COVID 19 PROJECT

CODE

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import datetime
covid_19 = pd.read_csv("covid_19_india_download.csv")

# Correction in state names
# creation of new column ST/UT which contains corrected state names
df = pd.DataFrame(covid_19, columns= ["Sno","State/UnionTerritory"])
df.rename(columns = {"State/UnionTerritory":'ST/UT'}, inplace = True)
df
```

OUTPUT

	Sno	ST/UT
0	1	Kerala
1	2	Kerala
2	3	Kerala
3	4	Kerala
4	5	Kerala
...	...	...
9286	9287	Telengana
9287	9288	Tripura
9288	9289	Uttarakhand
9289	9290	Uttar Pradesh
9290	9291	West Bengal

9291 rows × 2 columns

Q1 Check unique state/UT names, fix state names appearing twice or more due to spelling mistakes
Example -

Maharashtra
Maharashtra***

Here Maharashtra should be replaced by Maharashtra**

Example -

Telangana
Telangana*
Telengana
Telengana*

Here Telangana, Telengana, Telengana** should be replaced by telangana**

Do the same for all states, Hint - Use replace() function.

CODE

```
#check unique state/ut names , fix state names
def get_States(data_frame):

#Find all States
    States=[]
      for state in data_frame["ST/UT"]:
        if state not in States:
          States.append(state)
    return data_frame, States

    df,States = get_States(df)


def rep_States(data):
    data["ST/UT"].replace(["Telengana", "Telangana***",
"Telengana***","Maharashtra***","Chandigarh***", "Punjab***"],

            ["Telangana","Telangana","Telangana","Maharashtra","Chandigarh","Punjab"],
inplace = True)

    return data
df = rep_States(df)
covid_19_corr = covid_19.merge(df, on="Sno")
covid_19_corr
```

OUTPUT

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed	ST/UT
0	1	30/01/20	6:00 PM	Kerala	1	0	0	0	1	Kerala
1	2	31/01/20	6:00 PM	Kerala	1	0	0	0	1	Kerala
2	3	01/02/20	6:00 PM	Kerala	2	0	0	0	2	Kerala
3	4	02/02/20	6:00 PM	Kerala	3	0	0	0	3	Kerala
4	5	03/02/20	6:00 PM	Kerala	3	0	0	0	3	Kerala
...	...	...	...	...	...	...	...	...	...	...
9286	9287	09/12/20	8:00 AM	Telengana	-	-	266120	1480	275261	Telangana
9287	9288	09/12/20	8:00 AM	Tripura	-	-	32169	373	32945	Tripura
9288	9289	09/12/20	8:00 AM	Uttarakhand	-	-	72435	1307	79141	Uttarakhand
9289	9290	09/12/20	8:00 AM	Uttar Pradesh	-	-	528832	7967	558173	Uttar Pradesh
9290	9291	09/12/20	8:00 AM	West Bengal	-	-	475425	8820	507995	West Bengal

9291 rows × 10 columns

DATA ANALYSIS

CODE

covid_19_corr.info()

OUTPUT

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9291 entries, 0 to 9290
Data columns (total 10 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Sno                      9291 non-null   int64
 1   Date                     9291 non-null   object
 2   Time                     9291 non-null   object
 3   State/UnionTerritory     9291 non-null   object
 4   ConfirmedIndianNational  9291 non-null   object
 5   ConfirmedForeignNational 9291 non-null   object
 6   Cured                    9291 non-null   int64
 7   Deaths                   9291 non-null   int64
 8   Confirmed                9291 non-null   int64
 9   ST/UT                    9291 non-null   object
dtypes: int64(4), object(6)
memory usage: 798.4+ KB
```

CODE

covid_19_corr.describe()

OUTPUT

	Sno	Cured	Deaths	Confirmed
count	9291.000000	9.291000e+03	9291.000000	9.291000e+03
mean	4646.000000	7.863266e+04	1487.620385	9.183978e+04
std	2682.225009	1.931102e+05	4713.813690	2.166014e+05
min	1.000000	0.000000e+00	0.000000	0.000000e+00
25%	2323.500000	1.520000e+02	2.000000	5.385000e+02
50%	4646.000000	4.308000e+03	66.000000	6.832000e+03
75%	6968.500000	5.772650e+04	926.500000	7.885600e+04
max	9291.000000	1.737080e+06	47827.000000	1.859367e+06

CODE

covid_19_corr.head(5)

OUTPUT

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed	ST/UT
0	1	30/01/20	6:00 PM	Kerala	1	0	0	0	1	Kerala
1	2	31/01/20	6:00 PM	Kerala	1	0	0	0	1	Kerala
2	3	01/02/20	6:00 PM	Kerala	2	0	0	0	2	Kerala
3	4	02/02/20	6:00 PM	Kerala	3	0	0	0	3	Kerala
4	5	03/02/20	6:00 PM	Kerala	3	0	0	0	3	Kerala

CODE

covid_19_corr.tail(5)

OUTPUT

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed	ST/UT
9286	9287	09/12/20	8:00 AM	Telengana	-	-	266120	1480	275261	Telangana
9287	9288	09/12/20	8:00 AM	Tripura	-	-	32169	373	32945	Tripura
9288	9289	09/12/20	8:00 AM	Uttarakhand	-	-	72435	1307	79141	Uttarakhand
9289	9290	09/12/20	8:00 AM	Uttar Pradesh	-	-	528832	7967	558173	Uttar Pradesh
9290	9291	09/12/20	8:00 AM	West Bengal	-	-	475425	8820	507995	West Bengal

Q2. Calculate per day average confirmed cases for all states/UT.
 # Calculate per day average confirmed cases for all states /UT
CODE

```
df = pd.DataFrame(covid_19_corr)
df.groupby(["Date","ST/UT"], sort = False)["Confirmed"].mean()
```

OUTPUT

```
Date      ST/UT
30/01/20  Kerala               1
31/01/20  Kerala               1
01/02/20  Kerala               2
02/02/20  Kerala               3
03/02/20  Kerala               3
                             ...
09/12/20  Telangana       275261
          Tripura          32945
          Uttarakhand      79141
          Uttar Pradesh    558173
          West Bengal      507995
Name: Confirmed, Length: 9291, dtype: int64
```

Q3. Plot a linegraph that shows distribution of per day confirmed cases in Maharashtra in 2020.
#linegraph that shows distribution of per day confirmed cases in Maharashtra in 2020

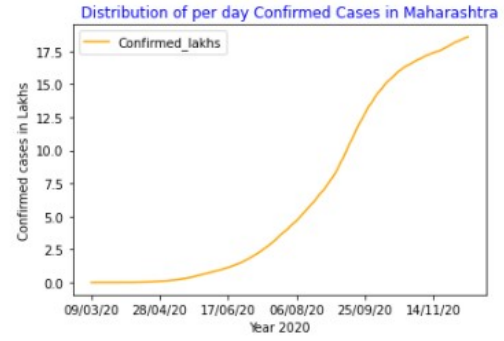
CODE

```
df = pd.DataFrame(covid_19_corr)
rf = df[df["ST/UT"] == "Maharashtra"]
rf["Confirmed_lakhs"] = rf["Confirmed"]/100000
rf.plot(x = "Date" , y = "Confirmed_lakhs", color = "Orange")
plt.title("    Distribution of per day Confirmed Cases in Maharashtra" , color = "Blue")
plt.xlabel('Year 2020')
plt.ylabel('Confirmed cases in Lakhs')
plt.show()
```

OUTPUT

```
<ipython-input-42-7db641056e49>:4: 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
  rf["Confirmed_lakhs"] = rf["Confirmed"]/100000
```



Q4. Plot a pie-chart displaying percentage of total cured and total death cases in Maharashtra.
#Piechart displaying % total cured and % total death cases
CODE

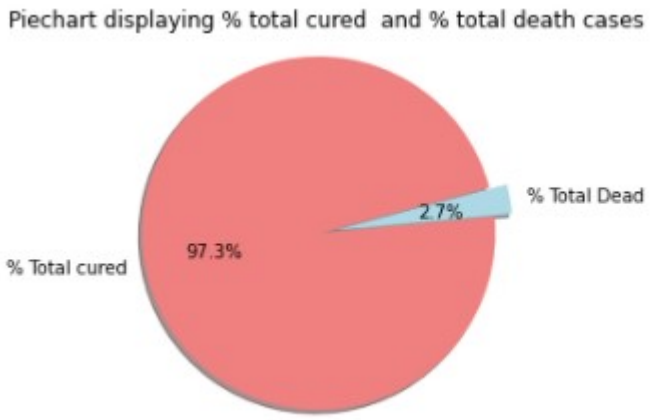
```
df = pd.DataFrame(covid_19_corr)
rf = df[df["ST/UT"] == "Maharashtra"]

total_cured = rf["Cured"].max()
total_dead = rf["Deaths"].max()
total_cases = rf["Confirmed"].max()
data = []
a1 = total_cured* 100/total_cases
data.append(a1)
a1 = total_dead*100/total_cases
data.append(a1)

my_labels = '% Total cured','% Total Dead'
my_colors = ['lightcoral','lightblue']

plt.pie( x = data , labels=my_labels, autopct='%1.1f%%', startangle=15, shadow = True,
colors=my_colors, explode=(0,0.1))
plt.title('Piechart displaying % total cured  and % total death cases')
plt.axis('equal')
plt.show()
```

OUTPUT



Q5. Plot a barplot showing top-5 states with maximum number of total confirmed cases.

#Bar plot showing top 5 states with maximum no of total confirmed cases

CODE

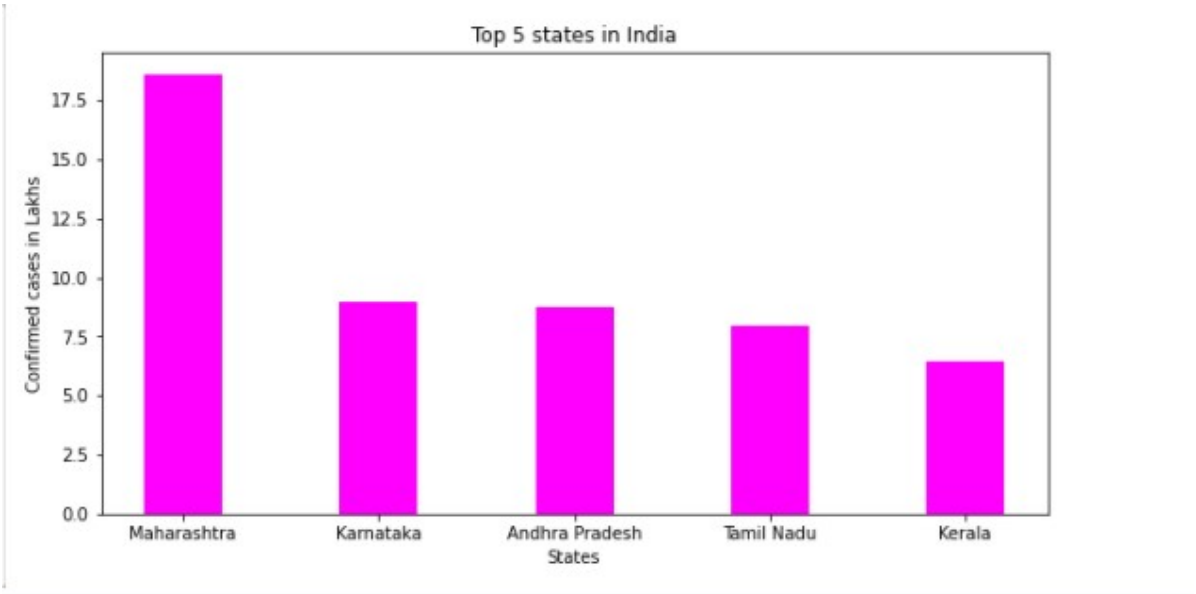
```
df = pd.DataFrame(covid_19_corr)
group2 = df.groupby("ST/UT")["Confirmed"].max()
df1 = pd.DataFrame(group2)
df2 = df1.sort_values(by=["Confirmed","ST/UT"],ascending = False).iloc[:5,:]
df3 = pd.DataFrame(df2)
state_names = []
for row in df3.index:
    state_names.append(row)
cases = df3["Confirmed"]/100000

fig =plt.figure(figsize =(10, 5))

# creating the bar plot
plt.bar(state_names, cases, color ="Magenta", width =0.4)

plt.xlabel("States")
plt.ylabel("Confirmed cases in Lakhs")
plt.title("Top 5 states in India")
plt.show()
```

OUTPUT



Q6. Which 3 states have lowest total death cases ?

#which 3 states have lowest death cases
CODE

```
df = pd.DataFrame(covid_19_corr)
group2 = df.groupby("ST/UT")["Deaths"].max()
df1 = pd.DataFrame(group2)
df2 = df1.sort_values(by="Deaths").head(3)
df2
```

OUTPUT

	Deaths
ST/UT	
Unassigned	0
Daman & Diu	0
Cases being reassigned to states	0

Q7. Plot multi linegraph that shows distribution of per day confirmed cases, death cases and cured cases in India in 2020.

 # multi linegraph that shows distribution of per day confirmed cases,
 # death cases and cured cases in India in 2020.

CODE

```
df2 = pd.DataFrame(covid_19_corr)
y1 = df2.groupby(["Date"], sort = False)["Confirmed"].sum()
y2 = df2.groupby(["Date"], sort = False)["Deaths"].sum()
y3 = df2.groupby(["Date"], sort = False)["Cured"].sum()

yy1 = pd.DataFrame(y1)
yy2 = pd.DataFrame(y2)
yy3 = pd.DataFrame(y3)

yy4 = yy1.merge(yy2 , on = "Date")
yy5 = yy4.merge(yy3 , on = "Date")
yy5["Dates"] = yy5.index


plt.plot("Dates", "Confirmed", data = yy5, label = "Confirmed",linestyle="-")
plt.plot("Dates", "Deaths", data = yy5, label = "Deaths",linestyle="--")
plt.plot("Dates", "Cured", data = yy5, label = "Cured",linestyle="-.")
plt.legend()
plt.xlabel('Dates in year 2020')
```

```
plt.ylabel('Cases')
plt.show()
```

OUTPUT