Archiving and Purging Data Using Azure Data Factory

**Introduction**

In the era of big data, organizations are constantly generating and managing large volumes of data. To ensure smooth operations and optimized storage usage, it is crucial to archive historical data and purge obsolete data efficiently. This project was undertaken to design and implement an automated solution for archiving and purging datasets using **Azure Data Factory (ADF)**.

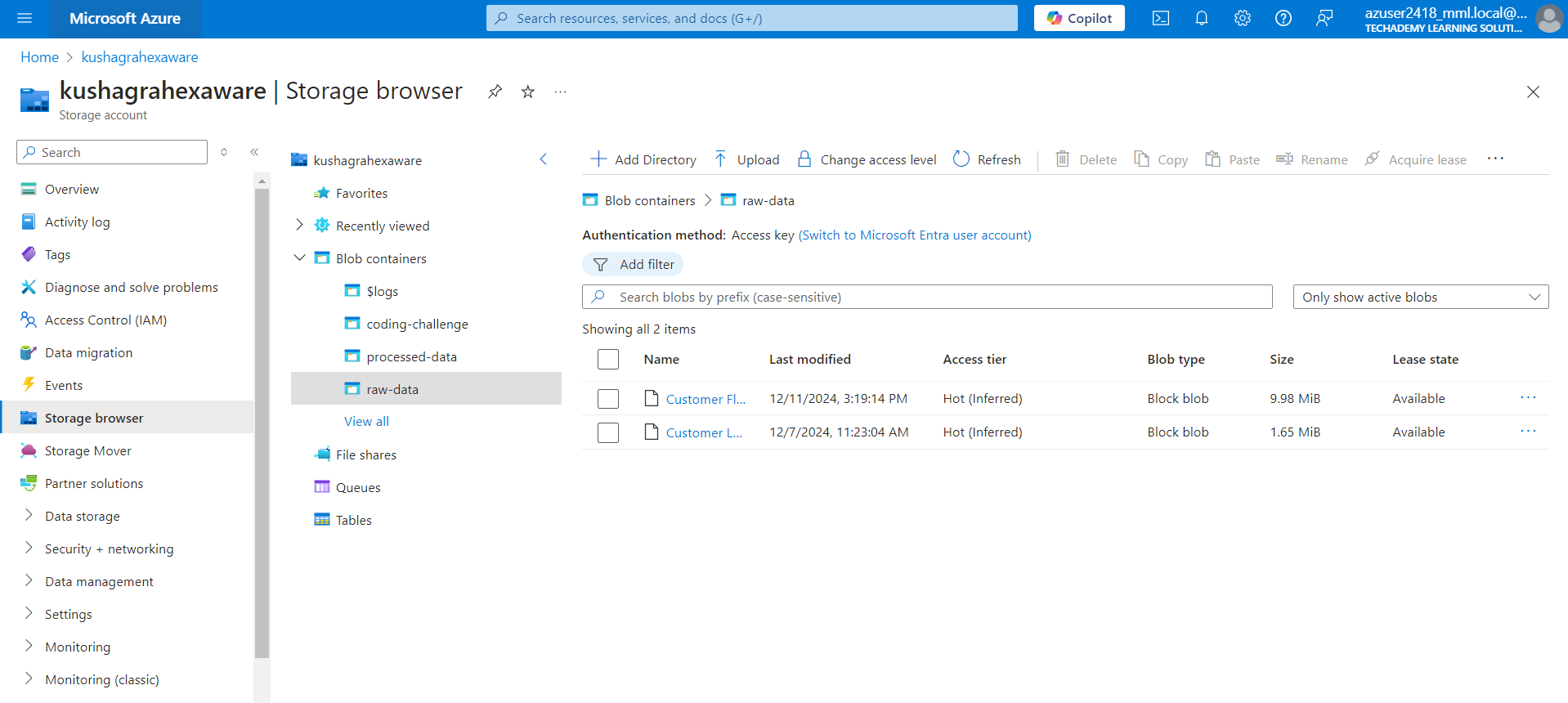
The project aimed to handle data generated between 2010 and 2020 of an Airline Loyalty Program, focusing on:

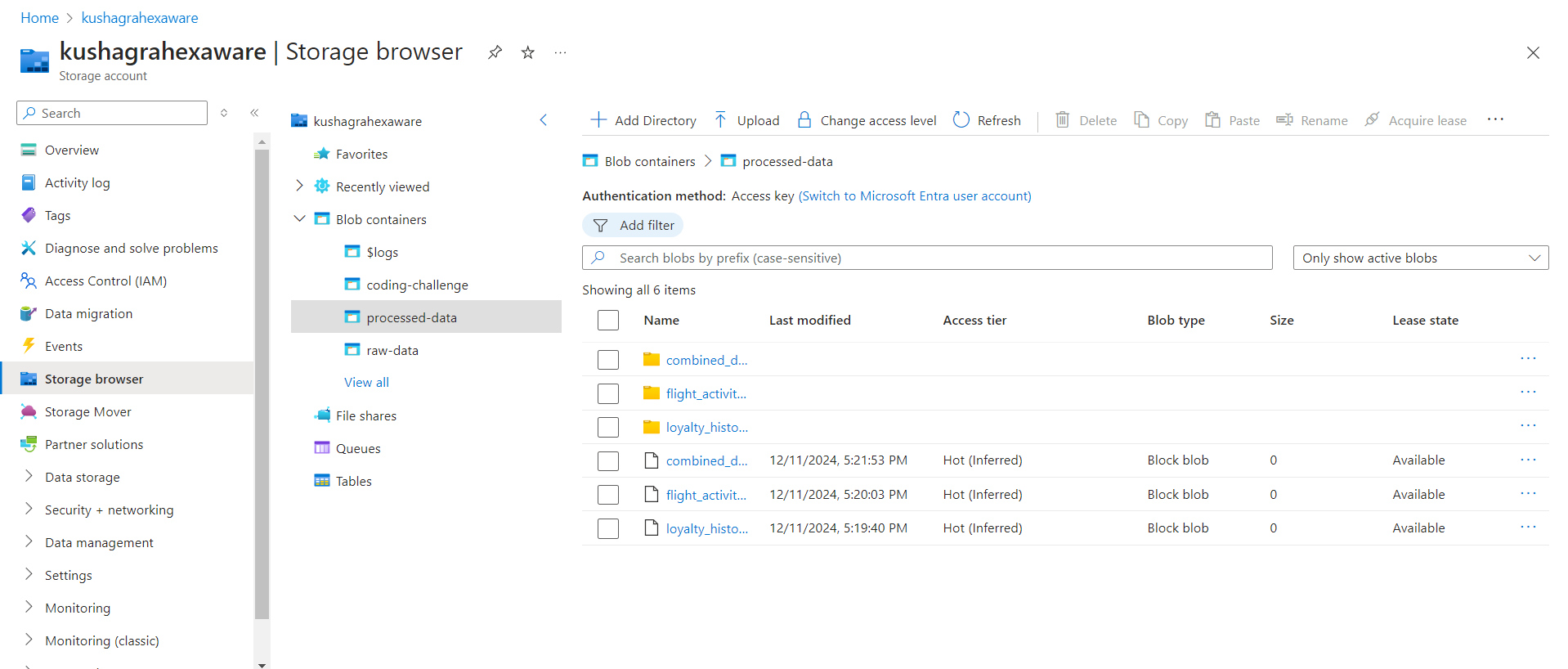
* Archiving historical data (2013–2017) into a dedicated storage location for long-term retention.
* Purging older, obsolete data (2010–2012) while logging its details for audit purposes.
* Performing **EDA** on relevant data (2018-2020) based on selected use-case.
* Automating the entire process to reduce manual intervention.

**Objectives**

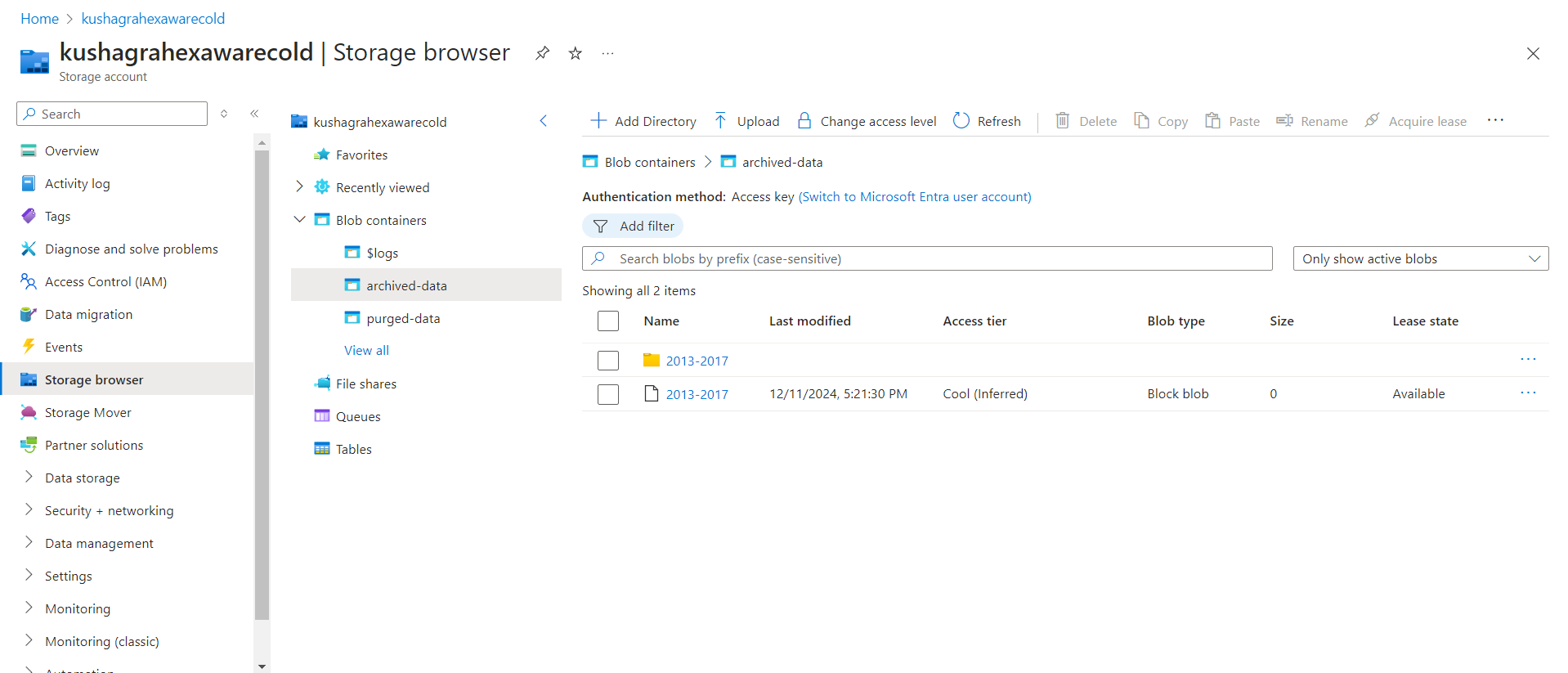
The primary objectives of the project were:

1. **Clean Data**: Process raw data stored in the raw-data container, clean it, and store the output in the processed-data container as Parquet files.

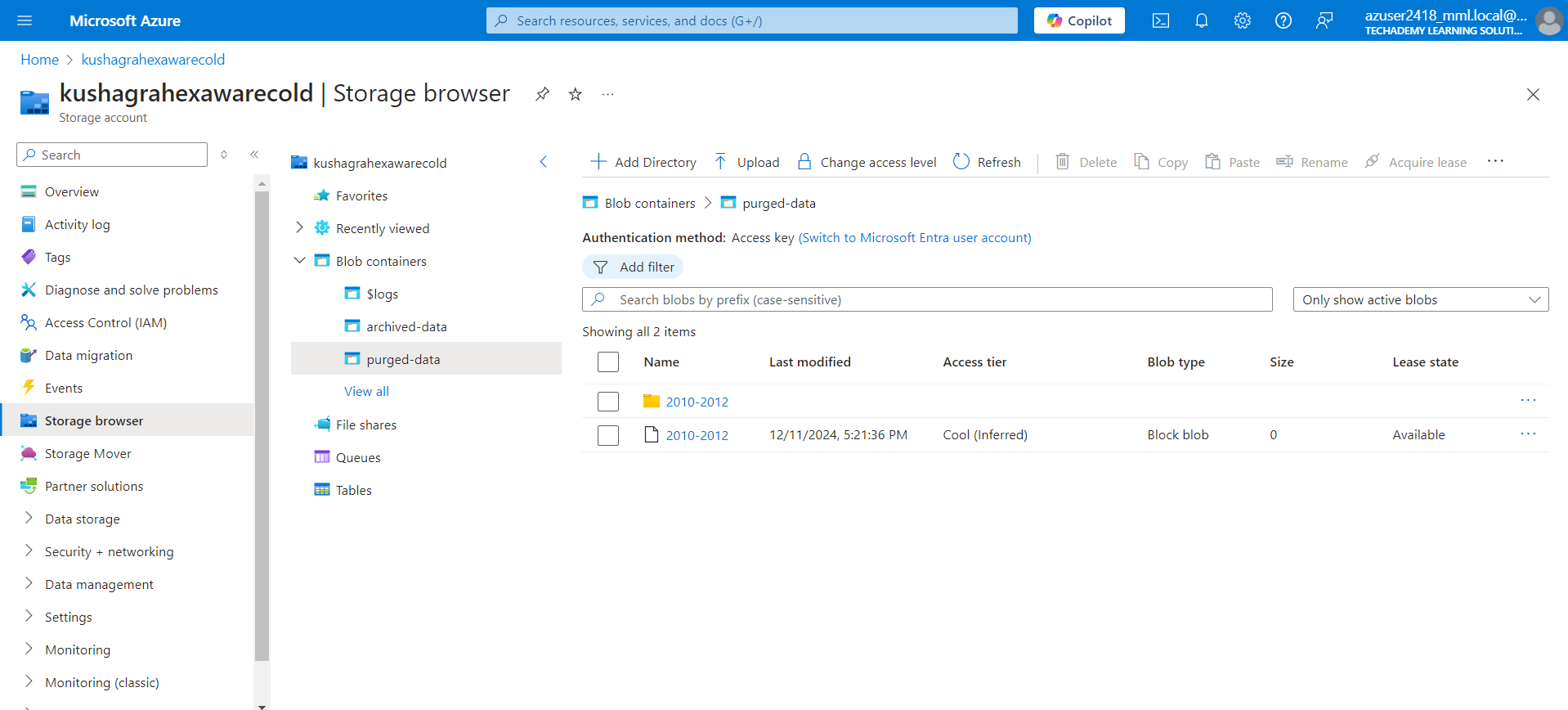




1. **Archive Data**: Move historical data (from specific years) to a dedicated archival location using cost-effective cool storage.



1. **Purge Data**: Remove obsolete data while logging its metadata for traceability.



1. **Automation**: Design pipelines in Azure Data Factory to automate the cleaning, archiving, and purging workflows.
2. **Cost Optimization**: Leverage separate storage accounts for archived and purged data containers, utilizing cool storage for reduced costs.

**System Design**

The solution consisted of a main pipeline orchestrated using Azure Data Factory that performed cleaning, archiving and purging activity effectively.

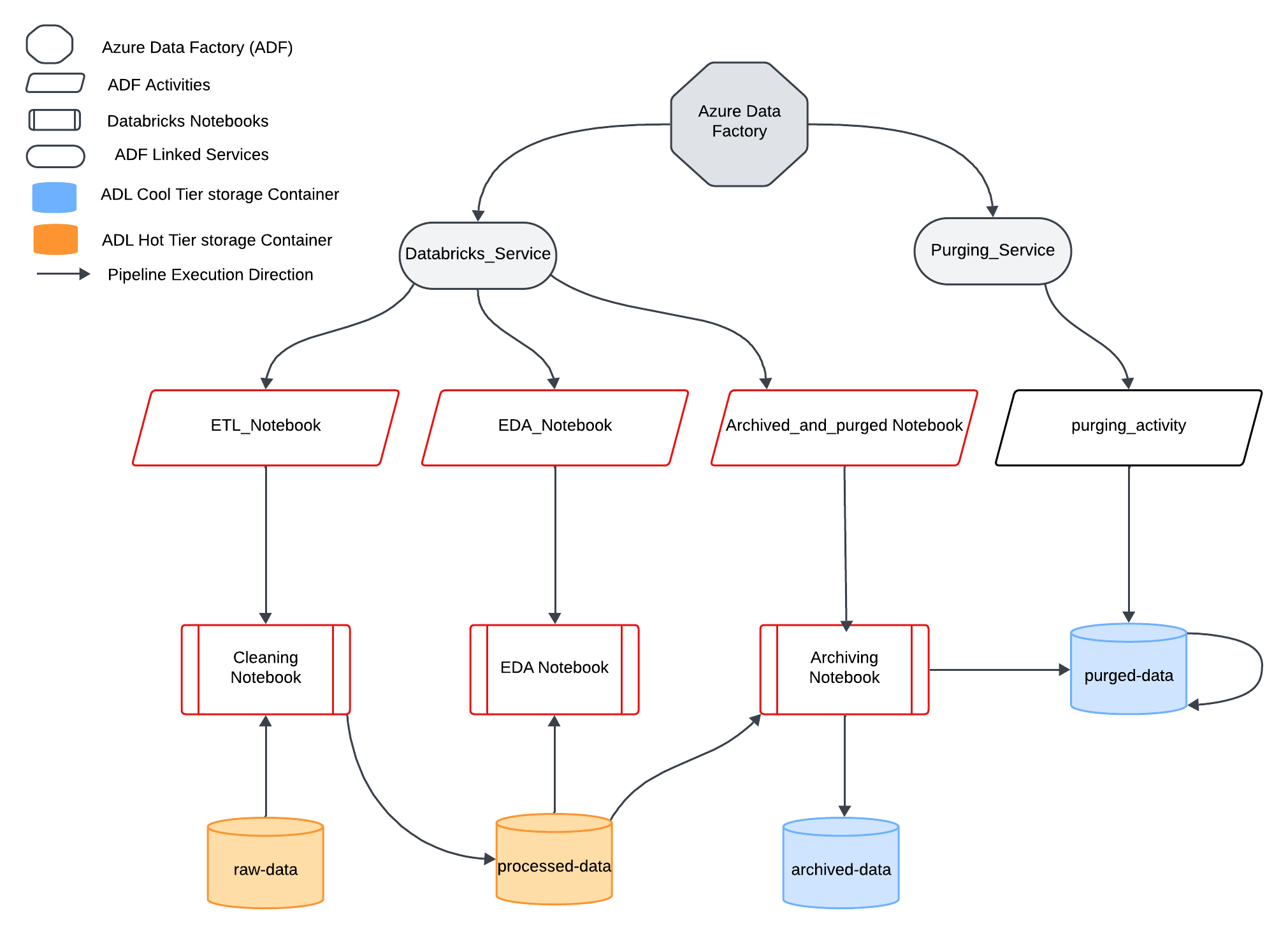
Folder Structure

The project used the following folder structure across multiple Azure Blob Storage accounts:

* **raw-data (Hot Storage Account)**: Stores unprocessed raw datasets.
* **processed-data (Hot Storage Account)**: Stores cleaned data in Parquet format for further processing.
* **archived-data (Cool Storage Account)**: Stores archived data for long-term retention.
* **purged-data (Cool Storage Account)**: Temporarily holds obsolete datasets before deletion.

This segregation ensured optimized performance, cost savings, and compliance with data lifecycle policies.

**Architecture Diagram**



**Implementation**

The implementation was carried out using a single pipeline in Azure Data Factory (ADF), which orchestrated the execution of three notebooks:

1. **Data Cleaning Notebook**
2. **Data Archiving Notebook**
3. **Exploratory Data Analysis (EDA) Notebook**

Additionally, a Linked Service deletion activity was configured within the pipeline to handle data purging.

**Pipeline Overview**

The pipeline was designed to execute the notebooks sequentially, ensuring smooth data transformation, archival, and analysis. The steps in the pipeline are as follows:

1. **Data Cleaning Notebook Execution**
2. **Data Archiving Notebook Execution**
3. **EDA Notebook Execution**
4. **Data Purging Activity**

This sequential structure ensured that the datasets flowed seamlessly through the data lifecycle, from cleaning to archiving, analysis, and eventual purging of obsolete data.

1. Data Cleaning Notebook

The **Data Cleaning Notebook** was the first step in the pipeline. It processed raw data stored in the raw-data container and transformed it into a cleaned format for downstream operations.

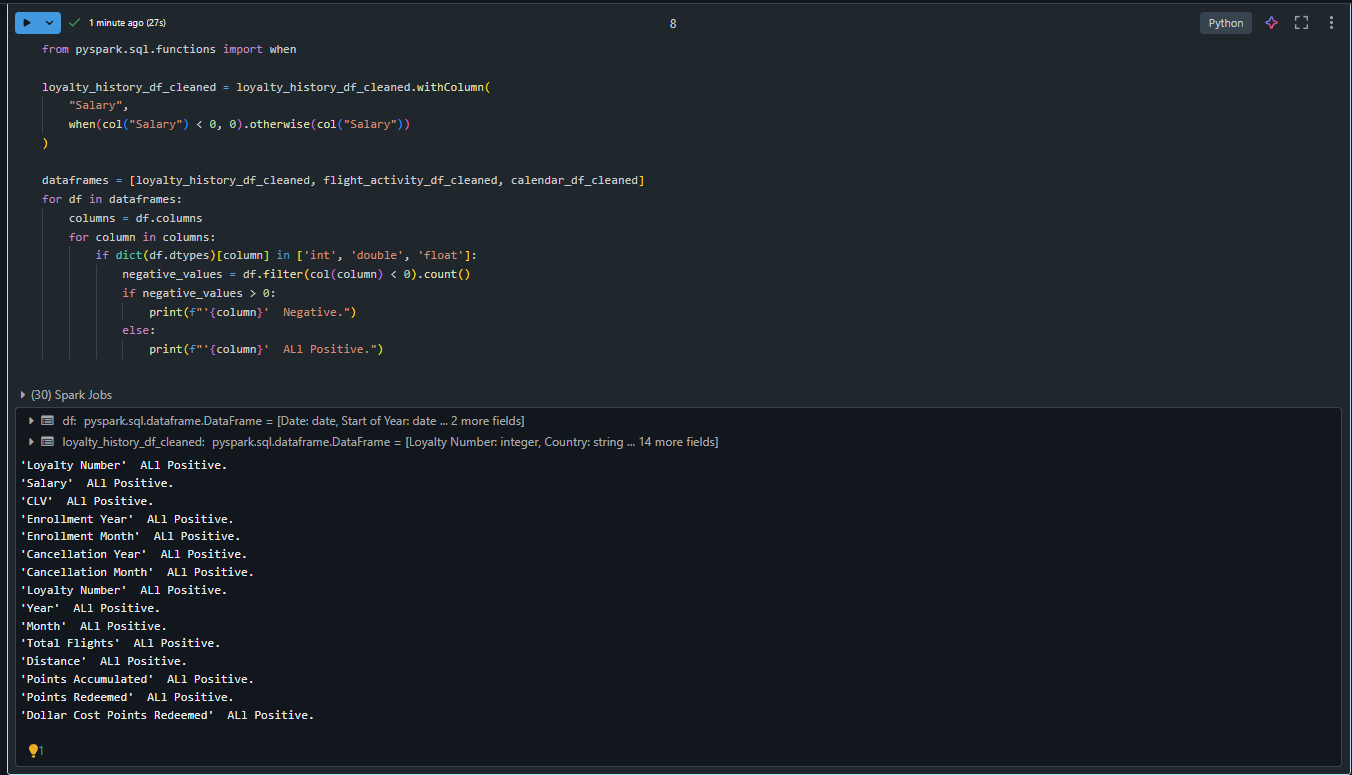
**Key Steps:**

* Reading the raw data.

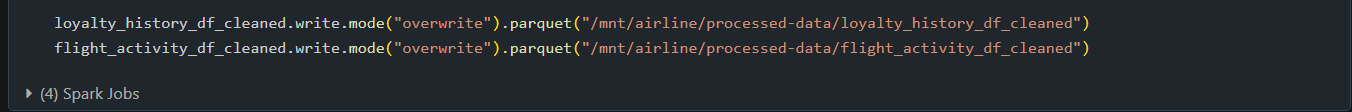




* Applying transformations to remove null values, and resolve inconsistencies.



* Saving the cleaned data in **Parquet format** in the processed-data container.

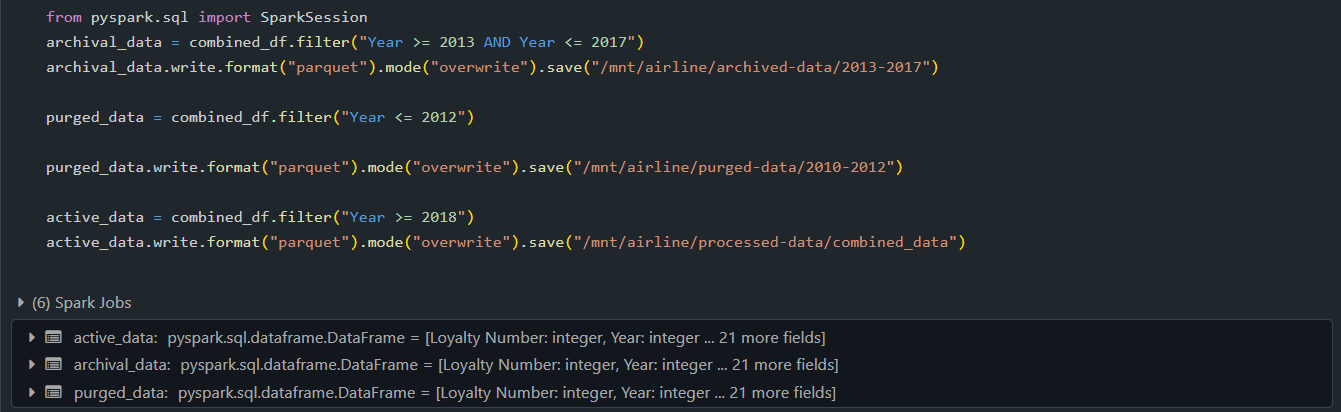


2. Data Archiving Notebook

The **Data Archiving Notebook** was the second step, responsible for moving historical data to cost-efficient cool storage in the archived-data container.

**Key Steps:**

* Filtering data for archival (e.g., 2013–2017).



* Saving the archived data in the archived-data container in Parquet format.

3. EDA Notebook

The **EDA Notebook** was designed to analyze a subset of relevant data (2018–2020) to showcase a use case. This step demonstrated how cleaned and structured data could be leveraged for generating actionable insights.

**Use Case**: *What impact did the campaign have on booked flights during summer?*

**Key Steps:**

* Filtering relevant data for the years 2018–2020.
* Analyzing summer months (June–August) to evaluate trends in flight bookings.
* Visualizing insights with graphs and charts to identify the campaign's impact.

The analysis provided clear trends that highlighted the success of the campaign.

4. Data Purging Activity

The purging process was automated using a Linked Service deletion activity in Azure Data Factory. This activity identified and removed obsolete data (e.g., 2010–2012) from the processed-data container while ensuring compliance with traceability requirements.

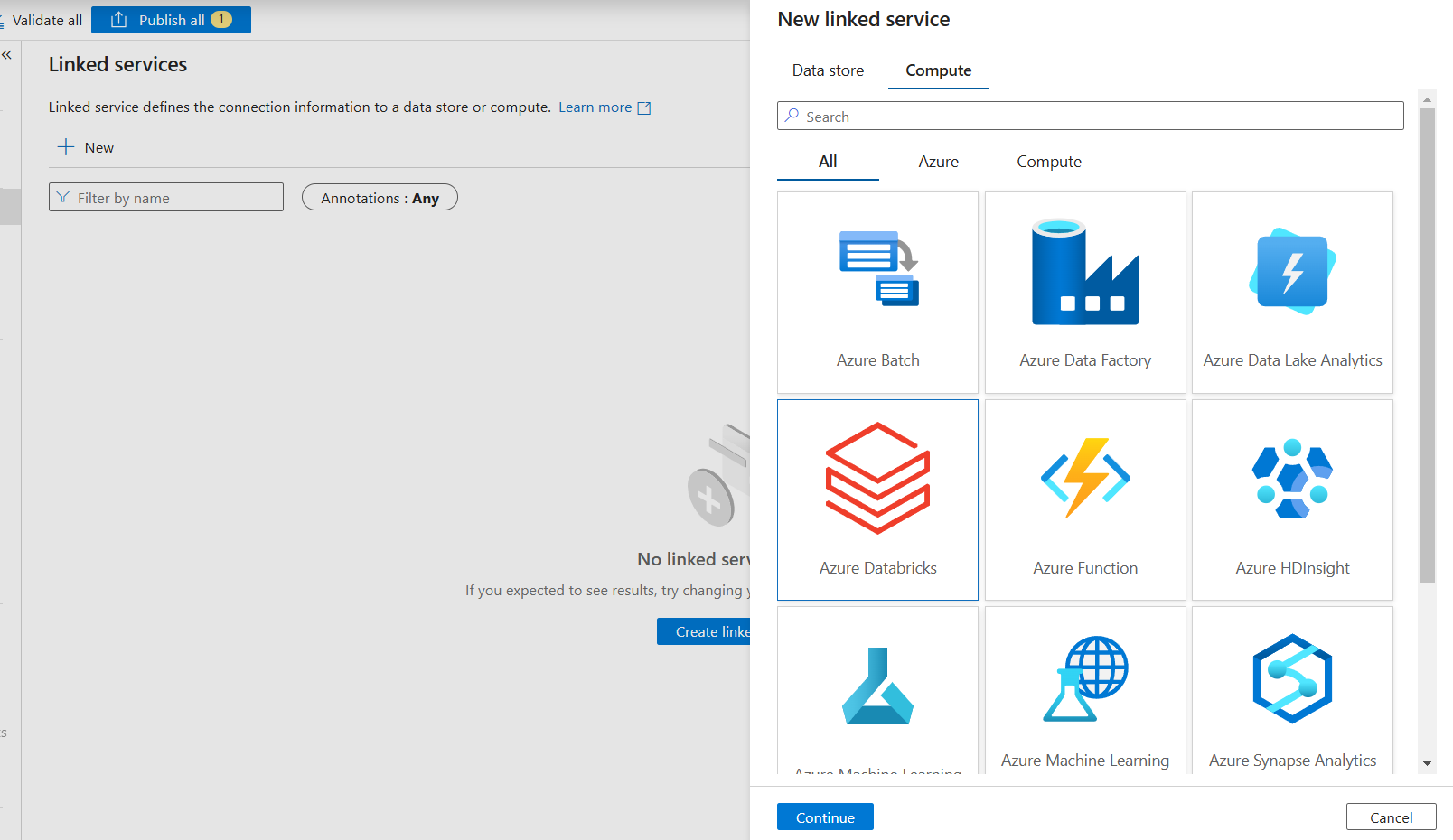
**Steps:**

1. **Identify Data for Purging**: Configured filters in ADF to target specific data ranges (2010–2012).
2. **Log Metadata**: Logged details of deleted files (e.g., file paths, sizes, and timestamps) in an Azure SQL Database table named PurgedDataLogs.
3. **Delete Data**: Removed the identified files permanently from the storage container.

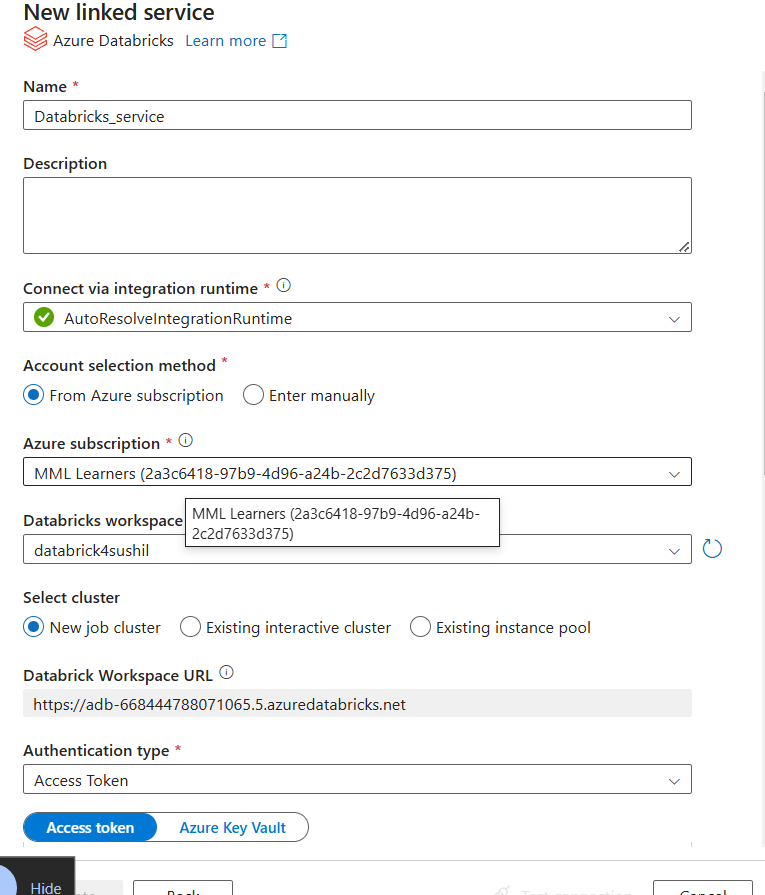
**Automating the entire process in ADF (Azure Data Factory) to reduce manual intervention.**

1)Creation of Linked Service in ADF for accessing Azure Databricks Notebooks.

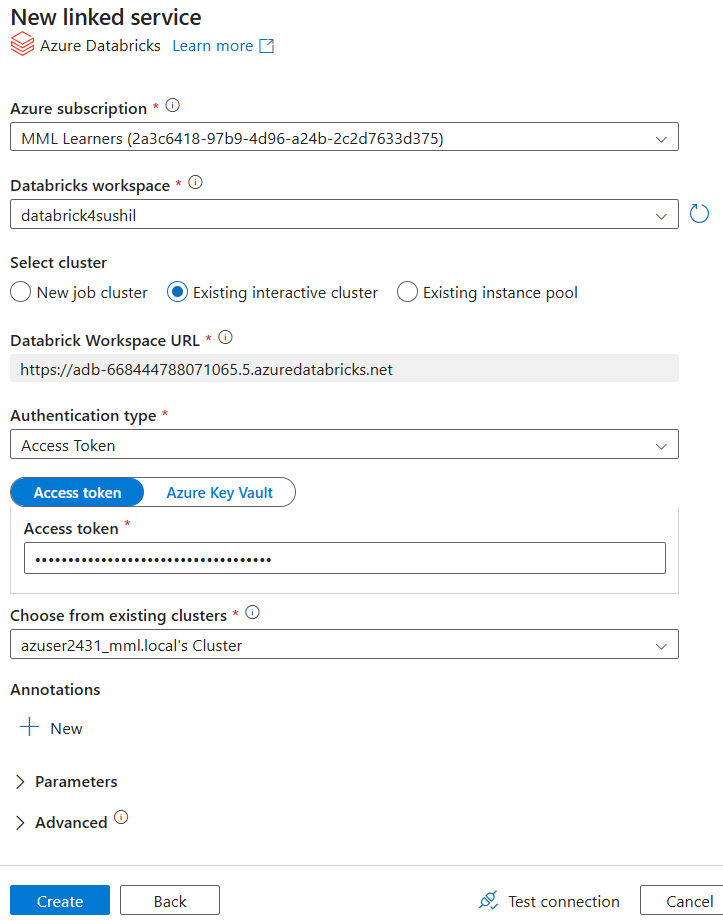
–used Azure Databricks compute for new linked service



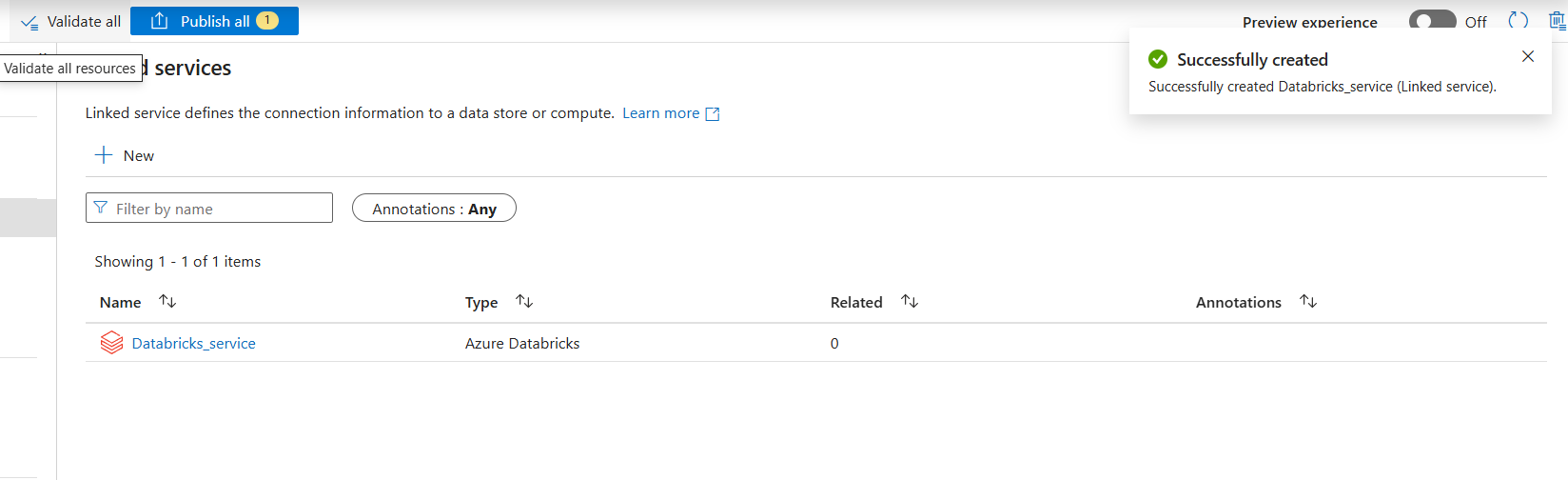
–Named it Databricks\_service, selected subscription provided by Hexaware, selected workspace (Azure resource) used for Databricks.



– Selected existing cluster, accessed using access token we can get in databricks (setting→user→developer→access\_tokens→generate\_new\_token)

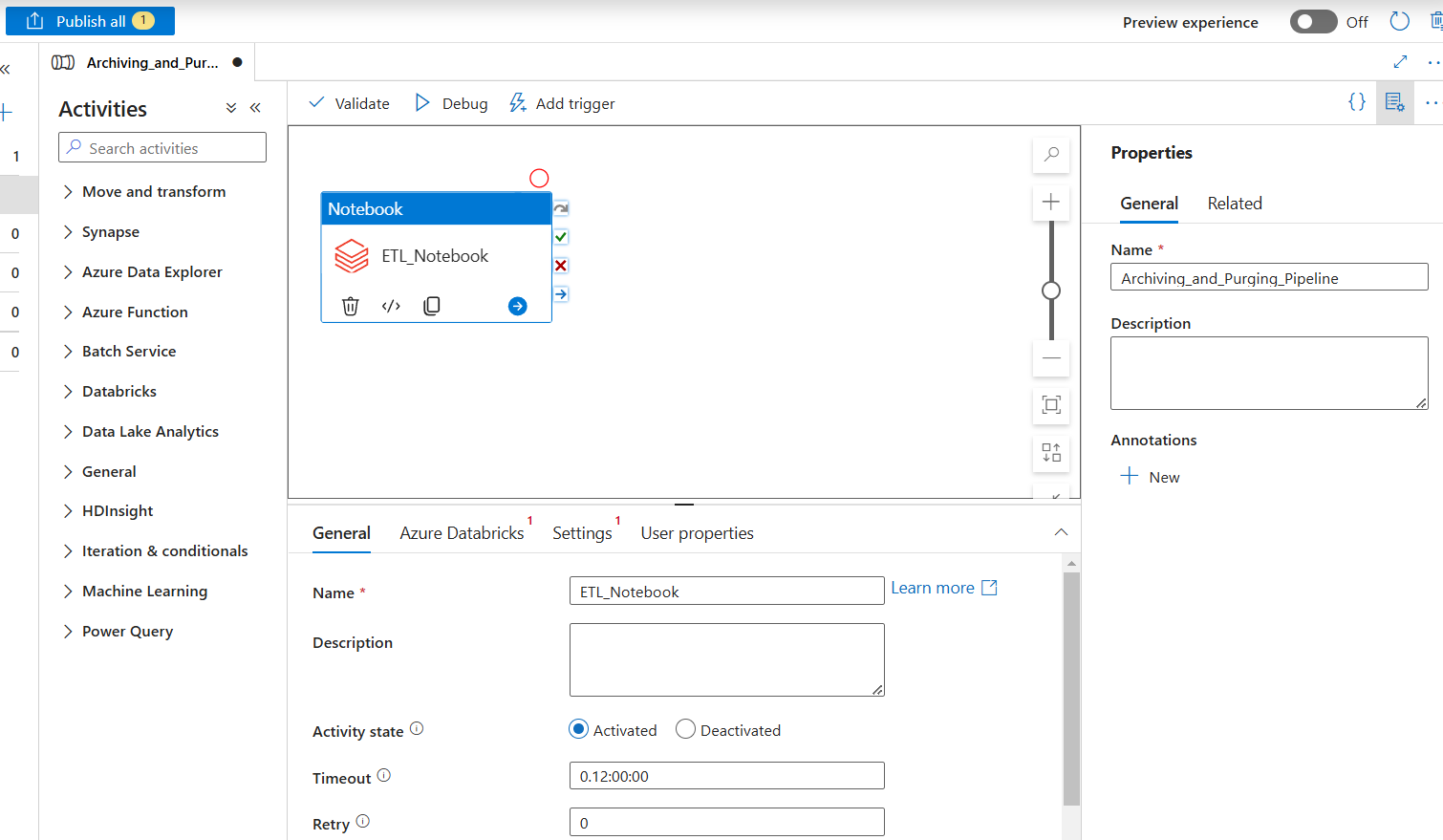


–Linked service for Databricks created Successfully .

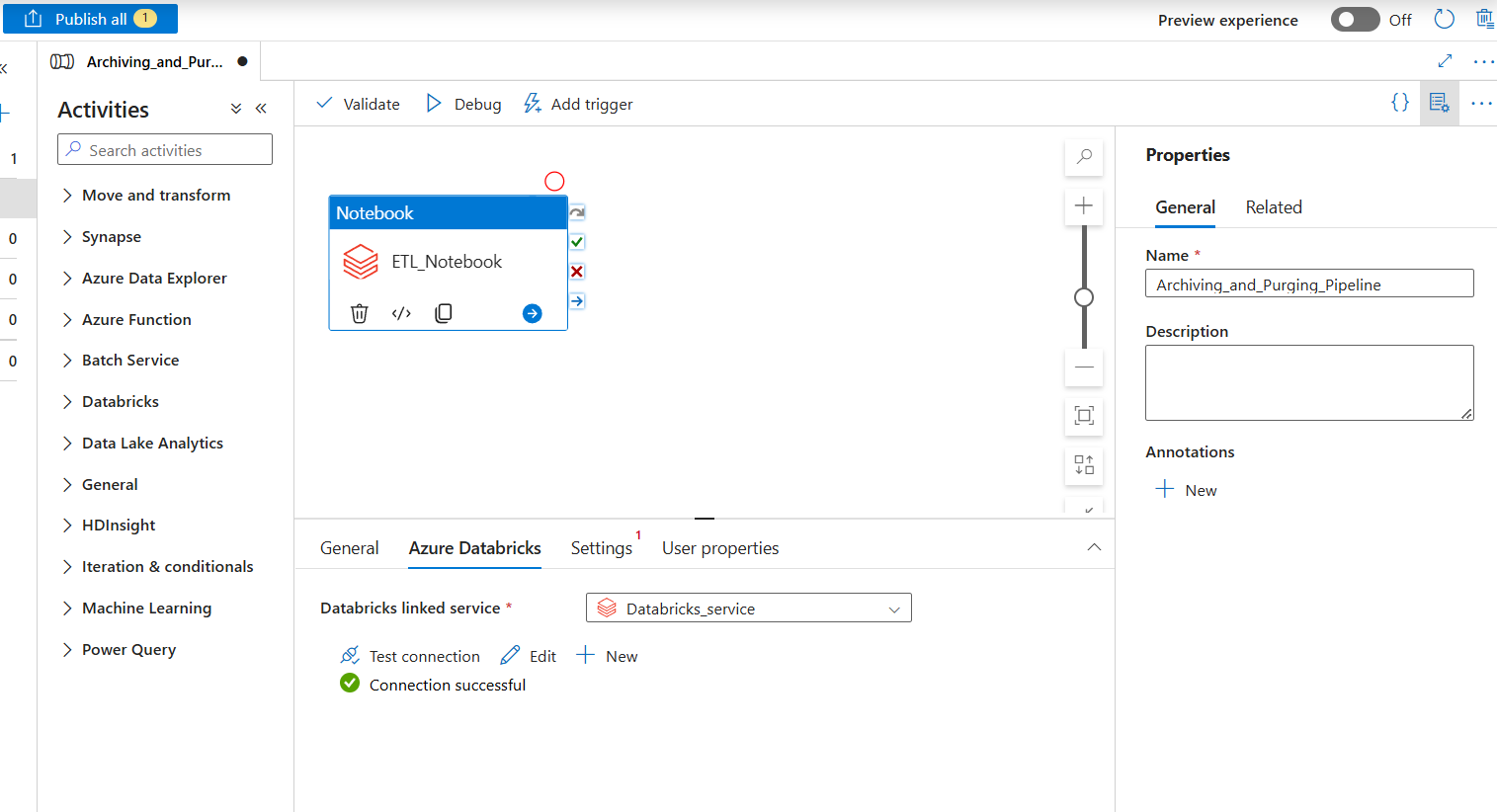


2)Creation of New Pipeline named ‘Archiving\_and\_Purging\_Pipeline’ in ADF.

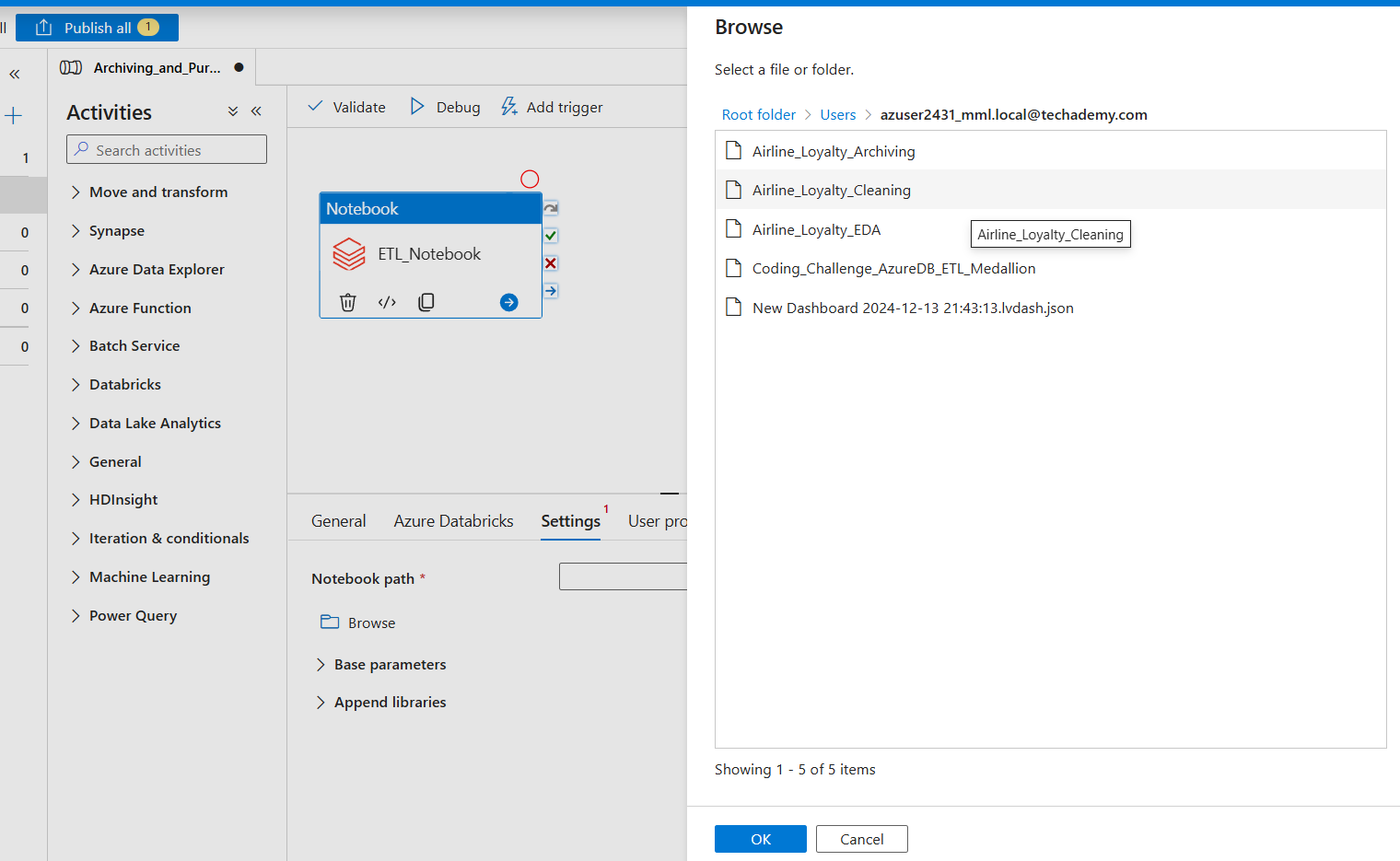
– Added New Databricks Notebook activity in the pipeline, named ETL\_Notebook.

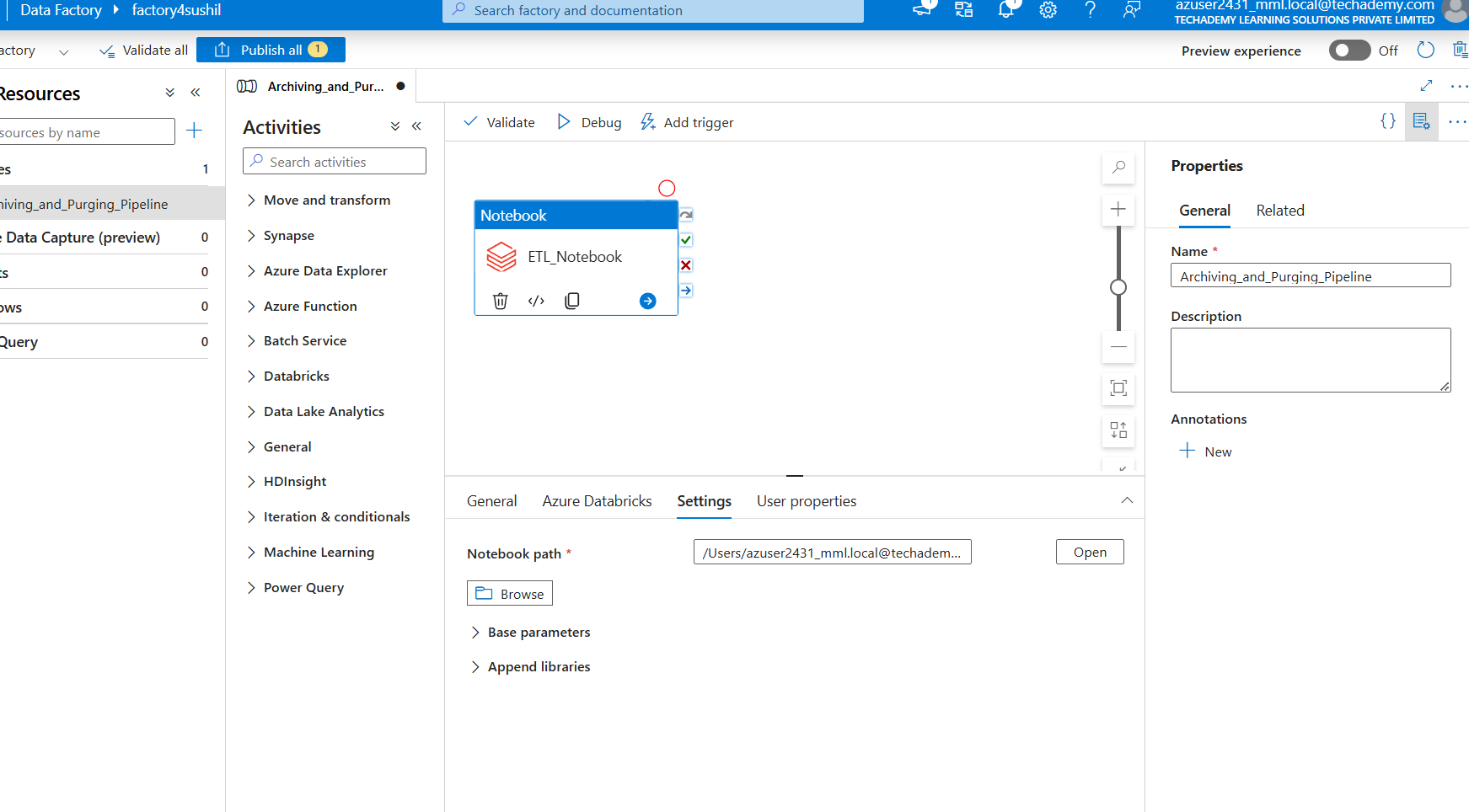


– Used already created linked service ‘Databricks\_service’ and tested connection with service.

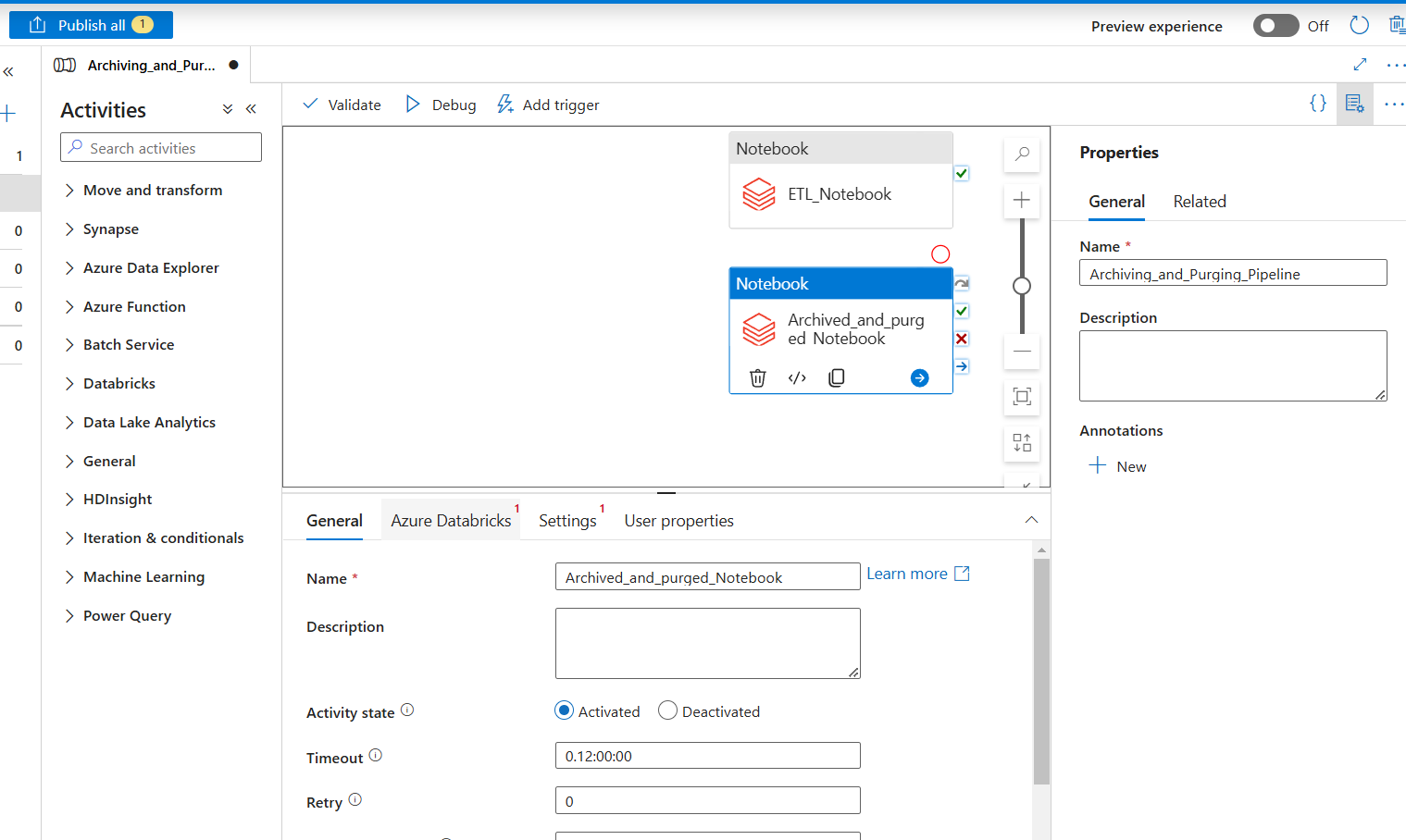


– In Settings gave Notebook path by browsing files to use (Linked service root→Users→username→Airline\_Loyalty\_Cleaning)

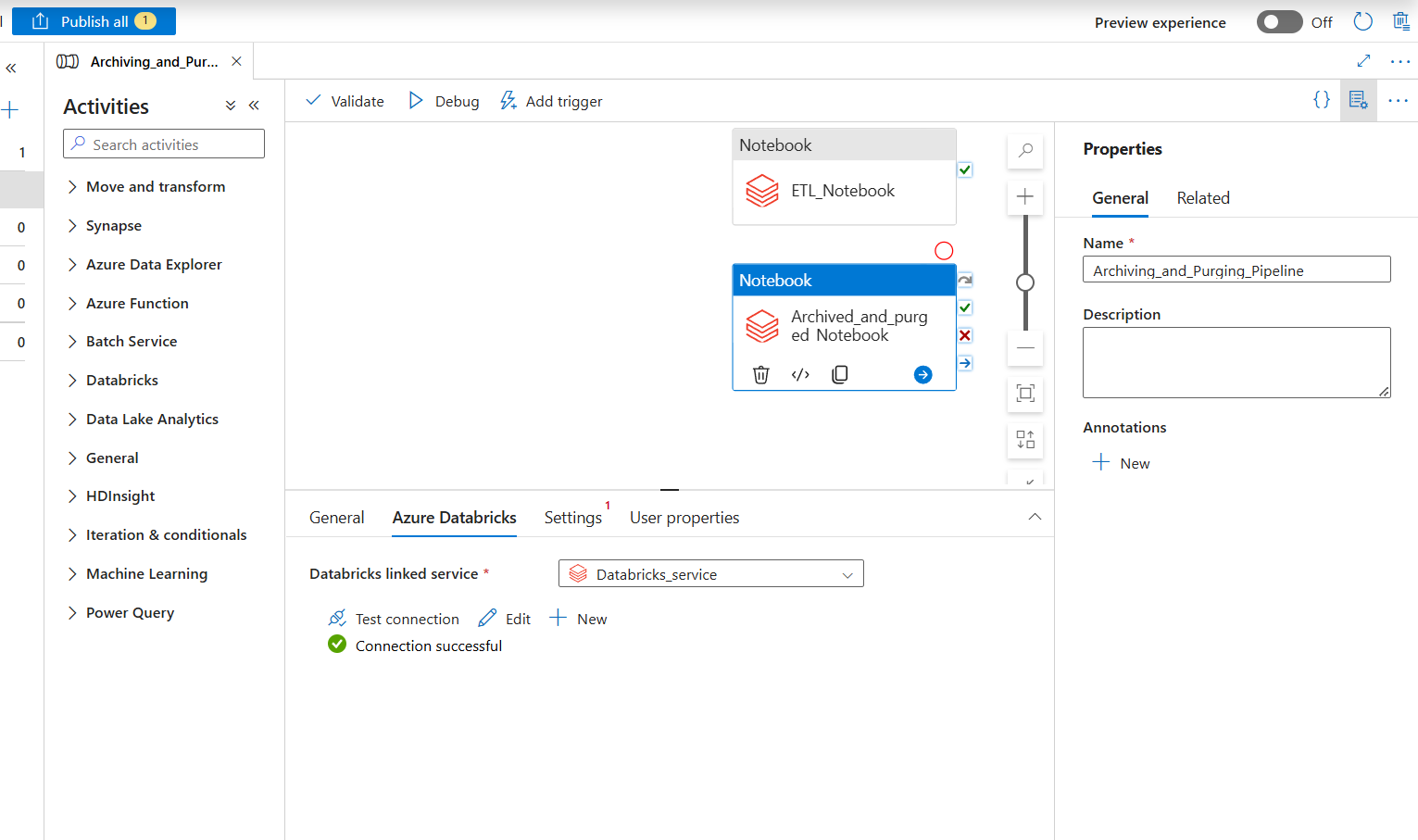




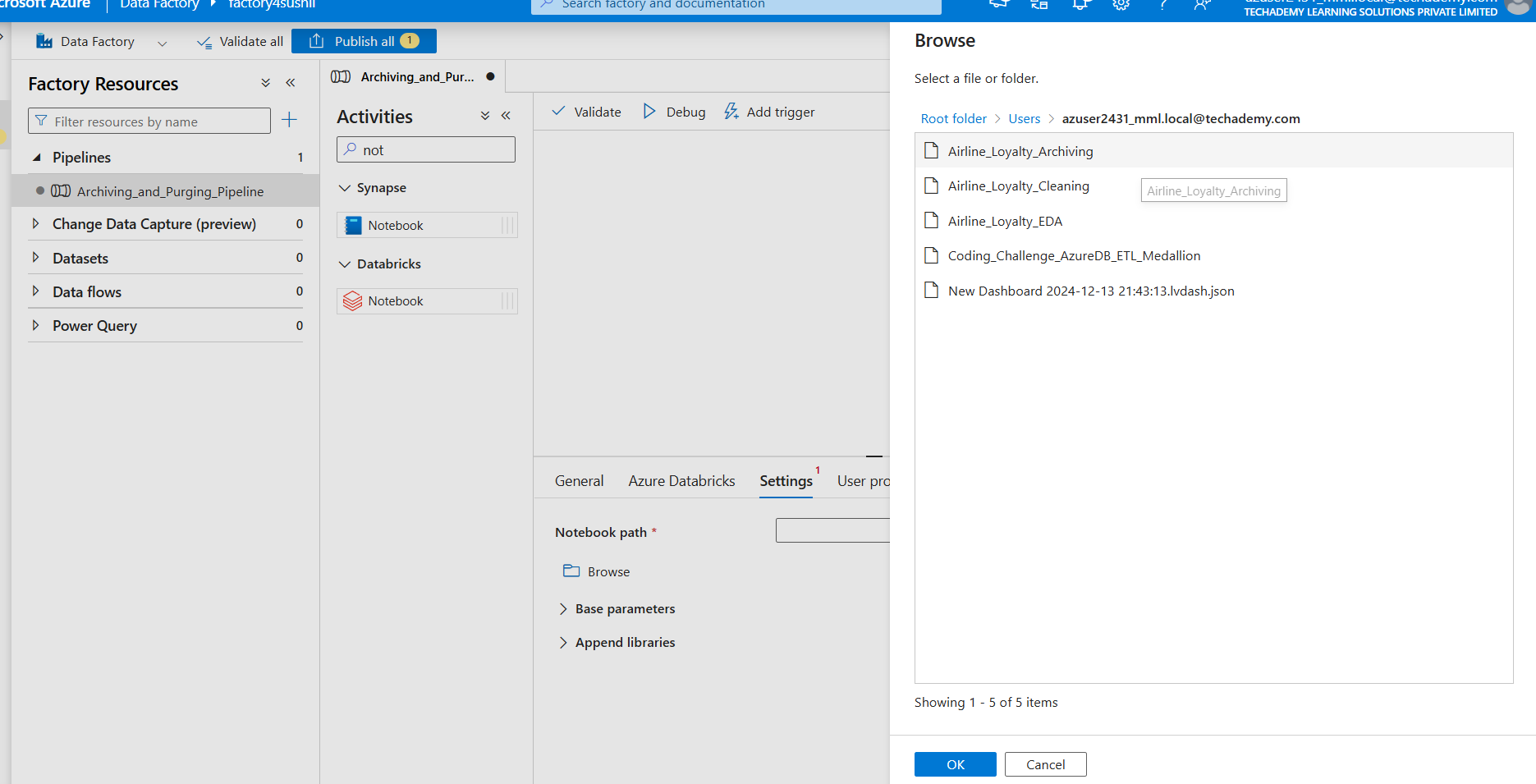
– Added New Databricks Notebook activity in the pipeline, named Archived\_and\_purged\_Notebook.



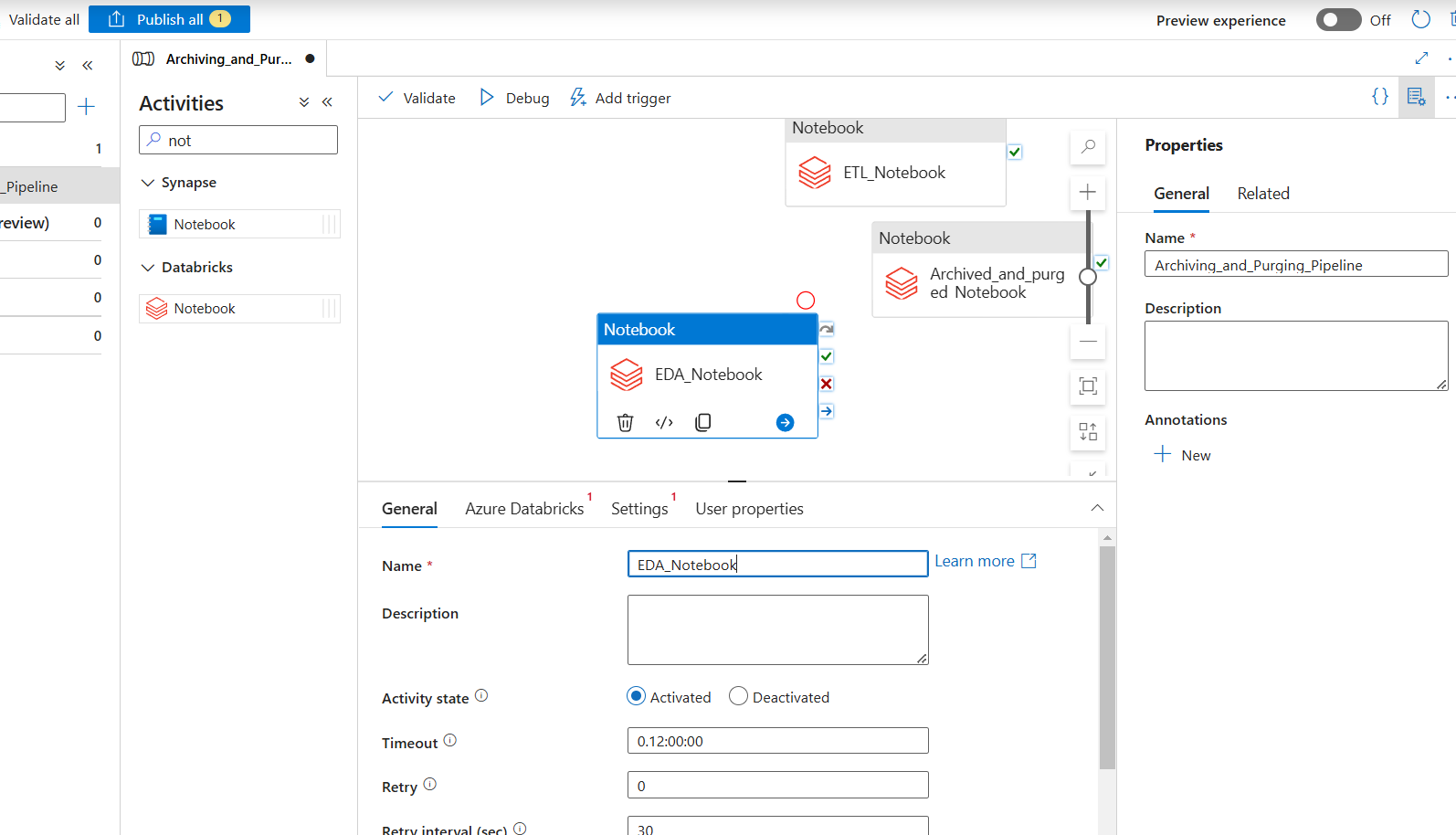
–Linked same Databricks linked service , checked connection.



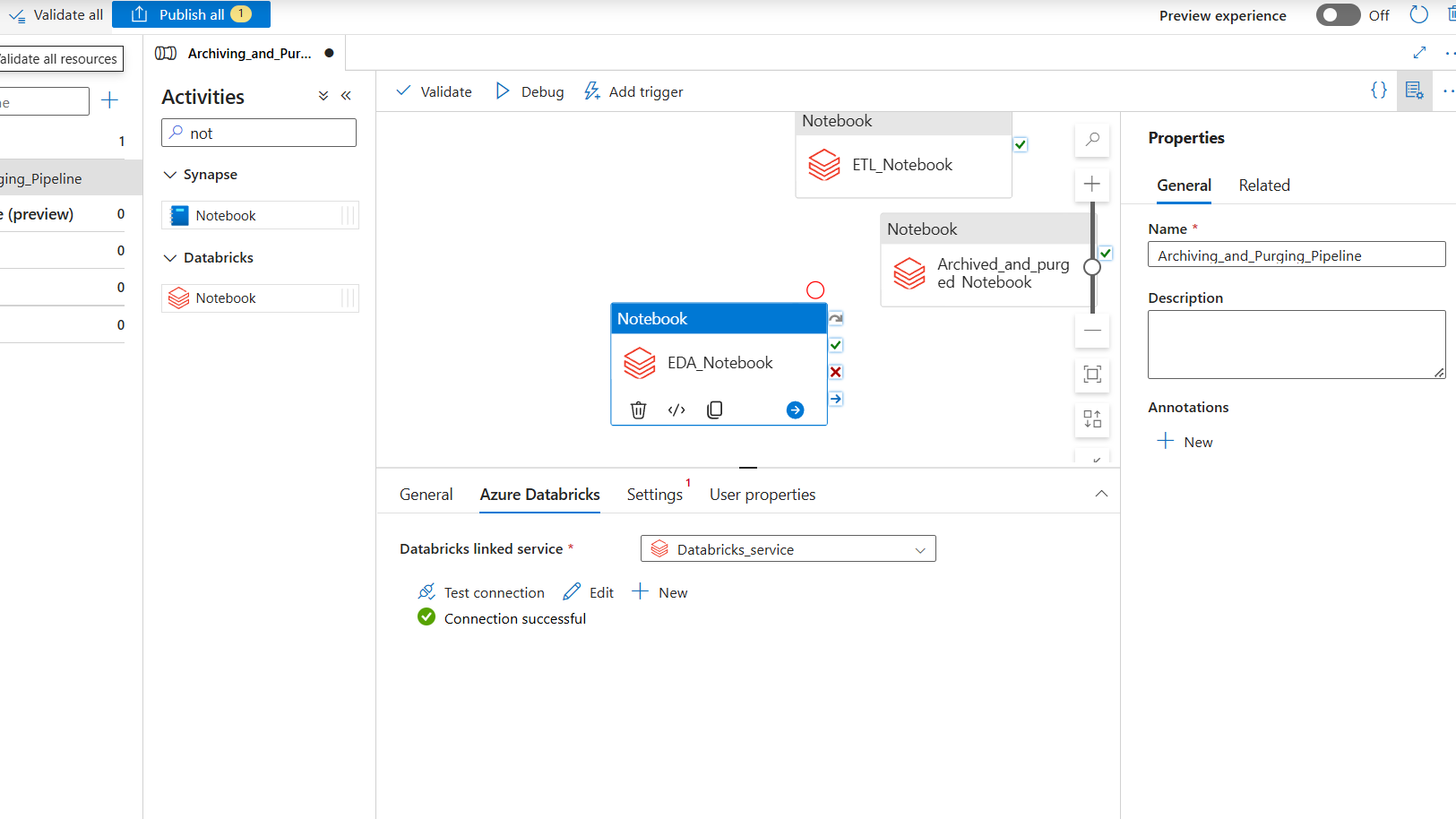
–Selected ‘Airline\_Loyalty\_Archiving’ Notebook path same as we did above.



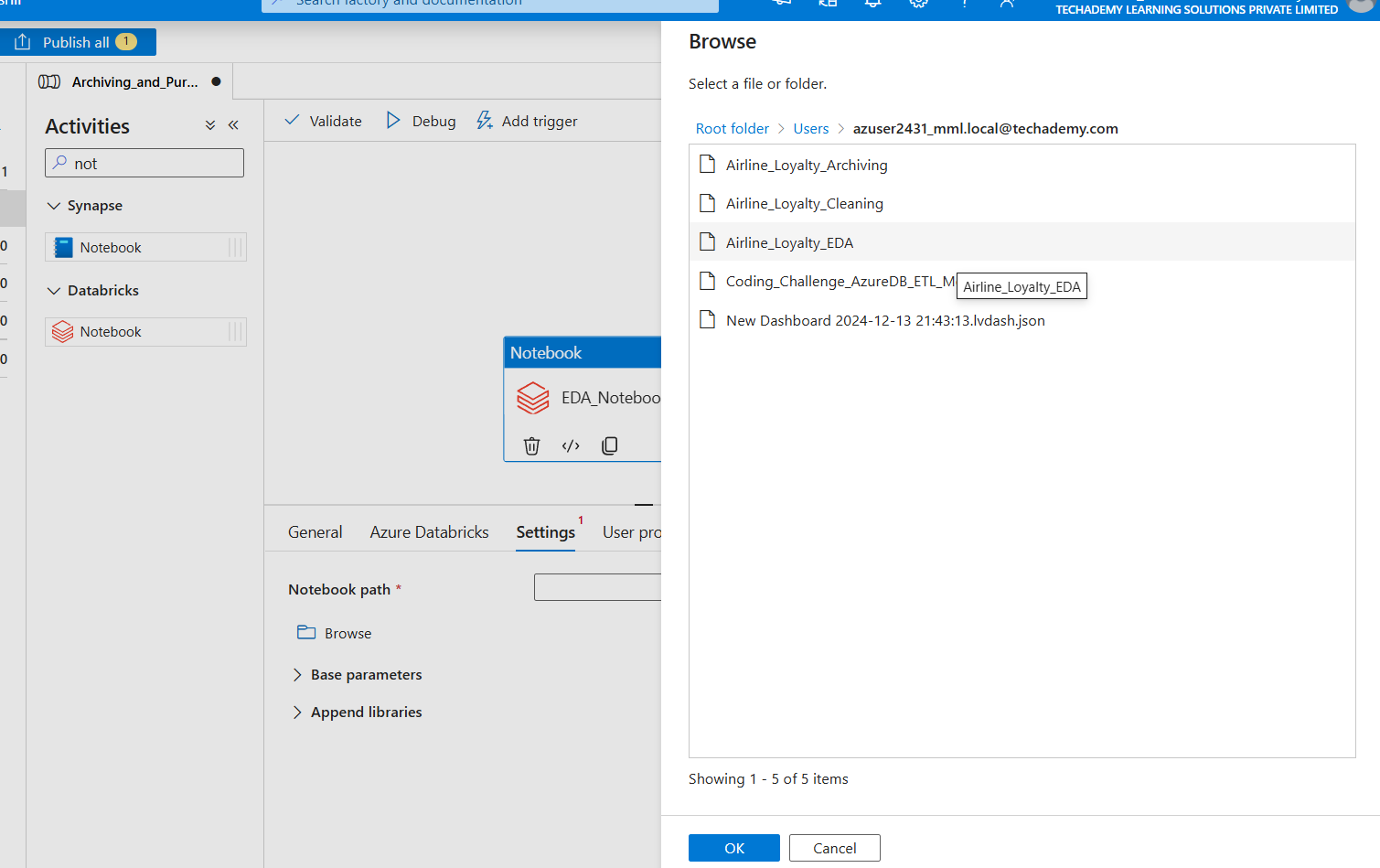
–Now our final 3rd Notebook we executed through the same pipeline. Created a new Activity named EDA\_Notebook.



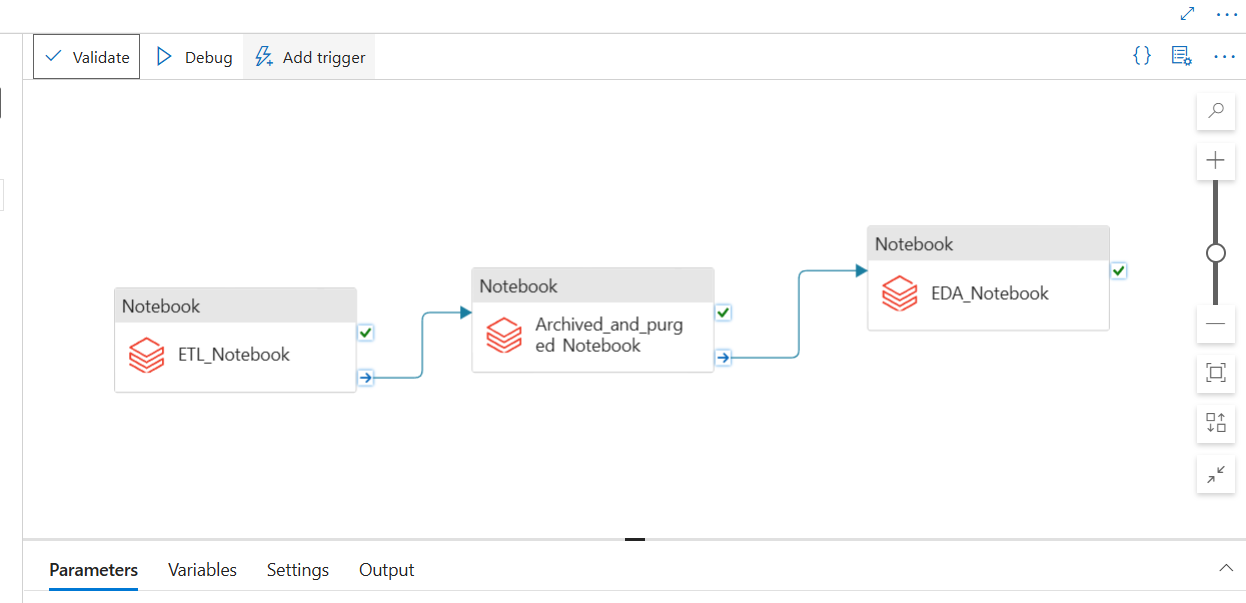
–Here also we used the same linked service (so we had created only one linked service for databricks). Each time we tested the connection .



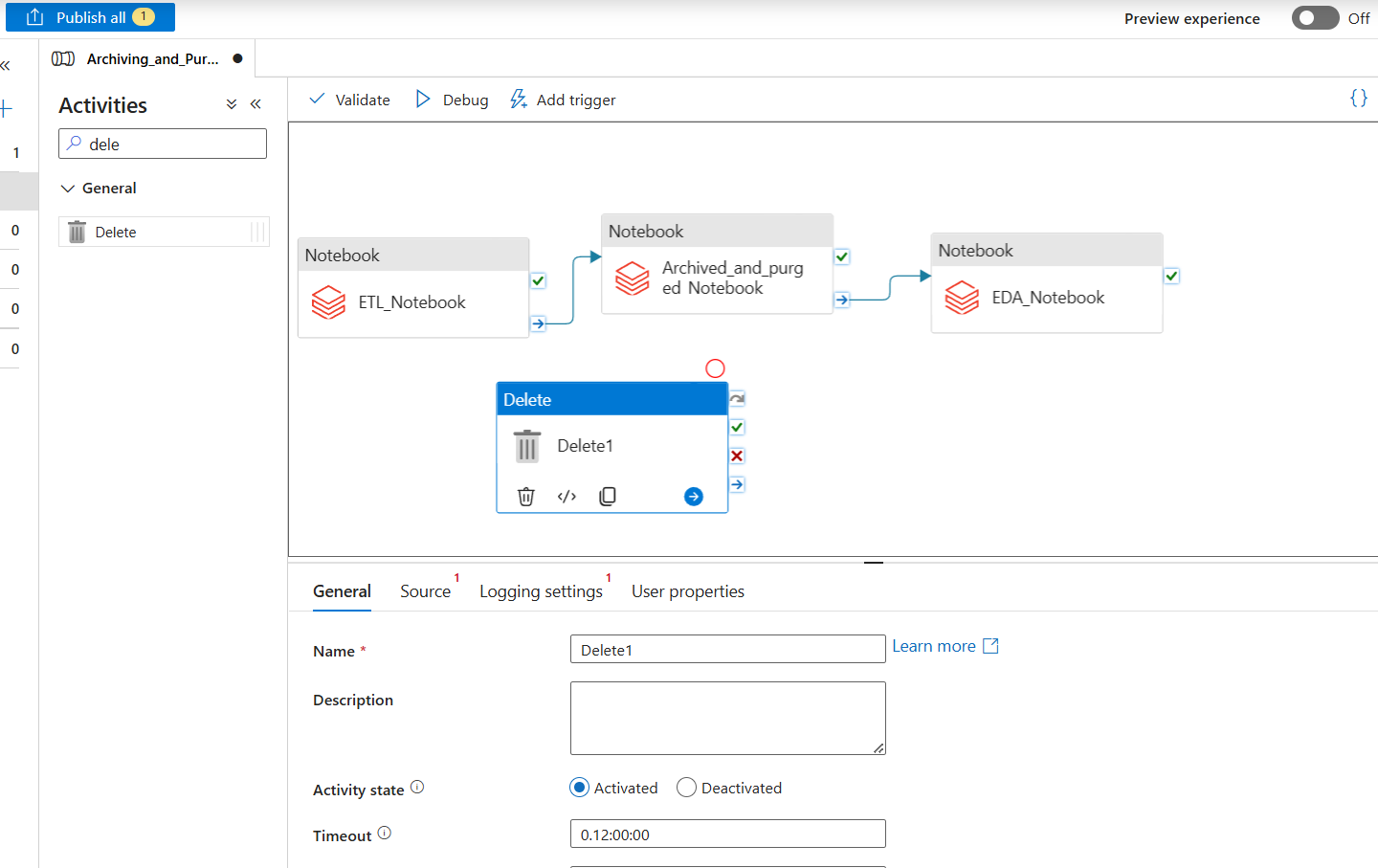
–Browsed Airline\_Loyalty\_EDA notebook path for activity.



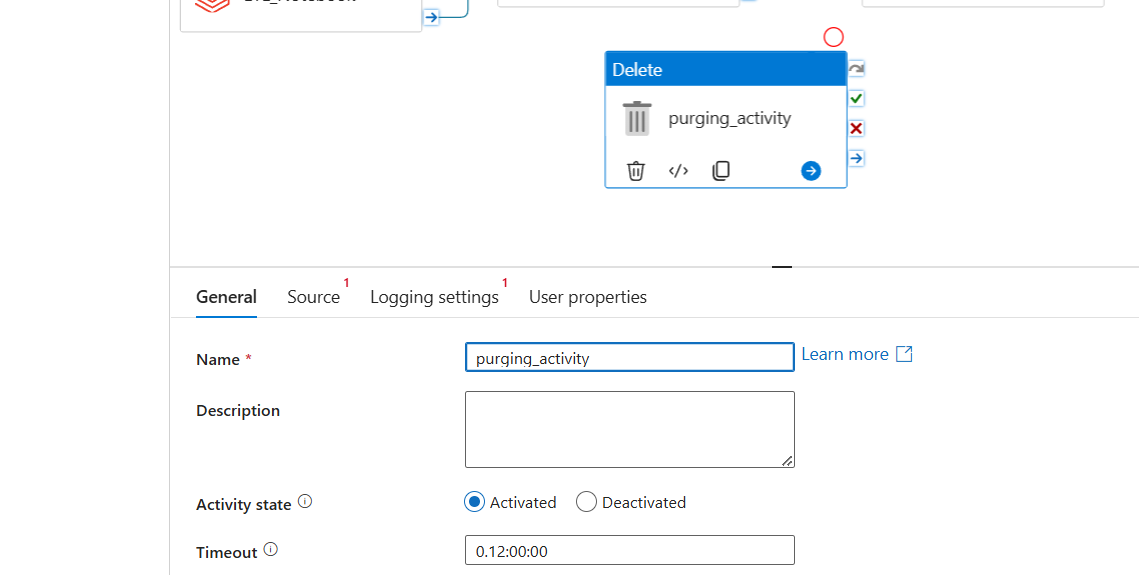
–Now we have all 3 Notebooks in the Pipeline connected according to sequence of execution.



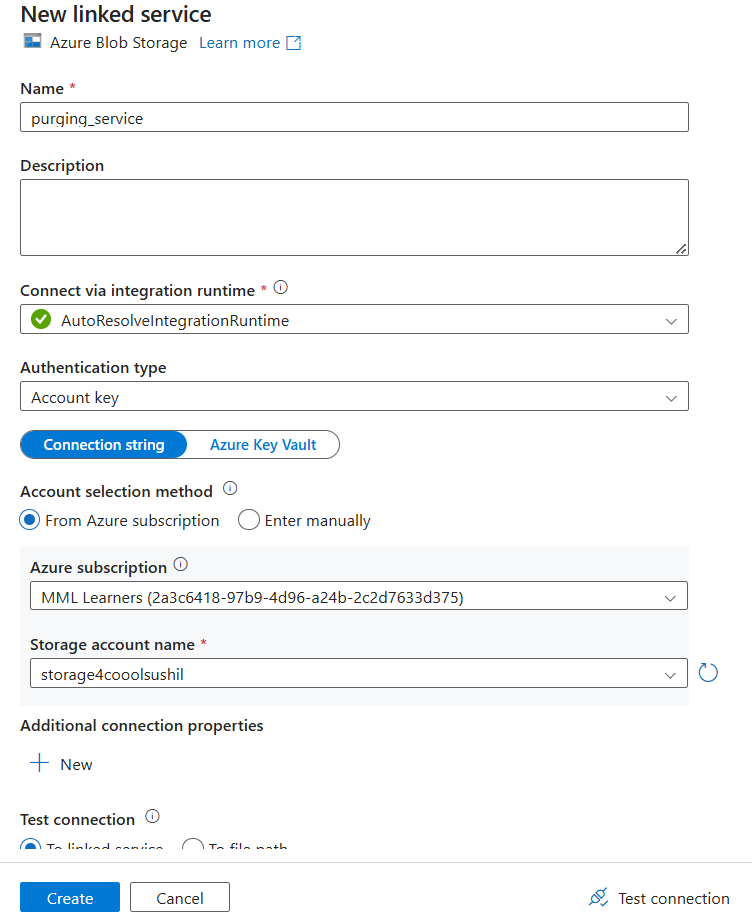
– We have a purged container where we stored data that needed to be purged using a purging process using a Linked Service deletion activity in Azure Data Factory. This activity identified and removed obsolete data (e.g., 2010–2012) from the processed-data container while ensuring compliance with traceability requirements.



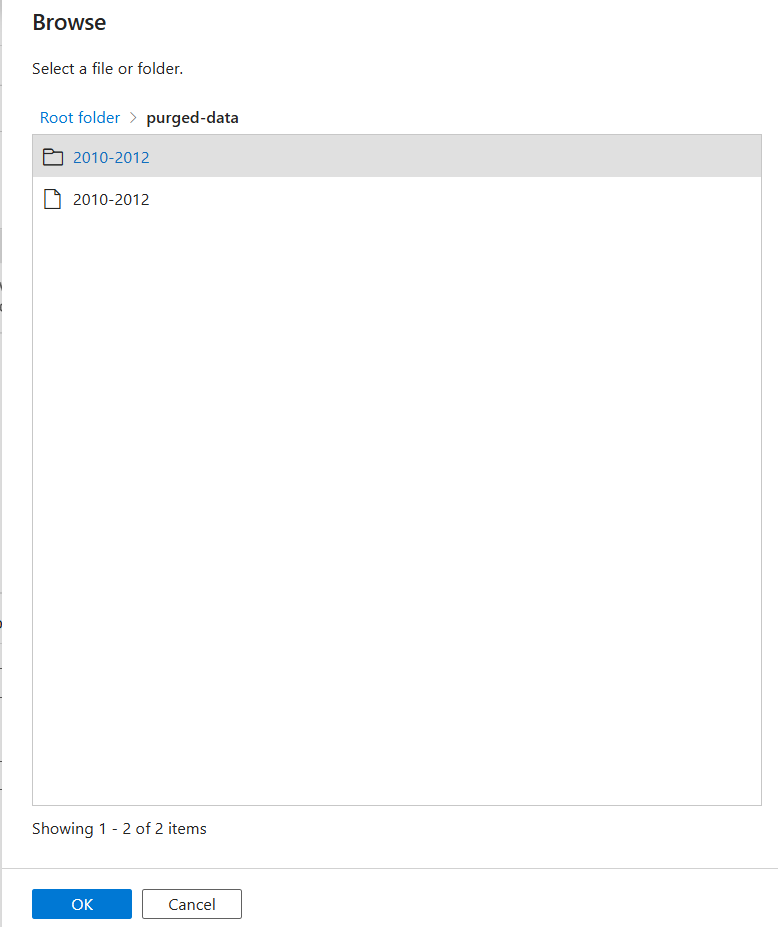
–Named delete activity as ‘purging\_activity’.



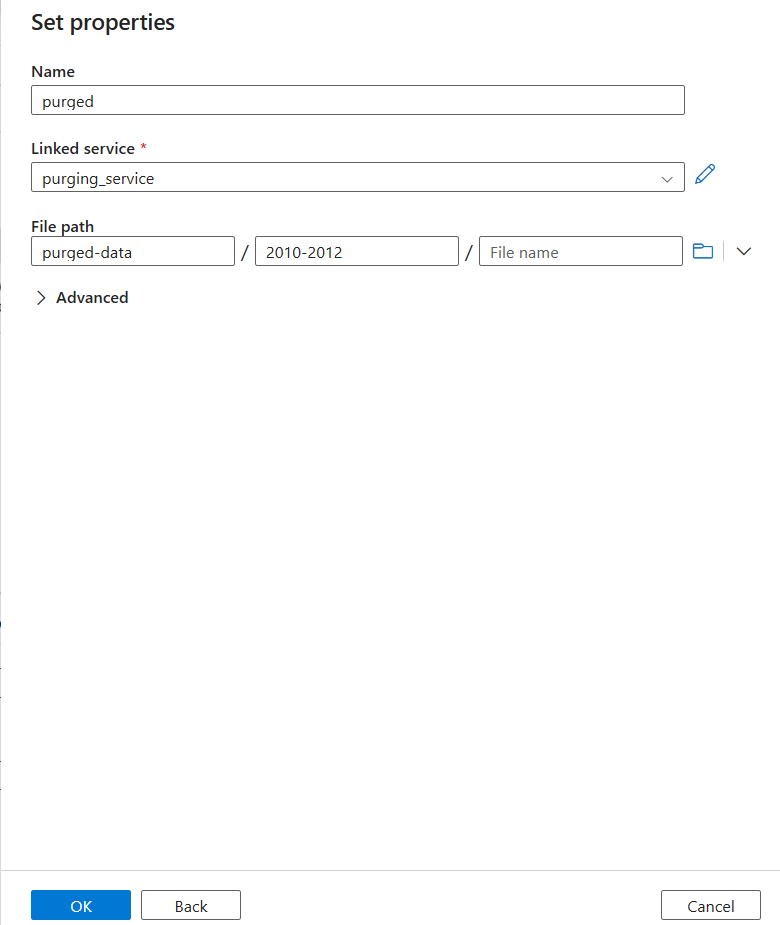
–Created Second linked service of Azure Blob Storage to have access to ADL storage to delete data from purged container. Named linked service ‘purging\_service’. Used storage account ‘storage4cooolsushil’ -Cool Tier ADL: Ideal for infrequently accessed data, providing a balance of cost and access latency ; where we have our purged container.



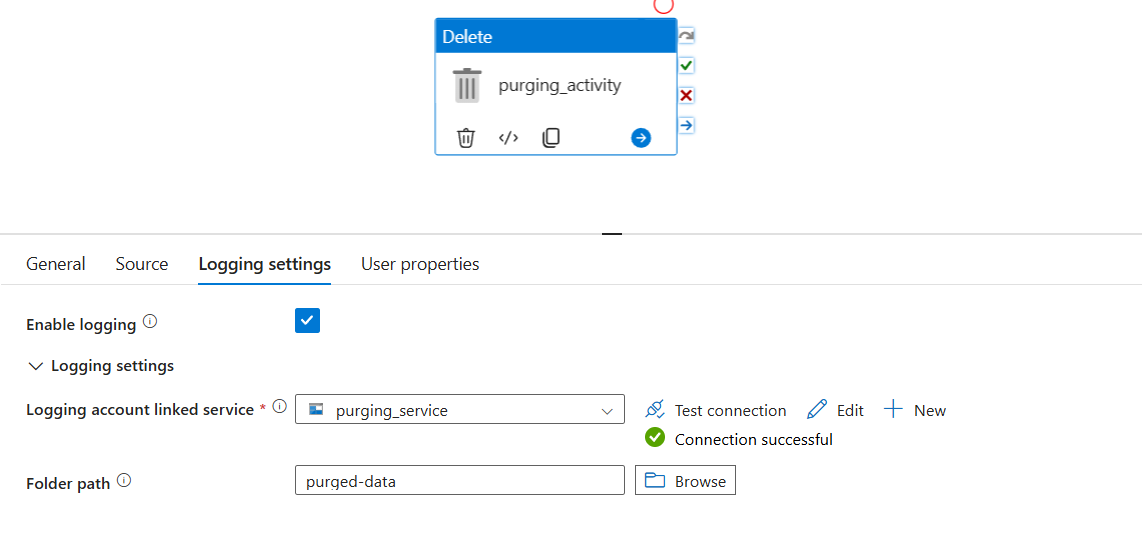
– Selected source for purging activity (ADL root→purged-data→2010-2012 file)



–We can see the path which we browsed and all our properties for purging\_activity.

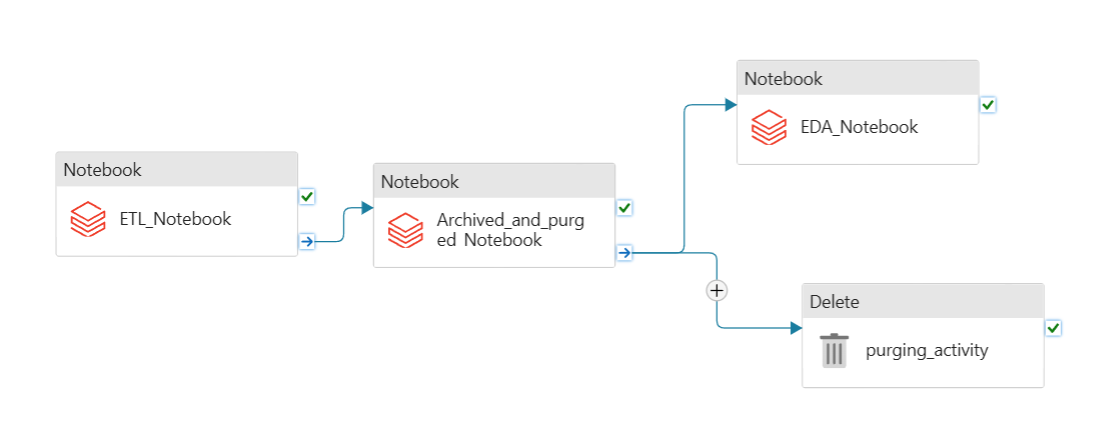


–We did a logging setting for this activity because we need to maintain purging (deletion) logs, that's why given logging file storage path within linked service.

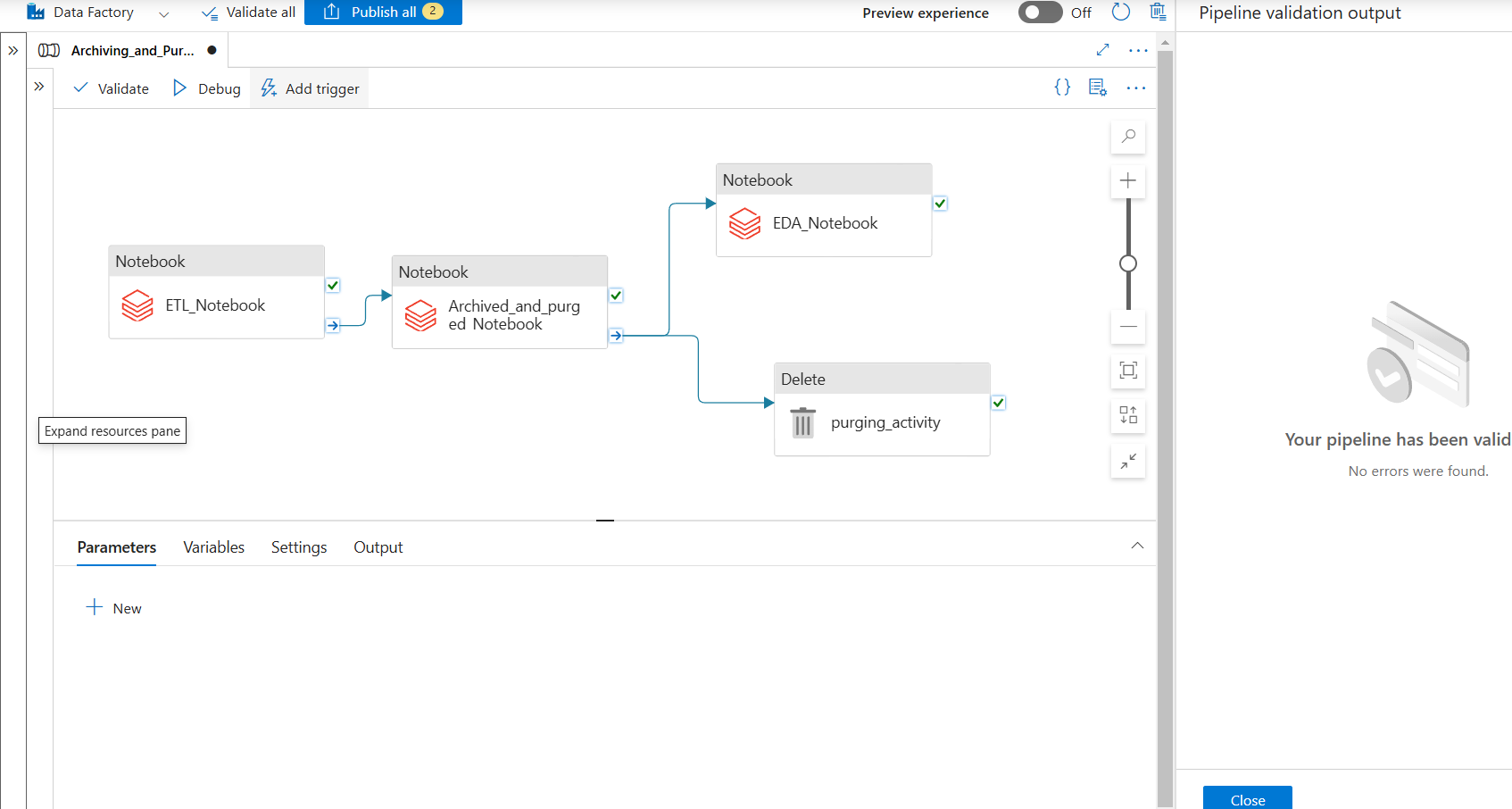


– Now we have constructed our pipeline workflow successfully and given a sequence of execution as per project requirement.

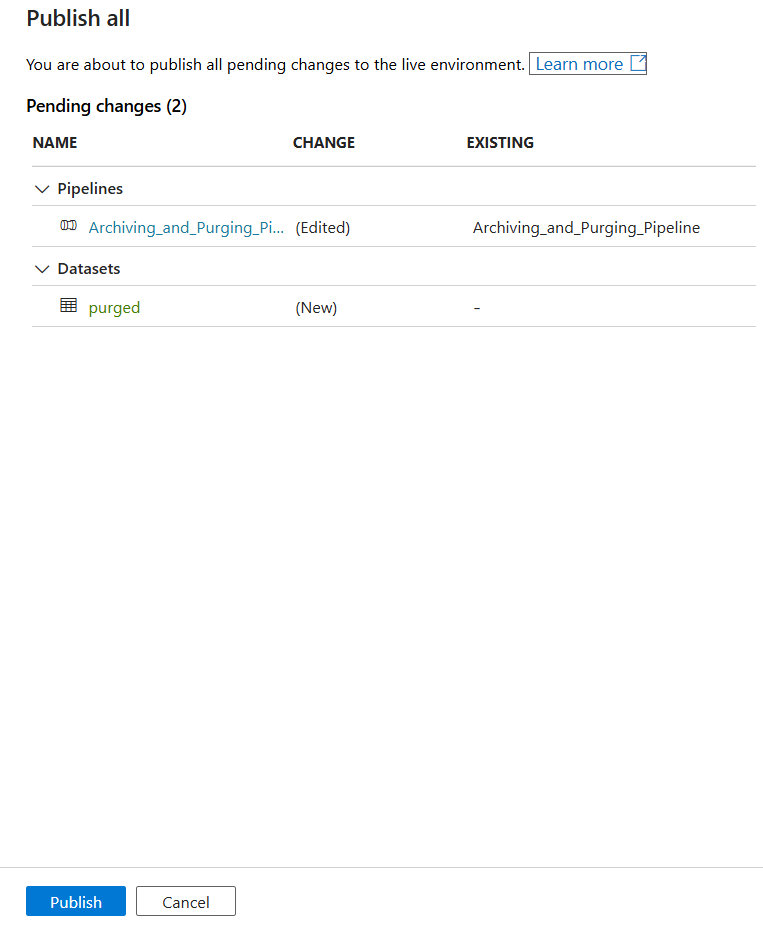
We first Extract and Clean data using **ETL\_Notebook** then we divided data and stored it in archived and purged container using **Archived\_and\_purged Notebook** after we got relevant data we did EDA on active data i.e(2018-2020) using **EDA\_Notebook**, parallely we purged(deleted) old/irrelevant data from purged container while maintaining logs using **purging\_activity**.



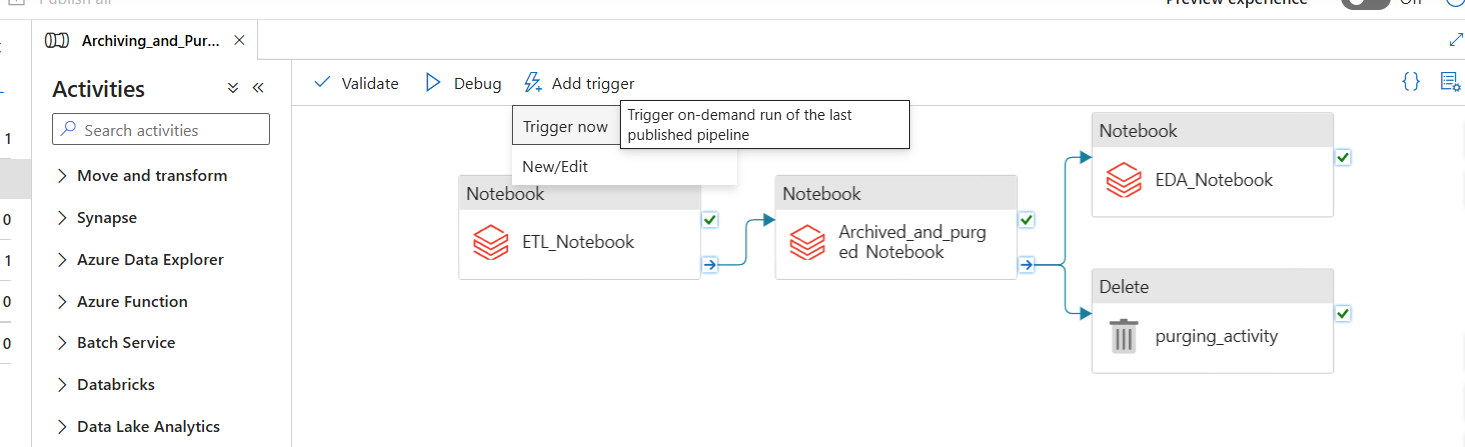
–Validated the pipeline.



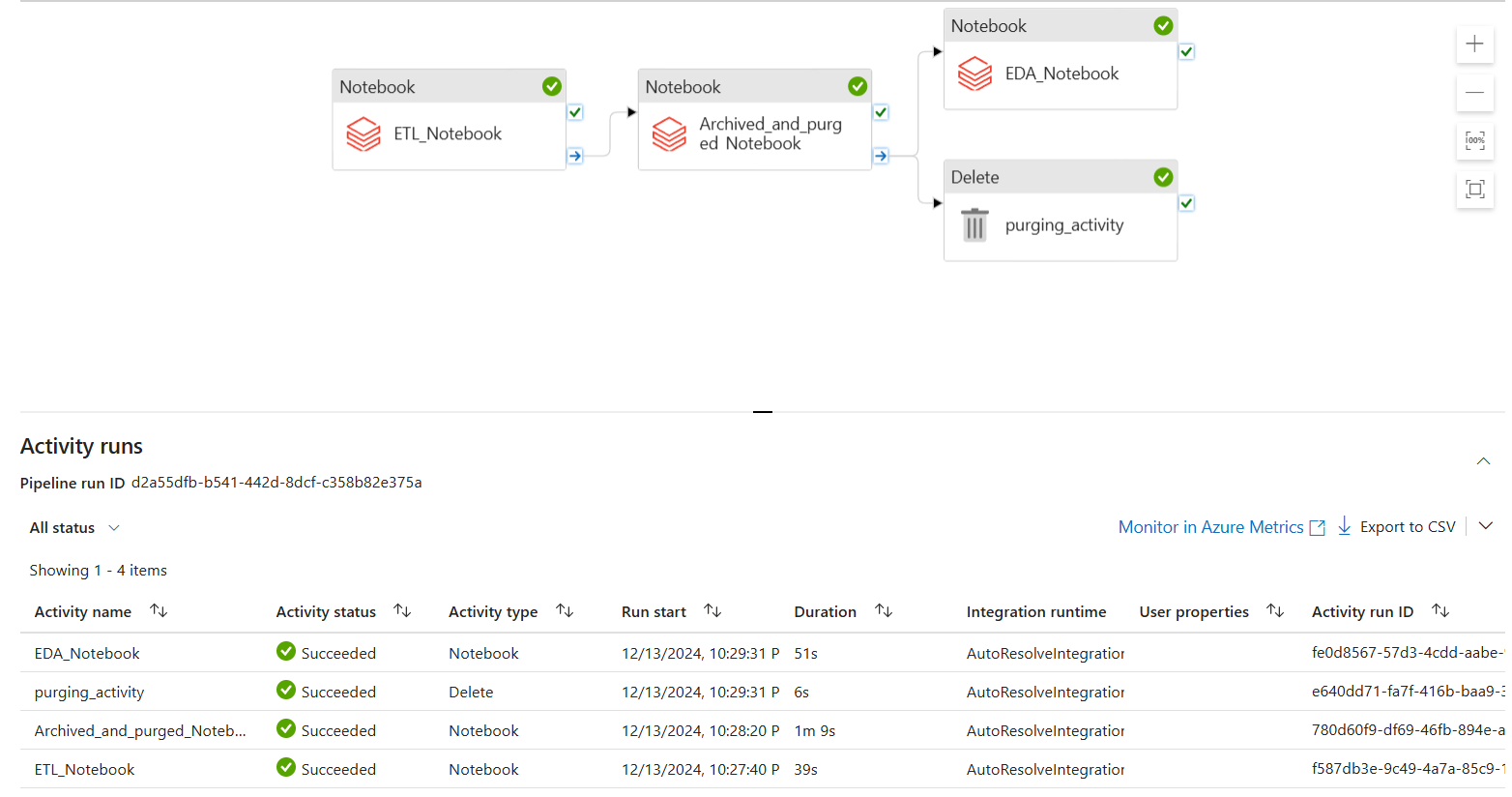
–Published all changes we made in Pipeline.



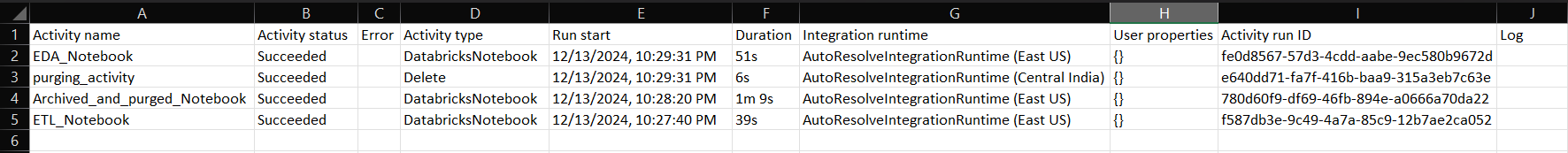
–Manually Triggered the pipeline using Trigger now in Add trigger drop down.



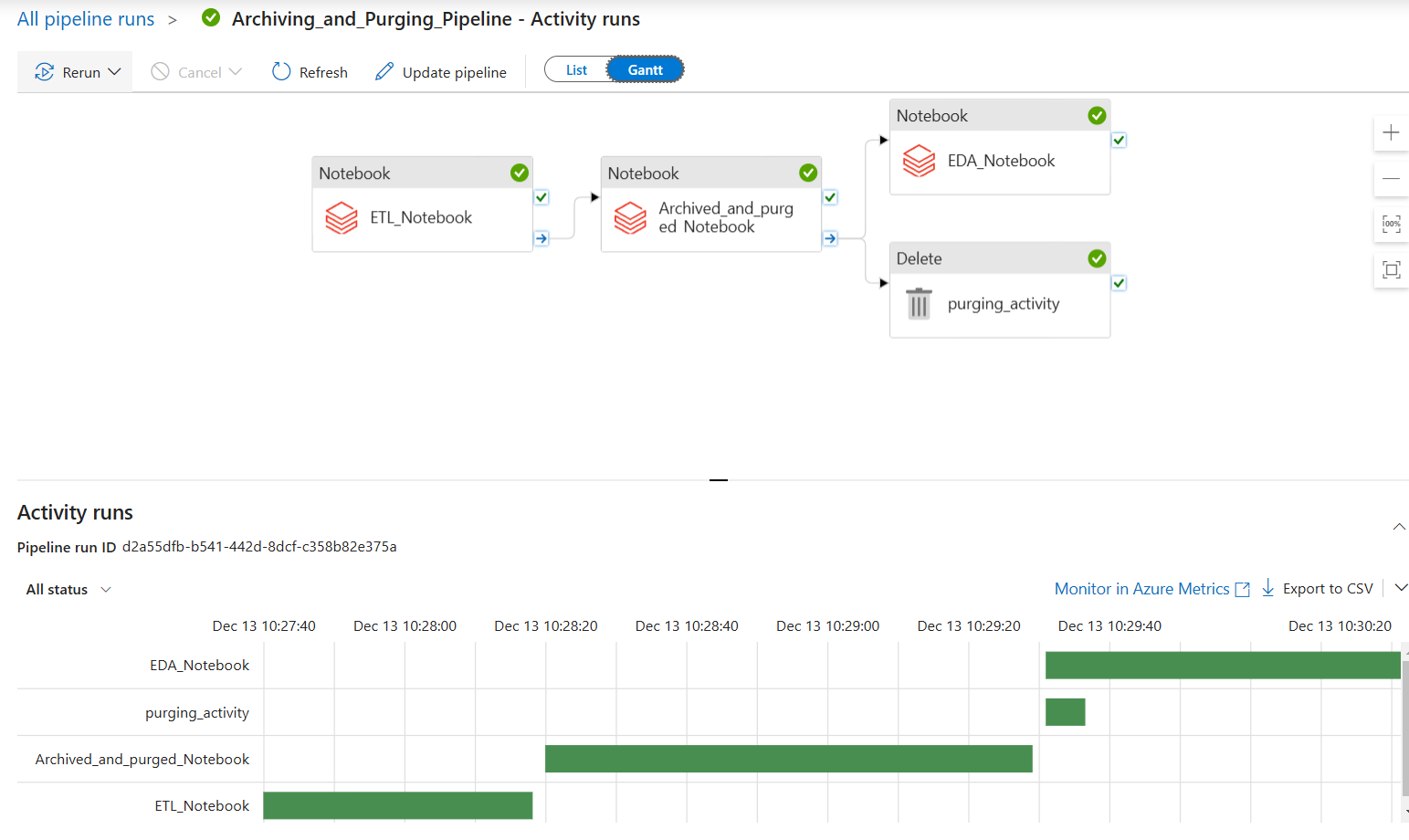
– We can see running pipelines in the Monitor section, All activity runs and their status.



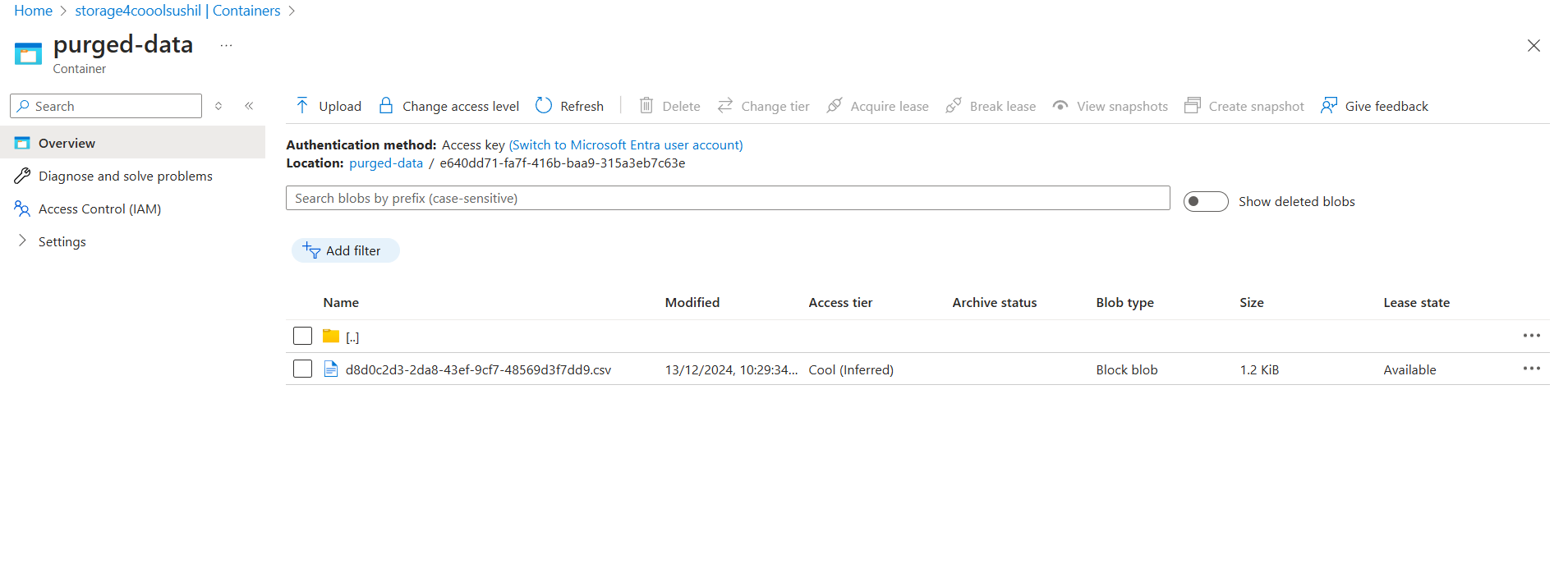
–After downloading Activity runs CSV we can see logs of our pipeline execution and everything we did using automation.



–We also saw a Gantt chart that shows the time duration of each activity execution.



– Here now in purged data existing purged files deleted and new log file created in CSV format.



– We can download a log file and see information about the purging activity we did earlier.

