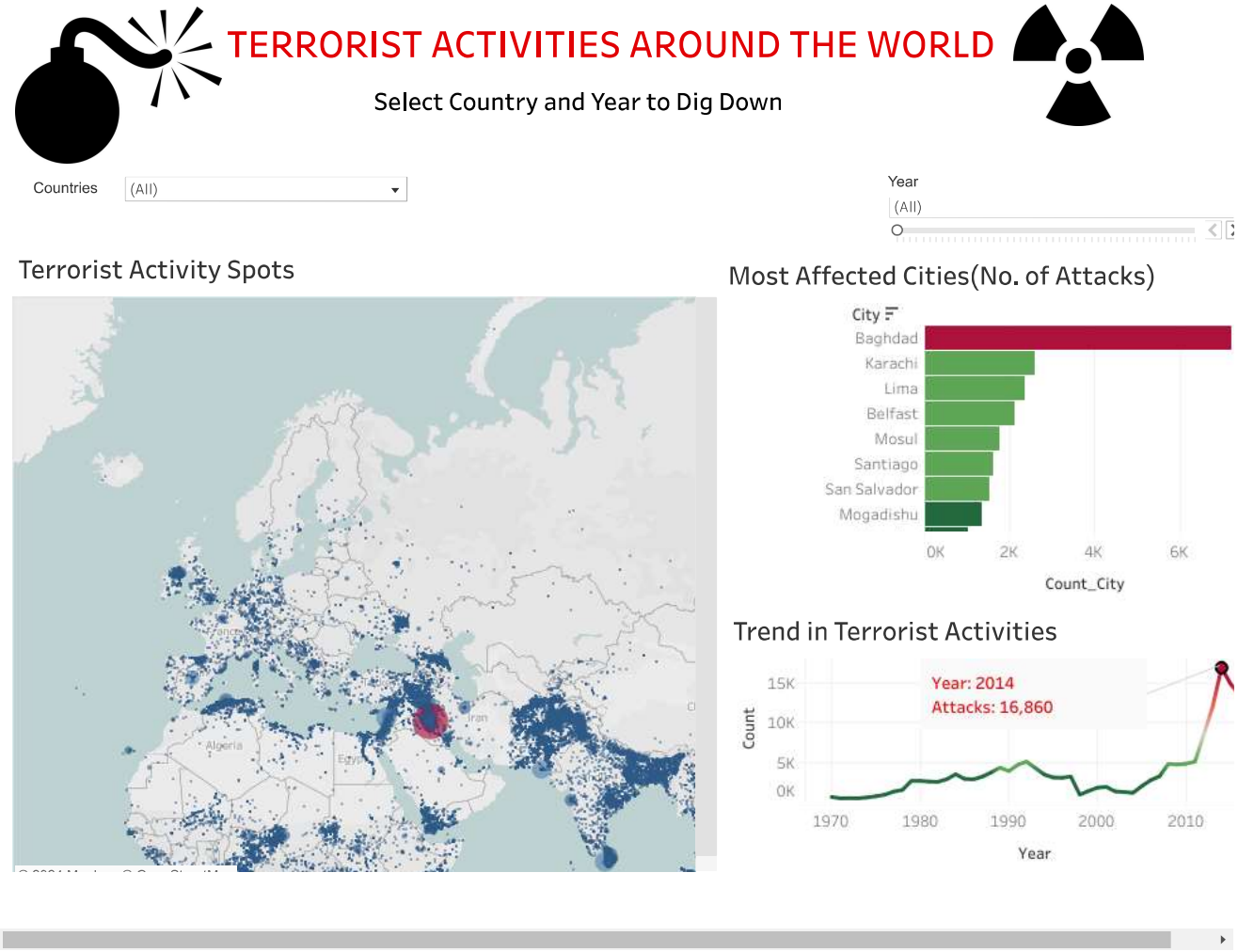
In [92]:
```
%%HTML
<div class='tableauPlaceholder' id='viz1512233423027' style='position: relative'>
<noscript><a href='#'><img alt='Dashboard 1 ' src='https:&#47;&#47;public.tableau.com&#47;static&#47;images&#47;RD&#47;RDGKRRHB7&
<object class='tableauViz'  style='display:none;'>
<param name='host_url' value='https%3A%2F%2Fpublic.tableau.com%2F' />
<param name='embed_code_version' value='3' />
<param name='path' value='shared&#47;RDGKRRHB7' />
<param name='toolbar' value='yes' />
<param name='static_image' value='https:&#47;&#47;public.tableau.com&#47;static&#47;images&#47;RD&#47;RDGKRRHB7&#47;1.png' />
<param name='animate_transition' value='yes' />
<param name='display_static_image' value='yes' />
<param name='display_spinner' value='yes' />
<param name='display_overlay' value='yes' />
<param name='display_count' value='yes' /></object></div>
<script type='text/javascript'>var divElement = document.getElementById('viz1512233423027');
var vizElement = divElement.getElementsByTagName('object')[0];
vizElement.style.width='1020px';vizElement.style.minHeight='687px';
vizElement.style.maxHeight='1587px';vizElement.style.height=(divElement.offsetWidth*0.75)+'px';
var scriptElement = document.createElement('script');
scriptElement.src = 'https://public.tableau.com/javascripts/api/viz_v1.js';
vizElement.parentNode.insertBefore(scriptElement, vizElement);                </script>
```



Getting Data ready

In [6]:
```
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
import seaborn as sns
import numpy as np
plt.style.use('fivethirtyeight')
import plotly.offline as py
py.init_notebook_mode(connected=True)
import plotly.graph_objs as go
import plotly.tools as tls
# from mpl_toolkits.basemap import Basemap
import folium
import folium.plugins
from matplotlib import animation,rc
import io
import base64
from IPython.display import HTML, display
import warnings
warnings.filterwarnings('ignore')
 #from scipy.misc import imread
import codecs
from subprocess import check_output
#print(check_output(["ls", "../input"]).decode("utf8"))
```

In [9]:
```
terror=pd.read_csv(r'C:\Users\Samdure\Downloads\globalterrorismdb_0617dist.csv',encoding='ISO-8859-1')
terror.rename(columns={'iyear':'Year','imonth':'Month','iday':'Day','country_txt':'Country','region_txt':'Region','attacktype1_tx
terror=terror[['Year','Month','Day','Country','Region','city','latitude','longitude','AttackType','Killed','Wounded','Target','Su
terror['casualities']=terror['Killed']+terror['Wounded']
terror.head(3)
```

Out[9]:

	Year	Month	Day	Country	Region	city	latitude	longitude	AttackType	Killed	Wounded	Target	Summary	Group	Target_type	Wea
0	1970	7	2	Dominican Republic	Central America & Caribbean	Santo Domingo	18.456792	-69.951164	Assassination	1.0	0.0	Julio Guzman	NaN	MANO-D	Private Citizens & Property	
1	1970	0	0	Mexico	North America	Mexico city	19.432608	-99.133207	Hostage Taking (Kidnapping)	0.0	0.0	Nadine Chaval, daughter	NaN	23rd of September Communist League	Government (Diplomatic)	
2	1970	1	0	Philippines	Southeast Asia	Unknown	15.478598	120.599741	Assassination	1.0	0.0	Employee	NaN	Unknown	Journalists & Media	

In [10]:
```
terror.isnull().sum()
```

Out[10]:
```
Year              0
Month             0
Day               0
Country           0
Region            0
city            447
latitude       4606
longitude      4606
AttackType        0
Killed         9682
Wounded       15325
Target          638
Summary       66138
Group             0
Target_type       0
Weapon_type       0
Motive       121764
casualities   15826
dtype: int64
```

In [11]:
```
terror.describe()
```

Out[11]:

	Year	Month	Day	latitude	longitude	Killed	Wounded	casualities
count	170350.000000	170350.000000	170350.000000	165744.000000	165744.000000	160668.000000	155025.000000	154524.000000
mean	2001.709997	6.474365	15.466845	23.399774	26.350909	2.387246	3.200239	5.312327
std	13.144146	3.392364	8.817929	18.844885	58.570068	11.327709	34.647365	40.798412
min	1970.000000	0.000000	0.000000	-53.154613	-176.176447	0.000000	0.000000	0.000000
25%	1990.000000	4.000000	8.000000	11.263580	2.396199	0.000000	0.000000	0.000000
50%	2007.000000	6.000000	15.000000	31.472680	43.130000	0.000000	0.000000	1.000000
75%	2014.000000	9.000000	23.000000	34.744167	68.451297	2.000000	2.000000	4.000000
max	2016.000000	12.000000	31.000000	74.633553	179.366667	1500.000000	7366.000000	8749.000000

In [12]:
```
terror.info()
```

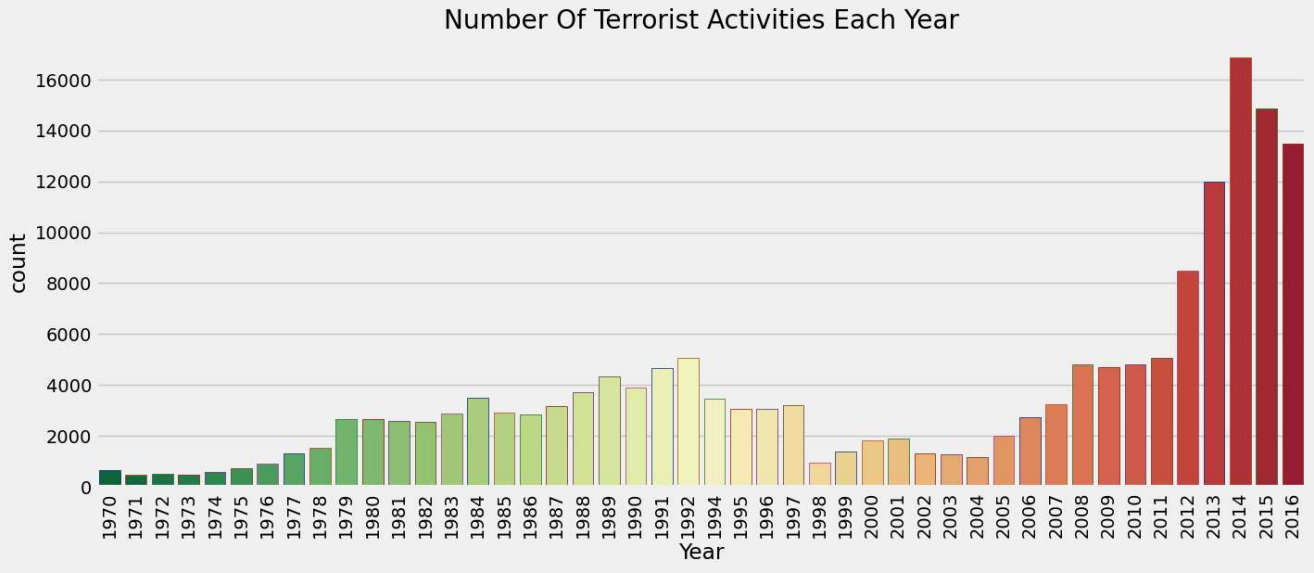
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 170350 entries, 0 to 170349
Data columns (total 18 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   Year         170350 non-null  int64
 1   Month        170350 non-null  int64
 2   Day          170350 non-null  int64
 3   Country      170350 non-null  object
 4   Region       170350 non-null  object
 5   city         169903 non-null  object
 6   latitude     165744 non-null  float64
 7   longitude    165744 non-null  float64
 8   AttackType   170350 non-null  object
 9   Killed       160668 non-null  float64
 10  Wounded      155025 non-null  float64
 11  Target       169712 non-null  object
 12  Summary      104212 non-null  object
 13  Group        170350 non-null  object
 14  Target_type  170350 non-null  object
 15  Weapon_type  170350 non-null  object
 16  Motive       48586 non-null   object
 17  casualities  154524 non-null  float64
dtypes: float64(5), int64(3), object(10)
memory usage: 23.4+ MB
```

EDA

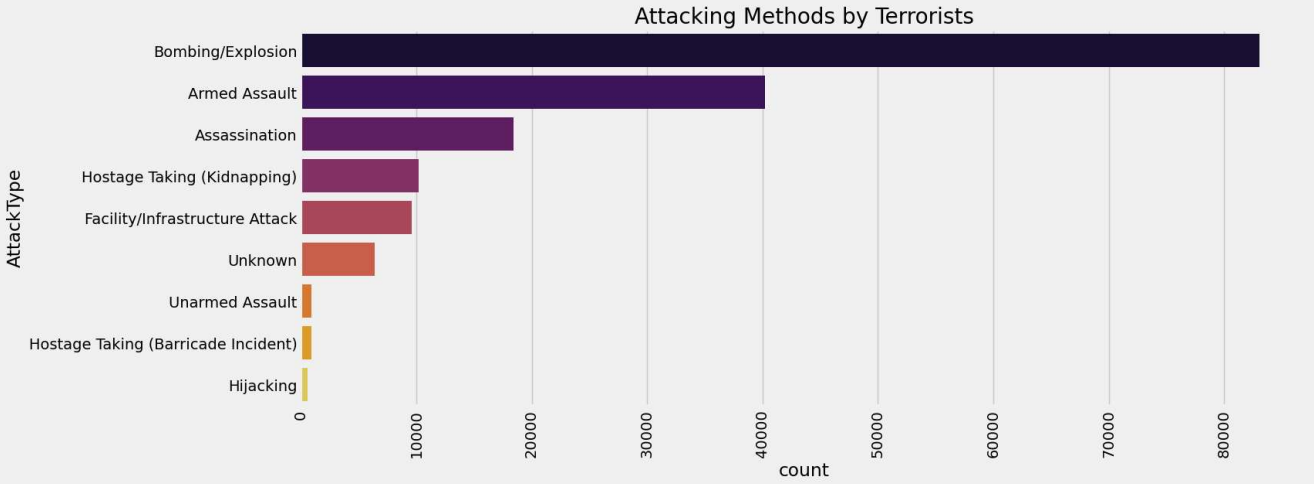
In [13]:
```
untry with Highest Terrorist Attacks:',terror['Country'].value_counts().index[0])
gions with Highest Terrorist Attacks:',terror['Region'].value_counts().index[0])
ximum people killed in an attack are:',terror['Killed'].max(),'that took place in',terror.loc[terror['Killed'].idxmax()].Country)
```

```
Country with Highest Terrorist Attacks: Iraq
Regions with Highest Terrorist Attacks: Middle East & North Africa
Maximum people killed in an attack are: 1500.0 that took place in Iraq
```

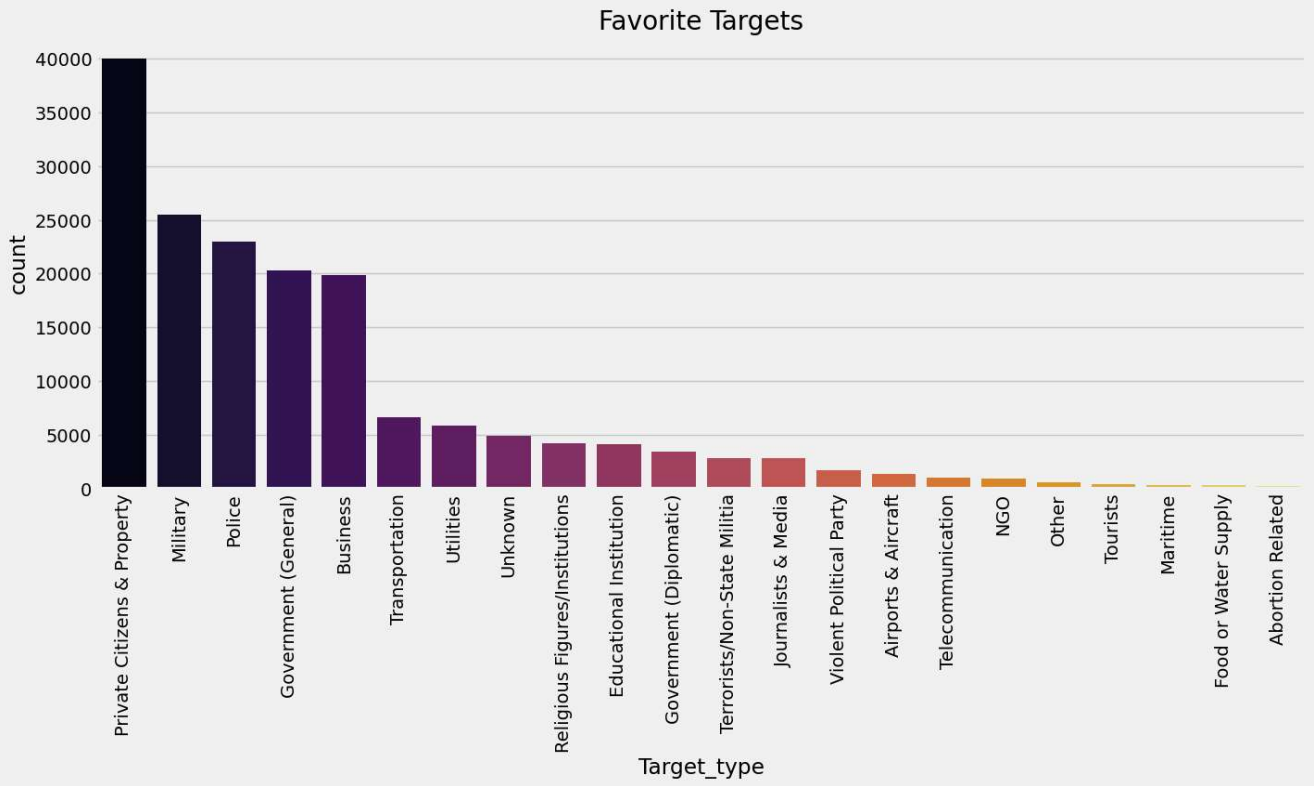
In [19]:
```
plt.subplots(figsize=(15,6))
sns.countplot(x='Year', data=terror, palette='RdYlGn_r', edgecolor=sns.color_palette('dark', 7))
plt.xticks(rotation=90)
plt.title('Number Of Terrorist Activities Each Year')
plt.show()
```



In [20]:
```
plt.subplots(figsize=(15,6))
sns.countplot(y='AttackType',data=terror,palette='inferno',order=terror['AttackType'].value_counts().index)
plt.xticks(rotation=90)
plt.title('Attacking Methods by Terrorists')
plt.show()
```



In [22]:
```
plt.subplots(figsize=(15, 6))
sns.countplot(x=terror['Target_type'], palette='inferno', order=terror['Target_type'].value_counts().index)
plt.xticks(rotation=90)
plt.title('Favorite Targets')
plt.show()
```

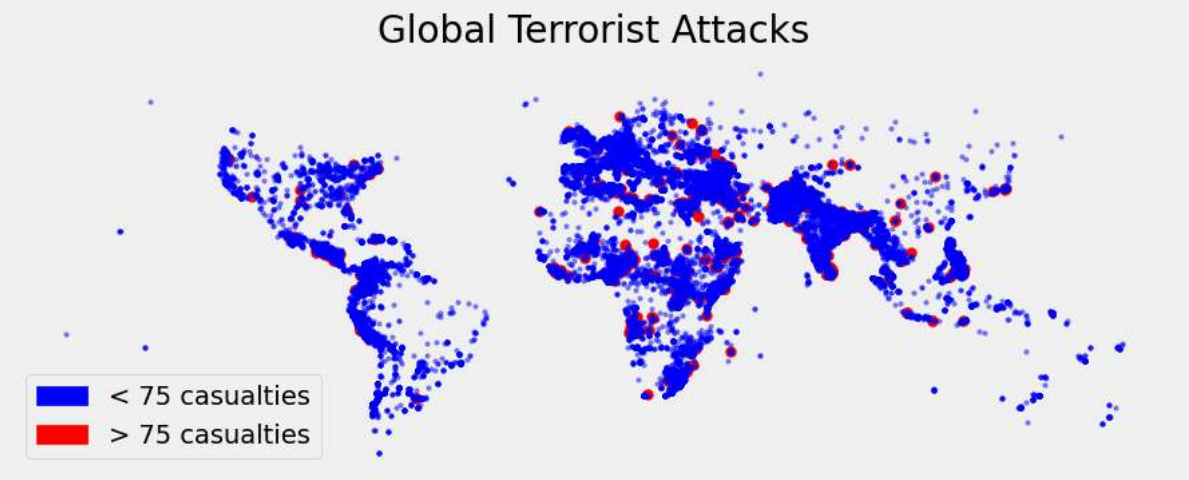


Global terror attacks

In [34]:
```
!pip install cartopy
```

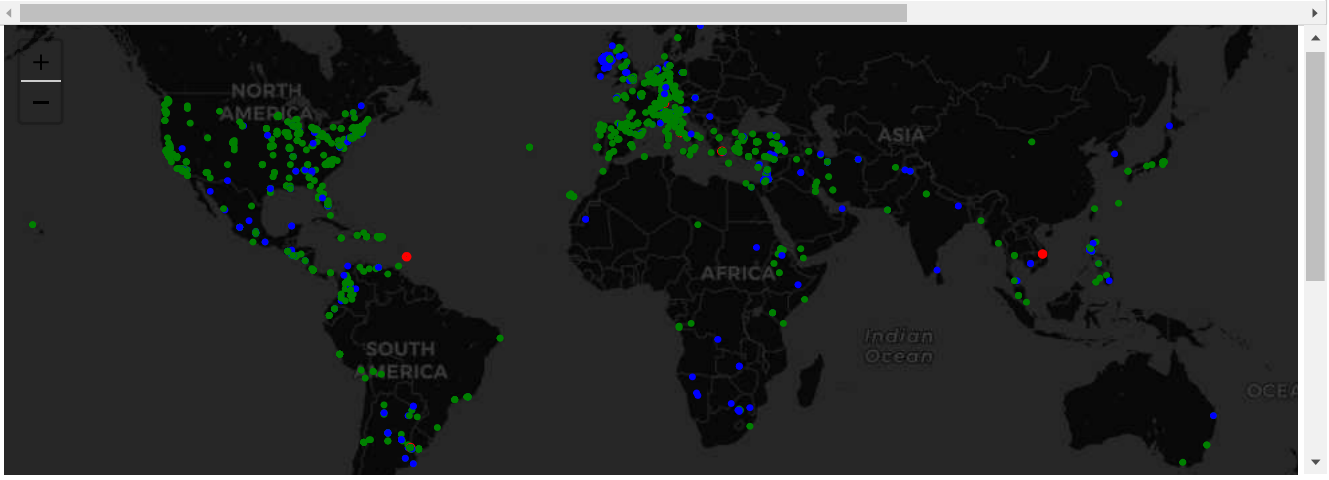
```
Requirement already satisfied: cartopy in c:\users\samdure\anaconda3\anaconda\lib\site-packages (0.22.0)
Requirement already satisfied: numpy>=1.21 in c:\users\samdure\appdata\roaming\python\python311\site-packages (from cartopy)
(1.26.3)
Requirement already satisfied: matplotlib>=3.4 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from cartopy) (3.7.2)
Requirement already satisfied: shapely>=1.7 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from cartopy) (2.0.3)
Requirement already satisfied: packaging>=20 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from cartopy) (23.1)
Requirement already satisfied: pyshp>=2.1 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from cartopy) (2.3.1)
Requirement already satisfied: pyproj>=3.1.0 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from cartopy) (3.6.1)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from matplotlib>=3.4-
>cartopy) (1.0.5)
Requirement already satisfied: cycler>=0.10 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from matplotlib>=3.4->car
topy) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from matplotlib>=3.4
->cartopy) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from matplotlib>=3.4
->cartopy) (1.4.4)
Requirement already satisfied: pillow>=6.2.0 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from matplotlib>=3.4->ca
rtopy) (9.4.0)
Requirement already satisfied: pyparsing<3.1,>=2.3.1 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from matplotlib>
=3.4->cartopy) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\samdure\appdata\roaming\python\python311\site-packages (from ma
tplotlib>=3.4->cartopy) (2.8.2)
Requirement already satisfied: certifi in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from pyproj>=3.1.0->cartopy)
(2023.7.22)
Requirement already satisfied: six>=1.5 in c:\users\samdure\appdata\roaming\python\python311\site-packages (from python-dateuti
l>=2.7->matplotlib>=3.4->cartopy) (1.16.0)


[notice] A new release of pip is available: 23.3.2 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [32]:
```
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
import matplotlib.patches as mpatches


# Create the plot
plt.figure(figsize=(10, 6))
ax = plt.axes(projection=ccrs.Miller())

# Plot terrorist attacks with casualties >= 75
casualties_high = terror[terror['casualities'] >= 75]
ax.plot(casualties_high['longitude'], casualties_high['latitude'], 'go', markersize=5, color='r')

# Plot terrorist attacks with casualties < 75
casualties_low = terror[terror['casualities'] < 75]
ax.plot(casualties_low['longitude'], casualties_low['latitude'], 'go', markersize=2, color='b', alpha=0.4)

# Create custom legend handles
legend_handles = [
    mpatches.Patch(color='b', label="< 75 casualties"),
    mpatches.Patch(color='red', label="> 75 casualties")
]

plt.legend(handles=legend_handles, loc='lower left')
plt.title('Global Terrorist Attacks')
plt.show()
```



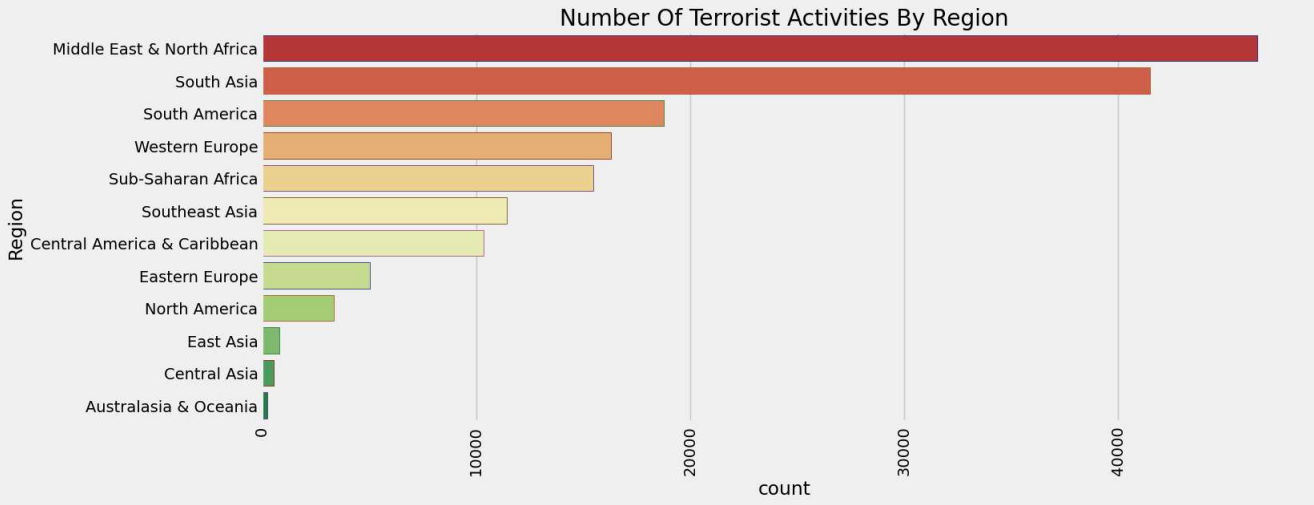
In [33]:
```
terror_fol=terror.copy()
terror_fol.dropna(subset=['latitude','longitude'],inplace=True)
location_fol=terror_fol[['latitude','longitude']][:5000]
country_fol=terror_fol['Country'][:5000]
city_fol=terror_fol['city'][:5000]
killed_fol=terror_fol['Killed'][:5000]
wound_fol=terror_fol['Wounded'][:5000]
def color_point(x):
    if x>=30:
        color='red'
    elif ((x>0 and x<30)):
        color='blue'
    else:
        color='green'
    return color
def point_size(x):
    if (x>30 and x<100):
        size=2
    elif (x>=100 and x<500):
        size=8
    elif x>=500:
        size=16
    else:
        size=0.5
    return size
map2 = folium.Map(location=[30,0],tiles='CartoDB dark_matter',zoom_start=2)
for point in location_fol.index:
    info='<b>Country: </b>'+str(country_fol[point])+'<br><b>City: </b>: '+str(city_fol[point])+'<br><b>Killed </b>: '+str(killed_
    iframe = folium.IFrame(html=info, width=200, height=200)
    folium.CircleMarker(list(location_fol.loc[point].values),popup=folium.Popup(iframe),radius=point_size(killed_fol[point]),colo
map2
```

Out[33]:



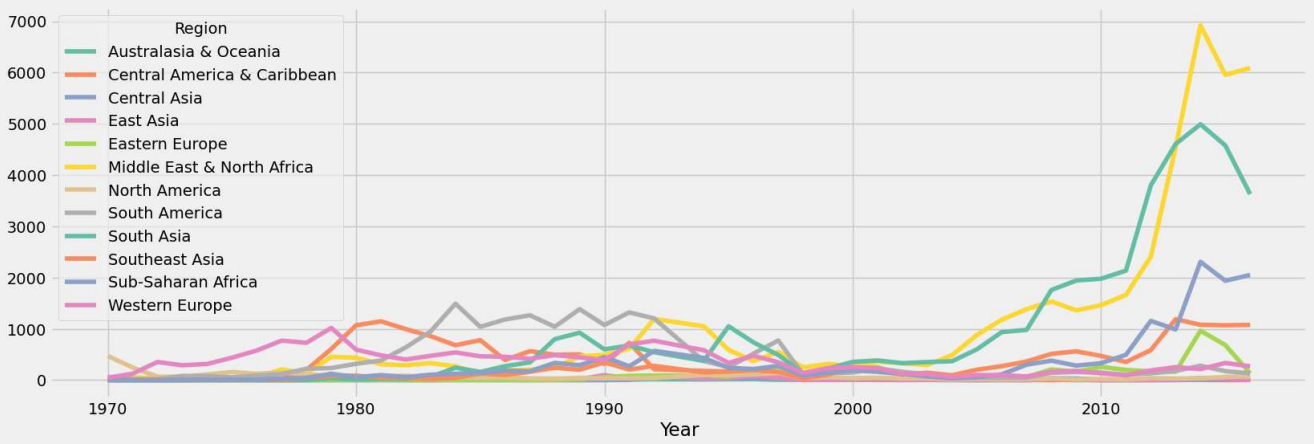
Terrorism by region

In [39]:
```
plt.subplots(figsize=(15,6))
sns.countplot(y='Region',data=terror,palette='RdYlGn',edgecolor=sns.color_palette('dark',7),order=terror['Region'].value_counts()
plt.xticks(rotation=90)
plt.title('Number Of Terrorist Activities By Region')
plt.show()
```



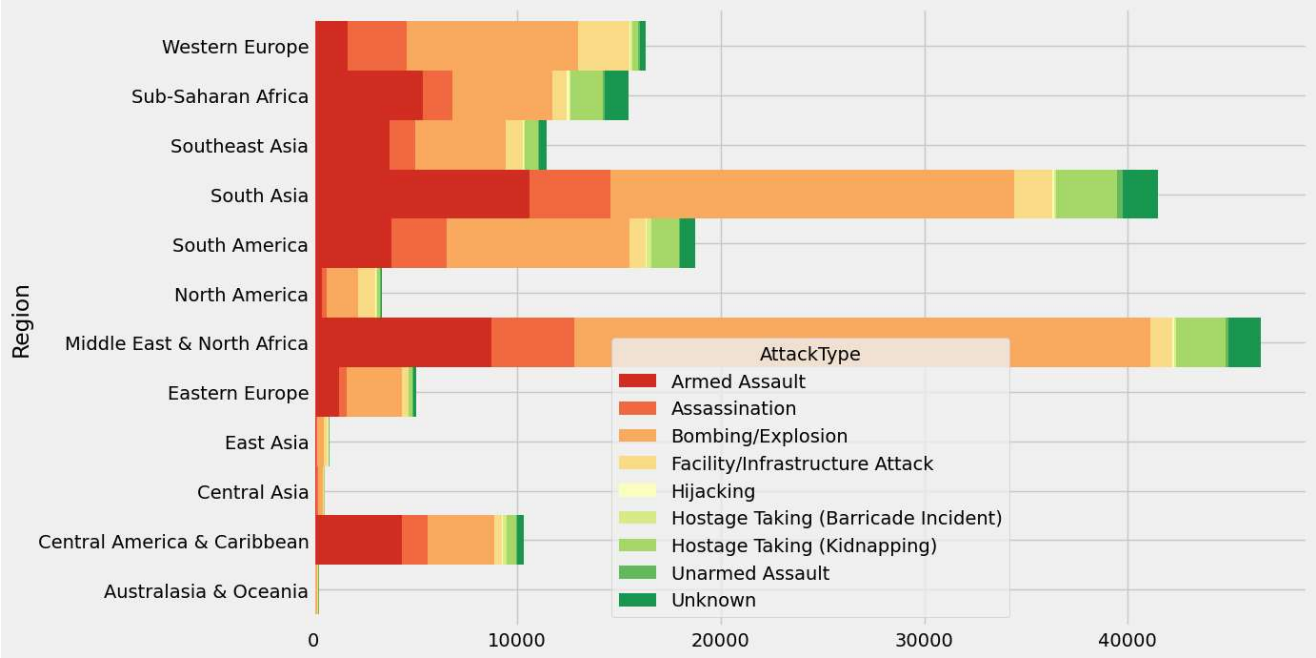
Trend in Terrorist activities

In [40]:
```
terror_region=pd.crosstab(terror.Year,terror.Region)
terror_region.plot(color=sns.color_palette('Set2',12))
fig=plt.gcf()
fig.set_size_inches(18,6)
plt.show()
```



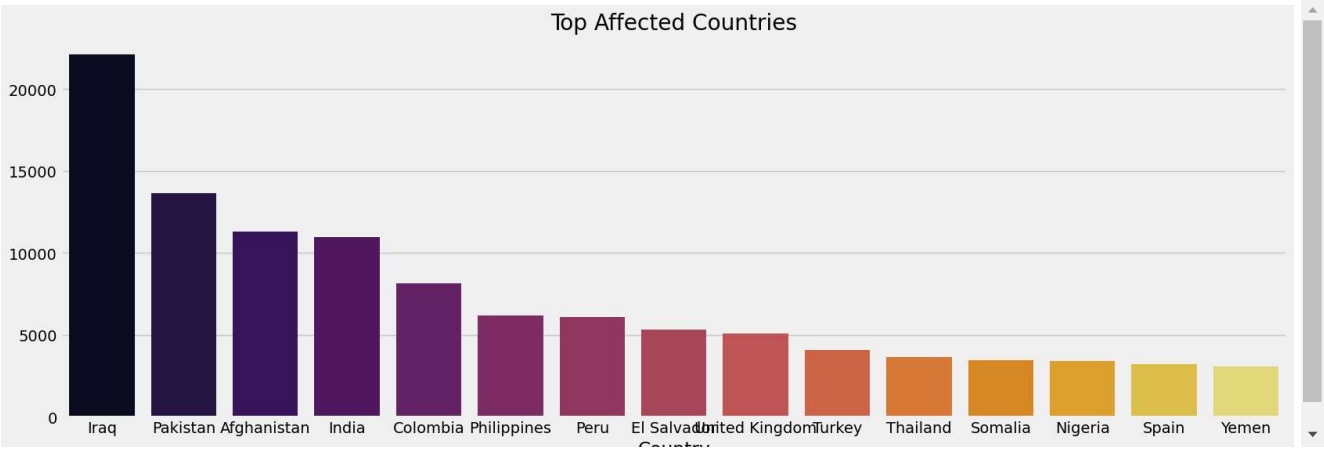
Attack type vs Region

In [41]:
```
pd.crosstab(terror.Region,terror.AttackType).plot.barh(stacked=True,width=1,color=sns.color_palette('RdYlGn',9))
fig=plt.gcf()
fig.set_size_inches(12,8)
plt.show()
```



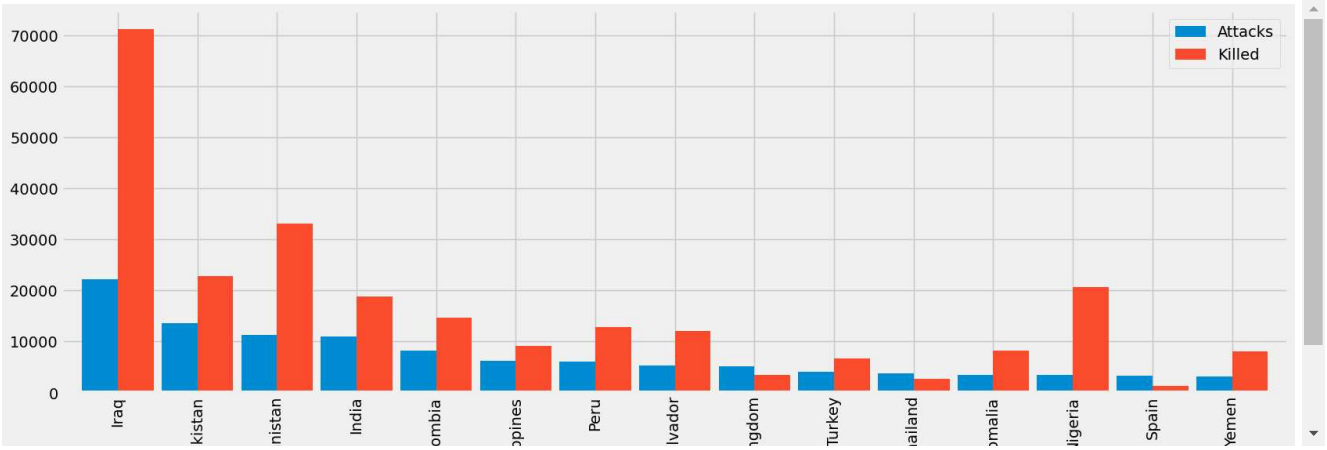
Terrorism by country

In [45]:
```
plt.subplots(figsize=(18, 6))
sns.barplot(x=terror['Country'].value_counts()[:15].index, y=terror['Country'].value_counts()[:15].values, palette='inferno')
plt.title('Top Affected Countries')
plt.show()
```



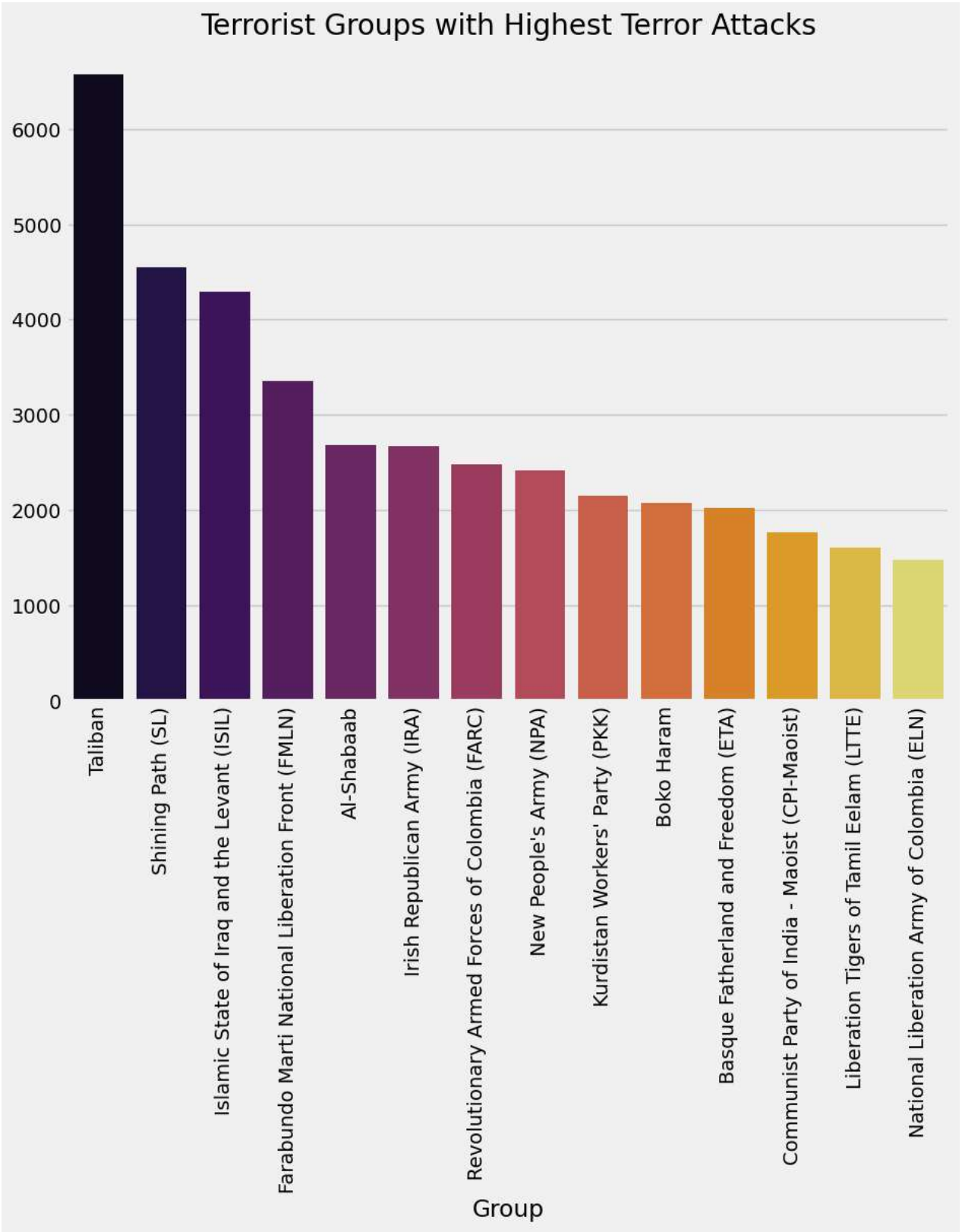
Attacks vs killed

In [46]:
```
coun_terror=terror['Country'].value_counts()[:15].to_frame()
coun_terror.columns=['Attacks']
coun_kill=terror.groupby('Country')['Killed'].sum().to_frame()
coun_terror.merge(coun_kill,left_index=True,right_index=True,how='left').plot.bar(width=0.9)
fig=plt.gcf()
fig.set_size_inches(18,6)
plt.show()
```



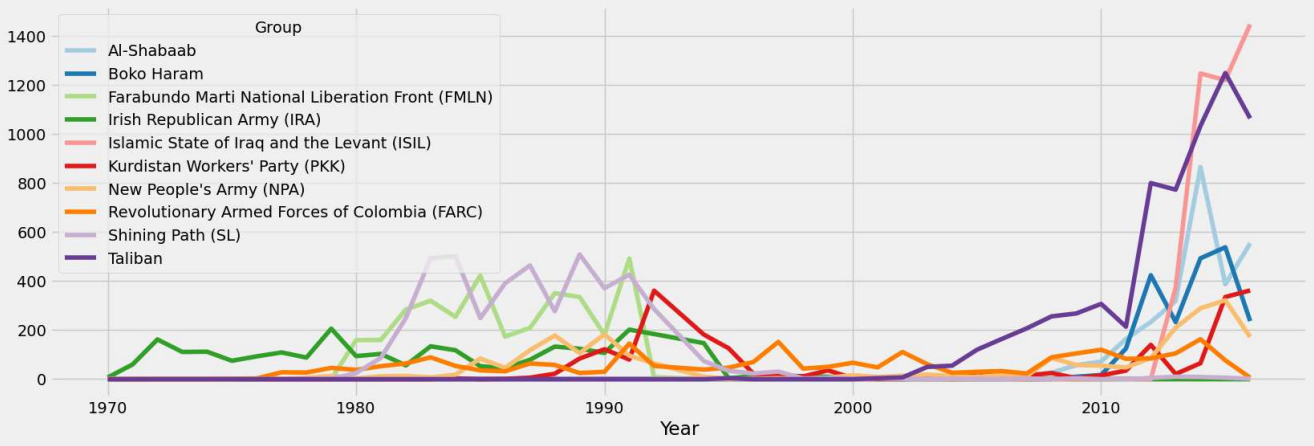
Most Notorious groups

In [50]:
```
import seaborn as sns
import matplotlib.pyplot as plt

plt.subplots(figsize=(10, 8))
sns.barplot(x=terror['Group'].value_counts()[1:15].index, y=terror['Group'].value_counts()[1:15].values, palette='inferno')
plt.xticks(rotation=90)
plt.title('Terrorist Groups with Highest Terror Attacks')
plt.show()
```


Terrorist Groups with Highest Terror Attacks
6000
5000
4000
3000
2000
1000
0
Taliban
Shining Path (SL)
Islamic State of Iraq and the Levant (ISIL)
Farabundo Marti National Liberation Front (FMLN)
Al-Shabaab
Irish Republican Army (IRA)
Revolutionary Armed Forces of Colombia (FARC)
New People's Army (NPA)
Kurdistan Workers' Party (PKK)
Boko Haram
Basque Fatherland and Freedom (ETA)
Communist Party of India - Maoist (CPI-Maoist)
Liberation Tigers of Tamil Eelam (LTTE)
National Liberation Army of Colombia (ELN)
Group

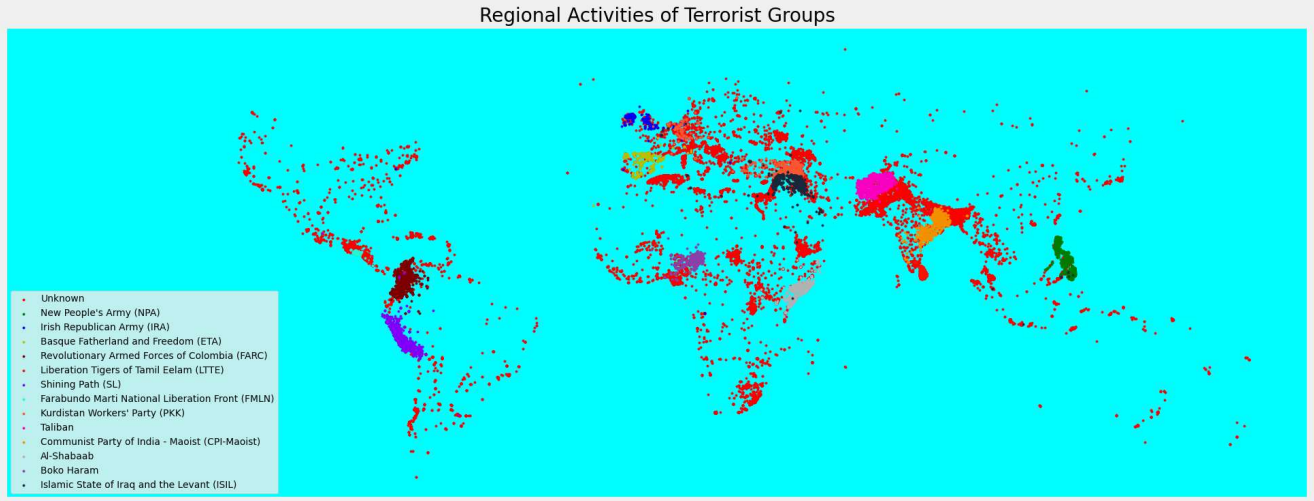
Activity of Top terrorist group

In [51]:
```
top_groups10=terror[terror['Group'].isin(terror['Group'].value_counts()[1:11].index)]
pd.crosstab(top_groups10.Year,top_groups10.Group).plot(color=sns.color_palette('Paired',10))
fig=plt.gcf()
fig.set_size_inches(18,6)
plt.show()
```



Regions attacked by Terrorist activities

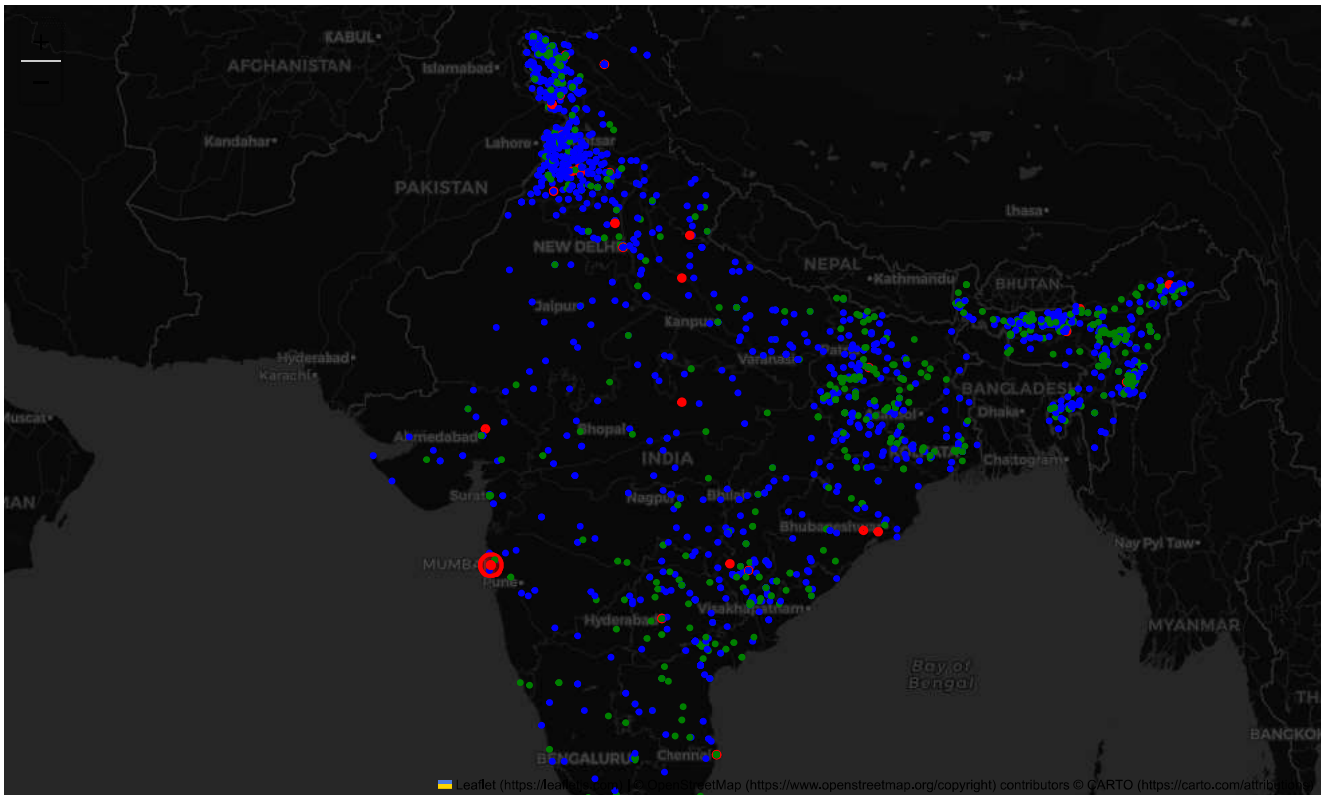
In [53]:
```
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
import cartopy.feature as cfeature

top_groups = terror[terror['Group'].isin(terror['Group'].value_counts()[:14].index)]

plt.figure(figsize=(22, 10))
ax = plt.axes(projection=ccrs.Miller())

# Draw coastlines, countries, and fill continents
ax.coastlines()
ax.add_feature(cfeature.BORDERS)
ax.add_feature(cfeature.LAND, edgecolor='black', facecolor='aqua')
ax.add_feature(cfeature.OCEAN, facecolor='aqua')

colors = ['r', 'g', 'b', 'y', '#800000', '#ff1100', '#8202fa', '#20fad9', '#ff5733', '#fa02c6', "#f99504", '#b3b6b7', '#8e44ad',
group = list(top_groups['Group'].unique())

for i, j in zip(group, colors):
    lat_group = list(top_groups[top_groups['Group'] == i]['latitude'])
    long_group = list(top_groups[top_groups['Group'] == i]['longitude'])
    ax.scatter(long_group, lat_group, color=j, label=i, s=5)

plt.legend(loc='lower left', frameon=True, prop={'size': 10})
plt.title('Regional Activities of Terrorist Groups')
plt.show()
```



Terror activites in INDIA

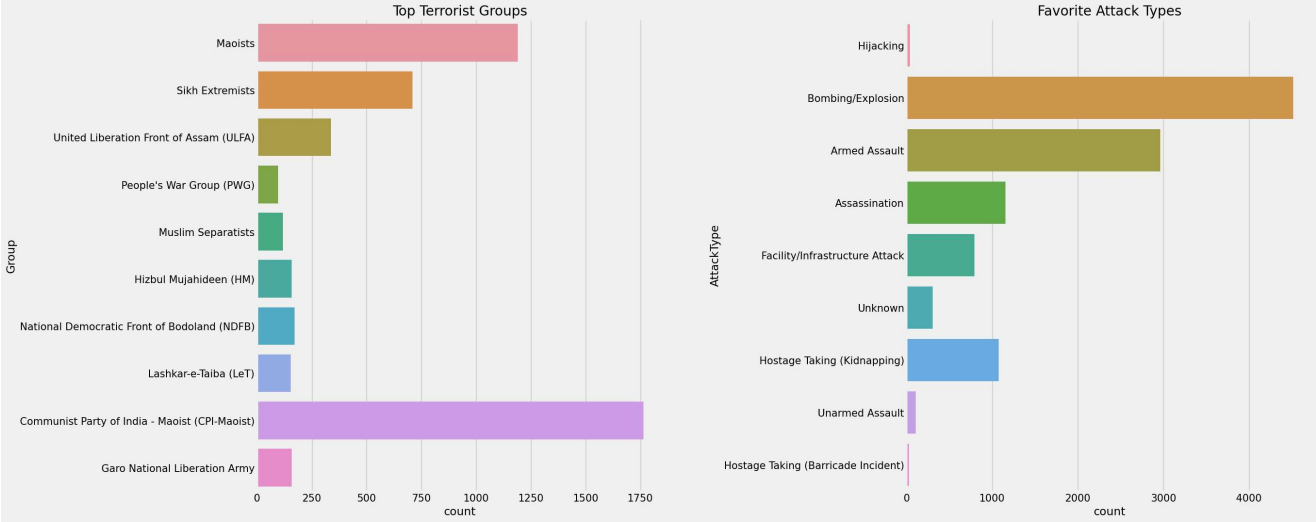
In [54]:
```
terror_india=terror[terror['Country']=='India']
terror_india_fol=terror_india.copy()
terror_india_fol.dropna(subset=['latitude','longitude'],inplace=True)
location_ind=terror_india_fol[['latitude','longitude']][:5000]
city_ind=terror_india_fol['city'][:5000]
killed_ind=terror_india_fol['Killed'][:5000]
wound_ind=terror_india_fol['Wounded'][:5000]
target_ind=terror_india_fol['Target_type'][:5000]

map4 = folium.Map(location=[20.59, 78.96],tiles='CartoDB dark_matter',zoom_start=4.5)
for point in location_ind.index:
    folium.CircleMarker(list(location_ind.loc[point].values),popup='<b>City: </b>'+str(city_ind[point])+'<br><b>Killed: </b>'+str
                        '<br><b>Injured: </b>'+str(wound_ind[point])+'<br><b>Target: </b>'+str(target_ind[point]),radius=point_si
map4
```

Out[54]:



Most Notorious Groups in India and Favorite Attack Types

In [55]:
```
f,ax=plt.subplots(1,2,figsize=(25,12))
ind_groups=terror_india['Group'].value_counts()[1:11].index
ind_groups=terror_india[terror_india['Group'].isin(ind_groups)]
sns.countplot(y='Group',data=ind_groups,ax=ax[0])
ax[0].set_title('Top Terrorist Groups')
sns.countplot(y='AttackType',data=terror_india,ax=ax[1])
ax[1].set_title('Favorite Attack Types')
plt.subplots_adjust(hspace=0.3,wspace=0.6)
ax[0].tick_params(labelsize=15)
ax[1].tick_params(labelsize=15)
plt.show()
```

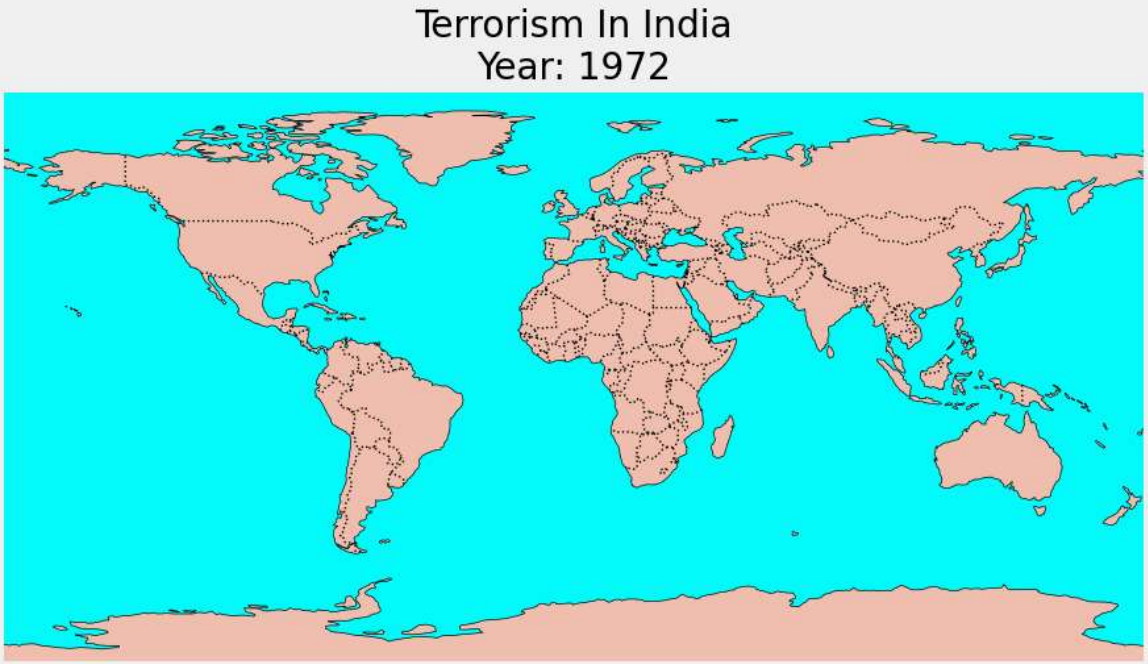


How did terrorism spread in India(Animation)

In [58]:
```
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
import cartopy.feature as cfeature
import pandas as pd
import numpy as np
from matplotlib.animation import FuncAnimation
import io
import base64
from IPython.display import HTML

# Assuming terror_india is your DataFrame containing terrorism data in India

fig = plt.figure(figsize=(10, 8))
ax = plt.axes(projection=ccrs.PlateCarree())

def animate(year):
    ax.clear()
    ax.set_title('Terrorism In India\nYear: {}'.format(year))
    ax.set_global()
    ax.add_feature(cfeature.COASTLINE)
    ax.add_feature(cfeature.BORDERS, linestyle=':', linewidth=1)
    ax.add_feature(cfeature.LAND, facecolor='coral', alpha=0.4)
    ax.add_feature(cfeature.OCEAN, facecolor='aqua')

    # Filter data for the current year
    year_data = terror_india[terror_india['Year'] == year]

    # Scatter plot incidents
    ax.scatter(year_data['longitude'], year_data['latitude'], s=year_data['Killed'] + year_data['Wounded'],
               color='red', transform=ccrs.PlateCarree())

years = sorted(terror_india['Year'].unique())

# Create the animation
ani = FuncAnimation(fig, animate, frames=years, interval=1500)

# Save the animation to a GIF file
ani.save('animation.gif', writer='imagemagick', fps=1)

# Close the figure
plt.close(fig)

# Display the GIF
filename = 'animation.gif'
video = io.open(filename, 'r+b').read()
encoded = base64.b64encode(video)
HTML(data='''<img src="{0}" type="gif" />'''.format(encoded.decode('ascii')))
```

```
MovieWriter imagemagick unavailable; using Pillow instead.
```

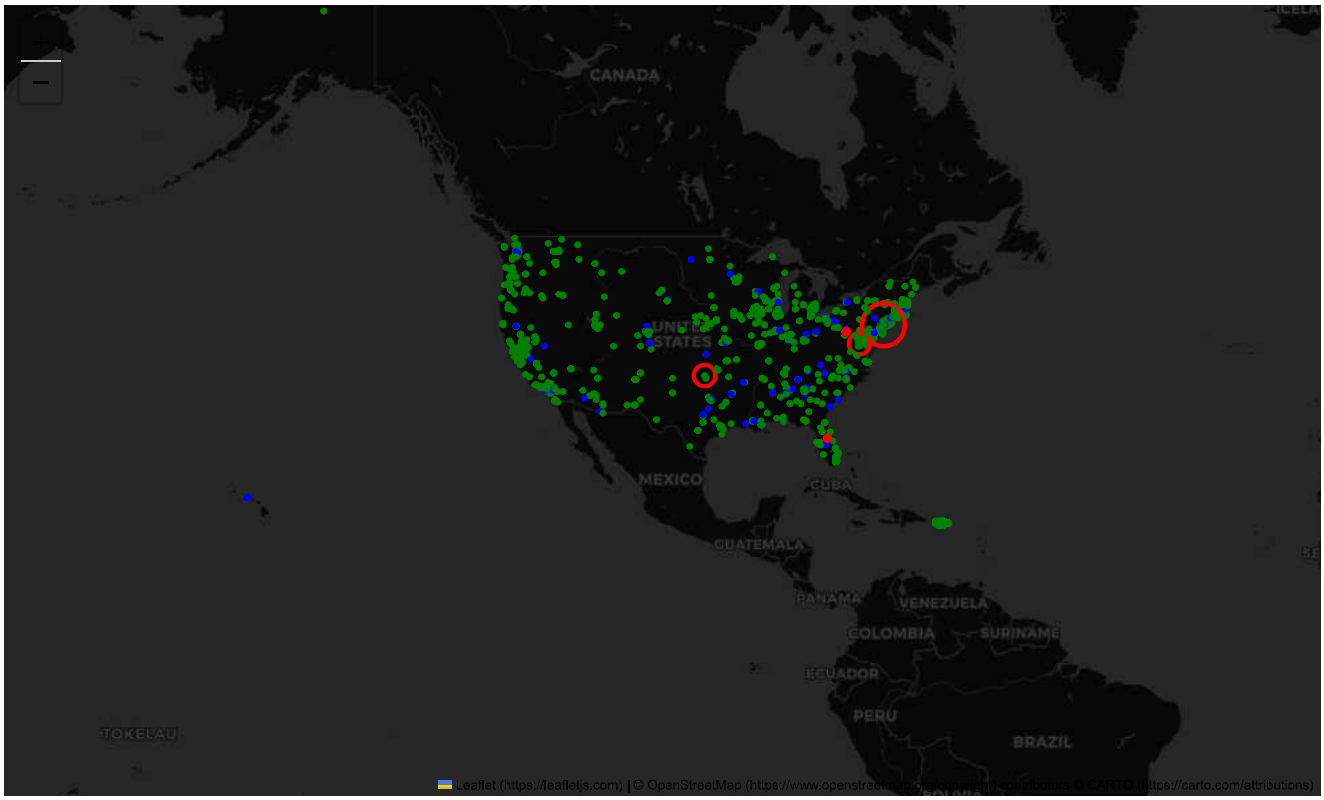
Out[58]:



Terror Activities in USA

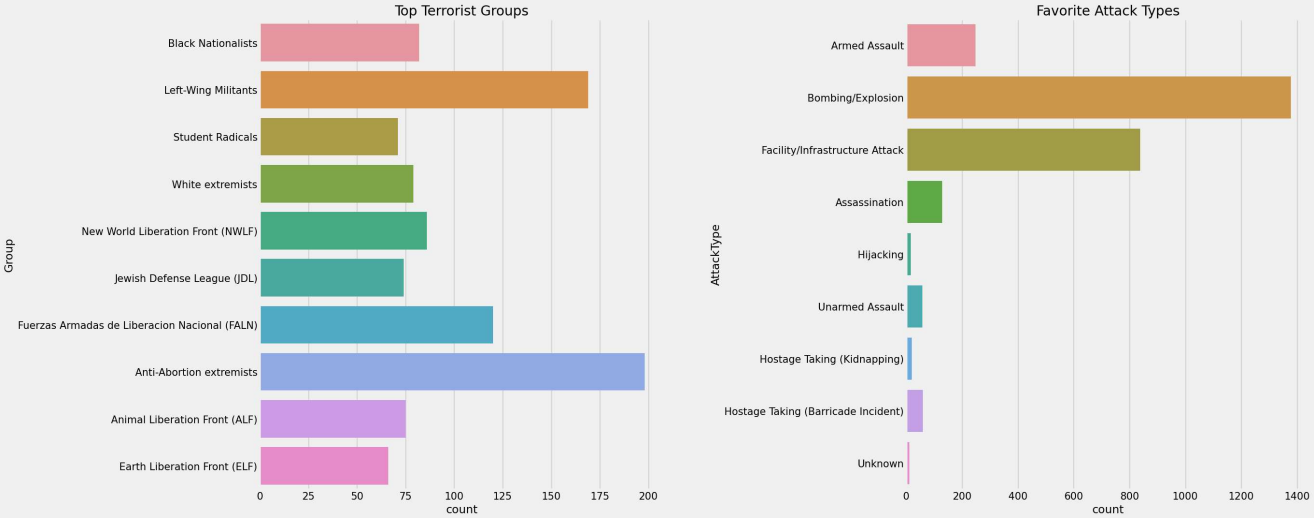
In [90]:
```
terror_usa=terror[terror['Country']=='United States']
terror_usa_fol=terror_usa.copy()
terror_usa_fol.dropna(subset=['latitude','longitude'],inplace=True)
location_usa=terror_usa_fol[['latitude','longitude']]
city_usa=terror_usa_fol['city']
killed_usa=terror_usa_fol['Killed']
wound_usa=terror_usa_fol['Wounded']
target_usa=terror_usa_fol['Target_type']

map5 = folium.Map(location=[39.50, -98.35],tiles='CartoDB dark_matter',zoom_start=3.5)
for point in location_usa.index:
    info='<b>City:</b>'+str(city_usa[point])+'<br><b>Killed</b>: '+str(killed_usa[point])+'<br><b>Wounded</b>: '+str(wound_usa[pc
    iframe = folium.IFrame(html=info, width=200, height=200)
    folium.CircleMarker(list(location_usa.loc[point].values),popup=folium.Popup(iframe),radius=point_size(killed_usa[point]),colc

map5
```

Out[90]:



Most Notorious Groups in USA and Favorite Attack Type

In [61]:
```
f,ax=plt.subplots(1,2,figsize=(25,12))
usa_groups=terror_usa['Group'].value_counts()[1:11].index
usa_groups=terror_usa[terror_usa['Group'].isin(usa_groups)]
sns.countplot(y='Group',data=usa_groups,ax=ax[0])
sns.countplot(y='AttackType',data=terror_usa,ax=ax[1])
plt.subplots_adjust(hspace=0.3,wspace=0.6)
ax[0].set_title('Top Terrorist Groups')
ax[1].set_title('Favorite Attack Types')
ax[0].tick_params(labelsize=15)
ax[1].tick_params(labelsize=15)
plt.show()
```

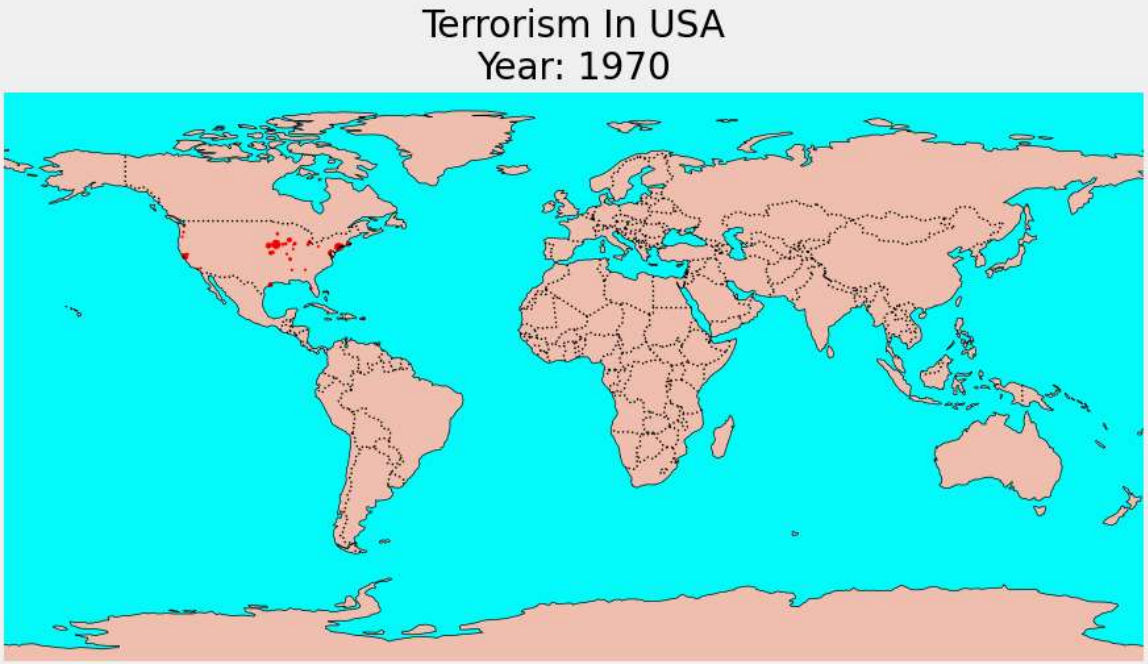


How did Terrorism Spread in USA(Animation)

In [65]:
```
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
import cartopy.feature as cfeature
import pandas as pd
import numpy as np
from matplotlib.animation import FuncAnimation
import io
import base64
from IPython.display import HTML

# Assuming terror_usa is your DataFrame containing terrorism data in the USA

fig = plt.figure(figsize=(10, 8))
ax = plt.axes(projection=ccrs.PlateCarree())

def animate(year):
    ax.clear()
    ax.set_title('Terrorism In USA\nYear: {}'.format(year))
    ax.set_global()
    ax.add_feature(cfeature.COASTLINE)
    ax.add_feature(cfeature.BORDERS, linestyle=':', linewidth=1)
    ax.add_feature(cfeature.LAND, facecolor='coral', alpha=0.4)
    ax.add_feature(cfeature.OCEAN, facecolor='aqua')

    # Filter data for the current year
    year_data = terror_usa[terror_usa['Year'] == year]

    # Scatter plot incidents
    ax.scatter(year_data['longitude'], year_data['latitude'], s=year_data['Killed'] + year_data['Wounded'],
               color='red', transform=ccrs.PlateCarree())

years = sorted(terror_usa['Year'].unique())

# Create the animation
ani = FuncAnimation(fig, animate, frames=years, interval=1500)

# Save the animation to a GIF file
ani.save('animation_usa.gif', writer='imagemagick', fps=1)

# Close the figure
plt.close(fig)

# Display the GIF
filename = 'animation_usa.gif'
video = io.open(filename, 'r+b').read()
encoded = base64.b64encode(video)
HTML(data='''<img src="{0}" type="gif" />'''.format(encoded.decode('ascii')))
```

```
MovieWriter imagemagick unavailable; using Pillow instead.
```

Out[65]:



Motive Behind Attacks

In [81]:
```
!pip install wordcloud
```

```
Collecting wordcloud
  Downloading wordcloud-1.9.3-cp311-cp311-win_amd64.whl.metadata (3.5 kB)
Requirement already satisfied: numpy>=1.6.1 in c:\users\samdure\appdata\roaming\python\python311\site-packages (from wordcloud)
(1.26.3)
Requirement already satisfied: pillow in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from wordcloud) (9.4.0)
Requirement already satisfied: matplotlib in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from wordcloud) (3.7.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from matplotlib->word
cloud) (1.0.5)
Requirement already satisfied: cycler>=0.10 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from matplotlib->wordclou
d) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from matplotlib->wor
dcloud) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from matplotlib->wor
dcloud) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from matplotlib->wordc
loud) (23.1)
Requirement already satisfied: pyparsing<3.1,>=2.3.1 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from matplotlib-
>wordcloud) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\samdure\appdata\roaming\python\python311\site-packages (from ma
tplotlib->wordcloud) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\samdure\appdata\roaming\python\python311\site-packages (from python-dateuti
l>=2.7->matplotlib->wordcloud) (1.16.0)
Downloading wordcloud-1.9.3-cp311-cp311-win_amd64.whl (300 kB)
   ---------------------------------------- 0.0/300.2 kB ? eta -:--:--
   ---------------------- ---------------- 174.1/300.2 kB 5.3 MB/s eta 0:00:01
   ---------------------------------------  297.0/300.2 kB 3.7 MB/s eta 0:00:01
   ---------------------------------------- 300.2/300.2 kB 3.1 MB/s eta 0:00:00
Installing collected packages: wordcloud
Successfully installed wordcloud-1.9.3


[notice] A new release of pip is available: 23.3.2 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [83]:
```
import nltk
nltk.download('punkt')
```

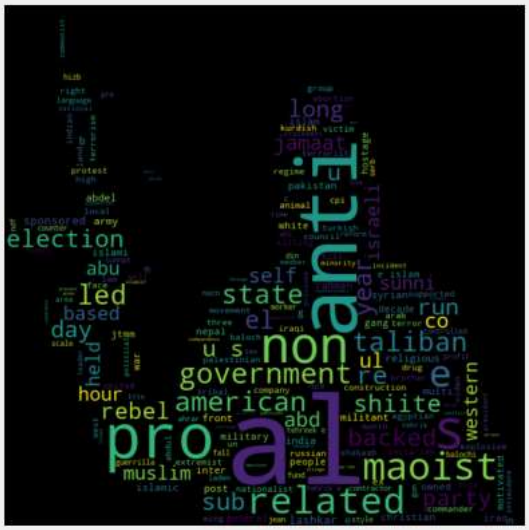
```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Samdure\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping tokenizers\punkt.zip.
```

Out[83]:
```
True
```

In [89]:
```
!pip install imageio
```

```
Requirement already satisfied: imageio in c:\users\samdure\anaconda3\anaconda\lib\site-packages (2.26.0)
Requirement already satisfied: numpy in c:\users\samdure\appdata\roaming\python\python311\site-packages (from imageio) (1.26.3)
Requirement already satisfied: pillow>=8.3.2 in c:\users\samdure\anaconda3\anaconda\lib\site-packages (from imageio) (9.4.0)


[notice] A new release of pip is available: 23.3.2 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [87]:
```
import nltk
import codecs
import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from imageio import imread

# Load data
motive = terror['Motive'].str.lower().str.replace(r'\|', ' ').str.cat(sep=' ')

# Tokenize words
words = word_tokenize(motive)

# Calculate word frequency
word_dist = FreqDist(words)

# Load stopwords
stopwords = set(stopwords.words('english'))

# Remove stopwords
words_except_stop_dist = FreqDist(w for w in words if w.lower() not in stopwords)

# Load mask image
img = imread("kaggle.png")

# Generate word cloud
wordcloud = WordCloud(stopwords=STOPWORDS, background_color='black', mask=img).generate(" ".join(words_except_stop_dist))

# Display word cloud
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```



World Terrorism Spread(Animation)

In [77]:
```
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
import cartopy.feature as cfeature
import pandas as pd
import numpy as np
from matplotlib.animation import FuncAnimation
import io
import base64
from IPython.display import HTML

# Assuming 'terror' is your DataFrame containing terrorist activities

fig = plt.figure(figsize=(10, 6))
ax = plt.axes(projection=ccrs.PlateCarree())

def animate(year):
    ax.clear()
    ax.set_title('Animation Of Terrorist Activities\nYear: {}'.format(year))
    ax.set_global()
    ax.add_feature(cfeature.COASTLINE)
    ax.add_feature(cfeature.BORDERS, linestyle=':', linewidth=1)
    ax.add_feature(cfeature.LAND, facecolor='coral', alpha=0.4)
    ax.add_feature(cfeature.OCEAN, facecolor='aqua')

    # Filter data for the current year
    year_data = terror[terror['Year'] == year]

    # Scatter plot incidents
    ax.scatter(year_data['longitude'], year_data['latitude'], s=(year_data['Killed'] + year_data['Wounded']) * 0.1,
               color='red', transform=ccrs.PlateCarree())

years = sorted(terror['Year'].unique())

# Create the animation
ani = FuncAnimation(fig, animate, frames=years, interval=1500)

# Save the animation to a GIF file
ani.save('animation.gif', writer='imagemagick', fps=1)

# Close the figure
plt.close(fig)

# Display the GIF
filename = 'animation.gif'
video = io.open(filename, 'r+b').read()
encoded = base64.b64encode(video)
HTML(data='''<img src="{0}" type="gif" />'''.format(encoded.decode('ascii')))
```

```
MovieWriter imagemagick unavailable; using Pillow instead.
```

Out[77]: