**Project Overview:** Mapping Data Flow with Azure Data Factory Activities using ADF Pipeline

**Project Objective:** The objective of this project is to demonstrate the use of Mapping Data Flow within Azure Data Factory pipelines for efficient data transformation tasks. By leveraging Mapping Data Flow activities, the project aims to showcase how to design, develop, and deploy data transformation processes using a visual interface and orchestrate them within ADF pipelines.

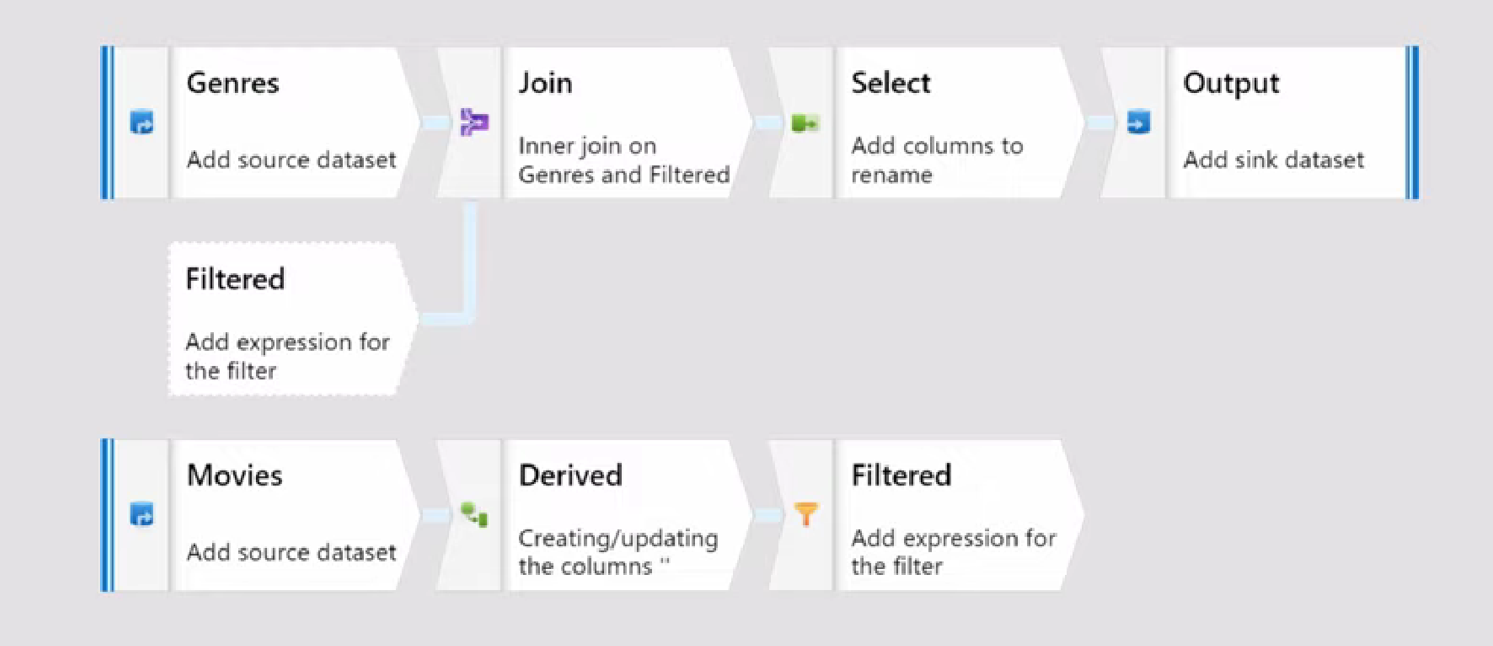
**Introduction:**

In the realm of data engineering and data processing, efficient management, transformation, and movement of data are essential. Azure Data Factory (ADF) is a cloud-based data integration service that allows you to create, schedule, and orchestrate data pipelines for ingesting, preparing, transforming, and analyzing data. Mapping Data Flow is a feature within Azure Data Factory that provides a visually interactive data transformation experience, enabling users to design, debug, and execute data transformation processes without writing code.

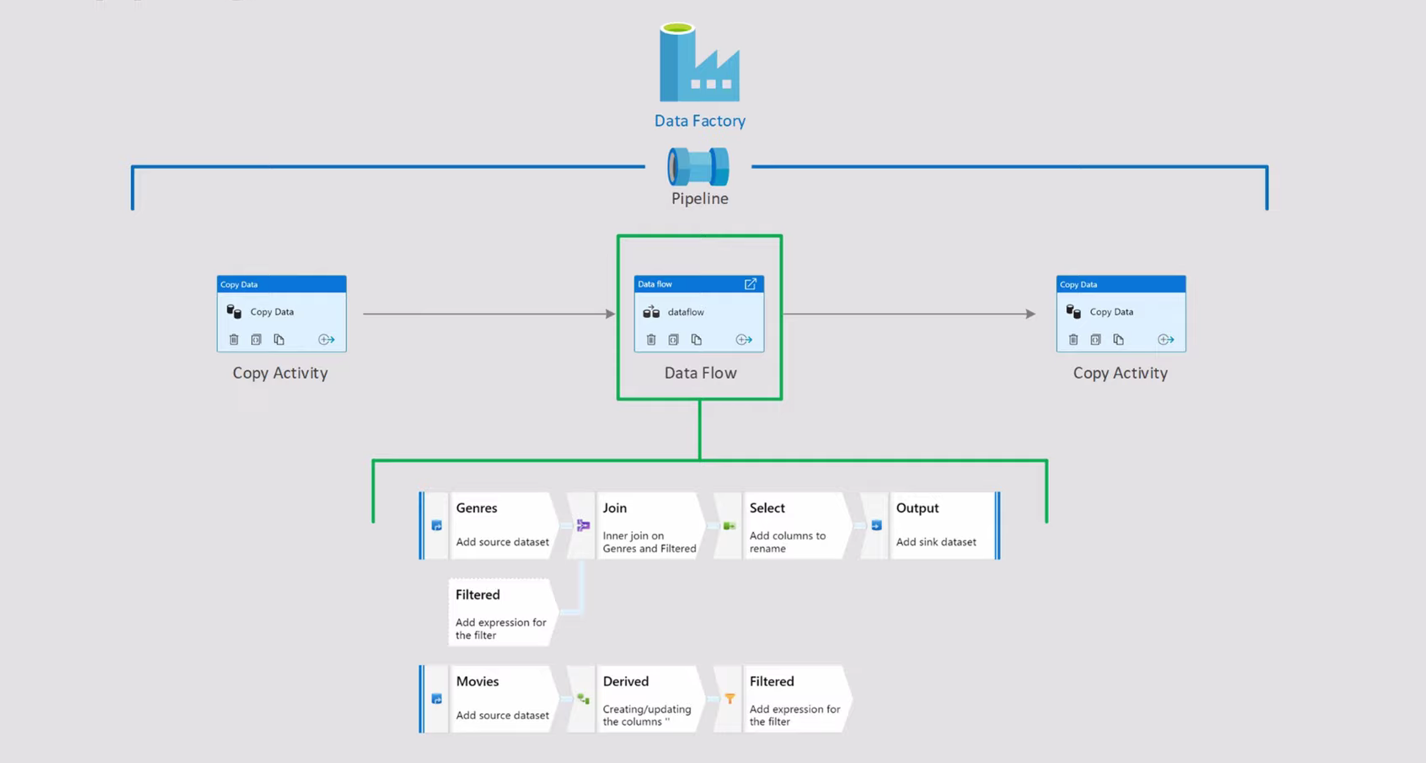
**Project Components and Workflow:**

1. **Data Source Definition:**
   * Define the data sources from which data will be ingested and transformed. These sources can include various Azure services like Azure SQL Database, Azure Blob Storage, Azure Data Lake Storage, or external sources like SQL Server, Oracle, etc.
2. **Mapping Data Flow Design:**
   * Utilize the Mapping Data Flow feature within Azure Data Factory to visually design the data transformation logic.
   * Define data transformations such as filtering, aggregations, joins, data cleansing, and schema transformations using a drag-and-drop interface provided by Mapping Data Flow.
3. **Data Transformation Logic:**
   * Implement the necessary data transformation logic within Mapping Data Flow activities to meet the business requirements and data processing needs.
   * Ensure efficient data processing by optimizing transformations and considering performance considerations.
4. **Error Handling and Data Quality Checks:**
   * Implement error handling mechanisms within the Mapping Data Flow to handle data quality issues, errors, and exceptions gracefully.
   * Incorporate data quality checks and validations to ensure the integrity and accuracy of transformed data.
5. **Integration with ADF Pipeline:**
   * Integrate the Mapping Data Flow activities within Azure Data Factory pipelines.
   * Define the sequence of activities, dependencies, and triggers within the ADF pipeline to orchestrate the data transformation process.
6. **Execution and Monitoring:**
   * Execute the ADF pipeline to trigger the data transformation process.
   * Monitor the execution progress, track data flow, and analyze performance metrics using Azure Data Factory monitoring capabilities.
7. **Deployment and Automation:**
   * Automate the deployment of ADF pipelines and Mapping Data Flow configurations using Azure DevOps or Azure Resource Manager templates.
   * Implement continuous integration and continuous deployment (CI/CD) pipelines to streamline the deployment process and ensure consistency across environments.

**Architectural Diagram:**

****

*Fig 1 shows the mapping data flow*



*Fig 2 it also shows the mapping of data flow*



*Fig 3 it shows the supported data sources*

**Technologies and Tools Used:**

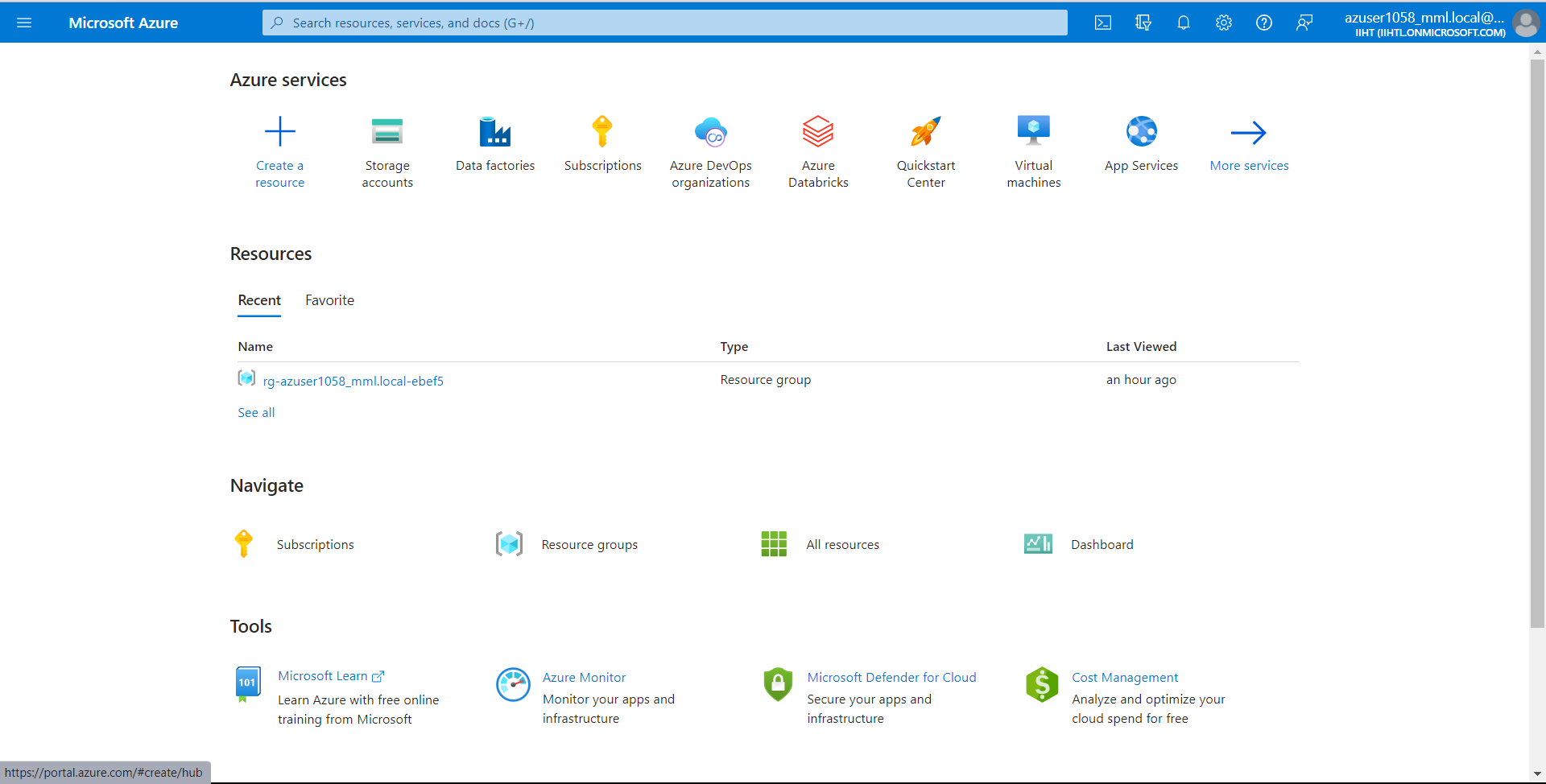
1. **Azure Data Factory (ADF):**
   * A cloud-based data integration service provided by Microsoft Azure for building, scheduling, and orchestrating data pipelines.
2. **Mapping Data Flow:**
   * A feature within Azure Data Factory that provides a visual interface for designing and executing data transformation processes without writing code.
3. **Azure Storage Services:**
   * Azure Blob Storage: A scalable object storage service for storing large amounts of unstructured data.
   * Azure Data Lake Storage: A scalable and secure data lake service for big data analytics.
4. **Azure SQL Database:**
   * A fully managed relational database service provided by Azure for building data-driven applications.
5. **Data Integration and ETL Tools:**
   * Various data integration and ETL (Extract, Transform, Load) tools may be used for data movement and transformation tasks as per project requirements.
6. **Azure DevOps:**
   * A suite of cloud-based collaboration tools provided by Azure for software development, including source control, build automation, and release management.
7. **Monitoring and Logging Tools:**
   * Azure Monitor: Provides comprehensive monitoring and logging capabilities for Azure services and applications.
   * Azure Log Analytics: A service in Azure that collects and analyzes data generated by resources in a cloud and on-premises environments.
8. **Azure Resource Manager (ARM) Templates:**
   * JSON templates used for deploying, managing, and configuring Azure resources in a repeatable and consistent manner.

By leveraging these technologies and tools, the project aims to achieve efficient data transformation, pipeline orchestration, monitoring, and deployment within the Azure environment.

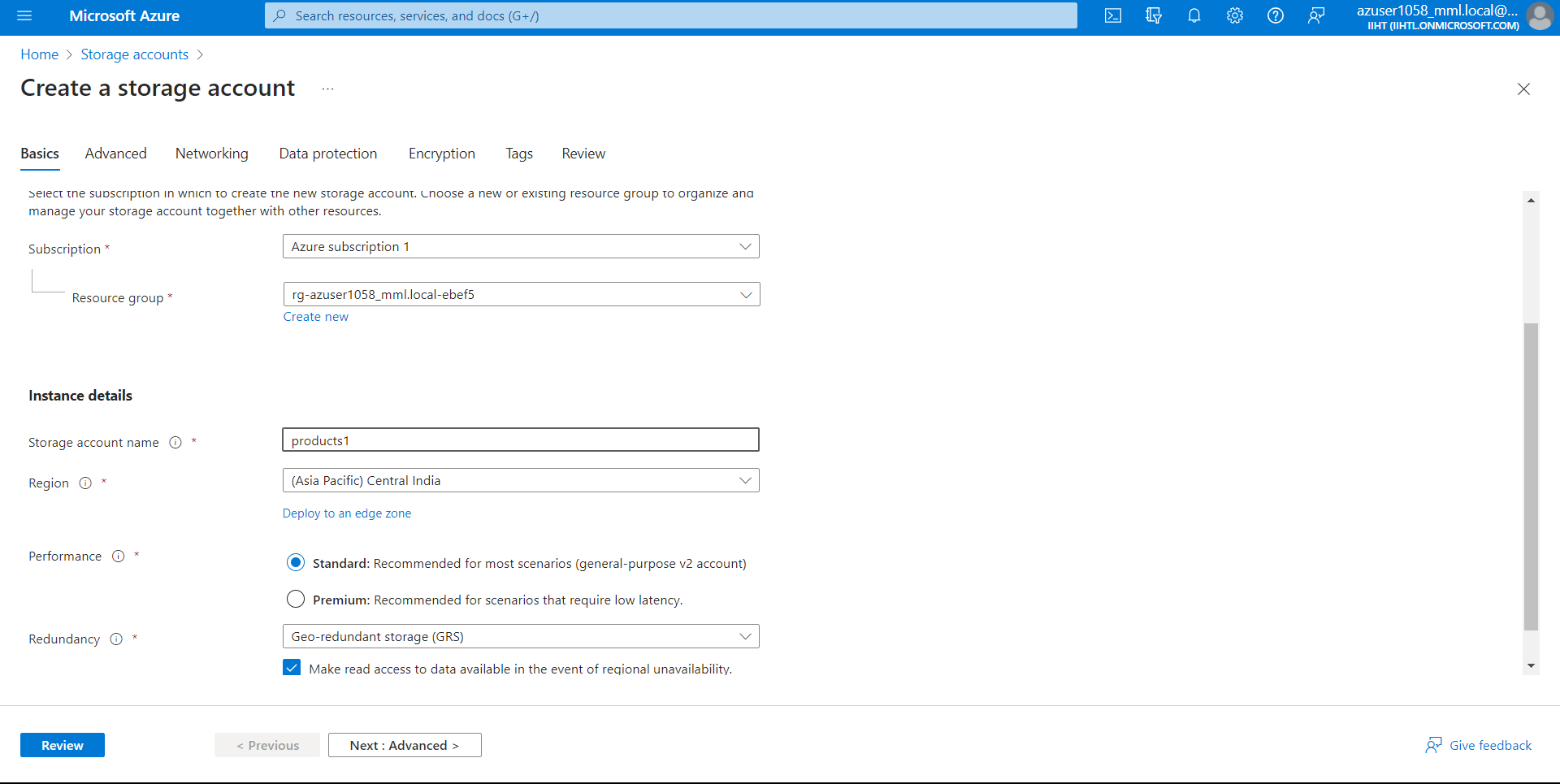
**Tasks performed:**

**Output/Result:**

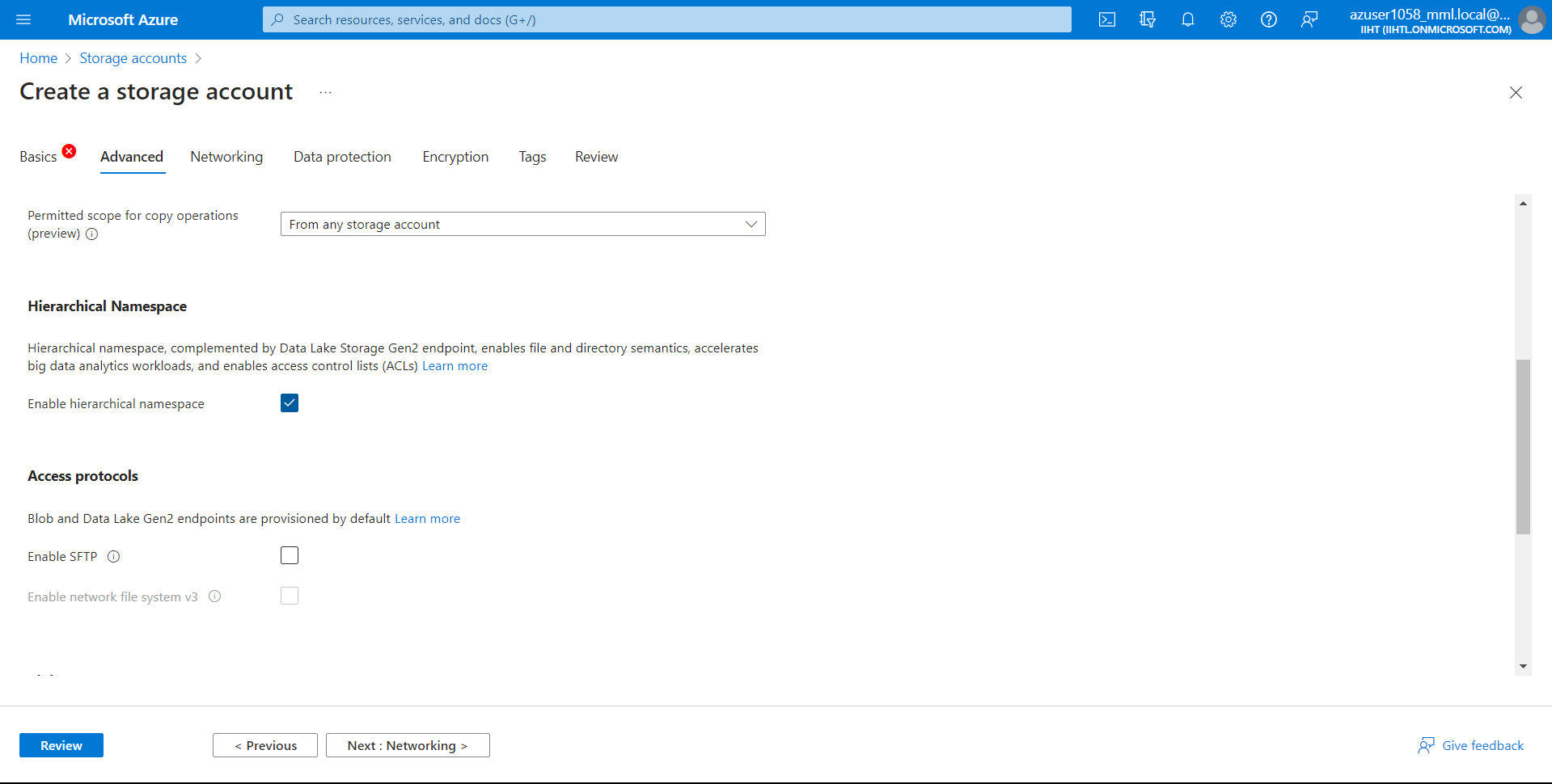
Firstly, Go to Microsoft Azure Portal.

****

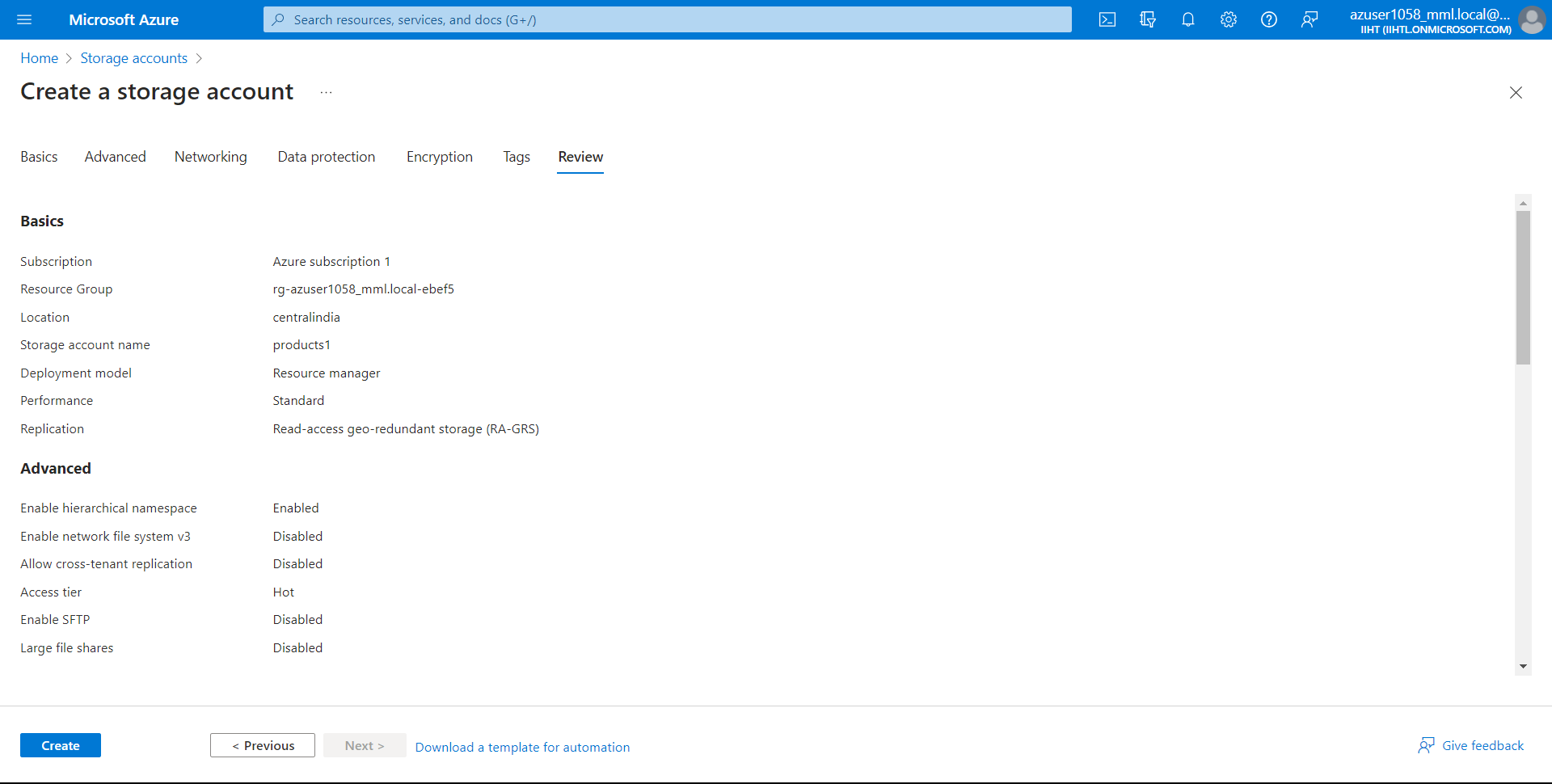
Now lets create a storage account from portal, enter your resource group name and select any region. We have given storage account name as **“products1”**



