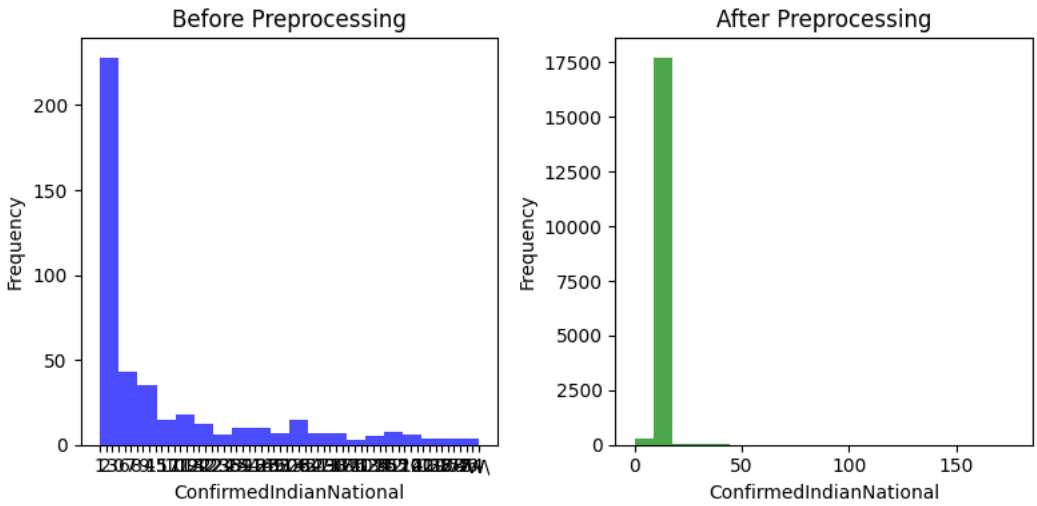
```
import pandas as pd
import matplotlib.pyplot as plt
covid_df = pd.read_csv('/content/covid_19_india (1).csv', parse_dates=['Date'], infer_datetime_format=True)
print("Missing Values Before Preprocessing:")
print(covid_df.isnull().sum())
plt.figure(figsize=(12,4))
plt.subplot(1, 3, 1)
plt.title('Before Preprocessing')
plt.hist(covid_df['ConfirmedIndianNational'].dropna(), bins=20, color='blue', alpha=0.7)
plt.xlabel('ConfirmedIndianNational')
plt.ylabel('Frequency')
columns_to_fill = ['ConfirmedIndianNational', 'ConfirmedForeignNational', 'Cured', 'Deaths', 'Confirmed']
for col in covid_df.columns:
    if covid_df[col].isnull().any():
        covid_df[col] = pd.to_numeric(covid_df[col], errors='coerce')
        col_mean = covid_df[col].mean()
        covid_df[col].fillna(col_mean, inplace=True)
print("\nMean Values used for filling missing values:")
print({col: covid_df[col].mean() for col in columns_to_fill})
plt.subplot(1, 3, 2)
plt.title('After Preprocessing')
plt.hist(covid_df['ConfirmedIndianNational'], bins=20, color='green', alpha=0.7)
plt.xlabel('ConfirmedIndianNational')
plt.ylabel('Frequency')

plt.tight_layout()
plt.show()
```

```
Missing Values Before Preprocessing:
Sno                            0
Date                           0
Time                           0
State/UnionTerritory           0
ConfirmedIndianNational    17663
ConfirmedForeignNational   17664
Cured                          0
Deaths                         0
Confirmed                      0
latitude                   18110
longitude                  18110
dtype: int64

Mean Values used for filling missing values:
{'ConfirmedIndianNational': 12.188340807174885, 'ConfirmedForeignNational': 1.4955156950672648, 'Cured': 278637.5180563225, 'Deaths
```




```
#describing data
covid_df.head()
```

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed	latitude	lon
0	1	2020-01-30	6:00 PM	Kerala	1.0	0.0	0	0	1	NaN	
1	2	2020-01-31	6:00 PM	Kerala	1.0	0.0	0	0	1	NaN	
2	3	2020-02-01	6:00 PM	Kerala	2.0	0.0	0	0	2	NaN	

```
covid_df.tail()
```

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedFo
18105	18106	2021-08-11	8:00 AM	Telangana	12.188341	
18106	18107	2021-08-11	8:00 AM	Tripura	12.188341	
18107	18108	2021-08-11	8:00 AM	Uttarakhand	12.188341	

```
covid_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18110 entries, 0 to 18109
Data columns (total 11 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Sno                      18110 non-null  int64
 1   Date                     18110 non-null  datetime64[ns]
 2   Time                     18110 non-null  object
 3   State/UnionTerritory     18110 non-null  object
 4   ConfirmedIndianNational  18110 non-null  float64
 5   ConfirmedForeignNational  18110 non-null  float64
 6   Cured                    18110 non-null  int64
 7   Deaths                   18110 non-null  int64
 8   Confirmed                18110 non-null  int64
 9   latitude                 0 non-null      float64
 10  longitude                0 non-null      float64
dtypes: datetime64[ns](1), float64(4), int64(4), object(2)
memory usage: 1.5+ MB
```

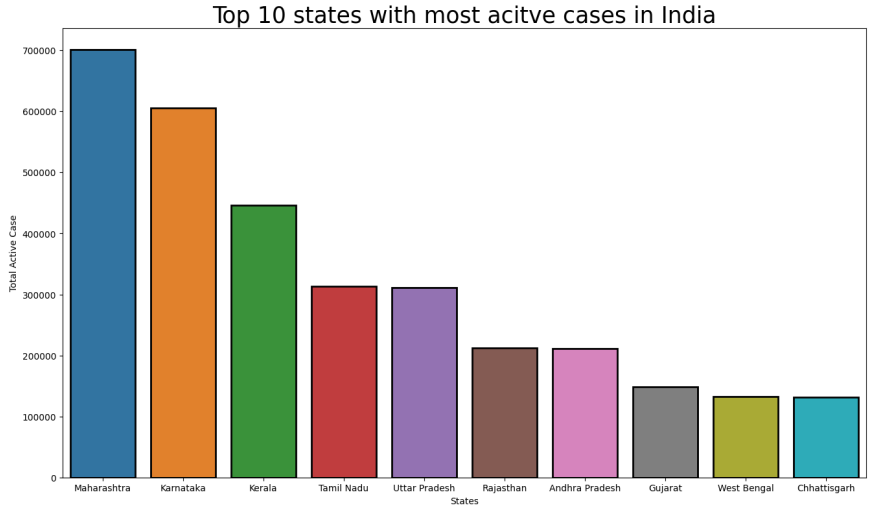
```
covid_df.describe()
```

	Sno	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed	latitude	longitud
count	18110.000000	18110.000000	18110.000000	1.811000e+04	18110.000000	1.811000e+04	0.0	0.(
mean	9055.500000	12.188341	1.495516	2.786375e+05	4052.402264	3.010314e+05	NaN	NaN
std	5228.051023	3.383215	0.560616	6.148909e+05	10919.076411	6.561489e+05	NaN	NaN
min	1.000000	0.000000	0.000000	0.000000e+00	0.000000	0.000000e+00	NaN	NaN
25%	4528.250000	12.188341	1.495516	3.360250e+03	32.000000	4.376750e+03	NaN	NaN
50%	9055.500000	12.188341	1.495516	3.336400e+04	588.000000	3.977350e+04	NaN	NaN
75%	13582.750000	12.188341	1.495516	2.788698e+05	3643.750000	3.001498e+05	NaN	NaN
max	18110.000000	177.000000	14.000000	6.159676e+06	134201.000000	6.363442e+06	NaN	NaN

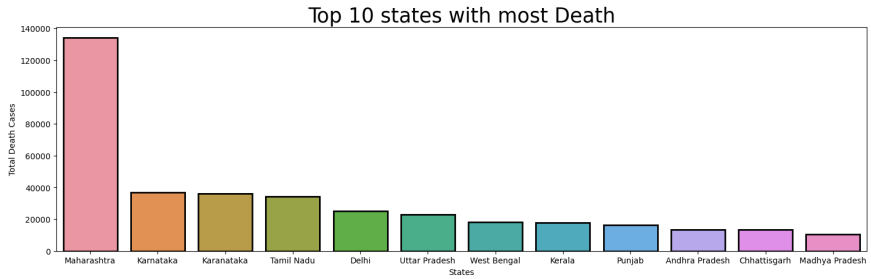
```
covid_df['Active_Cases']=covid_df['Confirmed']-(covid_df['Cured']+covid_df['Deaths'])
covid_df.tail()
#top 10 active cases stores
top_10_active_cases=covid_df.groupby(by="State/UnionTerritory").max()[["Active_Cases","Date"]].sort_values(by=["Active_Cases"],ascending
fig=plt.figure(figsize=(16,9))
plt.title("Top 10 states with most acitve cases in India",size=25)
ax=sns.barplot(data=top_10_active_cases.iloc[:10],y="Active_Cases",x="State/UnionTerritory",linewidth=2,edgecolor="black")

plt.xlabel("States")
plt.ylabel("Total Active Case")
```

```
Text(0, 0.5, 'Total Active Case')
```

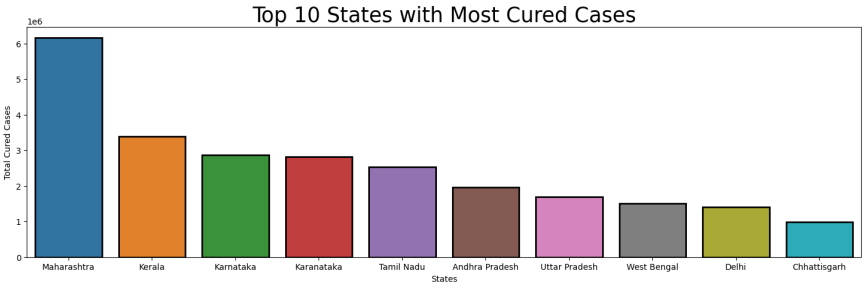



```
top_10_deaths=covid_df.groupby(by='State/UnionTerritory').max()[['Deaths','Date']].sort_values(by=['Deaths'],ascending=False).reset_inde
fig=plt.figure(figsize=(18,5))
plt.title("Top 10 states with most Death",size=25)
ax=sns.barplot(data=top_10_deaths.iloc[:12],y="Deaths",x='State/UnionTerritory',linewidth=2,edgecolor='black')
plt.xlabel("States")
plt.ylabel("Total Death Cases")
plt.show()
```




```
top_10_cured = covid_df.groupby(by='State/UnionTerritory').max()[['Cured', 'Date']].sort_values(by=['Cured'], ascending=False).reset_ind

fig = plt.figure(figsize=(18, 5))
plt.title("Top 10 States with Most Cured Cases", size=25)
ax = sns.barplot(data=top_10_cured.iloc[:10], y="Cured", x='State/UnionTerritory', linewidth=2, edgecolor='black')
plt.xlabel("States")
plt.ylabel("Total Cured Cases")
plt.show()
```




```
bivariate_df = covid_df[['Confirmed', 'Cured', 'State/UnionTerritory']]
state_mean_df = bivariate_df.groupby('State/UnionTerritory').mean()
plt.figure(figsize=(12, 6))
sns.scatterplot(x='Confirmed', y='Cured', hue='State/UnionTerritory', data=bivariate_df, palette='viridis', s=100)
plt.title('Bivariate Analysis: Confirmed vs Cured Cases for Each State')
plt.xlabel('Confirmed Cases')
plt.ylabel('Cured Cases')
plt.legend(bbox_to_anchor=(1, 1), loc='upper left')
plt.show()
```