

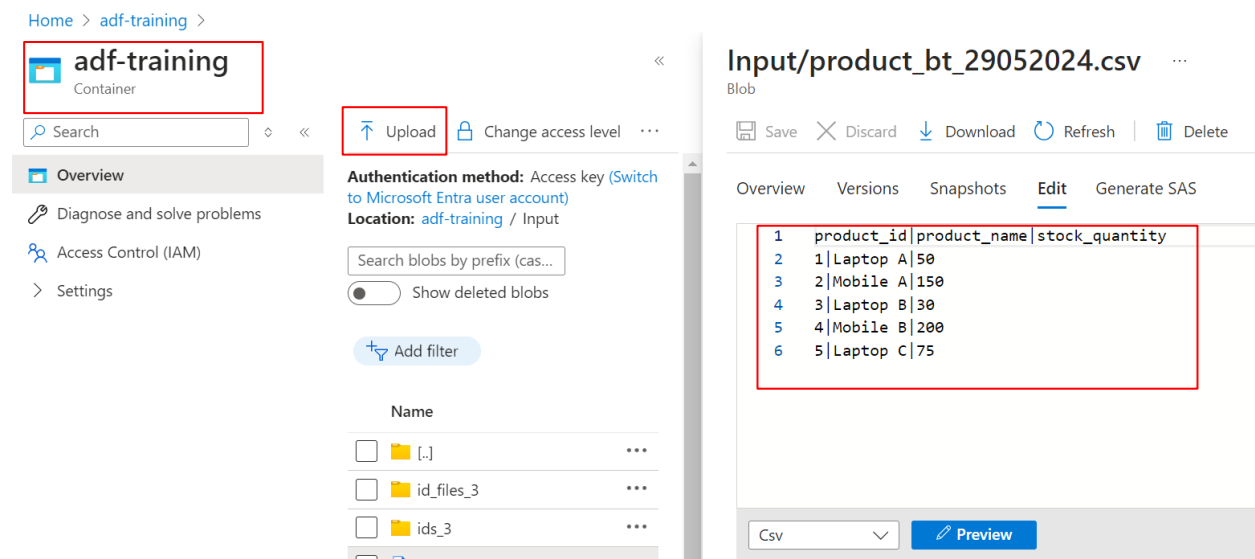
## Formal Documentation of Azure Data Factory Pipeline – Training

**Use Case:** Calculating the sum of product quantity using dataflow.

**Steps:**

### 1. Uploading Input File in the Container

- A container named adf-training was previously created in the Azure storage account.
- For this pipeline, the file was uploaded into the container in the input folder
- Once the upload was successful, the file was added to the input folder. We are transforming the data in the input file.
- The task was to calculate the sum of the stock quantity of Laptops and Mobiles (refer image 1)

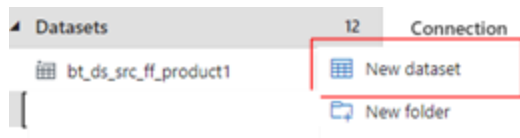


-Image 1

### 2. Dataset Creation

#### Source Dataset

- A new source dataset was created for the source data file.
- 
- Source dataset name: bt\_ds\_src\_ff\_product\_sum
- Inside Azure Data Factory, in the Author tab, I selected the Dataset option and clicked on "New Dataset" (refer to image 1).
- I chose the Azure Blob Storage option (refer to image 2).
- Next, I selected the Delimited Text file format, which brought me to the properties page where I defined the dataset name and path (refer to image 3).
- I specified the dataset name, selected the linked service, and provided the path of my input file. These steps created my source dataset (refer to image 4).
- I opened my source file and updated the connection, changing the column delimiter option to pipe (|) because my CSV file is pipe delimited (refer image 5)



- Image 1

### New dataset

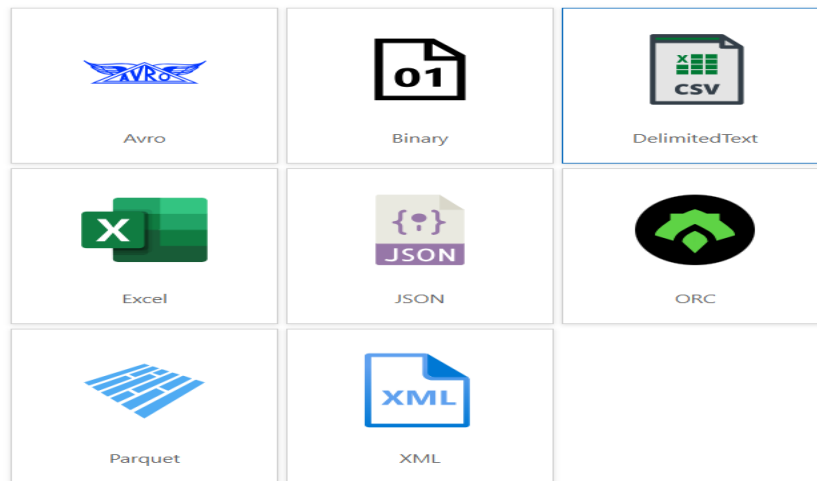
In pipeline activities and data flows, reference a dataset to specify the location and structure of your data within a data store. [Learn more](#)



- Image 2

### Select format

Choose the format type of your data



- Image 3

**Set properties**

Name

Linked service \*

File path  
 /  /

First row as header ☒

Import schema  
☒ From connection/store ☐ From sample file ☐ None

> Advanced

- Image 4

**Connection** Schema Parameters

Linked service \*  
 Test connection Edit + New Learn more

File path \*  
 /  /

Compression type

Column delimiter ⓘ

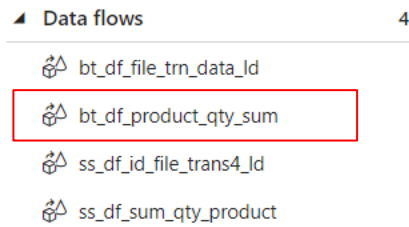
- image 5

### Target Dataset

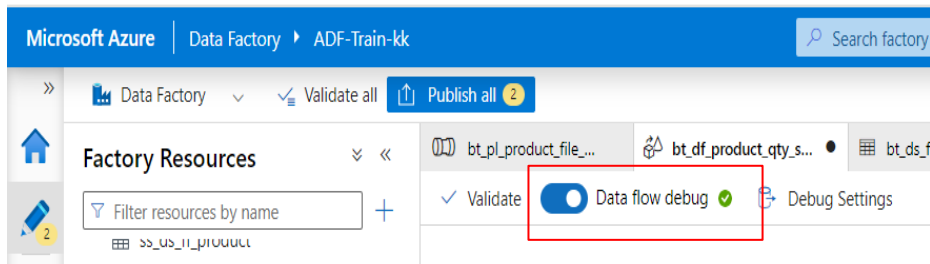
- I followed similar steps for the target dataset.
- In the Author tab, I selected the datasets, clicked on "New Dataset," selected Azure Blob Storage, and then selected the Delimited Text format, which brought me to the properties page where I defined the dataset name and path.
- Assigned path: adf-training/output
- Target dataset name: bt\_ds\_tgt\_product\_df\_sum
- I opened my source file and updated the connection, changing the column delimiter option to pipe (|) because my CSV file is pipe delimited. All other options remained unchanged.

## 3. Dataflow Creation

- Dataflow name: bt\_df\_product\_qty\_sum (refer image 1)
- The mandatory step is to enable dataflow debug (refer image 2)



- Image 1



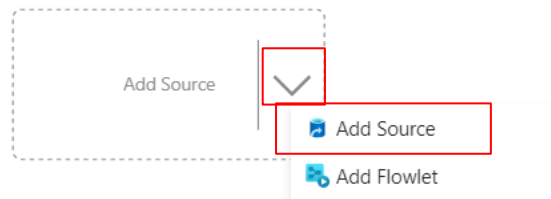
- Image 2

### Dataflow Steps:

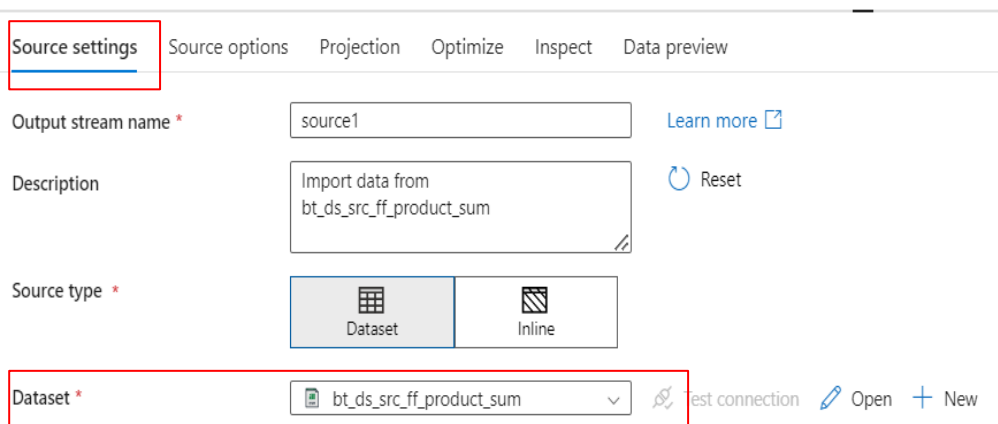
#### A. Selecting Source.

- Added source to add my source file (refer image 1)
- In source settings, I added my source dataset (refer image 2)

✓ Validate ☐ Data flow debug



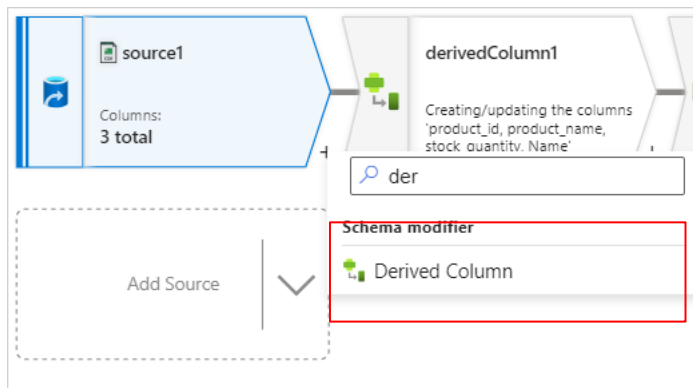
- Image 1



-Image 2

#### B. Derived Column

- I then selected the derived column (refer image 1) and added a new column named "Name" to extract the product\_name without the last Alphabet (e.g., from Laptop A to Laptop).
- I had 3 columns earlier in my file: product\_id, product\_name, stock\_quantity.
- In derived column's settings, I clicked on add column, assigned the column name as "Name" and clicked on expression (refer image 2).
- I wrote an expression which correctly extracted the product\_name and created a new column as "Name" (Eg: Laptop A to Laptop) (refer image 3).



-Image 1

This screenshot shows the 'Derived column's settings' interface. It includes tabs for 'Optimize', 'Inspect', and 'Data preview'. The 'Derived column's settings' tab is active, showing:
 

- Output stream name:** derivedColumn1
- Description:** Creating/updating the columns: product\_id, product\_name, stock\_quantity, Name
- Incoming stream:** source1
- Columns:** A table with two columns: 'Column' and 'Expression'. A new column 'Name' is being added with the expression 'substring(product\_name, 1, length(product\_name) - 2)'. A red box highlights this section.

-Image 2

This screenshot shows the 'Dataflow expression builder' and 'Data preview' sections. The 'Dataflow expression builder' shows:
 

- Column name:** Name
- Expression:** substring(product\_name, 1, length(product\_name) - 2)

 The 'Data preview' section shows a table with columns 'ANY' and 'Name', displaying data for Laptop A, Mobile A, Laptop B, Mobile B, and Laptop C. A red box highlights the 'Data preview' section.

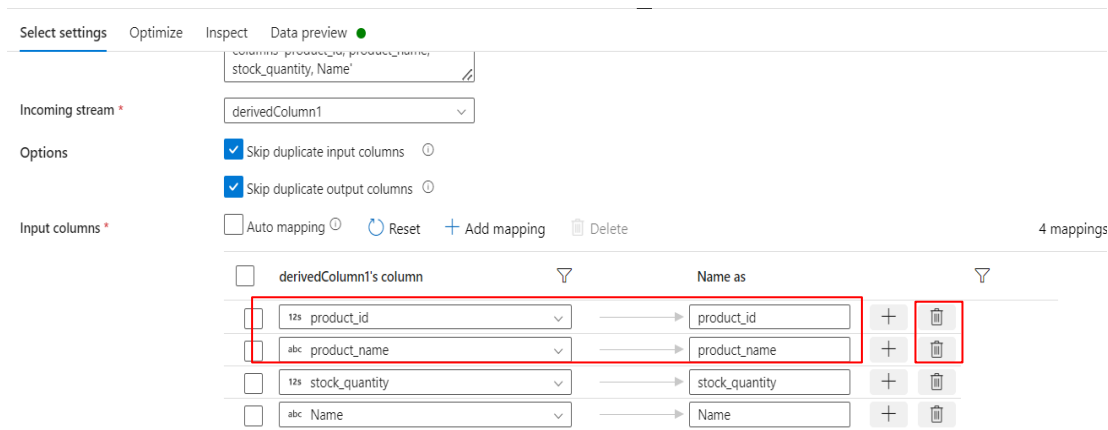
-Image 3

### C. Select

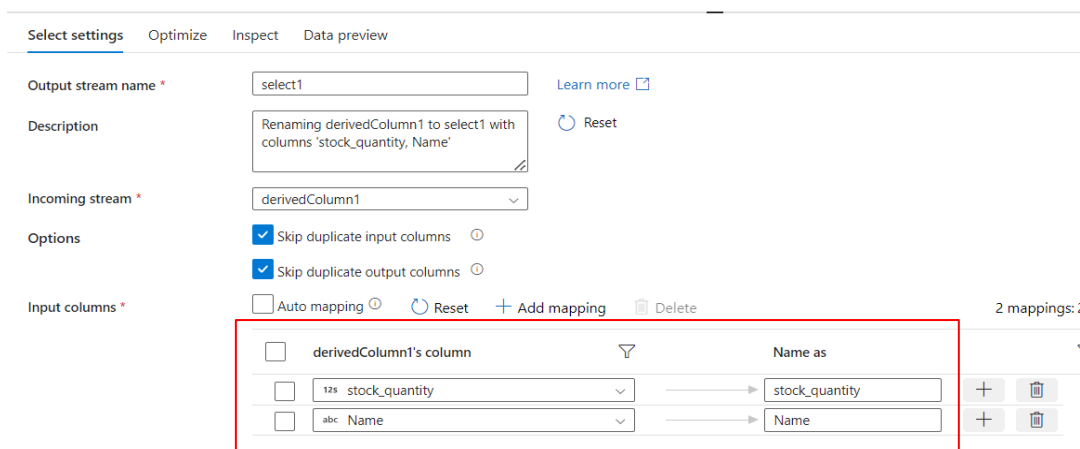
- I then used the select function to get only the required columns (refer image 1)
- After the derived function, I had 4 columns however to calculate the sum and clean the data, I only needed 2 columns "Name" and "stock\_quantity".
- In the select settings, I deleted the other 2 columns, product\_id and product\_name, and was able to get the data in the desired format.
- I checked the same from the data preview tab (refer image 4).



-Image 1



-Image 2



-Image 3



Aggregate settings
Optimize
Inspect
Data preview

Output stream name \*
[Learn more](#)

Description
Reset

Incoming stream \*

Group by
Aggregates

Grouped by: Name
+ Add
Clone
Delete
Open expression builder

Column	Expression
<input type="checkbox"/> stock_quantity	<input type="text" value="Enter expression..."/> <input type="button" value="ANY"/> <input type="button" value="+"/> <input type="button" value="🗑️"/>

-Image 3

Dataflow expression builder

aggregate1

Aggregate Columns
+ Create new
ANY stock\_quantity

Column name \*

Expression

+ - \* / || && ! ^ == === <=> !=

Expression elements
All
Functions
Input schema
Parameters
Cached lookup
Data flow library functions
Locals

Expression values
+ Create new
123 stock\_quantity
abc Name
123 abs(123 numeric\_value)
123 acos(123 numeric\_value)
ANY add(ANY first\_expression , ANY second\_expression)

Save and finish
Cancel
Clear contents

-Image 4

Aggregate settings
Optimize
Inspect
Data preview

Number of rows + INSERT 2
UPDATE 0

Refresh
Typecast
Modify
Map drifted

↑↓	Name	abc	↑↓	stock_quantity	121	↑↓
+	Laptop			155		
+	Mobile			350		

-Image 5



## E. Sink

- In sink, I selected my target dataset to copy the sum file to the output folder (refer image 1)

Sink Settings Errors Mapping Optimize Inspect Data preview ●

Output stream name \* sink1 [Learn more](#)

Description Export data to bt\_ds\_tgt\_product\_df\_sum [Reset](#)

Incoming stream \* aggregate1

Sink type \* Dataset Inline Cache

Dataset \* bt\_ds\_tgt\_product\_df\_sum [Test connection](#) [Open](#) [New](#)

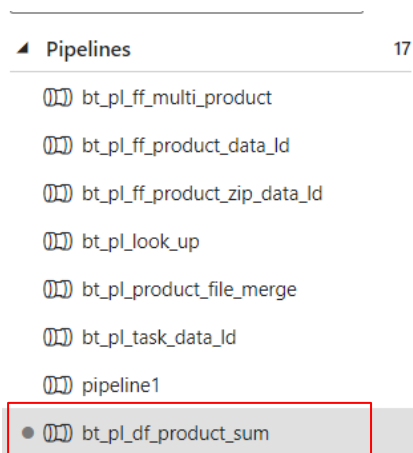
Skip line count

Options ☒ Allow schema drift ⓘ

-Image 1

## 4. Pipeline Creation

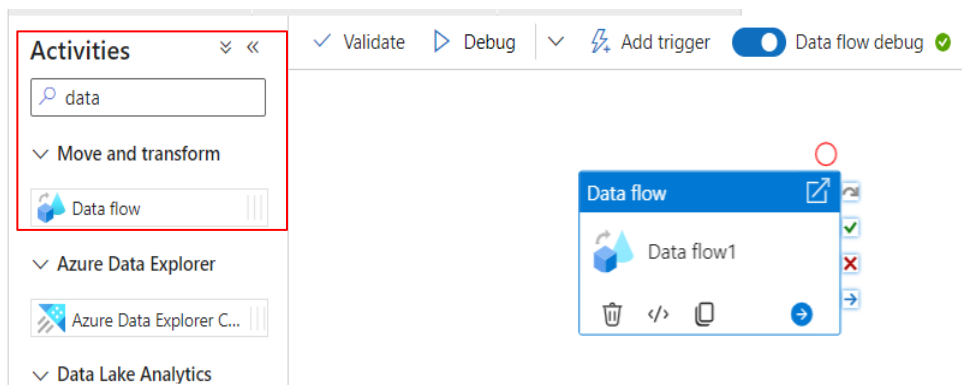
- I then created a pipeline named bt\_pl\_df\_product\_sum to execute the dataflow activities (refer image 1)



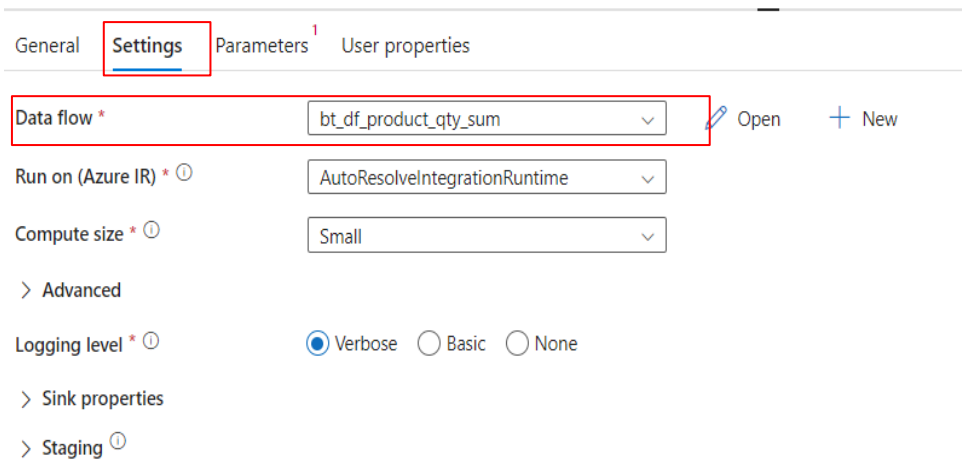
- Image 1

## 5. Dataflow Activity

- From the activities, I dragged and dropped the dataflow activity (refer image 1).
- Then, in the settings, I added the source file (refer image 2).



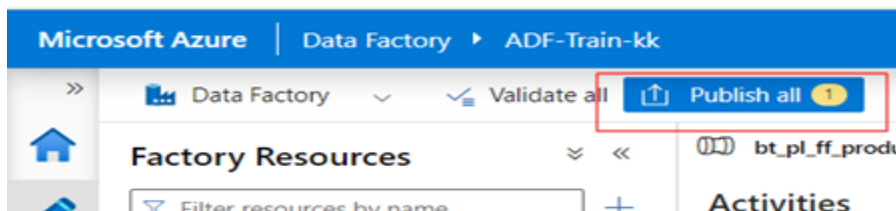
-Image 1



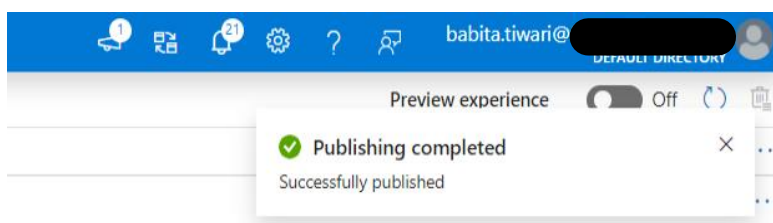
- Image 2

## 6. Publishing and Executing the Pipeline

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



- Image 1



-Image 2

Parameters

Variables

Settings

Output

Pipeline run ID: 288bee7f-3f9c-4c17-9994-aec4766f260b

Pipeline status

Succeeded

[View debug run consumption](#)

All status

[Monitor in Azure Metrics](#)

[Export to CSV](#)

Showing 1 - 1 of 1 items

Activity name	Activity status	Activity type	Run start	Duration	Integration runtime	User properties	Activity
Data flow1	<div><div></div>Succeeded</div>	Data flow	6/6/2024, 12:40:48 AM	1m 23s	AutoResolveIntegrator		2df38c

- Image 3

7. Verifying the Output

- I then visited the output folder in the adf-training container and checked the output.
- The file was generated correctly, and I got the sum of both items (refer image 1).

Output/part-00000-f841ffbe-d759-4d44-88e8-b4e1.

Blob

Save

Discard

Download

Refresh

Delete

Overview

Versions

Snapshots

Edit

Generate SAS

1

Name,stock\_quantity

2

Laptop,155

3

Mobile,350

4

-Image 1

Summary:

This document outlines the steps taken to create an Azure Data Factory pipeline to calculate the sum of product quantities. The process involved uploading the input file to a container, creating source and target datasets, setting up dataflow with source, derived column, select, and aggregate functions, creating and executing a pipeline, and verifying the output. The pipeline successfully calculated and outputted the sum of the stock quantities of Laptops and Mobiles.

