

Preview mode. Refresh to see the latest changes.



[FME Support Center](#) / [Data Types and Solutions](#) / [Databases and Data Warehouses](#) / [Esri Geodatabase](#)

Categories



Working with Geodatabase Domains: Writing A Coded Domain



Liz Sanderson

Updated 11 months ago

FME Version

FME 2022.0

Files



WritingACodedDomain.zip

1 MB · [Download](#)



WritingACodedDomain_2022.zip

1 MB · [Download](#)

Preview mode. Refresh to see the latest changes.

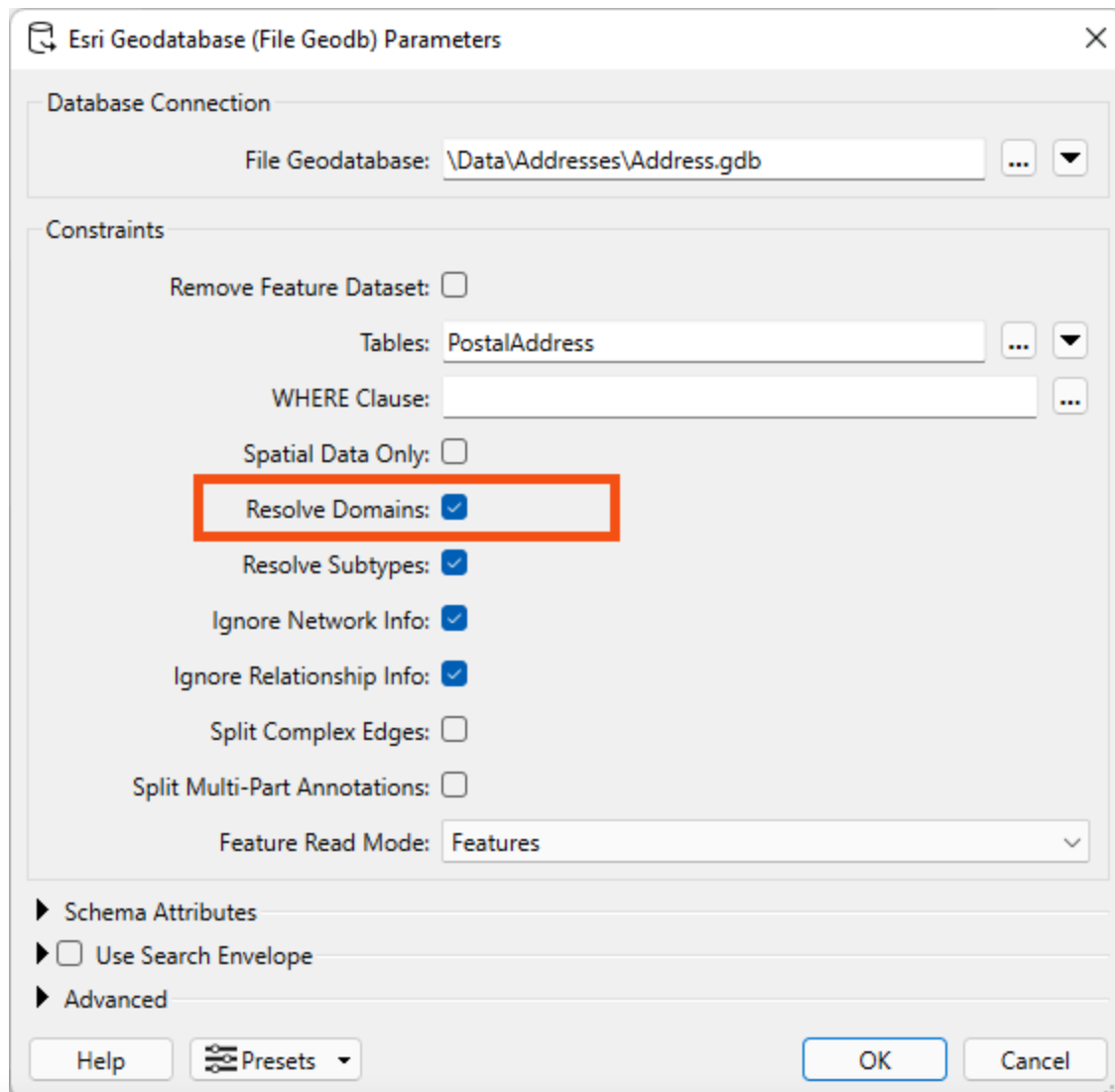
Introduction

A domain is a set of rules that define permitted values for an attribute. They are used to constrain data values in order to ensure data integrity. A domain is defined in a geodatabase as a unique entity and can be applied to any attribute in any feature class within the geodatabase that contains the domain definition.

There are two types of geodatabase domains (coded domains and range domains) and both are supported by FME. A coded domain is essentially a list of multiple valid values while a range domain is a single permitted range of numeric values.

Domain Reading

When reading a geodatabase, FME has an option to resolve domains.



Preview mode. Refresh to see the latest changes.

which contains the textual description of the coded attribute value.

Domain Writing

If you can, we would recommend using a geodatabase template as described in [How to Use an Esri Template](#). This is generally easier and more flexible than defining domains within FME feature types.

FME has the capability to write to existing or create new geodatabase tables. As such, when writing geodatabase domains, the workflow requirements will tend to fall under one of two scenarios:

- Write to an existing table using an existing domain.
- Write to a new table using an existing domain.

These scenarios will be controlled primarily by the following writer parameters:

- Data Type: coded_domain / range_domain
- Validate Features to Write parameter value: Yes / No

Despite the scenarios mentioned below, it should be noted that using an XML Workspace Document (containing the required domain definitions) or a file geodatabase as a template on a geodatabase writer, is the most efficient and highly recommended means of writing geodatabase domains. For more information on importing an XML document or a geodatabase Template with a schema, please see the Geodatabase Writer documentation.

Scenario 1: Writing to an existing table using an existing domain

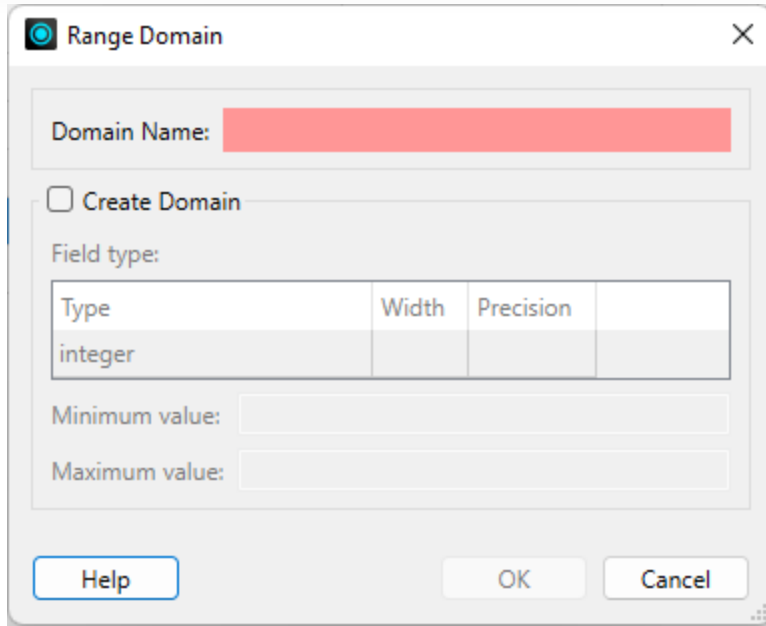
Any data written to an existing domain field is, by default, inserted as normal. As the table already exists, its attribute(s) will already be associated with the required domain and there is no need to set any parameter to define this connection.

However, if you wish to validate incoming data – for instance, compare it to a domain definition to ensure it has valid attribute values – then you must set the writer parameter “Validate Features to Write” to Yes. If the validation parameter is set to No, the data will pass into the Geodatabase without error, despite the fact that some data values may not meet domain rules.

Scenario 2: Writing to a new table using an existing domain

When creating a new table that uses an existing domain, the attribute that needs to be associated with a domain should be given the data type coded_domain or range_domain (depending on its

Preview mode. Refresh to see the latest changes.



The image shows a 'Range Domain' dialog box. It has a title bar with a blue icon and a close button. The main area contains a 'Domain Name' text field with a red background. Below it is a checkbox labeled 'Create Domain'. Under the checkbox is a section titled 'Field type:' containing a table with four columns: 'Type', 'Width', 'Precision', and an empty column. The 'Type' column has the value 'integer'. Below the table are two text fields labeled 'Minimum value:' and 'Maximum value:'. At the bottom are three buttons: 'Help', 'OK', and 'Cancel'.

Type	Width	Precision	
integer			

Limitations

There are a few limitations in regard to domain writing.

Firstly, it is not possible to write to an existing table and to either create an association with an existing domain or create an entirely new domain. That is because this association is wrapped up in the table definition, and an existing table definition cannot be changed by FME. You would need to drop the existing table and re-create it entirely in order to be able to do this.

Secondly, for the same reason stated above, creating a domain is a one-off translation. You would set the data type to coded_domain/range_domain for the initial process, but subsequent loads of the data should be done with the data type changed back the actual type of data (char, integer, etc.)

Finally, it is not possible to create a domain dynamically (i.e., the domain definition should be manually defined prior to execution and not defined during the workspace process).

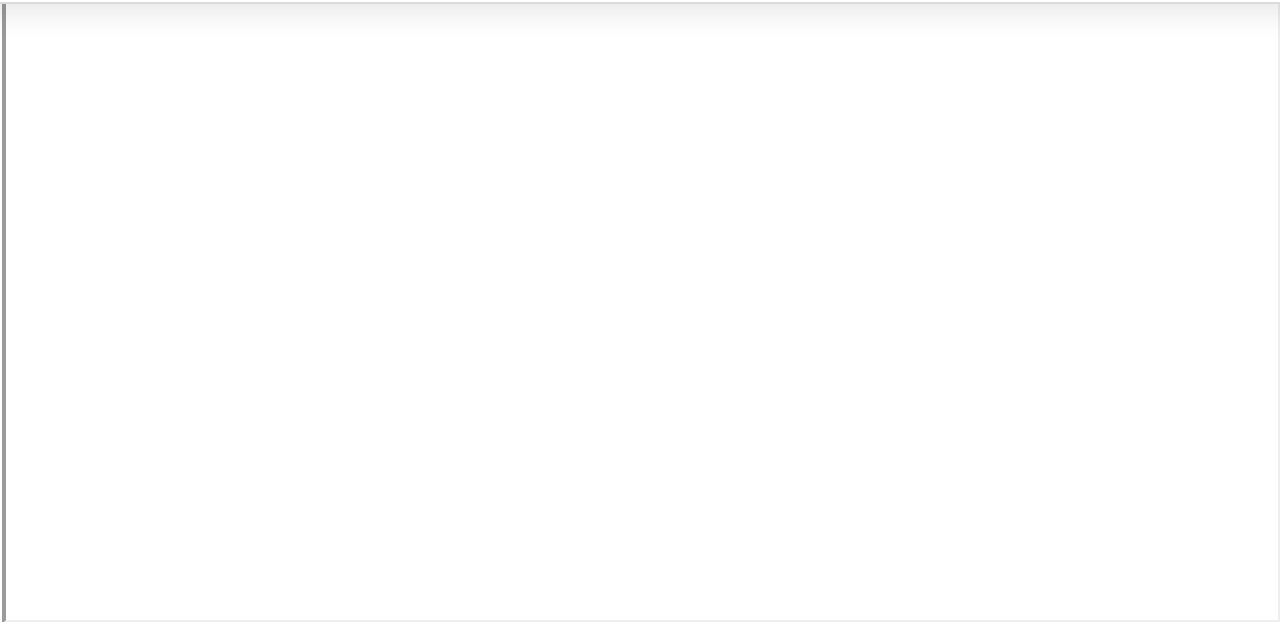
The following example demonstrates Scenario 2: Writing to a new table and using an existing domain through the use of a template.

Requirements

The Esri Geodatabase (File Geodb) reader/writer used in the following example requires that a licensed version of ArcGIS be available to the user. For more information on required ArcGIS license levels, please see [Comparison of Esri-Based Readers and Writers in FME](#).

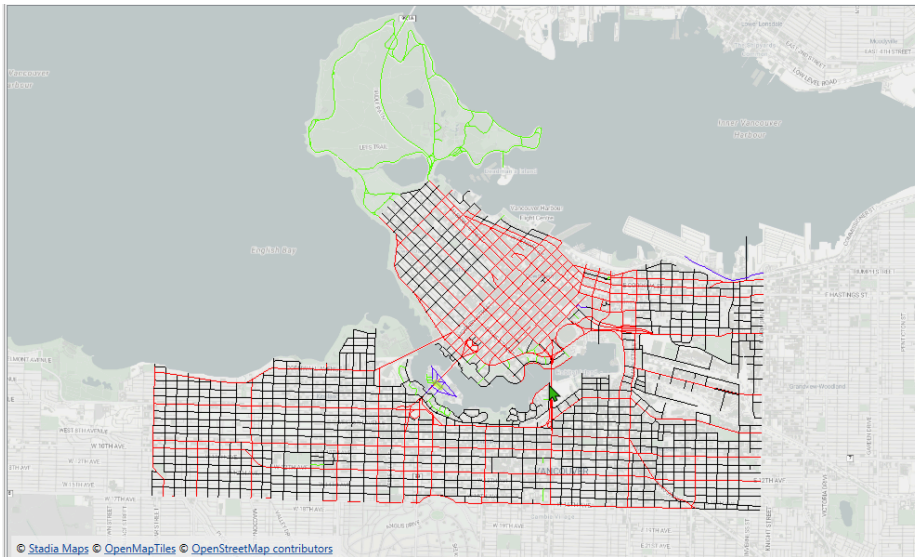
Video

Preview mode. Refresh to see the latest changes.



Source Data

Roads (Autodesk AutoCAD DWG/DXF)



© Stadia Maps © OpenMapTiles © OpenStreetMap contributors

Features Selected: 2 of 2

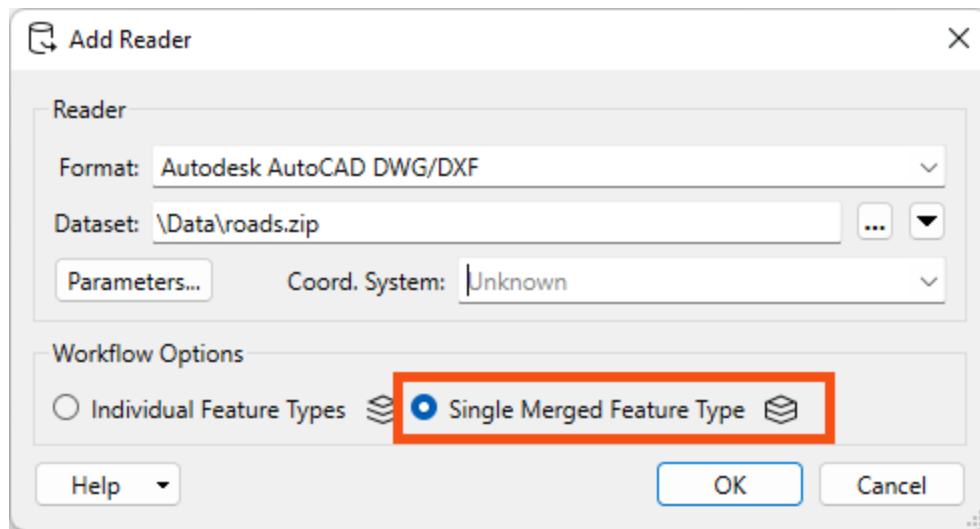
Property	Data Type	Value
Exposed Attributes (6)		
From_HBlock	varchar(10)	
HBlock	varchar(27)	
StreetId	number(6,0)	10002
StreetName	varchar(19)	Railspur Alley
fme_feature_type	varchar(50)	NonCity
autocad_layer	varchar(50)	NonCity
Unexposed Attributes (45)		
autocad_color		180
autocad_entity		autocad_line
autocad_entity_handle		68
autocad_entity_visibility		visible
autocad_extended_data_list[] (5)		
autocad_layer_desc		
autocad_layer_frozen		no
autocad_layer_hidden		no
autocad_layer_locked		no
autocad_layer_on		yes
autocad_layer_plottable		yes
autocad_layer_type		not_frozen
autocad_linetype		ByLayer
autocad_linetype_scale		1
autocad_lineweight		-3
autocad_original_color		180
autocad_original_entity_type		autocad_line
autocad_resolved_linetype		Continuous
autocad_resolved_transparency		-1
autocad_source_filename		C:\...
autocad_space		model_space
autocad_thickness		0

In the above image, we see a visualization of the source Roads AutoCAD DWG. The values that will be subject to the destination geodatabase domain definition can be seen in the autocad_layer format attribute.

Step-by-step Instructions

Preview mode. Refresh to see the latest changes.

Road.zip dataset provided in the Files section of this tutorial. You can leave the file zipped. Change the Workflow Options to Single Merged Feature Type. Then, open the parameters.



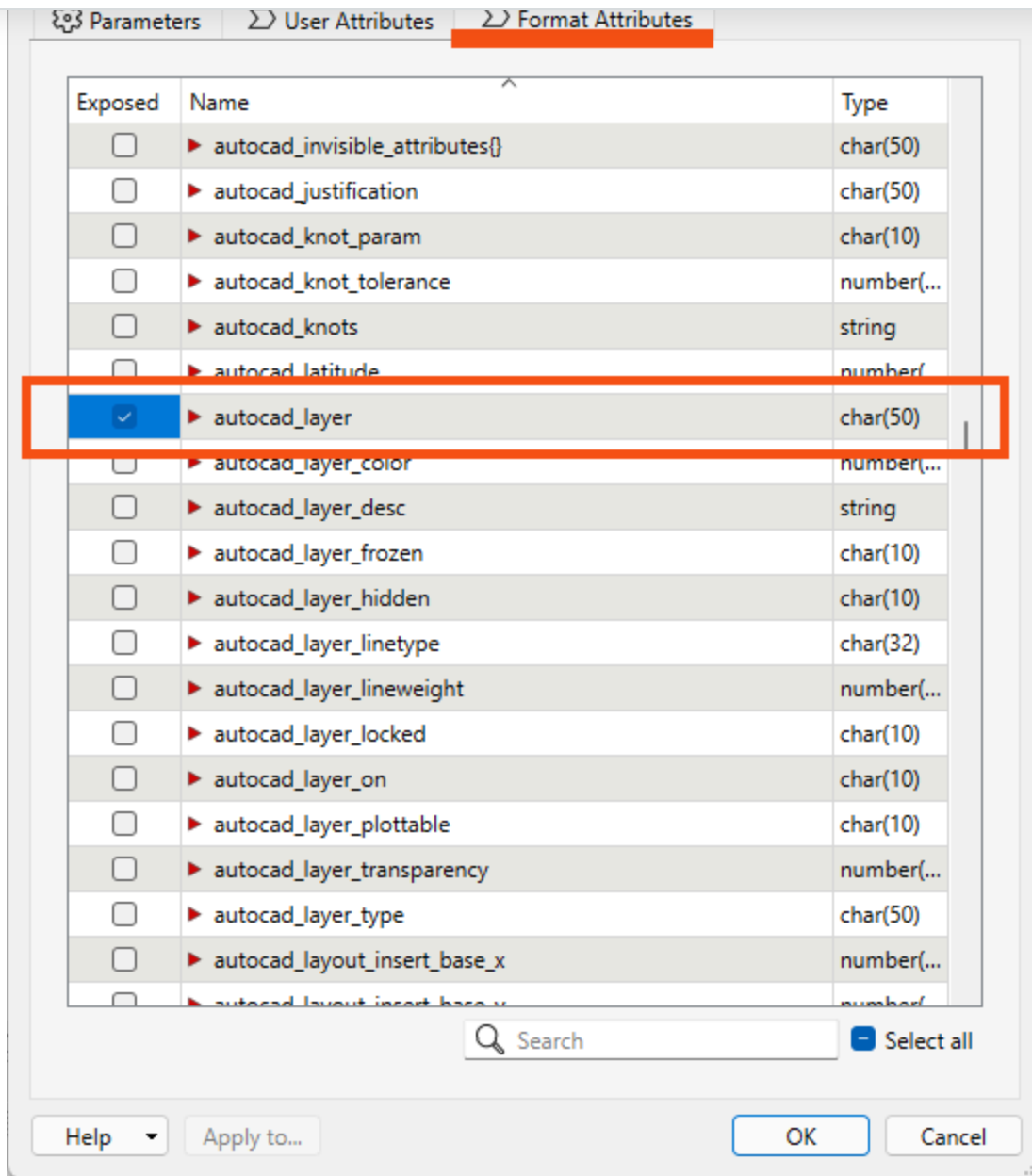
In the parameters, set the Group Entities By to Attribute Schema, then click OK twice to add the reader.

Preview mode. Refresh to see the latest changes.

The screenshot shows the 'Format Attributes' tab of the FME Reader Parameters dialog. The 'Group Entities by' section has three radio buttons: 'Layer Name', 'Geometry', and 'Attribute Schema'. The 'Attribute Schema' option is selected and highlighted with a red rectangle. Below this are three expandable sections: 'Entity Options', 'Model and Paper Space Options', and 'Attribute Options'. The 'Entity Options' section contains four checkboxes: 'Explode Blocks into Entities' (checked), 'Read Visible Attributes as Text Entities' (checked), 'Explode MText Entities' (checked), and 'Read Polylines as 2.5D' (unchecked). The 'Model and Paper Space Options' section contains two checkboxes: 'Read Model Space' (checked) and 'Read Paper Space' (unchecked). The 'Attribute Options' section contains six checkboxes: 'Read Interpreted Extended Entity Data' (checked), 'Read Extended Entity Data As List' (checked), 'Read Attribute Entity Data As List' (checked), 'Entity XRecord Data Reading' (unchecked), 'Read Dynamic Block Attributes' (unchecked), and 'Read Dynamic Block Attributes As List' (checked). At the bottom, there are three expandable sections: 'Schema Attributes', 'Use Search Envelope' (unchecked), and 'Advanced'. The bottom of the dialog features a 'Help' button, a 'Presets' dropdown menu, and 'OK' and 'Cancel' buttons.

Once the reader has been added to the workspace canvas, double-click on the reader feature type to open the parameters. Switch to the Format Attributes tab, enable the `autocad_layer`, and click OK.

Preview mode. Refresh to see the latest changes.



2. Map Attributes

There are two different ways to map attributes to domains, either by domain code or by resolved domain values. If following along with the demo, only use one method.

Option 1: Map Attributes to Domain Codes

Next, we will use an AttributeValueMapper to map the incoming values from the source DWG to the domain codes, which we will apply to the domain definition.

Add an AttributeValueMapper to the canvas and connect it to the <ALL> reader feature type. In the parameters, set the Source Attribute to autocad_layer and enter StreetCategory as the Destination Attribute. Enter 99 as the Default Value as, this will be the domain code we will use for Unknown

Preview mode. Refresh to see the latest changes.

Source Value	Destination Value
Arterial	1
Secondary	2
Residential	3
Private	4
NonCity	5
Other	6
Collector	7
Unknown	99

Preview mode. Refresh to see the latest changes.

The screenshot shows the 'Attribute Value Mapper' transformer dialog. The 'Transformer Name' is 'AttributeValueMapper'. Under 'Attribute Selection', 'Input Attribute' is 'autocad_layer' and 'Output Attribute' is 'StreetCategory'. Under 'Value Map', the 'Default Output Value' is '99' and 'Mapping Direction' is 'Forward (Input To Output)'. A table maps input values to output values:

Input Value	Output Value
<input type="checkbox"/> Arterial	<input type="checkbox"/> 1
<input type="checkbox"/> Secondary	<input type="checkbox"/> 2
<input type="checkbox"/> Residential	<input type="checkbox"/> 3
<input type="checkbox"/> Private	<input type="checkbox"/> 4
<input type="checkbox"/> NonCity	<input type="checkbox"/> 5
<input type="checkbox"/> Other	<input type="checkbox"/> 6
<input type="checkbox"/> Collector	<input type="checkbox"/> 7
<input type="checkbox"/> Unknown	<input type="checkbox"/> 99

At the bottom, there are buttons for '+', '-', '^', 'v', '▲', '▼', '✂', '📄', '📁', 'Import...', 'Help', 'Presets', 'OK', and 'Cancel'.

Option 2: Map Attributes to Resolved Domain Values

As an alternative to the step described above, one could instead use an AttributeManager, to take the value from the source DWG file and map it to the domain definition based on the domain definitions resolved value. To do so, add an AttributeManager to the workspace and open the parameters dialog. Rename autocad_layer to StreetCategory_resolved - the resolved version of the domain.

Preview mode. Refresh to see the latest changes.

Transformer Name: AttributeManager

► Advanced - Attribute Value Handling

Attribute Actions

Input Attribute	Output Attribute	Value	Type	Action
From_HBlock	From_HBlock	<Enter value (optiona...	varchar(10)	Do Nothing
HBlock	HBlock	<Enter value (optiona...	varchar(27)	Do Nothing
StreetId	StreetId	<Enter value (optiona...	number(6,0)	Do Nothing
StreetName	StreetName	<Enter value (optiona...	varchar(19)	Do Nothing
fme_feature_type	fme_feature_type	<Enter value (optiona...	varchar(50)	Do Nothing
autocad_layer	StreetCategory_resolved	Enter new value (op...	varchar(50)	Rename
<Expose existing attribute>	<Add new attribute>			

+ − ^ v ▴ ▾ ✂ 📄 📁

Search Import

Help Presets

The above tends to be most applicable when your workflow falls under Scenario #1 (writing to an existing table using an existing domain definition) or Scenario #2 (writing to a new table in the geodatabase using an existing domain). In both scenarios, it would be assumed that one does not know the actual domain codes contained in the domain definition and instead knows the resolved value.

3. Write Features and Create Domain

Add an Esri Geodatabase (File Geoddb) writer to the canvas, browse to a location to save the geodatabase. Leave the Feature Class or Table Definition set to Copy from Reader then open the parameters.

In the parameters, enable Overwrite Existing Geodatabase, then browse to the RoadsTemplate.gdb for the Template File Geodatabase. Next, expand the Advanced section and set Validate Features to Write to Yes. This setting will ensure that the incoming source data adheres to our domain definition. Then click OK twice to add the writer.

Preview mode. Refresh to see the latest changes.

Database Connection

File Geodatabase: ... ▼

☐ Fanout Dataset

☒ Overwrite Existing Geodatabase

Template File Geodatabase: ... ▼

☐ Import XML Workspace Document

Transaction Type: ▼

Coordinate System

Geometry Settings

Advanced

Geodatabase Version: ▼

Default Z Value: ▼

Transaction Number: ▼

Features Per Transaction: ▼

Ignore Failed Features: ▼

Max number of features to ignore: ▼

Dump Failed Features to File: ▼

Failed Feature Dump filename: ... ▼

Annotation Units: ▼

Contains Measures: ▼

Compact Database When Done: ▼

Enable Fast Deletes: ▼

Enable Load Only Mode: ▼

▼

Simplify Network Features: ▼

SQL To Run Before Write: ... ▼

SQL To Run After Write: ... ▼

Help ▼

OK Cancel

Connect the writer to either the AttributeValueMapper or the AttributeManager, then open the writer feature type parameters.

In the feature type parameters, change the Feature Class or Table Name to something more

Preview mode. Refresh to see the latest changes.

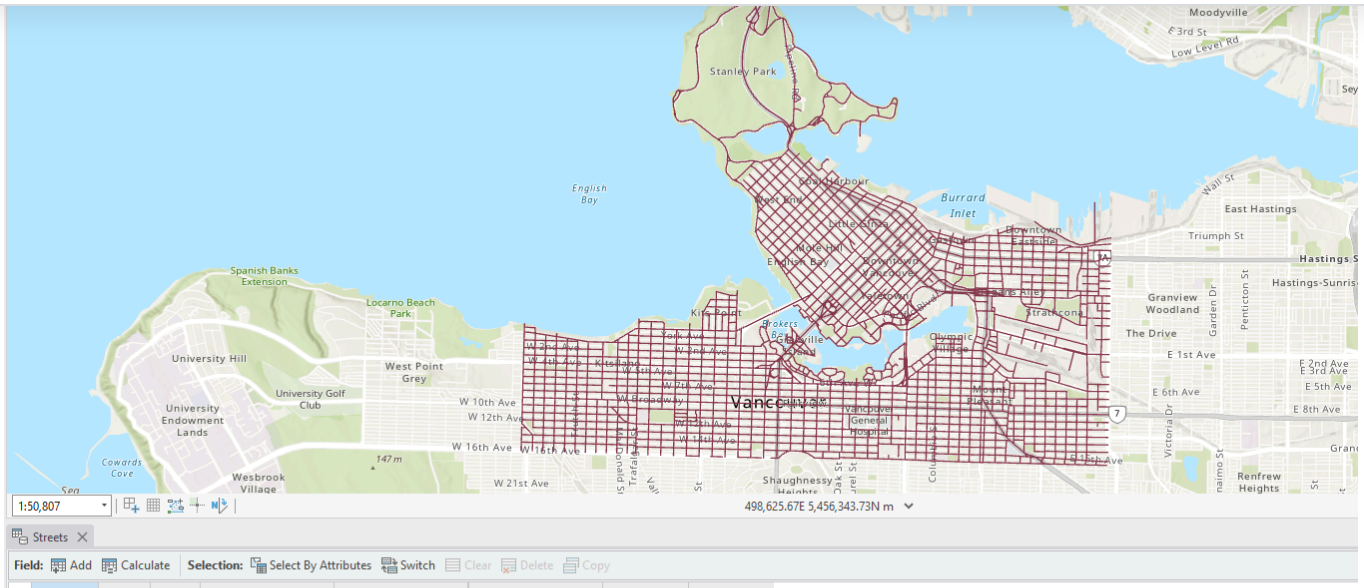
The screenshot shows the 'Feature Type' dialog box with the 'User Attributes' tab selected. The 'Attribute Definition' section has 'Manual' selected. A table lists attributes: 'From_HBlock' (char, width 10), 'HBlock' (char, width 27), 'StreetCategory' (char, width 50, highlighted with a red box), 'StreetId' (integer), and 'StreetName' (char, width 19). The bottom of the dialog includes a toolbar with icons for adding, removing, and sorting attributes, a search bar, and 'Help', 'Apply to...', 'OK', and 'Cancel' buttons.

Name	Type	Width	Precision	Value
▶ From_HBlock	char	10		
▶ HBlock	char	27		
▶ StreetCategory	char	50		
▶ StreetId	integer			
▶ StreetName	char	19		

4. Save and Run the Workspace

Save the workspace and then run it. View your output in ArcGIS Pro.

Preview mode. Refresh to see the latest changes.



A product of

Safe Software

[Company](#) [Products](#) [Giving Back](#) [Careers](#) [Contact](#)

Safe Software respectfully acknowledges that we live, learn and work on the traditional and unceded territories of the Kwantlen, Katzie, and Semiahmoo First Nations.

The data used here originates from data made available by the [City of Vancouver, British Columbia](#). It contains information licensed under the Open Government License - Vancouver.



Preview mode. Refresh to see the latest changes.

Was this article helpful?

☒ Yes


☐ No

Related articles

- How to use an Esri Template Geodatabase
- Working with Geodatabase Subtypes: Writing A Subtype
- Notes on FME and Esri Versions and Compatibility
- Introduction to Working with Geodatabase Relationship Classes
- Python Scripted Parameters in FME

Comments

0 comments



Submit

Preview mode. Refresh to see the latest changes.
