TASK : 01

Exploratory Data Analysis On "Global Terrorism Dataset"

Problem Statement : As a Security/Defense Analyst, try to find out the hot zone of terrorism.

Name : Rahul Gayaval

```
# Preparing all the necessary libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.simplefilter("ignore")

# Reading the data

data = pd.read_csv("globalterrorismdb.csv", encoding="latin1")

# Elaborating the data

data.head()
```

	eventid	iyear	imonth	iday	approach	extended	resolution	country	...	country_txt	region	...	addnotes	scite1	scite2	scite3	dbsource	INT_LOG	INT_IDEO	INT_MISC	INT_A	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	0	0	0	0	NaN
3	197001000002	1970	1	0	NaN	0	NaN	78	...	Greece	8	...	NaN	NaN	NaN	NaN	PGIS	0	1	1	1	NaN
4	197001000003	1970	1	0	NaN	0	NaN	101	...	Japan	4	...	NaN	NaN	NaN	NaN	PGIS	0	0	0	0	NaN

5 rows × 135 columns

Let's see how many rows and columns are there in the dataset.

```
print("There are {} rows and {} columns in this dataset".format(data.shape[0],data.shape[1]))
```

```
There are 181691 rows and 135 columns in this dataset
```

Now let's see the total columns and their names in the dataset.

```
data.columns.tolist()
```

```
['eventid', 'iyear', 'imonth', 'iday', 'approach', 'extended',
 'resolution', 'country', 'country_txt', 'region',
 'region_txt', 'provstate', 'city', 'latitude', 'longitude',
 'specificity', 'vicinity', 'location', 'summary', 'crit1', 'crit2',
 'crit3', 'doubtterr', 'alternative', 'alternative_txt',
 'multinat', 'targtype1', 'targtype1_txt', 'targtype2',
 'targtype2_txt', 'targtype3', 'targtype3_txt', 'target1',
 'target2', 'target3', 'corp1', 'corp2', 'corp3', 'target1_txt',
 'target2_txt', 'target3_txt', 'natlty1', 'natlty1_txt',
 'natlty2', 'natlty2_txt', 'natlty3', 'natlty3_txt', 'gsubtype1',
 'gsubtype1_txt', 'gsubtype2', 'gsubtype2_txt', 'gsubtype3',
 'gsubtype3_txt', 'guerrtype1', 'guerrtype1_txt', 'guerrtype2',
 'guerrtype2_txt', 'guerrtype3', 'guerrtype3_txt', 'peinagg',
 'peinjur', 'peinjurtx', 'peinjurlt', 'peinjurus', 'peinjurtot',
 'property', 'propextent', 'propextent_txt', 'propvalue',
 'propcomment', 'ishostkid', 'nhostkid', 'nhostkidus',
 'nhours', 'ransom', 'ransomamt', 'ransomamtus',
 'ransomamtus2', 'ransomdur', 'ransomdur2', 'ransomoth',
 'ransomto', 'ransom2', 'ransomto2', 'ransomoth2',
 'weapeventid', 'weapindate', 'weaptype1', 'weaptype1_txt',
 'weapsubtype1', 'weapsubtype1_txt', 'weapdetail',
 'nkill', 'nkillus', 'nkillter', 'nwound', 'nwoundus',
 'nwoundte', 'property1', 'property1_txt', 'ransomeventid',
 'hostkidevent', 'ishostkid2', 'hostage', 'hostage2',
 'addnotes', 'scite1', 'scite2', 'scite3', 'dbsource',
 'INT_LOG', 'INT_IDEO', 'INT_MISC', 'INT_ANY', 'related']
```

Let's see how many unique columns this dataset contains.

```
total_columns = data.columns.shape[0]
print("There are {} unique columns in the data".format(total_columns))
```

```
There are total 135 unique columns in the data
```

We will use the following columns for the Analysis :

1. Year
2. Month
3. Day
4. Country
5. Region
6. City
7. Attack Type
8. Target Type
9. Group Name
10. Weapon Type
11. Gang name

Renaming the above mentioned columns.

```
data = data.rename(columns={'iyear' : 'Year',
                            'imonth' : 'Month',
                            'iday' : 'Day',
                            'country_txt' : 'Country',
                            'region_txt' : 'Region',
                            'provstate' : 'Province',
                            'city' : 'City',
                            'attack' : 'Attack',
                            'attack_type' : 'Attack_Type',
                            'target1' : 'Target_Type',
                            'gsubtype1_txt' : 'Group_Name',
                            'gsubtype2_txt' : 'Gang_Name',
                            'weap' : 'Weapon_Type'
                            'weapdetail' : 'Weapon_Type'
                            }
data
```

	eventid	Year	Month	Day	approach	extended	resolution	country	Country	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	0	0	0	0	NaN
3	197001000002	1970	1	0	NaN	0	NaN	78	Greece	8	...	NaN	NaN	NaN	NaN	PGIS	0	1	1	1	NaN
4	197001000003	1970	1	0	NaN	0	NaN	101	Japan	4	...	NaN	NaN	NaN	NaN	PGIS	0	0	0	0	NaN
181690	201512150002	2015	12	31	NaN	0	NaN	182	Somalia	11	...	NaN	"Somalia Al-Shabaab Militants Attack Army Base."	"Highlights Sunday Daily Media Highlights"	"Highlights Sunday Daily Media Highlights"	START Primary Collection	0	0	1	1	NaN
181691	201512150004	2015	12	31	NaN	0	NaN	200	Syria	10	...	NaN	"Putin's 'victory' in Syria has turned into a quagmire"	"Two Russian soldiers killed as militants blast..."	"Two Russian servicemen killed in Syria mortar..."	START Primary Collection	0	0	1	1	NaN
181692	201512150005	2015	12	31	NaN	0	NaN	78	Greece	8	...	NaN	"Megalomania claims Day One overdoes: PM."	NaN	NaN	START Primary Collection	0	0	1	1	NaN
181693	201512150006	2015	12	31	NaN	0	NaN	92	India	6	...	NaN	"Textile escapes grenade attack in Imphal" Nov.	NaN	NaN	START Primary Collection	0	0	1	1	NaN
181694	201512150007	2015	12	31	NaN	0	NaN	160	Philippines	5	...	NaN	"Security tightened in Cotabato City" Manila	"Security tightened in Cotabato City" Manila	"Security tightened in Cotabato City" Manila	START Primary Collection	0	0	0	0	NaN

181691 rows × 135 columns

As you can see in the data, there are a lot of NaN values. Let's fill these NaN values with 'Others' and 'Unknown'

```
data['Target_subtype1'] = data['Target_subtype1_txt].fillna('Others')

# Filling the NaN values in attack_type as Unknown

data['Target'] = data['Target'].fillna('Unknown')

# Filling the NaN values in city column as Unknown

data['City'] = data['City'].fillna('Unknown')

# Replacing the NaN values in gsubtype1_txt column with "Unknown"

data['Group_Name'] = data['Group_Name'].fillna('Unknown')
```

```
data
```

	eventid	Year	Month	Day	approach	extended	resolution	country	Country	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	0	0	0	0	NaN
3	197001000002	1970	1	0	NaN	0	NaN	78	Greece	8	...	NaN	NaN	NaN	NaN	PGIS	0	1	1	1	NaN
4	197001000003	1970	1	0	NaN	0	NaN	101	Japan	4	...	NaN	NaN	NaN	NaN	PGIS	0	0	0	0	NaN
181690	201512150002	2015	12	31	NaN	0	NaN	182	Somalia	11	...	NaN	"Somalia Al-Shabaab Militants Attack Army Base."	"Highlights Sunday Daily Media Highlights"	"Highlights Sunday Daily Media Highlights"	START Primary Collection	0	0	1	1	NaN
181691	201512150004	2015	12	31	NaN	0	NaN	200	Syria	10	...	NaN	"Putin's 'victory' in Syria has turned into a quagmire"	"Two Russian soldiers killed as militants blast..."	"Two Russian servicemen killed in Syria mortar..."	START Primary Collection	0	0	1	1	NaN
181692	201512150005	2015	12	31	NaN	0	NaN	78	Greece	8	...	NaN	"Megalomania claims Day One overdoes: PM."	NaN	NaN	START Primary Collection	0	0	1	1	NaN
181693	201512150006	2015	12	31	NaN	0	NaN	92	India	6	...	NaN	"Textile escapes grenade attack in Imphal" Nov.	NaN	NaN	START Primary Collection	0	0	1	1	NaN
181694	201512150007	2015	12	31	NaN	0	NaN	160	Philippines	5	...	NaN	"Security tightened in Cotabato City" Manila	"Security tightened in Cotabato City" Manila	"Security tightened in Cotabato City" Manila	START Primary Collection	0	0	0	0	NaN

181691 rows × 135 columns

Checking how many columns are left with NaN values so that we can drop it.

```
# Checking number of NaN values
print(data.isnull())
```

```
        eventid  iyear  imonth  iday  approach  extended  resolution  \
0         False  False   False  False      True     False        True
1         False  False   False  False      True     False        True
2         False  False   False  False      True     False        True
3         False  False   False  False      True     False        True
4         False  False   False  False      True     False        True
...         ...    ...     ...    ...       ...       ...         ...
181690    False  False   False  False      True     False        True
181691    False  False   False  False      True     False        True
181692    False  False   False  False      True     False        True
181693    False  False   False  False      True     False        True
181694    False  False   False  False      True     False        True

        country  country_txt  region  ...  addnotes  scite1  scite2  \
0         False        False   False  ...      True    True    True
1         False        False   False  ...      True    True    True
2         False        False   False  ...      True    True    True
3         False        False   False  ...      True    True    True
4         False        False   False  ...      True    True    True
...         ...          ...     ...  ...       ...     ...     ...
181690    False        False   False  ...      True    True    True
181691    False        False   False  ...      True    True    True
181692    False        False   False  ...      True    True    True
181693    False        False   False  ...      True    True    True
181694    False        False   False  ...      True    True    True

        scite3  dbsource  INT_LOG  INT_IDEO  INT_MISC  INT_ANY  related
0        False     False    False     False     False    False     True
1        False     False    False     False     False    False     True
2        False     False    False     False     False    False     True
3        False     False    False     False     False    False     True
4        False     False    False     False     False    False     True
...        ...       ...      ...       ...       ...      ...      ...
181690   False     False    False     False     False    False     False
181691   False     False    False     False     False    False     False
181692   False     False    False     False     False    False     False
181693   False     False    False     False     False    False     False
181694   False     False    False     False     False    False     False

[181691 rows x 135 columns]
```

```
for col in data:
    if data[col] == True:
        data[col] = data[col]
```

```
print(data.isnull().sum())
```

```
eventid            0
iyear              0
imonth             0
iday               0
approach      166969
extended           0
resolution    179565
country            0
country_txt        0
region             0
region_txt         0
provstate        421
city             434
latitude        4606
longitude       4608
specificity        5
vicinity           0
location       96996
summary        66135
crit1              0
crit2              0
crit3              0
doubtterr        788
alternative    165567
alternative_txt
multinat       163733
targtype1          0
targtype1_txt      0
targtype2      111780
targtype2_txt  111780
targtype3      166602
targtype3_txt  166602
target1         14078
corp1          126320
target1_txt        0
natlty1         12628
natlty1_txt     12628
gsubtype1       84627
gsubtype1_txt   84627
weapsubtype1    84632
weapsubtype1_txt
property           0
ishostkid        421
INT_LOG            0
INT_IDEO           0
INT_MISC           0
INT_ANY            0
dtype: int64
```

Let's check the shape

```
data.shape
```

```
(181691, 135)
```

Let's look at the unique value of the columns now

```
data.columns.unique()
```

```
Index(['eventid', 'iyear', 'imonth', 'iday', 'approach', 'extended',
       'resolution', 'country', 'country_txt', 'region',
       'region_txt', 'provstate', 'city', 'latitude', 'longitude',
       'specificity', 'vicinity', 'location', 'summary', 'crit1',
       'crit2', 'crit3', 'doubtterr', 'alternative', 'alternative_txt',
       'multinat', 'targtype1', 'targtype1_txt', 'targtype2',
       'targtype2_txt', 'targtype3', 'targtype3_txt', 'target1',
       'target1_txt', 'corp1', 'natlty1', 'natlty1_txt',
       'gsubtype1', 'gsubtype1_txt', 'weapsubtype1',
       'weapsubtype1_txt', 'property', 'ishostkid', 'INT_LOG',
       'INT_IDEO', 'INT_MISC', 'INT_ANY', 'related'],
      dtype='object')
```

Number of countries in the dataset

```
total_countries = data['Country'].nunique()
print("There are {} countries in the dataset".format(total_countries))
```

```
There are total 205 countries in the dataset
```

Number of Regions in the dataset

```
total_regions = data['Region'].nunique()
print("There are {} regions in the dataset".format(total_regions))
```

```
There are 12 regions in the dataset
```

Basic Statistics

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 181691 entries, 0 to 181694
Data columns (total 135 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   approach      14722 non-null   float64
 5   extended      181691 non-null  int64
 6   resolution    2126 non-null    object
 7   country       181691 non-null  int64
 8   country_txt   181691 non-null  object
 9   region        181691 non-null  int64
 10  region_txt    181691 non-null  object
 11  provstate     181270 non-null  object
 12  city          181257 non-null  object
 13  latitude      177085 non-null  float64
 14  longitude     177083 non-null  float64
 15  specificity   181686 non-null  int64
 16  vicinity      181691 non-null  int64
 17  location      84695 non-null   object
 18  summary       115556 non-null  object
 19  crit1         181691 non-null  int64
 20  crit2         181691 non-null  int64
 21  crit3         181691 non-null  int64
 22  doubtterr     180903 non-null  float64
 23  alternative   16124 non-null   object
 24  alternative_txt 16124 non-null object
 25  multinat      18158 non-null   object
 26  targtype1     181691 non-null  int64
 27  targtype1_txt 181691 non-null  object
 28  targtype2     69911 non-null   float64
 29  targtype2_txt 69911 non-null   object
 30  targtype3     15089 non-null   float64
 31  targtype3_txt 15089 non-null   object
 32  target1       167613 non-null  object
 33  corp1         55371 non-null   object
 34  target1_txt   181691 non-null  object
 35  natlty1       169063 non-null  float64
 36  natlty1_txt   169063 non-null  object
 37  gsubtype1     97064 non-null   float64
 38  gsubtype1_txt 97064 non-null   object
 39  weapsubtype1  97059 non-null   float64
 40  weapsubtype1_txt 97059 non-null object
 41  property      181691 non-null  int64
 42  ishostkid     181270 non-null  float64
 43  INT_LOG       181691 non-null  int64
 44  INT_IDEO      181691 non-null  int64
 45  INT_MISC      181691 non-null  int64
 46  INT_ANY       181691 non-null  int64
 47  related       91839 non-null   object
dtype: int64
```

Top 10 Countries

```
data['Country'].value_counts().head(10)
```

```
Iraq            24636
Pakistan        14368
Afghanistan     12731
India           11960
colombia        8306
Philippines     6908
Peru            6096
El Salvador     5320
United Kingdom  5235
Turkey          4292
Name: Country, dtype: int64
```

Top 10 Regions

```
data['Region'].value_counts().head(10)
```

```
Middle East & North Africa     50474
South Asia                     44974
South America                  18978
Sub-Saharan Africa             13915
Western Europe                 13444
Southeast Asia                 12323
Central America & Caribbean    10344
Eastern Europe                  8916
North America                   7731
East Asia                       7530
Name: Region, dtype: int64
```

Top 10 States

```
data['Province'].value_counts().head(10)
```

```
Baghdad            9536
Saladin            4561
Northern Ireland   4458
Al Anbar           4297
Nineveh            3917
Diyala             3466
Arauca             3331
Sinaloa            3239
Khyber Pakhtunkhwa 3154
Punjab             3151
Name: Province, dtype: int64
```

Attack types

```
data['Attack_Type'].value_counts().head(10)
```

```
Bombing/Explosion                      88253
Armed Assault                          42507
Assassination                          41980
Hostage Taking (Kidnapping)            11836
Hostage Taking (Barricade Incident)     9914
Unknown                                 9319
Facility/Infrastructure Attack          8654
Hostage Taking (Hostage Incident)       6591
Unarmed Assault                         1075
Name: Attack_Type, dtype: int64
```

Target Types

```
data['Target_Type'].value_counts().head(10)
```

```
Private Citizens & Property       27453
Military                          22423
Police                            21549
Government (General)              16635
Business                          15562
Transportation                     8749
Utilities                          5929
Religious                          5784
Terrorists/Non-State Militia       2557
Educational Institution            2521
Name: Target_Type, dtype: int64
```

Top 10 Gangs that were involved.

```
data['Gang_Name'].value_counts().head(10)
```

```
Unknown                                            82782
Taliban                                             7478
Shiite extremists this and that the sentiment       3290
Jemaat Islamiya                                     3092
Revolutionary Armed Forces of Colombia (FARC)       2671
Al-Shabaab                                          2652
New People's Army (NPA)                             2772
Liberation Tigers of Tamil Eelam (LTTE)             2393
Revolutionary Armed Forces of Colombia (FARC)       2371
Al-Qaeda                                            2258
Name: Gang_Name, dtype: int64
```

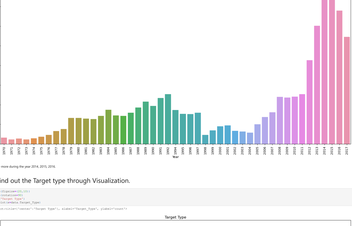
Top 10 Weapons that were used.

```
data['Weapon_Type'].value_counts().head(10)
```

```
Explosives          92424
Firearms            58524
Unknown             15157
Incendiary          12147
Other               3855
Melee               3455
Chemical            1307
Sabotage Equipment  1140
Vehicle             1130
Biological          140
Name: Weapon_Type, dtype: int64
```

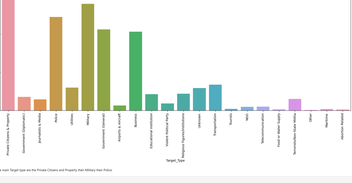
Now let us find out the hot zones of attack through visualization.

```
plt.figure(figsize=(8,5))
sns.countplot(data['Weapon_Type'],
              palette='rocket')
plt.xticks(rotation=45,fontsize=7,
           fontweight='bold')
plt.title('Attacks By Weapon',
          fontsize=20,fontweight='bold',
          color='black')
```



Most of the attacks had done through 'Explosives' and 'Firearms'

```
plt.figure(figsize=(8,5))
sns.countplot(data['Year'],
              palette='rocket')
plt.xticks(rotation=45,fontsize=7,
           fontweight='bold')
plt.title('Attacks By Year',
          fontsize=20,fontweight='bold',
          color='black')
```



Attacks were more during the year 2014, 2015, 2016.

Let's find out the Target type through Visualization.

```
plt.figure(figsize=(8,5))
sns.countplot(data['Target_Type'],
              palette='rocket')
plt.xticks(rotation=45,fontsize=7,
           fontweight='bold')
plt.title('Target Type',
          fontsize=20,fontweight='bold',
          color='black')
```


Therefore the main Target type are the Private Citizens and Property then Military then Police.