

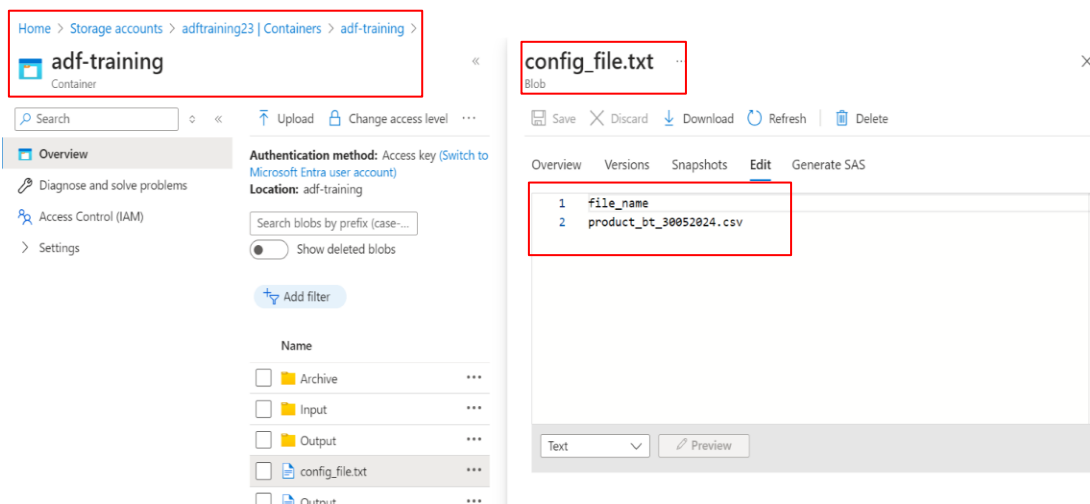
## Formal Documentation of Azure Data Factory Pipeline – Training.

**Use case:** Transfer the data from source to target based on the file name specified in the config file.

### Steps:

#### 1. Creating a config file:

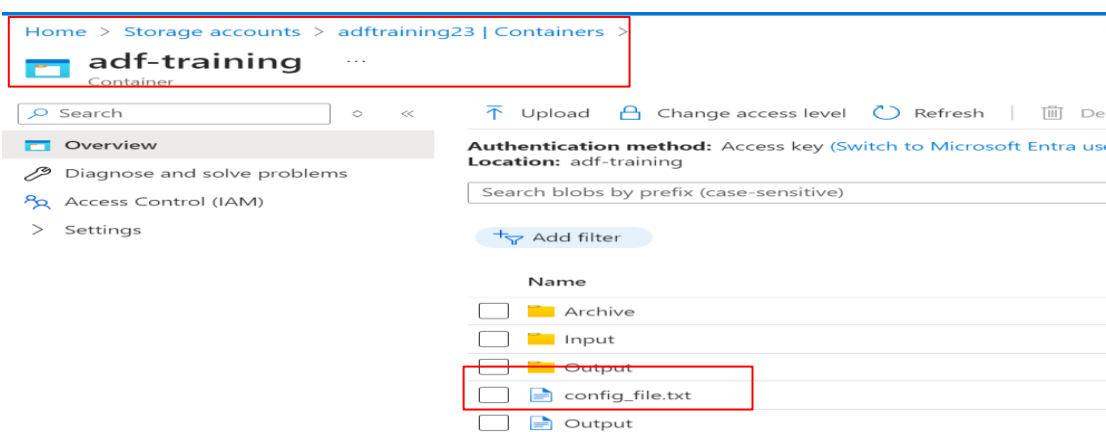
- Created a config file via Notepad, specifying the file name from which I want to copy the data to the target (refer image 1).
- Saved the file as config\_file.txt.



-Image 1

#### 2. Loading the file in the container:

- Uploaded a config file named config\_file.txt to a container named adf-training which was previously created in the Azure storage account (refer image 1)
- Note: Not in the input or output folder but in the container along with the input and output folder.