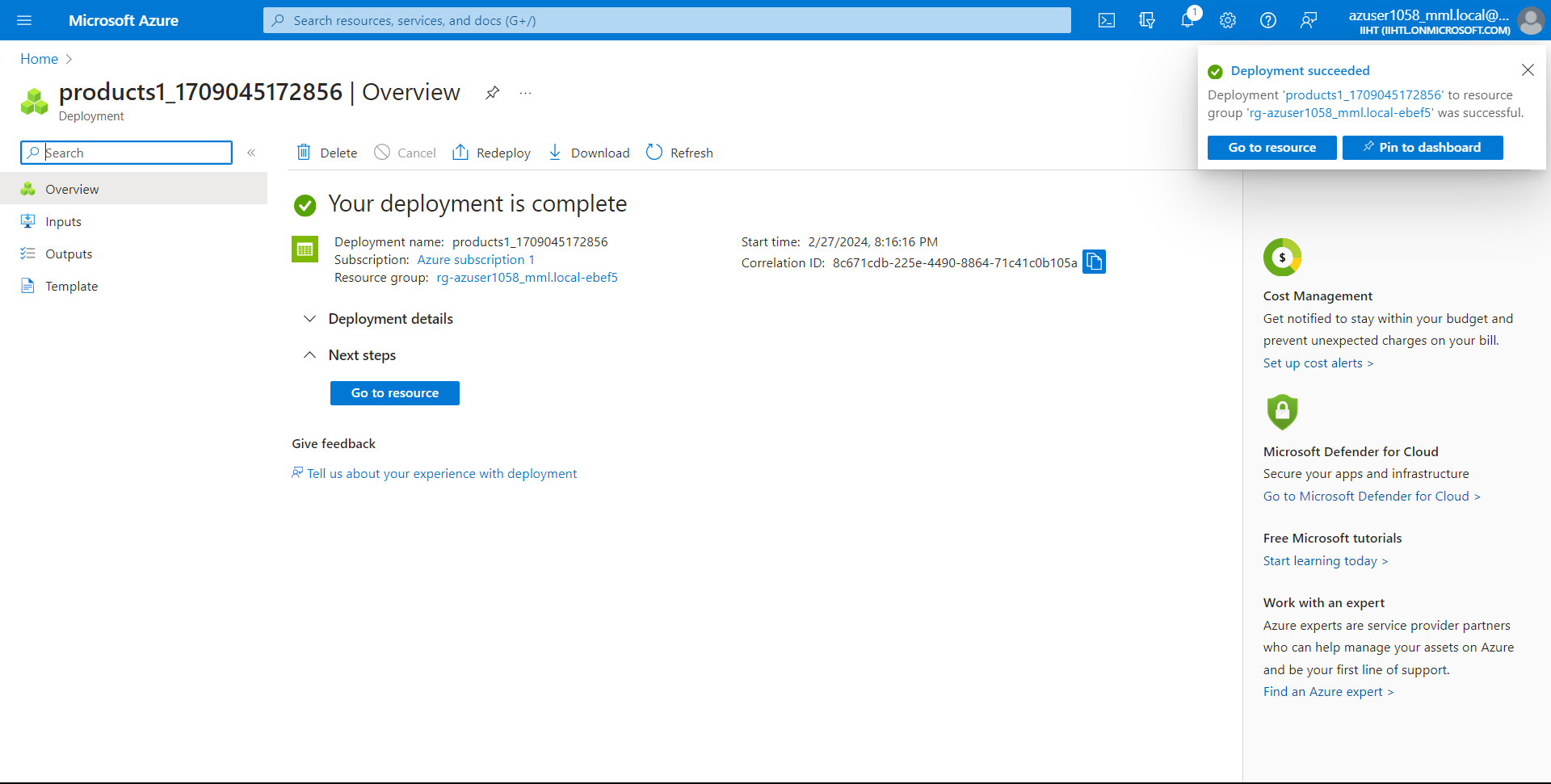
Now for performing Data flow we need to use Data Lake Gen2 so for this we need to go to Advanced and we need to enable the **“Hierarchical namespace”**



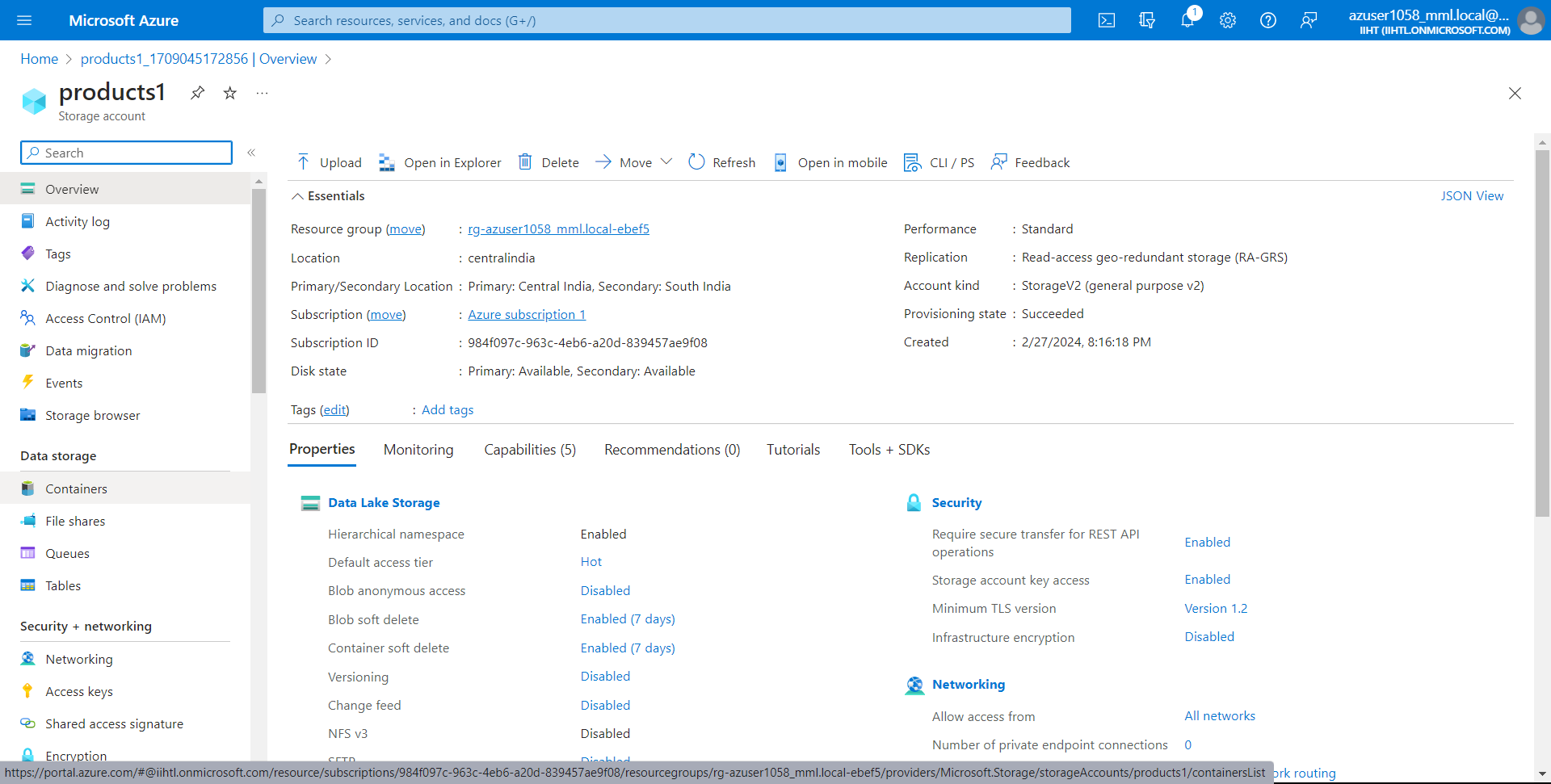
Now Review it and click on create.



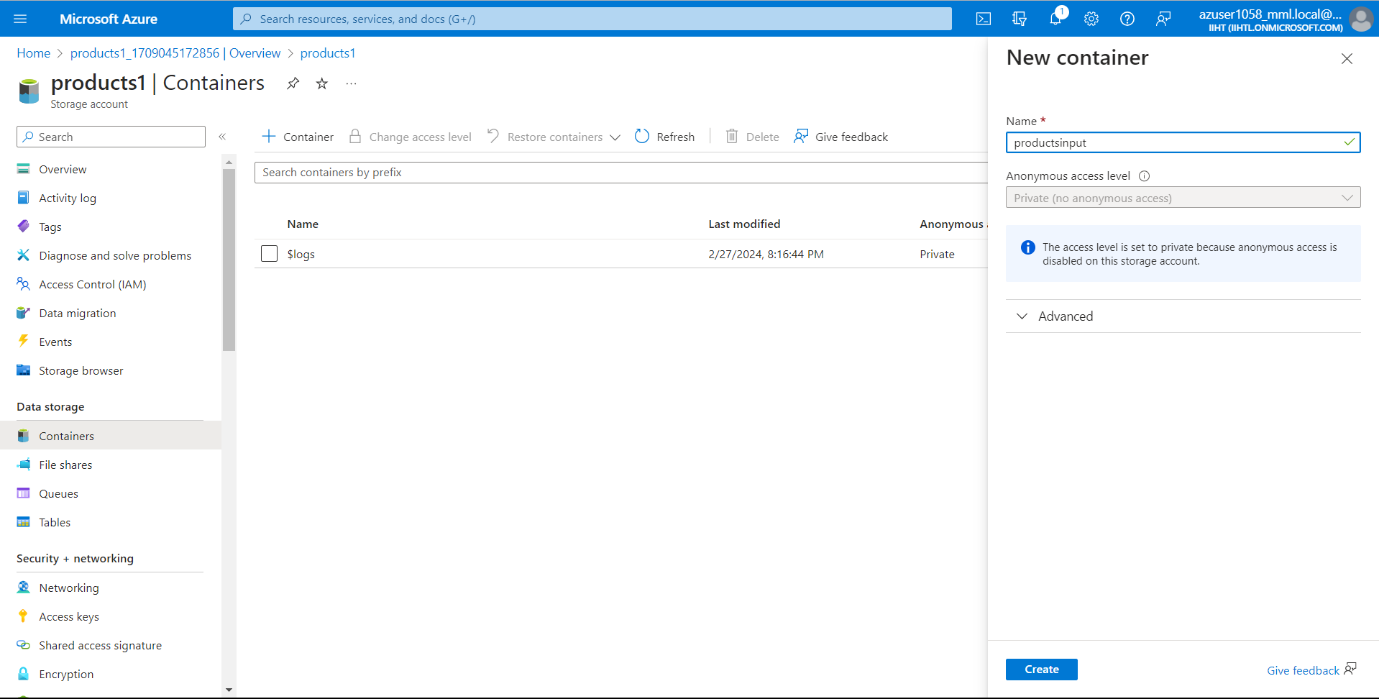
Now we have successfully created the storage account. Click on go to resource



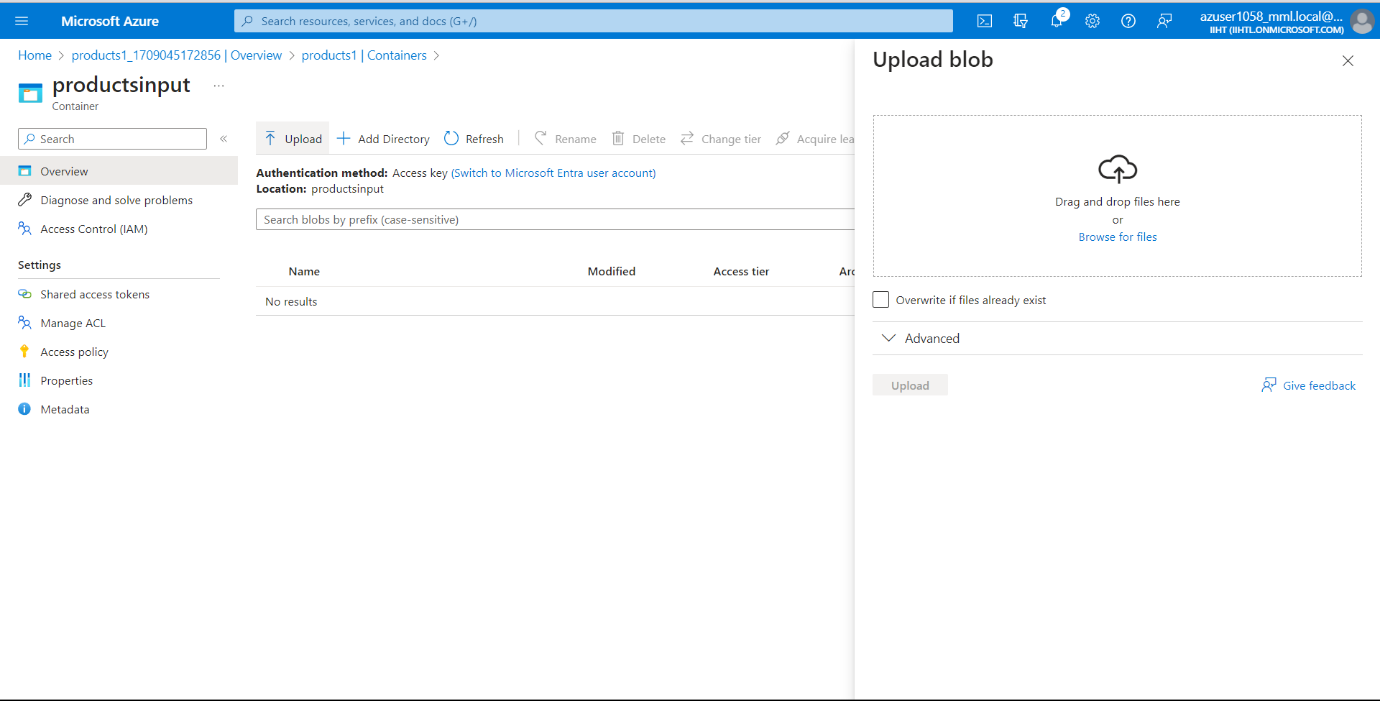
Now go to containers.



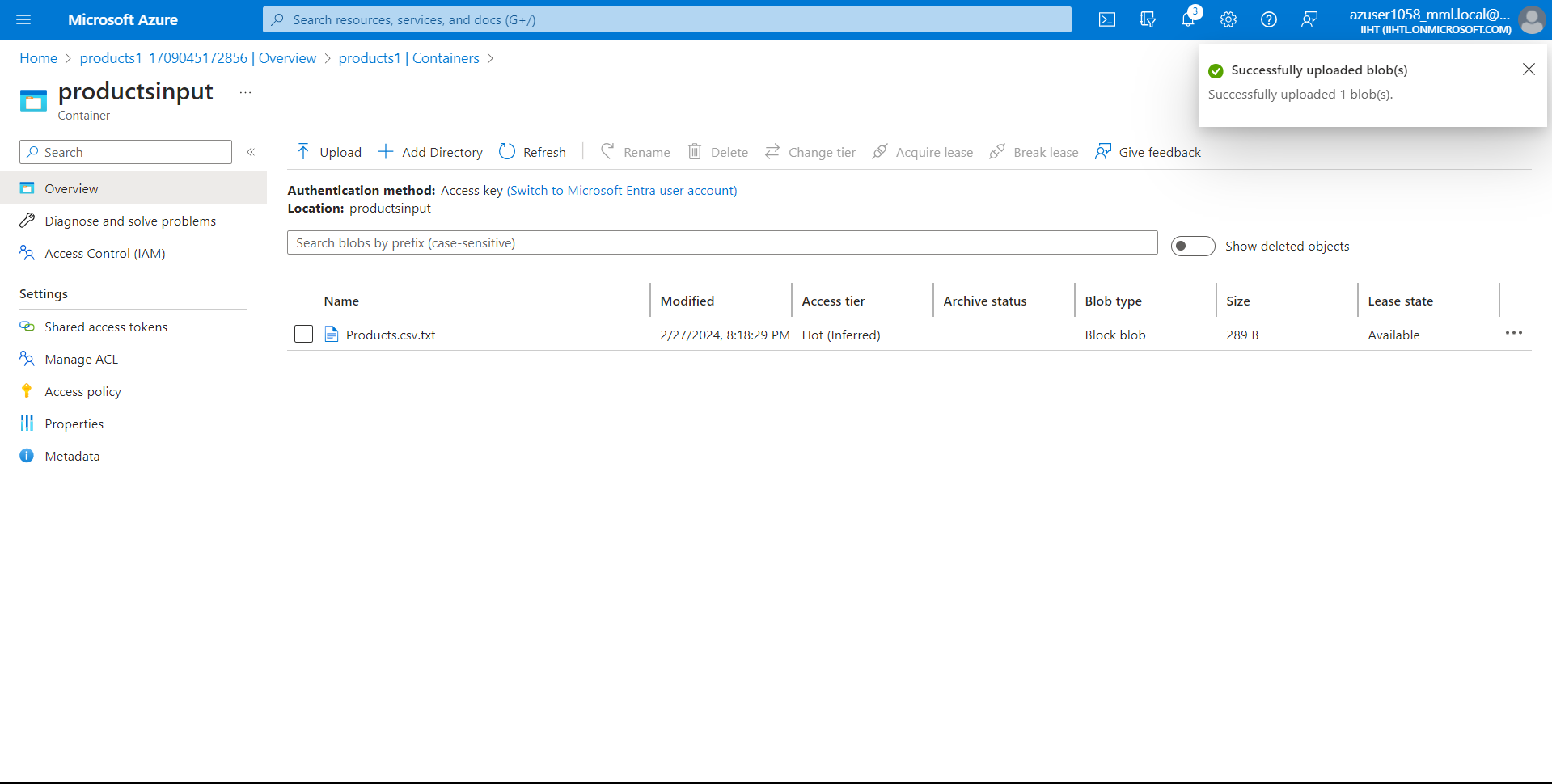
Click + Container and create a New Container on desired name. here I named the container as **“productsinput”**



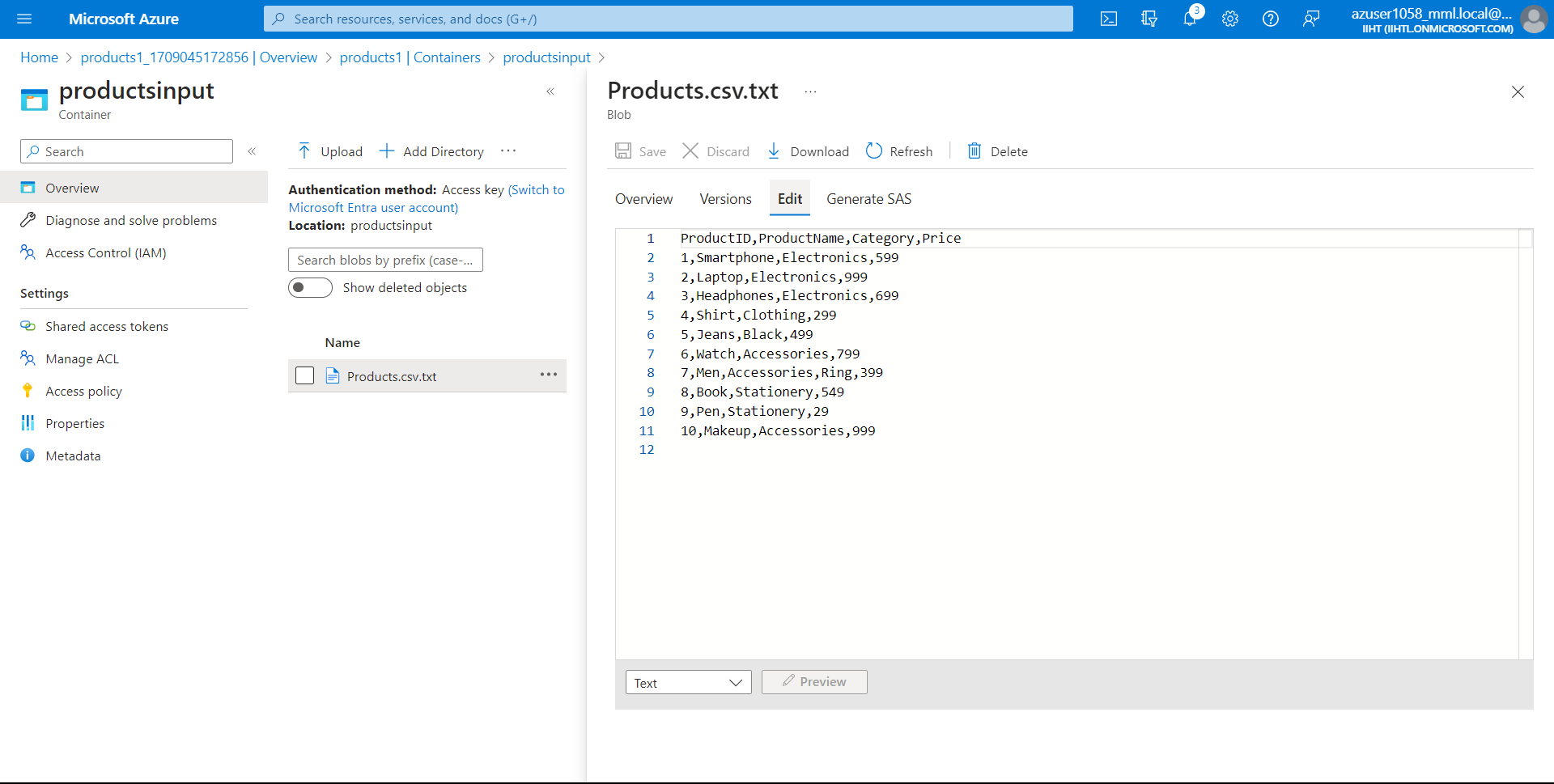
Now I uploaded a CSV File named **“products.csv.txt”** into the container from local device. We can also use any resource.



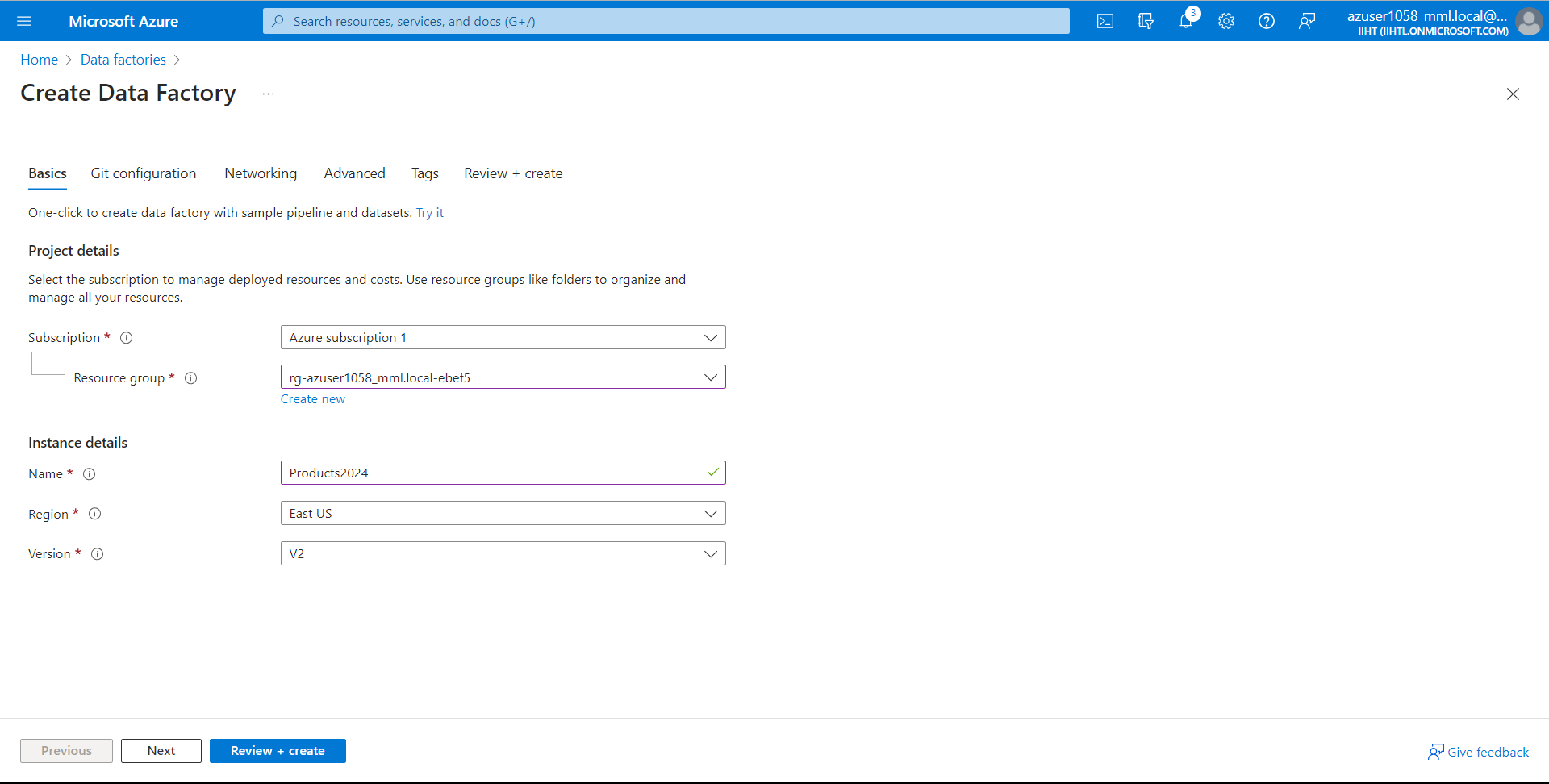
After uploading the CSV File click on Upload.



