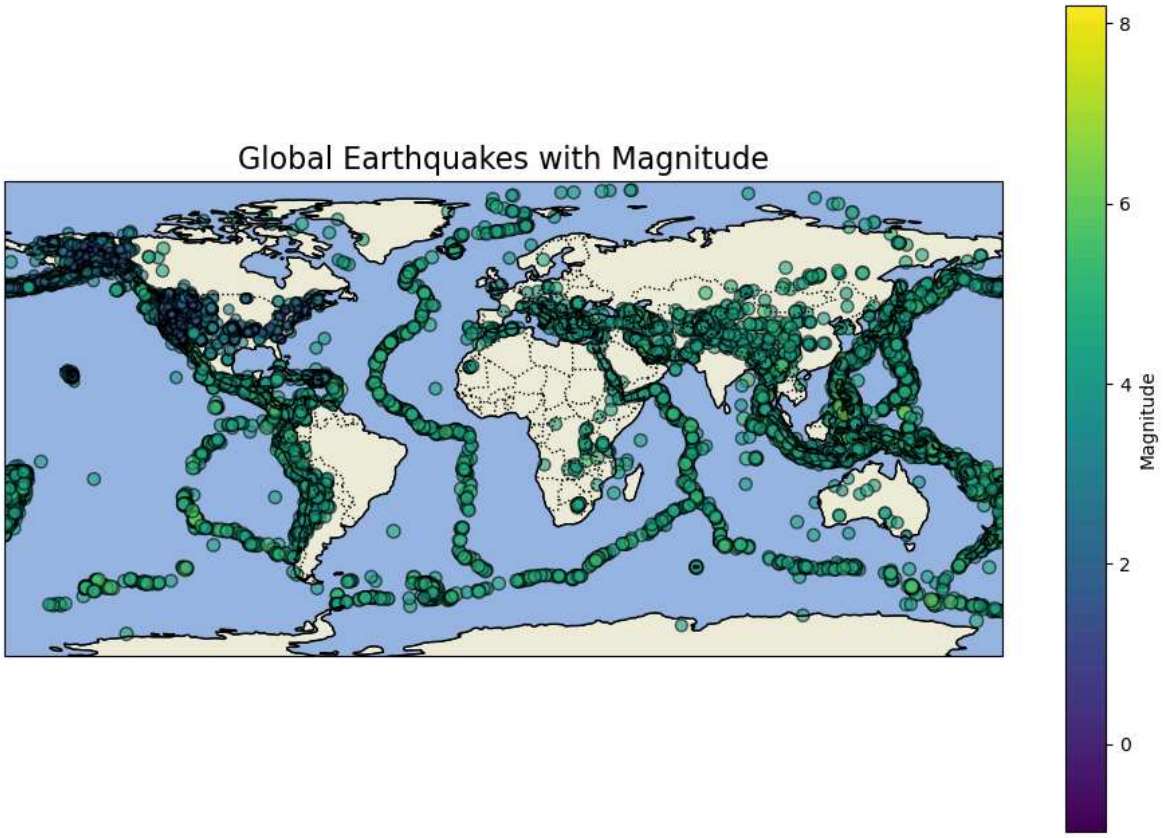
In [3]:
```python
# 1
import pandas as pd
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
import cartopy.feature as cfeature
from matplotlib.colors import Normalize
from matplotlib.cm import ScalarMappable

# 读取数据
data = pd.read_csv('usgs_earthquakes.csv')

# 提取经度、纬度和震级
longitudes = data['longitude']
latitudes = data['latitude']
magnitudes = data['mag']

# 创建一个颜色映射
norm = Normalize(vmin=magnitudes.min(), vmax=magnitudes.max())
cmap = plt.get_cmap('viridis')

# 创建地图
plt.figure(figsize=(12, 8))
# 使用 PlateCarree 投影
ax = plt.axes(projection=ccrs.PlateCarree())

# 添加海岸线和国家边界
ax.add_feature(cfeature.COASTLINE)
ax.add_feature(cfeature.BORDERS, linestyle=':')
ax.add_feature(cfeature.LAND, edgecolor='black')
ax.add_feature(cfeature.OCEAN)

# 绘制地震点
# 使用 scatter 绘制地震点，大小根据震级调整，颜色根据震级映射
scatter = ax.scatter(longitudes, latitudes, s=magnitudes * 10,
                     c=magnitudes, cmap=cmap, norm=norm,
                     alpha=0.6, edgecolors='k', transform=ccrs.PlateCarree())

# 添加颜色条
cbar = plt.colorbar(ScalarMappable(norm=norm, cmap=cmap), ax=ax, label='Magnitud
cbar.set_label('Magnitude')

# 添加标题
plt.title('Global Earthquakes with Magnitude', fontsize=16)

# 显示地图
plt.show()
```

```
D:\ANACONDA\Lib\site-packages\matplotlib\collections.py:996: RuntimeWarning: inva
lid value encountered in sqrt
  scale = np.sqrt(self._sizes) * dpi / 72.0 * self._factor
D:\ANACONDA\Lib\site-packages\matplotlib\collections.py:996: RuntimeWarning: inva
lid value encountered in sqrt
  scale = np.sqrt(self._sizes) * dpi / 72.0 * self._factor
D:\ANACONDA\Lib\site-packages\matplotlib\collections.py:996: RuntimeWarning: inva
lid value encountered in sqrt
  scale = np.sqrt(self._sizes) * dpi / 72.0 * self._factor
```



In [5]:
```python
# 这里我使用了Assignment 3的数据
import xarray as xr
import glob
# 定义文件路径模式
file_pattern = 'D:\\ESE5023\\output_file\\*.nc4'

# 获取所有匹配的文件路径
files = glob.glob(file_pattern)

# 读取并合并多个文件
CO2 = xr.open_mfdataset(files, combine='by_coords')

# 查看合并后的数据集
CO2
```

Out[5]: xarray.Dataset

▸ Dimensions: (**lat**: 361, **lon**: 576, **time**: 86)

▾ Coordinates:

lat	(lat)	float64	-90.0 -89.5 -89.0 ... ...
lon	(lon)	float64	-180.0 -179.4 ... 17...
time	(time)	datetime64[ns]	2015-01-16T12:00:...

▾ Data variables:

XCO2	(time, lat, lon)	float64	dask.array<chunksi...
XCO2PREC	(time, lat, lon)	float64	dask.array<chunksi...

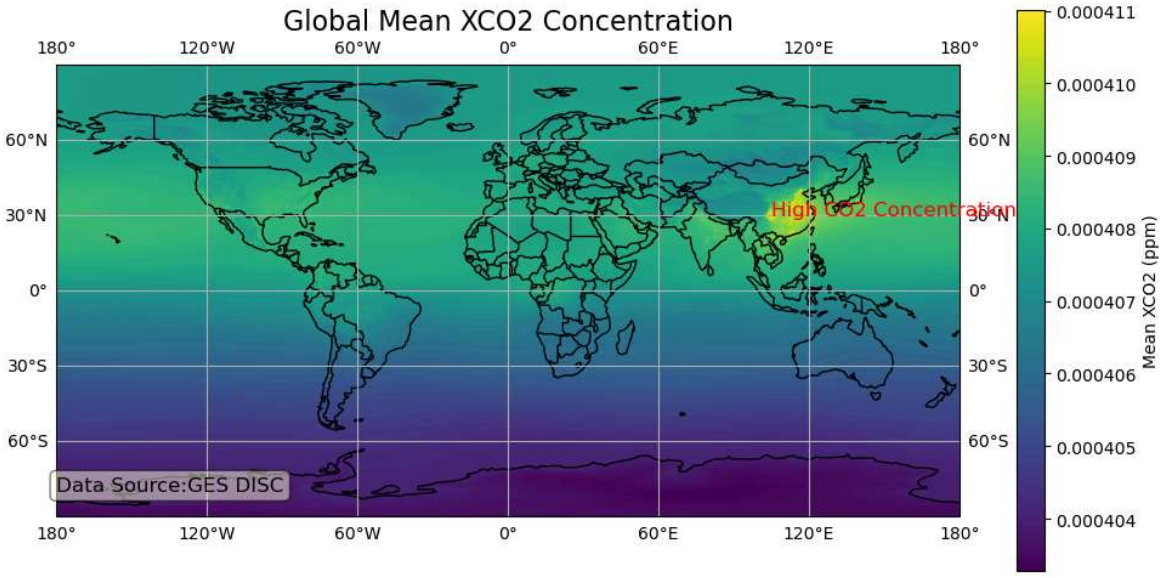
▸ Indexes: (3)