-Image 1

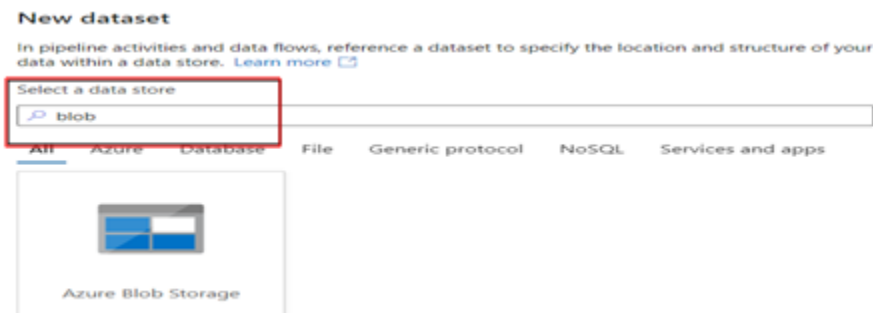
### 3. Dataset Creation:

#### Source dataset:

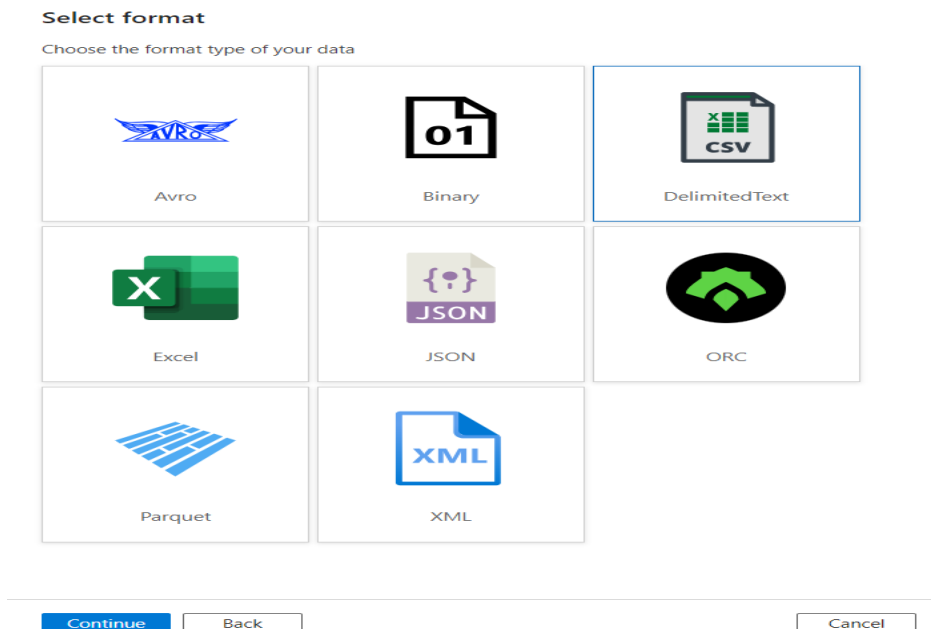
- Created a new source dataset for the source data files with the name bt\_ds\_src\_look\_up.
- Inside Azure Data Factory, in the Author tab, I selected the Dataset option and clicked on "New Dataset" (refer to image 1).
- Chose the Azure Blob Storage option.
- Next, I selected the Delimited Text file format, which brought me to the properties page where I defined the dataset name and path (refer image 2,3)



- Image 1



- Image 2



-Image 3

### Target dataset:

- Followed similar steps for the target dataset.
- Selected the existing target dataset named ss\_ds\_tg\_product\_adf\_train.
- In the file path, dynamically added the file name. Clicked on add dynamic content and selected the filename option. It gave me an expression @dataset().filename. Selected it and clicked OK.
- In Azure Data Factory (ADF), the expression @dataset().filename is used to dynamically reference the filename of the dataset being processed. (refer image 1)
- By using @dataset().filename, we ensure that the filename used in our operations is always consistent with what is defined in the dataset's configuration file (refer image 2)
- In parameters, clicked on add parameter and assigned name as filename and gave a default value product.csv (refer image 3).

Connection Schema Parameters

Linked service \* ADF\_Training Test connection Edit + New Learn more

File path \* adf-training / Output @dataset().filename

Compression type Select...

Column delimiter ① Comma (,)

Row delimiter ① Default (\r\n, or \r\n)

-Image 1

Pipeline expression builder

Add dynamic content below using any combination of expressions, functions and system variables.

@dataset().filename

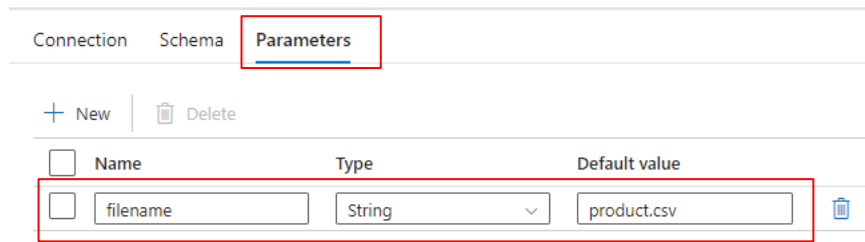
Clear contents

Parameters Functions

Search +

filename

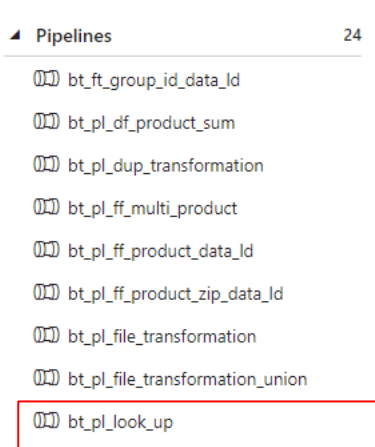
- Image 2



- Image 3

#### 4. Pipeline creation:

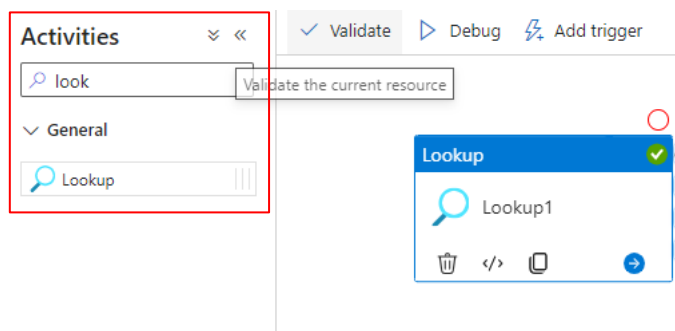
- Created a new pipeline named bt\_pl\_look\_up in Azure Data Factory (ADF) to load the data into the output folder (refer image 1).



- Image 1

#### 5. Lookup activity:

- Used lookup activity Lookup activity for the purpose of retrieving specific data or configuration settings that are required for the pipeline to process the file correctly (refer image 1).
- This ensures that the correct file is referenced and processed based on the dynamic content specified in the dataset's configuration.
- In the settings, I specified the source data and deselected the first row only option (refer image 2).



-Image 1

General **Settings** User properties

Source dataset \* bt\_ds\_src\_look\_up Open New Preview data [Learn](#)

First row only ☐

File path type ☒ File path in dataset ☐ Prefix ☐ Wildcard file path ☐ List of files <sup>①</sup>

Filter by last modified <sup>①</sup> Start time (UTC) End time (UTC)

Recursively <sup>①</sup> ☒

- Image 2

## 6. Copy activity

- Selected the copy activity to copy data from a source to a destination.
- In the settings, source option, I added the source file and in dataset properties, I added an expression by clicking on add dynamic content (refer image 1).
- Expression: @activity('Lookup1').output.value[0].file\_name (refer image 2).
- The expression @activity('Lookup1').output.value[0].file\_name is used to dynamically reference the file\_name field from the output of a Lookup activity named Lookup1.
- I did the same in the sink option, selected the target dataset and wrote an expression (refer image 3)
- Also, connected the copy activity with the lookup activity (refer image 4).

General **Source** Sink Mapping Settings User properties

Source dataset \* ss\_ds\_src\_ff\_multi\_product Open New Preview data

Dataset properties <sup>①</sup>

Name	Value
filename	@activity('Lookup1').output.value[0].f...

File path type ☒ File path in dataset ☐ Prefix ☐ Wildcard file path ☐ List of files <sup>①</sup>

- Image 1

**Pipeline expression builder**

Add dynamic content below using any combination of [expressions](#), [functions](#) and [system variables](#).

@activity('Lookup1').output.value[0].file\_name

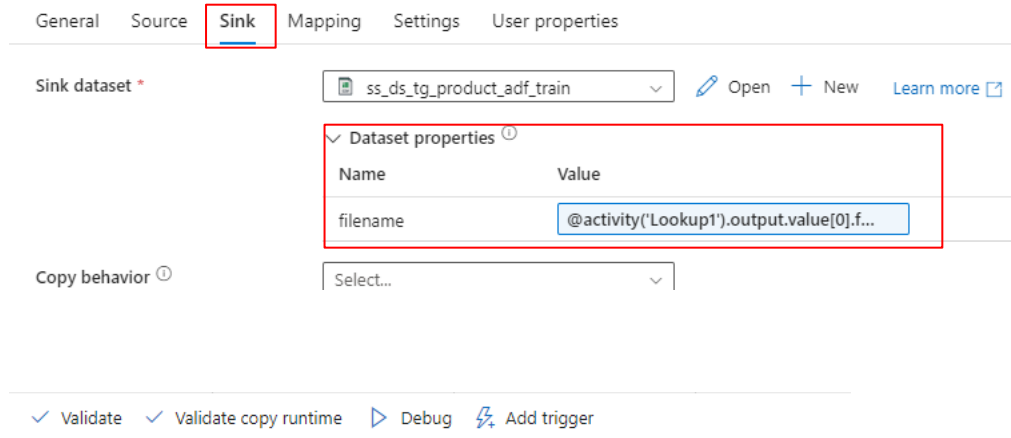
[Clear contents](#)

Activity outputs Parameters System variables Functions Variables

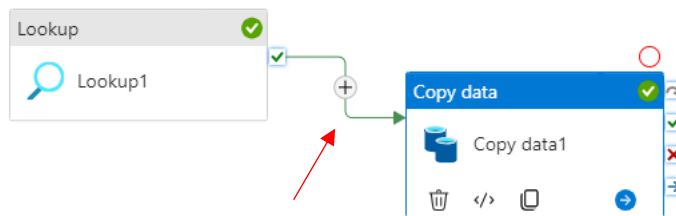
Search

Lookup1  
Lookup1 activity output

- Image 2



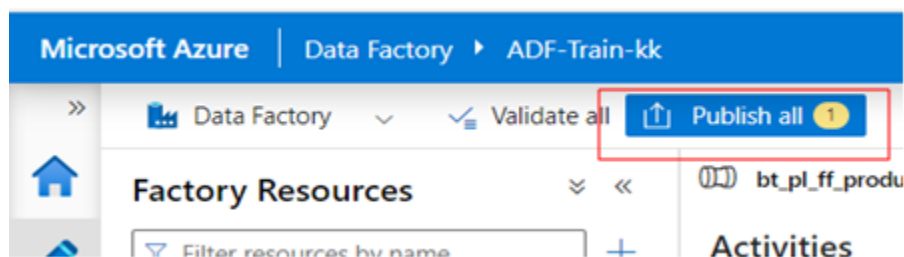
-Image 3



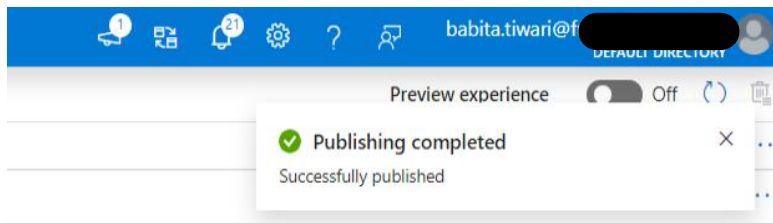
- Image 4

## 7. Publishing and Executing the Pipeline:

- The activities were saved/published, and all were successfully published and executed (Image 1, 2, 3).



- Image 1



- Image 2

Parameters

Variables

Settings

Output

Pipeline run ID:

6d999324-7670-4c5d-ab50-284d2d111721

Pipeline status

Succeeded

[View debug run consumption](#)

Data flow activity for this debug run will start as soon as the data flow debug session is ready.

All status

[Monitor in Azure Metrics](#)

[Export to CSV](#)

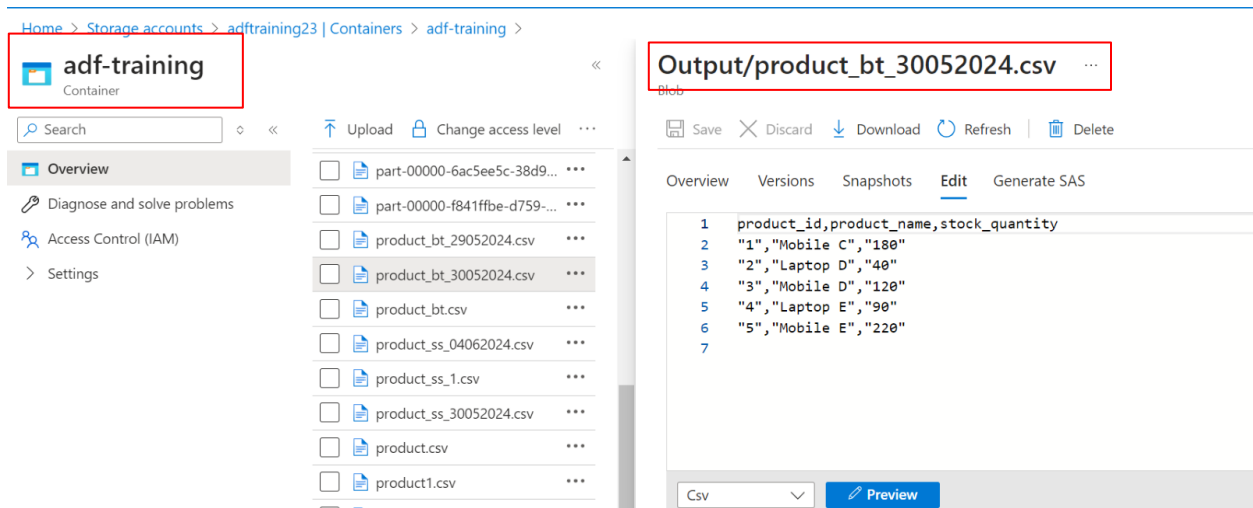
Showing 1 - 2 of 2 items

Activity name	Activity status	Activity type	Run start	Duration	Integration runtime	User properties	Activity run ID
Copy data1	<div></div> Succeeded	Copy data	6/10/2024, 6:01:08 PM	22s	AutoResolveIntegration		f2257965-c7d2-4310-822c-5ee771cd4b92
Lookup1	<div></div> Succeeded	Lookup	6/10/2024, 6:01:04 PM	4s	AutoResolveIntegration		e690c645-699d-480c-ad63-3a89563362db

-Image 3

## 8. Verifying the output:

- I then visited the output folder in the adf-training container and could see that the file was successfully loaded into the output folder (refer image 1).



-Image 1

## Summary:

The use case involves transferring data from a source to a target based on the file name specified in a config file.

