In [1]:
```
pip install pandas
```

```
Requirement already satisfied: pandas in c:\users\samil\anaconda3\lib\site-packages (2.0.3)Note: you may need to restart the kernel to use updated packages.

Requirement already satisfied: numpy>=1.20.3; python_version < "3.10" in c:\users\samil\anaconda3\lib\site-packages (from pandas) (1.24.4)
Requirement already satisfied: tzdata>=2022.1 in c:\users\samil\anaconda3\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\samil\anaconda3\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\samil\anaconda3\lib\site-packages (from pandas) (2020.1)
Requirement already satisfied: six>=1.5 in c:\users\samil\anaconda3\lib\site-packages (from python-dateutil>=2.8.2->pandas) (1.15.0)
```

In [2]:
```
pip install plotly
```

```
Requirement already satisfied: plotly in c:\users\samil\anaconda3\lib\site-packages (5.18.0)Note: you may need to restart the kernel to use updated packages.

Requirement already satisfied: packaging in c:\users\samil\anaconda3\lib\site-packages (from plotly) (20.4)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\samil\anaconda3\lib\site-packages (from plotly) (8.2.3)
Requirement already satisfied: pyparsing>=2.0.2 in c:\users\samil\anaconda3\lib\site-packages (from packaging->plotly) (2.4.7)
Requirement already satisfied: six in c:\users\samil\anaconda3\lib\site-packages (from packaging->plotly) (1.15.0)
```

In [3]:
```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib.animation import FuncAnimation
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
C:\Users\Samil\anaconda3\lib\site-packages\pandas\core\computation\expressions.py:20: UserWarning: Pandas requires version '2.7.3' or newer of 'numexpr' (version '2.7.1' current
ly installed).
  from pandas.core.computation.check import NUMEXPR_INSTALLED
```

In [4]:
```
raw_dataset=pd.read_csv("LosAnglesAll.csv",sep=",")
```

In [5]:
```
LosAnglesAll = raw_dataset.copy()
LosAnglesAll.head()
```

Out[5]:

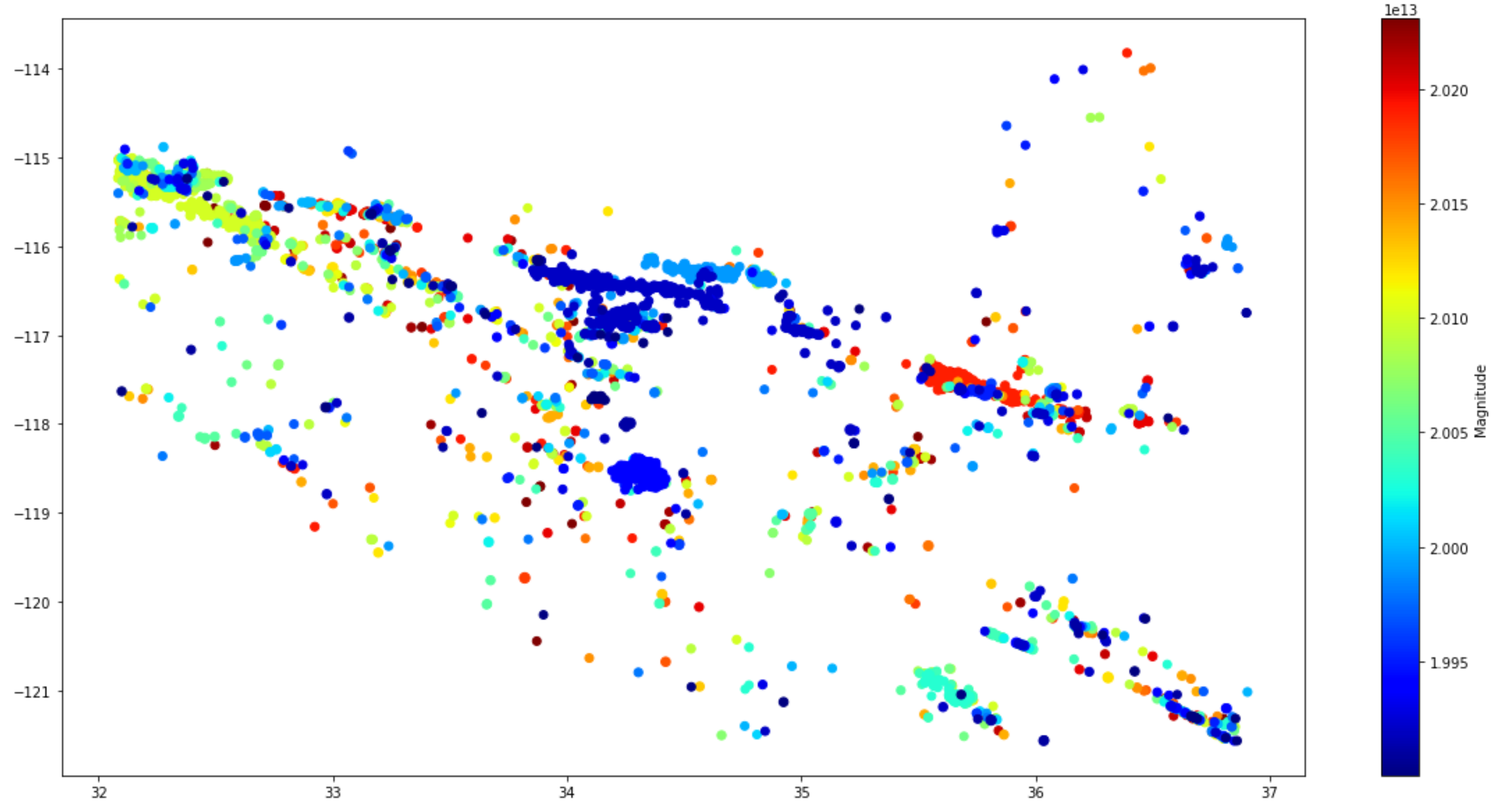
	time	Altitude	Longititude	Deptm_km	Magnitude	Unnamed: 5
0	2023-08-02T05:38:08	33.186000	-115.573500	3.11	4.12	NaN
1	2023-07-02T09:29:49	33.827000	-118.881000	10.73	3.72	NaN
2	2023-06-19T06:49:11	35.498000	-118.145000	6.81	3.57	NaN
3	2023-05-30T20:24:52	34.021833	-119.124833	13.56	3.56	NaN
4	2023-05-15T00:13:39	32.467167	-115.956667	1.00	3.63	NaN

In [11]:
```
LosAnglesAll.shape
```

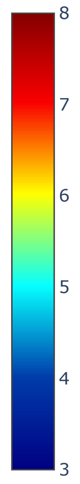
Out[11]:
```
(3639, 6)
```

In [12]:
```
x = LosAnglesAll.iloc[:,1].values
y = LosAnglesAll.iloc[:,2].values
z = LosAnglesAll.iloc[:,3].values
colors = LosAnglesAll.iloc[:,4].values
sizes = LosAnglesAll.iloc[:,4].values*10
```

In [25]:
```
fig = plt.figure(figsize=(20, 10))
my_cmap = plt.get_cmap('jet')

plt.scatter(x, y, c=colors, s=sizes, cmap= 'jet')

ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet')

cbar = plt.colorbar(ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet'))

ax.set_xlabel('Longitude')
ax.set_ylabel('Latitude')
cbar.set_label('Magnitude')
ax.set_zlabel('Depth_km')
font_size = 700
dpi = (5000)
font_size = 1000
plt.show()
```

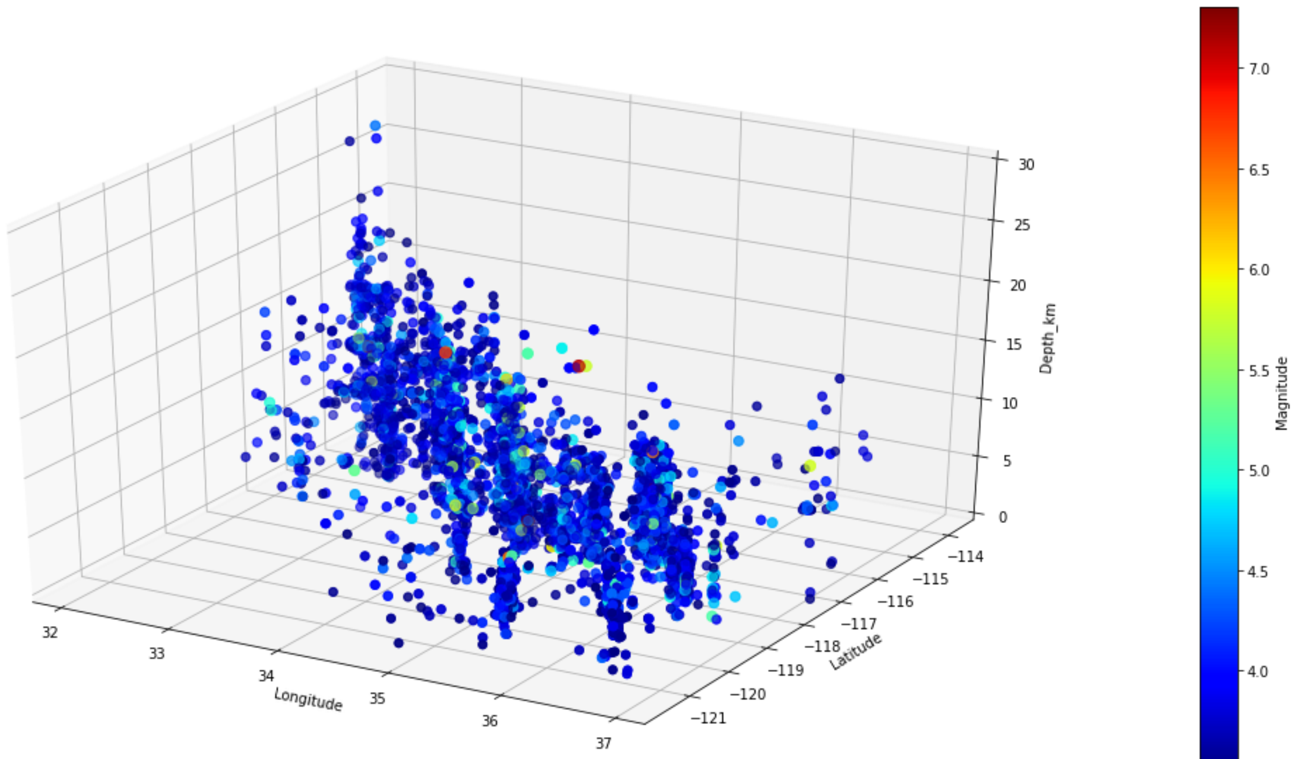


In [14]:
```
import plotly.graph_objects as go
# Yüksek çözünürlüklü dünya haritası verilerini çevrimiçi olarak alın
fig = go.Figure(go.Choroplethmapbox(
 geojson="https://raw.githubusercontent.com/johan/world.geo.json/master/countries.geo.json",
 locations=["USA", "CAN", "MEX", "RUS", "CHN"], # Örnek ülke kodları (ABD, Kanada, Meksika, Rusya, Çin)
 z=[1, 1, 1, 1, 1], # Ülkelere atanacak değerler (hepsi 1 olarak ayarlanmıştır)
 colorscale='Jet', # Renk skalası adı (Viridis, YlGnBu, Jet vb.)
 zmin=3,
 zmax=8,
 marker_opacity=0.9, # Ülke sınırlarının opaklığı
 marker_line_width=1, # Ülke sınırlarının kenarlık kalınlığı
))
# Örnek deprem verilerini oluşturun
deprem_verileri = {
 'Longitude': x,
 'Latitude': y,
 'Magnitude': colors,
 }
# Scatter plot ile deprem verilerini ekleyin
fig.add_trace(go.Scattermapbox(
 lat=deprem_verileri['Latitude'],
 lon=deprem_verileri['Longitude'],
 mode='markers',
 marker=dict(
 size=deprem_verileri['Magnitude'] * 2, # Magnitude değerine göre nokta boyutlarını belirleme
 color=deprem_verileri['Magnitude'], # Magnitude değerine göre renk skalasını belirleme
 colorscale='Jet', # Renk skalası adı (Viridis, YlGnBu, Jet vb.)
 ),
 ))
# Harita düzenini ve stilini belirleyin
fig.update_layout(
 mapbox_style="carto-positron", # Harita stilini belirleme (diğer stiller için: "open-street-map", "stamen-terrain" vb.)
 mapbox_zoom=6, # Harita yakınlaştırma düzeyini belirleme
 mapbox_center={"lat": 30.000, "lon": 30.0000}, # Harita merkezini belirleme (ABD'nin merkezi)
)
dpi = (9000)
font_size = 1000
# Grafiği görüntüleyin
fig.show()
```



```
In [15]: x = LosAnglesAll.iloc[:,1].values
         y = LosAnglesAll.iloc[:,2].values
         z = LosAnglesAll.iloc[:,3].values
         colors = LosAnglesAll.iloc[:,4].values
         sizes = LosAnglesAll.iloc[:,4].values*10
```

In [16]:
```
fig = plt.figure(figsize=(20, 10))

ax = fig.add_subplot(111, projection='3d')

ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet')


cbar = plt.colorbar(ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet'))

cbar.set_label('Magnitude')


ax.set_xlabel('Longitude')
ax.set_ylabel('Latitude')
ax.set_zlabel('Depth_km')




dpi = (5000)
font_size = 1000

ax.set_zlim(0, 30)


plt.show()
```



In [17]:
```
x = LosAnglesAll.iloc[:,1].values
y = LosAnglesAll.iloc[:,2].values
z = LosAnglesAll.iloc[:,3].values
colors = LosAnglesAll.iloc[:,4].values
sizes = LosAnglesAll.iloc[:,4].values*2
```

In [18]:
```
import plotly.graph_objs as go
import numpy as np

# Veri oluşturma (x, y, z, colors, sizes tanımlanmış olarak varsayıldı)

trace = go.Scatter3d(
    x=x,
    y=y,
    z=z,
    mode='markers',
    marker=dict(
        size=sizes,
        color=colors,
        colorscale='Jet',
        opacity=0.5,
        colorbar=dict(title='Magnitude')
    )
)

layout = go.Layout(
    scene=dict(
        xaxis=dict(title='Longitude'),
        yaxis=dict(title='Latitude'),
        zaxis=dict(title='Depth_km'),
        aspectmode='manual',
        aspectratio=dict(x=1, y=2, z=1),
        camera=dict(eye=dict(x=2, y=1, z=1))
    ),
    coloraxis=dict(colorbar=dict(len=0.75))
)

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

# Grafik döndürme
frames = []
for angle in np.linspace(0, 360, 36):
    frame = go.Frame(layout=dict(scene=dict(camera=dict(eye=dict(x=2*np.cos(np.radians(angle)), y=2*np.sin(np.radians(angle)), z=2)))))
    frames.append(frame)

fig.frames = frames
fig.update_layout(updatemenus=[dict(type='buttons', showactive=False, buttons=[dict(label='Play', method='animate', args=[None, dict(frame=dict(duration=200, redraw=True), fromcu

fig.show()
```

```
In [19]: x = LosAnglesAll.iloc[:,1].values
         y = LosAnglesAll.iloc[:,2].values
         z = LosAnglesAll.iloc[:,0].values
         colors = LosAnglesAll.iloc[:,4].values
         sizes = LosAnglesAll.iloc[:,4].values*10
```

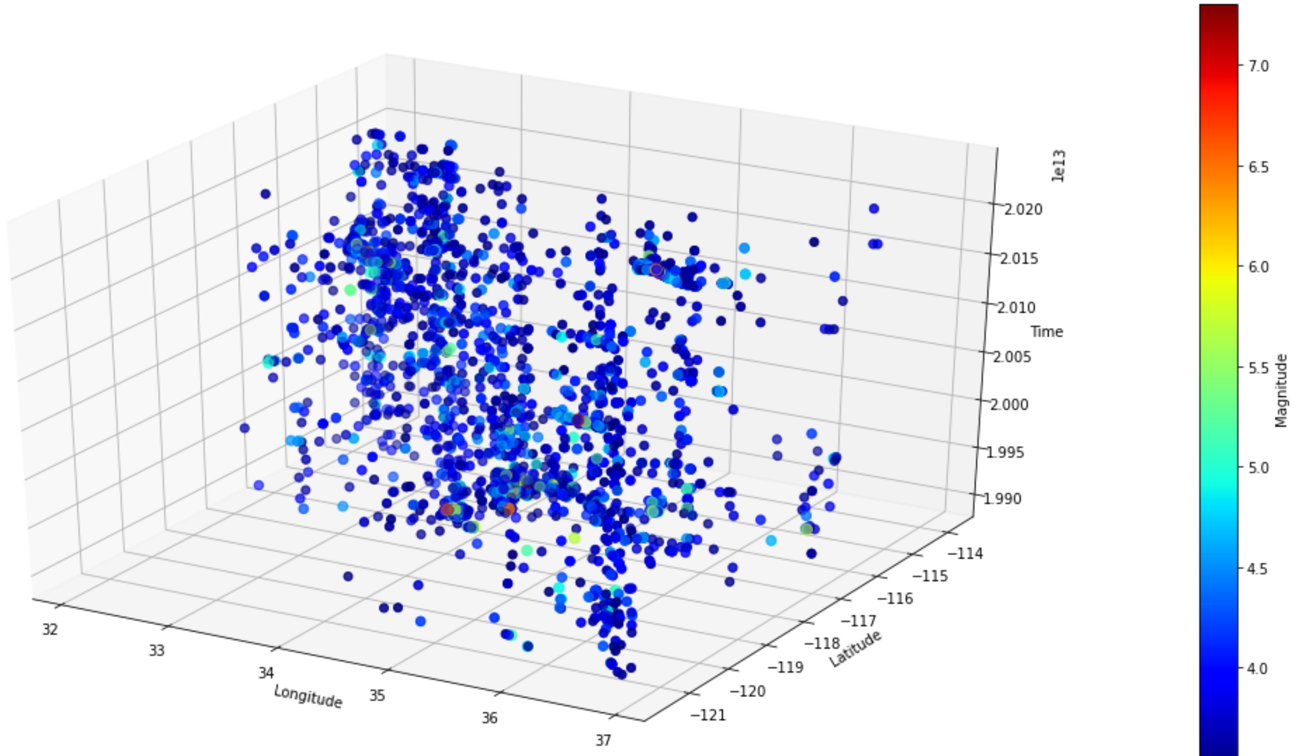
```
In [20]: z
```

```
Out[20]: array(['2023-08-02T05:38:08', '2023-07-02T09:29:49',
                '2023-06-19T06:49:11', ..., '1990-01-11T01:22:10',
                '1990-01-03T11:54:27', '1990-01-02T09:50:53'], dtype=object)
```

In [21]:
```
import pandas as pd
from datetime import datetime

# Örnek tarih listesi
dates_list = z

# Tarihleri tek rakam temsiline dönüştürme
single_digit_list = []
for date in dates_list:
    datetime_object = datetime.fromisoformat(date)
    single_digit_rep = int(
        f"{datetime_object.year}{datetime_object.month:02d}{datetime_object.day:02d}"
        f"{datetime_object.hour:02d}{datetime_object.minute:02d}{datetime_object.second:02d}"
    )
    single_digit_list.append(single_digit_rep)

# Tek rakam temsilini içeren DataFrame oluşturma
df = pd.DataFrame({'Single_Digit_Representation': single_digit_list})
print(df)
```

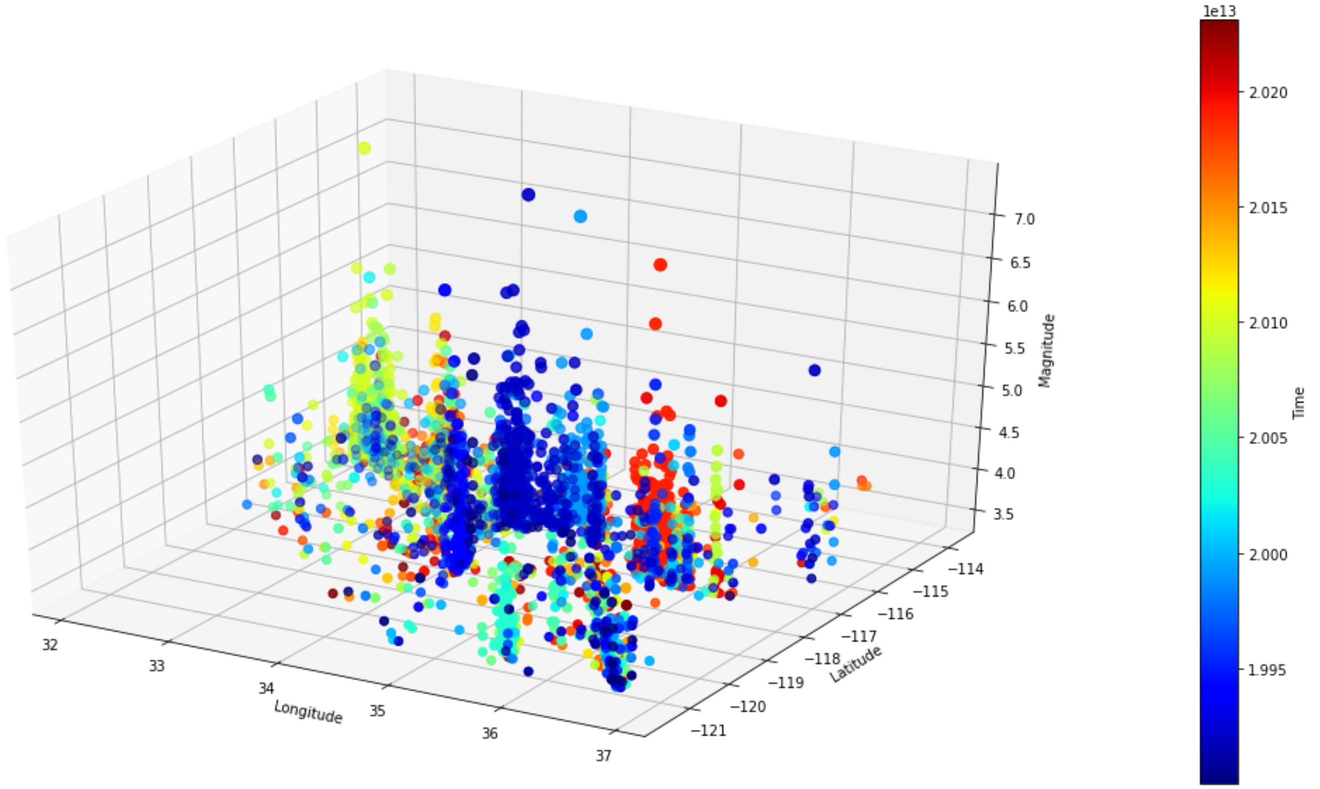
```
      Single_Digit_Representation
0                  20230802053808
1                  20230702092949
2                  20230619064911
3                  20230530202452
4                  20230515001339
...                           ...
3634               19900112091022
3635               19900111230857
3636               19900111012210
3637               19900103115427
3638               19900102095053

[3639 rows x 1 columns]
```

In [22]:
```
x = LosAnglesAll.iloc[:,1].values
y = LosAnglesAll.iloc[:,2].values
z = single_digit_list
colors =  LosAnglesAll.iloc[:,4].values
sizes = LosAnglesAll.iloc[:,4].values*10

fig = plt.figure(figsize=(20, 10))

ax = fig.add_subplot(111, projection='3d')

ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet')


cbar = plt.colorbar(ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet'))

cbar.set_label('Magnitude')


ax.set_xlabel('Longitude')
ax.set_ylabel('Latitude')
ax.set_zlabel('Time')

font_size = 700

dpi = (5000)
font_size = 1000




plt.show()
```



In [23]:
```
x = LosAnglesAll.iloc[:,1].values
y = LosAnglesAll.iloc[:,2].values
z = LosAnglesAll.iloc[:,4].values
colors =   single_digit_list
sizes = LosAnglesAll.iloc[:,4].values*10
```

In [24]:
```
fig = plt.figure(figsize=(20, 10))

ax = fig.add_subplot(111, projection='3d')

ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet')


cbar = plt.colorbar(ax.scatter(x, y, z, c=colors, s=sizes, cmap='jet'))

cbar.set_label('Time')


ax.set_xlabel('Longitude')
ax.set_ylabel('Latitude')
ax.set_zlabel('Magnitude')

font_size = 700

dpi = (5000)
font_size = 1000




plt.show()
```


1e13
2.020
2.015
2.010
2.005
2.000
1.995
Time
Magnitude
7.0
6.5
6.0
5.5
5.0
4.5
4.0
3.5
Longitude
32
33
34
35
36
37
Latitude
−114
−115
−116
−117
−118
−119
−120
−121

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```