Task-06 Timeline Analysis Covid-19.

Create a EDA showing spread of Covid-19 cases in your country or any region (Asia, Europe, BRICS etc).

Identify interesting patterns and possible reasons helping Covid-19 spreadwith basic as well as advanced charts.

Dataset: https://bit.ly/30d2gdi

Coronavirus is a family of viruses that can cause illness, which can vary from common cold and cough to sometimes more severe disease. Middle East Respiratory Syndrome (MERS-CoV) and Severe Acute Respiratory Syndrome (SARS-CoV) were such severe cases with the world already has faced.

SARS-CoV-2 (n-coronavirus) is the new virus of the coronavirus family, which first discovered in 2019, which has not been identified in humans before. It is a contiguous virus which started from Wuhan in December 2019. Which later declared as Pandemic by WHO due to high rate spreads throughout the world. Currently (on date 08 October 2020), this leads to a total of 3M+ cases across the globe, including 100K+ deaths around the globe.

Pandemic is spreading all over the world; it becomes more important to understand about this spread. This NoteBook is an effort to analyze the cumulative data of confirmed, deaths, and recovered cases over time. In this notebook, the main focus is to analyze the spread trend of this virus all over the world.

Downloding and Installing Prerequisite

Install:

pip install pycountry_convert

pip install folium

In [1]:

```
# Install pycountry_convert
!pip3 install pycountry_convert
!pip3 install folium
!wget https://raw.githubusercontent.com/tarunk04/COVID-19-CaseStudy-and-Predictions/master/models/model_
!wget https://raw.githubusercontent.com/tarunk04/COVID-19-CaseStudy-and-Predictions/master/models/model_
!wget https://raw.githubusercontent.com/tarunk04/COVID-19-CaseStudy-and-Predictions/master/models/model_
```

```
Requirement already satisfied: pycountry_convert in /home/anshal/.local/lib/python3.8/site-packages (0.7.
2)
Requirement already satisfied: pprintpp>=0.3.0 in /home/anshal/.local/lib/python3.8/site-packages (from p
ycountry_convert) (0.4.0)
Requirement already satisfied: repoze.lru>=0.7 in /home/anshal/.local/lib/python3.8/site-packages (from p
ycountry_convert) (0.7)
Requirement already satisfied: pytest-mock>=1.6.3 in /home/anshal/.local/lib/python3.8/site-packages (fro
m pycountry_convert) (3.3.1)
Requirement already satisfied: pycountry>=16.11.27.1 in /home/anshal/.local/lib/python3.8/site-packages
(from pycountry_convert) (20.7.3)
Requirement already satisfied: pytest-cov>=2.5.1 in /home/anshal/.local/lib/python3.8/site-packages (from
pycountry_convert) (2.10.1)
Requirement already satisfied: pytest>=3.4.0 in /home/anshal/.local/lib/python3.8/site-packages (from pyc
ountry_convert) (6.1.1)
Requirement already satisfied: wheel>=0.30.0 in /usr/lib/python3/dist-packages (from pycountry_convert) (
0.34.2)
Requirement already satisfied: coverage>=4.4 in /home/anshal/.local/lib/python3.8/site-packages (from pyt
est-cov>=2.5.1->pycountry_convert) (5.3)
Requirement already satisfied: py>=1.8.2 in /home/anshal/.local/lib/python3.8/site-packages (from pytest>
=3.4.0->pycountry_convert) (1.9.0)
Requirement already satisfied: attrs>=17.4.0 in /home/anshal/.local/lib/python3.8/site-packages (from pyt
est>=3.4.0->pycountry_convert) (20.2.0)
Requirement already satisfied: toml in /home/anshal/.local/lib/python3.8/site-packages (from pytest>=3.4.
0->pycountry_convert) (0.10.1)
Requirement already satisfied: packaging in /home/anshal/.local/lib/python3.8/site-packages (from pytest>
=3.4.0->pycountry_convert) (20.4)
Requirement already satisfied: pluggy<1.0,>=0.12 in /home/anshal/.local/lib/python3.8/site-packages (from
pytest>=3.4.0->pycountry_convert) (0.13.1)
Requirement already satisfied: iniconfig in /home/anshal/.local/lib/python3.8/site-packages (from pytest>
=3.4.0->pycountry_convert) (1.1.1)
Requirement already satisfied: six in /usr/lib/python3/dist-packages (from packaging->pytest>=3.4.0->pyco
untry_convert) (1.14.0)
Requirement already satisfied: pyparsing>=2.0.2 in /home/anshal/.local/lib/python3.8/site-packages (from
packaging->pytest>=3.4.0->pycountry_convert) (2.4.7)
Requirement already satisfied: folium in /home/anshal/.local/lib/python3.8/site-packages (0.11.0)
Requirement already satisfied: requests in /usr/lib/python3/dist-packages (from folium) (2.22.0)
Requirement already satisfied: branca>=0.3.0 in /home/anshal/.local/lib/python3.8/site-packages (from fol
ium) (0.4.1)
```

```
Requirement already satisfied: numpy in /usr/lib/python3/dist-packages (from folium) (1.17.4)
Requirement already satisfied: jinja2>=2.9 in /home/anshal/.local/lib/python3.8/site-packages (from foliu
m) (2.11.2)
Requirement already satisfied: MarkupSafe>=0.23 in /usr/lib/python3/dist-packages (from jinja2>=2.9->foli
um) (1.1.0)
--2020-10-20 14:22:17--  https://raw.githubusercontent.com/tarunk04/COVID-19-CaseStudy-and-Predictions/ma
ster/models/model_deaths.h5
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 199.232.252.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|199.232.252.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 111008 (108K) [application/octet-stream]
Saving to: 'model_deaths.h5.3'

model_deaths.h5.3   100%[===================>] 108.41K  --.-KB/s    in 0.07s

2020-10-20 14:22:18 (1.48 MB/s) - 'model_deaths.h5.3' saved [111008/111008]

--2020-10-20 14:22:18--  https://raw.githubusercontent.com/tarunk04/COVID-19-CaseStudy-and-Predictions/ma
ster/models/model_confirmed.h5
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 199.232.252.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|199.232.252.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 111008 (108K) [application/octet-stream]
Saving to: 'model_confirmed.h5.3'

model_confirmed.h5. 100%[===================>] 108.41K  --.-KB/s    in 0.1s

2020-10-20 14:22:18 (917 KB/s) - 'model_confirmed.h5.3' saved [111008/111008]

--2020-10-20 14:22:19--  https://raw.githubusercontent.com/tarunk04/COVID-19-CaseStudy-and-Predictions/ma
ster/models/model_usa_c.h5
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 199.232.252.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|199.232.252.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 111008 (108K) [application/octet-stream]
Saving to: 'model_usa_c.h5.3'

model_usa_c.h5.3    100%[===================>] 108.41K  --.-KB/s    in 0.1s

2020-10-20 14:22:19 (989 KB/s) - 'model_usa_c.h5.3' saved [111008/111008]
```

Imports and Datasets

Pandas - for dataset handeling

Numpy - Support for Pandas and calculations

Matplotlib - for visualization (Platting graphas)

pycountry_convert - Library for getting continent (name) to from their country names

folium - Library for Map

keras - Prediction Models

plotly - for interative plots

In [2]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import ticker
import pycountry_convert as pc
import folium
import branca
from datetime import datetime, timedelta,date
from scipy.interpolate import make_interp_spline, BSpline
import plotly.express as px
import json, requests
```

In [3]:

```
# from keras.layers import Input, Dense, Activation, LeakyReLU
# from keras import models
# from keras.optimizers import RMSprop, Adam
```

```
%matplotlib inline
```

In [4]:

```
df_confirmed = pd.read_csv('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_1
df_deaths = pd.read_csv('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_c

# Depricated
# df_recovered = pd.read_csv('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covi
df_covid19 = pd.read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/web-data/data/cases_c
df_table = pd.read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/web-data/data/cases_tim
```

```
/home/anshal/.local/lib/python3.8/site-packages/IPython/core/interactiveshell.py:3145: DtypeWarning: Colu
mns (11) have mixed types.Specify dtype option on import or set low_memory=False.
  has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

In [5]:

```
# new dataset
df_covid19.head(2)
```

Out[5]:

	Country_Region	Last_Update	Lat	Long_	Confirmed	Deaths	Recovered	Active	Incident_Rate	People_Tested	People_Hospitalized
0	Afghanistan	2020-10-20 08:24:14	33.93911	67.709953	40287.0	1497.0	33760.0	5030.0	103.490154	NaN	NaN
1	Albania	2020-10-20 08:24:14	41.15330	20.168300	17350.0	454.0	10167.0	6729.0	602.891097	NaN	NaN

In [6]:

```
df_confirmed.head(2)
```

Out[6]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	10/10/20	10/11/20	10/12/2
0	NaN	Afghanistan	33.93911	67.709953	0	0	0	0	0	0	...	39703	39799	3987
1	NaN	Albania	41.15330	20.168300	0	0	0	0	0	0	...	15231	15399	1557

2 rows × 276 columns

Preprocessing

In [7]:

```
df_confirmed = df_confirmed.rename(columns={"Province/State":"state","Country/Region": "country"})
df_deaths = df_deaths.rename(columns={"Province/State":"state","Country/Region": "country"})
df_covid19 = df_covid19.rename(columns={"Country_Region": "country"})
df_covid19["Active"] = df_covid19["Confirmed"]-df_covid19["Recovered"]-df_covid19["Deaths"]
# df_recovered = df_recovered.rename(columns={"Province/State":"state","Country/Region": "country"})
```

In [8]:

```
# Changing the conuntry names as required by pycountry_convert Lib
df_confirmed.loc[df_confirmed['country'] == "US", "country"] = "USA"
df_deaths.loc[df_deaths['country'] == "US", "country"] = "USA"
df_covid19.loc[df_covid19['country'] == "US", "country"] = "USA"
df_table.loc[df_table['Country_Region'] == "US", "Country_Region"] = "USA"
# df_recovered.loc[df_recovered['country'] == "US", "country"] = "USA"


df_confirmed.loc[df_confirmed['country'] == 'Korea, South', "country"] = 'South Korea'
df_deaths.loc[df_deaths['country'] == 'Korea, South', "country"] = 'South Korea'
df_covid19.loc[df_covid19['country'] == "Korea, South", "country"] = "South Korea"
df_table.loc[df_table['Country_Region'] == "Korea, South", "Country_Region"] = "South Korea"
# df_recovered.loc[df_recovered['country'] == 'Korea, South', "country"] = 'South Korea'

df_confirmed.loc[df_confirmed['country'] == 'Taiwan*', "country"] = 'Taiwan'
df_deaths.loc[df_deaths['country'] == 'Taiwan*', "country"] = 'Taiwan'
df_covid19.loc[df_covid19['country'] == "Taiwan*", "country"] = "Taiwan"
df_table.loc[df_table['Country_Region'] == "Taiwan*", "Country_Region"] = "Taiwan"
# df_recovered.loc[df_recovered['country'] == 'Taiwan*', "country"] = 'Taiwan'

df_confirmed.loc[df_confirmed['country'] == 'Congo (Kinshasa)', "country"] = 'Democratic Republic of the
df_deaths.loc[df_deaths['country'] == 'Congo (Kinshasa)', "country"] = 'Democratic Republic of the Congo'
df_covid19.loc[df_covid19['country'] == "Congo (Kinshasa)", "country"] = "Democratic Republic of the Cong
df_table.loc[df_table['Country_Region'] == "Congo (Kinshasa)", "Country_Region"] = "Democratic Republic o
# df_recovered.loc[df_recovered['country'] == 'Congo (Kinshasa)', "country"] = 'Democratic Republic of t
```

```
df_confirmed.loc[df_confirmed['country'] == "Cote d'Ivoire", "country"] = "Côte d'Ivoire"
df_deaths.loc[df_deaths['country'] == "Cote d'Ivoire", "country"] = "Côte d'Ivoire"
df_covid19.loc[df_covid19['country'] == "Cote d'Ivoire", "country"] = "Côte d'Ivoire"
df_table.loc[df_table['Country_Region'] == "Cote d'Ivoire", "Country_Region"] = "Côte d'Ivoire"
# df_recovered.loc[df_recovered['country'] == "Cote d'Ivoire", "country"] = "Côte d'Ivoire"

df_confirmed.loc[df_confirmed['country'] == "Reunion", "country"] = "Réunion"
df_deaths.loc[df_deaths['country'] == "Reunion", "country"] = "Réunion"
df_covid19.loc[df_covid19['country'] == "Reunion", "country"] = "Réunion"
df_table.loc[df_table['Country_Region'] == "Reunion", "Country_Region"] = "Réunion"
# df_recovered.loc[df_recovered['country'] == "Reunion", "country"] = "Réunion"

df_confirmed.loc[df_confirmed['country'] == 'Congo (Brazzaville)', "country"] = 'Republic of the Congo'
df_deaths.loc[df_deaths['country'] == 'Congo (Brazzaville)', "country"] = 'Republic of the Congo'
df_covid19.loc[df_covid19['country'] == "Congo (Brazzaville)", "country"] = "Republic of the Congo"
df_table.loc[df_table['Country_Region'] == "Congo (Brazzaville)", "Country_Region"] = "Republic of the Co
# df_recovered.loc[df_recovered['country'] == 'Congo (Brazzaville)', "country"] = 'Republic of the Congo

df_confirmed.loc[df_confirmed['country'] == 'Bahamas, The', "country"] = 'Bahamas'
df_deaths.loc[df_deaths['country'] == 'Bahamas, The', "country"] = 'Bahamas'
df_covid19.loc[df_covid19['country'] == "Bahamas, The", "country"] = "Bahamas"
df_table.loc[df_table['Country_Region'] == "Bahamas, The", "Country_Region"] = "Bahamas"
# df_recovered.loc[df_recovered['country'] == 'Bahamas, The', "country"] = 'Bahamas'

df_confirmed.loc[df_confirmed['country'] == 'Gambia, The', "country"] = 'Gambia'
df_deaths.loc[df_deaths['country'] == 'Gambia, The', "country"] = 'Gambia'
df_covid19.loc[df_covid19['country'] == "Gambia, The", "country"] = "Gambia"
df_table.loc[df_table['Country_Region'] == "Gambia", "Country_Region"] = "Gambia"
# df_recovered.loc[df_recovered['country'] == 'Gambia, The', "country"] = 'Gambia'

# getting all countries
countries = np.asarray(df_confirmed["country"])
countries1 = np.asarray(df_covid19["country"])
# Continent_code to Continent_names
continents = {
    'NA': 'North America',
    'SA': 'South America',
    'AS': 'Asia',
    'OC': 'Australia',
    'AF': 'Africa',
    'EU' : 'Europe',
    'na' : 'Others'
}

# Defininng Function for getting continent code for country.
def country_to_continent_code(country):
    try:
        return pc.country_alpha2_to_continent_code(pc.country_name_to_country_alpha2(country))
    except :
        return 'na'

#Collecting Continent Information
df_confirmed.insert(2,"continent", [continents[country_to_continent_code(country)] for country in countri
df_deaths.insert(2,"continent",  [continents[country_to_continent_code(country)] for country in countries
df_covid19.insert(1,"continent",  [continents[country_to_continent_code(country)] for country in countrie
df_table.insert(1,"continent",  [continents[country_to_continent_code(country)] for country in df_table["
# df_recovered.insert(2,"continent",  [continents[country_to_continent_code(country)] for country in cou
```

In [9]:

```
df_table = df_table[df_table["continent"] != "Others"]
```

In [10]:

```
df_deaths[df_deaths["continent" ]== 'Others']
```

Out[10]:

	state	country	continent	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	...	10/10/20	10/11/20	10/12/20	10/13/
34	NaN	Burma	Others	21.916200	95.956000	0	0	0	0	0	...	598	646	664	6
102	NaN	Diamond Princess	Others	0.000000	0.000000	0	0	0	0	0	...	13	13	13	
139	NaN	Holy See	Others	41.902900	12.453400	0	0	0	0	0	...	0	0	0	
156	NaN	Kosovo	Others	42.602636	20.902977	0	0	0	0	0	...	645	647	648	6
168	NaN	MS Zaandam	Others	0.000000	0.000000	0	0	0	0	0	...	2	2	2	
238	NaN	Timor-Leste	Others	-8.874217	125.727539	0	0	0	0	0	...	0	0	0	
262	NaN	West Bank and Gaza	Others	31.952200	35.233200	0	0	0	0	0	...	378	381	387	3
263	NaN	Western Sahara	Others	24.215500	-12.885800	0	0	0	0	0	...	1	1	1	

8 rows × 277 columns

In [11]:

```
#df_active = df_confirmed.copy()
#df_active.iloc[:,5:] = df_active.iloc[:,5:] - df_recovered.iloc[:,5:] - df_deaths.iloc[:,5:]
#df_active.head(5)
```

In [12]:

```
df_confirmed = df_confirmed.replace(np.nan, '', regex=True)
df_deaths = df_deaths.replace(np.nan, '', regex=True)
# df_recovered = df_recovered.replace(np.nan, '', regex=True)
# df_active = df_active.replace(np.nan, '', regex=True)
```

Defining Functions

plot_params()

visualize_covid_cases()

get_mortality_rate()

In [13]:

```
def plot_params(ax,axis_label= None, plt_title = None,label_size=15, axis_fsize = 15, title_fsize = 20, s
    # Tick-Parameters
    ax.xaxis.set_minor_locator(ticker.AutoMinorLocator())
    ax.yaxis.set_minor_locator(ticker.AutoMinorLocator())
    ax.tick_params(which='both', width=1,labelsize=label_size)
    ax.tick_params(which='major', length=6)
    ax.tick_params(which='minor', length=3, color='0.8')

    # Grid
    plt.grid(lw = 1, ls = '-', c = "0.7", which = 'major')
    plt.grid(lw = 1, ls = '-', c = "0.9", which = 'minor')

    # Plot Title
    plt.title( plt_title,{'fontsize':title_fsize})

    # Yaxis sacle
    plt.yscale(scale)
    plt.minorticks_on()
    # Plot Axes Labels
    xl = plt.xlabel(axis_label[0],fontsize = axis_fsize)
    yl = plt.ylabel(axis_label[1],fontsize = axis_fsize)

def visualize_covid_cases(confirmed, deaths, continent=None , country = None , state = None, period = Non
    x = 0
    if figure == None:
        f = plt.figure(figsize=(10,10))
        # Sub plot
        ax = f.add_subplot(111)
    else :
        f = figure[0]
```

```
        # Sub plot
        ax = f.add_subplot(figure[1],figure[2],figure[3])

    plt.tight_layout(pad=10, w_pad=5, h_pad=5)

    stats = [confirmed, deaths]
    label = ["Confirmed", "Deaths"]

    if continent != None:
        params = ["continent",continent]
    elif country != None:
        params = ["country",country]
    else:
        params = ["All", "All"]
    color = ["darkcyan","crimson"]
    marker_style = dict(linewidth=3, linestyle='-', marker='o',markersize=4, markerfacecolor='#ffffff')
    for i,stat in enumerate(stats):
        if params[1] == "All" :
            cases = np.sum(np.asarray(stat.iloc[:,5:]),axis = 0)[x:]
        else :
            cases = np.sum(np.asarray(stat[stat[params[0]] == params[1]].iloc[:,5:]),axis = 0)[x:]
        date = np.arange(1,cases.shape[0]+1)[x:]
        plt.plot(date,cases,label = label[i]+" (Total : "+str(cases[-1])+")",color=color[i],**marker_styl

    if params[1] == "All" :
        Total_confirmed = np.sum(np.asarray(stats[0].iloc[:,5:]),axis = 0)[x:]
        Total_deaths = np.sum(np.asarray(stats[1].iloc[:,5:]),axis = 0)[x:]
    else :
        Total_confirmed =  np.sum(np.asarray(stats[0][stat[params[0]] == params[1]].iloc[:,5:]),axis = 0)
        Total_deaths = np.sum(np.asarray(stats[1][stat[params[0]] == params[1]].iloc[:,5:]),axis = 0)[x:]

    text = "From "+stats[0].columns[5]+" to "+stats[0].columns[-1]+"\n"
    text += "Mortality rate : "+ str(int(Total_deaths[-1]/(Total_confirmed[-1])*10000)/100)+"\n"
    text += "Last 5 Days:\n"
    text += "Confirmed : " + str(Total_confirmed[-1] - Total_confirmed[-6])+"\n"
    text += "Deaths : " + str(Total_deaths[-1] - Total_deaths[-6])+"\n"
    text += "Last 24 Hours:\n"
    text += "Confirmed : " + str(Total_confirmed[-1] - Total_confirmed[-2])+"\n"
    text += "Deaths : " + str(Total_deaths[-1] - Total_deaths[-2])+"\n"

    plt.text(0.02, 0.78, text, fontsize=15, horizontalalignment='left', verticalalignment='top', transfor

    # Plot Axes Labels
    axis_label = ["Days ("+df_confirmed.columns[5]+" - "+df_confirmed.columns[-1]+")","No of Cases"]

    # Plot Parameters
    plot_params(ax,axis_label,scale = scale)

    # Plot Title
    if params[1] == "All" :
        plt.title("COVID-19 Cases World",{'fontsize':25})
    else:
        plt.title("COVID-19 Cases for "+params[1] ,{'fontsize':25})

    # Legend Location
    l = plt.legend(loc= "best",fontsize = 15)

    if figure == None:
        plt.show()

def get_total_cases(cases, country = "All"):
    if(country == "All") :
        return np.sum(np.asarray(cases.iloc[:,5:]),axis = 0)[-1]
    else :
        return np.sum(np.asarray(cases[cases["country"] == country].iloc[:,5:]),axis = 0)[-1]

def get_mortality_rate(confirmed,deaths, continent = None, country = None):
    if continent != None:
        params = ["continent",continent]
    elif country != None:
        params = ["country",country]
    else :
        params = ["All", "All"]

    if params[1] == "All" :
        Total_confirmed = np.sum(np.asarray(confirmed.iloc[:,5:]),axis = 0)
```

```
        Total_deaths = np.sum(np.asarray(deaths.iloc[:,5:]),axis = 0)
        mortality_rate = np.round((Total_deaths/Total_confirmed)*100,2)
    else :
        Total_confirmed =  np.sum(np.asarray(confirmed[confirmed[params[0]] == params[1]].iloc[:,5:]),axi
        Total_deaths = np.sum(np.asarray(deaths[deaths[params[0]] == params[1]].iloc[:,5:]),axis = 0)
        mortality_rate = np.round((Total_deaths/Total_confirmed)*100,2)

    return np.nan_to_num(mortality_rate)
def dd(date1,date2):
    return (datetime.strptime(date1,'%m/%d/%y') - datetime.strptime(date2,'%m/%d/%y')).days


out = ""+"output/"
```

General Analysis of Data

Getting country wise and continent wise data.

In [14]:

```
df_countries_cases = df_covid19.copy().drop(['Lat','Long_','continent','Last_Update'],axis =1)
df_countries_cases.index = df_countries_cases["country"]
df_countries_cases = df_countries_cases.drop(['country'],axis=1)

df_continents_cases = df_covid19.copy().drop(['Lat','Long_','country','Last_Update'],axis =1)
df_continents_cases = df_continents_cases.groupby(["continent"]).sum()
```

Global Reported Cases till Date

Total number of confirmed cases, deaths reported, revoveries and active cases all across the world

In [15]:

```
pd.DataFrame(df_countries_cases.sum()).transpose().style.background_gradient(cmap='Wistia',axis=1)
```

Out[15]:

	Confirmed	Deaths	Recovered	Active	Incident_Rate	People_Tested	People_Hospitalized	Mortality_Rate	
0	40411186.000000	1118398.000000	27707599.000000	11452393.000000	124451.141931	0.000000	0.000000	463.268616	9

Coninent Wise Reported Cases

Coninent Wise reported confirmed cases, recovered cases, deaths, active cases

In [16]:

```
df_continents_cases.style.background_gradient(cmap='Wistia')
```

Out[16]:

continent	Confirmed	Deaths	Recovered	Active	Incident_Rate	People_Tested	People_Hospitalized	Mortality_Rat
Africa	1655970.000000	39930.000000	1359734.000000	256306.000000	9597.074855	0.000000	0.000000	118.18828
Asia	12520839.000000	224191.000000	10944097.000000	1352551.000000	33613.059564	0.000000	0.000000	94.36002
Australia	29932.000000	939.000000	27526.000000	1464.000000	157.389173	0.000000	0.000000	12.07910
Europe	7189497.000000	240507.000000	2872388.000000	3943816.000000	44392.680353	0.000000	0.000000	105.11532
North America	9883695.000000	330287.000000	4616656.000000	4936752.000000	15026.859821	0.000000	0.000000	52.77733
Others	102617.000000	1999.000000	73850.000000	26761.000000	5282.628242	0.000000	0.000000	41.22886
South America	9028636.000000	280545.000000	7813348.000000	934743.000000	16381.449922	0.000000	0.000000	39.51968

Country Wise Reported Cases

Country Wise reported confirmed cases, recovered cases, deaths, active cases

In [17]:

```
df_countries_cases.sort_values('Confirmed', ascending= False).style.background_gradient(cmap='Wistia')
```

```
/home/anshal/.local/lib/python3.8/site-packages/pandas/io/formats/style.py:1126: RuntimeWarning: All-NaN
slice encountered
  smin = np.nanmin(s.to_numpy()) if vmin is None else vmin
/home/anshal/.local/lib/python3.8/site-packages/pandas/io/formats/style.py:1127: RuntimeWarning: All-NaN
```

```
slice encountered
  smax = np.nanmax(s.to_numpy()) if vmax is None else vmax
```

Out[17]:

country	Confirmed	Deaths	Recovered	Active	Incident_Rate	People_Tested	People_Hospitalized	Mortality_Ra
USA	8214754.000000	220133.000000	3272603.000000	4722018.000000	2493.351952	nan	nan	2.6797
India	7597063.000000	115197.000000	6733328.000000	748538.000000	550.510062	nan	nan	1.5163
Brazil	5250727.000000	154176.000000	4526393.000000	570158.000000	2470.239744	nan	nan	2.9362
Russia	1406667.000000	24205.000000	1070920.000000	311542.000000	963.903248	nan	nan	1.7207
Argentina	1002662.000000	26716.000000	803965.000000	171981.000000	2218.486032	nan	nan	2.6645
Spain	974449.000000	33992.000000	150376.000000	790081.000000	2084.169656	nan	nan	3.4883
Colombia	965883.000000	29102.000000	867961.000000	68820.000000	1898.247356	nan	nan	3.0129
France	952600.000000	33647.000000	109611.000000	809342.000000	1459.397496	nan	nan	3.5321
Peru	868675.000000	33759.000000	784056.000000	50860.000000	2634.596195	nan	nan	3.8862
Mexico	854926.000000	86338.000000	727759.000000	40829.000000	668.996562	nan	nan	10.0988
United Kingdom	744122.000000	43816.000000	2613.000000	697693.000000	1096.134632	nan	nan	5.8882
South Africa	705254.000000	18492.000000	635257.000000	51505.000000	1189.124224	nan	nan	2.6220
Iran	534631.000000	30712.000000	431360.000000	72559.000000	636.518876	nan	nan	5.7445
Chile	493305.000000	13676.000000	465021.000000	14608.000000	2580.558729	nan	nan	2.7723
Iraq	430678.000000	10317.000000	363532.000000	56829.000000	1070.738934	nan	nan	2.3955
Italy	423578.000000	36616.000000	252959.000000	134003.000000	700.570945	nan	nan	8.6444
Bangladesh	390206.000000	5681.000000	305599.000000	78926.000000	236.934521	nan	nan	1.4558
Germany	377068.000000	9842.000000	298599.000000	68627.000000	450.048037	nan	nan	2.6101
Indonesia	365240.000000	12617.000000	289243.000000	63380.000000	133.531429	nan	nan	3.4544
Philippines	360775.000000	6690.000000	310642.000000	43443.000000	329.231089	nan	nan	1.8543
Turkey	349519.000000	9371.000000	305427.000000	34721.000000	414.421231	nan	nan	2.6811
Saudi Arabia	342583.000000	5201.000000	328895.000000	8487.000000	984.041790	nan	nan	1.5181
Pakistan	324034.000000	6673.000000	308010.000000	9351.000000	146.693187	nan	nan	2.0593
Ukraine	317756.000000	5947.000000	134517.000000	177292.000000	726.569148	nan	nan	1.8715
Israel	305348.000000	2268.000000	279729.000000	23351.000000	3527.774867	nan	nan	0.7427
Netherlands	242217.000000	6828.000000	5294.000000	230095.000000	1413.590868	nan	nan	2.8189
Belgium	230480.000000	10443.000000	21214.000000	198823.000000	1988.676760	nan	nan	4.5309
Canada	204111.000000	9832.000000	172406.000000	21873.000000	539.181654	nan	nan	4.8169
Poland	183248.000000	3614.000000	94014.000000	85620.000000	484.186098	nan	nan	1.9721
Romania	182854.000000	5931.000000	132082.000000	44841.000000	950.499130	nan	nan	3.2435
Czechia	181962.000000	1513.000000	74908.000000	105541.000000	1699.153103	nan	nan	0.8314
Morocco	175749.000000	2976.000000	146421.000000	26352.000000	476.148315	nan	nan	1.6933
Ecuador	153423.000000	12395.000000	134187.000000	6841.000000	869.594050	nan	nan	8.0789
Bolivia	139890.000000	8502.000000	104957.000000	26431.000000	1198.403602	nan	nan	6.0776
Nepal	136036.000000	757.000000	94501.000000	40778.000000	466.887107	nan	nan	0.5564
Qatar	129671.000000	224.000000	126650.000000	2797.000000	4500.808730	nan	nan	0.1727
Panama	125181.000000	2574.000000	101545.000000	21062.000000	2901.222036	nan	nan	2.0562
Dominican Republic	121667.000000	2203.000000	98880.000000	20584.000000	1121.571504	nan	nan	1.8106
Kuwait	116832.000000	710.000000	108606.000000	7516.000000	2735.751703	nan	nan	0.6077
United Arab Emirates	116517.000000	466.000000	108811.000000	7240.000000	1178.081776	nan	nan	0.3999
Oman	110594.000000	1114.000000	96400.000000	13080.000000	2165.697794	nan	nan	1.0072

country	Confirmed	Deaths	Recovered	Active	Incident_Rate	People_Tested	People_Hospitalized	Mortality_Ra
Kazakhstan	109623.000000	1768.000000	105145.000000	2710.000000	583.824416	nan	nan	1.6128
Egypt	105547.000000	6130.000000	98314.000000	1103.000000	103.139313	nan	nan	5.8078
Sweden	103200.000000	5918.000000	nan	nan	1021.856035	nan	nan	5.7344
Portugal	101860.000000	2198.000000	59966.000000	39696.000000	998.949955	nan	nan	2.1578
Guatemala	101599.000000	3541.000000	91032.000000	7026.000000	567.098993	nan	nan	3.4852
Costa Rica	97075.000000	1204.000000	59580.000000	36291.000000	1905.630695	nan	nan	1.2402
Japan	93607.000000	1676.000000	85310.000000	6621.000000	74.011402	nan	nan	1.7904
China	91006.000000	4739.000000	85840.000000	427.000000	6.478788	nan	nan	5.2073
Ethiopia	89860.000000	1365.000000	43149.000000	45346.000000	78.163883	nan	nan	1.5190
Honduras	89381.000000	2576.000000	35398.000000	51407.000000	902.418349	nan	nan	2.8820
Belarus	88290.000000	933.000000	80130.000000	7227.000000	934.352849	nan	nan	1.0567
Venezuela	87161.000000	741.000000	80316.000000	6104.000000	306.517002	nan	nan	0.8501
Switzerland	83159.000000	2138.000000	53400.000000	27621.000000	960.862744	nan	nan	2.5709
Bahrain	78224.000000	302.000000	74683.000000	3239.000000	4597.131024	nan	nan	0.3860
Moldova	67302.000000	1600.000000	48493.000000	17209.000000	1668.384167	nan	nan	2.3773
Armenia	66694.000000	1101.000000	48734.000000	16859.000000	2250.716616	nan	nan	1.6508
Austria	65927.000000	904.000000	50359.000000	14664.000000	732.001688	nan	nan	1.3712
Uzbekistan	63737.000000	533.000000	60717.000000	2487.000000	190.434793	nan	nan	0.8362
Lebanon	62944.000000	526.000000	28855.000000	33563.000000	922.196687	nan	nan	0.8356
Nigeria	61558.000000	1125.000000	56697.000000	3736.000000	29.862289	nan	nan	1.8275
Singapore	57921.000000	28.000000	57819.000000	74.000000	990.044515	nan	nan	0.0483
Paraguay	55452.000000	1207.000000	36663.000000	17582.000000	777.452040	nan	nan	2.1766
Algeria	54616.000000	1865.000000	38215.000000	14536.000000	124.548919	nan	nan	3.4147
Kyrgyzstan	52526.000000	1111.000000	45863.000000	5552.000000	805.095988	nan	nan	2.1151
Ireland	50993.000000	1852.000000	23364.000000	25777.000000	1032.707710	nan	nan	3.6318
Libya	49949.000000	732.000000	27262.000000	21955.000000	726.923501	nan	nan	1.4654
Hungary	48757.000000	1211.000000	14637.000000	32909.000000	504.712562	nan	nan	2.4837
West Bank and Gaza	47616.000000	413.000000	40861.000000	6342.000000	933.387906	nan	nan	0.8673
Ghana	47372.000000	310.000000	46664.000000	398.000000	152.454169	nan	nan	0.6543
Azerbaijan	45295.000000	630.000000	40144.000000	4521.000000	446.732599	nan	nan	1.3908
Kenya	45076.000000	839.000000	32084.000000	12153.000000	83.829106	nan	nan	1.8613
Tunisia	42727.000000	687.000000	5032.000000	37008.000000	361.522811	nan	nan	1.6078
Afghanistan	40287.000000	1497.000000	33760.000000	5030.000000	103.490154	nan	nan	3.7158
Jordan	38937.000000	380.000000	7006.000000	31551.000000	381.617816	nan	nan	0.9759
Burma	37205.000000	914.000000	17568.000000	18723.000000	68.379233	nan	nan	2.4566
Denmark	36345.000000	686.000000	30048.000000	5611.000000	627.481461	nan	nan	1.8874
Serbia	36282.000000	778.000000	nan	nan	415.250813	nan	nan	2.1443
Bosnia and Herzegovina	34661.000000	997.000000	25442.000000	8222.000000	1056.475297	nan	nan	2.8764
El Salvador	31666.000000	926.000000	27000.000000	3740.000000	488.205654	nan	nan	2.9242
Slovakia	31400.000000	98.000000	8004.000000	23298.000000	575.129180	nan	nan	0.3121
Bulgaria	30527.000000	1008.000000	17153.000000	12366.000000	439.335707	nan	nan	3.3019
Australia	27429.000000	905.000000	25126.000000	1398.000000	107.734969	nan	nan	3.2994
Croatia	26863.000000	382.000000	20962.000000	5519.000000	654.354356	nan	nan	1.4220
Greece	25802.000000	520.000000	1347.000000	23935.000000	247.547360	nan	nan	2.0153
South Korea	25333.000000	447.000000	23466.000000	1420.000000	49.411749	nan	nan	1.7644
North	23788.000000	846.000000	17392.000000	5550.000000	1141.798424	nan	nan	3.5564

	Confirmed	Deaths	Recovered	Active	Incident_Rate	People_Tested	People_Hospitalized	Mortality_Ra
Macedonia								
Cameroon country	21506.000000	424.000000	20117.000000	965.000000	81.014504	nan	nan	1.9715
Malaysia	21363.000000	190.000000	13717.000000	7456.000000	66.004453	nan	nan	0.8893
Côte d'Ivoire	20324.000000	121.000000	20029.000000	174.000000	77.048253	nan	nan	0.5953
Georgia	19857.000000	158.000000	8666.000000	11033.000000	497.772096	nan	nan	0.7956
Albania	17350.000000	454.000000	10167.000000	6729.000000	602.891097	nan	nan	2.6167
Kosovo	17009.000000	656.000000	14711.000000	1642.000000	939.533774	nan	nan	3.8567
Madagascar	16810.000000	238.000000	16215.000000	357.000000	60.705603	nan	nan	1.4158
Norway	16603.000000	278.000000	11863.000000	4462.000000	306.258234	nan	nan	1.6743
Zambia	15897.000000	346.000000	15031.000000	520.000000	86.472139	nan	nan	2.1765
Montenegro	15760.000000	240.000000	11288.000000	4232.000000	2509.306406	nan	nan	1.5228
Senegal	15432.000000	319.000000	13865.000000	1248.000000	92.164743	nan	nan	2.0671
Sudan	13724.000000	836.000000	6764.000000	6124.000000	31.298127	nan	nan	6.0915
Slovenia	13679.000000	190.000000	6385.000000	7104.000000	657.982079	nan	nan	1.3889
Finland	13555.000000	351.000000	9100.000000	4104.000000	244.643384	nan	nan	2.5894
Namibia	12326.000000	131.000000	10426.000000	1769.000000	485.100649	nan	nan	1.0627
Guinea	11518.000000	70.000000	10427.000000	1021.000000	87.704123	nan	nan	0.6077
Maldives	11232.000000	37.000000	10201.000000	994.000000	2077.914390	nan	nan	0.3294
Mozambique	11080.000000	75.000000	8836.000000	2169.000000	35.449835	nan	nan	0.6768
Democratic Republic of the Congo	11052.000000	303.000000	10357.000000	392.000000	12.340137	nan	nan	2.7415
Luxembourg	11010.000000	135.000000	8471.000000	2404.000000	1758.853375	nan	nan	1.2261
Uganda	10691.000000	97.000000	6992.000000	3602.000000	23.372904	nan	nan	0.9073
Tajikistan	10533.000000	80.000000	9563.000000	890.000000	110.436102	nan	nan	0.7595
Haiti	8976.000000	231.000000	7303.000000	1442.000000	78.719351	nan	nan	2.5735
Gabon	8884.000000	54.000000	8452.000000	378.000000	399.150300	nan	nan	0.6078
Jamaica	8321.000000	173.000000	3951.000000	4197.000000	281.004646	nan	nan	2.0790
Zimbabwe	8159.000000	232.000000	7683.000000	244.000000	54.894975	nan	nan	2.8434
Lithuania	7928.000000	118.000000	3276.000000	4534.000000	291.225295	nan	nan	1.4883
Angola	7829.000000	248.000000	3031.000000	4550.000000	23.820776	nan	nan	3.1677
Cabo Verde	7800.000000	87.000000	6620.000000	1093.000000	1402.907976	nan	nan	1.1153
Mauritania	7621.000000	163.000000	7355.000000	103.000000	163.904458	nan	nan	2.1388
Cuba	6258.000000	127.000000	5780.000000	351.000000	55.250394	nan	nan	2.0294
Malawi	5860.000000	181.000000	4757.000000	922.000000	30.632586	nan	nan	3.0887
Eswatini	5788.000000	116.000000	5427.000000	245.000000	498.894984	nan	nan	2.0041
Bahamas	5773.000000	123.000000	3339.000000	2311.000000	1468.030352	nan	nan	2.1306
Sri Lanka	5625.000000	13.000000	3440.000000	2172.000000	26.268782	nan	nan	0.2311
Botswana	5609.000000	21.000000	915.000000	4673.000000	238.515920	nan	nan	0.3743
Djibouti	5469.000000	61.000000	5379.000000	29.000000	553.541390	nan	nan	1.1153
Nicaragua	5353.000000	154.000000	4225.000000	974.000000	80.805440	nan	nan	2.8768
Trinidad and Tobago	5298.000000	97.000000	3696.000000	1505.000000	378.566207	nan	nan	1.8308
Republic of the Congo	5156.000000	92.000000	3887.000000	1177.000000	93.438094	nan	nan	1.7843
Syria	5134.000000	251.000000	1565.000000	3318.000000	29.336041	nan	nan	4.8889
Suriname	5133.000000	109.000000	4959.000000	65.000000	874.991903	nan	nan	2.1235
Equatorial Guinea	5070.000000	83.000000	4954.000000	33.000000	361.372360	nan	nan	1.6370

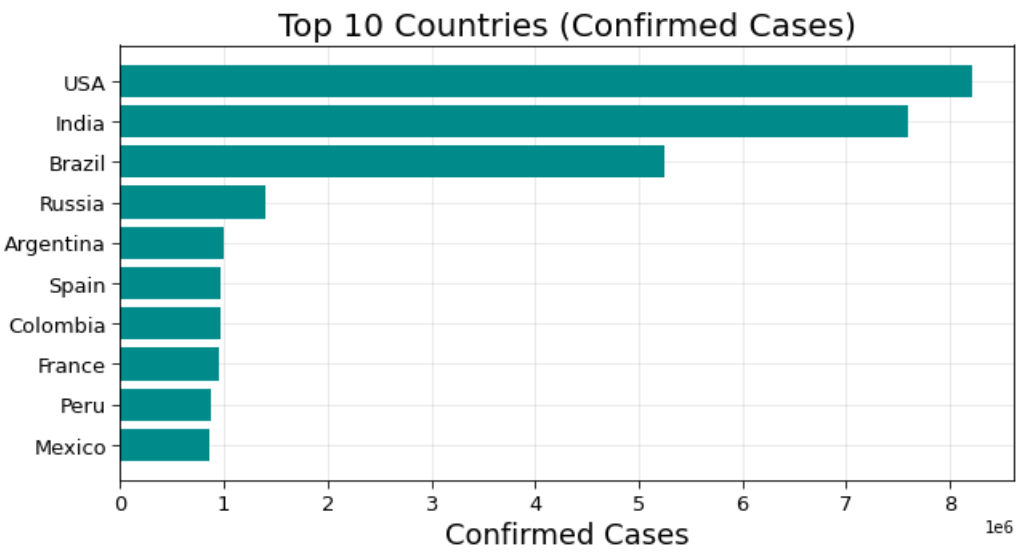
country	Confirmed	Deaths	Recovered	Active	Incident_Rate	People_Tested	People_Hospitalized	Mortality_Ra
Rwanda	4992.000000	34.000000	4797.000000	161.000000	38.541688	nan	nan	0.6810
Central African Republic	4856.000000	62.000000	1924.000000	2870.000000	100.543215	nan	nan	1.2767
Malta	4737.000000	45.000000	3242.000000	1450.000000	1072.838413	nan	nan	0.9499
Estonia	4127.000000	68.000000	3270.000000	789.000000	311.110341	nan	nan	1.6476
Iceland	4101.000000	11.000000	2856.000000	1234.000000	1201.758242	nan	nan	0.2682
Somalia	3890.000000	99.000000	3089.000000	702.000000	24.475847	nan	nan	2.5449
Guyana	3765.000000	111.000000	2749.000000	905.000000	478.667207	nan	nan	2.9482
Thailand	3700.000000	59.000000	3491.000000	150.000000	5.300861	nan	nan	1.5945
Gambia	3649.000000	118.000000	2649.000000	882.000000	150.993270	nan	nan	3.2337
Andorra	3623.000000	62.000000	2273.000000	1288.000000	4689.057141	nan	nan	1.7112
Latvia	3609.000000	47.000000	1341.000000	2221.000000	191.336877	nan	nan	1.3023
Mali	3407.000000	132.000000	2588.000000	687.000000	16.823998	nan	nan	3.8743
South Sudan	2847.000000	55.000000	1290.000000	1502.000000	25.433884	nan	nan	1.9318
Belize	2833.000000	45.000000	1692.000000	1096.000000	712.487520	nan	nan	1.5884
Cyprus	2687.000000	25.000000	1444.000000	1218.000000	222.551499	nan	nan	0.9304
Uruguay	2560.000000	51.000000	2121.000000	388.000000	73.696062	nan	nan	1.9921
Benin	2496.000000	41.000000	2330.000000	125.000000	20.588627	nan	nan	1.6426
Guinea-Bissau	2403.000000	41.000000	1818.000000	544.000000	122.103783	nan	nan	1.7062
Burkina Faso	2387.000000	65.000000	1802.000000	520.000000	11.419262	nan	nan	2.7230
Sierra Leone	2331.000000	73.000000	1760.000000	498.000000	29.221567	nan	nan	3.1317
Togo	2071.000000	51.000000	1541.000000	479.000000	25.015893	nan	nan	2.4625
Yemen	2056.000000	597.000000	1338.000000	121.000000	6.893322	nan	nan	29.0369
New Zealand	1887.000000	25.000000	1829.000000	33.000000	39.131249	nan	nan	1.3248
Lesotho	1833.000000	42.000000	961.000000	830.000000	85.564163	nan	nan	2.2913
Chad	1390.000000	93.000000	1194.000000	103.000000	8.462267	nan	nan	6.6906
Liberia	1381.000000	82.000000	1271.000000	28.000000	27.305026	nan	nan	5.9377
Niger	1211.000000	69.000000	1128.000000	14.000000	5.002760	nan	nan	5.6977
Vietnam	1140.000000	35.000000	1046.000000	59.000000	1.171170	nan	nan	3.0701
Sao Tome and Principe	933.000000	15.000000	898.000000	20.000000	425.714429	nan	nan	1.6077
San Marino	766.000000	42.000000	688.000000	36.000000	2257.056986	nan	nan	5.4830
Diamond Princess	712.000000	13.000000	659.000000	40.000000	nan	nan	nan	1.8258
Papua New Guinea	581.000000	7.000000	541.000000	33.000000	6.493777	nan	nan	1.2048
Burundi	549.000000	1.000000	497.000000	51.000000	4.617022	nan	nan	0.1821
Taiwan	540.000000	7.000000	493.000000	40.000000	2.267309	nan	nan	1.2962
Tanzania	509.000000	21.000000	183.000000	305.000000	0.852108	nan	nan	4.1257
Comoros	502.000000	7.000000	485.000000	10.000000	57.728023	nan	nan	1.3944
Eritrea	452.000000	0.000000	388.000000	64.000000	12.745222	nan	nan	0.0000
Mauritius	419.000000	10.000000	379.000000	30.000000	32.946287	nan	nan	2.3866
Bhutan	330.000000	0.000000	301.000000	29.000000	42.767609	nan	nan	0.0000
Mongolia	326.000000	0.000000	312.000000	14.000000	9.944203	nan	nan	0.0000
Cambodia	285.000000	0.000000	280.000000	5.000000	1.704650	nan	nan	0.0000
Monaco	268.000000	2.000000	222.000000	44.000000	682.906941	nan	nan	0.7462
Liechtenstein	224.000000	1.000000	142.000000	81.000000	587.356111	nan	nan	0.4464
Barbados	222.000000	7.000000	203.000000	12.000000	77.253054	nan	nan	3.1531

	Confirmed	Deaths	Recovered	Active	Incident_Rate	People_Tested	People_Hospitalized	Mortality_Ra
Barbados	222.000000	7.000000	205.000000	12.000000	77.252054	nan	nan	3.1531
Seychelles	149.000000	0.000000	148.000000	1.000000	151.515152	nan	nan	0.0000
country Brunei	147.000000	3.000000	143.000000	1.000000	33.601306	nan	nan	2.0408
Antigua and Barbuda	119.000000	3.000000	101.000000	15.000000	121.517850	nan	nan	2.5210
Saint Vincent and the Grenadines	67.000000	0.000000	64.000000	3.000000	60.389195	nan	nan	0.0000
Saint Lucia	36.000000	0.000000	27.000000	9.000000	19.604747	nan	nan	0.0000
Dominica	33.000000	0.000000	29.000000	4.000000	45.839063	nan	nan	0.0000
Fiji	32.000000	2.000000	30.000000	0.000000	3.569660	nan	nan	6.2500
Timor-Leste	29.000000	0.000000	28.000000	1.000000	2.199566	nan	nan	0.0000
Holy See	27.000000	0.000000	15.000000	12.000000	3337.453646	nan	nan	0.0000
Grenada	27.000000	0.000000	24.000000	3.000000	23.995947	nan	nan	0.0000
Laos	23.000000	0.000000	22.000000	1.000000	0.316127	nan	nan	0.0000
Saint Kitts and Nevis	19.000000	0.000000	19.000000	0.000000	35.719657	nan	nan	0.0000
Western Sahara	10.000000	1.000000	8.000000	1.000000	1.674116	nan	nan	10.0000
MS Zaandam	9.000000	2.000000	nan	nan	nan	nan	nan	22.2222
Solomon Islands	3.000000	0.000000	nan	nan	0.459518	nan	nan	0.0000

Top 10 countries (Confirmed Cases and Deaths)

In [18]:

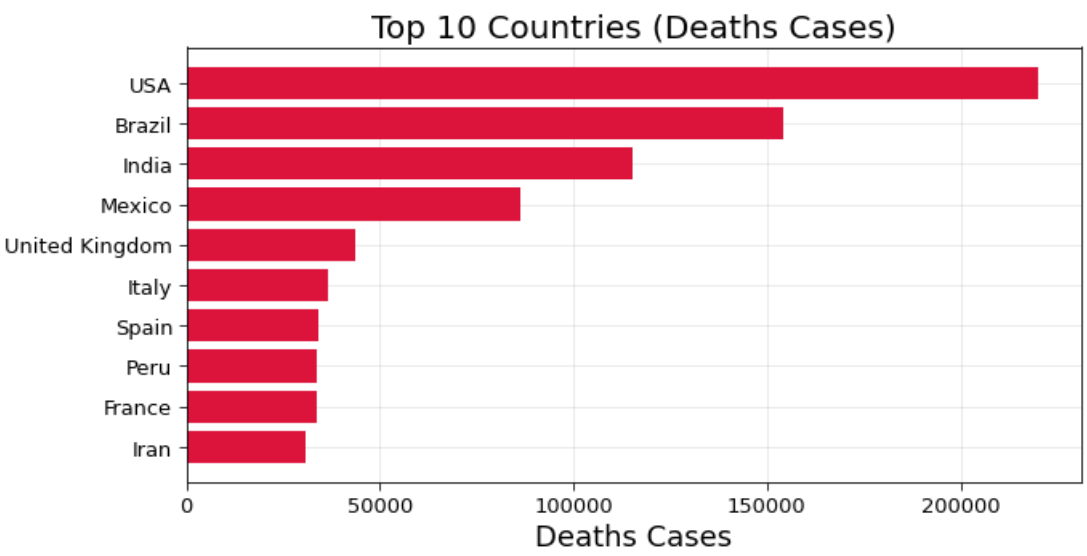
```
f = plt.figure(figsize=(10,5))
f.add_subplot(111)

plt.axes(axisbelow=True)
plt.barh(df_countries_cases.sort_values('Confirmed')["Confirmed"].index[-10:],df_countries_cases.sort_val
plt.tick_params(size=5,labelsize = 13)
plt.xlabel("Confirmed Cases",fontsize=18)
plt.title("Top 10 Countries (Confirmed Cases)",fontsize=20)
plt.grid(alpha=0.3)
```



In [19]:

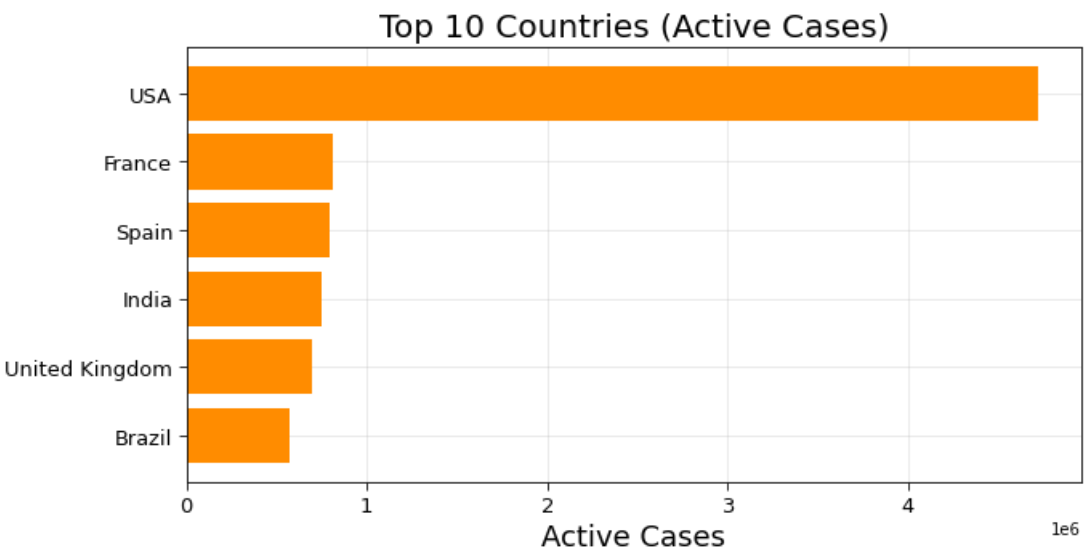
```
f = plt.figure(figsize=(10,5))
f.add_subplot(111)

plt.axes(axisbelow=True)
plt.barh(df_countries_cases.sort_values('Deaths')["Deaths"].index[-10:],df_countries_cases.sort_values('D
plt.tick_params(size=5,labelsize = 13)
plt.xlabel("Deaths Cases",fontsize=18)
```

```
plt.title("Top 10 Countries (Deaths Cases)",fontsize=20)
plt.grid(alpha=0.3,which='both')
```



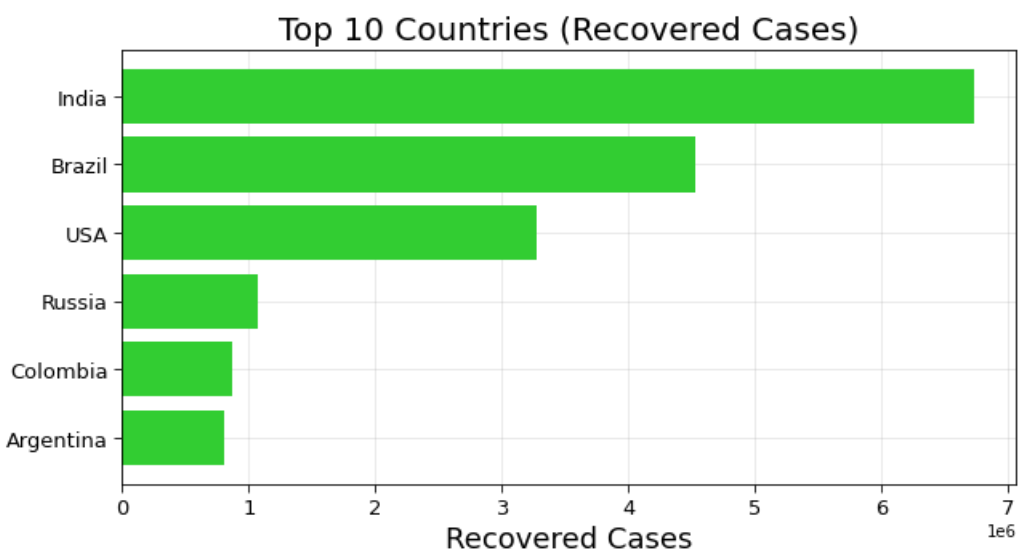
In [20]:

```
f = plt.figure(figsize=(10,5))
f.add_subplot(111)

plt.axes(axisbelow=True)
plt.barh(df_countries_cases.sort_values('Active')["Active"].index[-10:],df_countries_cases.sort_values('A
plt.tick_params(size=5,labelsize = 13)
plt.xlabel("Active Cases",fontsize=18)
plt.title("Top 10 Countries (Active Cases)",fontsize=20)
plt.grid(alpha=0.3,which='both')
```



In [21]:

```
f = plt.figure(figsize=(10,5))
f.add_subplot(111)

plt.axes(axisbelow=True)
plt.barh(df_countries_cases.sort_values('Recovered')["Recovered"].index[-10:],df_countries_cases.sort_val
plt.tick_params(size=5,labelsize = 13)
plt.xlabel("Recovered Cases",fontsize=18)
plt.title("Top 10 Countries (Recovered Cases)",fontsize=20)
plt.grid(alpha=0.3,which='both')
```



Correlation Analysis

Plotting Heat map of correlation of confirmed cases, recovered cases, deaths and active cases.

Country wise Correlation

In [22]:

```
df_countries_cases.corr().style.background_gradient(cmap='Reds')
```

Out[22]:

	Confirmed	Deaths	Recovered	Active	Incident_Rate	People_Tested	People_Hospitalized	Mortality_Rate	UID
Confirmed	1.000000	0.933806	0.935365	0.783330	0.228404	nan	nan	0.024794	-0.013884
Deaths	0.933806	1.000000	0.821013	0.816850	0.254287	nan	nan	0.114383	-0.014337
Recovered	0.935365	0.821013	1.000000	0.512967	0.192615	nan	nan	0.025633	-0.022155
Active	0.783330	0.816850	0.512967	1.000000	0.212759	nan	nan	0.039653	0.029959
Incident_Rate	0.228404	0.254287	0.192615	0.212759	1.000000	nan	nan	-0.066788	-0.097105
People_Tested	nan	nan	nan	nan	nan	nan	nan	nan	nan
People_Hospitalized	nan	nan	nan	nan	nan	nan	nan	nan	nan
Mortality_Rate	0.024794	0.114383	0.025633	0.039653	-0.066788	nan	nan	1.000000	0.324959
UID	-0.013884	-0.014337	-0.022155	0.029959	-0.097105	nan	nan	0.324959	1.000000

Continent Wise Correlation

In [23]:

```
df_continents_cases.corr().style.background_gradient(cmap='Reds')
```

Out[23]:

	Confirmed	Deaths	Recovered	Active	Incident_Rate	People_Tested	People_Hospitalized	Mortality_Rate	UID
Confirmed	1.000000	0.912685	0.914999	0.600352	0.688374	nan	nan	0.280171	-0.114169
Deaths	0.912685	1.000000	0.712945	0.785378	0.619909	nan	nan	0.171571	-0.251449
Recovered	0.914999	0.712945	1.000000	0.226839	0.544131	nan	nan	0.220776	-0.093408
Active	0.600352	0.785378	0.226839	1.000000	0.574722	nan	nan	0.237744	-0.087056
Incident_Rate	0.688374	0.619909	0.544131	0.574722	1.000000	nan	nan	0.635821	0.304745
People_Tested	nan	nan	nan	nan	nan	nan	nan	nan	nan
People_Hospitalized	nan	nan	nan	nan	nan	nan	nan	nan	nan
Mortality_Rate	0.280171	0.171571	0.220776	0.237744	0.635821	nan	nan	1.000000	0.783808
UID	-0.114169	-0.251449	-0.093408	-0.087056	0.304745	nan	nan	0.783808	1.000000

Visualization on Map

In [24]:

```
world_map = folium.Map(location=[10,0], tiles="cartodbpositron", zoom_start=2,max_zoom=6,min_zoom=2)
for i in range(0,len(df_confirmed)):
    folium.Circle(
        location=[df_confirmed.iloc[i]['Lat'], df_confirmed.iloc[i]['Long']],
        tooltip = "<h5 style='text-align:center;font-weight: bold'>"+df_confirmed.iloc[i]['country']+"</h
                  "<div style='text-align:center;'>"+str(np.nan_to_num(df_confirmed.iloc[i]['state']))+
                  "<hr style='margin:10px;'>"+
                  "<ul style='color: #444;list-style-type:circle;align-item:left;padding-left:20px;pad
        "<li>Confirmed: "+str(df_confirmed.iloc[i,-1])+"</li>"+
        "<li>Deaths:   "+str(df_deaths.iloc[i,-1])+"</li>"+
        "<li>Mortality Rate:   "+str(np.round(df_deaths.iloc[i,-1]/(df_confirmed.iloc[i,-1]+1.00001)*100,
        "</ul>"
        ,
        radius=(int((np.log(df_confirmed.iloc[i,-1]+1.00001)))+0.2)*50000,
        color='#ff6600',
        fill_color='#ff8533',
        fill=True).add_to(world_map)

world_map
```

Out[24]:

Make this Notebook Trusted to load map: File -> Trust Notebook

Global Confirmed Cases Heat Map

In [25]:

```
temp_df = pd.DataFrame(df_countries_cases['Confirmed'])
temp_df = temp_df.reset_index()
fig = px.choropleth(temp_df, locations="country",
                    color=np.log10(temp_df.iloc[:,-1]), # lifeExp is a column of gapminder
                    hover_name="country", # column to add to hover information
                    hover_data=["Confirmed"],
                    color_continuous_scale=px.colors.sequential.Plasma,locationmode="country names")
fig.update_geos(fitbounds="locations", visible=False)
fig.update_layout(title_text="Confirmed Cases Heat Map (Log Scale)")
fig.update_coloraxes(colorbar_title="Confirmed Cases(Log Scale)",colorscale="Reds")
# fig.to_image("Global Heat Map confirmed.png")
fig.show()
```

COVID-19: Spread Progression

In [26]:

```
df_data = df_table.groupby(['Last_Update', 'Country_Region'])['Confirmed', 'Deaths'].max().reset_index()
df_data["Last_Update"] = pd.to_datetime( df_data["Last_Update"]).dt.strftime('%m/%d/%Y')

fig = px.scatter_geo(df_data, locations="Country_Region", locationmode='country names',
                     color=np.power(df_data["Confirmed"],0.3)-2 , size= np.power(df_data["Confirmed"]+1,0
                     hover_data=["Confirmed"],
                     range_color= [0, max(np.power(df_data["Confirmed"],0.3))],
                     projection="natural earth", animation_frame="Last_Update",
                     color_continuous_scale=px.colors.sequential.Plasma,
                     title='COVID-19: Progression of spread'
                    )
fig.update_coloraxes(colorscale="hot")
fig.update(layout_coloraxis_showscale=False)
fig.show()
```

```
<ipython-input-26-183ed341048f>:1: FutureWarning:

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

COVID-19 Spread Analysis

Spread Analysis is in two sections

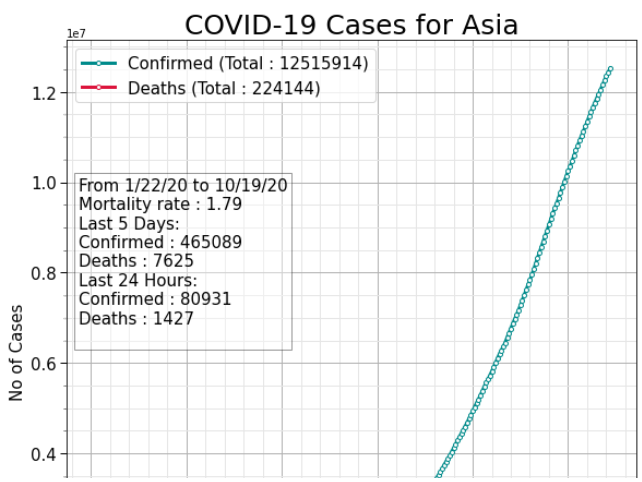
Spread Across Globe

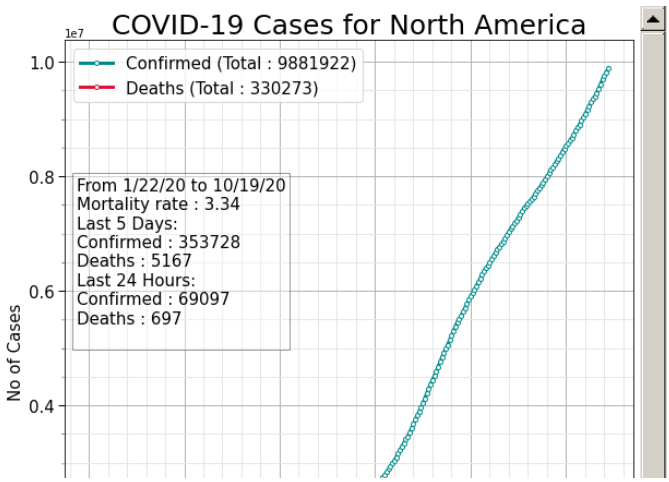
Spread Trends in the World, Continents and few most affected Countries

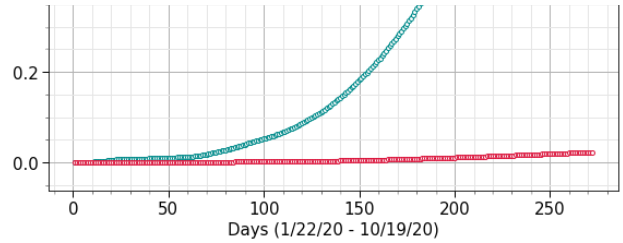
2 . COVID-19 Spread Trends in Different Continents

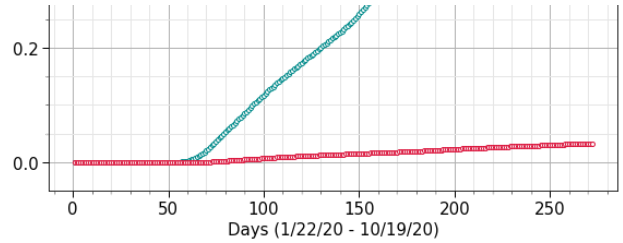
In [27]:

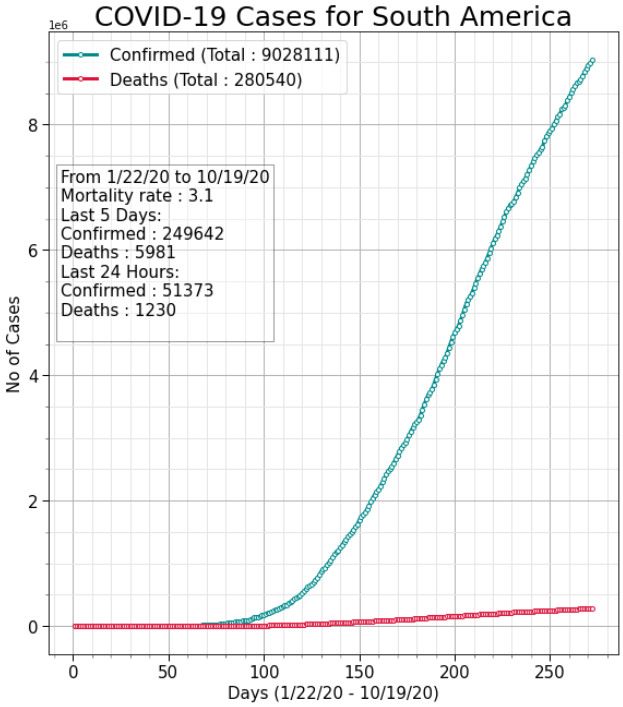
```
df_continents= df_confirmed.groupby(["continent"]).sum()
continents = df_continents.sort_values(df_continents.columns[-1],ascending = False).index

cols =2
rows = int(np.ceil(continents.shape[0]/cols))
f = plt.figure(figsize=(20,10*rows))
for i,continent in enumerate(continents):
    visualize_covid_cases(df_confirmed, df_deaths, continent = continent,figure = [f,rows,cols, i+1])

plt.show()
```

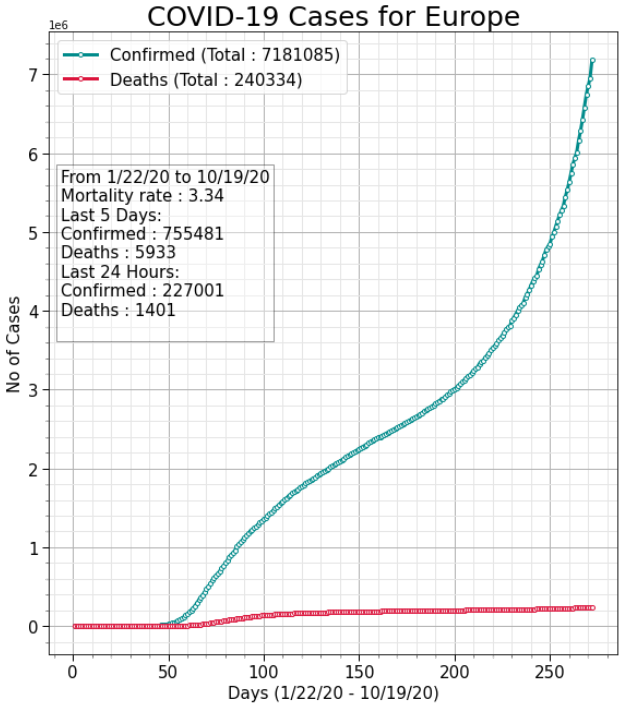


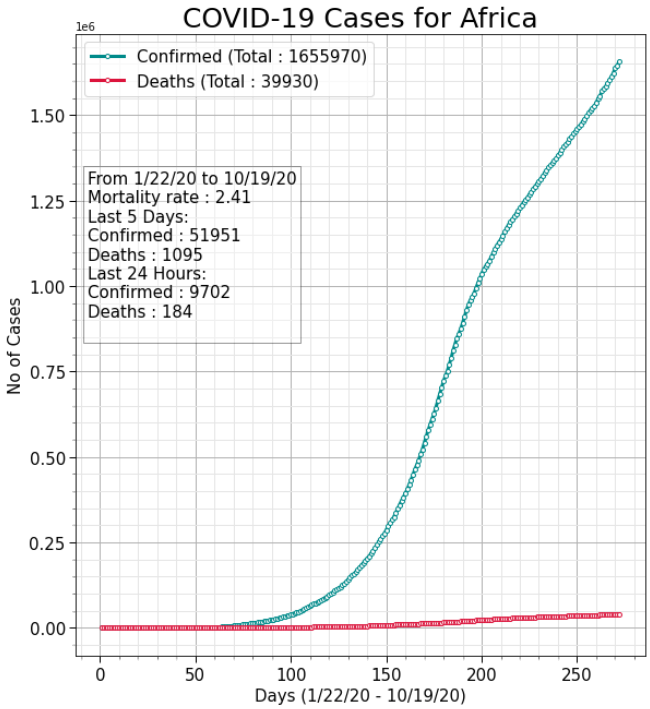


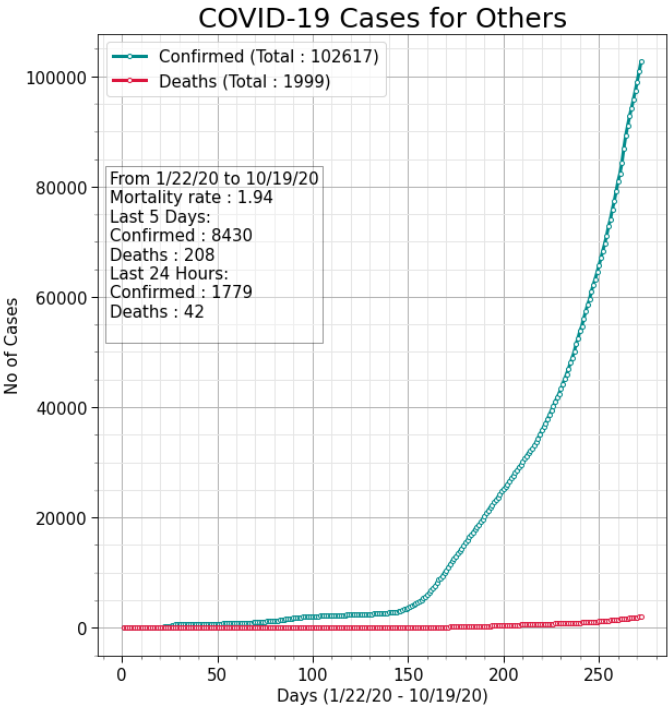

0.2
0.0
0
50
100
150
200
250
Days (1/22/20 - 10/19/20)

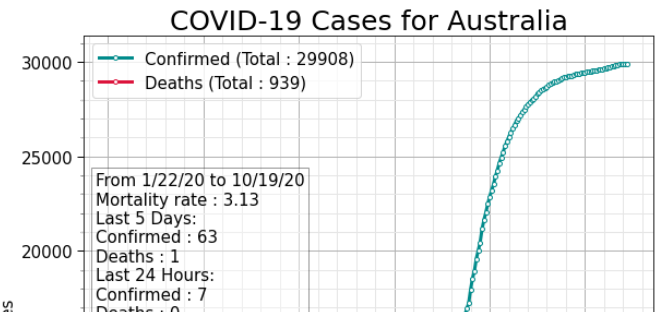

0.2
0.0
0
50
100
150
200
250
Days (1/22/20 - 10/19/20)

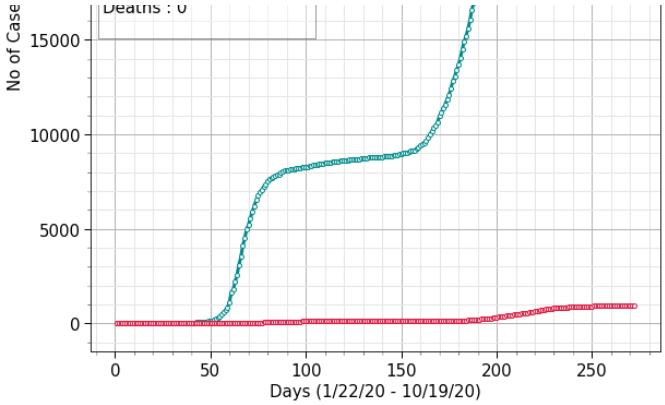

COVID-19 Cases for South America
1e6
Confirmed (Total : 9028111)
Deaths (Total : 280540)
From 1/22/20 to 10/19/20
Mortality rate : 3.1
Last 5 Days:
Confirmed : 249642
Deaths : 5981
Last 24 Hours:
Confirmed : 51373
Deaths : 1230
No of Cases
0
2
4
6
8
0
50
100
150
200
250
Days (1/22/20 - 10/19/20)


COVID-19 Cases for Europe
1e6
Confirmed (Total : 7181085)
Deaths (Total : 240334)
From 1/22/20 to 10/19/20
Mortality rate : 3.34
Last 5 Days:
Confirmed : 755481
Deaths : 5933
Last 24 Hours:
Confirmed : 227001
Deaths : 1401
No of Cases
0
1
2
3
4
5
6
7
0
50
100
150
200
250
Days (1/22/20 - 10/19/20)


COVID-19 Cases for Africa
1e6
Confirmed (Total : 1655970)
Deaths (Total : 39930)
From 1/22/20 to 10/19/20
Mortality rate : 2.41
Last 5 Days:
Confirmed : 51951
Deaths : 1095
Last 24 Hours:
Confirmed : 9702
Deaths : 184
No of Cases
0.00
0.25
0.50
0.75
1.00
1.25
1.50
0
50
100
150
200
250
Days (1/22/20 - 10/19/20)


COVID-19 Cases for Others
Confirmed (Total : 102617)
Deaths (Total : 1999)
From 1/22/20 to 10/19/20
Mortality rate : 1.94
Last 5 Days:
Confirmed : 8430
Deaths : 208
Last 24 Hours:
Confirmed : 1779
Deaths : 42
No of Cases
0
20000
40000
60000
80000
100000
0
50
100
150
200
250
Days (1/22/20 - 10/19/20)


COVID-19 Cases for Australia
Confirmed (Total : 29908)
Deaths (Total : 939)
From 1/22/20 to 10/19/20
Mortality rate : 3.13
Last 5 Days:
Confirmed : 63
Deaths : 1
Last 24 Hours:
Confirmed : 7
20000
25000
30000

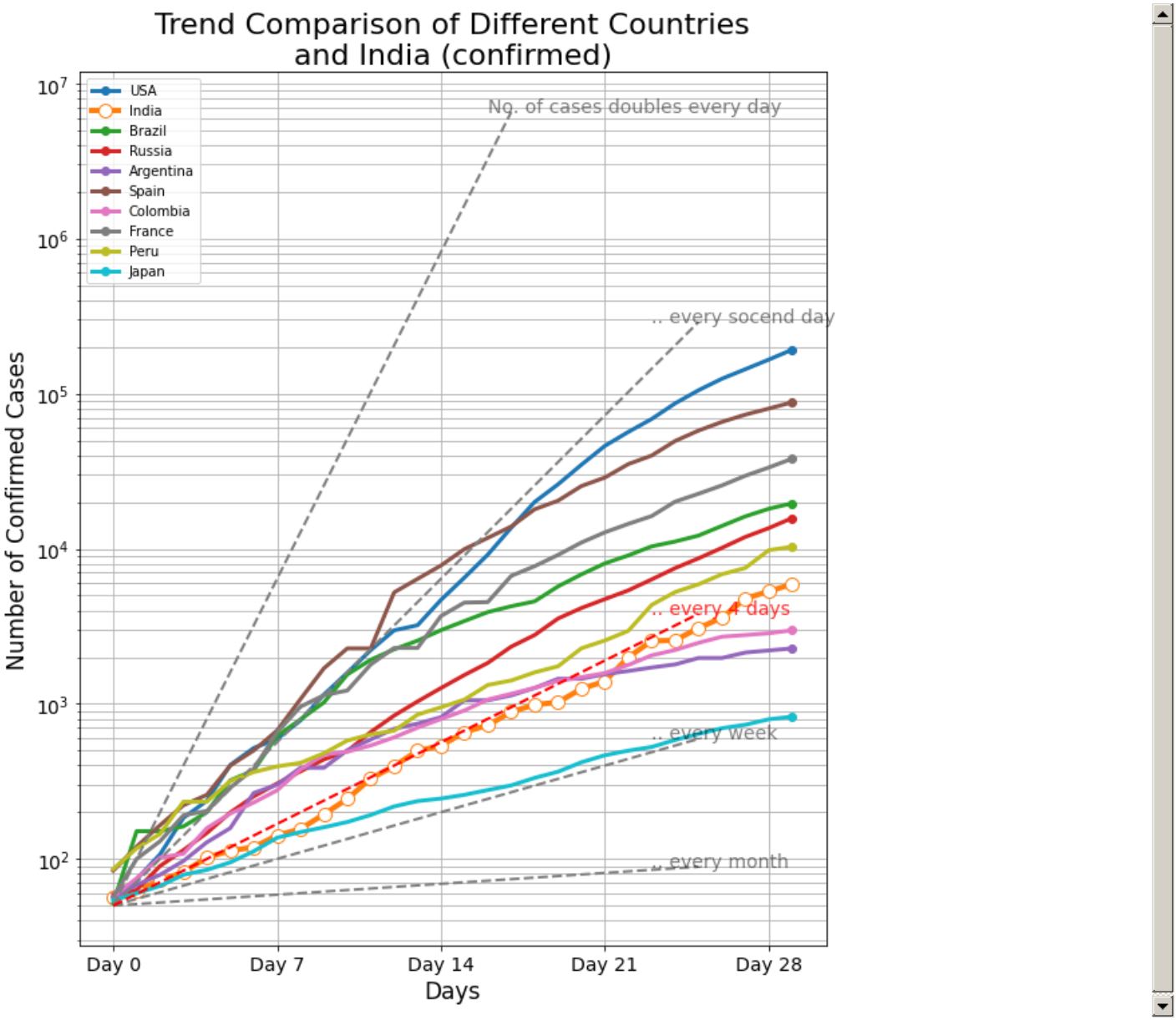


4. COVID-19 Spread Comparison of few most affected countries and INDIA

In [28]:

```
temp = df_confirmed.groupby('country').sum().drop(["Lat","Long"],axis =1).sort_values(df_confirmed.column

threshold = 50
f = plt.figure(figsize=(10,12))
ax = f.add_subplot(111)
for i,country in enumerate(temp.index):
    if i >= 9:
        if country != "India" and country != "Japan" :
            continue
    x = 30
    t = temp.loc[temp.index== country].values[0]
    t = t[t>threshold][:x]

    date = np.arange(0,len(t[:x]))
    xnew = np.linspace(date.min(), date.max(), 30)
    spl = make_interp_spline(date, t, k=1)  # type: BSpline
    power_smooth = spl(xnew)
    if country != "India":
        plt.plot(xnew,power_smooth,'-o',label = country,linewidth =3, markevery=[-1])
    else:
        marker_style = dict(linewidth=4, linestyle='-', marker='o',markersize=10, markerfacecolor='#fffff
        plt.plot(date,t,"-.",label = country,**marker_style)

plt.tick_params(labelsize = 14)
plt.xticks(np.arange(0,30,7),[ "Day "+str(i) for i in range(30)][::7])

# Reference lines
x = np.arange(0,18)
y = 2**(x+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate("No. of cases doubles every day",(x[-2],y[-1]),xycoords="data",fontsize=14,alpha = 0.5)

x = np.arange(0,26)
y = 2**(x/2+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate(".. every socend day",(x[-3],y[-1]),xycoords="data",fontsize=14,alpha = 0.5)

x = np.arange(0,26)
y = 2**(x/7+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate(".. every week",(x[-3],y[-1]),xycoords="data",fontsize=14,alpha = 0.5)

x = np.arange(0,26)
y = 2**(x/30+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate(".. every month",(x[-3],y[-1]),xycoords="data",fontsize=14,alpha = 0.5)


# India is following trend similar to doulbe the cases in 4 days but it may increase the rate
x = np.arange(0,26)
y = 2**(x/4+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "Red")
plt.annotate(".. every 4 days",(x[-3],y[-1]),color="Red",xycoords="data",fontsize=14,alpha = 0.8)

# plot Params
plt.xlabel("Days",fontsize=17)
plt.ylabel("Number of Confirmed Cases",fontsize=17)
plt.title("Trend Comparison of Different Countries\n and India (confirmed) ",fontsize=22)
```

```
plt.legend(loc = "upper left")
plt.yscale("log")
plt.grid(which="both")
#plt.savefig(out+'Trend Comparison with India (confirmed).png')
plt.show()
```

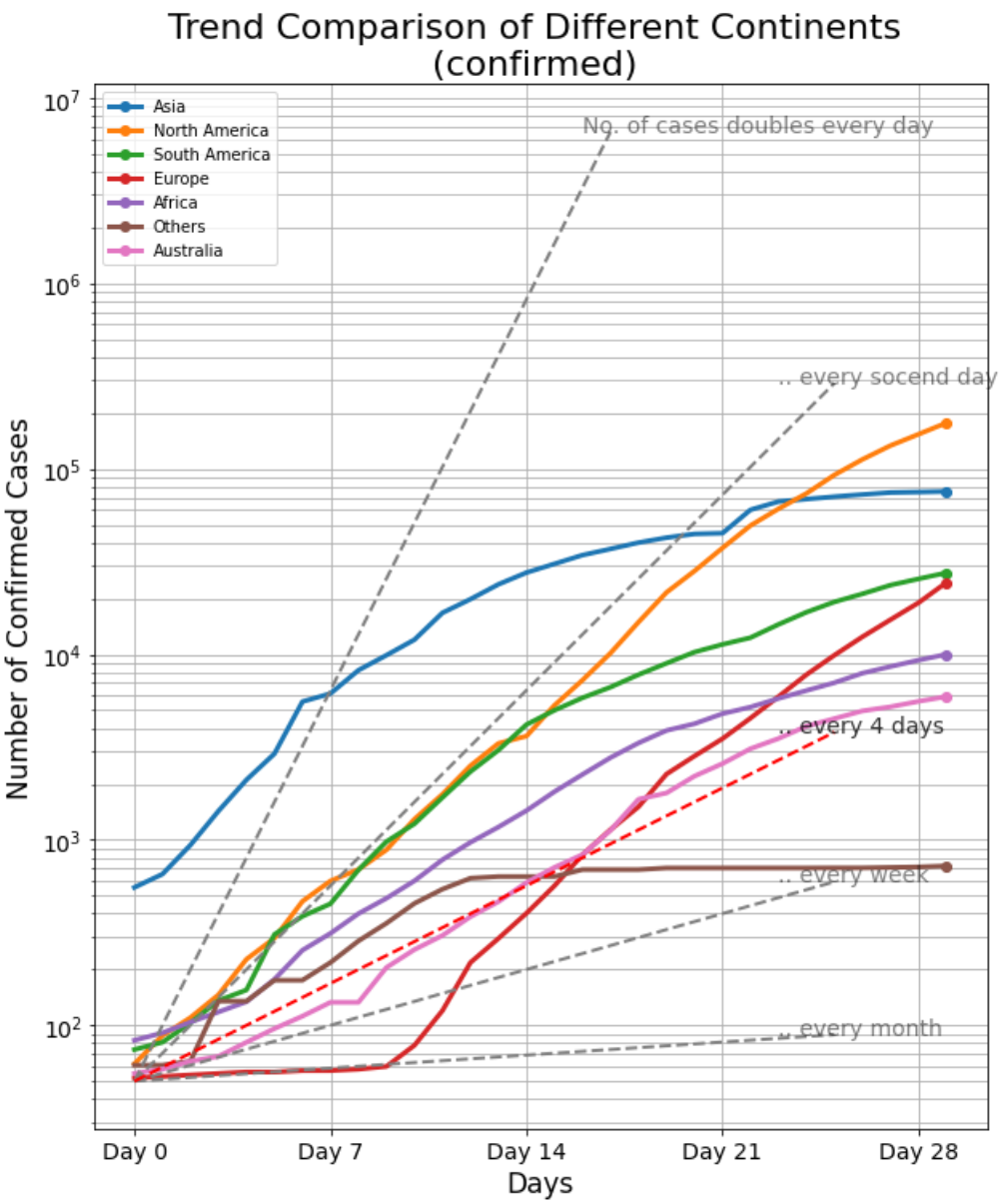


5. COVID-19 Spread Comparison of in different continents

In [29]:

```
temp = df_confirmed.groupby('continent').sum().drop(["Lat","Long"],axis =1).sort_values(df_confirmed.colu

threshold = 50
f = plt.figure(figsize=(10,12))
ax = f.add_subplot(111)
for i,country in enumerate(temp.index):
    if i > 10:
        break
    x = 30
    t = temp.loc[temp.index== country].values[0]
    t = t[t>threshold][:x]

    date = np.arange(0,len(t[:x]))
    xnew = np.linspace(date.min(), date.max(), 30)
    spl = make_interp_spline(date, t, k=1)  # type: BSpline
    power_smooth = spl(xnew)
    plt.plot(xnew,power_smooth,'-o',label = country,linewidth =3, markevery=[-1])

plt.tick_params(labelsize = 14)
plt.xticks(np.arange(0,30,7),[ "Day "+str(i) for i in range(30)][::7])

# Reference lines
```

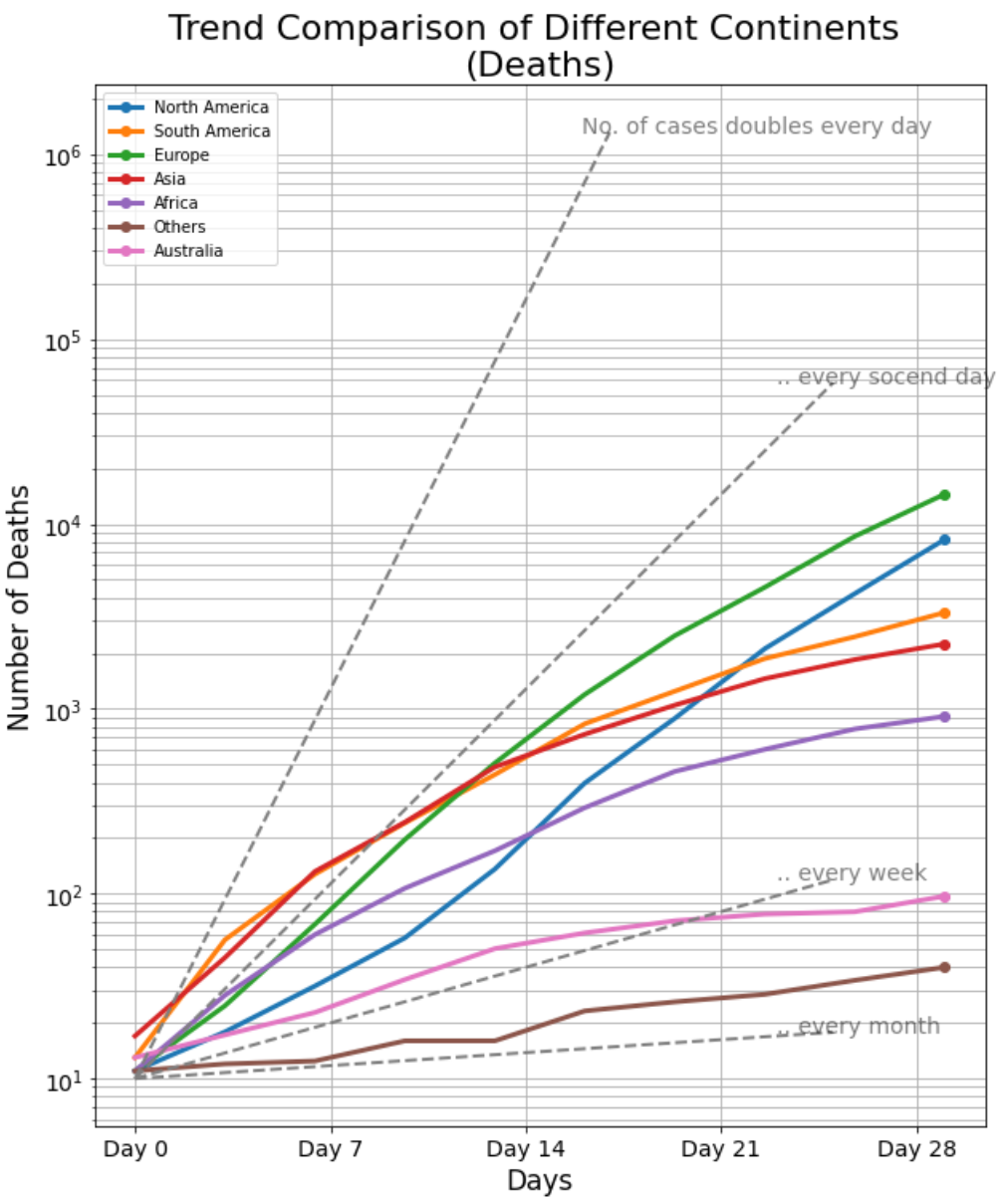
```
x = np.arange(0,18)
y = 2**(x+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate("No. of cases doubles every day",(x[-2],y[-1]),xycoords="data",fontsize=14,alpha = 0.5)

x = np.arange(0,26)
y = 2**(x/2+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate(".. every socend day",(x[-3],y[-1]),xycoords="data",fontsize=14,alpha = 0.5)

x = np.arange(0,26)
y = 2**(x/7+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate(".. every week",(x[-3],y[-1]),xycoords="data",fontsize=14,alpha = 0.5)

x = np.arange(0,26)
y = 2**(x/30+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate(".. every month",(x[-3],y[-1]),xycoords="data",fontsize=14,alpha = 0.5)


# India is following trend similar to doulbe the cases in 4 days but it may increase the rate
x = np.arange(0,26)
y = 2**(x/4+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "Red")
plt.annotate(".. every 4 days",(x[-3],y[-1]),xycoords="data",fontsize=14,alpha = 0.8)

# plot Params
plt.xlabel("Days",fontsize=17)
plt.ylabel("Number of Confirmed Cases",fontsize=17)
plt.title("Trend Comparison of Different Continents \n(confirmed) ",fontsize=22)
plt.legend(loc = "upper left")
plt.yscale("log")
plt.grid(which="both")
#plt.savefig(out+'Trend Comparison of continents (Confirmed).png')
plt.show()
```



In [30]:

```
temp = df_deaths.groupby('continent').sum().drop(["Lat","Long"],axis =1).sort_values(df_deaths.columns[-1

threshold = 10
f = plt.figure(figsize=(10,12))
ax = f.add_subplot(111)
for i,country in enumerate(temp.index):
    if i > 10:
        break
    x = 30
    t = temp.loc[temp.index== country].values[0]
    t = t[t>threshold][:x]

    date = np.arange(0,len(t[:x]))
    xnew = np.linspace(date.min(), date.max(), 10)
    spl = make_interp_spline(date, t, k=1)  # type: BSpline
    power_smooth = spl(xnew)
    plt.plot(xnew,power_smooth,'-o',label = country,linewidth =3, markevery=[-1])


plt.tick_params(labelsize = 14)
plt.xticks(np.arange(0,30,7),[ "Day "+str(i) for i in range(30)][::7])

# Reference lines
x = np.arange(0,18)
y = 2**(x+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate("No. of cases doubles every day",(x[-2],y[-1]),xycoords="data",fontsize=14,alpha = 0.5)

x = np.arange(0,26)
y = 2**(x/2+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
```

```
plt.annotate(".. every socend day",(x[-3],y[-1]),xycoords="data",fontsize=14,alpha = 0.5)

x = np.arange(0,26)
y = 2**(x/7+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate(".. every week",(x[-3],y[-1]),xycoords="data",fontsize=14,alpha = 0.5)

x = np.arange(0,26)
y = 2**(x/30+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate(".. every month",(x[-3],y[-1]),xycoords="data",fontsize=14,alpha = 0.5)

# plot Params
plt.xlabel("Days",fontsize=17)
plt.ylabel("Number of Deaths",fontsize=17)
plt.title("Trend Comparison of Different Continents \n(Deaths)",fontsize=22)
plt.legend(loc = "upper left")
plt.yscale("log")
plt.grid(which="both")
#plt.savefig(out+'Trend Comparison continents (deaths).png')
plt.show()
```



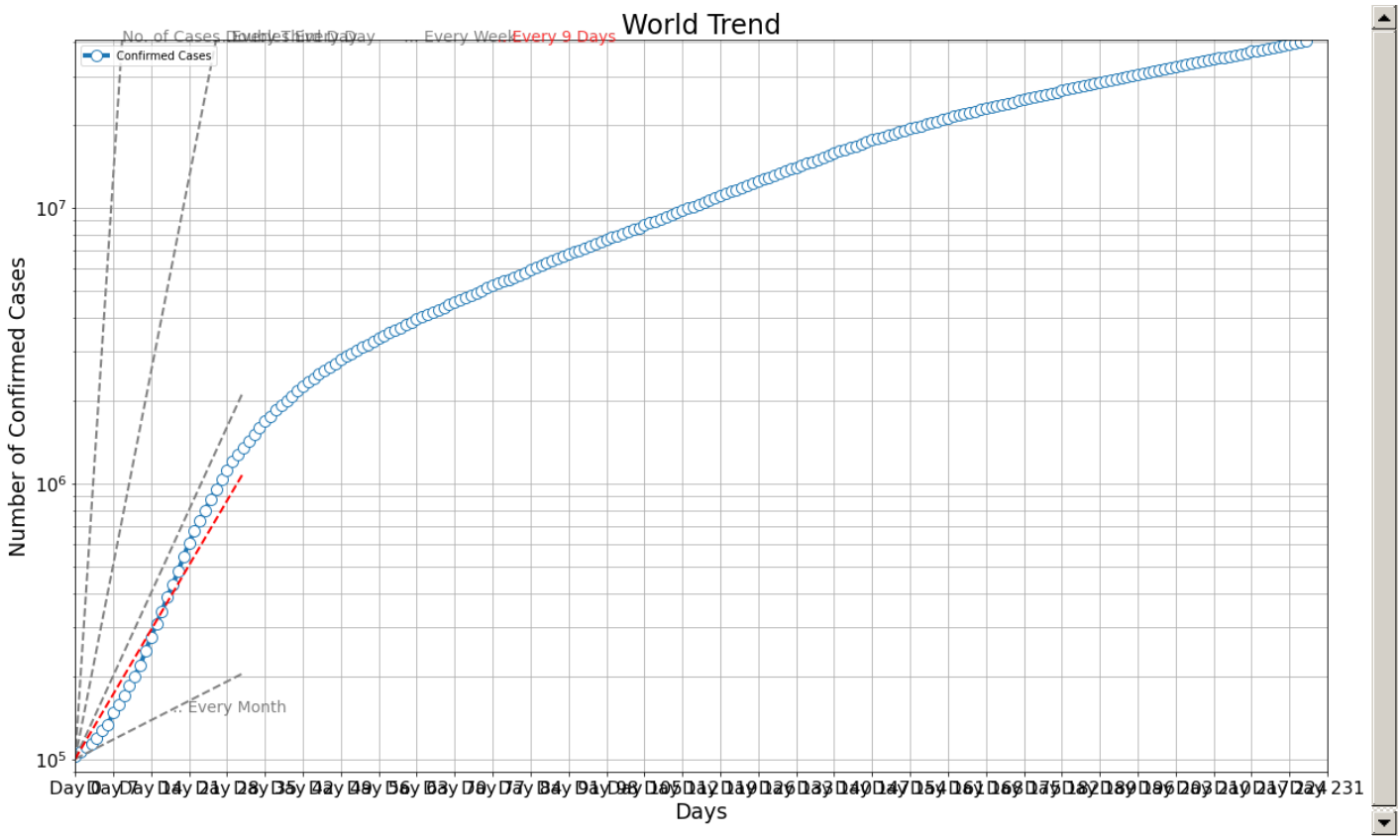
Global Prediction

Global Trend:

It is useful to understand the global trend of an increase in the number of cases over time. There is always a pattern in any data, but the concern is how strongly data follows a pattern. COVID-19 spreads exponentially, positive cases of COVID-19 takes 67 days to reach 1 Lakhs while it takes only 11 days to reach 2 Lakhs, 4 days to reach 3 Lakhs, and just 2 days to reach 5 Lakhs. This trend shows how fast it spreads.

In [31]:

```
temp_data = df_confirmed.iloc[:,5:].sum(axis =0)
f = plt.figure(figsize=(20,12))
f.add_subplot(111)

threshold = 100000

t = temp_data.values
t = t[t >threshold]

date = np.arange(0,len(t[:]))
xnew = np.linspace(date.min(), date.max(), 10)
spl = make_interp_spline(date, t, k=1)  # type: BSpline
power_smooth = spl(xnew)

marker_style = dict(linewidth=4, linestyle='-', marker='o',markersize=10, markerfacecolor='#ffffff')
plt.plot(date,t,"-.",label="Confirmed Cases",**marker_style)


# Reference lines
x = np.arange(0,32)
y = 2**(x+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate("No. of Cases Doubles Every Day",(np.log2((t.max()-threshold)/threshold),t.max()-threshold/2

x = np.arange(0,32)
y = 2**(x/3+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate("...Every Third Day",(np.log2((t.max()-threshold)/threshold)*3,t.max()-threshold),xycoords="

x = np.arange(0,32)
y = 2**(x/7+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate("... Every Week",(np.log2((t.max()-threshold)/threshold)*7,t.max()-threshold),xycoords="data

x = np.arange(0,32)
y = 2**(x/30+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate(".. Every Month",(18,2**(17/30+np.log2(threshold))),xycoords="data",fontsize=14,alpha = 0.5)

x = np.arange(0,32)
y = 2**(x/9+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "Red")
plt.annotate(".. Every 9 Days",(np.log2((t.max()-threshold)/threshold)*9,t.max()-threshold),color="Red",x


plt.xlim(date[0],date[-1])
plt.ylim(threshold - threshold/10,t.max()+threshold)
# plot Params
# plt.tight_layout()
plt.tick_params(labelsize = 16)
plt.xticks(np.arange(0,len(t[:])+7,7),[ "Day "+str(i) for i in range(len(t[:])+7)][::7])
plt.xlabel("Days",fontsize=19)
plt.ylabel("Number of Confirmed Cases",fontsize=19)
plt.title("World Trend",fontsize=24)
plt.legend(loc = "upper left")
plt.yscale("log")
plt.grid(which="both")
#plt.savefig(out+"World Trend Confirmed cases.png")
plt.show()
```



COVID-19 Mortality Rate Variation Over Period of Time

For any epidemic the one of the most important evaluation is Mortality Rate. It is the measure of number of deaths in a particular population during a specific interval.

1st curve shows how the mortality rate varies from 22 JAN 2020 to till date all over the world.

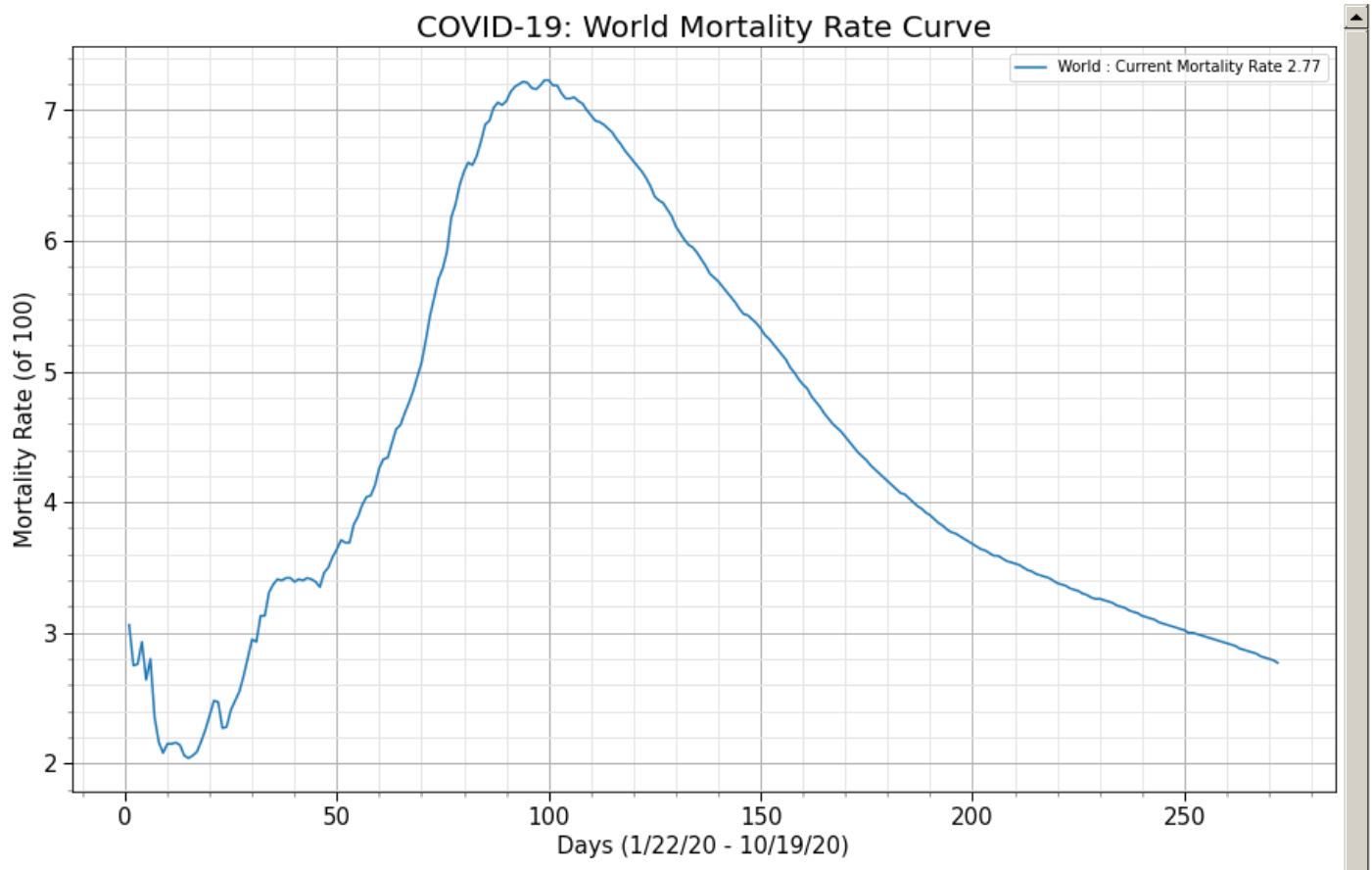
2nd Curve shows the variation of mortality rate in different continents over time.

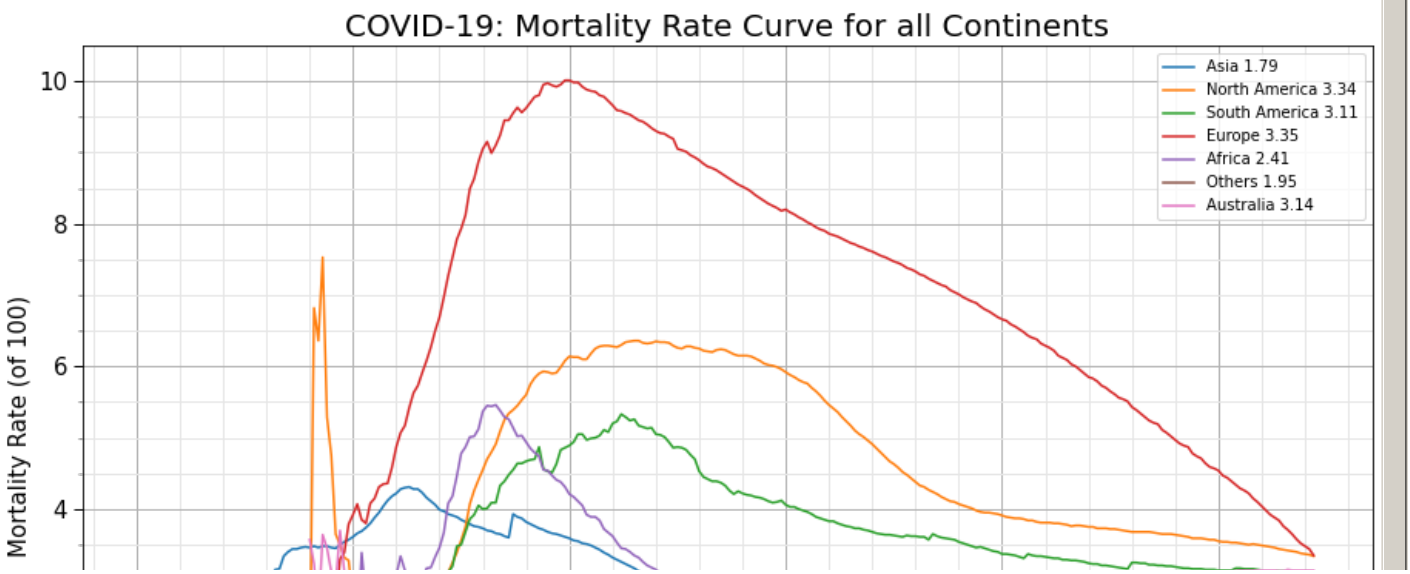
In [32]:

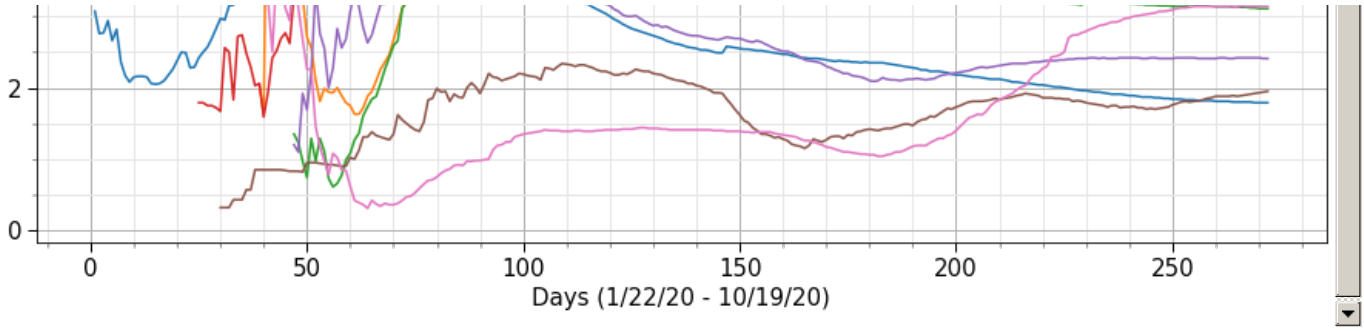
```
df_continents= df_confirmed.groupby(["continent"]).sum()
continents = df_continents.sort_values(df_continents.columns[-1],ascending = False).index
continents = ["All"]+list(continents)

cols =1
rows = 2
axis_label = ["Days ("+df_confirmed.columns[5]+" - "+df_confirmed.columns[-1]+")","Mortality Rate (of 100

f = plt.figure(figsize=(15,10*rows))

#SubPlot 1
ax = f.add_subplot(211)
mortality_rate = get_mortality_rate(df_confirmed,df_deaths,continent=continents[0])
plt.plot(np.arange(1,mortality_rate.shape[0]+1),mortality_rate,label = "World : Current Mortality Rate "+

plt_title = "COVID-19: World Mortality Rate Curve"
plot_params(ax,axis_label,plt_title)
# Legend Location
l = plt.legend(loc= "best")


#SubPlot 2
ax = f.add_subplot(212)
for i, continent in enumerate(continents[1:]):
    mortality_rate = get_mortality_rate(df_confirmed,df_deaths,continent=continent)
    plt.plot(np.arange(1+mortality_rate[mortality_rate == 0].shape[0],mortality_rate[mortality_rate == 0]

plt_title = "COVID-19: Mortality Rate Curve for all Continents"
plot_params(ax,axis_label,plt_title)

# Legend Location
l = plt.legend(loc= "best")

plt.minorticks_on()
```

```
#plt.savefig(out+'Mortality rate.png')
plt.show()
```

```
<ipython-input-13-23285493a0fd>:112: RuntimeWarning:

invalid value encountered in true_divide

<ipython-input-13-23285493a0fd>:112: RuntimeWarning:

invalid value encountered in true_divide

<ipython-input-13-23285493a0fd>:112: RuntimeWarning:

invalid value encountered in true_divide

<ipython-input-13-23285493a0fd>:112: RuntimeWarning:

invalid value encountered in true_divide

<ipython-input-13-23285493a0fd>:112: RuntimeWarning:

invalid value encountered in true_divide
```







COVID-19: Change in Mortality Rate of Each Countries Over Time

In [33]:

```
df_data = df_table.groupby(['Last_Update', 'Country_Region'])['Confirmed', 'Deaths','continent'].max().re
df_data["Last_Update"] = pd.to_datetime( df_data["Last_Update"]).dt.strftime('%m/%d/%Y')

fig = px.scatter(df_data, y=100*df_data["Deaths"]/(df_data["Confirmed"]+1),
                     x= df_data["Confirmed"]+1,
                     range_y = [-1,18],
                     range_x = [1,df_data["Confirmed"].max()+10000],
                     color= "continent", hover_name="Country_Region",
                     hover_data=["Confirmed","Deaths"],
                     range_color= [0, max(np.power(df_data["Confirmed"],0.3))],
                     animation_frame="Last_Update",
                     animation_group="Country_Region",
                     color_continuous_scale=px.colors.sequential.Plasma,
                     title='COVID-19: Change in Mortality Rate of Each Countries Over Time',
                     size = np.power(df_data["Confirmed"]+1,0.3)-0.5,
                     size_max = 30,
                     log_x=True,
                     height =700,
                     )
fig.update_coloraxes(colorscale="hot")
fig.update(layout_coloraxis_showscale=False)
fig.update_xaxes(title_text="Confirmed Cases (Log Scale)")
fig.update_yaxes(title_text="Mortality Rate (%)")
fig.show()
```

```
<ipython-input-33-459809667884>:1: FutureWarning:

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

Cumulative Confirmed Cases and Cumulative Recovery Vs Cumulative Deaths Analysis

The variation of Cumulative Confirmed Cases and Cumulative Recovery with Cumulative Deaths can show a trend. These 2 curves depict the same. Also, these curves should be a straight line as shown in the 1st curve, but the 2nd curve is not showing that trend, and as the number of recovered cases is increasing, death is increasing at a faster rate.

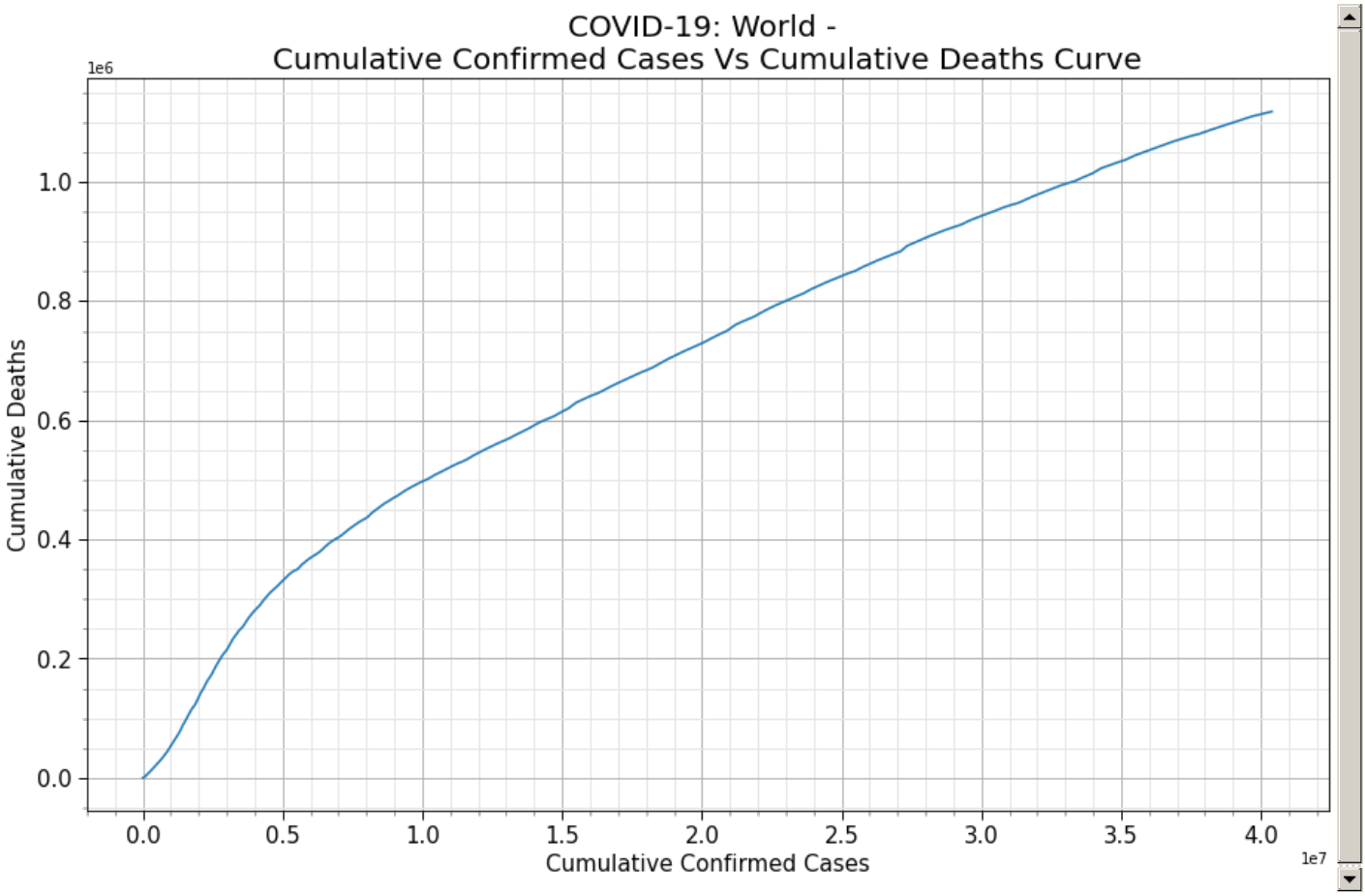
1st curve: Cumulative Confirmed Cases VS Cumulative Deaths

2nd curve: Cululative Recovery VS Cumulative Deaths

In [34]:

```
cols =1
rows = 2

f = plt.figure(figsize=(15,10*rows))

# SubPlot 1
ax = f.add_subplot(211)
plt.plot(np.sum(np.asarray(df_confirmed.iloc[:,5:]),axis = 0),np.sum(np.asarray(df_deaths.iloc[:,5:]),axi

axis_label = ["Cumulative Confirmed Cases","Cumulative Deaths"]
plt_title = "COVID-19: World - \nCumulative Confirmed Cases Vs Cumulative Deaths Curve"
plot_params(ax,axis_label,plt_title)


# # SubPlot 2
# ax = f.add_subplot(212)
# mortality_rate = get_mortality_rate(df_confirmed,df_deaths,continent=continents[0])
# plt.plot(np.sum(np.asarray(df_recovered.iloc[:,5:]),axis = 0),np.sum(np.asarray(df_deaths.iloc[:,5:]),
```

```
# axis_label = ["Cumulative Recoveries","Cumulative Deaths"]
# plt_title = "COVID-19: World - Cumulative Recovery Vs Cumulative Deaths Curve"

# plot_params(ax,axis_label,plt_title)
plt.minorticks_on()
#plt.savefig(out+'Cumulative Confirmed Cases Vs Cumulative Deaths Curve.png')
plt.show()
```



Variation of Deaths vs Confirmed cases of different cuntries over time

In [35]:

```
df_data = df_table.groupby(['Last_Update', 'Country_Region'])['Confirmed', 'Deaths','continent'].max().re
df_data["Last_Update"] = pd.to_datetime( df_data["Last_Update"]).dt.strftime('%m/%d/%Y')

fig = px.scatter(df_data, y=df_data["Deaths"],
                 x= df_data["Confirmed"]+1,
                 range_y = [1,df_data["Deaths"].max()+1000],
                 range_x = [1,df_data["Confirmed"].max()+10000],
                 color= "continent", hover_name="Country_Region",
                 hover_data=["Confirmed","Deaths"],
                 range_color= [0, max(np.power(df_data["Confirmed"],0.3))],
                 animation_frame="Last_Update",
                 animation_group="Country_Region",
                 color_continuous_scale=px.colors.sequential.Plasma,
                 title='COVID-19: Change Deaths vs Cofirmed of Each Countries Over Time',
                 size = np.power(df_data["Confirmed"]+1,0.3)-0.5,
                 size_max = 30,
                 log_x=True,
                 log_y=True,
                 height =700,
                 )
fig.update_coloraxes(colorscale="hot")
fig.update(layout_coloraxis_showscale=False)
fig.update_xaxes(title_text="Confirmed Cases (Log Scale)")
fig.update_yaxes(title_text="Deaths Rate (Log Scale)")
fig.show()
```

```
<ipython-input-35-09592e284058>:1: FutureWarning:

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

COVID-19 : INDIA

Few basic visualization related to India. I will be updating with more visualization in further commits. i am also working on Paitent Data Insights. Adding Soon

Dataset: This dataset is provided by https://api.rootnet.in/

Analysis of Tests done in India

In [36]:

```
#For runing this command connect to your internet
df_india_test = pd.io.json.json_normalize(requests.get('https://api.rootnet.in/covid19-in/stats/testing/h
```

```
<ipython-input-36-ab5ff6ac8823>:2: FutureWarning:

pandas.io.json.json_normalize is deprecated, use pandas.json_normalize instead
```

In [37]:

```
df_india_test["p2t_ratio"]= np.round(100*df_india_test["c_positive"]/df_india_test["c_tests"],2)
df_india_test["positive"] = df_india_test["c_positive"].diff()
df_india_test["tests"] = df_india_test["c_tests"].diff()
df_india_test["p2t_ratio"]= np.round(100*df_india_test["positive"]/df_india_test["tests"],2)
```

Total tests done till date (30 March 2020) in India

In [38]:

```
df_india_test["c_tests"][-1:].values[0]
```

Out[38]:

```
96116771.0
```

Test Conducted per Million People

In [39]:

```
np.round(1000000*df_india_test["c_tests"][-1:].values[0]/1300000000,2)
```

Out[39]:

```
73935.98
```

COVID19 Cases in India

In [40]:

```
india_data_json = requests.get('https://api.rootnet.in/covid19-in/unofficial/covid19india.org/statewise')
df_india = pd.io.json.json_normalize(india_data_json['data']['statewise'])
df_india = df_india.set_index("state")
```

```
<ipython-input-40-77e15bea6e01>:2: FutureWarning:

pandas.io.json.json_normalize is deprecated, use pandas.json_normalize instead
```

In [41]:

```
total = df_india.sum()
total.name = "Total"
pd.DataFrame(total).transpose().style.background_gradient(cmap='Wistia',axis=1)
```

Out[41]:

	confirmed	recovered	deaths	active
Total	7596667	6730617	115252	749537

In [42]:

```
df_india.style.background_gradient(cmap='Wistia')
```

Out[42]:

state	confirmed	recovered	deaths	active
Maharashtra	1601365	1384879	42240	173759
Andhra Pradesh	786050	744532	6453	35065
Karnataka	770604	653829	10542	106214
Tamil Nadu	690936	642152	10694	38090
Uttar Pradesh	456865	418685	6685	31495
Delhi	333171	304561	6040	22570
Kerala	346882	252868	1183	92732
West Bengal	325028	284325	6119	34584
Odisha	272250	249575	1221	21454
Telangana	223059	200686	1275	21098
Bihar	205124	193789	1003	10331
Assam	201407	173210	875	27319
Rajasthan	175226	152573	1760	20893
Gujarat	160722	142899	3646	14177
Madhya Pradesh	161203	145421	2786	12996
Chhattisgarh	162772	135259	1534	25979
Haryana	151234	139511	1648	10075
Punjab	128103	118767	4029	5307
Jharkhand	96842	89780	842	6220
Jammu and Kashmir	88369	78667	1388	8314
Uttarakhand	58360	51486	933	5527
Goa	40746	36914	549	3283
Puducherry	33247	28520	575	4152
Tripura	29550	26527	328	2672
Himachal Pradesh	19119	16238	267	2577
Manipur	15778	11913	117	3748
Chandigarh	13686	12617	208	861
Arunachal Pradesh	13643	10780	30	2833
Meghalaya	8536	6392	75	2069
Nagaland	7953	6208	21	1652
Ladakh	5647	4701	66	880
Andaman and Nicobar Islands	4126	3892	56	178
Sikkim	3601	3195	62	263
Dadra and Nagar Haveli and Daman and Diu	3183	3115	2	41
Mizoram	2280	2151	0	129
State Unassigned	0	0	0	0
Lakshadweep	0	0	0	0

States with Reported Deaths

In [43]:

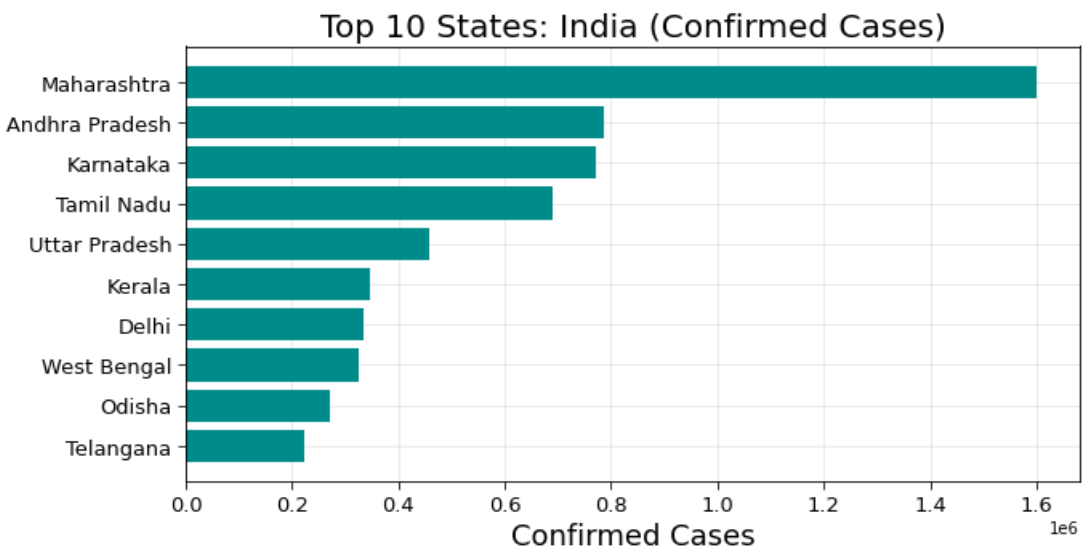
```
df_india[df_india['deaths'] > 0].style.background_gradient(cmap='Wistia')
```

Out[43]:

state	confirmed	recovered	deaths	active
Maharashtra	1601365	1384879	42240	173759
Andhra Pradesh	786050	744532	6453	35065
Karnataka	770604	653829	10542	106214
Tamil Nadu	690936	642152	10694	38090
Uttar Pradesh	456865	418685	6685	31495
Delhi	333171	304561	6040	22570
Kerala	346882	252868	1183	92732
West Bengal	325028	284325	6119	34584
Odisha	272250	249575	1221	21454
Telangana	223059	200686	1275	21098
Bihar	205124	193789	1003	10331
Assam	201407	173210	875	27319
Rajasthan	175226	152573	1760	20893
Gujarat	160722	142899	3646	14177
Madhya Pradesh	161203	145421	2786	12996
Chhattisgarh	162772	135259	1534	25979
Haryana	151234	139511	1648	10075
Punjab	128103	118767	4029	5307
Jharkhand	96842	89780	842	6220
Jammu and Kashmir	88369	78667	1388	8314
Uttarakhand	58360	51486	933	5527
Goa	40746	36914	549	3283
Puducherry	33247	28520	575	4152
Tripura	29550	26527	328	2672
Himachal Pradesh	19119	16238	267	2577
Manipur	15778	11913	117	3748
Chandigarh	13686	12617	208	861
Arunachal Pradesh	13643	10780	30	2833
Meghalaya	8536	6392	75	2069
Nagaland	7953	6208	21	1652
Ladakh	5647	4701	66	880
Andaman and Nicobar Islands	4126	3892	56	178
Sikkim	3601	3195	62	263
Dadra and Nagar Haveli and Daman and Diu	3183	3115	2	41

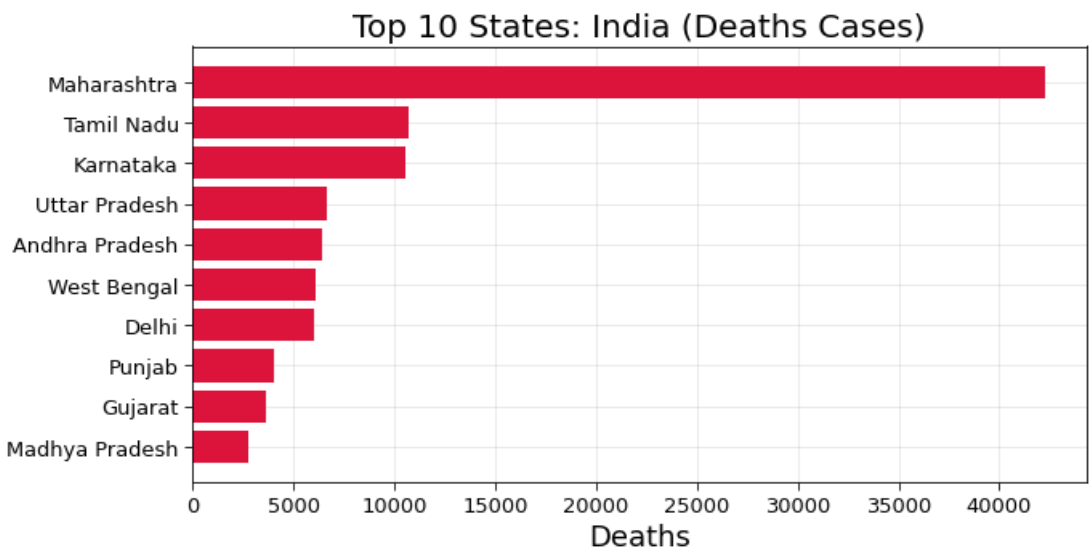
In [44]:

```
f = plt.figure(figsize=(10,5))
f.add_subplot(111)

plt.axes(axisbelow=True)
plt.barh(df_india.sort_values('confirmed')["confirmed"].index[-10:],df_india.sort_values('confirmed')["co
plt.tick_params(size=5,labelsize = 13)
plt.xlabel("Confirmed Cases",fontsize=18)
plt.title("Top 10 States: India (Confirmed Cases)",fontsize=20)
plt.grid(alpha=0.3)
#plt.savefig(out+'Top 10 States_India (Confirmed Cases).png')
```



In [45]:

```
f = plt.figure(figsize=(10,5))
f.add_subplot(111)

plt.axes(axisbelow=True)
plt.barh(df_india.sort_values('deaths')["deaths"].index[-10:],df_india.sort_values('deaths')["deaths"].va
plt.tick_params(size=5,labelsize = 13)
plt.xlabel("Deaths",fontsize=18)
plt.title("Top 10 States: India (Deaths Cases)",fontsize=20)
plt.grid(alpha=0.3)
#plt.savefig(out+'Top 10 States_India (Deaths Cases).png')
```



Correlation

In [46]:

```
df_india.corr().style.background_gradient(cmap='Reds')
```

Out[46]:

	confirmed	recovered	deaths	active
confirmed	1.000000	0.998444	0.920923	0.904265
recovered	0.998444	1.000000	0.912604	0.879439
deaths	0.920923	0.912604	1.000000	0.855973
active	0.904265	0.879439	0.855973	1.000000

Conclusion

There are hundreds of coronaviruses, most of which circulate in animals. Only seven of these viruses infect humans and four of them cause symptoms of the common cold. But, three times in the last 20 years, a coronavirus has jumped from animals to humans to cause severe disease.

SARS, a beta coronavirus emerged in 2002 and was controlled mainly by aggressive public health measures. There have been no new cases since 2004. MERS emerged in 2012, still exists in camels, and can infect people who have close contact with them.

COVID-19, a new and sometimes deadly respiratory illness that is believed to have originated in a live animal market in China, has spread rapidly throughout that country and the world.

The new coronavirus was first detected in Wuhan, China in December 2019. Tens of thousands of people were infected in China, with the virus spreading easily from person-to-person in many parts of that country.

The novel coronavirus infections were at first associated with travel from Wuhan, but the virus has now established itself in 177 countries and territories around the world in a rapidly expanding pandemic. Health officials in the United States and around the world are working to contain the spread of the virus through public health measures such as social distancing, contact tracing, testing, quarantines and travel restrictions. Scientists are working to find medications to treat the disease and to develop a vaccine.

The World Health Organization declared the novel coronavirus outbreak “a public health emergency of international concern” on January 30. On March 11, 2020 after sustained spread of the disease outside of China, the World Health Organization declared the COVID-19 epidemic a pandemic. Public health measures like ones implemented in China and now around the world, will hopefully blunt the spread of the virus while treatments and a vaccine are developed to stop it.

In [ ]:

In [ ]: