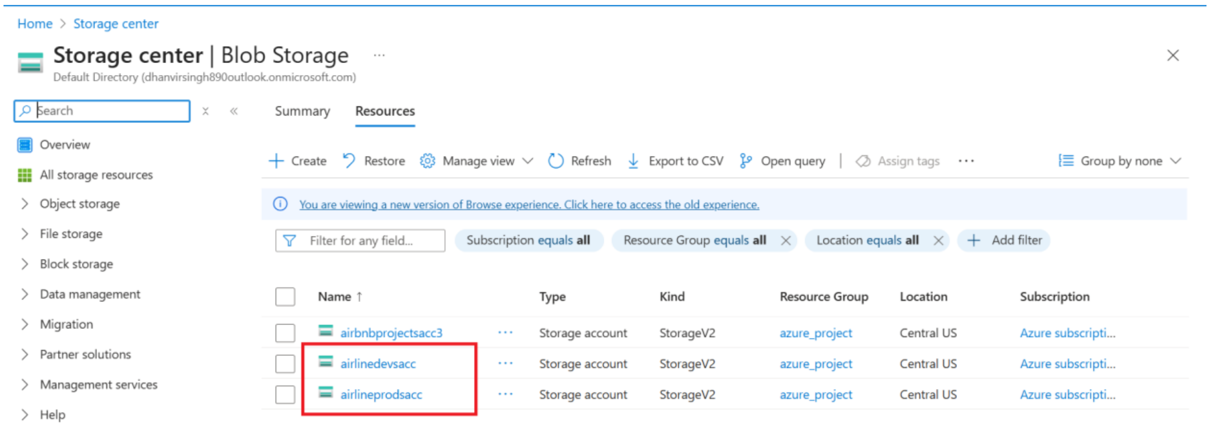
Project:

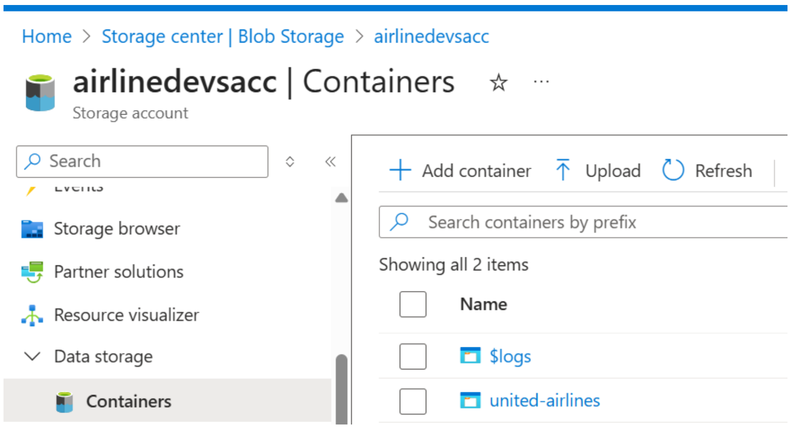
Firstly, I created 2 storage account

1. airlinedevacc
2. airlineprodacc



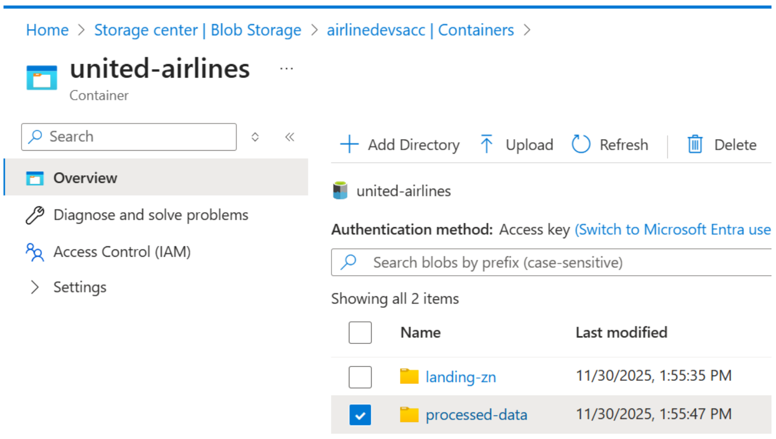
Name ↑	Type	Kind	Resource Group	Location	Subscription
airbnbprojectsacc3	Storage account	StorageV2	azure_project	Central US	Azure subscripti...
airlinedevsacc	Storage account	StorageV2	azure_project	Central US	Azure subscripti...
airlineprodsacc	Storage account	StorageV2	azure_project	Central US	Azure subscripti...

Now i created a container name: **united-airlines** and under it i created **landing-zn and processed-data**



Showing all 2 items

Name
$logs
united-airlines



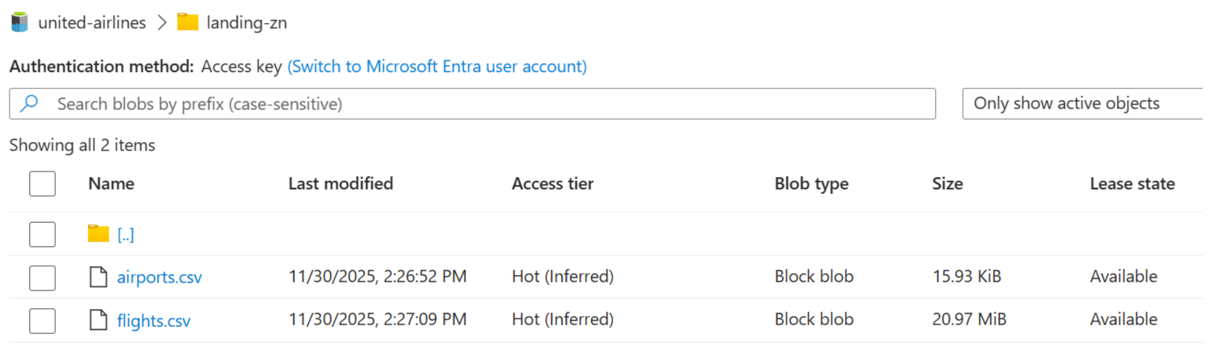
Home > Storage center | Blob Storage > airlinedevsacc | Containers >
united-airlines
Container
Search
Overview
Diagnose and solve problems
Access Control (IAM)
Settings
Add Directory
Upload
Refresh
Delete
united-airlines
Search blobs by prefix (case-sensitive)

Authentication method: Access key (Switch to Microsoft Entra use

Showing all 2 items

Name	Last modified
landing-zn	11/30/2025, 1:55:35 PM
processed-data	11/30/2025, 1:55:47 PM

Under **landing-zn** I upload 2 csv files


united-airlines > landing-zn

Authentication method: Access key (Switch to Microsoft Entra user account)

Search blobs by prefix (case-sensitive) | Only show active objects

Showing all 2 items

Name	Last modified	Access tier	Blob type	Size	Lease state
[..]					
airports.csv	11/30/2025, 2:26:52 PM	Hot (Inferred)	Block blob	15.93 KiB	Available
flights.csv	11/30/2025, 2:27:09 PM	Hot (Inferred)	Block blob	20.97 MiB	Available

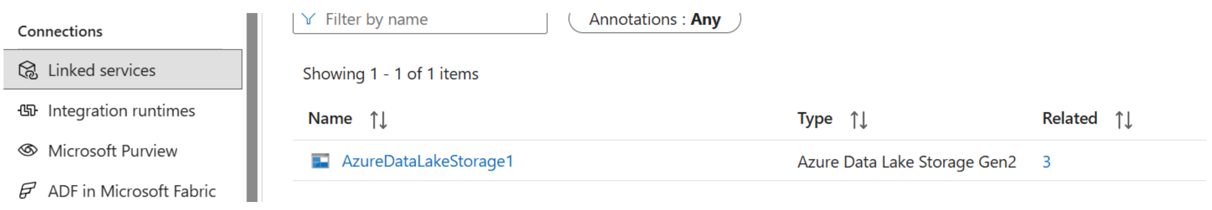
DailyFlightSource (flight.csv)

```
Carrier | OriginAirportID | DestAirportID | DepDelay | ArrDelay
DL      | 11433           | 13303         | -3       | 1
DL      | 14869           | 12478         | 0        | -8
DL      | 14057           | 14869         | -4       | -15
```

2. AirlineDimSource (airport.csv)

```
airport_id | city       | state | name
11433      | Hilo       | HI    | Hilo International
13303      | Miami      | FL    | Miami International
14869      | San Diego  | CA    | San Diego International
```

Now i created a linked service as to connect data factory with adls so as to fetch dataset files



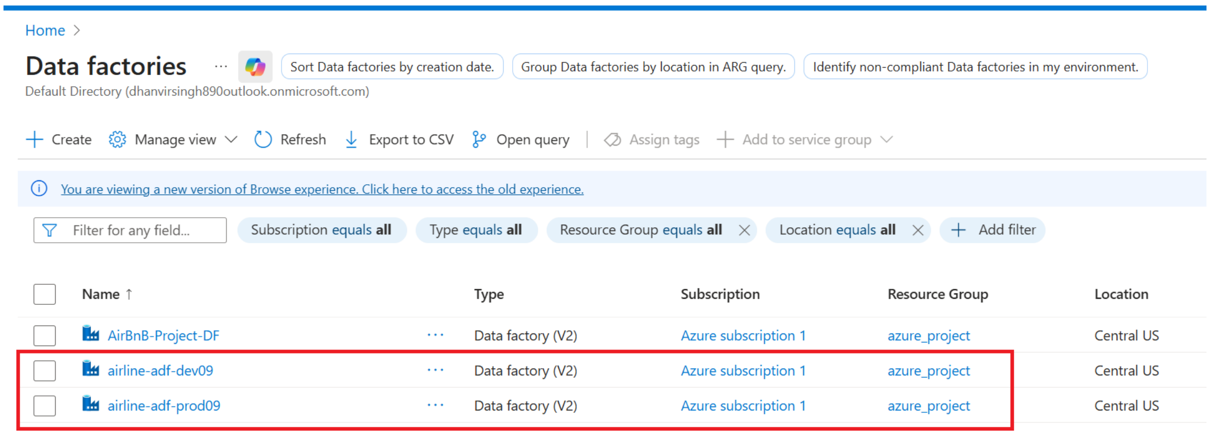
Showing 1 - 1 of 1 items

Name	Type	Related
AzureDataLakeStorage1	Azure Data Lake Storage Gen2	3

Now I created two Azure Data Factories – one for development and one for production. The development Data Factory is connected to Azure DevOps for source control.

Whenever I make changes in the Dev environment, I publish them, and through a CI/CD pipeline in Azure DevOps, those changes are automatically deployed to the production Data Factory.

This setup ensures proper environment separation, version control, and automated deployment without manual intervention.


Home >

Data factories ··· Sort Data factories by creation date. | Group Data factories by location in ARG query. | Identify non-compliant Data factories in my environment.

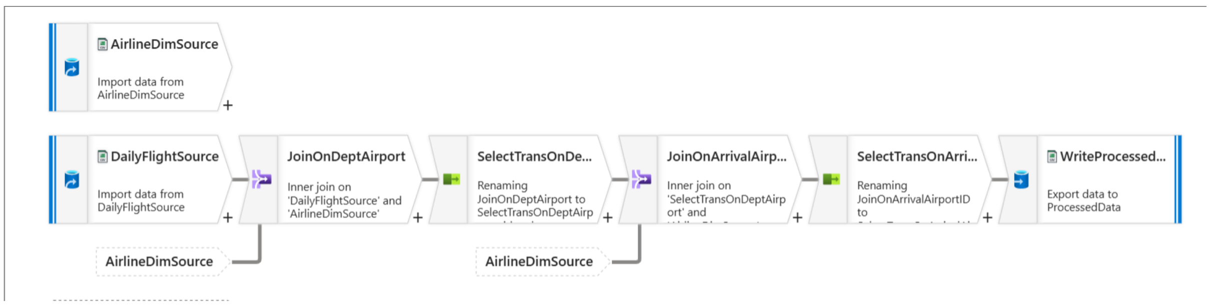
Default Directory (dhanvirsingh890outlook.onmicrosoft.com)

+ Create | Manage view | Refresh | Export to CSV | Open query | Assign tags | + Add to service group

You are viewing a new version of Browse experience. Click here to access the old experience.

Filter for any field... | Subscription equals **all** | Type equals **all** | Resource Group equals **all** | Location equals **all** | + Add filter

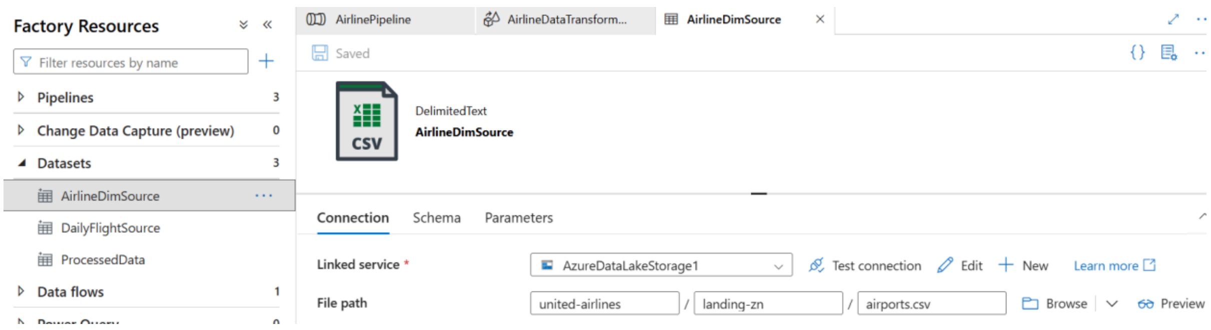
Name ↑	Type	Subscription	Resource Group	Location
AirBnB-Project-DF	Data factory (V2)	Azure subscription 1	azure_project	Central US
airline-adf-dev09	Data factory (V2)	Azure subscription 1	azure_project	Central US
airline-adf-prod09	Data factory (V2)	Azure subscription 1	azure_project	Central US


AirlineDimSource
Import data from AirlineDimSource
DailyFlightSource
Import data from DailyFlightSource
JoinOnDeptAirport
Inner join on 'DailyFlightSource' and 'AirlineDimSource'
SelectTransOnDe...
Renaming JoinOnDeptAirport to SelectTransOnDeptAirp
JoinOnArrivalAirp...
Inner join on 'SelectTransOnDeptAirport' and
SelectTransOnArri...
Renaming JoinOnArrivalAirportID to
WriteProcessed...
Export data to ProcessedData
AirlineDimSource
AirlineDimSource

Data Flow Overview

I created a **Mapping Data Flow** in Azure Data Factory to transform and combine airline-related data. The flow consists of:

1. **Source Datasets**
   a. **AirlineDimSource** → Reads airports.csv from Azure Data Lake (landing-zn folder).
   b. **DailyFlightSource** → Reads flights.csv from Azure Data Lake (landing-zn folder).
   c. **ProcessedData** → Stores transformed data in Azure Data Lake.


Factory Resources

Filter resources by name

Pipelines	3
Change Data Capture (preview)	0
Datasets	3
AirlineDimSource	
DailyFlightSource	
ProcessedData	
Data flows	1

AirlinePipeline | AirlineDataTransform... | AirlineDimSource

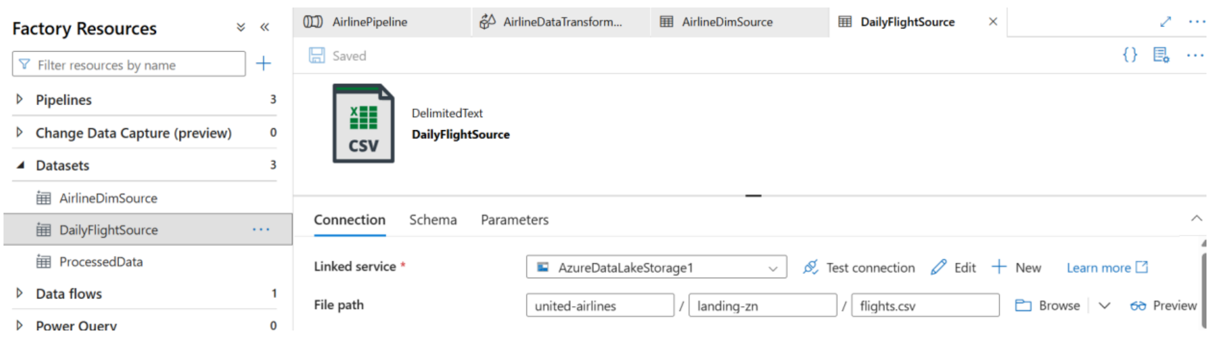
Saved

DelimitedText
AirlineDimSource

Connection | Schema | Parameters

Linked service * AzureDataLakeStorage1 | Test connection | Edit | New | Learn more

File path united-airlines / landing-zn / airports.csv | Browse | Preview


Factory Resources

Pipelines 3
Change Data Capture (preview) 0
Datasets 3
AirlineDimSource
DailyFlightSource
ProcessedData
Data flows 1
Power Query 0

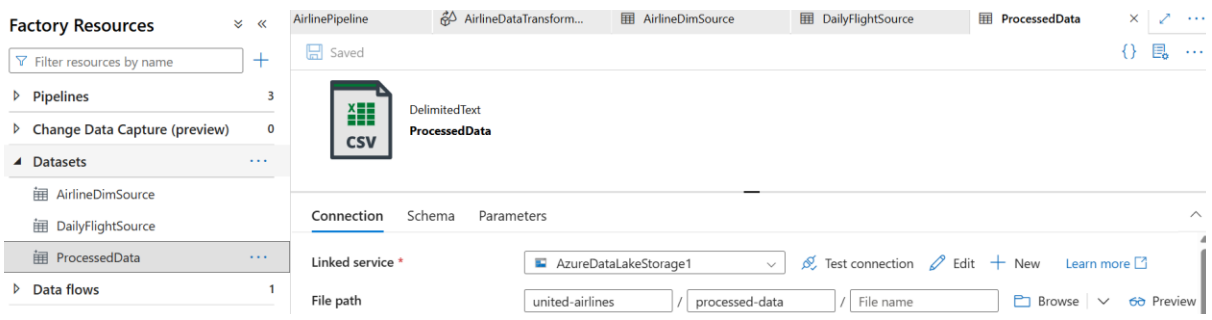
AirlinePipeline | AirlineDataTransform... | AirlineDimSource | DailyFlightSource

Saved

DelimitedText
DailyFlightSource

Connection | Schema | Parameters

Linked service *: AzureDataLakeStorage1
File path: united-airlines / landing-zn / flights.csv


Factory Resources

Pipelines 3
Change Data Capture (preview) 0
Datasets
AirlineDimSource
DailyFlightSource
ProcessedData
Data flows 1

AirlinePipeline | AirlineDataTransform... | AirlineDimSource | DailyFlightSource | ProcessedData

Saved

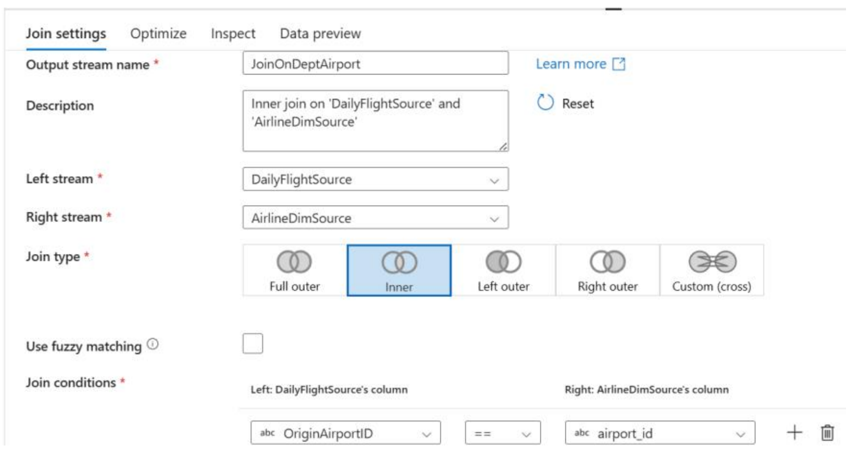
DelimitedText
ProcessedData

Connection | Schema | Parameters

Linked service *: AzureDataLakeStorage1
File path: united-airlines / processed-data / File name

JoinOnDeptAirport

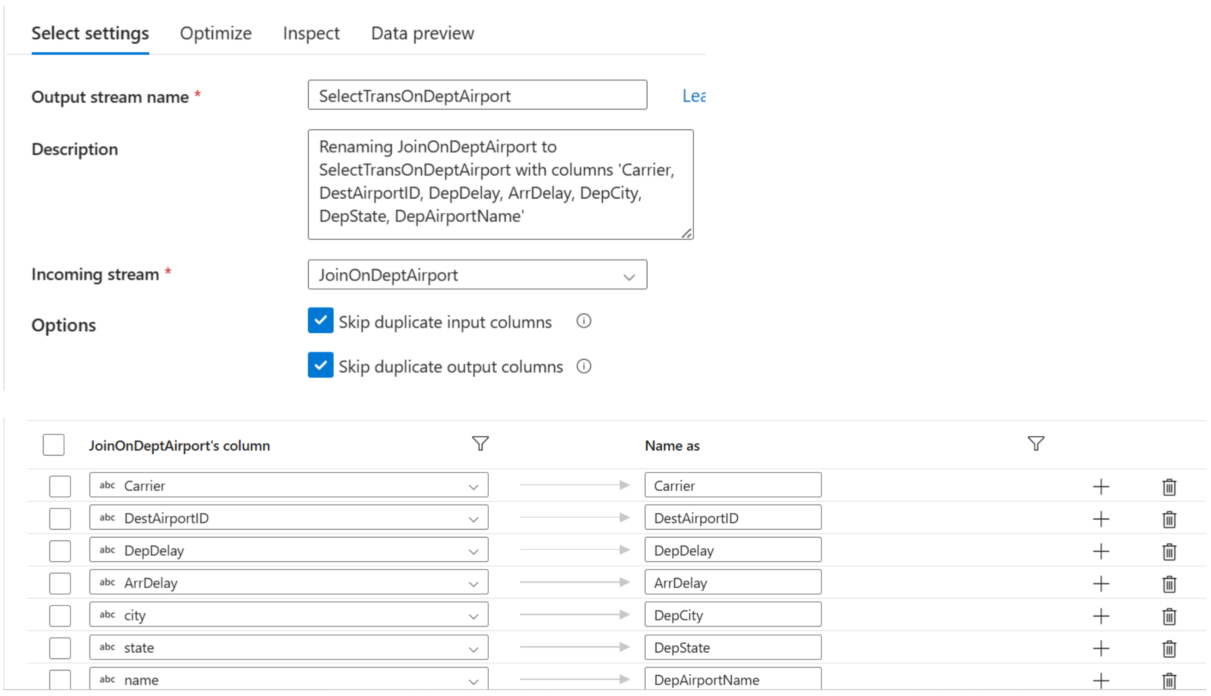
Under it I configured an inner join between the flight data and the airport dimension data. The join condition matches the OriginAirportID from the flight dataset with airport_id from the dimension dataset. This helps combine flight records with airport details.


Join settings | Optimize | Inspect | Data preview

Output stream name *: JoinOnDeptAirport
Description: Inner join on 'DailyFlightSource' and 'AirlineDimSource'
Left stream *: DailyFlightSource
Right stream *: AirlineDimSource
Join type *: Full outer [✓ Inner] Left outer Right outer Custom (cross)
Use fuzzy matching: ☐
Join conditions *:
Left: DailyFlightSource's column: OriginAirportID ==
Right: AirlineDimSource's column: airport_id

Select activity

After joining the datasets, I used a Select transformation to choose relevant columns and rename them for clarity. For example, the airport name column was renamed to DepAirportName, and city/state were renamed to DepCity and DepState.


Select settings | Optimize | Inspect | Data preview

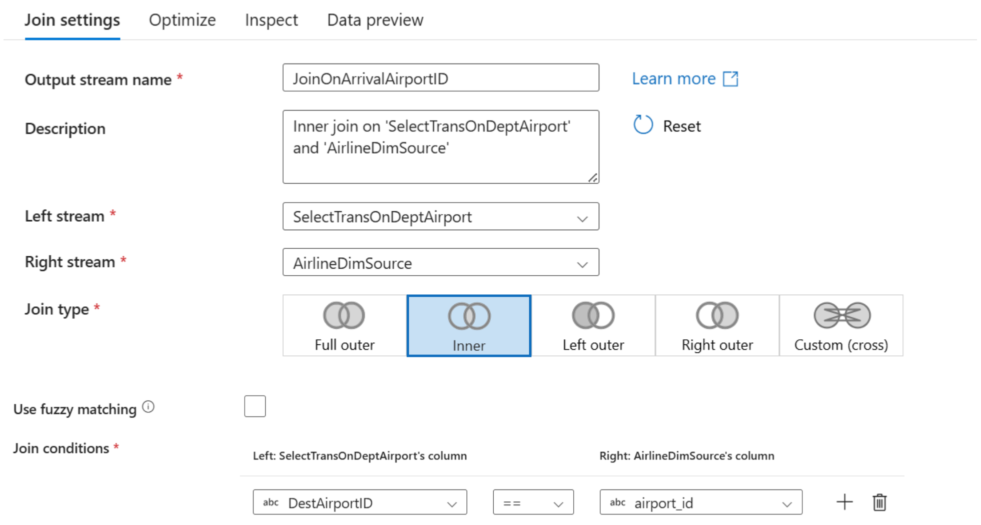
Output stream name *	SelectTransOnDeptAirport
Description	Renaming JoinOnDeptAirport to SelectTransOnDeptAirport with columns 'Carrier, DestAirportID, DepDelay, ArrDelay, DepCity, DepState, DepAirportName'
Incoming stream *	JoinOnDeptAirport
Options	✓ Skip duplicate input columns
	✓ Skip duplicate output columns

JoinOnDeptAirport's column	Name as
Carrier	Carrier
DestAirportID	DestAirportID
DepDelay	DepDelay
ArrDelay	ArrDelay
city	DepCity
state	DepState
name	DepAirportName

Join Activity

I added another inner join to match the destination airport ID from the flight dataset with airport_id from the dimension dataset. This step adds destination airport details to the data.

The second join is needed because the flight dataset has two keys: OriginAirportID and DestAirportID. The first join adds departure airport details, and the second join adds arrival airport details.


Join settings | Optimize | Inspect | Data preview

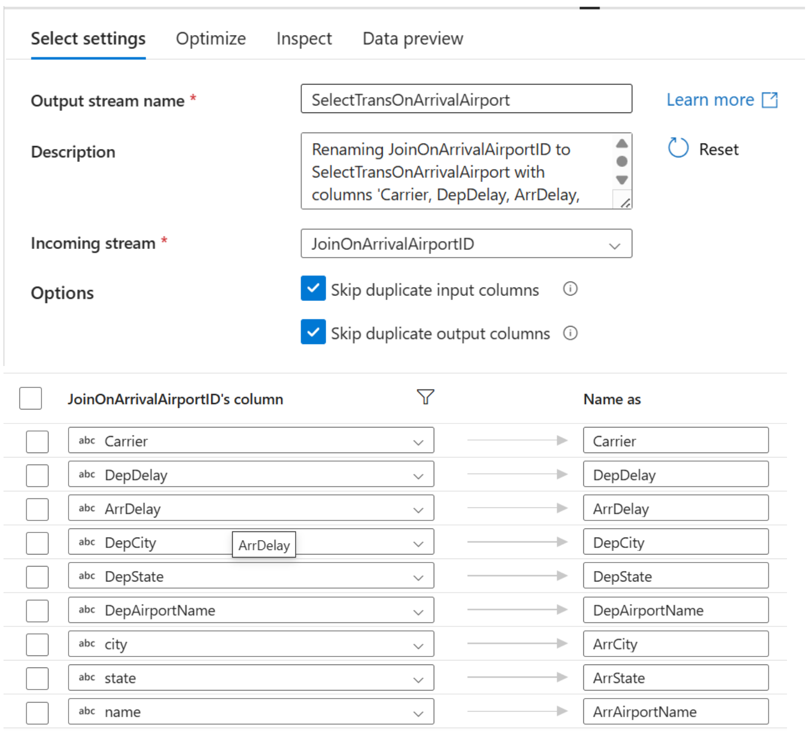
Output stream name *	JoinOnArrivalAirportID
Description	Inner join on 'SelectTransOnDeptAirport' and 'AirlineDimSource'
Left stream *	SelectTransOnDeptAirport
Right stream *	AirlineDimSource
Join type *	Full outer [✓ Inner] Left outer Right outer Custom (cross)
Use fuzzy matching	☐

Join conditions *

Left: SelectTransOnDeptAirport's column		Right: AirlineDimSource's column
DestAirportID	==	airport_id

Select Activity

I used Select to keep only the required columns and rename them for clarity. For example, city/state/name from the arrival airport were renamed to ArrCity, ArrState, and ArrAirportName.



Output stream name * SelectTransOnArrivalAirport

Description: Renaming JoinOnArrivalAirportID to SelectTransOnArrivalAirport with columns 'Carrier, DepDelay, ArrDelay,

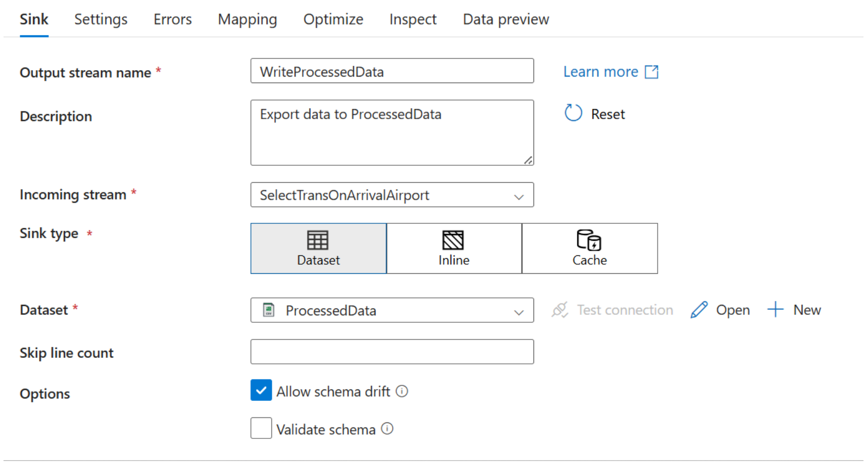
Incoming stream * JoinOnArrivalAirportID

Options: ✓ Skip duplicate input columns; ✓ Skip duplicate output columns

JoinOnArrivalAirportID's column	Name as
abc Carrier	Carrier
abc DepDelay	DepDelay
abc ArrDelay	ArrDelay
abc DepCity	DepCity
abc DepState	DepState
abc DepAirportName	DepAirportName
abc city	ArrCity
abc state	ArrState
abc name	ArrAirportName

Sink Configuration Details

Finally, I used a Sink transformation to write the processed data back to Azure Data Lake in the processed-data folder. I enabled schema drift, auto mapping, and cleared the folder before writing to ensure clean output.



Output stream name * WriteProcessedData

Description: Export data to ProcessedData

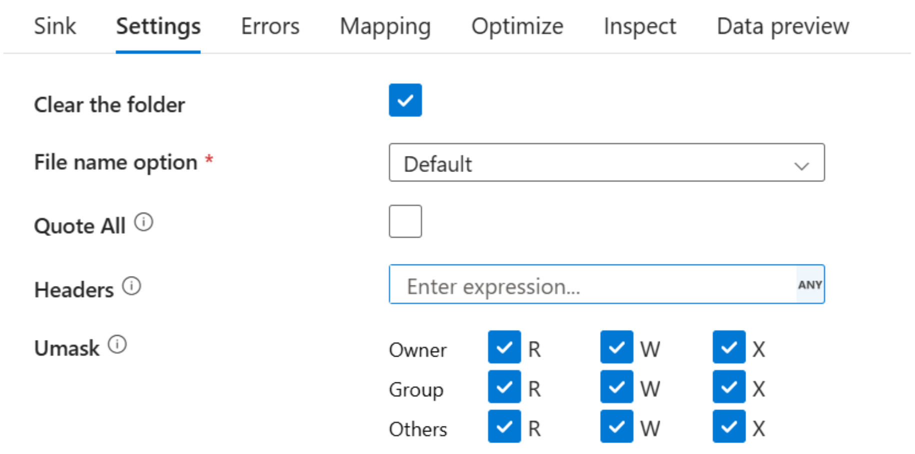
Incoming stream * SelectTransOnArrivalAirport

Sink type * [✓ Dataset] Inline Cache

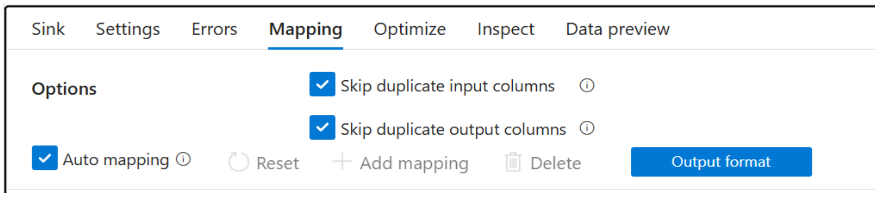
Dataset * ProcessedData

Skip line count

Options: ✓ Allow schema drift; Validate schema


Sink | **Settings** | Errors | Mapping | Optimize | Inspect | Data preview

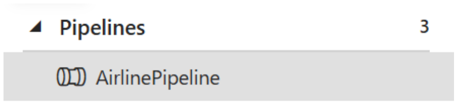
Clear the folder	✓
File name option *	Default
Quote All ⓘ	☐
Headers ⓘ	Enter expression...
Umask ⓘ	Owner ✓ R ✓ W ✓ X
	Group ✓ R ✓ W ✓ X
	Others ✓ R ✓ W ✓ X


Sink | Settings | Errors | **Mapping** | Optimize | Inspect | Data preview

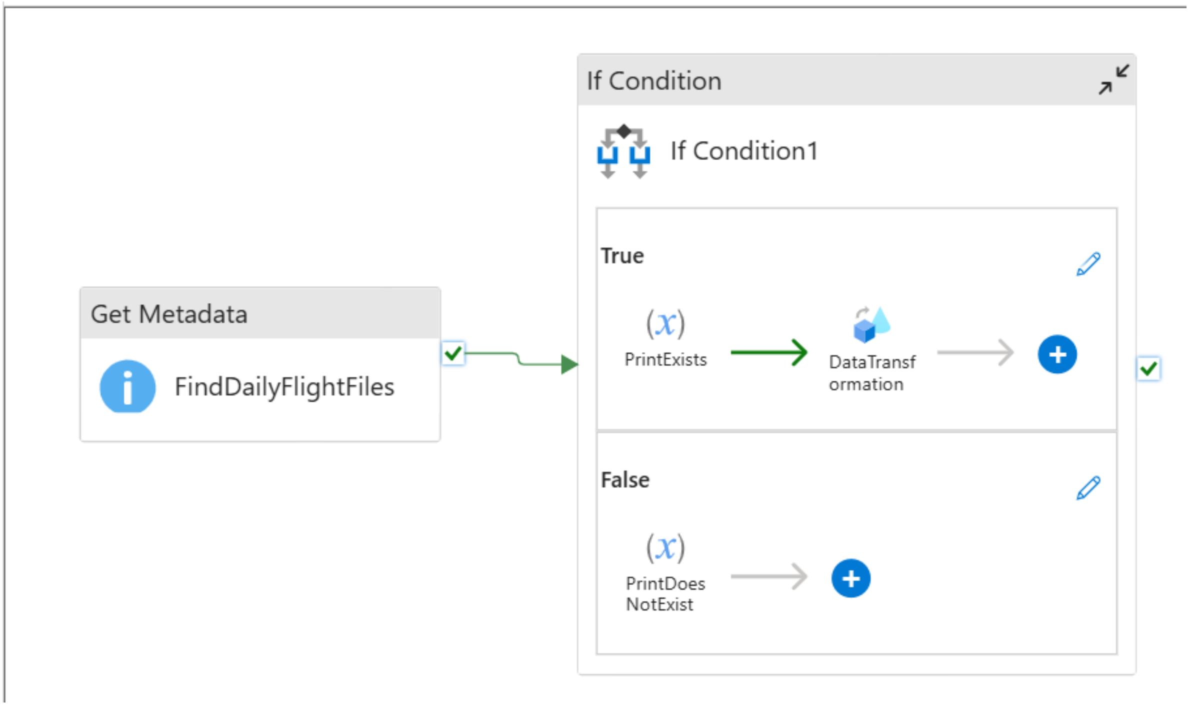
Options
- ✓ Skip duplicate input columns
- ✓ Skip duplicate output columns

✓ Auto mapping ⓘ | Reset | Add mapping | Delete | Output format

Now I started creating pipeline

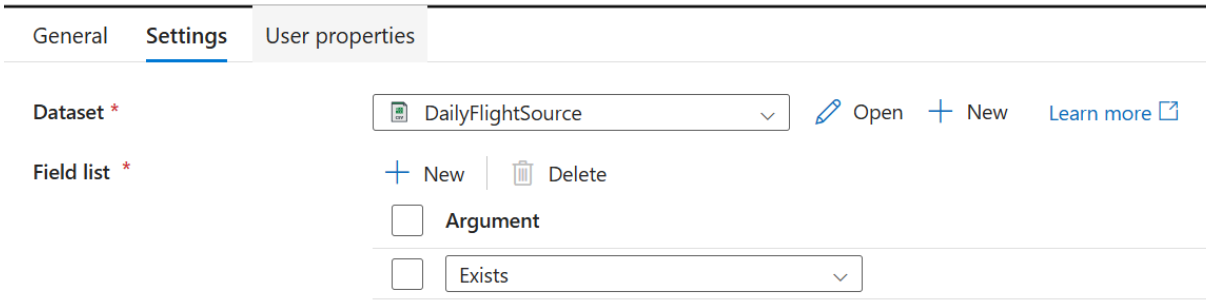

Pipelines 3

AirlinePipeline



FindDailyFlightFiles

I used Get Metadata to check if the source file exists by retrieving the 'Exists' property. This value drives an If Condition activity, ensuring the pipeline only runs when the file is available

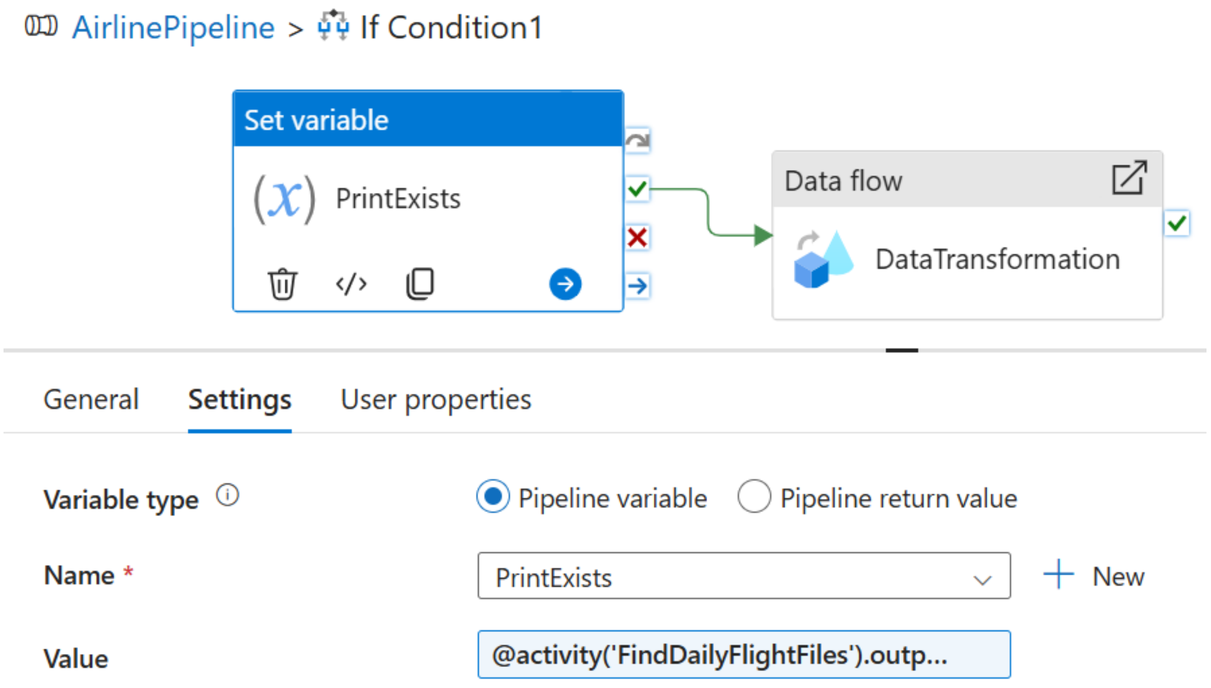

General | Settings | User properties

Dataset * DailyFlightSource | Open | New | Learn more

Field list * | New | Delete

Argument
Exists

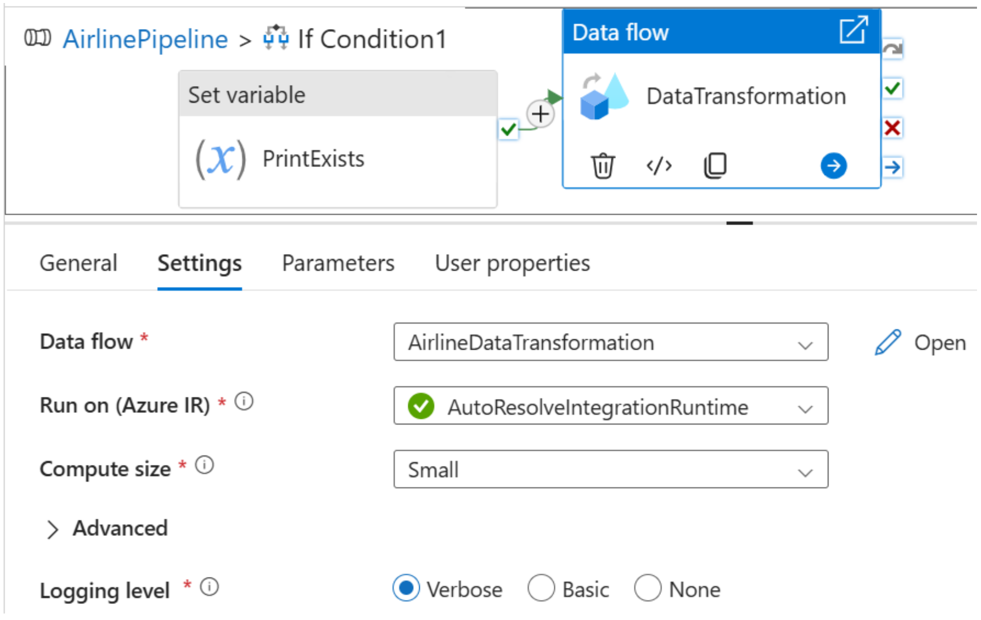
I used a Set Variable activity to store the result of the Get Metadata check using the expression `@activity('FindDailyFlightFiles').output.exists`. This value drives the If Condition logic, ensuring the pipeline only runs when the file exists.



General | Settings | User properties

Variable type	[✓ Pipeline variable] Pipeline return value
Name *	PrintExists
Value	@activity('FindDailyFlightFiles').outp...

I configured the Data Flow activity to run on AutoResolveIntegrationRuntime with a small compute size for cost efficiency. It executes the AirlineDataTransformation flow only when the source file exists, and verbose logging helps in monitoring and troubleshooting.


AirlinePipeline > If Condition1

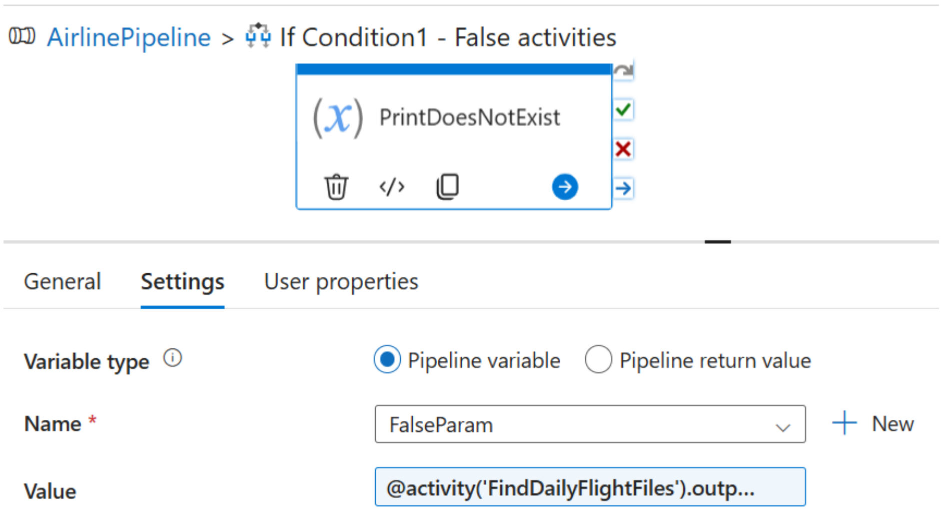
Set variable
PrintExists

Data flow
DataTransformation

General | **Settings** | Parameters | User properties

Data flow *	AirlineDataTransformation
Run on (Azure IR) *	AutoResolveIntegrationRuntime
Compute size *	Small
> Advanced	
Logging level *	[✓ Verbose] Basic None

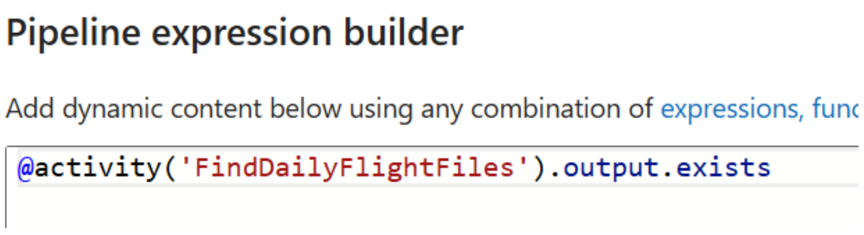
In the False branch, I set a variable using the same dynamic expression to capture the file existence status. This allows me to log or trigger alerts when the source file is missing, making the pipeline more reliable.


AirlinePipeline > If Condition1 - False activities

PrintDoesNotExist

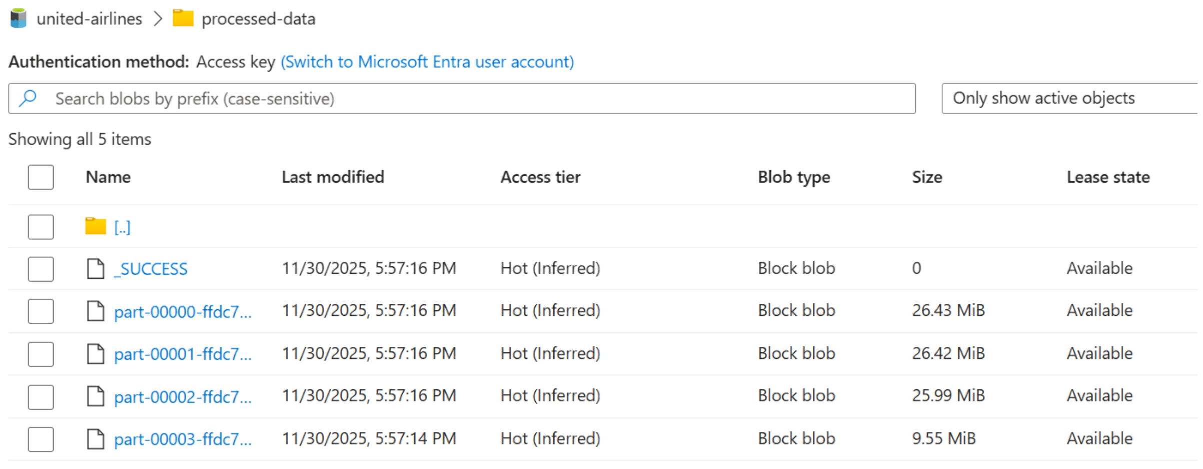
General | **Settings** | User properties

Variable type	[✓ Pipeline variable] Pipeline return value
Name *	FalseParam
Value	@activity('FindDailyFlightFiles').outp...


Pipeline expression builder

Add dynamic content below using any combination of expressions, func

```
@activity('FindDailyFlightFiles').output.exists
```



united-airlines > processed-data

Authentication method: Access key (Switch to Microsoft Entra user account)

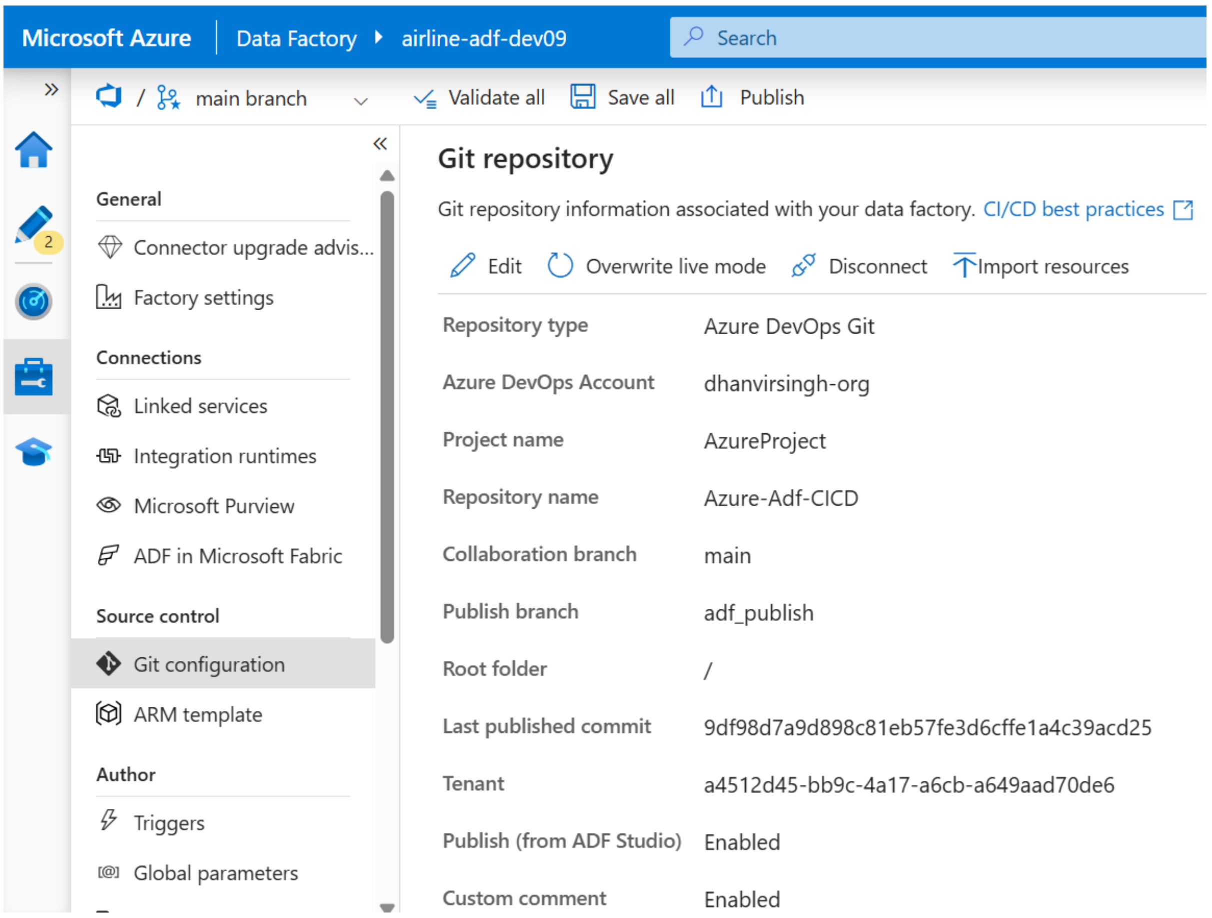
Search blobs by prefix (case-sensitive) | Only show active objects

Showing all 5 items

Name	Last modified	Access tier	Blob type	Size	Lease state
[..]					
_SUCCESS	11/30/2025, 5:57:16 PM	Hot (Inferred)	Block blob	0	Available
part-00000-ffdc7...	11/30/2025, 5:57:16 PM	Hot (Inferred)	Block blob	26.43 MiB	Available
part-00001-ffdc7...	11/30/2025, 5:57:16 PM	Hot (Inferred)	Block blob	26.42 MiB	Available
part-00002-ffdc7...	11/30/2025, 5:57:16 PM	Hot (Inferred)	Block blob	25.99 MiB	Available
part-00003-ffdc7...	11/30/2025, 5:57:14 PM	Hot (Inferred)	Block blob	9.55 MiB	Available

1. Connected Dev Data Factory to Azure DevOps

I linked my Development Data Factory to an Azure DevOps Git repository. This allows all pipelines, datasets, and data flows to be version-controlled. Any changes I make in ADF are first saved in the collaboration branch before publishing.



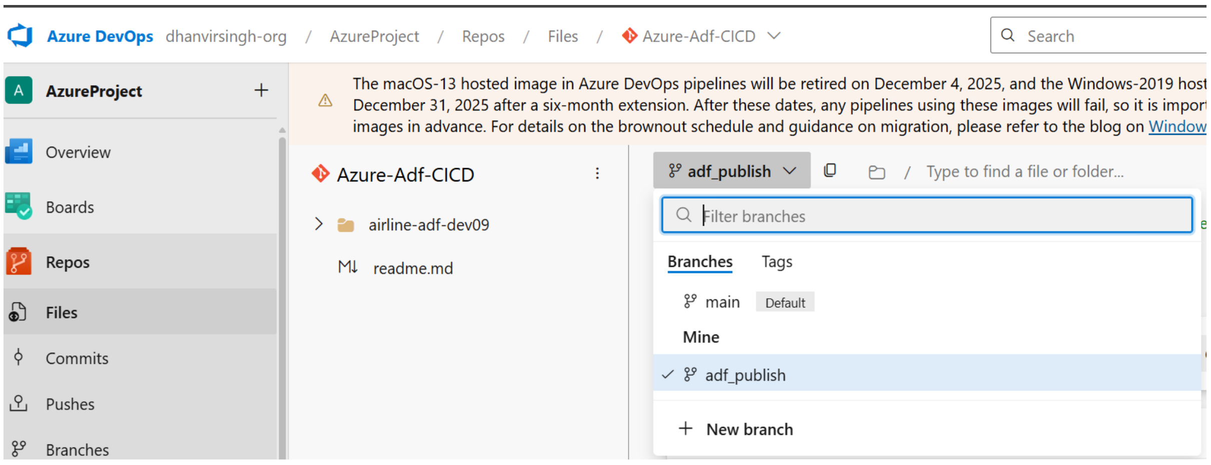
Git repository

Git repository information associated with your data factory. CI/CD best practices

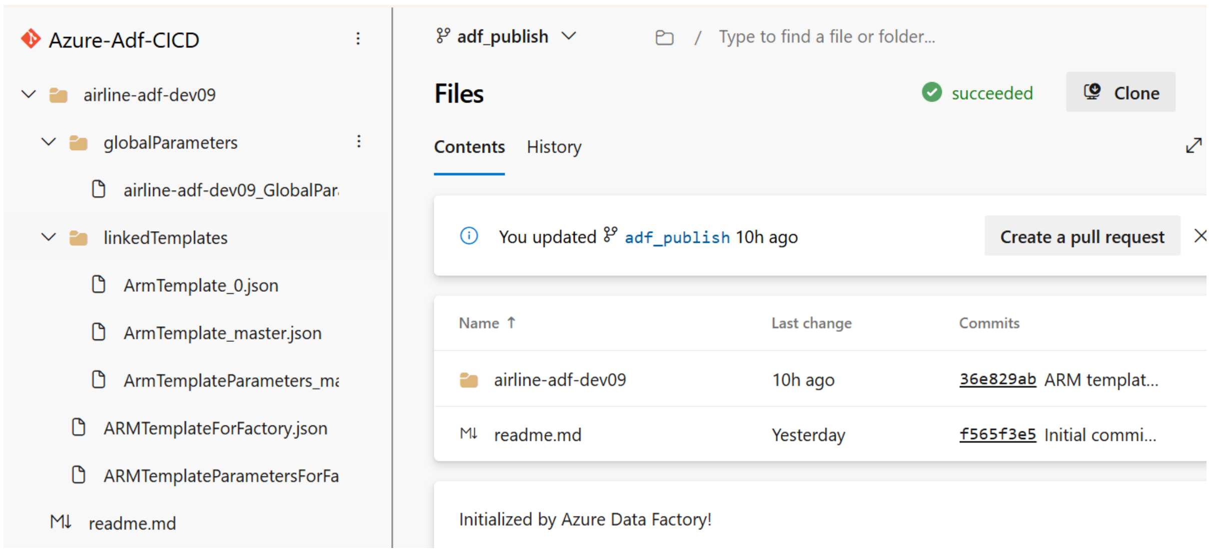
Repository type	Azure DevOps Git
Azure DevOps Account	dhanvirsingh-org
Project name	AzureProject
Repository name	Azure-Adf-CICD
Collaboration branch	main
Publish branch	adf_publish
Root folder	/
Last published commit	9df98d7a9d898c81eb57fe3d6cffe1a4c39acd25
Tenant	a4512d45-bb9c-4a17-a6cb-a649aad70de6
Publish (from ADF Studio)	Enabled
Custom comment	Enabled

I connected my Development Data Factory to an Azure DevOps Git repository for source control. The collaboration branch is `main`, where I make changes, and the publish branch is `adf_publish`, which stores ARM templates generated when I publish. This setup ensures version control and enables CI/CD automation.

When I publish changes from Dev Data Factory, it automatically generates ARM templates and commits them to the `adf_publish` branch in Azure DevOps. These templates represent the entire ADF configuration, including pipelines, datasets, and linked services. This branch acts as the source for the CI/CD pipeline.



The macOS-13 hosted image in Azure DevOps pipelines will be retired on December 4, 2025, and the Windows-2019 host
December 31, 2025 after a six-month extension. After these dates, any pipelines using these images will fail, so it is impor
images in advance. For details on the brownout schedule and guidance on migration, please refer to the blog on Window

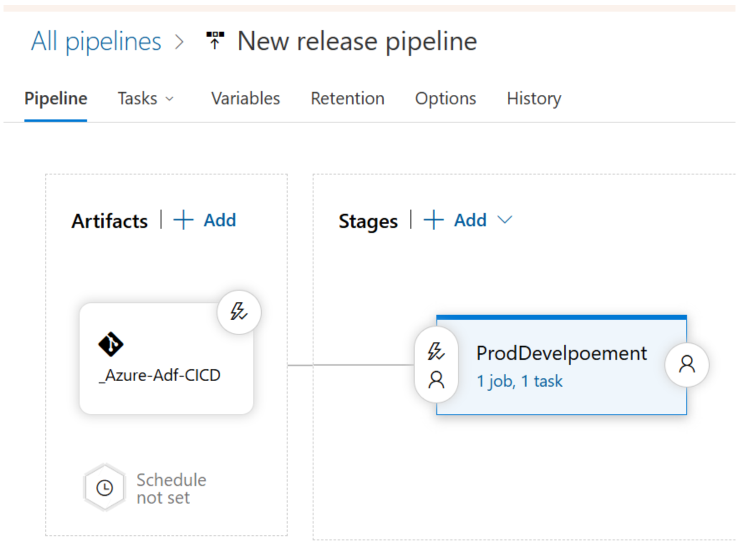


You updated adf_publish 10h ago

Name ↑	Last change	Commits
airline-adf-dev09	10h ago	36e829ab ARM templat...
readme.md	Yesterday	f565f3e5 Initial commi...

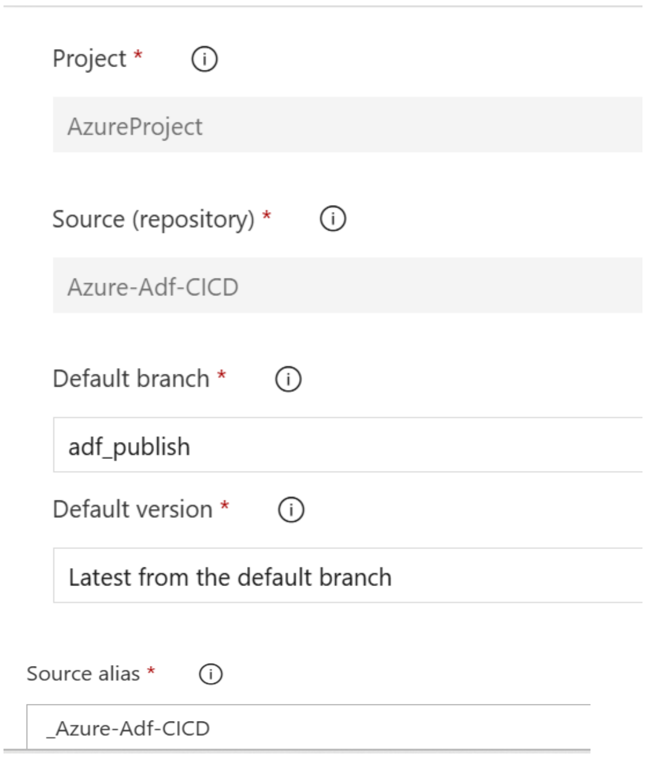
Initialized by Azure Data Factory!

Now I created a Release Pipeline in Azure DevOps that takes the ARM templates from the **artifact** generated during the build stage and deploys them to the Production Data Factory. This **stage** runs automatically after publishing changes in Dev Data Factory, ensuring a smooth and consistent deployment process.

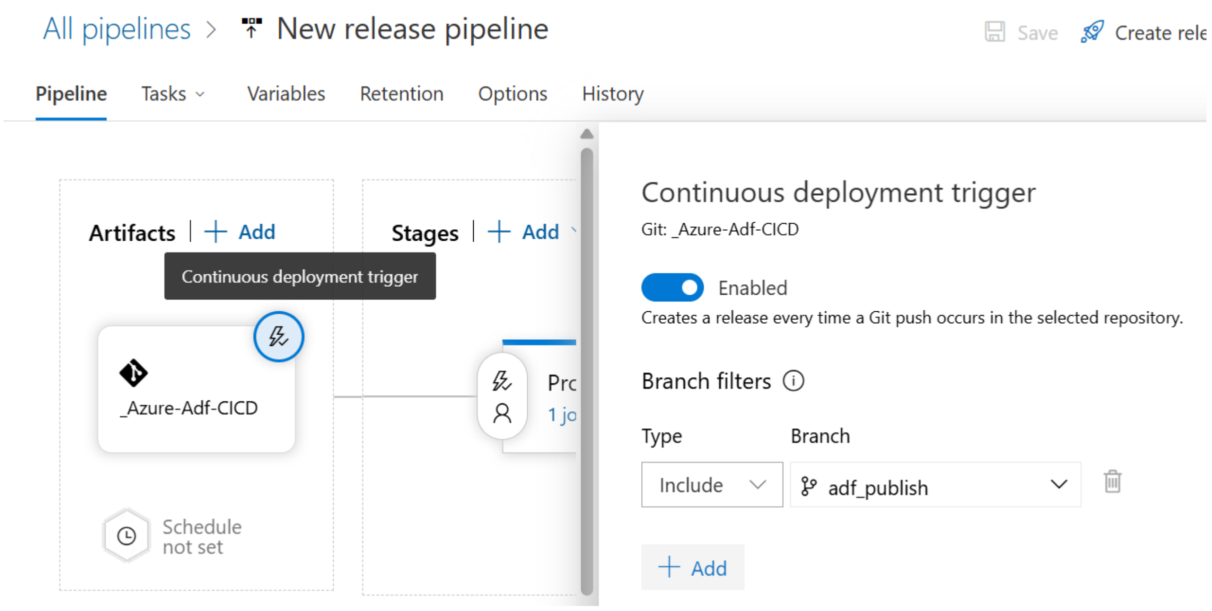


Under Artifact

After adding these settings in the artifact section, the pipeline knows which repository and branch to use. This is important because without linking the artifact, the release pipeline cannot fetch the latest ARM templates. Once this is set, every time I publish from the development Data Factory, the pipeline picks up the new changes and deploys them to production.



Project *	AzureProject
Source (repository) *	Azure-Adf-CICD
Default branch *	adf_publish
Default version *	Latest from the default branch
Source alias *	_Azure-Adf-CICD



Continuous deployment trigger
Git: _Azure-Adf-CICD

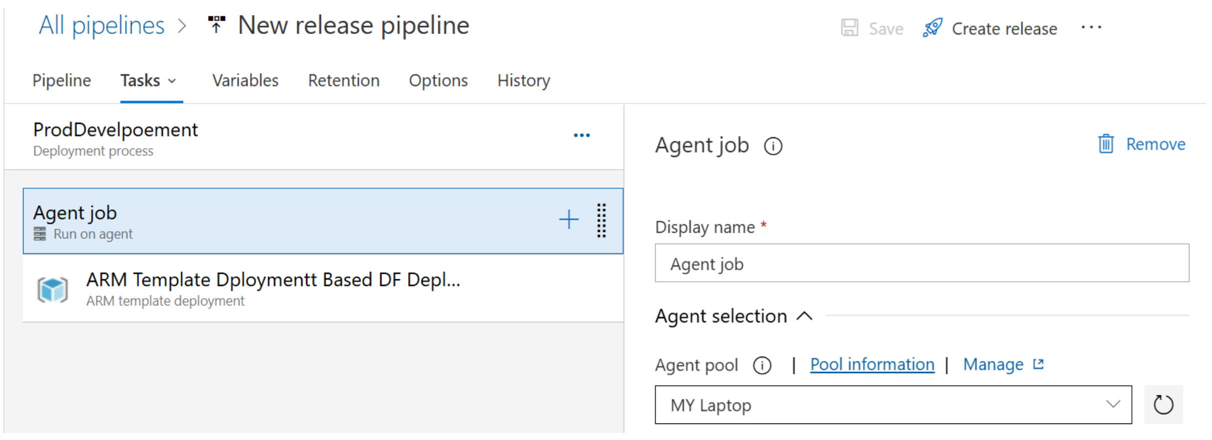
Enabled
Creates a release every time a Git push occurs in the selected repository.

Branch filters

Type	Branch
Include	adf_publish

+ Add

After linking the artifact and enabling the continuous deployment trigger, the pipeline automatically creates a release whenever there is a new commit in the `adf_publish` branch. This means every time I publish changes from the development Data Factory, the updated ARM templates are pushed to `adf_publish`, and the pipeline deploys those changes to production without manual intervention.



ProdDevelpoement
Deployment process

Agent job
Run on agent

ARM Template Dploymentt Based DF Depl...
ARM template deployment

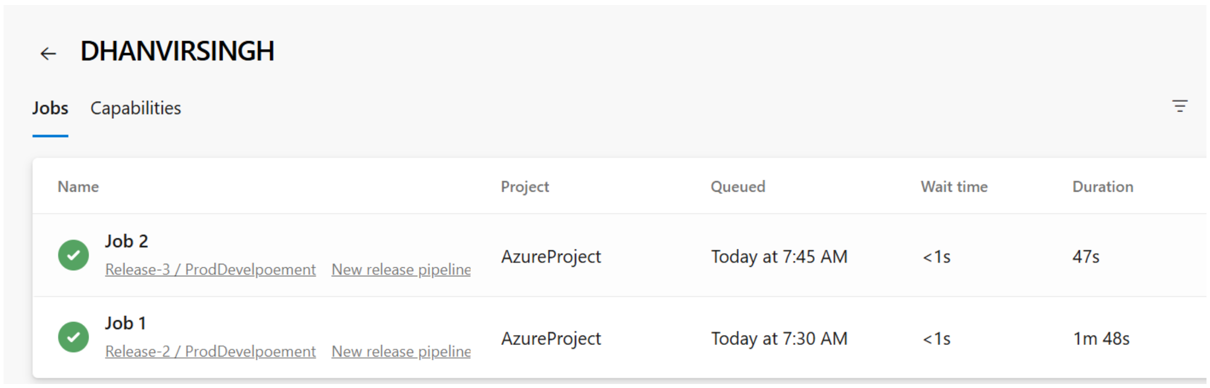
Agent job

Display name *
Agent job

Agent selection

Agent pool
MY Laptop

After configuring the artifact and enabling the trigger, I created a stage called `ProdDevelopement` for production deployment. I am the stage owner, which means I control approvals and deployment settings. Inside this stage, I added an agent job that runs on my selected agent pool. The main task in this job is an ARM Template Deployment, which deploys the Data Factory resources to the production environment using the templates from the `adf_publish` branch.

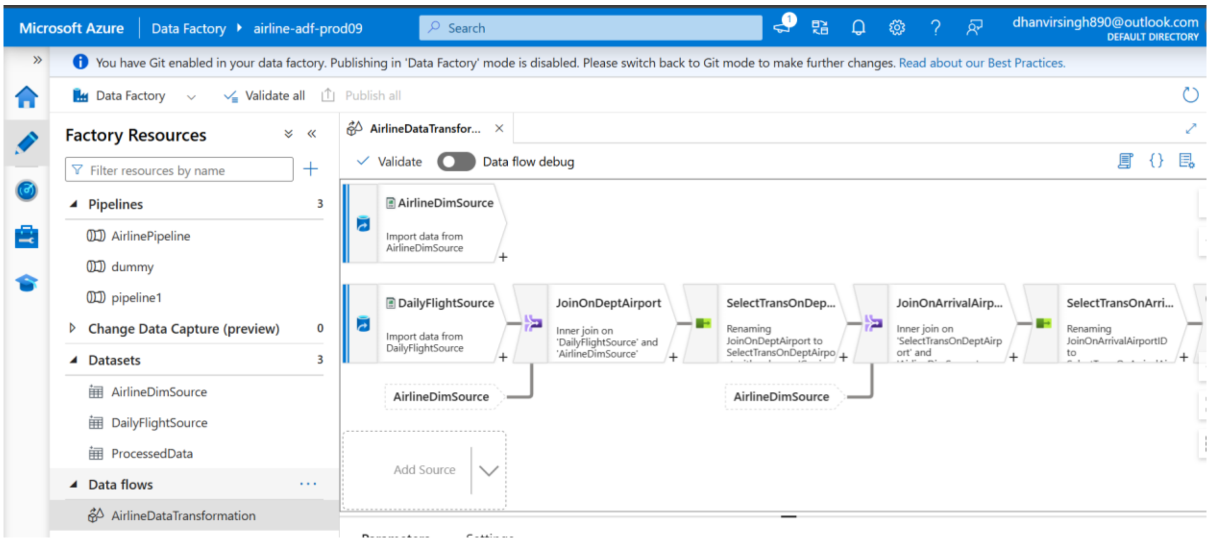


← DHANVIRSINGH

Jobs Capabilities

Name	Project	Queued	Wait time	Duration
Job 2 Release-3 / ProdDevelpoement New release pipeline	AzureProject	Today at 7:45 AM	<1s	47s
Job 1 Release-2 / ProdDevelpoement New release pipeline	AzureProject	Today at 7:30 AM	<1s	1m 48s

These jobs show that my self-hosted agent executed the release pipeline tasks for production deployment. Each job corresponds to a release triggered by changes in the `adf_publish` branch. The agent picked up the job instantly and completed the ARM template deployment task successfully.



Microsoft Azure | Data Factory ▸ airline-adf-prod09

dhanvirsingh890@outlook.com
DEFAULT DIRECTORY

You have Git enabled in your data factory. Publishing in 'Data Factory' mode is disabled. Please switch back to Git mode to make further changes. Read about our Best Practices.

Factory Resources

- Pipelines 3
  - AirlinePipeline
  - dummy
  - pipeline1
- Change Data Capture (preview) 0
- Datasets 3
  - AirlineDimSource
  - DailyFlightSource
  - ProcessedData
- Data flows
  - AirlineDataTransformation

Data Factory
Validate all
Publish all
Search
AirlineDataTransfor...
Validate
Data flow debug
AirlineDimSource
Import data from AirlineDimSource
DailyFlightSource
Import data from DailyFlightSource
JoinOnDeptAirport
Inner join on 'DailyFlightSource' and 'AirlineDimSource'
SelectTransOnDep...
Renaming JoinOnDeptAirport to SelectTransOnDeptAirpo
JoinOnArrivalAirp...
Inner join on 'SelectTransOnDeptAirport' and
SelectTransOnArri...
Renaming JoinOnArrivalAirportID to
AirlineDimSource
AirlineDimSource
Add Source

Whatever I was having in dev data factory is also now in prod data factory