[1] Business understanding

What is novel coronavirus?

2019 Novel Coronavirus (2019-nCoV) is a virus (more specifically, a coronavirus) identified as the cause of an outbreak of respiratory illn first detected in Wuhan, China. Early on, many of the patients in the outbreak in Wuhan, China reportedly had some link to a large seafoo animal market, suggesting animal-to-person spread. However, a growing number of patients reportedly have not had exposure to anima markets, indicating person-to-person spread is occurring. At this time, it's unclear how easily or sustainably this virus is spreading betwe people - CDC

Problem Statement:

Questions that will be answered through the analysis are:

1. Which date has recorded the highest single-day coronavirus death so far?
2. What is the Biggest one-day recovery in Covid-19 cases worldwide?
3. What is the current total number of active cases worldwide?
4. Which Country has hight Covid-19 positive cases?
5. How many countries have recorded zero death case?

Real world/Business Objectives and Constraints:

- The cost of a mis-analysis can be very high.
- No strict latency concerns.
- Interpretability is important.

[2] Data understanding

- In order to be able to answer these questions, a more convenient data set is necessary.
- Data Sources: https://github.com/datasets/covid-19
- This dataset includes time series data tracking the number of people affected by COVID-19 worldwide, including:
  - confirmed tested cases of Coronavirus infection
  - the number of people who have reportedly died while sick with Coronavirus
  - the number of people who have reportedly recovered from it
- The data is available from 22 Jan, 2020.

[2.1] Importing libraries

```
import pandas as pd
import numpy as np

#plotting libs
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go

#Read Data
df = pd.read_csv("countries-aggregated.csv")
```

```
# What kind of columns are there in the dataset?
df.columns
```

```
Index(['Date', 'Country', 'Confirmed', 'Recovered', 'Deaths'], dtype='object')
```

```
# Print the top 2 rows
df.head(2)
```

	Date	Country	Confirmed	Recovered	Deaths
0	2020-01-22	Afghanistan	0	0	0
1	2020-01-22	Albania	0	0	0

```
# Print the last 2 rows
df.tail(2)
```

	Date	Country	Confirmed	Recovered	Deaths
24250	2020-05-29	Zambia	1057	779	7
24251	2020-05-29	Zimbabwe	149	28	4

```
#Print the number of rows and columns in this dataset.
print(f"There are total {df.shape[0]} rows and {df.shape[1]} columns in the dataset.")
```

```
There are total 24252 rows and 5 columns in the dataset.
```

```
#check the data type of Dataset Columns.
print(df.dtypes)
```

```
Date         object
Country      object
Confirmed     int64
Recovered     int64
Deaths        int64
dtype: object
```

[3] Data preparation

[3.1] Clean data

- Which columns had no missing values?

```
#Provide a set of column names that have no missing values.
no_nulls = set(df.columns[df.isnull().mean()==0]) #Provide a set of columns with 0 missing values.
no_nulls
```

```
{'Confirmed', 'Country', 'Date', 'Deaths', 'Recovered'}
```

Observation: Hence there are no NAN value in the dataset.

- Find the unique Counties.

```
#List unique values in the df['Country'] column
print(f"We have total {len(df.Country.unique())} Countries data.")
```

```
We have total 188 Countries data.
```

[3.2] Prepar a new dataframe and group it by Dates.

```
#date_index_df
#Grouping different types of cases as per the date.
date_index_df = df.groupby(["Date"]).agg({"Confirmed":'sum',"Recovered":'sum',"Deaths":'sum'})
date_index_df.head()
```

Date	Confirmed	Recovered	Deaths
2020-01-22	555	28	17
2020-01-23	654	30	18
2020-01-24	941	36	26
2020-01-25	1434	39	42

[3.3] Prepar a new dataframe and group it by Countries.

```
# country_index_df
#Calculating countrywise positive cases
#Sort the values by confirmed cases in ascending order
country_index_df =df[df["Date"]==df["Date"].max()].groupby(["Country"]).agg({"Confirmed":'sum',"Recovered":'sum',"Deaths":'su
country_index_df.head()
```

