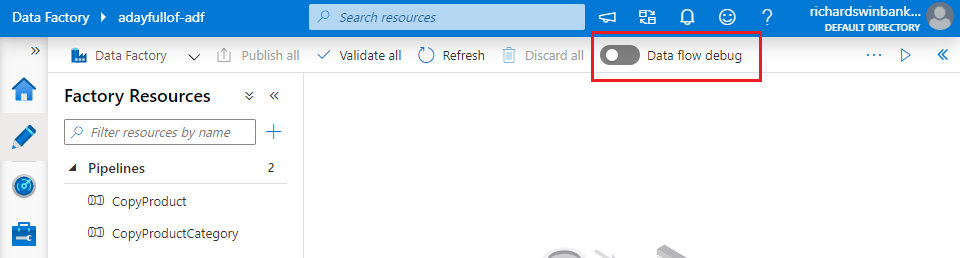
Lab 4 – Build a Mapping Data Flow

In this lab you will use Azure Data Factory’s Mapping Data Flows feature to implement a familiar data warehousing process: maintaining a dimension.

# Lab 4.1 – Enable data flow debugging

Mapping data flows are executed – and debugged – on Apache Spark clusters. A cluster takes several minutes to warm up, so start this lab by switching data flow debug on for your ADF UX session.

1. In the ADF UX, toggle the “Data flow debug” slider to “On”.



1. When the ADF UX prompts you for confirmation, click “OK”.

While the debug cluster is warming up, continue with Labs 4.2 & 4.3.

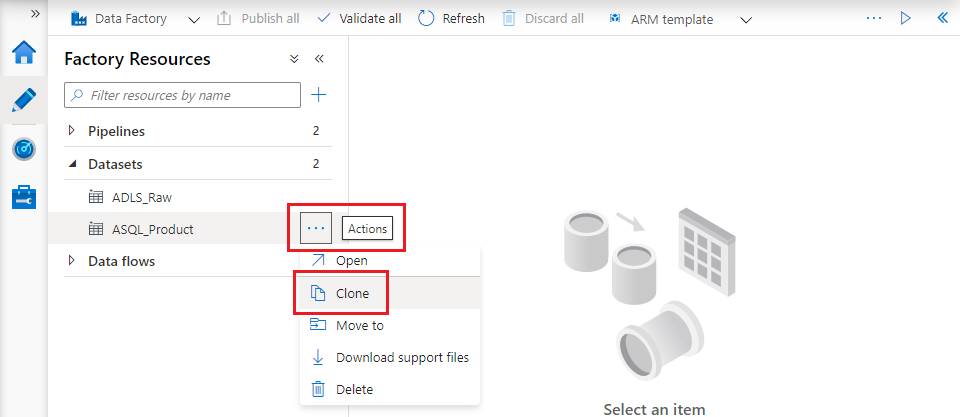
# Lab 4.2 – Copy source data to the data lake

The product dimension will be built using data from three source tables:

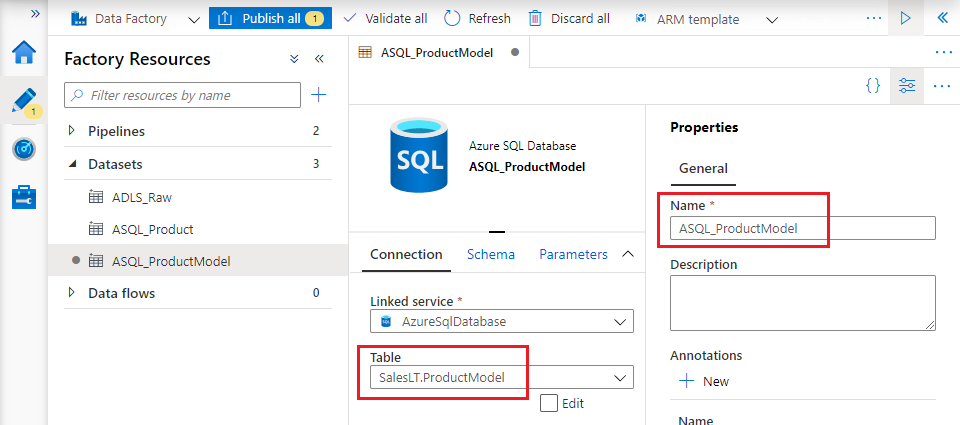
* [SalesLT].[Product]
* [SalesLT].[ProductCategory]
* [SalesLT].[ProductModel]

In Labs 2 & 3 you imported data from the first two tables into the data lake. Import data for [SalesLT].[ProductModel] now.

1. Create a copy of the “ASQL\_Product” dataset by clicking its ellipsis “Actions” button in the “Factory Resources” list, then selecting “Clone”.



1. The cloned dataset opens automatically with its “Properties” pane displayed. Change its name to “ASQL\_ProductModel”, then on the “Connections” tab choose the corresponding [AdventureWorks] table.

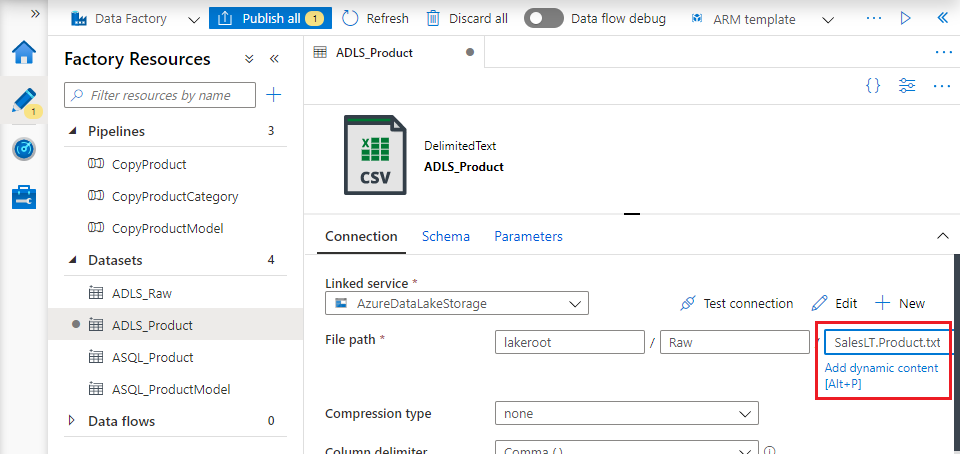


1. Create a new pipeline in the same way as Lab 2.3, using the new “ASQL\_ProductModel” dataset as source and your Azure Data Lake Storage dataset as sink. Save your changes.
2. Run the pipeline in debug mode and verify that file “SalesLT.ProductModel.txt” has been created in the “lakeroot” container’s “Raw” folder.

# Lab 4.3 – Create ADLS datasets

To use data from the three files now created in “/lakeroot/Raw”, you need datasets to represent them.

1. Create a new dataset by cloning your existing Azure Data Lake Storage dataset. Name it “ADLS\_Product” and add file name “SalesLT.Product.txt” to the “File path” specified on the dataset’s “Connection” tab.



1. Repeat step 1 to create two further datasets:
   * “ASQL\_ProductCategory”, to represent file “/lakeroot/Raw/SalesLT.ProductCategory.txt”
   * “ASQL\_ProductModel”, to represent file “/lakeroot/Raw/SalesLT.ProductModel.txt”

# Lab 4.4 – Combine ADLS datasets

The product dimension combines product, model and category information to support different aggregations of facts that have a product attribute. This SQL query combines this information within the [AdventureWorks] database:

SELECT

p.ProductID

, p.[Name] AS Product

, pm.[Name] AS ProductModel

, pc.[Name] AS ProductCategory

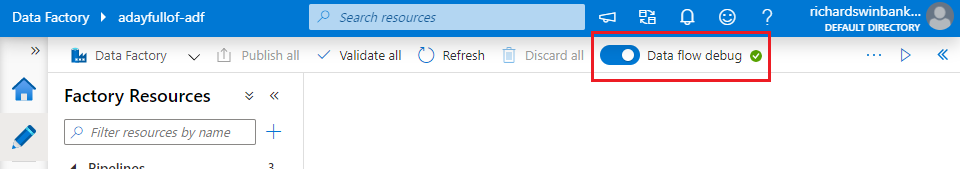
FROM SalesLT.Product p

INNER JOIN SalesLT.ProductModel pm ON pm.ProductModelID = p.ProductModelID

INNER JOIN SalesLT.ProductCategory pc ON pc.ProductCategoryID = p.ProductCategoryID

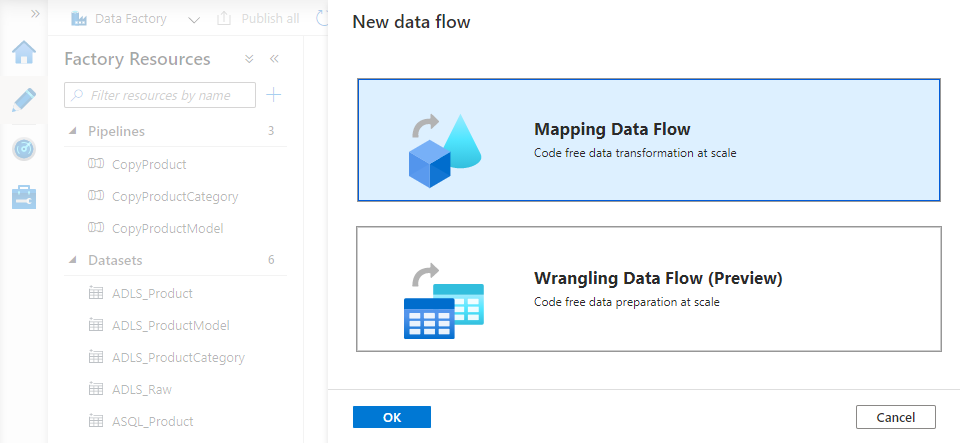
In this section you will use a Mapping Data Flow to reproduce this effect in Azure Data Factory.

1. Check that the debug cluster has finished warming up. When the cluster is available, a tick mark in a green circle appears to the right of the “Data flow debug” slider.

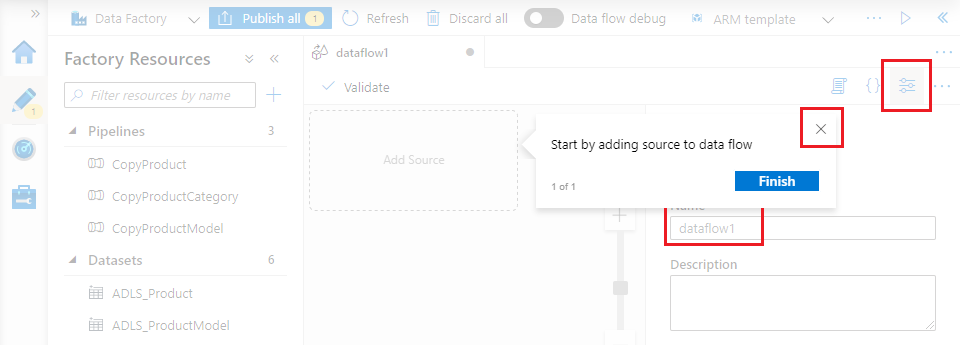


If the cluster is not ready yet, wait for it to finish warming up.

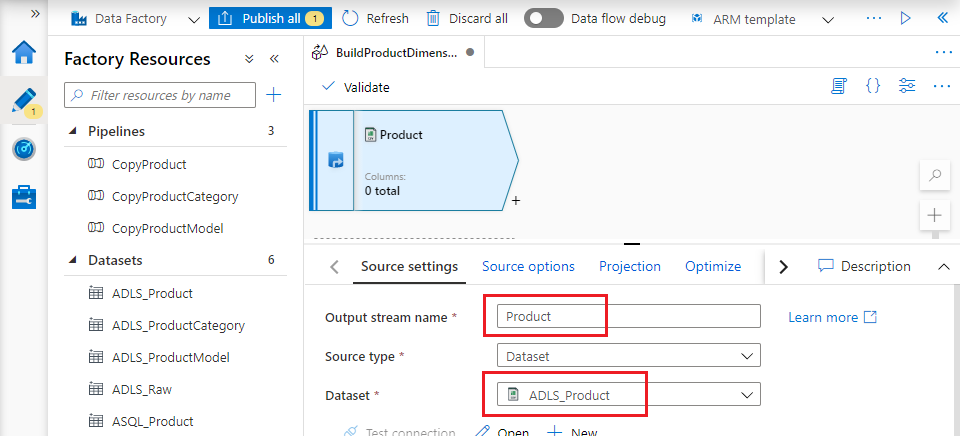
1. In the “Factory resources” sidebar, click the “+” button to the right of “Filter resources by name”, then choose “Data flow”.
2. On the “New data flow” blade, select the “Mapping Data Flow” tile. Click “OK”.



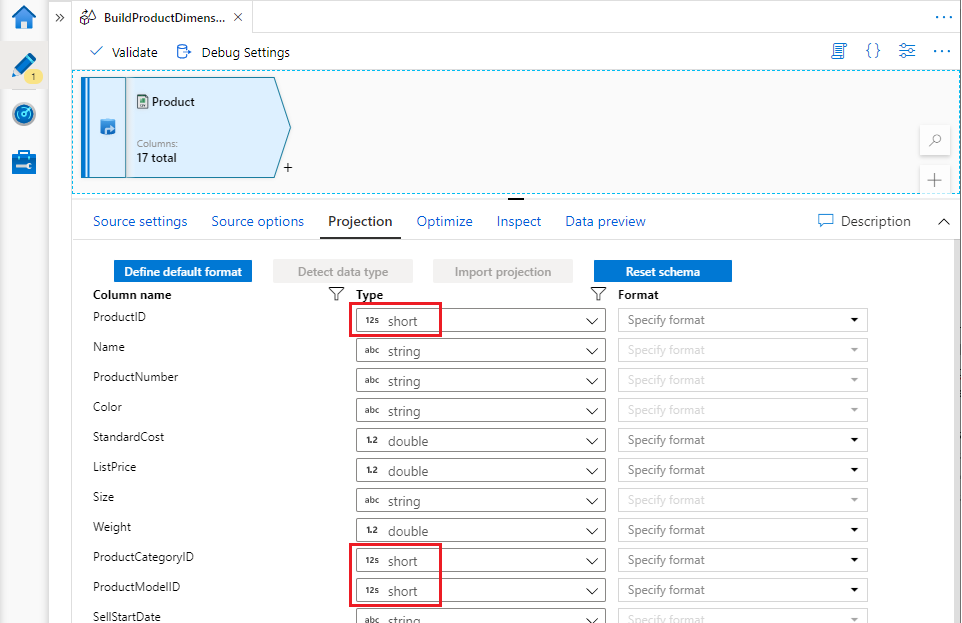
1. The data flow canvas opens, displaying the callout “Start by adding source to data flow”. Dismiss the callout using its close button, then replace the data flow default name (“dataflow1”) with something more descriptive. Use the “Properties” slider button to close the data flow properties blade.



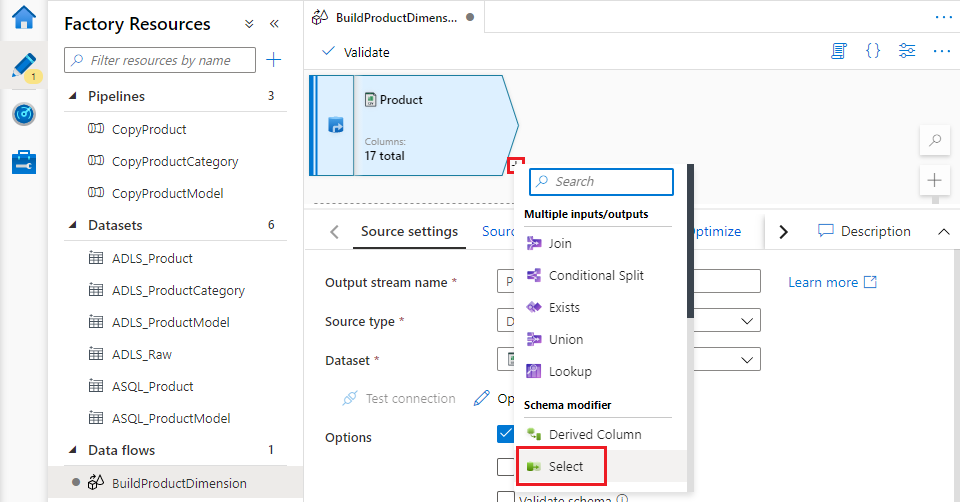
1. Click the “Add source” tile on the data flow canvas and close the callout that appears. On the source transformation’s **Source settings** tab, change its “Output stream name” to “Product” and select the corresponding “ADLS\_Product” dataset.



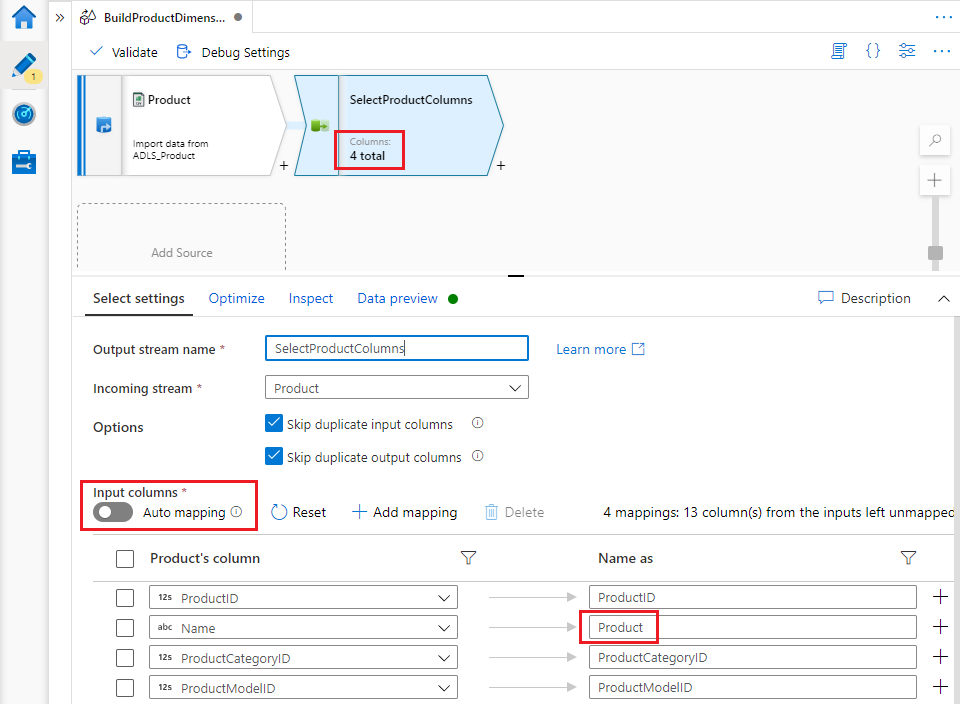
1. On the **Projection** tab, click “Import projection” to import the source file’s schema. Check carefully that the type of the three fields “ProductID”, “ProductCategoryID” and “ProductModelID” has correctly been inferred as “short” – if this is not the case, use the “Type” dropdown to make the necessary correction(s).



1. The source transformation provides a stream of rows for consumption by downstream transformations. To add a transformation to consume the source stream, click on the small “+” button to the bottom right of the source transformation. Choose the “Select” transformation from the popup menu of available options.

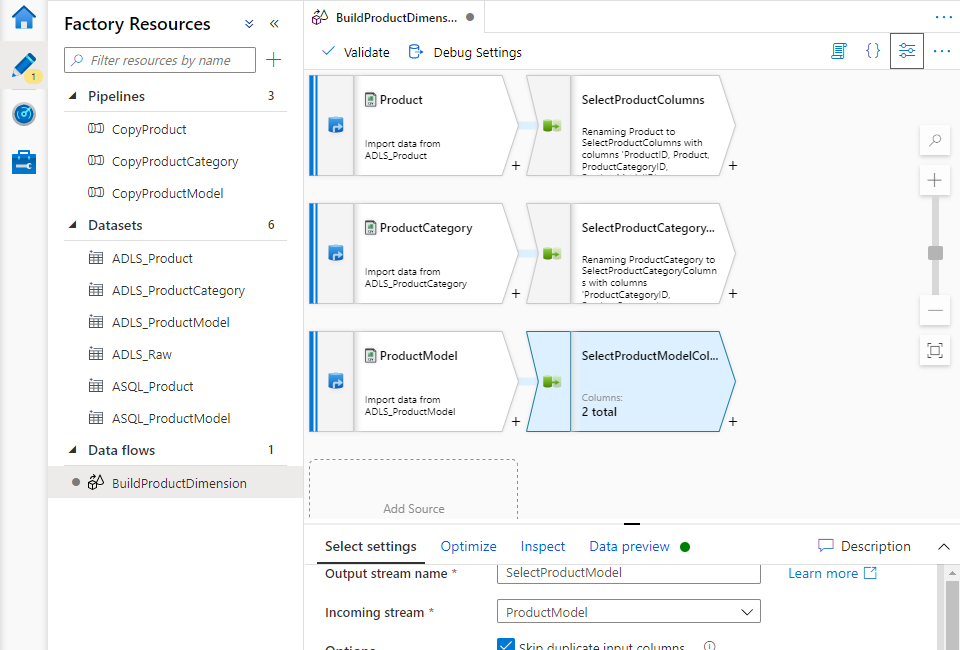


1. On the Select transformation’s **Select settings** tab, change its “Output stream name” to “SelectProductColumns”, then scroll down to the “Input columns” section.



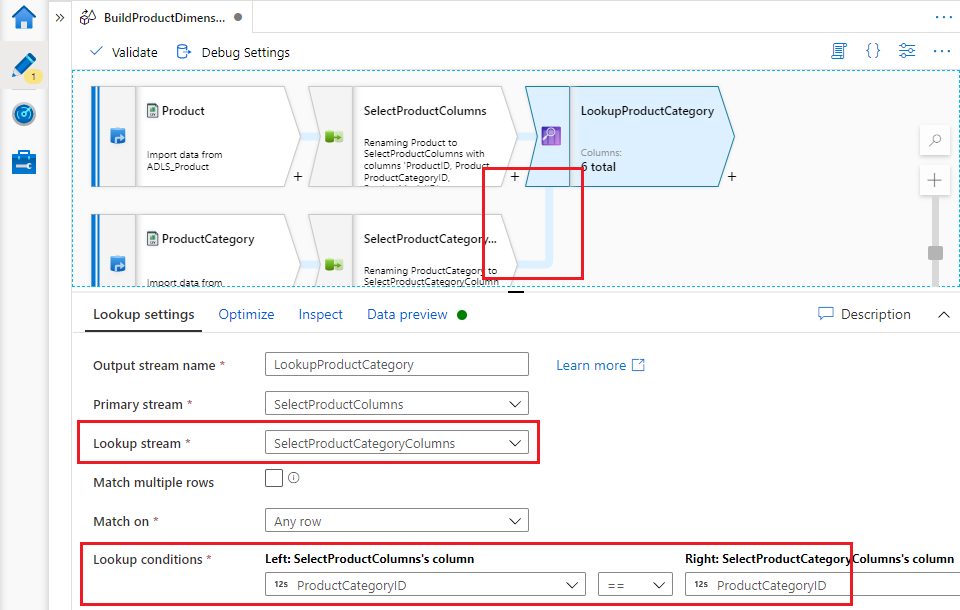
The Select transformation enables you to rename, reorder or remove columns from a stream. Disable the “Auto mapping” setting to enable you to see the transformation’s column mappings, then remove all columns except “ProductID”, “Name”, “ProductCategoryID” and “ProductModelID”. Rename the “Name” column by setting its “Name as” value to “Product”.

1. Repeat steps 5-7 for the “ADLS\_ProductCategory” dataset:
   * Add a source (using the “Add Source” tile displayed on the data flow canvas beneath the Product source transformation)
   * Set its dataset to “ADLS\_ProductCategory”, import the file’s schema and check that ProductCategoryID is of type “short”
   * Add a select transformation, rename it and disable auto-mapping
   * Remove all columns except “ProductCategoryID” and “Name”. Rename the “Name” field to “ProductCategory”.
2. Repeat steps 5-7 using the “ADLS\_ProductModel” dataset, checking that column “ProductModelID” is of type “short”. Use a Select transformation to emove all columns except “ProductModelID” and “Name”. Rename “Name” to “ProductModel”. You now have three parallel streams, loading and modifying data from the three source files.

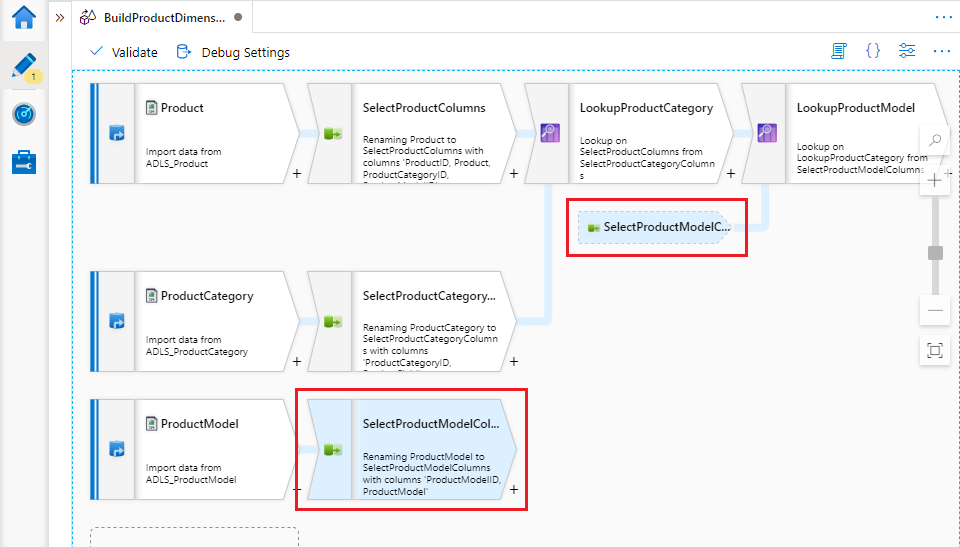


1. You can combine data into the Product stream from the other two streams using the Mapping Data Flow “Lookup” transformation. Click the small “+” button below the product column stream’s Select transformation, then select “Lookup” from the popup menu.
2. On the **Lookup settings** tab, name the transformation then set “Lookup stream” to use ProductCategory columns. (You can choose from any of the other five previous transformations, so take care to pick the ProductCategory stream’s Select transformation, and not the earlier Source for the stream).

“Lookup conditions” specifies the lookup fields from each transformation and the operator to compare them – choose the streams’ respective ProductCategoryId fields. Notice that the lookup relationship also appears on the data flow canvas.

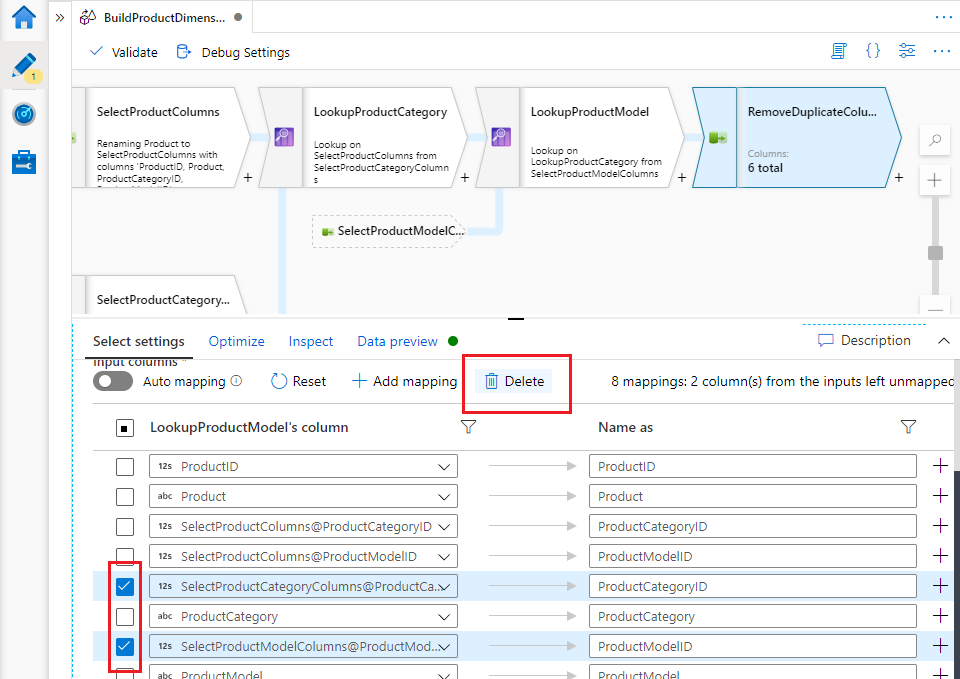


1. Add a second “Lookup” activity, also on the Product stream, this time performing a lookup against the ProductModel stream based on matching ProductModelId. This time the canvas displays a “reference node” instead of showing a direct link between the two transformations. This is just for readability – if you hover over the reference node or the transformation it refers to, both light up in blue to indicate that they mean the same thing.

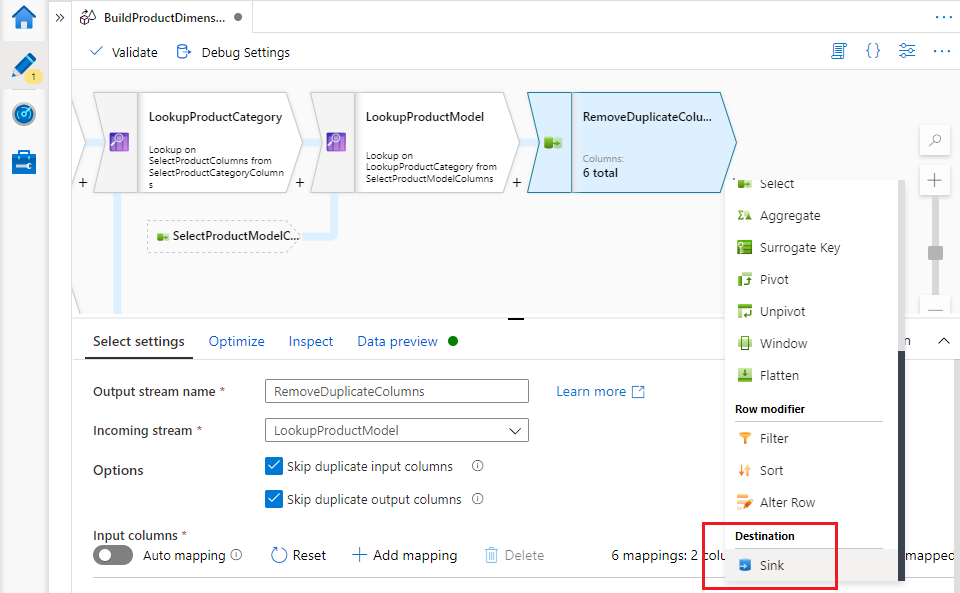


1. Open the new Lookup transformation’s “Inspect” tab to view the set of columns present in the combined stream – notice it includes two copies of each of the join fields, one from each stream participating in the lookup. Add a final Select transformation to clean this up.

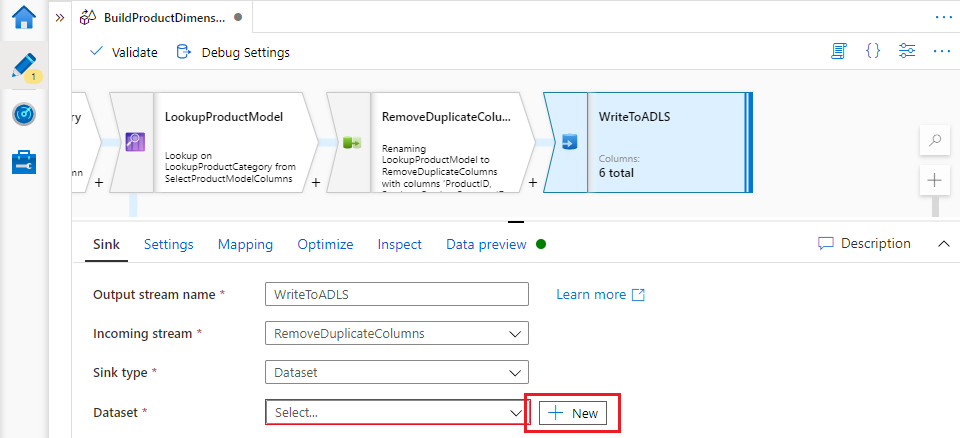
The Select transformation’s input columns indicate the source for each of the duplicated column names by prefixing it with the source transformation name. Delete one duplicate column from each pair.



1. Finally, write the transformed dimension data back to the data lake using a “Sink” transformation. Add the transformation in the usual way, using the small “+” button following the Select transformation that removes duplicate columns. Sink is at the bottom of the list on the popup menu.



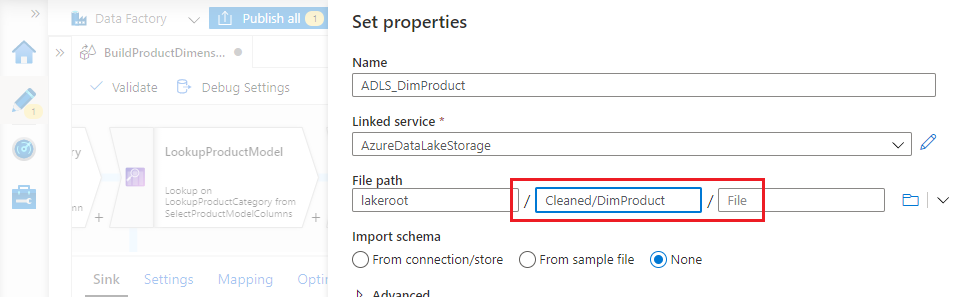
You haven’t yet created a dataset to use as the data flow sink, but you can do so from the Sink transformation by clicking the “+ New” button. This opens the “New dataset” blade familiar from earlier labs.



1. Select data store type “Azure Data Lake Storage Gen2” and select the “Parquet” file format. Parquet is a column-oriented, highly-compressible file format, offering significant performance benefits in data lake stores.

Choose your data lake linked service, then specify a file location. I’m writing the dimension into the “Cleaned” folder of my “lakeroot” container, to reflect the fact that this dataset has

passed beyond the raw state of its source files. Parquet is a multi-file storage format, so the dataset will not accept a file name – instead I’ve specified folder path “Cleaned/DimProduct”, so that the dimension’s Parquet files are written into a directory with a descriptive name.



Click “OK” to create your dataset, then save/publish your changes.

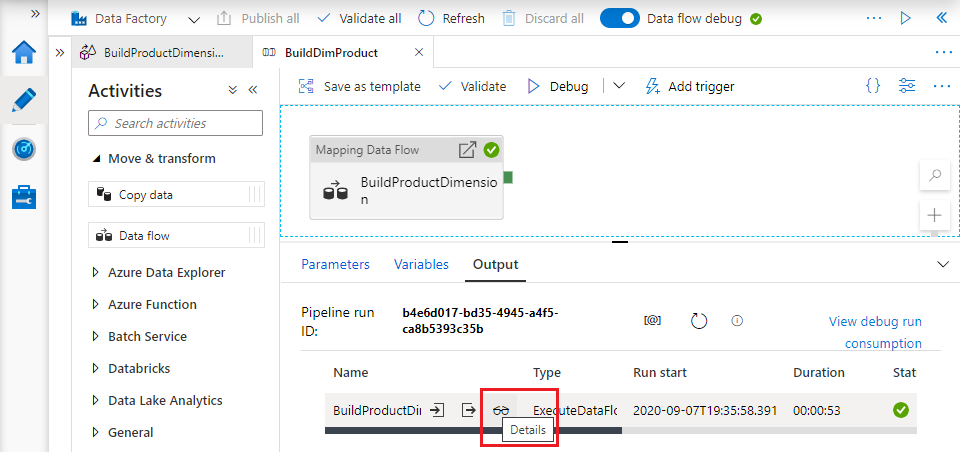
# Lab 4.5 – Run the Mapping Data Flow

Mapping data flows run within an ADF pipeline. To run your data flow, create a pipeline for it.

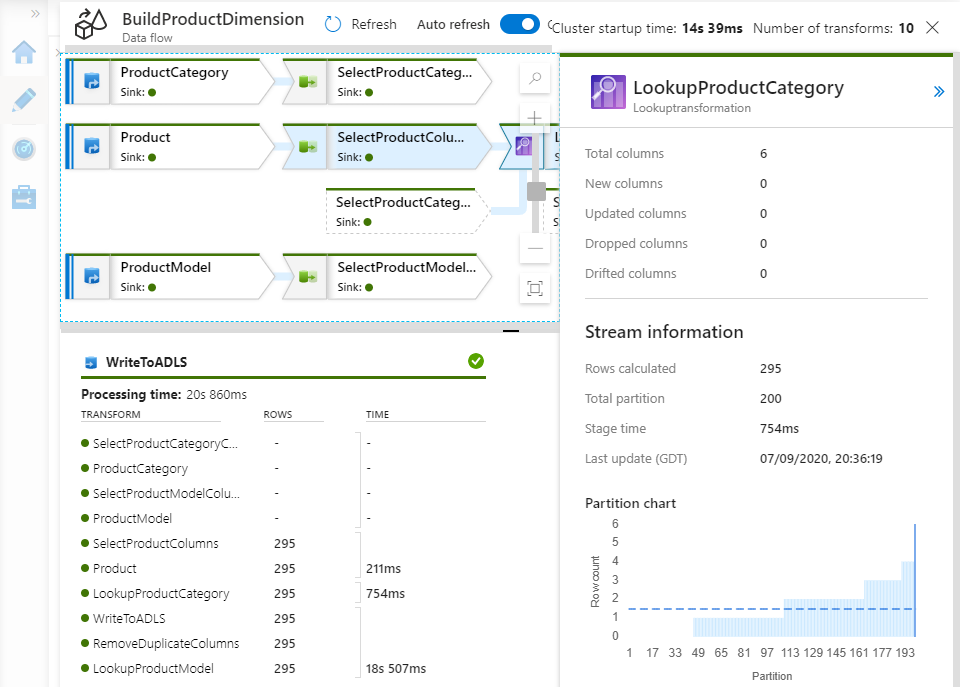
1. Create a new ADF pipeline.
2. Expand the “Move & transform” group in the activity toolbox, then drag a “Data flow” activity onto the pipeline canvas. When prompted, select “Use existing data flow” and select your new data flow from the “Existing data flow” dropdown. Click OK.
3. Click “Debug” to run the pipeline in debugging mode. A warm data flow debug cluster is required to run pipelines containing data flows, just as when you are developing them.

A warm cluster is also required to run published pipelines – you can publish and trigger your data flow now if you wish, but be prepared for a few minutes’ delay while a cluster spins up in the published environment. This may feel cumbersome for a workload the size of the product dimension – in the real world, mapping data flows are designed to support workloads that are considerably more demanding.

1. When the pipeline has finished running, a row of data appears in its “Output” pane for the Data flow activity. Hover over the activity’s name to reveal the “Details” button (“glasses” icon).



1. Click on the button to view more detailed data flow performance information – a visualisation of the data flow is displayed, on which you can inspect individual transformation details.

****

This page contains a wealth of information – highlight different transformations to see the number of rows processed, how long that stage lasted and how the data was partitioned. In this example, the data was very small and so we could perform everything on a single box. For larger examples, we can configure how datasets are distributed to optimise spark executor partitioning, which is very powerful indeed!

You’ve now got the basics for creating a Data Factory data flow, but there’s a lot more to learn! Try out some of the other transformation types and, when you’re ready, try using the DerivedColumn transformation to see the large number of functions available!