Global Seismic Activities 2023 Analysis

The dataset "Seismic Activities 2023 global" provides valuable information about earthquakes and other seismic activity worldwide in the year 2023. This dataset includes various parameters such as time, location (latitude and longitude), depth, magnitude, magnitude type, and more. In this exploration, we will delve into the dataset to gain insights into the seismic activity observed throughout the year.

Here we will understand the data and prepare it for creating report in Power BI

Steps we will follow here

1.Import the libraries. 2.Read the data. 3.Check the head and tail of data. 4.Check the shape of data. 5.Check the describe and info function. 6.Understand the number of unique values in a feature. 7.Check the null values and fill accordingly. 8.Check for the outlier and fill accordingly.

In [1]:

```
# Import the required libraries
import pandas as pd
import numpy as np
```

In [2]:

```
# Read the data
data = pd.read_csv('Seismic_Activities_2023_global.csv')
data.head() # Helpful for a quick visual inspection of the dataset
```

Out[2]:

	time	latitude	longitude	depth	mag	magType	nst	gap	dmin	rms	...	updated	place	type	horizonta
0	2023-01-01T00:49:25.294Z	52.0999	178.5218	82.770	3.10	ml	14.0	139.0	0.8700	0.18	...	2023-03-11T22:51:52.040Z	Rat Islands, Aleutian Islands, Alaska	earthquake	
1	2023-01-01T01:41:43.755Z	7.1397	126.7380	79.194	4.50	mb	32.0	104.0	1.1520	0.47	...	2023-03-11T22:51:45.040Z	23 km ESE of Manay, Philippines	earthquake	
2	2023-01-01T03:29:31.070Z	19.1631	-66.5251	24.000	3.93	md	23.0	246.0	0.8479	0.22	...	2023-03-11T22:51:29.040Z	Puerto Rico region	earthquake	

	time	latitude	longitude	depth	mag	magType	nst	gap	dmin	rms	...	updated	place	type	horizonta
3	2023-01-01T04:09:32.814Z	-4.7803	102.7675	63.787	4.30	mb	17.0	187.0	0.4570	0.51	...	2023-03-11T22:51:45.040Z	99 km SSW of Pagar Alam, Indonesia	earthquake	
4	2023-01-01T04:29:13.793Z	53.3965	-166.9417	10.000	3.00	ml	19.0	190.0	0.4000	0.31	...	2023-03-11T22:51:38.040Z	59 km SSW of Unalaska, Alaska	earthquake	

5 rows × 22 columns

In [3]:

```
data.tail() # Sometimes there is aggregate row in end, we can check that by using this
```

Out[3]:

	time	latitude	longitude	depth	mag	magType	nst	gap	dmin	rms	...	updated	place	type	ho
26637	2023-12-29T03:37:19.334Z	-6.9527	154.9829	10.000	5.2	mb	72.0	60.0	3.924	0.93	...	2023-12-29T04:05:57.040Z	89 km SW of Panguna, Papua New Guinea	earthquake	
26638	2023-12-29T04:38:54.109Z	32.3262	141.7386	10.000	5.1	mb	74.0	121.0	1.803	0.70	...	2023-12-29T10:59:44.533Z	Izu Islands, Japan region	earthquake	
26639	2023-12-29T08:42:05.747Z	-7.2411	68.0663	10.000	5.1	mb	60.0	54.0	12.776	0.57	...	2023-12-29T08:57:05.040Z	Chagos Archipelago region	earthquake	
26640	2023-12-29T11:02:48.679Z	-19.1602	169.0428	153.264	4.7	mb	40.0	61.0	3.746	0.82	...	2023-12-29T11:22:46.040Z	49 km NNW of Isangel, Vanuatu	earthquake	
26641	2023-12-29T16:31:16.656Z	25.1050	96.5309	10.000	5.0	mb	53.0	64.0	4.156	0.78	...	2023-12-29T16:45:27.040Z	92 km WSW of Myitkyina, Myanmar	earthquake	

5 rows × 22 columns

In [4]:

```
# Check the shape of data
data.shape # 26642 rows and 22 columns
```

Out[4]:

```
(26642, 22)
```

In [5]:

```
# Basic info about raw dataset
data.info()
# Here we check, whether the data types assigned to respective columns are correct
# Which columns have null values
# Memory usage
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26642 entries, 0 to 26641
Data columns (total 22 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   time             26642 non-null  object 
 1   latitude         26642 non-null  float64
 2   longitude        26642 non-null  float64
 3   depth            26642 non-null  float64
 4   mag              26642 non-null  float64
 5   magType          26642 non-null  object 
 6   nst              25227 non-null  float64
 7   gap              25225 non-null  float64
 8   dmin             24776 non-null  float64
 9   rms              26642 non-null  float64
 10  net              26642 non-null  object 
 11  id               26642 non-null  object 
 12  updated          26642 non-null  object 
 13  place            25034 non-null  object 
 14  type             26642 non-null  object 
 15  horizontalError  25093 non-null  float64
 16  depthError       26642 non-null  float64
 17  magError         24970 non-null  float64
 18  magNst           25065 non-null  float64
 19  status           26642 non-null  object 
 20  locationSource   26642 non-null  object 
 21  magSource        26642 non-null  object 
dtypes: float64(12), object(10)
memory usage: 4.5+ MB
```

In [6]:
```python
# To get summarized analysis of numerical column
data.describe()
```

Out[6]:

	latitude	longitude	depth	mag	nst	gap	dmin	rms	horizontalError	deptl
count	26642.000000	26642.000000	26642.000000	26642.000000	25227.000000	25225.000000	24776.000000	26642.000000	25093.000000	26642.0
mean	16.852798	-11.487497	67.491224	4.007395	42.571332	124.930971	2.692908	0.581575	7.017267	4.4
std	30.389200	130.053399	116.762456	0.794423	37.662352	67.430145	4.043568	0.256276	4.072365	4.4
min	-65.849700	-179.998700	-3.370000	2.600000	0.000000	8.000000	0.000000	0.010000	0.000000	0.0
25%	-6.415275	-149.608650	10.000000	3.220000	19.000000	73.000000	0.612000	0.410000	4.140000	1.8
50%	18.884167	-64.811833	21.998000	4.300000	30.000000	111.000000	1.579000	0.590000	7.060000	2.0
75%	41.827950	126.965100	66.833000	4.500000	52.000000	165.000000	3.172000	0.750000	9.730000	6.6
max	86.593900	179.999400	681.238000	7.800000	423.000000	350.000000	50.820000	1.880000	99.000000	60.6

In [7]:
```python
# View the unique values
data['magType'].unique()
```

Out[7]:
```
array(['ml', 'mb', 'md', 'mw', 'mwr', 'mb_lg', 'mh', 'mlr', 'mlv',
       'ms_vx', 'ml(texnet)', 'mww', 'mwc', 'mwb', 'mwp'], dtype=object)
```

In [8]:
```python
data['type'].unique()
```

Out[8]:
```
array(['earthquake', 'mining explosion', 'quarry blast', 'ice quake',
       'explosion', 'Landslide', 'volcanic eruption', 'landslide'],
      dtype=object)
```

In [9]:
```python
data['id'].nunique() # To get the total number of unique 'id' values
```

Out[9]:
```
24682
```

In [10]:
```python
# Dealing with null values
data.isna().any()
# Here there are frew columns which has null values i.e. nst, gap, dmin, place, horizontalError, magError and magNst.
```

Out[10]:
```
time               False
latitude           False
longitude          False
depth              False
mag                False
magType            False
nst                 True
gap                 True
dmin                True
rms                False
net                False
id                 False
updated            False
place               True
type               False
horizontalError     True
depthError         False
magError            True
magNst              True
status             False
locationSource     False
magSource          False
dtype: bool
```

In [11]:
```
data.isna().sum()
```

Out[11]:
```
time                  0
latitude              0
longitude             0
depth                 0
mag                   0
magType               0
nst                1415
gap                1417
dmin               1866
rms                   0
net                   0
id                    0
updated               0
place              1608
type                  0
horizontalError    1549
depthError            0
magError           1672
```

```
magNst              1577
status                 0
locationSource         0
magSource              0
dtype: int64
```

In [12]:

```python
# Creating copy of data
data_temp = data.copy()
```

In [13]:

```python
# Filling Null values

data_temp['dmin'] = data_temp['dmin'].fillna(data_temp['dmin'].mean())
data_temp['place'] = data_temp['place'].replace(np.nan, 'Unknown')
data_temp['horizontalError'] = data_temp['horizontalError'].fillna(data_temp['horizontalError'].median())
data_temp['magError'] = data_temp['magError'].fillna(data_temp['magError'].median())
data_temp['magNst'] = data_temp['magNst'].fillna(33)
```

In [14]:

```python
data_temp.isna().sum()
```

Out[14]:
```
time                  0
latitude              0
longitude             0
depth                 0
mag                   0
magType               0
nst                1415
gap                1417
dmin                  0
rms                   0
net                   0
id                    0
updated               0
place                 0
type                  0
horizontalError       0
depthError            0
magError              0
magNst                0
status                0
locationSource        0
magSource             0
dtype: int64
```

In [15]:
```python
# Checking the distribution of data
kurtosis_value = data_temp.kurtosis() # Check kurtosis
print(kurtosis_value)
```

```
latitude            -0.745274
longitude           -1.630079
depth                9.845202
mag                 -0.491595
nst                 10.931679
gap                  0.000923
dmin                30.652838
rms                 -0.146381
horizontalError     11.477214
depthError          11.250269
magError           667.043719
magNst              45.109141
dtype: float64
```

```
C:\Users\SAKETK~1\AppData\Local\Temp/ipykernel_5124/1336331859.py:2: FutureWarning: Dropping of nuisance columns in DataF
rame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only val
```

```
id columns before calling the reduction.
  kurtosis_value = data_temp.kurtosis() # Check kurtosis
```

In [16]:

```python
skewness_value = data_temp.skew() # Check skew
print(skewness_value)
```

```
latitude           -0.344790
longitude           0.095953
depth               3.126641
mag                -0.276278
nst                 2.686683
gap                 0.806147
dmin                4.691464
rms                 0.129647
horizontalError     0.769693
depthError          2.622667
magError           20.714969
magNst              5.275833
dtype: float64
```

```
C:\Users\SAKETK~1\AppData\Local\Temp/ipykernel_5124/1886591829.py:1: FutureWarning: Dropping of nuisance columns in DataF
rame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only val
id columns before calling the reduction.
  skewness_value = data_temp.skew() # Check skew
```

In [17]:

```python
# Removing outliers
# Finding out IQR for different variables.
Q1 = data_temp.quantile(0.25)
Q3 = data_temp.quantile(0.75)
IQR = Q3 - Q1
print(IQR)
```

```
latitude             48.243225
longitude           276.573750
depth                56.833000
mag                   1.280000
nst                  33.000000
gap                  92.000000
dmin                  2.354275
rms                   0.340000
horizontalError       5.200000
depthError            4.821000
magError              0.066000
magNst               23.000000
dtype: float64
```

In [18]:

```python
# Removing outliers
data_cleaned = data_temp[~((data_temp < (Q1 - 10 * IQR)) |(data_temp > (Q3 + 10 * IQR))).any(axis=1)]
# Normally 1.5*IQR is taken but here I am taking 10*IQR  to accomodate some geniune values
```

```
C:\Users\SAKETK~1\AppData\Local\Temp/ipykernel_5124/3722644091.py:2: FutureWarning: Automatic reindexing on DataFrame vs
Series comparisons is deprecated and will raise ValueError in a future version.  Do `left, right = left.align(right, axis
=1, copy=False)` before e.g. `left == right`
  data_cleaned = data_temp[~((data_temp < (Q1 - 10 * IQR)) |(data_temp > (Q3 + 10 * IQR))).any(axis=1)]
```

In [19]:

```python
# Displaying shape before and after removing outliers
print("Shape with outliers: ", data_temp.shape)
print("Shape without outliers: ", data_cleaned.shape) # We have removed 394 outliers
```

```
Shape with outliers:  (26642, 22)
Shape without outliers:  (26248, 22)
```

In [20]:

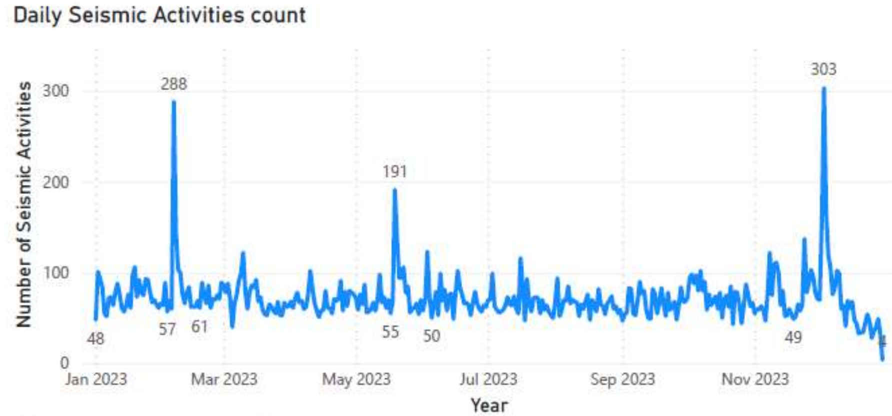
```python
# Converting the cleaned data to csv
data_cleaned.to_csv("Earthquake 2023 cleaned data")
```

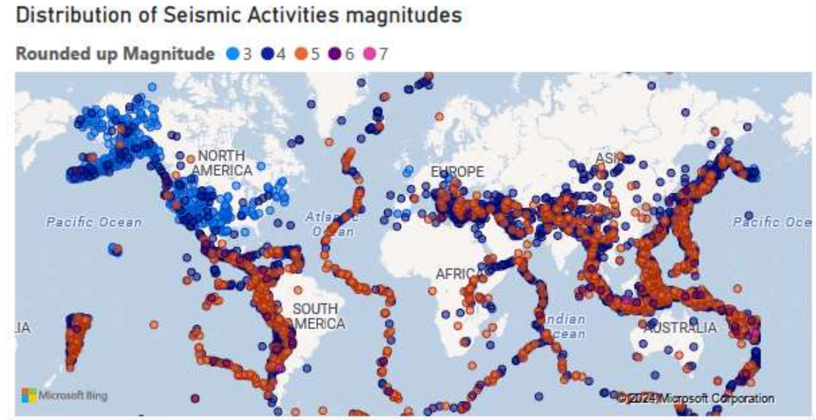
Importing report generated in Power Bi.

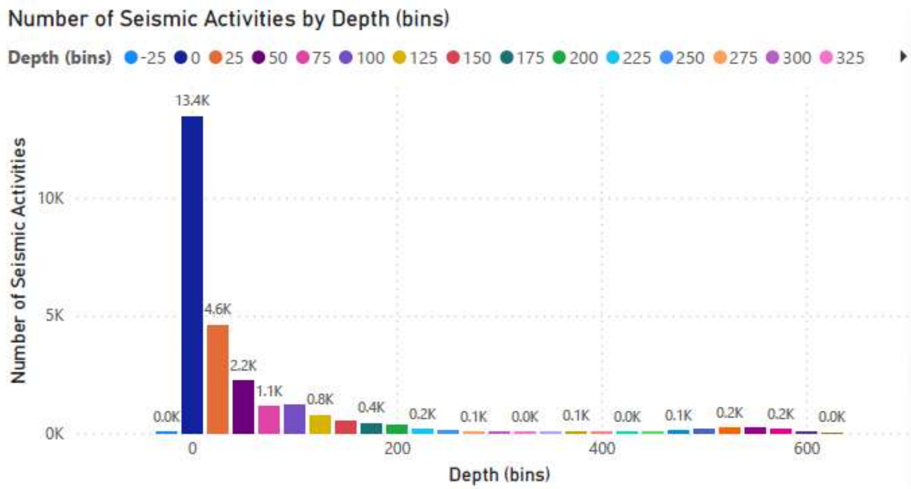
In [27]:

```python
from IPython.display import Image, display

image_path = 'Power BI Report.png'

# Display the image
display(Image(filename=image_path))
```

Global Seismic Activities 2023 Report







Different types of seismic activities

Type	Total	Max. Magnitude	Med. depth	Med. Depth Error	Med. Horizontal Error
earthquake	26034	7.80	23.02	2.02	7.06
mining explosion	167	4.10	0.00	1.73	4.93
ice quake	26	3.00	0.00	0.00	7.06
volcanic eruption	13	4.80	10.00	5.02	7.04
explosion	3	2.77	-0.70	31.61	0.93
Landslide	3	4.30	0.00	0.00	7.06
quarry blast	2	3.07	-1.20	31.61	1.72

earthquake	explosion	ice quake	Landslide
mining explosion	quarry blast	volcanic eruption	

Summary:

Temporal Insights: The peak of seismic activity occurred on December 3, 2023.

Geospatial Patterns: Comparison of seismic activities between North and South America reveals that the South American region experiences relatively higher magnitudes of earthquake events compared to the North American region.

Depth Distribution: Approximately 50% of seismic activities are concentrated within the depth range from ground level to 25 meters below the surface. This suggests that a significant portion of earthquakes occurs near the Earth's surface.

Magnitude Dominance: Earthquake occurs approximately 99% out of the total seismic activities. This indicates the dominance of one highly impactful earthquake in the dataset.