

Attachment Migration for Attendee Mapping

Introduction

In this exercise, we will map the distribution of course attendees actively taking the FME & ArcGIS Integration course. Use the Attendee Submission App to submit your data.

Data & Privacy

Reminder: No personal information is required for app submission.

When submitting your app, you are not expected to use any real personal information, such as your own first and last name, personal photos, or any other identifying details.

To keep things simple and privacy-friendly, you can use staged responses instead. For example:

- ✓ Use a fictional first and last name (e.g., "Alex Doo").
- ✓ Upload a placeholder image or an avatar instead of a real photo.
- ✓ Provide sample data instead of actual personal details.

This ensures a safe and consistent experience for everyone. If you have any questions, feel free to ask!

All data will be removed from the PostGRES database after the course concludes.

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 [Required ArcGIS License Types for FME Geodatabase Formats](#).

Step-by-step Instructions

Create Feature Class

1. Open FME Form 2025.0

Create a new workspace in FME Form 2025.0

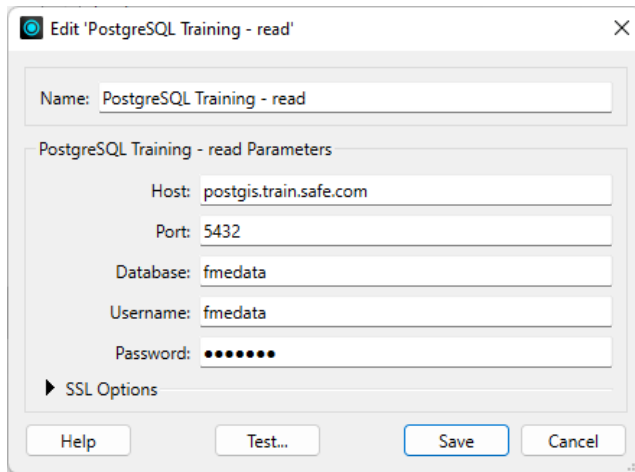
2. Connect to Training Database

Navigate to Tools > FME Options > Database Connections > + > Add Connection. Use this window to select and connect to the Postgres Training Database.

Host: postgis.train.safe.com

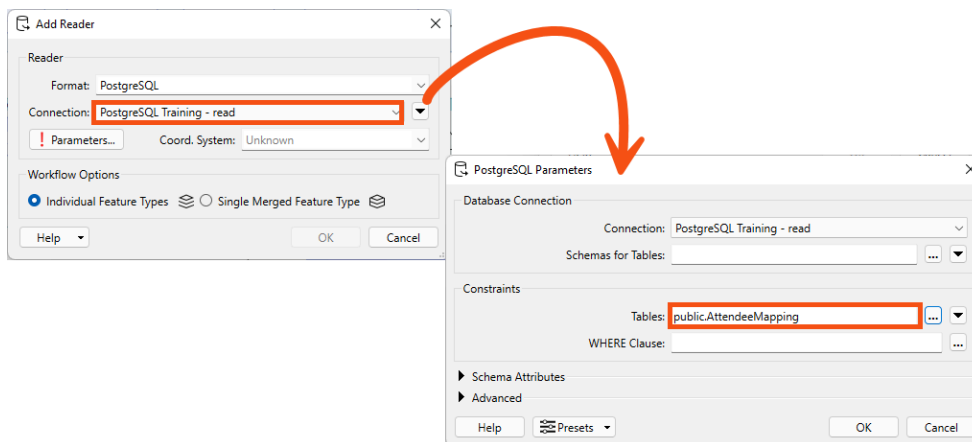
Port: 5432

Database: fmedata
Username: fmedata
Password: fmedata



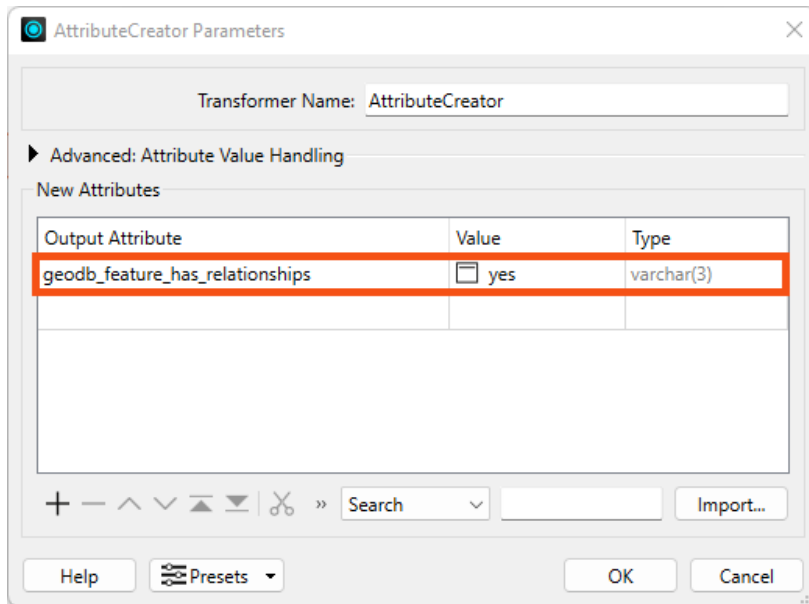
3. Add PostgreSQL Reader

Now that a connection is created, add a PostgreSQL Reader to the FME canvas. Use the 'Table' parameter within the PostgreSQL Reader parameters window to select the 'public.AttendeeMapping' table.



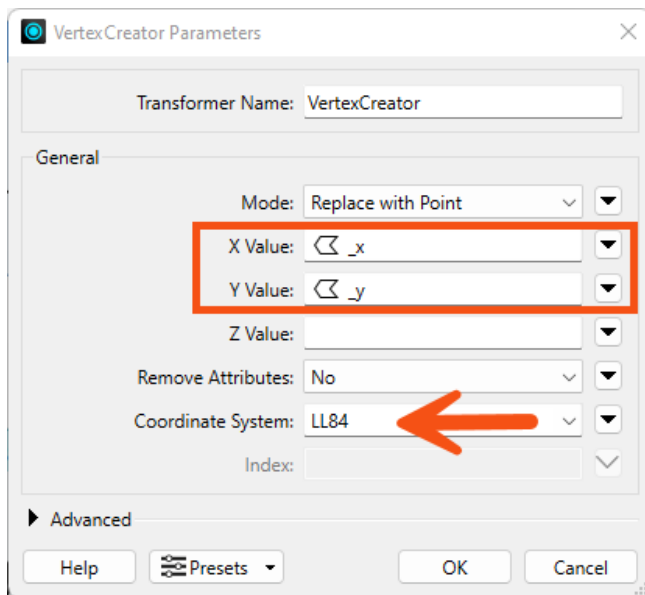
4. Add Relationship-Based Attributes for Feature Classes

FME needs to be told that the features participate in a relationship class. Add an AttributeCreator transformer. Connect the AttributeCreator to the output port of the postgresQL reader feature type. In its parameters, under the New Attributes parameter table, create a new attribute called 'geodb_feature_has_relationship' under the Output Attribute column; then, set the Value column to 'yes'.



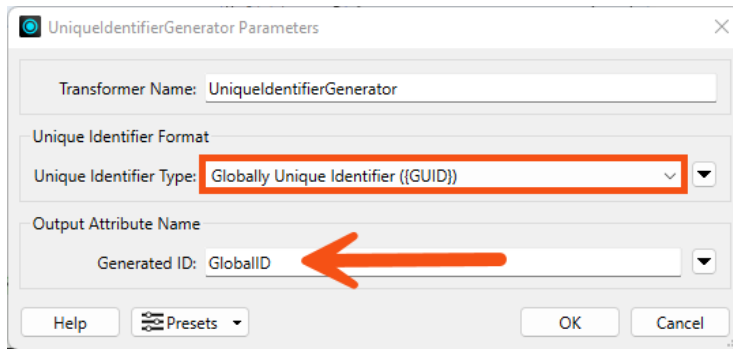
5. Create Point Geometry

Add a VertexCreator. Open its parameters. Assign '_x' to X Value and '_y' to Y Value, and 'LL84' as the Coordinate System. This will create a geometry point for each attendee record.



6. Create GUID

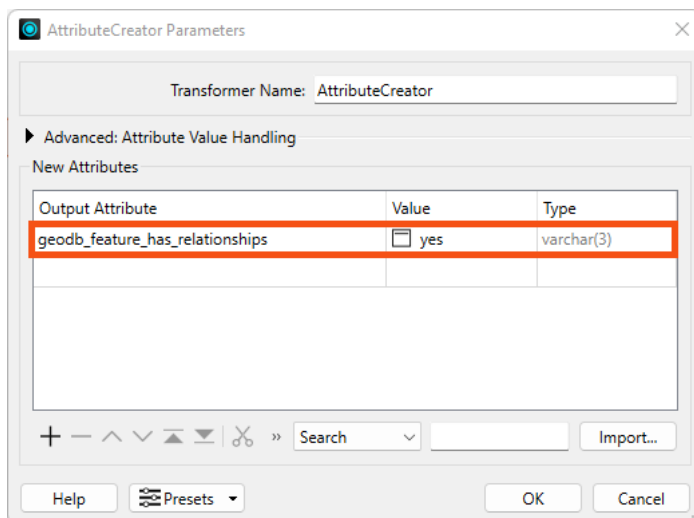
Create a GlobalID using a UniqueIdentifierGenerator transformer.



Create Attachment Table

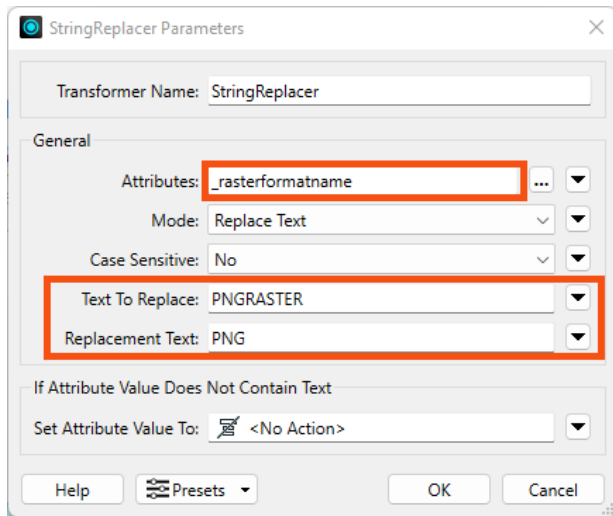
7. Add Relationship-Based Attributes for Attachment Table

As we previously mentioned, FME needs to be told that the features participate in a relationship class. Add an AttributeCreator transformer. Connect the AttributeCreator to the output port of the UniqueIdentifierGenerator transformer. In its parameters, under the New Attributes parameter table, create a new attribute called 'geodb_feature_has_relationship' under the Output Attribute column; then, set the Value column to 'yes'.



8. Add a StringReplacer

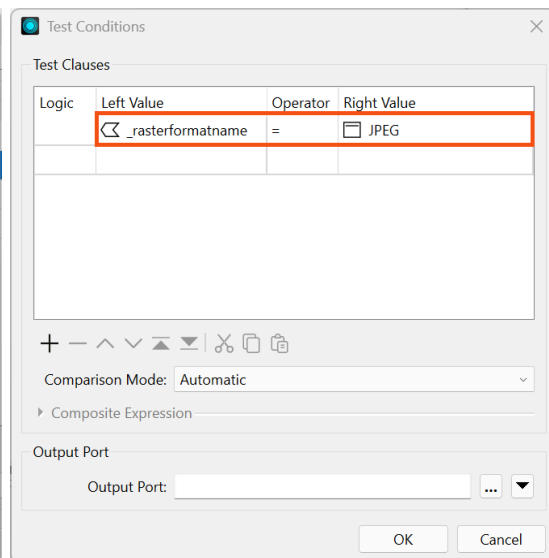
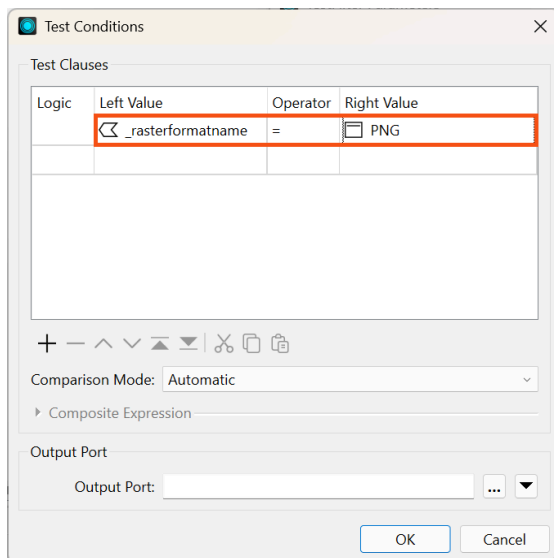
Add a StringReplacer and connect it to the output of the AttributeCreator_2. We will use the '_rasterformatname' attribute when we create the CONTENT_TYPE field downstream. The StringReplacer needs to shorten any occurrence of 'PNGRASTER' to 'PNG', so it becomes an acceptable value based on the Esri specification.

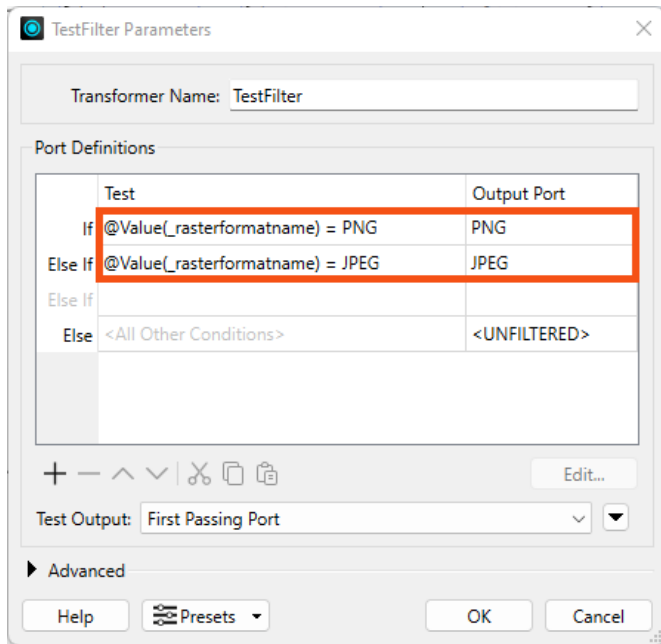


9. Separate Images Based on Extension & Extract BLOBs

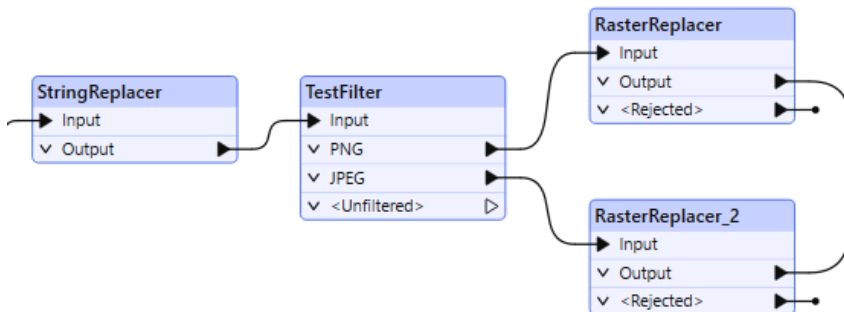
Add a TestFilter to appropriately route features based on the image extension. Since RasterReplacers are isolated to a single format, we need to separate PNG and JPEGs into separate data flows.

Connect the output of the StringReplacer to the TestFilter transformer. Use the TestFilter to conditionally route features based on the `_rasterformatname`.

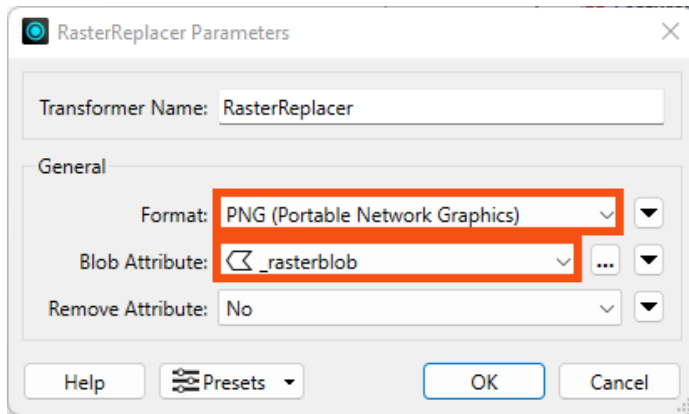




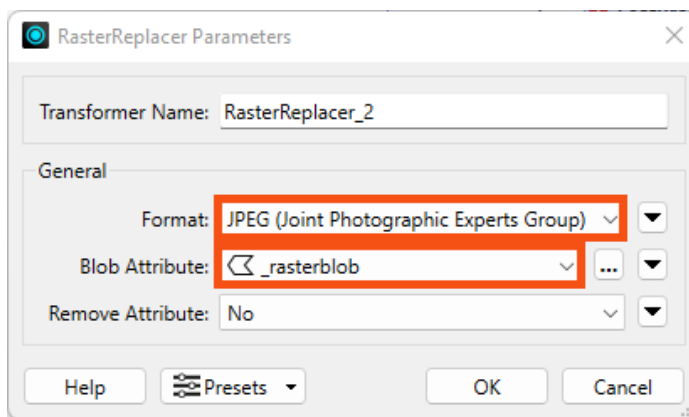
Add a separate RasterReplacer onto each output port of the TestFilter transformer.



Configure the RasterReplacer connected to the PNG output port of the TestFilter. In the Parameters, set the Format as 'PNG (Portable Network Graphics)' and Blob Attribute to '_rasterblob'. Leave all other parameters as default.



Configure the additional RasterReplacer to extract JPEG images. Connect the RasterReplacer to the PNG output port of the TestFilter. In the Parameters, set the Format as 'JPEG (Joint Photographic Experts Group)' and Blob Attribute to '_rasterblob'. Leave all other parameters as default.



10. Set Up the Fields in the Attachment Table

At the moment, we're writing a relationship between features, but the destination features (attachments) are currently a reference to a file, not the file itself. We need to use these references to read the attachment file contents.

First, we need to do some basic schema mapping. Add an AttributeManager. Connect both RasterReplacer output ports to the input port of the AttributeManager. Configure the attributes as follows:

INPUT ATTRIBUTE	OUTPUT ATTRIBUTE	Action
geodb_rel_destination_oid	REL_GLOBALID	RENAME
_fullname	ATT_NAME	RENAME

INPUT ATTRIBUTE	OUTPUT ATTRIBUTE	Action
_rasterblob	DATA	RENAME
_rasterformatname	CONTENT_TYPE	RENAME
_image		REMOVE
_session		REMOVE
_location		REMOVE
_x		REMOVE
_y		REMOVE
geodb_feature_has_relationships		REMOVE
	DATA_SIZE	SET VALUE
	GLOBALID	SET VALUE

Now that these attributes have been mapped or created, we can set the required values. Note that the DATA_SIZE attribute will be left blank once created.

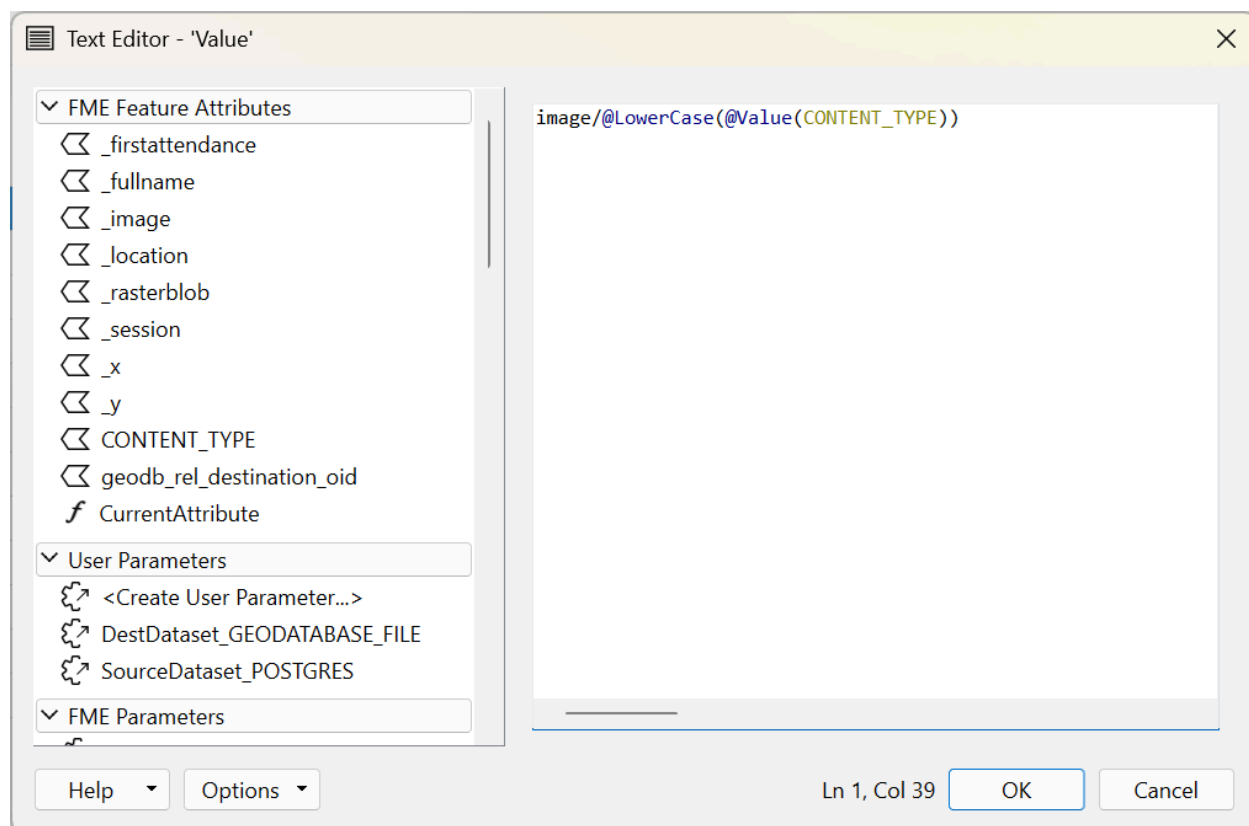
Create a new Output Attribute called GLOBALID. We will use FME's @UUID() function to create a new unique identifier or GlobalID for each attachment. Set the value as follows:

{@UUID()}

Ensure curly braces are added around the function to meet ArcGIS Pro standards.

Next, open the CONTENT_TYPE's Text Editor by clicking on the dropdown arrow in its Value field, and enter the following expression:

image/@LowerCase(@Value(CONTENT_TYPE))

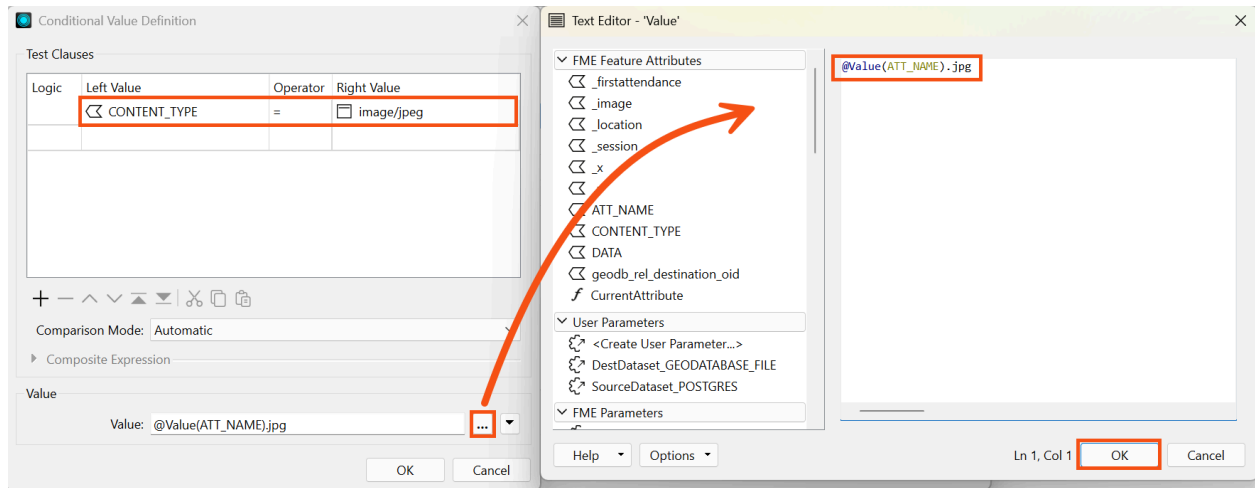


Lastly, we need to conditionally set the ATT_NAME, based on the format extension of the original image. This will ensure each image receives the correct format extension and can be successfully displayed in ArcGIS Pro.

Access Conditional Values by clicking the dropdown arrow in the Value field and navigating to 'Conditional Value...'.

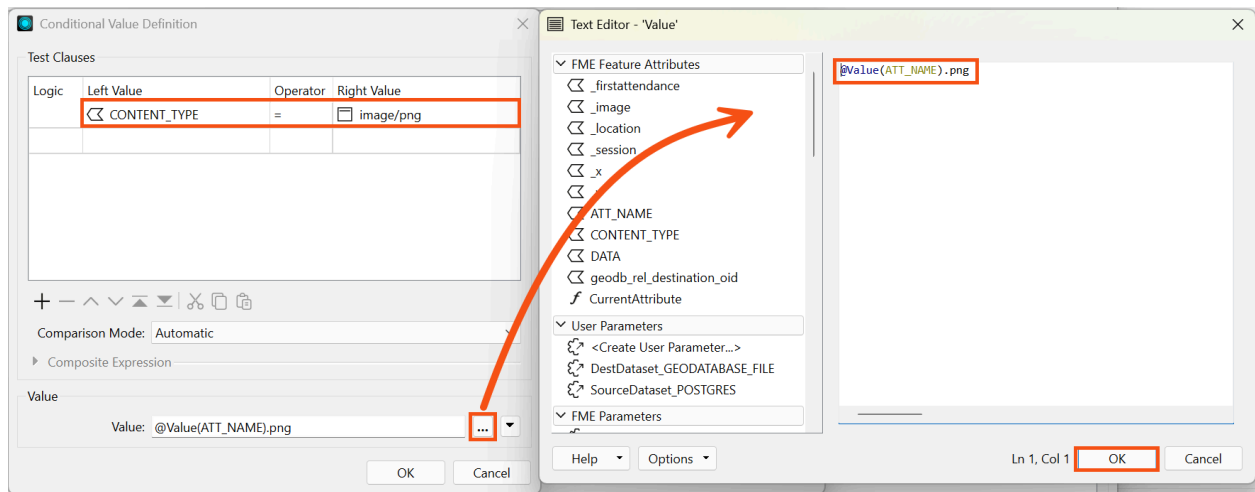
For the if statement set 'CONTENT_TYPE' to equal image/jpeg and for value click on the ellipsis (...), and enter the following expression:

@Value(ATT_NAME).jpg






For the else if statement set 'CONTENT_TYPE' to = image/png and for value click on the ellipsis (...), and enter the following expression:

@Value(ATT_NAME).png



Conditional Value Definition

Condition Statement

	Test	Value
If	@Value(CONTENT_TYPE) = image/jpeg	 @Value(ATT_NAME).jpg
Else If	@Value(CONTENT_TYPE) = image/png	 @Value(ATT_NAME).png
Else If		
Else	<All Other Conditions>	 <No Action>

+

−

^

∨

✂

📄

📋

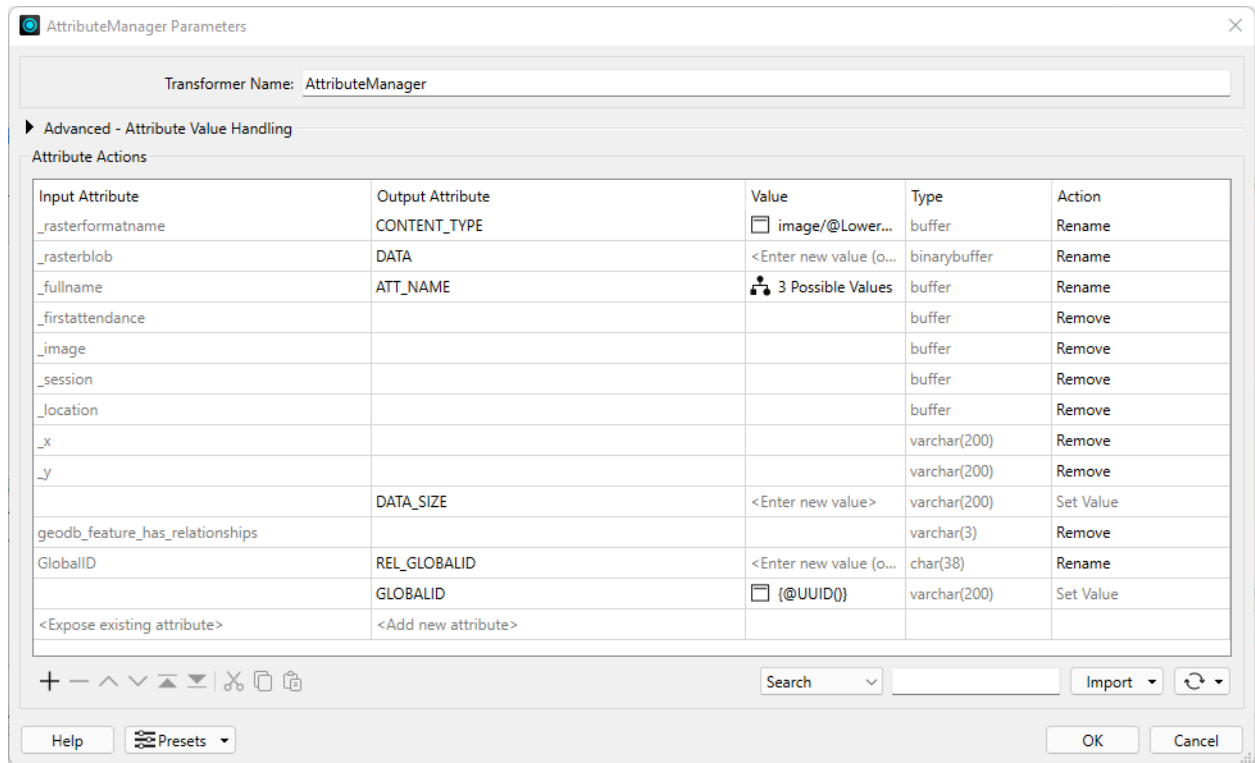
Edit...

Help

OK

Cancel

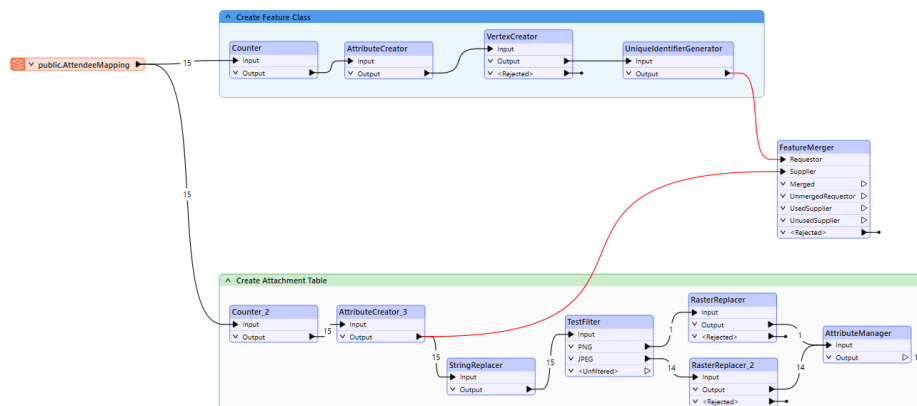
Overall, the AttributeManager should reflect the image below:



Create Relationship Class

1. Build Relationship Information

Add a FeatureMerger transformer. This is how the relationship will be built. Connect the UniqueIdentifierGenerator to the Requestor port and the AttributeCreator_2 to the Supplier.



In the Parameters, set the Requestor to '_fullname' and the Supplier to '_fullname'.

The other outstanding issue to take care of is the case where there are multiple attachments for a particular attendee (in the event someone submitted to the app twice).

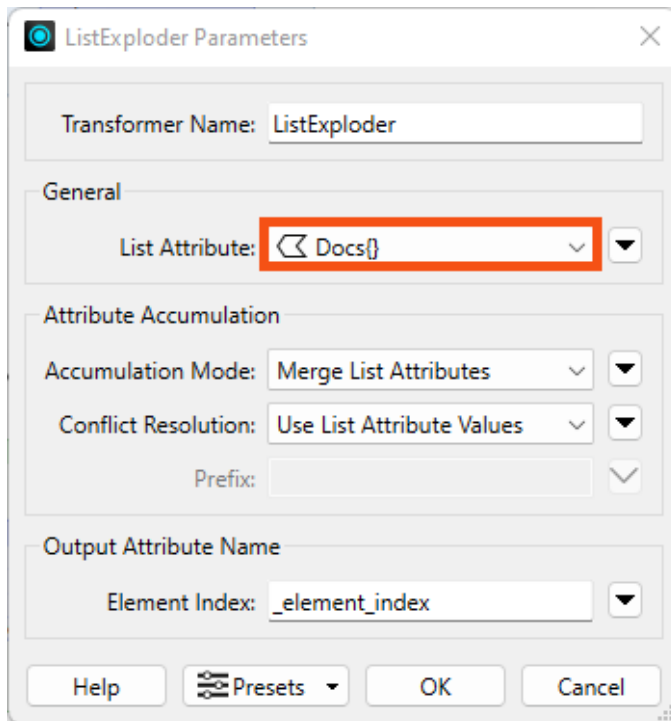
So, while still in the FeatureMerger parameters dialog, enable 'Process Duplicate Suppliers', then enable 'Generate List'. Set the List Name to 'Docs' and select 'geodb_rel_destination_oid' and '_rasterblob' as the Selected Attributes. Click OK.

2. Explode Docs List

Now when the workspace is run, for features with multiple attachments, multiple files are stored in a list, like so:

- Docs{0}.path_filename
- Docs{0}.geodb_rel_destination_oid
- Docs{1}.path_filename
- Docs{1}.geodb_rel_destination_oid

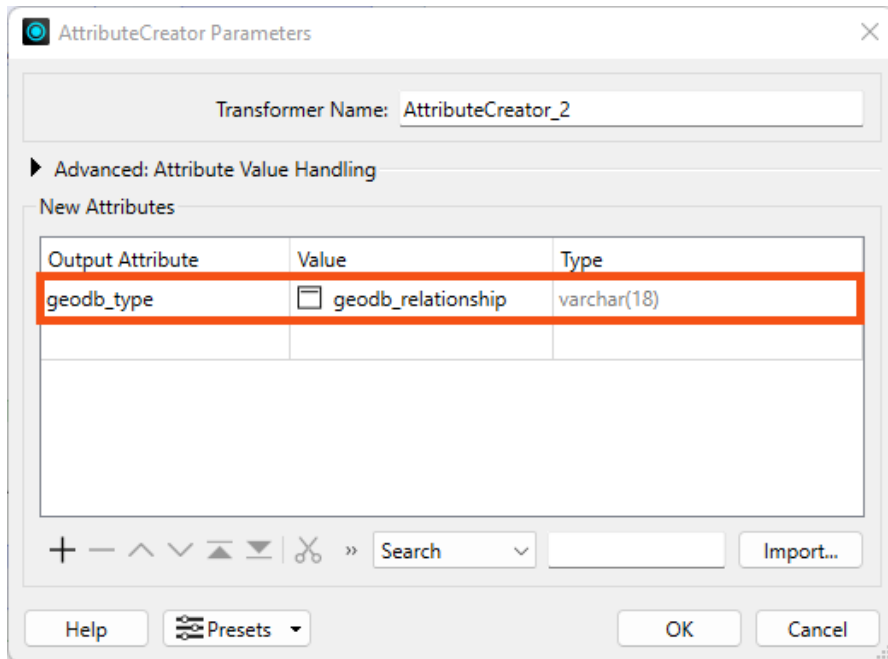
What we now need to do is explode the Docs list into individual features so that there is a relationship record for each attachment. Add a ListExploder transformer to the canvas and connect it to the Merged output port on the FeatureMerger. In its Parameters, set the List Attribute to 'Docs{}', then click OK.



3. Create Relationship Type

Each feature that gets sent to an FME writer should have an attribute indicating the geometry type. We need to tell FME there are actually non-geometry, relationship features.

Add an AttributeCreator and connect it to the ListExploder. In the Parameters, create a New Attribute called 'geodb_type' and set its Value to 'geodb_relationship'.



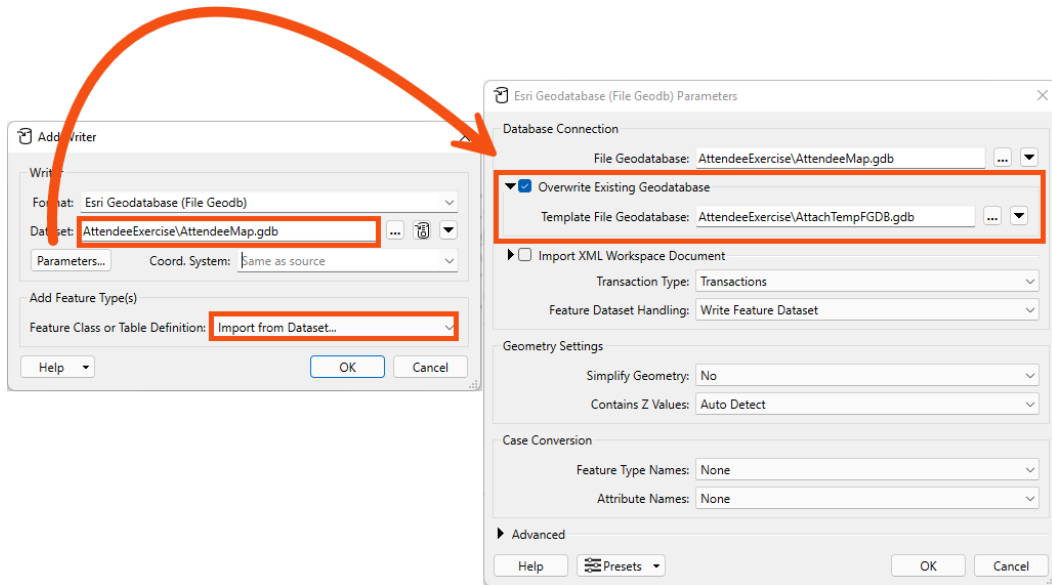
Write the Geodatabase

1. Import the Dataset on Writer

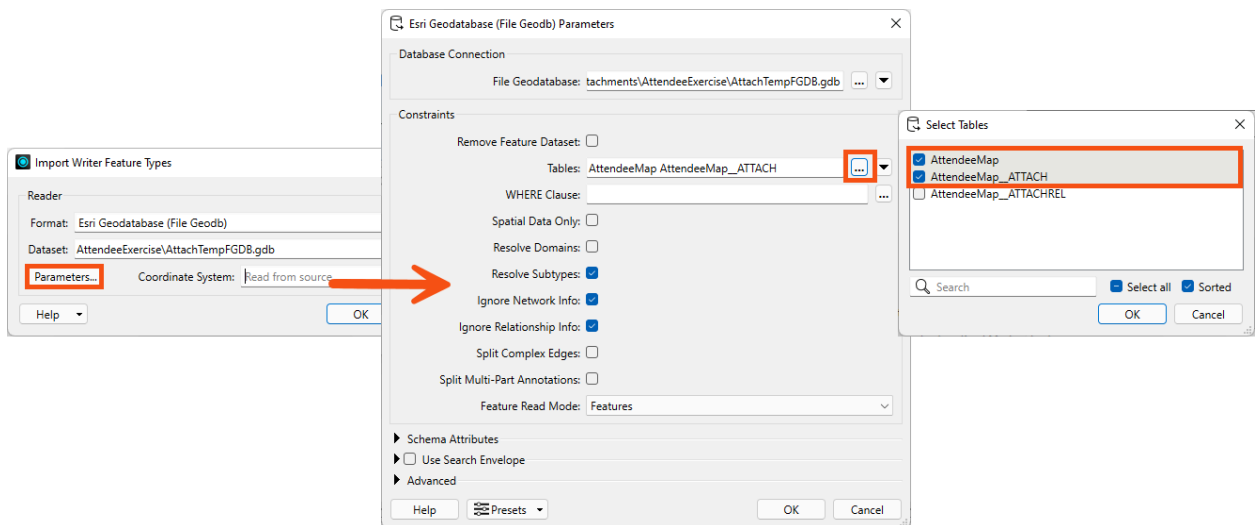
Add an Esri Geodatabase (File Geodb) Writer. The schema for this geodatabase will come from the AttachTempFGDB.gdb file. Add a dataset destination.

Open the Parameters. Enable 'Overwrite Existing Geodatabase' and assign the Template File Geodatabase to = 'AttachTempFGDB.gdb' template file. Click OK.

Next, change the Feature Class or Table Definition to 'Import from Dataset...'. Click OK.



In the 'Import Writer Feature Types' window, leave the format as Esri Geodatabase (File Geodb) and select the 'AttachTempFGDB.gdb' file. Within the parameters, use the Tables parameter to import: 'AttendeeMap' and 'AttendeeMap__ATTACH'. Leave all other parameters as default.



Click OK to add the writer feature types. Connect each writer feature type to its appropriate data stream.

Lastly, edit the writers and ensure the translation type is set to 'Edit Session'.

