Fluree Sense Ingest

End User Guide

Version 2.0

May 2024

Contents

1. Introduction 3
1.1 Overview 3
1.2 Key Terms and Concepts 3
2. Key User Activities 5
2.1 Getting Started 5
2.2 Creating a New Pipeline 6
2.3 Designing Your Pipeline 12
2.3.1 Origins 12
2.3.1.1 ADLS Gen 2 12
2.3.1.2 Delta Lake 16
2.3.2 Processors 18
2.3.2.1 Configuring an Aggregate Processor 19
2.3.2.2 Configuring a Deduplicate Processor 20
2.3.2.3 Configuring a Field Remover Processor 21
2.3.2.4 Configuring a Filter Processor 21
2.3.2.5 Configuring a Join Processor 21
2.3.2.6 Configuring a Type Converter Processor 22
2.3.2.7 Configuring a Field Renamer Processor 22
2.3.2.8 Configuring a Field Order Processor 23
2.3.2.9 Configuring a Delta Lake Lookup Processor 23
2.3.2.10 Configuring a Stream Selector Processor 25
2.3.2.11 Configuring a Spark SQL Expression Processor 25
2.3.3 Destinations 25
2.3.3.1 Configuring an ADLS Gen2 Destination 26
2.3.3.2 Configuring a Delta Lake Destination 28
3. Running and Executing Pipelines 31

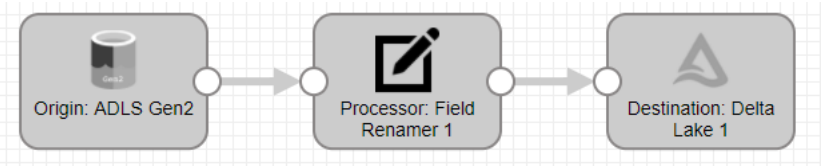
1. Introduction

1.1 Overview

- Fluree Sense is a full end to end platform designed to Ingest, Classify, Resolve, and Consume Big Data.
- The Fluree Sense Ingest component is focused on processing, curating, and transforming data using pipelines from various data sources to create dataset to be cleansed, and mastered.
- Ingest works by creating pipelines that load data from an origin (Can be a file, a delta lake). Then the data can be transformed using many different Ingest processors. Then the data will land in a specified destination.
- Ingest runs data processing pipelines on Apache Spark, an open-source cluster-computing framework. Because Ingest pipelines run on Spark deployed on a cluster, the pipelines can perform transformations that require heavy processing on the entire data set
- Once data fully curated it can be ready to be used in the master data management process.

1.2 Key Terms and Concepts

- Pipeline: A pipeline describes the flow of data from origin systems (or source) to destination systems (or sink) and defines how to transform the data along the way, through processors. Each pipeline must have at least one origin and destination. Pipelines can run in either Batch or Streaming mode, where the pipelines can be turned on and run as data changes.
- Stage: A node in a pipeline graph. A typical pipeline is comprised of three stages, an Origin stage, a Processor stage, and a Destination stage – as illustrated below:



- Origin: An origin stage represents the source for the pipeline. You must use at least one origin stage in a pipeline, although a pipeline may have many origins. The most commonly used origins will be (1) ADLS Gen2, (2) Delta Lake, and (3) Files.
- Processor: A processor stage represents a type of data processing that you want to perform on data between its origin and destination. You can use as many processors in a pipeline as you need. The list of the most commonly used processors are shown below:

Processors	Description
Aggregate	Performs aggregate or grouping calculations. This is the equivalent of doing GROUPSUM or GROUPMAX in SQL.
Deduplicate	Removes duplicate records.

Processors	Description
Field Remover	Removes fields or columns from a record. This is useful for if you have a record with 100 columns and you want to remove 1, or keep 10.
Filter	Passes only the records that match a filter condition.
Join	Joins data from two input streams.
Type Converter	Converts the data types of specified fields to compatible types. This is the equivalent of a CAST command in SQL.
Field Renamer	Renames a field from one name to another specified name. This is useful for doing a column mapping from a source model to a target model
Field Order	Reorders the fields in final output of data. This is useful if you want to sequence the columns in a specific order before writing to file or Delta Table.
Delta Lake Lookup	Performs a lookup on a Delta Lake table. This is like a join, but for a single record value. The Lookup can return one or more result records.
Stream Selector	Routes data to output streams based on conditions. This is useful for if you want to separate a flow based on a filter condition, e.g., if a condition is met follow Stream 1, otherwise route to Stream 2.
Spark SQL Expression	Performs record-level calculations using Spark SQL expressions. This allows you to enter direct SELECT SQL commands for specific transformations (e.g., of you want to add the values of two variables, set a constant value, call a timestamp function or other UDF, etc.)

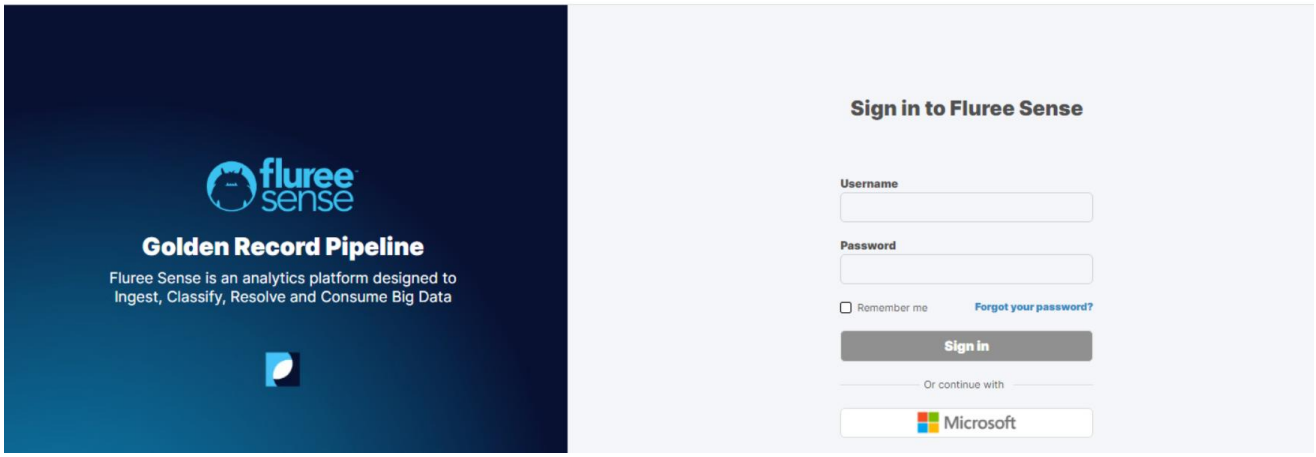
- Destinations: A destination is the path where the pipeline ends and where the data will land. You can use one or more destinations in a single pipeline. The most commonly used destinations are: (1) ADLS Gen2, (2) Delta Lake, (3) Files and (4) JDBC, for relational databases.
- Preview: This is a mode in Ingest which allows you to preview a limited number of records throughout the stages of your pipeline without actually running the full pipeline. This tool is used to tracks the records through the various stages as well as debugging any errors which might have occurred.
- Parameters: This allows you to set environment or configuration variables for the pipeline, such as the name of the container where Data Lake data is being stored. Global parameters can be set if they will be used by various stages in a pipeline. Or, specific parameters can be created from within each stage if it only affects each stage individually.

- Cluster: The cluster is the processer used to run the pipeline. They can be a Databricks cluster, Azure HDInsight, AWS EMR, or Apache Hadoop (for systems such as Cloudera or MapR).
- Batch Mode: Ingest can run pipelines in batch mode. A batch pipeline processes all available data in a single batch, and then stops. A batch pipeline is typically used to process data that has already been stored over a period of time, often in a relational database or in a raw or staging area in a Hadoop Distributed File System (HDFS).
- Streaming: Ingest can run pipelines in streaming mode. A streaming pipeline maintains connections to origin systems and processes data at user-defined intervals. The pipeline runs continuously until you manually stop it. A streaming pipeline is typically used to process data in stream processing platforms such as Apache Kafka.
- Databricks: You can run Ingest pipelines using Spark deployed on a Databricks cluster. Ingest supports Databricks Runtime versions 5.x or 6.x. To run a pipeline on a Databricks cluster, configure the pipeline to use Databricks as the cluster manager type on the Cluster tab of pipeline properties.

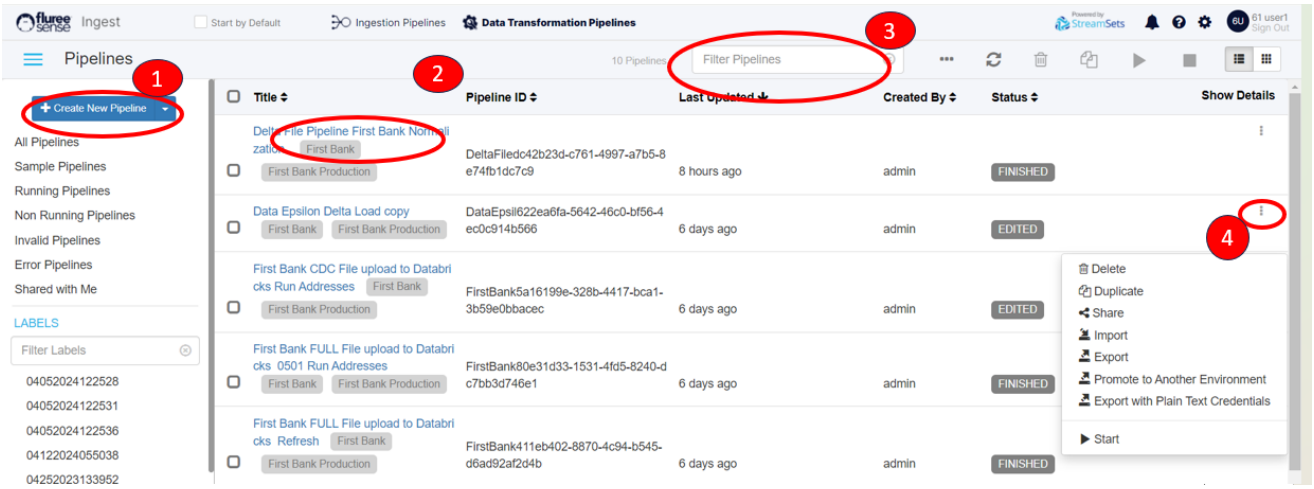
2. Key User Activities

2.1 Getting Started

Login to your account by accessing the URL provided to you, and entering the provisioned User ID and password as shown below.



From here, you are taken to the home screen. Here you will see all your pipelines that this user has access to. There are four main areas of focus on the Home Page:



1. Creating a New Pipeline: This panel allows you to create a new pipeline from scratch or import one from your computer. This will be the first step if the user has not previously created any pipelines before.

2. Core Pipeline Area: Here you will be able to see any pipelines the user has created as well as being able to click on the pipelines to go into each one to edit it and then eventually run the pipelines. In addition, you can see:
    a. The pipeline ID
    b. How long ago was the pipeline last edited
    c. Which user created the pipeline
    d. The status of the pipeline. This can be Running, Finished, Edited, Stopped, Cancelled, Error.

3. Filter Pipelines: Here you will be able to filter pipelines based on keywords in their name and go into a specific one you need at any given time.

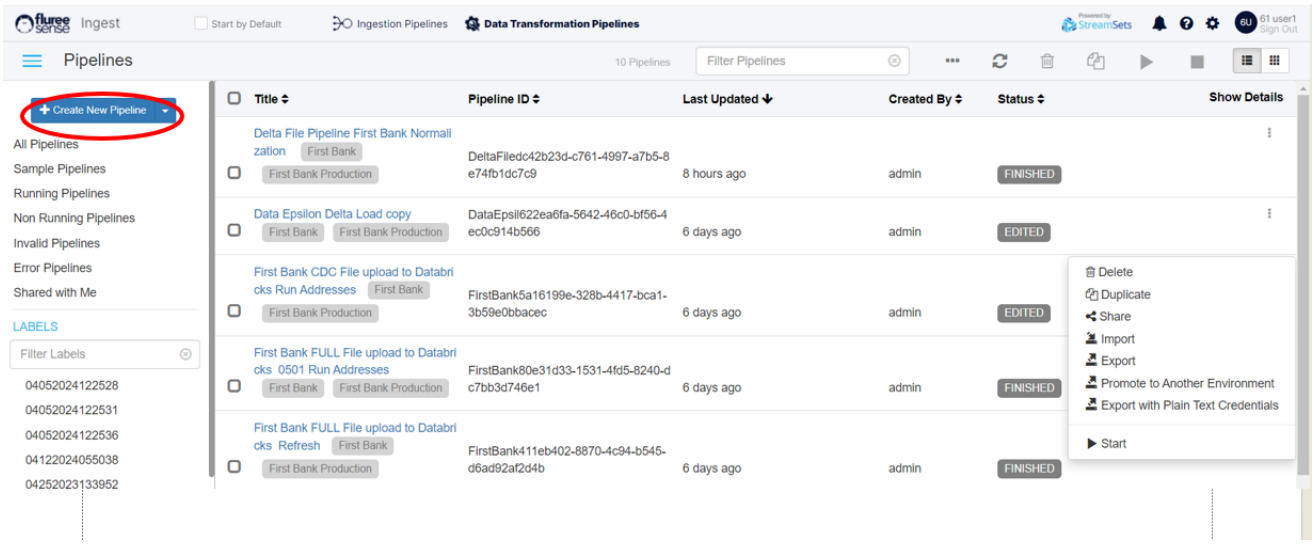
4. Action Menu: Displays additional actions that can be executed for a pipeline, such as starting a pipeline, importing or exporting pipelines, deleting a pipeline, or cloning a new one using an existing one as a template.

Now, let's get started by creating a Ingest Pipeline.

2.2 Creating a New Pipeline

In most cases, the recommended approach is to clone an existing pipeline that matches the conditions for the new pipeline that you are looking to create. For example, if you are creating a pipeline to move from data from a raw staging zone to a semantic zone for a specific table in a source such as SAP, then you should start by cloning a pipeline created to move an existing SAP table from Raw to Semantic.

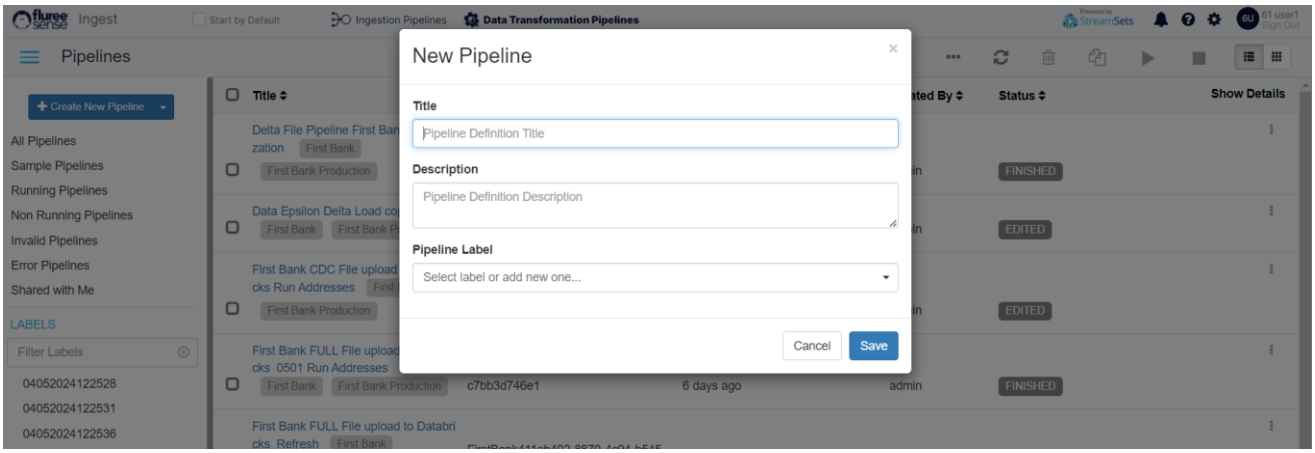
However, in the beginning when no pipeline templates exist from where to draw down from, you can create a new pipeline by clicking on the Create New Pipeline Button on the top left of the screen as shown below:



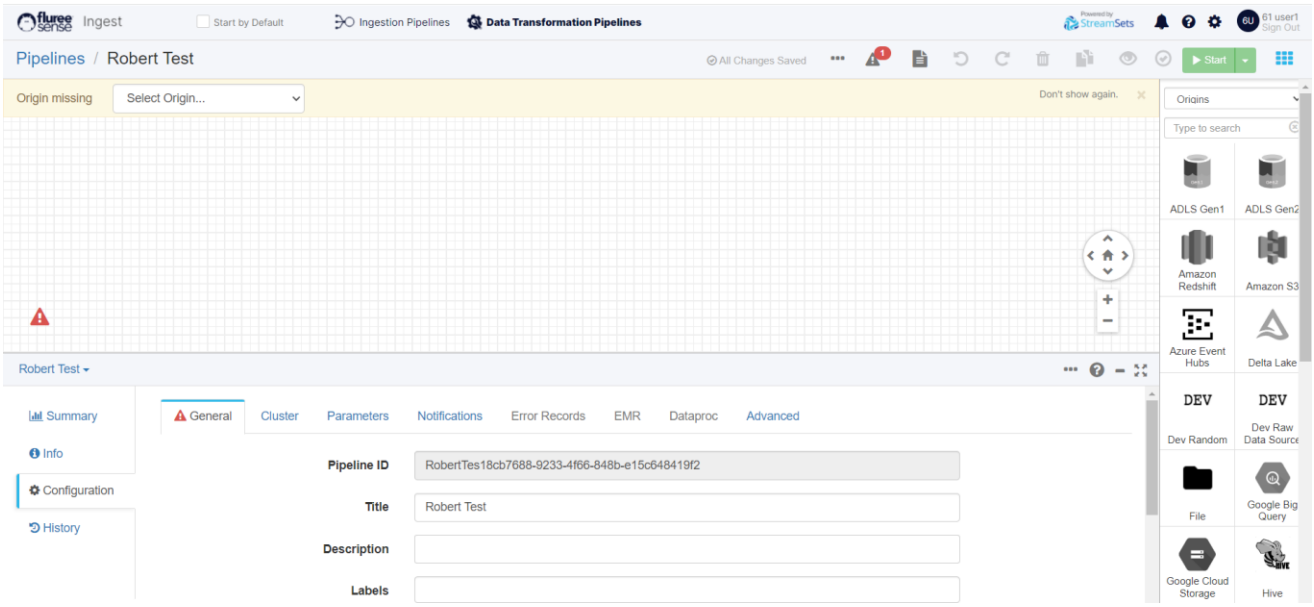
After that you are given the option of inserting the pipeline title, description, and label. The pipeline title will be the display name of the pipeline, and should follow the Naming Convention as described in Section 2.2.

The description is an optional text description that you can provide to better describe the pipeline.

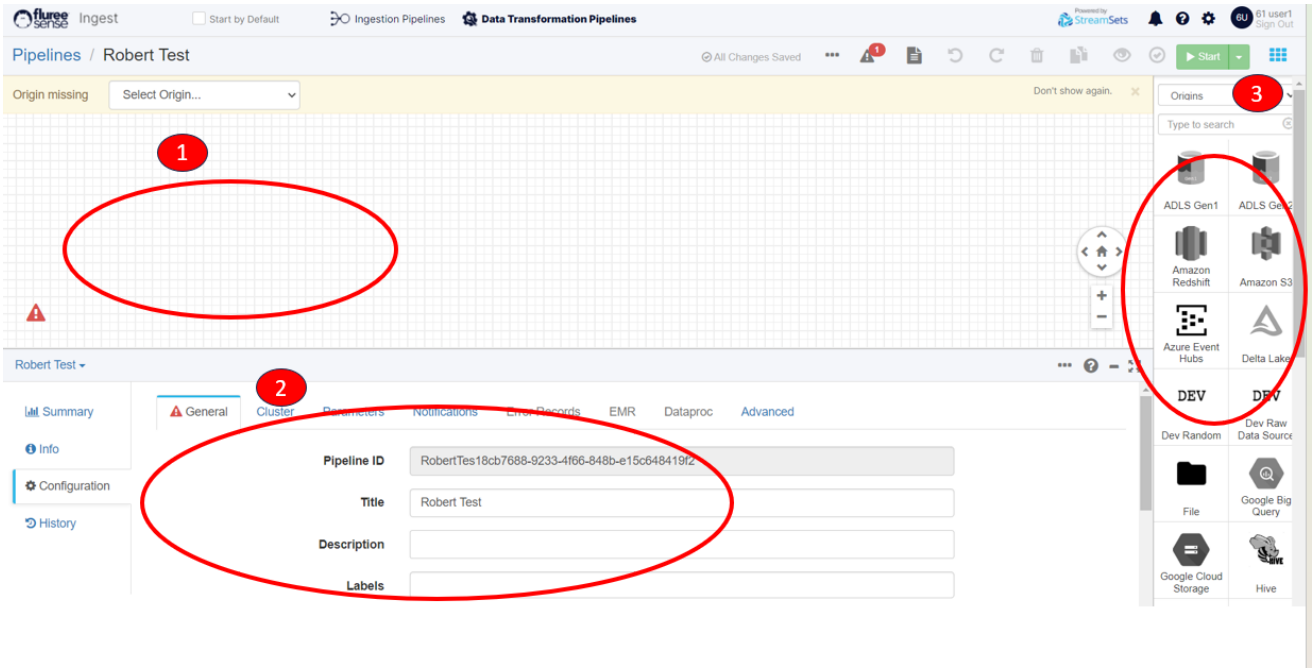
A label is a simple tag that you can add to the pipeline. Adding tags can help you also locate specific pipelines faster by filtering by label, versus a keyword filter on the name. In this case, the labels will appear in the left column below the Create New Pipeline button.



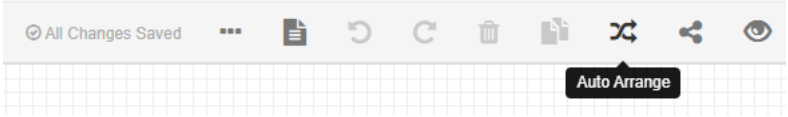
After selecting a title and clicking save the pipeline will be created with that name and the user will be taken to the default blank pipeline design screen:



The pipeline design screen is made up of three major components:



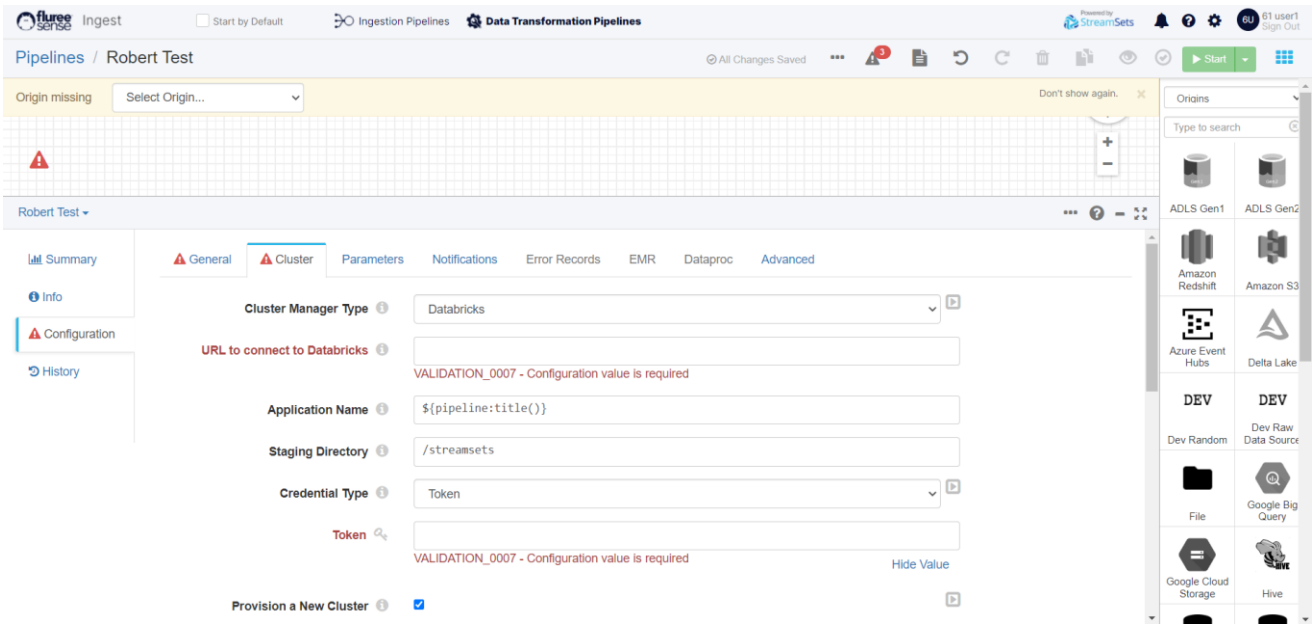
1. The Main Canvas where the user will add and connect stages to build the pipeline. You can drag, drop and connect stages from here, and then select the Auto-Arrange button to organize and align the stages nicely:



2. The Configuration Panel where the user will define the environmental variables, configuration parameters and the required form field elements needed to define how the pipeline or pipeline stage will function. The configuration panel is unique to each stage and describes what needs to be defined in order for the stage to operate correctly. There is also a default configuration panel for the pipeline as a whole, where global configuration settings can be entered. There are multiple tabs that must be filled in. In order for the pipeline to run, certain mandatory fields must be entered. If there is any information missing in any tab, this will be visually indicated through a warning icon

3. The Stage Selector on the right panel. This is the gallery that displays all of the possible stage options, where the user can drag and drop to the Main Canvas. The Stage Selector can be hidden by clicking on the icon of the 9-box grid to the right of Start button, and exposed by clicking on the same icon. The user must select pick at least one origin and one destination for the pipeline to validate and run.

The first step is to define the Global Parameters at the pipeline level in the default Configuration Panel before dragging and dropping stages onto the canvas. There are four specific tabs that need to be populated in order to successfully set the pipeline Global Parameters:



1. General: The name, description and labels used to create the pipeline should automatically appear in this tab. Other than that, you can also change the pipeline to run in either Batch or Streaming mode by selecting from the Execution Mode drop-down.

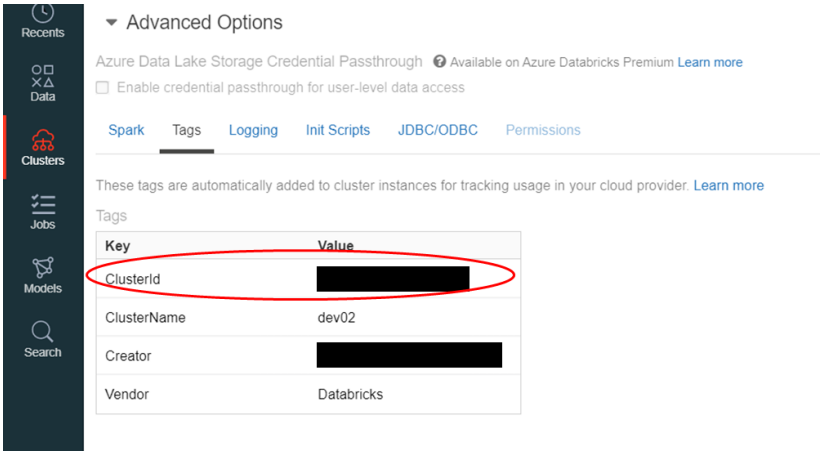
2. Advanced: Insert the Cluster Callback URL, which is the URL or IP address assigned to connect to Ingest.

3. Parameters: Users can create any global parameters as key-value pairs. Once you define the key value pair, the key can then be used as required as a parameter in any stage of the pipeline. This is useful for when you don't want to copy and paste tokens, passwords or secret keys as local parameters inside of a stage.

Once the key value is created, you can use the parameter key as a configuration element in any other form field, by using the following syntax: ${parameter_key}:

4. Cluster: This is where the user will select the cluster manager type which will execute the Spark processing, which can be Databricks, Azure HD Insight, AWS Elastic Map Reduce (EMR) or Apache Hadoop (Cloudera or MapR). In this document, we will refer to a Databricks cluster. You will need the following four data elements in order to connect to your cluster:
   a) The URL to the Databricks API Rest Server
   b) The Staging directory which contains the FlureeSense Ingest libraries needed to execute the Spark pipelines. The default directory is /ingest, which should not be modified.
   c) The Databricks Token, which is the service principal ID from Databricks which the API uses to authenticate the incoming job request from Ingest, and
   d) The Databricks Cluster ID. When defining the cluster, you can either select to create a new cluster on the fly and then terminate the cluster when the pipeline job completes, or you can provide the Databricks Cluster ID for the cluster that you want to re-use to execute the pipeline. (Note: The cluster must be activated from the Databricks console in order for the pipeline to run).

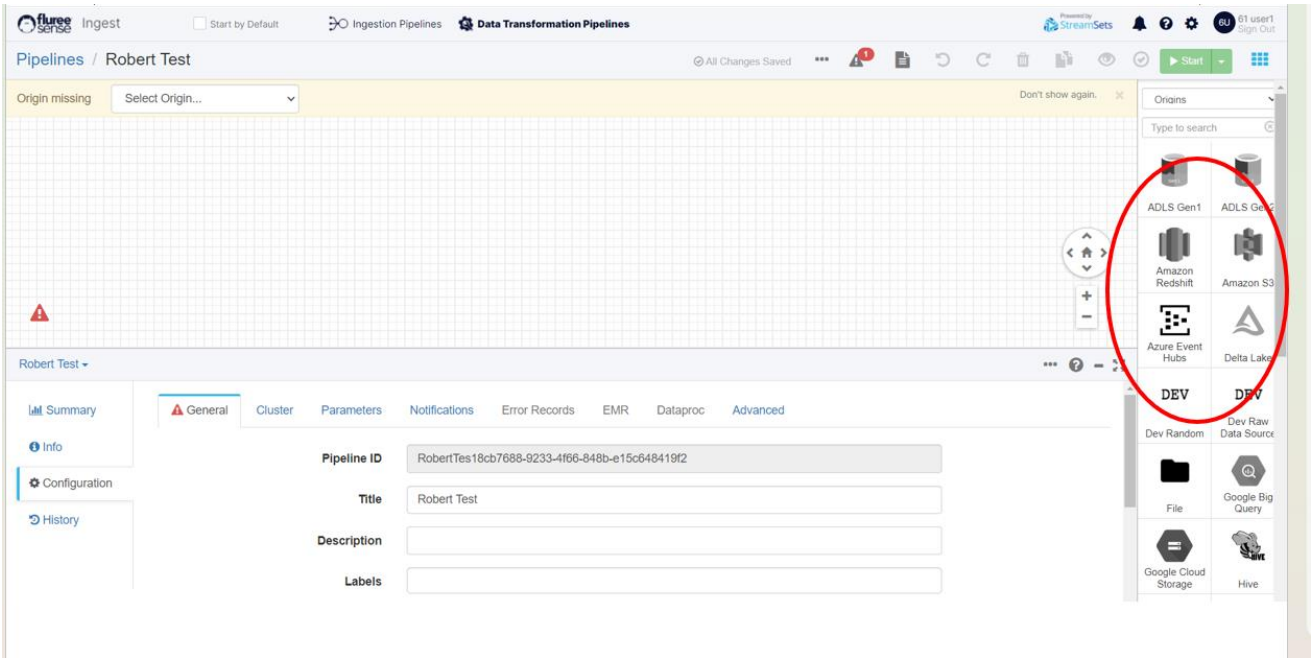
      Creating a pipeline on the fly can save money by not requiring fixed or policy managed cluster resources, but it increases the overhead required to start and execute pipelines. If you are using an existing cluster, then collect the Cluster ID from the Advanced Configuration settings in your Database management console screen:


Recents
Data
Clusters
Jobs
Models
Search
Advanced Options
Azure Data Lake Storage Credential Passthrough
Available on Azure Databricks Premium Learn more
Enable credential passthrough for user-level data access
Spark
Tags
Logging
Init Scripts
JDBC/ODBC
Permissions
These tags are automatically added to cluster instances for tracking usage in your cloud provider. Learn more
Tags
Key
Value
ClusterId
[redacted]
ClusterName
dev02
Creator
[redacted]
Vendor
Databricks

2.3 Designing Your Pipeline

2.3.1 Origins

After these global pipeline settings are inserted the user can advance to creating the flow of the pipeline by adding in an origin stage. The user can select an origin be going to the right panel and either toggling down to the origin stages or searching for a specific origin:



The 2 predominant origins a user will select will either be ADLS Gen2 or Delta Lake. The user can select either from the right panel and drag and drop into the main canvas to begin the pipeline.

After selecting an origin the user will need to define the configurations for the origin selected.

2.3.1.1 ADLS Gen 2

The ADLS Gen2 origin reads data from Microsoft Azure Data Lake Store Gen2. Every file must be fully written, include data of the same supported format, and use the same schema.

Configure an ADLS Gen2 origin to read files in Azure Data Lake Storage Gen2. Before you use the origin in a pipeline, complete the required prerequisites. Before using the origin in a local pipeline, complete these additional prerequisites.
Note: All processed files must share the same schema.

Configuring a ADLS Gen2 Origin

1. On the Properties panel, on the General tab, configure the following properties:

General Property	Description
Name	Stage name.
Description	Optional description.
Stage Library	Stage library to use: • ADLS cluster-provided libraries - The cluster where the pipeline runs has Apache Hadoop Azure Data Lake libraries installed, and therefore has all of the necessary libraries to run the pipeline.
Load Data Only Once	Reads data in a single batch and caches the results for reuse. Use to perform lookups in streaming execution mode pipelines.
Cache Data	Caches processed data so the data can be reused for multiple downstream stages. Use to improve performance when the stage passes data to multiple stages.
Skip Offset Tracking	Skips tracking offsets. In a batch pipeline, the origin reads all available data each time the pipeline starts.

2. On the ADLS tab, configure the following properties:

ADLS Property	Description
Storage Account	Azure Data Lake Storage Gen2 storage account name.
File System	Name of the file system to use.
Authentication Method	Authentication method to use to connect to Azure: • Shared Key - Can be used to run pipelines on a cluster.
Account Shared Key	Shared access key that Azure generated for the storage account.

ADLS Property	Description
Directory Path	Path to the directory that stores the source files. The origin reads files in the specified directory and subdirectories.
File Name Pattern	Glob pattern that defines the file names to process. Default is *, which processes all files.
Exclusion Pattern	Glob pattern that defines the file names to exclude from processing. Use to exclude a subset of files that are included by the File Name Pattern property. For example, if you set File Name Pattern to * to process all files in the directory, you can set File Name Exclude Pattern to *.log to exclude log files from processing.
Max Files Per Batch	Maximum number of files to process in a batch.
Post Processing	Action to take on successfully read files: • Delete - Remove the file from the directory. • Archive - Move the file to an archive directory. None - Leave the file in the directory where found.
Archive Directory	Directory to store successfully read files when the origin archives files.
Additional Configuration	Additional HDFS properties to pass to an HDFS-compatible file system. Specified properties override those in Hadoop configuration files.

3. On the Data Format tab, configure the following property:

Data Format Property	Description
Data Format	Format of the data. Select one of the following formats: • Delimited, JSON, Parquet, Text, XML

4. For delimited data, on the Data Format tab, optionally configure the following properties:

Delimited Property	Description
Delimiter Character	Delimiter character used in the data. Select one of the available options or select Other to enter a custom character.
Quote Character	Quote character used in the data.
Escape Character	Escape character used in the data
Includes Header	Indicates that the data includes a header line. When selected, the origin uses the first line to create field names and begins reading with the second line.

5. For JSON data, on the Data Format tab, configure the following property:

JSON Property	Description
Multiline	Enables processing multiline JSON Lines data. By default, the origin expects a single JSON object on each line of the file. Use this option to process JSON objects that span multiple lines.

6. Schema tab and configure the following properties:

Schema Property	Description
Schema Mode	Mode that determines the schema to use when processing data: • Infer from Data • Use Custom Schema - DDL Format • Use Custom Schema - JSON Format
Schema	Custom schema to use to process the data.

Schema Property	Description
	Enter the schema in DDL or JSON format, depending on the selected schema mode.

If the default files are in CSV format, then the Gen 2 origin stage will contain the following settings:

- Data Format: Delimited
- Includes Header: TRUE
- Delimiter: Comma
- Quote Character: Other “
- Escape Character: Other \
- Schema: Infer from Data

2.3.1.2 Delta Lake

1. The Delta Lake origin reads data from a Delta Lake table. The origin can read from a managed or unmanaged table.
2. When you configure the Delta Lake origin, you specify the path to the table to read.

Configuring a Delta Lake Origin

1. On the Properties panel, on the General tab, configure the following properties:

General Property	Description
Name	Stage name.
Description	Optional description.
Stage Library	Stage library to use to connect to Delta Lake: • Delta Lake cluster-provided libraries - The cluster where the pipeline runs has Delta Lake libraries installed, and therefore has all of the necessary libraries to run the pipeline.

General Property	Description
Load Data Only Once	Reads data in a single batch and caches the results for reuse. Use to perform lookups in streaming execution mode pipelines.
Cache Data	Caches processed data so the data can be reused for multiple downstream stages. Use to improve performance when the stage passes data to multiple stages.

2. On the Delta Lake tab, configure the following properties:

Delta Lake Property	Description
Table Directory Path	Path to the Delta Lake table.

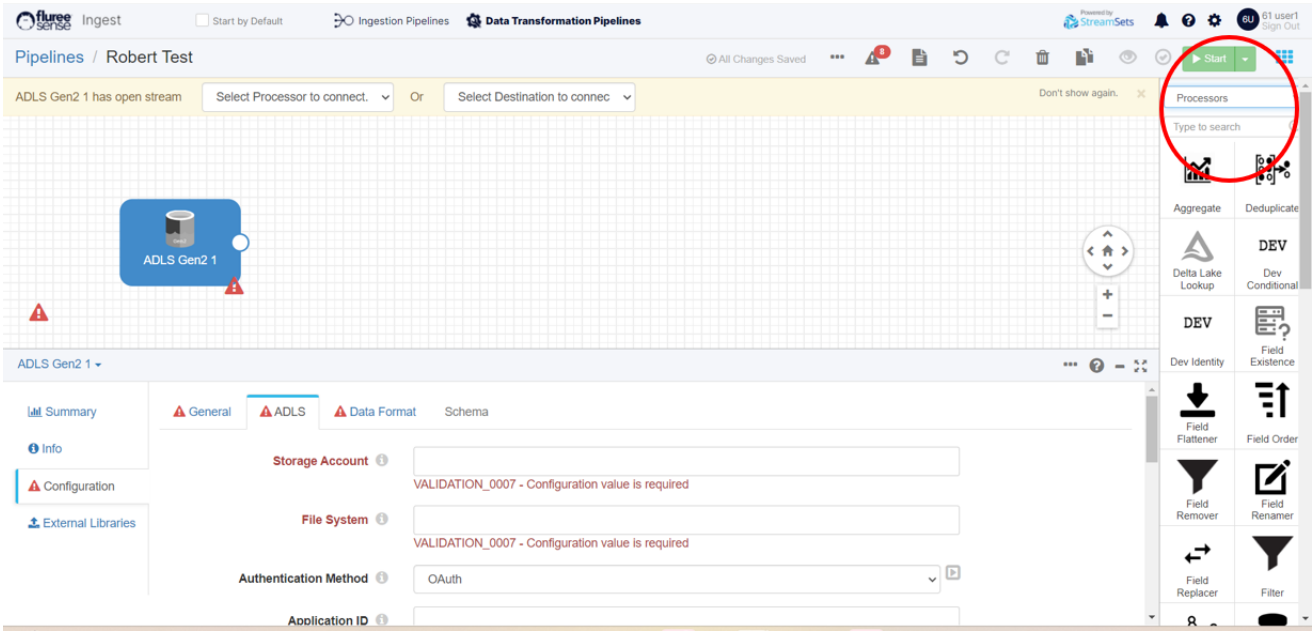
3. On the Storage tab, configure storage and connection information:

Storage	Description
Storage System	Storage system for the Delta Lake table: • ADLS Gen2 - Use for a table stored on Azure Data Lake Storage Gen2. To connect, Ingest uses the specified connection details.
Storage Account	Azure Data Lake Storage Gen2 storage account name.
File System	Name of the file system to use.
Authentication Method	Authentication method to use to connect to Azure: Shared Key - Can be used to run pipelines on a cluster.
Account Shared Key	Shared access key that Azure generated for the storage account.

After the user has set up the origin of the pipeline and which files the pipeline will read from they can go to the next step and start adding different processors before setting a final destination for the data.

2.3.2 Processors

After the origins point are set up within the pipeline the user can advance the flow of the pipeline by adding in processors. The user can select a processor from the right panel and view the gallery choices or type to search for a specific processor name:



The most common processors a user will use in the pipelines are:

Aggregate	Performs aggregate or grouping calculations. This is the equivalent of doing GROUPSUM or GROUPMAX in SQL.
Deduplicate	Removes duplicate records.
Field Remover	Removes fields or columns from a record. This is useful for if you have a record with 100 columns and you want to remove 1, or keep 10.
Filter	Passes only the records that match a filter condition.
Join	Joins data from two input streams.
Type Converter	Converts the data types of specified fields to compatible types. This is the equivalent of a CAST command in SQL.
Field Renamer	Renames a field from one name to another specified name. This is useful for doing a column mapping from a source model to a target model

Field Order	Reorders the fields in final output of data. This is useful if you want to sequence the columns in a specific order before writing to file or Delta Table.
Delta Lake Lookup	Performs a lookup on a Delta Lake table. This is like a join, but for a single record value. The Lookup can return one or more result records.
Stream Selector	Routes data to output streams based on conditions. This is useful for if you want to separate a flow based on a filter condition, e.g., if a condition is met follow Stream 1, otherwise route to Stream 2.
Spark SQL Expression	Performs record-level calculations using Spark SQL expressions. This allows you to enter direct SELECT SQL commands for specific transformations (e.g., of you want to add the values of two variables, set a constant value, call a timestamp function or other UDF, etc.)

2.3.2.1 Configuring an Aggregate Processor

1. On the Aggregate tab, configure the following properties:

Aggregate Property	Description

Aggregate Type and Field	The aggregate calculations to perform. Configure the following properties:
	• Aggregate function - The aggregation function to use in the calculation. Use one of the following functions:
	  ○ Approximate Count Version - Returns the approximate number of distinct items in a group.
	  ○ Approximate Variance - Returns the variance of a values from a group. Approximate variance uses a sample of a group rather than the full population.
	  ○ Average - Returns the average of the values in a group.
	  ○ Count - Returns the number of items in a group.
	  ○ First - Returns the value of the first record in a group.
	  ○ Last - Returns the value of the last record in a group.

Aggregate Property	Description
	○ Max - Returns the maximum value of a group. ○ Median - Returns the median of the values of a group. ○ Min - Returns the minimum value of a group. ○ Sum - Returns the sum of the values of a group. ○ Standard Deviation - Returns the standard deviation of a value from the group. Standard deviation uses a sample of a group rather than the full population. • Aggregate Field - The field to use in the aggregate calculation. • Output Field - The field for the results of the calculation.
Group By Fields	Optional fields to group by. Use to perform calculations on subsets of the data.

2.3.2.2 Configuring a Deduplicate Processor

On the Deduplicate tab, configure the following options:

Deduplicate Property	Description
Evaluate	Determines how the processor evaluates records: • All Fields - Evaluates all fields, removing only records with the exact same fields and values. The processor removes records that have the same values for all fields. • Specified Fields - Evaluates only the specified fields for matching values. The processor removes records that have the same values for every specified field.
Fields to Evaluate	One or more fields to evaluate for duplicate values when evaluating specified fields. Click the Add icon to add additional fields.

2.3.2.3 Configuring a Field Remover Processor

1. On the Remove/Keep tab, click the Add icon and then enter the name of the field to remove.
2. To remove an additional field, click the Add icon and specify another field name.

2.3.2.4 Configuring a Filter Processor

1. On the Filter tab, specify the filter condition to use.

2.3.2.5 Configuring a Join Processor

On the Join tab, configure the following properties:

Join Property	Description
Join Type	Type of join to perform: • Cross - Returns the Cartesian product of two sets of data. • Inner - Returns records that have matching values in both inputs. • Full outer - Returns all records, including records that have matching values in both inputs and records from either input that do not have a match. • Left anti - Returns records from the left input that do not have a match in the right input. • Left outer - Returns records from the left input, and the matched records from the right input. • Left semi - Returns records that have matching values in both inputs, but includes only the data from the left input. • Right anti - Returns records from the right input that do not have a match in the left input. • Right outer - Returns records from the right input, and the matched records from the left input.
Add Prefix to Field Names	Adds specified prefixes to the names of fields not specified as matching fields. Use to avoid duplicate field names from the two inputs in the joined record.
Join Criteria	Criteria used to perform the join:

Join Property	Description
	• Matching Fields - Join data based on one or more matching field names. • Condition - Join data based on a condition that you define.
Matching Fields	Names of the matching fields used to perform the join. Click the Add icon to specify each field name. The field names must be identical in both inputs.
Condition	Condition used to perform the join. The field names used in the condition must be unique in each input.

2.3.2.6 Configuring a Type Converter Processor

On the Conversions tab, configure the following properties:

Conversion Property	Description
Field Name	Name of the field to convert.
Target Type	Data type to convert to.
Precision	Precision for a Decimal field.
Scale	Scale for a Decimal field. For the Decimal data type only.

2.3.2.7 Configuring a Field Renamer Processor

On the Rename tab, configure the following properties:

Conversion Property	Description
Rename Type	Type of Filed Renamer the User is trying to perform
Filed Name	Name of Field user is targeting
New Field Name	The name of the new field the user wants to replace the old field nd name with

2.3.2.8 Configuring a Field Order Processor

1. On the Field Order tab, specify the order of the fields the user wants. Note: if the field isn't present in this list this processors will drop that field from continuing throughout the next steps of the pipeline.

2.3.2.9 Configuring a Delta Lake Lookup Processor

1. On the Lookup tab, configure the following properties:

Lookup Property	Description
Lookup Behavior	Lookup task to perform: • Return the first matching row • Return all matching rows, generating a record for each • Return a count of matching rows • Return true if matches exist, otherwise false
Lookup Keys	Lookup keys to use to find matching records. Specify the following information: • Lookup Field - Field in the record to use for the lookup. • Lookup Column - Column in the table to use for the lookup. • Operator - Spark SQL operator to use for the lookup.

Lookup Property	Description
Return Columns	Columns in the table to return and include in the matching record. Click the Add icon to specify additional columns. You can use simple or bulk edit mode to configure the columns.
Add Prefix to Column Names	Adds the specified prefix to the returned columns before adding fields to records. Use when the original record might include a field with the same name.
Prefix	Prefix to add to the returned columns.
Sorting	Sorts the matching rows in the specified order to determine how records are generated. Specify the following information: • Column to Sort - Column to use for the sort. • Sort Order - Sort order to use: ascending or descending.
Target Field	Name of the field to store the results of the lookup. Available when Lookup behavior is set to perform a count of matching rows or to return true if a match exists.

2. On the Delta Lake tab, configure the following properties:

Delta Lake Property	Description
Table Directory Path	Path to the Delta Lake lookup table.
Timestamp String	Date or timestamp to use to find matching time travel data.

3. On the Storage tab, configure storage and connection information:

Storage	Description
Storage System	Storage system for the Delta Lake table:

Storage	Description
	• ADLS Gen2 - Use for a table stored on Azure Data Lake Storage Gen2. To connect, Ingest uses the specified connection details.
Storage Account	Azure Data Lake Storage Gen2 storage account name.
File System	Name of the file system to use.
Account Shared Key	Shared access key that Azure generated for the storage account.

2.3.2.10 Configuring a Stream Selector Processor

1. On the Conditions tab, click the Add icon for each condition that you want to create, then specify the condition to use. Each new condition creates a corresponding output location. You cannot edit or delete the condition for the default stream.

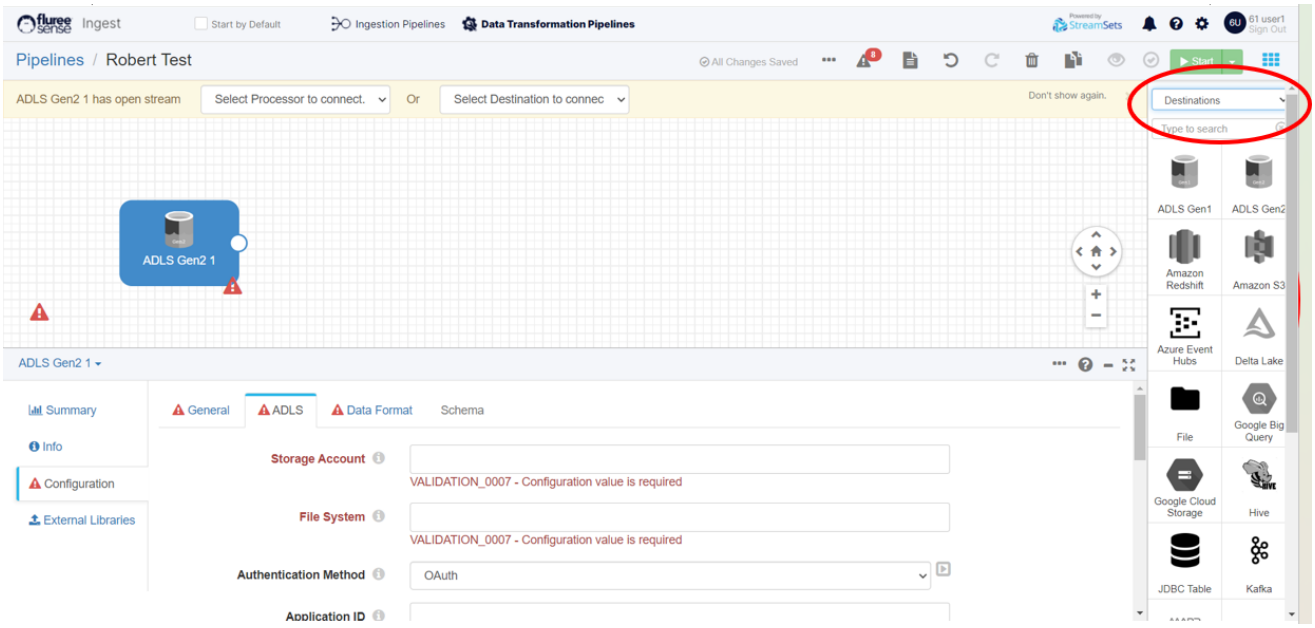
2.3.2.11 Configuring a Spark SQL Expression Processor

On the Expressions tab, configure the following properties for every expression:

Expressions Property	Description
Output Field	Output field for the results of the calculation.
SQL Expression	Spark SQL expression to use.

2.3.3 Destinations

After the user has defined the processors, the final step is to close the pipeline by providing at least one destination stage. In the Stage Selector, the user can scroll and look for, or search by keyword for the specific destination stage icon:



The 2 most common destinations that will be used are: ADLS Gen2 and Delta Lake. This will be the stage where the files land after running the pipeline.

2.3.3.1 Configuring an ADLS Gen2 Destination

1. On the General tab, configure the following properties:

General Property	Description
Name	Stage name.
Description	Optional description.
Stage Library	Stage library to use: • ADLS cluster-provided libraries - The cluster where the pipeline runs has Apache Hadoop Azure Data Lake libraries installed, and therefore has all of the necessary libraries to run the pipeline.

2. On the ADLS tab, configure the following properties:

ADLS Property	Description
Storage Account	Azure Data Lake Storage Gen2 storage account name.
File System	Name of the file system to use.
OAuth Provider Type	Authentication method to use to connect to Azure: • Shared Key - Can be used to run pipelines on a cluster.
Account Shared Key	Shared access key that Azure generated for the storage account.
Directory Path	Path to the directory for the output files. Use the following format:
Write Mode	Mode to write files: • Overwrite files - Removes all files in the directory before creating new files. • Overwrite related partitions - Removes all files in a partition before creating new files for the partition. Partitions with no data to be written are left intact. • Write new or append to existing files - Creates new files in an existing directory. • Write new files to new directory - Creates a new directory and writes new files to the directory. Generates an error if the specified directory exists when you start the pipeline.
Additional Configuration	Additional HDFS properties to pass to an HDFS-compatible file system. Specified properties override those in Hadoop configuration files.
Partition by Fields	Fields to use to partition the data.

3. On the Data Format tab, configure the following property:

Data Format Property	Description
Data Format	Format of the data. Select one of the following formats: Delimited, JSON, Parquet, Text, XML

4. For delimited data, on the Data Format tab, configure the following property:

Delimited Property	Description
Delimiter Character	Delimiter character to use in the data. Select one of the available options or select Other to enter a custom character.
Quote Character	Quote character to use in the data.
Escape Character	Escape character to use in the data

5. For text data, on the Data Format tab, configure the following property:

Text Property	Description
Text Field	String field in the record that contains the data to be written. All data must be incorporated into the specified field.

2.3.3.2 Configuring a Delta Lake Destination

1. On the Properties panel, on the General tab, configure the following properties:

General Property	Description
Name	Stage name.

General Property	Description
Description	Optional description.
Stage Library	Stage library to use to connect to Delta Lake: • Delta Lake cluster-provided libraries - The cluster where the pipeline runs has Delta Lake libraries installed, and therefore has all of the necessary libraries to run the pipeline.

2. On the Delta Lake tab, configure the following properties:

Delta Lake Property	Description
Table Directory Path	Path to the Delta Lake table to write to.
Write Mode	Mode to write to an existing Delta Lake table: • Append Data - Appends new data to the table, leaving existing data in place. • Overwrite Data - Removes all existing data from the table before writing new data, by default. You can use an overwrite condition to remove only parts of the existing table. • Upsert Using Merge - Merges new data with existing data in the table based on the specified merge condition, merge clauses, and timestamp field. You can use this write mode to perform inserts, updates, and deletes. • Update Table - Updates an existing table based on an update condition and update set rules. This write mode executes the configured updates each time the destination receives a batch of records, but does not use the records to perform the updates. • Delete from Table - Deletes rows from an existing table that match the specified condition. This write mode deletes records each time the destination

Delta Lake Property	Description
	receives a batch of records, but does not delete the records that it receives. It performs the deletes based only on the specified condition.
Partition Columns	List of table columns to partition the data by. The destination partitions the data so that records with the same value for the specified columns are in the same partition.
Merge Schema	Adds new columns to the existing table when records contain unexpected fields or data types.
Update Schema	Method used to update the schema of existing tables: • Overwrite Schema - When the pipeline starts, the destination updates the table schema to match the schema of the first batch of data. The schema update occurs only once, at the beginning of the pipeline run. Subsequent records with incompatible schemas cause the pipeline to stop. • Merge Schema - When the data contains unexpected fields, the destination creates matching columns in the table to enable writing the data. The destination does not delete columns from the table. • No Schema Update - The destination performs no updates to the schema. Data with unexpected fields or data types causes the pipeline to stop.

3. On the Storage tab, configure storage and connection information:

Storage	Description
Storage System	Storage system for the Delta Lake table: • ADLS Gen2 - Use for a table stored on Azure Data Lake Storage Gen2. To connect, Ingest uses the specified connection details.
Storage Account	• Azure Data Lake Storage Gen2 storage account name.

Storage	Description
File System	Name of the file system to use.

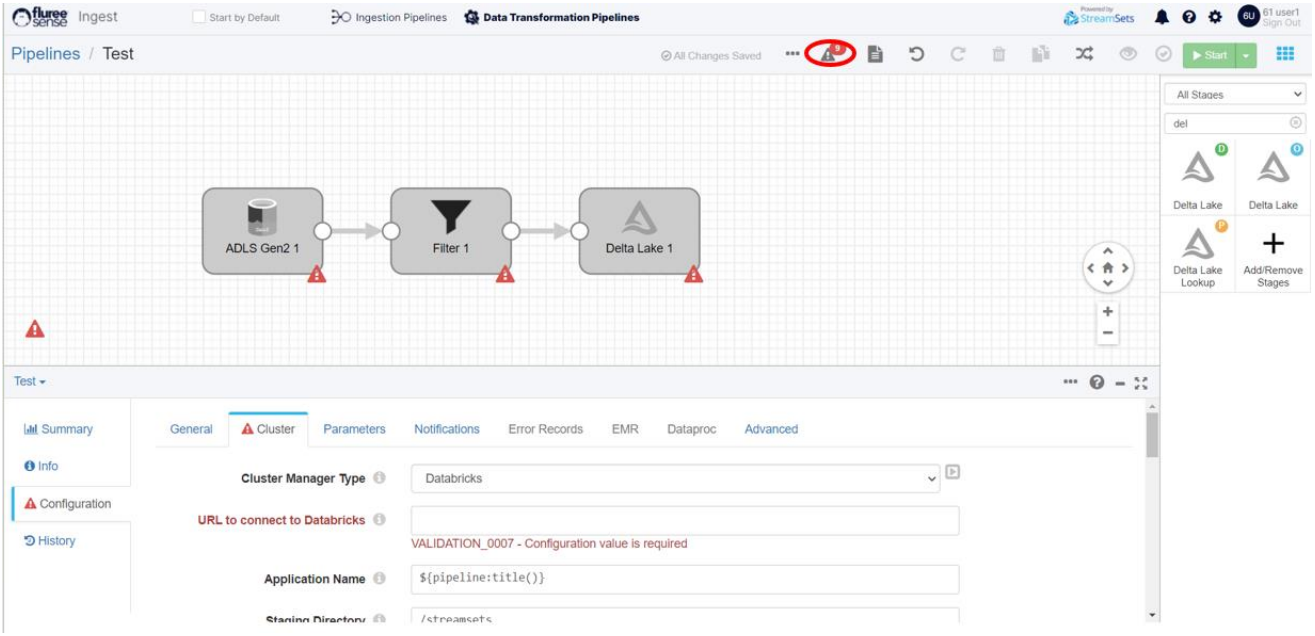
4. Optionally configure the following advanced properties:

Advanced Property	Description
Create Managed Table	Creates a Delta Lake managed table at the specified directory path. Use only when the pipeline runs on a Databricks cluster.
Table Name	Name of the table to create.
Additional Configuration	Additional HDFS configuration properties to use. Properties set here override the Hadoop configuration file.

3. Running and Executing Pipelines

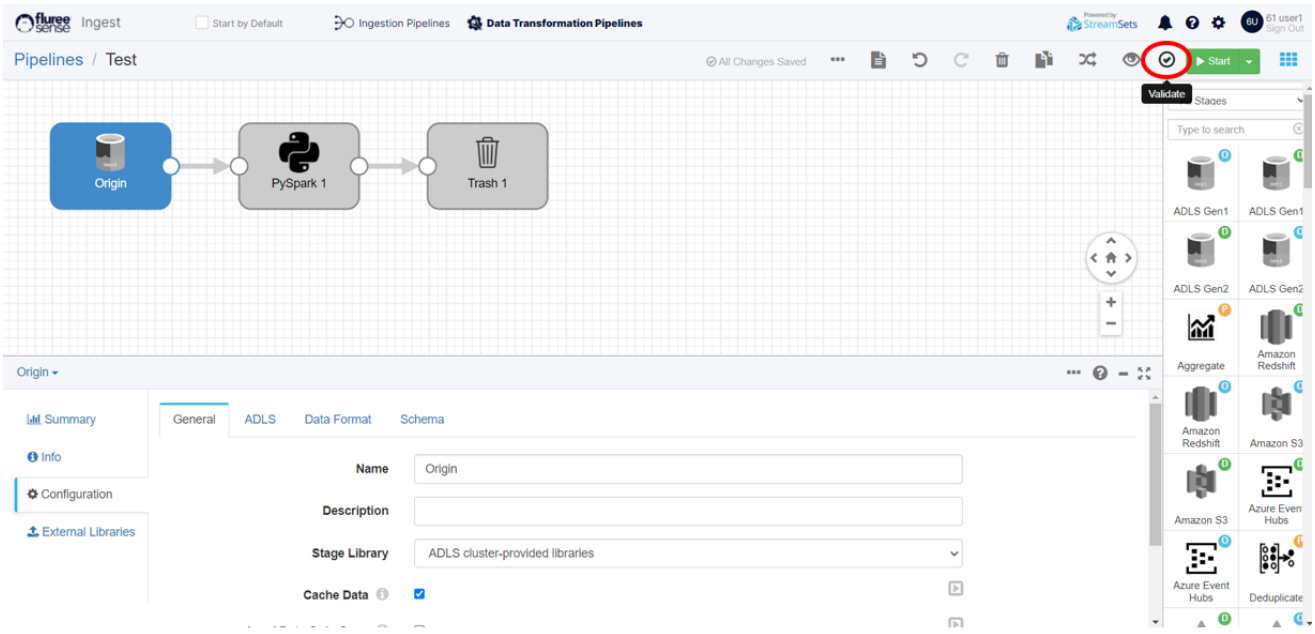
After the stages are completed and the pipeline has an origin and a destination the user is ready to try to run the pipeline.

1. The first thing the user will check is to see if there are any errors appearing on the UI that wouldn't allow the pipeline to run:



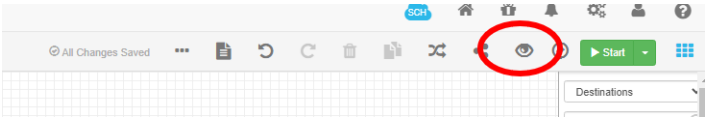
Looking at the top panel the user can find the Issues tab. This will allow the user to see if the pipeline is detecting any errors and in which stages they are occurring in. Before running the pipeline the user will need to correct any of this issues and make sure the tab doesn't show any errors.

2. After the pipeline is checked for issue the user can test the pipeline by clicking on the validate button on the top right of the pipeline. This will allow the user to run the stages of the pipeline without any data to check if the pipeline will run smoothly from origin to destinations. This can be where the user finds any errors that weren't apparent at first.



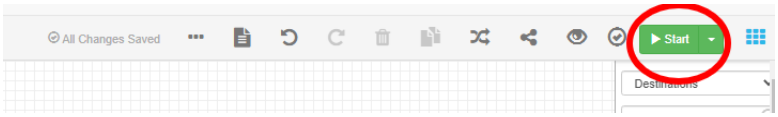
3. After the pipeline is validated the user may choose to run a preview of the pipeline. This will allow the user to track the actual data flowing throughout the pipeline at the various stages. This is only a preview of the pipeline so the user will only pick a sample data amount to run through the preview. The user can choose which stages the preview should run through, it can run through the entire pipeline end to end, or a specific stage in the pipeline that the user wants to debug. The preview is especially helpful when the user wants to look at the specific data to ty to view how it will look at the end of the pipeline flow as well as to debug any errors that are arising in the pipeline.

To run the preview the user will click on the eye button located in the top right of the pipeline:



When the pipeline runs… (need to say something about the results screen and how to navigate it)

4. After the validations and previews are completed and there are no errors in the pipeline, the pipeline now can be turned on and ran. The user can trigger the pipeline by hitting the Green Start button on the upper right:



When starting the pipeline, the user can choose from several options:

- Start Pipeline: Turns on a pipeline and resumes from wherever it left off. If the pipeline was partially through processing a file, it will set an offset for that file and pickup from the offset position. This is usually the way to start a Streaming or Change Data Capture pipeline.
- Reset Offsets & Start: Starts the pipeline process from scratch and the pipeline from The user can choose to start the pipeline, or reset offsets and starting by clicking the down arrow attached to the button. That is recommended if the user has many any changes or edits to the pipeline.