Country	Confirmed	Recovered	Deaths
US	1746019	406446	102809
Brazil	465166	189476	27878
Russia	387623	159257	4374
United Kingdom	272607	1172	38243
Spain	238564	150376	27121

```
#creating a new column for Active Cases
# Active Cases = Number of Confirmed Cases - (Number of Recovered Cases - Number of Death Cases)
country_index_df["Active"] = country_index_df["Confirmed"]-(country_index_df["Recovered"]-country_index_df["Deaths"])
country_index_df.head()
```

Country	Confirmed	Recovered	Deaths	Active
US	1746019	406446	102809	1442382
Brazil	465166	189476	27878	303568
Russia	387623	159257	4374	232740
United Kingdom	272607	1172	38243	309678
Spain	238564	150376	27121	115309

[4] Analysis

[4.1] Date wise Analysis

```
def datewise_cases(x, y, title, x_label, y_label):
    """
    ================================
    A function that will plot the the number of cases as per the date.
    ================================
    Parameters:
    x = x-axis, the Date Column to plot the Scatter plot
    y = y-axis, the Case Columns to plot the Scatter plot
    title = Title of the plot
    x_label= xaxis_title
    y_label= yaxis_title
    """

    fig=go.Figure()
    fig.add_trace(go.Scatter(x=x, y = y, mode='lines+markers'))
    fig.update_layout(title = title, xaxis_title= x_label, yaxis_title = y_label)
    fig.show()
```

Observation:

April 17 2020, reported **8858** coronavirus deaths, highest in one day so far.

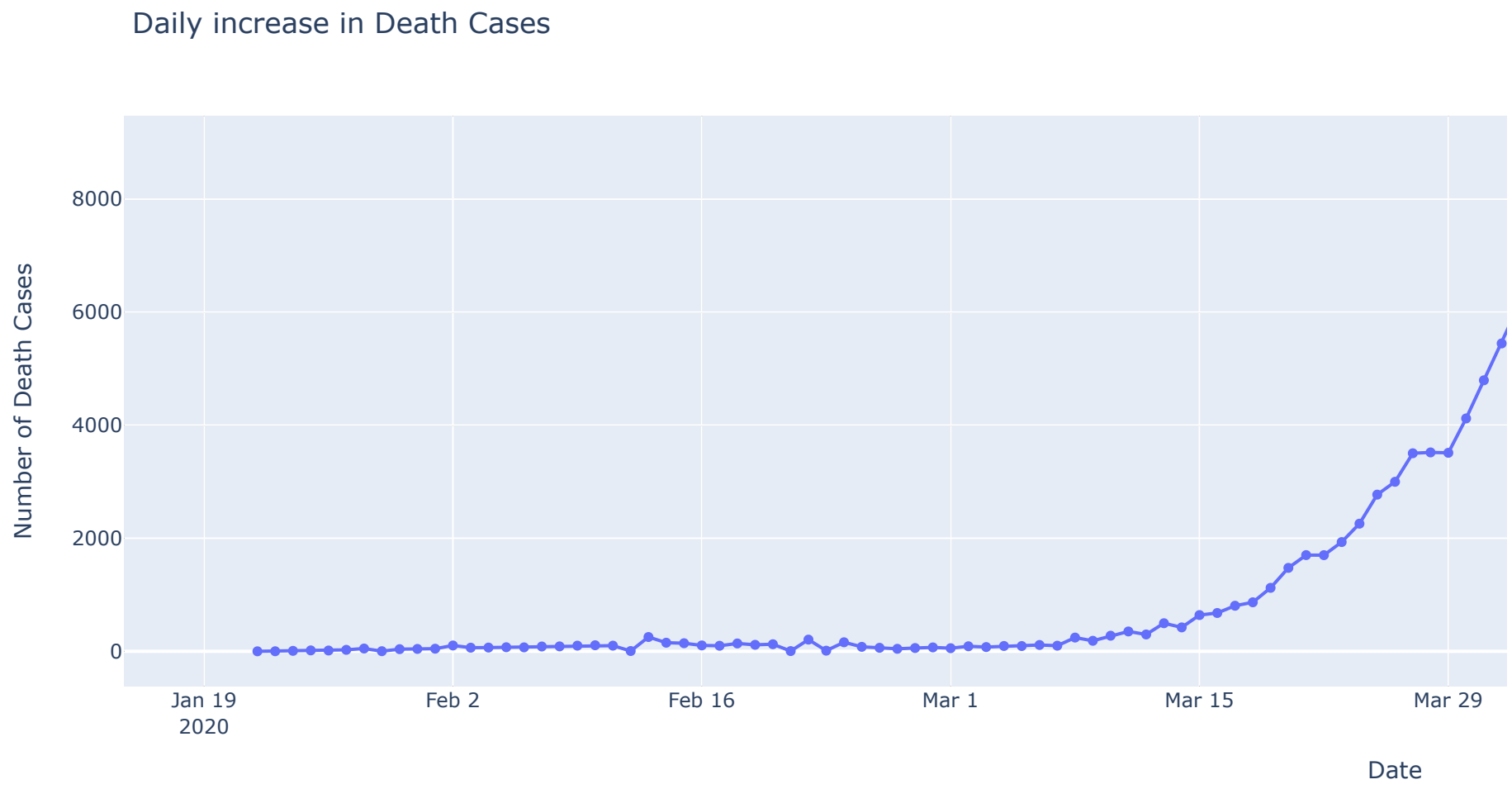
Q2: What is the Biggiest one-day recovery in Covid-19 cases worldwide?

```
datewise_cases(x = date_index_df.index, y = date_index_df["Recovered"].diff().fillna(0), "Daily increase in Recovered Cases"
```

Q1: Which date has the highest single-day coronavirus death?

reference: https://plotly.com/python/line-and-scatter/#line-and-scatter-plots

```
datewise_cases(date_index_df.index, date_index_df["Deaths"].diff().fillna(0), "Daily increase in Death Cases", "Date", "Numb
```



Q4: What are the top 10 countries that have the highest active cases so far?

```
Countrywise_cases(country_index_df, 10)
```

Observation:

Biggiest one-day jump in Covid-19 cases on **May 22 2020**, reported **108.245k** number of patients recovered.

Q3: What is the current total number of active cases worldwide?

> By removing deaths and recoveries from total cases, we can get the "current infected cases" or "active cases".
> Active Cases = Number of Confirmed Cases - (Number of Recovered Cases - Number of Death Cases)

```
datewise_cases(x = date_index_df.index, y = date_index_df["Confirmed"]-(date_index_df["Recovered"]-date_index_df["Deaths"]),
```

Observation:

We are having total **3.795607M** active cases worldwide.

[4.2] Country wise Analysis

```
def Countrywise_cases(df , top_n):
    """
    ================================
    This function will plot the top_n number of countries with the cases so far.
    ================================

    Parameters:
    df: Dataframe which contains Country as index.
    top_n: Top "n" number of counties with the cases (Active, Deaths, Recovered, Confirmed).

    Usage:
    After calling the function, the user have to input the case type (Active, Deaths, Recovered, Confirmed) that he/she wants
    to see the plot.

    """
    print("Confirmed\nRecovered\nDeaths\nActive\n")
    y = input("Enter the case type: ")
    df = df.head(top_n)
    fig = px.bar(df, y= y, x= df.index, text=y)
    fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')
    fig.update_layout(title=f"Distribution of Number of {str(y)} Cases per Country", xaxis_title ="Countries" ,yaxis_title=
    fig.show()
```

```
Confirmed
Recovered
Deaths
```

```
#print top 10 active cases countries with the active cases
list(zip(country_index_df.head(10).index, country_index_df.head(10).Active))
```

```
[('US', 1442382),
 ('Brazil', 303568),
 ('Russia', 232740),
 ('United Kingdom', 309678),
 ('Spain', 115309),
 ('Italy', 112633),
 ('France', 147719),
 ('Germany', 27181),
 ('India', 95844),
 ('Turkey', 40646)]
```

Observation:

Top 10 active cases countries:

('US', 1442382),

('Brazil', 303568),

('Russia', 232740),

('United Kingdom', 309678),

('Spain', 115309),

('Italy', 112633),

('France', 147719),

('Germany', 27181),

('India', 95844),

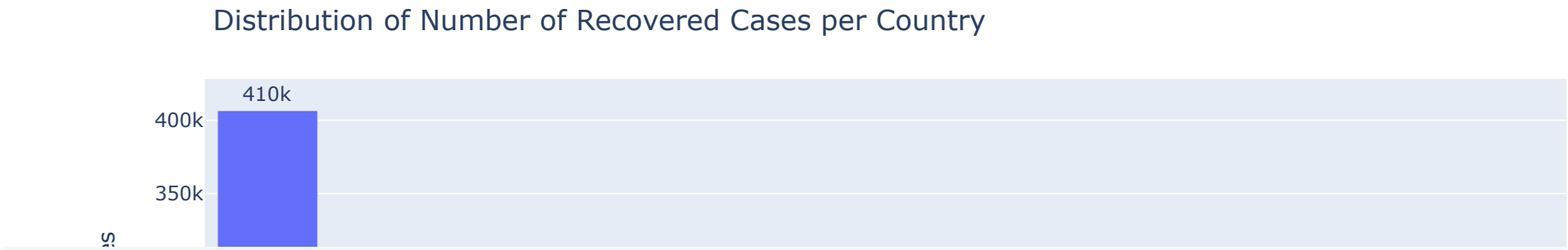
('Turkey', 40646)

```
print(Countrywise_cases.__doc__)
```

```
        ================================
        This function will plot the top_n number of countries with the cases so far.
        ================================

        Parameters:
        df: Dataframe which contains Country as index.
        top_n: Top "n" number of counties with the cases (Active, Deaths, Recovered, Confirmed).

        Usage:
        After calling the function, the user have to input the case type (Active, Deaths, Recovered, Confirmed) that he/she
        to see the plot.
```

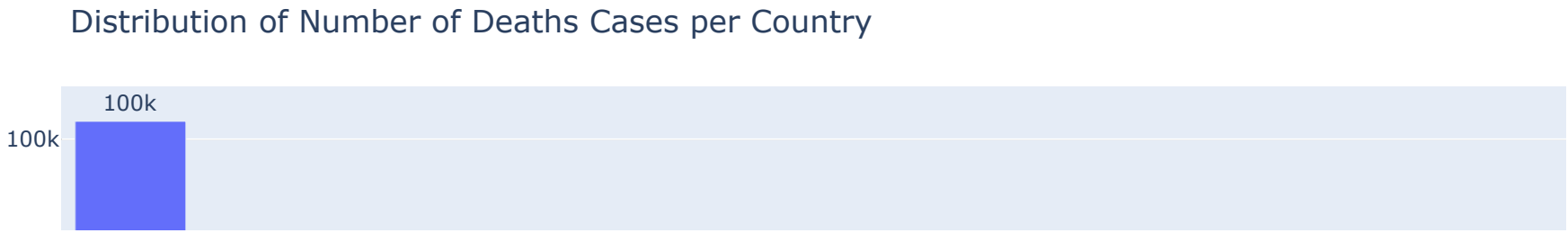
```
# top 20 Recovered cases countries so far
Countrywise_cases(country_index_df, 20)
```

```
Confirmed
Recovered
Deaths
Active

Enter the case type: Recovered
```




```
# top 20 Death cases countries so far
Countrywise_cases(country_index_df, 20)
```

```
Confirmed
Recovered
Deaths
Active

Enter the case type: Deaths
```



Q5: Howmany countries has recorded zero death case?

```
#making a new df with 0 death countries
Zero_death_Country = country_index_df.loc[country_index_df['Deaths'] == 0]
```

```
Zero_death_Country
```

Country	Confirmed	Recovered	Deaths	Active
Rwanda	355	247	0	108
Uganda	329	72	0	257
Vietnam	328	279	0	49
Mongolia	179	43	0	136
Cambodia	124	122	0	2
Eritrea	39	39	0	0
Bhutan	31	6	0	25
Saint Vincent and the Grenadines	26	14	0	12
Timor-Leste	24	24	0	0
Namibia	23	14	0	9
Grenada	23	18	0	5

Observation

```
print(f"There are total {len(Zero_death_Country.index.values)} countries with no Death Cases.")
```

```
There are total 20 countries with no Death Cases.
```

[5] Conclusion:

Analysis like these is useful because they help us understand the most likely outcomes as well as best- and worst-case possibilities. I h find it convenient and helpful to gain a better understanding of the recent coronavirus pandemic.

Lesotho	2	1	0	1