
# Sensing the Earth II

## 11/18/22 - Haskell Indian Nations University

Nate Quarderer (ESIIL/Earth Lab/CIRES/CU Boulder)

---

# Introduction to GIS in Open Source Python

Opening and plotting spatial data

- Vector data (shapefiles, .shp)
- Raster data (LiDAR, .tiff)

Read more about shapefiles and LiDAR data in our free, open, Earth Data Science Textbook (www.earthdatascience.org).

---

## Code of Conduct (borrowed from the Carpentries)


- Use welcoming and inclusive language
- Be respectful of different viewpoints and experiences
- Gracefully accept constructive criticism
- Focus on what is best for the community
- Show courtesy and respect towards other community members

---

# Let's get started coding

In [1]:
```
# When using Colab un-comment the following lines and run this cell
```

```
# Otherwise, skip this cell and run the next cell.

#%%capture

# Install libraries not included w/ Colab
#%pip install geopandas
#%pip install rioxarray
#%pip install earthpy
```

In [2]:
```
# Import Python libraries
import os  #for working with operating system

import matplotlib.pyplot as plt  #for general plotting
import geopandas as gpd  #for opening and plotting geodataframes
import rioxarray as rxr  #for opening raster data
import earthpy.plot as ep  #for plotting raster data
import earthpy.spatial as es #for creating hillshade from DEM/DSM
```

# Vector data (shapefiles, .shp)

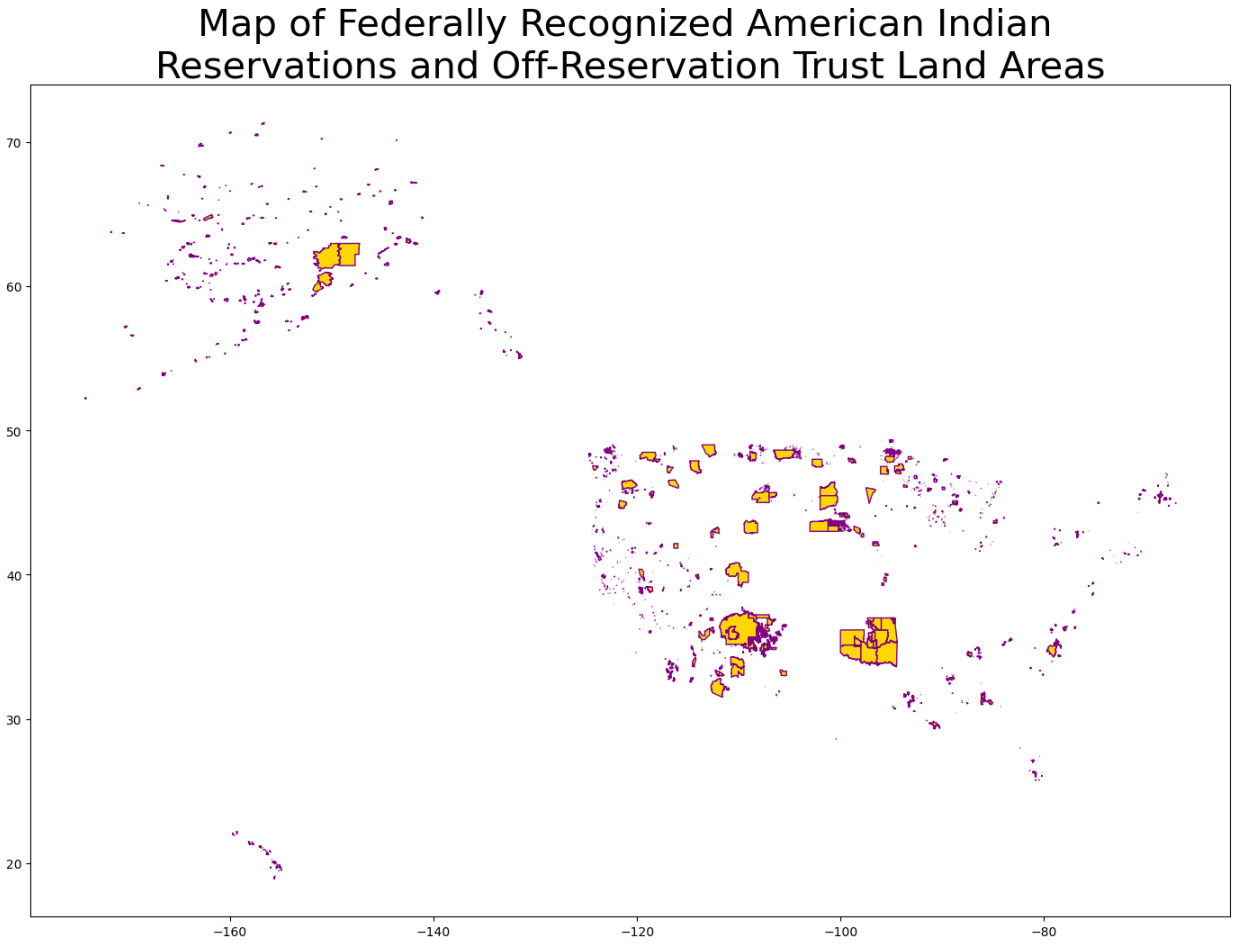
Data are being brought in as a url from the U.S. Census Bureau and data.gov, and opened using geopandas. Here we are working with the The American Indian/Alaska Native/Native Hawaiian (AIANNH) Areas Shapefile which contains boundaries of all *Federally Recognized American Indian Reservations and Off-Reservation trust land areas*.

In [3]:
```
# Open and plot AIANNH areas

# Land areas url
aiannh_url = "https://www2.census.gov/geo/tiger/GENZ2018/shp/cb_2018_us_aiannh_

# Open land area boundaries
aiannh_boundary = gpd.read_file(aiannh_url)
print(aiannh_boundary)

# Plot land area boundaries
fig, ax = plt.subplots(figsize=(20,12))
aiannh_boundary.plot(color='gold',
                     edgecolor='purple',
                     ax=ax)
ax.set_title("Map of Federally Recognized American Indian \nReservations and Of
plt.show()
```

| | AIANNHCE | AIANNHNS | AFFGEOID | GEOID | NAME | LSAD | ALAND |
|---|---|---|---|---|---|---|---|
| 0 | 0555 | 00220808 | 2500000US0555 | 0555 | Cedarville | 85 | 139809 |
| 1 | 4760 | 02419530 | 2500000US4760 | 4760 | Yurok | 86 | 217781851 |
| 2 | 2495 | 02419036 | 2500000US2495 | 2495 | North Fork | 85 | 937332 |
| 3 | 3520 | 00247949 | 2500000US3520 | 3520 | Santa Rosa | 85 | 1620467 |
| 4 | 3220 | 00231831 | 2500000US3220 | 3220 | Rohnerville | 85 | 176274 |
| .. | ... | ... | ... | ... | ... | ... | ... |
| 690 | 7565 | 02419252 | 2500000US7565 | 7565 | Telida | 79 | 4335608 |
| 691 | 7315 | 02419202 | 2500000US7315 | 7315 | Russian Mission | 79 | 14648690 |
| 692 | 6825 | 02418673 | 2500000US6825 | 6825 | Koyuk | 79 | 12337367 |
| 693 | 6480 | 02418573 | 2500000US6480 | 6480 | Emmonak | 79 | 11877389 |
| 694 | 6935 | 02418705 | 2500000US6935 | 6935 | Mekoryuk | 79 | 16472057 |

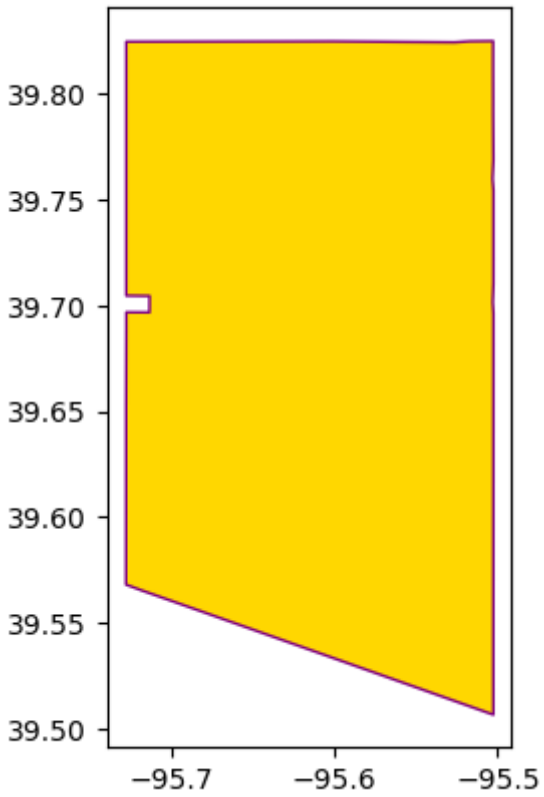
| | AWATER | geometry |
|---|---|---|
| 0 | 0 | POLYGON ((-120.18599 41.52737, -120.18323 41.5... |
| 1 | 8668719 | POLYGON ((-124.09301 41.56119, -124.08355 41.5... |
| 2 | 0 | MULTIPOLYGON (((-119.47685 37.22337, -119.4763... |
| 3 | 0 | POLYGON ((-119.76745 36.23807, -119.76522 36.2... |
| 4 | 0 | MULTIPOLYGON (((-124.12420 40.58205, -124.1212... |
| .. | ... | ... |
| 690 | 0 | POLYGON ((-153.30883 63.38777, -153.30182 63.3... |
| 691 | 945542 | POLYGON ((-161.39417 61.80578, -161.39113 61.8... |
| 692 | 0 | POLYGON ((-161.19933 64.95135, -161.19415 64.9... |
| 693 | 2052508 | POLYGON ((-164.56228 62.79060, -164.46920 62.7... |
| 694 | 33631 | POLYGON ((-166.35250 60.36385, -166.35131 60.3... |

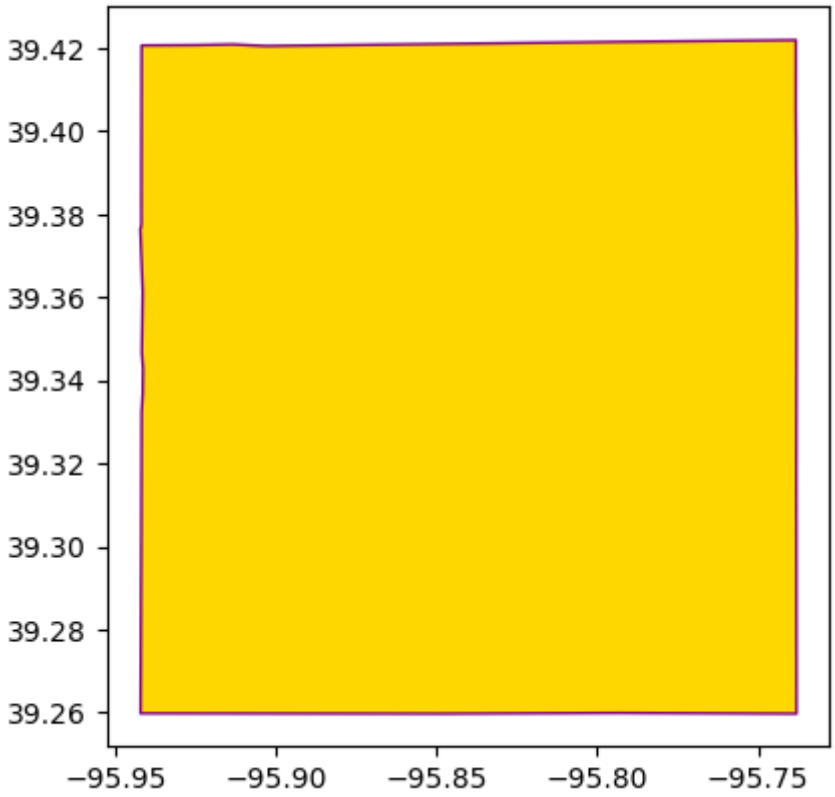
[695 rows x 9 columns]

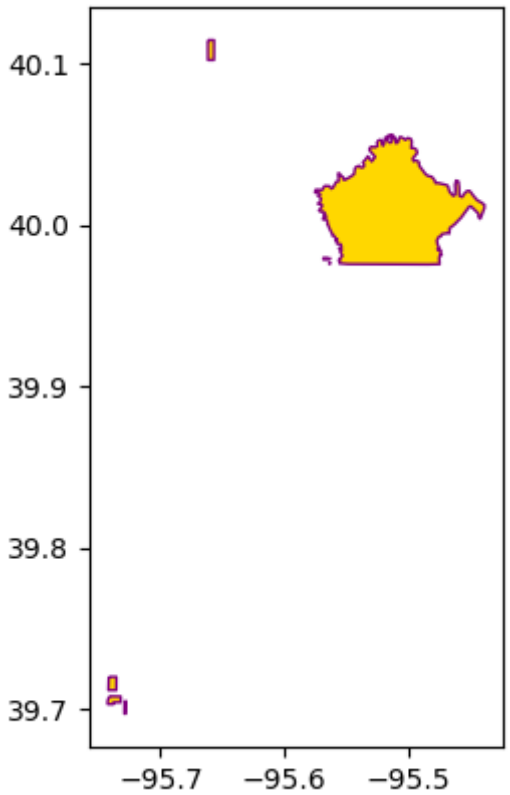


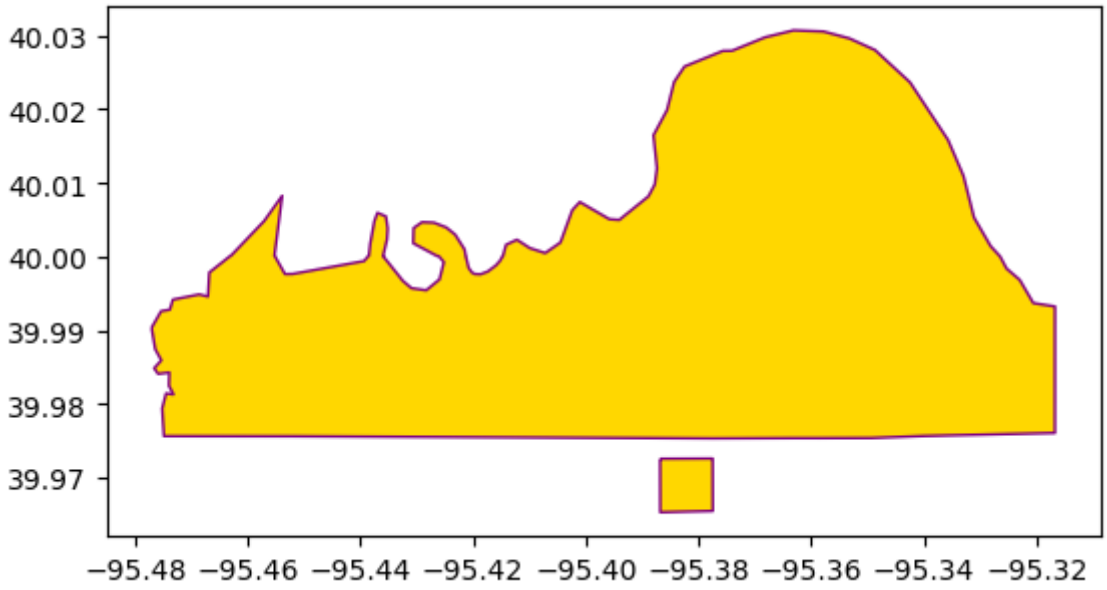
In [4]:
```
# Select and plot specified AIANNH boundaries (KS)

# Kickapoo (KS)
kickapoo_bndry = aiannh_boundary.loc[aiannh_boundary['NAME'] == 'Kickapoo (KS)'
kickapoo_bndry.plot(color='gold',
                    edgecolor='purple')

# Prairie Band Potawatomi Nation
pbpn_bndry = aiannh_boundary.loc[aiannh_boundary['NAME'] == 'Prairie Band of Po
pbpn_bndry.plot(color='gold',
                edgecolor='purple')

# Sac and Fox Nation
sfn_bndry = aiannh_boundary.loc[aiannh_boundary['NAME'] == 'Sac and Fox Nation'
sfn_bndry.plot(color='gold',
               edgecolor='purple')

# The Iowa Tribe of Kansas and Nebraska (KS-NE)
ikn_bndry = aiannh_boundary.loc[aiannh_boundary['NAME'] == 'Iowa (KS-NE)']
ikn_bndry.plot(color='gold',
               edgecolor='purple')
```

Out[4]: <AxesSubplot: >

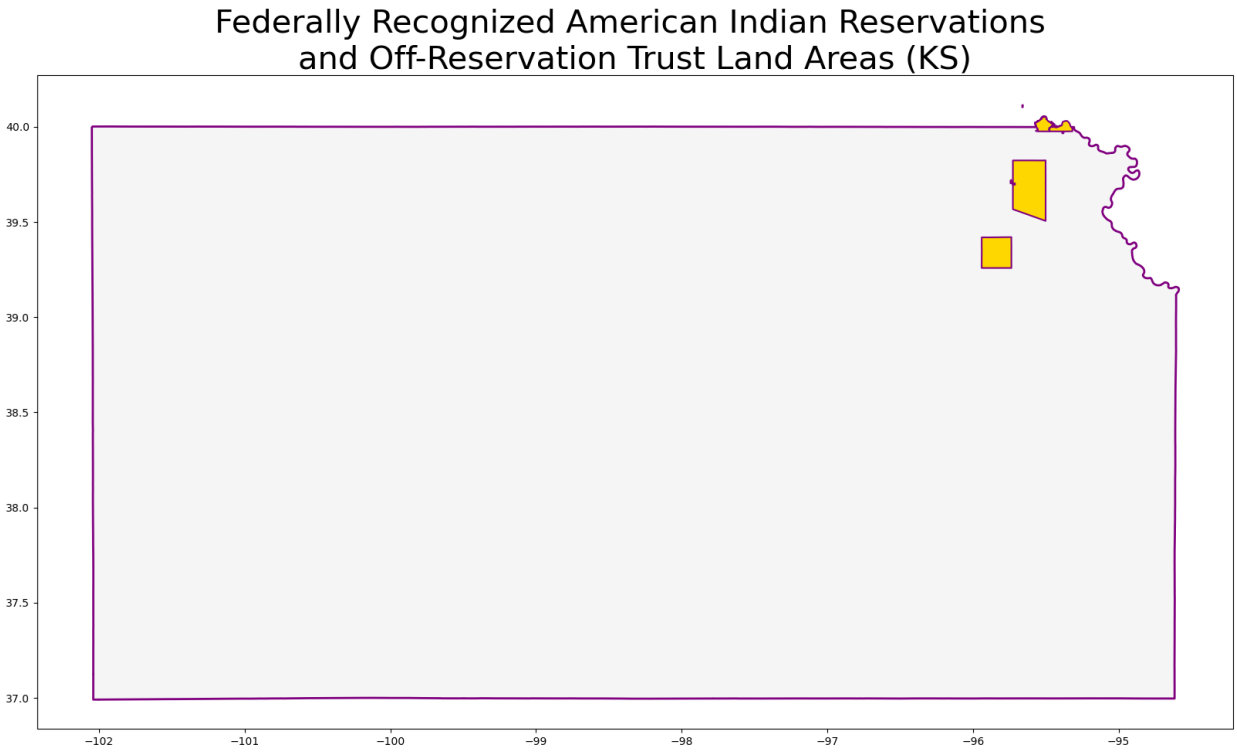



39.42
39.40
39.38
39.36
39.34
39.32
39.30
39.28
39.26
−95.95
−95.90
−95.85
−95.80
−95.75


40.1
40.0
39.9
39.8
39.7
−95.7
−95.6
−95.5




In [5]:
```
# Define urls to state boundaries
states_url = "https://www2.census.gov/geo/tiger/GENZ2018/shp/cb_2018_us_state_5

# Open state boundary data
states_shp = gpd.read_file(states_url)
states_shp

# Select specified states (KS)
ks_bndry = states_shp.loc[states_shp['NAME'] == 'Kansas']

# Plot selected AIANNH boundaries & KS boundary
fig, ax = plt.subplots(figsize=(20,12))
ks_bndry.plot(ax=ax,
              color='whitesmoke',
              edgecolor='purple',
              linewidth=2)
kickapoo_bndry.plot(ax=ax,
            color='gold',
            edgecolor='purple',
            linewidth=1.5)
pbpn_bndry.plot(ax=ax,
            color='gold',
            edgecolor='purple',
            linewidth=1.5)
sfn_bndry.plot(ax=ax,
            color='gold',
            edgecolor='purple',
            linewidth=1.5)
ikn_bndry.plot(ax=ax,
            color='gold',
            edgecolor='purple',
            linewidth=1.5)
ax.set_title("Federally Recognized American Indian Reservations \nand Off-Reser
plt.show()
```




# Raster data (LiDAR, .tiff)

Data were downloaded prior to the workshop from the Equator Studios web portal and saved within the CyVerse public file sharing environment.

**Bonus Challenge:** Click here to explore the Equator Studios website and try to download some data for a location that is important to you and your students.

**NOTE:** The National Ecological Observatory Network (NEON) also collects and hosts lidar and other spatial and ecological data through their data portal.

In [6]:
```
# Bring in Equator Studio's Hillshade Data
!wget https://data.cyverse.org/dav-anon/iplant/commons/community_released/earth
```

```
--2022-11-14 06:11:05--  https://data.cyverse.org/dav-anon/iplant/commons/comm
unity_released/earthlab/haskell/2022-11-07-A9CE8-equator-surface.tiff
Resolving data.cyverse.org (data.cyverse.org)... 206.207.252.35
Connecting to data.cyverse.org (data.cyverse.org)|206.207.252.35|:443... conne
cted.
HTTP request sent, awaiting response... 200 OK
Length: 21425395 (20M) [application/octet-stream]
Saving to: '2022-11-07-A9CE8-equator-surface.tiff'

2022-11-07-A9CE8-eq 100%[===================>]  20.43M  --.-KB/s    in 0.08s

2022-11-14 06:11:05 (270 MB/s) - '2022-11-07-A9CE8-equator-surface.tiff' saved
[21425395/21425395]
```

In [7]:
```
# Set path to data
lidar_path = os.path.join("2022-11-07-A9CE8-equator-surface.tiff")
```

```
print("the path to the lidar data is:", lidar_path)

# Open data using rioxarray
haskell_lidar = rxr.open_rasterio(lidar_path, masked=True).squeeze()
haskell_lidar
```

```
the path to the lidar data is: 2022-11-07-A9CE8-equator-surface.tiff
```

Out[7]: xarray.DataArray (**y**: 3232, **x**: 5311)

[17165152 values with dtype=float32]

▾ Coordinates:

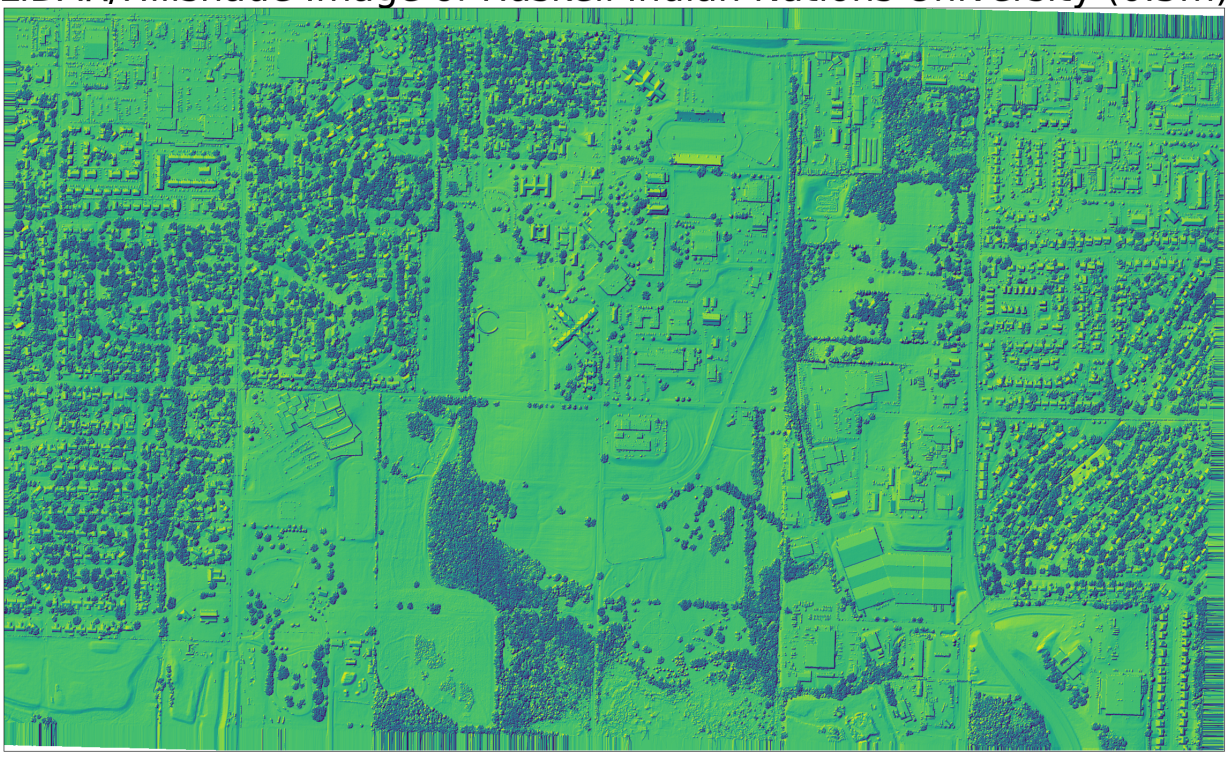
| | | | | |
|---|---|---|---|---|
| band | () | int64 | 1 | |
| **x** | (x) | float64 | 3.052e+05 3.052e+05 ... 3.078e+05 | |
| **y** | (y) | float64 | 4.313e+06 4.313e+06 ... 4.311e+06 | |
| spatial_ref | () | int64 | 0 | |

▾ Attributes:

| | |
|---|---|
| AREA_OR_POI... | Area |
| STATISTICS_M... | 255 |
| STATISTICS_M... | 154.15938934445 |
| STATISTICS_MI... | 1 |
| STATISTICS_ST... | 63.485080922075 |
| STATISTICS_VA... | 99.71 |
| scale_factor : | 1.0 |
| add_offset : | 0.0 |
| units : | metre |

In [8]:
```
# Plot data using earthpy.plot
f, ax = plt.subplots(figsize=(30,18))
ep.plot_bands(haskell_lidar,
              #alpha=.6,
              cbar=False,
              ax=ax,
              cmap='viridis')
ax.set_title("LiDAR/Hillshade Image of Haskell Indian Nations University (0.5m)
plt.show()
```

LiDAR/Hillshade Image of Haskell Indian Nations University (0.5m)



# Processing DEM/DSM to Hillshade Using Earthpy

## Opening and plotting Digital Elevation Model (DEM)

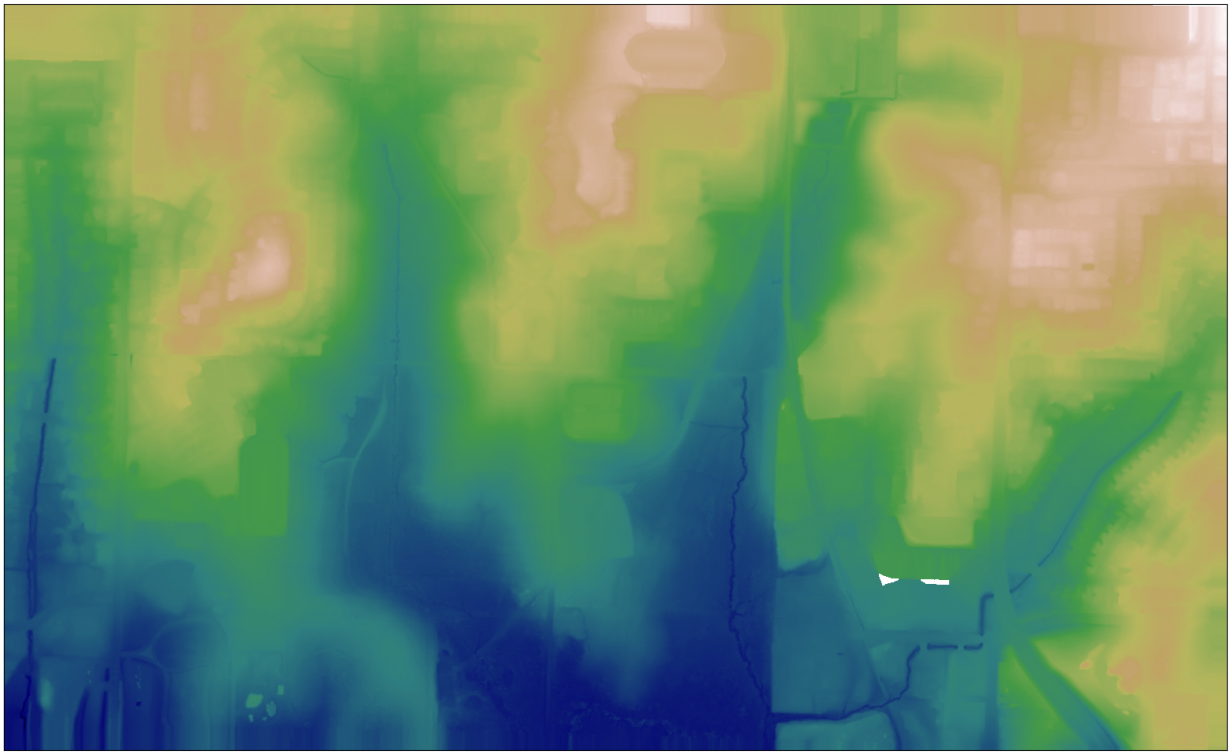
In [9]:
```
# Bring in Equator Studio's DEM data
!wget https://data.cyverse.org/dav-anon/iplant/commons/community_released/earth
```

```
--2022-11-14 06:11:08--  https://data.cyverse.org/dav-anon/iplant/commons/comm
unity_released/earthlab/haskell/2022-11-08-8B53E-equator-digital.tiff
Resolving data.cyverse.org (data.cyverse.org)... 206.207.252.35
Connecting to data.cyverse.org (data.cyverse.org)|206.207.252.35|:443... conne
cted.
HTTP request sent, awaiting response... 200 OK
Length: 51658389 (49M) [application/octet-stream]
Saving to: '2022-11-08-8B53E-equator-digital.tiff'

2022-11-08-8B53E-eq 100%[===================>]  49.26M   323MB/s    in 0.2s

2022-11-14 06:11:08 (323 MB/s) - '2022-11-08-8B53E-equator-digital.tiff' saved
[51658389/51658389]
```

In [10]:
```
# Processing DEM to hillshade

# Set path to DEM data
dem_path = os.path.join("2022-11-08-8B53E-equator-digital.tiff")
dem_path
```

```
# Open DEM data using rioxarray
haskell_dem = rxr.open_rasterio(dem_path, masked = True).squeeze()
haskell_dem

# Plot using earthpy.plot
# Add code to plot the DEM
fig, ax = plt.subplots(figsize=(20,20))

ep.plot_bands(haskell_dem,
              ax=ax,
              cmap="gist_earth",
              cbar=False)
```

Out[10]: <AxesSubplot: >

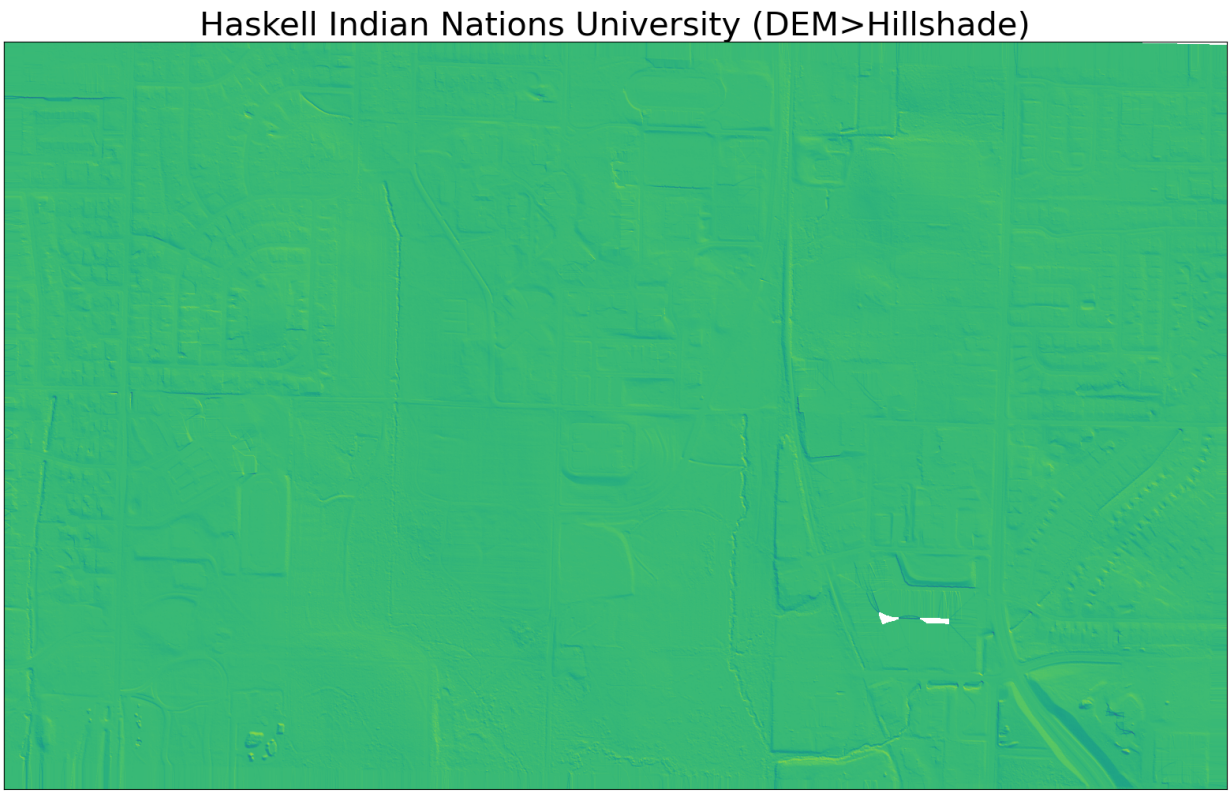


In [11]:
```
# Create hillshade from DEM with es.hillshade()
haskell_dem_hillshade = es.hillshade(haskell_dem)
haskell_dem_hillshade
```

Out[11]:
```
array([[191.05728, 191.05435, 191.05435, ...,       nan,       nan,
              nan],
       [190.81474, 190.8139 , 190.81306, ...,       nan,       nan,
              nan],
       [190.19713, 190.1986 , 190.1986 , ...,       nan,       nan,
              nan],
       ...,
       [190.41483, 190.41483, 191.66183, ..., 191.25   , 191.25   ,
        191.25   ],
       [190.41483, 190.41483, 191.66183, ..., 191.25   , 191.25   ,
        191.25   ],
       [190.41483, 190.41483, 191.66183, ..., 191.25   , 191.25   ,
        191.25   ]], dtype=float32)
```

In [12]:
```
# Add code to plot the hillshade made from DEM
```

```
fig, ax = plt.subplots(figsize=(20,20))

ep.plot_bands(haskell_dem_hillshade,
              ax=ax,
              cmap='viridis',
              cbar=False)
ax.set_title("Haskell Indian Nations University (DEM>Hillshade)",
             fontsize=30)
plt.show()
```




# Opening and plotting Digital Surface Model (DSM)

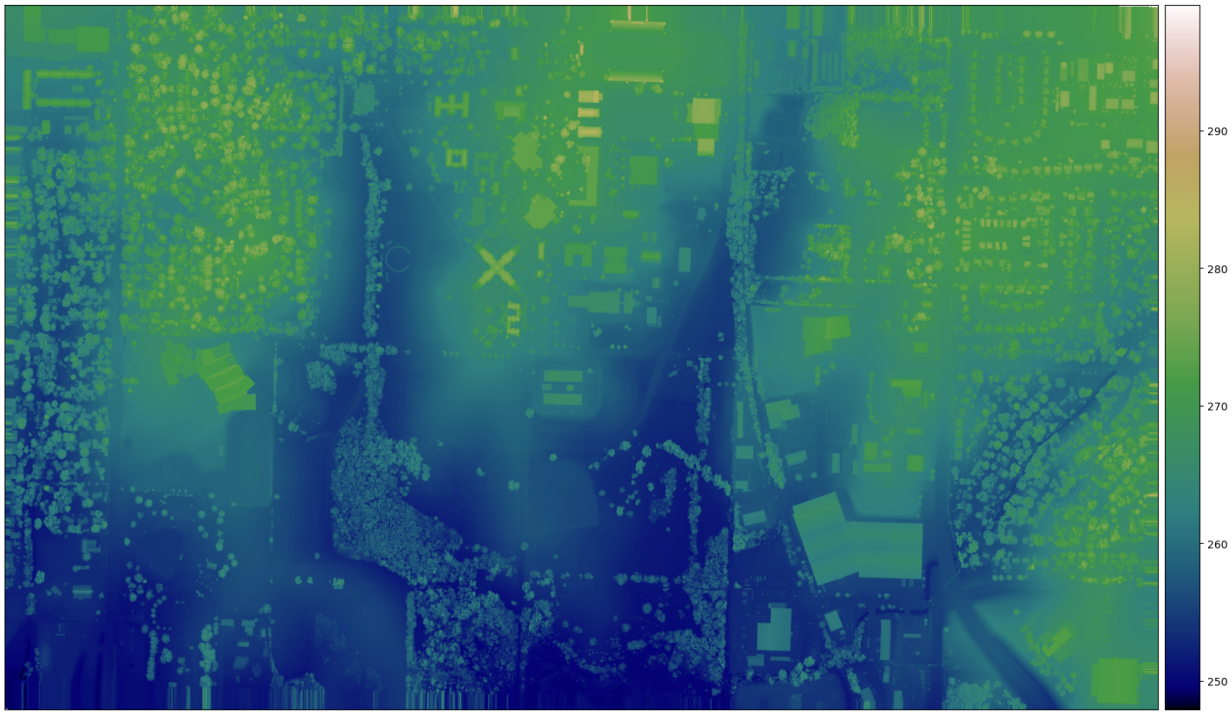
In [13]:
```
# Bring in Equator Studio's DSM data
!wget https://data.cyverse.org/dav-anon/iplant/commons/community_released/earth
```

```
--2022-11-14 06:11:12--  https://data.cyverse.org/dav-anon/iplant/commons/comm
unity_released/earthlab/haskell/2022-11-08-B4BB7-equator-digital.tiff
Resolving data.cyverse.org (data.cyverse.org)... 206.207.252.35
Connecting to data.cyverse.org (data.cyverse.org)|206.207.252.35|:443... conne
cted.
HTTP request sent, awaiting response... 200 OK
Length: 55867022 (53M) [application/octet-stream]
Saving to: '2022-11-08-B4BB7-equator-digital.tiff'

2022-11-08-B4BB7-eq 100%[===================>]  53.28M   308MB/s    in 0.2s

2022-11-14 06:11:12 (308 MB/s) - '2022-11-08-B4BB7-equator-digital.tiff' saved
[55867022/55867022]
```

In [14]:
```
# Processing DSM to hillshade
```

```
# Set path to data
dsm_path = os.path.join("2022-11-08-B4BB7-equator-digital.tiff")
dsm_path

# Open data using rioxarray
haskell_dsm = rxr.open_rasterio(dsm_path, masked = True).squeeze()
haskell_dsm

# Plot using earthpy.plot
# Add code to plot the DEM overlaid by the hillshade
fig, ax = plt.subplots(figsize=(20,20))

ep.plot_bands(haskell_dsm,
              ax=ax,
              cmap="gist_earth",
              scale=False)
```

Out[14]: <AxesSubplot: >




In [15]:
```
# Create hillshade from DSM with es.hillshade()
haskell_dsm_hillshade = es.hillshade(haskell_dsm)
haskell_dsm_hillshade
```

Out[15]:
```
array([[ 65.59727,  65.60778,  65.61866, ...,       nan,       nan,
              nan],
       [ 97.59776,  97.61395,  97.63014, ...,       nan,       nan,
              nan],
       [190.27351, 190.27351, 190.27351, ...,       nan,       nan,
              nan],
       ...,
       [190.7853 , 190.70784, 190.75679, ..., 190.914  , 190.914  ,
        190.914  ],
       [190.60133, 190.49101, 190.44057, ..., 190.36015, 190.36015,
        190.36015],
       [190.45538, 190.35088, 189.99713, ..., 190.28969, 190.28969,
        190.28969]], dtype=float32)
```

In [16]:
```
# Add code to plot the hillshade from DSM
fig, ax = plt.subplots(figsize=(20,20))

ep.plot_bands(haskell_dsm_hillshade,
              ax=ax,
              cmap='viridis',
              cbar=False)
ax.set_title("Haskell Indian Nations University (DSM>Hillshade)",
             fontsize=30)
plt.show()
```