342829697
Kerala                                      344319045
Karnataka                                   387597335
Maharashtra                                 908892470
Name: Confirmed, dtype: int64
```

In [49]:
```
Statedata.plot.bar(figsize = (16,7))
plt.show()
```



```
In [50]: max_count=df.groupby("State/UnionTerritory")[["Cured","Deaths","Confirmed"]].max().reset_index()
```

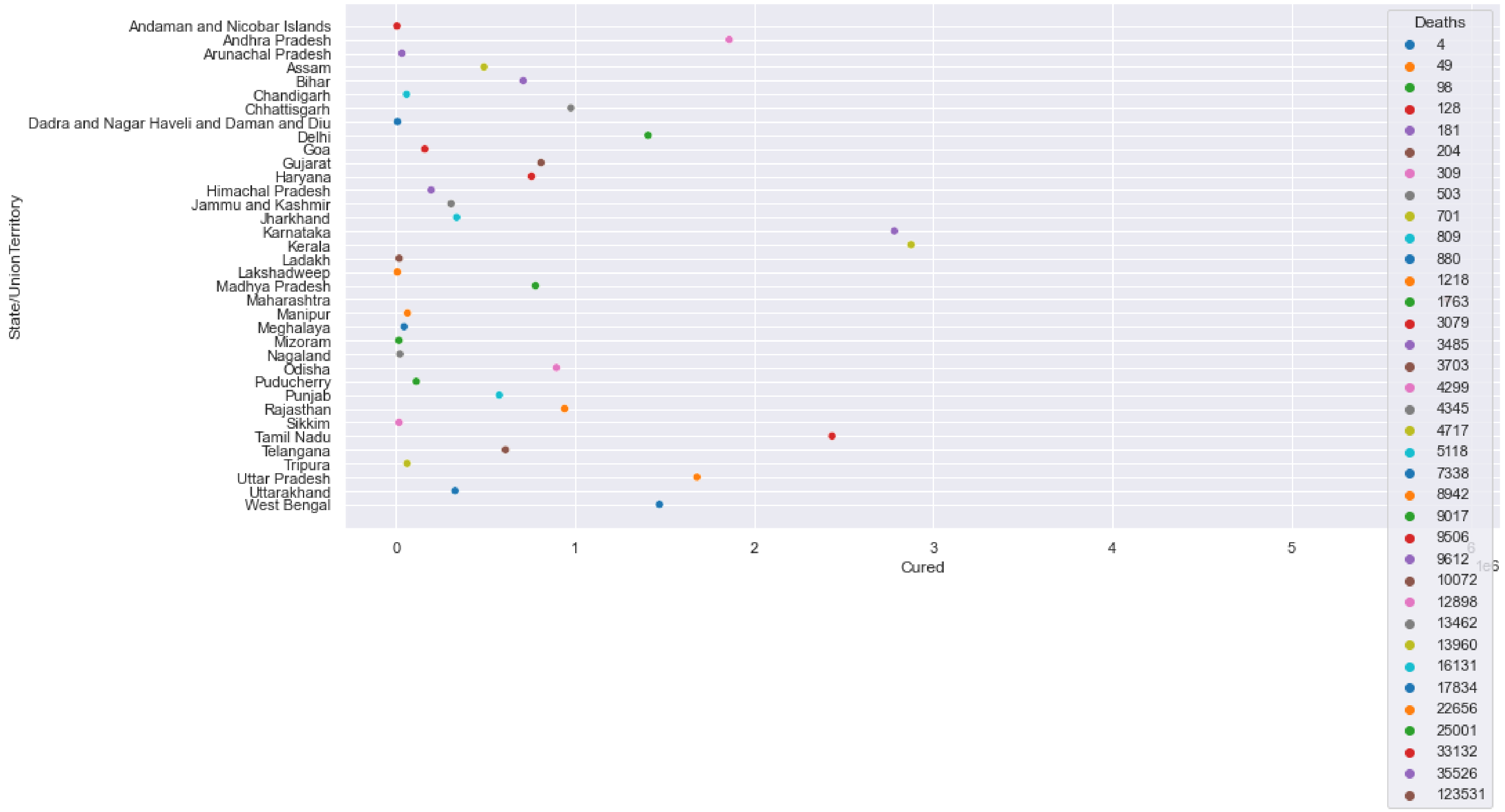
In [51]:
```
max_count
```

Out[51]:

	State/UnionTerritory	Cured	Deaths	Confirmed
0	Andaman and Nicobar Islands	7343	128	7487
1	Andhra Pradesh	1861937	12898	1908065
2	Arunachal Pradesh	34525	181	37879
3	Assam	493306	4717	522267
4	Bihar	711913	9612	722746
5	Chandigarh	60837	809	61752
6	Chhattisgarh	977893	13462	996359
7	Dadra and Nagar Haveli and Daman and Diu	10532	4	10575
8	Delhi	1408853	25001	1434687
9	Goa	162787	3079	167823
10	Gujarat	811699	10072	823964
11	Haryana	758442	9506	769030
12	Himachal Pradesh	198134	3485	202945
13	Jammu and Kashmir	309554	4345	317481
14	Jharkhand	340365	5118	346038
15	Karnataka	2784030	35526	2859595
16	Kerala	2877557	13960	2996094
17	Ladakh	19733	204	20137
18	Lakshadweep	9643	49	9947
19	Madhya Pradesh	780578	9017	790042
20	Maharashtra	5872268	123531	6113335
21	Manipur	66132	1218	73581
22	Meghalaya	47173	880	52358
23	Mizoram	18383	98	22155

	State/UnionTerritory	Cured	Deaths	Confirmed
24	Nagaland	23982	503	25619
25	Odisha	897362	4299	927186
26	Puducherry	114673	1763	118227
27	Punjab	578590	16131	596736
28	Rajasthan	942882	8942	952836
29	Sikkim	19200	309	21403
30	Tamil Nadu	2435872	33132	2503481
31	Telangana	613124	3703	628282
32	Tripura	63964	701	68612
33	Uttar Pradesh	1682130	22656	1706818
34	Uttarakhand	332006	7338	340882
35	West Bengal	1472132	17834	1507241

In [52]:
```
fig = plt.figure(figsize=(15,7))
sns.scatterplot(data=max_count,x= 'Cured',hue='Deaths', y ='State/UnionTerritory',palette='tab10')
plt.show()
```



In [53]:
```
max_count.sort_values(by='Deaths' ,ascending= False)
```

Out[53]:

	State/UnionTerritory	Cured	Deaths	Confirmed
20	Maharashtra	5872268	123531	6113335
15	Karnataka	2784030	35526	2859595
30	Tamil Nadu	2435872	33132	2503481
8	Delhi	1408853	25001	1434687
33	Uttar Pradesh	1682130	22656	1706818
35	West Bengal	1472132	17834	1507241
27	Punjab	578590	16131	596736
16	Kerala	2877557	13960	2996094
6	Chhattisgarh	977893	13462	996359
1	Andhra Pradesh	1861937	12898	1908065
10	Gujarat	811699	10072	823964
4	Bihar	711913	9612	722746
11	Haryana	758442	9506	769030
19	Madhya Pradesh	780578	9017	790042
28	Rajasthan	942882	8942	952836
34	Uttarakhand	332006	7338	340882
14	Jharkhand	340365	5118	346038
3	Assam	493306	4717	522267
13	Jammu and Kashmir	309554	4345	317481
25	Odisha	897362	4299	927186
31	Telangana	613124	3703	628282
12	Himachal Pradesh	198134	3485	202945
9	Goa	162787	3079	167823
26	Puducherry	114673	1763	118227

	State/UnionTerritory	Cured	Deaths	Confirmed
21	Manipur	66132	1218	73581
22	Meghalaya	47173	880	52358
5	Chandigarh	60837	809	61752
32	Tripura	63964	701	68612
24	Nagaland	23982	503	25619
29	Sikkim	19200	309	21403
17	Ladakh	19733	204	20137
2	Arunachal Pradesh	34525	181	37879
0	Andaman and Nicobar Islands	7343	128	7487
23	Mizoram	18383	98	22155
18	Lakshadweep	9643	49	9947
7	Dadra and Nagar Haveli and Daman and Diu	10532	4	10575

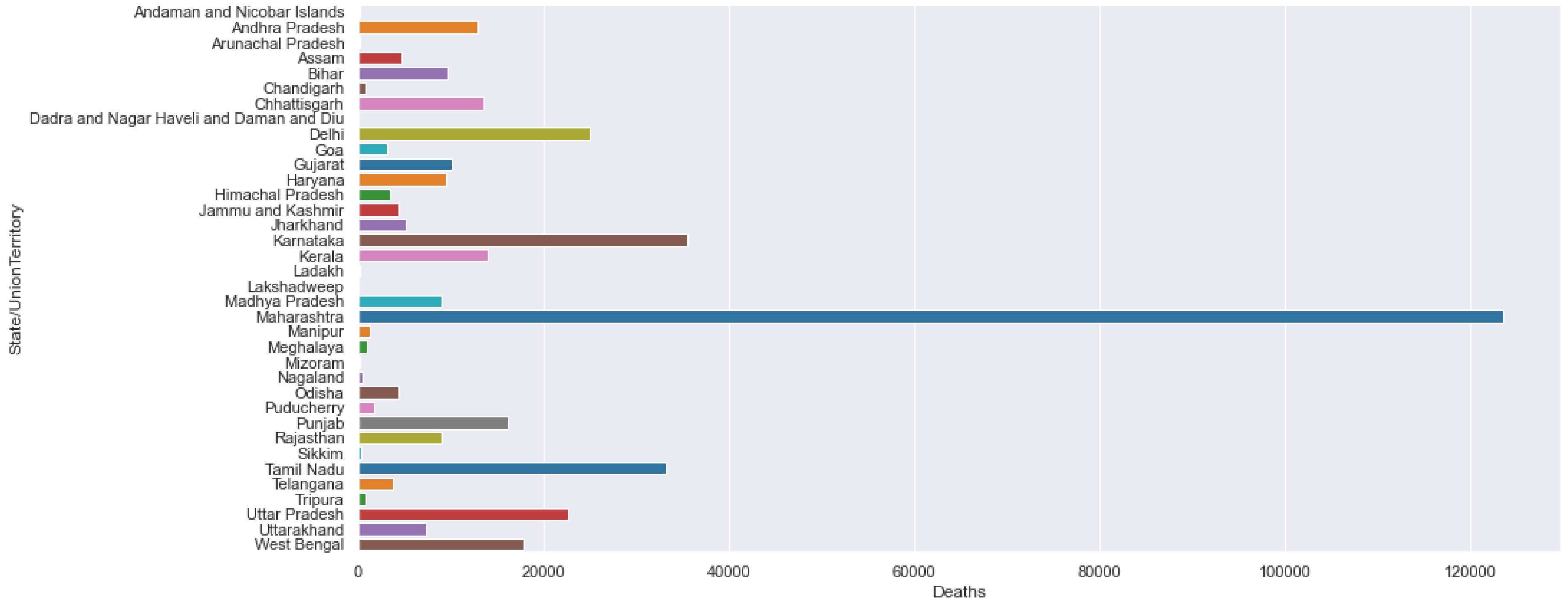
In [54]:
```
max_count.sort_values(by='Cured' ,ascending= False)
```

Out[54]:

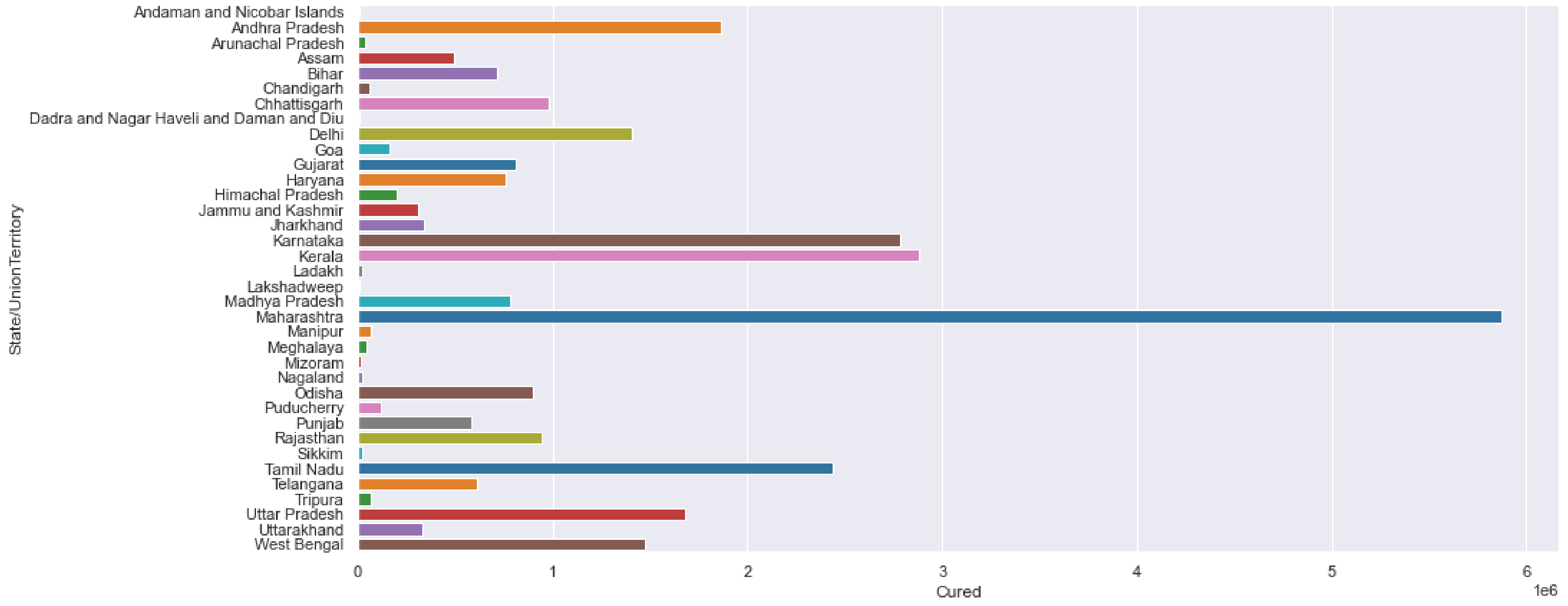
	State/UnionTerritory	Cured	Deaths	Confirmed
20	Maharashtra	5872268	123531	6113335
16	Kerala	2877557	13960	2996094
15	Karnataka	2784030	35526	2859595
30	Tamil Nadu	2435872	33132	2503481
