organize imports

In [49]:
```
import pandas as pd
import numpy as np
import wget
```

define the datafile urls

In [23]:
```
urls = [
    'https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_cc
    'https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_cc
    'https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_cc
]
```

Download the datafiles locally

In [24]:
```
[wget.download(url) for url in urls]
```

```
100% [..........................................................................] 865777 / 865777
```

Out[24]:
```
['time_series_covid19_confirmed_global (1).csv',
 'time_series_covid19_deaths_global (1).csv',
 'time_series_covid19_recovered_global (1).csv']
```

Load the datasets

In [25]:
```
confirmed_df = pd.read_csv('time_series_covid19_confirmed_global.csv')
deaths_df = pd.read_csv('time_series_covid19_deaths_global.csv')
recovered_df = pd.read_csv('time_series_covid19_recovered_global.csv')
```

Peek through the datasets

In [26]:
```
confirmed_df
```

Out[26]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	6/7/22	6/8/22	6/9/22	6/10/22	6/11/22	6/
0	NaN	Afghanistan	33.939110	67.709953	0	0	0	0	0	0	...	180741	180784	180864	180864	180864	18
1	NaN	Albania	41.153300	20.168300	0	0	0	0	0	0	...	276468	276518	276583	276638	276690	27
2	NaN	Algeria	28.033900	1.659600	0	0	0	0	0	0	...	265904	265909	265920	265925	265925	26
3	NaN	Andorra	42.506300	1.521800	0	0	0	0	0	0	...	43067	43224	43224	43224	43224	4
4	NaN	Angola	-11.202700	17.873900	0	0	0	0	0	0	...	99761	99761	99761	99761	99761	9
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
280	NaN	West Bank and Gaza	31.952200	35.233200	0	0	0	0	0	0	...	657879	657879	657879	657879	657879	65
281	NaN	Winter Olympics 2022	39.904200	116.407400	0	0	0	0	0	0	...	535	535	535	535	535	
282	NaN	Yemen	15.552727	48.516388	0	0	0	0	0	0	...	11822	11822	11822	11822	11822	1
283	NaN	Zambia	-13.133897	27.849332	0	0	0	0	0	0	...	322562	322790	322919	323058	323058	32
284	NaN	Zimbabwe	-19.015438	29.154857	0	0	0	0	0	0	...	253637	253779	253779	254031	254031	25

285 rows × 881 columns

In [27]:
```
deaths_df
```

Out[27]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	6/7/22	6/8/22	6/9/22	6/10/22	6/11/22	6/12/2
0	NaN	Afghanistan	33.939110	67.709953	0	0	0	0	0	0	...	7709	7709	7709	7709	7709	770
1	NaN	Albania	41.153300	20.168300	0	0	0	0	0	0	...	3497	3497	3497	3497	3497	349
2	NaN	Algeria	28.033900	1.659600	0	0	0	0	0	0	...	6875	6875	6875	6875	6875	687
3	NaN	Andorra	42.506300	1.521800	0	0	0	0	0	0	...	153	153	153	153	153	15
4	NaN	Angola	-11.202700	17.873900	0	0	0	0	0	0	...	1900	1900	1900	1900	1900	190
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
280	NaN	West Bank and Gaza	31.952200	35.233200	0	0	0	0	0	0	...	5660	5660	5660	5660	5660	566
281	NaN	Winter Olympics 2022	39.904200	116.407400	0	0	0	0	0	0	...	0	0	0	0	0	
282	NaN	Yemen	15.552727	48.516388	0	0	0	0	0	0	...	2149	2149	2149	2149	2149	214
283	NaN	Zambia	-13.133897	27.849332	0	0	0	0	0	0	...	3988	3988	3989	3989	3989	398
284	NaN	Zimbabwe	-19.015438	29.154857	0	0	0	0	0	0	...	5515	5515	5515	5518	5518	552

285 rows × 881 columns

In [28]:
```
recovered_df
```

Out[28]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	6/7/22	6/8/22	6/9/22	6/10/22	6/11/22	6/12/2
0	NaN	Afghanistan	33.939110	67.709953	0	0	0	0	0	0	...	0	0	0	0	0	
1	NaN	Albania	41.153300	20.168300	0	0	0	0	0	0	...	0	0	0	0	0	
2	NaN	Algeria	28.033900	1.659600	0	0	0	0	0	0	...	0	0	0	0	0	
3	NaN	Andorra	42.506300	1.521800	0	0	0	0	0	0	...	0	0	0	0	0	
4	NaN	Angola	-11.202700	17.873900	0	0	0	0	0	0	...	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
265	NaN	West Bank and Gaza	31.952200	35.233200	0	0	0	0	0	0	...	0	0	0	0	0	
266	NaN	Winter Olympics 2022	39.904200	116.407400	0	0	0	0	0	0	...	0	0	0	0	0	
267	NaN	Yemen	15.552727	48.516388	0	0	0	0	0	0	...	0	0	0	0	0	
268	NaN	Zambia	-13.133897	27.849332	0	0	0	0	0	0	...	0	0	0	0	0	
269	NaN	Zimbabwe	-19.015438	29.154857	0	0	0	0	0	0	...	0	0	0	0	0	

270 rows × 881 columns

Cleanup the datasets

In [29]:
```
dates = confirmed_df.columns[4:]
confirmed_df_long = confirmed_df.melt(
    id_vars=['Province/State', 'Country/Region', 'Lat', 'Long'],
    value_vars=dates,
    var_name='Date',
    value_name='Confirmed'
)
deaths_df_long = deaths_df.melt(
    id_vars=['Province/State', 'Country/Region', 'Lat', 'Long'],
    value_vars=dates,
    var_name='Date',
    value_name='Deaths'
)
recovered_df_long = recovered_df.melt(
    id_vars=['Province/State', 'Country/Region', 'Lat', 'Long'],
    value_vars=dates,
    var_name='Date',
    value_name='Recovered'
)
```

In [31]:
```
# Remove Canada due to inconsistent data state
recovered_df_long = recovered_df_long[recovered_df_long['Country/Region']!='Canada']
```

Merge the datasets

In [32]:
```
# Merging confirmed_df_long and deaths_df_long
full_table = confirmed_df_long.merge(
  right=deaths_df_long,
  how='left',
  on=['Province/State', 'Country/Region', 'Date', 'Lat', 'Long']
)
# Merging full_table and recovered_df_long
full_table = full_table.merge(
  right=recovered_df_long,
  how='left',
  on=['Province/State', 'Country/Region', 'Date', 'Lat', 'Long']
)
```

Peak into full table

In [33]:
```
full_table
```

Out[33]:

	Province/State	Country/Region	Lat	Long	Date	Confirmed	Deaths	Recovered
0	NaN	Afghanistan	33.939110	67.709953	1/22/20	0	0	0.0
1	NaN	Albania	41.153300	20.168300	1/22/20	0	0	0.0
2	NaN	Algeria	28.033900	1.659600	1/22/20	0	0	0.0
3	NaN	Andorra	42.506300	1.521800	1/22/20	0	0	0.0
4	NaN	Angola	-11.202700	17.873900	1/22/20	0	0	0.0
...	...	...	...	...	...	...	...	...
249940	NaN	West Bank and Gaza	31.952200	35.233200	6/16/22	658100	5660	0.0
249941	NaN	Winter Olympics 2022	39.904200	116.407400	6/16/22	535	0	0.0
249942	NaN	Yemen	15.552727	48.516388	6/16/22	11822	2149	0.0
249943	NaN	Zambia	-13.133897	27.849332	6/16/22	323654	3990	0.0
249944	NaN	Zimbabwe	-19.015438	29.154857	6/16/22	254502	5526	0.0

249945 rows × 8 columns

Further data cleaning

In [34]:
```
# Cleanup the date
full_table['Date'] = pd.to_datetime(full_table['Date'])
```

In [35]:
```
# Peek into cleaned up dataset
full_table
```

Out[35]:

	Province/State	Country/Region	Lat	Long	Date	Confirmed	Deaths	Recovered
0	NaN	Afghanistan	33.939110	67.709953	2020-01-22	0	0	0.0
1	NaN	Albania	41.153300	20.168300	2020-01-22	0	0	0.0
2	NaN	Algeria	28.033900	1.659600	2020-01-22	0	0	0.0
3	NaN	Andorra	42.506300	1.521800	2020-01-22	0	0	0.0
4	NaN	Angola	-11.202700	17.873900	2020-01-22	0	0	0.0
...	...	...	...	...	...	...	...	...
249940	NaN	West Bank and Gaza	31.952200	35.233200	2022-06-16	658100	5660	0.0
249941	NaN	Winter Olympics 2022	39.904200	116.407400	2022-06-16	535	0	0.0
249942	NaN	Yemen	15.552727	48.516388	2022-06-16	11822	2149	0.0
249943	NaN	Zambia	-13.133897	27.849332	2022-06-16	323654	3990	0.0
249944	NaN	Zimbabwe	-19.015438	29.154857	2022-06-16	254502	5526	0.0

249945 rows × 8 columns

In [37]:
```
# Find missing values
full_table.isna().sum()
```

Out[37]:
```
Province/State    171892
Country/Region         0
Lat                 1754
Long                1754
Date                   0
Confirmed              0
Deaths                 0
Recovered          18417
dtype: int64
```

In [38]:
```
# Replace the recovered values (missing values) with 0
full_table['Recovered'] = full_table['Recovered'].fillna(0)
```

In [39]:
```
# Check for missing values
full_table.isna().sum()
```

Out[39]:
```
Province/State    171892
Country/Region         0
Lat                 1754
Long                1754
Date                   0
Confirmed              0
Deaths                 0
Recovered              0
dtype: int64
```

Extract the ship data

In [41]:
```
ship_rows = full_table['Province/State'].str.contains('Grand Princess') | full_table['Province/State'].str.contains('Diamond Prir
full_ship = full_table[ship_rows]
```

Remove the ship data from the full table

In [42]:
```
full_table = full_table[~(ship_rows)]
```

We add an active cases column Active (calculated by active = confirmed — deaths — recovered)

In [43]:
```
# Active Case = confirmed - deaths - recovered
full_table['Active'] = full_table['Confirmed'] - full_table['Deaths'] - full_table['Recovered']
```

In [44]:
```
full_table
```

Out[44]:

	Province/State	Country/Region	Lat	Long	Date	Confirmed	Deaths	Recovered	Active
0	NaN	Afghanistan	33.939110	67.709953	2020-01-22	0	0	0.0	0.0
1	NaN	Albania	41.153300	20.168300	2020-01-22	0	0	0.0	0.0
2	NaN	Algeria	28.033900	1.659600	2020-01-22	0	0	0.0	0.0
3	NaN	Andorra	42.506300	1.521800	2020-01-22	0	0	0.0	0.0
4	NaN	Angola	-11.202700	17.873900	2020-01-22	0	0	0.0	0.0
...	...	...	...	...	...	...	...	...	...

249940	NaN	West Bank and Gaza	31.952200	35.233200	2022-06-16	658100	5660	0.0	652440.0
249941	NaN	Winter Olympics 2022	39.904200	116.407400	2022-06-16	535	0	0.0	535.0
249942	NaN	Yemen	15.552727	48.516388	2022-06-16	11822	2149	0.0	9673.0
249943	NaN	Zambia	-13.133897	27.849332	2022-06-16	323654	3990	0.0	319664.0
249944	NaN	Zimbabwe	-19.015438	29.154857	2022-06-16	254502	5526	0.0	248976.0

246437 rows × 9 columns

Aggregate data into Country/Region wise and group them by Date and Country/Region

In [46]:
```
# sum() is to get the total count of 'Confirmed', 'Deaths', 'Recovered', 'Active' for the given Date and Country/Region
# reset_index() reset the index and use the default one, which is Date and Country/Region

full_grouped = full_table.groupby(['Date', 'Country/Region'])['Confirmed', 'Deaths', 'Recovered', 'Active'].sum().reset_index()
```

In [47]:
```
full_grouped
```

Out[47]:

	Date	Country/Region	Confirmed	Deaths	Recovered	Active
0	2020-01-22	Afghanistan	0	0	0.0	0.0
1	2020-01-22	Albania	0	0	0.0	0.0
2	2020-01-22	Algeria	0	0	0.0	0.0
3	2020-01-22	Andorra	0	0	0.0	0.0
4	2020-01-22	Angola	0	0	0.0	0.0
...	...	...	...	...	...	...
172764	2022-06-16	West Bank and Gaza	658100	5660	0.0	652440.0
172765	2022-06-16	Winter Olympics 2022	535	0	0.0	535.0
172766	2022-06-16	Yemen	11822	2149	0.0	9673.0
172767	2022-06-16	Zambia	323654	3990	0.0	319664.0
172768	2022-06-16	Zimbabwe	254502	5526	0.0	248976.0

172769 rows × 6 columns

We add day wise New cases, New deaths and New recovered

In [50]:
```
# new cases
temp = full_grouped.groupby(['Country/Region', 'Date', ])['Confirmed', 'Deaths', 'Recovered']
temp = temp.sum().diff().reset_index()
mask = temp['Country/Region'] != temp['Country/Region'].shift(1)
temp.loc[mask, 'Confirmed'] = np.nan
temp.loc[mask, 'Deaths'] = np.nan
temp.loc[mask, 'Recovered'] = np.nan
# renaming columns
temp.columns = ['Country/Region', 'Date', 'New cases', 'New deaths', 'New recovered']
# merging new values
full_grouped = pd.merge(full_grouped, temp, on=['Country/Region', 'Date'])
# filling na with 0
full_grouped = full_grouped.fillna(0)
# fixing data types
cols = ['New cases', 'New deaths', 'New recovered']
full_grouped[cols] = full_grouped[cols].astype('int')
#
full_grouped['New cases'] = full_grouped['New cases'].apply(lambda x: 0 if x<0 else x)
```

Save the cleaned up data

In [51]:
```
full_grouped.to_csv('COVID-19-time-series-clean-complete.csv')
```

Data Exploration

Import Libraries

In [73]:
```
import pandas as pd
import altair as alt
alt.data_transformers.enable('default', max_rows=None)
```

Out[73]:
```
DataTransformerRegistry.enable('default')
```

In [74]:
```
full_grouped = pd.read_csv('COVID-19-time-series-clean-complete.csv', parse_dates=['Date'])
full_grouped
```

Out[74]:

	Unnamed: 0	Date	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered
0	0	2020-01-22	Afghanistan	0	0	0.0	0.0	0	0	0
1	1	2020-01-22	Albania	0	0	0.0	0.0	0	0	0
2	2	2020-01-22	Algeria	0	0	0.0	0.0	0	0	0
3	3	2020-01-22	Andorra	0	0	0.0	0.0	0	0	0
4	4	2020-01-22	Angola	0	0	0.0	0.0	0	0	0

...	...	...	...	...	...	...	...	...	...	...
172764	172764	2022-06-16	West Bank and Gaza	658100	5660	0.0	652440.0	0	0	0
172765	172765	2022-06-16	Winter Olympics 2022	535	0	0.0	535.0	0	0	0
172766	172766	2022-06-16	Yemen	11822	2149	0.0	9673.0	0	0	0
172767	172767	2022-06-16	Zambia	323654	3990	0.0	319664.0	0	0	0
172768	172768	2022-06-16	Zimbabwe	254502	5526	0.0	248976.0	115	1	0

172769 rows × 10 columns

Showing total and daily cases

In [75]:
```
uk = full_grouped[full_grouped['Country/Region'] == 'United Kingdom']
```

In [84]:
```
uk
```

Out[84]:

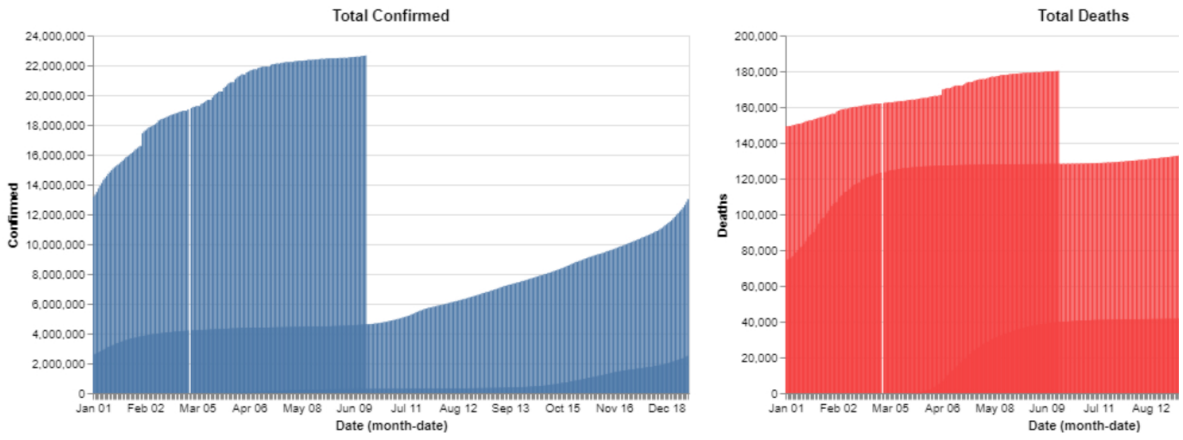
	Unnamed: 0	Date	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered
186	186	2020-01-22	United Kingdom	0	0	0.0	0.0	0	0	0
383	383	2020-01-23	United Kingdom	0	0	0.0	0.0	0	0	0
580	580	2020-01-24	United Kingdom	0	0	0.0	0.0	0	0	0
777	777	2020-01-25	United Kingdom	0	0	0.0	0.0	0	0	0
974	974	2020-01-26	United Kingdom	0	0	0.0	0.0	0	0	0
...	...	...	...	...	...	...	...	...	...	...
171970	171970	2022-06-12	United Kingdom	22571882	179884	0.0	22391998.0	2	0	0
172167	172167	2022-06-13	United Kingdom	22600145	179941	0.0	22420204.0	28263	57	0
172364	172364	2022-06-14	United Kingdom	22612173	180030	0.0	22432143.0	12028	89	0
172561	172561	2022-06-15	United Kingdom	22638832	180078	0.0	22458754.0	26659	48	0
172758	172758	2022-06-16	United Kingdom	22651908	180139	0.0	22471769.0	13076	61	0

877 rows × 10 columns

In [91]:
```
base = alt.Chart(uk).mark_bar().encode(
    x='monthdate(Date):O',
).properties(
    width=500
)
```

In [93]:
```
red = alt.value("#f54242")
base.encode(y = 'Confirmed').properties(title = 'Total Confirmed') | base.encode( y = 'Deaths', color = red).properties( title =
```

Out[93]:



Showing the coronavirus spread

In [78]:
```
full_grouped = pd.read_csv('COVID-19-time-series-clean-complete.csv', parse_dates=['Date'])
countries = ['US', 'Italy', 'China', 'Spain', 'Germany', 'France', 'Iran', 'United Kingdom', 'Switzerland']
selected_countries = full_grouped[full_grouped['Country/Region'].isin(countries)]
```

In [79]:
```
selected_countries
```

Out[79]:

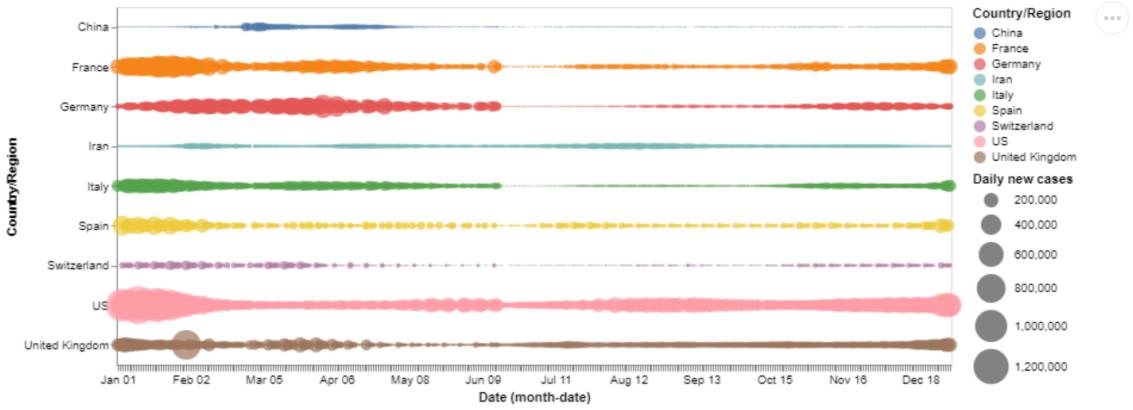
	Unnamed: 0	Date	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered
37	37	2020-01-22	China	548	17	28.0	503.0	0	0	0
62	62	2020-01-22	France	0	0	0.0	0.0	0	0	0
66	66	2020-01-22	Germany	0	0	0.0	0.0	0	0	0
81	81	2020-01-22	Iran	0	0	0.0	0.0	0	0	0
85	85	2020-01-22	Italy	0	0	0.0	0.0	0	0	0
...	...	...	...	...	...	...	...	...	...	...
172657	172657	2022-06-16	Italy	17773764	167617	0.0	17606147.0	37068	64	0

172736	172736	2022-06-16	Spain	12515127	107239	0.0	12407888.0	0	0	0
172742	172742	2022-06-16	Switzerland	3701895	13817	0.0	3688078.0	0	0	0
172754	172754	2022-06-16	US	86057735	1012647	0.0	85045088.0	116445	40	0
172758	172758	2022-06-16	United Kingdom	22651908	180139	0.0	22471769.0	13076	61	0

7893 rows × 10 columns

In [86]:
```
alt.Chart(selected_countries).mark_circle().encode(
    x='monthdate(Date):O',
    y='Country/Region',
    color='Country/Region',
    size=alt.Size('New cases:Q',
        scale=alt.Scale(range=[0, 1000]),
        legend=alt.Legend(title='Daily new cases')
    )
).properties(
    width=700,
    height=300
)
```

Out[86]:



In [ ]: