Analysts, Data Scientists and Developers

**Analyst**
- Uses graphical tools
- Can call functions, cut & paste code
- Can change some variables

Gets paid for:
- **Insight**

*Tools:
- Excel, VB, Tableau,*
- **Python**

**Data Scientist**
- Builds simple apps & workflows
- Used to be “just an analyst”
- Likes coding to solve problems
- Doesn’t want to be a “full-time programmer”

Gets paid (like a rock star) for:
- **Code that produces insight**

*Tools:
- SAS, R, Matlab,*
- **Python**

**Developer**
- Creates frameworks & compilers
- Uses IDEs
- Degree in CompSci
- Knows multiple languages

Gets paid for:
- **Code**

*Tools:
- C, C++, Java, JS,*
- **Python**

Source: *PyCon India 2018 Python And Data: Past, Present And Future By Peter Wang*
Why Python?

- Popularity
- Productivity
- Interoperability
- Solves the “two-language” problem
- Scientific Python ecosystem
- Community
A powerful Python library for spatial analysis, mapping and GIS

ArcGIS API for Python is a Python library for working with maps and geospatial data, powered by web GIS. It provides simple and efficient tools for sophisticated vector and raster analysis, geocoding, map making, routing and directions, as well as for organizing and managing a GIS with users, groups and information items. In addition to working with your own data, the library enables access to ready-to-use maps and curated geographic data from Esri and other authoritative sources. It also integrates well with the scientific Python ecosystem and includes rich support for Pandas and Jupyter notebook.

Understand your GIS
This "hello world" style notebook shows how to get started with the GIS and visualize its contents.
• Get started with the GIS class

Manage your GIS
The ArcGIS API for Python provides APIs and samples for ArcGIS Online administrators to manage their online organization.
• Clone a portal

Perform Spatial Analysis
Call sophisticated spatial analysis tools that work with online content, using a few lines of code.
• Chennai-floods analysis

geo enrichment
widgets
geometry
mapping
geocoding
gis
geo processing
geo analytics
realtime
schematics
raster
network
features
env
ArcGIS + Jupyter = ♥
It all starts with your GIS

```python
In [1]: from arcgis.gis import GIS
In [2]: gis = GIS('https://deldev.maps.arcgis.com', 'demo_deldev')
In [3]: enterprise = GIS('https://python.playground.esri.com/portal', 'arcgis_python',
```
In [4]:
   items = gis.content.search('San Diego')

In [5]:
   for item in items:
      display(item)

Places to see in San Diego

Places to see in San Diego
Feature Collection by deldev
Last Modified: July 01, 2017
0 comments, 512 views

San Diego attractions web map

Esri Story Maps team member and San Diego resident Rupert Essinger selects some places you might enjoy.
Web Map by deldev
Last Modified: July 01, 2017
0 comments, 3 views

San Diego Trolley stations

San Diego Trolley stations
Feature Collection by deldev
Last Modified: June 23, 2017
Visualize layers on map widget

In [7]:
   
   sdmr = gis.map('San Diego', zoomlevel=14)
   sdmr

In [8]:
   
   sdmr.add_layer(sd_attractions)

In [9]:
   
   sdmr.add_layer(trolley_stations)
Spatial Analysis
Discover relationships, patterns and trends in data
ar cgis.feature submodules
Spatial Analysis Tools

- Summarize Data
- Find Locations
- Data Enrichment
- Analyze Patterns
- Use Proximity
- Manage Data
Summarize Data
Calculate summary statistics for features and attributes

res = arcgis.features.summarize_data.aggregate_points(relief_centers, chennai_pop_featurelayer, False, ['No_of_persons_Sum'])

aggr_lyr = res['aggregated_layer']

reliefmap.add_layer(aggr_lyr, {'renderer': 'ClassedSizeRenderer', 'field_name': 'SUM_No_of_persons'})
Enrich Layer
Add detailed demographic data and statistics to your analysis

```python
enriched_crime = enrich_data.enrich_layer(polynomial_beats,
                                           analysis_variables=analysis_variables)
```

Submitted.
Executing...

```python
enriched_df = enriched_crime.query().df
enriched_df.head()
```

<table>
<thead>
<tr>
<th>ASSCDEG_CY</th>
<th>AVGFMSZ_CY</th>
<th>AVGHSZ_CY</th>
<th>AVGHINC_CY</th>
<th>BACHDEG_CY</th>
<th>DIVINDX_CY</th>
<th>EDUCBASECY</th>
<th>ENRICH_FID</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.00</td>
<td>0.00</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>0.0</td>
<td>3.57</td>
<td>2.29</td>
<td>93321.0</td>
<td>16.0</td>
<td>79.6</td>
<td>30.0</td>
<td>2.0</td>
</tr>
<tr>
<td>545.0</td>
<td>2.34</td>
<td>1.35</td>
<td>99836.0</td>
<td>1142.0</td>
<td>72.8</td>
<td>12007.0</td>
<td>3.0</td>
</tr>
<tr>
<td>869.0</td>
<td>2.55</td>
<td>1.64</td>
<td>112053.0</td>
<td>8333.0</td>
<td>63.8</td>
<td>20489.0</td>
<td>4.0</td>
</tr>
</tbody>
</table>
Use Proximity
“What is near what?"

```python
from arcgis.features.use_proximity import create_buffers

# buffer the active fire boundaries and add as new content

timestamp = '{:%Y_%m_%d %H_%M_%S}'.format(datetime.datetime.now())
firebuffers = create_buffers(fires, [4], None, 'Miles', output_name="Fire_Buffers_" + timestamp)
```
Manage Data
Manage geographic data, overlay layers

```python
from arcgis.features.manage_data import overlay_layers

# run analysis to determine critical infrastructure within the risk boundaries
riskinfra = overlay_layers(firebuffers, infra,
                           overlay_type="Intersect",
                           output_name="At_Risk_Infrastructure_" + timestamp)
```
Analyze Patterns
Identify, quantify, and visualize spatial patterns in your data.

```python
from arcgis.features.analyze_patterns import interpolate_points

interpolated_rf = interpolate_points(rainfall, field='RAINFALL')
```
Geocoding

Geocoding, Batch geocoding, reverse geocoding
arcgis.geocoding module
Geocode
Single line or multi field addresses

```javascript
multi_field_address = {
  "Address": "380 New York Street",
  "City": "Redlands",
  "Region": "CA",
  "Postal": 92373
}

single_line_address = "380 New York Street, Redlands, CA 92373"

# geocode the single line address and plot the location of the first geocode result on the map
esrihq = geocode(single_line_address)[0]

# add a popup to the matched location
popup = {
  "title": "Esri Headquarters",
  "content": esrihq['address']
}
map.draw(esrihq['location'], popup)
```
Geocode
Points of Interest

- Cultural or geographic landmarks
- Businesses by name or category
- Administrative divisions

```python
# find and plot upto 100 Starbucks(TM) locations around Union Square in San Francisco, CA
starbucks = geocode("Starbucks", unionsquare['extent'], max_locations=100)
for starbuck in starbucks:
    map.draw(starbuck['location'])
```
Batch Geocoding

Geocode an entire list of single line or multi field addresses

addresses = ["300 New York St, Redlands, CA",
            "1 World Way, Los Angeles, CA",
            "1200 Getty Center Drive, Los Angeles, CA",
            "5905 Wilshire Boulevard, Los Angeles, CA",
            "100 Universal City Plaza, Universal City, CA 91608",
            "4800 Oak Grove Dr, Pasadena, CA 91109"]

results = batch_geocode(addresses)

for address in results:
    map.draw(address['location'])
Reverse Geocode
Determines address at a particular x/y location

```python
result = reverse_geocode([4.366281, 50.851994], lang_code="fr")
```

Result:

```json
{
    "address": {
        "Address": "Rue de la Sablonnière 15",
        "City": "Bruxelles",
        "CountryCode": "BEL",
        "Loc_name": "BEL.PointAddress",
        "Match_addr": "Rue de la Sablonnière 15, 1000, Bruxelles",
        "Neighborhood": "Bruxelles",
        "Postal": "1000",
        "PostalExt": None,
        "Region": "Bruxelles",
        "Subregion": "Bruxelles"},
    "location": {
        "spatialReference": {
            "latestWkid": 4326,
            "wkid": 4326,
            "x": 4.366265813154625,
            "y": 50.85196404988331}
    }
}
```

"x": 43.233701, "y": 22.993717
Av Presidente João Goulart 76, Rio de Janeiro, 22450-242
Network Analysis
Routing and directions, location allocation, service areas...
arcgis.network.analysis module
Network analysis capabilities and tools
Commercial grade, traffic aware routing and directions for multiple travel modes
Network analysis – Service Areas

```
travel_modes = sa_layer.retrieve_travel_modes()
truck_mode = [t for t in travel_modes['supportedTravelModes'] if t['name'] == 'Trucking Time'][0]

result = sa_layer.solve_service_area(fire_station_fset, default_breaks=[5,10,15],
                                    travel_direction='esriNATravelDirectionFromFacility',
                                    travel_mode=truck_mode)
```
GeoAnalytics

Fast distributed spatio-temporal analysis of large vector and tabular data

arcgis.geoanalytics module
GeoAnalytics Tools

Tools that work with tabular data that is usually spatially enabled:

- **Summarize data** – calculate descriptive statistics of features and their attributes within areas or near other features
- **Analyze patterns** – identify, quantify, and visualize spatial patterns
- **Find locations** – identify areas that meet a number of different criteria
- **Use proximity** – answer “what is near what?”
- **Manage data** – day-to-day management of geographic data
Demo

Analysis of crime patterns in Houston
Attach data

In [27]:
datastores = arcgis.geoanalytics.get_datastores()
datastores.add_bigdata('Houston_crime_yearly',
                   r'\teton\atma_shared\datasets\HoustonCrime')

Big Data file share exists for Houston_crime_yearly

Out[27]:
<Datastore title="/bigDataFileShares/Houston_crime_yearly" type="bigDataFileShare"/>

In [28]:
houston_yearly = houston_gis.content.search('Houston_crime_yearly',
                                                'big data file share')[0]
houston_yearly

Out[28]:
bigDataFileShares_Houston_crime_yearly
Big Data File Share by admin
Last Modified: March 03, 2017
0 comments, 0 views

In [30]:
houston_yearly.layers

Out[30]:
[<Layer url="https://dev003247.esri.com/gax/rest/services/DataStoreCatalogs/big
DataFileShares_Houston_crime_yearly/BigDataCatalogServer/houstoncrime2010">,
 <Layer url="https://dev003247.esri.com/gax/rest/services/DataStoreCatalogs/big
DataFileShares_Houston_crime_yearly/BigDataCatalogServer/houstoncrime2011"],
 <Layer url="https://dev003247.esri.com/gax/rest/services/DataStoreCatalogs/big
DataFileShares_Houston_crime_yearly/BigDataCatalogServer/houstoncrime2012"],
Invoke batch analytics

```
In [ ]:
for category in df.Category.unique()[:-1]:
    lyrid = 0
    for year in range(2010, 2017):
        output_name='Houston_' + category.replace(' ', '_') + '_Hotspot_' + str
        print('Generating ' + output_name)
        layer = houston_yearly.layers[lyrid]
        layer.filter = "Category={}".format(category)

find_hot_spots(layer, bin_size=0.5, bin_size_unit='Miles',
               neighborhood_distance=1, neighborhood_distance_unit='Miles',
               output_name=output_name)

lyrid = lyrid + 1
```
View results
Hot Spots across crime categories

In [33]:

display(HBox([heatmap1, heatmap2]))
display(HBox(children=[Button(description='Burglary hot spots in 2016', layout=
                          Button(description='Auto theft hot spots in 2016', layout=Layout(width='100%'))]))

Burglary hot spots in 2016
Auto theft hot spots in 2016
for year in range(2014, 2017):
    layer = houston_gis.content.search('Houston_Auto_Theft_Hotspot_ ' + str(year)
    hotspotmap = houston_gis.map(houston)
    hotspotmap.add_layer(layer)
    hotspotmap.layout=Layout(flex='1 1', padding='3px')
    maps.append(hotspotmap)
    hotspotmap.basemap='gray'
    labels.append(Button(description='Auto theft hot spots in ' + str(year),
                        layout=items_layout, button_style='danger'))

display(HBox([maps[0], maps[1], maps[2]], layout=layout))
display(HBox(children=labels, layout=Layout(width='100%')))
GeoEnrichment
Enrich your analysis with demographic and business data
arcgis.geoenrichment module
GeoEnrichment

- Get facts about a location or area
  - Street addresses
  - Points, lines and polygon geometries
  - Within a drive time or service area
  - Named geographical areas
    - Counties or block groups in California
    - Districts and subdistricts in India
- Create charts and choropleth maps

```r
# print a few rows of the DataFrame
df.head()
```

<table>
<thead>
<tr>
<th>dataCollectionID</th>
<th>analysisVariable</th>
<th>alias</th>
<th>fieldCategory</th>
<th>vintage</th>
</tr>
</thead>
<tbody>
<tr>
<td>yea#1incomes</td>
<td>yea#1incomes.AG08_CY</td>
<td>2017 Population.Age &lt;1</td>
<td>2017 Age: 1 Year increments(Equi)</td>
<td>2017</td>
</tr>
<tr>
<td>yea#1incomes</td>
<td>yea#1incomes.AG09_CY</td>
<td>2017 Population.Age 1</td>
<td>2017 Age: 1 Year increments(Equi)</td>
<td>2017</td>
</tr>
<tr>
<td>yea#1incomes</td>
<td>yea#1incomes.AG10_CY</td>
<td>2017 Population.Age 2</td>
<td>2017 Age: 1 Year increments(Equi)</td>
<td>2017</td>
</tr>
</tbody>
</table>

```
print('Number of people in fire perimeter: ', str(popdf['male'].sum()) + ' popdf['male'].sum()'))
```

```R
# Age Pyramid of affected population
print('Number of people in fire perimeter: ', str(popdf['male'].sum()))
sns.barplot(x='female', y='age label', color='#CC0000', label='female', data=popdf, edgecolor='none')
sns.barplot(x='male', y='age label', color='#4000FF', label='male', data=popdf, edgecolor='none')
plt.xlabel('Age group')
plt.xlabel('Number of people')
```
GeoEnrichment – create reports

```
# if print a sample of the reports available for USA
usa.reports.head(10)

<table>
<thead>
<tr>
<th>id</th>
<th>title</th>
<th>categories</th>
<th>format</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>census2010_profile</td>
<td>[Demographics]</td>
<td>[pdf, vsl]</td>
</tr>
<tr>
<td>1</td>
<td>aca_housing</td>
<td>[Demographics]</td>
<td>[pdf, vsl]</td>
</tr>
<tr>
<td>2</td>
<td>aca_population</td>
<td>[Demographics]</td>
<td>[pdf, vsl]</td>
</tr>
<tr>
<td>3</td>
<td>S5plus</td>
<td>[Demographics]</td>
<td>[pdf, vsl]</td>
</tr>
<tr>
<td>4</td>
<td>agesexuate</td>
<td>[Demographics]</td>
<td>[pdf, vsl]</td>
</tr>
<tr>
<td>5</td>
<td>agesex</td>
<td>[Demographics]</td>
<td>[pdf, vsl]</td>
</tr>
<tr>
<td>6</td>
<td>cnu_auto</td>
<td>[Consumer Spending]</td>
<td>[pdf, vsl]</td>
</tr>
<tr>
<td>7</td>
<td>business_loc</td>
<td>[Business]</td>
<td>[pdf, vsl]</td>
</tr>
<tr>
<td>8</td>
<td>business_summary</td>
<td>[Business]</td>
<td>[pdf, vsl]</td>
</tr>
<tr>
<td>9</td>
<td>community_profile</td>
<td>[Demographics]</td>
<td>[pdf, vsl]</td>
</tr>
</tbody>
</table>

report = create_report(study_areas=['380 New York Street, Redlands, CA'],
report='tapesty_profileNEW',
export_format='PDF',
out_folder=r'c:\xc', out_name='esri_tapesty_profile.pdf')

report
```

'C:\xc\esri_tapesty_profile.pdf'
Imagery and Raster Analysis

On-the-fly and distributed batch analysis of raster data

arcgis.raster module
Imagery and Raster Analysis

In [3]: landsat_item = gis.content.search('title:Multispectral Landsat', 'Imagery Layer', outside_org=True)[0]

In [4]: landsat_item

Out[4]:

Multispectral Landsat
Landsat 8 OLI, 30m Multispectral 8 band scenes with visual renderings and indices. Updated daily. Based on the Landsat on AWS collections.

Image Layer by esri
Last Modified: December 07, 2016
0 comments, 118,273 views

In [5]: landsat = landsat_item.layers[0]
Visualizing imagery layers

In [6]:
imagery_map = gis.map('San Diego, CA', zoomlevel=12)
imagery_map.add_layer(landsat)
imagery_map

In [27]:
for rasterfunc in landsat.properties.rasterFunctionInfos:
    print(rasterfunc.name)
    imagery_map.add_layer(apply(landsat, rasterfunc.name))
time.sleep(2)

Agriculture with DRA
Bathymetric with DRA
Custom raster processing using Raster Functions
Running in San Diego

Least Important
20%

Moderately Important
30%

Most Important
50%

Low Elevation

Flat, Not Hilly

Natural, Not Built
**Inputs - Elevation**

```python
# Digital elevation model for the US

elevation_item = enterprise.content.search('elevation_270m')
elevation_lyr = elevation_item.layers[0]
elevation_lyr
```

Out[8]:

![Map of the United States with digital elevation model](image)
Natural areas

In [9]:
# Human Modified Index imagery layer
# This dataset is based on research on the degree of human modification to
# the landscape, on a scale of 0 - 1, where 0.0 indicates unmodified natural
# landscape and 1.0 indicates the landscape is completely modified by human aci

naturalareas_item = enterprise.content.search('human_modification_index')[0]
naturalareas_lyr = naturalareas_item.layers[0]
naturalareas_lyr

Out[9]:

![Map of the United States](image_url)
Interactive raster processing in Jupyter Notebook

In [12]:
   clipped_elev = clip(elevation_lyr, sd_geom)
   clipped_elev

Out[12]:

![Map Image]
Chaining raster functions

In [14]: output_values = [1,2,3,4,5,6,7,8,9]

colormap(remap(slope(clipped_elev,
    slope_type='DEGREE',
    z_factor=1),
    input_ranges=[0,1, 1,2, 2,3, 3,5, 5,7, 7,9, 9,12, 12,15, 15,100],
    output_values=output_values),
    colormap=red_green)

Out[14]:

![Map Image]
Prepare input layers

In [17]: elevation = remap(elevation_lyr,
    [-90,250, 250,500, 500,750, 750,1000, 1000,1500, 1500,2000],
    output_values)

In [18]: terrain = remap(slope(elevation_lyr, slope_type='DEGREE', z_factor=1), # Slope
    [0,1, 1,2, 2,3, 3,5, 5,7, 7,9, 9,12, 12,15, 15,100],
    output_values)

In [19]: natural_areas = remap(naturalareas_lyr,
    [0.0,0.1, 0.1,0.2, 0.2,0.3, 0.3,0.4, 0.4,0.5,0.5,0.6, 0.6],
    output_values)
Map Algebra for the Web GIS

In [20]:
result = 0.2*elevation + 0.3*terrain + 0.5*natural_areas

In [21]:
run_raster = colormap(clip(result, sd_geom), colormap=red_green)
run_raster

Out[21]:
Visualize results using map widget

```
In [22]:
surface_map = gis.map('San Diego, CA', zoomlevel=12)
surface_map
```

```
In [23]:
surface_map.add_layer(run_raster, {'opacity': 0.6})
```
Persist results as an imagery layer

In [24]:
# Generate a persistent result at source resolution using Raster Analytics
resultlyr = run_raster.save('SanDiego_PlaceToRun')

In [26]:
resultlyr

Out[26]:
SanDiego_PlaceToRun
Analysis Image Service generated from GenerateRaster
Imagery Layer by arcgis_python
Last Modified: July 06, 2017
0 comments, 0 views
Cross country mobility

Find most efficient paths for off-road vehicles
Use Geoprocessing for onroad routing

```python
In [25]:
    from arcgis.geoprocessing import import_toolbox
    ccmurl='https://maps.esri.com/apl3/rest/services/LCP/LCP/GPServer/LeastCostPath
    ccm = import_toolbox(ccmurl)

In [26]:
def find_path(m, pt):
    m.draw(pt, symbol=finish_symbol)
    paths = ccm.least_cost_path(destination=FeatureSet([Feature(pt)]),
                              origins=origins)
    m.draw(paths, symbol=dash_dot)
```
In [28]:
    ccm_map = gis.map('San Vicente Reservoir', zoomlevel=10)
    ccm_map.on_click(find_path)
    ccm_map

In [29]:
    ramona = geocode("Ramona, CA")
    poway = geocode("Poway, CA")
    barona = geocode("Barona Reservation, CA")

    ccm_map.draw(ramona, symbol=tank_symbol)
    ccm_map.draw(poway, symbol=tank_symbol)
    ccm_map.draw(barona, symbol=tank_symbol)
In [35]: ccm_map.add_layer(surface)
Inputs
Which factors affect cross country mobility?
Terrain
Flat, rolling or steep?

In[34]:
```
dtm_sd = enterprise_b.content.search('DTM_SD')[0]
elevation = dtm_sd.layers[0]
elevation.extent = sd_extent

elevation
```

Out[34]:
Land Cover
Barren, developed or cultivated?

```
In [35]: nlcd_sd = enterprise_b.content.search('NLCD_SD')[0]
   land_cover = nlcd_sd.layers[0]
   land_cover.extent = sd_extent

   land_cover
```

Out[35]:
Vegetation characteristics
Forest, shrub or pasture?

In [36]:

```
vegetation = pd.read_csv('CCM/LUTables/veg.csv')
```

Out[36]:

<table>
<thead>
<tr>
<th>mapunit</th>
<th>mapdesc</th>
<th>ttadmapunit</th>
<th>stemd</th>
<th>stems</th>
<th>vr</th>
<th>v1</th>
<th>v2</th>
<th>f3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>42</td>
<td>Evergreen Forest</td>
<td>C</td>
<td>0.183</td>
<td>3.030</td>
<td>0.60</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>1</td>
<td>43</td>
<td>Mixed Forest</td>
<td>E</td>
<td>0.167</td>
<td>3.515</td>
<td>0.60</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>2</td>
<td>52</td>
<td>Shrub/Scrub</td>
<td>B1, B2, H</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>0.68</td>
<td>NaN</td>
</tr>
<tr>
<td>3</td>
<td>71</td>
<td>Grassland/Herbaceous</td>
<td>G1, G2</td>
<td>NaN</td>
<td>NaN</td>
<td>0.75</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>4</td>
<td>81</td>
<td>Pasture/May</td>
<td>A1</td>
<td>NaN</td>
<td>NaN</td>
<td>0.80</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>5</td>
<td>82</td>
<td>Cultivated Crops</td>
<td>A2, A3, A5, A6, A7, FC, FD, FE, L</td>
<td>NaN</td>
<td>NaN</td>
<td>0.56</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>6</td>
<td>11</td>
<td>Open Water</td>
<td>W</td>
<td>NaN</td>
<td>NaN</td>
<td>0.00</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>7</td>
<td>21</td>
<td>Developed, Open Space</td>
<td>X</td>
<td>NaN</td>
<td>NaN</td>
<td>0.00</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>8</td>
<td>22</td>
<td>Developed, Low Intensity</td>
<td>X</td>
<td>NaN</td>
<td>NaN</td>
<td>0.00</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>9</td>
<td>23</td>
<td>Developed, Medium Intensity</td>
<td>X</td>
<td>NaN</td>
<td>NaN</td>
<td>0.00</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>10</td>
<td>24</td>
<td>Developed High Intensity</td>
<td>X</td>
<td>NaN</td>
<td>NaN</td>
<td>0.00</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>11</td>
<td>31</td>
<td>Barren Land (Rock/Sand/Clay)</td>
<td>N</td>
<td>NaN</td>
<td>NaN</td>
<td>1.00</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>12</td>
<td>41</td>
<td>Deciduous Forest</td>
<td>D</td>
<td>0.150</td>
<td>4.000</td>
<td>0.60</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>13</td>
<td>90</td>
<td>Woody Wetlands</td>
<td>IC, ID, IE</td>
<td>NaN</td>
<td>NaN</td>
<td>0.40</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>14</td>
<td>95</td>
<td>Emergent Herbaceous Wetlands</td>
<td>J, K</td>
<td>NaN</td>
<td>NaN</td>
<td>0.55</td>
<td>NaN</td>
<td>NaN</td>
</tr>
</tbody>
</table>
Soil Type
Rock, clay or sand?

In [37]:
   soils_sd = enterprise_b.content.search('Soils_SDv3')[0]
   soils = soils_sd.layers[0]

   soils
Transportation infrastructure

\begin{verbatim}
In [38]: transportation = enterprise_b.content.search('RoadsAndRails')[0]
transportation
Out[38]:
RoadsAndRails_SD
Binary Roads and Rails for SD county
Imagery Layer by bgerlRa
Last Modified: July 18, 2017
0 comments, 4 views
\end{verbatim}
# Vehicle Characteristics

**Jeep, truck or tank?**


```
In [39]: import pandas as pd
vehicle_characteristics = pd.read_csv('CCM/LUTables/VCTable.csv')
vehicle_characteristics
```

<table>
<thead>
<tr>
<th></th>
<th>name</th>
<th>maxkph</th>
<th>onslope</th>
<th>offslope</th>
<th>vehiclewidth</th>
<th>overriddeniameter</th>
<th>vc1</th>
<th>vci50</th>
<th>minturnrad</th>
<th>vehiclelength</th>
<th>mlc</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>M1</td>
<td>71.0</td>
<td>68.7</td>
<td>53.0</td>
<td>3.65</td>
<td>0.25</td>
<td>25.0</td>
<td>58.0</td>
<td>9.90</td>
<td>9.90</td>
<td>60.0</td>
</tr>
<tr>
<td>1</td>
<td>M60A1</td>
<td>48.0</td>
<td>60.0</td>
<td>45.0</td>
<td>3.63</td>
<td>0.15</td>
<td>20.0</td>
<td>48.0</td>
<td>9.40</td>
<td>9.40</td>
<td>54.0</td>
</tr>
<tr>
<td>2</td>
<td>M109</td>
<td>56.0</td>
<td>60.0</td>
<td>45.0</td>
<td>3.10</td>
<td>0.12</td>
<td>25.0</td>
<td>57.0</td>
<td>6.60</td>
<td>6.60</td>
<td>24.0</td>
</tr>
<tr>
<td>3</td>
<td>M113</td>
<td>48.0</td>
<td>60.0</td>
<td>45.0</td>
<td>2.69</td>
<td>0.10</td>
<td>17.0</td>
<td>40.0</td>
<td>4.00</td>
<td>4.00</td>
<td>12.0</td>
</tr>
<tr>
<td>4</td>
<td>M35A2</td>
<td>56.0</td>
<td>64.0</td>
<td>30.0</td>
<td>2.43</td>
<td>0.06</td>
<td>26.0</td>
<td>59.0</td>
<td>5.30</td>
<td>6.70</td>
<td>10.0</td>
</tr>
<tr>
<td>5</td>
<td>M191</td>
<td>50.0</td>
<td>60.0</td>
<td>28.0</td>
<td>1.69</td>
<td>0.04</td>
<td>19.0</td>
<td>44.0</td>
<td>5.80</td>
<td>3.35</td>
<td>NaN</td>
</tr>
<tr>
<td>6</td>
<td>T62</td>
<td>50.0</td>
<td>62.0</td>
<td>45.0</td>
<td>3.37</td>
<td>0.15</td>
<td>21.0</td>
<td>49.0</td>
<td>9.33</td>
<td>9.33</td>
<td>42.0</td>
</tr>
<tr>
<td>7</td>
<td>T72</td>
<td>60.0</td>
<td>62.0</td>
<td>45.0</td>
<td>3.38</td>
<td>0.18</td>
<td>25.0</td>
<td>60.0</td>
<td>9.20</td>
<td>9.20</td>
<td>45.0</td>
</tr>
</tbody>
</table>
Raster function chain for cross country mobility
Generate cost surface

```
In [54]: with open("CCM_FunctionChain.rft.xml", "r", encoding='utf-8-sig') as rft:
    raster_fn = rft.read()

In [55]: %time
    from arcgis.raster.analytics import generate_raster
    surface = generate_raster(raster_fn, output_name='Cross_Country_Mobility')

Wall time: 5min 48s

In [56]: surface
```

```
Out[56]:
```

**Cross_Country_Mobility**
Analysis Image Service generated from GenerateRaster
Imagery Layer by rsinghRA
Last Modified: October 22, 2017
0 comments, 0 views
In [57]:
cost_surface = surface.layers[0]
cost_surface.extent = sd_extent

cost_surface

Out[57]:
Spatial Data Frame

Feature layers as a Pandas dataframe

arcgis.features.SpatialDataFrame class
Demo

Using SpatialDataFrame
Data Science
Find the patterns hidden in data
Data Science with ArcGIS - Data

- **Esri curated content – Living Atlas**
  - Multi-spectral, temporal, dynamic imagery layers
  - Landsat, NAIP, MODIS, Elevation
  - Basemaps, Imagery, Demographics, Transport
  - Boundaries & places, Landscape, Oceans
  - Earth Observations, Urban Systems, Historical Maps, …

- **Your data, org's data, data shared with you**
  - Shapefiles, File geodatabase, CSV, Excel, HTML, …
  - File shares, cloud share
  - HDFS, Hive and databases

- **Public data**
  - Maps, layers and datasets shared by users worldwide
Data Science with ArcGIS - Visualization

• Visualize with ArcGIS
  - Map widget in Jupyter notebook
  - Web Maps and Web Scene
  - Feature layers
  - Raster and imagery layer
  - Smart mapping
  - Pythoic renderers and symbology

• Visualize with Python
  - Matplotlib, Seaborn, Bokeh, Plotly, …
  - Datashader, Holoviews, Mayavi, …
Data Science with ArcGIS - Analysis

• Analysis with ArcGIS
  - Geoprocessing in Web GIS
    - Spatial analysis, Routing and directions
    - Network analysis, Geocoding, Geoenrichment…
  - Imagery and Raster Analysis
    - On the fly dynamic image processing
    - Distributed raster analysis
  - GeoAnalytics – large tabular and vector data
Data Science with ArcGIS - Analysis

• Analysis with Python libraries
  - Data wrangling
    - Pandas, numpy, scipy
  - Machine learning
    - Scikit-learn, tensorflow, keras, pytorch, fastai
  - Geospatial analysis
    - PySAL, GDAL, Shapely, Fiona, …
  - Image processing and computer vision
    - PIL, OpenCV, scikit-image
Data Science workflows - Deployment

• Deploy as information products
  - WebMaps
  - Web scenes
  - Layers
• Deploy as web tools
  - Geoprocessing script tools
  - Binder projects
• Deploy as dashboards
  - ArcGIS Operations Dashboard
  - Jupyter dashboard
  - Plotly dashboard
Questions?
Please fill out the surveys!