Contents

[DATA CLASSIFICATION 2](#_Toc127029216)

[DATA FORMAT 2](#_Toc127029217)

[TERMINOLOGIES 2](#_Toc127029218)

[DATA LAKE 2](#_Toc127029219)

[AZURE DATA LAKE STORAGE ACCOUNT 3](#_Toc127029220)

[CREATING DATA LAKE GEN-2 STORAGE ACCOUNT 3](#_Toc127029221)

[CREATING CONTAINIER AND UPLOAD BLOB OBJECTS 4](#_Toc127029222)

[CONNECTING STORAGE EXPLORER WITH DATA LAKE STORAGE ACCOUNT 5](#_Toc127029223)

[SQL 5](#_Toc127029224)

[AZURE SQL DATABASE 5](#_Toc127029225)

[CONFIGURING THE DATABASE 6](#_Toc127029226)

[NETWORKING 7](#_Toc127029227)

[ADDITIONAL SETTINGS 7](#_Toc127029228)

[CONFIGURING SQL SERVER STUDIO 8](#_Toc127029229)

[ETL(EXTRACT-TRANSFORM-LOAD) 8](#_Toc127029230)

[AZURE SYNAPSE 9](#_Toc127029231)

[AZURE DATA FACTORY 9](#_Toc127029232)

[WHAT PROBLEM DOES IT SOLVE? 10](#_Toc127029233)

[USAGE SCENARIOS 10](#_Toc127029234)

[COMPONENTS OF AZURE DATA FACTORY 12](#_Toc127029235)

[PIPELINE 12](#_Toc127029236)

[ACTIVITIES 12](#_Toc127029237)

[DATASET 13](#_Toc127029238)

[LINKED SERVICE 14](#_Toc127029239)

[MAPPING DATA FLOWS 15](#_Toc127029240)

[INTEGRATION RUNTIMES 15](#_Toc127029241)

[ADDITIONAL CONCEPTS 15](#_Toc127029242)

[SETTING UP AZURE DATA FACTORY 16](#_Toc127029243)

[ADF OVERVIEW 17](#_Toc127029244)

[ADF STUDIO 17](#_Toc127029245)

[SETTING UP ADF COMPONENTS 18](#_Toc127029246)

# DATA CLASSIFICATION

|  |  |
| --- | --- |
| STRUCTURED DATA | * Tabular Data that is represented by rows and columns in a DB |
| SEMI-STRUCTURED DATA | * JSON * Key value data store- The values are based on the key. The key value data store   can be stored also in the form of tables, but over here, we don't have any sort of relationships between the tables as we have in a SQL based database system. |
| UNSTRUCTURED DATA | Data like having audio files, video files, images, etc. All of these are binary objects. |

## DATA FORMAT

|  |  |
| --- | --- |
| JSON | * JavaScript Object |
| AVRO | * This is row-based file format * Each record in the file contains a header that describes the structure of the data in the record * Stores data in compressed format |
| CSV | Comma Separated Values |
| PARQUET | Unlike row-based formats such as CSV or Avro, Apache Parquet is column-oriented – meaning the values of each table column are stored next to each other, rather than those of each record |

# TERMINOLOGIES

|  |  |
| --- | --- |
| DTU | * Database Transaction Unit. This unit is a combined measure of CPU, memory, IOPS, etc. |

# DATA LAKE

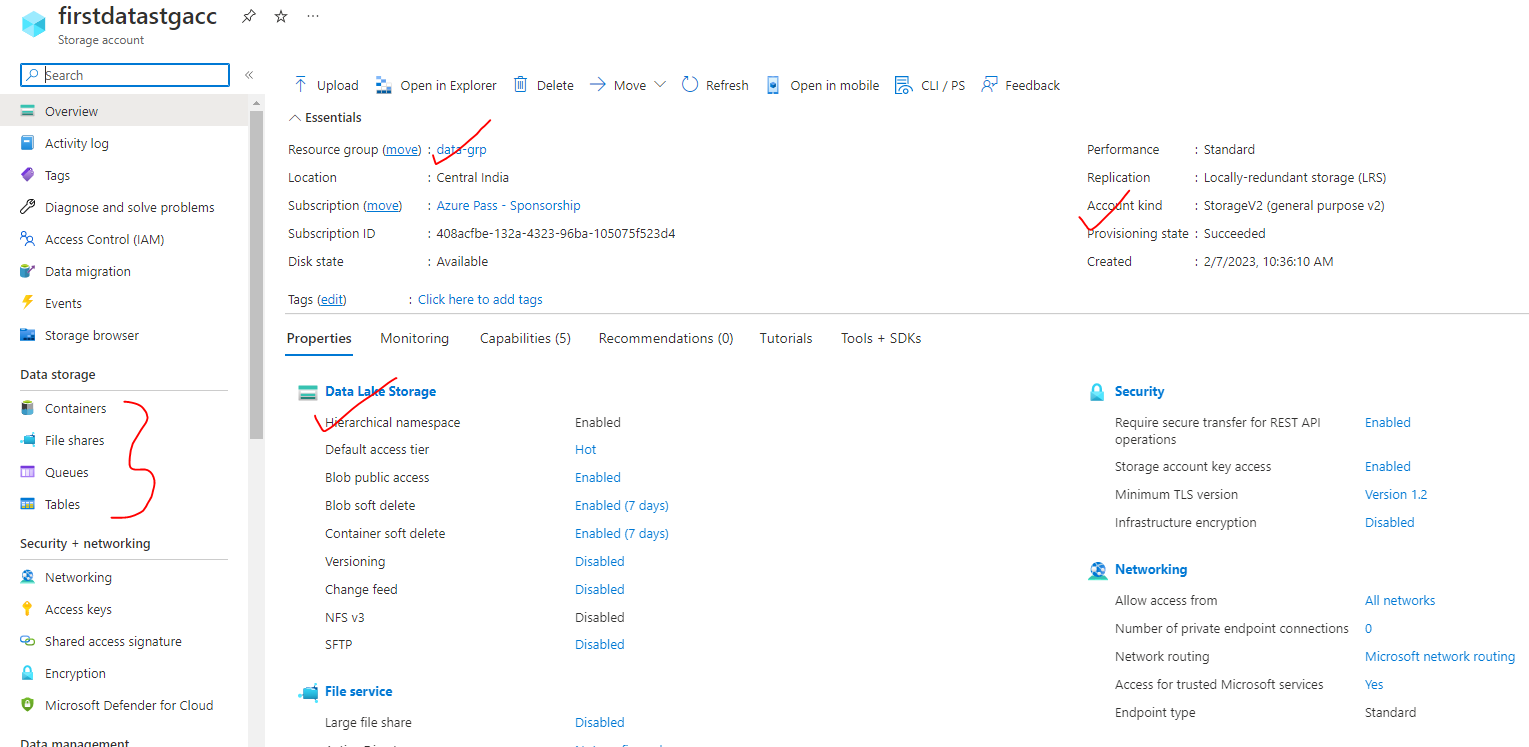
|  |  |
| --- | --- |
| **DATA LAKE IN AZURE**  Data lakes - Azure Architecture Center | Microsoft Learn | * A data lake is a system or [repository of data](https://en.wikipedia.org/wiki/Data_repository) stored in its natural/raw format(images, text, JSON format), usually object [blobs](https://en.wikipedia.org/wiki/Binary_large_object) or files. * Data lake is used when we are working with large data sets and data is arriving in large volumes and at fast rate. * The data lake has been provided as PAAS in Azure using Azure data lake storage account. |

# AZURE DATA LAKE STORAGE ACCOUNT

* Azure Data Lake Gen2 storage accounts is a service that provides the option of hosting a data lake on Azure.
* Data lake account can be used when we are working with large data sets and data is arriving in large volumes and at fast rate.
* ***Data Lake Storage account is built on top of Azure Blob Storage. Here we make use of features of a hierarchical name space on top of Azure block storage itself. This hierarchy helps to organize the objects and the files into a hierarchy of directories for efficient data access.***
* Data lake is used to store large amount of data in its native, raw format. It is optimized for storing terabytes and petabytes of data

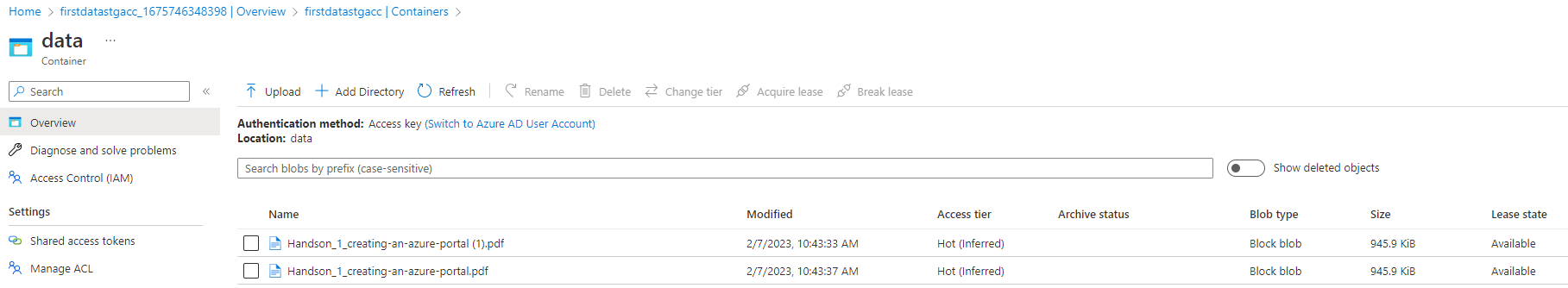
## CREATING DATA LAKE GEN-2 STORAGE ACCOUNT

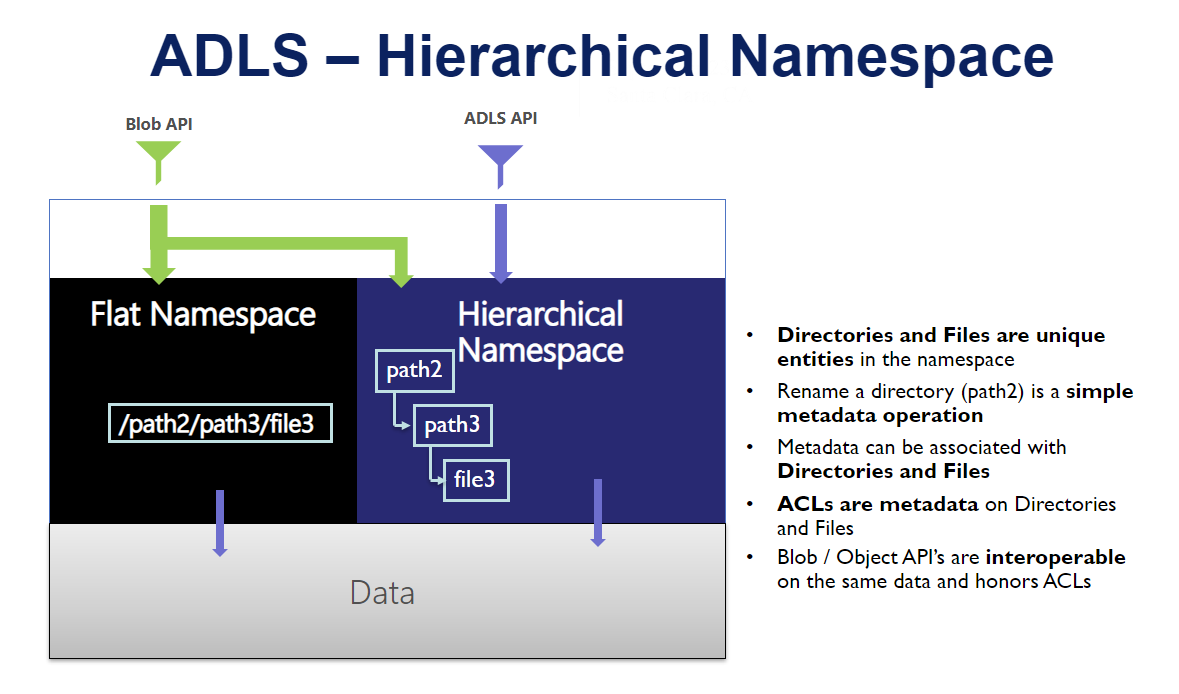
|  |  |
| --- | --- |
|  | * The data lake storage account is created same as storage account -with a difference |



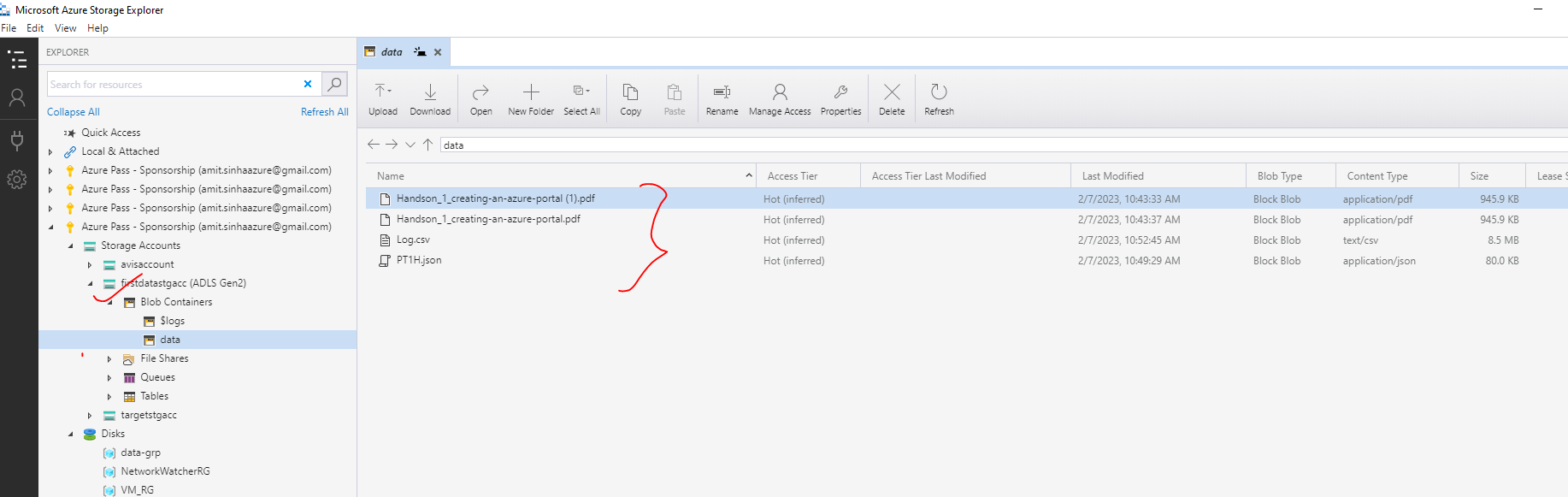
### CREATING CONTAINIER AND UPLOAD BLOB OBJECTS







## CONNECTING STORAGE EXPLORER WITH DATA LAKE STORAGE ACCOUNT



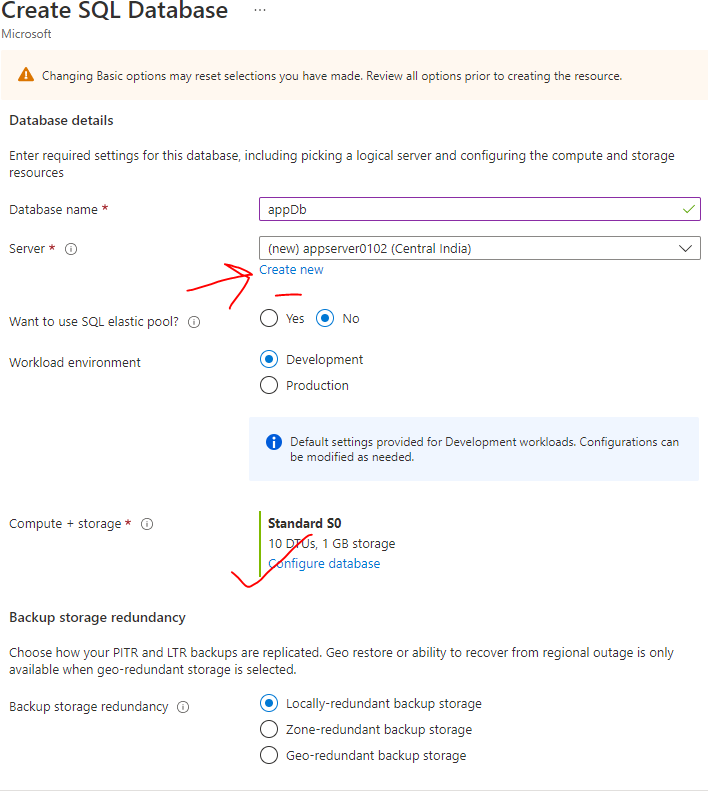
The data lake storage account can be connected to storage explorer- same as General Purpose Storage account. The different ways to connect are (refer the details in Azure Storage Account documentation)

1. **SUBSCRIPTION**
2. **ACCESS KEYS**
3. **SHARED ACCESS SIGNATURE**

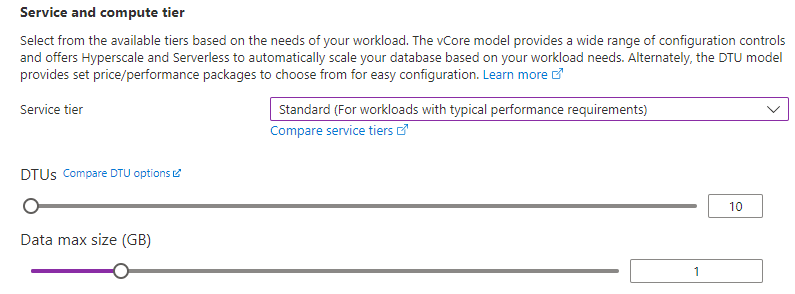
# SQL

## AZURE SQL DATABASE

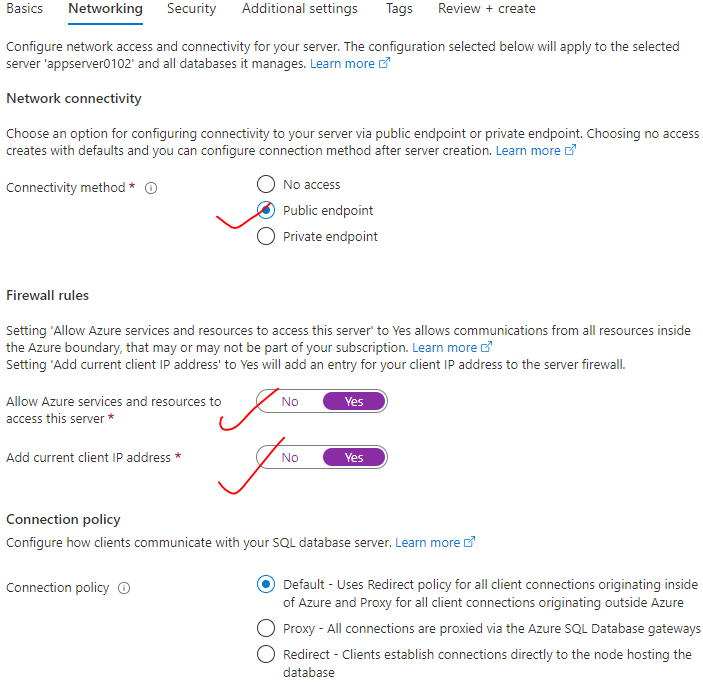
* It’s a PAAS in Azure. To create SQL database server - Select SQL database service 🡪 Create SQL Database.



### CONFIGURING THE DATABASE



### NETWORKING



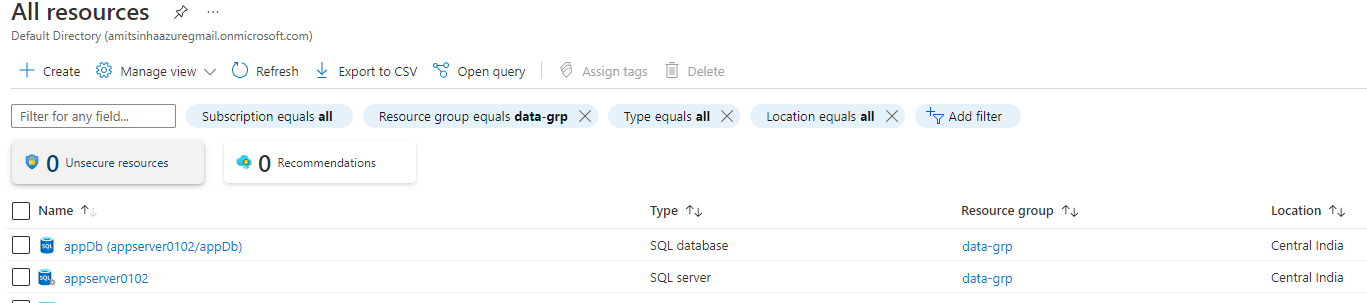
### ADDITIONAL SETTINGS

* We can make use of sample data as well. This will pre-populate the DB with tables with sample data

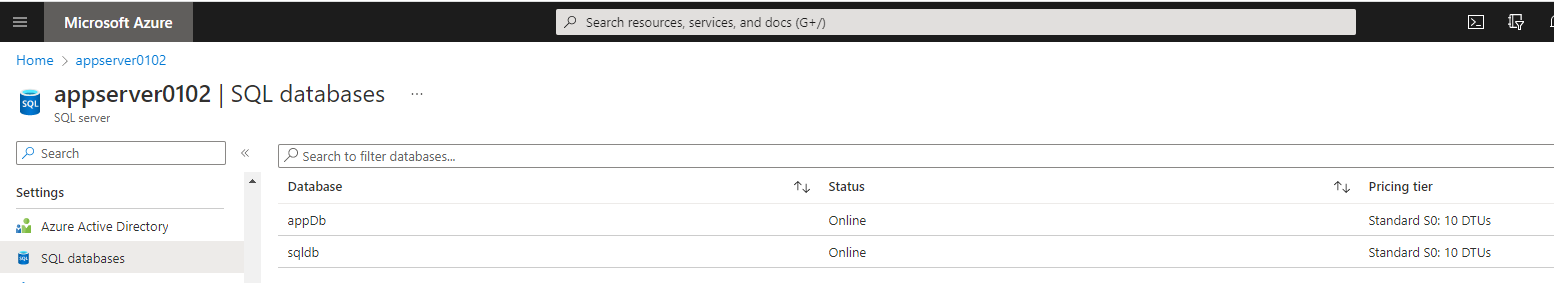
|  |  |
| --- | --- |
|  |  |

* Note - This implementation creates two resources.

1. **AZURE SQL DATABASE SERVER**
2. **AZURE SQL DATABASE.**

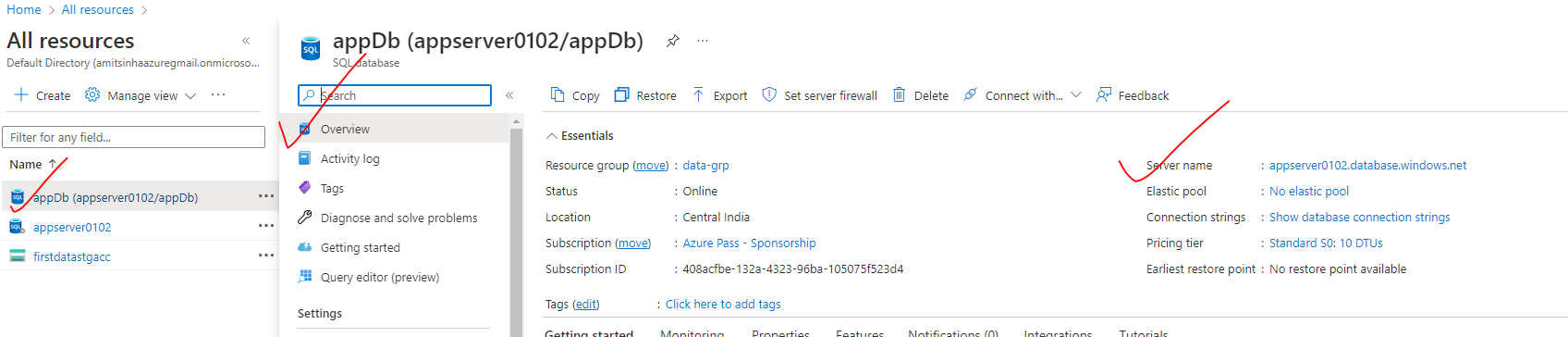


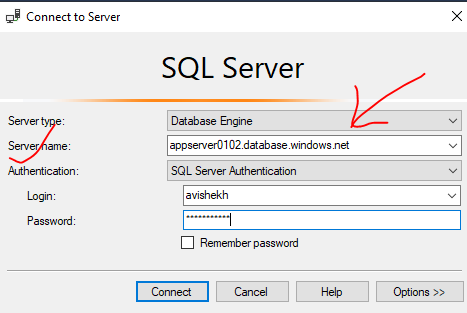
* **We can have one DB server with multiple SQL data base.**



### CONFIGURING SQL SERVER STUDIO

* To connect the SQL server studio with SQL Database server and hence access the SQL DB





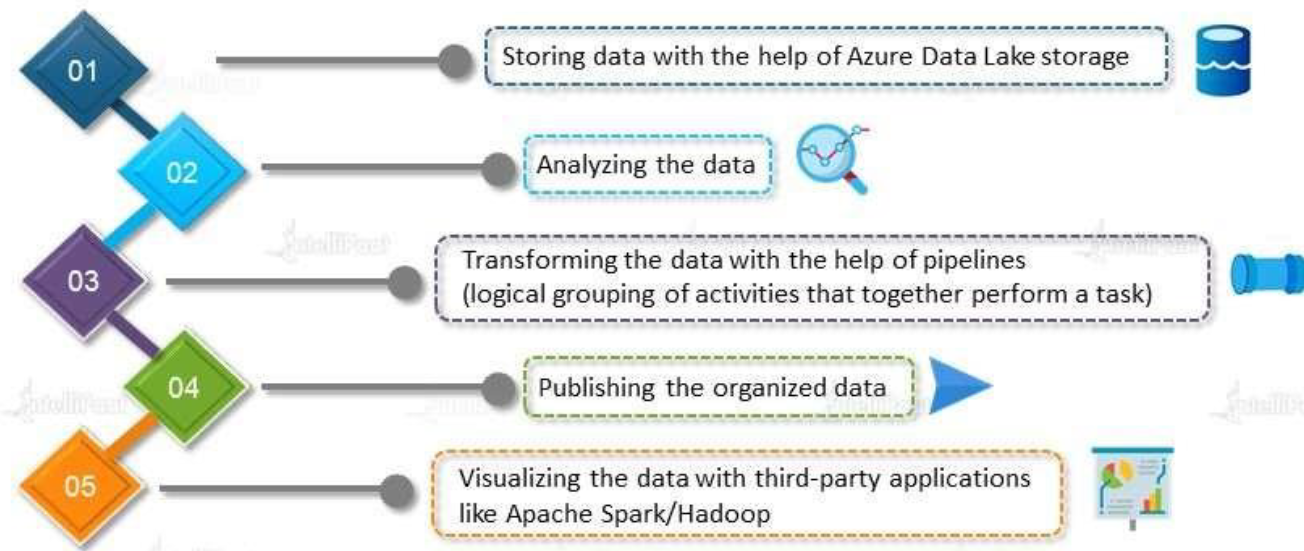
# ETL(EXTRACT-TRANSFORM-LOAD)

|  |  |
| --- | --- |
| What is ETL? (Extract Transform Load) | Informatica | ETL is a process of extracting data from a variety of data sources and load the data into a target area. Example -extracting data from an Azure SQL Database or SQL server instance hosted in an on-premise environment and load it on to an Azure SQL Data Warehouse after perform some sort of transformation of the data |

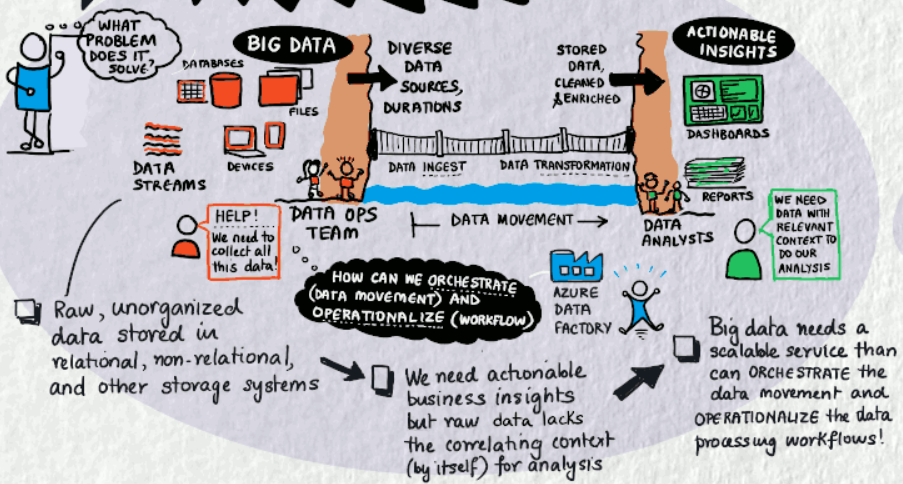
# AZURE SYNAPSE

# AZURE DATA FACTORY

|  |  |
| --- | --- |
|  | * In the world of big data, raw, unorganized data is often stored in relational, non-relational, and other storage systems. However, on its own, raw data doesn't have the proper context or meaning to provide meaningful insights to analysts, data scientists, or business decision makers. * Big data requires a service that can orchestrate and operationalize processes to refine these enormous stores of raw data into actionable business insights. Azure Data Factory is a managed cloud service that's built for these complex hybrid extract-transform-load (ETL), extract-load-transform (ELT), and data integration projects. |



## WHAT PROBLEM DOES IT SOLVE?



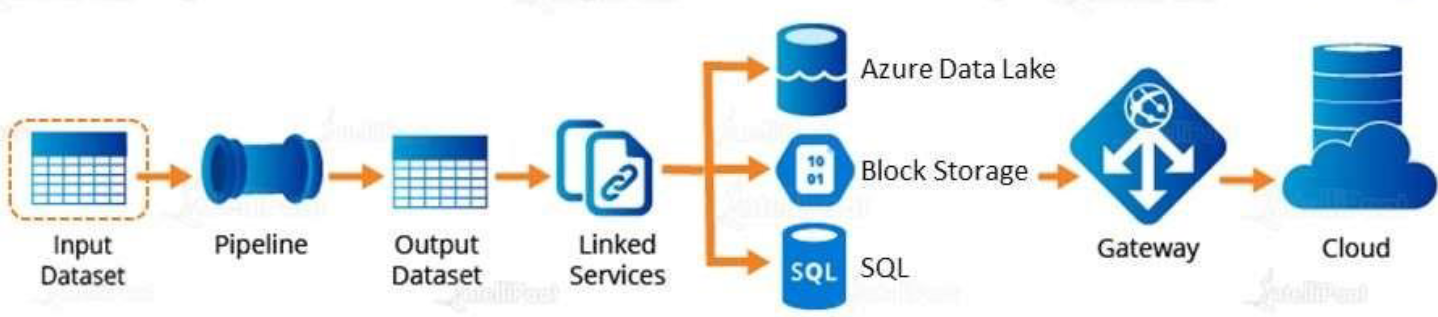
## USAGE SCENARIOS



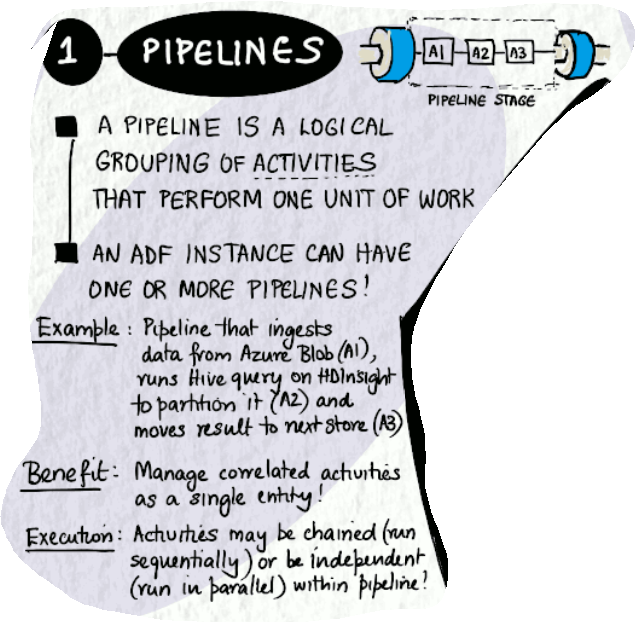
* For example, imagine a gaming company that collects petabytes of game logs that are produced by games in the cloud. The company wants to analyze these logs to gain insights into customer preferences, demographics, and usage behavior. It also wants to identify up-sell and cross-sell opportunities, develop compelling new features, drive business growth, and provide a better experience to its customers.
* To analyze these logs, the company needs to use reference data such as customer information, game information, and marketing campaign information that is in an on-premises data store. The company wants to utilize this data from the on-premises data store, combining it with additional log data that it has in a cloud data store.
* To extract insights, it hopes to process the joined data by using a Spark cluster in the cloud (Azure HDInsight), and publish the transformed data into a cloud data warehouse such as Azure Synapse Analytics to easily build a report on top of it. They want to automate this workflow and monitor and manage it on a daily schedule. They also want to execute it when files land in a blob store container.
* Azure Data Factory is the platform that solves such data scenarios. **It is the cloud-based ETL and data integration service that allows you to create data-driven workflows for orchestrating data movement and transforming data at scale**.
* Using Azure Data Factory, we can create and schedule data-driven workflows (called pipelines) that can ingest data from disparate data stores. We can build complex ETL processes that transform data visually with data flows or by using compute services such as Azure HDInsight Hadoop, Azure Databricks, and Azure SQL Database.
* Additionally, we can publish your transformed data to data stores such as Azure Synapse Analytics for business intelligence (BI) applications to consume. **Ultimately, through Azure Data Factory, raw data can be organized into meaningful data stores and data lakes for better business decisions**.

## COMPONENTS OF AZURE DATA FACTORY

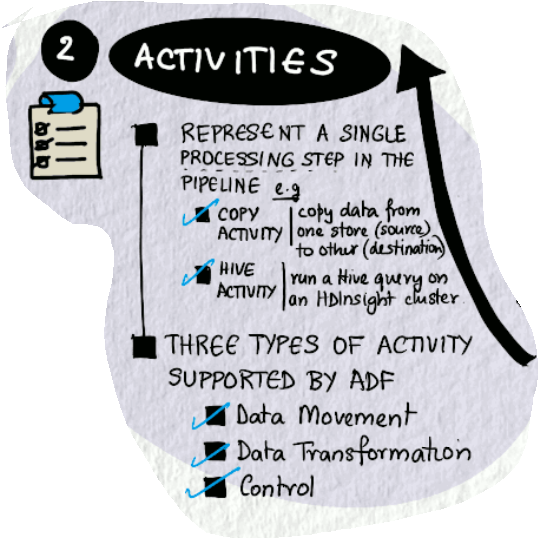
PROCESS FLOW



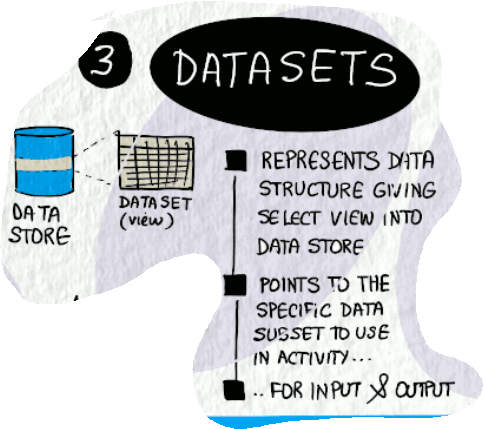
### PIPELINE



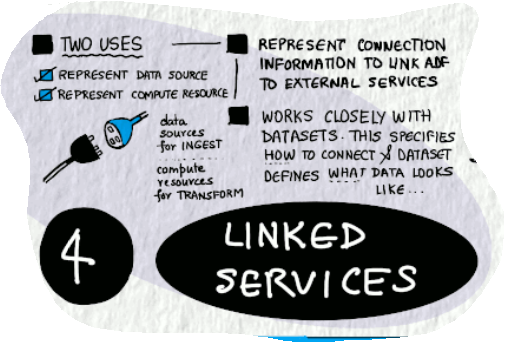
### ACTIVITIES



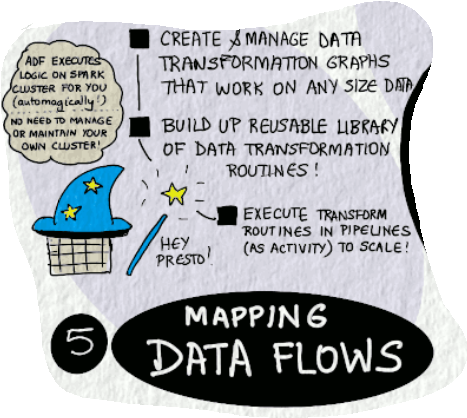
### DATASET



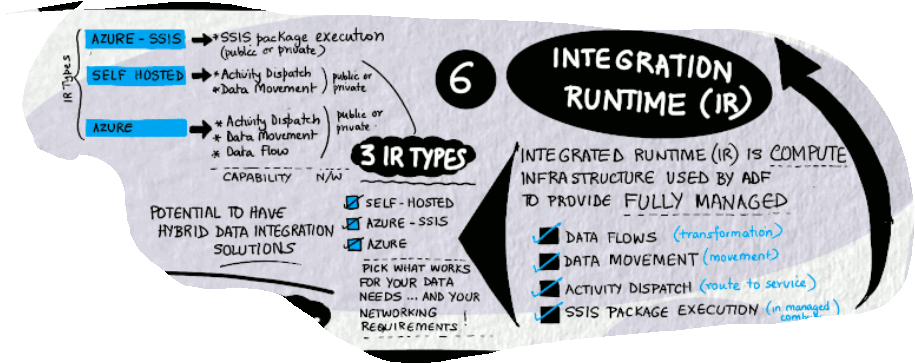
### LINKED SERVICE



### MAPPING DATA FLOWS



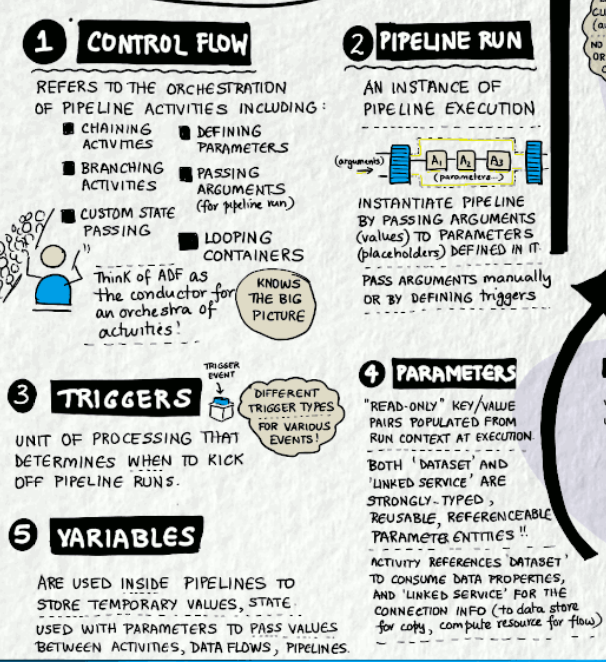
### INTEGRATION RUNTIMES



***These components work together to provide the platform on which you can compose data-driven workflows with steps to move and transform data.***

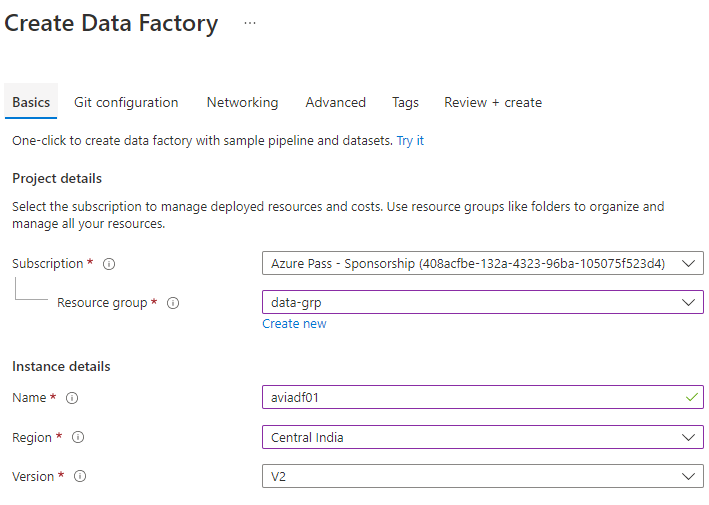
### ADDITIONAL CONCEPTS

1. **CONTROL FLOW**
2. **PIPELINE RUN**
3. **TRIGGERS**
4. **PARAMETERS**
5. **VARIABLES**

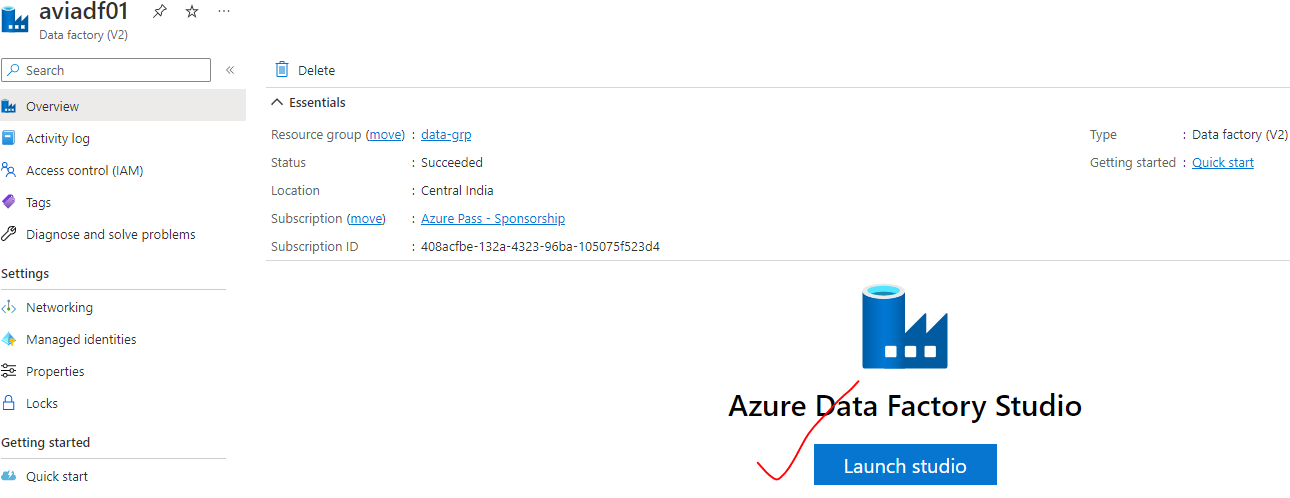


## SETTING UP AZURE DATA FACTORY

|  |  |
| --- | --- |
|  |  |

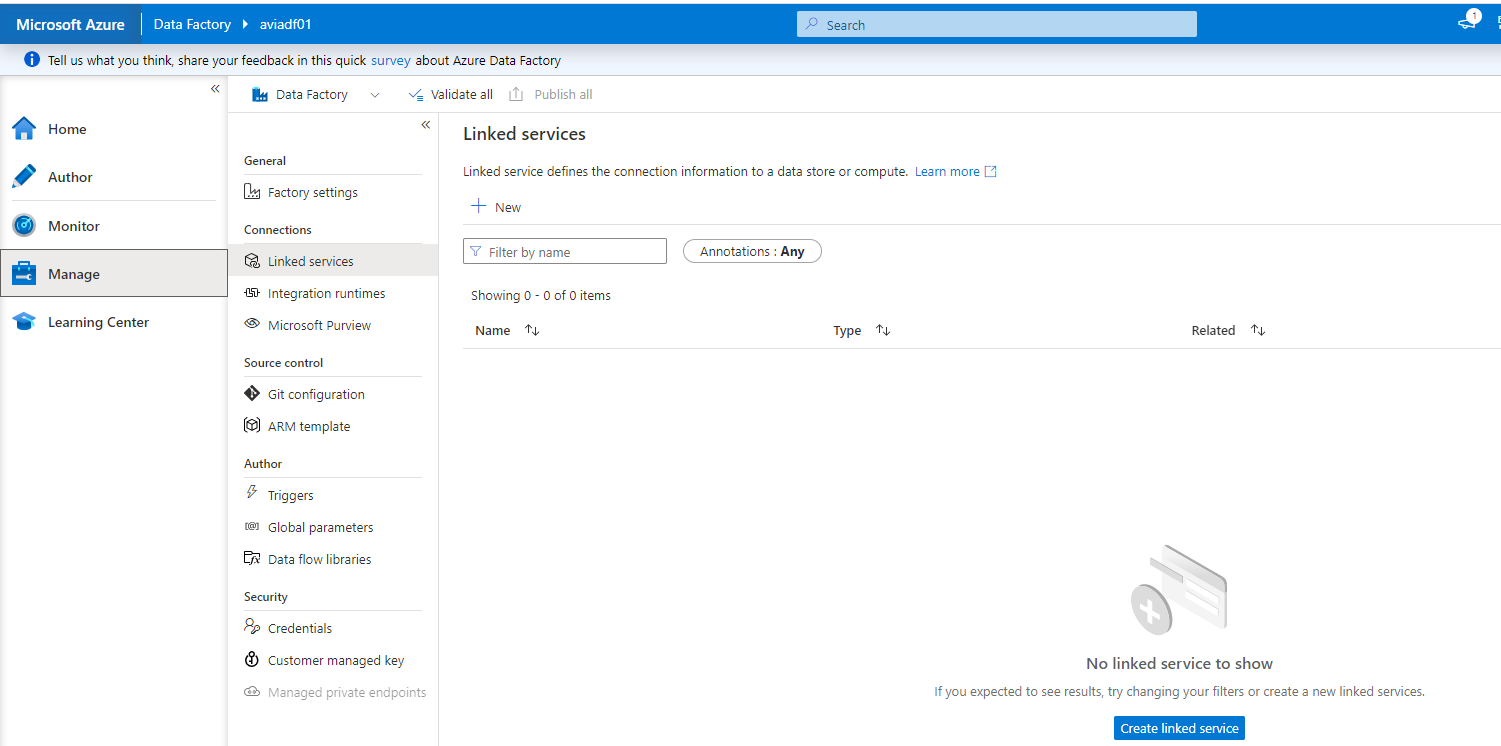


### ADF OVERVIEW



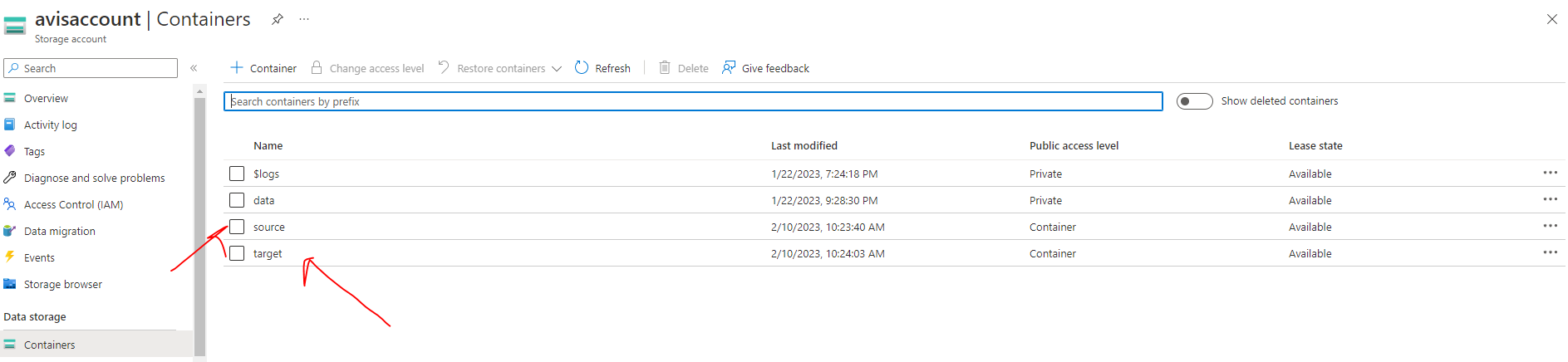
### ADF STUDIO

Azure Data factory Studio in web-based development IDE



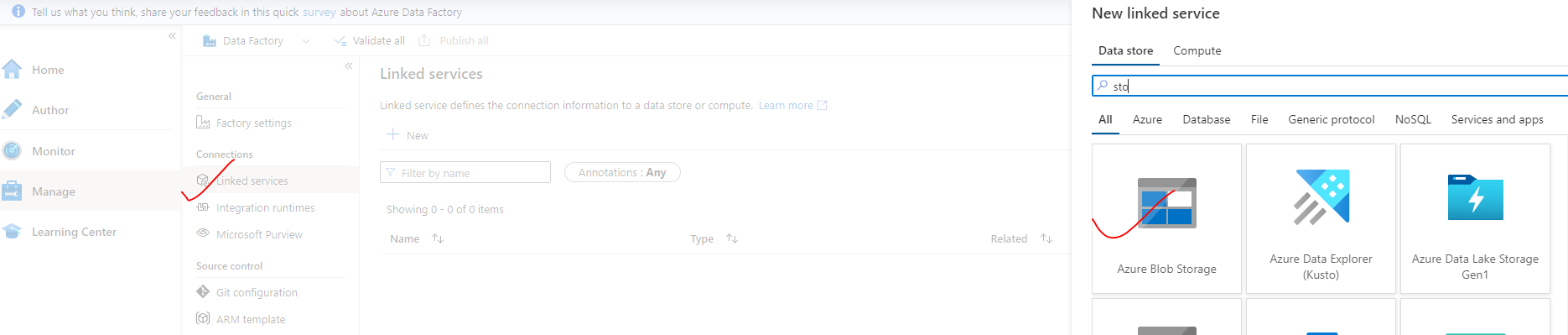
### SETTING UP ADF COMPONENTS

* To illustrate the ADF flow process – lets create demonstrate a use case
* **USE CASE – We will copy a file(e.g. JSON File) from source container to target container in storage account using ADF**



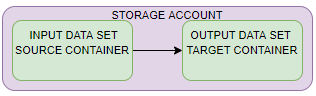
#### STEP 1: SET UP A LINKED SERVICE

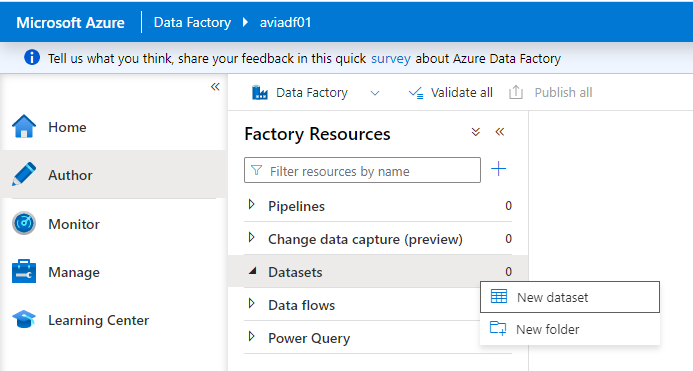
* The linked service helps us to link ADF with external service (here its Storage Account)



|  |  |
| --- | --- |
|  | * Note – ADF will have an Integration runtime created by default. This compute resource help as connect with any Azure PASS * For connection with on-prem external service we need to create a new IR. * While creating the Linked Service 🡪 Test Connection 🡪 Create |

#### STEP 2: SET UP DATASET

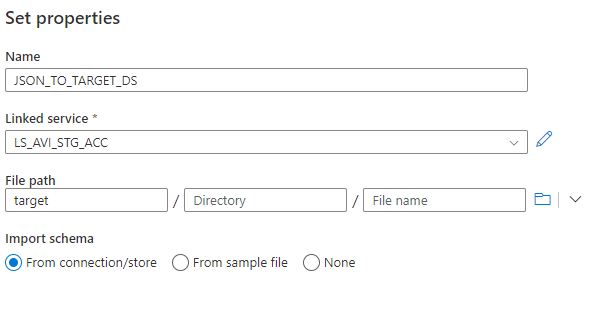




1. SET UP INPUT DATASET

|  |  |  |
| --- | --- | --- |
|  |  | Select the Storage Type for the Blob Storage and file format |
| SET UP THE PROPERTY   * Browse the path of the file from the “File path” field. | | |

1. SET UP OUTPUT DATASET



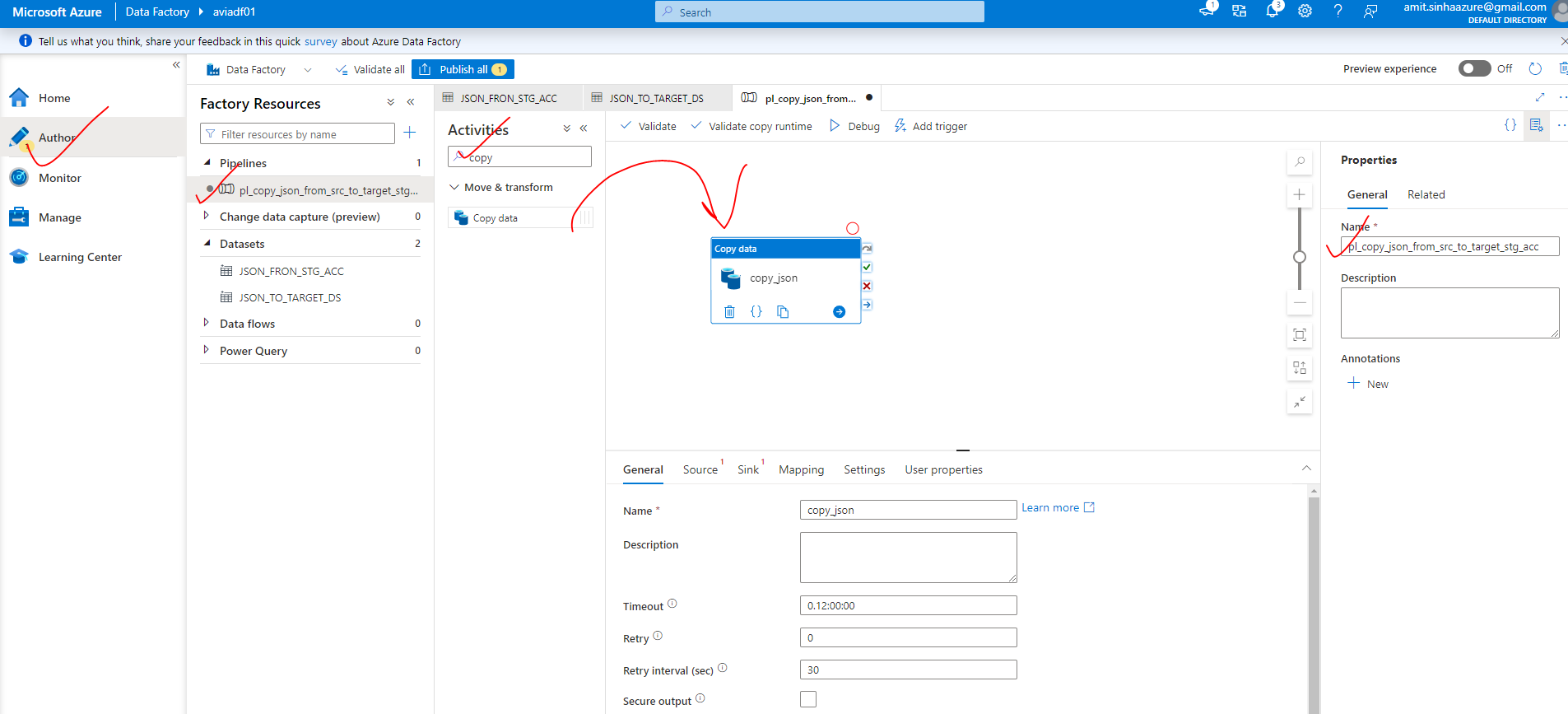
1. PUBLISH THE DATA SET

* If the dataset is not published – it will be lost if browser is refreshed.

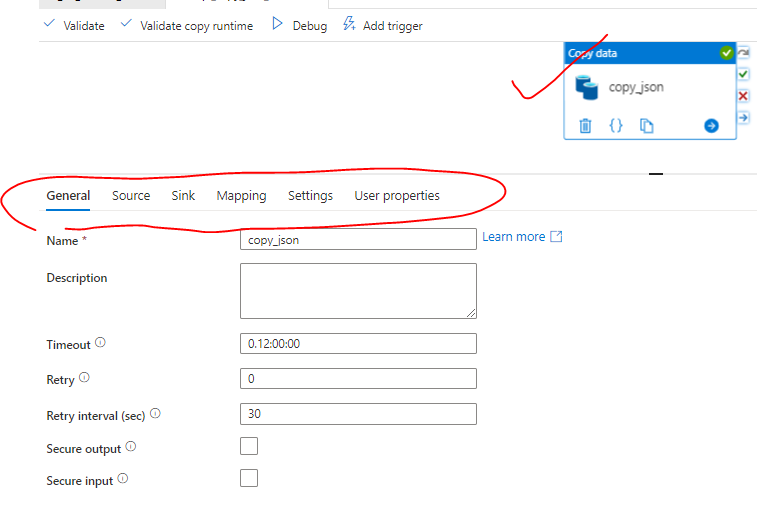


#### STEP 3: SET UP A PIPELINE

* Pipeline consists of activities . Here to for the copy the file – we will drag and drop the copy activity in the pipeline

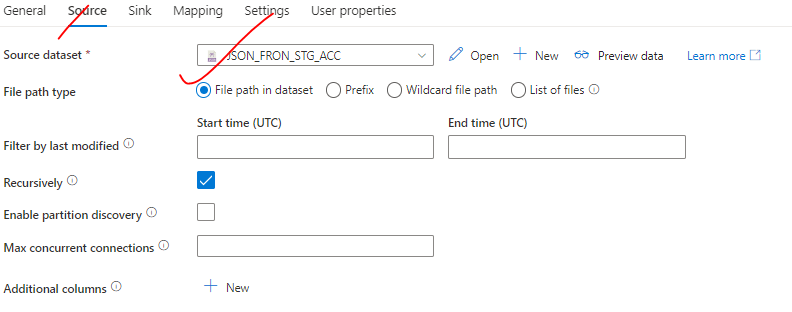


The activity is consisting of many properties

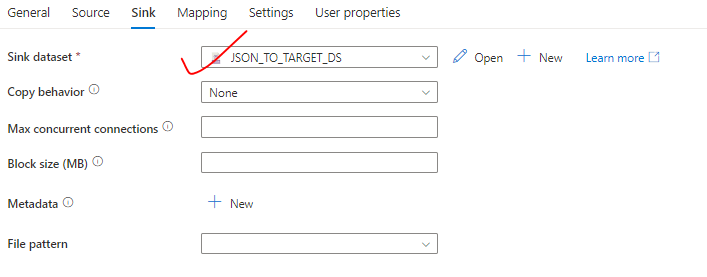


|  |  |
| --- | --- |
| Name | Name of the activity |
| Timeout | The time after which the activity will be timed out |
| Retry | Number of retries |
| Retry Interval | Interval after which next retry will happen |
| Secure output/input |  |

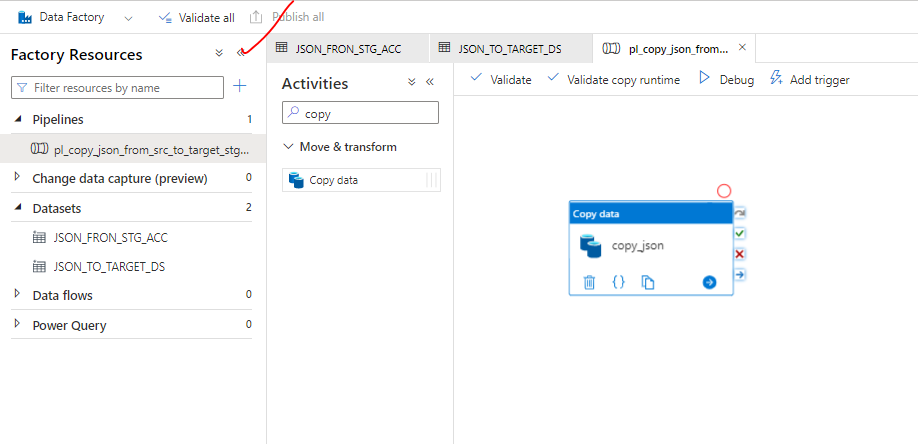
1. SELECT THE SOURCE DATA SET



1. SELECT THE TARGET DATA SET

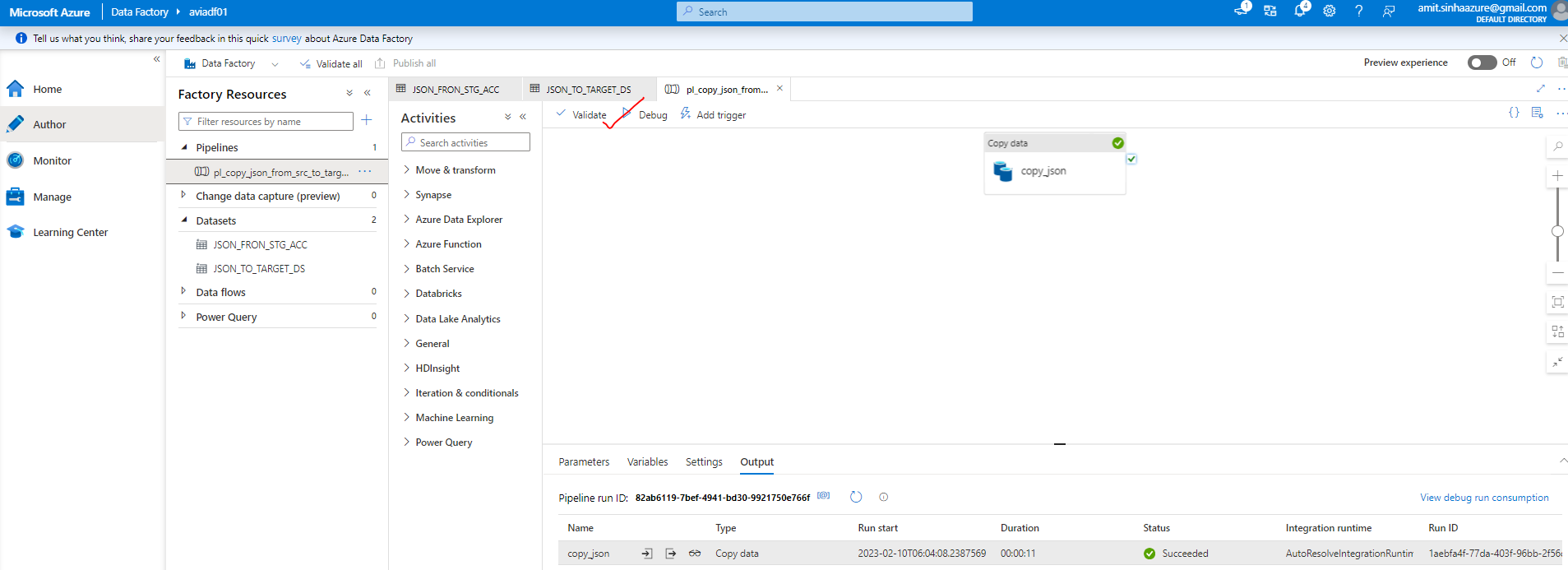


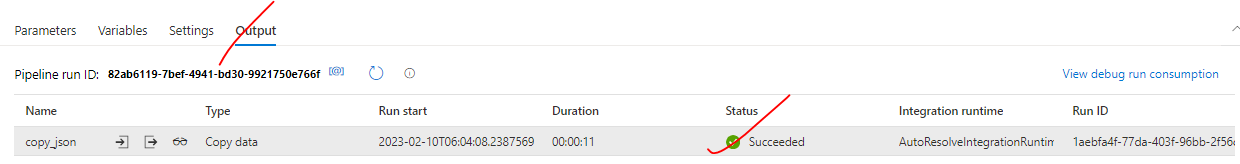
1. PUBLISH THE PIPELINE



1. RUNNING THE PIPELINE

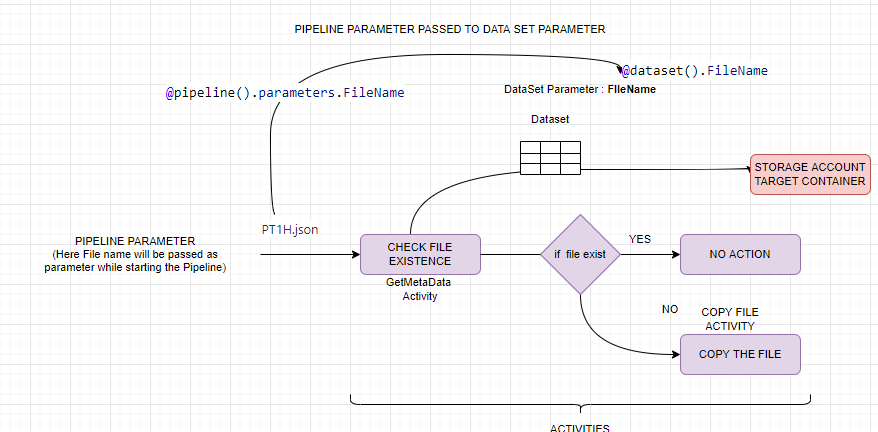
* In real time – We set up triggers to trigger the pipeline. But we can run the pipeline with debug mode for testing.
* This will copy the JSON file from source container to target container in the storage account.





#### USECASE 2: COPY THE FILES ONLY IF ITS NOT PRESENT IN TARGET CONTAINER OF STORAGE ACCOUNT

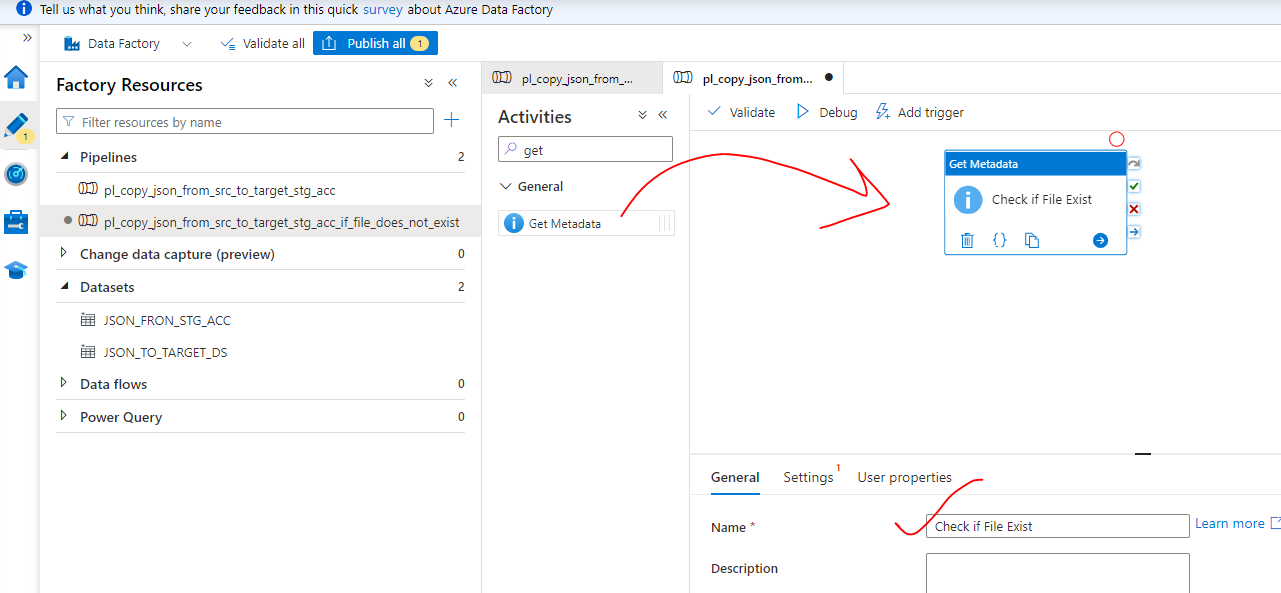
|  |  |
| --- | --- |
| **If(fileExist){**  **Do not copy**  **}else {**  **Copy the file**  **}** | * In the above use case – we copied / overridden the file in the target container. In this use case we want to copy the data only if the file is not present. * To extend the pipeline further- We can clone the existing pipeline and make changes in it. |



**CONCEPT**

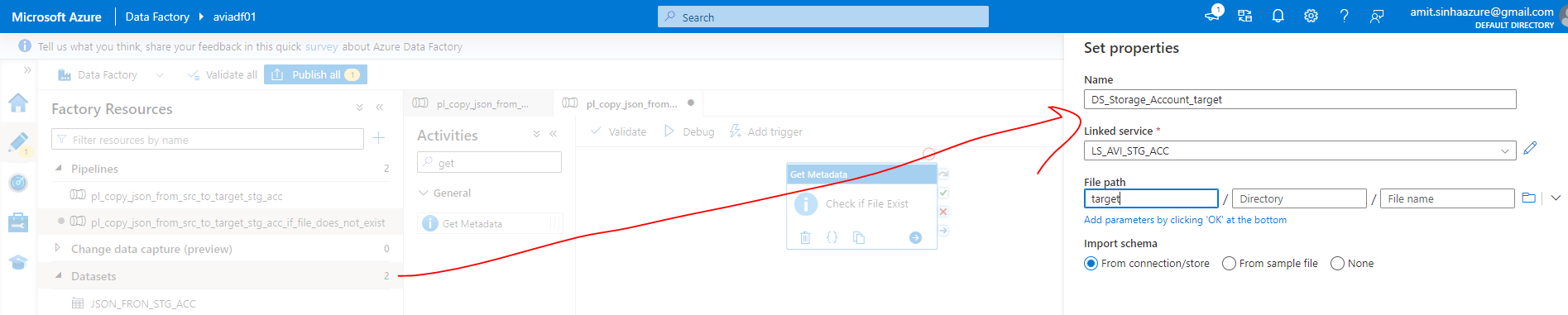
* When we trigger a pipeline – We can pass the value to activities using pipeline parameter
* The passed value to activity will be received by dataset parameter.

**STEP 1: ADD A NEW ACTIVITY CALLED “*Get Metadata*”**

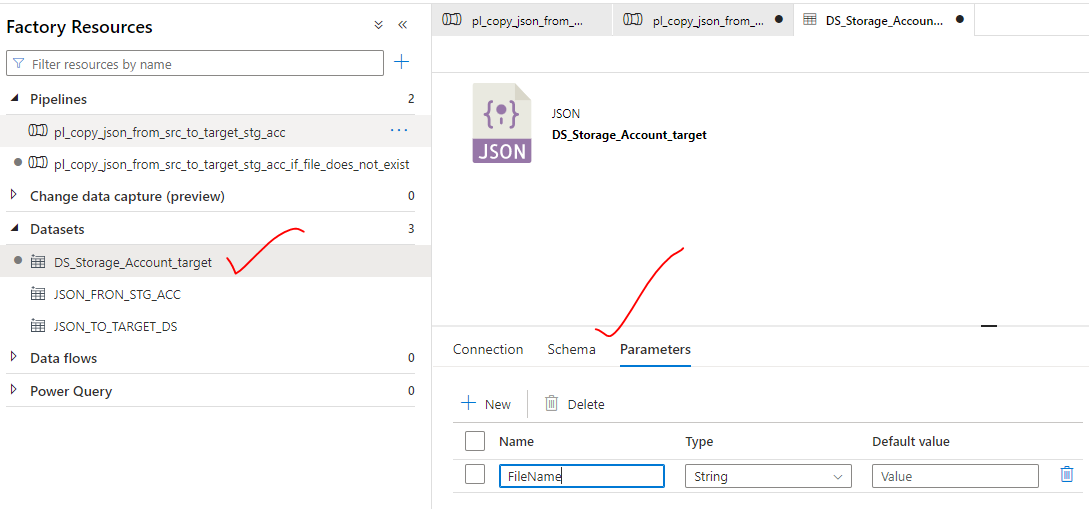


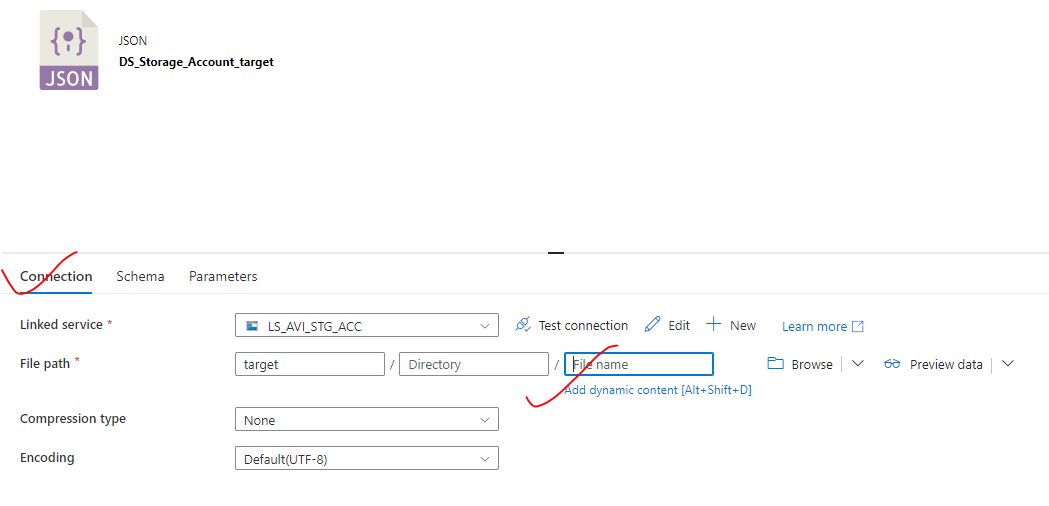
**Step 2: CREATE A NEW DATA SET –**

* The new data set will be on the target side to check the file existence in the target container of storage account.

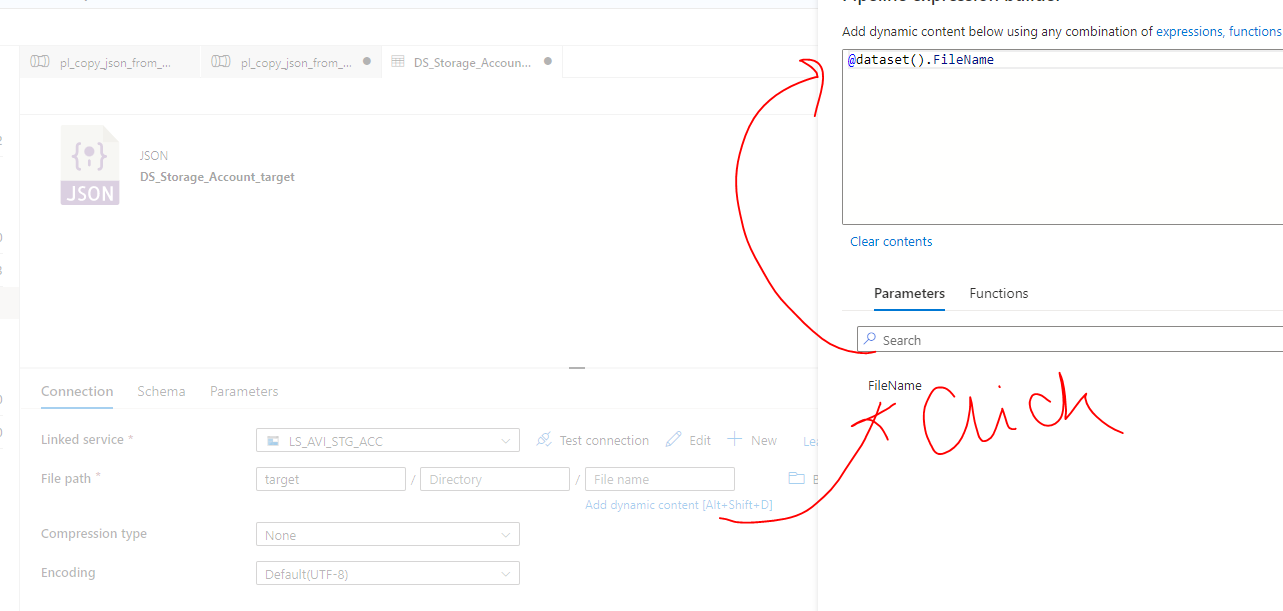


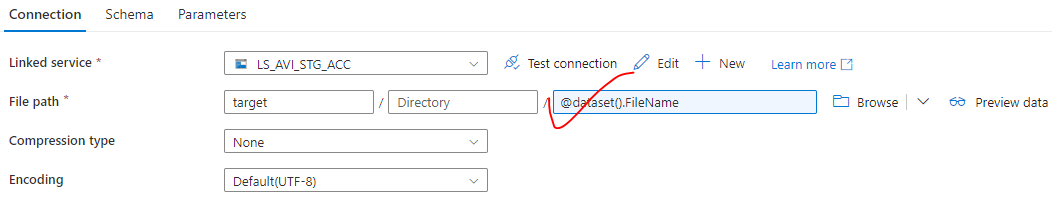
**STEP 3: ADD A “*FILENAME*” DATASET PARAMETER.**

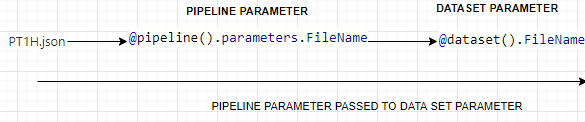




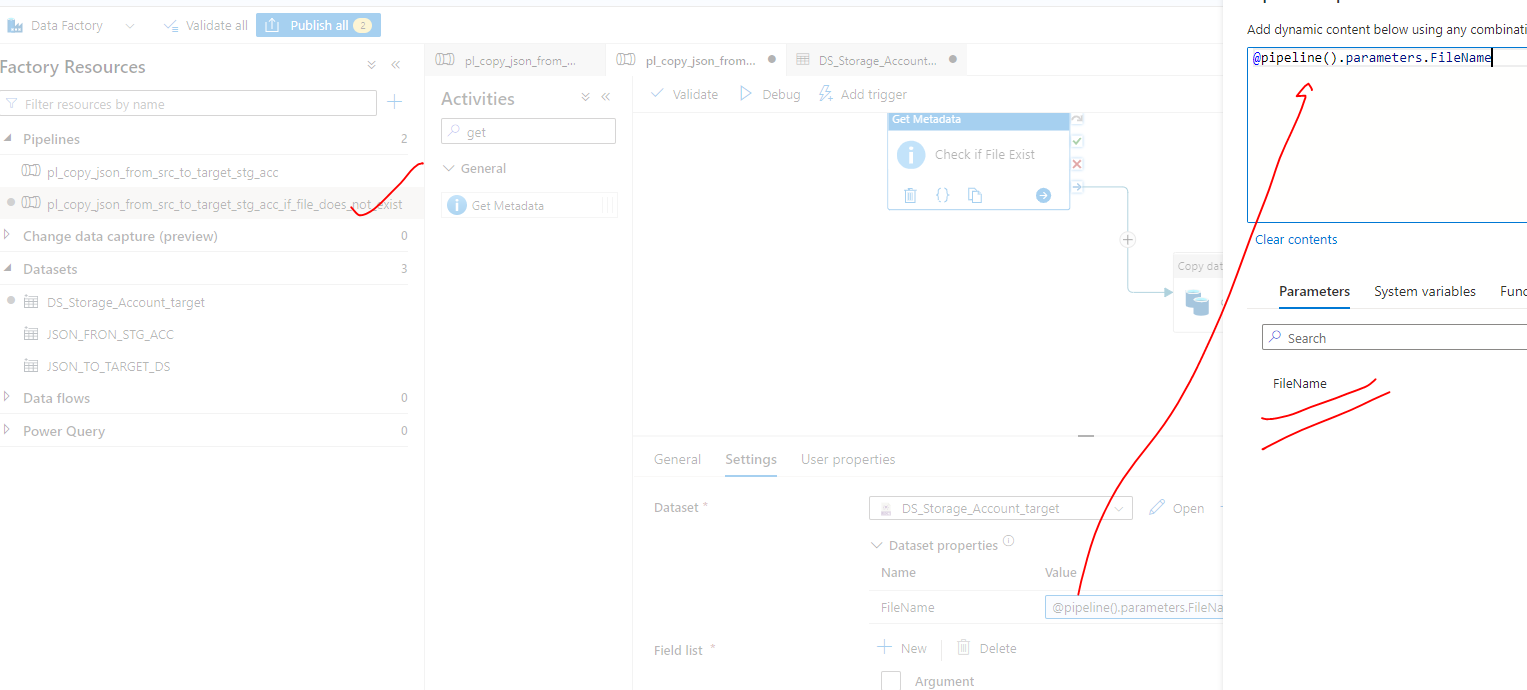
* The pipeline will pass the value to the dataset parameter called - “FileName ” dynamically

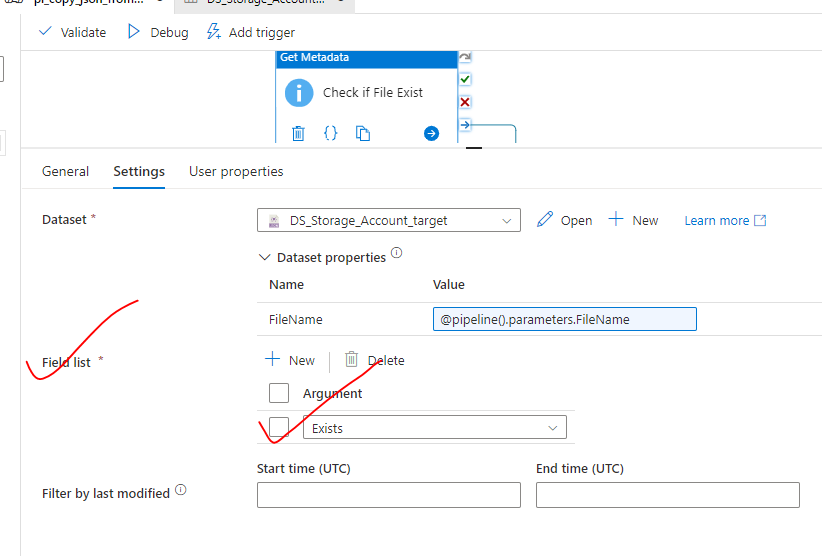






**SETTING UP PIPELINE PARAMETER**

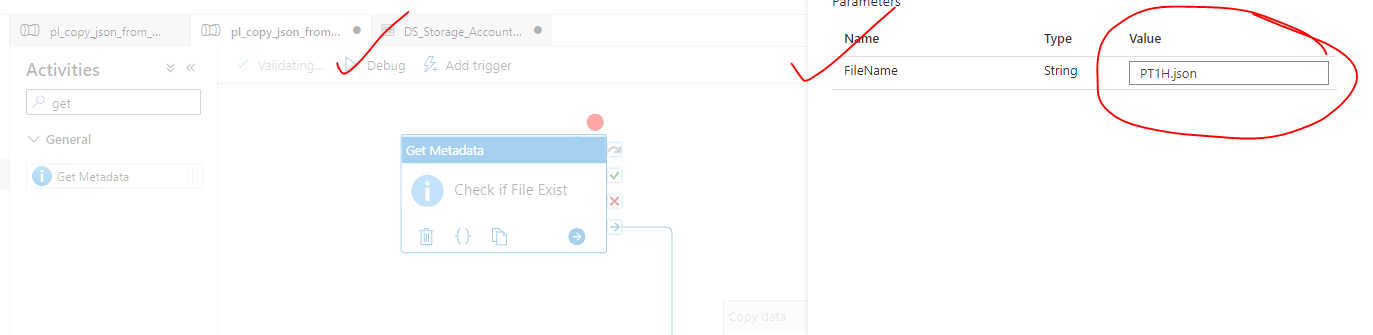




|  |  |
| --- | --- |
|  | **DEBUGING AN ACTIVITY –** An activity can be debugged in isolation (using red dot – Debug until Option) |

**RUNNING THE PIPELINE**

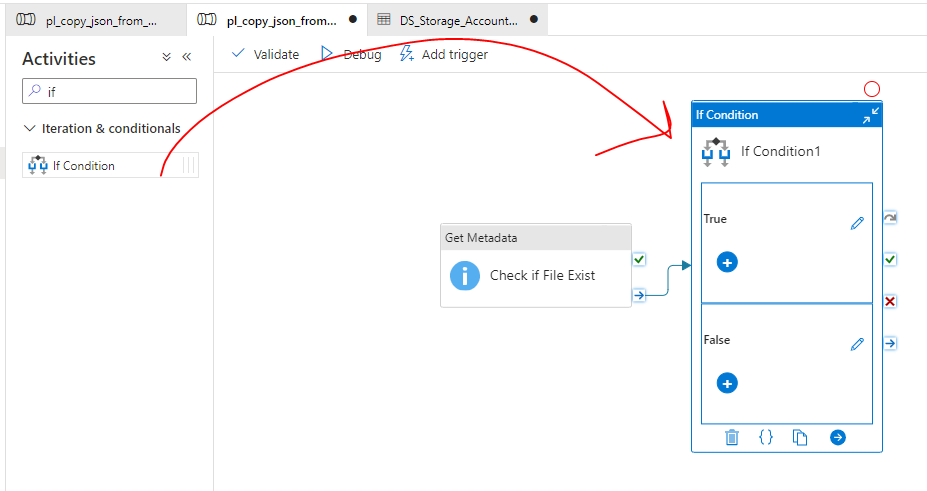
We need to pass the value(filename) while debugging / running the pipeline



|  |  |
| --- | --- |
|  | * Testing succeeded |

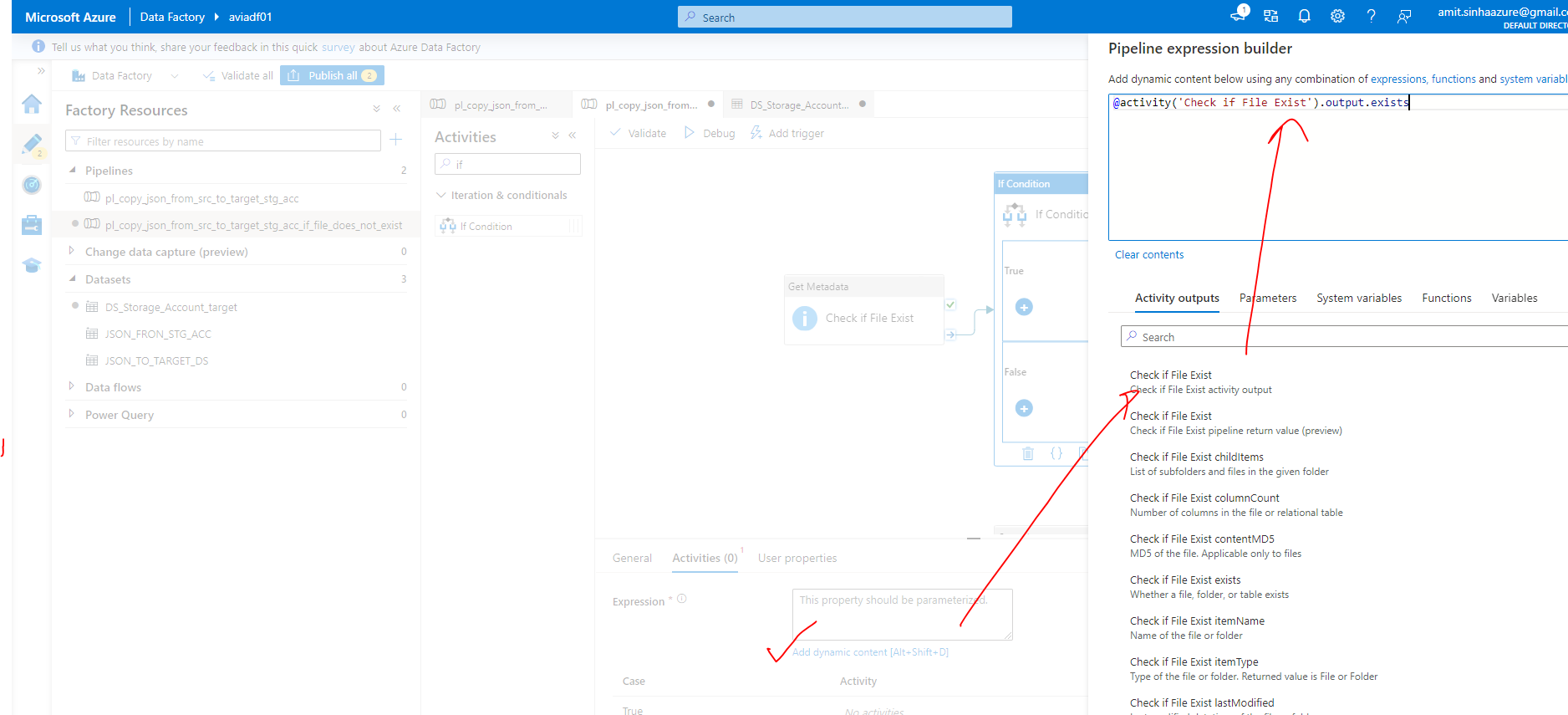
|  |  |
| --- | --- |
|  | * This is the output of the GetMeta Data activity. * The output value “exist=true” will be passed to the “if condition” activity * if the condition is true 🡪 the file will be copied to target container otherwise no action. |

**ADDING IF CONDITION ACTIVITY**



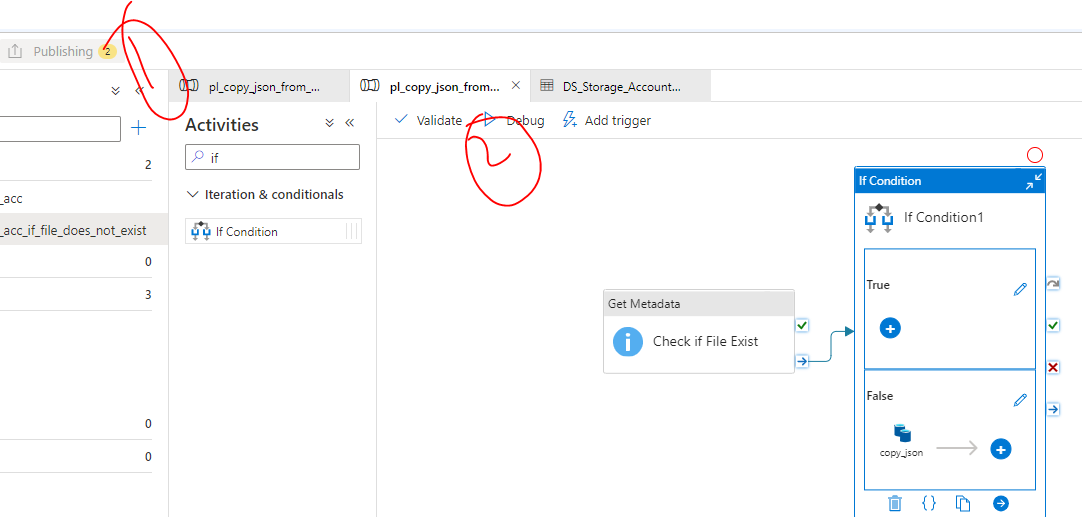
**IF CONDITION ACTIVITY PAREMETER**

* The exist property value from the previous activity (*GetMetaData*) – [@activity('Check if File Exist').output.exists]- will be used in if condition metadata



* Cut the copy data activity to the else block of the “if condition” activity

|  |  |
| --- | --- |
|  |  |



|  |  |
| --- | --- |
|  | * Passing a value to a pipeline |

**OUTPUT – *When the file is present – it executes three activities***