▸ Attributes: (25)

In [25]:

```
# 2.1
# 选择 XCO2 变量
xco2 = CO2['XCO2']

# 计算 XCO2 的整体平均值
xco2_mean = xco2.mean(dim='time')

# 创建绘图使用 PlateCarree 投影
fig, ax = plt.subplots(figsize=(12, 6), subplot_kw={'projection': ccrs.PlateCarr

# 绘制 XCO2 的平均浓度图
xco2_mean.plot(ax=ax, cmap='viridis', cbar_kwargs={'label': 'Mean XCO2 (ppm)'})

# 设置标题
ax.set_title('Global Mean XCO2 Concentration', fontsize=16)

# 添加坐标轴标签
ax.set_xlabel('Longitude', fontsize=12)
ax.set_ylabel('Latitude', fontsize=12)

# 添加网格线
ax.gridlines(draw_labels=True)

# 添加注释
high_concentration_lon = 105  # 经度
high_concentration_lat = 30    # 纬度
ax.annotate('High CO2 Concentration',
            xy=(high_concentration_lon, high_concentration_lat),
            xycoords='data',
            fontsize=12,
            color='red')

# 文本框
textstr = 'Data Source:GES DISC'
props = dict(boxstyle='round', facecolor='lightyellow', alpha=0.5)
ax.text(-180, -80, textstr, fontsize=12, bbox=props)

# 添加地图特征
ax.add_feature(cfeature.COASTLINE)
ax.add_feature(cfeature.BORDERS)
```

```
# 显示图形
plt.show()
```

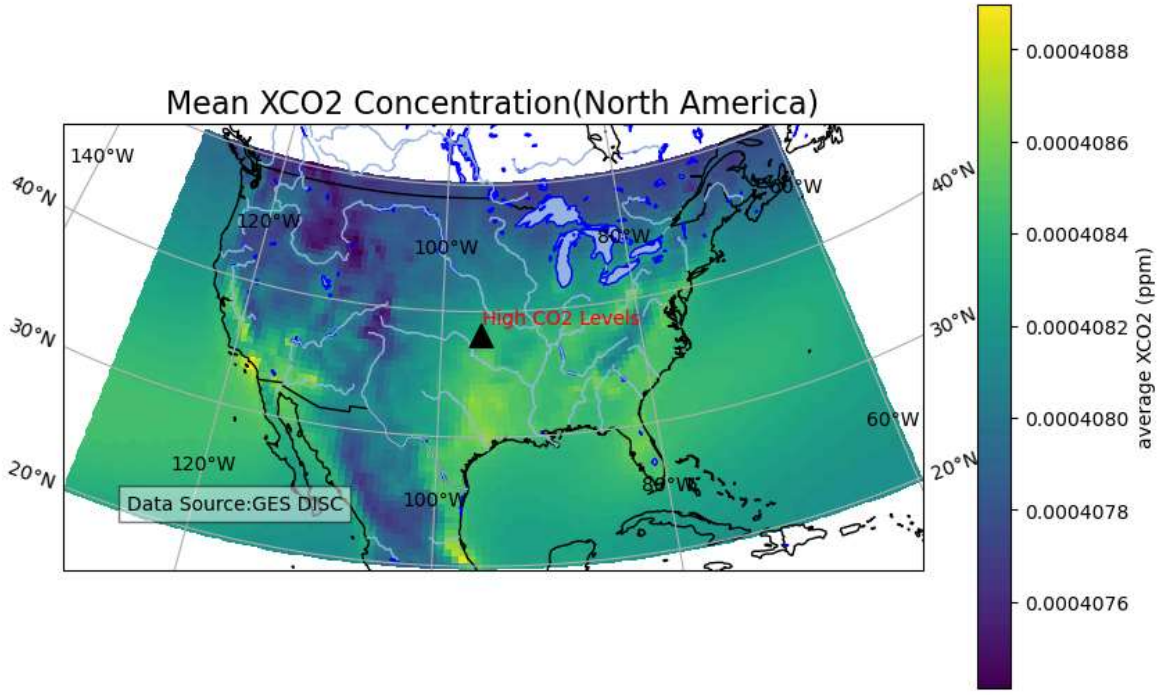


In [9]:

```
# 2.2
# 定义区域地图的范围 [西经, 东经, 南纬, 北纬]，这里我集中在北美区域
extent = [-130, -60, 20, 50]

# 选择LambertConformal投影
fig, ax = plt.subplots(figsize=(10, 8), subplot_kw={'projection': ccrs.LambertCc

# 绘制指定区域的平均 XCO2
xco2_mean.sel(lon=slice(extent[0], extent[1]), lat=slice(extent[2], extent[3])).
    ax=ax,
    transform=ccrs.PlateCarree(),
    cmap='viridis',
    cbar_kwargs={'label': 'average XCO2 (ppm)', 'shrink': 0.8}
)

# 设置标题
ax.set_title('Mean XCO2 Concentration(North America)', fontsize=16)

# 设置 x 轴和 y 轴标签
ax.set_xlabel('Longitude', fontsize=12)
ax.set_ylabel('Latitude', fontsize=12)

# 添加网格线
ax.gridlines(draw_labels=True)

# 添加地理特征
ax.add_feature(cfeature.COASTLINE)
ax.add_feature(cfeature.BORDERS)
ax.add_feature(cfeature.LAKES, edgecolor='blue')
ax.add_feature(cfeature.RIVERS)

# 添加注释
ax.annotate('High CO2 Levels', xy=(-100, 40), xytext=(-110, 45),
            arrowprops=dict(facecolor='black', shrink=0.05),
            fontsize=10, color='red')

# 将文本框放置在左下角，稍微向上移动以避免重叠
ax.text(-125, 20, 'Data Source:GES DISC', fontsize=10,
```

```
        bbox=dict(facecolor='white', alpha=0.5), transform=ccrs.PlateCarree())
# 设置 x 轴和 y 轴刻度
ax.set_xticks(range(-130, -60, 10), crs=ccrs.PlateCarree())
ax.set_yticks(range(20, 51, 5), crs=ccrs.PlateCarree())

# 显示图形
plt.show()
```

```
---------------------------------------------------------------------------
RuntimeError                              Traceback (most recent call last)
Cell In[9], line 41
     38 ax.text(-125, 20, 'Data Source:GES DISC', fontsize=10,
     39         bbox=dict(facecolor='white', alpha=0.5), transform=ccrs.PlateCarr
ee())
     40 # 设置 x 轴和 y 轴刻度
---> 41 ax.set_xticks(range(-130, -60, 10), crs=ccrs.PlateCarree())
     42 ax.set_yticks(range(20, 51, 5), crs=ccrs.PlateCarree())
     44 # 显示图形

File D:\ANACONDA\Lib\site-packages\cartopy\mpl\geoaxes.py:926, in GeoAxes.set_xti
cks(self, ticks, minor, crs)
    920 if crs is not None and crs != self.projection:
    921     if not isinstance(crs, (ccrs._RectangularProjection,
    922                             ccrs.Mercator)) or \
    923             not isinstance(self.projection,
    924                            (ccrs._RectangularProjection,
    925                             ccrs.Mercator)):
--> 926         raise RuntimeError('Cannot handle non-rectangular coordinate '
    927                            'systems.')
    928     proj_xyz = self.projection.transform_points(crs,
    929                                                 np.asarray(ticks),
    930                                                 np.zeros(len(ticks)))
    931     xticks = proj_xyz[..., 0]

RuntimeError: Cannot handle non-rectangular coordinate systems.
```



In [ ]:

In [ ]: