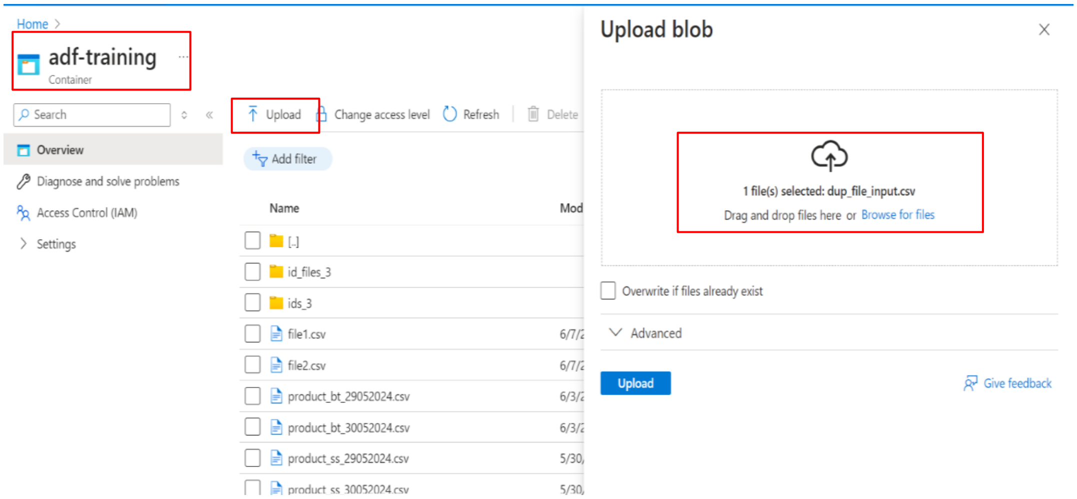
Formal Documentation of Azure Data Factory Pipeline – Training

Use Case: Removing Duplicate Data from a File

Steps:

1. Loading the File into the Container.

- A container named adf-training was previously created in the Azure storage account.
- The files were uploaded into the container, in the input folder (refer to image 1).
- File name: dup_file_input.csv.
- Once the upload was successful, the files were added to the input folder.
- Refer image 2 to review the input data that contains duplicate data.

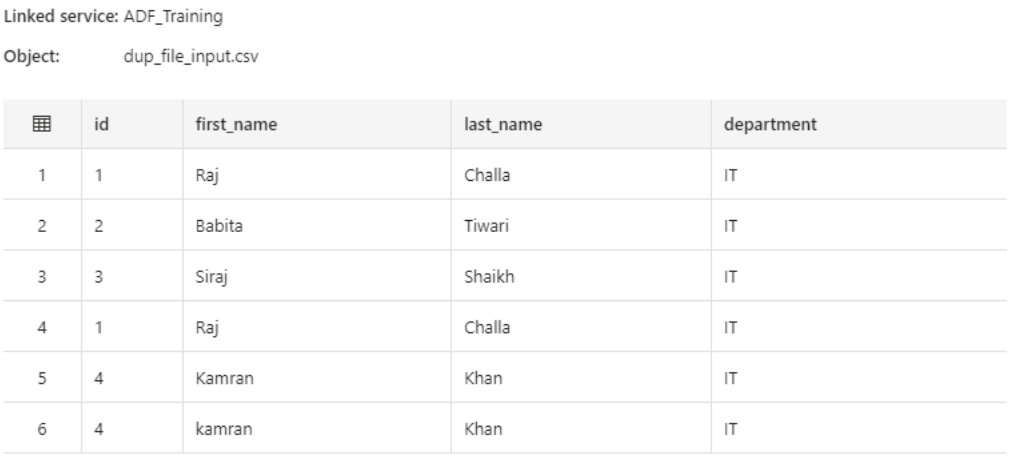


Name	Mod
[..]	
id_files_3	
ids_3	
file1.csv	6/7/2
file2.csv	6/7/2
product_bt_29052024.csv	6/3/2
product_bt_30052024.csv	6/3/2
product_ss_29052024.csv	5/30/
product_ss_30052024.csv	5/30/

1 file(s) selected: dup_file_input.csv

Drag and drop files here or Browse for files

-Image 1



Linked service: ADF_Training

Object: dup_file_input.csv

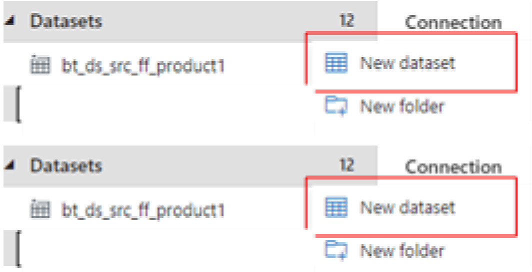
▦	id	first_name	last_name	department
1	1	Raj	Challa	IT
2	2	Babita	Tiwari	IT
3	3	Siraj	Shaikh	IT
4	1	Raj	Challa	IT
5	4	Kamran	Khan	IT
6	4	kamran	Khan	IT

- Image 2

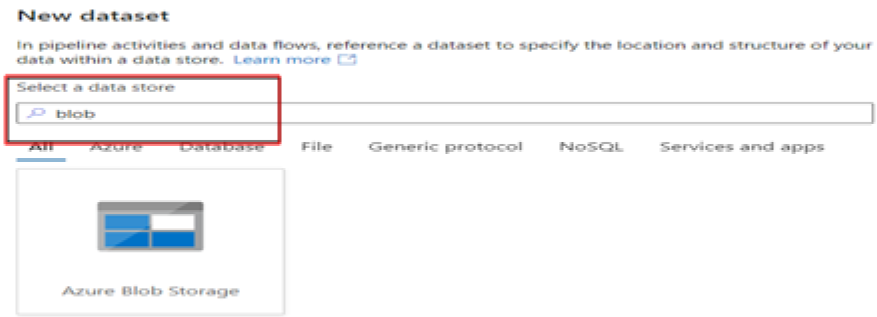
2. Dataset Creation

Source Dataset

- A new source dataset was created for the source data files.
- Source dataset name: bt_ds_src_dup_transformation.
- Inside Azure Data Factory, in the Author tab, select the Dataset option and click on "New Dataset" (refer to image 1).
- Choose the Azure Blob Storage option (refer to images 2 and 3). Next, select the Delimited Text file format, which brings you to the properties page where you define the dataset name and path.
- Specify the dataset name, select the linked service, and provide the path of the input file (refer image 4)
- These steps create the source dataset.



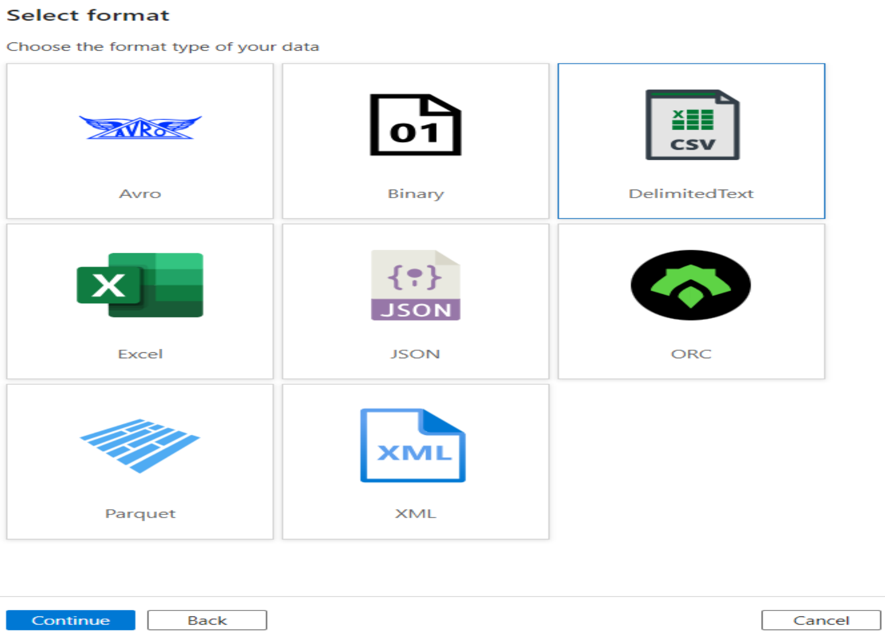
- Image 1


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

Select a data store: blob

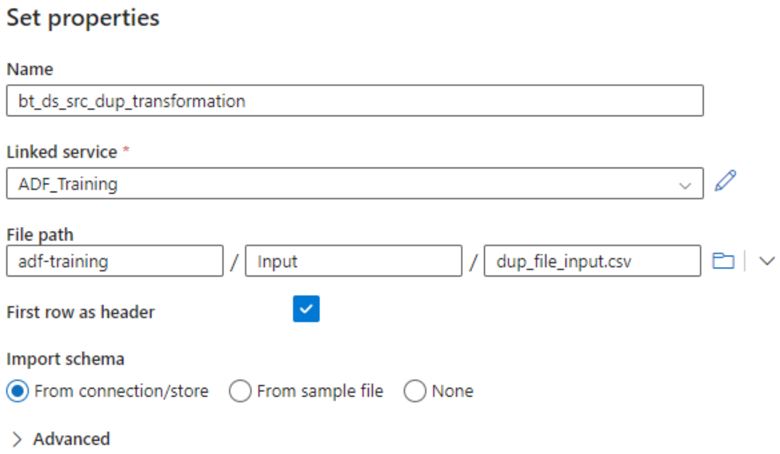
- Image 2



Select format

Choose the format type of your data

- Image 3



Set properties

Name: bt_ds_src_dup_transformation

Linked service *: ADF_Training

File path: adf-training / Input / dup_file_input.csv

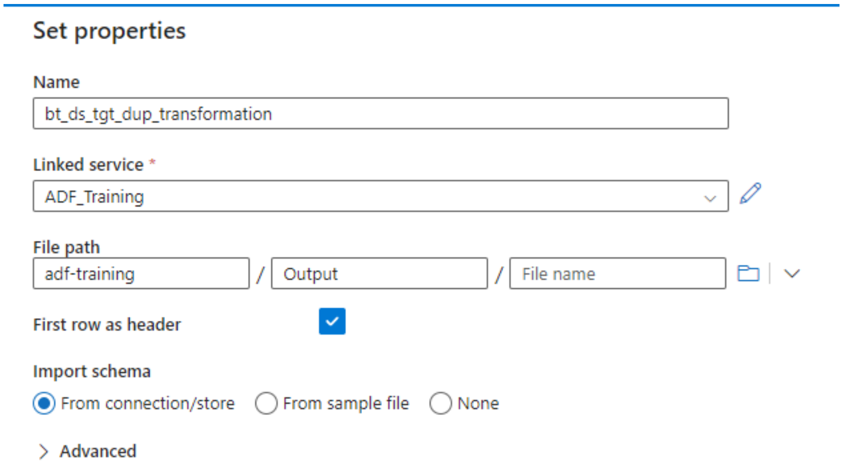
First row as header ☑

Import schema: ● From connection/store ○ From sample file ○ None

- Image 4.

<u>Target Dataset</u>

- Follow similar steps for the target dataset. In the Author tab, select the Datasets option, click on "New Dataset," select Azure Blob Storage, and then select the Delimited Text format, which brings you to the properties page where you define the dataset name and path.
- Assigned path: adf-training/output (refer image 1)
- Target dataset name: bt_ds_tgt_dup_transformation.



Set properties

Name: bt_ds_tgt_dup_transformation

Linked service *: ADF_Training

File path: adf-training / Output / File name

First row as header ☑

Import schema

● From connection/store ○ From sample file ○ None

- Image 1

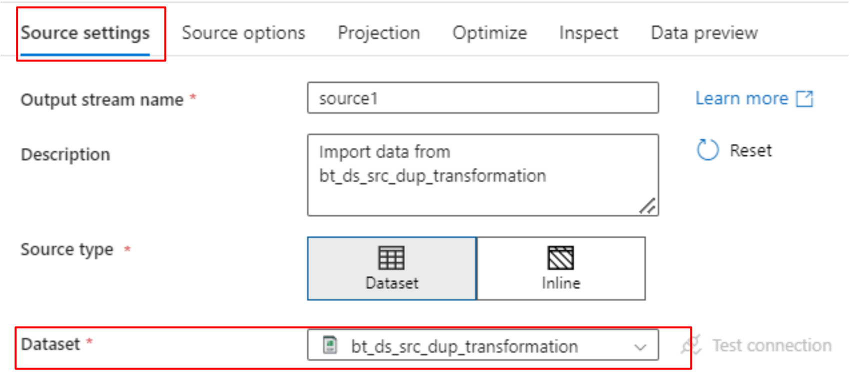
3. Creating Dataflow

- A dataflow named bt_df_dup_transformation was created to handle duplicate entries.
- The mandatory step is to enable dataflow debug.

Below are the steps:

A. Source

- In the source settings, the source file containing the duplicate data was added.



Output stream name *: source1

Description: Import data from bt_ds_src_dup_transformation

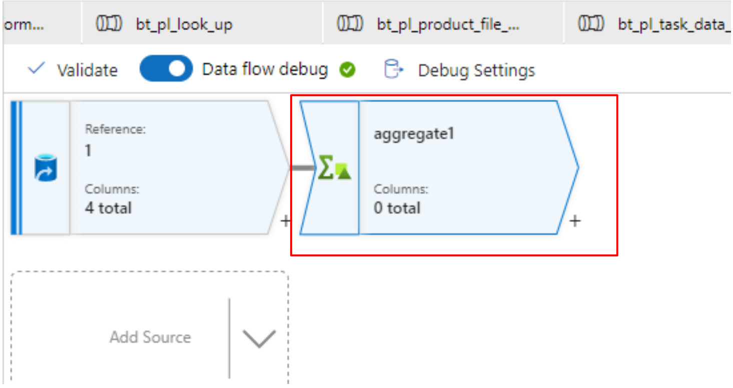
Source type *: Dataset / Inline

Dataset *: bt_ds_src_dup_transformation

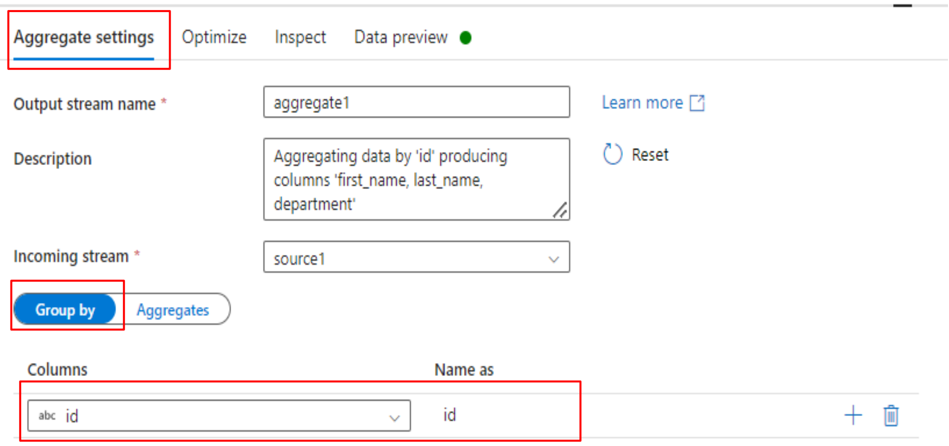
- Image 1

B. Aggregate Function

- Add an aggregate transformation to remove duplicates based on the "id" column (refer image 1)
- In "group by," specified the id column (refer image 2).
- In "aggregates," select three columns: first_name, last_name, department, and use the first expression (refer image 3).
- The first expression in the Aggregate function ensures that for each unique id, the first occurrence of the first_name, last_name, and department columns is selected (refer image 4).
- The duplicate data was successfully removed. I checked the same in the data preview tab (refer image 5).



- Image 1



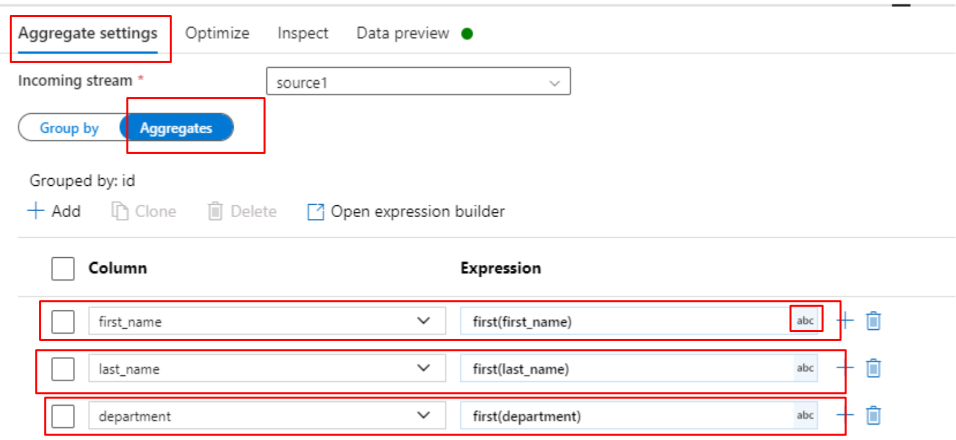
Output stream name * aggregate1

Description: Aggregating data by 'id' producing columns 'first_name, last_name, department'

Incoming stream * source1

Columns	Name as
abc id	id

- Image 2

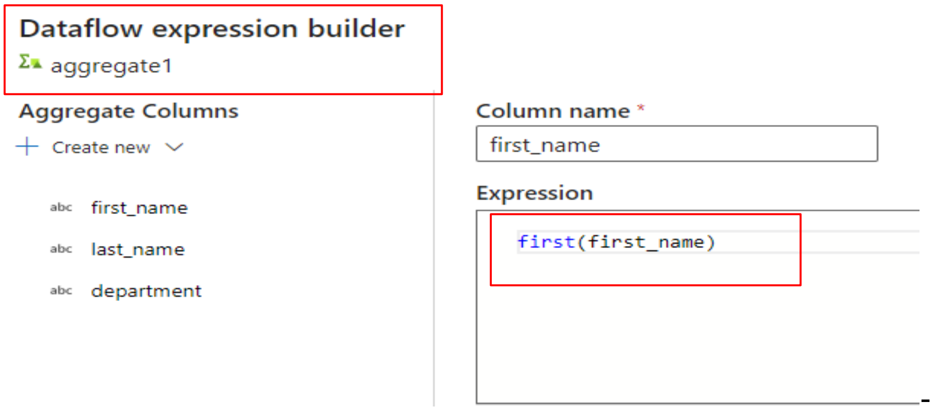


Incoming stream * source1

Grouped by: id

□ Column	Expression
□ first_name	first(first_name)
□ last_name	first(last_name)
□ department	first(department)

-Image 3



Dataflow expression builder

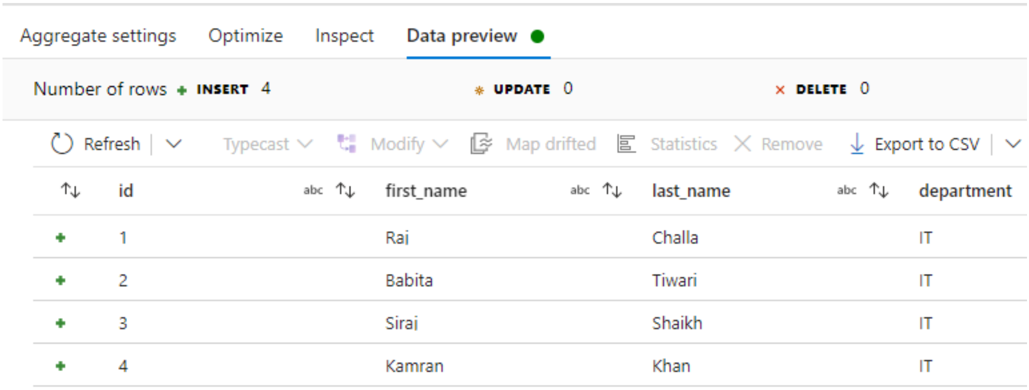
aggregate1

Column name *: first_name

Expression:

```
first(first_name)
```

- Image 4



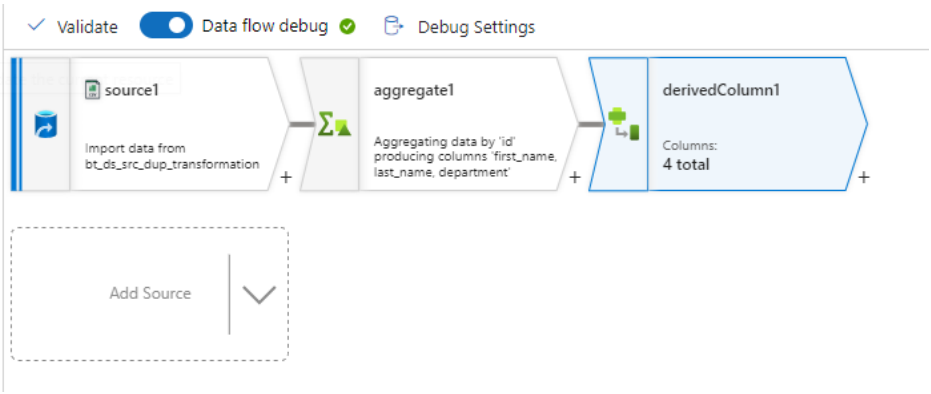
Number of rows INSERT 4 UPDATE 0 DELETE 0

	id	first_name	last_name	department
+	1	Raj	Challa	IT
+	2	Babita	Tiwari	IT
+	3	Siraj	Shaikh	IT
+	4	Kamran	Khan	IT

- Image 5

C. Derived Column

- Select the derived column (refer to image 1) and add a new column named full_name to concatenate the first and last names in the file.
- The original file had 4 columns: id, first_name, last_name, department.
- In derived column settings, click on "add column," assign the column name as full_name, and click on "expression" (refer to image 2).
- Wrote a concat expression to create a new column named full_name (refer image 3).
- I then checked my data through data preview and the new column named full_name was added (refer image 4).

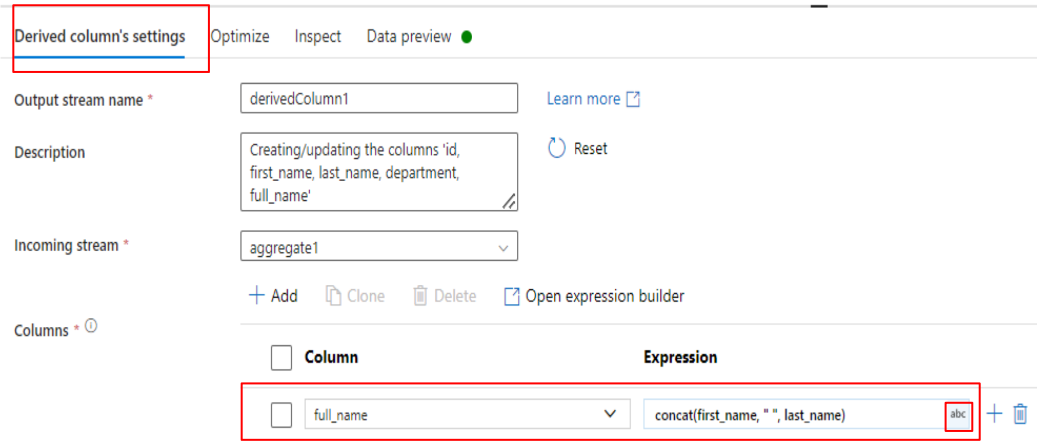


source1
Import data from bt_ds_src_dup_transformation

aggregate1
Aggregating data by 'id' producing columns 'first_name, last_name, department'

derivedColumn1
Columns:
4 total

- Image 1.



Output stream name * derivedColumn1

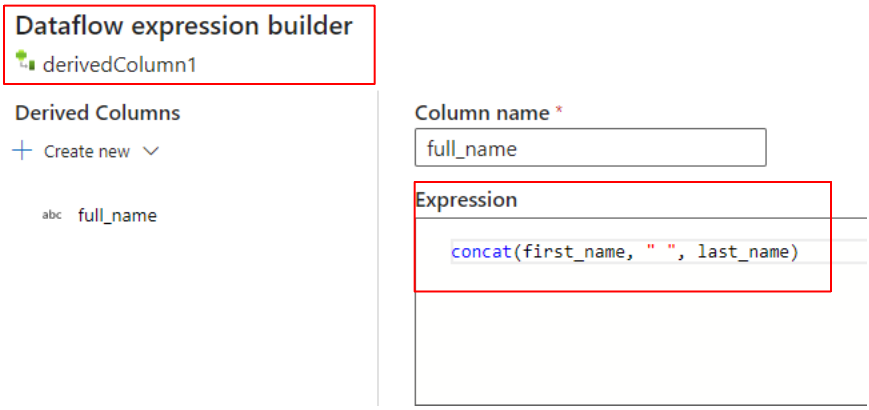
Description: Creating/updating the columns 'id, first_name, last_name, department, full_name'

Incoming stream * aggregate1

Columns *

□ Column	Expression
□ full_name	concat(first_name, " ", last_name) abc

- Image 2



Dataflow expression builder
derivedColumn1

Derived Columns

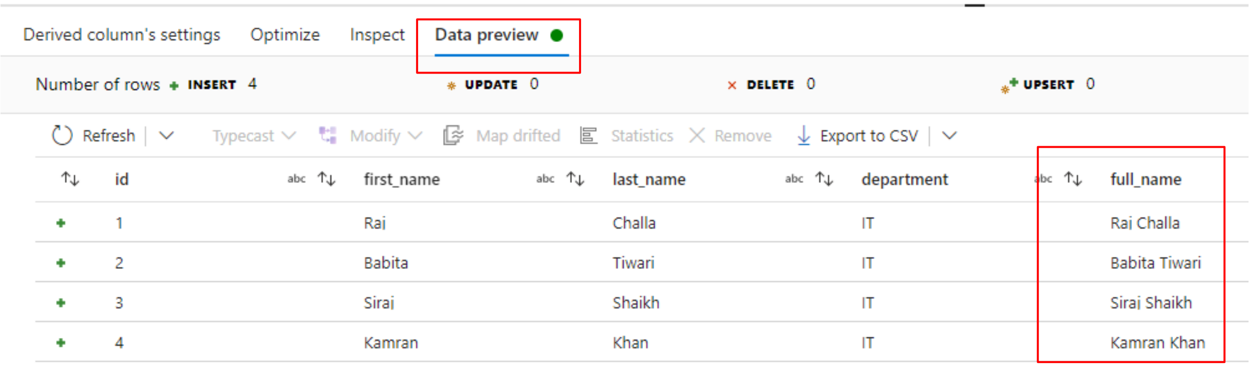
abc full_name

Column name * full_name

Expression

```
concat(first_name, " ", last_name)
```

- Image 3



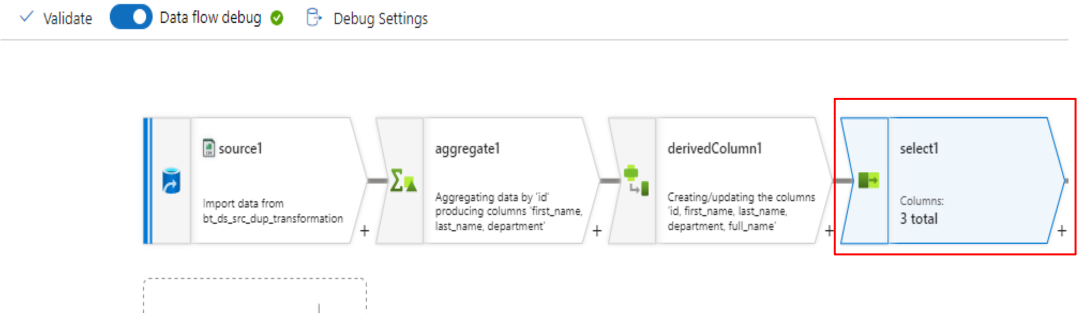
Number of rows + INSERT 4 UPDATE 0 × DELETE 0 UPSERT 0

	id	first_name	last_name	department	full_name
+	1	Raj	Challa	IT	Raj Challa
+	2	Babita	Tiwari	IT	Babita Tiwari
+	3	Siraj	Shaikh	IT	Siraj Shaikh
+	4	Kamran	Khan	IT	Kamran Khan

Image 4

D. Select

- Use the select function to get only the required columns (refer to image 1).
- After the derived function, there were 5 columns, but only id, full_name, and department were needed.
- In select settings, delete the other 2 columns, first_name and last_name, to get the data in the desired format (refer image 2).
- I also changed the position of the full name column from last to middle by just dragging and moving.
- Check the data in the Data Preview tab (refer to image 4).



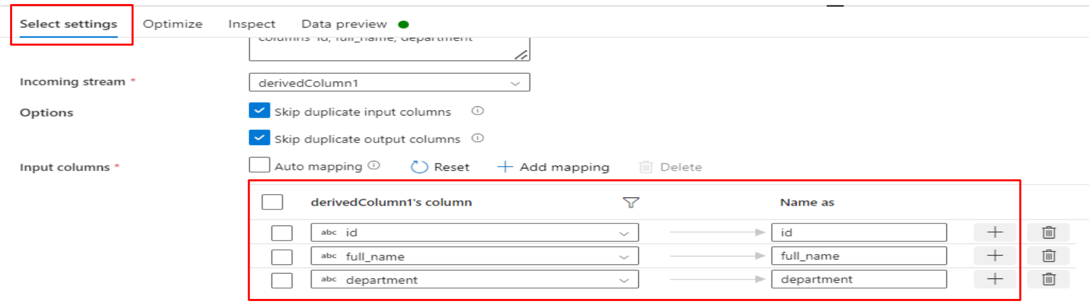
source1: Import data from bt_ds_src_dup_transformation

aggregate1: Aggregating data by 'id' producing columns 'first_name, last_name, department'

derivedColumn1: Creating/updating the columns 'id, first_name, last_name, department, full_name'

select1: Columns: 3 total

- Image 1



columns 'id, full_name, department'

Incoming stream *: derivedColumn1

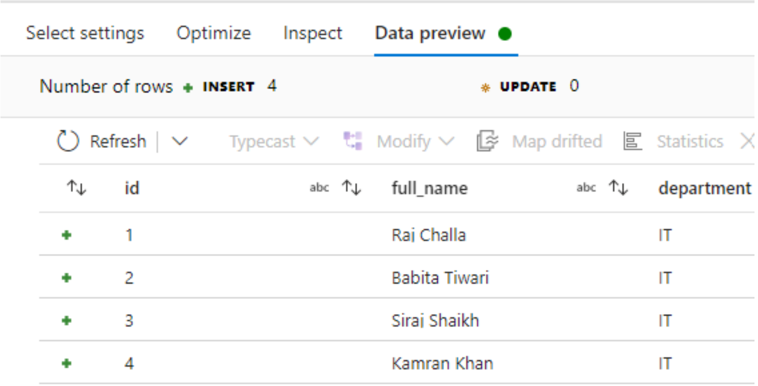
Options: ☑ Skip duplicate input columns

☑ Skip duplicate output columns

Input columns *: □ Auto mapping

derivedColumn1's column	Name as
abc id	id
abc full_name	full_name
abc department	department

- Image 2



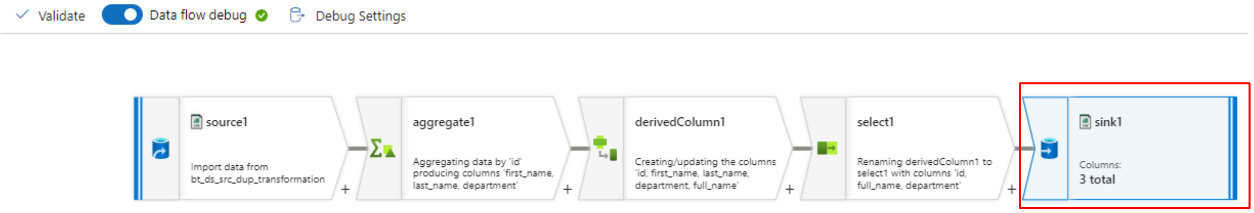
Number of rows + INSERT 4 * UPDATE 0

	id	full_name	department
+	1	Raj Challa	IT
+	2	Babita Tiwari	IT
+	3	Siraj Shaikh	IT
+	4	Kamran Khan	IT

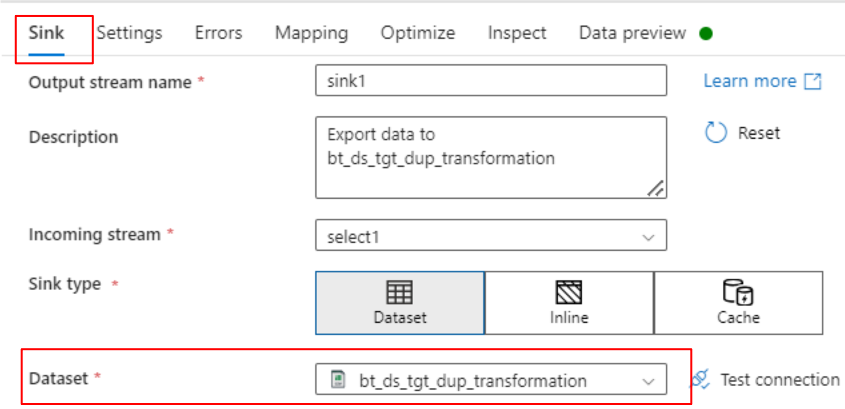
- Image 3

E. Sink

- In sink, select the target dataset to copy the cleaned file to the output folder (refer to image 1, 2).
- I also specified the name of my output file (refer image 3).


Validate
Data flow debug
Debug Settings
source1
Import data from bt_ds_src_dup_transformation
aggregate1
Aggregating data by 'id' producing columns 'first_name, last_name, department'
derivedColumn1
Creating/updating the columns 'id, first_name, last_name, department, full_name'
select1
Renaming derivedColumn1 to select1 with columns 'id, full_name, department'
sink1
Columns:
3 total

- Image 1



Output stream name * sink1

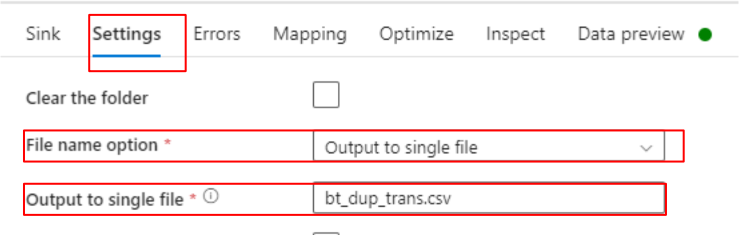
Description: Export data to bt_ds_tgt_dup_transformation

Incoming stream * select1

Sink type * Dataset | Inline | Cache

Dataset * bt_ds_tgt_dup_transformation

Image 2



Clear the folder □

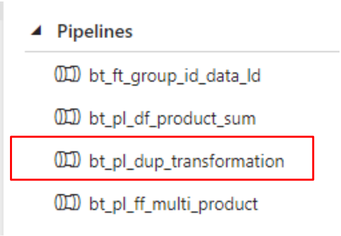
File name option * Output to single file

Output to single file * bt_dup_trans.csv

- Image 3

4. Pipeline Creation

- Create a pipeline named bt_pl_dup_transformation to execute the dataflow activities (refer to image 1).



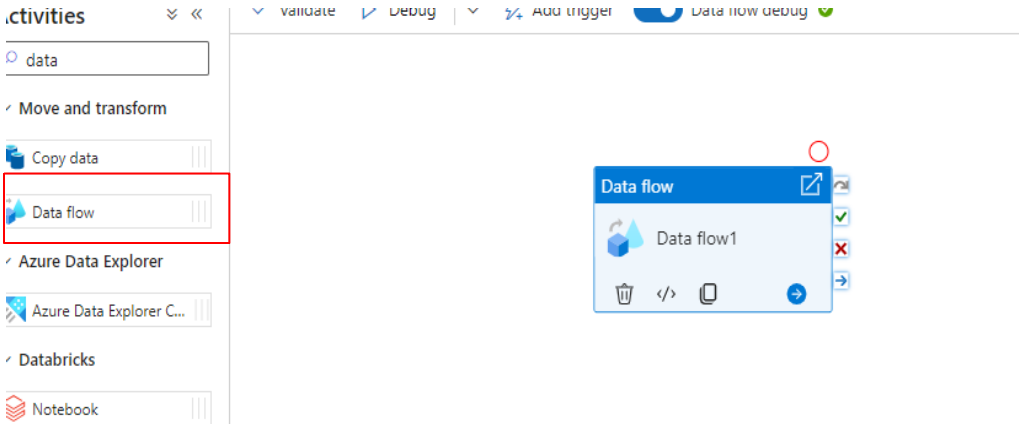
Pipelines

- bt_ft_group_id_data_ld
- bt_pl_df_product_sum
- bt_pl_dup_transformation
- bt_pl_ff_multi_product

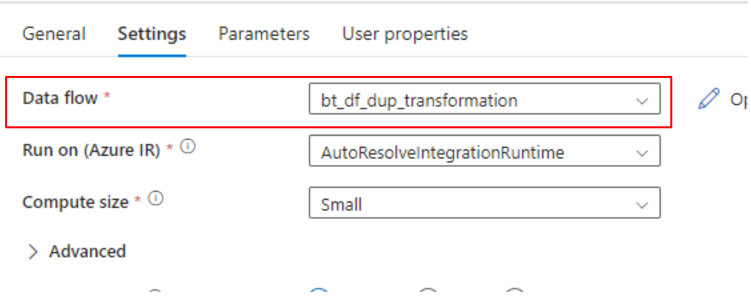
- Image 1

5. Dataflow Activity

- From the activities, drag and drop the dataflow activity (refer to image 1).
- In the settings, selected my dataflow. (refer to image 2).



- Image 1



Data flow * bt_df_dup_transformation

Run on (Azure IR) * AutoResolveIntegrationRuntime

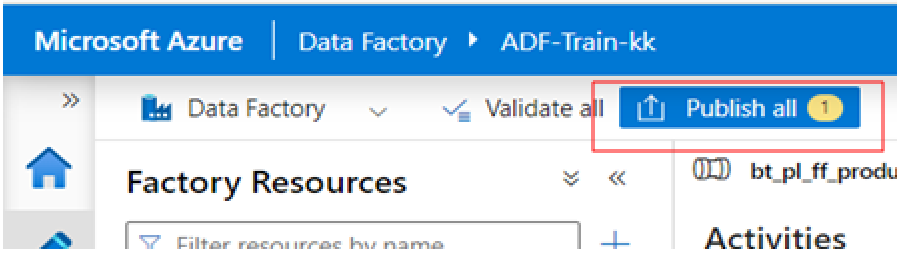
Compute size * Small

> Advanced

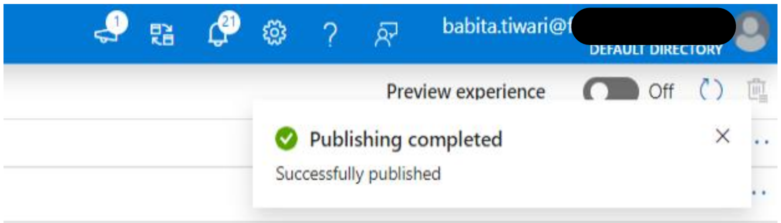
-Image 2

6. Publishing and Executing the Pipeline

- Save/publish the activities, and successfully publish and execute them (refer to images 1, 2, 3).



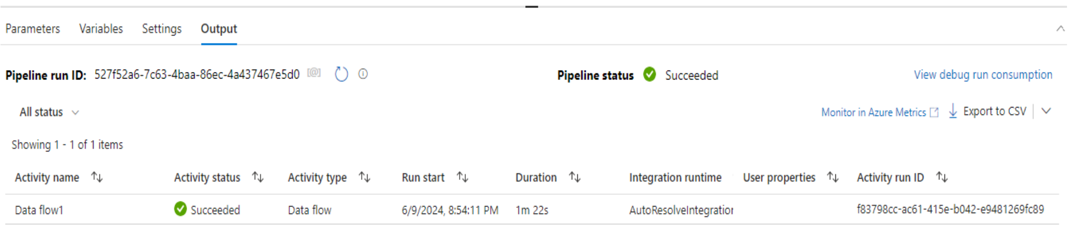
- Image 1



Publishing completed

Successfully published

- Image 2



Pipeline run ID: 527f52a6-7c63-4baa-86ec-4a437467e5d0

Pipeline status ✓ Succeeded

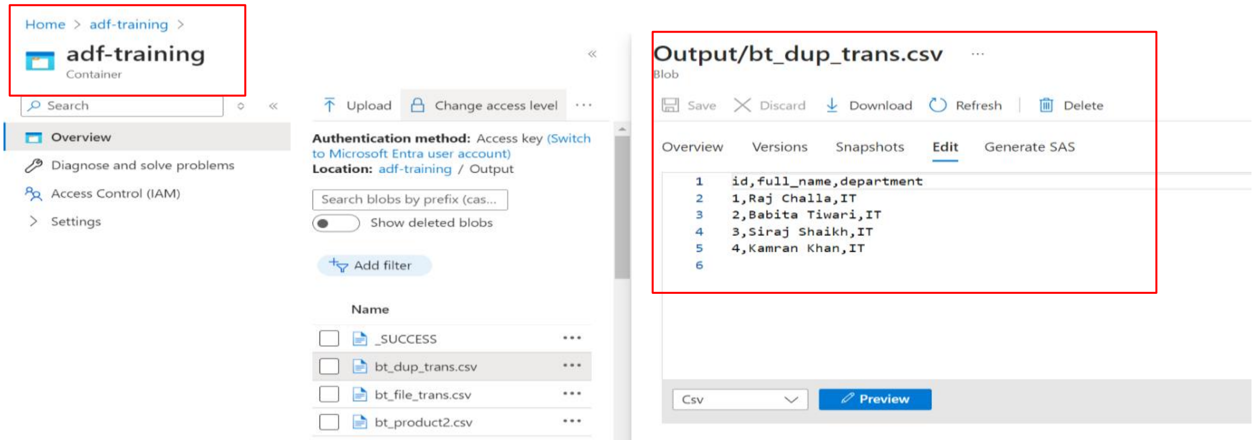
Showing 1 - 1 of 1 items

Activity name	Activity status	Activity type	Run start	Duration	Integration runtime	User properties	Activity run ID
Data flow1	✓ Succeeded	Data flow	6/9/2024, 8:54:11 PM	1m 22s	AutoResolveIntegratior		f83798cc-ac61-415e-b042-e9481269fc89

- Image 3

7. Verifying the Output

- Visited the output folder in the adf-training container and checked the output.
- The file was generated correctly, and duplicate data was removed from the file (refer image 1)



Authentication method: Access key (Switch to Microsoft Entra user account)
Location: adf-training / Output

Name
- _SUCCESS
- bt_dup_trans.csv
- bt_file_trans.csv
- bt_product2.csv

```
id,full_name,department
1,Raj Challa,IT
2,Babita Tiwari,IT
3,Siraj Shaikh,IT
4,Kamran Khan,IT
```

- Image 1

Summary:

In this use case, duplicate data was removed from a file using Azure Data Factory (ADF). The process included loading the file into a container, creating source and target datasets, setting up a dataflow for duplicate handling, and configuring aggregate and derived column transformations. A pipeline was created to execute these activities, and the results were verified in the output folder. The file was processed successfully, with duplicates removed and desired columns retained.