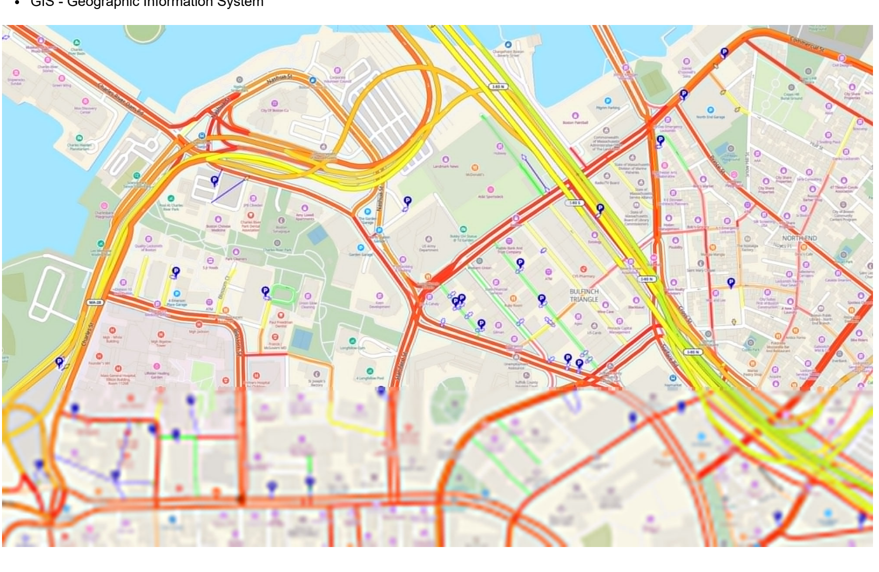
Today's agenda

- Location analytics
  - Real-time geographic DA
  - Historical geographic DA
- Important fields where LA is used
- Python pacakges for Geospatial visualization
- Earthquake data preparation
- Earthquake data visualization

In [ ]:

Location Analytics - Geospatial Visualization

- The ability to gain insights from the location or geographic component of business data.
- The important component is the location data.
- GIS - Geographic Information System



Credits - Image from Internet

In [ ]:

Real-time Geographic Data Analytics

Getting insights from the data that comes into the system and relating that to a particluar location is called Real-time Geographic data analytics.

- Getting route navigation in Google Maps
- Courier and postal services
- Military serives
  - Getting the exact location of the enemy movements on the map to get informed

Credits - Image from Internet

In [ ]:

Historic Geographic Data Analytics

Getting insights from the historic data so as to predict the furture occurrence or disaster (purely by chance) completely based on the geographic location.

- Predict and prevent (taking necessary steps) the occurrence based on the past data
- Disaster prevention efforts
  - Floods
  - Volcanoes
  - Earthquakes

- Chain retailers benefit from the historical spending habits based on location to increase the sales

Credits - Image from Internet

In [ ]:

Important fields/topics

Location analytics (Geospatial science) can be used in every field/study where data is generated with respect to the location.

Credits - Image from Internet

- Demographic study - statistical study of population of all the areas based on location
  - Analysis based on education details
  - Analysis based on ethnicity
  - Analysis based on religion
  - Analysis based on births and deaths
  - ...
- Measurement of **Natural Occurrences** that happen on timely basis
- Increase or decrease of pollution in a particular city or country
- Geographic crime analysis
  - Analyse the crime data
  - Identify the crime types and hotspots
  - Help law enforcement plan for crime prevention
- ETC

In [ ]:

Python packages for Geospatial Visualization

- GeoViews → explore & visualize geographical, meteorological, and oceanographic datasets
- Folium → widely used geospatial data visualization library built on Leaflet.js framework
- Plotly → Interactive map visualization - effectively uses Mapbox api
- KeplerGL → built exclusively for jupyter notebook to visualize big geospatial data
- GeoPandas → workhorse for working geo-data
- geonamescache → used to retrieve location datasets in the form of python dictionaries

In [ ]:

`import` packages

In [1]:
```
import pandas as pd
import plotly.graph_objects as go
```

If you do not have the above packages, you can install by typing these commands on `Command Prompt` (CMD) -

- `pip install pandas --user`
- `pip install plotly --user`

Dataset description

Earthquake data (from Yesterday) - The data is obtained from `USGS` datasources. The data is updated every 1 minute. In this example we don't deal with streaming data.

- time
- latitude
- longitude
- mag (magnitude)
- place

Data Source → https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all_day.csv

In [2]:
```
data_source = 'https://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all_day.csv'
eqdf = pd.read_csv(data_source)
eqdf.shape
```

Out[2]: (246, 22)

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

Out[3]:

	time	latitude	longitude	depth	mag	magType	nst	gap	dmin	rms	...	updated	place	typ
0	2021-04-19T15:24:10.390Z	38.835499	-122.803169	2.07	1.03	md	21.0	54.0	0.01043	0.02	...	2021-04-19T15:39:09.737Z	7km WNW of Cobb, CA	earthquak
1	2021-04-19T15:23:52.117Z	63.219100	-151.416500	2.60	1.40	ml	NaN	NaN	NaN	0.69	...	2021-04-19T15:31:35.806Z	39 km SSE of Denali National Park, Alaska	earthquak
2	2021-04-19T15:21:23.063Z	65.146300	-149.261800	6.90	1.50	ml	NaN	NaN	NaN	0.67	...	2021-04-19T15:25:34.620Z	4 km E of Minto, Alaska	earthquak
3	2021-04-19T15:20:25.600Z	19.404333	-155.251999	0.41	1.72	ml	15.0	110.0	NaN	0.38	...	2021-04-19T15:25:55.350Z	4 km SSW of Volcano, Hawaii	earthquak
4	2021-04-19T15:11:44.807Z	66.401900	-157.449200	8.60	1.40	ml	NaN	NaN	NaN	1.00	...	2021-04-19T15:21:43.796Z	55 km SSW of Shungnak, Alaska	earthquak

5 rows × 22 columns

In [4]:
```
print(eqdf.columns)
```

```
Index(['time', 'latitude', 'longitude', 'depth', 'mag', 'magType', 'nst',
       'gap', 'dmin', 'rms', 'net', 'id', 'updated', 'place', 'type',
       'horizontalError', 'depthError', 'magError', 'magNst', 'status',
       'locationSource', 'magSource'],
      dtype='object')
```

In [5]:
```
# take subset data (time, latitude, longitude, mag, place)
eqdf  = eqdf[['time', 'latitude', 'longitude', 'mag', 'place']]
```

In [6]:
```
# head()
eqdf.head()
```

Out[6]:

	time	latitude	longitude	mag	place
0	2021-04-19T15:24:10.390Z	38.835499	-122.803169	1.03	7km WNW of Cobb, CA
1	2021-04-19T15:23:52.117Z	63.219100	-151.416500	1.40	39 km SSE of Denali National Park, Alaska
2	2021-04-19T15:21:23.063Z	65.146300	-149.261800	1.50	4 km E of Minto, Alaska
3	2021-04-19T15:20:25.600Z	19.404333	-155.251999	1.72	4 km SSW of Volcano, Hawaii
4	2021-04-19T15:11:44.807Z	66.401900	-157.449200	1.40	55 km SSW of Shungnak, Alaska

In [7]:
```
# shape
eqdf.shape
```

Out[7]: (246, 5)

In [8]:
```
# take "place" separately and display head
eqdf['place'].head()
```

Out[8]:
```
0                          7km WNW of Cobb, CA
1    39 km SSE of Denali National Park, Alaska
2                      4 km E of Minto, Alaska
3                  4 km SSW of Volcano, Hawaii
4                55 km SSW of Shungnak, Alaska
Name: place, dtype: object
```

In [9]:
```
# example
dp = '33 km Hyderabad, India'
gp = dp.split(', ')
print(gp)
print('------------')
print("Sub Area →", gp[0])
print('------------')
print("Area →", gp[-1])
# or
print("Area →", gp[1])
```

```
['33 km Hyderabad', 'India']
------------
Sub Area → 33 km Hyderabad
------------
Area → India
Area → India
```

In [ ]:

Data Preperation

Separate `place` by → **comma** and **space**

In [10]:
```
# place_list → split
place_list = eqdf['place'].str.split(', ')
```

Print first `5` from `place_list`

In [11]:
```
place_list.head()
```

Out[11]:
```
0                          [7km WNW of Cobb, CA]
1    [39 km SSE of Denali National Park, Alaska]
2                      [4 km E of Minto, Alaska]
3                  [4 km SSW of Volcano, Hawaii]
4                [55 km SSW of Shungnak, Alaska]
Name: place, dtype: object
```

Make two columns `area` and `sub_area` from area_list

In [12]:
```
# apply
sub_area = place_list.apply(lambda place : place[0])
area = place_list.apply(lambda place : place[-1])
```

Print first `5` from `area`

In [13]:
```
area.head()
```

Out[13]:
```
0        CA
1    Alaska
2    Alaska
3    Hawaii
4    Alaska
Name: place, dtype: object
```

Print first `5` from `sub_area`

In [14]:
```
sub_area.head()
```

Out[14]:
```
0                        7km WNW of Cobb
1    39 km SSE of Denali National Park
2                        4 km E of Minto
3                    4 km SSW of Volcano
4                  55 km SSW of Shungnak
Name: place, dtype: object
```

Add `area` and `sub_area` as columns in `eqdf`

In [15]:
```
eqdf['sub_area'] = sub_area
eqdf['area'] = area
```

In [16]:
```
# head()
eqdf.head()
```

Out[16]:

	time	latitude	longitude	mag	place	sub_area	area
0	2021-04-19T15:24:10.390Z	38.835499	-122.803169	1.03	7km WNW of Cobb, CA	7km WNW of Cobb	CA
1	2021-04-19T15:23:52.117Z	63.219100	-151.416500	1.40	39 km SSE of Denali National Park, Alaska	39 km SSE of Denali National Park	Alaska
2	2021-04-19T15:21:23.063Z	65.146300	-149.261800	1.50	4 km E of Minto, Alaska	4 km E of Minto	Alaska
3	2021-04-19T15:20:25.600Z	19.404333	-155.251999	1.72	4 km SSW of Volcano, Hawaii	4 km SSW of Volcano	Hawaii
4	2021-04-19T15:11:44.807Z	66.401900	-157.449200	1.40	55 km SSW of Shungnak, Alaska	55 km SSW of Shungnak	Alaska

Remove the column `place` from `eqdf`

In [17]:
```
# prep_df
prep_df = eqdf.drop(columns=['place'], axis=0)
```

In [18]:
```
# head()
prep_df.head()
```

Out[18]:

	time	latitude	longitude	mag	sub_area	area
0	2021-04-19T15:24:10.390Z	38.835499	-122.803169	1.03	7km WNW of Cobb	CA
1	2021-04-19T15:23:52.117Z	63.219100	-151.416500	1.40	39 km SSE of Denali National Park	Alaska
2	2021-04-19T15:21:23.063Z	65.146300	-149.261800	1.50	4 km E of Minto	Alaska
3	2021-04-19T15:20:25.600Z	19.404333	-155.251999	1.72	4 km SSW of Volcano	Hawaii
4	2021-04-19T15:11:44.807Z	66.401900	-157.449200	1.40	55 km SSW of Shungnak	Alaska

In [19]:
```
# shape
prep_df.shape
```

Out[19]: (246, 6)

Consider the data where magnitude is greater than or equal to `2`

In [20]:
```
# prep_df
prep_df = prep_df[prep_df['mag'] >= 2]
```

In [21]:
```
# head
prep_df.head()
```

Out[21]:

	time	latitude	longitude	mag	sub_area	area
6	2021-04-19T15:01:11.040Z	19.203501	-155.449173	2.02	3 km E of Pāhala	Hawaii
7	2021-04-19T14:54:33.290Z	37.660999	-122.522163	2.37	6km WSW of Colma	CA
10	2021-04-19T14:31:52.965Z	44.296500	-115.196900	2.50	22 km WNW of Stanley	Idaho
17	2021-04-19T14:08:08.420Z	38.747167	-112.500833	2.38	8 km SW of Kanosh	Utah
20	2021-04-19T13:47:22.411Z	-33.976600	-72.362500	4.10	81 km WSW of San Antonio	Chile

In [22]:
```
prep_df.shape
```

Out[22]: (62, 6)

In [ ]:

Data Visualization

Check the frequency of occurrence by `area`

In [23]:
```
# occ_freq → value_counts().to_frame()
occ_freq = prep_df['area'].value_counts().to_frame()
```

In [24]:
```
occ_freq
```

Out[24]:

	area
Alaska	11
Hawaii	7
Chile	6
Iran	4
Indonesia	4
CA	4
Puerto Rico	4
south of the Fiji Islands	3
Idaho	2
Russia	2
Oklahoma	2
off the coast of Ecuador	1
Texas	1
Washington	1
New Zealand	1
Utah	1
Japan	1
Papua New Guinea	1
Peru	1
New Mexico	1
Peru-Bolivia border region	1
Greece	1
Timor Leste	1
Mexico	1

default plot will not be so interactive

In [25]:
```
# occ_freq.plot(kind='pie', subplots=True)
```

`Pie` chart with `Plotly`

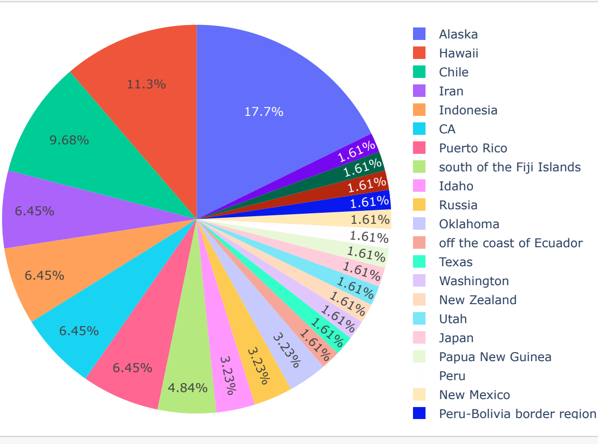
- labels → index
- values → area
- lrtb → 0

refer → https://plotly.com/python/pie-charts/

In [26]:
```
trace = go.Pie(
    labels=occ_freq.index,
    values=occ_freq.area
)

layout = go.Layout(
    height=400,
    width=600,
    margin=dict(l=0, r=0, t=0, b=0)
)

fig = go.Figure(data=[trace], layout=layout)
fig.show()
```



In [ ]:

`Bar` chart of magnitude based on area with `Plotly`

- x → sub_area
- y → mag
- lrtb → 0

refer → https://plotly.com/python/bar-charts/

In [27]:
```
region = 'Hawaii'

# rdf from prep_df
rdf = prep_df[prep_df['area'] == region]
```

In [28]:
```
# head()
rdf.head()
```

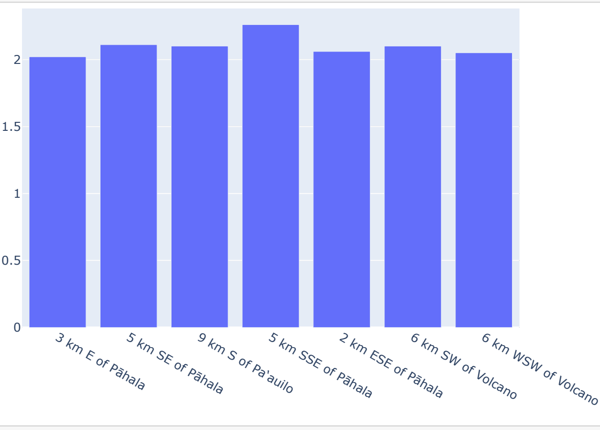
Out[28]:

	time	latitude	longitude	mag	sub_area	area
6	2021-04-19T15:01:11.040Z	19.203501	-155.449173	2.02	3 km E of Pāhala	Hawaii
50	2021-04-19T10:46:00.210Z	19.162500	-155.447006	2.11	5 km SE of Pāhala	Hawaii
52	2021-04-19T10:41:09.330Z	19.955500	-155.381500	2.10	9 km S of Pa'auilo	Hawaii
72	2021-04-19T08:58:53.410Z	19.160166	-155.453995	2.26	5 km SSE of Pāhala	Hawaii
110	2021-04-19T04:29:44.180Z	19.192833	-155.461166	2.06	2 km ESE of Pāhala	Hawaii

In [29]:
```
# bar chart
trace = go.Bar(
    x=rdf.sub_area,
    y=rdf.mag
)

layout = go.Layout(
    height=400,
    width=600,
    margin=dict(l=0, r=0, t=0, b=0)
)

fig = go.Figure(data=[trace], layout=layout)
fig.show()
```



In [ ]:

Map Visualization

- lat → latitudes
- lon → longitudes
- mode → markers
  - size → 10
  - color → red
- text → sub_area

refer → https://plotly.com/python/scattermapbox/

In [30]:
```
region = 'CA'

# filter it
rdf = prep_df[prep_df['area'] == region]
```

In [31]:
```
# lats, lons, text (sub_area)
region_lats = rdf['latitude'].to_list()
region_lons = rdf['longitude'].to_list()
region_text = rdf['sub_area'].to_list()
```

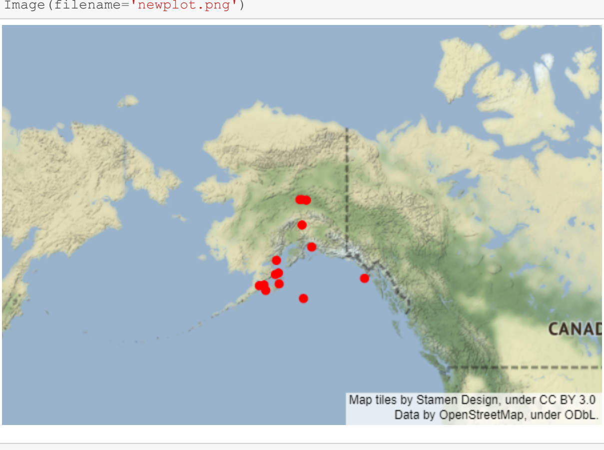
In [34]:
```
trace = go.Scattermapbox(
    lat=region_lats,
    lon=region_lons,
    mode='markers',
    marker=dict(
        size=10,
        color='red'
    ),
    text=region_text,
    hoverinfo='text'
)

layout = go.Layout(
    height=400,
    width=600,
    margin=dict(l=0, r=0, t=0, b=0),
    mapbox_style='stamen-terrain',
    mapbox=dict(
        center=dict(
            lat=region_lats[0],
            lon=region_lons[0]
        ),
        zoom=2
    )
)

# fig = go.Figure(data=[trace], layout=layout)
# fig.show()
```

In [35]:
```
from IPython.display import Image
Image(filename='newplot.png')
```

Out[35]:



In [ ]:

What did we learn?

- Location Analytics
- Where it is used and how it is important for the business or the govt agency
- Python modules that support geospatial visualization
- Earthquake data - using pandas
  - Data preparation
  - Data filtering
- Earthquake data visualization