

ADF – Azure Data Factory

It's cloud-based ETL and data integration service provided by Microsoft azure.

What is Azure Data Factory?

- **Azure Data Factory is a cloud-based data integration service provided by the Microsoft azure.**
- **It helps us to manage, schedule and create data pipeline that move and transform data across multiple source and destination.**

Azure Data Factory is a cloud-based data integration service designed to orchestrate (arrange or order) and automate data workflows.

It is used to collect, transform, and deliver data, ensuring that insights are readily accessible for analytics and decision-making.

With its scalable and serverless architecture, ADF can handle workflows of any size—from simple data migrations to complex data transformation pipelines.

ADF stands out for its user-friendly, no-code interface, which allows both technical and non-technical users to build data pipeline easily

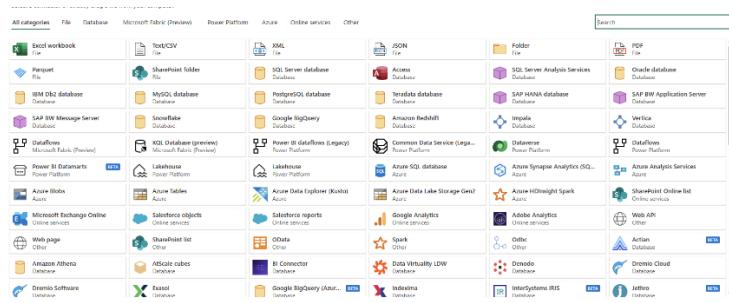
(Data integration is the process of combining data from different sources and providing a unified view of it.)

(Azure Data Factory is a cloud ETL and orchestration service that lets you create, schedule, and monitor data pipelines to move and transform data across on-prem and cloud systems without managing servers).

Features of ADF:

1. Data integration

Azure Data Factory supports integration with over 90 data sources, including cloud-based and on-premises systems. It includes support for SQL databases, NoSQL systems, REST APIs, and file-based data sources, allowing you to unify data workflows regardless of the source or format.



2. No-code pipeline authoring

ADF's drag-and-drop interface simplifies how users create data pipelines. With prebuilt templates, guided configuration wizards, and an intuitive visual editor, even users with no coding expertise can design comprehensive end-to-end workflows.

3. Scheduling

Azure Data Factory's scheduling tools offer workflow automation. Users can set up triggers based on specific conditions, such as a file's arrival in cloud storage or scheduled time intervals. These scheduling options eliminate the need for manual interventions and ensure workflows are executed consistently and reliably.

New trigger

Name *
trigger4

Description

Type *
 Schedule Tumbling window Event

Start date *
10/29/2020 3:30 PM

Time zone *
Pacific Time (US & Canada) (UTC-8)

Recurrence *
Every 15 Minute(s)

Specify an end date

End On *
10/31/2020 6:30 PM

Annotations
+ New

Name

Activated *
 Yes No

[Azure Data Factory: A Complete Guide for Beginners | DataCamp](#)

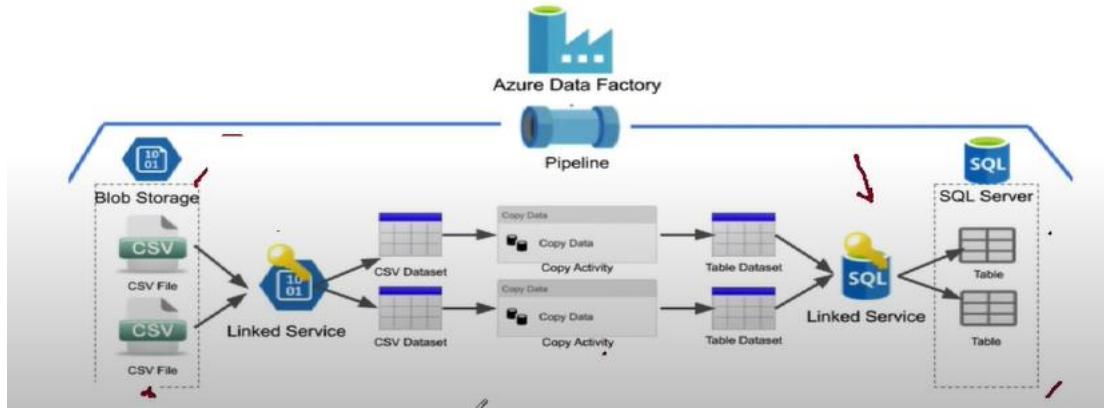
Features:

1. Data movement – copy data from on-premises system, SaaS apps, and cloud storages (e.g., SQL server, Oracle, Blob Storage, Data Lake).
2. Data transformation – supports:
Build-in transformation with Mapping Data Flows (low-code / no-code)
Running compute service like databricks, synapse or custom code.
3. Orchestration (order) & Monitoring – manage workflows, schedule pipelines, and monitor execution visually.
4. Integration runtime (IR) – provides the compute infrastructure for data movement and transformation.
5. Serverless – you don't need to manage servers; you pay only for the pipelines and runtime you use.

Common use cases:

- Migrating data from on-premises databases to the cloud.
- Building ETL/ELT pipelines for data warehousing.
- Moving data into azure data lake or synapse analytics for reporting.
- Automating data workflows for AI/ML model training.

- Real-time data ingestion with event triggers.



Functions in Azure Data Factory (ADF)

1. String Functions

Used to work with text.

Function	Description	Example
concat(a,b,...)	Joins multiple strings.	@concat('file_', pipeline().RunId, '.csv') → "file_1234.csv"
substring(string, start, length)	Extracts part of a string.	@substring('DataFactory', 0, 4) → "Data"
length(string)	Returns length of string.	@length('ADF') → 3
toLowerCase(string)	Lowercase.	@toLowerCase('ADF') → "adf"
toUpperCase(string)	Uppercase.	@toUpperCase('adf') → "ADF"
trim(string)	Removes spaces.	@trim(' hello ') → "hello"
replace(string, old, new)	Replaces text.	@replace('data_factory','_','') → "datafactory"
split(string, delimiter)	Splits into array.	@split('a,b,c','.') → ['a','b','c']
guid()	Generates unique ID.	@guid() → "7e8d...a2b"

2. Collection Functions (Arrays & Objects)

Work with arrays, objects, JSON.

Function	Description	Example
contains(collection, value)	Checks if value exists.	@contains(['a','b'],'b') → true

empty(collection)	Checks if empty.	@empty([]) → true
first(array)	First element.	@first(['x','y']) → "x"
last(array)	Last element.	@last(['x','y']) → "y"
intersection(arr1,arr2)	Common values.	@intersection([1,2],[2,3]) → [2]
union(arr1,arr2)	Combine unique values.	@union([1,2],[2,3]) → [1,2,3]
join(array, delimiter)	Join into string.	@join(['a','b'], '-') → "a-b"
length(array)	Array length.	@length([1,2,3]) → 3
addProperty(obj,key,value)	Adds property.	@addProperty(variables('obj'),'newKey','val')
removeProperty(obj,key)	Removes property.	@removeProperty(obj,'key')

3. Logical Functions

Used in conditions (If, Switch, Filter, etc.).

Function	Description	Example
and(a,b,...)	True if all true.	@and>equals(1,1),greater(2,1)) → true
or(a,b,...)	True if at least one true.	@or>equals(1,2),equals(2,2)) → true
not(expr)	Negates.	@not>equals(1,1)) → false
equals(a,b)	Checks equality.	@equals(5,5) → true
greater(a,b)	Greater than.	@greater(5,3) → true
less(a,b)	Less than.	@less(2,3) → true
if(cond, trueVal, falseVal)	Inline if.	@if>equals(2,2),'Yes','No') → "Yes"
coalesce(a,b,...)	First non-null value.	@coalesce(null,'ADF','DF') → "ADF"

4. Conversion Functions

Convert between data types.

Function	Description	Example
string(value)	Convert to string.	@string(123) → "123"
int(value)	Convert to integer.	@int('25') → 25
float(value)	Convert to float.	@float('12.34') → 12.34
bool(value)	Convert to boolean.	@bool('true') → true

<code>json(string)</code>	Convert to JSON.	<code>@json('{"name":"abc"})</code> → object
<code>base64(string)</code>	Encode base64.	<code>@base64('ADF')</code> → "QURG"
<code>base64ToString(string)</code>	Decode base64.	<code>@base64ToString('QURG')</code> → "ADF"

5. Date & Time Functions

Handle datetime.

Function	Description	Example
<code>utcNow()</code>	Current UTC datetime.	<code>@utcNow()</code> → "2025-09-02T06:30:00Z"
<code>addDays(datetime,n)</code>	Add days.	<code>@addDays(utcNow(),5)</code>
<code>addHours(datetime,n)</code>	Add hours.	<code>@addHours(utcNow(),2)</code>
<code>addMinutes(datetime,n)</code>	Add minutes.	<code>@addMinutes(utcNow(),30)</code>
<code>addMonths(datetime,n)</code>	Add months.	<code>@addMonths('2025-01-01',1)</code> → Feb 1
<code>addSeconds(datetime,n)</code>	Add seconds.	<code>@addSeconds(utcNow(),10)</code>
<code>subtractFromTime(datetime,n,unit)</code>	Subtract time.	<code>@subtractFromTime(utcNow(),2,'Day')</code>
<code>formatDateTime(datetime,format)</code>	Format datetime.	<code>@formatDateTime(utcNow(),'yyy-MM-dd')</code>
<code>dayOfWeek(datetime)</code>	Returns day (0=Sunday).	<code>@dayOfWeek('2025-09-02')</code> → 2
<code>dayOfMonth(datetime)</code>	Day of month.	<code>@dayOfMonth('2025-09-02')</code> → 2

6. Math Functions

Used for calculations.

Function	Description	Example
<code>add(a,b)</code>	Add.	<code>@add(5,3)</code> → 8
<code>sub(a,b)</code>	Subtract.	<code>@sub(5,3)</code> → 2
<code>mul(a,b)</code>	Multiply.	<code>@mul(2,3)</code> → 6
<code>div(a,b)</code>	Divide.	<code>@div(6,3)</code> → 2
<code>mod(a,b)</code>	Modulus.	<code>@mod(10,3)</code> → 1
<code>min(a,b,...)</code>	Minimum.	<code>@min(3,5,1)</code> → 1

<code>max(a,b,...)</code>	Maximum.	<code>@max(3,5,1) → 5</code>
<code>rand(min,max)</code>	Random number.	<code>@rand(1,100) → e.g., 42</code>
<code>ceiling(value)</code>	Round up.	<code>@ceiling(2.3) → 3</code>
<code>floor(value)</code>	Round down.	<code>@floor(2.9) → 2</code>
<code>round(value,decimals)</code>	Round with decimals.	<code>@round(2.567,2) → 2.57</code>

7. Pipeline Functions

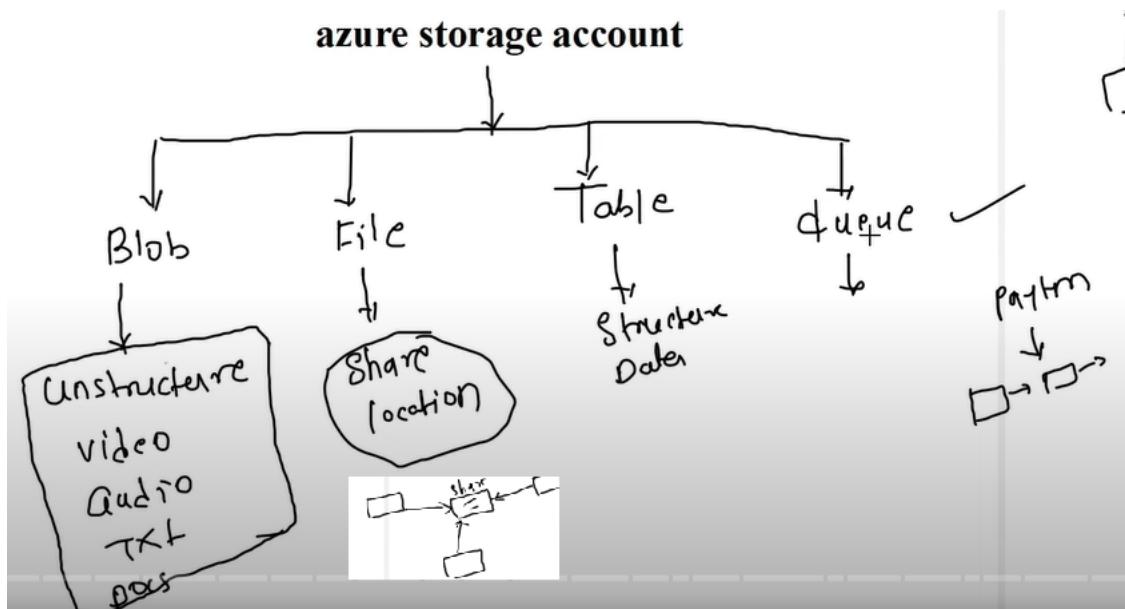
Access metadata inside pipelines.

Function	Description	Example
<code>pipeline().DataFactory</code>	Returns DF name.	"MyADF"
<code>pipeline().Pipeline</code>	Returns pipeline name.	"MyPipeline"
<code>pipeline().RunId</code>	Unique pipeline run ID.	"12345-abc"
<code>pipeline().TriggerTime</code>	Triggered datetime.	"2025-09-02T06:30:00Z"
<code>activity('ActivityName').output</code>	Output of activity.	<code>@activity('Lookup1').output.firstRow.id</code>
<code>trigger().outputs</code>	Trigger values.	From event/schedule trigger
<code>parameters('paramName')</code>	Pipeline parameter.	<code>@parameters('inputFile')</code>
<code>variables('varName')</code>	Pipeline variable.	<code>@variables('counter')</code>

Activity dependencies

Condition	Meaning	Typical Use Case
On Success	Next activity runs only if the previous activity succeeds (status = Succeeded).	Run a data transformation only if the copy activity was successful.
On Failure	Next activity runs only if the previous activity fails (status = Failed).	Send an alert email or trigger a rollback when something fails.
On Skip	Next activity runs if the previous activity was skipped (not executed at all, e.g., due to an If Condition or dependency rule).	Log or handle skipped branches in complex pipelines.

Condition	Meaning	Typical Use Case
On Completion (the “another” one)	Next activity runs regardless of success, failure, or skip.	Always clean up resources, update a log table, or send a “job finished” notification.



ADLS stands for **Azure Data Lake Storage**.

- It's a **cloud-based storage service** from Azure, specially designed for **big data analytics and processing**.
- Think of it like **Azure Blob Storage**, but **optimized for analytics, hierarchical structure, and security**.
- Is a scalable and secure data lake service designed to store and analyze large volumes of structured and unstructured data.

Types of ADLS

Earlier there were two generations, but now everything is under **ADLS Gen2** (built on top of Blob storage).

1. **ADLS Gen1** – older, separate service (now retired).
2. **ADLS Gen2** – combines the best of Blob storage + Data Lake features.

ADLS Gen2 = Azure Blob Storage + ADLS Gen1

1. Hadoop compatible:
ADLS Gen1 is based on Hadoop Distributed File System (HDFS),
ADLS Gen2 is built on top of Azure Blob Storage.
2. Namespace and hierarchy:
ADLS Gen1 has a flat namespace.

ADLS Gen2 has a hierarchical namespace. ADLS Gen2 uses directories to organize data, which provides a more efficient way to manage large amounts of data.

3. Security:

ADLS Gen1 uses access control lists (ACLs) for authorization.

ADLS Gen2 uses role-based access control (RBAC) and Azure Active Directory (AAD) authentication.

4. Data Lake Analytics:

ADLS Gen1 integrates with Azure Data Lake Analytics, a cloud-based analytics service that can process big data using U-SQL, a language that combines SQL and C#.

ADLS Gen2 also supports Data Lake Analytics, but it uses Apache Spark instead of U-SQL.

ADLS Gen 2 supports access methods such as REST APIs, .NET, Java, and Hadoop Distributed File System (HDFS).

ADF is like a pipeline tool in Azure that helps us **extract data from multiple sources, transform it, and load it into target systems like Data Lake or Synapse**.

It acts as an **orchestrator** – it doesn't always process the data itself, but it makes sure the right data moves and gets processed at the right time using services like Databricks, SQL, or Synapse.”

One-liner:

- **ADF = Azure's ETL + Orchestration tool for building and managing data pipelines.**
-

Core components of Azure Data Factory

1. **Pipeline** – the workflow or flowchart or container that tells ADF how to move and transform data.
In ADF, that whole workflow (Extract → Transform → Load → Report) = **Pipeline**.
2. **Activity** – the individual steps [tasks inside pipeline] (copy, transform, etc)
3. **Data sets** – represent the data structures (e.g., SQL table, CSV file, blob)
4. **Linked services** – connections to external resources (like storage accounts, APIs)
5. **Triggers** – define when the pipeline should run (schedule, event or manual).

Example Scenario

Let's say pipeline needs to:

1. Copy data from an **on-prem SQL database** to an **Azure Data Lake**.
2. Transform the data using a **Databricks notebook**.
3. Load the cleaned data into **Azure Synapse Analytics** for reporting.

In ADF, this would look like:

- **Pipeline** = *Daily Sales Data Pipeline*
 - **Activities**:
 - Copy activity (SQL → Data Lake)
 - Notebook activity (Databricks transformation)
 - Copy activity (Data Lake → Synapse)
 - **Trigger**: Schedule at 1 AM daily
-

Pipeline

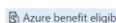
A data factory might have one or more pipelines. A pipeline is a logical grouping of activities that performs a unit of work. [we are only able to create 40 pipeline at max in single Data Factory]

Activity

Activities represent a processing step in a pipeline. (copy activity to copy data store)

Search for “data factory”

The screenshot shows the Azure Marketplace search interface. At the top, there is a search bar with the text "data factory". Below the search bar are several filter options: "Pricing : All", "Operating System : All", "Publisher name : All", "Azure benefit eligible only" (unchecked), and "Azure services only" (unchecked). A button labeled "View suggestions" is also present. Below the filters, a message says "New! Get AI-generated suggestions for 'data factory'" with a "View suggestions" button. The main area displays search results for "data factory", showing 1 to 20 of 121 results. The first result is highlighted with a red box: "Data Factory" by Microsoft, which is categorized as an "Azure Service". Other results include "Data Factory Monitoring Dashboard" by In2Intel (Azure Application) and "Factory Operations Agent in Azure AI (preview)" by Microsoft (Azure Service).

Data Factory  Add to Favorites
Microsoft | Azure Service
★ 3.6 (608 ratings)
 Azure benefit eligible
Subscription Plan
Azure for Students Data Factory **Create**

Overview Plans Usage Information + Support Ratings + Reviews

Integrate data silos with Azure Data Factory, a service built for all data integration needs and skill levels. Easily construct ETL and ELT processes code-free within the intuitive visual environment, or write your own code. Visually integrate data sources using more than 90+ naturally built and maintenance-free connectors at no added cost. Focus on

Select resource group

Create Data Factory ...

One-click to create data factory with sample pipeline and datasets. [Try it](#)

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * 

Azure for Students 

Resource group * 

 Select existing...

adf_assignment

bigdata

databricks-rg-azure_practice-gugdxleldrzy

databricks-rg-vanilla_workspace-cona2royf5f5k

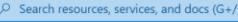
databricks-rg-vanilla_workspace-uldnqzwakxst4

[Previous](#)

[Next](#)

Review + create

Data factory is created

Microsoft Azure  Search resources, services, and docs (G+)

Home > Microsoft.DataFactory-20250827124701 | Overview ...

 Deployment

>Your deployment is complete

Deployment name : Microsoft.DataFactory-2025082... Start time : 27/08/2025, 14:33:27
Subscription : Azure for Students Correlation ID : 1a037a8a-b34b-4593-a8f0-f48...
Resource group : bigdata

 Deployment details

 Next steps

Go to resource

Give feedback  Tell us about your experience with deployment

Home > Microsoft.DataFactory-20250827124701 | Overview >

bigdata Resource group

Are there any alerts fired for this resource group? What are the best practices for managing this resource group? +1

Search Overview

+ Create Manage view Delete resource group Refresh Export to CSV Open query Assign tags

Essentials

Resources Recommendations (1)

Filter for any field... Type equals all Location equals all Add filter

Showing 1 to 2 of 2 records. Show hidden types No grouping List view

Name	Type	Location
createfirstdatafactory	Data factory (V2)	South India
stbigdata	Storage account	East US

Activity log Access control (IAM) Tags Resource visualizer Events Settings Cost Management Cost analysis Cost alerts (preview)

Home > Microsoft.DataFactory-20250827124701 | Overview > **bigdata** >

createfirstdatafactory Data factory (V2)

Show me integration runtime metrics for this Data factory (V2). List triggers associated with this Data factory

Search Delete

Overview

Activity log Access control (IAM) Tags Diagnose and solve problems Resource visualizer Settings Getting started

Resource group (move) **bigdata** Status Succeeded Location South India Subscription (move) Azure for Students Subscription ID b279b1f8-3e6d-4236-af0e-0dd64e534656

Subscription ID b279b1f8-3e6d-4236-af0e-0dd64e534656



Azure Data Factory Studio

→ Launch studio

Quick Starts Tutorials Template Gallery Training Modules

createfirstdatafactory - Microsoft Edge

adf.azure.com/en/home/factory=%2Fsubscriptions%2Fb279b1f8-3e6d-4236-af0e-0dd64e534656%2FresourceGroups%2Fbigdata%2Fpr...

Microsoft Azure Data Factory > createfirstdatafactory Set up code repository

Azure Data Factory allows you to configure a Git repository with either Azure DevOps or GitHub. Git is a version control system that allows for easier change tracking and collaboration. Learn more

Set up code repository

Data factory

createfirstdatafactory

New Pipeline Power Query Orchestrate Transform data Configure SSIS

Power flow Dataset

scale once or

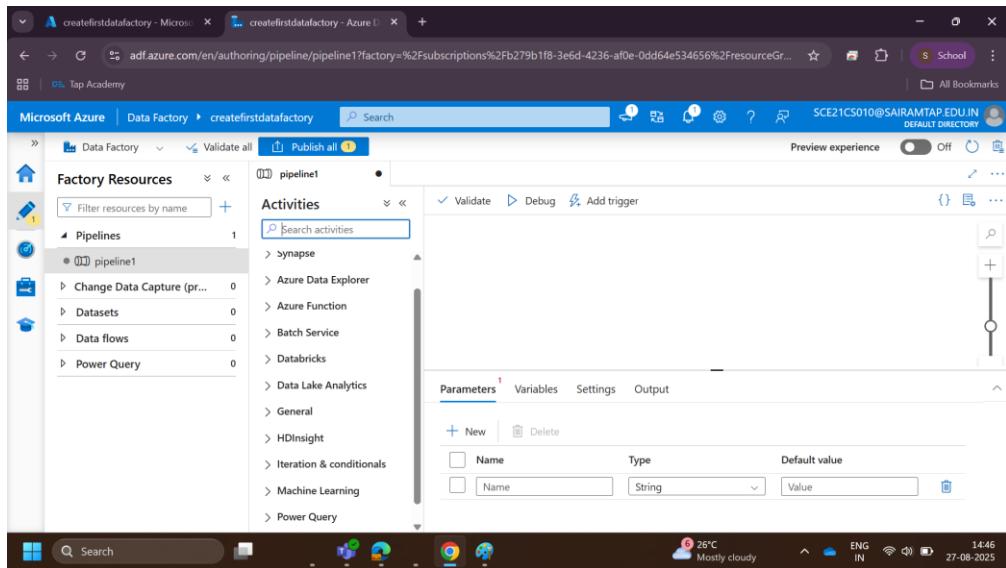
Orchestrate Code-free data pipelines.

Transform data Transform your data using data flows.

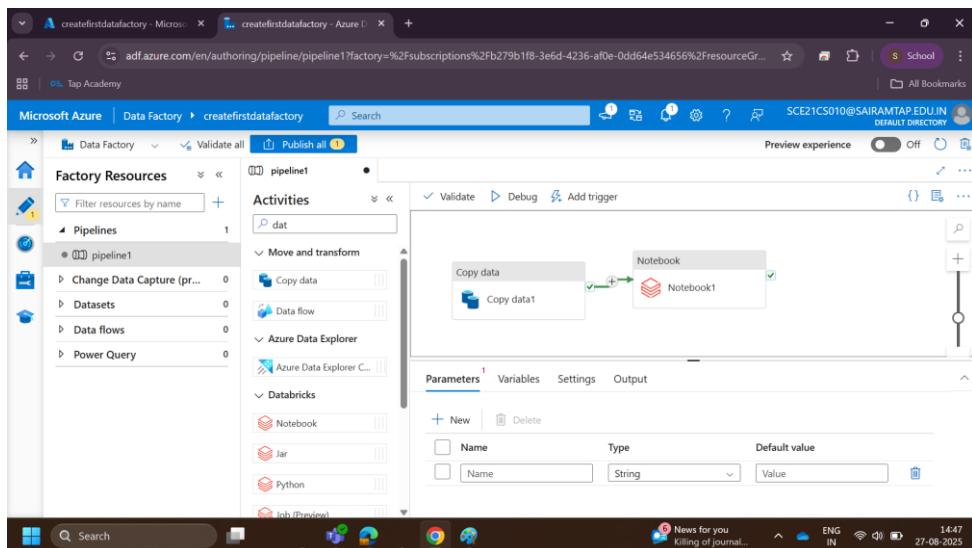
Configure SSIS Manage & run your SSIS packages in the cloud.

Recent resources

27°C Mostly cloudy ENG IN 14:41 27-08-2025



Drag and drop the activity



Linked services & data sets

Linked services are much like connection strings, which define the connection information that's needed for data factory to connect to external resources.

It defines how to connect to a data source.

Datasets represent data structures within the data stores, which simply point to or reference the data you want to use in your activities.

Think of it as a **pointer to actual data** inside the data source.

It represents the **structure & location** of the data you want to use.

For example,

- **Linked Service** → Tells ADF how to connect.
- **Dataset** → Tells ADF what data inside that source to use.

Task: Copy customer data from SQL Database to Azure Blob.

1. **Linked Service (SQL)** → Connection details to SQL DB.
2. **Dataset (SQL)** → Points to *dbo.Customers* table.
3. **Linked Service (Blob)** → Connection details to Blob Storage.
4. **Dataset (Blob)** → Points to *customers.csv* in container *raw-data*.
5. **Copy Activity** → Moves data from Dataset (SQL) → Dataset (Blob).

Click “New” to add the source data

The screenshot shows the Microsoft Azure Data Factory interface. The left sidebar has a tree view with nodes like 'General', 'Connections' (which is selected and highlighted with a red box), 'Factory settings', 'Integration runtimes', 'Microsoft Purview', 'ADF in Microsoft Fabric', 'Source control', 'Author', and 'Triggers'. The main content area is titled 'Linked services' and contains the message 'No linked service to show'. Below this message is a button labeled 'Create linked service'.

Trigger

A trigger defines when and how a pipeline should run.

Pipelines don't run automatically — you must either manually run them or use a trigger.

- Triggers – you can execute your pipeline.
- It determines when a pipeline execution needs to be kicked off.
- Multiple triggers can kick off a single pipeline, or a single trigger can kick off multiple pipelines.
- Pipelines and triggers have a many-to-many relationship (except for the tumbling window trigger).

The screenshot shows the Azure Data Factory interface with the 'Triggers' section selected. A 'New trigger' dialog is open, prompting for a name ('trigger1'), a description, and a type ('Schedule'). Under 'Schedule', the 'Tumbling window' option is selected, and the interval is set to 'Every 15 Minute(s)'. There are also options for 'Storage events' and 'Custom events'.

Types of triggers

- Schedule Trigger
- tumbling window Trigger
- Event Trigger or blob triggers
- custom events Trigger

Types of Triggers in ADF

1. Schedule Trigger

- Runs a pipeline at **specific times** based on a schedule you define.
- Example: Run every day at **9 AM**, or every **30 minutes**.
- It is **time-based only**.
- Doesn't care if previous runs are completed or not.
- No dependency between runs (they are **independent**).
- Best suited for:
 - Simple, fixed-time schedules.
 - Pipelines that do not depend on previous runs or data intervals.
- Example: Run a data backup job at **midnight daily**.

2. Tumbling Window Trigger

This trigger runs a pipeline on a periodic time interval from a specific start time.

- Runs a pipeline at a regular interval (window), but each run represents a specific, non-overlapping slice of time.
- Example:

Process data for each **hour**, where

10:00–11:00 is one window,
11:00–12:00 is the next window.

- Ensures exactly once execution per window (no overlaps).
- If a run for a window fails, the next window will wait until the failed one is re-run and succeeded.
- Supports dependency chaining (later runs can depend on the success of earlier windows).
- Useful for **incremental data processing**.
- You can **re-run** a specific failed window.
- Best suited for:
 - Time-partitioned data ingestion (hourly/daily batch data).
 - Scenarios where no data window should be skipped.
- Example: Process log files arriving **hourly**. If 2–3 PM failed, you can rerun just that window.

Feature	Schedule Trigger	Tumbling Window Trigger
Type	Calendar-based	Window-based (fixed intervals)
Dependency	No dependency between runs	Can depend on previous runs
Overlap	Runs can overlap	Non-overlapping windows
Use Case	Simple recurring jobs	Incremental, window-based processing
Retry/Re-run	Manual rerun of pipeline	Can rerun a specific failed window

New trigger

Type *
Tumbling window

Start Date (UTC) * ⓘ
8/29/2025, 10:00:00 AM

Recurrence * ⓘ
Every 15 Minute(s)

Specify an end date

End On (UTC) * ⓘ
8/30/2025, 5:55:54 AM

> Advanced

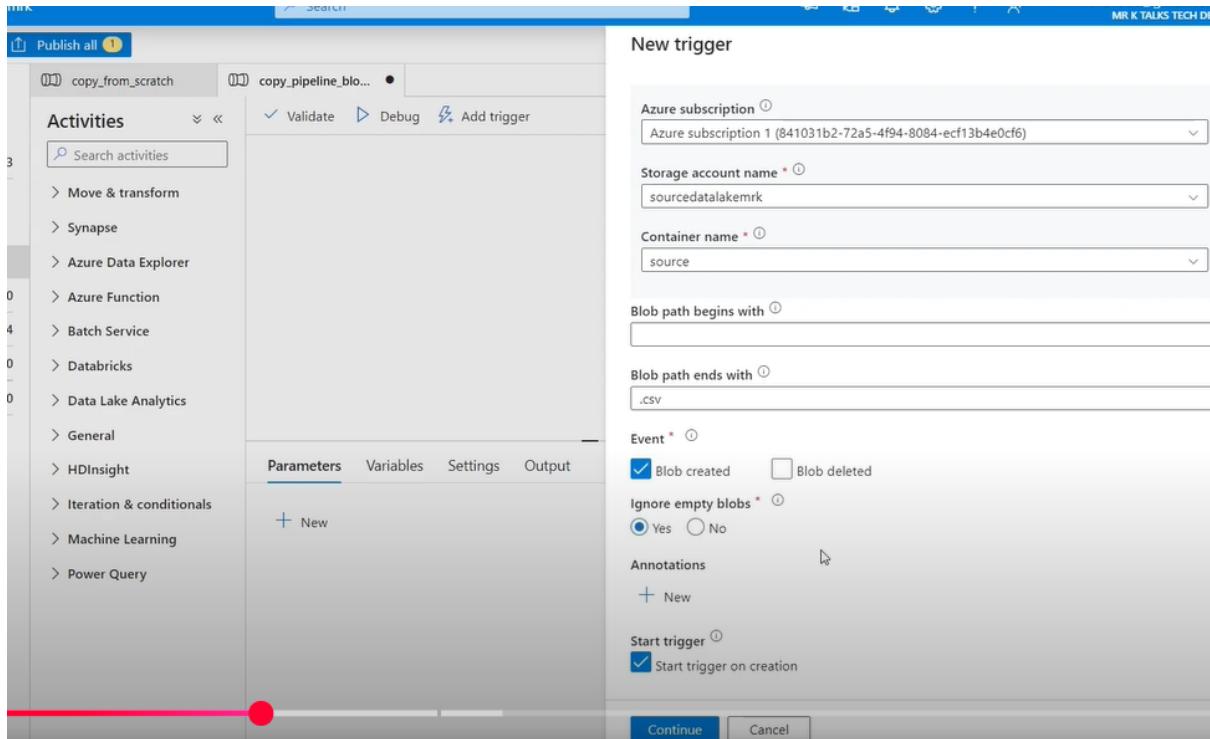
Annotations
[New](#)

[Start trigger](#) ⓘ

OK Cancel

3. Event-based Trigger

- Runs a pipeline **when an event occurs**, not on a schedule.
- Commonly used with **Azure Blob Storage** or **ADLS Gen2**.
- Typical events:
 - **Blob created** (new file uploaded)
 - **Blob deleted**
- Runs only when the event happens
- Example:
 - When a new file arrives in Azure Blob container, trigger pipeline to process it. (Run whenever a new orders.csv arrives in the raw-data container).
 - Ingest data automatically when a **new CSV file arrives** in raw/sales/ folder.
 - No need to schedule → the pipeline starts as soon as file lands.



Event:

blob created – (if the file comes to from the folder, then triggered)

blob deleted – (if the file removed from the folder, then trigger it)

4. Manual or custom Trigger

- **User-defined trigger** created using REST API, PowerShell, or SDK.
- Unlike Schedule, Tumbling, or Event triggers, you have **full control**.
- Useful when built-in triggers don't fit your needs.
- You can invoke them programmatically with **custom logic** (from another system, app, or script).

The screenshot shows the Microsoft Azure Data Factory interface. On the left, the navigation pane includes options like Data Factory, Git configuration, ARM template, Author, Triggers (which is selected), Global parameters, Data flow libraries, Security, Credentials, Customer managed key, Outbound rules, Managed private endpoints, Workflow orchestration manager, and Apache Airflow. The main area is titled 'Triggers' with a sub-instruction: 'To execute a pipeline set the trigger. Triggers represent a unit of execution'. It features a 'New' button and a 'Refresh' button. A search bar and a filter by name dropdown are also present. The right side of the screen displays the 'New trigger' dialog. The 'Name' field is filled with 'trigger1'. The 'Description' field is empty. The 'Type' dropdown is set to 'Schedule'. The 'Start date' is set to '8/27/2025, 9:39:38 AM'. The 'Time zone' is set to 'Chennai, Kolkata, Mumbai, New Delhi (UTC+5:30)'. The 'Recurrence' dropdown shows 'Every 15 Minute(s)'. At the bottom are 'OK' and 'Cancel' buttons.

Practical demo of linked service

Connect the blob storage to ADF (use lined service) – customer.csv file

The screenshot shows the Azure Storage Explorer interface. The left sidebar lists 'Overview', 'Diagnose and solve problems', 'Access Control (IAM)', and 'Settings'. The main area shows a blob container named 'datasetcontainer'. It contains one item, 'customers-100.csv', which was last modified on 27/08/2025, 19:48:35. The blob type is 'Block blob' and its size is 16.86 KB. The status is 'Available'. Below the blob list is a table header with columns: Name, Last modified, Access tier, Blob type, Size, Lease state. A checkbox is checked next to the first column.

The screenshot shows the Microsoft Azure Data Factory interface. The left sidebar includes General, Factory settings, Connections (with 'Linked services' selected), Source control, Author, and Triggers. The main area is titled 'Linked services' with a sub-instruction: 'Linked service defines the connection information to a data store'. It features a 'New' button and a 'Refresh' button. A search bar and a filter by name dropdown are also present. The right side of the screen displays the 'New linked service' dialog. The 'Data store' tab is selected. Under the 'Azure' tab, there are icons for Azure AI Search, Azure Blob Storage, Azure Cosmos DB for MongoDB, and other services. At the bottom are 'Continue' and 'Cancel' buttons.

The screenshot shows the Microsoft Azure Data Factory interface. The left sidebar includes General, Factory settings, Connections (with 'Linked services' selected), Source control, Author, and Triggers. The main area is titled 'Linked services' with a sub-instruction: 'Linked service defines the connection information to a data store'. It features a 'New' button and a 'Refresh' button. A search bar and a filter by name dropdown are also present. The right side of the screen displays the 'New linked service' dialog for 'Azure Blob Storage'. The 'Connection string' tab is selected. The 'Authentication type' is set to 'Account key'. The 'Account selection method' is set to 'From Azure subscription'. The 'Azure subscription' dropdown is set to 'Azure for Students (b279b1f8-3e6d-4236-af0e-0dd64e534656)'. The 'Storage account name' dropdown is set to 'stbigrdata'. A red box highlights the 'Azure subscription' dropdown. At the bottom are 'Create', 'Back', 'Test connection', and 'Cancel' buttons.

Microsoft Azure | Data Factory | createfirstdatafactory | Search

Validate all Publish all

Preview experience Off

Linked services

Linked service defines the connection information to a data store or compute. Learn more

+ New

Filter by name Annotations : Any

Showing 1 - 1 of 1 items

Name ↑↓	Type ↑↓	Related ↑↓	Annotations ↑↓
AzureBlobStorage1	Azure Blob Storage	0	

General
Connector upgrade advis...
Factory settings
Connections
Integration runtimes
Microsoft Purview
ADF in Microsoft Fabric
Source control
Git configuration
ARM template
Author
Triggers

Create dataset

Data Factory | Validate all | Publish all

Factory Resources

Pipelines 0
Change Data Capture (preview) 0
Datasets 0
Data flows 0
Power Query 0

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

All Azure Database File Generic protocol

Azure AI Search	Azure Blob Storage	Azure Cosmos DB for MongoDB

Continue Cancel

Select format

Select format

Choose the format type of your data

Avro	Binary	DelimitedText
Excel	JSON	ORC

Continue Back Cancel

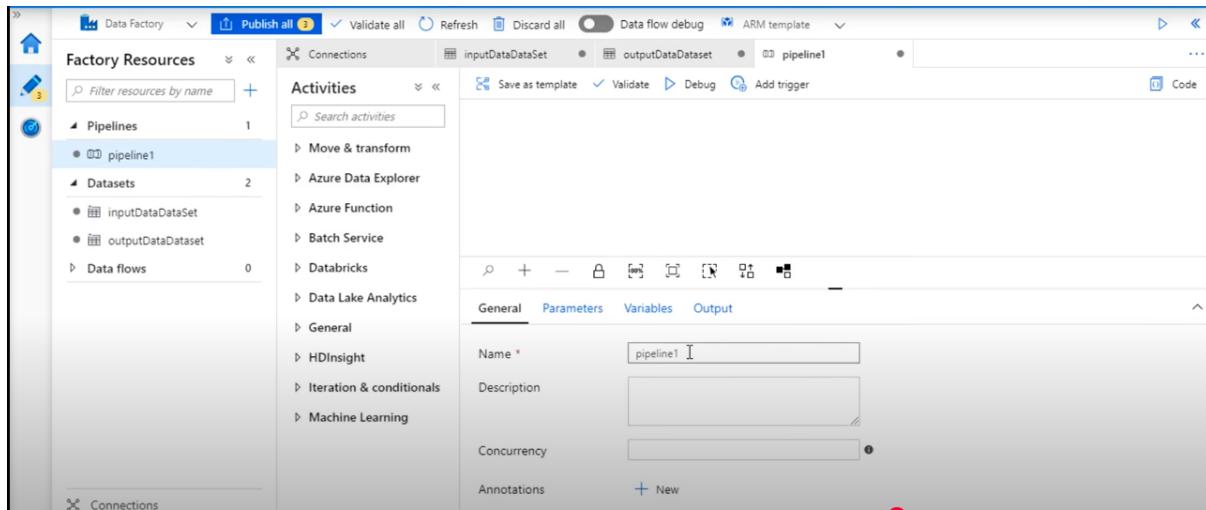
The screenshot shows the 'Set properties' dialog for creating a new dataset. The 'Name' field is set to 'inputdataset'. The 'Linked service' dropdown is set to 'AzureBlobStorage1'. The 'File path' field shows the path 'datasetcontainer / Directory / customers-100 (1).csv'. The 'First row as header' checkbox is checked. Under 'Import schema', the option 'From connection/store' is selected. At the bottom are 'OK', 'Back', and 'Cancel' buttons.

Same for input

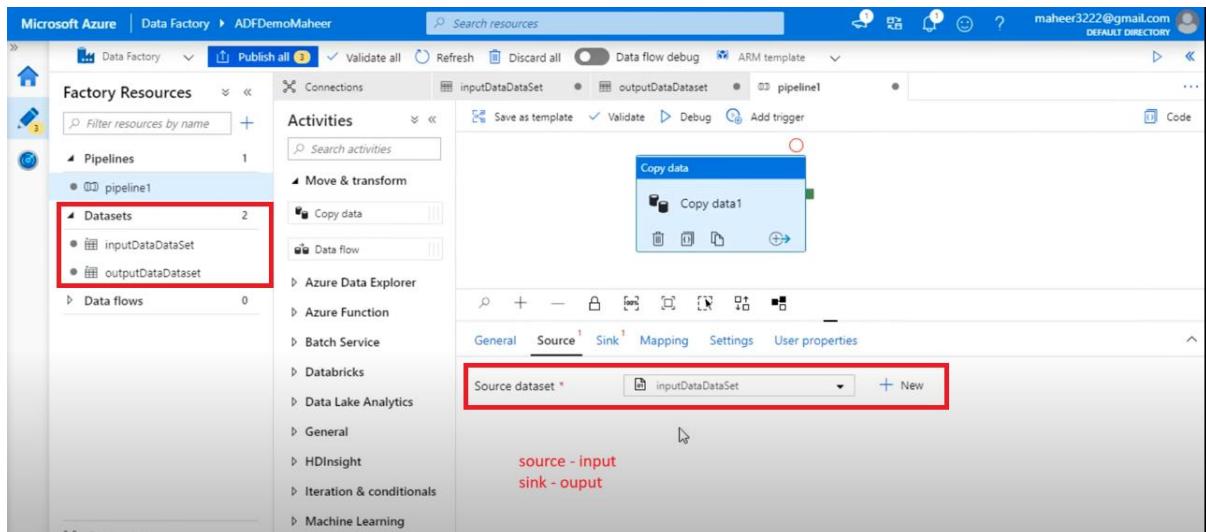
The screenshot shows the 'New linked service (Azure Blob Storage)' dialog. The 'Authentication method' is set to 'Account key'. The 'Connection string' tab is selected. The 'Account selection method' is set to 'From Azure subscription'. The 'Azure subscription' dropdown shows 'Free Trial (b082bb2a-59fa-488b-b269-f1a9d4365bc4)'. The 'Storage account name' is 'storagedemomaheer'. Under 'Additional connection properties', there is a 'New' button. The 'Test connection' section has 'To linked service' selected. The 'Annotations' section has a 'New' button. At the bottom are 'OK', 'Cancel', and 'Close' buttons.

One for output

The screenshot shows the 'Set properties' dialog for creating a new dataset. The 'Name' field is set to 'outputDataDataSet'. The 'Linked service' dropdown is set to 'linkedService_StorageDemoMaheer'. The 'File path' field shows the path 'adfdemo / output / File'. The 'Browse' button is highlighted with a mouse cursor. At the bottom are tabs for 'General', 'Connection', and 'Parameters', along with 'Name', 'Description', and 'Annotations' fields.



Copy activity – copy the data from the input folder to copy it output folder.



1st --- Validate

2nd --- Debug (whether data is copied or not)

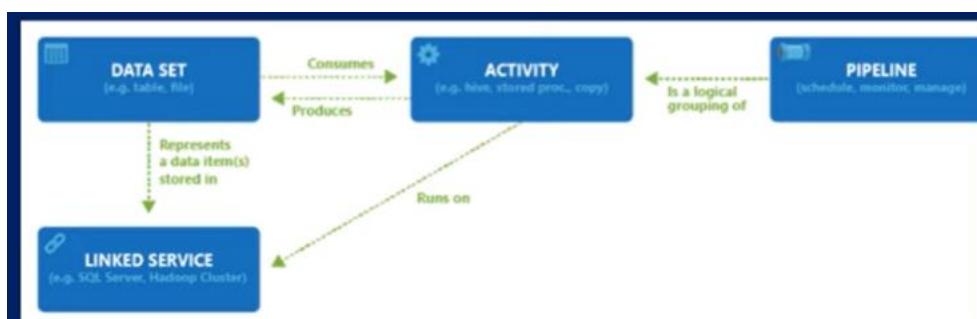
It is ok then publish all

The screenshot shows the Azure Data Factory pipeline editor interface. On the left, the 'Factory Resources' sidebar lists 'Pipelines' (1), 'Datasets' (2), and 'Data flows' (0). The main workspace displays a single 'Copy data' activity named 'Copy data1'. Below the activity, the 'Output' tab is selected, showing a table with one row:

NAME	TYPE	RUN START	DURATION	STATUS	INTEGRATION RI
Copy data1	Copy	2020-02-25T08:24:25.086	00:00:02	Queued	

This screenshot shows the same Azure Data Factory pipeline editor interface after the 'Copy data' activity has completed successfully. A modal dialog box in the top right corner indicates 'Succeeded pipeline1' with a green checkmark and a 'Dismiss' button. The table below the activity shows the status as 'Succeeded'.

NAME	TYPE	RUN START	DURATION	STATUS	INTEGRATION RI
Copy data1	Copy	2020-02-25T08:24:25.086	00:00:04	Succeeded	DefaultIntegration



Activities

Types of Activities

- Data Movement Activities
- Data Transformation Activities
- Control flow activities

First, think about Azure Data Factory (ADF)

ADF is just an **orchestrator**.

- You design **pipelines** (like a recipe: "copy this data, transform that, load here").
- But ADF itself does **not** have the compute power to move or process the data.

So... something has to **do the actual work** → that's the **Integration Runtime (IR)**.

Integration Runtime (IR)

The Integration Runtime (IR) is the compute infrastructure used by Azure Data Factory.

Azure Synapse uses to **move data, run transformations, and connect to data sources**.

[Integration Runtime (IR) is the compute engine of ADF/Synapse that provides the processing power and connectivity needed to move and transform data].

Note: (Think of it like the **engine** that executes your pipeline activities. Without an IR, your pipeline has the steps defined, but nothing to **run them**.)

Azure Synapse pipelines to provide the following data integration capabilities across different network environments:

- **Data Flow:** Execute a data flow in a managed Azure compute environment.
- **Data movement:** Copy data across data stores in a public or private network (for both on-premises or virtual private networks). The service provides support for built-in connectors, format conversion, column mapping, and performant and scalable data transfer.
- **Activity dispatch:** Dispatch and monitor transformation activities running on various compute services such as Azure Databricks, Azure HDInsight, ML Studio (classic), Azure SQL Database, SQL Server, and more.
- **SSIS package execution:** Natively execute SQL Server Integration Services (SSIS) packages in a managed Azure compute environment.

Without IR, your pipeline is only instructions; nothing will actually run.

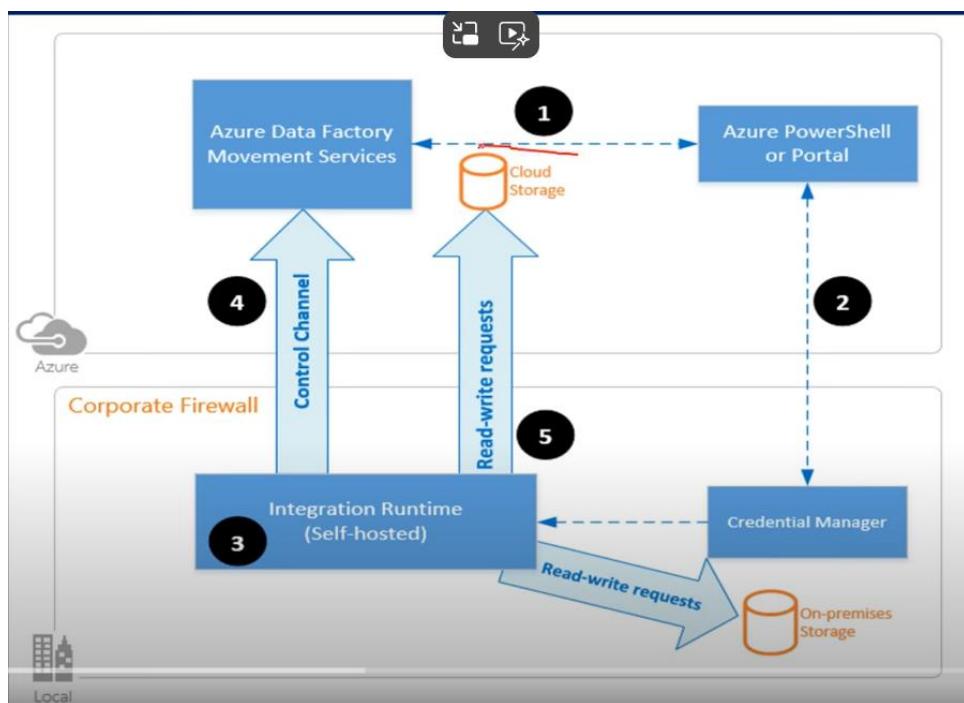
Environment

IR types:

ADF offers 3 types of IR, and you should choose the types that best serves your data integration capabilities and network environment requirements.

- **Azure IR:** Handles cloud-based data integration tasks and is managed entirely by Azure. (default)

- [Running dataflow
- Running copy activities between cloud data stores.
- Running transform activities]
- Best for **cloud-to-cloud data movement** and **transformations**.
- Supports **mapping data flows** and copying data between Azure & other cloud sources.
- **Self-hosted IR:** Supports data movement between on-premises systems and the cloud, making it ideal for hybrid scenarios.
 - Installed on your **own machine (on-prem or in VM)**.
 - Acts like a **bridge** between ADF and your **private network**.
 - Required when you need to connect to:
 - On-premises databases (SQL Server, Oracle, MySQL, etc.).
 - Private network sources behind firewalls/VPN.
 - Can also be used for **cloud-to-cloud** copy if you want data to pass through your network.
 - Example: Copy data from On-Prem SQL Server → Azure Data Lake.



we can two things to connect with local.
opt:1 and opt:2

Integration runtime setup

Settings Nodes Auto update Sharing

Install integration runtime on Windows machine or add further nodes using the Authentication Key.

Name: IR-selfhost

Option 1: Express setup
Click here to launch the express setup for this computer

Option 2: Manual setup

Step 1: Download and install integration runtime

Step 2: Use this key to register your integration runtime

NAME	AUTHENTICATION KEY
Key1	IR@bb84c923-99fd-4424-bb76-4a72170371fd@ADFDemoMaheer@cu
Key2	IR@bb84c923-99fd-4424-bb76-4a72170371fd@ADFDemoMaheer@cu

Close

Need to install the IR setup.

Microsoft Integration Runtime Setup

Completed the Microsoft Integration Runtime Setup Wizard

Click the Finish button to exit the Setup Wizard.

Back Finish Cancel

Close

Copy the key and paste it.

Microsoft Integration Runtime Configuration Manager

Register Integration Runtime (Self-hosted)

Welcome to Microsoft Integration Runtime Configuration Manager. Before you start, register your Integration Runtime (Self-hosted) node using a valid Authentication Key.

Authentication Key:

Show Authentication Key Learn how to find the Authentication Key

HTTP Proxy

Current Proxy: No proxy Change

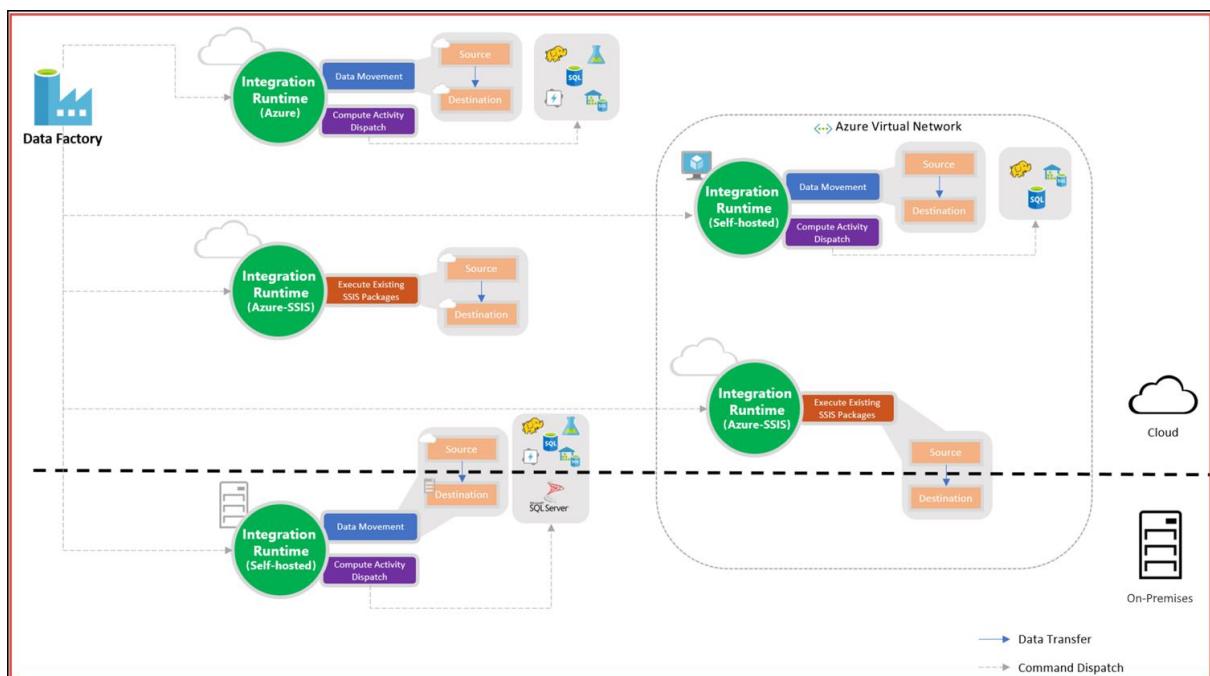
Verifying Authentication Key...

Register Cancel

Close

The screenshot shows the Microsoft Azure Data Factory portal. In the top navigation bar, it says "Microsoft Azure | Data Factory > ADFDemoMaheer". Below the navigation bar is a search bar labeled "Search resources". The main area is titled "Factory Resources" and shows a list of resources: "Pipelines" (1), "Datasets" (2), and "Data flows" (0). On the right, there's a "Connections" blade titled "Integration runtimes". It displays two items: "AutoResolveIntegrationR..." (Azure, Public, Running, Auto Resolve) and "IR-selfhost" (Self-Hosted, Running). There are buttons for "New", "Refresh", "Discard all", "Data flow debug", and "ARM template". A "Connections" tab is also visible at the bottom left.

- **SSIS IR:** Enables the execution of SQL Server Integration Service (SSIS) packages within Azure, allowing you to reuse existing SSIS workflows in the cloud.



Integration runtimes

The integration runtime (IR) is the compute infrastructure to provide the following data integration capabilities across different network environment. [Learn more](#)

Name	Type	Sub-type	Status	Related	Region	Version
AutoResolveIntegrationRuntime	Azure	Public	Running	0	Auto Resolve	---

Linked services

Linked service defines the connection information to a data store

Name
AzureBlobStorage1

Connect via integration runtime *

- AutoResolveIntegrationRuntime
- + New
- AutoResolveIntegrationRuntime

Create new IR

Integration runtime setup

Integration Runtime is the native compute used to execute or dispatch activities. Choose what integration runtime to create based on required capabilities. [Learn more](#)

Azure, Self-Hosted Perform data flows, data movement and dispatch activities to external compute.
Azure-SSIS Lift-and-shift existing SSIS packages to execute in Azure.
Airflow (Preview) Use this for running your existing DAGs

Parameterize Linked Services

Consider SQL server:

contains 10 databases, for that creating 10 linked service for all 10 DBs.

To avoid this we use, **parameterization** (by this create only one linked service and use it for 10 dbs).

Create storage (sql server):

New linked service

SQL Server [Learn more](#)

AutoResolveIntegrationRuntime

Version

2.0 (Recommended) 1.0

[Import from connection string](#)

Server name *

Database name *

Add dynamic content [Alt+Shift+D]

Authentication type

SQL authentication

User name *

[Test connection](#) [Cancel](#)

Pipeline expression builder

Learn about linked service parameterization [here](#)

Please specify an expression

Clear contents

Filter system variables and functions... +

Parameters

OK Cancel

The screenshot shows the Microsoft Azure Data Factory pipeline expression builder. On the left, there's a sidebar with options like General, Connections, and Source control. The main area is titled 'Pipeline expression builder' and contains a text input field with the expression `linkedService().dbname`. Below the input field, there's a section for 'Parameters' with a single entry: `dbname`. At the bottom right are 'OK' and 'Cancel' buttons.

Once created

The screenshot shows the Microsoft Azure Data Factory 'Factory Resources' blade. Under the 'Connections' tab, there are two entries: 'linkedService_SQL' (type: SQL Server) and 'linkedService_StorageDemoMaheer' (type: Azure Blob Storage). A confirmation message box is visible in the top right corner: 'linkedService_SQL has been saved.' with a 'Dismiss' button.

The screenshot shows the Microsoft Azure Data Factory 'Linked Service' blade for 'linkedService_SQL'. The JSON configuration is as follows:

```

1  "name": "linkedService_SQL",
2  "type": "Microsoft.DataFactory/factories/linkedservices",
3  "properties": {
4    "parameters": {
5      "dbName": {
6        "type": "string"
7      }
8    },
9    "annotations": [],
10   "type": "SqlServer",
11   "typeProperties": {
12     "connectionString": "Integrated security=False;data source=mydbserver.windows.db.net;initial catalog=@{linkedService().dbName};user id=maheer",
13     "encryptedCredential": "ew0KCAIVmVyc2lvbil6ClyMDE3LTExLTMwliwNCiAgIIByb3Ry3Rpz5Nb2RljlqgkteleSlsDQogICJTWNyZXRUb250ZW50HlwZSI6ICJQbGFpbnRI"
14   }
15 }
16
17

```

A red box highlights the `dbName` parameter in the JSON, and a red arrow points from it to the text 'dynamically change db name'.

We can create the parameters for all the things like username, password, server name and database name.

New linked service
SQL Server [Learn more](#)

Additional connection properties

Annotations

Parameters

Name	Type	Default value
dbname	String	Value
servername	String	Value
username	String	Value
password	String	Value

Advanced

Create Back Test connection Cancel

New linked service
SQL Server [Learn more](#)

Server name *
@linkedService().servername

Database name *
@linkedService().dbname

Authentication type
SQL authentication

User name *
@linkedService().username

Password [Azure Key Vault](#)
Password *
@linkedService().password

Always encrypted

Encrypt
Mandatory

Create Back Test connection Cancel

Parameterize Dataset and pipeline in ADF

You can parameterize a dataset and pass dynamic values at run time.

Supported file format in ADF

- **Avro format**
- **Binary format – text files**
- **Delimited text format – csv file**
- **JSON format**
- **ORC format**
- **Parquet format**

Copy Data Activity

The screenshot shows the Microsoft Azure Data Factory pipeline editor. On the left, the 'Factory Resources' sidebar lists 'Pipelines' (1), 'Datasets' (1), and 'Data flows' (0). The main workspace displays a pipeline named 'pipeline1' containing a single 'Copy data' activity named 'Copy data1'. The 'Activities' pane on the right shows various options like 'Move and transform', 'Copy data', and 'Data flow'. The 'General' tab of the 'Copy data' activity configuration is selected, showing fields for Name (Copy data1), Activity state (Activated), Timeout (0:12:00:00), and Retry (0).

Source

The 'Source' tab is active for the 'Copy data1' activity. Under 'Source dataset', the dropdown is set to 'source_customer_csv_file'. The 'File path type' section includes options for 'File path in dataset' (selected), 'Prefix', 'Wildcard file path', and 'List of files'. Below these are filters for 'Start time (UTC)', 'End time (UTC)', and 'Filter by last modified'. The 'Recursively' checkbox is checked. The 'Enable partitions discovery' checkbox is unchecked.

Destination - sink

The 'Sink' tab is active. Under 'Sink dataset', the dropdown is set to 'output_customer_csv_file'. Other configuration fields include 'Copy behavior' (set to 'Select...'), 'Max concurrent connections' (empty), 'Block size (MB)' (empty), 'Metadata' (with a 'New' button), and 'Quote all text' (unchecked).

click on import schema

The screenshot shows the Microsoft Azure Data Factory interface. In the center, there's a pipeline named 'pipeline1'. Under the 'Mapping' tab, there's a section titled 'Type conversion settings' with a sub-section 'Import schemas'. This section is highlighted with a red rectangle. Below it is a table mapping source columns to destination columns, including 'Customer Id', 'Index', 'First Name', 'Last Name', 'Company', and 'City'.

Source	Type	Destination	Type
Index	abc String	Index	abc String
Customer Id	abc String	Customer Id	abc String
First Name	abc String	First Name	abc String
Last Name	abc String	Last Name	abc String
Company	abc String	Company	abc String
City	abc String	City	abc String

We change the column order and data type. Also delete the not required column in destination.

The screenshot shows the same pipeline and mapping configuration as the previous one, but with changes made to the mapping table. The 'Customer Id' and 'Index' columns have been swapped in the mapping. Additionally, a new column 'Phone 2' has been added to the destination side, mapped from the 'Phone 1' source column.

Source	Type	Destination	Type
Customer Id	abc String	Customer Id	abc String
Index	12s Int16	Index	abc String
First Name	abc String	First Name	abc String
Last Name	abc String	Last Name	abc String
Company	abc String	Company	abc String
City	abc String	City	abc String
Country	abc String	Country	abc String
Phone 1	1.2 Double	Phone 1	abc String
Phone 2	1.2 Double	Phone 2	abc String

✓ Validate ⚙️ Cancel options ⚡ Add trigger

Copy data

Copy data1

Parameters Variables Settings **Output**

Pipeline run ID: f466233d-ba03-4eb8-9dc9-36feba2cb66a Pipeline status: In progress

All status ▾ Monitor in Azure Metrics Export to CSV

Showing 1 - 1 of 1 items

Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime
Copy data1	Queued	Copy data	8/28/2025, 10:47:52 PM	2s	

source_customer_csv... pipeline1

✓ Validate ✓ Validate copy runtime ▶ Debug ⚡ Add trigger

Copy data

Copy data1

General Source Sink Mapping Settings **User properties**

+ New Delete Auto generate

Name	Value
Source	datasetcontainer//customers-100 (1).csv
Destination	destinationcontainer//

Microsoft Azure | Data Factory > createfirstdatafactory

Factory Resources

- Pipelines: pipeline1
- Datasets: dest_output, output_customer_csv_file, source_customer_csv_file
- Data flows: 0
- Power Query: 0

source_customer_csv... pipeline1

✓ Validate ▶ Debug ⚡ Add trigger

Copy data

Copy data1

Parameters Variables Settings **Output**

Pipeline status: Succeeded

Showing 1 - 1 of 1 items

Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime
Copy data1	Succeeded	Copy data	8/28/2025, 10:49:19 PM	18s	AutoResolveIntegrationRuntime (South India)

Publish

The screenshot shows the Microsoft Azure Data Factory 'Publish' interface. On the left, the 'Factory Resources' sidebar lists Pipelines, Datasets, Data flows, and Power Query. In the center, a pipeline named 'pipeline1' is selected, showing a 'source_customer_csv...' dataset and a 'dest_output' dataset. The 'dest_output' dataset is configured as a 'DelimitedText' type with 'CSV' as the file format. The 'Connection' tab is selected, showing the 'Linked service' as 'AzureBlobStorage1', 'File path' as 'datasetcontainer', and other settings like compression type, column delimiter, row delimiter, and encoding. On the right, a 'Pending changes (3)' table shows three items: 'pipeline1' (New), 'source_customer_csv_file' (New), and 'dest_output' (New). Below the table are 'Publish' and 'Cancel' buttons.

Copied from src to destination

The screenshot shows the Azure Storage Explorer interface. A container named 'destinationcontainer' is selected. Inside, there is one blob named 'customers-100'. The blob's properties are shown: Last modified: 28/08/2025, 23:00:20, Access tier: Hot (Inferred), Blob type: Block blob, Size: 18.74 KiB, Lease state: Available.

The screenshot shows the Azure Data Factory Pipeline run details page. The 'Output' tab is selected. At the top, it shows the 'Pipeline run ID' as 2be52330-6154-4afd-a433-28859b986474, 'Pipeline status' as Succeeded, and 'View debug run consumption'. Below that, there are filters for 'All status' and buttons for 'Monitor in Azure Metrics' and 'Export to CSV'. The main area displays pipeline activity details. A red box highlights the 'Destination' field under the 'Activity run ID' section, which contains the value 'destinationcontainer//'. Other fields shown include 'Name' (migrationRuntime (South India)), 'User prop...', 'Source', 'Activity run ID', and 'Log'.

For copy the data from the API (use the base url)

The screenshot shows the Microsoft Azure Data Factory interface. On the left, the 'Factory Resources' sidebar lists 'Pipelines' (pipelineGit, pipelinemanager), 'Datasets', 'Data flows', and 'Power Query'. In the center, a pipeline named 'pipelineGit' is selected. The 'Activities' section shows a 'copy' activity. The 'Source' tab is active, displaying configuration for the source dataset. A red box highlights the 'Base URL' field, which contains 'https://raw.githubusercontent.com'. A warning message below it says: 'Information will be sent to the URL specified. Please ensure you trust the URL entered.' Other visible fields include 'Server certificate validation' (Enable selected), 'Authentication type' (Anonymous), and 'Auth headers'.

Relative url

This screenshot shows the same pipeline configuration as the previous one, but with a different URL. The 'Source' tab now displays a 'Relative URL' field containing 'https://raw.githubusercontent.com'. The rest of the configuration remains the same, including the 'Will be created' note for the linked service.

Monitor copy activity

The screenshot shows the 'Pipeline runs' page. The left sidebar has a red box around the 'Runs' section, which includes 'Pipeline runs', 'Monitor per runs', 'Change Data Capture (preview...)', 'Runtimes & sessions', 'Integration runtimes', and 'Data flow debug'. The main area shows a table of pipeline runs. The first run, 'pipeline1', has three entries: two successful runs ('Succeeded') and one failed run ('Failed'). The table includes columns for Pipeline name, Run start, Run end, Duration, Status, Triggered by, Run ID, and Parameters. A red box highlights the 'Status' column header. The table shows 'Showing 1 - 3 items' and was last refreshed 1 minute ago.

Check the details of the pipeline

Once you've created and published a pipeline in ADF, you can associate it with a trigger or manually kick off an hoc run. You can monitor all of your pipeline runs natively in the ADF user experience.

Delete Activity

You can use the delete activity in ADF to delete files or folders from on-premises storage or cloud storage.

Deleted files or folders cannot be restored.

Supported data stores

- Azure Blob storage
- Azure Data Lake Storage Gen1
- Azure Data Lake Storage Gen2
- Azure File Storage

File system data stores

- File System
- FTP
- SFTP
- Amazon S3
- Google Cloud Storage

The screenshot shows the Microsoft Azure Data Factory pipeline editor interface. On the left, the 'Factory Resources' sidebar lists 'Pipelines' (with 'pipeline1' and 'deletepipeline'), 'Datasets', 'Data flows', and 'Power Query'. The main workspace shows a pipeline named 'deletepipeline'. The 'Activities' tab is selected, displaying a list of available activities including 'Append variable', 'Delete', 'Execute Pipeline', 'Execute SSIS package', 'Fail', 'Get Metadata', 'Lookup', 'Stored procedure', 'Script', and 'Set variable'. A 'Delete' activity is currently selected and is shown in a detailed configuration pane. The configuration pane includes tabs for 'General', 'Source', 'Logging settings', and 'User properties'. Under 'General', the 'Name' field is set to 'Delete1', 'Activity state' is set to 'Activated', and the 'Timeout' is set to '0.12:00:00'. The 'Source' tab is visible but contains no specific configuration. The 'Logging settings' and 'User properties' tabs are also present but not detailed in the screenshot.

We can customize the file (*.txt – means delete only txt)

The screenshot shows the 'Source' blade in the Azure Data Factory interface. At the top, there are tabs for 'General', 'Source' (which is selected), 'Logging settings', and 'User properties'. Below the tabs, the 'Dataset' section is active, with a dropdown menu showing 'dest_output'. There are buttons for 'Open', 'New', 'Preview data', and 'Learn more'. Under 'File path type', the 'Wildcard file path' option is selected. The 'Wildcard file name' field contains a placeholder '[]' with a note 'Add dynamic content [Alt+Shift+D]'. Other options like 'File path in dataset', 'Prefix', and 'List of files' are available but not selected. Below this, there are sections for 'Filter by last modified', 'Recursively' (which is checked), and 'Max concurrent connections'.

Lookup Activity

The Lookup activity in Azure Data Factory is a powerful tool used to retrieve data from an external source, such as a database or file system, and pass it to subsequent activities in a pipeline.

Lookup activity reads and returns the content of a configuration file or table. It also returns the result of executing a query or stored procedure.

The output from lookup activity can be used in a subsequent activity.

What Lookup Activity Does:

- It runs a **query** or **reads a file** and returns the result.
- The result can be:
 - **Single row** → returns key-value pairs.
 - **Multiple rows** → returns an array of JSON objects.

Limitation or drawback of lookup activity

- **Row Limit:** It can fetch a maximum of **5,000 rows**.
- **Data Size Limit:** The total size of the fetched data cannot exceed **4 MB**.

Example Scenario

- **Use Case:** Retrieve a file name from a database table to dynamically copy the file from Azure Blob Storage.

Scenario

We had to build a pipeline that **processes multiple files dynamically**.

- File details (name, path, active flag) were stored in a **control/config table** in Azure SQL DB.

- Each file had to be copied from **source Blob storage** to **curated ADLS zone** only if it was **active**.

Step 1 – Lookup Activity

- First, we used a **Lookup activity** to fetch all the file details from the control table.
- Lookup returned JSON like:

```
{
  "value": [
    { "FileName": "sales.csv", "Path": "/input/sales/", "IsActive": 1 },
    { "FileName": "customer.csv", "Path": "/input/customers/", "IsActive": 0 }
  ]
}
```

- This allowed us to **parameterize the pipeline** and make it **dynamic**.
- Without Lookup, we would have to hardcode the file names.

Step 2 – ForEach Activity

- We passed the Lookup output array into a **ForEach activity**.
- ForEach iterated over each file record.

Step 3 – Copy Activity (inside ForEach)

- Inside ForEach, we placed a **Copy activity**.
- Source = Blob file path (parameterized from Lookup result).
- Sink = ADLS curated folder.
- Copy moved the actual file contents (GBs of data) from source → destination.

Get Metadata

The Get Metadata activity in Azure Data Factory (ADF) is a control flow activity that retrieves metadata from a specified data source.

This metadata can include details like file names, sizes, structure, or even the existence of a file or folder.

It is particularly useful for **dynamic pipelines** where decisions depend on the properties of the data.

1. **Supported Data Sources:** Works with various sources like Azure Blob Storage, Azure Data Lake, SQL databases, and more.
2. **Metadata Properties: (Field list)** You can retrieve properties such as:

- File name
- File size
- Last modified date
- Child items (for folders) – items which are available with this folder
- Column structure (for tables)
- Add dynamic

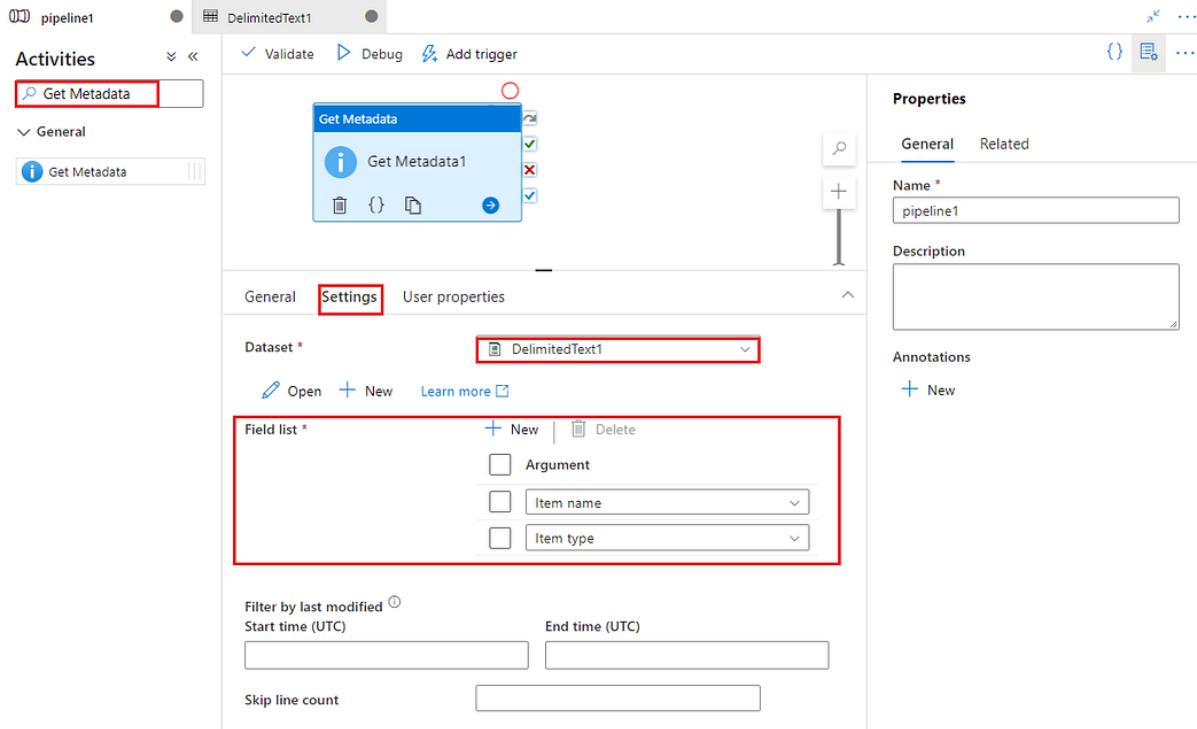
Metadata	
type	Description
itemName	Name of the file or folder.
itemType	Type of the file or folder. Returned value is <code>File</code> or <code>Folder</code> .
size	Size of the file, in bytes. Applicable only to files.
created	Created datetime of the file or folder.
lastModified	Last modified datetime of the file or folder.
childItems	List of subfolders and files in the given folder. Applicable only to folders. Returned value is a list of the name and type of each child item.
contentMD5	MD5 of the file. Applicable only to files.
structure	Data structure of the file or relational database table. Returned value is a list of column names and column types.
columnCount	Number of columns in the file or relational table.
exists	Whether a file, folder, or table exists. Note that if <code>exists</code> is specified in the Get Metadata field <code>list</code> , the activity won't fail even if the file, folder, or table doesn't exist. Instead, <code>exists: false</code> is returned in the output.

3. **Dynamic Pipelines:** Enables conditional logic based on metadata, such as checking if a file exists before processing.

Example Use Cases

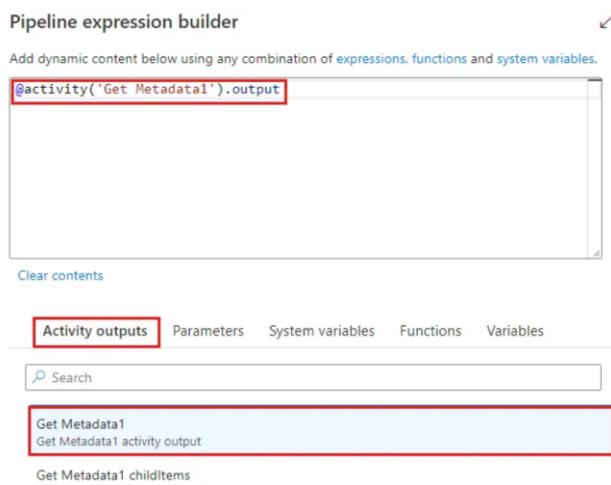
- **File Validation:** Check if a file exists before copying or transforming it.
- **Dynamic Processing:** Retrieve column names or structure to dynamically configure downstream activities.
- **Folder Iteration:** List all files in a folder for batch processing.

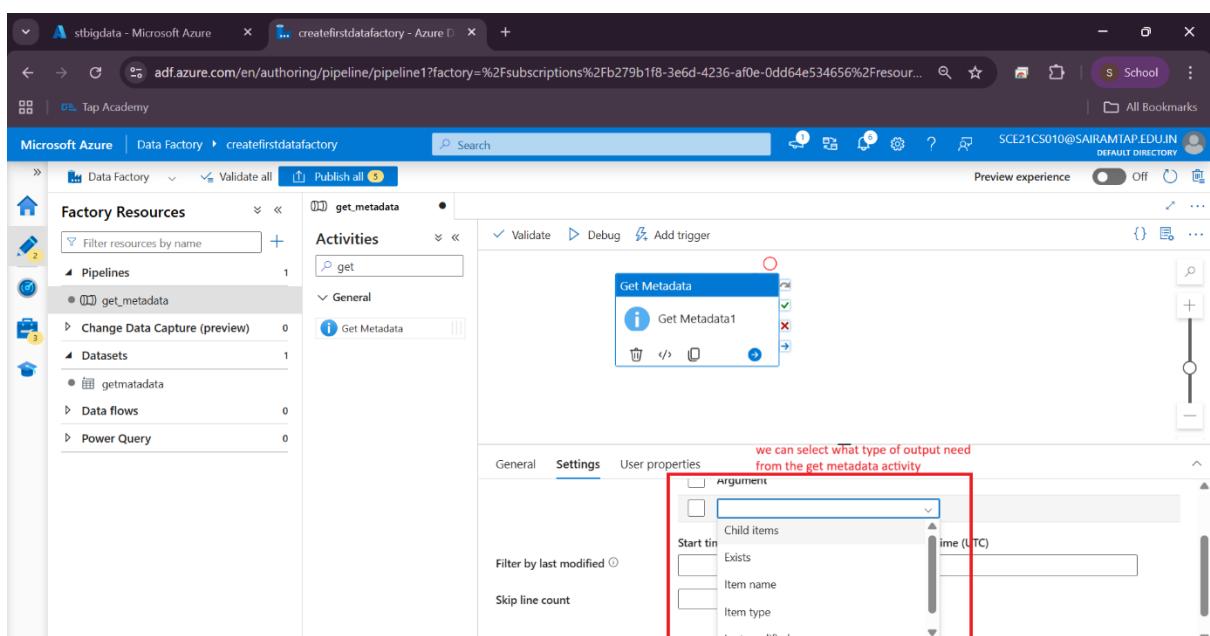
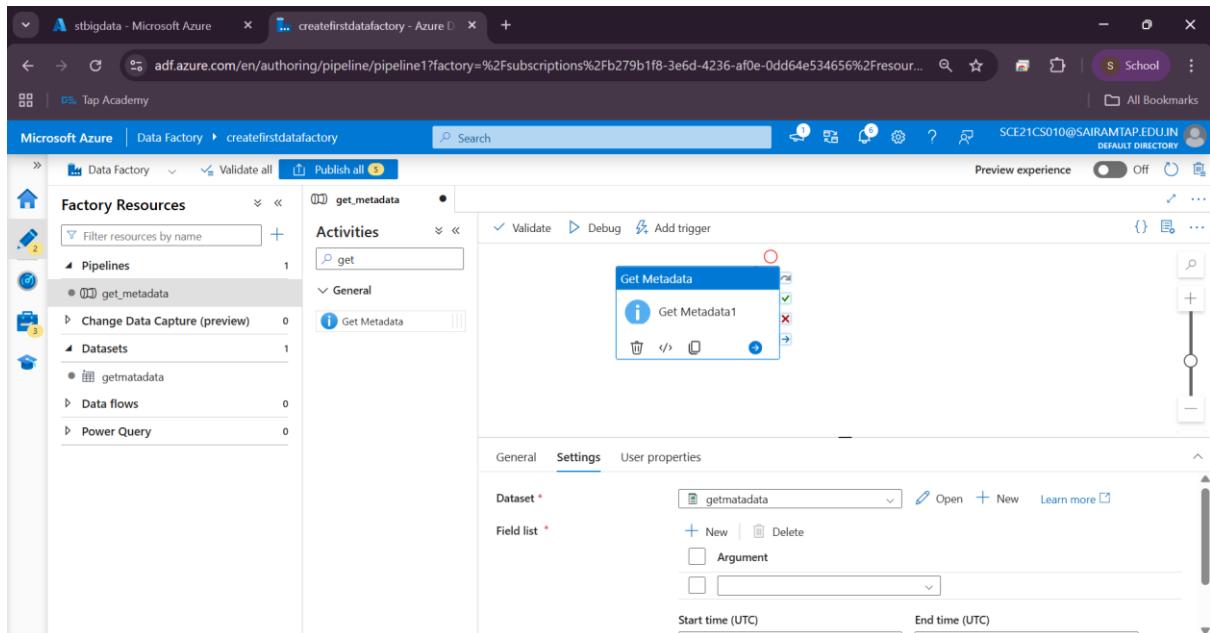
- Usage:** Useful when you need to dynamically get information about files or datasets, such as file names, size, last modified timestamps, or even schema information.



This activity is a powerful tool for building dynamic and intelligent data pipelines in ADF.

Use the Output: The output of this activity will be an array of file names. You can use this output to dynamically create pipelines or copy activities for each file.





Output

```
{
  "childItems": [
    {
      "name": "Fact_Sales_1.csv",
      "type": "File"
    },
    {
      "name": "file2.csv",
      "type": "File"
    }
  ],
  "effectiveIntegrationRuntime": "AutoResolveIntegrationRuntime (East US)",
  "executionDuration": 1,
  "durationInQueue": 0,
  "integrationRuntimeQueue": 0,
  "billingReference": [
    {
      "activityType": "PipelineActivity",
      "billableDuration": [
        {
          "meterType": "AzureIR",
          "duration": 0.01666666666666666,
          "unit": "Hours"
        }
      ]
    }
  ]
}
```

Now to connect the output of the get metadata to the for each activity (for taking the file name)

Pipeline expression builder

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

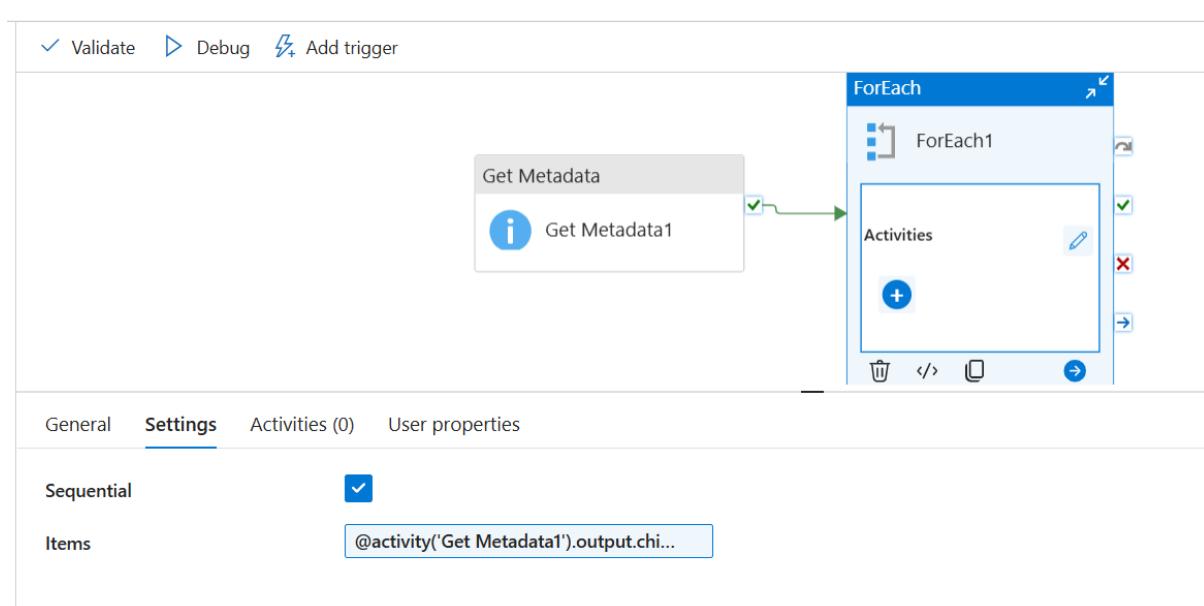
`@activity('Get Metadata1').output.childItems`

only going to pick childItems here
(based on requirement use it)

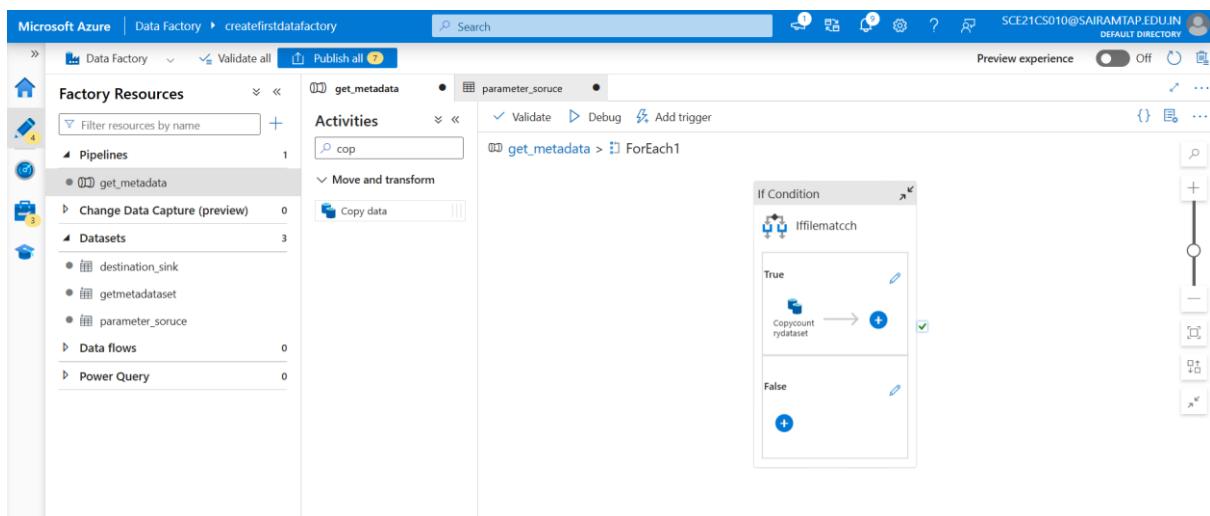
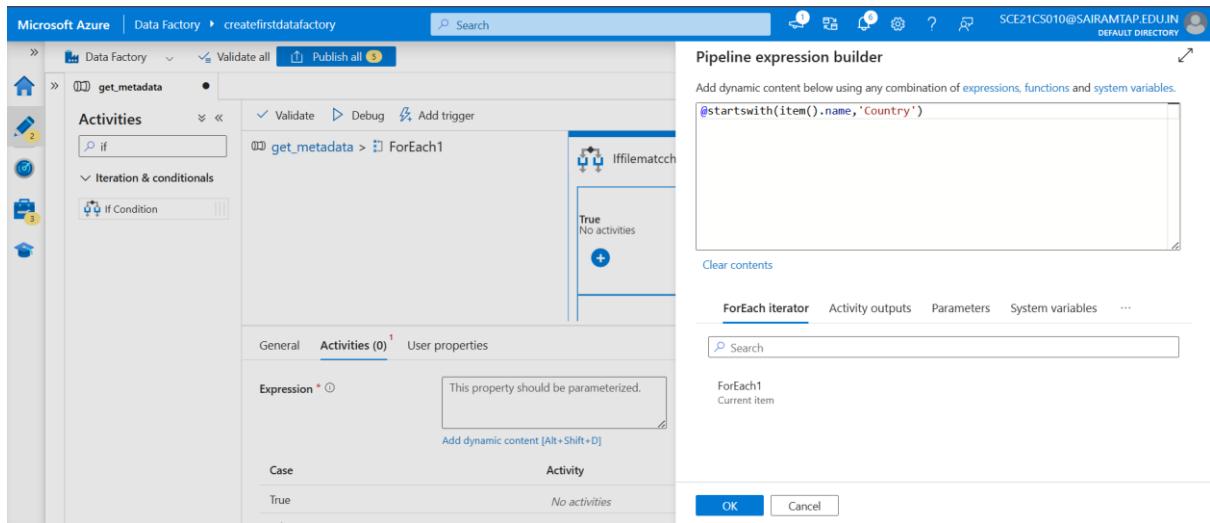
Activity outputs Parameters System variables Functions Variables

Get Metadata1
Get Metadata1 activity output
Get Metadata1 childItems
List of subfolders and files in the given folder

OK Cancel

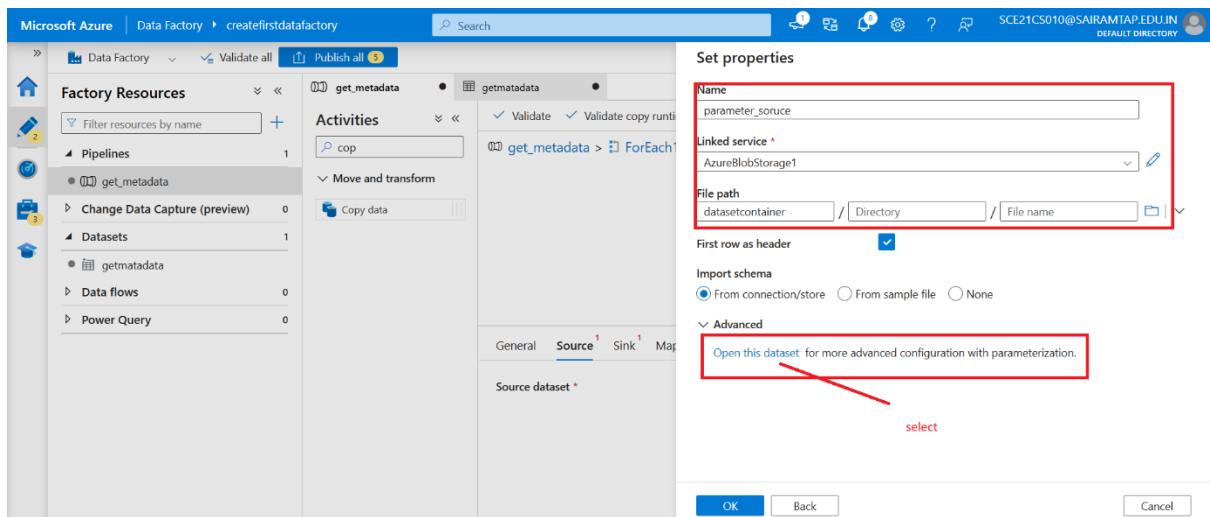


Set the condition inside the “if activity”



Step to Add the file name Dynamically

Create the dataset, while create it select the folder, directory, leave file name empty



The screenshot shows the Azure Data Factory pipeline editor. On the left, the 'Factory Resources' sidebar lists a single pipeline named 'get_metadata'. In the main workspace, a dataset named 'parameter_source' is selected. The 'Connection' tab is active, showing 'AzureBlobStorage1' as the linked service. The 'File path' field contains the expression `datasetcontainer / Directory / @dataset().parameter_file_name`. Other settings include 'No compression', 'Comma (,)', 'Default (\r\n, or \n\r)', and 'Default(UTF-8)'.

This screenshot shows the same pipeline 'get_metadata' in the editor. A 'Copy data' activity is added to the flow. The 'Source' tab is selected, showing the 'File path type' dropdown set to 'Dataset properties'. A red box highlights the 'parameter_file_name' entry in the 'Value' column of the 'Dataset properties' table. A red arrow points from the text 'file name that coming from the for loop' to this entry.

This screenshot shows the pipeline 'get_metadata' with the 'Sink' tab selected for the copy activity. The 'Sink dataset' is set to 'destination_sink'. Other sink settings include 'Select...' for 'Copy behavior', empty fields for 'Max concurrent connections' and 'Block size (MB)', and a 'Learn more' link.

Get metadata → for each → if (True) → copy activity output

Microsoft Azure | Data Factory > createfirstdatafactory

Factory Resources

- Pipelines (1)
- Datasets (3)
- Data flows (0)
- Power Query (0)

Activities

- get_metadata (1)
- Copy (1)
- Move and transform (1)
- Get Metadata (1)

Get Metadata Pipeline

Get Metadata Pipeline Status: In progress

Pipeline run ID: 033aafaa3-73a5-4e43-b955-582689c0557b

Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime
Copycountrydataset	Queued	Copy data	8/30/2025, 6:39:31 PM	8s	
Iffilematcch	In progress	If Condition	8/30/2025, 6:39:31 PM	8s	
ForEach1	In progress	ForEach	8/30/2025, 6:39:30 PM	9s	
Get Metadata1	Succeeded	Get Metadata	8/30/2025, 6:39:16 PM	14s	AutoResolveIntegrat

Microsoft Azure | Data Factory > createfirstdatafactory

Factory Resources

- Pipelines (1)
- Datasets (3)
- Data flows (0)
- Power Query (0)

Activities

- get_metadata (1)
- Copy (1)
- Move and transform (1)
- Get Metadata (1)

Get Metadata Pipeline

Get Metadata Pipeline Status: In progress

Pipeline run ID: 033aafaa3-73a5-4e43-b955-582689c0557b

Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime
Iffilematcch	Succeeded	If Condition	8/30/2025, 6:39:49 PM	Less than 1s	
Iffilematcch	Succeeded	If Condition	8/30/2025, 6:39:48 PM	Less than 1s	
Copycountrydataset	Succeeded	Copy data	8/30/2025, 6:39:31 PM	15s	AutoResolveIntegrat
Iffilematcch	Succeeded	If Condition	8/30/2025, 6:39:31 PM	17s	
ForEach1	Succeeded	ForEach	8/30/2025, 6:39:30 PM	21s	
Get Metadata1	Succeeded	Get Metadata	8/30/2025, 6:39:16 PM	14s	AutoResolveIntegrat



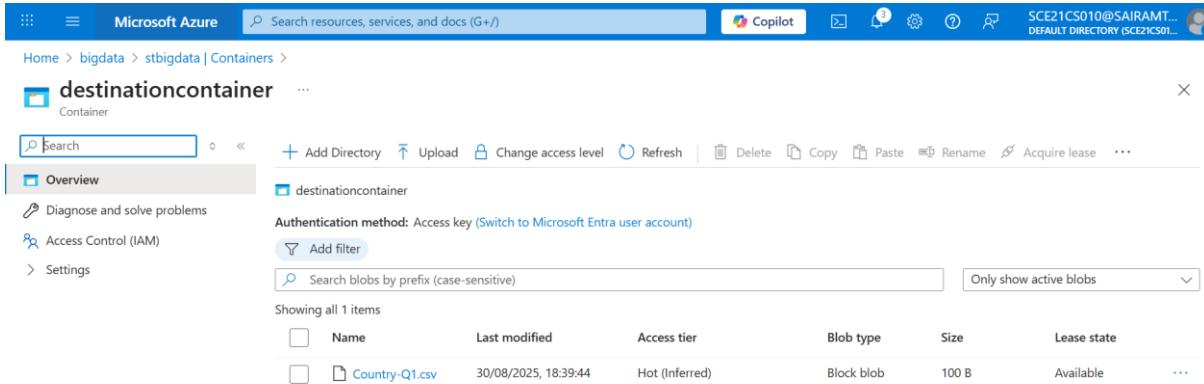
```

Output
Copy to clipboard

{
  "childItems": [
    {
      "name": "Country-Q1.csv",
      "type": "File"
    },
    {
      "name": "Department-Q1.csv",
      "type": "File"
    },
    {
      "name": "Employee-Q1.csv",
      "type": "File"
    }
  ],
  "effectiveIntegrationRuntime": "AutoResolveIntegrationRuntime (South India)",
  "executionDuration": 1,
  "durationInQueue": {
    "integrationRuntimeQueue": 10
  },
  "billingReference": {
    "activityType": "PipelineActivity",
    "billableDuration": [
      {
        "meterType": "AzureR",
        "duration": 0.01666666666666666
      }
    ]
  }
}

```

Only country dataset is copied



Microsoft Azure Search resources, services, and docs (G+)

Home > bigdata > stbigdata | Containers >

destinationcontainer Container

Search

Add Directory Upload Change access level Refresh Delete Copy Paste Rename Acquire lease ...

destinationcontainer

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

Add filter

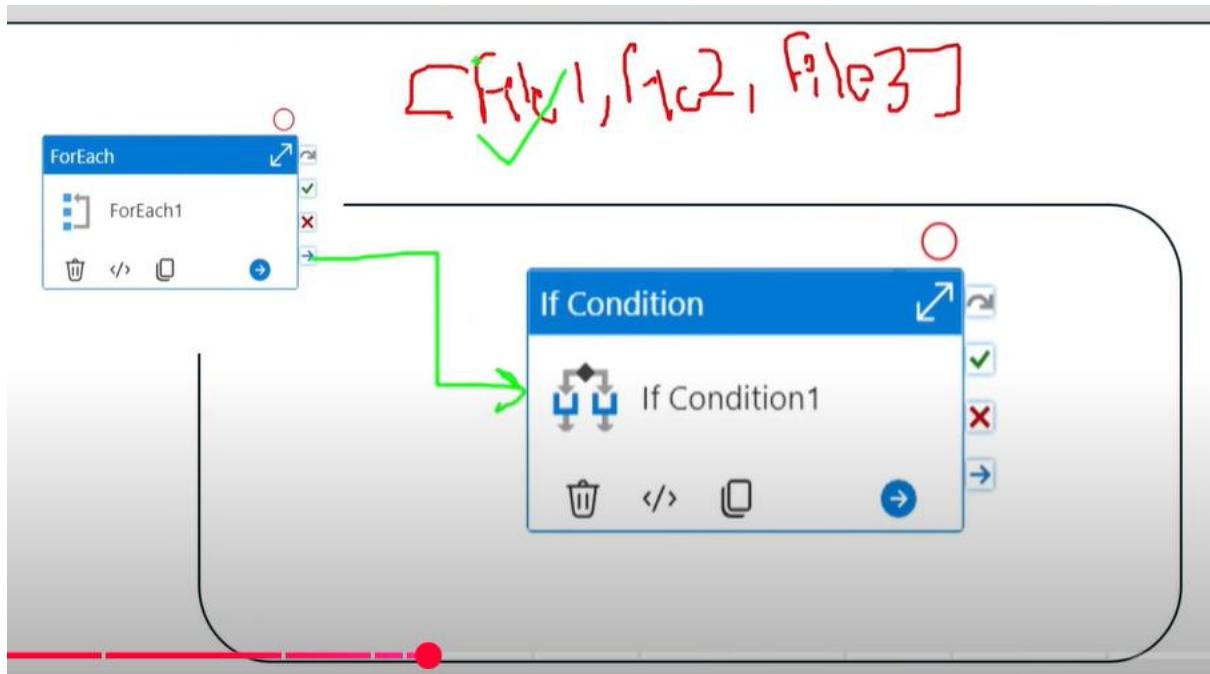
Search blobs by prefix (case-sensitive) Only show active blobs

Showing all 1 items

	Name	Last modified	Access tier	Blob type	Size	Lease state
<input type="checkbox"/>	Country-Q1.csv	30/08/2025, 18:39:44	Hot (Inferred)	Block blob	100 B	Available

Feature	Lookup Activity	Get Metadata Activity
---------	-----------------	-----------------------

Returns	Data values (rows from source) (5000rows or 4 MB)	Properties/metadata of dataset
Scope	Works on structured data sources (SQL, files, API)	Works on files, folders, tables
Typical Use	Drive pipeline logic based on data	Drive pipeline logic based on file/dataset properties
Example	Get list of customer IDs from SQL	Get list of files in blob storage. (If you want to get all CSV file names in a folder)



Key Differences:

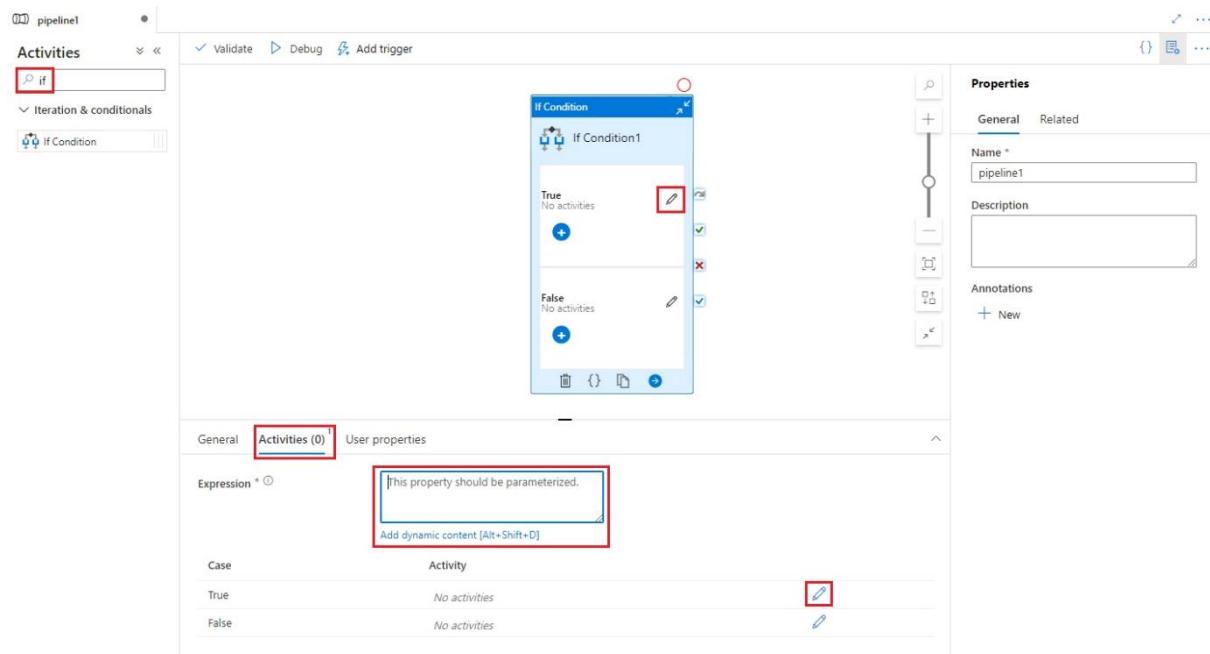
- **Functionality:** GetMetadata retrieves metadata about files or datasets, whereas Lookup retrieves data from a data source based on a query.
- **Use Case:** Use GetMetadata when you need information about files or datasets before processing. Use Lookup when you need to fetch specific data values or small datasets to use within your pipeline.
- **Output:** GetMetadata outputs metadata properties of files or datasets (like file names or schema details). Lookup outputs data rows from a query result.

If condition activity

The **If Condition activity** in Azure Data Factory (ADF) is a control flow activity that allows you to execute different sets of activities based on a logical condition. It evaluates an expression and executes one of two activity groups: one for when the condition evaluates to **true** and another for when it evaluates to **false**.

Key Features of If Condition Activity

- **Logical Evaluation:** You define an expression that evaluates to either true or false.
- **True and False Paths:**
 - If the condition is **true**, the activities in the **True** branch are executed.
 - If the condition is **false**, the activities in the **False** branch are executed.
- **Dynamic Content:** You can use dynamic expressions to evaluate variables, parameters, or outputs from other activities.

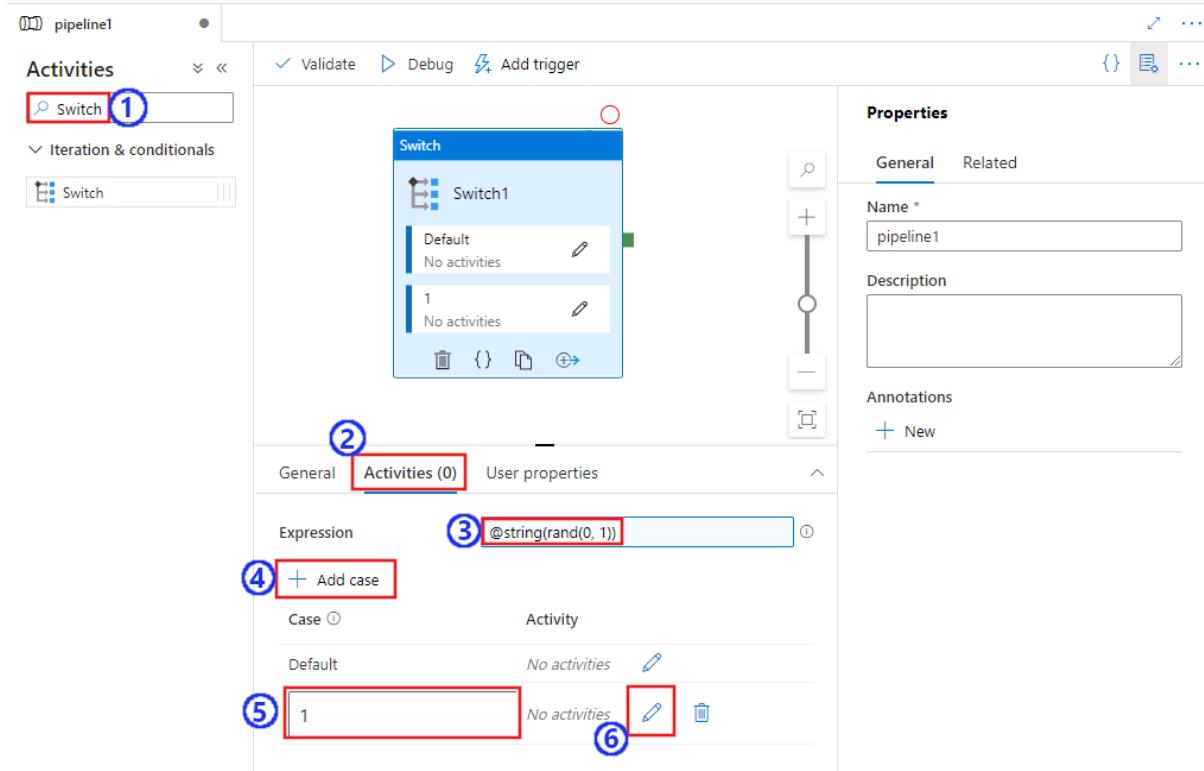


Key Points:

- **Expression-Based:** It evaluates an expression (typically using Azure Data Factory expressions) and executes activities based on the result (true or false).
- **Single Condition:** Supports only one condition evaluation per activity.
- **Control Flow:** Directs the workflow based on the evaluated condition.

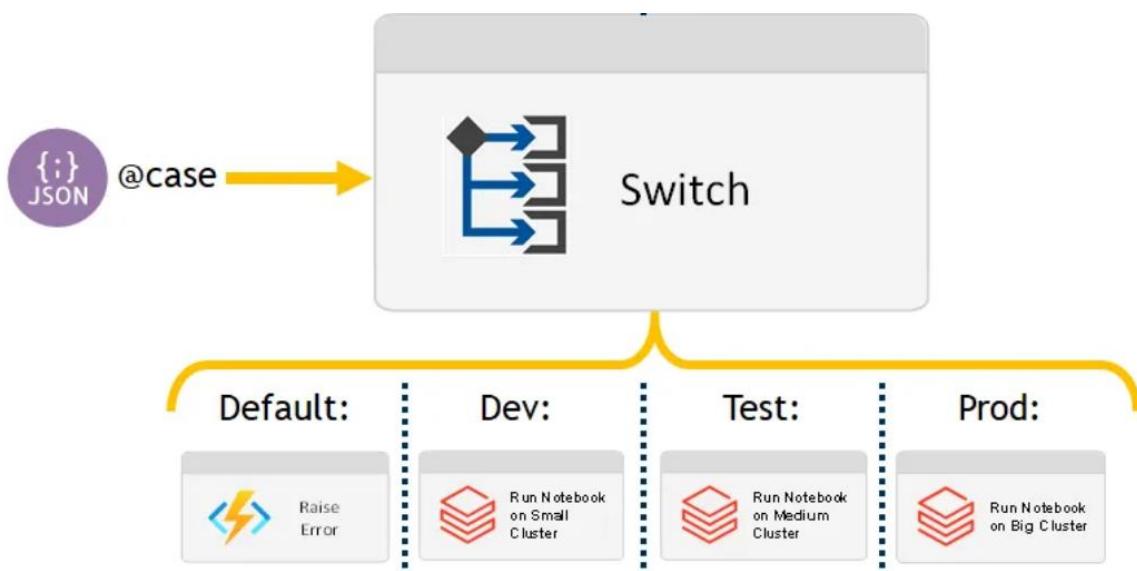
Switch Activity

When dealing with bigger sets of circumstances, this method is frequently thought to be more effective than a block of If, Else If, Else If, Else type logic tests.



- **Use Cases:**

- Multiple Conditions:** Use Switch when you have multiple conditions to evaluate and based on each condition, you want to execute different sets of activities. For instance, processing files of different types or performing different transformations based on data characteristics.
- Dynamic Branching:** It provides a structured way to handle complex branching logic where you need to evaluate multiple expressions and route based on different outcomes.



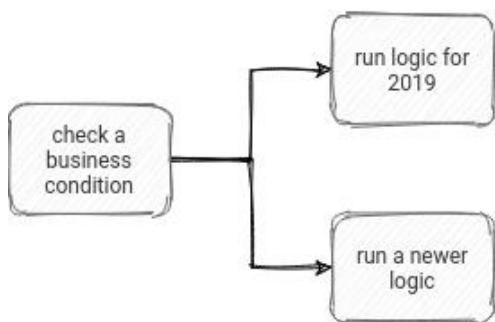
The screenshot shows the Databricks Switch interface. At the top, there's a header bar with a 'Switch' tab, a 'Switch Databricks Clusters' icon, and several small icons. Below the header is a toolbar with search, add, and filter buttons. The main area has tabs for 'General', 'Settings', 'Cases', and 'User properties', with 'Cases' being the active tab. A large button labeled '+ Add case' is visible. The 'Cases' table lists four entries:

Case	Activity	Actions
Default	Do Nothing	<button>Edit activities</button>
Dev	Transform Data on Dev Cluster	<button>Edit activities</button> <button>Delete</button>
Test	Transform Data on Test Cluster	<button>Edit activities</button> <button>Delete</button>
Prod	Transform Data on Prod Cluster	<button>Edit activities</button> <button>Delete</button>

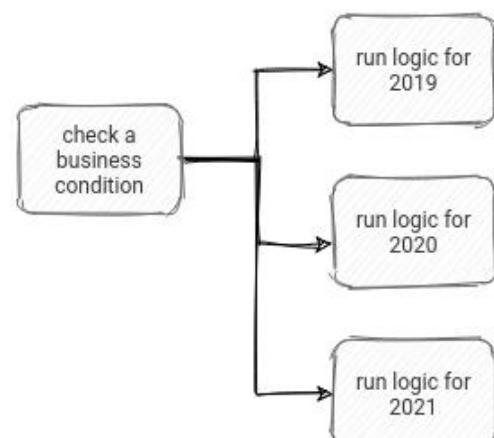
Key Points:

- **Multi-Condition:** Allows defining multiple conditions (cases) and corresponding activities or control flows for each case.
- **Structured Flow:** Offers a more structured approach to handle multiple scenarios compared to nested If Condition activities.
- **Enhanced Readability:** Improves pipeline readability when dealing with several branching scenarios by organizing them into distinct cases.

If-else conditional

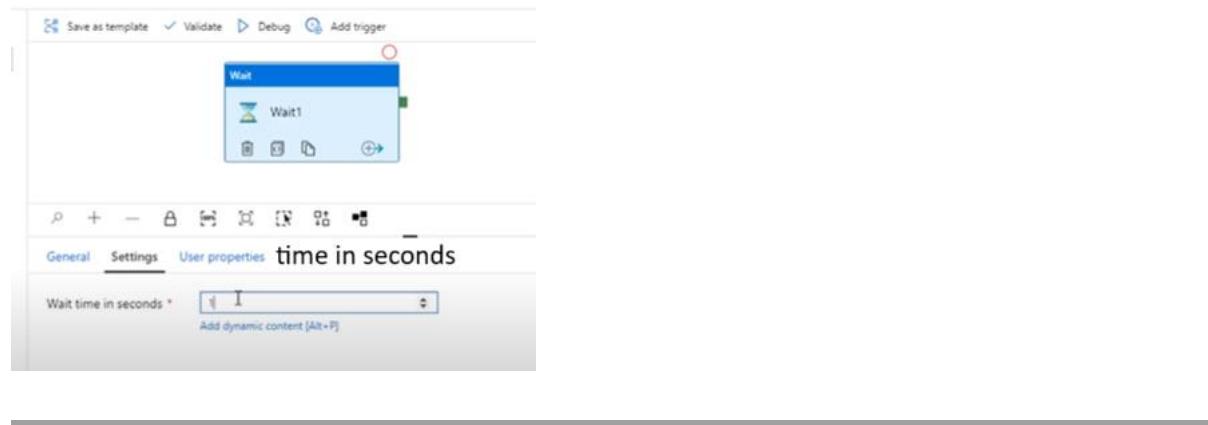


switch conditional



Wait Activity in ADF

When you use a wait activity in a pipeline, the pipeline waits for the specified period of time before continuing with execution of subsequent activities.

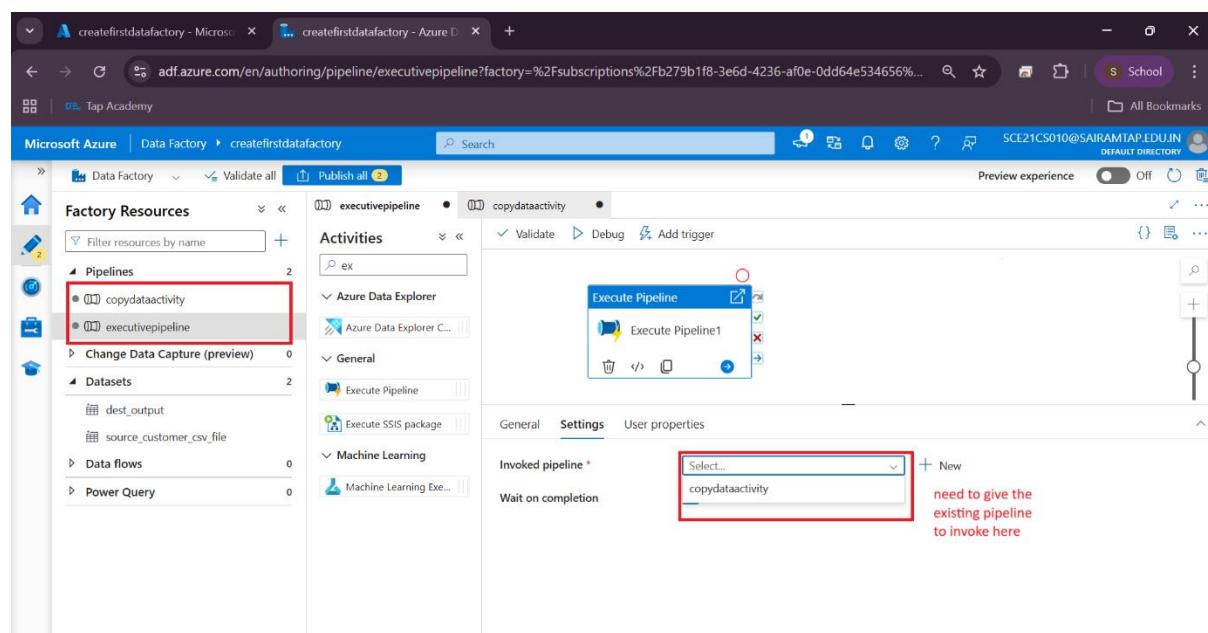


Execute Pipeline Activity:

This activity is essential when you need to chain pipelines together and execute them in a sequence, ensuring a smooth workflow from one process to another.

Execute Pipeline Activity in Azure Data Factory allows one pipeline to invoke and execute another pipeline, enabling modular, reusable, and sequential workflows.

This activity allows you to call and run another pipeline within a pipeline. It helps you break down complex workflows into smaller, modular components, which can then be executed in a sequence or based on certain conditions.



Wait on completion: wait for the selected pipeline(child pipeline) to complete (if checked)

There is difference between parameter and variable:

For Parameter value is passed in the Triggers

Parameter are external values passed into pipelines, datasets or linked services. The value cannot be changed inside a pipeline.

Variable → set values for these variables inside pipelines

variable are internal values set inside a pipeline. The value can be changed inside the pipeline using set variable or append variable activity.

Variable in ADF:

Variables are like internal to pipeline and they can be changed inside your pipeline.

Variables support 4 data type: **string, bool, array, integer**

@variables('variableName')

Set variable activity

If we want to set the value for the variable during the pipeline execution, we can use this.

Use the set variable activity to set the value of an existing variable of type **string, bool or array** defined in a Data factory pipeline.

The screenshot shows the Microsoft Azure Data Factory pipeline editor. On the left, the 'Factory Resources' sidebar lists 'Pipelines' (1), 'Datasets' (2), 'Data flows' (0), and 'Power Query' (0). The 'Pipelines' section is expanded, showing 'pipeline1'. In the main workspace, a pipeline named 'pipeline1' is open. The 'Activities' tab is selected, showing a 'set' activity. Below it, under 'General', is a 'Set variable' activity. At the bottom of the screen, a red box highlights the 'Variables' tab of the pipeline properties panel. This panel shows two variables: 'variable_copy' and 'variable_available', both of type 'String' with 'Value' as the default value. There is also a 'New' button for creating new variables.

Microsoft Azure | Data Factory > createfirstdatafactory

Search Preview experience Off

Factory Resources Pipelines pipeline1 Activities General Set variable

Set variable1 (X) Set variable1

show the available variable

General Settings User properties

Variable type Pipeline variable Pipeline return value

Name * variable_copy variable_available

```
graph TD; Start(( )) --> SetVariable1[Set variable1]; SetVariable1 --> End(( ));
```

✓ Validate ⚡ Debug ⚡ Add trigger

Set variable (X) Set variable1

General Settings User properties

Variable type Pipeline variable Pipeline return value

Name * a + New

Value @utcNow()

```
graph TD; Start(( )) --> SetVariable1[Set variable1]; SetVariable1 --> End(( ));
```

Microsoft Azure | Data Factory > createfirstdatafactory

Search

Validate all

Publish all 1

Factory Resources

- Pipelines 1
- pipeline1

Activities

- Set variable
- Set variable
- Set variable

Pipeline expression builder

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

```
@concat(variables('a'),variables('b'))
```

Clear contents

Parameters System variables Functions Variables

General Settings User properties

Variable type

Name *

Value *

OK Cancel

The screenshot shows the Pipeline expression builder interface. It displays the expression `@concat(variables('a'),variables('b'))`. Below the expression, there are tabs for Parameters, System variables, Functions, and Variables, with Variables selected. Under Variables, there are three entries: 'a', 'b', and 'concatvariable'. At the bottom are 'OK' and 'Cancel' buttons.

✓ Validate ▶ Debug ⚡ Add trigger



General Settings User properties

Variable type i

Pipeline variable Pipeline return value

Name *

concatvariable

+ New

Value

@concat(variables('a'),variables('b'))

✓ Validate ▶ Debug ⚡ Add trigger

Set variable (x) a variable
Set variable (x) concat variable
Set variable

Parameters Variables Settings Output

Pipeline run ID aac1da36-5e66-4edc-9adf-7360930348fd ⏪ Pipeline status ✓ Succeeded View debug run co

All status ▾ Monitor in Azure Metrics Export to

Showing 1 - 3 of 3 items

Activity name ↑↓	Activity st... ↑↓	Activit... ↑↓	Run start ↑↓	Duration ↑↓	Integration runtime
concat variable	✓ Succeeded	Set variable	9/2/2025, 12:43:04 PM	Less than 1s	
b variable	✓ Succeeded	Set variable	9/2/2025, 12:43:04 PM	Less than 1s	
a variable	✓ Succeeded	Set variable	9/2/2025, 12:43:04 PM	Less than 1s	

Output

Copy to clipboard

```
{ "name": "a", "value": "2025-09-02T07:13:04.512852Z" }
```

Pipeline status ✓ Succeeded Monitor in Azure Met

Run start ↑↓	Duration ↑↓
9/2/2025, 12:43:04 PM	Less than 1s
9/2/2025, 12:43:04 PM	Less than 1s

a variable	✓ Succeeded	Set variable	9/2/2025, 12:43:04 PM	Less than 1s
------------	-------------	--------------	-----------------------	--------------

Output

```
{
  "name": "b",
  "value": "2025-13-02"
}
```

Run History

Run ID	Status	Start Time	Duration
b variable	Succeeded	9/2/2025, 12:43:04 PM	Less than 1s
a variable	Succeeded	9/2/2025, 12:43:04 PM	Less than 1s

Output

```
{
  "name": "concatvariable",
  "value": "2025-09-02T07:13:04.512852Z2025-13-02"
}
```

Pipeline status Succeeded

Run History

Run ID	Status	Start Time	Duration	Integration run
concat variable	Succeeded	9/2/2025, 12:43:04 PM	Less than 1s	
b variable	Succeeded	9/2/2025, 12:43:04 PM	Less than 1s	
a variable	Succeeded	9/2/2025, 12:43:04 PM	Less than 1s	

Append Variable

The **Append Variable** activity is used to add (append) values to an **array variable** during pipeline execution.

- You **cannot use it with string or int variables**, only with **Array type variables**.
- Each execution appends a new element to the existing array.

Steps to Use Append Variable

1. **Create a Variable** in your pipeline (type = Array).

```
"variables": {
```

```
"FileList": {  
    "type": "Array",  
    "defaultValue": []  
}  
}
```

2. Add Append Variable Activity

- Select the variable (FileList).
- Provide the value to append (can be dynamic expression).

@concat('file_', pipeline().RunId, '.csv')

Example

Let's say you loop through a list of file names and want to store them:

- Initial variable (FileList): []
- First Append Variable: append "customer.csv" → ["customer.csv"]
- Second Append Variable: append "sales.csv" → ["customer.csv", "sales.csv"]
- Third Append Variable: append "product.csv" → ["customer.csv", "sales.csv", "product.csv"]

Real-time scenarios

- Collect file names processed in a loop.
- Store failed records file names for logging.
- Build a dynamic list of values to use later in the pipeline (for email notifications, logging tables, etc.).

Store Procedure

A stored procedure is a prepared SQL code that you can save, so the code can be reused again.

[What is stored procedure (sql)? A **stored procedure** is a **pre-compiled set of SQL statements** (and optional control logic) that you save inside a database and execute by name whenever you need it].

Stored Procedure activity

The **Stored Procedure activity** lets you call a stored procedure that already exists inside a database (for example Azure SQL Database, Azure Synapse SQL Pool, or SQL Server).

So, if you have an SQL query that you write again, save it as a stored procedure, and then just call it to execute it.

You can also pass parameters to a stored procedure, so that the stored procedure can act based on the parameters value that is passed.

You can use the stored procedure in

- Azure SQL Database
- Azure synapse analytics
- SQL server database

Microsoft Azure

Home > Microsoft.SQLDatabase.newDatabaseNewServer_cad2ee8df8ad42e992721 | Overview > pocdatabase (pocserver2671818/pocdatabase)

pocdatabase (pocserver2671818/pocdatabase) | Query editor (preview)

Search Login New Query Open query Feedback Getting started

Overview Activity log Tags Diagnose and solve problems Query editor (preview)

Settings Compute + storage Connection strings Properties Locks Data management Replicas Sync to other databases Integrations Azure Synapse Link Stream analytics (preview)

pocdatabase (manish)

Showing limited object explorer here. For full capability please click here to open Azure Data Studio.

Tables Views Stored Procedures

Query 1

Run Cancel query Save query Export data as Show only Editor

```
3 name varchar(128),
4 country varchar(128)
5 )
6
7 insert into employee values(1,'manish','india')
8 insert into employee values(2,'manish','USA')
9 insert into employee values(3,'manish','india')
10 insert into employee values(4,'manish','india')
11 insert into employee values(5,'manish','india')
12
13 create procedure emp_delete @id int as
14 delete from employee where id=@id
15
16 exec emp_delete @id=1
```

Results Messages

Query succeeded: Affected rows: 5

Microsoft Azure | Data Factory > pocdfsysu288292

Microsoft recently announced the public preview of Microsoft Fabric, a brand new and exciting way to build cloud-first data analytics. Click here to get started with Fabric Data Factory!

Validate all Publish all

Factory Resources

- Pipelines
 - pipeline1
 - Change Data Capture (preview) 0
 - Datasets 0
 - Data flows 0
 - Power Query 0
- Activities
 - Stored procedure
 - Stored procedure1
- General
 - Append variable
 - Delete
 - Execute Pipeline
 - Execute SSIS package
 - Fail
 - Get Metadata
 - Lookup
 - Script

Activities

Stored procedure

Stored procedure1

New linked service

Name * linkedService1

Description

Type * Amazon RDS for SQL Server Learn more

Amazon RDS for SQL Server

Amazon RDS for MySQL

Amazon RDS for PostgreSQL

Amazon RDS for Oracle

Azure SQL Database

Azure SQL Database Managed Instance

Azure Synapse Analytics

Microsoft Fabric Warehouse

SQL server

Authentication type * SQL authentication

User name *

Password * Azure Key Vault

Always encrypted

Additional connection properties

+ New

The screenshot shows the Azure Data Factory Studio interface. At the top, there is a connection configuration pane for a 'Connection string' with tabs for 'Connection string' and 'Azure Key Vault'. It includes fields for 'Account selection method' (set to 'From Azure subscription'), 'Azure subscription' (selected), 'Server name' (pocserver2671818), 'Database name' (pocdatabase), 'Authentication type' (SQL authentication), 'User name' (manish), and 'Password' (hidden). Below this is another pane for 'Azure Key Vault' with a 'Password' field containing '.....' and a note to 'Add dynamic content [Alt+Shift+D]'. A feedback rating bar is visible on the right.

The main workspace shows a pipeline named 'pipeline1'. On the left, the 'Activities' pane lists various options like 'Move and transform', 'Synapse', 'Azure Data Explorer', etc., with 'General' expanded. Under 'General', 'Stored procedure' is selected. In the center, a 'Stored procedure' activity is being configured. The 'Settings' tab is active, showing a 'Linked service' dropdown set to 'linkedService1', a 'Stored procedure name' dropdown with 'Select...' and a filter input, and a 'Stored procedure parameters' section with a dropdown listing '[dbo].[emp_delete]', '[dbo].[uspLogError]', and '[dbo].[uspPrintError]'. The 'General' tab shows 'Linked service' set to 'linkedService1', 'Stored procedure name' set to '[dbo].[emp_delete]', and 'Import' selected in the 'Stored procedure parameters' section.

Until activity

The until activity is a looping activity in ADF pipeline

It repeatedly executes one or more activity until a condition is met (like a WHILE LOOP)

(Until it work the given condition becomes TRUE)

Expression (Condition):

- Defines when the loop should **stop**.
- Written using ADF expression functions (e.g., @equals(variables('status'), 'Success')).

Activities inside Until:

- You can put multiple activities inside (Copy, Lookup, Stored Proc, Web activity, etc.).
- They will repeat until the condition is true.

Timeout:

- Prevents infinite looping.
- Example: 7.00:00:00 → loop stops after 7 days even if condition is not met.

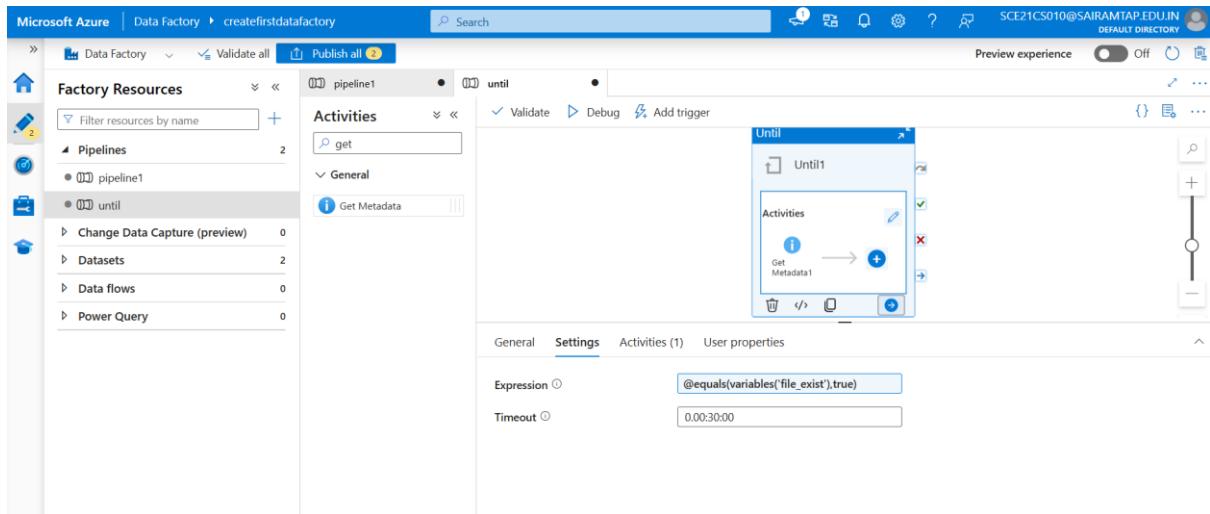
Real-Time Uses

- Waiting for files to land in storage.
(Wait Until File Arrives)
Suppose ADF must wait until a file is available in Azure Blob before proceeding)
- Polling an external API until processing completes.
- Checking for flag/status in a control table.
- Retrying logic until resource is ready.

Set the variable as **false** (until it becomes **TRUE** this will until activity will work)

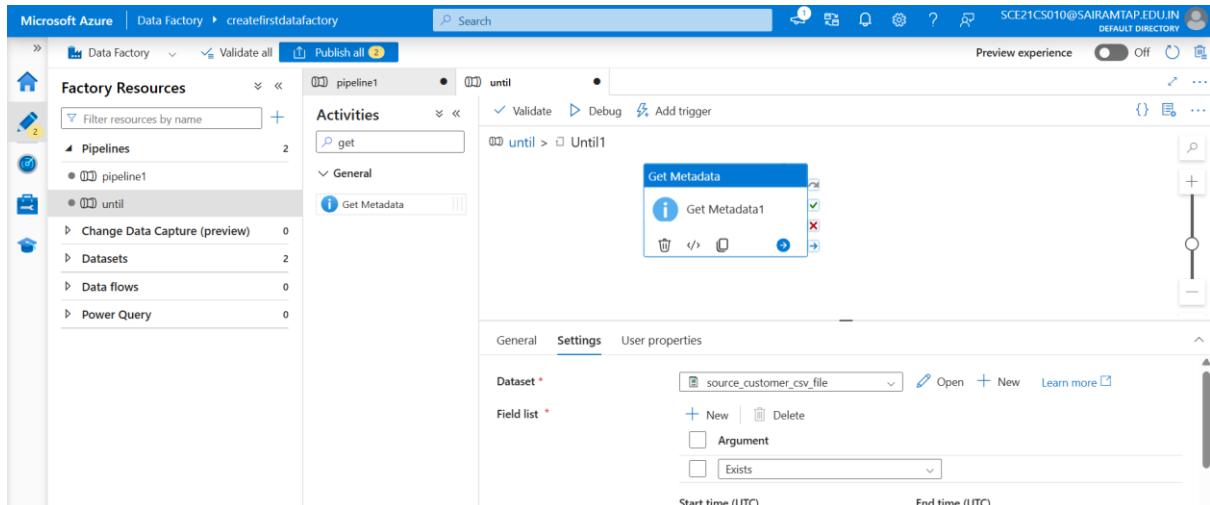
The screenshot shows the Microsoft Azure Data Factory pipeline editor. On the left, the 'Factory Resources' sidebar lists 'Pipelines' (pipeline1) and 'until' (2). The main workspace displays a pipeline named 'pipeline1'. An 'until' activity is selected, indicated by a red circle with a dot. The 'Activities' pane shows a 'set' activity. Below the activities, the 'Variables' tab is selected, showing a table with one row:

Name	Type	Default value
file_exist	Boolean	false



Get Metadata Activity

- Point it to the **file path** in your Blob dataset.
- Under *Field list*, check **Exists**.
- This will return "exists": true/false.



Output of this get metadata activity given to the set variable to update it dynamically.

Microsoft Azure | Data Factory > createfirstdatafactory

Search

SCE21CS010@SAIRAMTAP.EDU.IN
DEFAULT DIRECTORY

Preview experience Off

Factory Resources

- Pipelines
- until
- Change Data Capture (preview)
- Datasets
- Data flows
- Power Query

Filter resources by name

Activities

Get Metadata

General Settings User properties

Dataset * source_customer_csv_file

Field list *

- + New Delete
- Argument
- Exists

Start time (UTC) End time (UTC)

Filter by last modified

Skip line count

This screenshot shows the Microsoft Azure Data Factory pipeline editor. A 'Get Metadata' activity is selected in the 'Activities' pane. The 'General' tab is active in the settings panel, showing a dataset named 'source_customer_csv_file'. Below it, a 'Field list' section includes options for 'New', 'Delete', 'Argument', and 'Exists'. The 'Settings' tab is also visible.

Microsoft Azure | Data Factory > createfirstdatafactory

Search

SCE21CS010@SAIRAMTAP.EDU.IN
DEFAULT DIRECTORY

Preview experience Off

Factory Resources

- Pipelines
- until
- Change Data Capture (preview)
- Datasets
- Data flows
- Power Query

Filter resources by name

Activities

Set variable

General

Until

Activities

Get Metadata1 → Set variable1

General Settings User properties

Variable type Pipeline variable

Name * file_exist

Value @activity('Get Metadata1').output.exi...

This screenshot shows the Microsoft Azure Data Factory pipeline editor. A 'Set variable' activity is selected in the 'Activities' pane. The 'General' tab is active in the settings panel, showing a variable named 'file_exist' of type 'Pipeline variable'. The value is set to '@activity('Get Metadata1').output.exi...'. The 'Settings' tab is also visible.

Microsoft Azure | Data Factory > createfirstdatafactory

Search

SCE21CS010@SAIRAMTAP.EDU.IN
DEFAULT DIRECTORY

Preview experience Off

Factory Resources

- Pipelines
- until
- Change Data Capture (preview)
- Datasets
- Data flows
- Power Query

Filter resources by name

Activities

Set

General

Until

Activities

Get Metadata1 → Until1

Output

{ "exists": true, "effectiveIntegrationRuntime": "AutoResolveIntegrationRuntime (South India)", "executionDuration": 1, "durationInQueue": { "integrationRuntimeQueue": 10 }, "billingReference": { } }

Pipeline status Succeeded

Run start 9/2/2025, 6:04:36 PM Duration Less than 1s

Integration runtime AutoResolveIntegrat

Get Metadata1 Until1

This screenshot shows the Microsoft Azure Data Factory pipeline editor after the Until loop has been executed. The 'Until' activity is now succeeded. The 'Output' pane displays the results of the 'Get Metadata' activity. The 'Pipeline status' is shown as 'Succeeded'. The 'Run start' was at 9/2/2025, 6:04:36 PM, and the 'Duration' was 'Less than 1s'. The 'Integration runtime' is listed as 'AutoResolveIntegrat'.

The screenshot shows the Azure Data Factory Pipeline Editor interface. At the top, there's a toolbar with icons for copy to clipboard, refresh, and pipeline navigation. Below the toolbar, the pipeline canvas shows a single activity named 'variable1'. On the right side, there's a 'Run History' section titled 'Monitor in ADF' with a table showing three completed runs:

Activity	Status	Start Time
Set variable1	Succeeded	9/2/2025, 6:04:36 PM
Get Metadata1	Succeeded	9/2/2025, 6:04:21 PM
Until1	Succeeded	9/2/2025, 6:04:20 PM

Web Activity in ADF

The **Web Activity** lets us to **call REST API** (HTTP APIs) from inside your pipeline. It's useful when you need to **interact with external services or trigger jobs** from ADF.

Used to make a **simple HTTP request** and optionally capture the response for later activities.

You can pass datasets and linked services to be consumed and accessed by the activity.

Key Properties

1. **URL** → The endpoint you're calling (e.g., <https://www.dummy.restapiexample.com/>).
2. **Method** → GET, POST, PUT, DELETE, Patch(partially update a resource.)
3. **Headers** → You can pass things like Content-Type: application/json, **Authorization**: Bearer <token>.
4. **Body** → Used in POST or PUT calls to send data (must be valid JSON).
5. **Authentication** (supports multiple types):
 - None
 - Basic (username/password)
 - MSI (Managed Identity)
 - Service Principal

<https://www.dummy.restapiexample.com>

Mysore Pak, Laddu & More
Discover a wide variety of traditional sweets and snacks at Aswins. Aswins Sweets and Snacks

This page will list all of the rest services. These are fake online REST APIs for testing and prototyping sample applications that use rest calls to display listings and crud features. This rest api tutorials, faking a server, and sharing code examples can all be used.

There are following public apis

#	Route	Method	Type	Full route	Description	Details
1	/employee	GET	JSON	https://dummy.restapiexample.com/api/v1/employees	Get all employee data	Details
2	/employee/{id}	GET	JSON	https://dummy.restapiexample.com/api/v1/employee/1	Get a single employee data	Details
3	/create	POST	JSON	https://dummy.restapiexample.com/api/v1/create	Create new record in database	Details
4	/update/{id}	PUT	JSON	https://dummy.restapiexample.com/api/v1/update/1	Update an employee record	Details
5	/delete/{id}	DELETE	JSON	https://dummy.restapiexample.com/api/v1/delete/2	Delete an employee record	Details

Restapiexample © 2017-2019 [API testing tools](#) [Data modeling tools](#)

Microsoft Azure | Data Factory > createfirstdatafactory

Validate all Publish all

Factory Resources

- Pipelines
 - pipeline1
 - webactivity
- until
- Change Data Capture (preview)
- Datasets
- Data flows
- Power Query

Activities

- we
- General
- Web
- WebHook
- Power Query

pipeline1 until webactivity

Validate Debug Add trigger

Web

General Settings User properties

URL: <https://dummy.restapiexample.com/api/v...>
 Information will be sent to the URL specified. Please ensure you trust the URL entered.

Method: GET
 Authentication: None
 Headers: + New

Advanced

Web

Input

Copy to clipboard

```
{
  "method": "GET",
  "headers": {},
  "url": "https://dummy.restapiexample.com/api/v1/employees"
}
```

Run start

Web1 Queued Web 9/2/2025

The screenshot shows the Microsoft Azure Data Factory interface. On the left, the 'Factory Resources' sidebar lists 'Pipelines' (pipeline1, webactivity, until), 'Datasets' (dest_output, source_customer_csv_file), and other components. The main workspace displays a pipeline named 'pipeline1' with three activities: 'cop', 'until', and 'webactivity'. The 'Output' pane shows the results of the 'cop' activity, which copied data from 'source_customer_csv_file' to 'dest_output'. The output JSON is displayed as:

```

{
  "data": [
    {
      "id": 1,
      "employee_name": "Tiger Nixon",
      "employee_salary": 320800,
      "employee_age": 61,
      "profile_image": ""
    }
  ]
}

```

The pipeline status is 'Succeeded' with a duration of 6s. The run started on 9/2/2025, 6:22:08 PM.

For post

The screenshot shows the 'Settings' tab for a 'Web' activity named 'Web1' in the pipeline. The configuration includes:

- URL:** https://dummy.restapitexample.com/api/v...
- Method:** POST
- Body:** (empty)
- Authentication:** None

Web Hook activity

The **Webhook activity** in Azure Data Factory is a powerful feature that allows you to interact with external systems or services by **calling an endpoint and waiting for a callback response**.

Used when you need to wait for an external callback (for example, to wait for a long-running external process to signal completion).

Web Activity → just calls an API and moves on (doesn't wait).
Webhook Activity → calls an API (usually long-running jobs) and **waits for a callback response** before pipeline continues.

How It Works

- Trigger External Service:** The Webhook activity sends a request to an external endpoint (e.g., REST API) (post request).

2. **Callback URL:** ADF provides a callback URL in the request payload. The external service must invoke this URL once its processing is complete.
3. **Wait for Completion:** The pipeline execution pauses until the callback URL is invoked or a timeout occurs.

The screenshot shows the Microsoft Azure Data Factory pipeline editor. On the left, the 'Factory Resources' sidebar lists 'Pipelines' (pipeline1, webhook, until, webactivity), 'Datasets' (dest_output, source_customer_csv_file), and other components. In the main workspace, a pipeline named 'pipeline1' is selected. Under the 'Activities' section, a 'Web' activity is chosen, which is then expanded to show a 'WebHook' activity. The 'Settings' tab is active, displaying fields for 'URL' (a placeholder URL), 'Method' (set to 'POST'), and 'Headers' (empty). Below these, there's a 'Body' field with a placeholder 'POST'.

- **Scenario:** Trigger a Logic App to process data and wait for its completion before moving to the next pipeline step.

Supported HTTP Methods:

- POST
- PUT

Important notes:

- The external service **must call back** to the callbackUrl provided by ADF, sending a status payload (usually HTTP 200) to mark the activity as complete.
- If the service never calls back within the timeout, the activity fails.

Aspect	Web Activity	Webhook Activity
Purpose	Make a simple HTTP call and immediately get a response.	Start an external process and wait for that process to call back when it's done.
Execution flow	Pipeline continues as Pipeline stays in <i>Running</i> state until an response is returned.	Pipeline stays in <i>Running</i> state until an external system sends a callback to the provided URL or the timeout expires.
Supported HTTP methods	GET, POST, PUT, DELETE, PATCH	POST, PUT

Aspect	Web Activity	Webhook Activity
Callback required	No	Yes—external service must POST to the callbackUrl included in the request body/headers.
Typical cases	<ul style="list-style-type: none"> Call a REST API to fetch or update data. Trigger a microservice and move on. 	<ul style="list-style-type: none"> Trigger a long-running job (e.g., build, ML training) and wait for a “job complete” notification. Ensure downstream steps don’t start until an external workflow signals success/failure.
Timeout behavior	Fails only if the initial HTTP call fails or times out.	Fails if the external service doesn’t call back within the configured timeout.
Return value	Immediate response body Return value is available to later activities.	Final status is determined by the callback payload.

Use Web Activity when you just need to send/receive a single HTTP request/response.

Use Webhook Activity when you must wait for an external system to confirm completion before the pipeline proceeds.

Databricks activity

The screenshot shows the Databricks workspace interface. On the left is a sidebar with navigation links: + New, Workspace, Recents, Catalog, Jobs & Pipelines, Compute, Data Engineering, Job Runs, AI/ML, Playground, Experiments, Features, Models, and Serving. The main area displays the 'Welcome to Databricks' page with a search bar, a 'Set up your workspace' guide, and a list of recent items including 'Sales_Notebook /sales'. A red arrow points to the user profile icon in the top right corner.

The screenshot shows the user profile dropdown menu for 'Sales_Data'. It includes options: Stewart Prince, stewart.pm3003@gmail.com, Settings (with a red arrow pointing to it), Privacy policy, Previews, Send feedback, and Log out.

The screenshot shows the 'Settings' page in the Databricks workspace. The sidebar on the left is identical to the previous screenshots. The main content area is divided into two sections: 'Settings' on the left and 'Profile' on the right. The 'Settings' section contains links for Workspace admin, Identity and access, Security, Compute, Development, Notifications, Advanced, User (with 'Profile' highlighted), Preferences, and Developer (which is highlighted with a red box). The 'Profile' section shows the user's display name (Stewart Prince) and email (stewart.pm3003@gmail.com), group memberships (admins, users), and linked accounts.

The screenshot shows the Databricks Settings page under the Developer tab. A red box highlights the "Access tokens" section, which contains the sub-section "Set up secure authentication to Databricks API using access tokens". A "Manage" button is located in the top right corner of this section.

The screenshot shows the Databricks Settings page under the Developer tab, specifically the "Access tokens" section. A red arrow points to the "Generate new token" button. The table below shows no tokens exist:

Comment	Creation	Expiration
No tokens exist.		

Microsoft Azure databricks

Search data, notebooks, recents, and more... CTRL + P

+ New

- Workspace
- Recents
- Catalog
- Jobs & Pipelines *
- Compute
- Data Engineering
- Job Runs
- AI/ML
 - Playground
 - Experiments
 - Features
 - Models
 - Serving

Settings

User settings > Developer > Access tokens

Generate new token

Comment: toconnect adf to databricks

Lifetime (days): 1

Cancel Generate

Expiration

instead of passwords

Microsoft Azure databricks

Search data, notebooks, recents, and more... CTRL + P

+ New

- Workspace
- Recents
- Catalog
- Jobs & Pipelines *
- Compute
- Data Engineering
- Job Runs
- AI/ML
 - Playground
 - Experiments
 - Features
 - Models
 - Serving

Settings

User settings > Developer > Access tokens

Generate new token

Your token has been created successfully.

dapi5adde246fc3c68a46ff34ff62fb5e2cc

⚠ Make sure to copy the token now. You won't be able to see it again.

Done

Expiration

instead of passwords

User settings > Developer >

Access tokens

Personal access tokens can be used for secure authentication to the [Databricks API](#) instead of passwords.

[Generate new token](#)

Comment	Creation	Expiration
toconnect adf to databricks	2025-09-02 16:54:11 IST	2025-09-03 16:54:11 IST

Create linked service for databricks (go to compute)

New linked service

Compute

All Azure Compute

Azure Databricks

Continue Cancel

New linked service

Azure Databricks

Connect via integration runtime

AutoResolveIntegrationRuntime

Account selection method

From Azure subscription Enter manually

Azure subscription

Azure for Students (0327f472-b106-4e36-9ed5-5cf88535847b)

Databricks workspace

Sales_Data

Select cluster

New job cluster Existing interactive cluster Existing instance pool

Databrick Workspace URL

https://adb-3682507721380047.azuredatabricks.net

Create Back Test connection Cancel

Compute

All-purpose compute Job compute Pools

Filter compute you have access to Created by Only pinned

Create compute

No compute

Create compute

The screenshot shows the 'Create new compute' page in the Databricks UI. On the left, there's a sidebar with 'Compute' selected under 'Cloud'. The main area has tabs for 'Performance' (selected), 'Machine learning', and 'Advanced performance'. Under 'Performance', there are sections for 'Databricks runtime' (set to 11.3 LTS, Scala 2.12, Spark 3.3.0, with 'Photon acceleration' checked), 'Node type' (Standard_D3_v2, 14 GB Memory, 4 Cores, 'Single node' checked), and a checkbox for 'Terminate after 10 minutes of inactivity'. A red box highlights this section.

The screenshot shows the configuration page for a cluster named 'Stewart Prince's Cluster' created on 2025-09-02 at 17:10:26. The 'Configuration' tab is selected. It displays basic information like 'Summary' (14 GB Memory, 4 Cores) and 'Price' (1.5 DBU/h). The 'General' section includes 'Compute name' (Stewart Prince's Cluster 2025-09-02 17:10:26) and 'Policy' (Unrestricted). The 'Performance' section shows 'Machine learning' is off, and 'Databricks runtime' is set to 11.3 LTS (includes Apache Spark 3.3.0, Scala 2.12) with 'Photon acceleration' checked. A blue box highlights the 'Compute name' field.

Cluster created

Connect linked service

The screenshot shows the 'Linked services' page in the Microsoft Azure Data Factory UI. The left sidebar has 'Linked services' selected. The main area lists three items: 'AzureDataLakeStorage1_http', 'linked_git', and 'ls_blob_sales'. To the right, there's a 'New linked service' form for connecting to 'Azure Databricks'. It includes fields for 'Databrick Workspace URL' (https://adb-3682507721380047.azuredatabricks.net), 'Authentication type' (Access Token selected), 'Access token' (filled with a long string of asterisks), and 'Choose from existing clusters' (Stewart Prince's Cluster 2025-09-02 17:10:26 selected). A green checkmark indicates 'Connection successful'. A blue box highlights the 'Access token' field.

Linked services

Linked service defines the connection information to a data store or compute. [Learn more](#)

+ New

Filter by name Annotations : Any

Showing 1 - 4 of 4 items

Name ↑↓	Type ↑↓	Related ↑↓	Annotations ↑↓
AzureDataLakeStorage1_http	Azure Data Lake Storage Gen2	4	
linked_git	HTTP	1	
ls_blob_sales	Azure Blob Storage	2	
sales_linked_service	Azure Databricks	0	

Microsoft Azure | Data Factory > geneworkspace

Search Publish all

Factory Resources

Pipelines

- activity
- lookup
- selected_files

Change Data Capture (preview)

Datasets

- destination_sink
- ds_cleanedsales
- ds_raw_sales
- getmetadaaset
- param_csv
- sink_http
- src_git

Activities

- Move and transform
- Synapse
- Azure Data Explorer
- Azure Function
- Batch Service
- Databricks
- Data Lake Analytics
- General
- HDInsight
- Iteration & conditionals
- Machine Learning
- Power Query

Lookup

Notebook

Lookup1

Notebook1

General Azure Databricks Settings User properties

Databricks linked service * sales_linked_service

Test connection Edit New

Preview experience Off

Microsoft Azure | Data Factory > geneworkspace

Search Publish all

Factory Resources

Pipelines

- activity
- lookup
- selected_files

Change Data Capture (preview)

Datasets

- destination_sink
- ds_cleanedsales
- ds_raw_sales
- getmetadaaset
- param_csv
- sink_http
- src_git

Activities

- Move and transform
- Synapse
- Azure Data Explorer
- Azure Function
- Batch Service
- Databricks
- Data Lake Analytics
- General
- HDInsight
- Iteration & conditionals
- Machine Learning
- Power Query

Lookup

Notebook

Sales_Notebook

Browse

Select a file or folder.

Root folder > sales

Sales_Notebook

Showing 1 item

OK Cancel

The screenshot shows the Microsoft Azure Data Factory pipeline editor. On the left, the 'Factory Resources' sidebar lists Pipelines, Datasets, and other resources. In the main workspace, a pipeline is being edited with two activities: 'Lookup' and 'Notebook'. The 'Notebook' activity is selected. In the 'Settings' tab, the 'Notebook path' is specified as '/sales/Sales_Notebook'. A red box highlights this path entry.

The screenshot shows the Microsoft Azure Data Factory pipeline editor after a run has completed. The pipeline run ID is C960A8-C519-45D8-B2E3-05C0413eebc3. The Pipeline status is listed as 'Succeeded'. Below the status, a table provides details for each activity in the pipeline:

Activity name	Activity status	Activit...	Run start	Duration
Notebook1	Succeeded	Notebook	9/2/2025, 5:35:55 PM	38s
Lookup1	Succeeded	Lookup	9/2/2025, 5:35:47 PM	7s

Filter activity

The screenshot shows the Microsoft Azure Data Factory pipeline editor. A 'Get Metadata' activity is followed by a 'Filter' activity. The 'Filter' activity is selected, and its configuration is shown in the 'Settings' tab. To the right, the 'Pipeline expression builder' is open, displaying dynamic content for the 'Filter' activity. The builder shows the expression @activity('Get Metadata1').output.childItems.

The screenshot shows the Microsoft Azure Data Factory interface. On the left, the 'Factory Resources' sidebar lists Pipelines (pipeline1, pipeline2, pipeline3, pipeline4, pipeline5), Datasets (DelimitedText1, DelimitedText2), Data flows, and Power Query. The main workspace displays pipeline5, which contains a 'Get Metadata' activity followed by a 'Filter' activity. The 'General' tab of the pipeline properties is selected, showing the output path as '@activity('Get Metadata').output.ch...'. The 'Output' pane shows the JSON output of the 'Get Metadata' activity, which includes an array of items with names like 'department.csv' and types like 'File'. The 'Properties' pane on the right shows the pipeline's name as 'pipeline5' and its status as 'Succeeded'. The 'Pipeline status' section indicates a successful run on 5/17/2024 at 5:35:21 PM.

Azure Data Factory Pipeline Activities

- Move and transform

Activity	What it does	Real-time scenario
Copy Data	Moves data from a source to a sink (with optional schema mapping).	Copy CSV from Blob → Azure SQL Table.
Data Flow	Runs Mapping Data Flows (row-level transformations like join, aggregate, conditional split).	Cleanse customer data, handle SCD Type 2, load fact tables.

2. Azure Data Lake / Databricks

Activity	What it does	Scenario
Databricks Notebook / Jar / Python	Runs Databricks jobs.	Run PySpark to flatten JSON or apply SCD logic.
HDInsight Hive / Pig / MapReduce / Spark	Run big data jobs on HDInsight cluster.	Legacy Hadoop workloads.

3. General Activities

Activity	What it does	Scenario
Lookup	Reads a single row or dataset from a source for use in pipeline.	Get latest watermark value from SQL table.
Get Metadata	Fetches file/table properties (size, modified date, schema).	Check if file exists before processing.
Delete	Deletes files/folders from storage.	Clean up old files after load.
Web	Calls REST APIs (fire and forget).	Trigger an API that refreshes Power BI dataset.
Webhook	Calls API and waits for callback.	Trigger ML job and wait until training finishes.
Stored Procedure	Calls SQL stored procedure.	Run merge procedure after staging load.
Execute Pipeline	Runs another pipeline.	Modularize pipelines (ingest pipeline → transform pipeline).
Append Variable	Adds a value to an array variable.	Collect list of file names during iteration.
Set Variable	Assigns value to a variable.	Store dynamic file path for use later.
Validation	Checks existence/availability of data before continuing.	Ensure file is present before Copy Activity.
Wait	Pauses pipeline for set time.	Add 10 min delay before retrying API.

4. Iteration & Conditionals

Activity	What it does	Scenario
If Condition	Branch logic (if-else).	If file size > 1GB, process in Databricks, else Copy directly.
Switch	Case-based branching.	Route data load to bronze, silver, or gold container.
ForEach	Loops over a collection of items.	Process each file in a folder one by one.
Until	Loops until condition = true.	Keep checking for a file until it arrives.

Create sql database

The screenshot shows the Microsoft Azure Marketplace interface for creating a SQL Database. At the top, there's a navigation bar with 'Microsoft Azure', a search bar, and various icons including 'Copilot'. Below the navigation, the breadcrumb path is 'Home > Resource groups > bigdata > Marketplace > SQL Database'. The main content area features a large blue header with the text 'SQL Database' and a 'Create' button. To the left is a thumbnail icon of a database. Below the header, the product name 'SQL Database' is shown with a 'Microsoft | Azure Service' badge, a 4.4 rating from 2623 reviews, and an 'Azure benefit eligible' badge. There are dropdown menus for 'Subscription' (set to 'Azure for Students') and 'Plan' (set to 'SQL Database'). The bottom of the page has tabs for 'Overview' (which is selected), 'Plans', 'Usage Information + Support', and 'Ratings + Reviews'.



Home > Resource groups > bigdata > Marketplace > SQL Database >

Create SQL Database

Microsoft

Subscription * ⓘ

Azure for Students

Resource group * ⓘ

bigdata

[Create new](#)

on a combi
more about

Database details

Enter required settings for this database, including picking a logical server and configuring the compute and storage resources

Database name *

sqldatabasebj

Server * ⓘ

(new) bjsqldatabase (East Asia)

[Create new](#)

Want to use SQL elastic pool? ⓘ

Yes No

[Review + create](#)

[Next : Networking >](#)

Create SQL Database Server

Microsoft

Server details

Enter required settings for this server, including providing a name and location. This server will be created in the same subscription and resource group as your database.

Server name *

bjsqldatabase

.database.windows.net

- ✓ Server name specified
- ✓ The specified server name is valid
- ✓ Your server name ends with '.--' in third position

Location *

(Asia Pacific) East Asia

Authentication

i Azure Active Directory (Azure AD) is now Microsoft Entra ID. [Learn more](#)

[OK](#)

bjSQL@5@7

Create SQL Database Server

Microsoft

Select your preferred authentication methods for accessing this server. Create a server admin login and password to access your server with SQL authentication, select only Microsoft Entra authentication [Learn more](#) using an existing Microsoft Entra user, group, or application as Microsoft Entra admin [Learn more](#), or select both SQL and Microsoft Entra authentication.

Authentication method

- Use Microsoft Entra-only authentication
- Use both SQL and Microsoft Entra authentication
- Use SQL authentication

Server admin login *

bjsqlserver

Password *

Confirm password *

OK



Home > Resource groups > bigdata > Marketplace > SQL Database >

Create SQL Database

Microsoft

Workload environment

- Development
- Production

Default settings provided for Development workloads. Configurations can be modified as needed.

Compute + storage *

Basic

2 GB storage

[Configure database](#)

Backup storage redundancy

Choose how your PITR and LTR backups are replicated. Geo restore or ability to recover from regional outage is only available when geo-redundant storage is selected.

[Review + create](#)

[Next : Networking >](#)

Microsoft Azure Search resources, services, and docs (G+/) Copilot 20+ ⚙️ ⓘ 🔍 SCE21CS010@SAIRAMT DEFAULT DIRECTORY (SCE21CS01)

Home > Resource groups > bigdata > Marketplace > SQL Database >

Create SQL Database ...

Microsoft

Basics Networking Security Additional settings Tags Review + create

Product details

SQL database by Microsoft Estimated cost per month 4.90 USD Terms of use | Privacy policy

Cost summary

Basic (Basic) Cost per DTU (in USD) 0.98 DTUs selected x 5 ESTIMATED COST / MONTH 4.90 USD

Terms By clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. For additional details see [Azure Marketplace Terms](#).

Create < Previous Download a template for automation

Microsoft Azure Search resources, services, and docs (G+/) Copilot 20+ ⚙️ ⓘ 🔍 SCE21CS010@SAIRAMT DEFAULT DIRECTORY (SCE21CS01)

Home >

Microsoft.SQLDatabase.newDatabaseNewServer_3a93294439de421cba8d4 | Overview

Deployment

Search Delete Cancel Redeploy Download Refresh

Overview

Your deployment is complete

Deployment name : Microsoft.SQLDatabase.newDa... Start time : 03/09/2025, 17:28:11
Subscription : Azure for Students Correlation ID : 99c9ad5f-17d5-483c-b922-312...
Resource group : bigdata

Deployment details

Next steps

Go to resource

Microsoft Azure Search resources, services, and docs (G+/) Copilot 20+ ⚙️ ⓘ 🔍 SCE21CS010@SAIRAMT DEFAULT DIRECTORY (SCE21CS01)

Home > Microsoft.SQLDatabase.newDatabaseNewServer_3a93294439de421cba8d4 | Overview >

sqldatabasebj (bjsqldatabase/sqldatabasebj) Find query store timeouts Top CPU consuming queries +1

SQL database

Copy Restore Export Set server firewall Delete Connect with... Feedback

Overview

Mirror databases in Microsoft Fabric Easily replicate your existing databases in Fabric, and help your team achieve streamlined ETL and operational analysis goals. Learn more ↗

Essentials

Resource group (move)	Server name
bigdata	bjsqldatabase.database.windows.net
Status	Elastic pool
Online	No elastic pool
Location	Connection strings
East Asia	Show database connection strings
Subscription (move)	Pricing tier
Azure for Students	Basic
Subscription ID	Earliest restore point
b279b1f8-3e6d-4236-af0e-0dd64e534656	No restore point available

The screenshot shows the Microsoft Azure SQL Database Query Editor (preview) interface. At the top, there's a navigation bar with 'Microsoft Azure' and a search bar. Below it, the URL indicates the database is named 'sqldatabasebj'. The main area has a sidebar with options like Overview, Activity log, Tags, Diagnose and solve problems, and Query editor (preview), which is currently selected. The main content area displays a 'Welcome to SQL Database Query Editor' message and a 'SQL server authentication' form. It asks for a 'Login' (bjsqlserver) and a 'Password'. There's also an option for 'Microsoft Entra authentication' and a 'Continue as SCE21CS010@SAIRAMTAPE...' button. An 'OK' button is at the bottom of the auth form.

Scenario

1. We have data available in blob or ADLS as csv format
2. We need to copy data from storage to sql database (table format)

Source is created

Sink

(table created)

This screenshot shows the Microsoft Azure SQL Database Query Editor (preview) after a query has been run. The interface is similar to the previous one, with the 'Query editor (preview)' selected in the sidebar. The main area now shows a 'Query 1' tab with the following SQL code:

```
1 CREATE TABLE Customers (
2     [Index] INT PRIMARY KEY,
3     [CustomerId] INT NOT NULL,
4     [FirstName] NVARCHAR(100),
5     [LastName] NVARCHAR(100),
6     [Comments] nvarchar(200)
7 )
```

The code is highlighted with syntax coloring. Below the code, the 'Messages' tab shows the result: 'Query succeeded: Affected rows: 0'.

New linked service

Azure SQL Database [Learn more](#)

Version

2.0 (Recommended) 1.0

[Import from connection string](#)

Account selection method

From Azure subscription Enter manually

Azure subscription

Azure for Students (b279b1f8-3e6d-4236-af0e-0dd64e534656)

Server name *

bjsqldatabase



Database name *

sqldatabasebj



Authentication type *

SQL authentication

User name *

sa

Create

Cancel

[Test connection](#)

Microsoft Azure | Data Factory > createfirstdatafactory | Search

SCE21CS010@SAIRAMTAP.EDU.IN DEFAULT DIRECTORY

Factory Resources

- Pipelines
- pipeline1
- Change Data Capture (preview)
- Datasets

 - dest_output
 - source_customer_csv_file

- Data flows
- Power Query

Activities

- Move and transform
 - Copy data
 - Data flow
- Synapse
- Azure Data Explorer
- Azure Function
- Batch Service
- Databricks
- Data Lake Analytics
- General
- HDInsight
- Iteration & conditionals

pipeline1

source_customer_csv... pipeline1

Validate all Publish all 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

Search

All Azure Database File Generic protocol

- Azure File Storage
- Azure SQL Database
- Azure SQL Database Managed Instance
- File
- Generic protocol

Continue Cancel

Microsoft Azure | Data Factory > createfirstdatafactory | Search

SCE21CS010@SAIRAMTAP.EDU.IN DEFAULT DIRECTORY

Factory Resources

- Pipelines
- pipeline1
- Change Data Capture (preview)
- Datasets

 - dest_output
 - source_customer_csv_file

- Data flows
- Power Query

Activities

- Move and transform
 - Copy data
 - Data flow
- Synapse
- Azure Data Explorer
- Azure Function
- Batch Service
- Databricks
- Data Lake Analytics
- General
- HDInsight
- Iteration & conditionals

pipeline1

source_customer_csv... pipeline1

Validate all Publish all 1

Set properties

Name: AzureSqlTable1

Linked service: AzureSqlDatabase1

Table name: Select...
dbo.Customers

Import schema: From connection/store None

Advanced

OK Back Cancel

Microsoft Azure | Data Factory > createfirstdatafactory

Factory Resources

- Pipelines
- Change Data Capture (preview)
- Datasets
- Data flows
- Power Query

source_customer_csv... pipeline1

Activities

- Move and transform
- Copy data
- Data flow
- Synapse
- Azure Data Explorer
- Azure Function
- Batch Service
- Databricks
- Data Lake Analytics
- General
- HDInsight
- Iteration & conditionals

Set properties

Name: AzureSqlTable1

Linked service: AzureSqlDatabase1

Table name: dbo.Customers

Import schema: None

OK Back Cancel

Microsoft Azure | Data Factory > createfirstdatafactory

Factory Resources

- Pipelines
- Change Data Capture (preview)
- Datasets
- Data flows
- Power Query

source_customer_csv... pipeline1

Activities

- Move and transform
- Copy data
- Data flow
- Synapse
- Azure Data Explorer
- Azure Function
- Batch Service
- Databricks
- Data Lake Analytics
- General
- HDInsight
- Iteration & conditionals

Validate Copy runtime Debug Add trigger

Copy data

Sink dataset: AzureSqlTable1

Write behavior: Insert

Bulk insert table lock: No

Table option: Auto create table

Pre-copy script:

Write batch timeout: 00:30:00

Microsoft Azure | Data Factory > createfirstdatafactory

Factory Resources

- Pipelines
- Change Data Capture (preview)
- Datasets
- Data flows
- Power Query

source_customer_csv... pipeline1

Activities

- Move and transform
- Synapse
- Azure Data Explorer
- Azure Function
- Batch Service
- Databricks
- Data Lake Analytics
- General
- HDInsight
- Iteration & conditionals
- Machine Learning
- Power Query

Validate Debug Add trigger

Copy data

blob to sql

Parameters Variables Settings Output

Pipeline run ID: 6cc08641-df5f-44c5-b91b-640ca06965f6

Pipeline status: Succeeded

Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime
blob to sql	Succeeded	Copy data	9/3/2025, 6:30:24 PM	16s	AutoResolveIntegration

The screenshot shows a SQL query editor interface. At the top, there are navigation links for 'Feedback' and 'Getting started'. Below that, the title 'Query 1' is followed by a close button. A toolbar includes 'Run', 'Cancel query', 'Save query', 'Export data as', and 'Show only Editor'. The code area contains the following SQL script:

```
3 |     [mpg] varchar(100),
4 |     [cyl] varchar(100)
5 | );
6 |
7 drop table [dbo].[mtcars];
8 select * from [dbo].[mtcars];
```

Below the code, there are tabs for 'Results' and 'Messages', with 'Results' being selected. The results table has columns 'model', 'mpg', and 'cyl'. The data shows two rows: 'Mazda RX4' with mpg 21 and cyl 6, and 'Mazda RX4 Wag' with mpg 21 and cyl 6. A yellow bar at the bottom indicates 'Query succeeded | 0s'.

1) Azure DevOps

Azure DevOps is Microsoft's cloud-based platform that provides tools for collaboration, version control, build, testing, and deployment of applications. It is widely used for managing the end-to-end software development lifecycle (SDLC).

Key Components:

- Azure Repos → Git repositories for source code management.
- Azure Pipelines → Continuous Integration & Continuous Deployment (CI/CD).
- Azure Boards → Agile planning (work items, user stories, sprint planning, Kanban boards).
- Azure Artifacts → Package management (NuGet, npm, Maven, etc.).
- Azure Test Plans → Testing and quality assurance.

2) CI/CD Integration

CI/CD stands for Continuous Integration and Continuous Deployment/Delivery. It is a DevOps practice that automates building, testing, and deploying applications to ensure faster and more reliable delivery.

Breakdown:

- **CI (Continuous Integration):**
 - Developers push code frequently to a shared repo (e.g., Azure Repos, GitHub).
 - Code is automatically built and tested using pipelines.

- Ensures issues are caught early.
- **CD (Continuous Delivery/Deployment):**
 - Automates the release of validated code to different environments (Dev, QA, UAT, Prod).
 - Continuous Delivery → requires manual approval before production.
 - Continuous Deployment → fully automated deployment into production.

Azure DevOps CI/CD Example Flow:

1. Developer pushes code to Azure Repos (Git).
 2. Pipeline (CI) triggers → builds app, runs unit tests.
 3. Release pipeline (CD) → deploys to staging → approval → production.
 4. Monitoring & feedback loop via dashboards/logs.
-

In short:

- Azure DevOps = Platform with tools for DevOps.
 - CI/CD Integration = The practice of automating build/test/deployment pipelines, often implemented in Azure Pipelines.
-

Key Features and Services – Global Data Centers

- **Global Presence** – Azure has **data centers across the world** (regions).
- **Regions** – A geographical area containing one or more data centers (e.g., East US, Central India).
- **Availability Zones (AZs)** – Physically separate data centers within a region to provide **high availability** and **fault tolerance**.
- **Paired Regions** – Each region is paired with another (for disaster recovery).
- **Global Network** – Low-latency, high-speed connectivity between regions.
- **Compliance & Security** – Meets local laws (GDPR, HIPAA, etc.) by placing data in the right region.

In short: Azure's global data centers ensure **availability, redundancy, disaster recovery, and compliance** worldwide.

2) Key Features and Services – Azure Subscription and Resource Groups

- **Azure Subscription:**
 - Acts as a **container for resources** (VMs, databases, storage, etc.).
 - Defines **billing boundary** (you get billed per subscription).
 - Can have multiple subscriptions for **separation of environments** (Dev, Test, Prod).
 - Controls **access management** via Azure RBAC.
- **Resource Group (RG):**

- A **logical container** inside a subscription.
 - Used to group related resources (VM, Storage, Database for one app).
 - Supports **lifecycle management** (deploy, update, delete as a unit).
 - Helps in **organization, access control, and cost tracking**.
-

- 1) You have a pipeline that ingests files from an on-premises SQL Server every hour. Sometimes the files arrive late. How would you design the pipeline to handle late-arriving files without failing? – tumbling window triggers

Trigger (hourly with 15-min offset)

↓
Get Metadata (check file)

↓
Until Activity (wait & re-check)

↓
Copy Activity (SQL → Data Lake)

↓
Validation / Transform

You are using **Copy Activity** to move JSON files from Blob to SQL Database, but some rows fail because of schema drift (extra columns added in source). How will you handle these error rows?

Key Options in Copy Activity → **Fault tolerance Tab**

Option	What It Does
Skip incompatible rows	Ignores rows with errors; loads only valid rows.
Redirect rows to	Sends bad rows to another sink (Blob/ADLS/SQL) with an error report.
Maximum error rows	Limits how many bad rows to skip/redirect before failing the activity.

-
1. How to provide filename to the target file based on date function

How to provide filename to the target file based on date functions in azure data factory

SourceTableName: MonthRevenueTillDate

TargetFileName: MonthRevenueTillDate_Starting(PreviousMonthEndDate)_Ending(RunDate).csv

Ex:

MonthRevenueTillDate_Starting31Jul2022_Ending14Aug2022.csv : **Run on 14Aug2022**

MonthRevenueTillDate_Starting31Jul2022_Ending15Aug2022.csv : **Run on 15Aug2022**

MonthRevenueTillDate_Starting31Aug2022_Ending1Sep2022.csv : **Run on 1Sep2022**

MonthRevenueTillDate_Starting30Sep2022_Ending15Oct2022.csv : **Run on 15Oct2022**

Microsoft Azure | Data Factory > annuadf

Add dynamic content

```
Starting', adddays(startofMonth(utcnow()), -1), '_Ending', utcnow(), '.csv')
```

Clear contents

Add dynamic content above using any combination of expressions, functions and system variables. Click any of the available System variables or Functions below to add them directly.

utc

collapse all

Date Functions

- convertFromUtc
- Convert a timestamp from Universal Time Coordinated (UTC) to the target time zone. For example, convertT...
- convertTimeZone
- Convert a timestamp from the source time zone to the target time zone. For example, convertT...
- convertToUtc
- Convert a timestamp from the source time zone to Universal Time Coordinated (UTC). For ex...

```
ys(startofMonth(utcnow()),-1,'ddMMyyyy'), '_Ending', utcnow(), '.csv')
```

@concat('monthStart', adddays('startofMonth', (utcnow()),'Ending', utcnow()))

2. How to copy all the file present in folder to another folder using ADF

How to copy all the files present in a folder to another folder using Azure Data factory pipelines

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
<input type="checkbox"/> England.Batsman.csv	8/14/2022, 6:23:53 PM	Hot (Inferred)		Block blob	115 B	Available
<input type="checkbox"/> England.Bowler.csv	8/14/2022, 6:23:53 PM	Hot (Inferred)		Block blob	115 B	Available
<input type="checkbox"/> India.Batsman.csv	8/14/2022, 6:24:33 PM	Hot (Inferred)		Block blob	105 B	Available
<input type="checkbox"/> India.Bowler.csv	8/14/2022, 6:24:33 PM	Hot (Inferred)		Block blob	105 B	Available

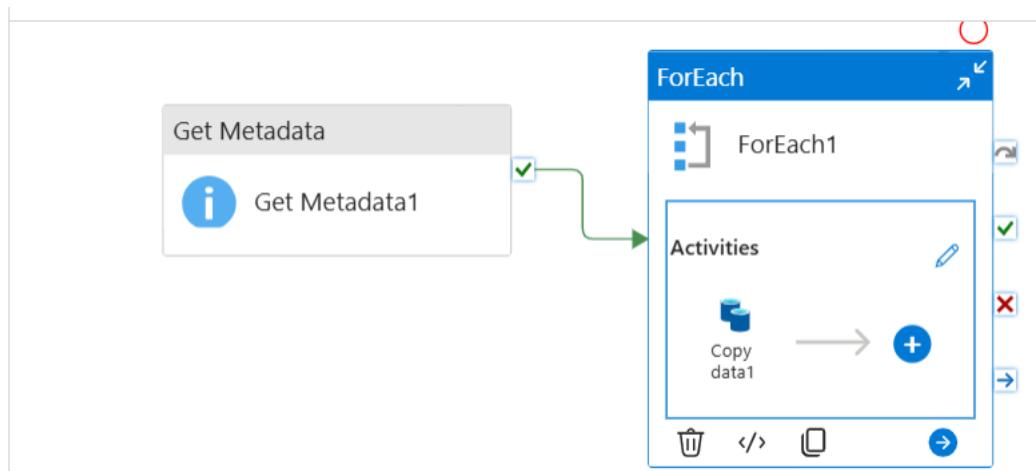
Method – 1

- Configure Source Dataset Create a dataset for your source (e.g., Azure Blob Storage, ADLS). Use the Wildcard File Path option in the source settings to specify multiple files or folders. For example: Folder Path: demo/source/ Wildcard File Name: *.csv This will select all .csv files in the specified folder.

The screenshot shows the 'Copy data' activity configuration screen. At the top, there are several status indicators: '✓ Validate', '✓ Validate copy runtime', 'Debug', and 'Add trigger'. Below this is the main configuration panel with tabs for General, Source, Sink, Mapping, Settings, and User properties. The 'Source' tab is selected. Under 'Source dataset', a dropdown menu is open, showing 'ds_source' as the selected item. To the right of the dropdown are buttons for 'Open', 'New', 'Preview data', and 'Learn more'. The 'File path type' section contains four radio buttons: 'File path in dataset' (unchecked), 'Prefix' (unchecked), 'Wildcard file path' (checked), and 'List of files' (unchecked). The 'Wildcard paths' field contains the value 'input / source / *'. At the bottom of the panel are 'Start time (UTC)' and 'End time (UTC)' fields.

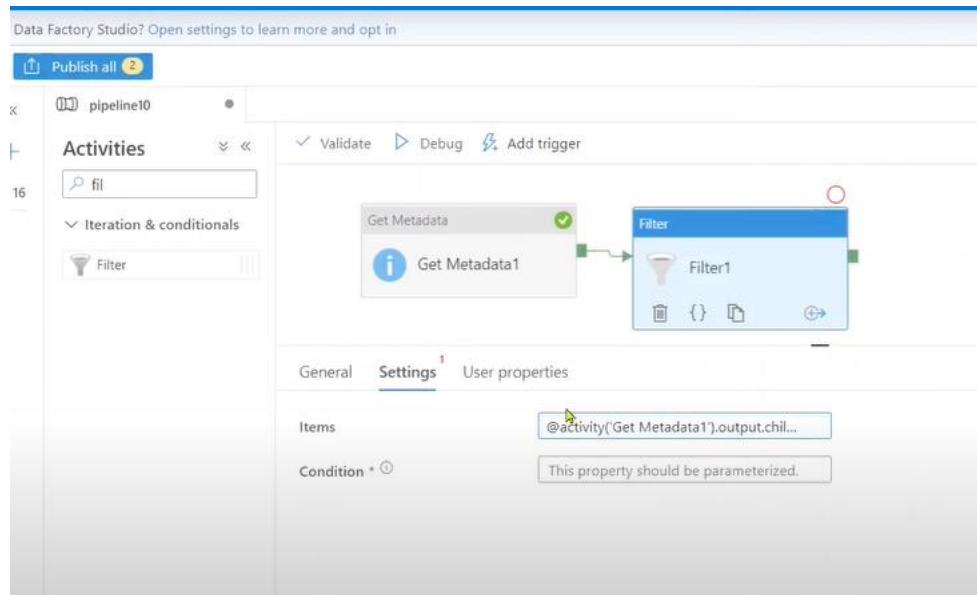
Method – 2 (dynamic copying)

- Use ForEach Activity for Dynamic Copying If you need to copy files from multiple folders dynamically: Use a get metadata Activity to fetch folder/file paths. Pass the output of the get metadata (child item) to a ForEach Activity. Inside the ForEach loop, configure the Copy Activity with dynamic parameters for source and sink paths: Source Path: @item().SourcePath Sink Path: @item().DestinationPath

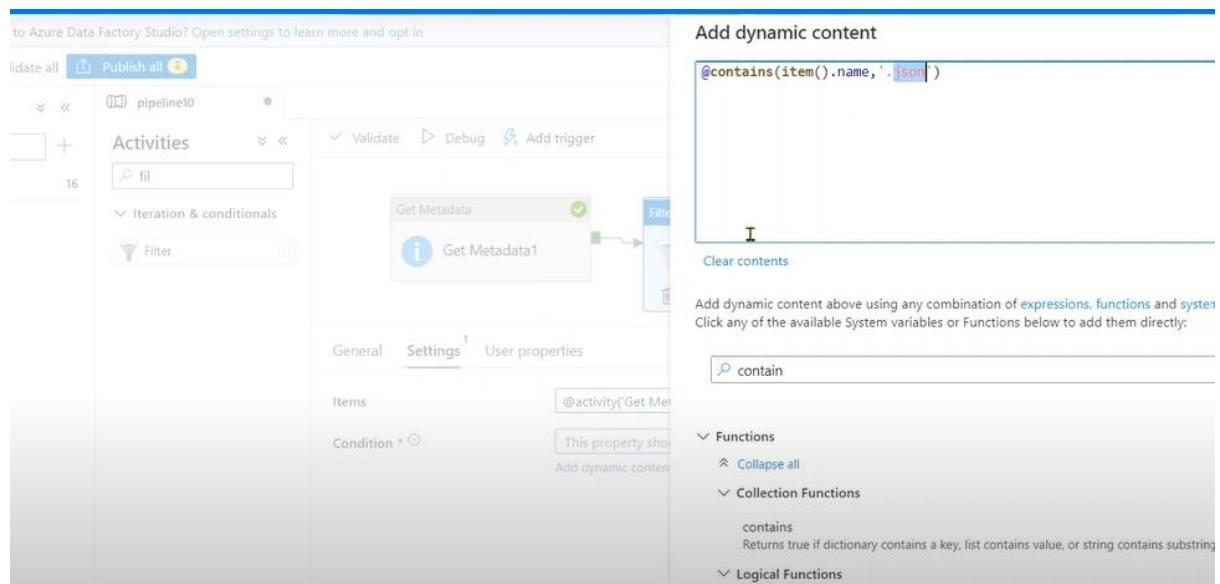


3. How to copy files of specific format from one ADLS folder to another using ADF pipeline

Step-1: get all file information using **Get Metadata activity** and the output to the **filter activity**



We can give dynamic file type also



4. How to load all the ADLS foldername and count of files inside each folder into SQL table.

Not given any file path

Microsoft Azure | Data Factory | annuadf

Would you like to try preview updates to Azure Data Factory Studio? Open settings to learn more and opt in.

Activities

Get Metadata

Get Metadata1

General Settings User properties

Dataset * Select... + New

Set properties

Name: DelimitedText9

Linked service *: D₂ADLS

File path: File system / Directory / File name

First row as header:

Import schema: From connection/store From sample file None

> Advanced

Validate Debug Add trigger

Get Metadata

Get Metadata1

General Settings User properties

Dataset *: DelimitedText9

Field list *

+ New | Delete

Argument

Child items

Filter by last modified (1)

Skip line count

Output

[Copy to clipboard](#)

```
{ "childItems": [ { "name": "adlsinput", "type": "Folder" }, { "name": "adlsoutput", "type": "Folder" }, { "name": "demo", "type": "Folder" }, { "name": "demofol", "type": "Folder" }, { "name": "dev", "type": "Folder" }, { "name": "test", "type": "Folder" } ], "effectiveIntegrationRuntime": "AutoResolveIntegrationRuntime (East US)", "executionDuration": 0, "durationInQueue": { }
```

The output of get metadata 1 is given to For each activity

If you like to try preview updates to Azure Data Factory Studio? Open settings to learn more and opt in

Data Factory Validate all Publish all

pipeline10 DelimitedText10

Activities <> Validate Debug Add trigger

General

Get Metadata

Output

Copy to clipboard

```
{
  "childItems": [
    {
      "name": "adlsinput",
      "type": "Folder"
    },
    {
      "name": "adlsoutput",
      "type": "Folder"
    },
    {
      "name": "demo",
      "type": "Folder"
    }
  ]
}
```

Get Metadata1

ForEach

ForEach1

Activities 1 activities

n start Duration Status

2022-08-19T18:45:51.5149 00:00:27 Succeeded

Inside that another get metadata is called

Validate all Publish all

pipeline10

Activities <> Validate Debug Add trigger

Get Metadata2

General Settings User properties

Select... New

General Settings User properties

Name Value

FolderName @item().name

New Delete

Argument Child items

Validate Debug Add trigger

pipeline10 > ForEach1

General **Settings** User properties

Items: @activity('Get Metadata2').output.chil...

Condition: This property should be parameterized.

Add dynamic content

```
@not(empty(item().name))
abc File
abc Folder
```

Validate all Publish all (3)

DelimitedText10

Output

```
{
  "ItemCount": 10,
  "FilteredItemsCount": 10,
  "Value": [
    {
      "name": "20211224-bollettino-widget-0900.xml",
      "type": "File"
    },
    {
      "name": "21july.xml",
      "type": "File"
    }
  ]
}
```

Get Metadata ForEach

start	Duration	Status
22-08-19T18:56:34.1829	00:00:01	Succeeded
22-08-19T18:56:26.2924	00:00:01	Succeeded
		Succeeded

5. How to retrieve folder having valid date in the FolderName

► **Use Case:**

How to retrieve only those folders which are having valid date in the foldernames in ADLS using ADF pipeline

1. Use Get Metadata on the parent container.

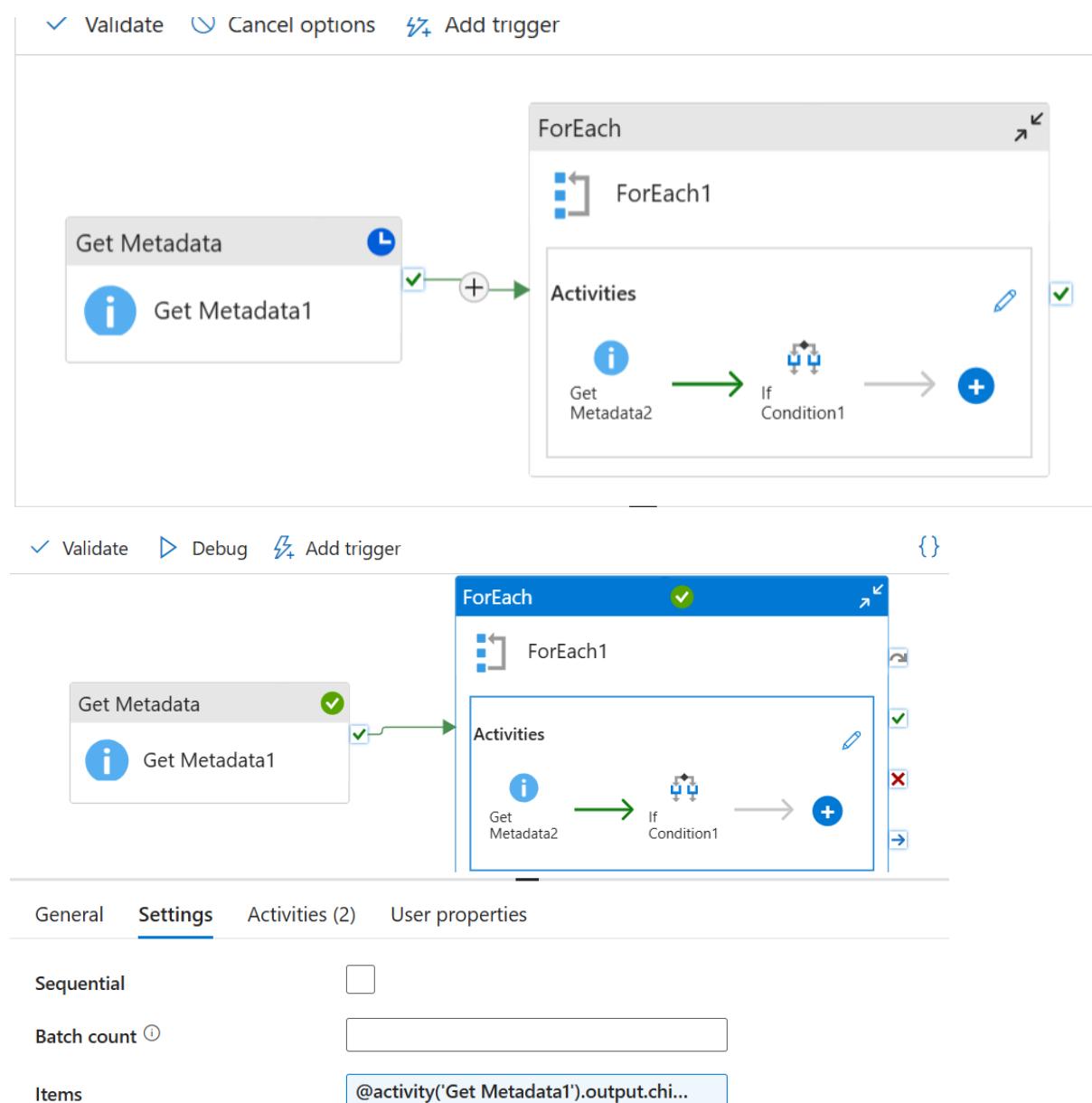
- Field list → Child items.
2. In a ForEach loop over @activity('Get Metadata1').output.childItems.
 3. Inside ForEach add If Condition with an expression that checks if the folder name is a valid date.

If your folders are yyyyMMdd: @not(empty(tryParseDateTime(item().name, 'yyyyMMdd')))

If your folders are yyyy-MM-dd: @not(empty(tryParseDateTime(item().name, 'yyyy-MM-dd')))

6. How to get the list of empty and non-empty ADLS folder.

Get metadata → for each → get metadata (directory) → IF (True) → append variable(file name) IF (false) → append variable(file name)



✓ Validate ▶ Debug ⚡ Add trigger

dynamic_file > ForEach1

The screenshot shows the configuration for the 'Get Metadata' activity. It includes a 'Get Metadata2' sub-activity and an 'Append variable1' action under an 'If Condition1' block. The 'Settings' tab is selected, showing a dataset named 'filecontainer' and its properties:

Name	Value	Type
directory_name	@item().name	String

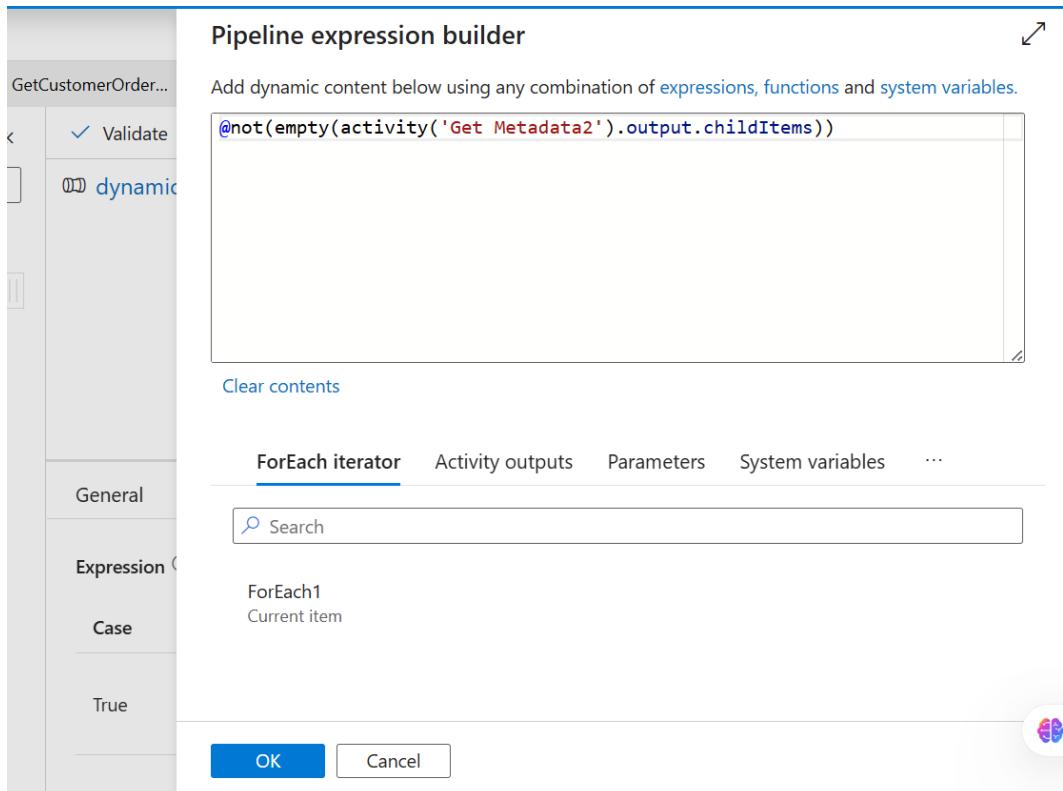
✓ Validate ▶ Debug ⚡ Add trigger

dynamic_file > ForEach1

The screenshot shows the configuration for the 'Get Metadata' activity. It includes a 'Get Metadata2' sub-activity and an 'Append variable1' action under an 'If Condition1' block. The 'Activities (2)' tab is selected, showing the following expression and activities:

Expression: @not(empty(activity('Get Metadata2')))

Case	Activity
True	Append variable1 1 Activity
	Append variable2



Preview experience Off

Activity name	Activity st...	Activit...	Run start	Duration
Append variable2	Succeeded	Append variabl	9/4/2025, 2:52:27 PM	Less than 1
Append variable2	Succeeded	Append variabl	9/4/2025, 2:52:26 PM	Less than 1
Append variable1	Succeeded	Append variabl	9/4/2025, 2:52:26 PM	Less than 1
If Condition1	Succeeded	If Condition	9/4/2025, 2:52:26 PM	2s
Append variable1	Succeeded	Append variabl	9/4/2025, 2:52:26 PM	Less than 1
If Condition1	Succeeded	If Condition	9/4/2025, 2:52:26 PM	2s
If Condition1	Succeeded	If Condition	9/4/2025, 2:52:26 PM	2s
If Condition1	Succeeded	If Condition	9/4/2025, 2:52:26 PM	2s
Get Metadata2	Succeeded	Get Metadata	9/4/2025, 2:52:21 PM	4s
Get Metadata2	Succeeded	Get Metadata	9/4/2025, 2:52:21 PM	4s
Get Metadata2	Succeeded	Get Metadata	9/4/2025, 2:52:21 PM	4s

This screenshot shows the pipeline run history. It lists 14 items, each with an activity name, status (all succeeded), run start time, and duration. Activities include Append variable, If Condition, and Get Metadata. The interface includes a toolbar with Validate, Debug, Add trigger, and a preview experience toggle.

Append variable2

Succeeded

Append variable2

Succeeded

Input

 Copy to clipboard

```
{  
  "variableName": "hasnofile",  
  "value": "store"  
}
```

Append variable2

Succeeded

Append variable1

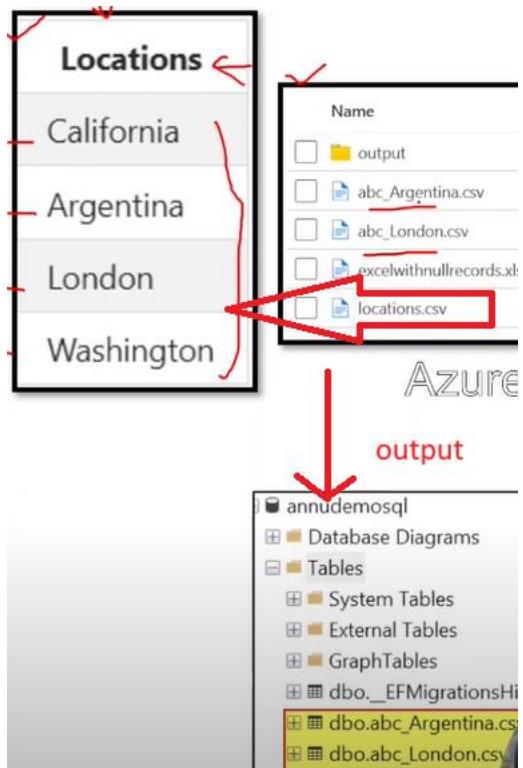
Succeeded

Input

 Copy to clipboard

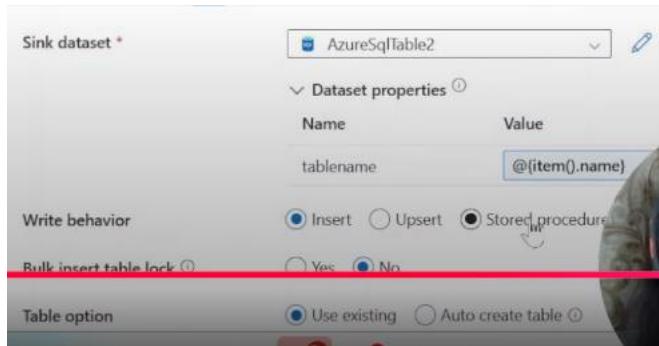
```
{  
  "variableName": "hasfile",  
  "value": "sales"  
}
```

How to dynamically copy only matching files from ADLS to SQL?



Lookup activity (to get the location from file location.csv) → for each → executive pipeline

Get meta-activity (to get the file name like abc_argentina, abc_london, location.csv,...) → for each → if (@contain(item().name, location) → copy activity



auto create table (to create a new table)

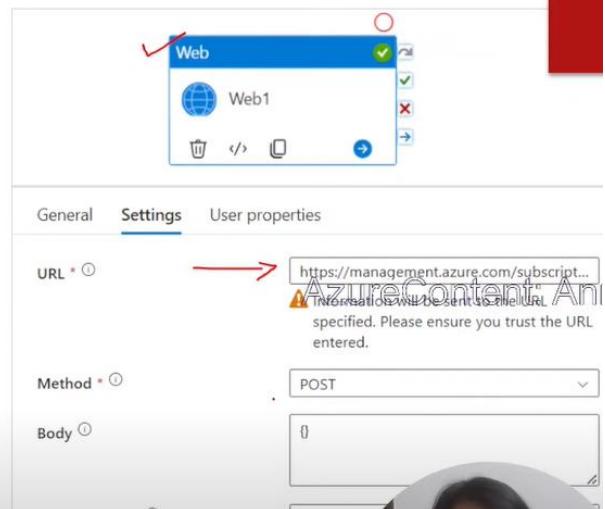
How to execute one ADF pipeline from other ADF pipeline via rest API??

Episode 29:

Use Case:

Execute ADF pipeline via REST API call

POST
`https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.DataFactory/factories/{factoryName}/pipelines/{pipelineName}/createRun?api-version=2018-06-01`



Episode 25:

Use Case:

Capture **Error Message** if the Dataflow failed , else get the **RowsRead** and **RowsWritten** numbers.

Azure Content Annu

Capture error message if the dataflow failed, else (if succeed) get the RowsRead and RowsWritten numbers (in SQL table).

Copy files uploaded three days back from ADLS Gen2. → @addDays(utcNow(),-7) end-@utcnow()

Validate all ↑ Publish all 2 Preview experience

Validate copy runtime Debug Add trigger

General Source Sink¹ Mapping Settings User properties

Source dataset * ds_source Open New Preview data Learn more

File path type File path in dataset Wildcard file path List of files

Wildcard paths demo / source / .csv

Start time (UTC) End time (UTC)

Filter by last modified @addDays(utcNow(),-7) last 7 days data

Recurisvely

Enable partitions discovery

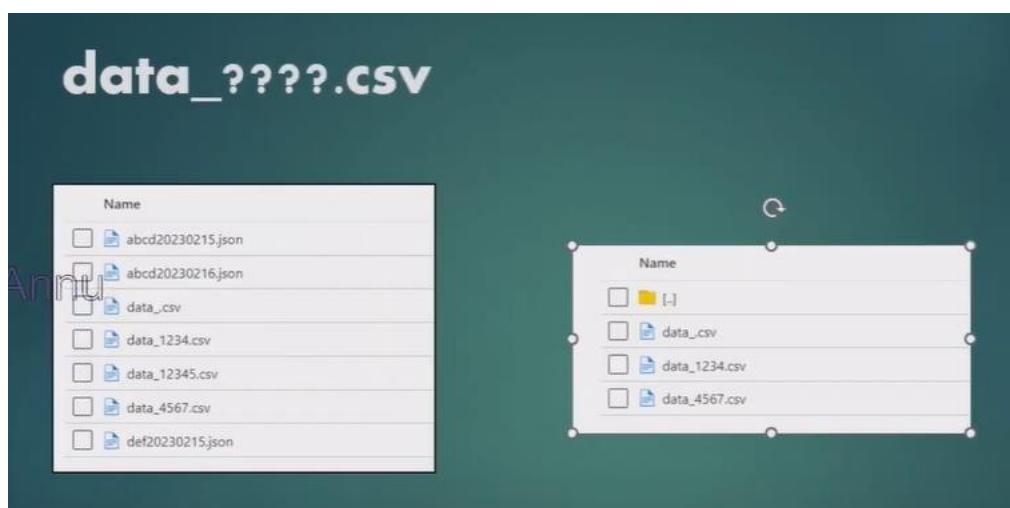
Max concurrent connections

Wildcard

The screenshot shows the 'Copy data' activity configuration in the Azure Data Factory pipeline editor. The 'Source' tab is selected. The 'Source dataset' is set to 'Binary4'. The 'File path type' is set to 'Wildcard file path', with the path specified as 'abccontainer / [] / data_????.csv'. The 'Recursively' checkbox is checked. Below the path, there are fields for 'Start time (UTC)' and 'End time (UTC)'. A note at the bottom says 'Add dynamic content [Alt+Shift+D]'. The pipeline navigation bar at the top shows 'Binary5', 'Binary4', 'pipeline25', and 'Validate', 'Validate copy runtime', 'Debug', 'Add trigger'.

data_????.csv → matches data_.csv, data_12.csv, data_123.csv, data_1234.csv,
data_tttt.csv,

But not matches data_abced.csv



We can also use **data_.*** instead of file name

data_????*.csv

is same as
data_* .csv

Name
<input type="checkbox"/> abcd20230215.json
<input checked="" type="checkbox"/> abcd20230216.json
<input type="checkbox"/> data_.csv
<input type="checkbox"/> data_1234.csv
<input type="checkbox"/> data_12345.csv
<input type="checkbox"/> data_4567.csv
<input type="checkbox"/> def20230215.json

Name
<input type="checkbox"/> data_.csv
<input type="checkbox"/> data_1234.csv
<input type="checkbox"/> data_12345.csv
<input type="checkbox"/> data_4567.csv

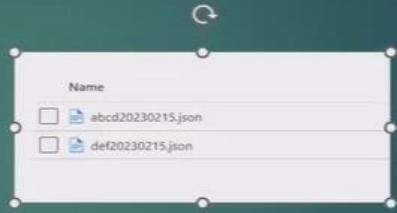
???20230215.json

Name
<input type="checkbox"/> abcd20230215.json
<input checked="" type="checkbox"/> abcd20230216.json
<input type="checkbox"/> data_.csv
<input type="checkbox"/> data_1234.csv
<input type="checkbox"/> data_12345.csv
<input type="checkbox"/> data_4567.csv
<input type="checkbox"/> def20230215.json

Name
<input type="checkbox"/> def20230215.json

***20230215.json**

Name
<input type="checkbox"/> abcd20230215.json
<input type="checkbox"/> abcd20230216.json
<input type="checkbox"/> data_.csv
<input type="checkbox"/> data_1234.csv
<input type="checkbox"/> data_12345.csv
<input type="checkbox"/> data_4567.csv
<input type="checkbox"/> def20230215.json



Wildcard

FolderPath	fileName	recursive	Source folder structure and filter result (files in bold are retrieved)
Folder*	(Empty, use default)	false	FolderA File1.csv File2.json Subfolder1 File3.csv File4.json File5.csv AnotherFolderB File6.csv
Folder*	(Empty, use default)	true	FolderA File1.csv File2.json Subfolder1 File3.csv File4.json File5.csv AnotherFolderB File6.csv
Folder*	*.csv	false	FolderA File1.csv File2.json Subfolder1 File3.csv File4.json File5.csv AnotherFolderB File6.csv
Folder*	*.csv	true	FolderA File1.csv File2.json Subfolder1 File3.csv File4.json File5.csv AnotherFolderB File6.csv

Azure Content An

pipeline25

- Validate
- Validate copy runtime
- Debug
- Add trigger

Copy data

Copy data1

General Source Sink Mapping Settings User properties

Source dataset * Binary4

File path type Wildcard file path

Wildcard paths abcccontainer / Folder* / Wildcard file name

Start time (UTC) End time (UTC)

Filter by last modified

Recursively

Delete files after completion

Max concurrent connections

How to get email notification for pipeline failure by creating alert rules within ADF?

Use Case:

Create Alert rules within Azure data factory to monitor pipeline failures and get email notifications

Manual approaches to monitoring data integration projects are inefficient and time consuming.

Creating alerts will ensure 24/7 monitoring of your data pipelines and make sure that you are notified of issues before they potentially corrupt your data or affect downstream processes

ADLS integration with Databricks

Method-3: using ADLS Access key directly

Once created ADLS storage account → get access key : “click tab on left menu Access Key”

Microsoft Azure

adlsrajade | Access keys

Storage account

Search (Ctrl+ /)

Queues

Tables

Security + networking

Networking

Access keys

Shared access signature

Encryption

Security

Data management

Geo-replication

Data protection

Blob inventory

Static website

Lifecycle management

Hide keys

Set rotation reminder

Refresh

Access keys authenticate your applications' requests to this storage account. Keep your keys in a secure location like Azure Key Vault, and replace them often with new keys. The two keys allow you to replace one while still using the other.

Remember to update the keys with any Azure resources and apps that use this storage account. Learn more

Storage account name: adlsrajade

key1

Last rotated: 1/15/2022 (0 days ago)

Rotate key

Key: HJeoH1r2fXWpZvu4lUPSQqIkJDS3pxLEmFZnJ4MRZOW5y4UrcekTqKAEOKT/YYDz...

Connection string: DefaultEndpointsProtocol=https;AccountName=adlsrajade;AccountKey=HJeoH1r2...

key2

Last rotated: 1/15/2022 (0 days ago)

Rotate key

ADLS Integration with Databricks Python

test

Cmd 1

Methods of ADLS Integration with Databricks

- Using a service principal
- Using Azure Active Directory credentials known as a credential passthrough
- Using ADLS access key directly
- Creating Mount Point using ADLS Access Key

Cmd 2

Method 1: Using ADLS access key directly

we have give storage account name

Cmd 3

```
1 spark.conf.set(  
2 "fs.azure.account.key.<storage account>.dfs.core.windows.net",  
3 "<access key>"  
4 )
```

give access key

Cmd 4

```
1 dbutils.fs.ls("abfss://<>@<>.dfs.core.windows.net/")
```

Cmd 5

Apps Google Exercise - Create Az... AzureDataFactoryH... Per

ADLS Integration with Databricks Python

test | File | Edit | View: Standard | Permissions | Run All | Clear

Methods of ADLS Integration with Databricks

- Using a service principal
- Using Azure Active Directory credentials known as a credential passthrough
- Using ADLS access key directly
- Creating Mount Point using ADLS Access Key

Method 1: Using ADLS access key directly

Cancel * Uploading command

```
1 spark.conf.set(  
2 "fs.azure.account.key.adlsrajade.dfs.core.windows.net",  
3 "HJeoH1r2fXWpZvu4IUPSQqIkJJD3pxLEMFnJ4MRZ0W5y4UrcekTqKAEOKT/YYDzAirUYdXPCpxe7Gcl4IbBA=="  
4 )
```

Cmd 4

```
1 dbutils.fs.ls("abfss://<>@<>.dfs.core.windows.net/")
```

ADLS Integration with Databricks Python

test | File | Edit | View: Standard | Permissions | Run All | Clear

```
3 "HJeoH1r2fXWpZvu4IUPSQqIkJJD3pxLEMFnJ4MRZ0W5y4UrcekTqKAEOKT/YYDzAirUYdXPCpxe7Gcl4IbBA=="  
4 )
```

Command took 0.04 seconds -- by audaciousazure@gmail.com at 1/15/2022, 1:51:27 PM on test

Cmd 4

```
1 dbutils.fs.ls("abfss://container-rajade@adlsrajade.dfs.core.windows.net/")
```

Out[2]: [FileInfo(path='abfss://container-rajade@adlsrajade.dfs.core.windows.net/world_population_data.csv')]

Command took 1.63 seconds -- by audaciousazure@gmail.com at 1/15/2022, 1:52:34 PM on test

Cmd 5

```
1 #set the data lake file location:  
2 file_location = "abfss://container-rajade@adlsrajade.dfs.core.windows.net/"  
3  
4 #read in the data to dataframe df  
5 df = spark.read.format("csv").option("inferSchema", "true").option("header",  
6 "true").option("delimiter", ",").load(file_location)  
7  
8 #display the dataframe  
9 display(df)
```

Method – 4

Method 2: Creating Mount Point using ADLS Access Key

```
Cmd 7
1 dbutils.fs.mount(
2     source = "wasbs://container-rajade@adlsrajade.blob.core.windows.net",
3     mount_point = "/mnt/adls_test",
4     extra_configs = {"fs.azure.account.key.<storageaccount.blob.core.windows.net>":<access key>"})
Cmd 8
1 dbutils.fs.ls("/mnt/adls_test")
```

Shift+Enter to run

In source give → “wasbs://<container-name>@<endpoint>”

Microsoft Azure

adlsrajade | Endpoints

Storage account

Geo-replication

Data protection

Blob inventory

Static website

Lifecycle management

Endpoints

Locks

Monitoring

Insights

Alerts

Search (Ctrl + /)

Refresh

Provisioning state	Succeeded
Created	1/15/2022, 1:41:35 PM
Last Failover	Never
Storage account resource ID	/subscriptions/53dde41e-916f-49f8-8108-558036f826ae/resourceGroups/learn-3b11e3a6-5f24-4c67-abfe-2159d688c1f7/providers/Mic...
Primary endpoint	https://adlsrajade.blob.core.windows.net/
Secondary endpoint	https://adlsrajade-secondary.blob.core.windows.net/
File service	https://adlsrajade.file.core.windows.net/