SDC Pipeline Example: SFTP to S3 External Stage to Snowflake

Introduction

This document describes an example Data Collector pipeline that retrieves files from an SFTP site using [Whole File Data Format](), copies the files to an S3-based external staging directory, and then executes COPY INTO commands to load the files' records into a Snowflake table.

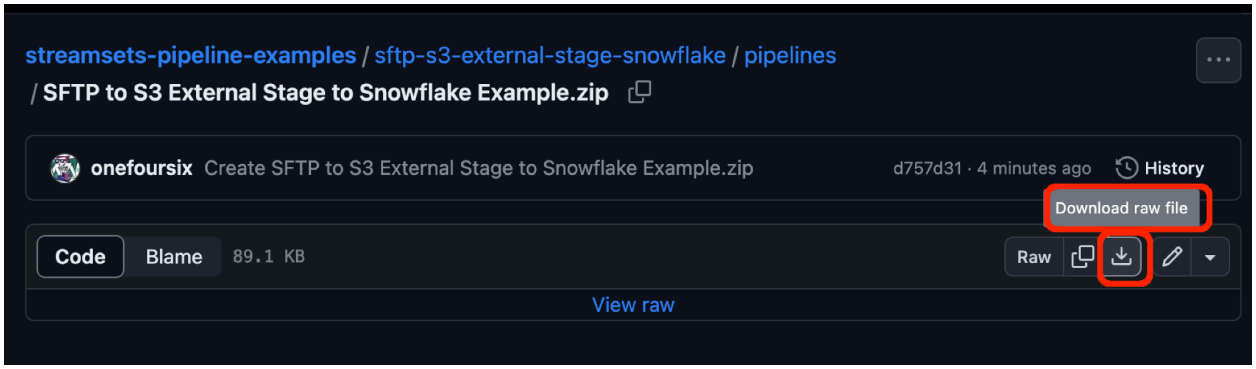
All artifacts are available in this GitHub repo:
https://github.com/onefoursix/streamsets-pipeline-examples

Prerequisites

- The target Snowflake table must exist in advance
- The S3-based External Stage must exist in advance
- A Snowflake File Format must exist that correctly parses the source files

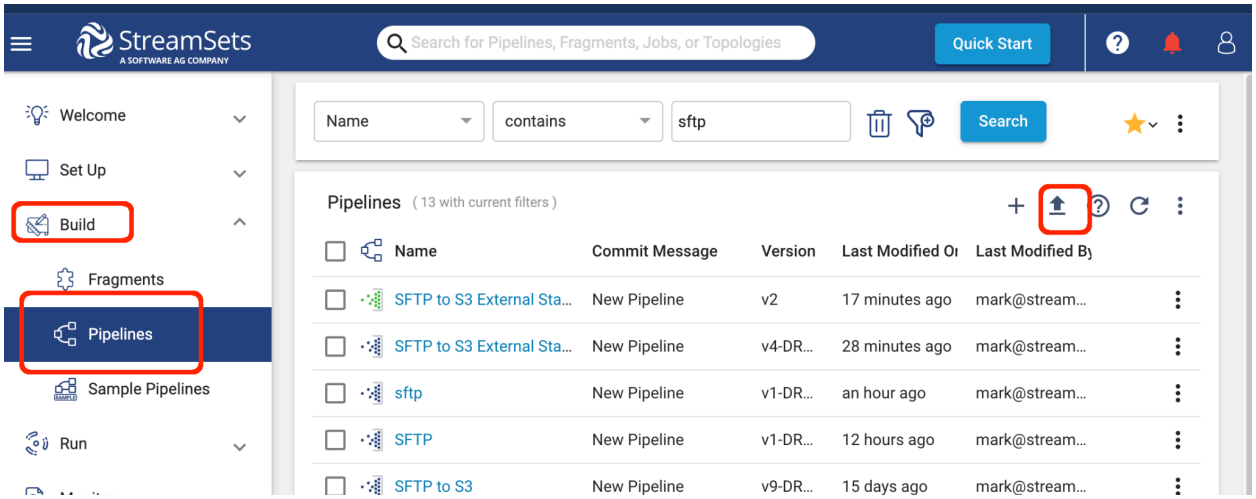
Example Pipeline Download Link

The pipeline can be downloaded (as a zip file) using the download button on [this page]():

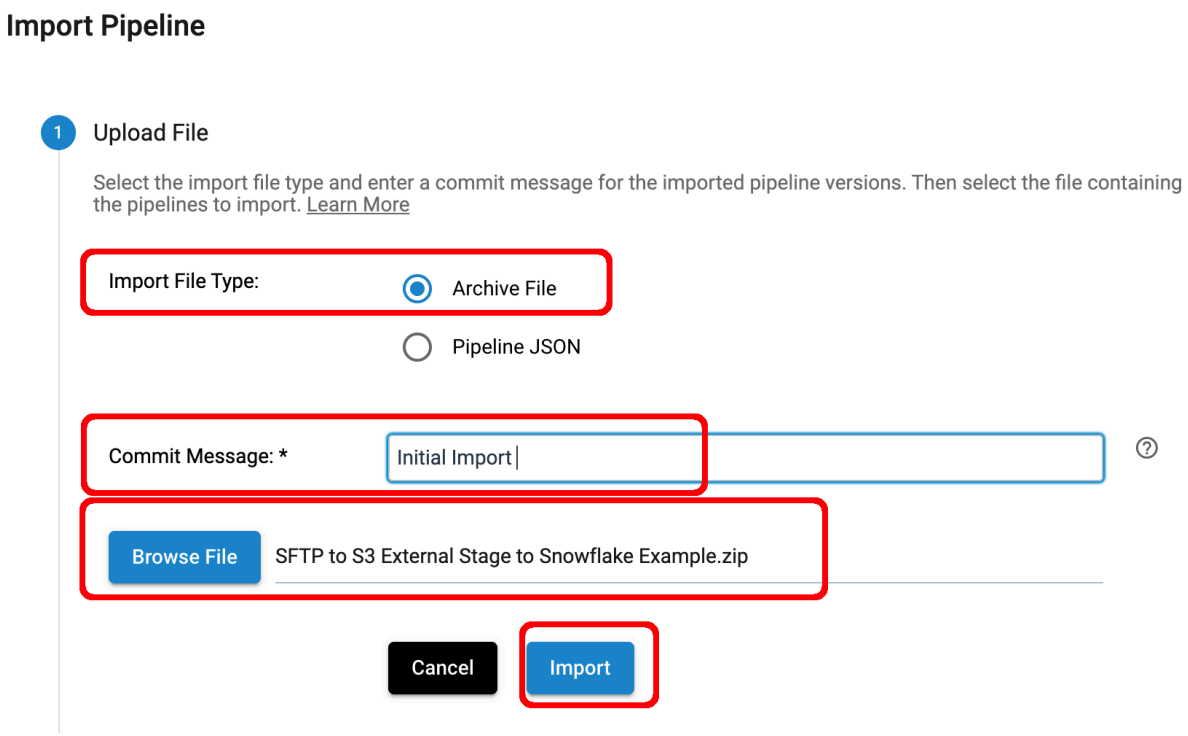


Import the Pipeline into your environment

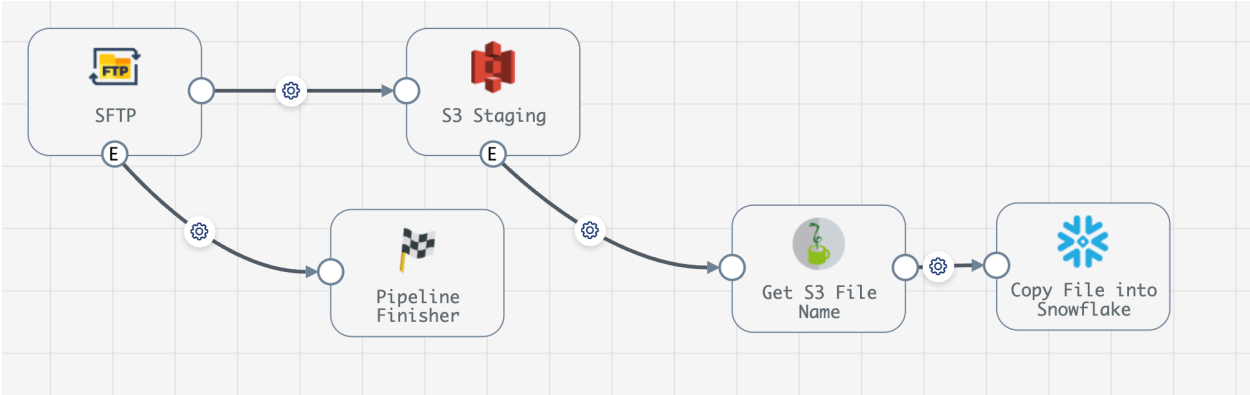
Use the upload button in the Pipeline List View to import the pipeline:



Browse to the download zip file, and import it as a Pipeline Archive:

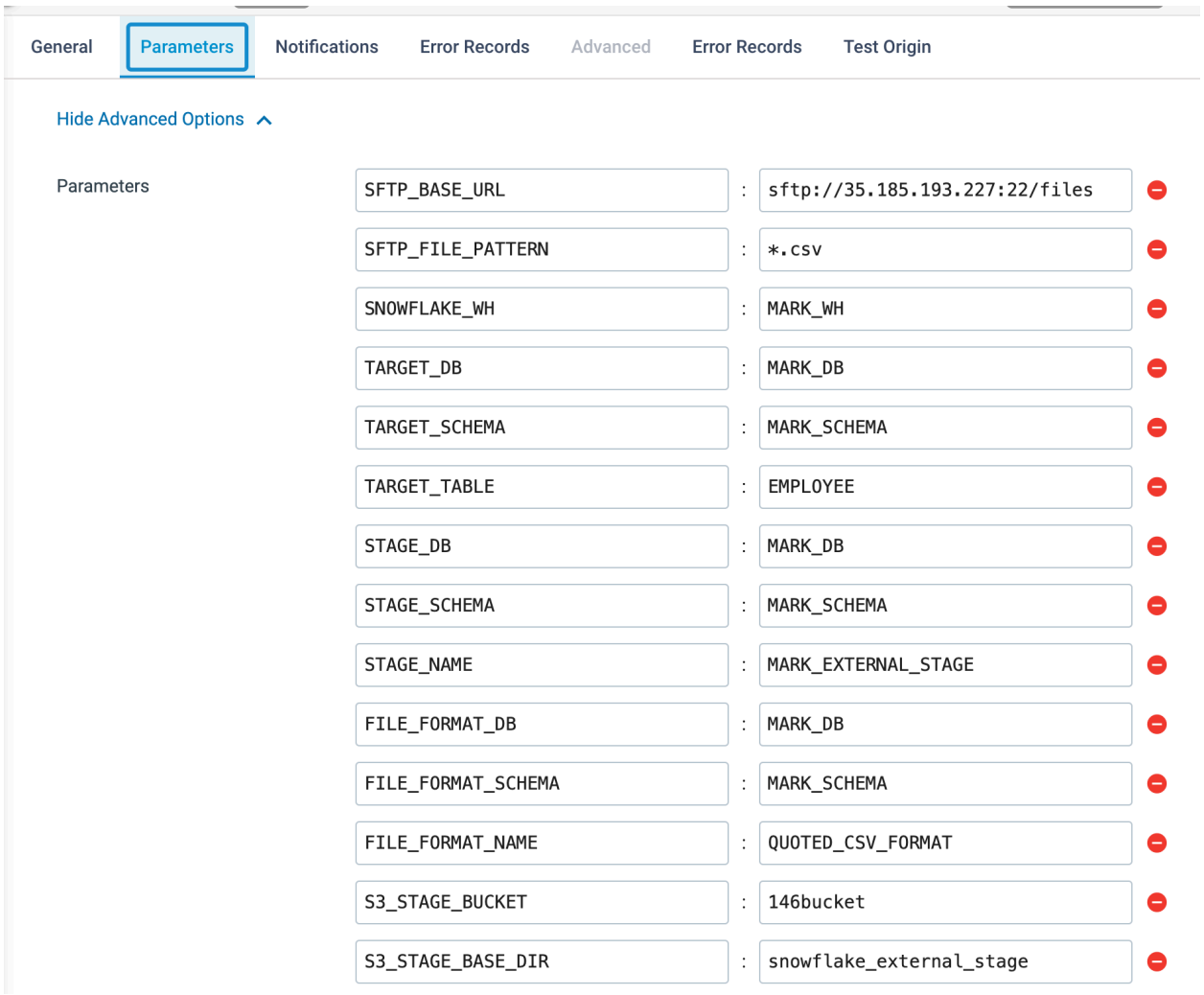


The imported pipeline should look like this:



Set the Pipeline's Parameters

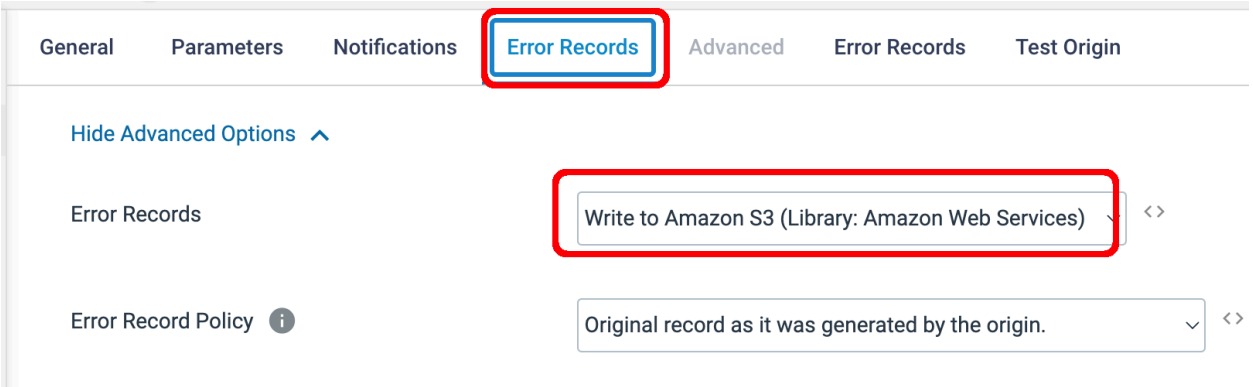
Edit the imported pipeline and set the pipeline's parameters to fit your environment. The example pipeline has these sample settings from my own environment:



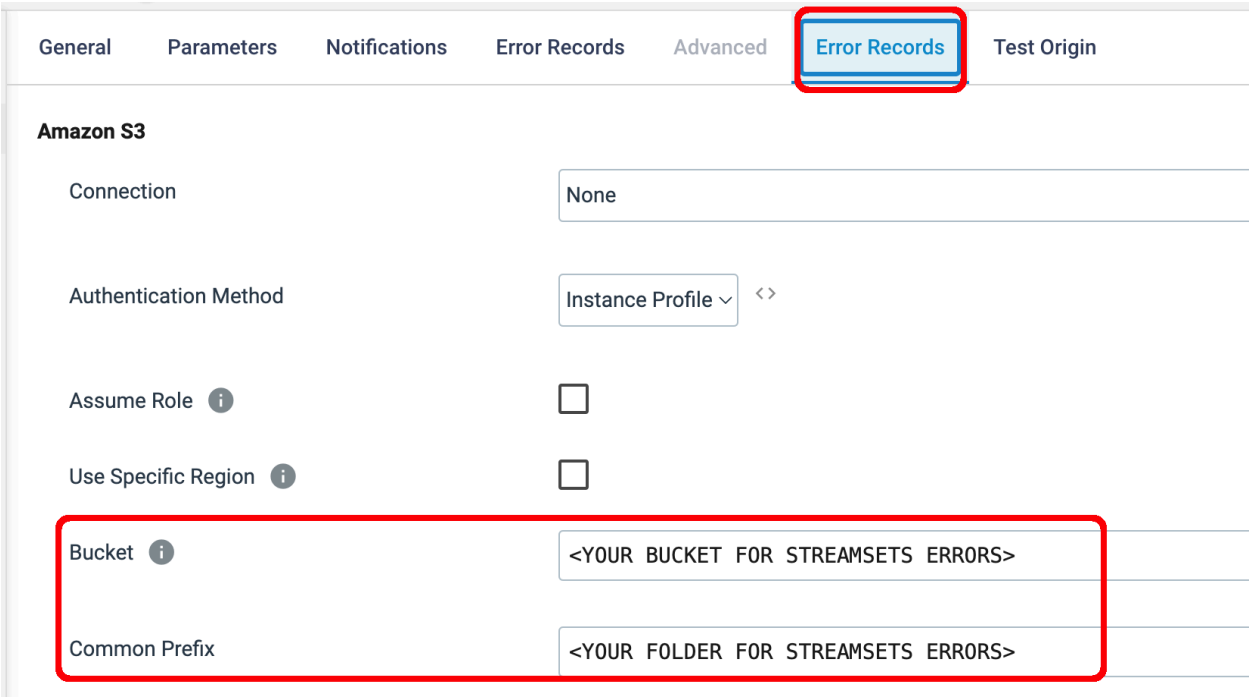
The use of each parameter will be shown in the steps that follow.

Set an Error Record Location

Set the error record location to S3:

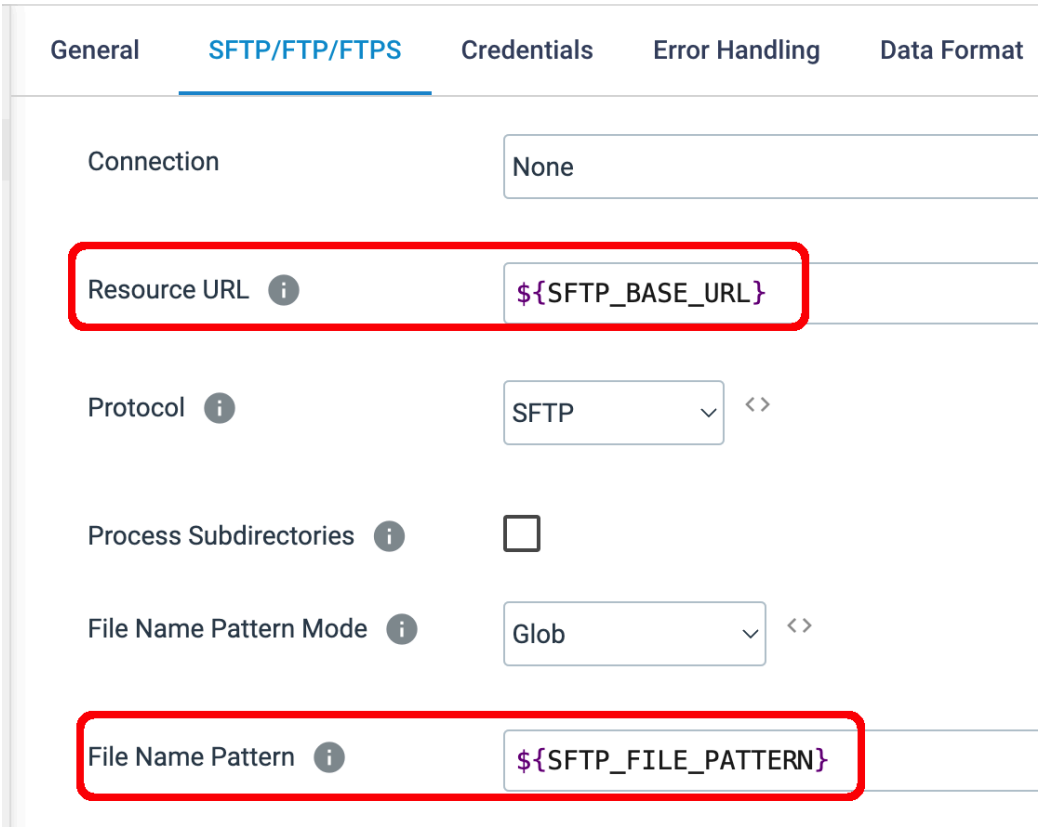


Set an S3 Bucket and folder to hold error records:

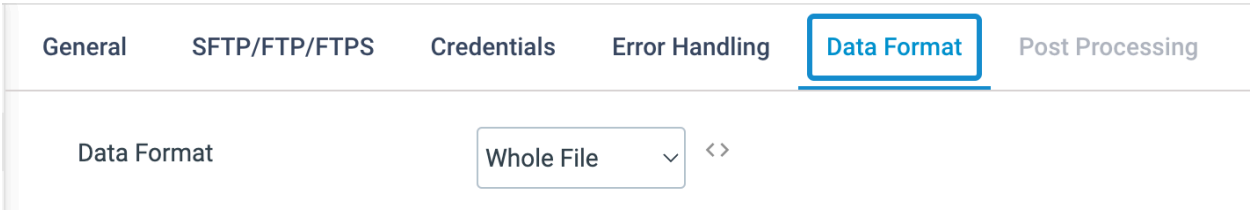


SFTP Origin Stage

The pipeline has an SFTP Origin configured like this:

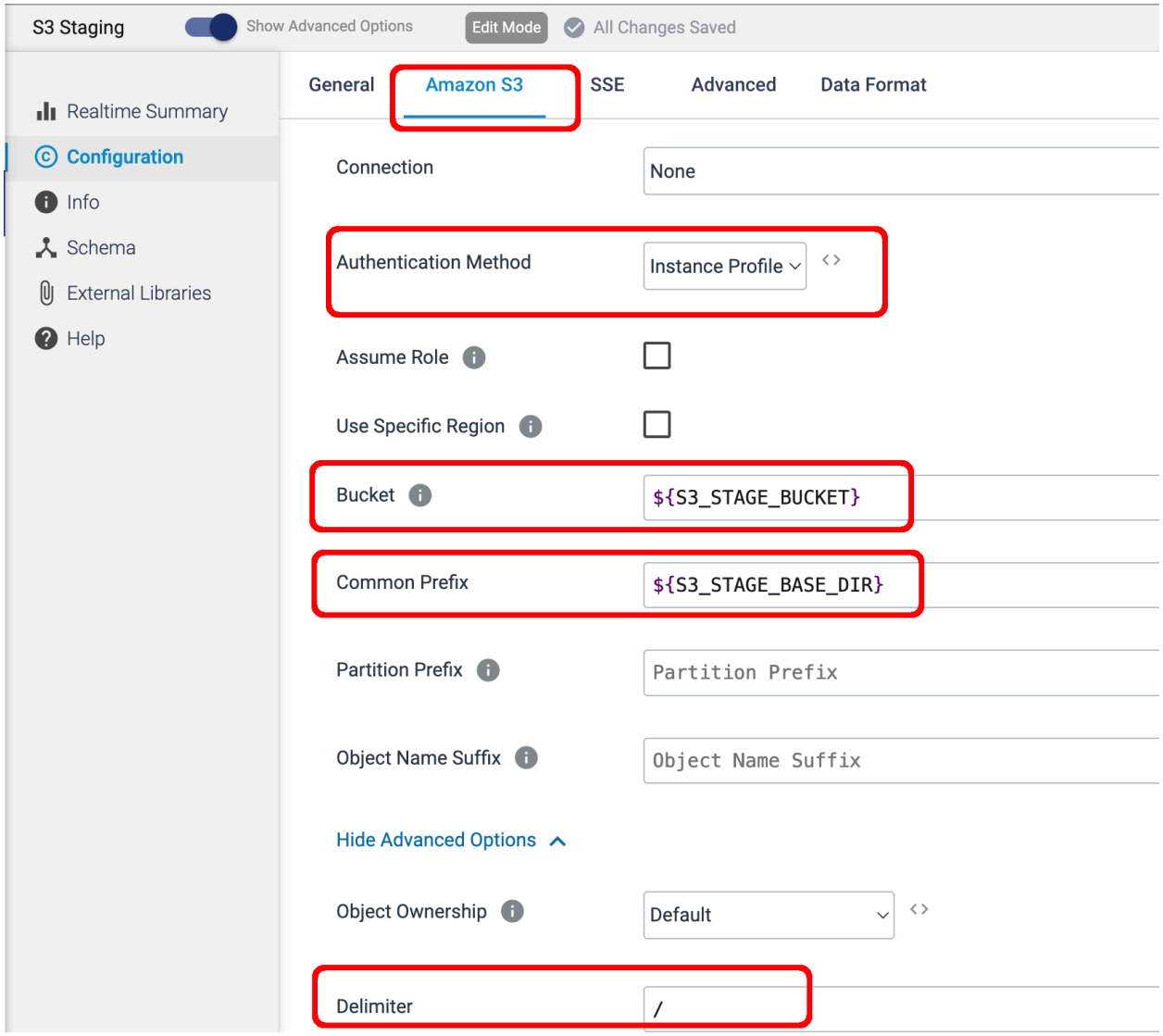


The Data Format is pre-set to Whole File Data Format:

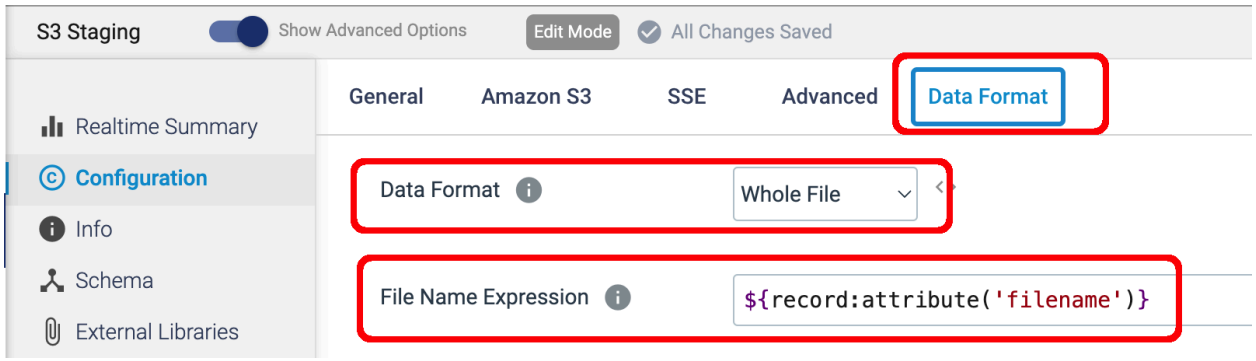


S3 Staging Stage

The `S3 Staging` stage is an [S3 Destination] configured as follows:



The Data Format is pre-set to Whole File Data Format and will preserve the source file names:



If you want to ensure unique file names, you could append a timestamp or a UUID to the File Name Expression. Info on these and other functions is here.

The S3 stage is configured to emit events, which will generate Whole File Processed events.

Get S3 File Name Stage

The `Get S3 File Name` stage is a Jython Evaluator configured to extract the S3 file name from the S3 Object Key. This step is needed in my example because the object key includes the base directory path of my Snowflake stage and I need to extract just the file name.

The current version of this example assumes files are written to the top level of the staging directory, though the example could easily be extended to recurse subdirectories. And there are many alternative implementations of this step that could be considered.

Here is the Jython script used in the example:

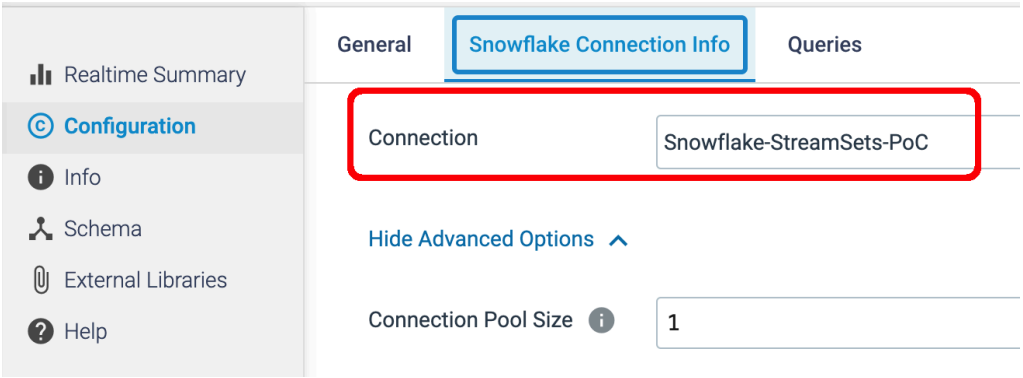
```
# Get the file name from the S3 objectKey.

for record in sdc.records:
  try:

    # Get the S3 objectKey
    object_key = record.value['targetFileInfo']['objectKey']

    # Split the objectKey path
    splits = object_key.split('/')

    # Grab the last split
    s3_file_name = splits[len(splits) - 1]

    # Assign the file_name to the record
    record.value['s3_file_name'] = s3_file_name

    # Write the record
    sdc.output.write(record)

  except Exception as e:
    sdc.error.write(record, str(e))
```

The result is this stage appends a field named `s3_file_name` to each record. That field is used in the COPY INTO command performed in the next step

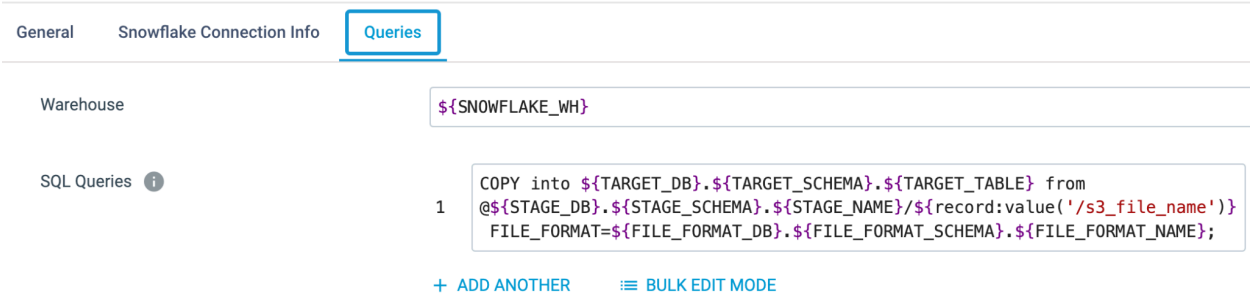
Copy File into Snowflake

The Copy File into Snowflake stage is a Snowflake Executor configured like this:

I used a pre-defines Snowflake Connection:



And here is the heavily parameterized COPY INTO query:



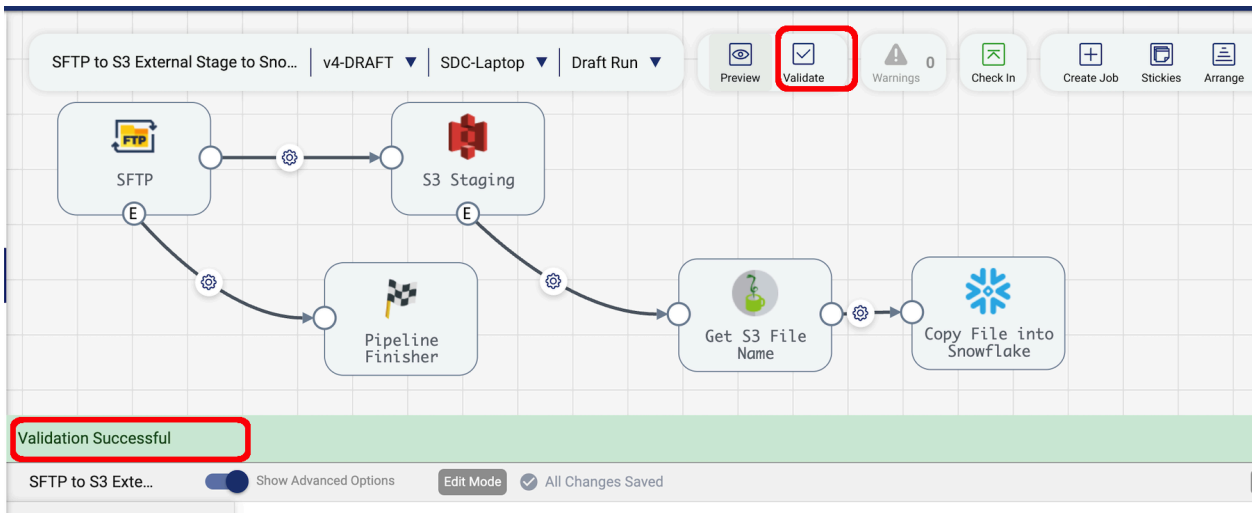
The SQL Query is this:

```
COPY into ${TARGET_DB}.${TARGET_SCHEMA}.${TARGET_TABLE} from
@${STAGE_DB}.${STAGE_SCHEMA}.${STAGE_NAME}/${record:value('/s3_file_name')}
FILE_FORMAT=${FILE_FORMAT_DB}.${FILE_FORMAT_SCHEMA}.${FILE_FORMAT_NAME};
```

Note that almost all values are parameterized in the query, and the name of the S3 file to COPY is retrieved from the `s3_file_name` field set by the previous stage.

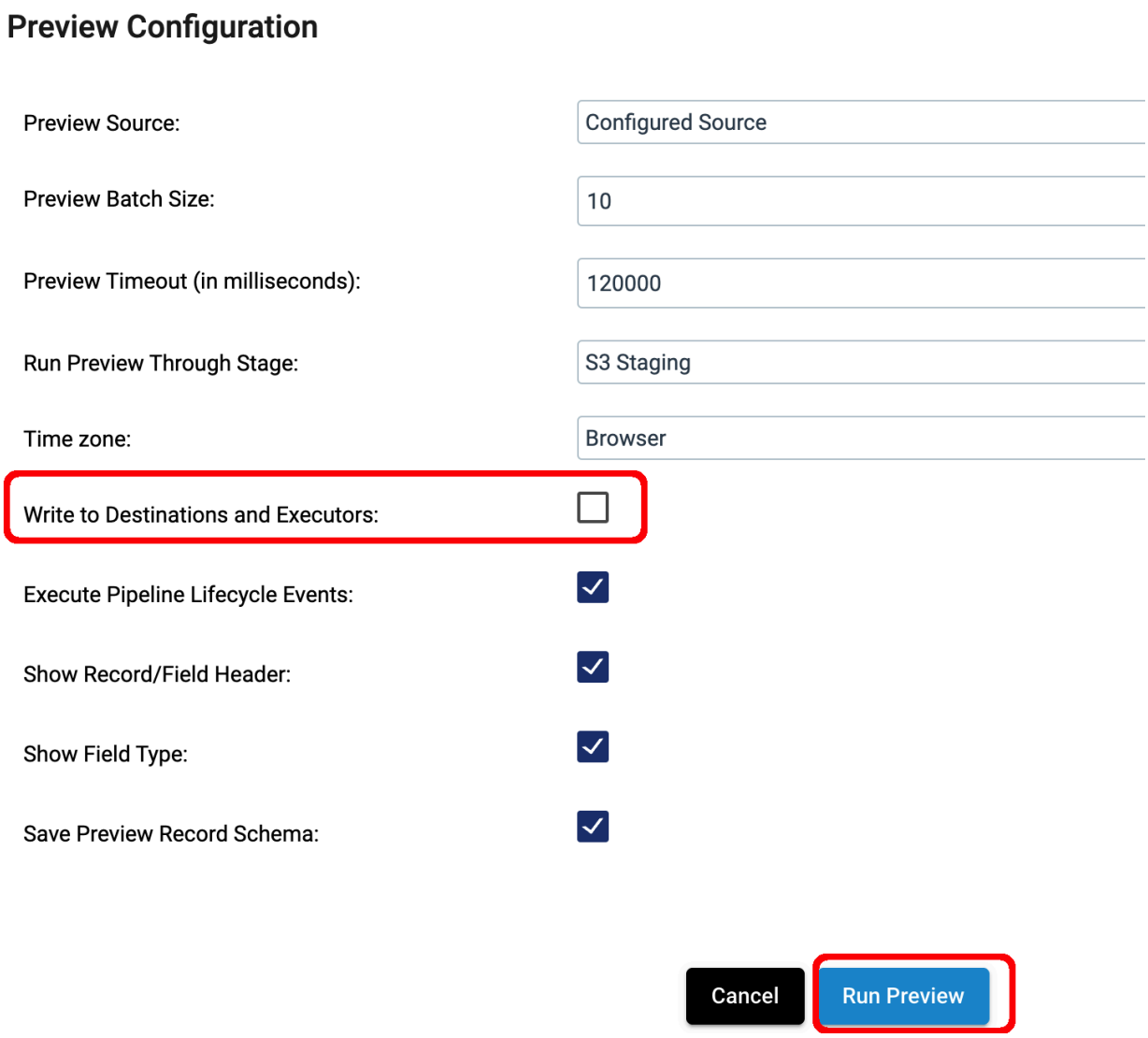
Validate the Pipeline

Validate the pipeline and make sure the validation is successful:

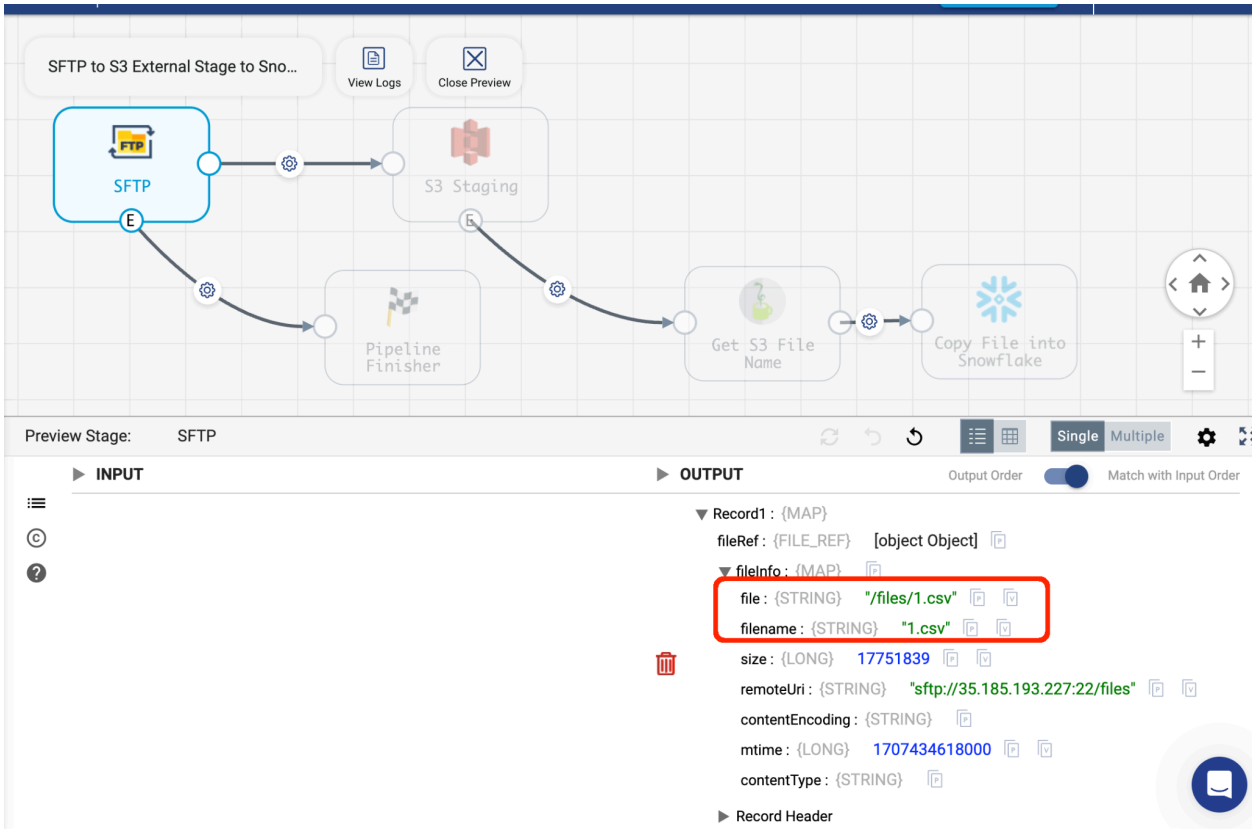


Preview the Pipeline

Choose `Preview > Configure Preview` and unset the Write to Destinations and Executors checkbox, the click Run Preview



You should be able to confirm that the SFTP connector is reading at least one file. For example, in my environment I can see I'm able to pick up the file `1.csv`:



Exit the preview.

Run the Pipeline

In my test, I have a single file named 1.csv on my SFTP server with 20,000 records in it.

Before I run the pipeline my S3 staging directory is empty, and my Snowflake target table has no rows in it.

Start the pipeline and when it has finished reading files from the SFTP server it will stop. In my environment, when the pipeline completes, I see the file in my S3 staging directory and 20,000 rows in my Snowflake table.

Click on the Snowflake stage to see it received one event and executed one COPY INTO command against Snowflake, and there were no errors: