sample covid Data analysis

Objective: conduct some analyses based on the characteristics of the data.

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, to_date, year, month, avg, sum as _sum, expr, lit
import requests
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go

#  Download JSON from GitHub
url = "https://raw.githubusercontent.com/Aless13260/covid-pipeline/main/sample_data.json"
local_path = "/tmp/sample_data.json"
with open(local_path, "w") as f:
    f.write(requests.get(url).text)
df=pd.read_json(url)
# mian info check
print(df.head(10))
print(f"\n Data types:\n{df.dtypes}")
print(f"Total rows: {len(df)}")
print(f"Total rows with missing value:\n{df.isna().sum()}")
missing_counts = df.isna().sum()
missing_ratio = (missing_counts / len(df) * 100).round(2)
print(f"\n Missing value ratio (%):\n{missing_ratio}")
```

```
         date         country                     state  confirmed source  \
0 2020-10-13  United Kingdom  Turks and Caicos Islands      696.0    JHU
1 2020-10-14  United Kingdom  Turks and Caicos Islands      696.0    JHU
2 2020-10-15  United Kingdom  Turks and Caicos Islands      696.0    JHU
3 2020-10-16  United Kingdom  Turks and Caicos Islands      697.0    JHU
4 2020-10-17  United Kingdom  Turks and Caicos Islands      698.0    JHU
5 2020-10-18  United Kingdom  Turks and Caicos Islands      698.0    JHU
6 2020-10-19  United Kingdom  Turks and Caicos Islands      698.0    JHU
7 2020-10-20  United Kingdom  Turks and Caicos Islands      698.0    JHU
8 2020-10-21  United Kingdom  Turks and Caicos Islands      698.0    JHU
9 2020-10-22  United Kingdom  Turks and Caicos Islands      698.0    JHU

   deaths
0     NaN
1     NaN
2     NaN
3     NaN
4     NaN
5     NaN
6     NaN
```

```
7     NaN
8     NaN
9     NaN

 Data types:
date         datetime64[ns]
country              object
state                object
confirmed           float64
source               object
deaths              float64
dtype: object
Total rows: 760920
Total rows with missing value:
date              0
country           0
state        656907
confirmed     17631
source            0
deaths       347958
dtype: int64

 Missing value ratio (%):
date          0.00
country       0.00
state        86.33
confirmed     2.32
source        0.00
deaths       45.73
dtype: float64
```

```
#unique values check

# for country
countries = df["country"].dropna().unique().tolist()
countries.sort()
print(f"Total unique countries: {len(countries)}")
print("Countries are:", countries[:])

#for state
states=df["state"].dropna().unique().tolist()
states.sort()
df["state"].unique()
print(f"Total unique sates: {len(states)}")
print("States are:", states[:])
```

```
Total unique countries: 272
Countries are: ['Afghanistan', 'Africa', 'Albania', 'Algeria', 'American Samoa', 'Andorra', 'Angola', 'Anguilla', 'Antarctica', 'Antigua and Barbuda', '
Total unique sates: 91
States are: ['Alberta', 'Anguilla', 'Anhui', 'Aruba', 'Australian Capital Territory', 'Beijing', 'Bermuda', 'Bonaire, Sint Eustatius and Saba', 'British
```

Observatone1: According to the results of the missing value check, the national data is basically unavailable, but most of the state data are missing. In this case , for the subsequent analysis, the geographical location analysis will mainly focus on the country.

Observatone2: Death data is more lacking than confirmed data, and data sources may not be comprehensive enough for death statistics.Similarly, some of the confirmed data is missing

Therefore, the data analysis will be conducted from the following perspectives:

Perspective 1: Anomaly detection

Perspective 2: Overview of cumulative confirmed cases vs cumulative deaths

Perspective 3: Identification of daily changes (including new confirmed cases)

```
# Libraries  load and data prepare

import pandas as pd
import plotly.express as px
import plotly.graph_objects as go


df["date"] = pd.to_datetime(df["date"])
df = df.sort_values(["country", "state", "date"])

#  Enrich the data,add daily_new_cases / daily_new_deaths

df["prev_conf"] = df.groupby(["country", "state"])["confirmed"].shift(1)
df["prev_death"] = df.groupby(["country", "state"])["deaths"]   .shift(1)

df["daily_new_cases"]  = (df["confirmed"] - df["prev_conf"]).fillna(0)
df["daily_new_deaths"] = (df["deaths"]    - df["prev_death"]).fillna(0)

df["daily_new_cases"]  = df["daily_new_cases"].clip(lower=0)
df["daily_new_deaths"] = df["daily_new_deaths"].clip(lower=0)

print(df.head(10))
print(f"Total rows with missing value:\n{df.isna().sum()}")
```

```
            date      country state  confirmed source  deaths  prev_conf  \
36374 2020-01-05  Afghanistan   NaN        0.0   OWID     0.0        NaN
36375 2020-01-06  Afghanistan   NaN        0.0   OWID     0.0        NaN
36376 2020-01-07  Afghanistan   NaN        0.0   OWID     0.0        NaN
36377 2020-01-08  Afghanistan   NaN        0.0   OWID     0.0        NaN
36378 2020-01-09  Afghanistan   NaN        0.0   OWID     0.0        NaN
36379 2020-01-10  Afghanistan   NaN        0.0   OWID     0.0        NaN
36380 2020-01-11  Afghanistan   NaN        0.0   OWID     0.0        NaN
36381 2020-01-12  Afghanistan   NaN        0.0   OWID     0.0        NaN
36382 2020-01-13  Afghanistan   NaN        0.0   OWID     0.0        NaN
```

```
       36383 2020-01-14  Afghanistan   NaN        0.0   OWID     0.0        NaN

             prev_death  daily_new_cases  daily_new_deaths
       36374         NaN              0.0               0.0
       36375         NaN              0.0               0.0
       36376         NaN              0.0               0.0
       36377         NaN              0.0               0.0
       36378         NaN              0.0               0.0
       36379         NaN              0.0               0.0
       36380         NaN              0.0               0.0
       36381         NaN              0.0               0.0
       36382         NaN              0.0               0.0
       36383         NaN              0.0               0.0
       Total rows with missing value:
       date                      0
       country                   0
       state                656907
       confirmed             17631
       source                    0
       deaths               347958
       prev_conf            656998
       prev_death           760920
       daily_new_cases           0
       daily_new_deaths          0
       dtype: int64
```

Perspective 1 – Anomaly Detection

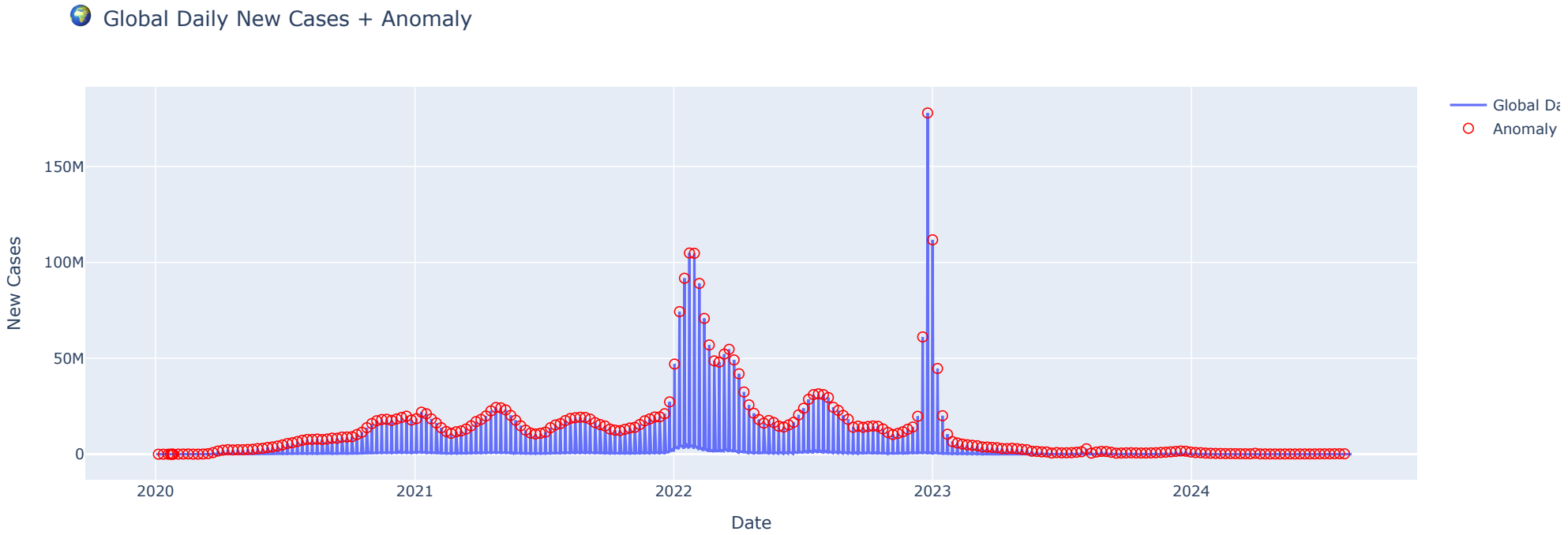
Goal: Quickly spot unusual spikes / drops in each country's daily new confirmed-case curve, so that analysts can:

(1) flag potential data-quality issues or reporting delays.

(2) highlight real epidemiological surges worth deeper investigation.

```
import plotly.graph_objects as go
import pandas as pd

# global cumulative confirmed cases by date
world_cum = (df.groupby("date")["confirmed"]
               .sum()
               .reset_index()
               .sort_values("date"))

# Calculate daily additions and trim negative values
world_cum["prev"]       = world_cum["confirmed"].shift(1)
world_cum["daily_new"]  = (world_cum["confirmed"] - world_cum["prev"]).clip(lower=0)

#  7-day rolling mean & anomaly threshold (1.5)
```

```
world_cum["roll"]     = world_cum["daily_new"].rolling(7, min_periods=1).mean()
world_cum["anomaly"]  = world_cum["daily_new"] > world_cum["roll"] * 1.5

#  Draw global curve & abnormal red point
fig_world = go.Figure()

fig_world.add_scatter(
    x=world_cum["date"],
    y=world_cum["daily_new"],
    mode="lines",
    name="Global Daily New"
)

fig_world.add_scatter(
    x=world_cum[world_cum["anomaly"]]["date"],
    y=world_cum[world_cum["anomaly"]]["daily_new"],
    mode="markers",
    name="Anomaly",
    marker=dict(color="red", size=8, symbol="circle-open")
)

fig_world.update_layout(
    title="🌍 Global Daily New Cases + Anomaly",
    xaxis_title="Date",
    yaxis_title="New Cases",
    height=500
)

fig_world.show()
```



The significant anomaly spikes are concentrated in two key periods:

(1) The first notable peak occurred around early 2022, where the red markers are densely clustered, indicating a sharp surge in daily new cases.

The second extreme anomaly appeared between late 2022 and early 2023. On one particular day, the number of new cases exceeded 160 million, representing an extreme outlier—likely caused by data backlog, bulk reporting, or statistical error.

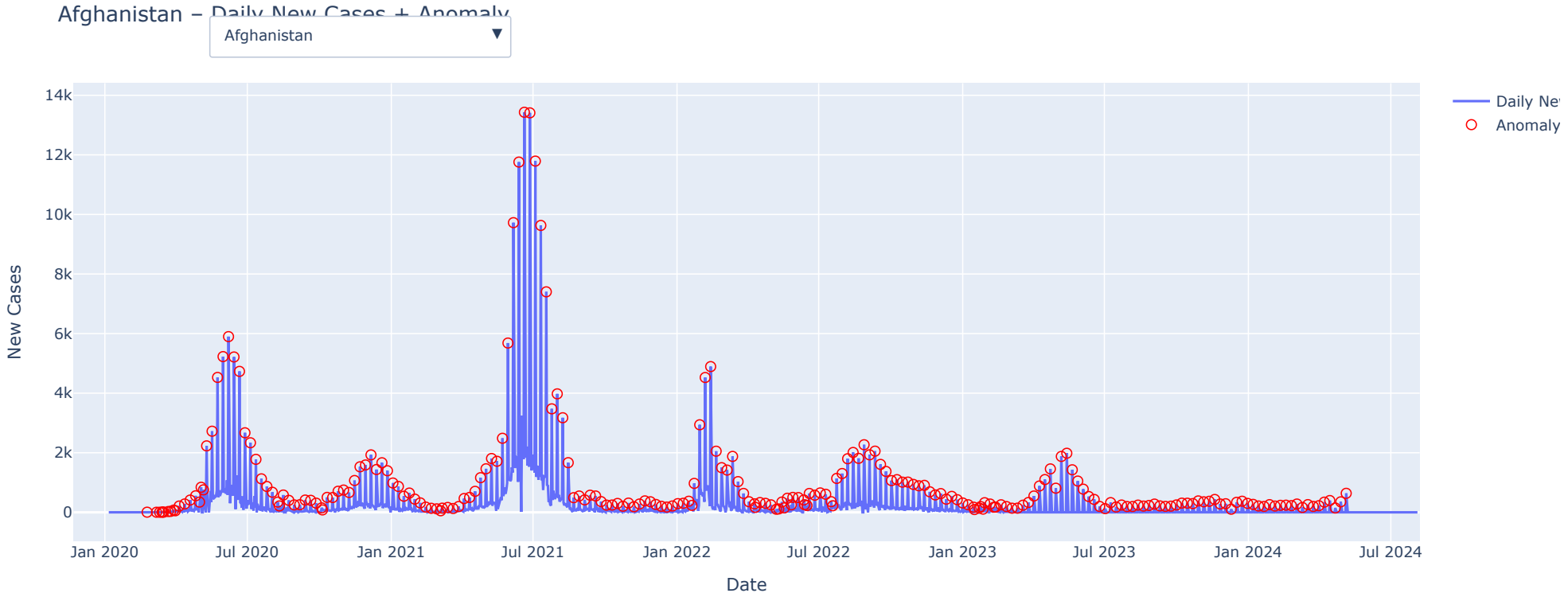
(2) Anomalies decreased significantly in the later period, with trends stabilizing

After mid-2023, the number of anomaly points dropped noticeably, suggesting that the global pandemic entered a phase of slower growth or better control.From a data reporting perspective, this trend may also reflect a reduction in reporting frequency or coverage.

```
# Anomalies for each country

#Prepare a list of countries
country_list = df["country"].dropna().unique().tolist()
country_list.sort()
country_list = country_list[:]
```

```
    # save all the  traces
    traces = []
    buttons = []

    for idx, country in enumerate(country_list):
        df_c = (
            df[df["country"] == country]
            .groupby("date")[["confirmed"]].sum().reset_index().sort_values("date")
        )
        df_c["prev"] = df_c["confirmed"].shift(1)
        df_c["daily_new"] = df_c["confirmed"] - df_c["prev"]
        df_c["daily_new"] = df_c["daily_new"].clip(lower=0)
        df_c["roll"] = df_c["daily_new"].rolling(7, min_periods=1).mean()
        df_c["anomaly"] = df_c["daily_new"] > df_c["roll"] * 1.5


        trace_main = go.Scatter(
            x=df_c["date"],
            y=df_c["daily_new"],
            mode="lines",
            name="Daily New Cases",
            visible=(idx == 0)
        )
        # Abnormal point: red circle
        trace_anom = go.Scatter(
            x=df_c[df_c["anomaly"]]["date"],
            y=df_c[df_c["anomaly"]]["daily_new"],
            mode="markers",
            name="Anomaly",
            marker=dict(color="red", size=8, symbol="circle-open"),
            visible=(idx == 0)
        )

        traces.extend([trace_main, trace_anom])


        visible_array = [False] * (2 * len(country_list))
        visible_array[2 * idx] = True
        visible_array[2 * idx + 1] = True

        buttons.append(dict(
            label=country,
            method="update",
            args=[
                {"visible": visible_array},
                {"title": f"{country} — Daily New Cases + Anomaly"}
            ]
```

```
        ))

    fig = go.Figure(data=traces)

    fig.update_layout(
        title=f"{country_list[0]} — Daily New Cases + Anomaly",
        xaxis_title="Date",
        yaxis_title="New Cases",
        updatemenus=[
            {
                "buttons": buttons,
                "direction": "down",
                "showactive": True,
                "x": 0.1,
                "xanchor": "left",
                "y": 1.15,
                "yanchor": "top"
            }
        ],
        height=550
    )

    fig.show()
```

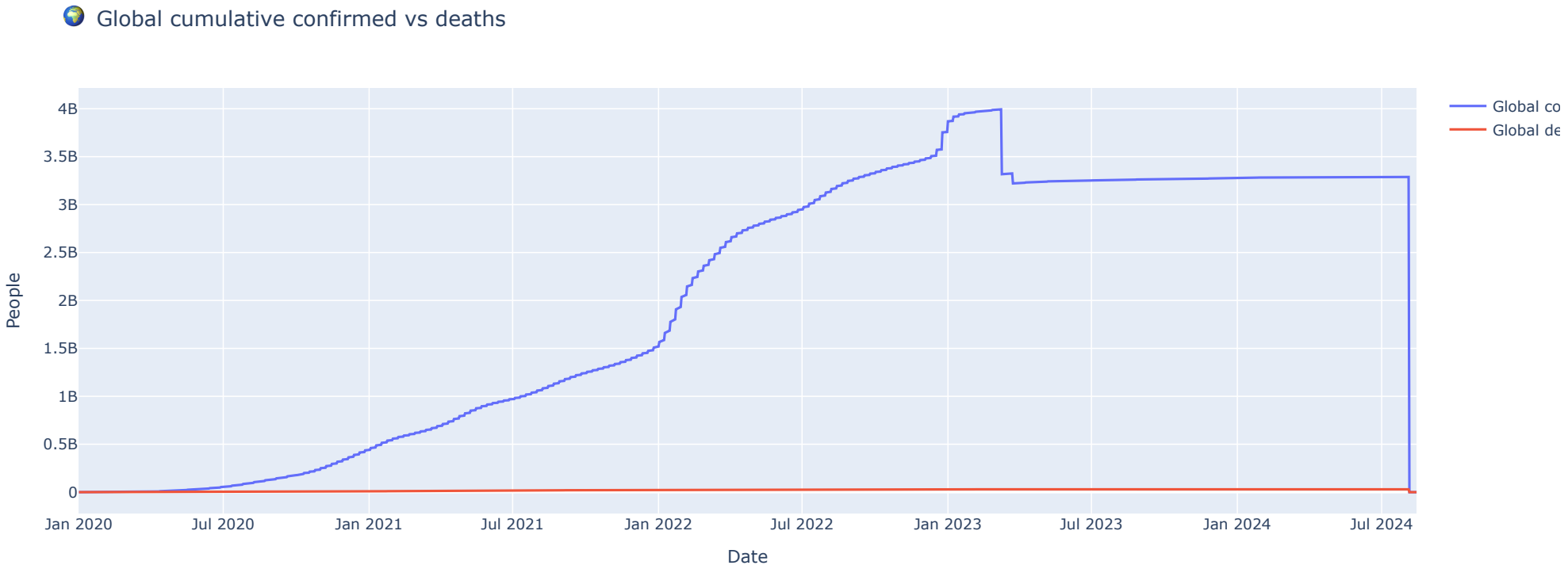


Perspective 2: Overview of cumulative confirmed cases vs cumulative deaths

Goal: To provide a clear and comparative view of how severely different countries or regions have been impacted by COVID-19 in terms of total infections and total deaths.

```
# Cumulative confirmed and deaths cases
# GLOBAL cumulative trend
global_daily = (df.groupby("date")[["confirmed", "deaths"]]
                  .sum()
                  .reset_index())

fig_global = go.Figure()
fig_global.add_scatter(x=global_daily["date"],
                       y=global_daily["confirmed"],
                       mode="lines",
```

```
                      name="Global confirmed")
fig_global.add_scatter(x=global_daily["date"],
                       y=global_daily["deaths"],
                       mode="lines",
                       name="Global deaths")
fig_global.update_layout(title="🌍 Global cumulative confirmed vs deaths",
                         xaxis_title="Date",
                         yaxis_title="People")
fig_global.show()
```



Overall trend :

(1) Confirmed cases continued to rise steadily, while deaths remained relatively stable.

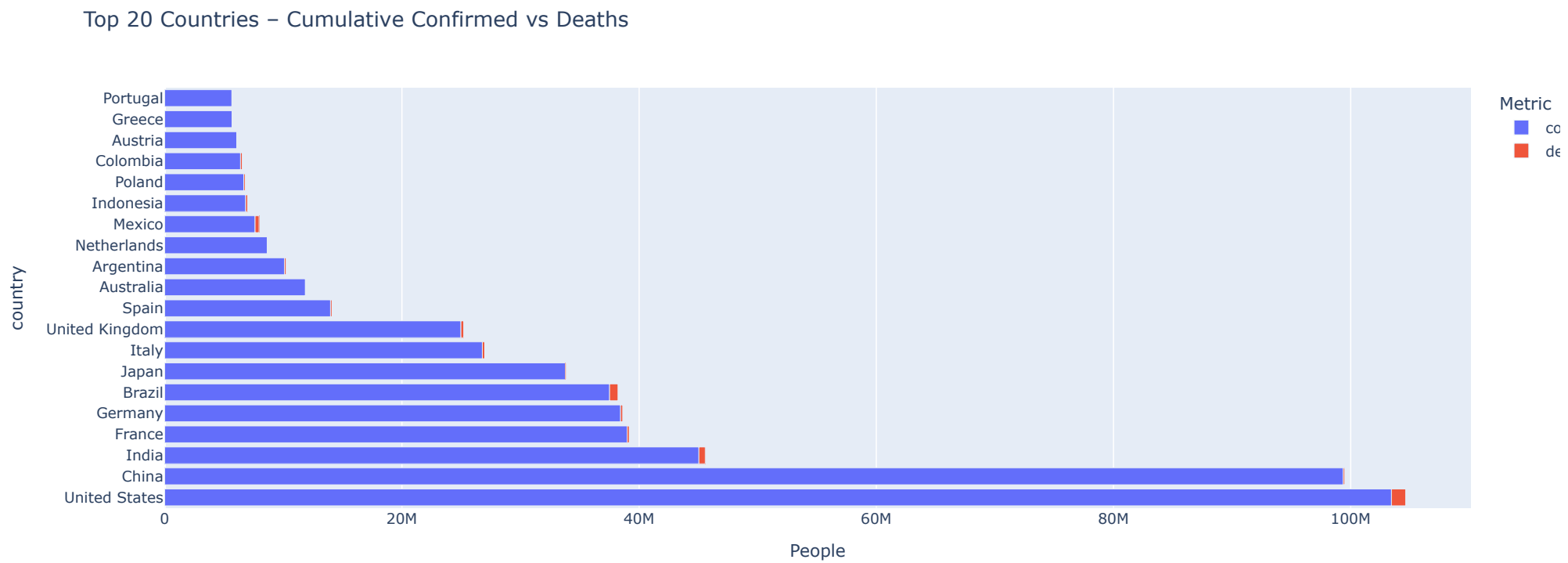
(2) The cumulative confirmed curve shows a steep upward trend, reflecting the wide spread and rapid transmission of COVID-19 globally.

(3) In contrast, the cumulative death count remained at a much lower level, likely due to a lower fatality rate compared to the transmission rate, or possibly due to gaps in death reporting.

(4) Rapid growth period (early 2022 – early 2023): During this period, the confirmed case curve rose most sharply, indicating a global surge in cases within a short timeframe.

```
# 2.Per-country cumulative totals

# Get the latest data for each country
latest = (df.sort_values("date")
            .groupby("country")
            .last()
            .reset_index())

!pip install pycountry
import pycountry

valid_countries = [c.name for c in pycountry.countries]

latest = latest[latest["country"].isin(valid_countries)]



# Top 20 country
topN = latest.sort_values("confirmed", ascending=False).head(20)

fig_bar = px.bar(
    topN,
    y="country",
    x=["confirmed", "deaths"],
    orientation="h",
    title="Top 20 Countries – Cumulative Confirmed vs Deaths",
    labels={"value": "People", "variable": "Metric"}
)
fig_bar.show()
```

```
Requirement already satisfied: pycountry in /usr/local/lib/python3.11/dist-packages (24.6.1)
```



Key Insights from Top 20 Countries

(1) The United States, India, and China rank top three in total confirmed cases The United States currently has the highest cumulative number of confirmed cases, exceeding 100 million. India and China follow closely, each with over 90 million cases. This is likely due to their large population base and the prolonged duration of virus transmission in these countries, resulting in high overall case counts.

(2) The proportion of deaths to confirmed cases varies significantly by country Although the red bars (deaths) are relatively small in the chart, we can observe that Mexico and Brazil show a noticeably higher death rate. In contrast, countries like Australia and Japan have very slim red bars, indicating a relatively low fatality rate.

(3) European countries account for a large portion of the Top 20 This suggests that the overall number of confirmed cases in Europe was high during the pandemic. Possible reasons include early virus spread, improved testing capacity, and higher data transparency in the region.

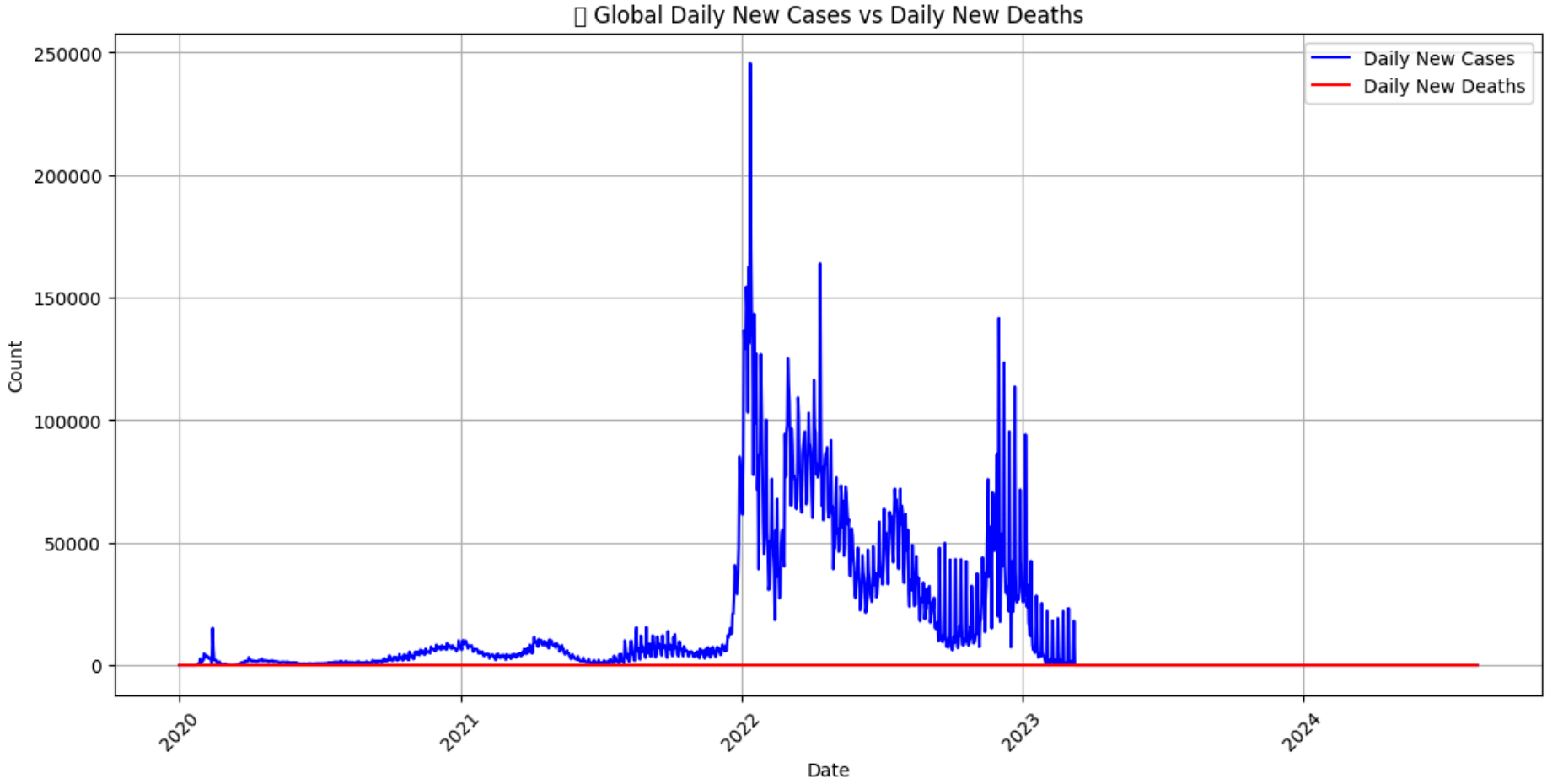
(4) Some mid-population countries also show high confirmed case numbers Countries such as Argentina, Indonesia, and Colombia are included in the Top 20, indicating that middle-income countries have faced challenges in controlling the pandemic at certain stages. Although their populations are smaller than China, India, or the U.S., their long-term cumulative case counts are still among the highest globally.

Perspective 3: Identification of daily changes (including new confirmed cases and new deaths)

Goal:To monitor the evolving patterns of the pandemic by tracking daily new confirmed cases and deaths, enabling timely detection of trends, cross-country comparisons, and identification of abnormal data patterns.

```
# Aggregate the number of newly confirmed cases and deaths worldwide by date
global_daily = df.groupby("date")[["daily_new_cases", "daily_new_deaths"]].sum().reset_index()

plt.figure(figsize=(12, 6))
plt.plot(global_daily["date"], global_daily["daily_new_cases"], label="Daily New Cases", color="blue")
plt.plot(global_daily["date"], global_daily["daily_new_deaths"], label="Daily New Deaths", color="red")
plt.title("🌍 Global Daily New Cases vs Daily New Deaths")
plt.xlabel("Date")
plt.ylabel("Count")
plt.legend()
plt.tight_layout()
plt.grid(True)
plt.xticks(rotation=45)
plt.show()
```

```
<ipython-input-50-1709123224>:11: UserWarning:
Glyph 127757 (\N{EARTH GLOBE EUROPE-AFRICA}) missing from font(s) DejaVu Sans.
/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarning:
Glyph 127757 (\N{EARTH GLOBE EUROPE-AFRICA}) missing from font(s) DejaVu Sans.
```



The trends of daily new confirmed cases and deaths are consistent with the overall cumulative trends. Specifically:

(1) The daily new confirmed cases curve (blue line) shows significant fluctuations and multiple waves of outbreaks. There are several distinct peaks, with the highest surge occurring in early 2022, when daily new confirmed cases nearly reached 250,000. The frequent fluctuations

indicate highly active virus transmission, possibly influenced by seasonal factors, new variants (such as Omicron), and changes in public health policies.

(2) The daily new deaths curve (red line) is almost flat and lies close to the horizontal axis. It is barely visible, suggesting that the number of deaths is extremely low compared to the number of confirmed cases. This may be due to effective control measures—such as widespread vaccination and timely medical intervention—or because death data is severely underreported or not updated, especially after 2023 when the red line almost disappears.

(3)A major outbreak period is concentrated in 2022. This year marked the most rapid increase in daily new cases, followed by several smaller resurgences. After late 2023 into 2024, the number of new confirmed cases gradually leveled off or showed data interruptions, which may reflect both pandemic stabilization and gaps in data reporting.

```
# Daily Newly confirmed cases visualization: by country & states
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go

# Ensure the sorting is correct and calculate the daily increase
df = df.sort_values(by=["country", "state", "date"])
df["prev_confirmed"] = df.groupby(["country", "state"])["confirmed"].shift(1)
df["daily_new_cases"] = (df["confirmed"] - df["prev_confirmed"]).fillna(0)
df["daily_new_cases"] = df["daily_new_cases"].clip(lower=0)  # 去掉负值

# Aggregate by country + date
country_daily = (
    df.groupby(["country", "date"])["daily_new_cases"]
      .sum()
      .reset_index()
      .sort_values(["country", "date"])
)

#Filter out countries with "no new cases"
valid_countries = (
    country_daily.groupby("country")["daily_new_cases"].sum()
    .loc[lambda x: x > 0]
    .index.tolist()
)
country_daily = country_daily[country_daily["country"].isin(valid_countries)]

country_list = sorted(valid_countries)
initial_country = country_list[0]
df_init = country_daily[country_daily["country"] == initial_country]

fig = px.line(
    df_init,
    x="date",
```

```
        y="daily_new_cases",
        title=f"📈 Daily New Cases — {initial_country}",
        markers=True
    )

    buttons = [
```