Sharing Parameters Across Jobs

Introduction

This example shows how to share parameters across Jobs.

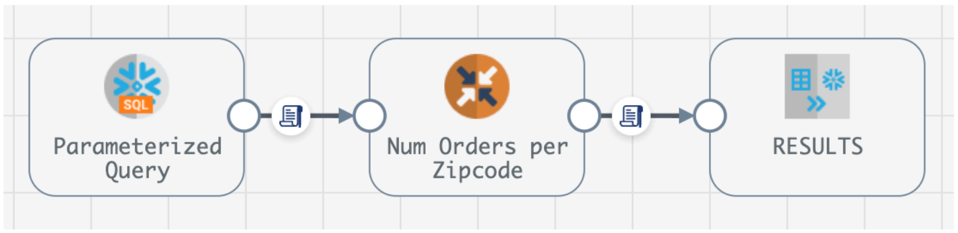
For example, let's say you want to start a Job that loads a file from S3 into a Snowflake table using Data Collector, and when that Job completes, you want to run an aggregation on the table you just loaded using a Transformer for Snowflake Job.

The design approach described in this example includes the following components:

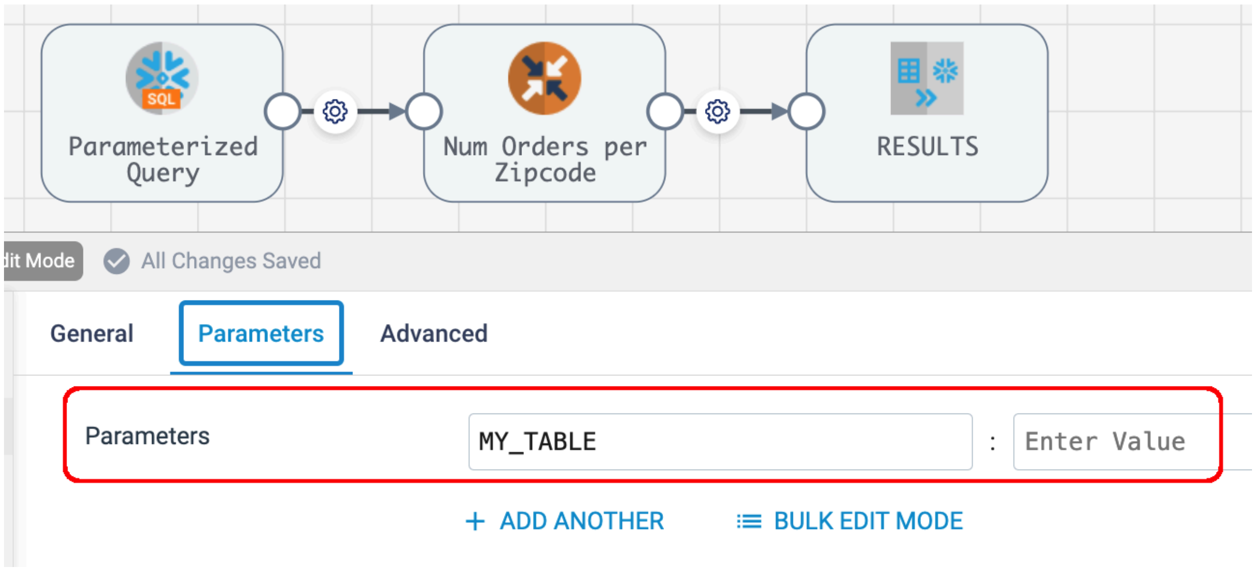
- A Transformer for Snowflake Job with a parameter named `MY_TABLE`
- A Data Collector Job with the parameters `S3_DIR`, `S3_FILE`, and `TARGET_TABLE`
- A `Load and Transform` Orchestration Pipeline that coordinates the work, and feeds the necessary parameters to the appropriate Jobs.

Create the Transformer for Snowflake Pipeline

I'll use a simple Transformer for Snowflake pipeline named `Query with Parameter` that looks like this:

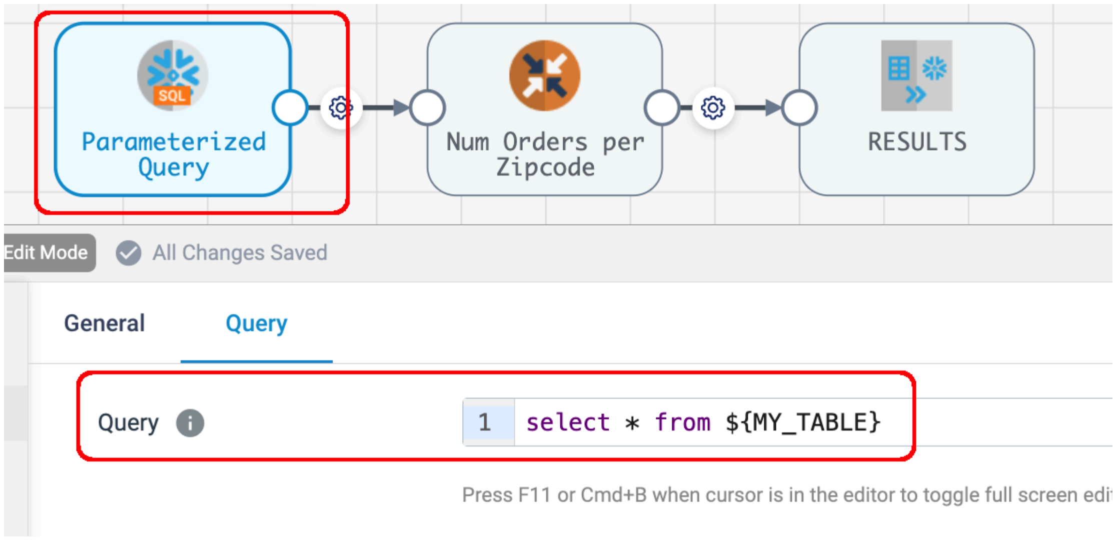


The pipeline has one parameter named `MY_TABLE` with no default value:



Parameters MY_TABLE : Enter Value

The Origin is a Snowflake Query with this config:



Query

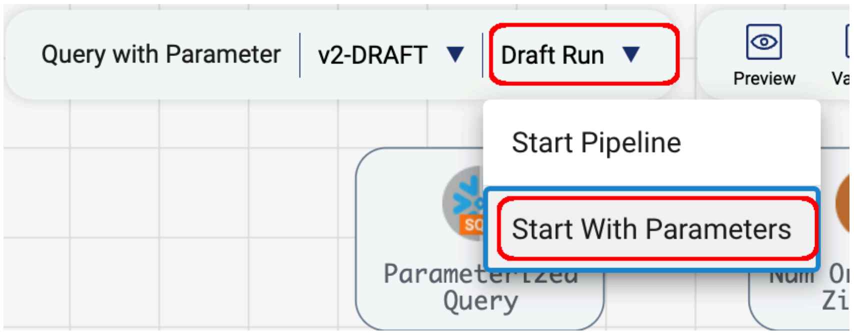
```
select * from ${MY_TABLE}
```

Press F11 or Cmd+B when cursor is in the editor to toggle full screen edi

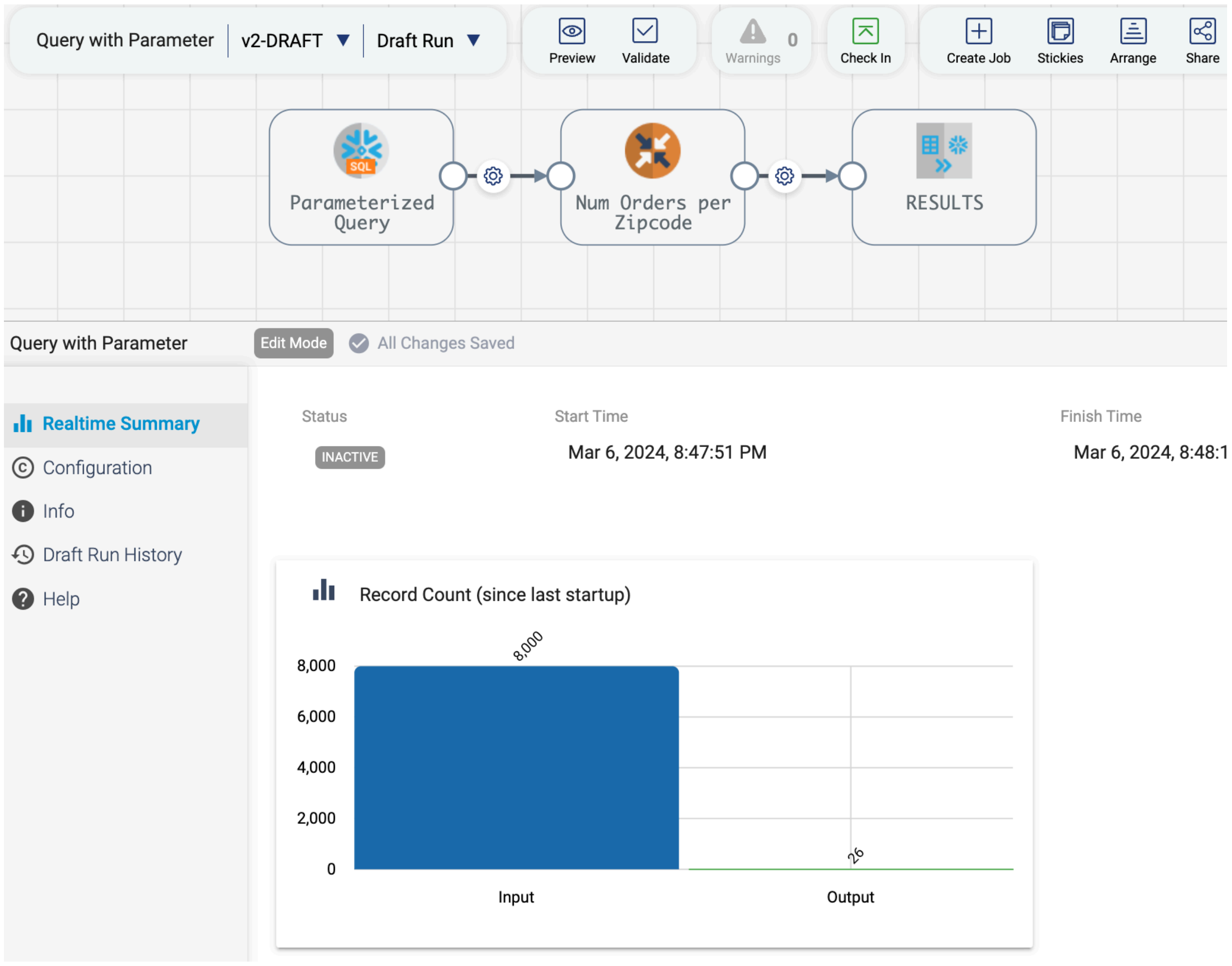
The subsequent stages perform an Aggregation and writes to a target table but are not relevant to the example.

Test the Transformer for Snowflake Pipeline

A virtue of this design approach is that every part can be tested independently. To test the Transformer for Snowflake pipeline, choose `Draft Run > Start with Parameters`



Provide a value for the `MY_TABLE` parameter and confirm the pipeline draft run completes successfully:



Status: INACTIVE

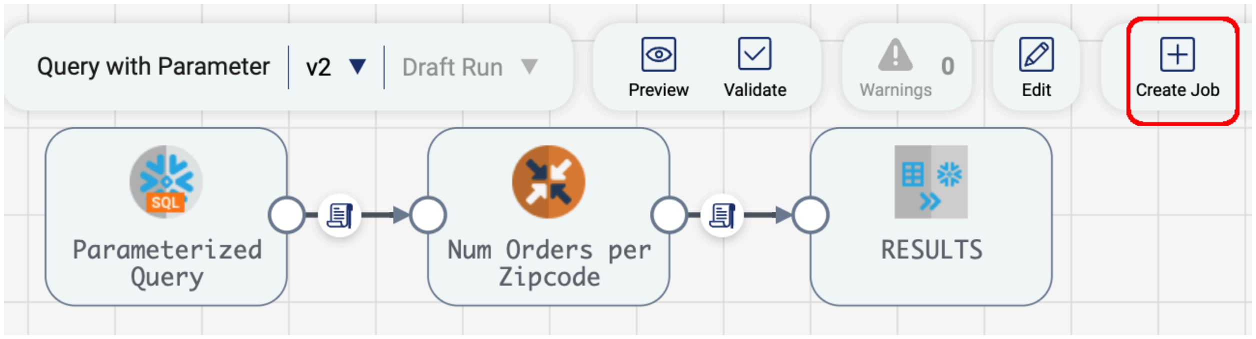
Start Time: Mar 6, 2024, 8:47:51 PM

Finish Time: Mar 6, 2024, 8:48:1

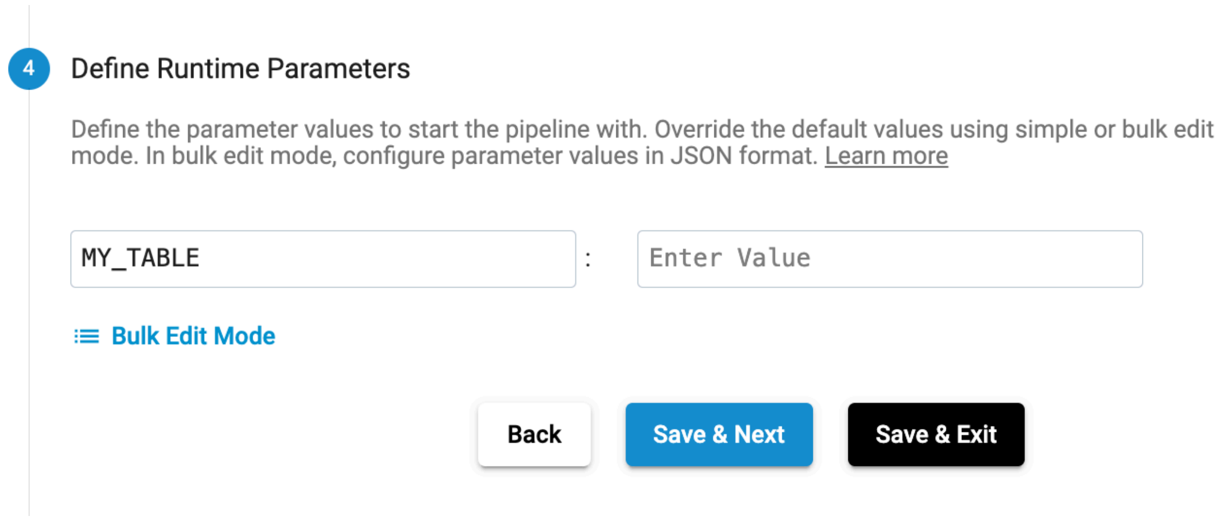
Record Count (since last startup): Input 8,000; Output 26

Create a Job for the Transformer for Snowflake Pipeline

Check in any changes to the pipeline and click the `Create Job` button:



Accept all the default values in the new Job, and leave the `MY_TABLE` parameter blank:



4 Define Runtime Parameters

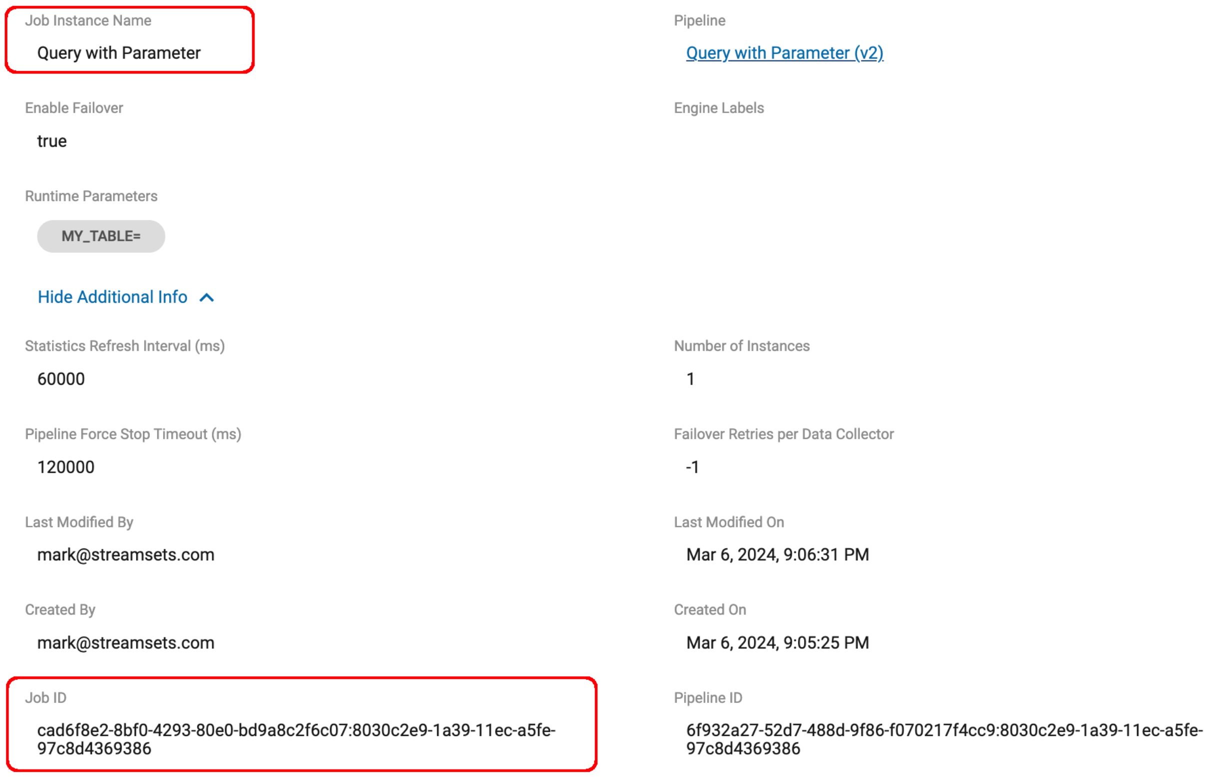
Define the parameter values to start the pipeline with. Override the default values using simple or bulk edit mode. In bulk edit mode, configure parameter values in JSON format. Learn more

MY_TABLE : Enter Value

Click `Save & Next` and then `Exit`

→ The Transformer for Snowflake Job has been created.

Click on the Transformer for Snowflake Job and copy its Job ID as we'll need it later:



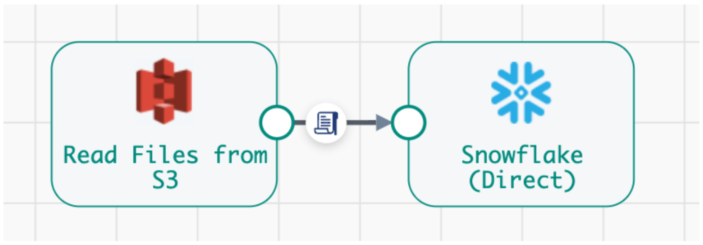
Job Instance Name Query with Parameter	Pipeline Query with Parameter (v2)
Enable Failover true	Engine Labels
Runtime Parameters MY_TABLE=	

Hide Additional Info

Statistics Refresh Interval (ms) 60000	Number of Instances 1
Pipeline Force Stop Timeout (ms) 120000	Failover Retries per Data Collector -1
Last Modified By mark@streamsets.com	Last Modified On Mar 6, 2024, 9:06:31 PM
Created By mark@streamsets.com	Created On Mar 6, 2024, 9:05:25 PM
Job ID cad6f8e2-8bf0-4293-80e0-bd9a8c2f6c07:8030c2e9-1a39-11ec-a5fe-97c8d4369386	Pipeline ID 6f932a27-52d7-488d-9f86-f070217f4cc9:8030c2e9-1a39-11ec-a5fe-97c8d4369386

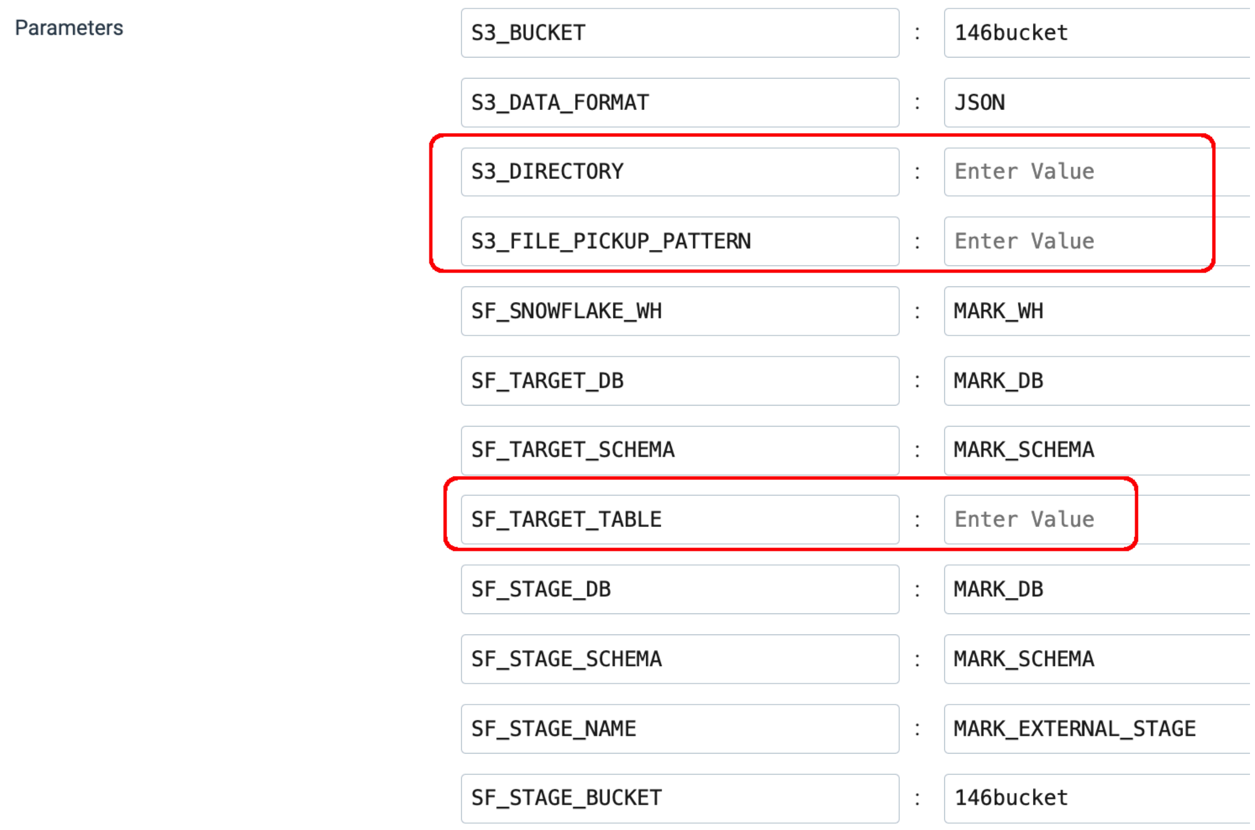
Create the Data Collector Pipeline

I'll create a simple Data Collector Pipeline using the fragments published earlier:



See the project here for descriptions and downloads of the two fragments.

By default, a pipeline with those two fragments will have a parameter list that looks like this:



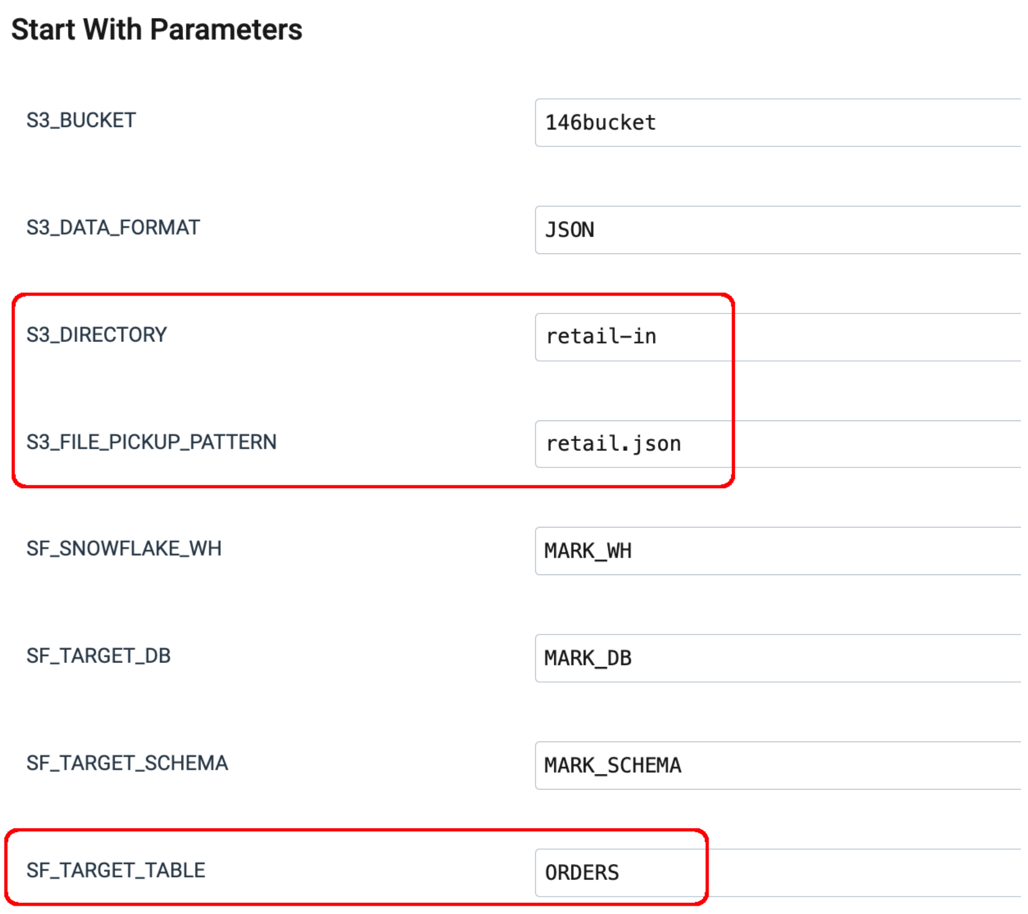
Parameters

S3_BUCKET	:	146bucket
S3_DATA_FORMAT	:	JSON
S3_DIRECTORY	:	Enter Value
S3_FILE_PICKUP_PATTERN	:	Enter Value
SF_SNOWFLAKE_WH	:	MARK_WH
SF_TARGET_DB	:	MARK_DB
SF_TARGET_SCHEMA	:	MARK_SCHEMA
SF_TARGET_TABLE	:	Enter Value
SF_STAGE_DB	:	MARK_DB
SF_STAGE_SCHEMA	:	MARK_SCHEMA
SF_STAGE_NAME	:	MARK_EXTERNAL_STAGE
SF_STAGE_BUCKET	:	146bucket

Set any parameters with default values, and clear the values for any parameters that should be passed in at runtime. For example, I cleared these three parameters: `S3_DIRECTORY`, `S3_FILE_PICKUP_PATTERN`, and `SF_TARGET_TABLE`.

Test the Data Collector Pipeline

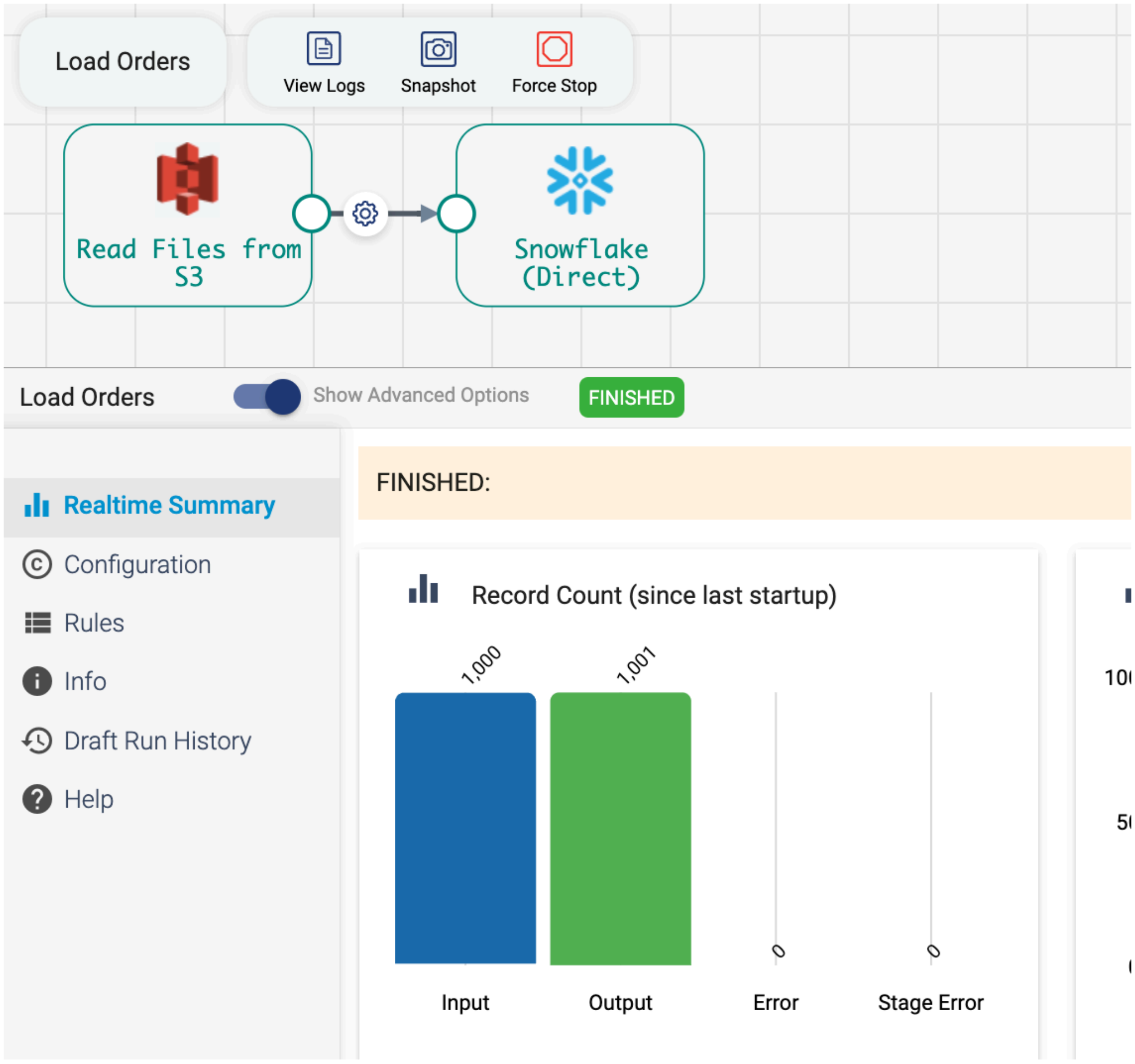
Once again, choose `Draft Run > Start with Parameters`. I provided the three parameter values highlighted here:



Start With Parameters

S3_BUCKET	146bucket
S3_DATA_FORMAT	JSON
S3_DIRECTORY	retail-in
S3_FILE_PICKUP_PATTERN	retail.json
SF_SNOWFLAKE_WH	MARK_WH
SF_TARGET_DB	MARK_DB
SF_TARGET_SCHEMA	MARK_SCHEMA
SF_TARGET_TABLE	ORDERS

Make sure the Data Collector pipeline runs correctly:



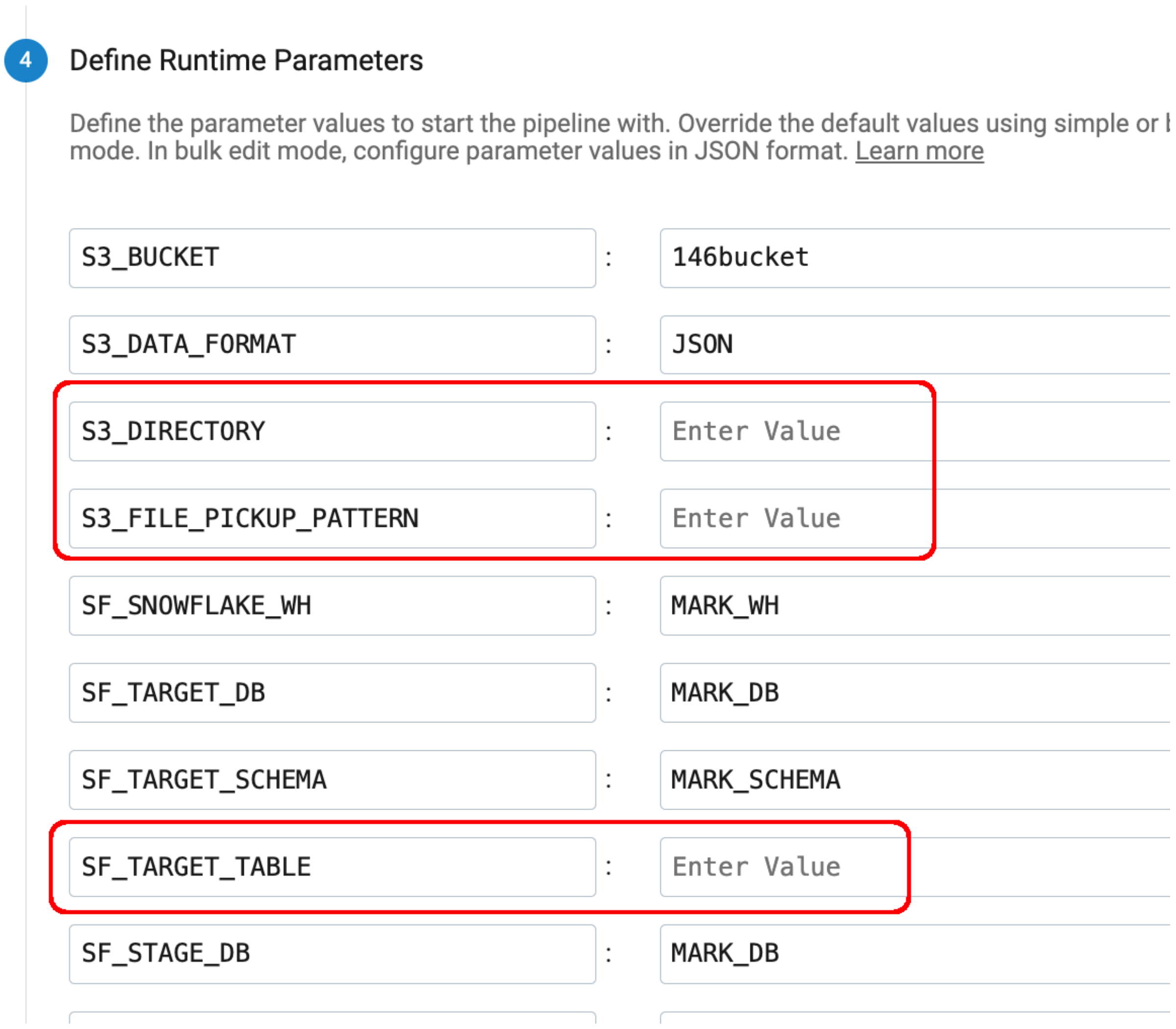
FINISHED:

Record Count (since last startup)

Input	Output	Error	Stage Error
1,000	1,001	0	0

Create a Job for the Data Collector Pipeline

Repeat the same steps as in the previous section to create a Job for the Data Collector pipeline. Once again, leave the non-default runtime parameters blank:



4 Define Runtime Parameters

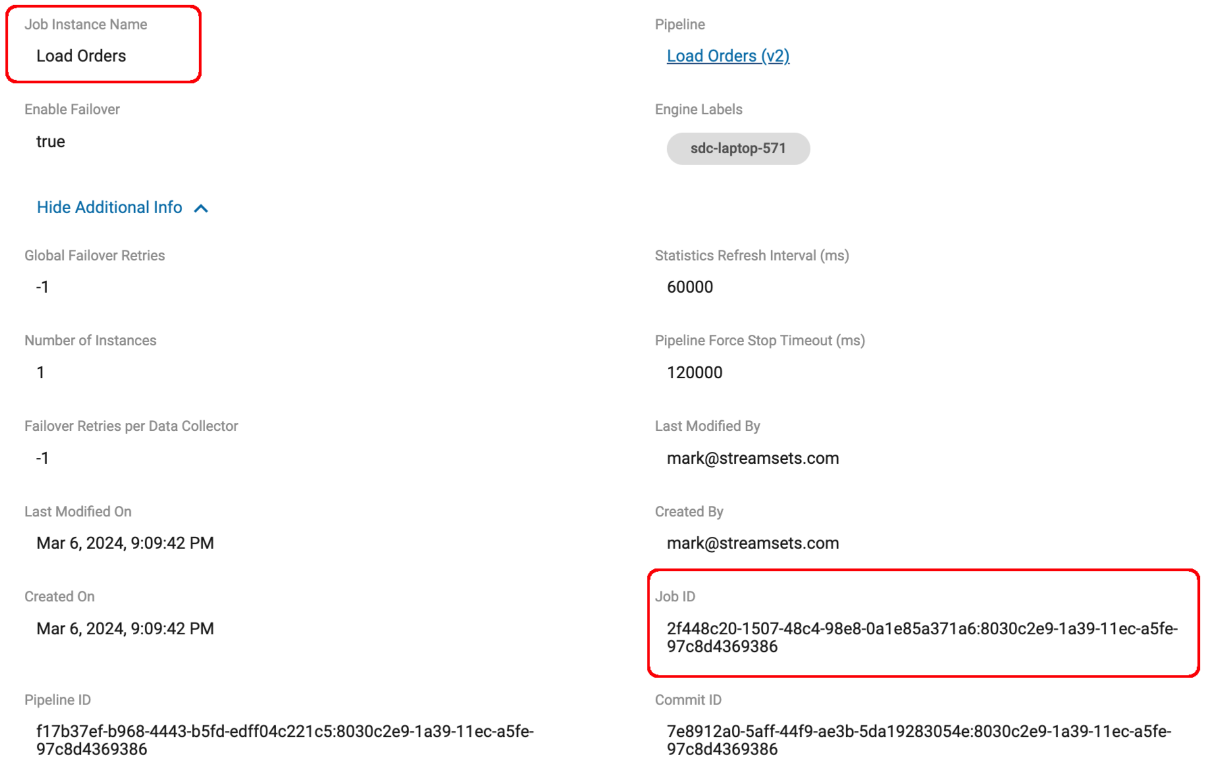
Define the parameter values to start the pipeline with. Override the default values using simple or b mode. In bulk edit mode, configure parameter values in JSON format. Learn more

S3_BUCKET	:	146bucket
S3_DATA_FORMAT	:	JSON
S3_DIRECTORY	:	Enter Value
S3_FILE_PICKUP_PATTERN	:	Enter Value
SF_SNOWFLAKE_WH	:	MARK_WH
SF_TARGET_DB	:	MARK_DB
SF_TARGET_SCHEMA	:	MARK_SCHEMA
SF_TARGET_TABLE	:	Enter Value
SF_STAGE_DB	:	MARK_DB

Click `Save & Next` and then `Exit`

→ The Data Collector Job has been created.

Click on the Data Collector Job and copy its Job ID as we'll need it later:



Job Instance Name	Pipeline
Load Orders	Load Orders (v2)
Enable Failover	Engine Labels
true	sdc-laptop-571

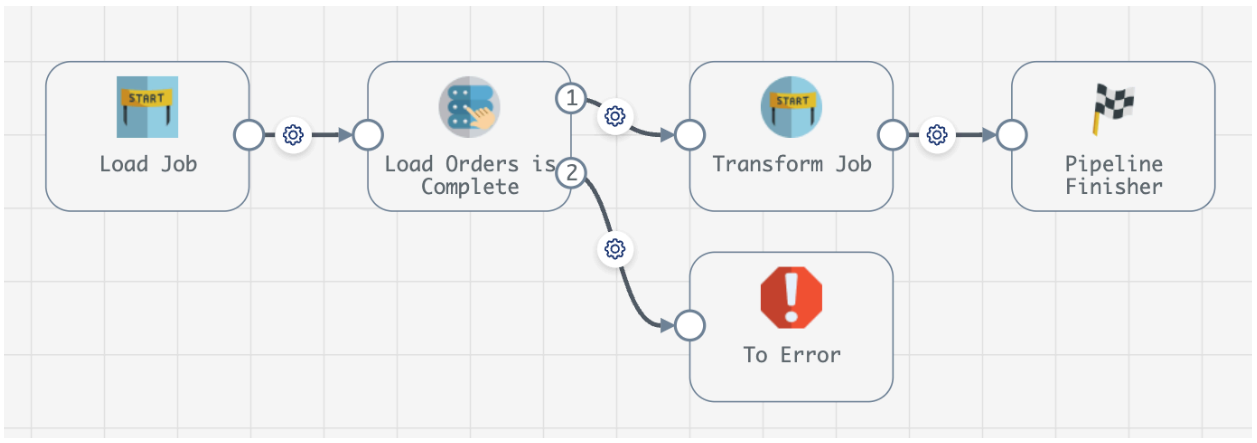
Hide Additional Info

Global Failover Retries	Statistics Refresh Interval (ms)
-1	60000
Number of Instances	Pipeline Force Stop Timeout (ms)
1	120000
Failover Retries per Data Collector	Last Modified By
-1	mark@streamsets.com
Last Modified On	Created By
Mar 6, 2024, 9:09:42 PM	mark@streamsets.com
Created On	Job ID
Mar 6, 2024, 9:09:42 PM	2f448c20-1507-48c4-98e8-0a1e85a371a6:8030c2e9-1a39-11ec-a5fe-97c8d4369386
Pipeline ID	Commit ID
f17b37ef-b968-4443-b5fd-edff04c221c5:8030c2e9-1a39-11ec-a5fe-97c8d4369386	7e8912a0-5aff-44f9-ae3b-5da19283054e:8030c2e9-1a39-11ec-a5fe-97c8d4369386

Create an Orchestration Pipeline

Create a Data Collector Orchestration pipeline to manage the process. You can download this pipeline from here.

The pipeline should look like this:



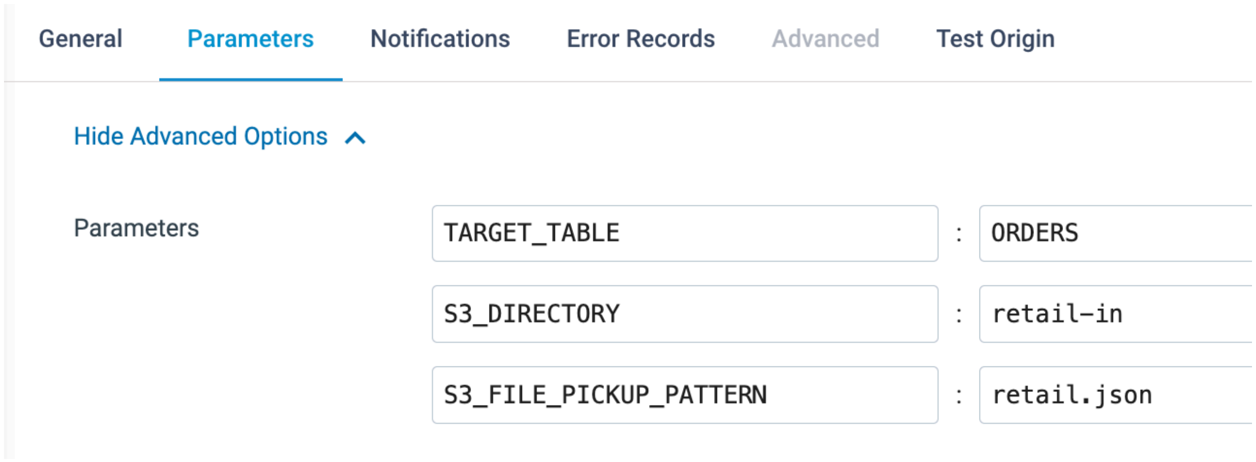
The `Load Job` stage is a Start Jobs Origin that will launch the Data Collector Job and wait for it to complete.

The `Load Orders is Complete` stage is a Stream Selector that tests if the Data Collector Job completed successfully or not.

The `Transform Job` stage is a Start Jobs Processor that launches the Transformer for Snowflake Job and immediately exits.

Additional details about these stages are provided below.

Set the pipeline's parameters. For example, in my environment I used these values:



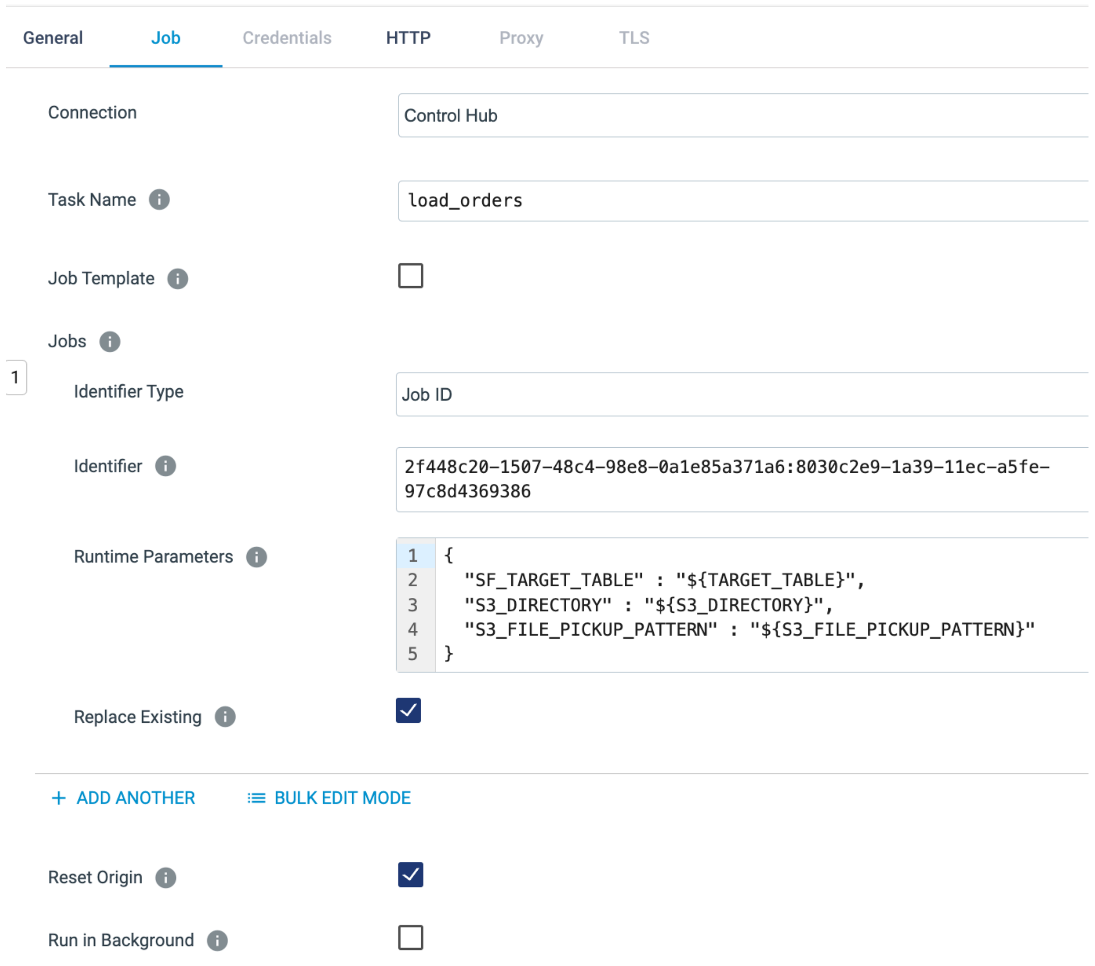
Parameters

TARGET_TABLE	ORDERS
S3_DIRECTORY	retail-in
S3_FILE_PICKUP_PATTERN	retail.json

The `TARGET_TABLE` value will be passed to both Jobs!

The two S3 parameters are the S3 directory and pickup pattern needed for the Data Collector Job.

In the `Load Job` stage, set the Job properties like this:



Connection: Control Hub

Task Name: load_orders

Job Template: ☐

Jobs

1

Identifier Type: Job ID

Identifier: 2f448c20-1507-48c4-98e8-0a1e85a371a6:8030c2e9-1a39-11ec-a5fe-97c8d4369386

Runtime Parameters:

```
{
  "SF_TARGET_TABLE" : "${TARGET_TABLE}",
  "S3_DIRECTORY" : "${S3_DIRECTORY}",
  "S3_FILE_PICKUP_PATTERN" : "${S3_FILE_PICKUP_PATTERN}"
}
```

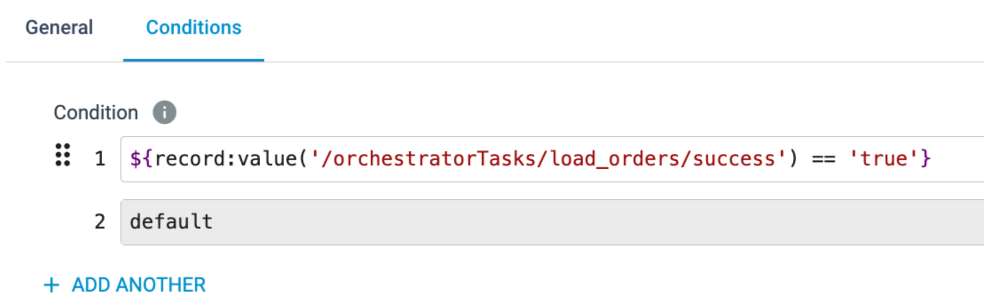
Replace Existing: ✓

Reset Origin: ✓

Run in Background: ☐

- Provide a Control Hub Connection and a user defined task name
- The `Identifier` is the Job ID captured earlier for the Data Collector Job
- `Runtime Parameters` is a JSON dictionary of the three parameter key/value pairs needed for the Data Collector pipeline. Note that the keys are all hard-coded to match what the Data Collector Job requires, and the three values are the parameters set in this orchestration pipeline itself. This is how the orchestration pipeline passes parameter values to a Job.
- Set the `Replace Existing` checkbox to have these parameters replace the empty ones in the Job
- Unset the `Run in Background` checkbox to have this stage block until the Job is complete

The `Load Orders is Complete` Stream Selector is configured like this:

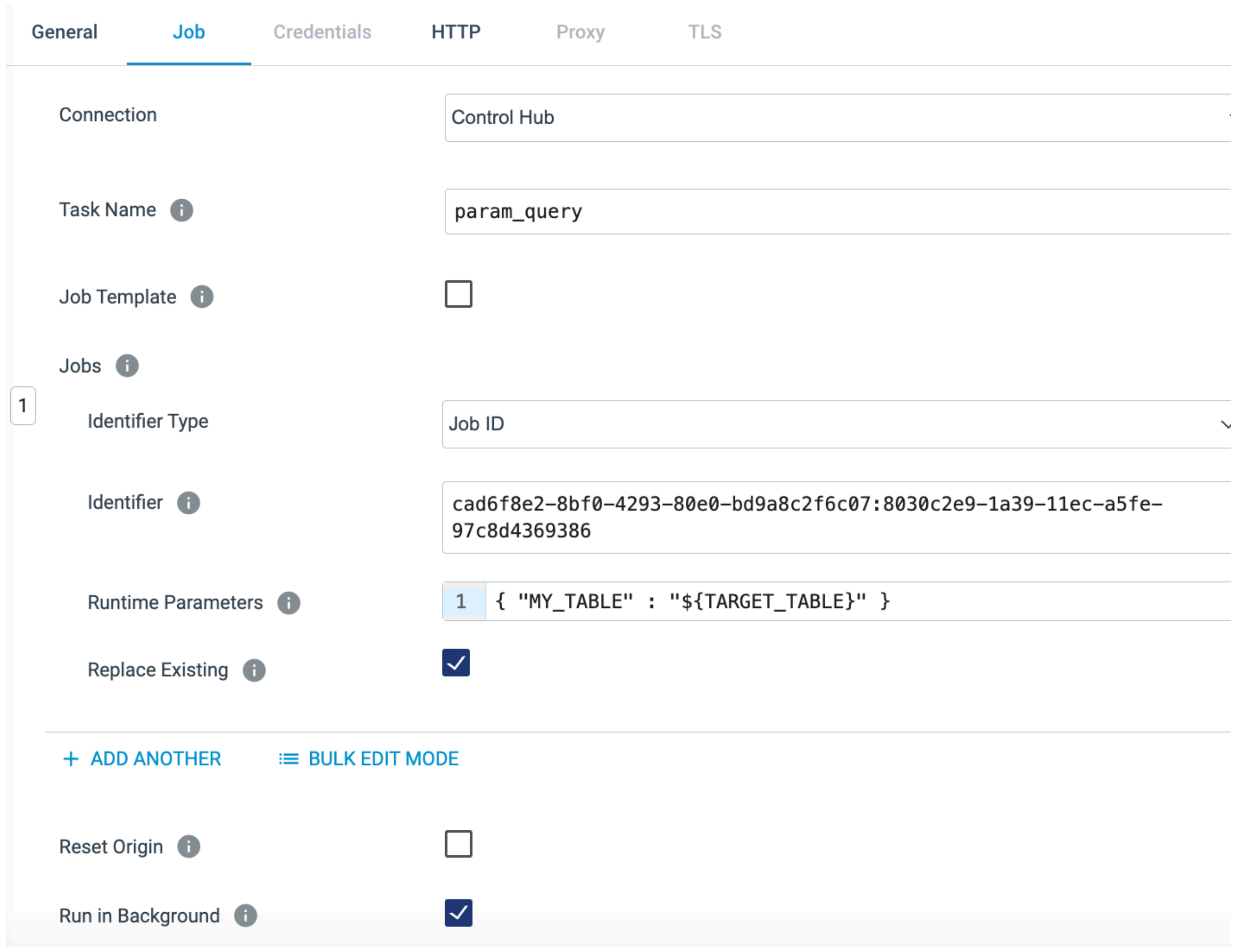


Condition

1 `${record:value('/orchestratorTasks/load_orders/success') == 'true'}`

2 `default`

The `Transform Job` stage is configured like this:



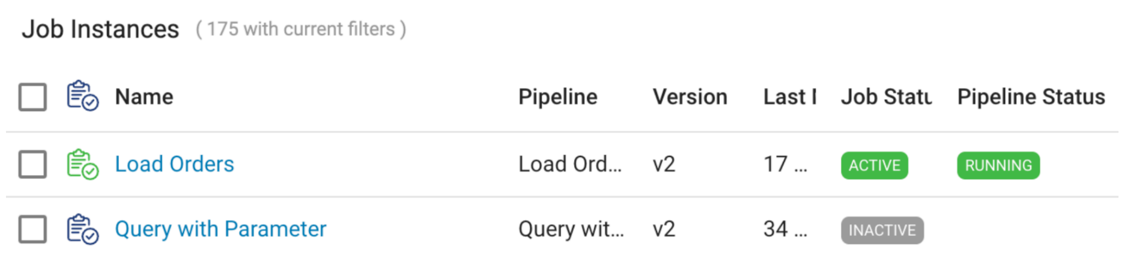
Connection	Control Hub
Task Name	param_query
Job Template	☐
Jobs	
1 Identifier Type	Job ID
Identifier	cad6f8e2-8bf0-4293-80e0-bd9a8c2f6c07:8030c2e9-1a39-11ec-a5fe-97c8d4369386
Runtime Parameters	`{ "MY_TABLE" : "${TARGET_TABLE}" }`
Replace Existing	✓
Reset Origin	☐
Run in Background	✓

Set the Transformer for Snowlake Job ID, Runtime Parameters and the checkboxes as shown.

Once again, note that the `MY_TABLE` parameter is set using a parameter from the Orchestration pipeline.

Run the Orchestration Pipeline

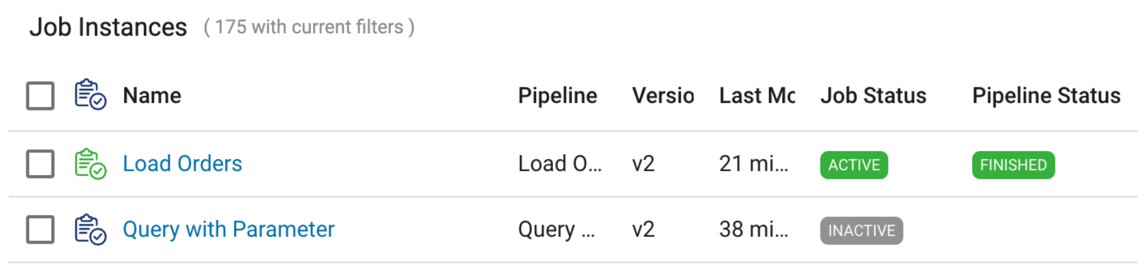
Start a Draft Run of the Orchestration Pipeline and then, in another browser tab, navigate to the Jobs Instances page. You should see the Data Collector Job transition to ACTIVE/RUNNING:



Job Instances (175 with current filters)

☐	Name	Pipeline	Version	Last I	Job Statu	Pipeline Status
☐	Load Orders	Load Ord...	v2	17 ...	ACTIVE	RUNNING
☐	Query with Parameter	Query wit...	v2	34 ...	INACTIVE	

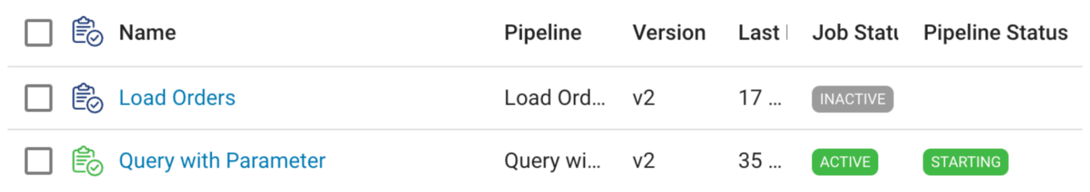
Then that Job should complete:



Job Instances (175 with current filters)

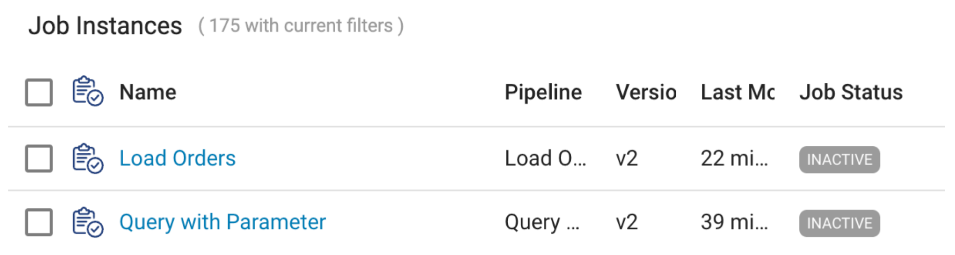
☐	Name	Pipeline	Versio	Last Mc	Job Status	Pipeline Status
☐	Load Orders	Load O...	v2	21 mi...	ACTIVE	FINISHED
☐	Query with Parameter	Query ...	v2	38 mi...	INACTIVE	

The the Transformer for Snowflake Job should start:



☐	Name	Pipeline	Version	Last I	Job Statu	Pipeline Status
☐	Load Orders	Load Ord...	v2	17 ...	INACTIVE	
☐	Query with Parameter	Query wi...	v2	35 ...	ACTIVE	STARTING

And then complete:



Job Instances (175 with current filters)

☐	Name	Pipeline	Versio	Last Mc	Job Status
☐	Load Orders	Load O...	v2	22 mi...	INACTIVE
☐	Query with Parameter	Query ...	v2	39 mi...	INACTIVE

Check each Jobs' history to confirm the run.