Global Terrorism Data Cleaning

Importing the Libraries

In [ ]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
```

- Load the datasets

In [ ]:
```
original_data = pd.read_csv("Global-terrorism-Datasets.csv")
```

```
C:\Users\Akash Pandey\AppData\Local\Temp\ipykernel_18356\1104420386.py:1: Dtype
Warning: Columns (4,6,31,33,61,62,63,76,79,90,92,94,96,114,115,121) have mixed
types. Specify dtype option on import or set low_memory=False.
  original_data = pd.read_csv("Global-terrorism-Datasets.csv")
```

In [ ]:
```
df = original_data.copy() # Make a copy of original datasets
```

In [ ]:
```
pd.set_option('display.max_columns', None) # for showing the all columns in data
```

In [ ]:
```
df.head() # top 5 rows
```

Out[ ]:

	eventid	iyear	imonth	iday	approxdate	extended	resolution	country	country_txt
0	197000000001	1970	7	2	NaN	0	NaN	58	Dominican Republic
1	197000000002	1970	0	0	NaN	0	NaN	130	Mexico
2	197001000001	1970	1	0	NaN	0	NaN	160	Philippines
3	197001000002	1970	1	0	NaN	0	NaN	78	Greece
4	197001000003	1970	1	0	NaN	0	NaN	101	Japan

In [ ]:
```
df.shape
```

Out[ ]: (181691, 135)

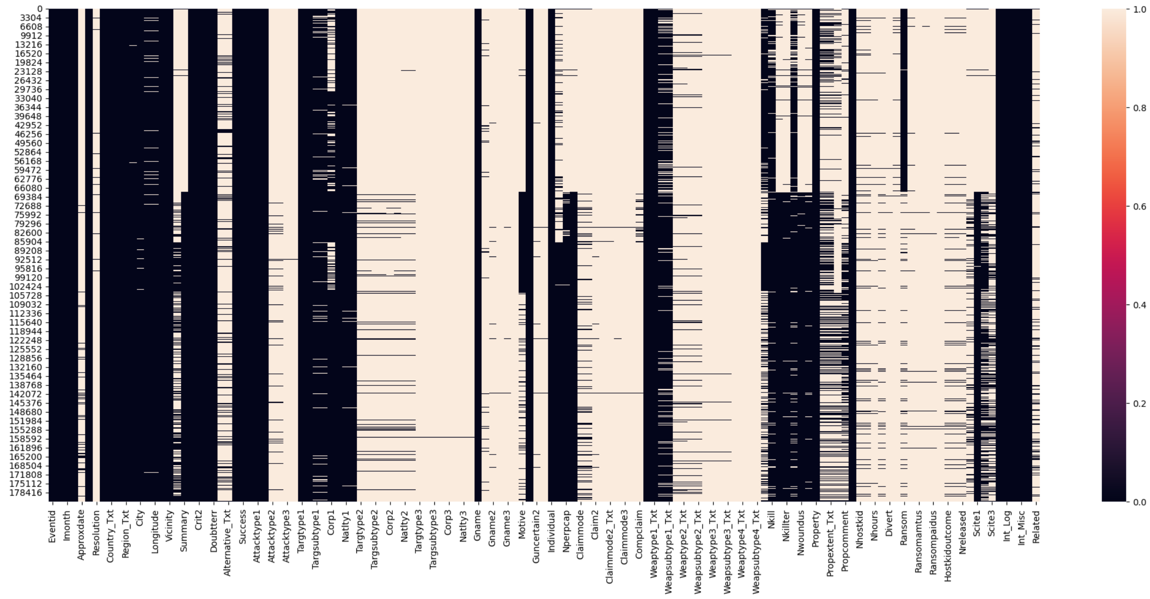
In [ ]:
```
df.columns = df.columns.str.title()
```

In [ ]:
```
df.isnull().sum()
```

```
Out[ ]: Eventid              0
        Iyear                0
        Imonth               0
        Iday                 0
        Approxdate      172452
                         ...
        Int_Log              0
        Int_Ideo             0
        Int_Misc             0
        Int_Any              0
        Related         156653
        Length: 135, dtype: int64
```

```
In [ ]: plt.figure(figsize=(25,10))



        sns.heatmap(df.isnull())
```

```
Out[ ]: <Axes: >
```



```
In [ ]: # Check the size of the dataset
        num_records, num_attributes = df.shape
        print(f"The dataset contains {num_records} records and {num_attributes} Columns.
```

```
The dataset contains 181691 records and 135 Columns.
```

```
In [ ]: # Check the data types of the attributes
        print("Data types of the attributes:")
        print(df.dtypes)
```

```
Data types of the attributes:
Eventid          int64
Iyear            int64
Imonth           int64
Iday             int64
Approxdate      object
                 ...
Int_Log          int64
Int_Ideo         int64
Int_Misc         int64
Int_Any          int64
Related         object
Length: 135, dtype: object
```

In [ ]:

```
# Check for missing values
num_missing_values = df.isnull().sum().sum()
print(f"\nThe dataset contains {num_missing_values} missing values.\n")
```

```
The dataset contains 13853997 missing values.
```

In [ ]:

```
# Check for duplicate records
num_duplicate_records = df.duplicated().sum()
print(f"The dataset contains {num_duplicate_records} duplicate records.")
```

```
The dataset contains 0 duplicate records.
```

In [ ]:

```
# changing the null columns in persentange form
null_col = df.isnull().sum()/len(df)*100
null_col
```

Out[ ]:

```
Eventid          0.000000
Iyear            0.000000
Imonth           0.000000
Iday             0.000000
Approxdate      94.914993
                  ...
Int_Log          0.000000
Int_Ideo         0.000000
Int_Misc         0.000000
Int_Any          0.000000
Related         86.219461
Length: 135, dtype: float64
```

In [ ]:

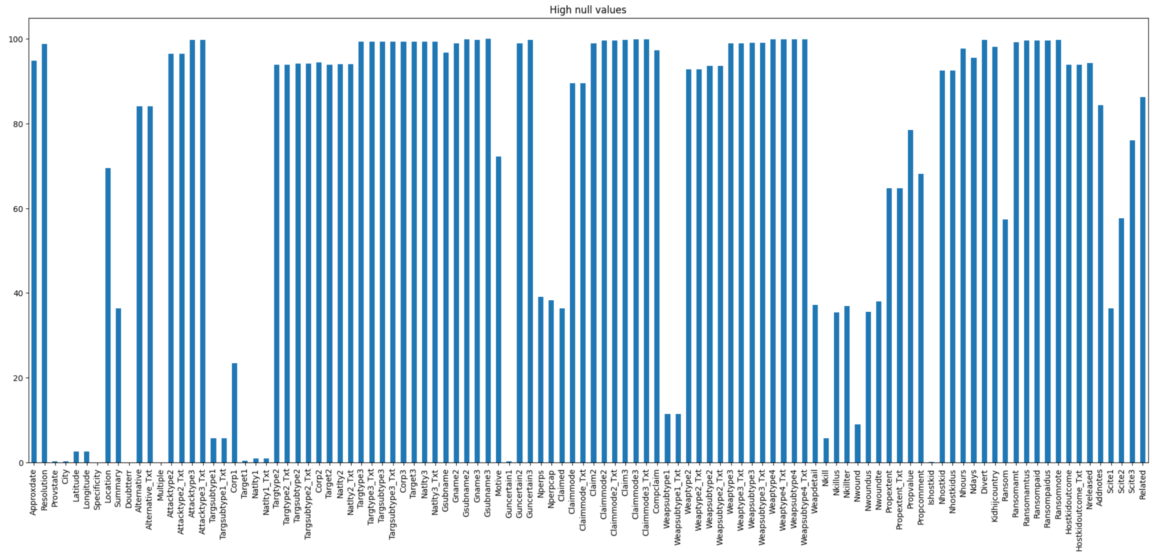
```
# filter the columns which columns has more the 0% values.
null_col = null_col.loc[null_col.values>0]
null_col
```

Out[ ]:

```
Approxdate      94.914993
Resolution      98.778145
Provstate        0.231712
City             0.238867
Latitude         2.507554
                  ...
Addnotes        84.430159
Scite1          36.430533
Scite2          57.657231
Scite3          76.049447
Related         86.219461
Length: 106, dtype: float64
```

In [ ]:
```python
# ploting the null columns which are having more than 0 % null values

plt.figure(figsize=(25,10))

null_col.plot(kind='bar', title='High null values')
```

Out[ ]:
```
<Axes: title={'center': 'High null values'}>
```



In [ ]:
```python
# filter that columns who have more than 50% null values in the datasets

null_col_50 = null_col.loc[null_col.values > 50]
null_col_50
```

Out[ ]:
```
Approxdate         94.914993
Resolution         98.778145
Location           69.456385
Alternative        84.032781
Alternative_Txt    84.032781
                     ...
Nreleased          94.275996
Addnotes           84.430159
Scite2             57.657231
Scite3             76.049447
Related            86.219461
Length: 77, dtype: float64
```

In [ ]:
```python
# there are that columns which are having more than 50 % null values
null_col_50.index
```

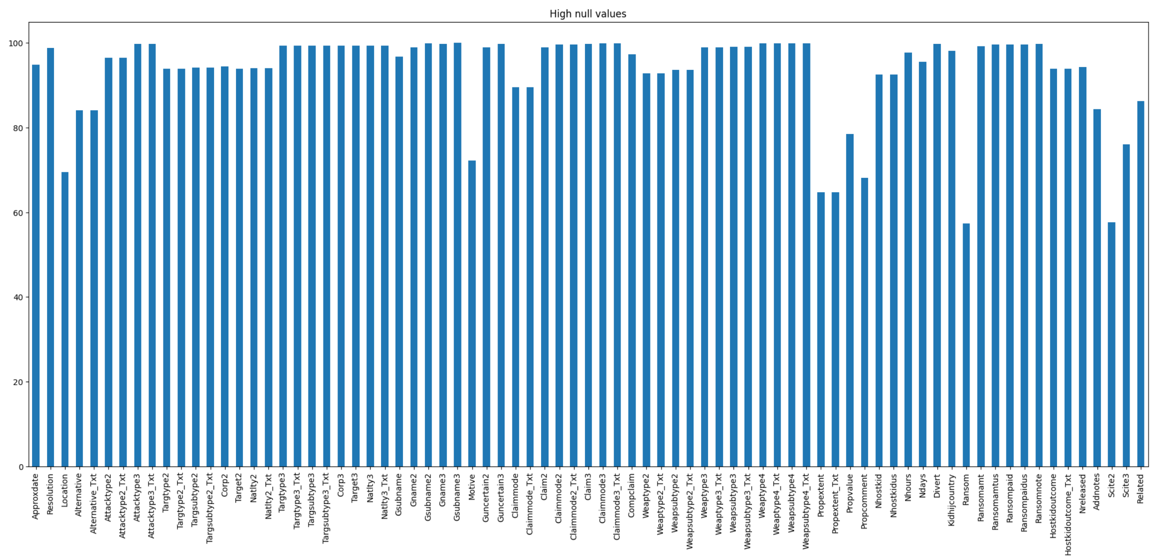
Out[ ]:
```
Index(['Approxdate', 'Resolution', 'Location', 'Alternative',
       'Alternative_Txt', 'Attacktype2', 'Attacktype2_Txt', 'Attacktype3',
       'Attacktype3_Txt', 'Targtype2', 'Targtype2_Txt', 'Targsubtype2',
       'Targsubtype2_Txt', 'Corp2', 'Target2', 'Natlty2', 'Natlty2_Txt',
       'Targtype3', 'Targtype3_Txt', 'Targsubtype3', 'Targsubtype3_Txt',
       'Corp3', 'Target3', 'Natlty3', 'Natlty3_Txt', 'Gsubname', 'Gname2',
       'Gsubname2', 'Gname3', 'Gsubname3', 'Motive', 'Guncertain2',
       'Guncertain3', 'Claimmode', 'Claimmode_Txt', 'Claim2', 'Claimmode2',
       'Claimmode2_Txt', 'Claim3', 'Claimmode3', 'Claimmode3_Txt', 'Compclaim',
       'Weaptype2', 'Weaptype2_Txt', 'Weapsubtype2', 'Weapsubtype2_Txt',
       'Weaptype3', 'Weaptype3_Txt', 'Weapsubtype3', 'Weapsubtype3_Txt',
       'Weaptype4', 'Weaptype4_Txt', 'Weapsubtype4', 'Weapsubtype4_Txt',
       'Propextent', 'Propextent_Txt', 'Propvalue', 'Propcomment', 'Nhostkid',
       'Nhostkidus', 'Nhours', 'Ndays', 'Divert', 'Kidhijcountry', 'Ransom',
       'Ransomamt', 'Ransomamtus', 'Ransompaid', 'Ransompaidus', 'Ransomnote',
       'Hostkidoutcome', 'Hostkidoutcome_Txt', 'Nreleased', 'Addnotes',
       'Scite2', 'Scite3', 'Related'],
      dtype='object')
```

In [ ]:
```python
print(f'The total number of columns is {len(null_col_50)} have more than 50% col
```

