Address Parsing Tool Guide

The Address Parsing Tool parses a full site address field into individual elements. It will parse any address contained as a field in a file geodatabase feature class into individual elements that fit the FGDC's United States Thoroughfare, Landmark, and Postal Address Data Standard.

Use this guide if your county's parcel site address data is not available as fully parsed address elements meeting the statewide parcel schema.

The tool, which is run in several cycles, is designed to accomplish as much automatic parsing as feasible. However, for some parcel feature's addresses, the workflow may require altering an input site address or manual parsing.

Why is This Guide Necessary?

The attribute schema of the statewide parcel layer requires parcel site addresses and sub-address elements to be modeled according to FGDC and other standards.

For those counties who wish to use Esri's ArcGIS geocoding tool called "Standardize Addresses," be aware that the tools from SCO—the Address Parsing Tool, and the Data Standardize Tool—were created to overcome some of the weaknesses in the Esri tool:

- The Esri tool does not parse to the FGDC addressing standard, but rather to Esri address locators
- Highways are intermingled with street names (when they should be included as street prefixes)
- Some prefix directions are incorrectly included as street names
- Some street names are incorrectly included as sub address type (unit type)
- Some street names and street types are incorrectly included as sub address unit (unit ID)
- Some street names split incorrectly and are distributed to various incorrect elements
- Grid addresses numbers are not parsed according to the FGDC standard

The Address Parsing Tool will parse addresses but achieves limited element standardization. To meet the standardization requirements of the Searchable Format, users are encouraged to employ the Data Standardize Tool.

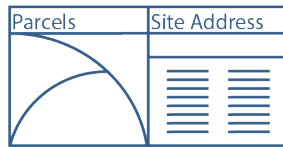**How to Use This Guide**

It is recommended you have the following materials at hand:
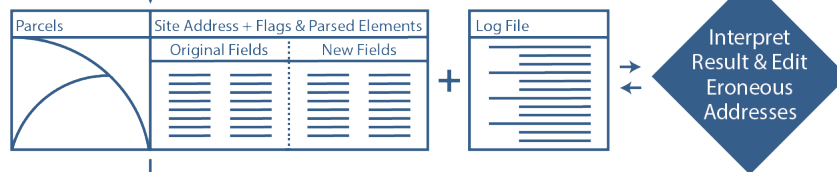
- The Submission Documentation and webpage
- Your GIS parcel data – Must be in file geodatabase feature class form but can be of any geometry type (point, line, polygon)
- The county staff member(s) who will submit the data
- The Address Parsing Tool from www.sco.wisc.edu/parcels/tools
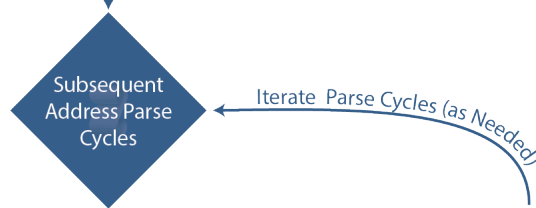
Figure 1. Address Parsing Tool workflow overview

❶ Install the Tool

1.1 Download and unzip the tool

Download the zipped package with the Address Parsing Tool, an ArcPy script tool, from
www.sco.wisc.edu/parcels/tools.

Unzip to the directory of your choice. Then open ArcCatalog and navigate to the new directory. You should see toolboxes, labeled with their respective ArcGIS version compatibility. Choose the toolbox that fits your ArcGIS install. If you do not see any toolboxes, hit F5 to refresh the directory. Once the tools are visible, move on to the next step.

1.2 Including Python library dependencies

The tool will not be ready to run immediately. Instead, a directory of Python library files need to be placed within your ArcGIS 10.X Python package. Note that copying files to this directory may require administrative privileges to the machine.

To install the files:
- Navigate to the directory the tools were unpacked in. Included within the .zip file is a directory called */libraries/.*
- Copy the **contents** of the */libraries/* directory.
- Navigate to your ArcGIS 10.X install of Python27. The full path to this directory may vary depending upon where ArcGIS is installed. A typical path to the Python27 install will look like this:
    - **C:\Python27\ArcGIS10.3**
    - If you cannot find the path to your ArcGIS 10.X install of Python27, from ArcCatalog, open the Python Window. Within the Python Window, enter:
        - **import re** (and hit enter)
        - **re.__file__** (and hit enter)
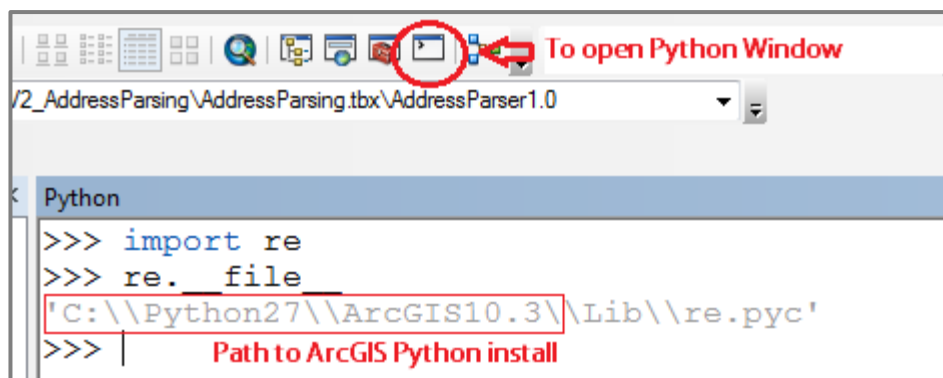 Figure 2 illustrates what this would look like in the ArcCatalog Python Window.

Figure 2. Locating Path to ArcGIS 10.X Install of Python27

- Once the path to Python27 is identified, navigate to this path, making sure to follow the path of the ArcGIS version that you are currently running. Navigate to the \Lib\site-packages\ sub-directory.
    - For example: C:\Python27\ArcGIS10.3**Lib\site-packages**
    - Copy and paste the contents of */libraries/* from the .zip file into this directory

The tool should now be ready to run. If you experience trouble starting or executing the tool, see the Troubleshooting section below.
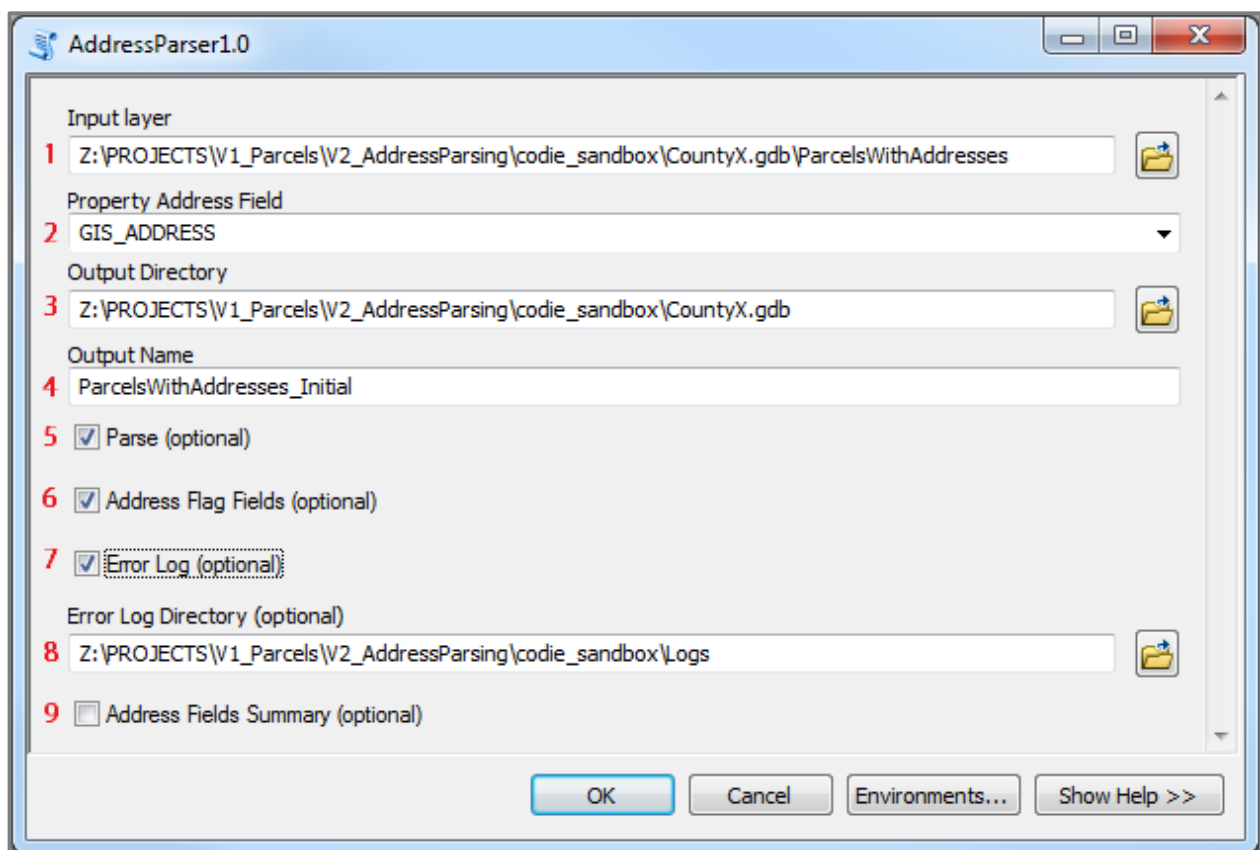
1.3 Tool overview

The Address Parsing Tool is designed to parse a site address within a file geodatabase feature class of type point, polygon, or line. The tool has the ability to create:

- A copy of the input feature class with:
    - New address fields containing the address elements parsed by the tool.
    - Address Flag Fields containing information to identify poorly constructed addresses. Note that new flag and address fields will be cleared and overwritten when subsequent parses are run over a previous tool's output feature class.
- An **Error Log** as output text file, containing information on addresses the tool was unable to parse.

As some outputs are optional, choosing to exclude optional elements may allow the tool to execute faster.

Figure 3 depicts the tool's parameters.

Figure 3. Address Parsing Tool parameters

Address Parsing Tool – Parameter Definitions		
1	**Input Layer**	The feature class containing the site address field to be parsed.
2	**Property Address Field**	The field containing the site address.
3	**Output Directory**	The path specified for the output feature class. This must be a path to a file geodatabase, or file geodatabase feature dataset.
4	**Output Name**	The name given to the output feature class and error log file (if applicable).
5	**Parse (Optional)**	Optional functionality to parse the site address field into the appropriate address elements. By default, this field is checked. **Parse** adds these fields to the feature class: • NEW_ADDNUMPREF • NEW_ADDNUM • NEW_ADDNUMSUFF • NEW_PREFIX • NEW_STREETNAME • NEW_STREETTYPE • NEW_SUFFIX • NEW_UNITTYPE • NEW_UNITID Consult the Address Reference Guide for definitions of address fields that are added to the feature class.
6	**Address Flag Fields (Optional)**	Optional functionality to create additional fields in the output feature class containing detail on potential poorly constructed addresses. By default, this field is left unchecked. **Address Flag Fields** generates the following: • **Character_Flag**: Any characters found in the site address that are not found in properly constructed addresses. Some characters flagged in this field are acceptable depending on the location of the character within the address. For example, hyphens in an address number range (100-110 Main St.) are unacceptable, but in a unit ID range (100 Main St. Apt 1-10) they are acceptable. See the Address Reference Guide for specific information regarding the character in question. • **Incomplete_Data_Flag**: An indication of potential missing address elements based on the notion every address must have an address number and street name. Valid addresses must contain these elements. See the Address Reference Guide for more information on required address elements. • **Parse_Error_Flag**: A numeric field indicating parse error with a "1" or a "0" / Null for no parse error. • **Extraneous_Data**: Any elements not parsed to schema address elements are dumped into this field. The field often contains data not included in a valid address and can be removed from the site address altogether. However, there are cases where an address will be parsed, but parsed incorrectly placing some elements in this field. See the Address Reference Guide for more information regarding extraneous address data.
7	**Error Log (Optional)**	Optional functionality to output a text file containing information on addresses the tool was unable to parse. By default, this option is unchecked.
8	**Error Log Directory (Optional)**	The path for the output error log file. This field only needs to be completed if the **Error Log** checkbox is selected. If left blank, the default directory is the ArcGIS current workspace.
9	**Address Fields Summary (Optional)**	Optional functionality to summarize address fields.

❷ Run initial address parse (to locate addresses in need of correction)

2.1 Run initial parse

If the addresses to be parsed were constructed in a consistent, conventional, error-free manner, it may only be necessary to run the Address Parsing Tool once. However, **if the addresses contain extraneous data, or are occasionally incomplete or erroneous, this tool will help identify where problematic addresses exist, so that they can be corrected within the full (input) site address and parsed again**. This tool may be able to uncover problematic addresses that would otherwise go unnoticed.

To begin the parsing workflow and establish a baseline for site address corrections needed in the file geodatabase feature class, run an initial address parse. When running an initial address parse, the optional parameters should be checked, as shown in Figure 4.
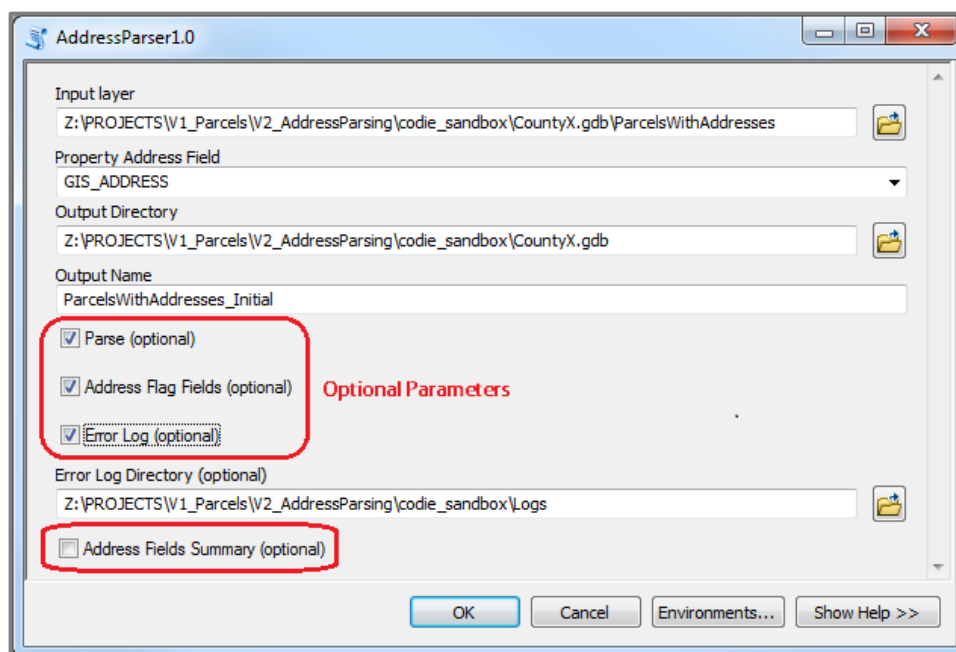

Figure 4. Select optional tool parameters for initial parse

2.2 Interpret results of initial parse

Upon completion of the initial parse, the parse tool will output a new feature class and error log.

First, find addresses the tool was unable to parse, by going into the attribute table of the output feature class and sorting the Parse_Error_Flag field. A value of "1" indicates a parse error, while a value of "0" or Null means the address was parsed correctly or skipped, respectively.

Make edits in the site address field to allow for proper parsing. To get a better understanding of why a parse error occurred, the **Error Log** provides additional information about the parse attempted and where in the address an error occurred.

At the bottom of each log is a count of the total number of parse errors found. Figure 5 shows a sample Error Log with a single parse error.

```
----------------------------------- OBJECTID: 153 -----------------------------------

ERROR: Unable to tag this string because more than one area of the string has the same label

ORIGINAL STRING:  586 W 10TH ST & 729 S Thunderbird AVE
PARSED TOKENS:    [(u'586', 'AddressNumber'), (u'w', 'StreetNamePreDirectional'), (u'10TH', 'StreetName'), (u'ST', 'StreetNamePostType'),
(u'&', 'AddressNumber'),(u'729', 'AddressNumber'),(u's', 'StreetNamePreDirectional'), (u'Thunderbird', 'StreetName'), (u'AVE', 'StreetNamePostType')]
UNCERTAIN LABEL:  AddressNumber

When this error is raised, it's likely that either (1) the string is not a valid address or (2) some tokens were labeled incorrectly

To report an error in labeling a valid address, open an issue at https://github.com/datamade/usaddress/issues/new - it'll help us continue to improve usaddress!

For more information, see the documentation at http://usaddress.readthedocs.org/


Total Number of Parse Errors: 1
```

Figure 5. Sample Error Log output

Each log record begins with the **OBJECTID** of the record that could not be parsed, followed by a statement about the error. Most errors are caused when the parser tries to place more than one address element into the same address field.

In Figure 5, the parser encountered an address in the form of an intersection, making the parser try to assign multiple elements to the **AddressNumber** field. Standards dictate that an intersection is not an acceptable address. In order to remedy this particular error, one address must be removed from the site address field. Consult the Address Reference Guide for more specific information regarding standards and address formatting.

Sometimes the error will indicate elements parsed to address elements not found in the schema. This is a result of using a separate Python module to complete the parsing.

The error log option should be used as many times as necessary, until all parse errors are removed from the data.

Look through the Extraneous_Data field after the first iteration of the address parser. Any address elements that do not fit the schema are placed in this field. Like with parse errors, sort by the field to find the extraneous data. There are two scenarios in which data ends up in the **Extraneous_Data** field:
- Data is not considered part of an address (e.g., a property descriptor such as "barn")
- Address elements parsed incorrectly

Make note of any records in which adjustments made to the site address still result in errors for the final step of the address parsing process.

❸ Run subsequent address parse cycles (until errors and flags are resolved)

3.1 Configure another Address Parsing Tool cycle

After all problems identified by the **Error Log** have been eliminated or corrected, configure another parse through the Address Parsing Tool. This time, the **Error Log** option may be left unchecked.

This parse should be executed on the output of the previous parse (or the feature class last edited). It is recommended that a versioned naming convention be adopted to keep input and output feature classes and their versions in order, as depicted below.

Feature Class Input Name	Cycle	Feature Class Output Name
ParcelsWithAddresses	Initial Parse ›	ParcelsWithAddresses_Initial »
» ParcelsWithAddresses_Initial	Cycle 1	ParcelsWithAddresses_v1
ParcelsWithAddresses_v1	Cycle 2	ParcelsWithAddresses_v2
ParcelsWithAddresses_v2	Cycle 3	ParcelsWithAddresses_v3

3.2 Interpret results of parse cycle

Again, look for incorrectly parsed addresses in the Extraneous_Data field. This field may contain different address elements depending on the adjustments made from the first tool iteration. Similar steps should be taken with addresses that contain extraneous data as above (section 2.2) in when interpreting results of a parse.

Make corrections based on extraneous data.

Next, sort on the Character_Flag field. This field will include a list of characters found in the address that are not acceptable in a properly formatted address. In some specific cases, characters flagged by the tool are acceptable. Consult the Address Reference Guide for details regarding character acceptance.

Remove unacceptable characters from the site address(es).

Sort through the Incomplete_Data_Flag field to locate missing elements. In this field, comments will indicate if an address is potentially missing a key element (e.g., an address number, street name, or street type). These results are based on the initial parse and do not include post-parsing processing within the tool.

This process should be followed with each iteration of the parse cycle.

Once no new flags have been thrown and the rest of the flags accounted for, it is time to move to the final parsing step.

3.3 Final run

To configure the parse tool, check the **Parse** option and **Address Fields Summary** option, and execute the final run.

❹ Manual parsing (if necessary)

4.1 Identifying manual edits

The final run of the Address Parsing Tool will contain parsed addresses in their most accurate form. However, it is possible that a valid address is an anomaly to the logic within the Address Parsing Tool. **The only way to handle especially challenging addresses is through manual editing of the parsed address fields.**

Having a list of known records from previous steps can help with the identification process. The text file outputs from the **Address Fields Summary** option are another source for locating elements in fields they should not be in.

It is important to scan through all of the records to confirm an accurately parsed dataset. If a correct address is identified to be parsed incorrectly or causing an error, make note of the address (in tabular form if several exist) and submit them to the SCO. The project team will work to better support these addresses.

For your records, save a copy of the table of parsed addresses along with parcel IDs.

❺ Proceed in creating Searchable Format Submission

Troubleshooting for Address Parsing Tool

My tool does not show-up in the directory I have unpacked to.

If the tool does not appear in the directory you have unpacked it to, first try refreshing the directory in ArcCatalog (Right click directory » Refresh…). If the problem continues, it may be because you are using a legacy version of ArcGIS (ArcGIS 9.0 – ArcGIS 10.1). Tools supporting ArcGIS legacy versions are available at www.sco.wisc.edu/parcels/tools

I get an error when attempting to run the tool.

If running the tool results in an error, first ensure that the tool runs correctly on the test data provided in the zipped package.

- If the tool does not run successfully over the test data, try to interpret the error message in finding a solution and submit the error message (via screen capture or cut and paste) to David Vogel at djvogel2@wisc.edu.

I get a properties dialogue instead of a tool input dialogue when opening the tool.

Refresh the directory in ArcCatalog (hit F5 to refresh).

Who can I contact for help?

David Vogel, State Cartographer's Office, 608-890-3793, djvogel2@wisc.edu

Credits

Supporting libraries for tool development, thanks to: the datamade/parserator project, https://github.com/datamade/parserator

Address Reference Guide

This reference accompanies the Address Parsing Tool. It describes the address components participating in full street addresses, the elements the Address Parsing Tool outputs, how to check the parsing results, and troubleshoot address parsing errors and flags.

Use this guide if you are utilizing the Address Parsing Tool to parse street addresses and want to learn about how to optimize the results of your address parse.

The Address Parsing Tool was developed as a part of the Wisconsin Statewide Parcel Initiative as a pathway for data contributors to process unparsed addresses into address sub-components meeting the statewide layer's attribute schema. The schema's address elements are based on the FGDC-endorsed United States Thoroughfare, Landmark, and Postal Address Data Standard and include common elements with definitions. This tool is designed to interpret local addresses and output address components meeting the FGDC and Parcel Initiative standards to the highest degree possible, although variations in data at the local level may impede accuracy.

Why is This Guide Necessary?

In order to achieve the most optimal parsing results from this tool, it is important for the user to understand the nature of the address elements within the statewide parcel schema as well as the various outputs provided as a result of this tool.

This is a reference guide specifically for those using the Address Parsing Tool, developed to meet the needs of the statewide schema in the Submission Documentation.

Address Element Examples

1.1 Address standards and formatting

The statewide parcel attribute schema can be found in the Submission Documentation. The address elements in the table below are the targeted outputs of the Address Parsing Tool.

Address Element Outputs	
Schema Name (and Alias)	**Corresponding Tool Output**
ADDNUMPREFIX (Address Number Prefix)	NEW_ADDNUMPREFIX
ADDNUM (Address Number)	NEW_ADDNUM
ADDNUMSUFFIX (Address Number Suffix)	NEW_ADDNUMSUFF
PREFIX (Prefix)	NEW_PREFIX
STREETNAME (Street Name)	NEW_STREETNAME
STREETTYPE (Street Type)	NEW_STREETTYPE
SUFFIX (Suffix)	NEW_SUFFIX
UNITTYPE (Unit Type)	NEW_UNITTYPE
UNITID (Unit ID)	NEW_UNITID

Note that this tool aims to parse street addresses only and does NOT parse the following to output fields:
- Landmark Names
- Place Names (City/Town/Village Name)
- Zip Codes

If Landmark Names, Place Names, or Zip-Codes exist within the address, they will be written to the extraneous data field while street address will be parsed to their respective elements.

Some address examples, and their expected Address Parsing Tool outputs are depicted in Figure 6. The address and ordering of sub-components to outputs depicted meet the specifications of the statewide schema.
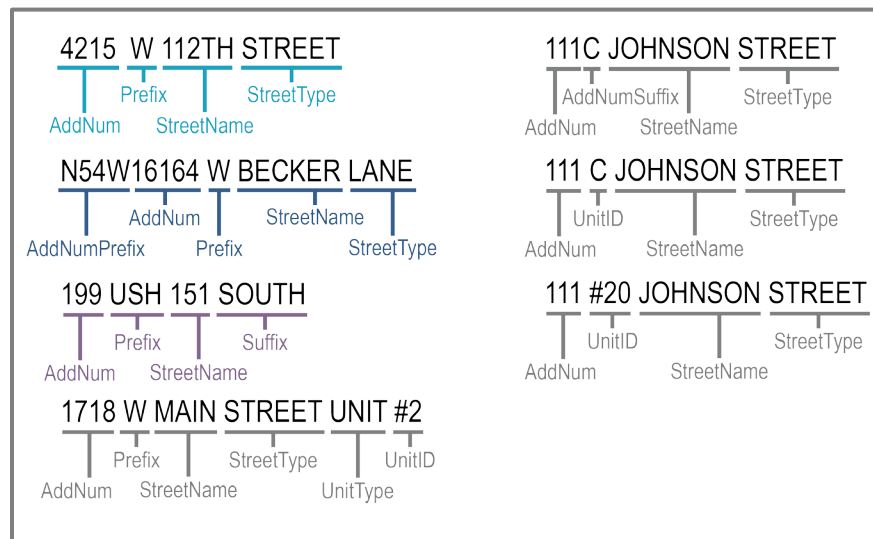

Figure 6. Sub-components of address outputs

Correcting Flags and Errors

2.1 Special characters

The Address Parsing Tool is very sensitive to the inclusion or location of non-alphanumeric characters in the site address field. For this reason, character cleanup is vital to achieving optimal results. In many cases, you will need to manually remove characters flagged by the tool.

Characters flagged by the tool should be removed, with some exceptions where a character is acceptable.

Special Characters		
	Acceptable	**Unacceptable – Requires Change**
Hyphen (-) The hyphen character is most often used in a site address for indicating a range or compound words.	Hyphens are acceptable in compound street names, concurrently running streets, or unit ID ranges. 100 W MAIN ST UNIT A-E 100 TRI-COUNTY RD 100 BUS 18-151	Hyphens are not allowed to indicate an address number range. A range of address numbers indicates multiple site addresses. There should only be one site address per parcel. To solve this problem, either remove the address number range or duplicate the parcel assigning each parcel a different number in the range. 100-104 W MAIN ST should be 100 W MAIN ST
Forward Slash (/) The forward slash is used frequently in a site address to denote different street names.	Forward slashes are acceptable when denoting concurrently running streets. 100 STH 25/58	Forward slashes are unacceptable when indicating a street name alias. To solve this problem, remove one of the street names and the forward slash. 100 STH 26/MAIN ST should be 100 STH 26 or 100 MAIN ST
Pound sign (#) The pound sign is used to indicate a unit. For example, instead of "Unit 1" another way to write it would be "#1."	Pound symbols are acceptable when used to indicate a unit. 100 W MAIN ST #1 100 #1 W MAIN ST	All other situations. To solve this problem, remove the character and other associated text. Then add number if known. - NO# W MAIN ST should be 100 W MAIN ST or W MAIN ST - UNKNOWN # W MAIN ST should be 100 W MAIN ST or W MAIN ST

2.2 Extraneous Data

The **Extraneous_Data_Flag** field is generated when you run the parser over the site address field. It contains various data that the parser was unable to assign to a common address element. **In most cases this extraneous information should be removed from the full address prior to re-running the parse tool.**
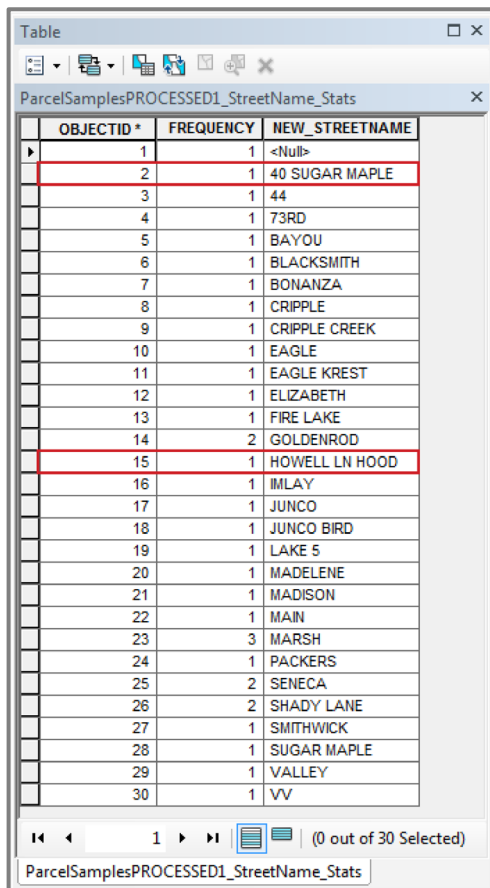
 A few examples of data that could be found in this field include the latter portion of an address range or some type of property descriptor, or an alternate street name attached to the full address.

Please note that while in most cases this information is not an acceptable address element, it is possible that the parser incorrectly placed an acceptable element into this field. It is recommended you take time after your final parse to manually move elements that may have inadvertently ended up in the **Extraneous_Data_Flag** field to their appropriate field.

Leverage the Tool's Summary Tables

3.1 Address Fields Summary output tables

Once you have successfully run through as many iterations as necessary to remove **Parse_Error_Flags**, corrected issues associated with **Incomplete_Data_Flags** and **Character_Flags**, it is recommended that you select the tool check box that will create **Address Fields Summary** for your Prefix, StreetName, StreetType, and Suffix fields. These summary tables will provide an additional method for double checking the parse success and identify any potential errors that may not have been identified by the various flags listed above. Below are some examples of the summary tables from the (SampleData.gdb » ParcelSamples) feature class provided with the tool.

Figure 7. Highlighting of incorrect street names

Figure 7 shows a summary of the **StreetNames** from the **ParcelSamples** feature class. The two highlighted records "40 SUGAR MAPLE" and "HOWELL LN HOOD" are incorrect street names. Once you have read through the summary table and identified incorrect names, you can make corrections in various ways:

Option1
Sort the attributed field alphabetically, open an editing session, navigate to the problematic street name and make the necessary correction. Be sure to make the correction to the full address field that you are running the parse over too, especially if you plan to re-run the parse.
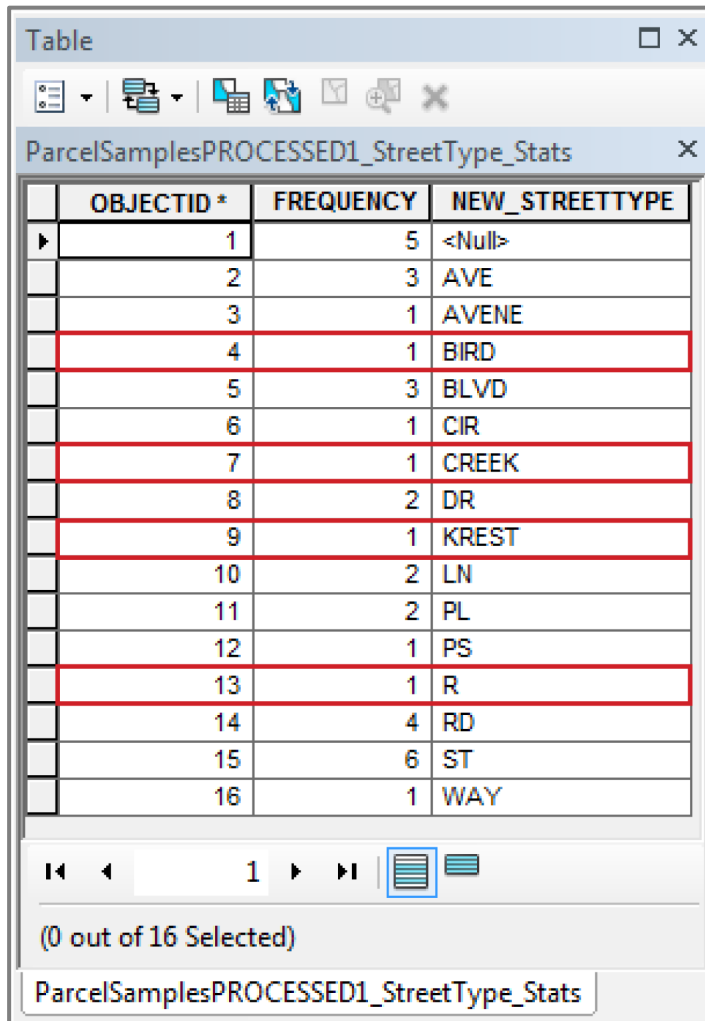
Option 2
Another option is to use the **Select by Attributes** option within the attribute table, open an editing session and construct a definition query that will select the incorrect records and allow you to make the necessary corrections. Be sure to make the correction to the full address field that you are running the parse over too, especially if you plan to re-run the parse.

Figure 8 shows a summary of the **StreetTypes** from the **ParcelSamples** feature class. The highlighted records are uncommon/incorrect street types. Once you have identifier incorrect street types, you can make corrections in various ways.

Note that often when you see uncommon street types this is a result of a multi-word street name that is missing its street type. In many cases, the second word of the street name will be placed in the STREETTYPE attribute field.

Option1

Sort the attributed field alphabetically, open an editing session, navigate to the problematic street name and make the necessary correction. Be sure to make the correction to the full address field that you are running the parse over too, especially if you plan to re-run the parse.

Option 2

Another option is to use the **Select by Attributes** option within the attribute table, open an editing session and construct a definition query that will select the incorrect records and allow you to make the necessary corrections. Be sure to make the correction to the full address field that you are running the parse over too, especially if you plan to re-run the parse.

Figure 8. Highlighting of incorrect street types