Workng with Vectors in arcpy

Working with Vectors in arcpy

Overview

1. Using More Toolbox Tools
2. Feature Layers
3. SQL Queries with arcpy
4. Selecting Data
5. An Example Vector Workflow

Using More Toolbox Tools

Using More Toolbox Tools

- Online documentation of all tools is geared toward arcpy
    - See the Tool Reference

Layers and Views in arcpy

Layers and Views in arcpy

- When a dataset is opened in ArcMap it becomes a Layer
- A table is opened as a View
- Layers and Views allow some non-destructive operations
    - Selection
    - Symbolization
- Layers do not persist after a session is closed unless explicitly saved

Layers and Views in arcpy

- Many types of layers/views:
    - Feature Layer
    - Image Server Layer
    - LAS Dataset Layer
    - Mosaic Layer
    - Query Layer
    - Query Table
    - Raster Catalog Layer
    - Raster Layer
    - Table View
    - WCS Layer
    - XY Event Layer

Layers and Views in arcpy

- Create a feature layer in ArcMap by adding feature class to map document
- How to create in arcpy?

```
>>> import arcpy

# an point feature class
>>> featureclass = r"C:\GIS\Data\NatlForests.gdb\Trees"

>>> featurelayer = arcpy.MakeFeatureLayer_management(featureclass,
                                                     "TreesLayer")


>>> featurelayer
(Result 'TreesLayer')

# the layer is a Layer object from the arcpy.mapping module
>>> featurelayer[0]
(map layer u'TreesLayer')
```

Layers and Views in arcpy

- Most tools will accept a layer/view or a path
    - In a tool GUI, when you click a drop-down to select a dataset, that is one of the open layers in the map document
    - Can also browse for a dataset, which supplies a path
- Some tools work explicitly with layers

SQL Queries

SQL Queries

- ArcGIS uses SQL as its query language, with some rather variable syntax:
    - A field name in a file geodatabase is wrapped in `""`
    - Fields names in personal geodatabase are wrapped with `[]`
    - SDE doesn't have field name delimiters
    - In SDE, table names are of the format `<database_name>.<schema>.<table_name>`
    - Joined field names are similarly difficult: `<orig_tbl>.<field_name>`

SQL Queries

- Field name delimiters are easily dealt with:

```
>>> import arcpy

>>> data = r"C:\Data.gdb\Marshes"

>>> query = "{statefld} = '{val1}' AND {areafld} >= {size}"\
                .format(arcpy.AddFieldDelimiters(data, "STATE"),
                        "OR",
                        arcpy.AddFireldDelimiters(data, "SHAPE_AREA"),
                        10000)

>>> query
'"STATE" = \'OR\' AND "SHAPE_AREA" >= 10000'
```

SQL Queries

- SDE table names and joined fields are not dealt with easily
- Three ideas:
    - The code is specific to a given application, so hard code the table name: `datastore.DBO.CoffeeShops`
    - Use the listing functions: `field_name = arcpy.ListFields(table, "*" + search_field_name)`
    - Maybe enough information can be gathered to reconstruct the field name: `field_name = table_name + "." + search_field_name)`

Selecting Data

Selecting Data

- Four ways to select data:
    - Definition Query when making a feature layer/table view
    - By Attributes
    - By Location
    - Using `arcpy.Select_analysis()`
      (Not to be confused with `arcpy.SelectData_management()`)

Selecting Data

- Remember: definition queries, select by attributes, select by location do not create a permanent selection
  - Use `arcpy.CopyFeatures_management()` to copy selected features to a new layer
  - Use `arcpy.DeleteFeatures_management()` to delete selected features from the original data
    - Caution: this IS permanent
    - Not to be confused with `arcpy.Delete_management()`
- Use `arcpy.GetCount_management()` to check if any features were selected

An Example Vector Workflow

An Example Vector Workflow

Scenario

You work for a regional water provider which has a file geodatabase with data representing the water system. Your supervisor has asked you to create a dataset representing the area within 100 feet of any active main segment that has experienced a shear break leak.

An Example Vector Workflow

```python
import arcpy

mains = r"C:\Data\Water.gdb\Mains"
leaks = r"C:\Data\Water.gdb\Leaks"
output = r"C:\Data\Analysis.gdb\Leak_areas"

# def query to get active mains
lyr_mains = arcpy.MakeFeatureLayer_management(mains,
                                              "lyr_mains",
                                              "\"Status\" = 'ACTIVE'")

# def query to get shear breaks
lyr_leaks = arcpy.MakeFeatureLayer_management(leaks,
                                              "lyr_leaks",
                                              "\"Type\" = 'SHEAR_BREAK'")

# select by location to get leaky mains
leaky_mains = arcpy.SelectLayerByLocation_management(lyr_mains,
                                                     "INTERSECT",
                                                     lyr_leaks)

# if leaky mains, buffer and dissolve all buffers
if int(arcpy.GetCount_management(leaky_mains).getOutput(0)):

    # "#" in an arcpy function means use default
    arcpy.Buffer_analysis(leaky_mains, output, "100 FEET", "#", "#", "ALL")
```