NAME: SHAILESH PANDEY

Data Analytics

Topic : Exploratory Data Analysis on Dataset - Terrorism

EDA - Terrorim

Objective:

- Perform 'Exploratory Data Analysis' on dataset 'Global Terrorism'
- As a security/defense analyst, try to find out the hot zone of terrorism.
- What all security issues and insights you can derive by EDA?

In [1]:
```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import math
import warnings
import numpy as np           # linear algebra
import pandas as pd          # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns
import plotly.offline as py
import plotly.graph_objs as go
import matplotlib.pyplot as plt
warnings.filterwarnings('ignore')

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory
import os
for dirname, _, filenames in os.walk(' '):
    for filename in filenames:
        print(os.path.join(dirname, filename))


# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

In [2]:
```
# Let's import to our data and check the basics.
terror = pd.read_csv('globalterrorismdb_0718dist.csv',encoding='ISO-8859-1')
```

In [3]:
```
terror.head()
```

Out[3]:

	eventid	iyear	imonth	iday	approxdate	extended	resolution	country	country_txt	region	...	addnotes	scite1	scite2	scite3	dbsource	INT_LOG	INT_IDEO	INT_MISC	INT_ANY	related
0	197000000001	1970	7	2	NaN	0	NaN	58	Dominican Republic	2	...	NaN	NaN	NaN	NaN	PGIS	0	0	0	0	NaN
1	197000000002	1970	0	0	NaN	0	NaN	130	Mexico	1	...	NaN	NaN	NaN	NaN	PGIS	0	1	1	1	NaN
2	197001000001	1970	1	0	NaN	0	NaN	160	Philippines	5	...	NaN	NaN	NaN	NaN	PGIS	-9	-9	1	1	NaN
3	197001000002	1970	1	0	NaN	0	NaN	78	Greece	8	...	NaN	NaN	NaN	NaN	PGIS	-9	-9	1	1	NaN
4	197001000003	1970	1	0	NaN	0	NaN	101	Japan	4	...	NaN	NaN	NaN	NaN	PGIS	-9	-9	1	1	NaN

5 rows × 135 columns

In [4]:
```
terror.tail()
```

Out[4]:

	eventid	iyear	imonth	iday	approxdate	extended	resolution	country	country_txt	region	...	addnotes	scite1	scite2	scite3	dbsource	INT_LOG	INT_IDEO	INT_MISC	INT_ANY	related
181686	201712310022	2017	12	31	NaN	0	NaN	182	Somalia	11	...	NaN	"Somalia: Al-Shabaab Militants Attack Army Che...	"Highlights: Somalia Daily Media Highlights 2 ...	"Highlights: Somalia Daily Media Highlights 1 ...	START Primary Collection	0	0	0	0	NaN
181687	201712310029	2017	12	31	NaN	0	NaN	200	Syria	10	...	NaN	"Putin's 'victory' in Syria has turned into a ...	"Two Russian soldiers killed at Hmeymim base i...	"Two Russian servicemen killed in Syria mortar...	START Primary Collection	-9	-9	1	1	NaN
181688	201712310030	2017	12	31	NaN	0	NaN	160	Philippines	5	...	NaN	"Maguindanao clashes trap tribe members," Phil...	NaN	NaN	START Primary Collection	0	0	0	0	NaN
181689	201712310031	2017	12	31	NaN	0	NaN	92	India	6	...	NaN	"Trader escapes grenade attack in Imphal," Bus...	NaN	NaN	START Primary Collection	-9	-9	0	-9	NaN
181690	201712310032	2017	12	31	NaN	0	NaN	160	Philippines	5	...	NaN	"Security tightened in Cotabato following IED ...	"Security tightened in Cotabato City," Manila ...	NaN	START Primary Collection	-9	-9	0	-9	NaN

5 rows × 135 columns

In [5]:
```
terror.columns
```

Out[5]:
```
Index(['eventid', 'iyear', 'imonth', 'iday', 'approxdate', 'extended',
       'resolution', 'country', 'country_txt', 'region',
       ...
       'addnotes', 'scite1', 'scite2', 'scite3', 'dbsource', 'INT_LOG',
       'INT_IDEO', 'INT_MISC', 'INT_ANY', 'related'],
      dtype='object', length=135)
```

There are to many columns, I didn't count them yet. But important things are the columns. Therefore we should look the columns and check what are they.

In [6]:
```
terror.shape
```

Out[6]:
```
(181691, 135)
```

In [7]:
```
terror.rename(columns={'iyear':'Year','imonth':'Month','iday':'Day','country_txt':'Country','provstate':'state',
                       'region_txt':'Region','attacktype1_txt':'AttackType','target1':'Target','nkill':'Killed',
                       'nwound':'Wounded','summary':'Summary','gname':'Group','targtype1_txt':'Target_type',
                       'weaptype1_txt':'Weapon_type','motive':'Motive'},inplace=True)
```

In [8]:
```
# I'm just take important data in whole dataset those I'm using further processing.

terror=terror[['Year','Month','Day','Country','state','Region','city','latitude','longitude','AttackType','Killed',
               'Wounded','Target','Summary','Group','Target_type','Weapon_type','Motive']]
```

In [9]:
```
# Checking the null velues in data

terror.isnull().sum()
```

Out[9]:
```
Year               0
Month              0
Day                0
Country            0
state            421
Region             0
city             434
latitude        4556
longitude       4557
AttackType         0
Killed         10313
Wounded        16311
Target           636
Summary        66129
Group              0
Target_type        0
Weapon_type        0
Motive        131130
dtype: int64
```

In [10]:
```
terror.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 181691 entries, 0 to 181690
Data columns (total 18 columns):
 #   Column       Non-Null Count   Dtype  
---  ------       --------------   -----  
 0   Year         181691 non-null  int64  
 1   Month        181691 non-null  int64  
 2   Day          181691 non-null  int64  
 3   Country      181691 non-null  object 
 4   state        181270 non-null  object 
 5   Region       181691 non-null  object 
 6   city         181257 non-null  object 
 7   latitude     177135 non-null  float64
 8   longitude    177134 non-null  float64
 9   AttackType   181691 non-null  object 
 10  Killed       171378 non-null  float64
 11  Wounded      165380 non-null  float64
 12  Target       181055 non-null  object 
 13  Summary      115562 non-null  object 
 14  Group        181691 non-null  object 
 15  Target_type  181691 non-null  object 
 16  Weapon_type  181691 non-null  object 
 17  Motive       50561 non-null   object 
dtypes: float64(4), int64(3), object(11)
memory usage: 25.0+ MB
```

Destructive Feature of data.....

In [11]:
```
print("Country with the most attacks:",terror['Country'].value_counts().idxmax())
print("City with the most attacks:",terror['city'].value_counts().index[1])        #as first entry is 'unknown'
print("Region with the most attacks:",terror['Region'].value_counts().idxmax())
print("Year with the most attacks:",terror['Year'].value_counts().idxmax())
print("Month with the most attacks:",terror['Month'].value_counts().idxmax())
print("Group with the most attacks:",terror['Group'].value_counts().index[1])
print("Most Attack Types:",terror['AttackType'].value_counts().idxmax())
```

```
Country with the most attacks: Iraq
City with the most attacks: Baghdad
Region with the most attacks: Middle East & North Africa
Year with the most attacks: 2014
Month with the most attacks: 5
Group with the most attacks: Taliban
Most Attack Types: Bombing/Explosion
```

In [12]:
```
from wordcloud import WordCloud
from scipy import signal
cities = terror.state.dropna(False)
plt.subplots(figsize=(10,10))
wordcloud = WordCloud(background_color = 'white',
                      width = 512,
                      height = 384).generate(' '.join(cities))
plt.axis('off')
plt.imshow(wordcloud)
plt.show()
```

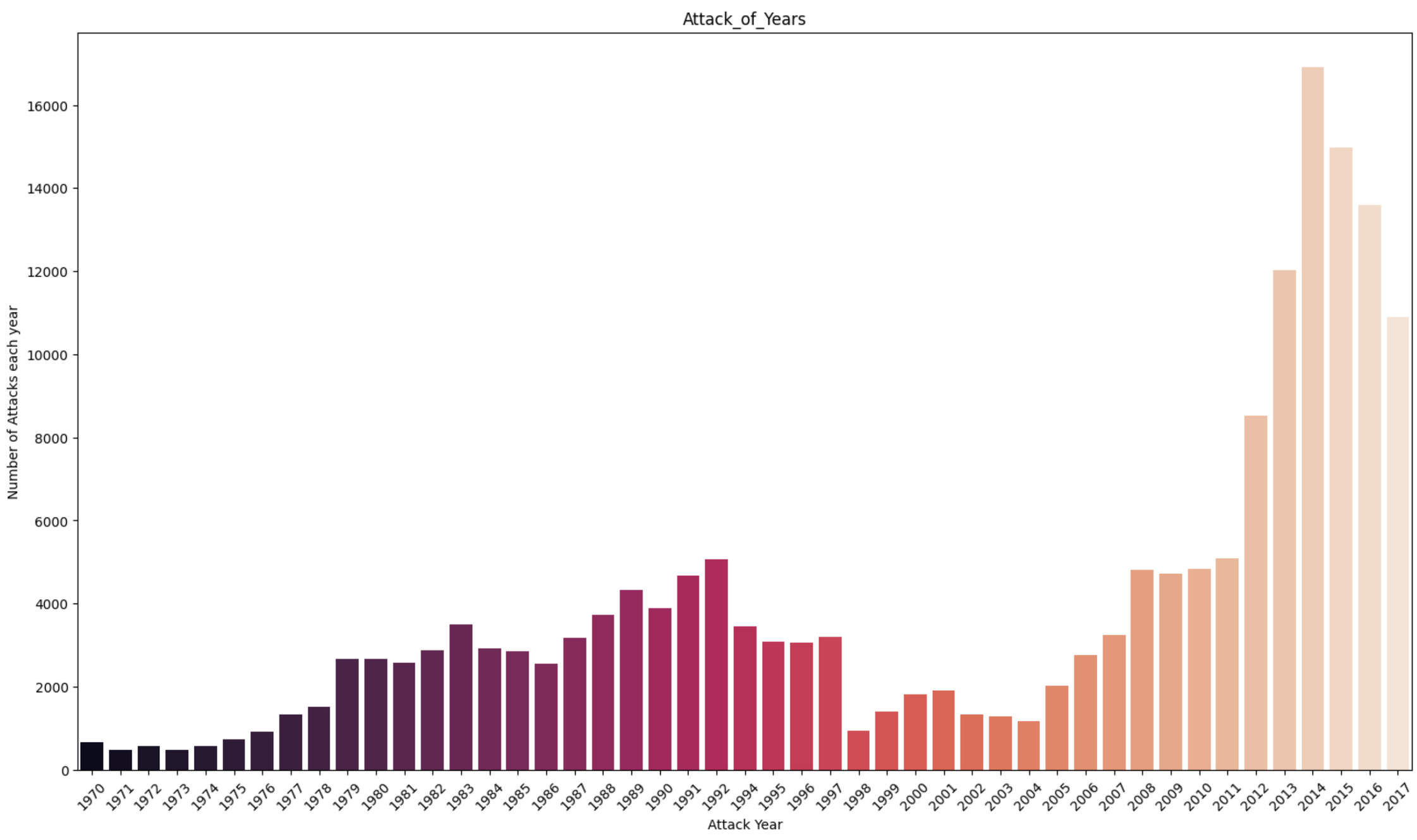
In [13]:
```
terror['Year'].value_counts(dropna = False).sort_index()
```

Out[13]:
```
1970     651
1971     471
1972     568
1973     473
1974     581
1975     740
1976     923
1977    1319
1978    1526
1979    2662
1980    2662
1981    2586
1982    2544
1983    2870
1984    3495
1985    2915
1986    2860
1987    3183
1988    3721
1989    4324
1990    3887
1991    4683
1992    5071
1994    3456
1995    3081
1996    3058
1997    3197
1998     934
1999    1395
2000    1814
2001    1906
2002    1333
2003    1278
2004    1166
2005    2017
2006    2758
2007    3242
2008    4805
2009    4721
2010    4826
2011    5076
2012    8522
2013   12036
2014   16903
2015   14965
2016   13587
2017   10900
Name: Year, dtype: int64
```

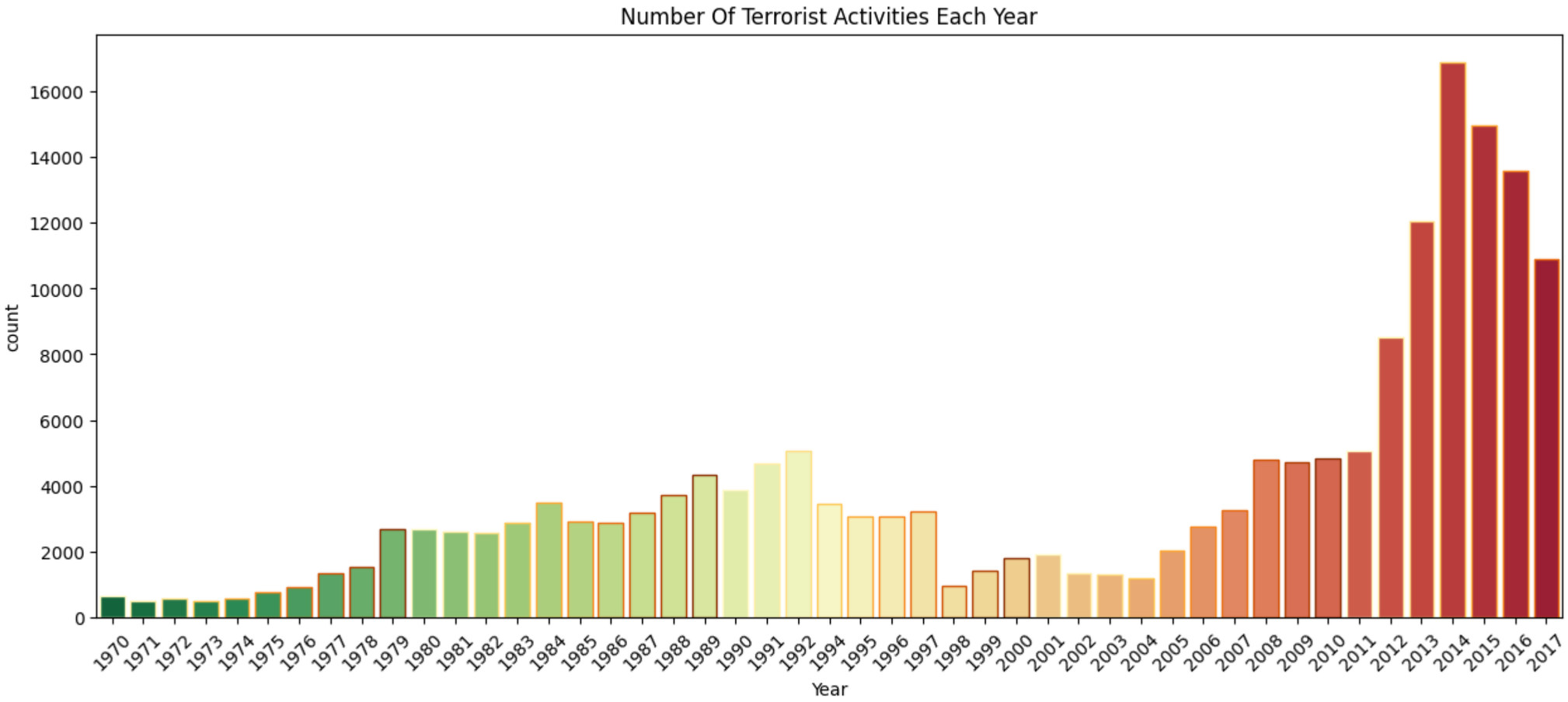
DATA VISUALISATION

Number of Terrorist Activities each Year

In [14]:
```
x_year = terror['Year'].unique()
y_count_years = terror['Year'].value_counts(dropna = False).sort_index()
plt.figure(figsize = (18,10))
sns.barplot(x = x_year,
            y = y_count_years,
            palette = 'rocket')
plt.xticks(rotation = 45)
plt.xlabel('Attack Year')
plt.ylabel('Number of Attacks each year')
plt.title('Attack_of_Years')
plt.show()
```

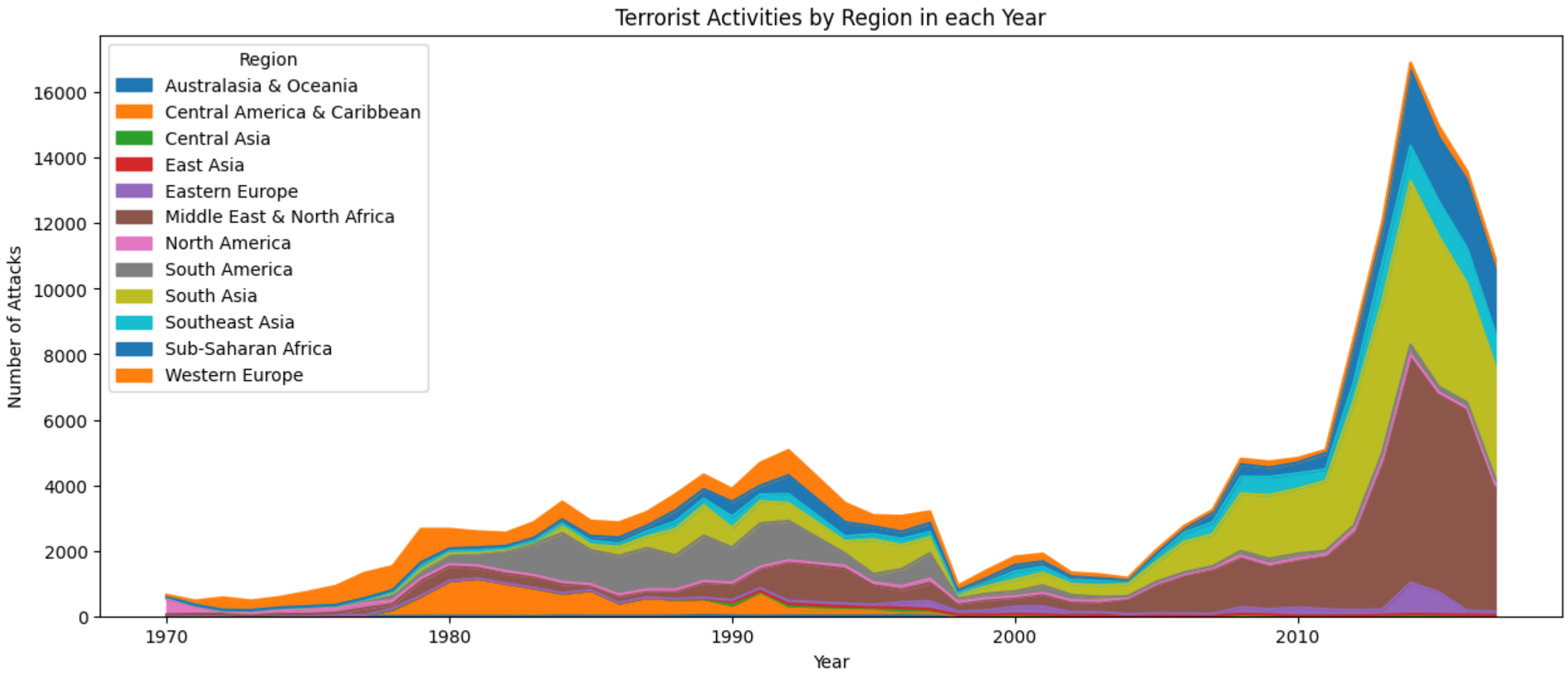


In [15]:
```
plt.subplots(figsize=(15, 6))
sns.countplot(data=terror, x='Year', palette='RdYlGn_r', edgecolor=sns.color_palette("YlOrBr", 10))
plt.xticks(rotation=45)
plt.title('Number Of Terrorist Activities Each Year')
plt.show()
```



Terrorist Activities by Region in each Year through Area Plot

In [16]:
```
pd.crosstab(terror.Year, terror.Region).plot(kind='area',figsize=(15,6))
plt.title('Terrorist Activities by Region in each Year')
plt.ylabel('Number of Attacks')
plt.show()
```



In [17]:
```
terror['Wounded'] = terror['Wounded'].fillna(0).astype(int)
terror['Killed'] = terror['Killed'].fillna(0).astype(int)
terror['casualities'] = terror['Killed'] + terror['Wounded']
```

Values are sorted by the top 40 worst terror attacks as to keep the heatmap simple and easy to visualize

In [18]:
```
terror1 = terror.sort_values(by='casualities',ascending=False)[:40]
```

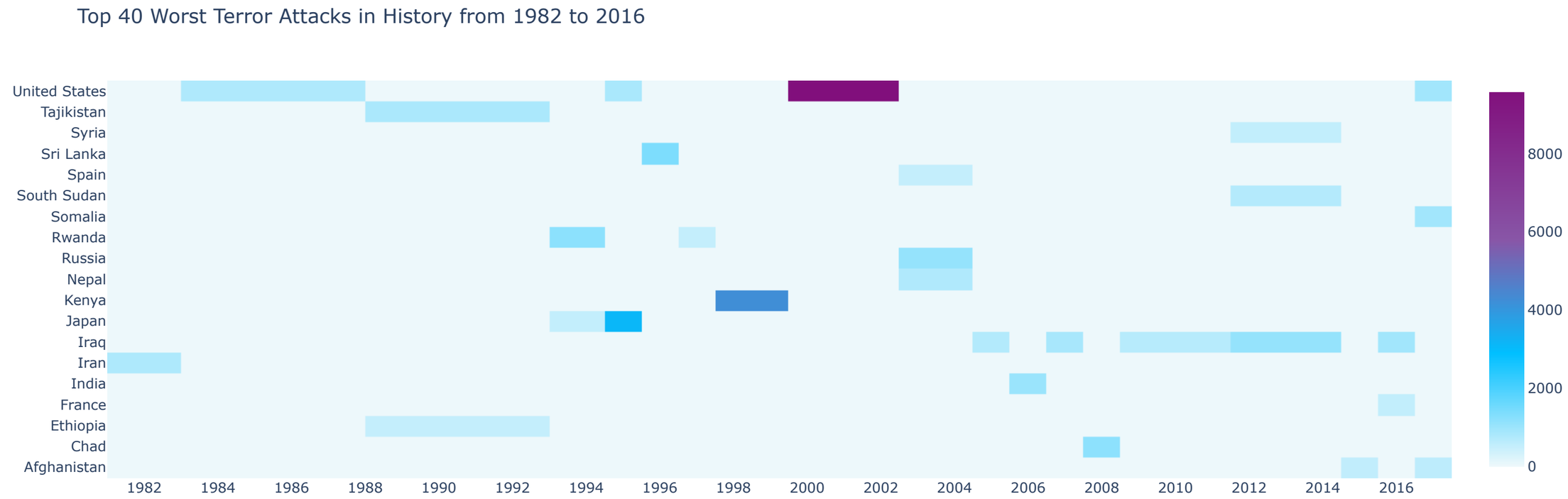
In [19]:
```
heat=terror1.pivot_table(index='Country',columns='Year',values='casualities')
heat.fillna(0,inplace=True)
```

In [20]:
```
heat.head()
```

Out[20]:

Year	1982	1984	1992	1994	1995	1996	1997	1998	2001	2004	2005	2006	2007	2008	2009	2014	2015	2016	2017
Country																			
Afghanistan	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	536.0	0.0	584.0
Chad	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1161.0	0.0	0.0	0.0	0.0	0.0
Ethiopia	0.0	0.0	500.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
France	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	520.0	0.0
India	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1005.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

In [21]:
```
import plotly.offline as py
py.init_notebook_mode(connected=True)
import plotly.graph_objs as go
colorscale = [[0, '#edf8fb'], [.3, '#00BFFF'],  [.6, '#8856a7'],  [1, '#810f7c']]
heatmap = go.Heatmap(z=heat.values, x=heat.columns, y=heat.index, colorscale=colorscale)
data = [heatmap]
layout = go.Layout(
    title='Top 40 Worst Terror Attacks in History from 1982 to 2016',
    xaxis = dict(ticks='', nticks=20),
    yaxis = dict(ticks='')
)
fig = go.Figure(data=data, layout=layout)
py.iplot(fig, filename='heatmap',show_link=False)
```

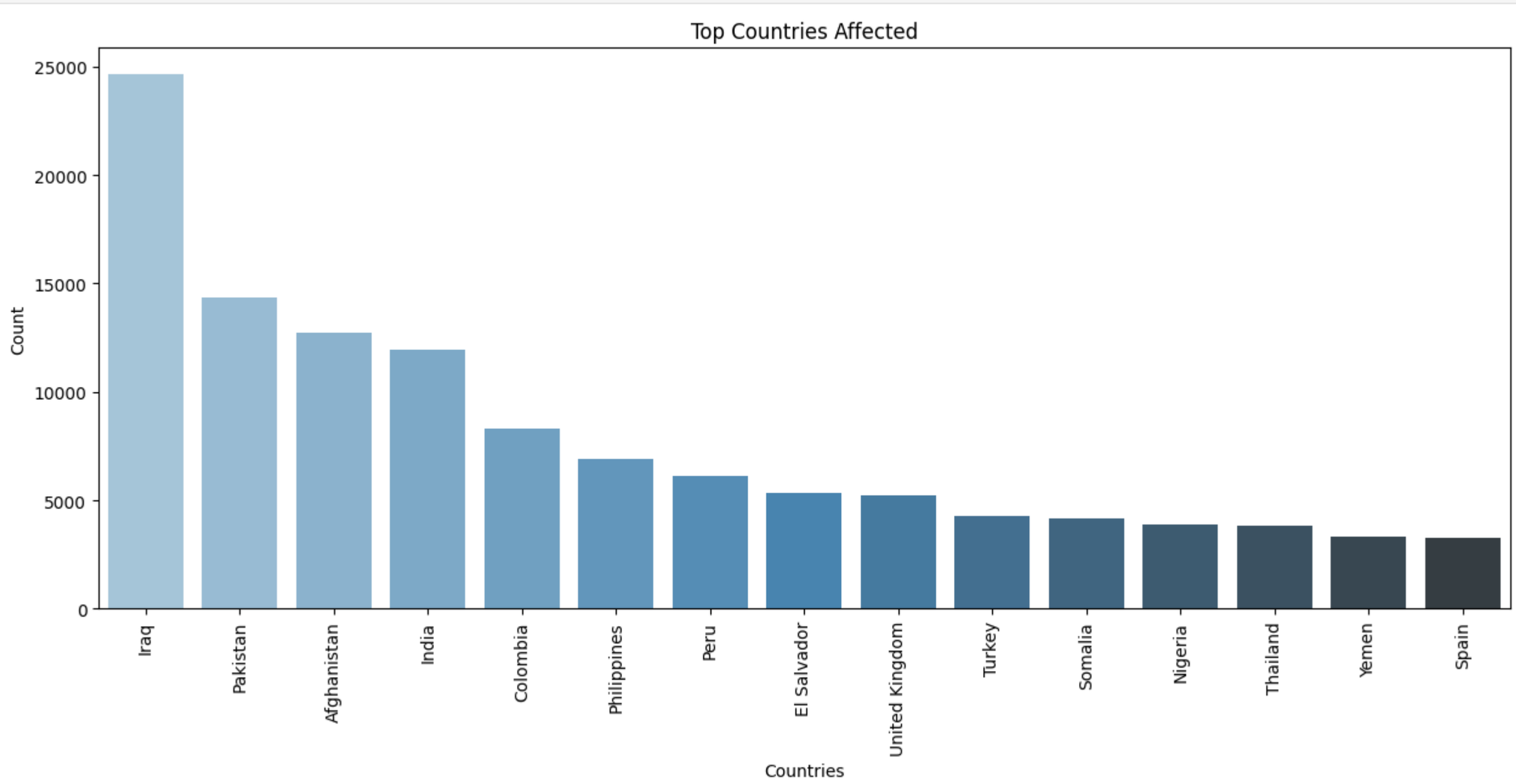


In [22]:
```
terror.Country.value_counts()[:15]
```

Out[22]:
```
Iraq              24636
Pakistan          14368
Afghanistan       12731
India             11960
Colombia           8306
Philippines        6908
Peru               6096
El Salvador        5320
United Kingdom     5235
Turkey             4292
Somalia            4142
Nigeria            3907
Thailand           3849
Yemen              3347
Spain              3249
Name: Country, dtype: int64
```

Top Countries affected by Terror Attacks

In [23]:
```
plt.subplots(figsize=(15, 6))
sns.barplot(
    x=terror['Country'].value_counts()[:15].index.tolist(),
    y=terror['Country'].value_counts()[:15].values.tolist(),
    palette='Blues_d'
)
plt.title('Top Countries Affected')
plt.xlabel('Countries')
plt.ylabel('Count')
plt.xticks(rotation=90)
plt.show()
```



ANALYSIS ON CUSTOMIZED DATA

Terrorist Attacks of a Particular year and their Locations

Let's look at the terrorist acts in the world over a certain year.

In [24]:
```
import folium
from folium.plugins import MarkerCluster
filterYear = terror['Year'] == 1970
```

In [25]:
```
filterData = terror[filterYear]                          # filter data

# filterData.info()
reqFilterData = filterData.loc[:,'city':'longitude']    # We are getting the required fields
reqFilterData = reqFilterData.dropna()                  # drop NaN values in latitude and longitude
reqFilterDataList = reqFilterData.values.tolist()
# reqFilterDataList
```

In [26]:
```
map = folium.Map(location = [0, 30], tiles='CartoDB positron', zoom_start=2)
```

```
# clustered marker
markerCluster = folium.plugins.MarkerCluster().add_to(map)
for point in range(0, len(reqFilterDataList)):
    folium.Marker(location=[reqFilterDataList[point][1],reqFilterDataList[point][2]],
                  popup = reqFilterDataList[point][0]).add_to(markerCluster)
map
```

Out[26]:

84% of the terrorist attacks in 1970 were carried out on the American continent. In 1970, the Middle East and North Africa, currently the center of wars and terrorist attacks, faced only one terrorist attack.

Now let us check out which terrorist organizations have carried out their operations in each country. A value count would give us the terrorist organizations that have carried out the most attacks. we have indexed from 1 as to negate the value of 'Unknown'

In [27]:
```
terror.Group.value_counts()[1:15]
```

Out[27]:
```
Taliban                                        7478
Islamic State of Iraq and the Levant (ISIL)    5613
Shining Path (SL)                              4555
Farabundo Marti National Liberation Front (FMLN)    3351
Al-Shabaab                                     3288
New People's Army (NPA)                        2772
Irish Republican Army (IRA)                    2671
Revolutionary Armed Forces of Colombia (FARC)    2487
Boko Haram                                     2418
Kurdistan Workers' Party (PKK)                 2310
Basque Fatherland and Freedom (ETA)            2024
Communist Party of India - Maoist (CPI-Maoist)    1878
Maoists                                        1630
Liberation Tigers of Tamil Eelam (LTTE)        1606
Name: Group, dtype: int64
```

In [28]:
```
test = terror[terror.Group.isin(['Shining Path (SL)','Taliban','Islamic State of Iraq and the Levant (ISIL)'])]
```

In [29]:
```
test.Country.unique()
```

Out[29]:
```
array(['Peru', 'Bolivia', 'Colombia', 'Argentina', 'Brazil', 'Mexico',
       'Afghanistan', 'Pakistan', 'Syria', 'Iraq', 'Turkey', 'Tunisia',
       'Lebanon', 'Turkmenistan', 'Israel', 'Belgium', 'Egypt', 'Libya',
       'Saudi Arabia', 'West Bank and Gaza Strip', 'France', 'Bahrain',
       'Jordan', 'Somalia', 'Germany', 'Yemen', 'Philippines', 'Malaysia',
       'Indonesia', 'Russia', 'Georgia', 'United Kingdom', 'Iran',
       'Australia'], dtype=object)
```

In [30]:
```
terror_df_group = terror.dropna(subset=['latitude','longitude'])
terror_df_group = terror_df_group.drop_duplicates(subset=['Country','Group'])
terrorist_groups = terror.Group.value_counts()[1:8].index.tolist()
terror_df_group = terror_df_group.loc[terror_df_group.Group.isin(terrorist_groups)]
print(terror_df_group.Group.unique())
```

```
["New People's Army (NPA)" 'Irish Republican Army (IRA)'
 'Shining Path (SL)' 'Farabundo Marti National Liberation Front (FMLN)'
 'Taliban' 'Al-Shabaab' 'Islamic State of Iraq and the Levant (ISIL)']
```

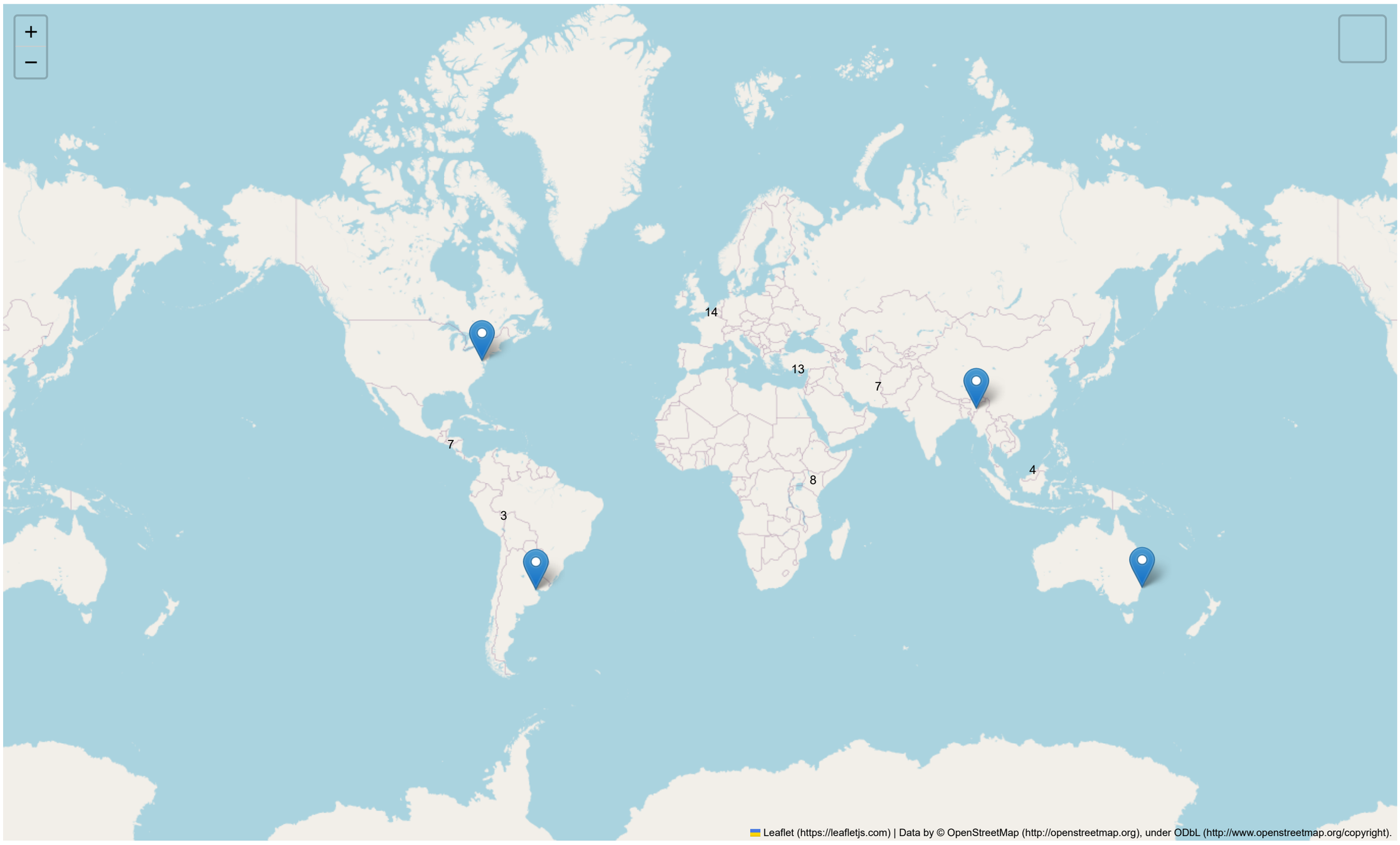
In [31]:
```
map = folium.Map(location=[20, 0], tiles="CartoDB positron", zoom_start=2)
markerCluster = folium.plugins.MarkerCluster().add_to(map)
for i in range(0,len(terror_df_group)):
    folium.Marker([terror_df_group.iloc[i]['latitude'],terror_df_group.iloc[i]['longitude']],
                  popup='Group:{}<br>Country:{}'.format(terror_df_group.iloc[i]['Group'],
                  terror_df_group.iloc[i]['Country'])).add_to(map)
map
```

Out[31]:

The Above map looks untidy even though it can be zoomed in to view the Country in question. Hence in the next chart, I have used Folium's Marker Cluster to cluster these icons. This makes it visually pleasing and highly interactive.

In [32]:
```
# Create a map centered at [20, 0] with default OpenStreetMap tiles and zoom level
m1 = folium.Map(location=[20, 0], zoom_start=2)

# Create a marker cluster layer
marker_cluster = MarkerCluster(
    name='clustered icons',
    overlay=True,
    control=False,
    icon_create_function=None
)

# Loop through the rows in terror_df_group and add markers to the map
for i in range(0, len(terror_df_group)):
    marker = folium.Marker([terror_df_group.iloc[i]['latitude'], terror_df_group.iloc[i]['longitude']])
    popup = 'Group: {}<br>Country: {}'.format(terror_df_group.iloc[i]['Group'], terror_df_group.iloc[i]['Country'])
    folium.Popup(popup).add_to(marker)
    marker_cluster.add_child(marker)

# Add the marker cluster layer to the map
marker_cluster.add_to(m1)

# Add different tile layers and a layer control to switch between them
folium.TileLayer('openstreetmap').add_to(m1)
folium.TileLayer('cartodbdark_matter').add_to(m1)
folium.TileLayer('stamentoner').add_to(m1)

# Add a layer control to toggle between different tile layers
folium.LayerControl().add_to(m1)

# Display the map
m1
```

Out[32]:



In [33]:
```
terror.head()
```

Out[33]:

	Year	Month	Day	Country	state	Region	city	latitude	longitude	AttackType	Killed	Wounded	Target	Summary	Group	Target_type	Weapon_type	Motive	casualities
0	1970	7	2	Dominican Republic	NaN	Central America & Caribbean	Santo Domingo	18.456792	-69.951164	Assassination	1	0	Julio Guzman	NaN	MANO-D	Private Citizens & Property	Unknown	NaN	1
1	1970	0	0	Mexico	Federal	North America	Mexico city	19.371887	-99.086624	Hostage Taking (Kidnapping)	0	0	Nadine Chaval, daughter	NaN	23rd of September Communist League	Government (Diplomatic)	Unknown	NaN	0
2	1970	1	0	Philippines	Tarlac	Southeast Asia	Unknown	15.478598	120.599741	Assassination	1	0	Employee	NaN	Unknown	Journalists & Media	Unknown	NaN	1
3	1970	1	0	Greece	Attica	Western Europe	Athens	37.997490	23.762728	Bombing/Explosion	0	0	U.S. Embassy	NaN	Unknown	Government (Diplomatic)	Explosives	NaN	0
4	1970	1	0	Japan	Fukouka	East Asia	Fukouka	33.580412	130.396361	Facility/Infrastructure Attack	0	0	U.S. Consulate	NaN	Unknown	Government (Diplomatic)	Incendiary	NaN	0

In [34]:
```
# Total Number of people killed in terror attack

killData = terror.loc[:,'Killed']
print('Number of people killed by terror attack:', int(sum(killData.dropna())))# drop the NaN values
```

```
Number of people killed by terror attack: 411868
```

In [35]:
```
# Let's look at what types of attacks these deaths were made of.

attackData = terror.loc[:,'AttackType']
# attackData
typeKillData = pd.concat([attackData, killData], axis=1)
```

In [36]:
```
typeKillData.head()
```

Out[36]:

	AttackType	Killed
0	Assassination	1
1	Hostage Taking (Kidnapping)	0
2	Assassination	1
3	Bombing/Explosion	0
4	Facility/Infrastructure Attack	0

In [37]:
```
typeKillFormatData = typeKillData.pivot_table(columns='AttackType', values='Killed', aggfunc='sum')
typeKillFormatData
```

Out[37]:

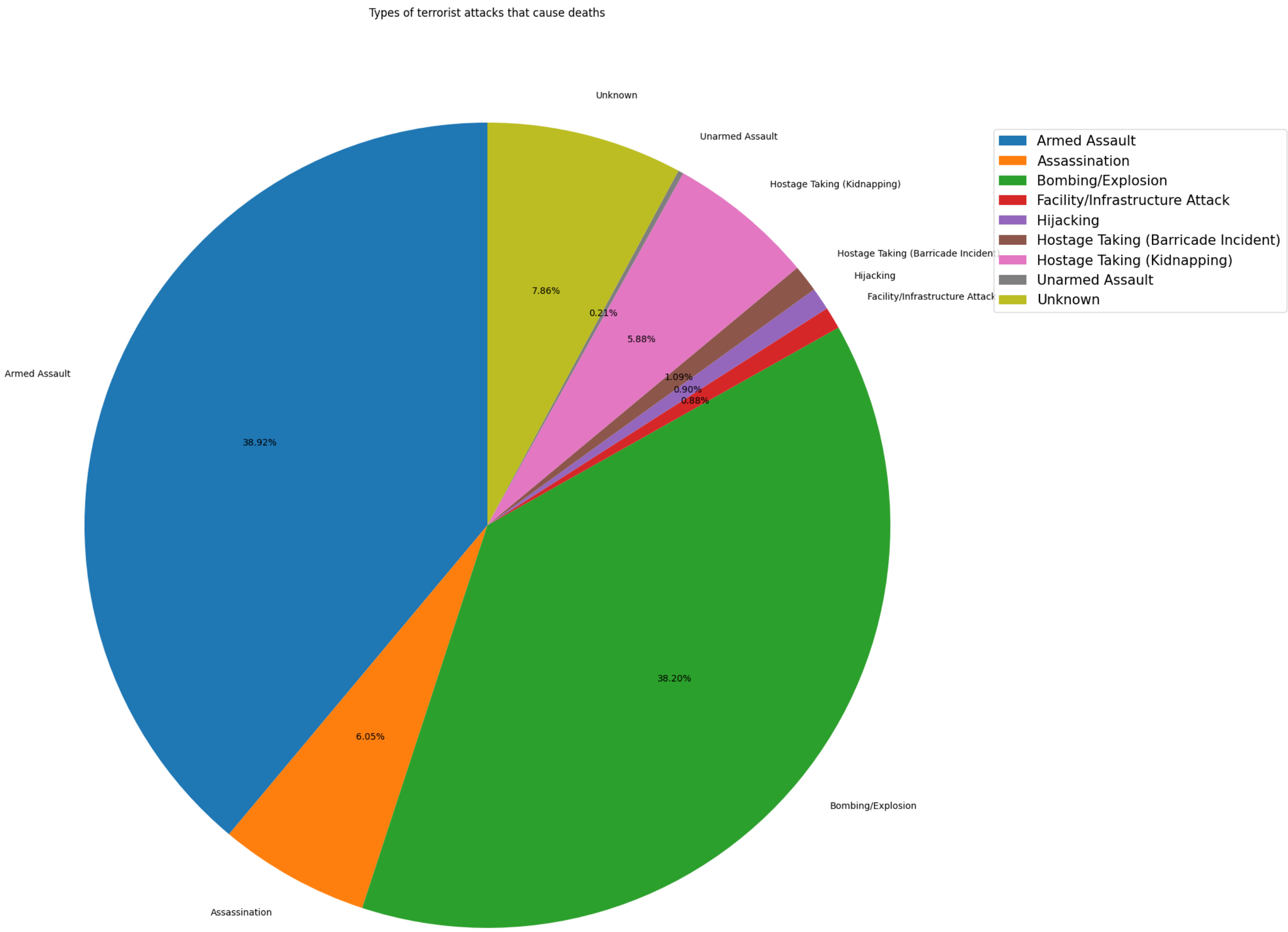
AttackType	Armed Assault	Assassination	Bombing/Explosion	Facility/Infrastructure Attack	Hijacking	Hostage Taking (Barricade Incident)	Hostage Taking (Kidnapping)	Unarmed Assault	Unknown
Killed	160297	24920	157321	3642	3718	4478	24231	880	32381

In [38]:
```
typeKillFormatData.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1 entries, Killed to Killed
Data columns (total 9 columns):
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0   Armed Assault                       1 non-null      int32
 1   Assassination                       1 non-null      int32
 2   Bombing/Explosion                   1 non-null      int32
 3   Facility/Infrastructure Attack      1 non-null      int32
 4   Hijacking                           1 non-null      int32
 5   Hostage Taking (Barricade Incident)  1 non-null      int32
 6   Hostage Taking (Kidnapping)         1 non-null      int32
 7   Unarmed Assault                     1 non-null      int32
 8   Unknown                             1 non-null      int32
dtypes: int32(9)
memory usage: 152.0+ bytes
```

In [39]:
```
labels = typeKillFormatData.columns.tolist()  # Convert columns to list
transposed = typeKillFormatData.T  # Transpose the data
values = transposed.values.flatten().tolist()  # Flatten the 2D array to a 1D list

fig, ax = plt.subplots(figsize=(20, 20), subplot_kw=dict(aspect="equal"))
plt.pie(values, labels=labels, startangle=90, autopct='%.2f%%')
```

```
plt.title('Types of terrorist attacks that cause deaths')
plt.legend(labels, loc='upper right', bbox_to_anchor=(1.3, 0.9), fontsize=15)  # Location of the legend
plt.show()
```



Armed assault and bombing/explosion are seen to be the cause of 77% of the deaths in these attacks. This rate is why these attacks are used so many times in terrorist actions. This is how dangerous weapons and explosives are to the world.

In [40]:
```
#Number of Killed in Terrorist Attacks by Countries
countryData = terror.loc[:,'Country']
# countyData
countryKillData = pd.concat([countryData, killData], axis=1)
```

In [41]:
```
countryKillFormatData = countryKillData.pivot_table(columns='Country', values='Killed', aggfunc='sum')
countryKillFormatData
```

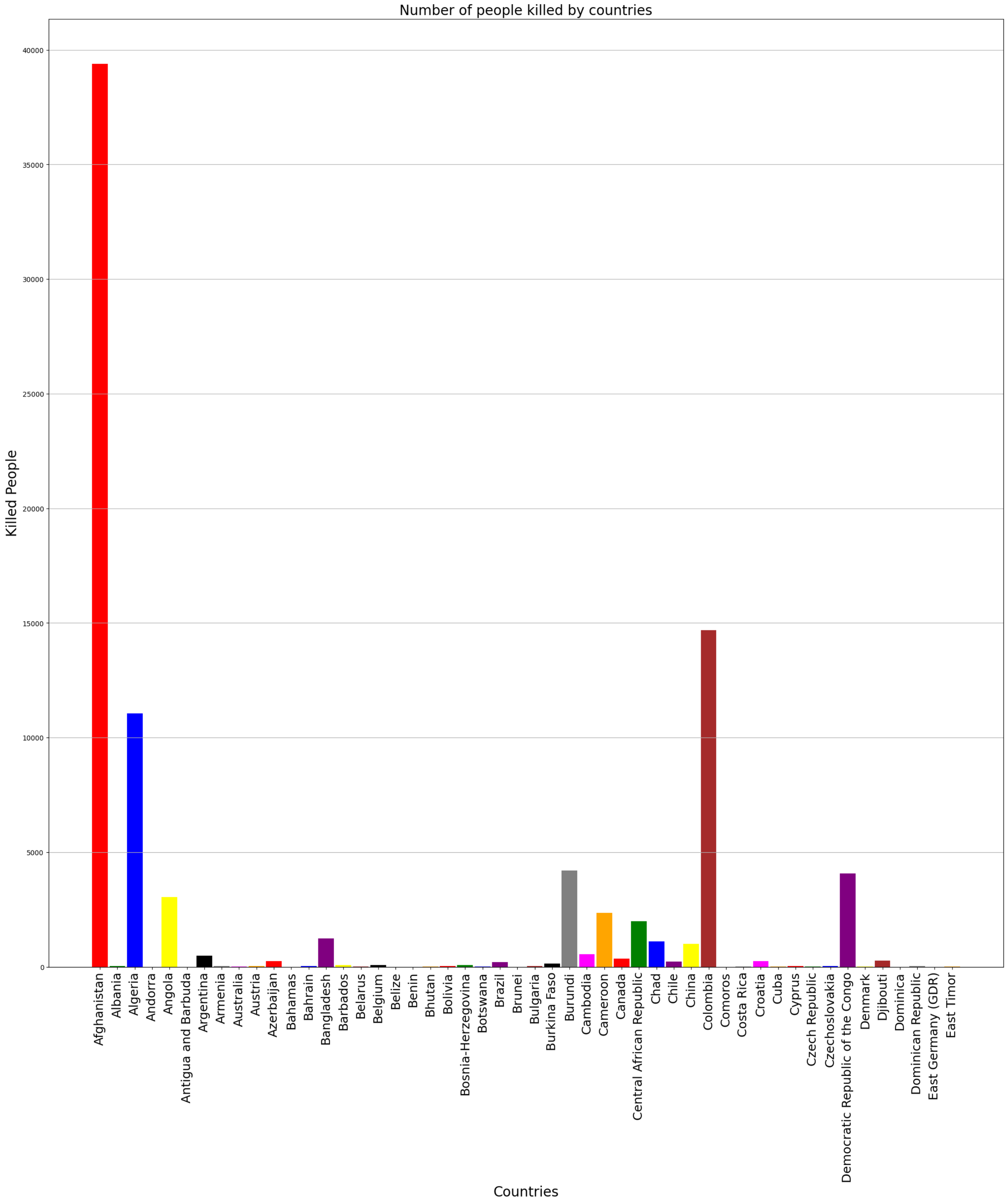
Out[41]:

Country	Afghanistan	Albania	Algeria	Andorra	Angola	Antigua and Barbuda	Argentina	Armenia	Australia	Austria	...	Vietnam	Wallis and Futuna	West Bank and Gaza Strip	West Germany (FRG)	Western Sahara	Yemen	Yugoslavia	Zaire	Zambia	Zimbabwe
Killed	39384	42	11066	0	3043	0	490	37	23	30	...	1	0	1500	97	1	8776	119	324	70	154

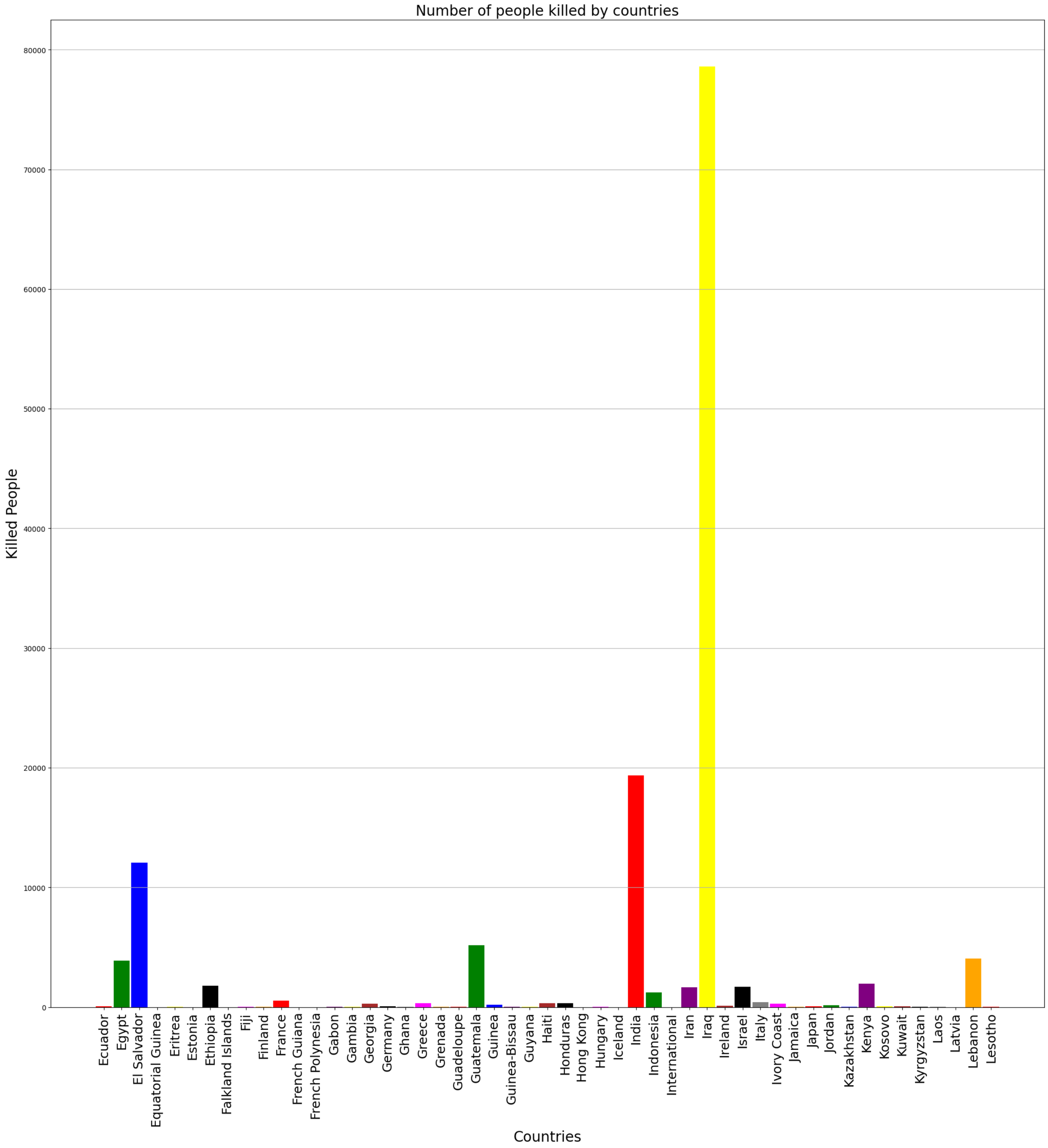
1 rows × 205 columns

In [42]:
```
fig_size = plt.rcParams["figure.figsize"]
fig_size[0]=25
fig_size[1]=25
plt.rcParams["figure.figsize"] = fig_size
```

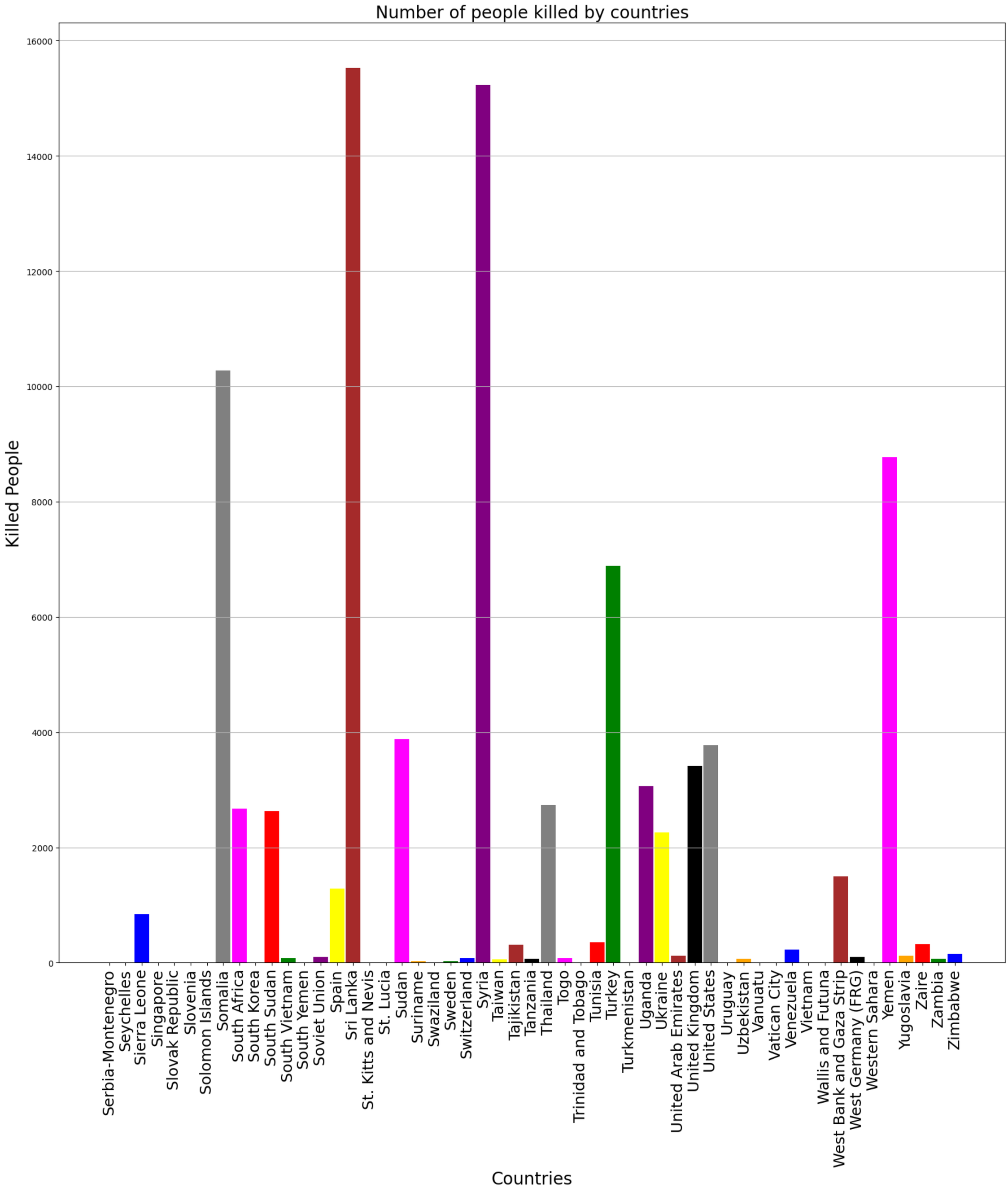
In [43]:
```
labels = countryKillFormatData.columns.tolist()
labels = labels[:50] #50 bar provides nice view
index = np.arange(len(labels))
transpoze = countryKillFormatData.T
values = transpoze.values.tolist()
values = values[:50]
values = [int(i[0]) for i in values] # convert float to int
colors = ['red', 'green', 'blue', 'purple', 'yellow', 'brown', 'black', 'gray', 'magenta', 'orange'] # color list for bar chart bar color
fig, ax = plt.subplots(1, 1)
ax.yaxis.grid(True)
fig_size = plt.rcParams["figure.figsize"]
fig_size[0]=25
fig_size[1]=25
plt.rcParams["figure.figsize"] = fig_size
plt.bar(index, values, color = colors, width = 0.9)
plt.ylabel('Killed People', fontsize=20)
plt.xlabel('Countries', fontsize = 20)
plt.xticks(index, labels, fontsize=18, rotation=90)
plt.title('Number of people killed by countries', fontsize = 20)
# print(fig_size)
plt.show()
```



In [44]:
```
labels = countryKillFormatData.columns.tolist()
labels = labels[50:101]
index = np.arange(len(labels))
transpoze = countryKillFormatData.T
values = transpoze.values.tolist()
values = values[50:101]
values = [int(i[0]) for i in values]
colors = ['red', 'green', 'blue', 'purple', 'yellow', 'brown', 'black', 'gray', 'magenta', 'orange']
fig, ax = plt.subplots(1, 1)
ax.yaxis.grid(True)
fig_size = plt.rcParams["figure.figsize"]
fig_size[0]=20
fig_size[1]=20
plt.rcParams["figure.figsize"] = fig_size
plt.bar(index, values, color = colors, width = 0.9)
plt.ylabel('Killed People', fontsize=20)
plt.xlabel('Countries', fontsize = 20)
plt.xticks(index, labels, fontsize=18, rotation=90)
plt.title('Number of people killed by countries', fontsize = 20)
plt.show()
```



In [45]:
```
labels = countryKillFormatData.columns.tolist()
labels = labels[152:206]
index = np.arange(len(labels))
transpoze = countryKillFormatData.T
values = transpoze.values.tolist()
values = values[152:206]
values = [int(i[0]) for i in values]
colors = ['red', 'green', 'blue', 'purple', 'yellow', 'brown', 'black', 'gray', 'magenta', 'orange']
fig, ax = plt.subplots(1, 1)
ax.yaxis.grid(True)
fig_size = plt.rcParams["figure.figsize"]
fig_size[0]=25
fig_size[1]=25
plt.rcParams["figure.figsize"] = fig_size
plt.bar(index, values, color = colors, width = 0.9)
plt.ylabel('Killed People', fontsize=20)
plt.xlabel('Countries', fontsize = 20)
plt.xticks(index, labels, fontsize=18, rotation=90)
plt.title('Number of people killed by countries', fontsize = 20)
plt.show()
```



Conclusion and Results :

Year with Maximum Attacks: 2014

Country with the Most Attacks: Iraq

City with the Most Attacks: Baghdad

Most Common Attack: Bombing/Explosion

Maximum Casualties in an Attack: Armed Assault

Most Wounded in an Attack: Bombing/Explosion

Most Common Targets in an Attack: Private Citizens and Property

Terrorist Group responsible for Maximum Attacks: Taliban

Terrorist acts in the Middle East and northern Africa have been seen to have fatal consequences. The Middle East and North Africa are seen to be the places of serious terrorist attacks. In addition, even though there is a perception that Muslims are supporters of terrorism, Muslims are the people who are most damaged by terrorist attacks. If you look at the graphics, it appears that Iraq, Afghanistan and Pakistan are the most damaged countries. All of these countries are Muslim countries.Thank you.

Thank you.