```
The total number of columns is 77 have more than 50% columns
```

In [ ]:
```python
# ploting the null columns which are having more than 50 % null values

plt.figure(figsize=(25,10))

null_col_50.plot(kind='bar', title='High null values')
```

Out[ ]:
```
<Axes: title={'center': 'High null values'}>
```



In [ ]:
```python
# delete the columns who are having more than 50% null values in the datasets

df = df.drop(columns=list(null_col_50.index))
```

In [ ]:
```python
# chenking the columns number after deleting
num_records, num_attributes = df.shape
print(f"The dataset contains {num_records} records and {num_attributes} attribut
```

```
The dataset contains 181691 records and 58 attributes after the deleteing the h
igh null vaules columns.
```

In [ ]:
```
# again check the null values after deleting teh 50 % null values
df.isnull().sum().sum()
```

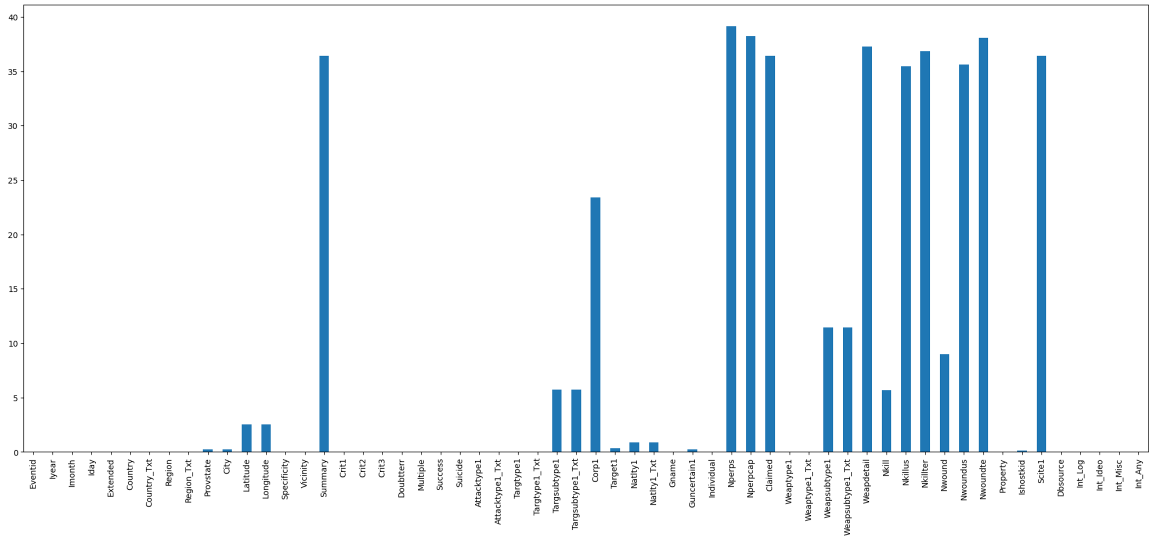
Out[ ]: 817707

In [ ]:
```
# changing the null values in %
null_col1 = df.isnull().sum()/len(df)*100
null_col1
```

Out[ ]:
```
Eventid               0.000000
Iyear                 0.000000
Imonth                0.000000
Iday                  0.000000
Extended              0.000000
Country               0.000000
Country_Txt           0.000000
Region                0.000000
Region_Txt            0.000000
Provstate             0.231712
City                  0.238867
Latitude              2.507554
Longitude             2.508104
Specificity           0.003302
Vicinity              0.000000
Summary              36.396409
Crit1                 0.000000
Crit2                 0.000000
Crit3                 0.000000
Doubtterr             0.000550
Multiple              0.000550
Success               0.000000
Suicide               0.000000
Attacktype1           0.000000
Attacktype1_Txt       0.000000
Targtype1             0.000000
Targtype1_Txt         0.000000
Targsubtype1          5.709144
Targsubtype1_Txt      5.709144
Corp1                23.418882
Target1               0.350045
Natlty1               0.858050
Natlty1_Txt           0.858050
Gname                 0.000000
Guncertain1           0.209146
Individual            0.000000
Nperps               39.140629
Nperpcap             38.245703
Claimed              36.391456
Weaptype1             0.000000
Weaptype1_Txt         0.000000
Weapsubtype1         11.430396
Weapsubtype1_Txt     11.430396
Weapdetail           37.244553
Nkill                 5.676120
Nkillus              35.470111
Nkillter             36.852678
Nwound                8.977330
Nwoundus             35.611010
Nwoundte             38.055270
Property              0.000000
Ishostkid             0.097969
Scite1               36.430533
Dbsource              0.000000
Int_Log               0.000000
Int_Ideo              0.000000
Int_Misc              0.000000
Int_Any               0.000000
dtype: float64
```

In [ ]:
```python
# ploting againt null values

plt.figure(figsize=(25,10))

null_col1.plot(kind='bar')
```

Out[ ]: <Axes: >



In [ ]:
```python
# filter that columns who have string datatypes

df.select_dtypes(include=object).columns
```

Out[ ]:
```
Index(['Country_Txt', 'Region_Txt', 'Provstate', 'City', 'Summary',
       'Attacktype1_Txt', 'Targtype1_Txt', 'Targsubtype1_Txt', 'Corp1',
       'Target1', 'Natlty1_Txt', 'Gname', 'Weaptype1_Txt', 'Weapsubtype1_Txt',
       'Weapdetail', 'Scite1', 'Dbsource'],
      dtype='object')
```

In [ ]:
```python
# Now, replace null values with "Unknown"

df[df.select_dtypes(include=object).columns] = df[df.select_dtypes(include=objec
```

In [ ]:
```python
# checking the null values for string datatypes columns
df.select_dtypes(include=object).isnull().sum()
```

Out[ ]:
```
Country_Txt          0
Region_Txt           0
Provstate            0
City                 0
Summary              0
Attacktype1_Txt      0
Targtype1_Txt        0
Targsubtype1_Txt     0
Corp1                0
Target1              0
Natlty1_Txt          0
Gname                0
Weaptype1_Txt        0
Weapsubtype1_Txt     0
Weapdetail           0
Scite1               0
Dbsource             0
dtype: int64
```

In [ ]:
```
# filter the intiger and float columns
df.select_dtypes(exclude=object).columns
```

Out[ ]:
```
Index(['Eventid', 'Iyear', 'Imonth', 'Iday', 'Extended', 'Country', 'Region',
       'Latitude', 'Longitude', 'Specificity', 'Vicinity', 'Crit1', 'Crit2',
       'Crit3', 'Doubtterr', 'Multiple', 'Success', 'Suicide', 'Attacktype1',
       'Targtype1', 'Targsubtype1', 'Natlty1', 'Guncertain1', 'Individual',
       'Nperps', 'Nperpcap', 'Claimed', 'Weaptype1', 'Weapsubtype1', 'Nkill',
       'Nkillus', 'Nkillter', 'Nwound', 'Nwoundus', 'Nwoundte', 'Property',
       'Ishostkid', 'Int_Log', 'Int_Ideo', 'Int_Misc', 'Int_Any'],
      dtype='object')
```

In [ ]:
```
# check info for intiger and float datatype columns

df.select_dtypes(exclude=object).info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 181691 entries, 0 to 181690
Data columns (total 41 columns):
 #   Column        Non-Null Count   Dtype  
---  ------        --------------   -----  
 0   Eventid       181691 non-null  int64  
 1   Iyear         181691 non-null  int64  
 2   Imonth        181691 non-null  int64  
 3   Iday          181691 non-null  int64  
 4   Extended      181691 non-null  int64  
 5   Country       181691 non-null  int64  
 6   Region        181691 non-null  int64  
 7   Latitude      177135 non-null  float64
 8   Longitude     177134 non-null  float64
 9   Specificity   181685 non-null  float64
 10  Vicinity      181691 non-null  int64  
 11  Crit1         181691 non-null  int64  
 12  Crit2         181691 non-null  int64  
 13  Crit3         181691 non-null  int64  
 14  Doubtterr     181690 non-null  float64
 15  Multiple      181690 non-null  float64
 16  Success       181691 non-null  int64  
 17  Suicide       181691 non-null  int64  
 18  Attacktype1   181691 non-null  int64  
 19  Targtype1     181691 non-null  int64  
 20  Targsubtype1  171318 non-null  float64
 21  Natlty1       180132 non-null  float64
 22  Guncertain1   181311 non-null  float64
 23  Individual    181691 non-null  int64  
 24  Nperps        110576 non-null  float64
 25  Nperpcap      112202 non-null  float64
 26  Claimed       115571 non-null  float64
 27  Weaptype1     181691 non-null  int64  
 28  Weapsubtype1  160923 non-null  float64
 29  Nkill         171378 non-null  float64
 30  Nkillus       117245 non-null  float64
 31  Nkillter      114733 non-null  float64
 32  Nwound        165380 non-null  float64
 33  Nwoundus      116989 non-null  float64
 34  Nwoundte      112548 non-null  float64
 35  Property      181691 non-null  int64  
 36  Ishostkid     181513 non-null  float64
 37  Int_Log       181691 non-null  int64  
 38  Int_Ideo      181691 non-null  int64  
 39  Int_Misc      181691 non-null  int64  
 40  Int_Any       181691 non-null  int64  
dtypes: float64(19), int64(22)
memory usage: 56.8 MB
```

In [ ]:

```
# replace the null values with 0 for intiger and float datatype columns in the a

df[df.select_dtypes(exclude=object).columns] = df[df.select_dtypes(exclude=objec
```

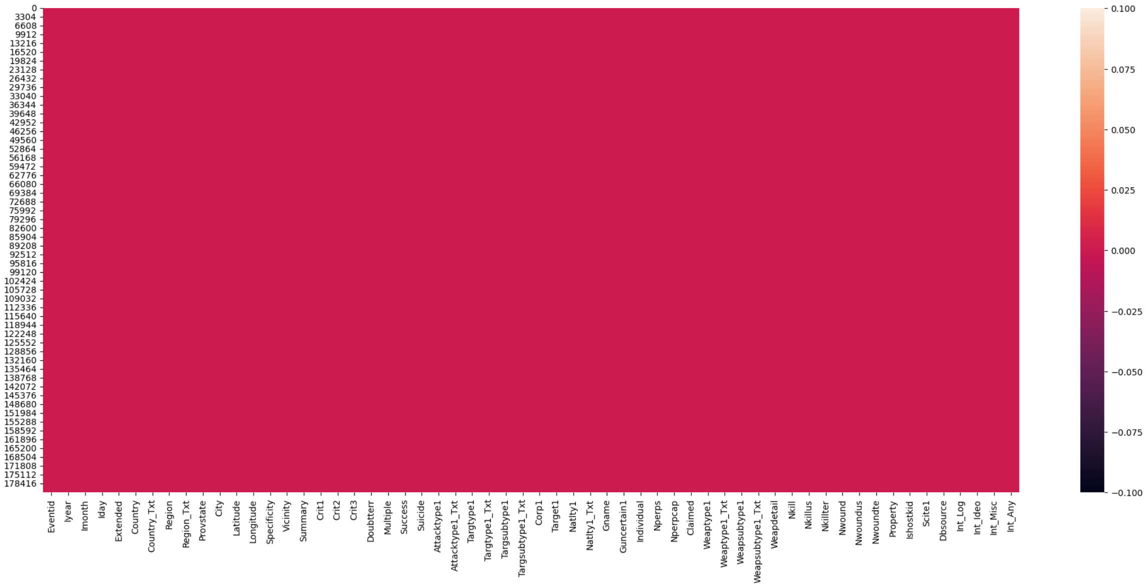
In [ ]:

```
# checking the final null values

df.isnull().sum()
```

```
Out[ ]: Eventid             0
        Iyear               0
        Imonth              0
        Iday                0
        Extended            0
        Country             0
        Country_Txt         0
        Region              0
        Region_Txt          0
        Provstate           0
        City                0
        Latitude            0
        Longitude           0
        Specificity         0
        Vicinity            0
        Summary             0
        Crit1               0
        Crit2               0
        Crit3               0
        Doubtterr           0
        Multiple            0
        Success             0
        Suicide             0
        Attacktype1         0
        Attacktype1_Txt     0
        Targtype1           0
        Targtype1_Txt       0
        Targsubtype1        0
        Targsubtype1_Txt    0
        Corp1               0
        Target1             0
        Natlty1             0
        Natlty1_Txt         0
        Gname               0
        Guncertain1         0
        Individual          0
        Nperps              0
        Nperpcap            0
        Claimed             0
        Weaptype1           0
        Weaptype1_Txt       0
        Weapsubtype1        0
        Weapsubtype1_Txt    0
        Weapdetail          0
        Nkill               0
        Nkillus             0
        Nkillter            0
        Nwound              0
        Nwoundus            0
        Nwoundte            0
        Property            0
        Ishostkid           0
        Scite1              0
        Dbsource            0
        Int_Log             0
        Int_Ideo            0
        Int_Misc            0
        Int_Any             0
        dtype: int64
```

In [ ]:
```python
plt.figure(figsize=(25,10))

#ploting the final checking null values with heatmap
sns.heatmap(df.isnull())
```

Out[ ]: <Axes: >



- The graph is look Red it means that we dont have any null values in datasets

In [ ]:
```python
print(f'Now we have {df.isnull().sum().sum()} values in the Datasets')
```

```
Now we have 0 values in the Datasets
```

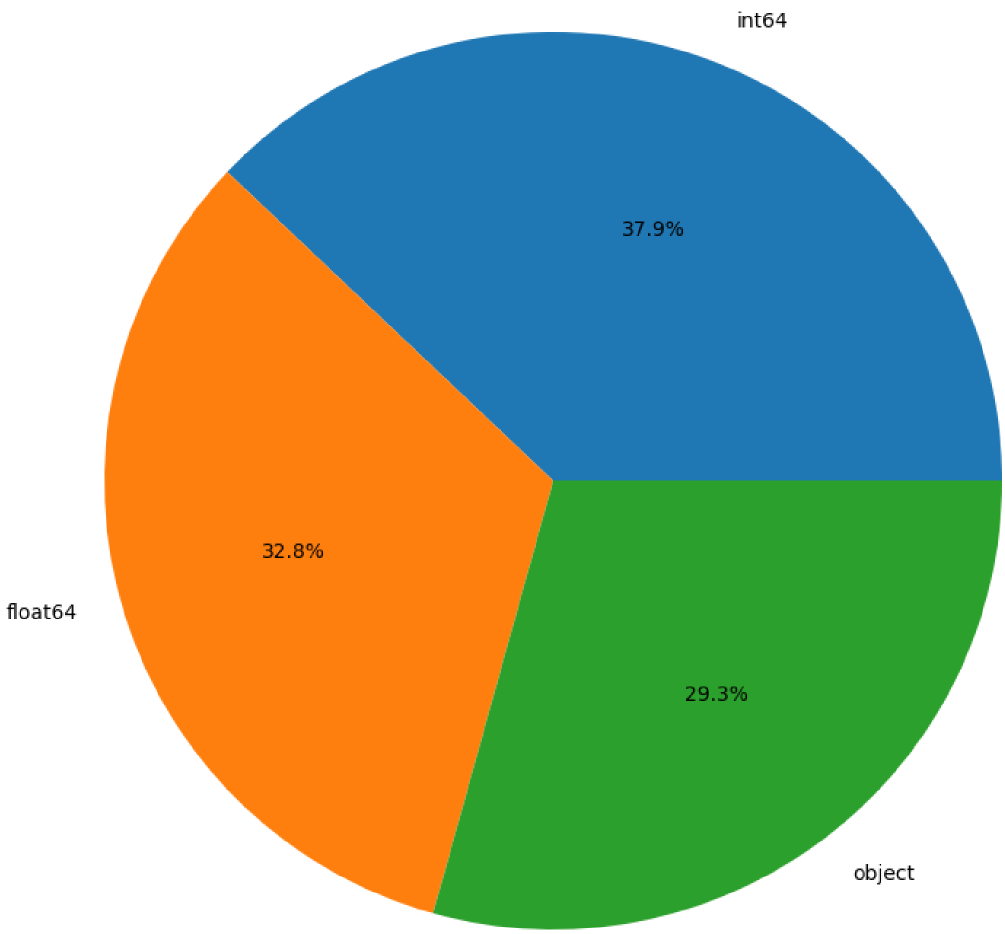
In [ ]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 181691 entries, 0 to 181690
Data columns (total 58 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   Eventid           181691 non-null  int64  
 1   Iyear             181691 non-null  int64  
 2   Imonth            181691 non-null  int64  
 3   Iday              181691 non-null  int64  
 4   Extended          181691 non-null  int64  
 5   Country           181691 non-null  int64  
 6   Country_Txt       181691 non-null  object 
 7   Region            181691 non-null  int64  
 8   Region_Txt        181691 non-null  object 
 9   Provstate         181691 non-null  object 
 10  City              181691 non-null  object 
 11  Latitude          181691 non-null  float64
 12  Longitude         181691 non-null  float64
 13  Specificity       181691 non-null  float64
 14  Vicinity          181691 non-null  int64  
 15  Summary           181691 non-null  object 
 16  Crit1             181691 non-null  int64  
 17  Crit2             181691 non-null  int64  
 18  Crit3             181691 non-null  int64  
 19  Doubtterr         181691 non-null  float64
 20  Multiple          181691 non-null  float64
 21  Success           181691 non-null  int64  
 22  Suicide           181691 non-null  int64  
 23  Attacktype1       181691 non-null  int64  
 24  Attacktype1_Txt   181691 non-null  object 
 25  Targtype1         181691 non-null  int64  
 26  Targtype1_Txt     181691 non-null  object 
 27  Targsubtype1      181691 non-null  float64
 28  Targsubtype1_Txt  181691 non-null  object 
 29  Corp1             181691 non-null  object 
 30  Target1           181691 non-null  object 
 31  Natlty1           181691 non-null  float64
 32  Natlty1_Txt       181691 non-null  object 
 33  Gname             181691 non-null  object 
 34  Guncertain1       181691 non-null  float64
 35  Individual        181691 non-null  int64  
 36  Nperps            181691 non-null  float64
 37  Nperpcap          181691 non-null  float64
 38  Claimed           181691 non-null  float64
 39  Weaptype1         181691 non-null  int64  
 40  Weaptype1_Txt     181691 non-null  object 
 41  Weapsubtype1      181691 non-null  float64
 42  Weapsubtype1_Txt  181691 non-null  object 
 43  Weapdetail        181691 non-null  object 
 44  Nkill             181691 non-null  float64
 45  Nkillus           181691 non-null  float64
 46  Nkillter          181691 non-null  float64
 47  Nwound            181691 non-null  float64
 48  Nwoundus          181691 non-null  float64
 49  Nwoundte          181691 non-null  float64
 50  Property          181691 non-null  int64  
 51  Ishostkid         181691 non-null  float64
 52  Scite1            181691 non-null  object 
 53  Dbsource          181691 non-null  object 
 54  Int_Log           181691 non-null  int64  
```

```
 55   Int_Ideo           181691 non-null  int64
 56   Int_Misc           181691 non-null  int64
 57   Int_Any            181691 non-null  int64
dtypes: float64(19), int64(22), object(17)
memory usage: 80.4+ MB
```

In [ ]:
```
plt.figure(figsize=(25,10))

df.dtypes.value_counts().plot(kind='pie', autopct='%1.1f%%')
```

Out[ ]: <Axes: >



In [ ]:
```
df.isnull().sum().sum()
```

Out[ ]: 0

Now Data cleaning is Over So we export the cleaned dataset for perform EDA analysis on that

In [ ]:
```
df.to_csv('Global-Terrorism-Cleaned-Datasets.csv', index= False)
```

In [ ]:
```
df
```

Out[ ]:

	Eventid	Iyear	Imonth	Iday	Extended	Country	Country_Txt	Region	Region_
0	197000000001	1970	7	2	0	58	Dominican Republic	2	Cen Americ Caribb
1	197000000002	1970	0	0	0	130	Mexico	1	No Ame
2	197001000001	1970	1	0	0	160	Philippines	5	Southe A
3	197001000002	1970	1	0	0	78	Greece	8	West Eur
4	197001000003	1970	1	0	0	101	Japan	4	East A
...	...	...	...	...	...	...	...	...	
181686	201712310022	2017	12	31	0	182	Somalia	11	S Saha Af
181687	201712310029	2017	12	31	0	200	Syria	10	Middle I & No Af
181688	201712310030	2017	12	31	0	160	Philippines	5	Southe A
181689	201712310031	2017	12	31	0	92	India	6	South A
181690	201712310032	2017	12	31	0	160	Philippines	5	Southe A

181691 rows × 58 columns

Global Terrorist EDA Analysis

In [ ]:
```
# import all libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [ ]:
```
# Load the Datasets
cleaned_datasets = pd.read_csv("Global-Terrorism-Cleaned-Datasets.csv")
```

In [ ]:
```
# for showing all columns

pd.set_option('display.max_columns', None)
```

In [ ]:
```
# Copy from the original Datasets
df = cleaned_datasets.copy()
```

In [ ]:
```
df.sample(5)
```

Out[ ]:

	Eventid	Iyear	Imonth	Iday	Extended	Country	Country_Txt	Region	Region_
11825	198009120007	1980	9	12	0	61	El Salvador	2	Cer Americ Caribb
87607	200806270009	2008	6	27	0	45	Colombia	3	Sc Ame
157509	201601100018	2016	1	10	0	95	Iraq	10	Middle I & Nc Af
178989	201709190028	2017	9	19	0	200	Syria	10	Middle I & Nc Af
129061	201403200043	2014	3	20	0	95	Iraq	10	Middle I & Nc Af

In [ ]:
```
# check the null values

print(f" Here is the {df.isnull().sum().sum() } values in the datasets ")
```

```
 Here is the 0 values in the datasets
```

In [ ]:
```
# check the info

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 181691 entries, 0 to 181690
Data columns (total 58 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   Eventid           181691 non-null  int64  
 1   Iyear             181691 non-null  int64  
 2   Imonth            181691 non-null  int64  
 3   Iday              181691 non-null  int64  
 4   Extended          181691 non-null  int64  
 5   Country           181691 non-null  int64  
 6   Country_Txt       181691 non-null  object 
 7   Region            181691 non-null  int64  
 8   Region_Txt        181691 non-null  object 
 9   Provstate         181691 non-null  object 
 10  City              181691 non-null  object 
 11  Latitude          181691 non-null  float64
 12  Longitude         181691 non-null  float64
 13  Specificity       181691 non-null  float64
 14  Vicinity          181691 non-null  int64  
 15  Summary           181691 non-null  object 
 16  Crit1             181691 non-null  int64  
 17  Crit2             181691 non-null  int64  
 18  Crit3             181691 non-null  int64  
 19  Doubtterr         181691 non-null  float64
 20  Multiple          181691 non-null  float64
 21  Success           181691 non-null  int64  
 22  Suicide           181691 non-null  int64  
 23  Attacktype1       181691 non-null  int64  
 24  Attacktype1_Txt   181691 non-null  object 
 25  Targtype1         181691 non-null  int64  
 26  Targtype1_Txt     181691 non-null  object 
 27  Targsubtype1      181691 non-null  float64
 28  Targsubtype1_Txt  181691 non-null  object 
 29  Corp1             181691 non-null  object 
 30  Target1           181691 non-null  object 
 31  Natlty1           181691 non-null  float64
 32  Natlty1_Txt       181691 non-null  object 
 33  Gname             181691 non-null  object 
 34  Guncertain1       181691 non-null  float64
 35  Individual        181691 non-null  int64  
 36  Nperps            181691 non-null  float64
 37  Nperpcap          181691 non-null  float64
 38  Claimed           181691 non-null  float64
 39  Weaptype1         181691 non-null  int64  
 40  Weaptype1_Txt     181691 non-null  object 
 41  Weapsubtype1      181691 non-null  float64
 42  Weapsubtype1_Txt  181691 non-null  object 
 43  Weapdetail        181691 non-null  object 
 44  Nkill             181691 non-null  float64
 45  Nkillus           181691 non-null  float64
 46  Nkillter          181691 non-null  float64
 47  Nwound            181691 non-null  float64
 48  Nwoundus          181691 non-null  float64
 49  Nwoundte          181691 non-null  float64
 50  Property          181691 non-null  int64  
 51  Ishostkid         181691 non-null  float64
 52  Scite1            181691 non-null  object 
 53  Dbsource          181691 non-null  object 
 54  Int_Log           181691 non-null  int64  
```

```
 55   Int_Ideo          181691 non-null  int64
 56   Int_Misc          181691 non-null  int64
 57   Int_Any           181691 non-null  int64
dtypes: float64(19), int64(22), object(17)
memory usage: 80.4+ MB
```

Now, We Can start the EDA Analysis

Discriptive Analysis

In [ ]:
```
df.describe().T
```

Out[ ]:

	count	mean	std	min	25%	
Eventid	181691.0	2.002705e+11	1.325957e+09	1.970000e+11	1.991021e+11	2.009022
Iyear	181691.0	2.002639e+03	1.325943e+01	1.970000e+03	1.991000e+03	2.009000
Imonth	181691.0	6.467277e+00	3.388303e+00	0.000000e+00	4.000000e+00	6.000000
Iday	181691.0	1.550564e+01	8.814045e+00	0.000000e+00	8.000000e+00	1.500000
Extended	181691.0	4.534622e-02	2.080629e-01	0.000000e+00	0.000000e+00	0.000000
Country	181691.0	1.319685e+02	1.124145e+02	4.000000e+00	7.800000e+01	9.800000
Region	181691.0	7.160938e+00	2.933408e+00	1.000000e+00	5.000000e+00	6.000000
Latitude	181691.0	2.290911e+01	1.869944e+01	-5.315461e+01	9.518645e+00	3.112665
Longitude	181691.0	-4.471911e+02	2.021946e+05	-8.618590e+07	1.231572e+00	4.314357
Specificity	181691.0	1.451404e+00	9.954480e-01	0.000000e+00	1.000000e+00	1.000000
Vicinity	181691.0	6.829727e-02	2.845529e-01	-9.000000e+00	0.000000e+00	0.000000
Crit1	181691.0	9.885300e-01	1.064825e-01	0.000000e+00	1.000000e+00	1.000000
Crit2	181691.0	9.930927e-01	8.282305e-02	0.000000e+00	1.000000e+00	1.000000
Crit3	181691.0	8.756680e-01	3.299608e-01	0.000000e+00	1.000000e+00	1.000000
Doubtterr	181691.0	-5.231685e-01	2.455813e+00	-9.000000e+00	0.000000e+00	0.000000
Multiple	181691.0	1.377724e-01	3.446619e-01	0.000000e+00	0.000000e+00	0.000000
Success	181691.0	8.895983e-01	3.133907e-01	0.000000e+00	1.000000e+00	1.000000
Suicide	181691.0	3.650704e-02	1.875486e-01	0.000000e+00	0.000000e+00	0.000000
Attacktype1	181691.0	3.247547e+00	1.915772e+00	1.000000e+00	2.000000e+00	3.000000
Targtype1	181691.0	8.439719e+00	6.653838e+00	1.000000e+00	3.000000e+00	4.000000
Targsubtype1	181691.0	4.428981e+01	3.197157e+01	0.000000e+00	2.100000e+01	3.400000
Natlty1	181691.0	1.265908e+02	8.969172e+01	0.000000e+00	8.300000e+01	9.800000
Guncertain1	181691.0	8.126985e-02	2.732498e-01	0.000000e+00	0.000000e+00	0.000000
Individual	181691.0	2.950064e-03	5.423446e-02	0.000000e+00	0.000000e+00	0.000000
Nperps	181691.0	-3.977839e+01	1.719109e+02	-9.900000e+01	-9.900000e+01	0.000000
Nperpcap	181691.0	-9.372616e-01	1.010951e+01	-9.900000e+01	0.000000e+00	0.000000
Claimed	181691.0	3.159210e-02	8.722035e-01	-9.000000e+00	0.000000e+00	0.000000
Weaptype1	181691.0	6.447325e+00	2.173435e+00	1.000000e+00	5.000000e+00	6.000000
Weapsubtype1	181691.0	9.846426e+00	7.062747e+00	0.000000e+00	4.000000e+00	1.100000
Nkill	181691.0	2.266860e+00	1.122706e+01	0.000000e+00	0.000000e+00	0.000000
Nkillus	181691.0	2.967126e-02	4.564308e+00	0.000000e+00	0.000000e+00	0.000000
Nkillter	181691.0	3.208249e-01	3.346474e+00	0.000000e+00	0.000000e+00	0.000000
Nwound	181691.0	2.883296e+00	3.430975e+01	0.000000e+00	0.000000e+00	0.000000
Nwoundus	181691.0	2.507554e-02	2.453378e+00	0.000000e+00	0.000000e+00	0.000000
Nwoundte	181691.0	6.638193e-02	1.172976e+00	0.000000e+00	0.000000e+00	0.000000

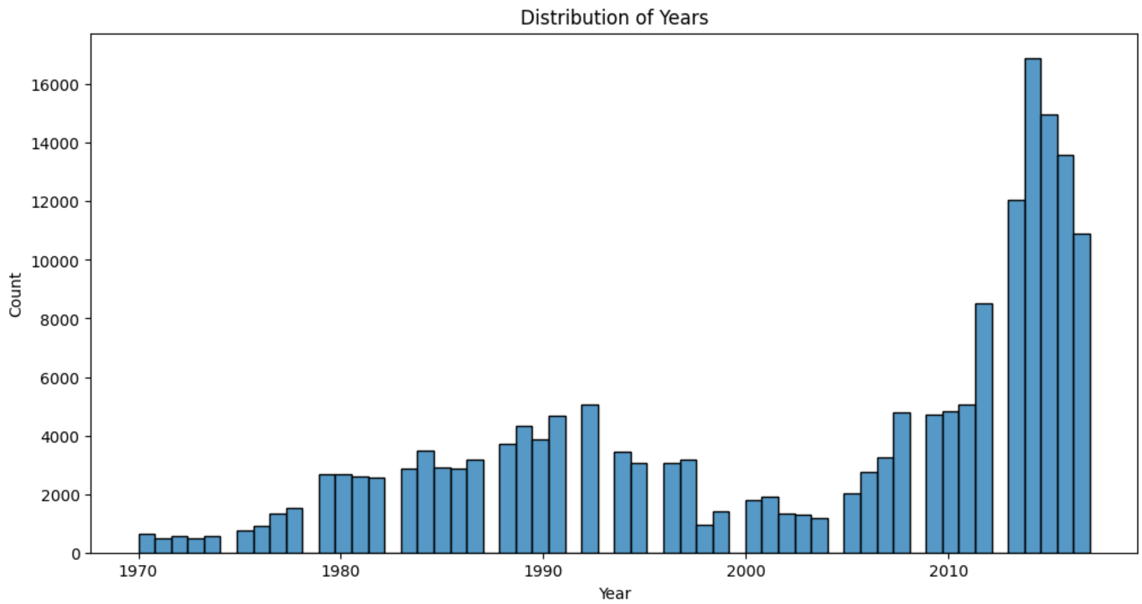
	count	mean	std	min	25%	
Property	181691.0	-5.445564e-01	3.122889e+00	-9.000000e+00	0.000000e+00	1.000000
Ishostkid	181691.0	5.899577e-02	4.610220e-01	-9.000000e+00	0.000000e+00	0.000000
Int_Log	181691.0	-4.543731e+00	4.543547e+00	-9.000000e+00	-9.000000e+00	-9.000000
Int_Ideo	181691.0	-4.464398e+00	4.637152e+00	-9.000000e+00	-9.000000e+00	-9.000000
Int_Misc	181691.0	9.000996e-02	5.684573e-01	-9.000000e+00	0.000000e+00	0.000000
Int_Any	181691.0	3.945952e+00	4.691325e+00	9.000000e+00	9.000000e+00	0.000000

Distributions

In [ ]:
```python
df.columns
```

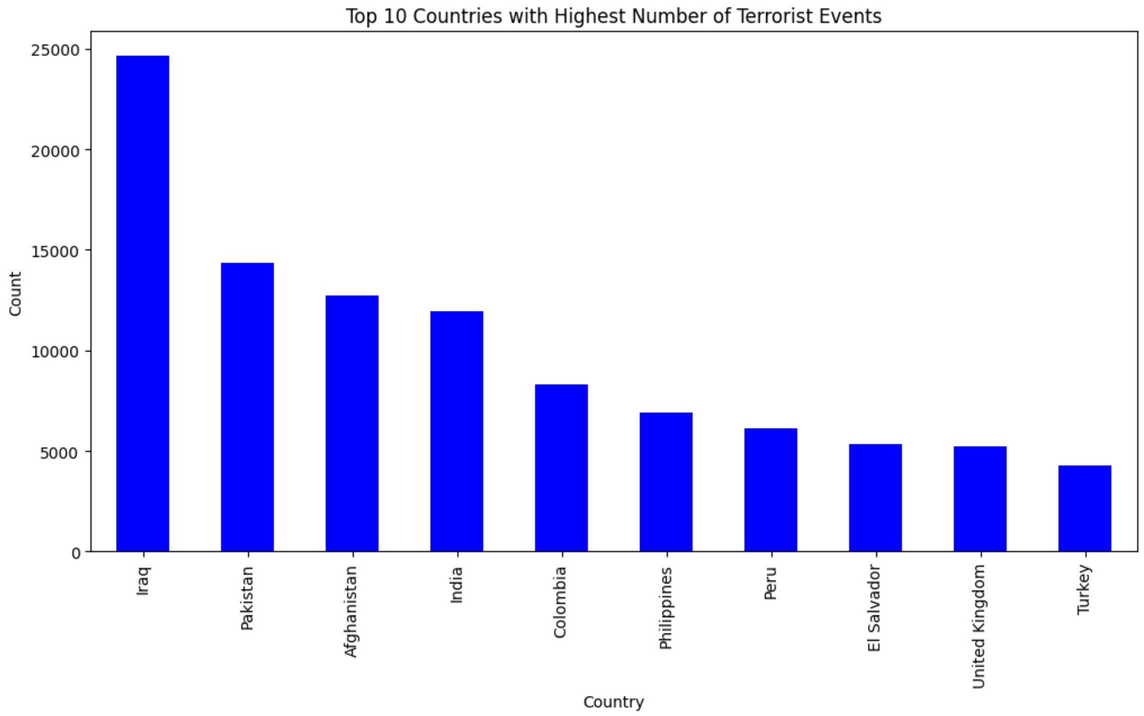
Out[ ]:
```
Index(['Eventid', 'Iyear', 'Imonth', 'Iday', 'Extended', 'Country',
       'Country_Txt', 'Region', 'Region_Txt', 'Provstate', 'City', 'Latitude',
       'Longitude', 'Specificity', 'Vicinity', 'Summary', 'Crit1', 'Crit2',
       'Crit3', 'Doubtterr', 'Multiple', 'Success', 'Suicide', 'Attacktype1',
       'Attacktype1_Txt', 'Targtype1', 'Targtype1_Txt', 'Targsubtype1',
       'Targsubtype1_Txt', 'Corp1', 'Target1', 'Natlty1', 'Natlty1_Txt',
       'Gname', 'Guncertain1', 'Individual', 'Nperps', 'Nperpcap', 'Claimed',
       'Weaptype1', 'Weaptype1_Txt', 'Weapsubtype1', 'Weapsubtype1_Txt',
       'Weapdetail', 'Nkill', 'Nkillus', 'Nkillter', 'Nwound', 'Nwoundus',
       'Nwoundte', 'Property', 'Ishostkid', 'Scite1', 'Dbsource', 'Int_Log',
       'Int_Ideo', 'Int_Misc', 'Int_Any'],
      dtype='object')
```

In [ ]:
```python
# Ploting the distribution of the 'iyear' attribute
plt.figure(figsize=(12,6))
sns.histplot(x=df['Iyear'])
plt.title('Distribution of Years')
plt.xlabel('Year')
plt.ylabel('Count')
plt.show()
```



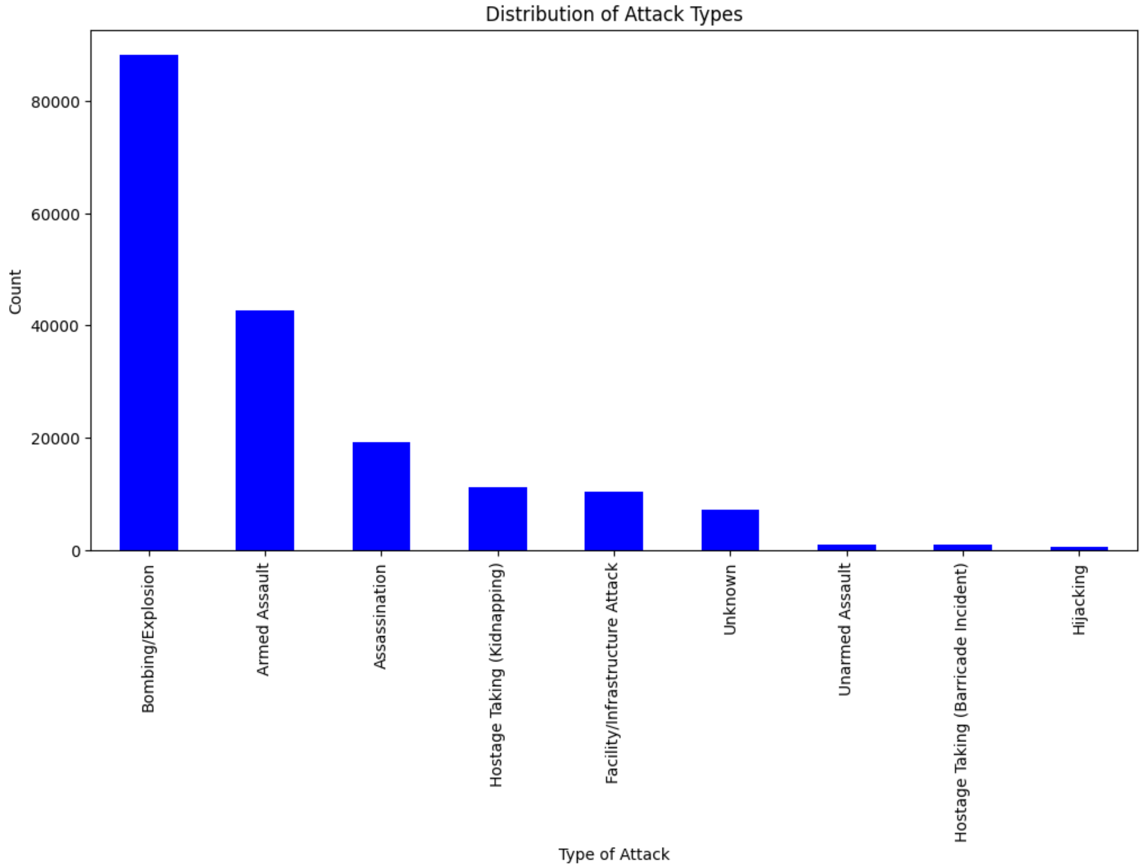
- The histogram depicts the distribution of terrorist events by year. We can observe that the frequency of terrorist events has increased significantly since the early 2000s, peaking around 2014.

In [ ]:
```
# Plot the top 10 countries with the highest number of terrorist events
plt.figure(figsize=(12,6))
df['Country_Txt'].value_counts().head(10).plot(kind='bar', color='blue')
plt.title('Top 10 Countries with Highest Number of Terrorist Events')
plt.xlabel('Country')
plt.ylabel('Count')
plt.xticks(rotation=90)
plt.show()
```



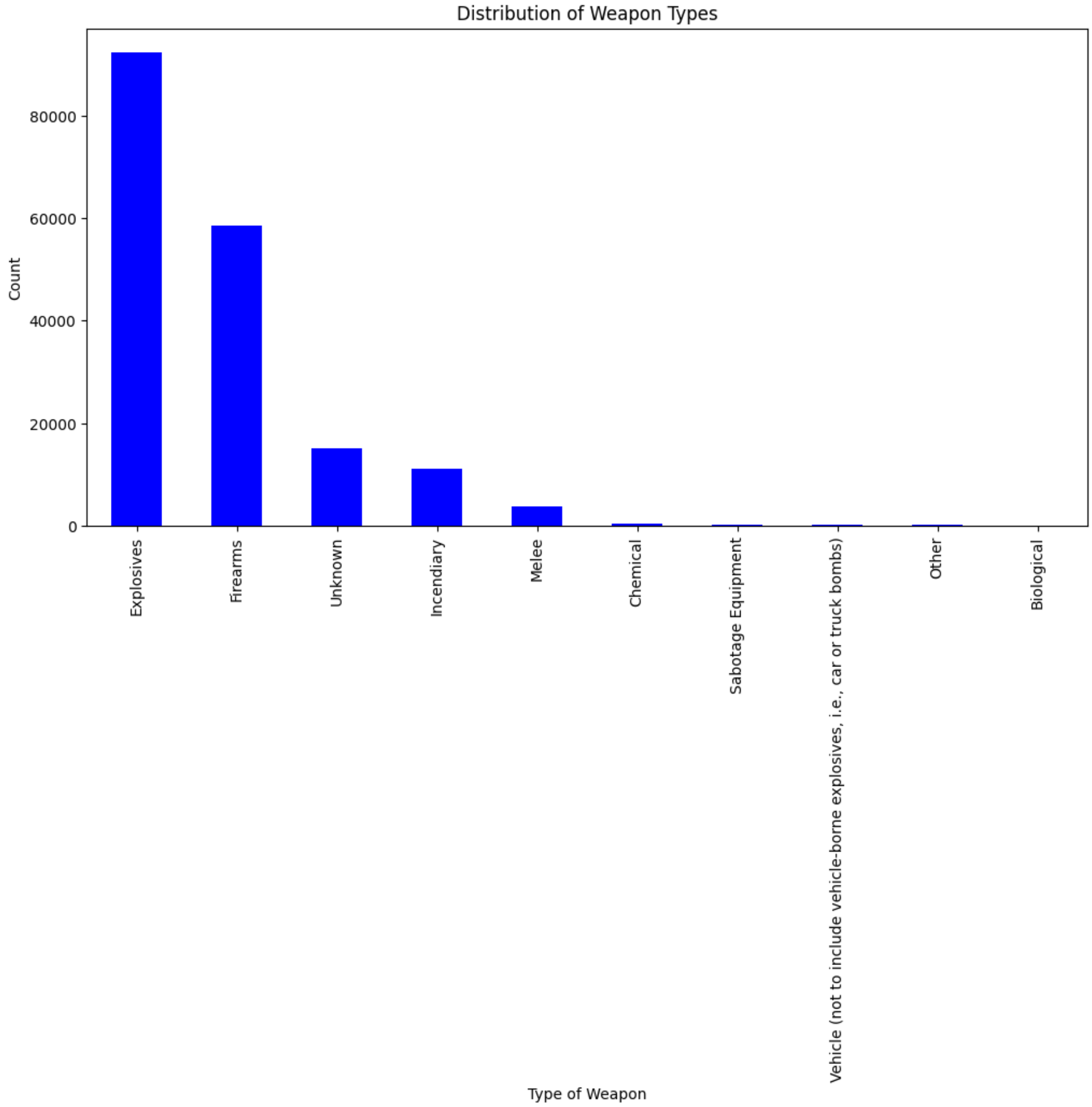
- The bar chart displays the top 10 countries with the highest number of recorded terrorist events. According to the data, Iraq, Pakistan, Afghanistan, and India are among the countries most affected by terrorism.

In [ ]:
```
# Plot the distribution of attack types
plt.figure(figsize=(12,6))
df['Attacktype1_Txt'].value_counts().plot(kind='bar', color='blue')
plt.title('Distribution of Attack Types')
plt.xlabel('Type of Attack')
plt.ylabel('Count')
plt.xticks(rotation=90)
plt.show()
```



- The bar chart displays the distribution of types of attacks. The most common types of attacks, according to the data, are bombings/explosions, armed assaults, and assassinations.

In [ ]:
```
# Plot the distribution of weapon types
plt.figure(figsize=(12,6))
df['Weaptype1_Txt'].value_counts().head(10).plot(kind='bar', color='blue')
plt.title('Distribution of Weapon Types')
plt.xlabel('Type of Weapon')
plt.ylabel('Count')
plt.xticks(rotation=90)
plt.show()
```



- The bar chart shows the distribution of the types of weapons used in these events. Explosives and firearms are the most commonly used weapons, according to the data.

Understanding the Distribution Graph

- The relationship between the year (iyear) and the country (country_txt), specifically for the top 5 countries with the highest number of terrorist events.
- The relationship between the type of attack (attacktype1_txt) and the type of weapon used (weaptype1_txt).

Temporal Analysis:

- How has the number of attacks evolved over time globally, and in the top 5 most affected countries?

In [ ]:
```
# Calculate the yearly number of terrorist attacks globally
global_attacks = df.groupby('Iyear')['Eventid'].count()
global_attacks
```

Out[ ]:
```
Iyear
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: Eventid, dtype: int64
```

In [ ]:
```
# Get the top 5 countries with the highest number of terrorist events
top_countries = df['Country_Txt'].value_counts().head(5).index
top_countries
```

Out[ ]:
```
Index(['Iraq', 'Pakistan', 'Afghanistan', 'India', 'Colombia'], dtype='object')
```

In [ ]:
```
# Filter the data for these top 5 countries
df_top_countries = df[df['Country_Txt'].isin(top_countries)]
df_top_countries
```

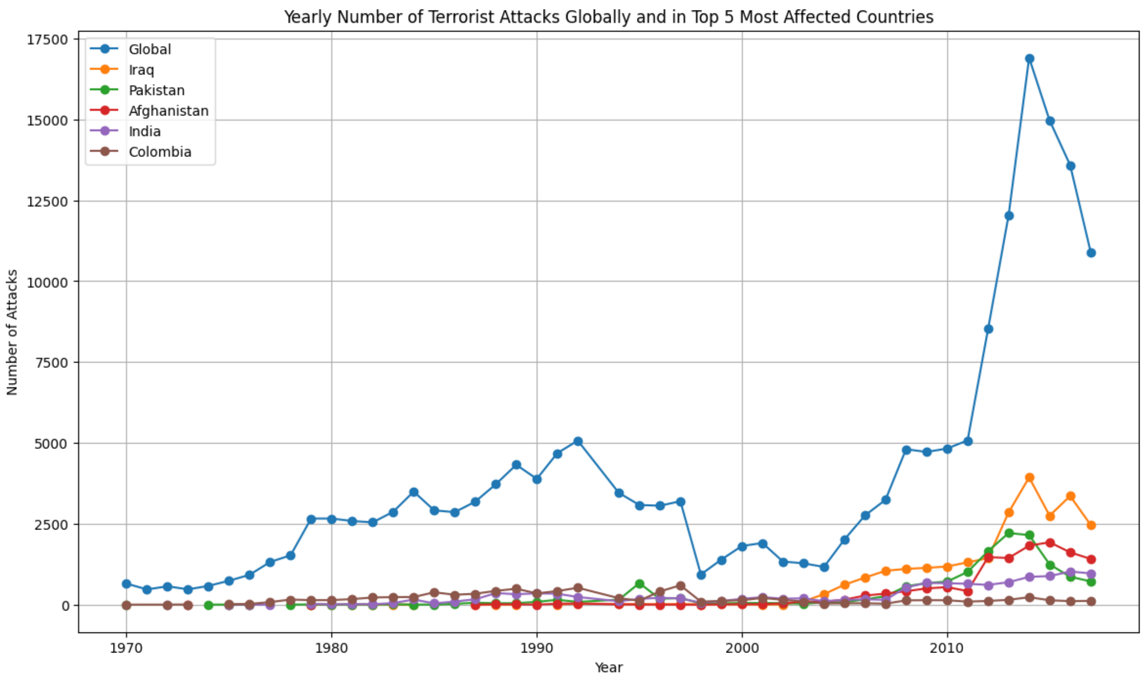
Out[ ]:

	Eventid	Iyear	Imonth	Iday	Extended	Country	Country_Txt	Region	Region_
405	197007100001	1970	7	10	0	45	Colombia	3	Sc Ame
585	197011010001	1970	11	1	0	153	Pakistan	6	South /
1186	197202220004	1972	2	22	0	92	India	6	South /
1392	197207170001	1972	7	17	0	45	Colombia	3	Sc Ame
1640	197212000001	1972	12	0	1	45	Colombia	3	Sc Ame
...	...	...	...	...	...	...	...	...	
181679	201712310012	2017	12	31	0	95	Iraq	10	Middle I & Nc Af
181683	201712310018	2017	12	31	0	4	Afghanistan	6	South /
181684	201712310019	2017	12	31	0	92	India	6	South /
181685	201712310020	2017	12	31	0	4	Afghanistan	6	South /
181689	201712310031	2017	12	31	0	92	India	6	South /

Eventid	Iyear	Imonth	Iday	Extended	Country	Country_Txt	Region	Region_

In [ ]:
```
# Calculate the yearly number of terrorist attacks in the top 5 most affected co
attacks_in_top_countries = df_top_countries.groupby(['Iyear', 'Country_Txt'])['E
```

In [ ]:
```
# Plot the yearly number of terrorist attacks
plt.figure(figsize=(14,8))
plt.plot(global_attacks.index, global_attacks.values, label='Global', marker='o'
for country in top_countries:
    plt.plot(attacks_in_top_countries.index, attacks_in_top_countries[country],
plt.title('Yearly Number of Terrorist Attacks Globally and in Top 5 Most Affecte
plt.xlabel('Year')
plt.ylabel('Number of Attacks')
plt.legend()
plt.grid()
plt.show()
```



What can we find with line chart

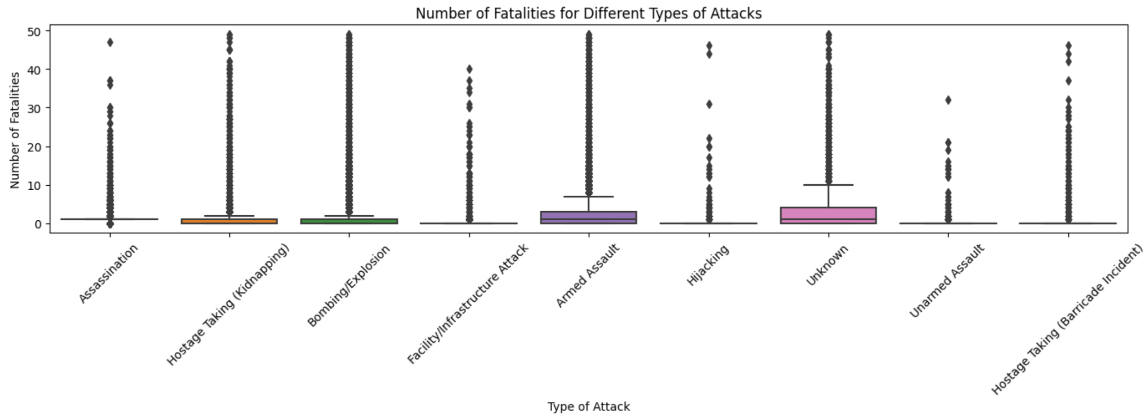
- Globally, the number of terrorist attacks has generally increased over time, with some fluctuations. There were particularly sharp increases in the late 1980s and early 1990s, and again in the 2010s.
- Iraq has seen a significant increase in terrorist attacks since 2003, likely related to the Iraq War and the rise of ISIS.
- Pakistan, Afghanistan, and India have also seen increases in terrorist attacks, although not as dramatic as in Iraq.
- The United States saw a spike in terrorist attacks around 2001, likely due to the 9/11 attacks, but has had relatively few attacks since then.

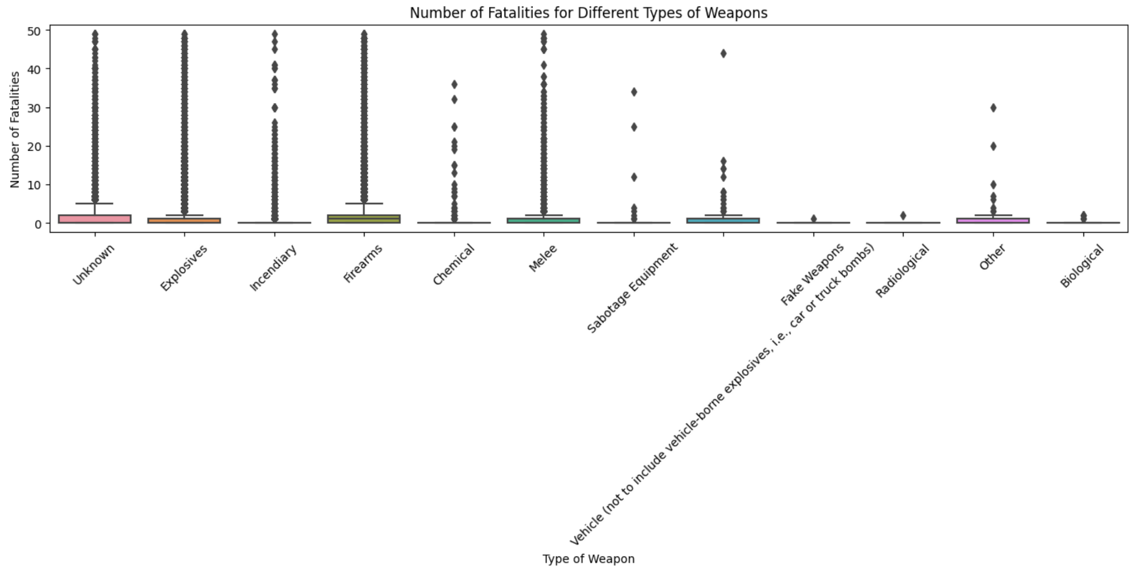
Casualties analysis

In [ ]:
```
# Display the names of all columns in the dataset
df.columns
```

Out[ ]:
```
Index(['Eventid', 'Iyear', 'Imonth', 'Iday', 'Extended', 'Country',
       'Country_Txt', 'Region', 'Region_Txt', 'Provstate', 'City', 'Latitude',
       'Longitude', 'Specificity', 'Vicinity', 'Summary', 'Crit1', 'Crit2',
       'Crit3', 'Doubtterr', 'Multiple', 'Success', 'Suicide', 'Attacktype1',
       'Attacktype1_Txt', 'Targtype1', 'Targtype1_Txt', 'Targsubtype1',
       'Targsubtype1_Txt', 'Corp1', 'Target1', 'Natlty1', 'Natlty1_Txt',
       'Gname', 'Guncertain1', 'Individual', 'Nperps', 'Nperpcap', 'Claimed',
       'Weaptype1', 'Weaptype1_Txt', 'Weapsubtype1', 'Weapsubtype1_Txt',
       'Weapdetail', 'Nkill', 'Nkillus', 'Nkillter', 'Nwound', 'Nwoundus',
       'Nwoundte', 'Property', 'Ishostkid', 'Scite1', 'Dbsource', 'Int_Log',
       'Int_Ideo', 'Int_Misc', 'Int_Any'],
      dtype='object')
```

In [ ]:
```
# Create box plots of the number of fatalities for different types of attacks an
fig, ax = plt.subplots(2, 1, figsize=(14, 12))

# Box plot for type of attack
sns.boxplot(data=df[df['Nkill'] < 50], x='Attacktype1_Txt', y='Nkill', ax=ax[0])
ax[0].set_title('Number of Fatalities for Different Types of Attacks')
ax[0].set_xlabel('Type of Attack')
ax[0].set_ylabel('Number of Fatalities')
ax[0].tick_params(axis='x', rotation=45)

# Box plot for type of weapon
sns.boxplot(data=df[df['Nkill'] < 50], x='Weaptype1_Txt', y='Nkill', ax=ax[1])
ax[1].set_title('Number of Fatalities for Different Types of Weapons')
ax[1].set_xlabel('Type of Weapon')
ax[1].set_ylabel('Number of Fatalities')
ax[1].tick_params(axis='x', rotation=45)

# Adjust the layout
plt.tight_layout()
plt.show()
```





What can we find with this box plot

Some insights from these plots are:

- For attack types, bombings/explosions, armed assault, and hijacking tend to result in a higher number of fatalities.
- For weapon types, explosives and firearms tend to result in a higher number of fatalities.

Group analysis

In [ ]:

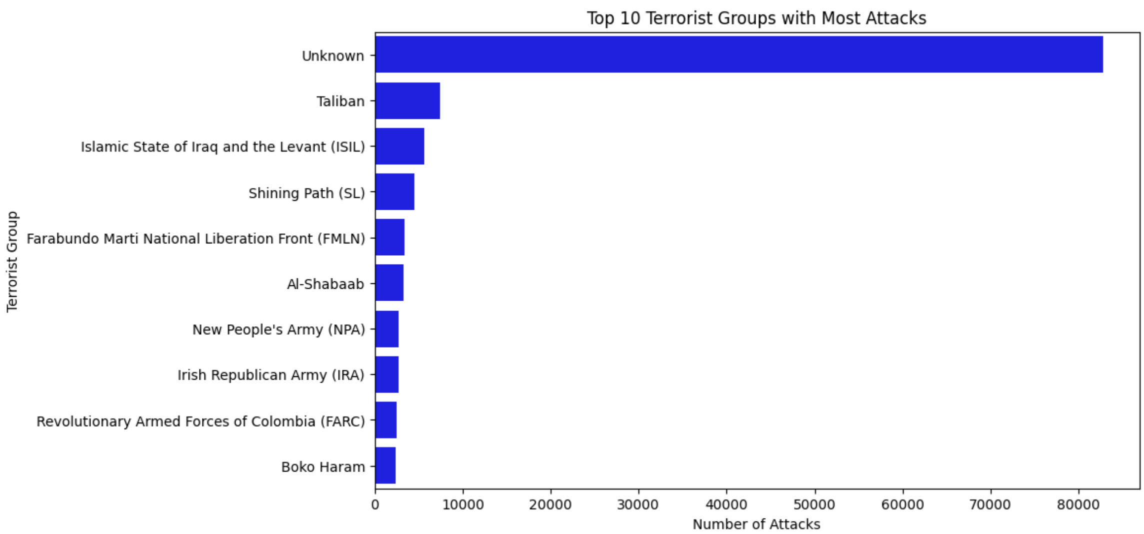
```
# Calculate the number of attacks for each terrorist group
group_attacks = df['Gname'].value_counts()
group_attacks
```

Out[ ]:
```
Unknown                                             82782
Taliban                                              7478
Islamic State of Iraq and the Levant (ISIL)          5613
Shining Path (SL)                                    4555
Farabundo Marti National Liberation Front (FMLN)     3351
                                                    ...
Ansar Sarallah                                          1
Sword of Islam                                          1
Support of Ocalan-The Hawks of Thrace                   1
Arab Revolutionary Front                                1
MANO-D                                                  1
Name: Gname, Length: 3537, dtype: int64
```

In [ ]:
```
# Get the top 10 terrorist groups with the most attacks
top_groups = group_attacks.head(10)
top_groups
```

Out[ ]:
```
Unknown                                             82782
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
Name: Gname, dtype: int64
```

In [ ]:
```
# Plot the number of attacks for the top 10 terrorist groups
plt.figure(figsize=(10,6))
sns.barplot(x=top_groups.values, y=top_groups.index, color='b', orient='h')
plt.title('Top 10 Terrorist Groups with Most Attacks')
plt.xlabel('Number of Attacks')
plt.ylabel('Terrorist Group')
plt.show()
```



Some insights from this plot are:

- The group labeled as "Unknown" has carried out the most attacks. This label is used when the group responsible for an attack could not be determined.

- Among the known groups, the Taliban, ISIL (Islamic State of Iraq and the Levant), and SL (Shining Path) have carried out the most attacks.

In [ ]:
```
# Get the top 5 types of attacks and weapons
top_attacks = df['Attacktype1_Txt'].value_counts().head(5).index
top_weapons = df['Weaptype1_Txt'].value_counts().head(5).index
```

In [ ]:
```
top_attacks
```

Out[ ]:
```
Index(['Bombing/Explosion', 'Armed Assault', 'Assassination',
       'Hostage Taking (Kidnapping)', 'Facility/Infrastructure Attack'],
      dtype='object')
```

In [ ]:
```
top_weapons
```

Out[ ]:
```
Index(['Explosives', 'Firearms', 'Unknown', 'Incendiary', 'Melee'], dtype='obje
ct')
```

In [ ]:
```
# Filter the data for these top 5 types of attacks and weapons
df_top_attacks_weapons = df[df['Attacktype1_Txt'].isin(top_attacks) & df['Weapty
df_top_attacks_weapons
```

Out[ ]:

	Eventid	Iyear	Imonth	Iday	Extended	Country	Country_Txt	Region	Region_
0	197000000001	1970	7	2	0	58	Dominican Republic	2	Cer Americ Caribb
1	197000000002	1970	0	0	0	130	Mexico	1	No Ame
2	197001000001	1970	1	0	0	160	Philippines	5	Southe /
3	197001000002	1970	1	0	0	78	Greece	8	West Eur
4	197001000003	1970	1	0	0	101	Japan	4	East /
...	...	...	...	...	...	...	...	...	
181686	201712310022	2017	12	31	0	182	Somalia	11	S Saha Af
181687	201712310029	2017	12	31	0	200	Syria	10	Middle I & No Af
181688	201712310030	2017	12	31	0	160	Philippines	5	Southe /
181689	201712310031	2017	12	31	0	92	India	6	South /
181690	201712310032	2017	12	31	0	160	Philippines	5	Southe /

171374 rows × 58 columns

In [ ]:

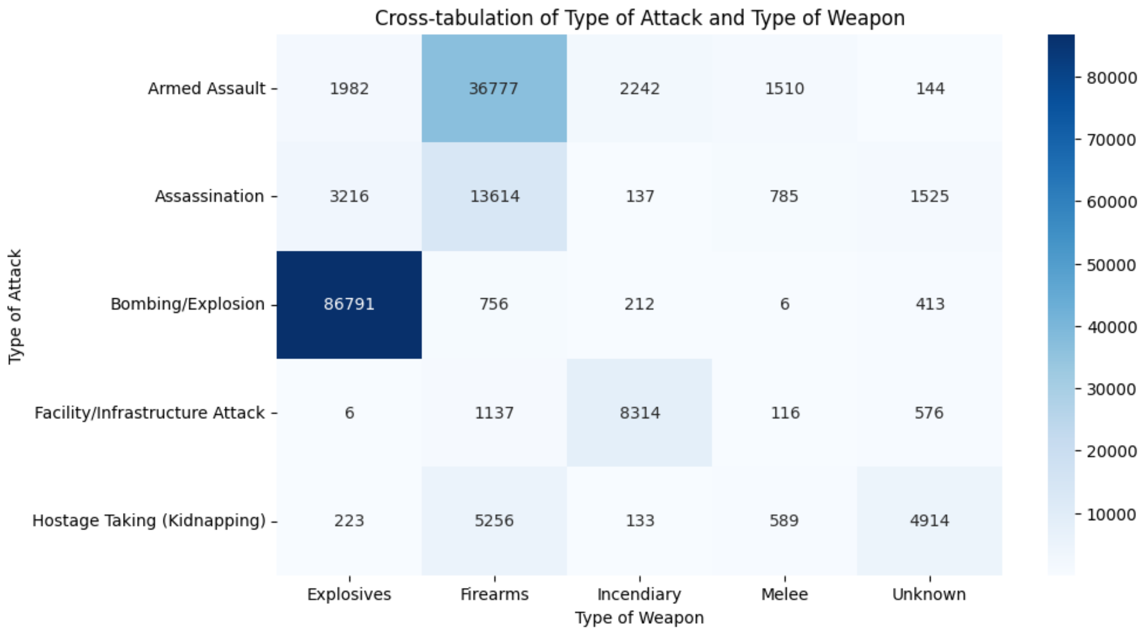
```
# Create a cross-tabulation of the type of attack and the type of weapon
cross_tab = pd.crosstab(df_top_attacks_weapons['Attacktype1_Txt'], df_top_attack
cross_tab
```

Out[ ]:

Weaptype1_Txt	Explosives	Firearms	Incendiary	Melee	Unknown
Attacktype1_Txt					
Armed Assault	1982	36777	2242	1510	144
Assassination	3216	13614	137	785	1525
Bombing/Explosion	86791	756	212	6	413
Facility/Infrastructure Attack	6	1137	8314	116	576
Hostage Taking (Kidnapping)	223	5256	133	589	4914

In [ ]:

```
# Plot the cross-tabulation as a heatmap
plt.figure(figsize=(10,6))
sns.heatmap(cross_tab, annot=True, fmt='d', cmap='Blues')
plt.title('Cross-tabulation of Type of Attack and Type of Weapon')
plt.xlabel('Type of Weapon')
plt.ylabel('Type of Attack')
plt.show()
```



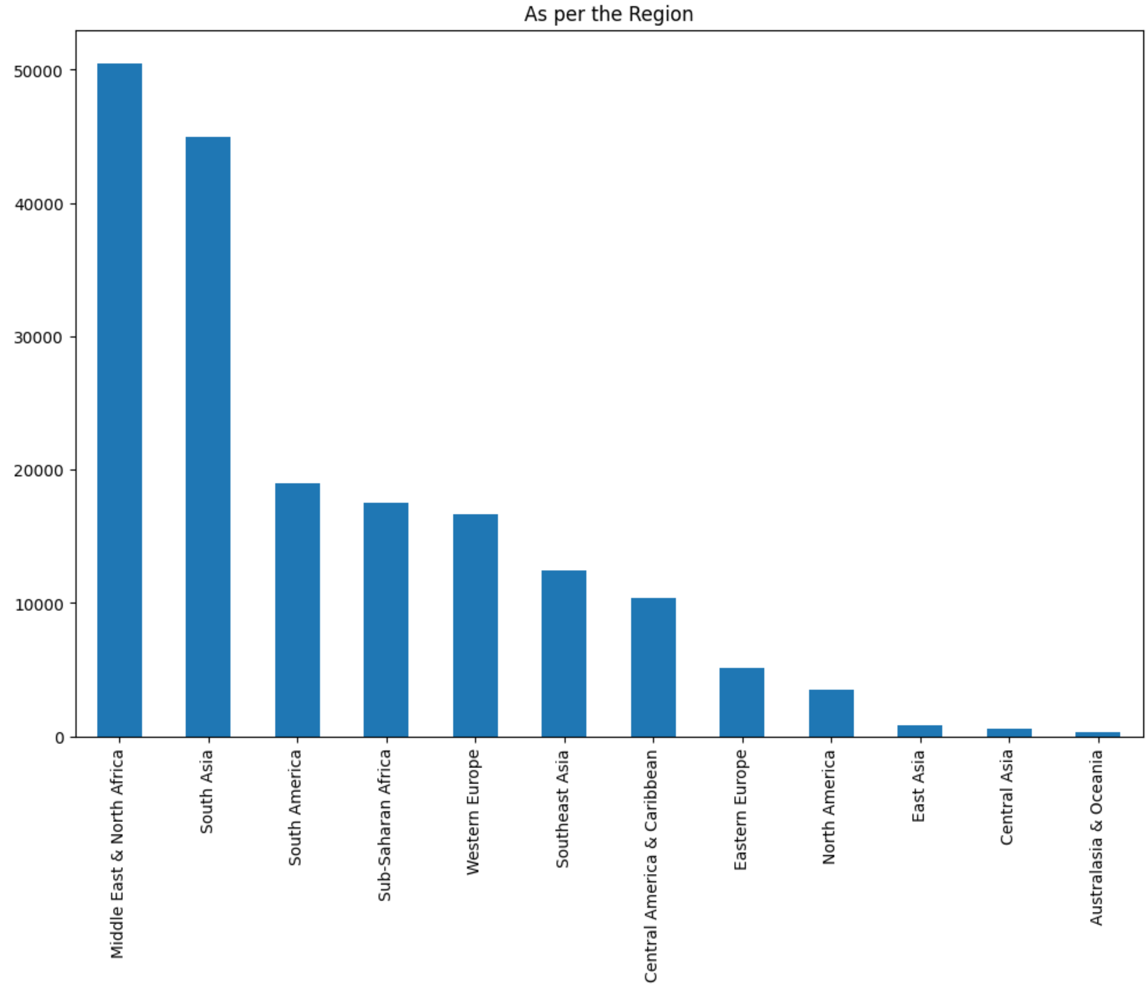
This heatmap represents the cross-tabulation of the types of attacks and the types of weapons. Each cell in the heatmap shows the number of terrorist events for a specific combination of attack type and weapon type. The darker the color, the higher the number of events.

Here are some insights from the heatmap:

- Explosives are most commonly used in bombings and explosions. This is expected because the very definition of these types of attacks involves the use of explosives.
- Firearms are frequently used in armed assaults and assassinations. This is also intuitive as these types of attacks often involve direct confrontation with the targets.
- Incendiary devices are often used in facility and infrastructure attacks, which might involve arson or other methods to cause property damage.

In [ ]:
```
plt.figure(figsize=(12,8))

# ploting the graff with Region
df['Region_Txt'].value_counts().plot(kind='bar', title='As per the Region')
```

Out[ ]: <Axes: title={'center': 'As per the Region'}>



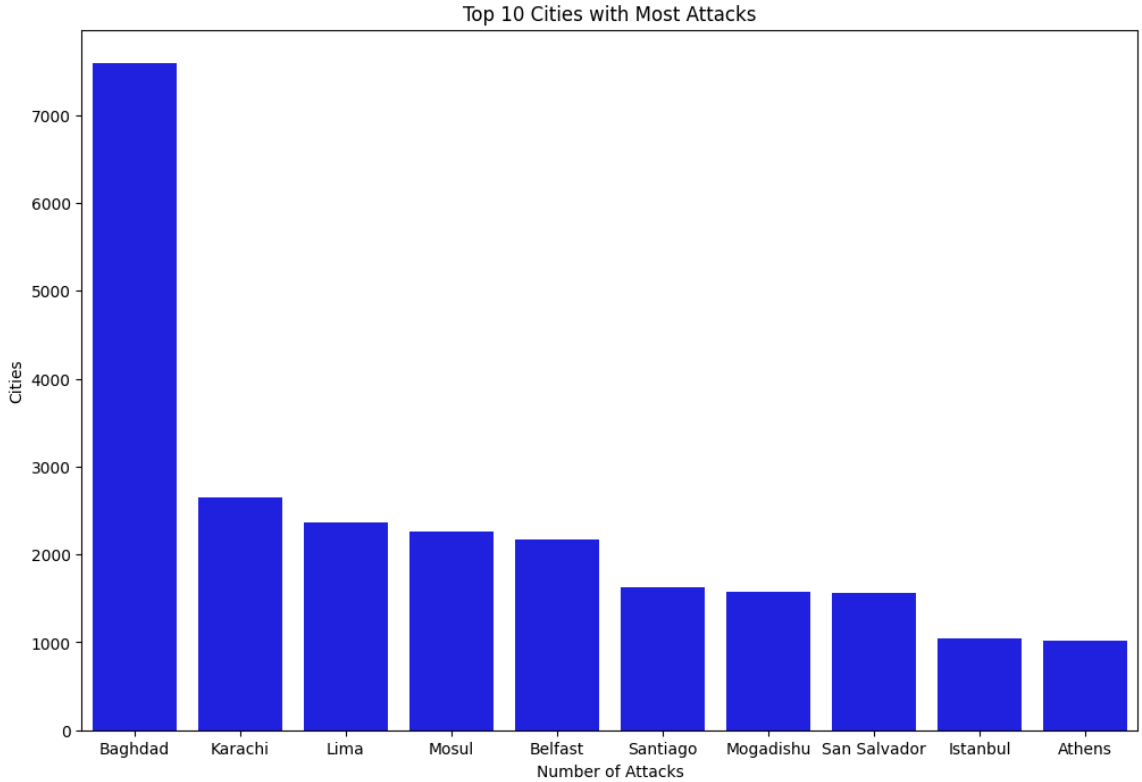
In [ ]:
```
# counting the city
df['City'].value_counts()
```

Out[ ]:
```
Unknown      10209
Baghdad       7589
Karachi       2652
Lima          2359
Mosul         2265
             ...
Hotwag           1
Ostend           1
Balughata        1
Jikoyi           1
Kubentog         1
Name: City, Length: 36674, dtype: int64
```

In [ ]:
```
# filter the top 10 city who faced most terrorist attack in the world
top10_city = df['City'].value_counts()[1:].head(10)
top10_city
```

Out[ ]:
```
Baghdad         7589
Karachi         2652
Lima            2359
Mosul           2265
Belfast         2171
Santiago        1621
Mogadishu       1581
San Salvador    1558
Istanbul        1048
Athens          1019
Name: City, dtype: int64
```

In [ ]:
```
# Plot the number of attacks for the top 10 terrorist groups
plt.figure(figsize=(12,8))
sns.barplot(x=top10_city.index, y=top10_city.values, color='b')
plt.title('Top 10 Cities with Most Attacks')
plt.xlabel('Number of Attacks')
plt.ylabel('Cities')
plt.show()
```



What we get from this chart

- Baghdad and Karachi has faced most attacked

In [ ]:
```
# Get the top 5 regions and weapons
top_regions = df['Region'].value_counts().head(5).index
top_weapons = df['Weaptype1_Txt'].value_counts().head(5).index
```

In [ ]:
```
top_regions
```

Out[ ]:
```
Int64Index([10, 6, 3, 11, 8], dtype='int64')
```

In [ ]:
```
top_weapons
```

Out[ ]:
```
Index(['Explosives', 'Firearms', 'Unknown', 'Incendiary', 'Melee'], dtype='obje
ct')
```

In [ ]:
```
# Filter the data for these top 5 regions and weapons
df_top_regions_weapons = df[df['Region'].isin(top_regions) & df['Weaptype1_Txt']
df_top_regions_weapons
```

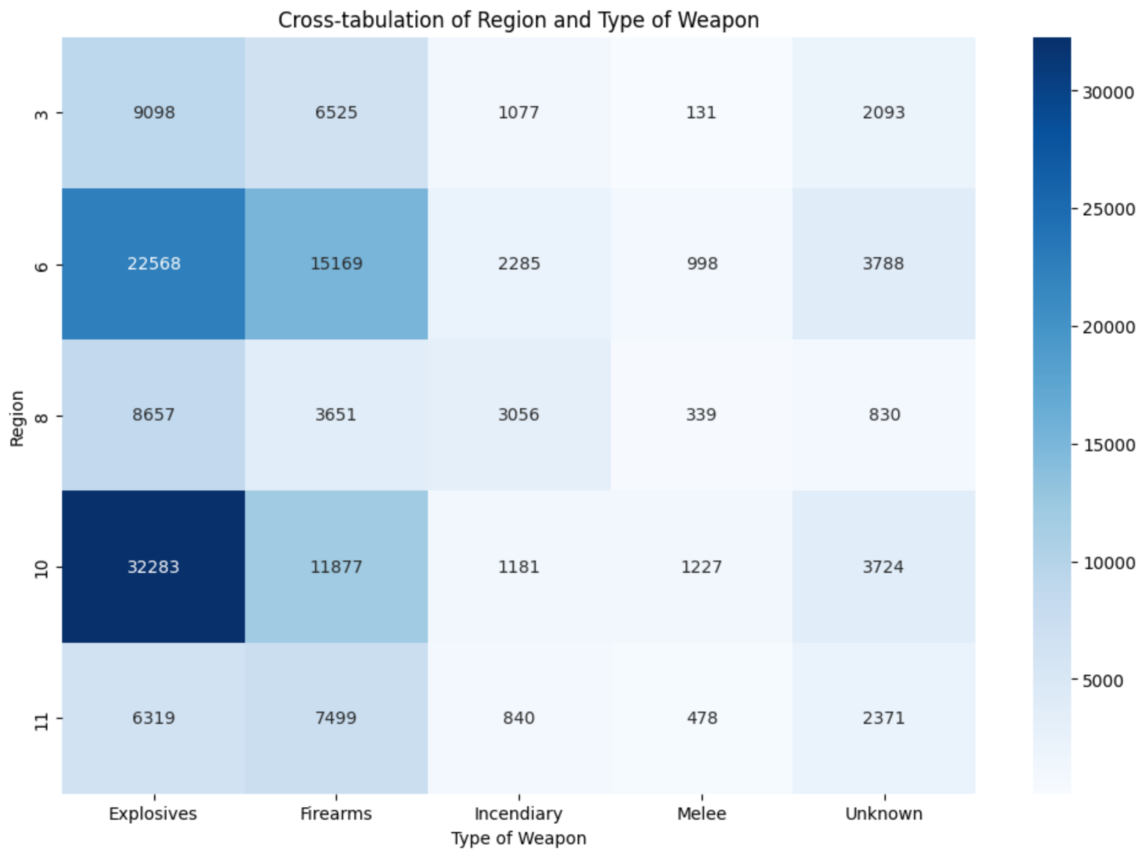
Out[ ]:

	Eventid	Iyear	Imonth	Iday	Extended	Country	Country_Txt	Region	Region_
3	197001000002	1970	1	0	0	78	Greece	8	West Euro
6	197001020001	1970	1	2	0	218	Uruguay	3	So Ame
12	197001080001	1970	1	8	0	98	Italy	8	West Euro
16	197001110001	1970	1	11	0	65	Ethiopia	11	S Saha Af
21	197001150001	1970	1	15	0	218	Uruguay	3	So Ame
...	...	...	...	...	...	...	...	...	
181684	201712310019	2017	12	31	0	92	India	6	South A
181685	201712310020	2017	12	31	0	4	Afghanistan	6	South A
181686	201712310022	2017	12	31	0	182	Somalia	11	S Saha Af
181687	201712310029	2017	12	31	0	200	Syria	10	Middle I & No Af
181689	201712310031	2017	12	31	0	92	India	6	South A

148064 rows × 58 columns

In [ ]:
```
# Create a cross-tabulation of the region and the type of weapon
cross_tab = pd.crosstab(df_top_regions_weapons['Region'], df_top_regions_weapons
```

In [ ]:
```
# Plot the cross-tabulation as a heatmap
plt.figure(figsize=(12,8))
sns.heatmap(cross_tab, annot=True, fmt='d', cmap='Blues')
plt.title('Cross-tabulation of Region and Type of Weapon')
plt.xlabel('Type of Weapon')
plt.ylabel('Region')
plt.show()
```



What we get from this heatmap

Here are some insights from the heatmap:

- In the Middle East and North Africa, explosives are by far the most commonly used weapon. This might be related to the prevalence of certain types of attacks in this region, such as bombings.
- Firearms are widely used in South Asia, Sub-Saharan Africa, and South America, although not as extensively as explosives in the Middle East and North Africa.
- Incendiary weapons appear to be more commonly used in Western Europe compared to other regions. This could reflect the fact that these weapons can cause significant disruption and attract media attention, which might be a strategic goal of terrorist groups operating in these regions.
- The category 'Unknown' for weapon type is most prevalent in South America. This might suggest that information about the type of weapon used is often unavailable or not reported for incidents in this region.

In [ ]:
```
df.sample(2)
```

Out[ ]:

	Eventid	Iyear	Imonth	Iday	Extended	Country	Country_Txt	Region	Region_
114530	201302170017	2013	2	17	0	205	Thailand	5	Southe A
86419	200804150031	2008	4	15	0	205	Thailand	5	Southe A