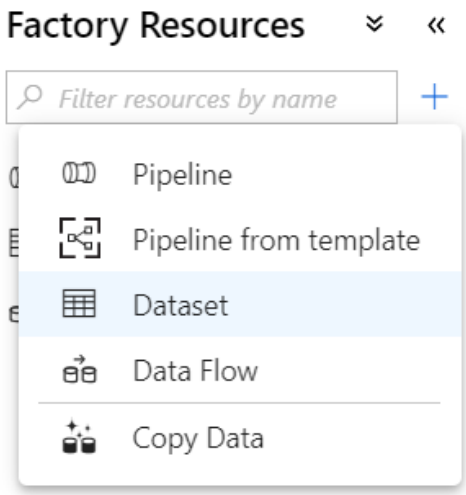
Lab 03 – Making Dynamic Pipelines

Lab 03.A – Configure Dynamic Data Factory Datasets

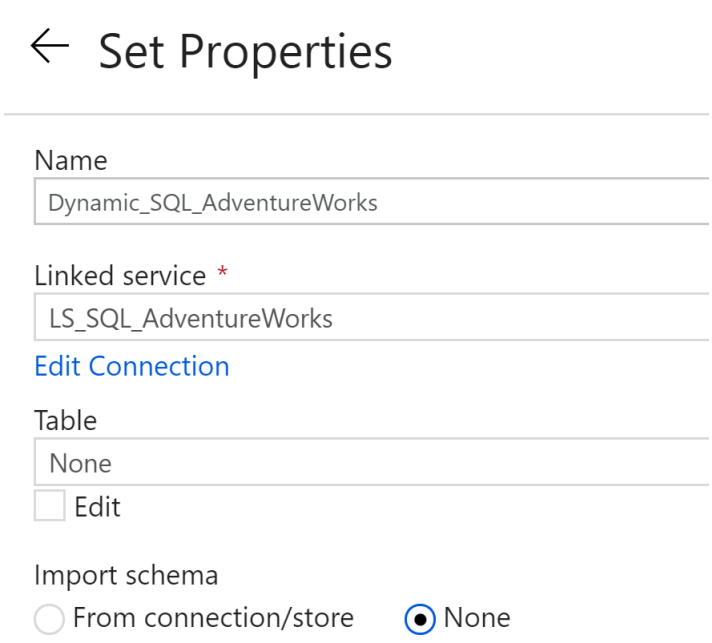
We are going to create 2 datasets and then a pipeline that will copy from one dataset to the other. We can make this pipeline very generic, accepting a parameter to determine which table to load from.

1. Back in our Data Factory workspace, click on the "+" icon next to Data Factory Resources and select "Dataset"



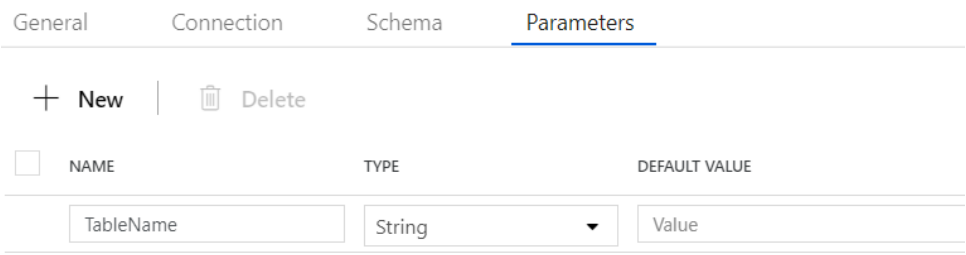
2. Select "Azure SQL Database"

3. This will create a new Dataset. We're going to use this same dataset for any table coming from our local AdventureWorks, so let's call it something generic:



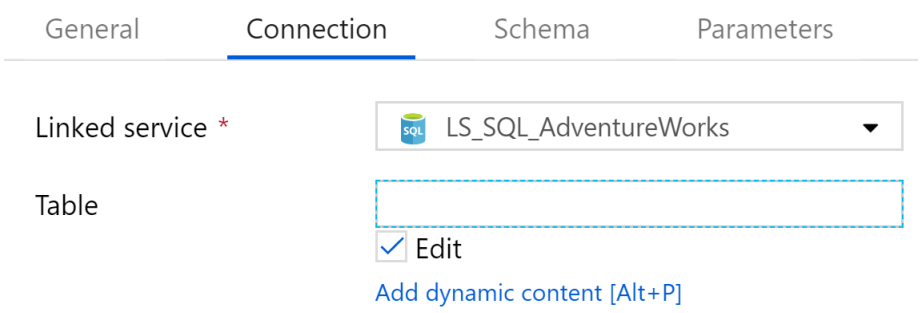
Note that we're not picking a table, as we don't want to tie the dataset to any specific data.

4. Next we need to set up a parameter so that we can tell the Dataset which table to use. Click on "Parameters" tab then create one called "TableName"

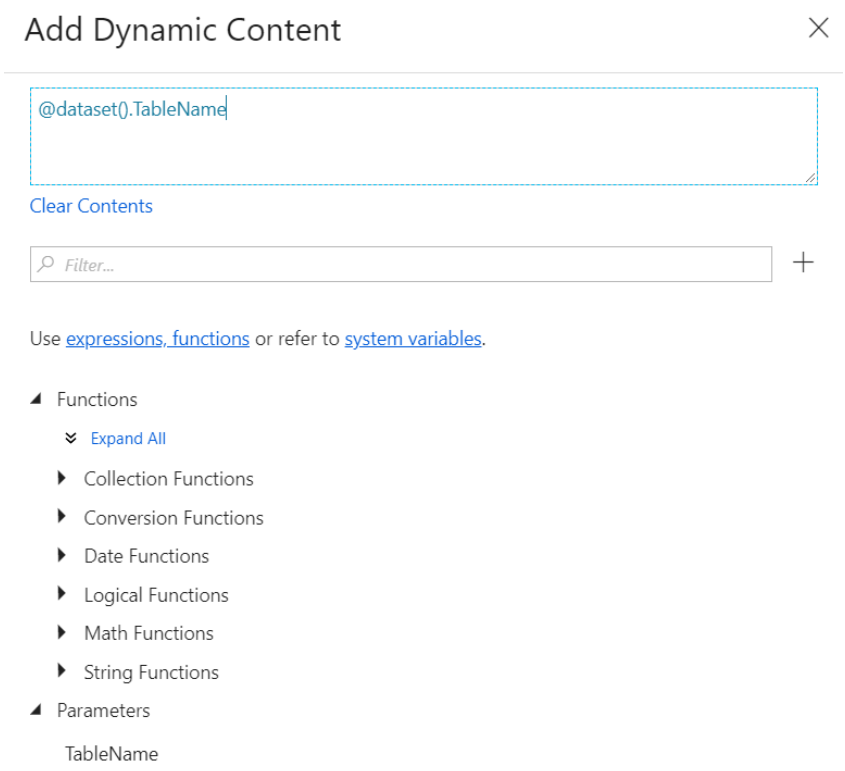


5. Now head back to the "Connections" tab within the Dataset. Select your local LinkedService, then instead of picking a table name, click on the "Edit" checkbox. You'll see an option for dynamic content appear:

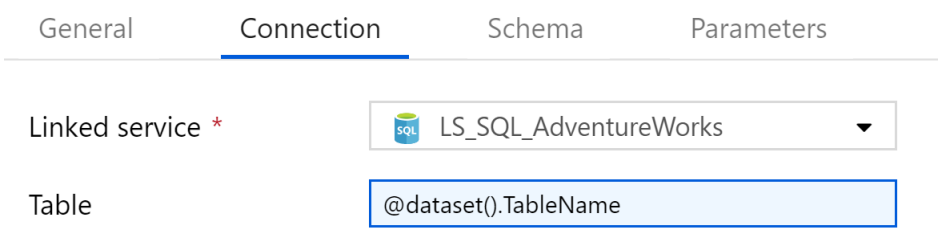
Lab 03 – Making Dynamic Pipelines



6. This will load the Dynamic Content blade, where you can click on "TableName" to add a snippet of code selecting the local parameter for you:

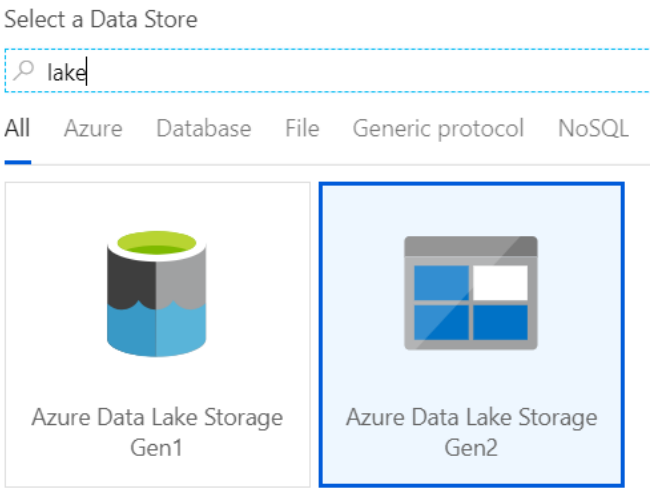


7. Click Finished and your dataset is now ready to go, with a dynamic connection



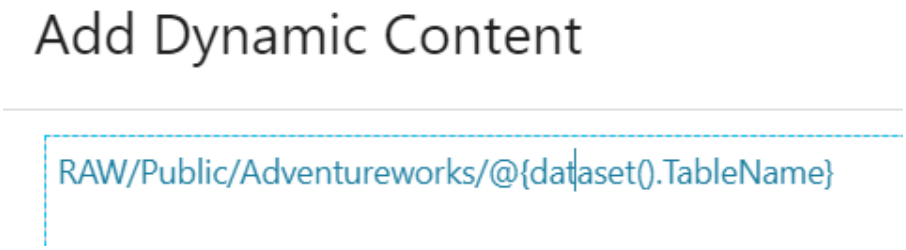
Lab 03 – Making Dynamic Pipelines

8. We can now do the same for a Data Lake Store Gen 2 dataset:



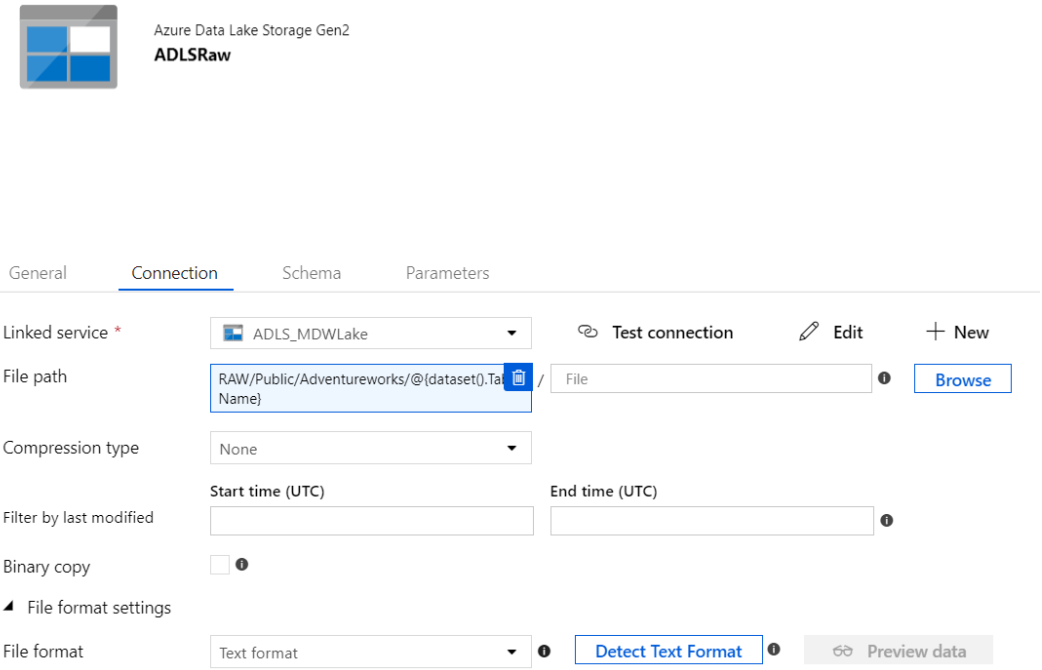
9. Create the dataset with another generic name and create a parameter. Connect to our ADLS LinkedService then we can be a bit clever with the folder names.

   We want each SQL table to go into a different table, so we can use slightly different syntax to concatenate our parameter with a static folder path.



   Leave the file settings as they are and it will create the files as a default CSV that we can work with further in later steps.

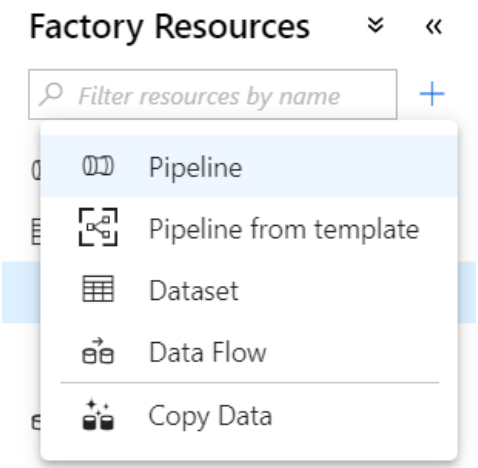
Lab 03 – Making Dynamic Pipelines



Lab 03.B – Build a Dynamic Pipeline

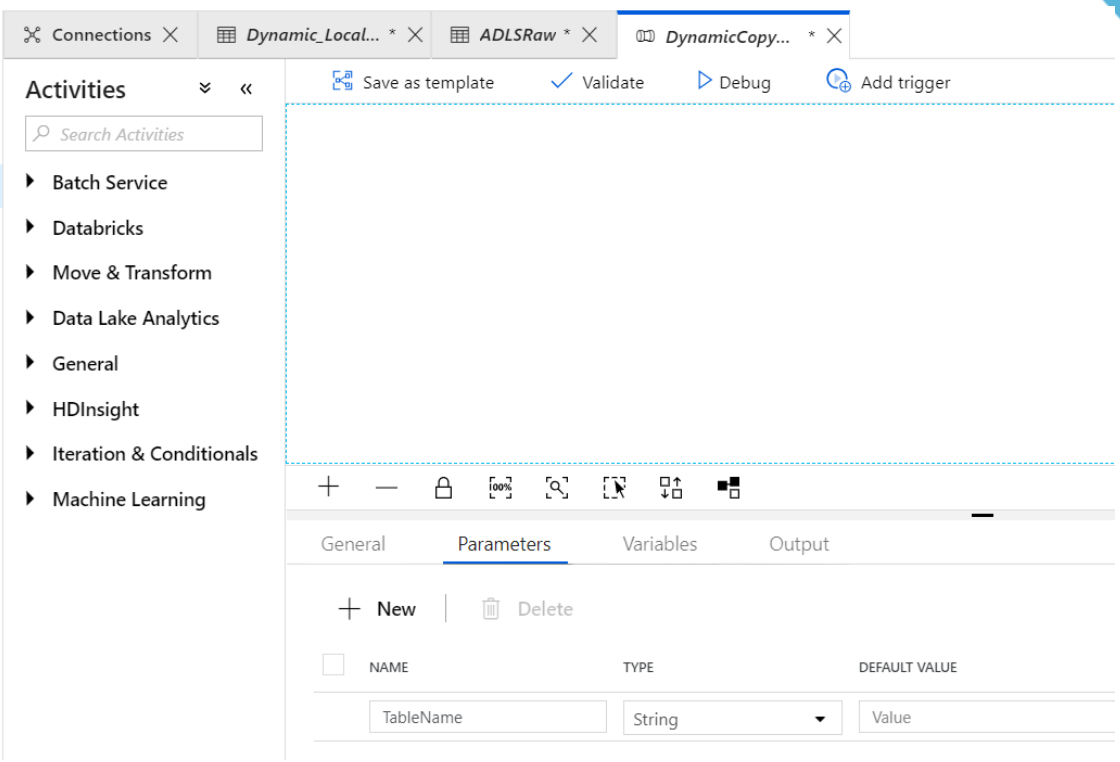
We now have everything we need – generic datasets, our linked services and a whole host of plumbing. We now just need to use it!

1. Let's create a new pipeline, from the same menu where we created our datasets

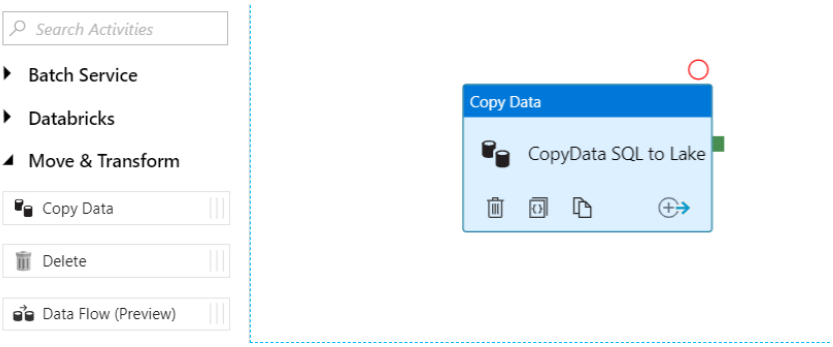


2. Give it a suitable name, and create a parameter called TableName, just as we did with our Dataset

Lab 03 – Making Dynamic Pipelines

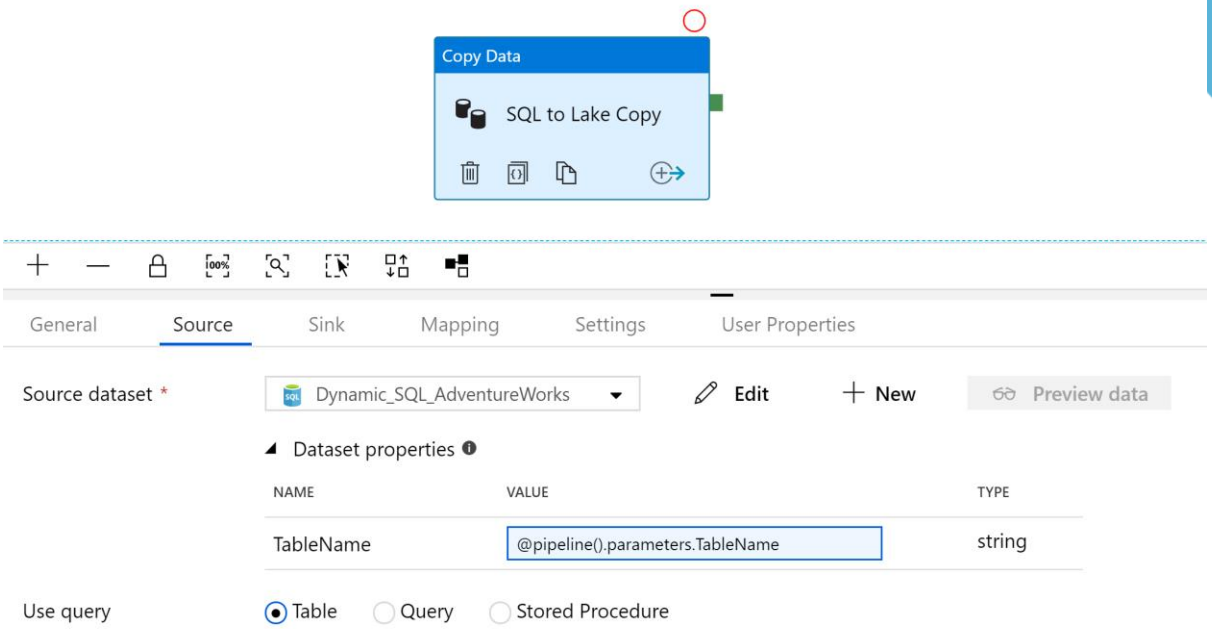


3. We can now drag an activity into our workflow – open the "Move & Transform" category and drag a "Copy Data" activity onto our workspace and give it a sensible name.



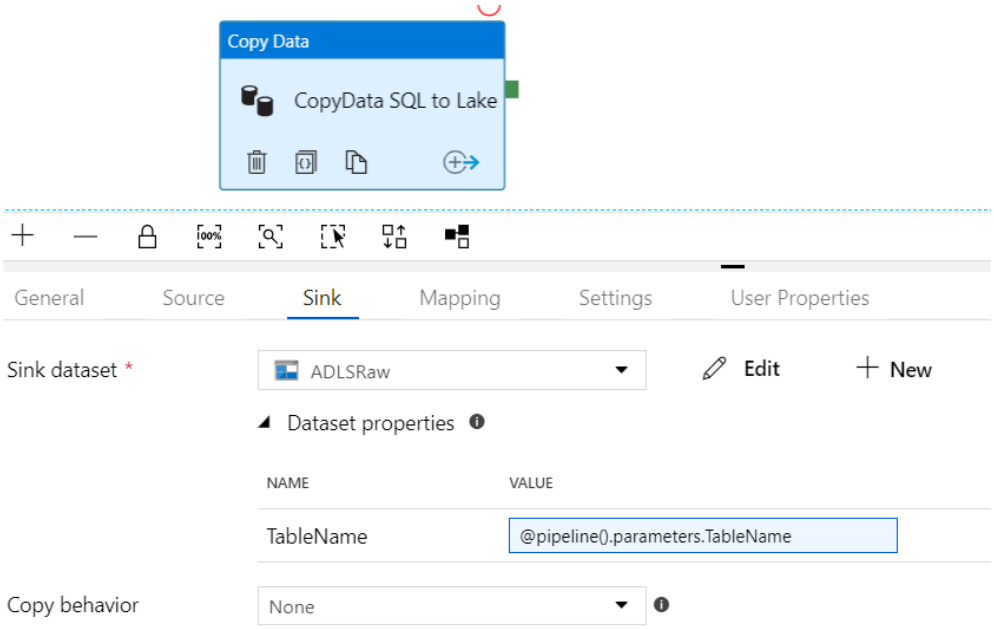
4. With the activity selected, go to the "Source" and pick our dynamic SQL dataset. It will automatically recognise that a Dataset is required for.

Lab 03 – Making Dynamic Pipelines



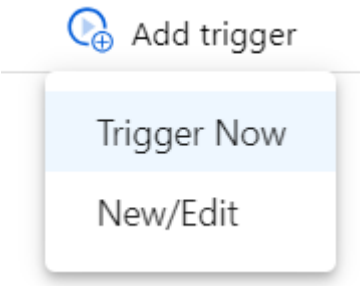
Pop in the parameter and that will now pass any parameter we provide to the pipeline through to that dataset.

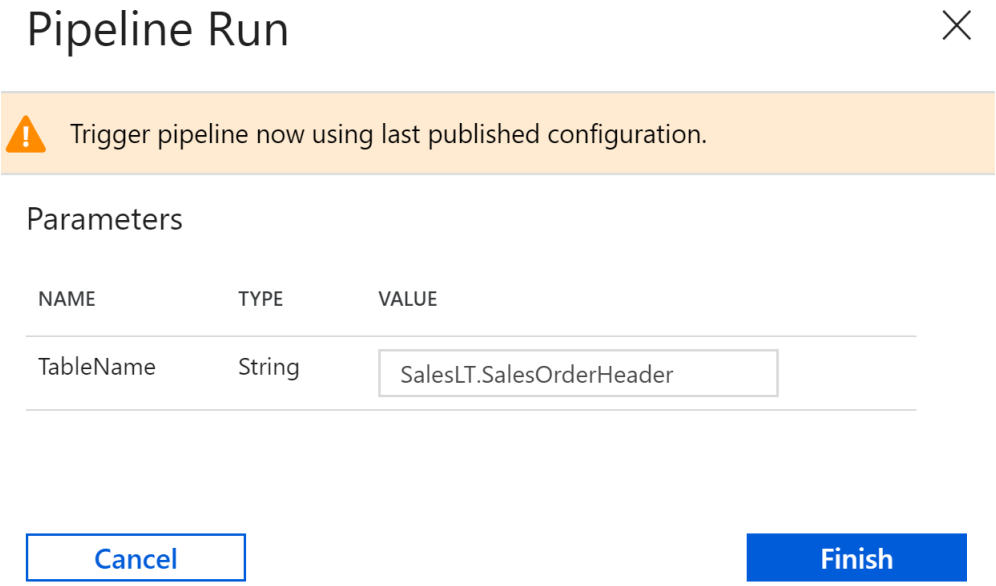
5. We can do the same for the "Sink", this time using our ADLS Dataset:



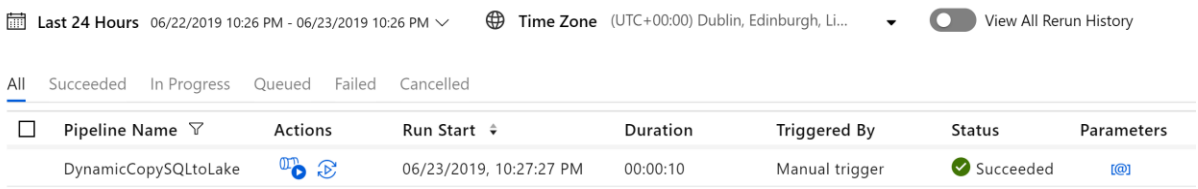
6. Now that this is complete, our pipeline is ready. We can hit the "Publish" button to validate our objects and commit those changes to our data factory instance.

7. Finally, we can click the "Add Trigger" button to perform a one-off run of the Data Factory. This will ask us for a parameter, we can input the name of one of our AdventureWorksDW tables to test it out:



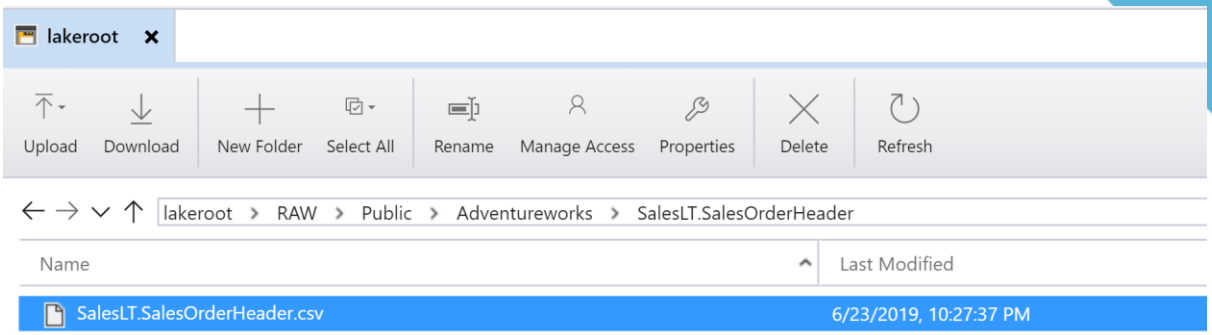


Clicking Finish will commit that request and execute the pipeline. We can view it's progress in the Monitoring tab.



Finally, we can check our lake using Azure Storage Explorer and we will see:

Lab 03 – Making Dynamic Pipelines



Our file has been created, along with the dynamic directory mapping