1	Andhra Pradesh	1861937	12898	1908065
33	Uttar Pradesh	1682130	22656	1706818
35	West Bengal	1472132	17834	1507241
8	Delhi	1408853	25001	1434687
6	Chhattisgarh	977893	13462	996359
28	Rajasthan	942882	8942	952836
25	Odisha	897362	4299	927186
10	Gujarat	811699	10072	823964
19	Madhya Pradesh	780578	9017	790042
11	Haryana	758442	9506	769030
4	Bihar	711913	9612	722746
31	Telangana	613124	3703	628282
27	Punjab	578590	16131	596736
3	Assam	493306	4717	522267
14	Jharkhand	340365	5118	346038
34	Uttarakhand	332006	7338	340882
13	Jammu and Kashmir	309554	4345	317481
12	Himachal Pradesh	198134	3485	202945
9	Goa	162787	3079	167823
26	Puducherry	114673	1763	118227

	State/UnionTerritory	Cured	Deaths	Confirmed
21	Manipur	66132	1218	73581
32	Tripura	63964	701	68612
5	Chandigarh	60837	809	61752
22	Meghalaya	47173	880	52358
2	Arunachal Pradesh	34525	181	37879
24	Nagaland	23982	503	25619
17	Ladakh	19733	204	20137
29	Sikkim	19200	309	21403
23	Mizoram	18383	98	22155
7	Dadra and Nagar Haveli and Daman and Diu	10532	4	10575
18	Lakshadweep	9643	49	9947
0	Andaman and Nicobar Islands	7343	128	7487

In [55]:
```
fig = plt.figure(figsize=(15,7))
sns.barplot(data=max_count,x= 'Deaths', y ='State/UnionTerritory', palette='tab10')
plt.show()
```



In [56]:
```
fig = plt.figure(figsize=(15,7))
sns.barplot(data=max_count,x= 'Cured', y ='State/UnionTerritory', palette='tab10')
plt.show()
```



HIGHEST RECORDED DATA

In [57]:
```
max_sum=df.groupby("State/UnionTerritory")[["Cured","Deaths","Confirmed"]].sum().sort_values(by= "Confirmed" , ascending
```

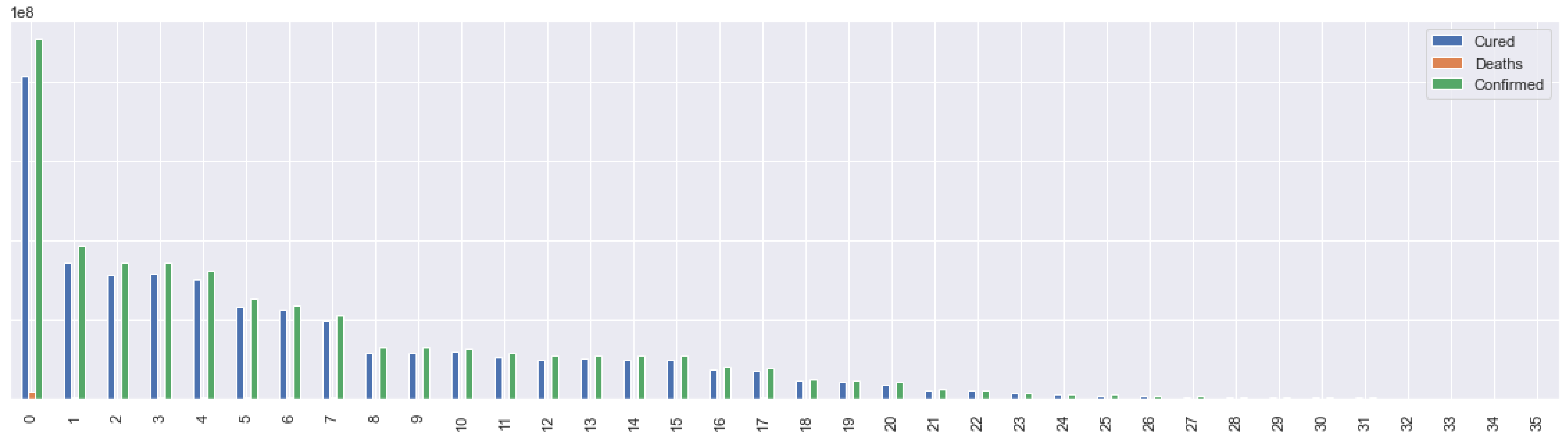
In [58]:
```
max_sum
```

Out[58]:

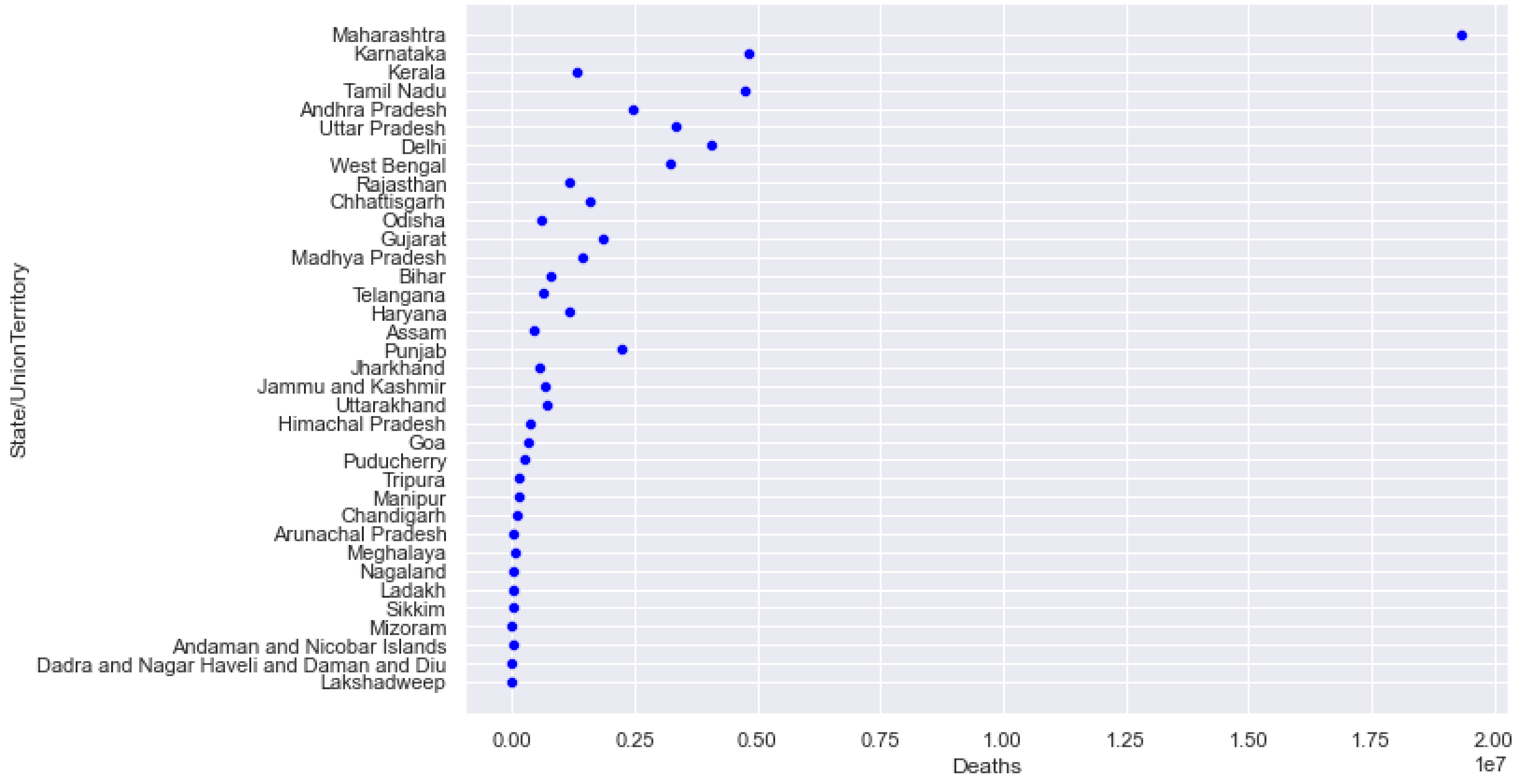
	State/UnionTerritory	Cured	Deaths	Confirmed
0	Maharashtra	813788907	19314532	908892470
1	Karnataka	345648926	4819018	387597335
2	Kerala	311127643	1327754	344319045
3	Tamil Nadu	317067499	4731627	342829697
4	Andhra Pradesh	303427899	2475816	324146783
5	Uttar Pradesh	232529439	3347656	252843682
6	Delhi	224062704	4066907	236972842
7	West Bengal	195296839	3214840	209822848
8	Rajasthan	117312772	1159823	128998101
9	Chhattisgarh	117163544	1591126	128751782
10	Odisha	117984789	600149	126408397
11	Gujarat	103995131	1866811	114557615
12	Madhya Pradesh	100169697	1427780	108712983
13	Bihar	101533848	775163	108312449
14	Telangana	100211245	617882	108152726
15	Haryana	100010131	1166573	107408371
16	Assam	74011348	459575	80418492
17	Punjab	71108712	2216735	78999515
18	Jharkhand	46083978	569298	49971564
19	Jammu and Kashmir	42295048	686680	46899925
20	Uttarakhand	36684388	728512	41179396
21	Himachal Pradesh	20682770	371931	23052151
22	Goa	20224042	338359	22280065
23	Puducherry	14376916	249683	15858688

	State/UnionTerritory	Cured	Deaths	Confirmed
24	Tripura	10479169	124444	11397656
25	Manipur	8420223	122089	9440912
26	Chandigarh	7980284	119356	8691806
27	Arunachal Pradesh	5150519	19303	5598324
28	Meghalaya	4606548	66293	5221064
29	Nagaland	3628619	39420	4089547
30	Ladakh	3059045	38578	3344131
31	Sikkim	1983899	41530	2315519
32	Mizoram	1534630	5073	1822190
33	Andaman and Nicobar Islands	1589935	22624	1675248
34	Dadra and Nagar Haveli and Daman and Diu	1491338	882	1587570
35	Lakshadweep	471712	2178	561459

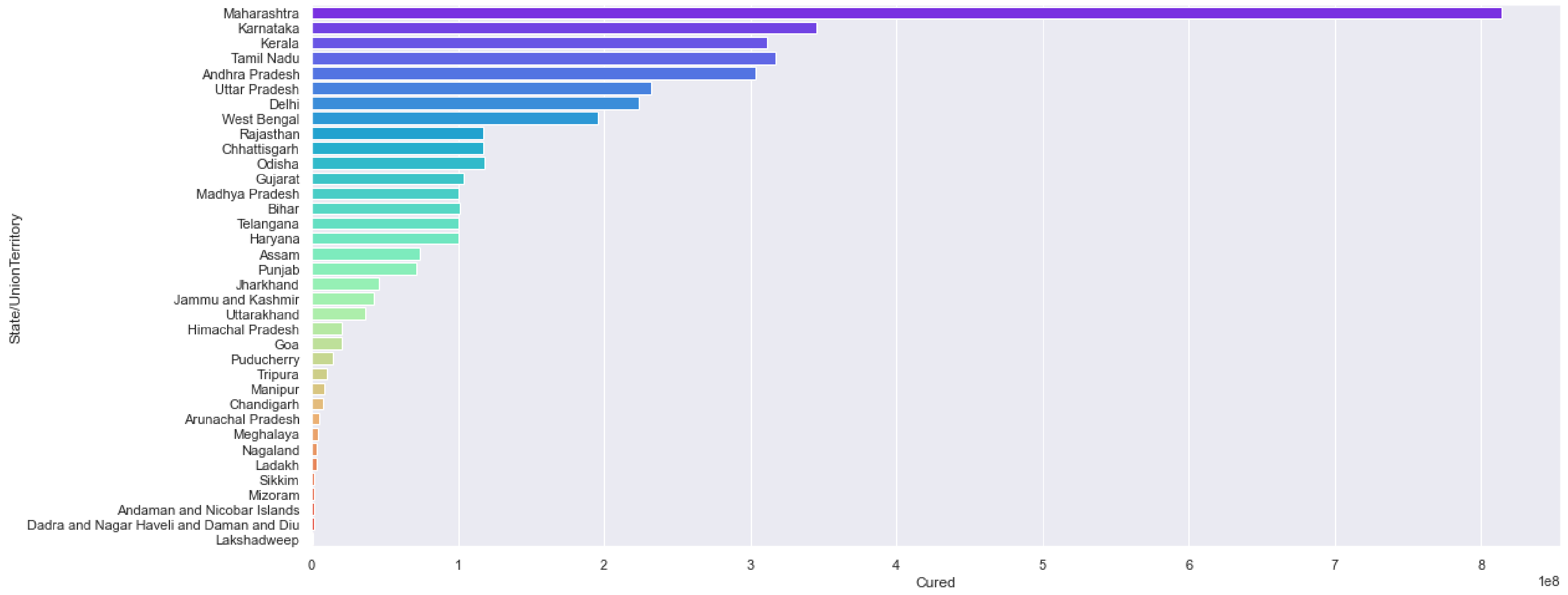
In [59]:
```
max_sum.plot.bar(figsize = (20,5))
plt.show()
```



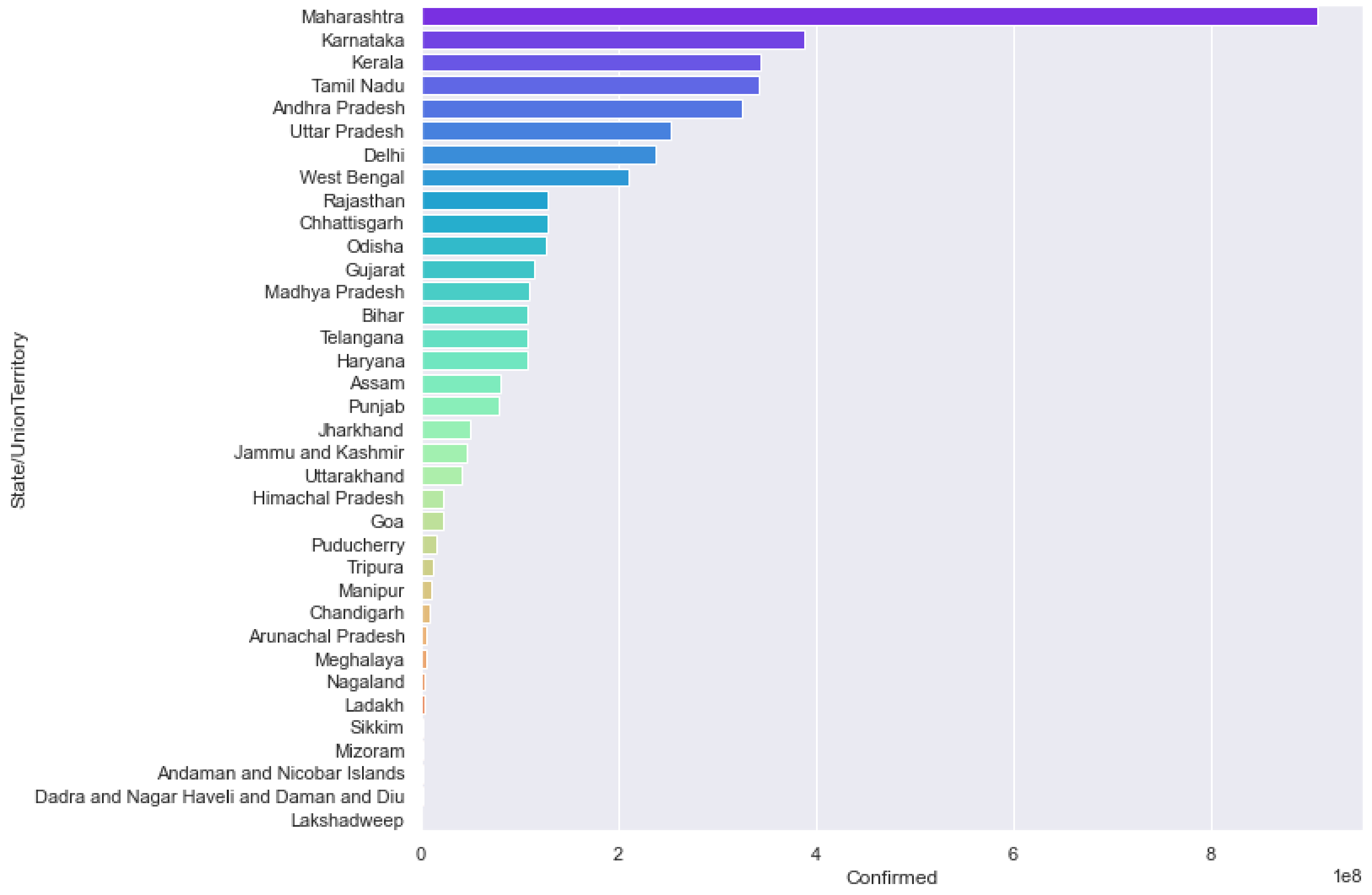
```
In [60]: fig = plt.figure(figsize=(10,7))
         sns.scatterplot(data=max_sum,x ='Deaths',y = 'State/UnionTerritory',color='blue')
         plt.show()
```



In [175]:
```
fig = plt.figure(figsize=(18,8))
sns.barplot(data=max_sum,x ='Cured',y = 'State/UnionTerritory',palette = 'rainbow')
plt.show()
```



```
In [62]: fig = plt.figure(figsize=(10,9))
         sns.barplot(data=max_sum,x ='Confirmed',y = 'State/UnionTerritory',palette = 'rainbow')
         plt.show()
```

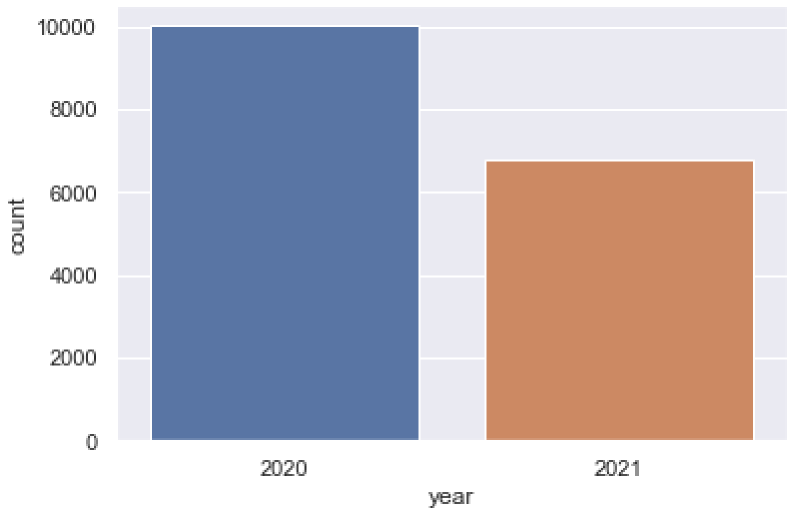


In [63]:
```
df['year']=pd.DatetimeIndex(df['Date']).year
```

In [64]:
```
sns.countplot(df['year'])
```

```
C:\Users\Admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:

Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in an error or misinterpretation.
```

Out[64]:
```
<AxesSubplot:xlabel='year', ylabel='count'>
```



Top 5 States in hited list**

Cured cases in top 5 states

In [65]:
```
df2=df.groupby(['State/UnionTerritory'])['Cured'].sum()
```

In [66]:
```
df2
```

Out[66]:
```
State/UnionTerritory
Andaman and Nicobar Islands                   1589935
Andhra Pradesh                              303427899
Arunachal Pradesh                             5150519
Assam                                        74011348
Bihar                                       101533848
Chandigarh                                    7980284
Chhattisgarh                                117163544
Dadra and Nagar Haveli and Daman and Diu      1491338
Delhi                                       224062704
Goa                                          20224042
Gujarat                                     103995131
Haryana                                     100010131
Himachal Pradesh                             20682770
Jammu and Kashmir                            42295048
Jharkhand                                    46083978
Karnataka                                   345648926
Kerala                                      311127643
Ladakh                                        3059045
Lakshadweep                                    471712
Madhya Pradesh                              100169697
Maharashtra                                 813788907
Manipur                                       8420223
Meghalaya                                     4606548
Mizoram                                       1534630
Nagaland                                      3628619
Odisha                                      117984789
Puducherry                                   14376916
Punjab                                       71108712
Rajasthan                                   117312772
Sikkim                                        1983899
Tamil Nadu                                  317067499
Telangana                                   100211245
Tripura                                      10479169
Uttar Pradesh                               232529439
Uttarakhand                                  36684388
West Bengal                                 195296839
Name: Cured, dtype: int64
```

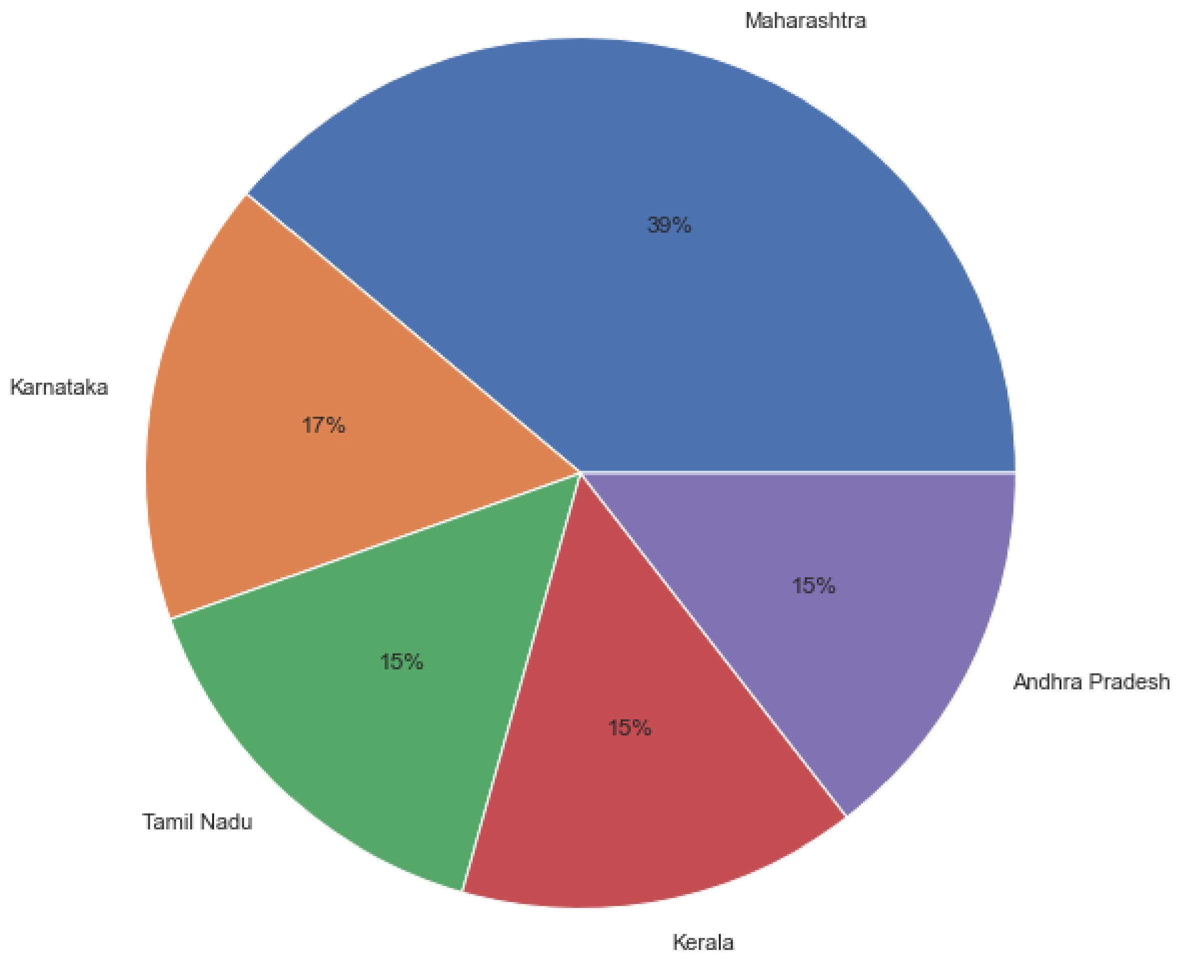
```
In [67]: pie = df2.sort_values(ascending=False).head()
```

```
In [68]: pie
```

```
Out[68]: State/UnionTerritory
         Maharashtra       813788907
         Karnataka         345648926
         Tamil Nadu        317067499
         Kerala            311127643
         Andhra Pradesh    303427899
         Name: Cured, dtype: int64
```

In [69]:
```
plt.figure(figsize=(20,10))
plt.pie(pie, labels=pie.index,autopct = '%0.0f%%')
```

Out[69]:
```
([<matplotlib.patches.Wedge at 0x1e77427a2e0>,
  <matplotlib.patches.Wedge at 0x1e77427aa60>,
  <matplotlib.patches.Wedge at 0x1e7742641c0>,
  <matplotlib.patches.Wedge at 0x1e7742648e0>,
  <matplotlib.patches.Wedge at 0x1e774264fd0>],
 [Text(0.37529252432419213, 1.0339997684653397, 'Maharashtra'),
  Text(-1.082807554452896, 0.19372093335450194, 'Karnataka'),
  Text(-0.7515473573470732, -0.8032288401599079, 'Tamil Nadu'),
  Text(0.20950008853258523, -1.079865599463581, 'Kerala'),
  Text(0.9876671935373039, -0.4842659546263253, 'Andhra Pradesh')],
 [Text(0.20470501326774113, 0.563999873708367, '39%'),
  Text(-0.5906223024288522, 0.10566596364791013, '17%'),
  Text(-0.4099349221893126, -0.43812482190540425, '15%'),
  Text(0.11427277556322829, -0.5890175997074077, '15%'),
  Text(0.5387275601112567, -0.26414506615981376, '15%')])
```


Maharashtra
39%
Karnataka
17%
Tamil Nadu
15%
Kerala
15%
Andhra Pradesh
15%

Deaths rate in top 5 states

In [70]:
```
df3=df.groupby(['State/UnionTerritory'])['Deaths'].sum()
```

In [71]:
```
df3
```

Out[71]:
```
State/UnionTerritory
Andaman and Nicobar Islands                    22624
Andhra Pradesh                               2475816
Arunachal Pradesh                              19303
Assam                                         459575
Bihar                                         775163
Chandigarh                                    119356
Chhattisgarh                                 1591126
Dadra and Nagar Haveli and Daman and Diu         882
Delhi                                        4066907
Goa                                           338359
Gujarat                                      1866811
Haryana                                      1166573
Himachal Pradesh                              371931
Jammu and Kashmir                             686680
Jharkhand                                     569298
Karnataka                                    4819018
Kerala                                       1327754
Ladakh                                         38578
Lakshadweep                                     2178
Madhya Pradesh                               1427780
Maharashtra                                 19314532
Manipur                                       122089
Meghalaya                                      66293
Mizoram                                         5073
Nagaland                                       39420
Odisha                                        600149
Puducherry                                    249683
Punjab                                       2216735
Rajasthan                                    1159823
Sikkim                                         41530
Tamil Nadu                                   4731627
Telangana                                     617882
Tripura                                       124444
Uttar Pradesh                                3347656
Uttarakhand                                   728512
West Bengal                                  3214840
Name: Deaths, dtype: int64
```

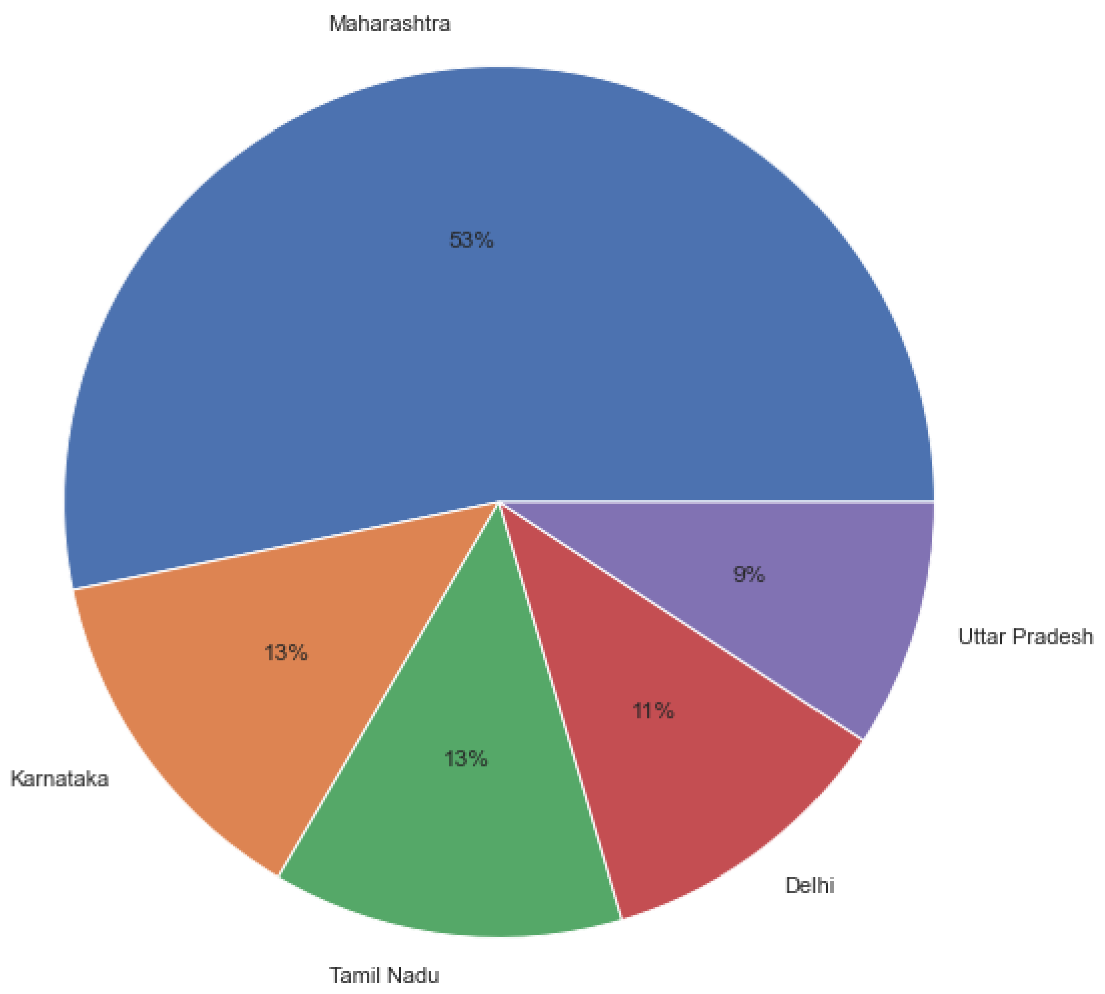
```
In [72]: pie2= df3.sort_values(ascending=False).head()
```

```
In [73]: pie2
```

```
Out[73]: State/UnionTerritory
         Maharashtra      19314532
         Karnataka         4819018
         Tamil Nadu        4731627
         Delhi             4066907
         Uttar Pradesh     3347656
         Name: Deaths, dtype: int64
```

In [74]:
```
plt.figure(figsize=(20,10))
plt.pie(pie2, labels=pie2.index, autopct = '%0.0f%%')
```

Out[74]:
```
([<matplotlib.patches.Wedge at 0x1e773cafa00>,
  <matplotlib.patches.Wedge at 0x1e773fcf220>,
  <matplotlib.patches.Wedge at 0x1e773fcf940>,
  <matplotlib.patches.Wedge at 0x1e773fdc0a0>,
  <matplotlib.patches.Wedge at 0x1e773fdc7c0>],
 [Text(-0.11169713877583874, 1.0943142826397227, 'Maharashtra'),
  Text(-0.8947979665098296, -0.6397941849766017, 'Karnataka'),
  Text(-0.13500207736441258, -1.0916842213329336, 'Tamil Nadu'),
  Text(0.6559062505051835, -0.8830554855433669, 'Delhi'),
  Text(1.0541042486968961, -0.3144268323142166, 'Uttar Pradesh')],
 [Text(-0.0609257120595484, 0.5968986996216669, '53%'),
  Text(-0.48807161809627064, -0.3489786463508736, '13%'),
  Text(-0.07363749674422503, -0.5954641207270546, '13%'),
  Text(0.35776704573010004, -0.4816666284782001, '11%'),
  Text(0.5749659538346705, -0.17150554489866357, '9%')])
```



Confirmed cases in top 5 states

```
In [75]: df4=df.groupby(['State/UnionTerritory'])['Confirmed'].sum()
```

In [76]:
```
df4
```

Out[76]:
```
State/UnionTerritory
Andaman and Nicobar Islands                   1675248
Andhra Pradesh                              324146783
Arunachal Pradesh                             5598324
Assam                                        80418492
Bihar                                       108312449
Chandigarh                                    8691806
Chhattisgarh                                128751782
Dadra and Nagar Haveli and Daman and Diu      1587570
Delhi                                       236972842
Goa                                          22280065
Gujarat                                     114557615
Haryana                                     107408371
Himachal Pradesh                             23052151
Jammu and Kashmir                            46899925
Jharkhand                                    49971564
Karnataka                                   387597335
Kerala                                      344319045
Ladakh                                        3344131
Lakshadweep                                    561459
Madhya Pradesh                              108712983
Maharashtra                                 908892470
Manipur                                       9440912
Meghalaya                                     5221064
Mizoram                                       1822190
Nagaland                                      4089547
Odisha                                      126408397
Puducherry                                   15858688
Punjab                                       78999515
Rajasthan                                   128998101
Sikkim                                        2315519
Tamil Nadu                                  342829697
Telangana                                   108152726
Tripura                                      11397656
Uttar Pradesh                               252843682
Uttarakhand                                  41179396
West Bengal                                 209822848
Name: Confirmed, dtype: int64
```

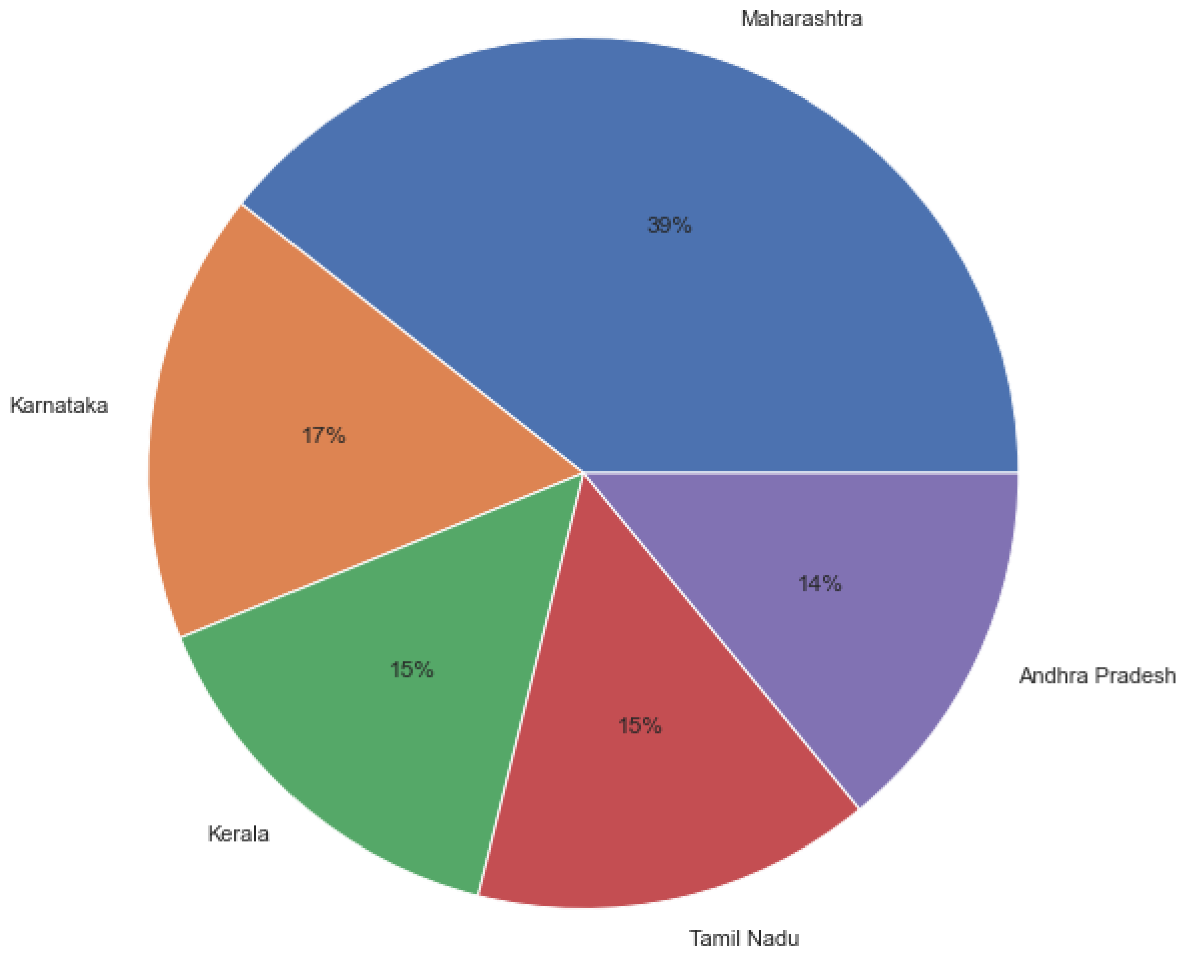
In [77]:
```
pie3= df4.sort_values(ascending=False).head()
```

In [78]:
```
pie3
```

Out[78]:
```
State/UnionTerritory
Maharashtra       908892470
Karnataka         387597335
Kerala            344319045
Tamil Nadu        342829697
Andhra Pradesh    324146783
Name: Confirmed, dtype: int64
```

In [79]:
```
plt.figure(figsize=(20,10))
plt.pie(pie3, labels=pie3.index, autopct = '%0.0f%%')
```

Out[79]:
```
([<matplotlib.patches.Wedge at 0x1e77402a460>,
  <matplotlib.patches.Wedge at 0x1e77402abe0>,
  <matplotlib.patches.Wedge at 0x1e774037340>,
  <matplotlib.patches.Wedge at 0x1e774037a60>,
  <matplotlib.patches.Wedge at 0x1e7740441c0>],
 [Text(0.36010745738549543, 1.039385693155026, 'Maharashtra'),
  Text(-1.0893292566033088, 0.15284557798013923, 'Karnataka'),
  Text(-0.7202141474315729, -0.8314394637250585, 'Kerala'),
  Text(0.24174631982526534, -1.0731070388600294, 'Tamil Nadu'),
  Text(0.9946347841842009, -0.4697889378123418, 'Andhra Pradesh')],
 [Text(0.1964222494829975, 0.5669376508118323, '39%'),
  Text(-0.5941795945108956, 0.08337031526189412, '17%'),
  Text(-0.3928440804172215, -0.4535124347591228, '15%'),
  Text(0.13186162899559925, -0.5853311121054705, '15%'),
  Text(0.5425280641004732, -0.2562485115340046, '14%')])
```

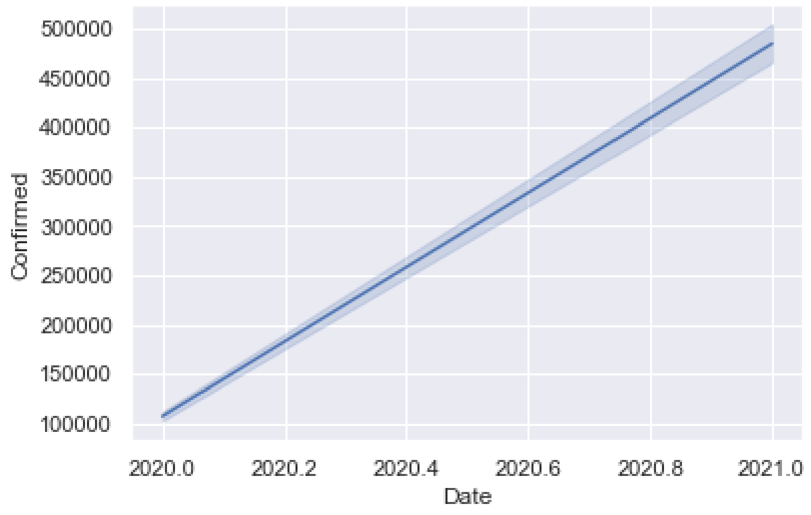

Maharashtra
39%
Karnataka
17%
15%
15%
14%
Andhra Pradesh
Kerala
Tamil Nadu

```
In [ ]:
```

Ploting with Date wise year

In [80]:
```
sns.lineplot(data=df, x=pd.DatetimeIndex(df['Date']).year, y="Confirmed")
```

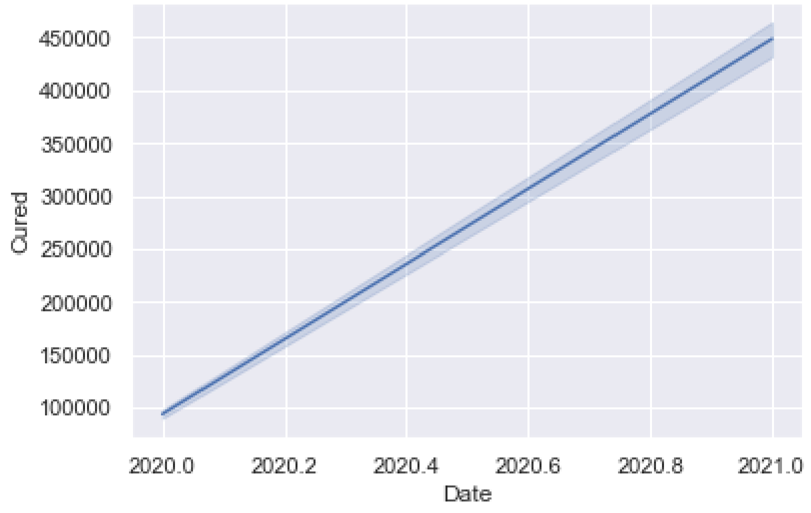
Out[80]:
```
<AxesSubplot:xlabel='Date', ylabel='Confirmed'>
```



In [ ]:

In [81]:
```
sns.lineplot(data=df, x=pd.DatetimeIndex(df['Date']).year, y="Cured")
```

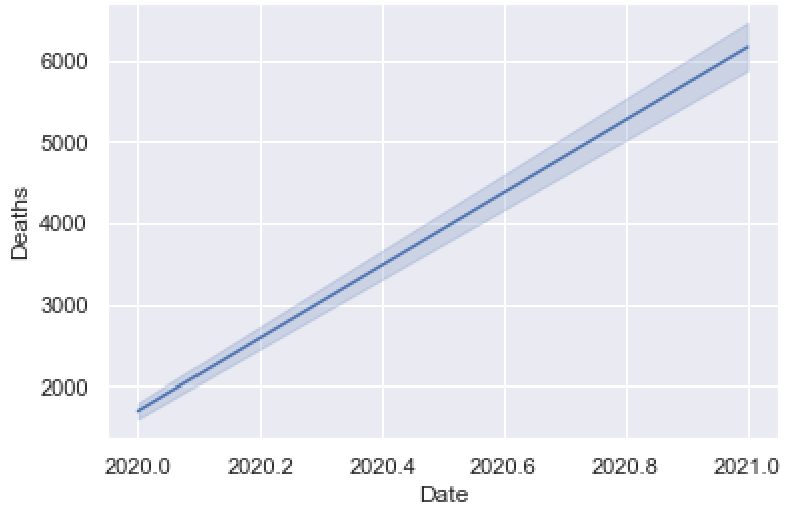
Out[81]:
```
<AxesSubplot:xlabel='Date', ylabel='Cured'>
```



In [ ]:

In [82]:
```
sns.lineplot(data=df, x=pd.DatetimeIndex(df['Date']).year, y="Deaths")
```

Out[82]:
```
<AxesSubplot:xlabel='Date', ylabel='Deaths'>
```



In [ ]:

Extracting Maharashtra Data

In [83]:
```
Maharashtra_data=df[df['State/UnionTerritory'] == 'Maharashtra']
```

In [84]:
```
Maharashtra_data
```

Out[84]:

	Date	Time	State/UnionTerritory	Cured	Deaths	Confirmed	Day	Month	Year	year
76	2020-03-09	6:00 PM	Maharashtra	0	0	2	9	3	2020	2020
91	2020-03-10	6:00 PM	Maharashtra	0	0	5	10	3	2020	2020
97	2020-03-11	6:00 PM	Maharashtra	0	0	2	11	3	2020	2020
120	2020-03-12	6:00 PM	Maharashtra	0	0	11	12	3	2020	2020
133	2020-03-13	6:00 PM	Maharashtra	0	0	14	13	3	2020	2020
...	...	...	...	...	...	...	...	...	...	...
16690	2021-07-03	8:00 AM	Maharashtra	5836920	122353	6079352	3	7	2021	2021
16726	2021-07-04	8:00 AM	Maharashtra	5845315	122724	6088841	4	7	2021	2021
16762	2021-07-05	8:00 AM	Maharashtra	5848693	123030	6098177	5	7	2021	2021
16798	2021-07-06	8:00 AM	Maharashtra	5861720	123136	6104917	6	7	2021	2021
16834	2021-07-07	8:00 AM	Maharashtra	5872268	123531	6113335	7	7	2021	2021

486 rows × 10 columns

In [85]:
```
Maharashtra_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 486 entries, 76 to 16834
Data columns (total 10 columns):
 #   Column                Non-Null Count  Dtype         
---  ------                --------------  -----         
 0   Date                  486 non-null    datetime64[ns]
 1   Time                  486 non-null    object        
 2   State/UnionTerritory  486 non-null    object        
 3   Cured                 486 non-null    int64         
 4   Deaths                486 non-null    int64         
 5   Confirmed             486 non-null    int64         
 6   Day                   486 non-null    int64         
 7   Month                 486 non-null    int64         
 8   Year                  486 non-null    int64         
 9   year                  486 non-null    int64         
dtypes: datetime64[ns](1), int64(7), object(2)
memory usage: 41.8+ KB
```

In [86]:
```
Maharashtra_data.describe()
```

Out[86]:

	Cured	Deaths	Confirmed	Day	Month	Year	year
count	4.860000e+02	486.000000	4.860000e+02	486.000000	486.000000	486.000000	486.000000
mean	1.674463e+06	39741.835391	1.870149e+06	15.744856	6.080247	2020.386831	2020.386831
std	1.710989e+06	31861.231600	1.831266e+06	8.810065	3.146548	0.487526	0.487526
min	0.000000e+00	0.000000	2.000000e+00	1.000000	1.000000	2020.000000	2020.000000
25%	1.197165e+05	9299.500000	2.187718e+05	8.000000	4.000000	2020.000000	2020.000000
50%	1.556812e+06	44884.500000	1.706879e+06	16.000000	6.000000	2020.000000	2020.000000
75%	2.066541e+06	52468.500000	2.216942e+06	23.000000	8.750000	2021.000000	2021.000000
max	5.872268e+06	123531.000000	6.113335e+06	31.000000	12.000000	2021.000000	2021.000000

In [87]:
```
Maharashtra_data.min()
```

Out[87]:
```
Date                     2020-03-09 00:00:00
Time                                10:00 AM
State/UnionTerritory             Maharashtra
Cured                                      0
Deaths                                     0
Confirmed                                  2
Day                                        1
Month                                      1
Year                                    2020
year                                    2020
dtype: object
```

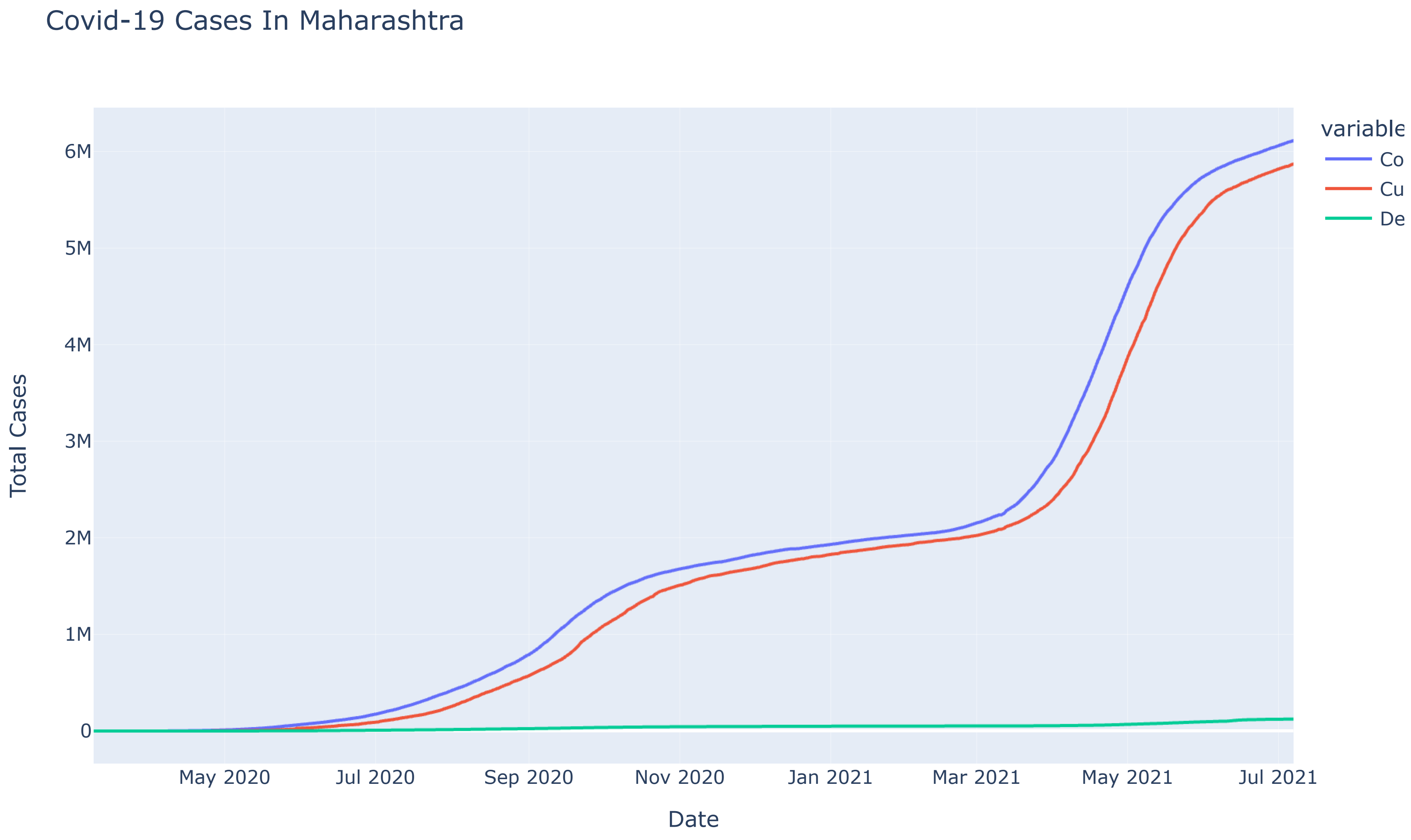
In [88]:
```
Maharashtra_data.max()
```

Out[88]:
```
Date                     2021-07-07 00:00:00
Time                                 9:30 PM
State/UnionTerritory             Maharashtra
Cured                                5872268
Deaths                                123531
Confirmed                            6113335
Day                                       31
Month                                     12
Year                                    2021
year                                    2021
dtype: object
```

In [89]:
```
Maharashtra_data.columns
```

Out[89]:
```
Index(['Date', 'Time', 'State/UnionTerritory', 'Cured', 'Deaths', 'Confirmed',
       'Day', 'Month', 'Year', 'year'],
      dtype='object')
```

```
In [90]: px.line(data_frame=df[df['State/UnionTerritory'] == 'Maharashtra'],
                x='Date', y=['Confirmed','Cured','Deaths'],
                labels={'value':'Total Cases'},
               height=600,title='Covid-19 Cases In Maharashtra')
```



In [179]:
```
Maharashtra_data.groupby(['Year','State/UnionTerritory'])[['Deaths', 'Confirmed',"Cured"]].sum().sort_values(by=['Year']
```

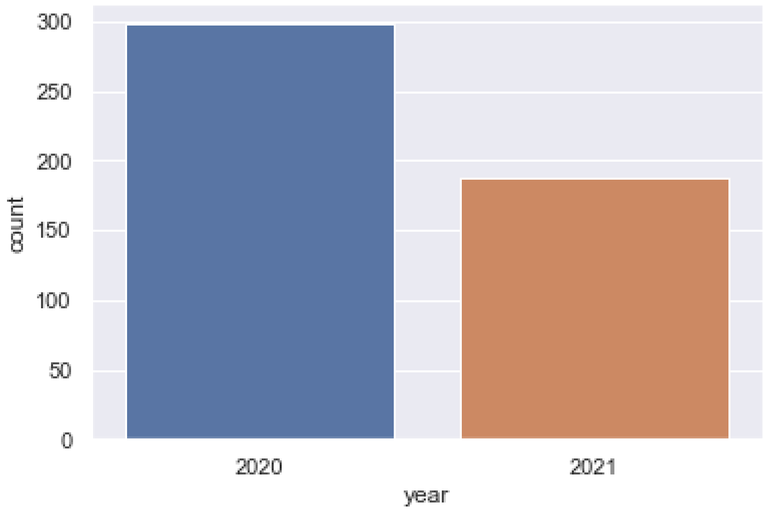
Out[179]:

	Year	State/UnionTerritory	Deaths	Confirmed	Cured
0	2021	Maharashtra	13129594	685991838	626754637
1	2020	Maharashtra	6184938	222900632	187034270

In [190]:
```
sns.countplot(Maharashtra_data['year'])
```

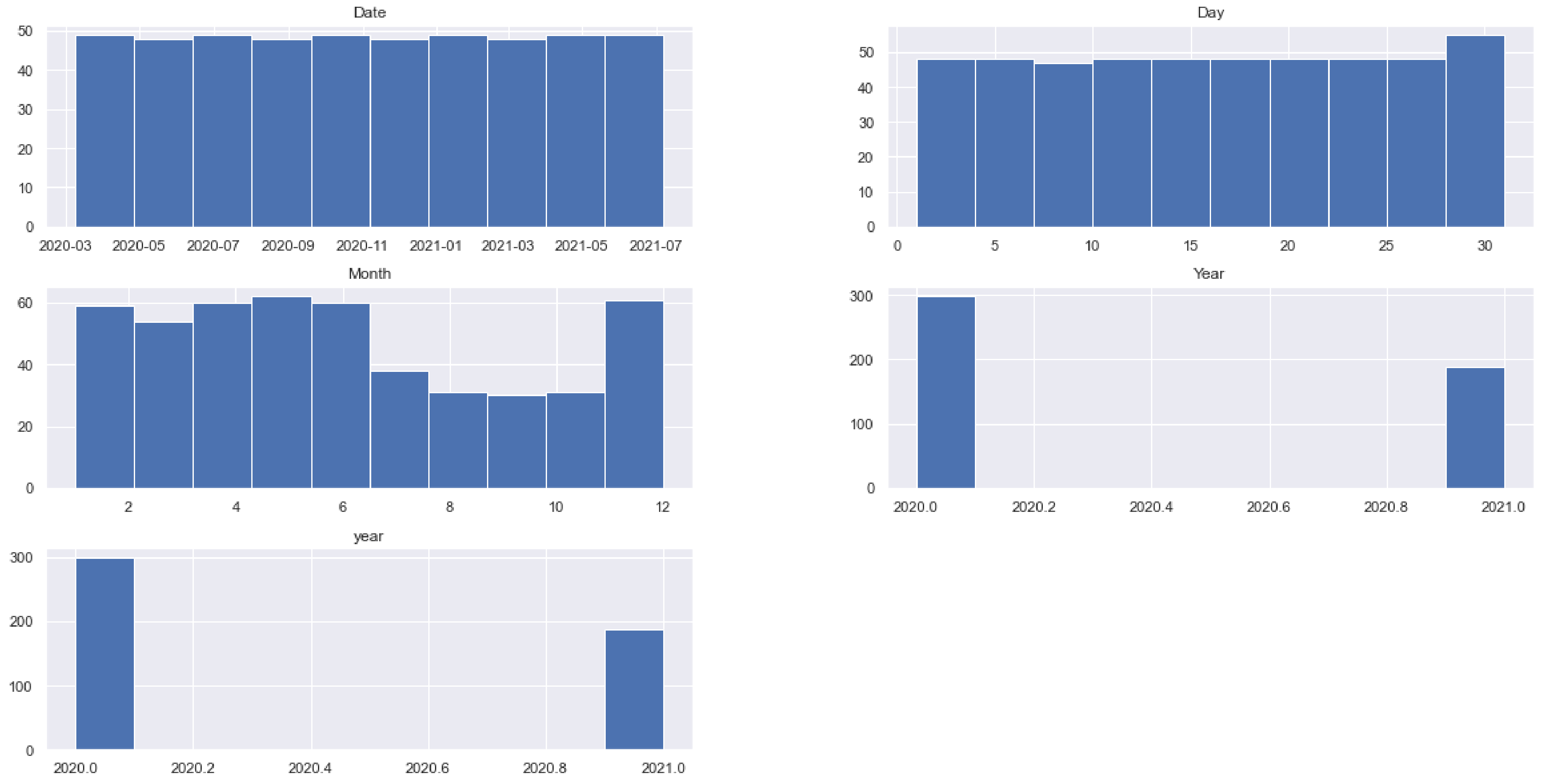
```
C:\Users\Admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:

Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in an error or misinterpretation.
```

Out[190]:
```
<AxesSubplot:xlabel='year', ylabel='count'>
```



```
In [191]: Maharashtra_data.hist(figsize = (20,10))
```

```
Out[191]: array([[<AxesSubplot:title={'center':'Date'}>,
                  <AxesSubplot:title={'center':'Day'}>],
                 [<AxesSubplot:title={'center':'Month'}>,
                  <AxesSubplot:title={'center':'Year'}>],
                 [<AxesSubplot:title={'center':'year'}>, <AxesSubplot:>]],
                dtype=object)
```



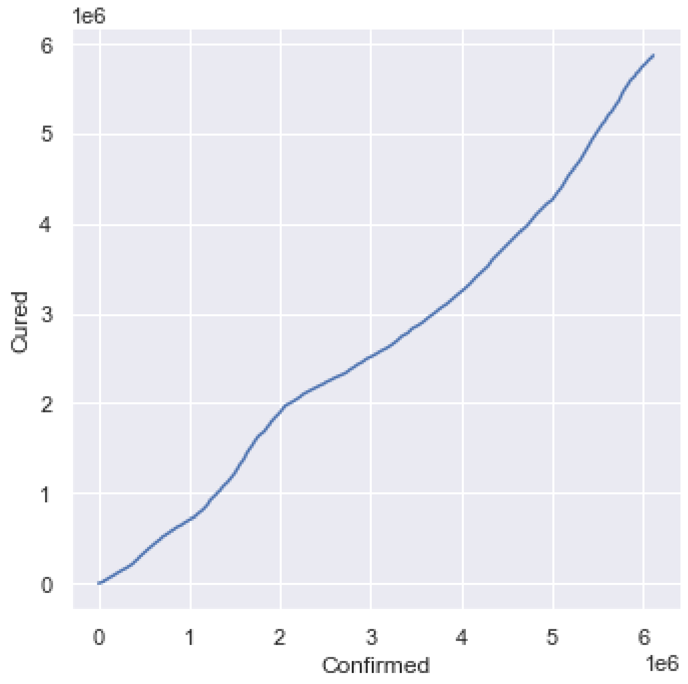
In [92]:
```
Maharashtra_data['Date'] = pd.to_datetime(Maharashtra_data['Date'])
```

```
C:\Users\Admin\AppData\Local\Temp/ipykernel_6380/3835735414.py:1: SettingWithCopyWarning:


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 (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py)
```

Ploting Between Confirmed and Cured

```
In [93]: sns.relplot(data=Maharashtra_data, x="Confirmed", y="Cured", kind="line",)
```

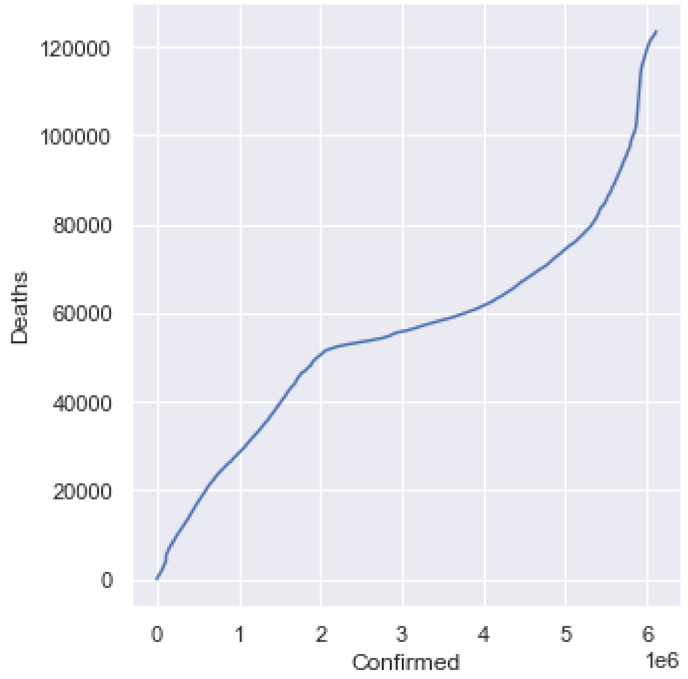
```
Out[93]: <seaborn.axisgrid.FacetGrid at 0x1e771b5a760>
```



Ploting Between Confirmed and Deathas

```
In [94]: sns.relplot(data=Maharashtra_data, x="Confirmed", y="Deaths", kind="line",)
```

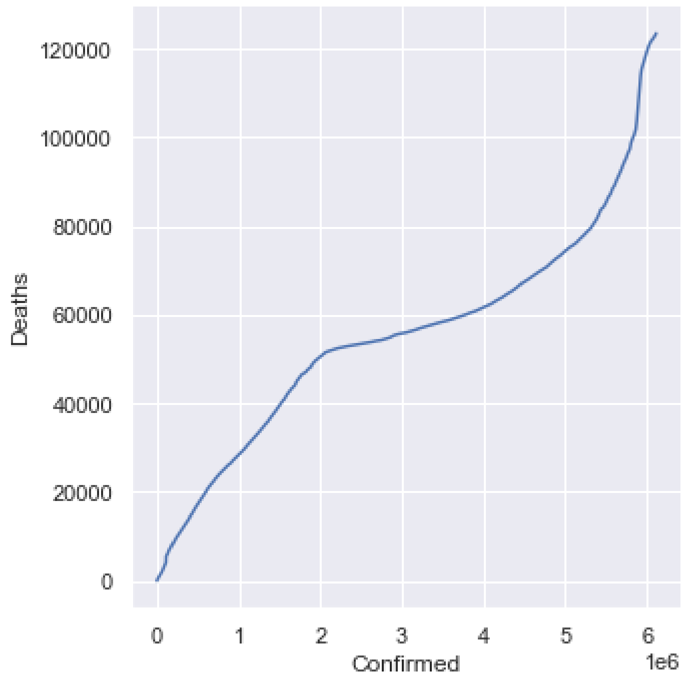
```
Out[94]: <seaborn.axisgrid.FacetGrid at 0x1e7718d4790>
```



Ploting Between Confirmed and Recovery cases

```
In [95]: sns.relplot(data=Maharashtra_data, x="Confirmed", y="Deaths", kind="line",)
```

```
Out[95]: <seaborn.axisgrid.FacetGrid at 0x1e7742a0bb0>
```

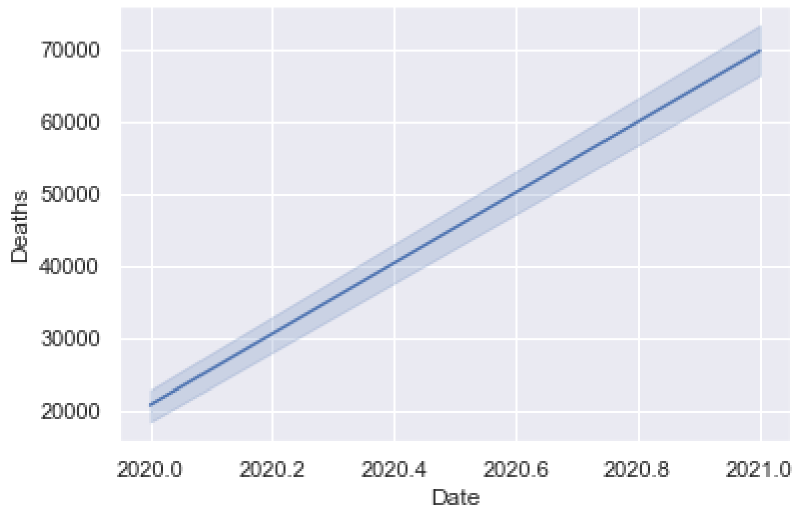


Ploting with Datetime/Year

Death rate in Maharashtra

```
In [177]: sns.lineplot(data=Maharashtra_data, x=pd.DatetimeIndex(Maharashtra_data['Date']).year, y="Deaths")
```

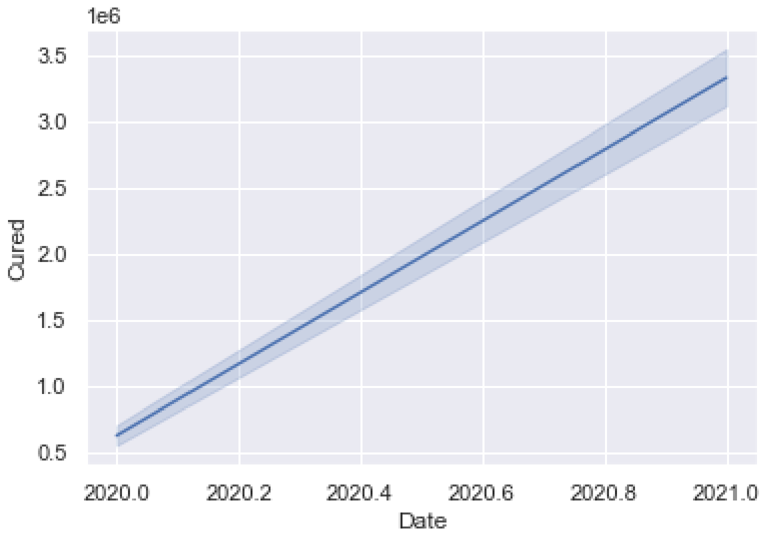
```
Out[177]: <AxesSubplot:xlabel='Date', ylabel='Deaths'>
```



Cured in Maharashtra

```
In [97]: sns.lineplot(data=Maharashtra_data, x=pd.DatetimeIndex(Maharashtra_data['Date']).year, y="Cured")
```

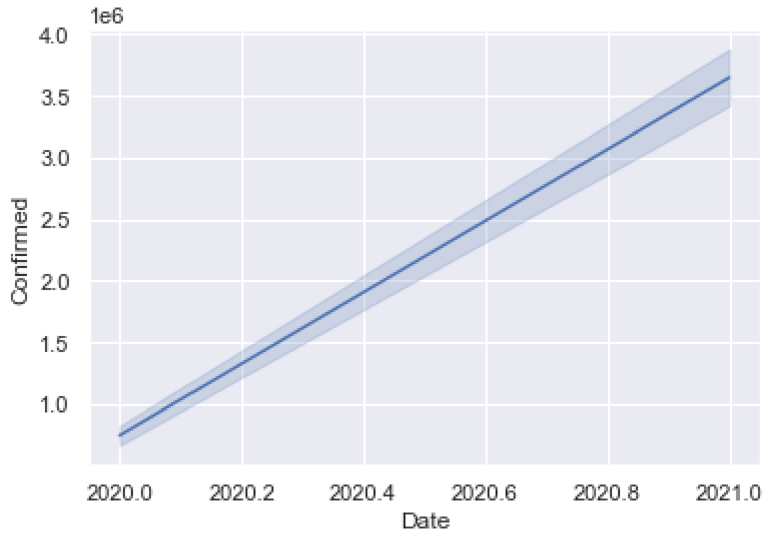
```
Out[97]: <AxesSubplot:xlabel='Date', ylabel='Cured'>
```



Confirmed cases in Maharashtra

```
In [98]: sns.lineplot(data=Maharashtra_data, x=pd.DatetimeIndex(Maharashtra_data['Date']).year, y="Confirmed")
```

```
Out[98]: <AxesSubplot:xlabel='Date', ylabel='Confirmed'>
```



Extracting Year

In [99]:
```
y_twenty_one=Maharashtra_data[Maharashtra_data['Year'] == 2021]
y_twenty_one
```

Out[99]:

	Date	Time	State/UnionTerritory	Cured	Deaths	Confirmed	Day	Month	Year	year
10102	2021-01-01	8:00 AM	Maharashtra	1828546	49521	1932112	1	1	2021	2021
10138	2021-01-02	8:00 AM	Maharashtra	1832825	49580	1935636	2	1	2021	2021
10174	2021-01-03	8:00 AM	Maharashtra	1834935	49631	1938854	3	1	2021	2021
10210	2021-01-04	8:00 AM	Maharashtra	1836999	49666	1942136	4	1	2021	2021
10246	2021-01-05	8:00 AM	Maharashtra	1847361	49695	1947011	5	1	2021	2021
...	...	...	...	...	...	...	...	...	...	...
16690	2021-07-03	8:00 AM	Maharashtra	5836920	122353	6079352	3	7	2021	2021
16726	2021-07-04	8:00 AM	Maharashtra	5845315	122724	6088841	4	7	2021	2021
16762	2021-07-05	8:00 AM	Maharashtra	5848693	123030	6098177	5	7	2021	2021
16798	2021-07-06	8:00 AM	Maharashtra	5861720	123136	6104917	6	7	2021	2021
16834	2021-07-07	8:00 AM	Maharashtra	5872268	123531	6113335	7	7	2021	2021

188 rows × 10 columns

In [100]:
```
y_twenty=Maharashtra_data[Maharashtra_data['Year'] == 2020]
y_twenty
```

Out[100]:

	Date	Time	State/UnionTerritory	Cured	Deaths	Confirmed	Day	Month	Year	year
76	2020-03-09	6:00 PM	Maharashtra	0	0	2	9	3	2020	2020
91	2020-03-10	6:00 PM	Maharashtra	0	0	5	10	3	2020	2020
97	2020-03-11	6:00 PM	Maharashtra	0	0	2	11	3	2020	2020
120	2020-03-12	6:00 PM	Maharashtra	0	0	11	12	3	2020	2020
133	2020-03-13	6:00 PM	Maharashtra	0	0	14	13	3	2020	2020
...	...	...	...	...	...	...	...	...	...	...
9922	2020-12-27	8:00 AM	Maharashtra	1807824	49189	1916236	27	12	2020	2020
9958	2020-12-28	8:00 AM	Maharashtra	1809948	49255	1919550	28	12	2020	2020
9994	2020-12-29	8:00 AM	Maharashtra	1814449	49305	1922048	29	12	2020	2020
10030	2020-12-30	8:00 AM	Maharashtra	1820021	49373	1925066	30	12	2020	2020
10066	2020-12-31	8:00 AM	Maharashtra	1824934	49463	1928603	31	12	2020	2020

298 rows × 10 columns

In [101]:
```
Maharashtra_data.columns
```

Out[101]:
```
Index(['Date', 'Time', 'State/UnionTerritory', 'Cured', 'Deaths', 'Confirmed',
       'Day', 'Month', 'Year', 'year'],
      dtype='object')
```

In [113]:
```
Maharashtra_data['Date'] = pd.to_datetime(Maharashtra_data['Date'])
```

```
C:\Users\Admin\AppData\Local\Temp/ipykernel_6380/3835735414.py:1: SettingWithCopyWarning:


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 (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py)
```

In [114]:
```
Maharashtra_data['Date']
```

Out[114]:
```
76      2020-03-09
91      2020-03-10
97      2020-03-11
120     2020-03-12
133     2020-03-13
           ...
16690   2021-07-03
16726   2021-07-04
16762   2021-07-05
16798   2021-07-06
16834   2021-07-07
Name: Date, Length: 486, dtype: datetime64[ns]
```

Groupby with cured,confirmed,deaths

In [132]:
```
by_month=y_twenty.groupby(['Date'])['Cured','Deaths','Confirmed'].sum()
```

...

```
In [133]: by_month
```

```
Out[133]:
```

Date	Cured	Deaths	Confirmed
2020-03-09	0	0	2
2020-03-10	0	0	5
2020-03-11	0	0	2
2020-03-12	0	0	11
2020-03-13	0	0	14
...	...	...	...
2020-12-27	1807824	49189	1916236
2020-12-28	1809948	49255	1919550
2020-12-29	1814449	49305	1922048
2020-12-30	1820021	49373	1925066
2020-12-31	1824934	49463	1928603

298 rows × 3 columns

From 2020 Extracting quarter end frequency

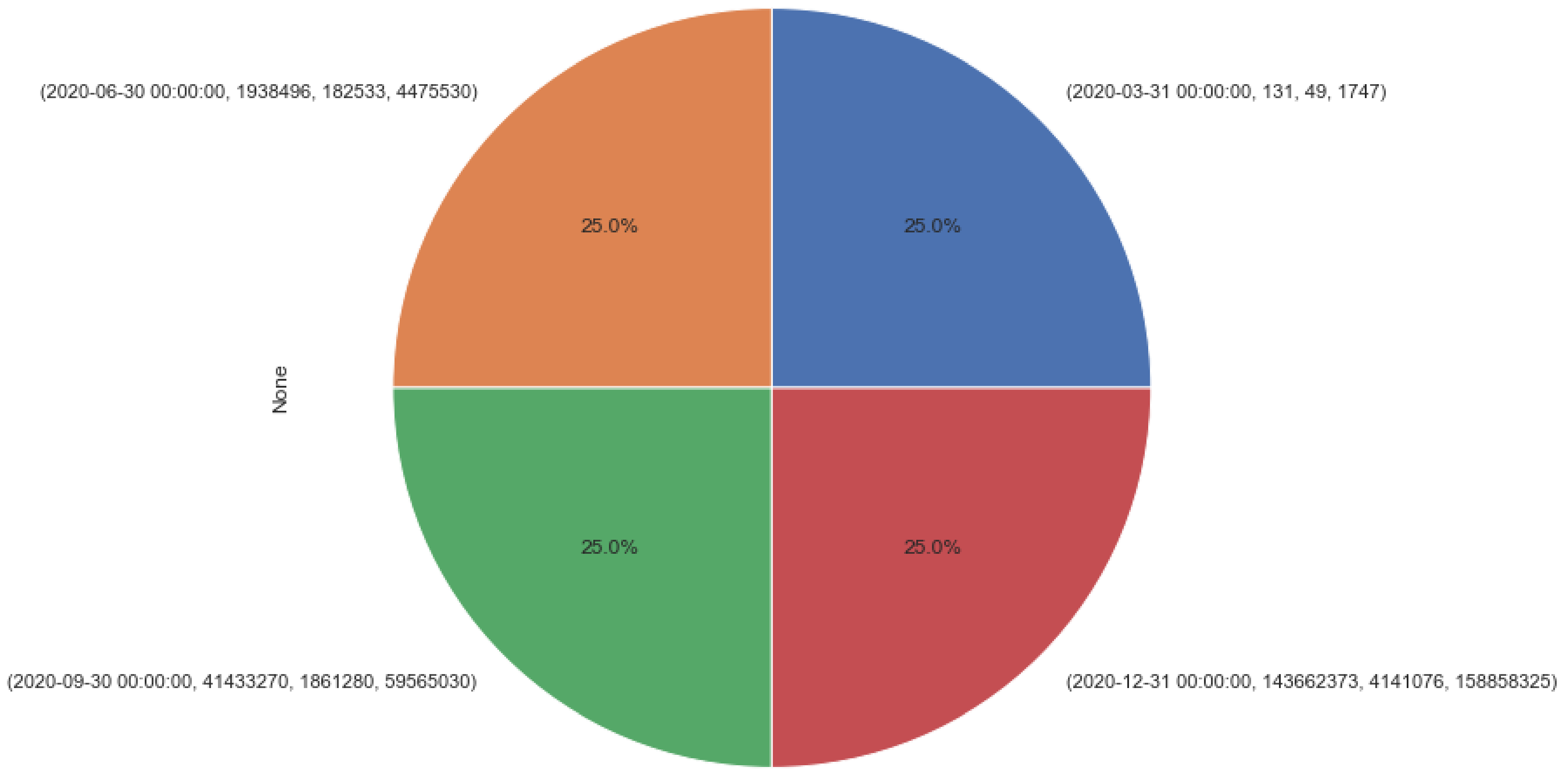
In [172]:
```
seperate=by_month.resample('Q').sum().reset_index()
seperate
```

Out[172]:

	Date	Cured	Deaths	Confirmed
0	2020-03-31	131	49	1747
1	2020-06-30	1938496	182533	4475530
2	2020-09-30	41433270	1861280	59565030
3	2020-12-31	143662373	4141076	158858325

In [210]:
```
plt.figure(figsize=(20,10))
seperate.value_counts().plot.pie(autopct='%1.1f%%')
```

Out[210]:
```
<AxesSubplot:ylabel='None'>
```



```
In [165]: by_month_next=y_twenty_one.groupby(['Date'])['Cured','Deaths','Confirmed'].sum()
```

```
C:\Users\Admin\AppData\Local\Temp/ipykernel_6380/632991346.py:1: FutureWarning:

Indexing with multiple keys (implicitly converted to a tuple of keys) will be deprecated, use a list instead.
```

```
In [166]: by_month_next
```

Out[166]:

Date	Cured	Deaths	Confirmed
2021-01-01	1828546	49521	1932112
2021-01-02	1832825	49580	1935636
2021-01-03	1834935	49631	1938854
2021-01-04	1836999	49666	1942136
2021-01-05	1847361	49695	1947011
...	...	...	...
2021-07-03	5836920	122353	6079352
2021-07-04	5845315	122724	6088841
2021-07-05	5848693	123030	6098177
2021-07-06	5861720	123136	6104917
2021-07-07	5872268	123531	6113335

188 rows × 3 columns

From 2021 Extracting 6 months

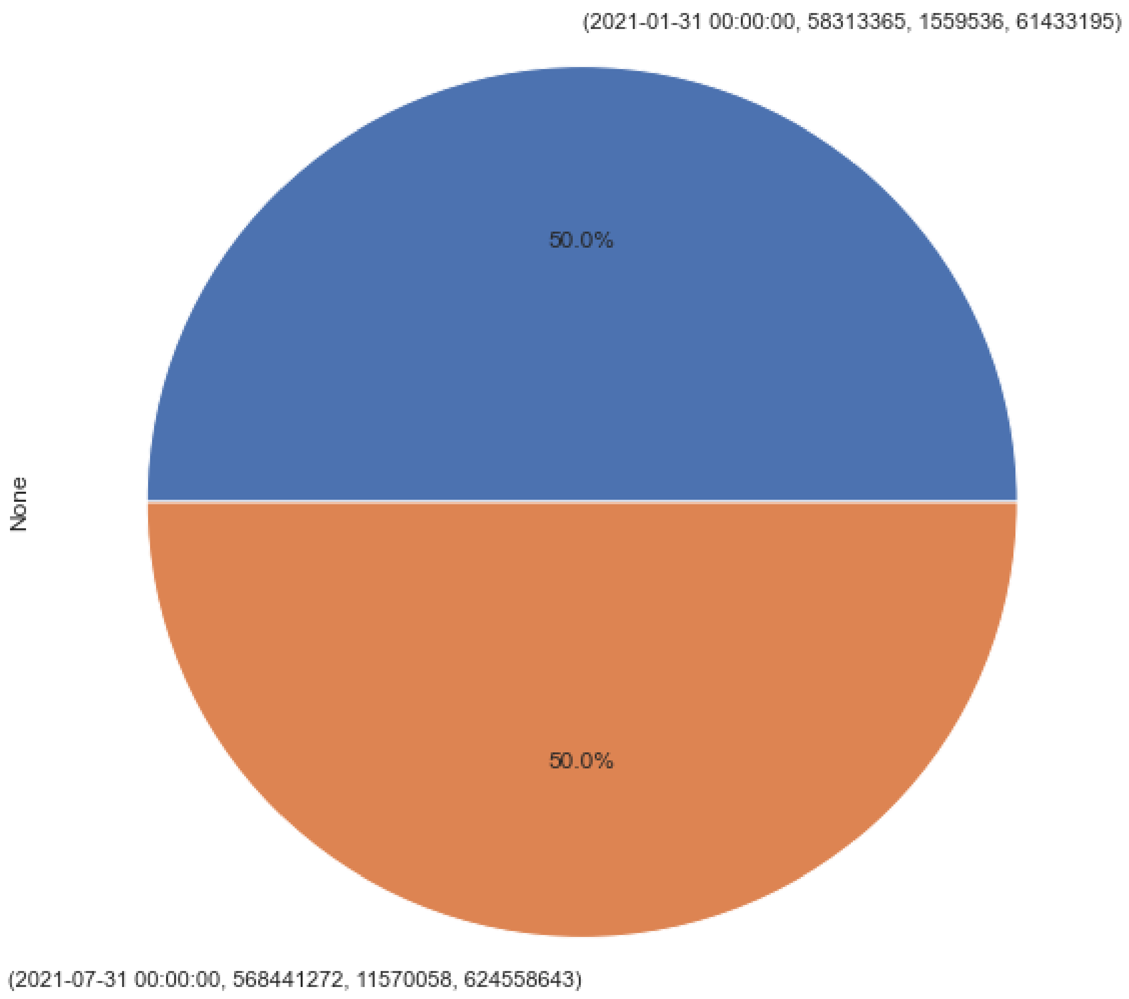
```
In [170]: next_sep=by_month_next.resample('6M').sum().reset_index()
          next_sep
```

Out[170]:

	Date	Cured	Deaths	Confirmed
0	2021-01-31	58313365	1559536	61433195
1	2021-07-31	568441272	11570058	624558643

```
In [211]: plt.figure(figsize=(20,10))
          next_sep.value_counts().plot.pie(autopct='%1.1f%%')
```

```
Out[211]: <AxesSubplot:ylabel='None'>
```



```
In [ ]:
```