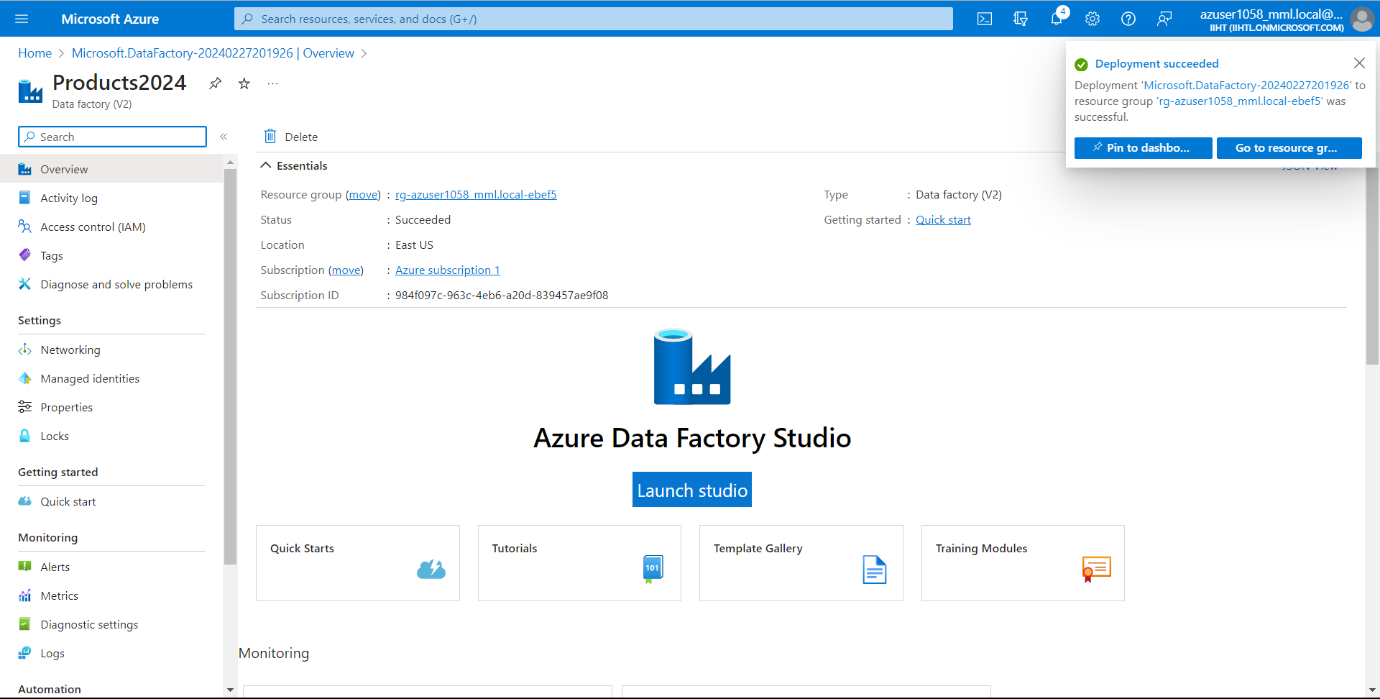
Click on uploaded file and you can click on Edit to view the data present in the CSV File.



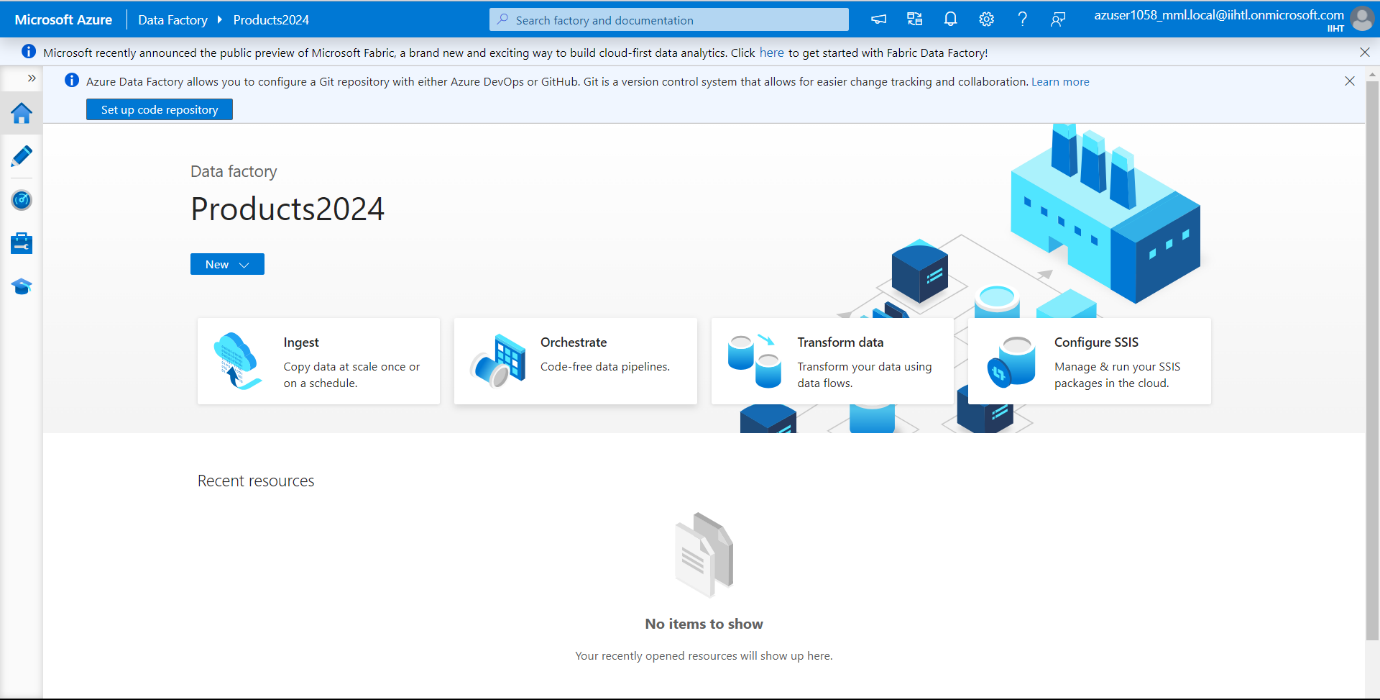
Now, create a Data Factory on your Resource Group and enter your desired name, Region and click on Review + Create



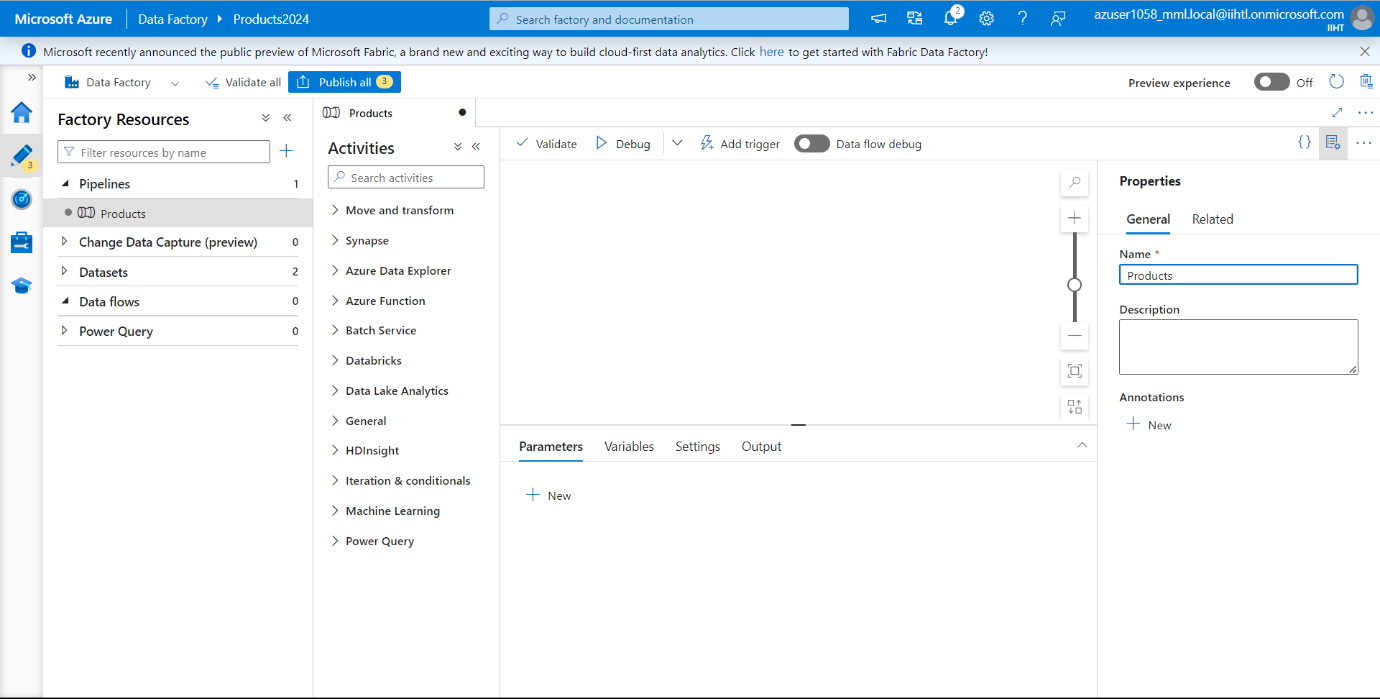
Now After successful creating the Data Factory, Launch the studio



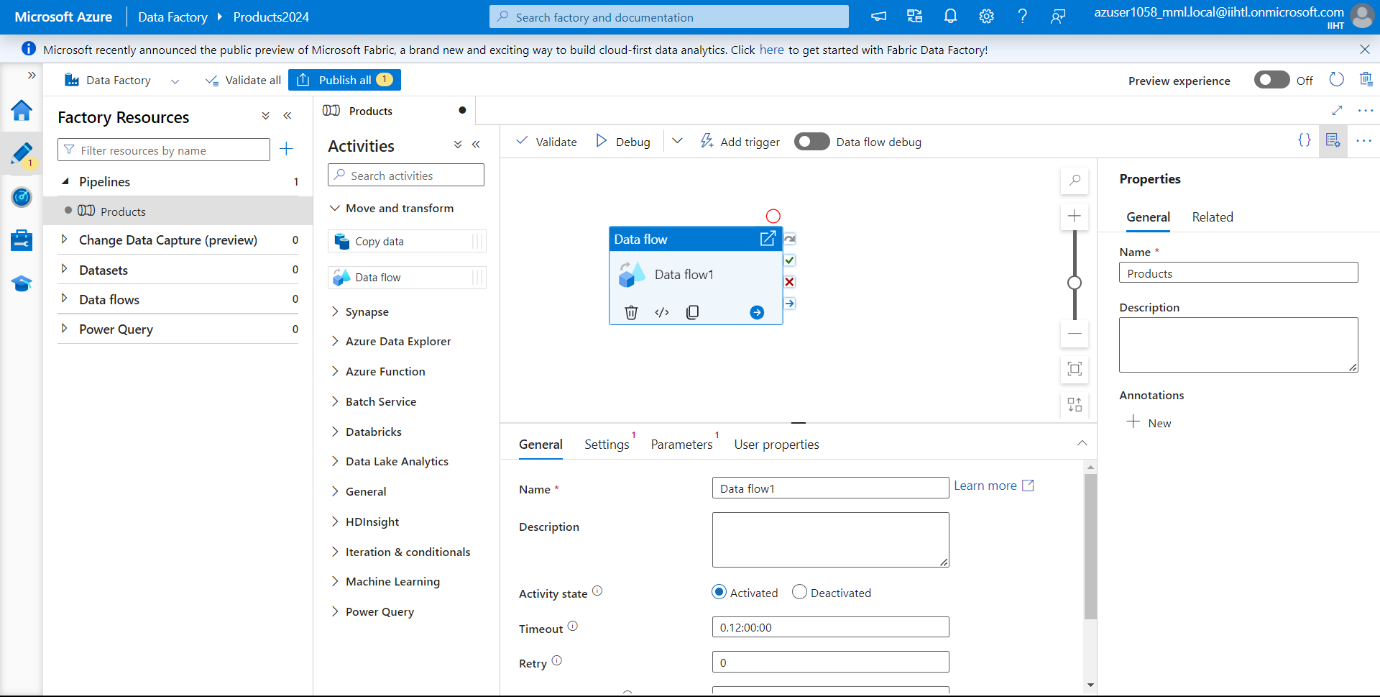
Now once we are on the Home Page of Azure Data Factory, select **Orchestrate.**



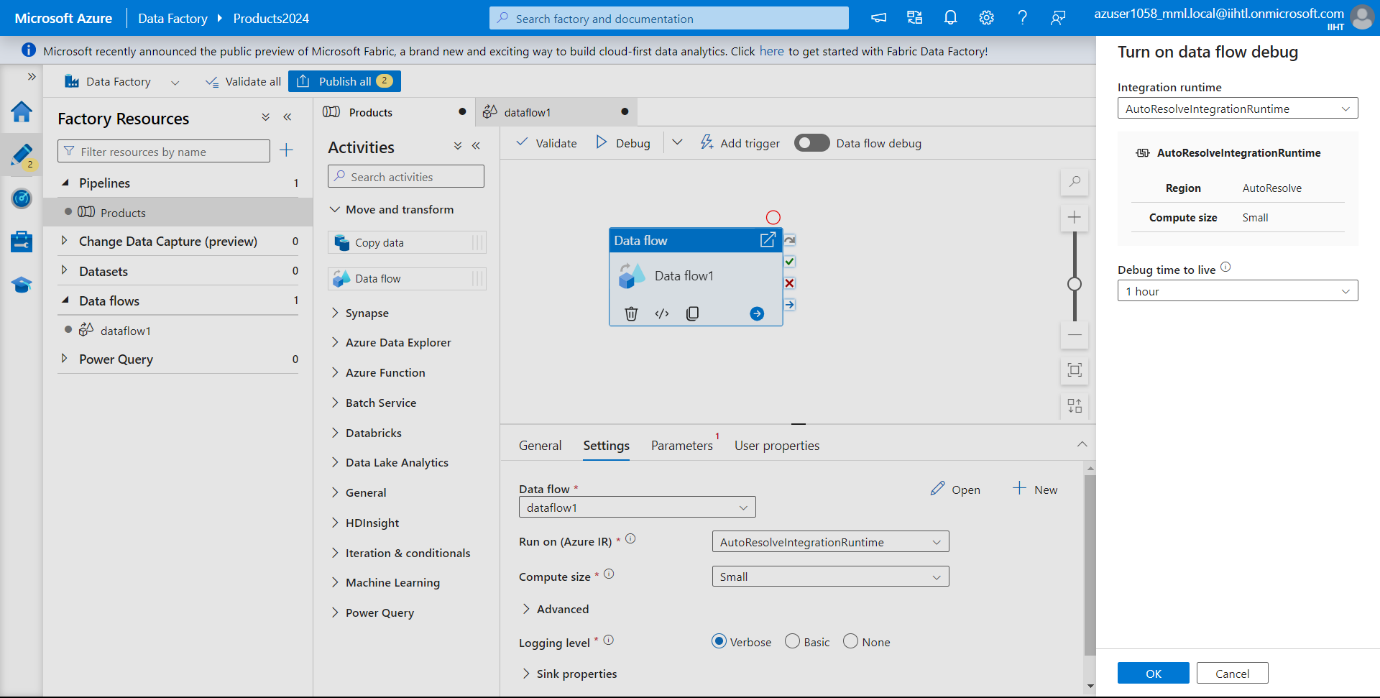
In General, Name your pipeline. I have named it as **“Products”**



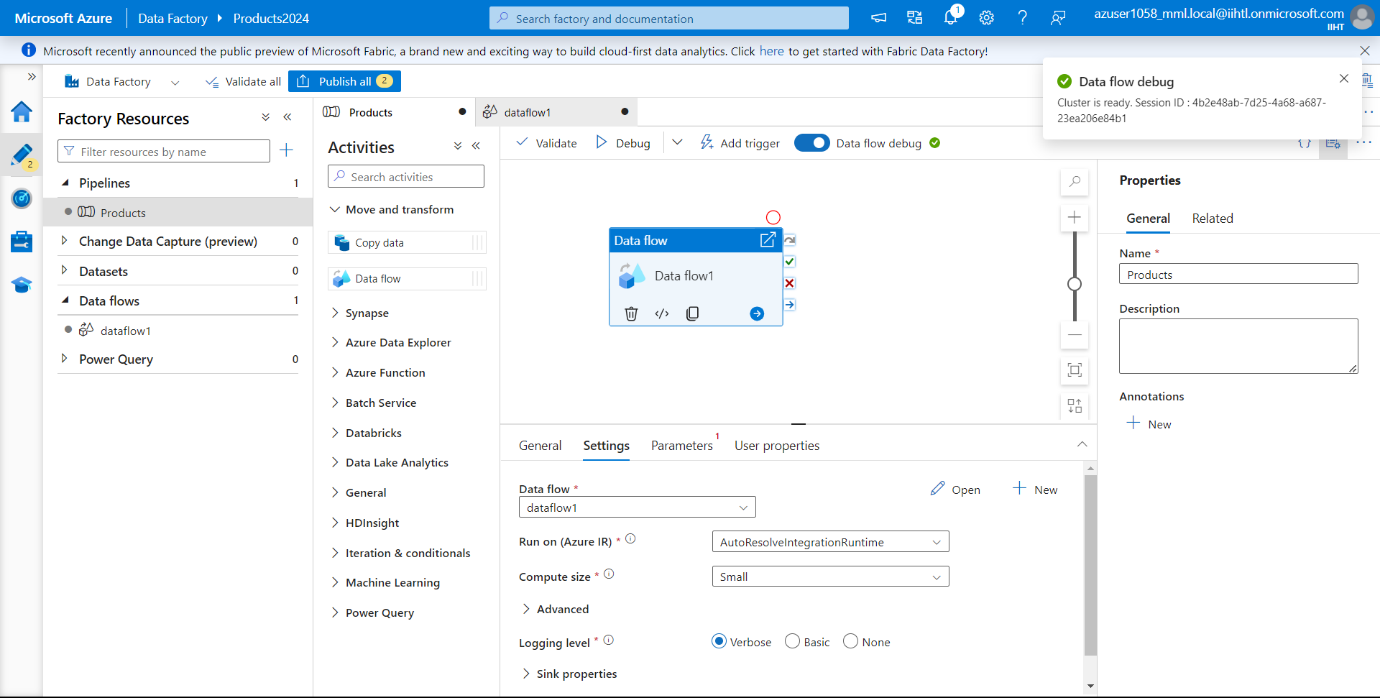
Now in Activities, we see option as **“Move and transform”** and Select **Data flow** and drag to the canvas.



Now for data flow we need slide the Data Flow debug on and you will get a Pop up and click on Ok.



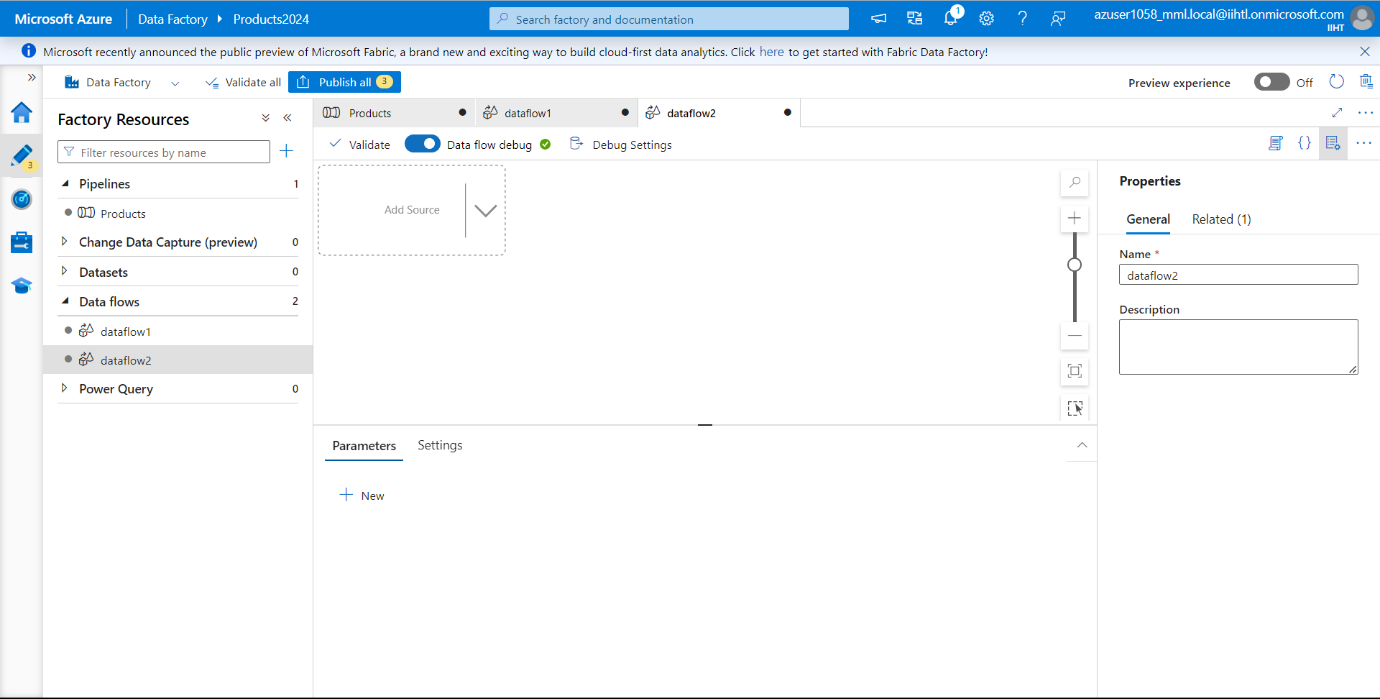
Wait for few mins and we can see here we have successfully enabled Data Flow debug.



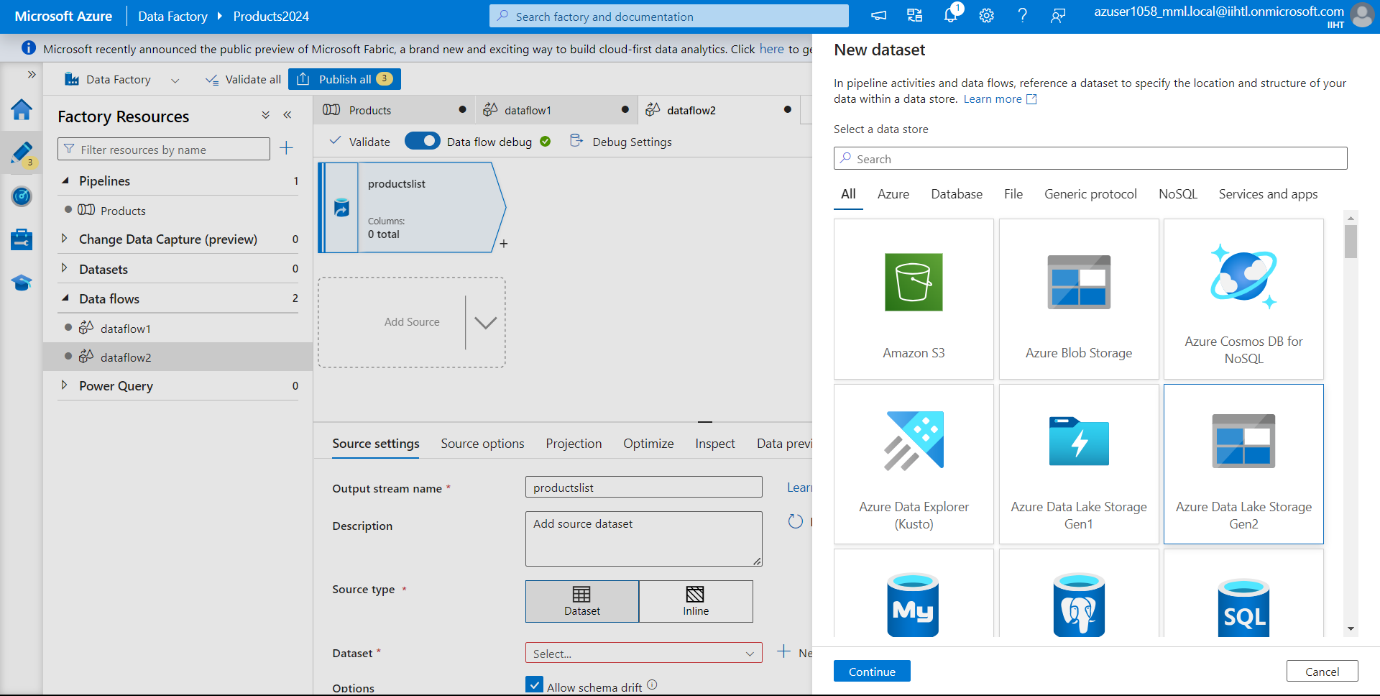
Now Go to settings and Click on + New



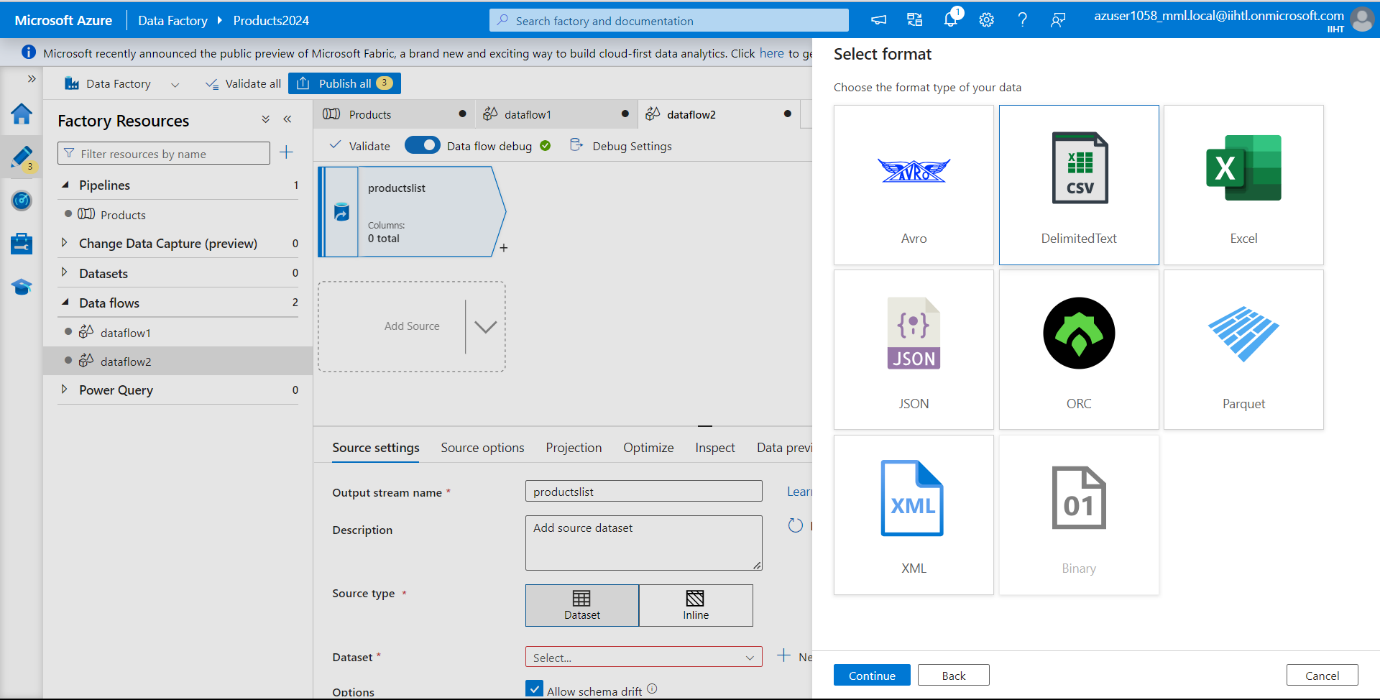
Now add a source by clicking on the Add Source box.



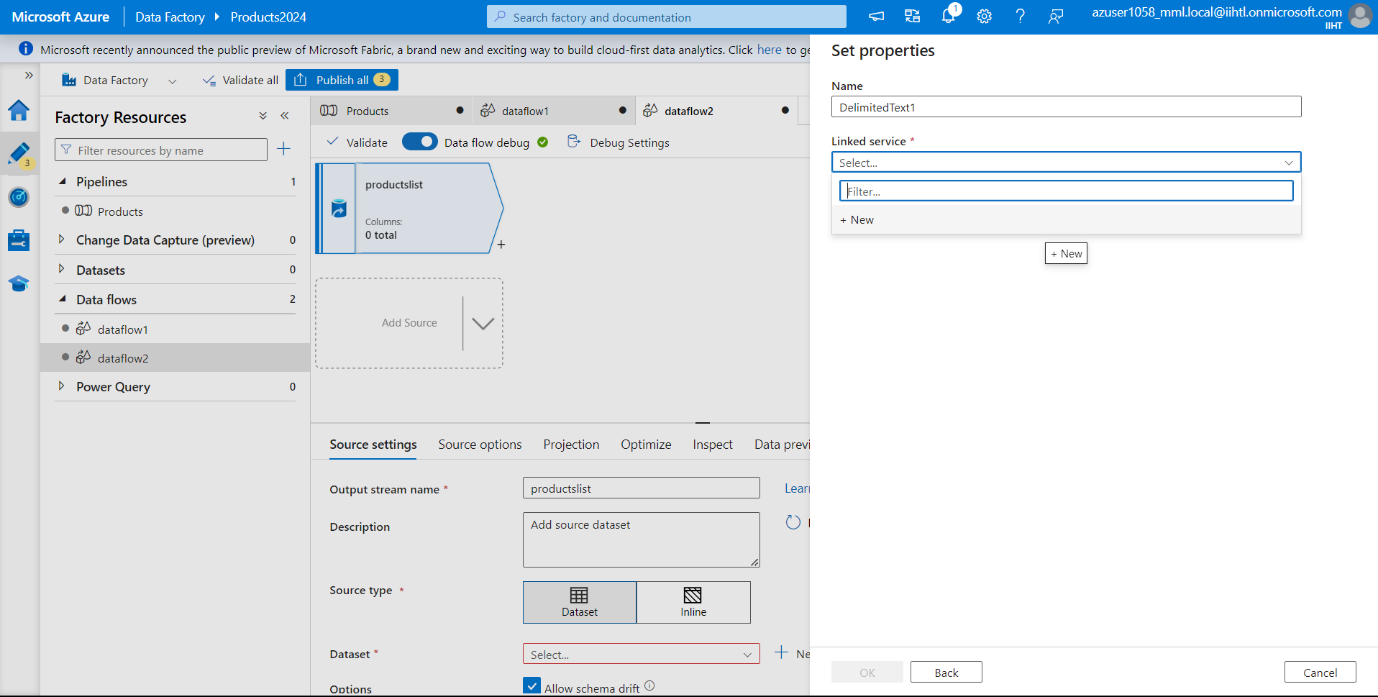
Now in source settings, click on + New Dataset to create a new dataset and Select Azure Data Lake Storage Gen2 and click on continue.



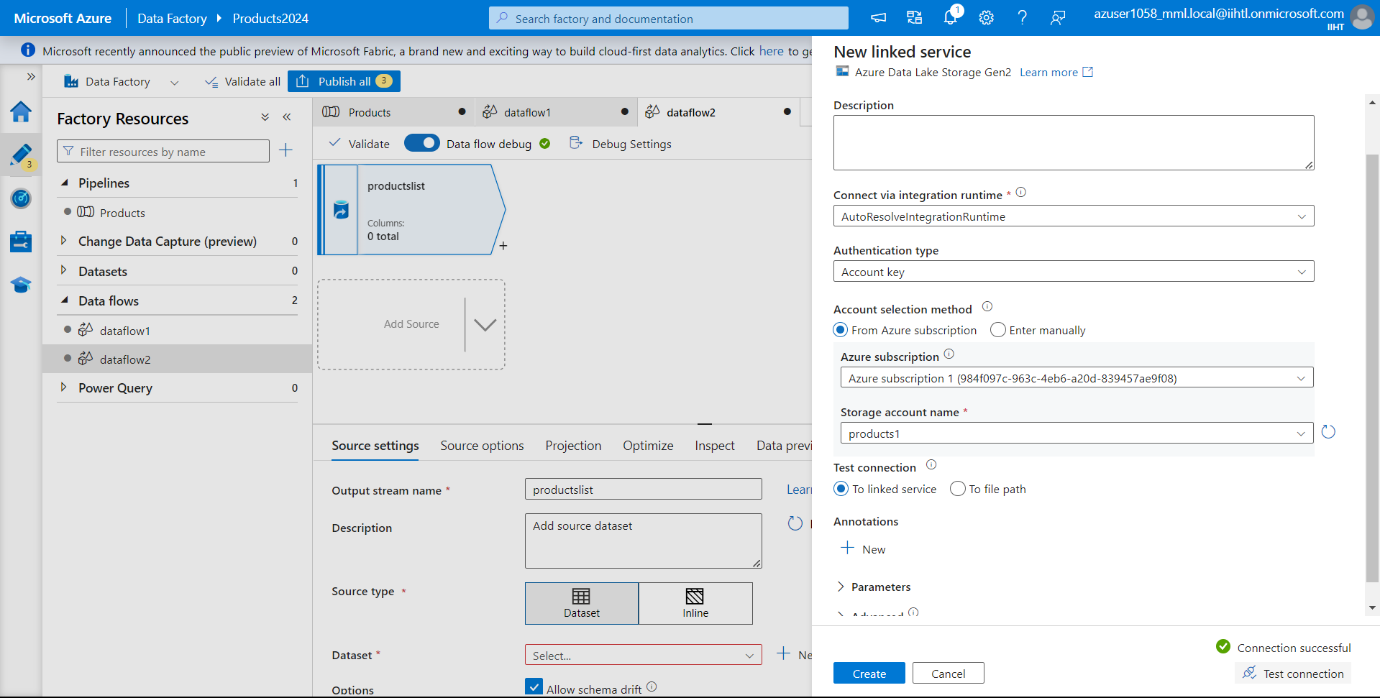
Now select format as CSV File and click on continue.



Now click on + New to create a New Linked Service.



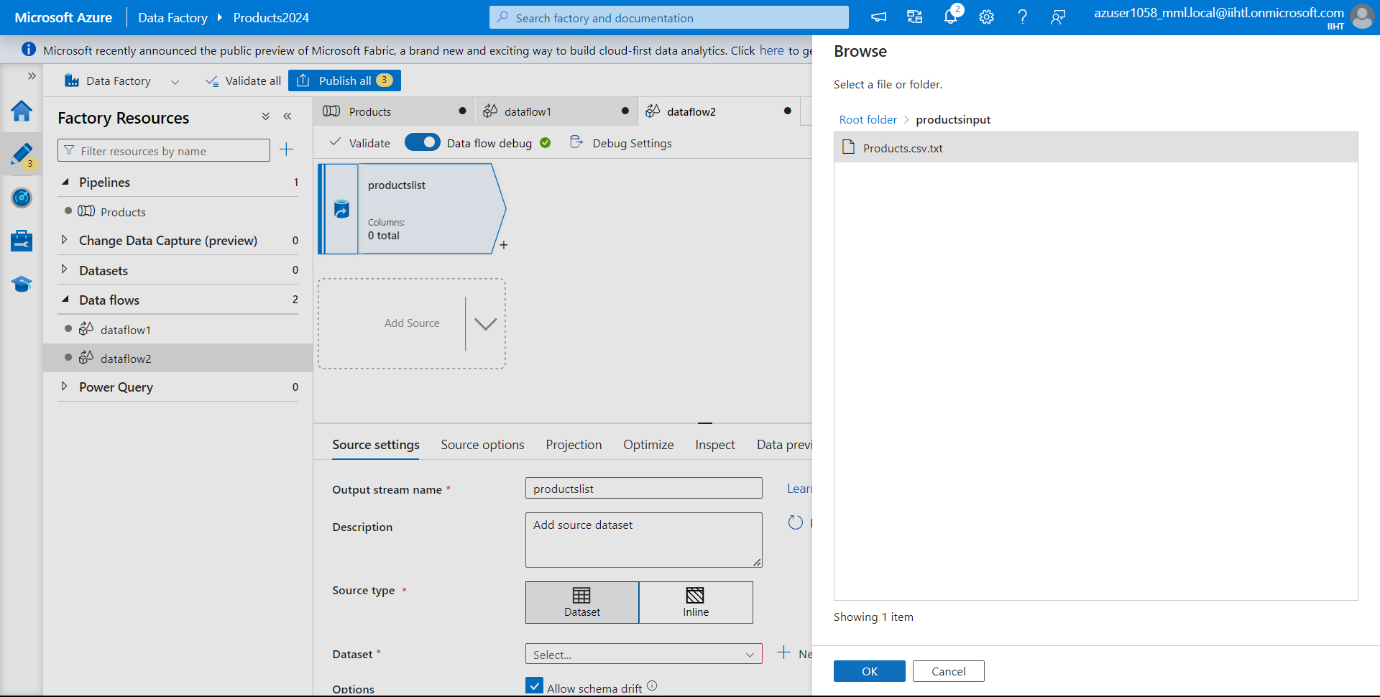
And now select your source. We have **“products1”** Storage account where we created and we tested the connection and it is successful. Click on Create.



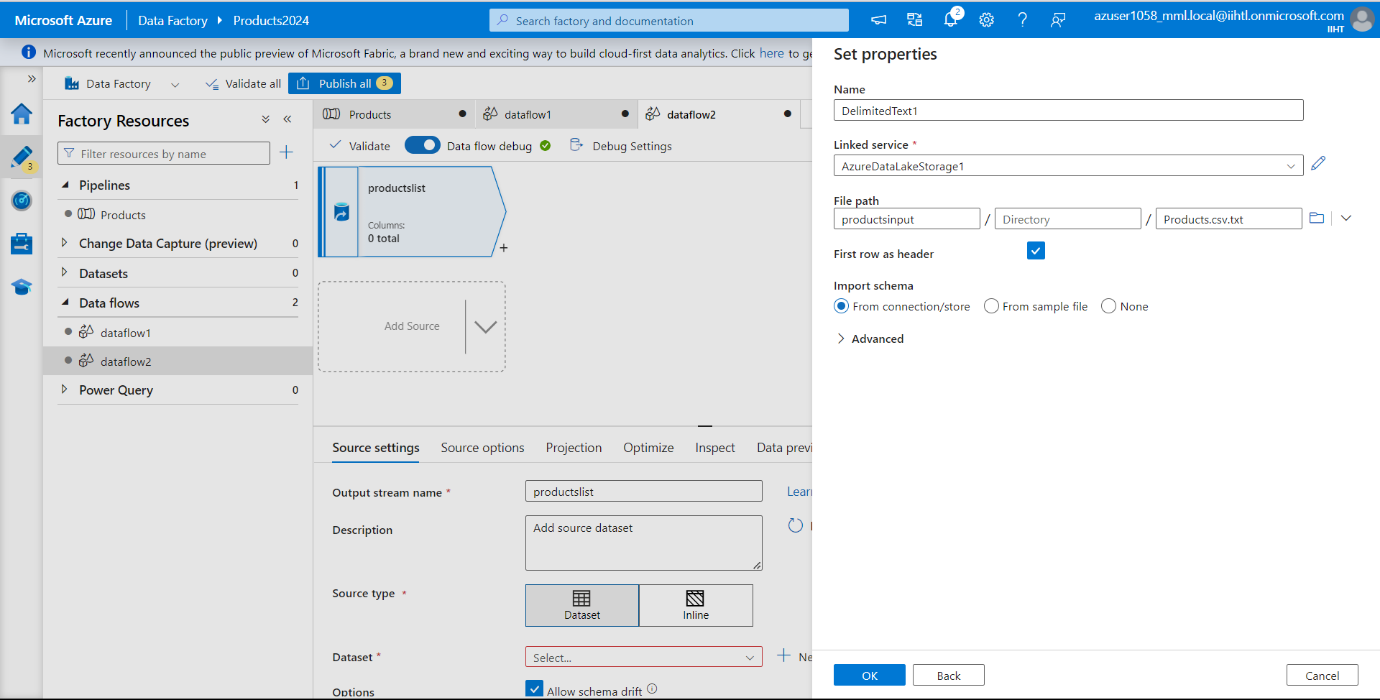
Now browse to the source path.



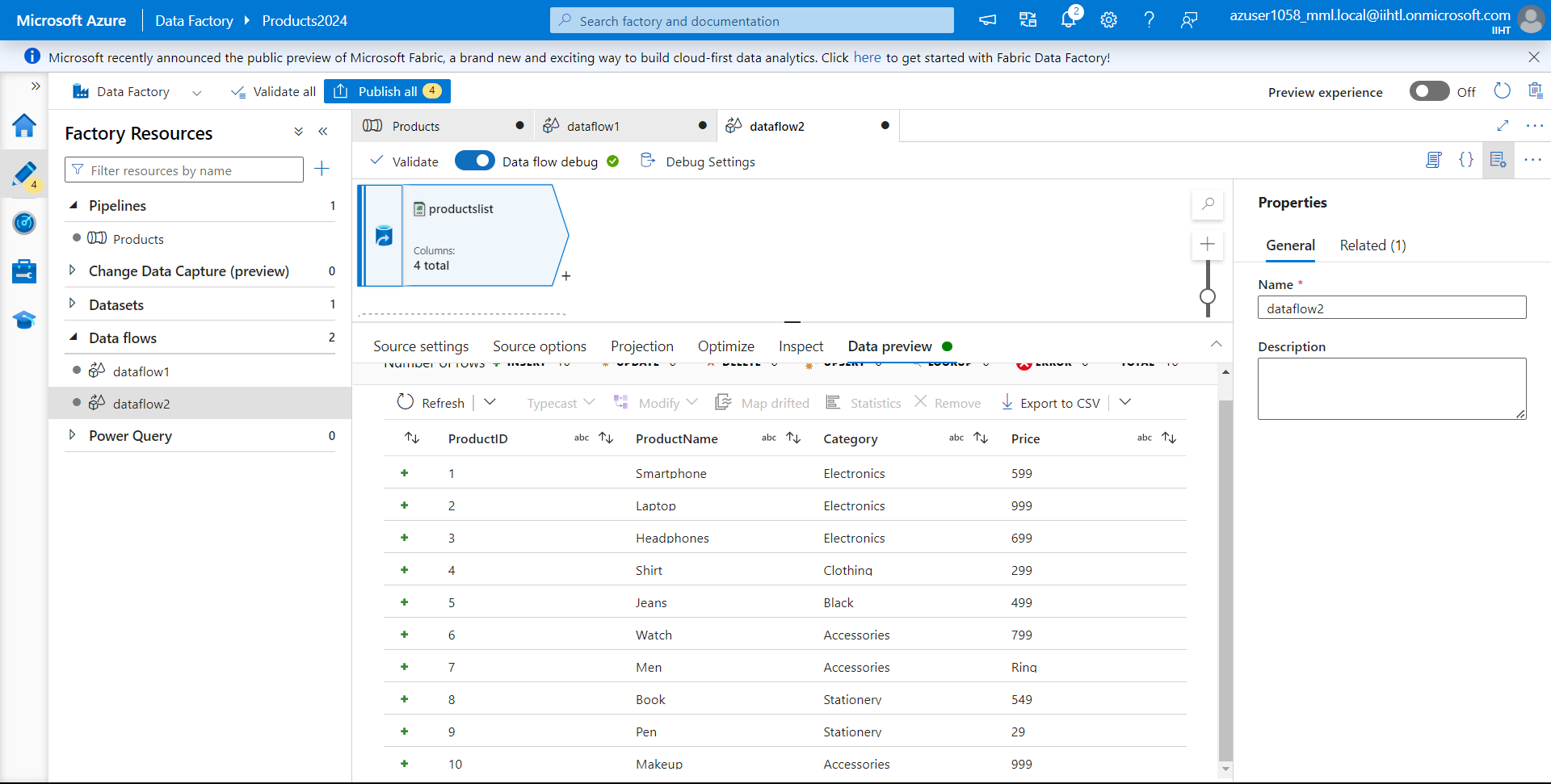
Now select the file from source path.



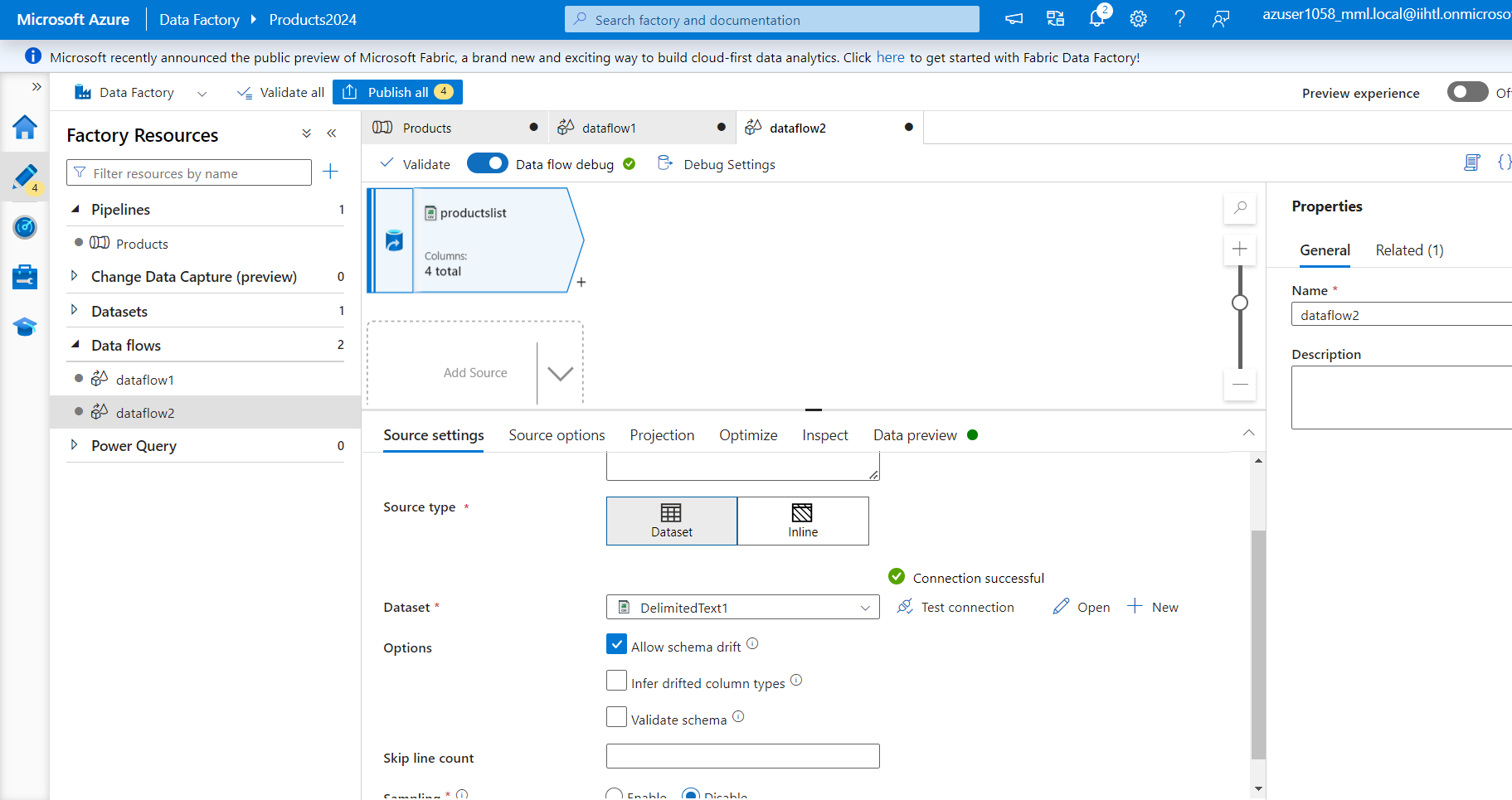
Click on ok.



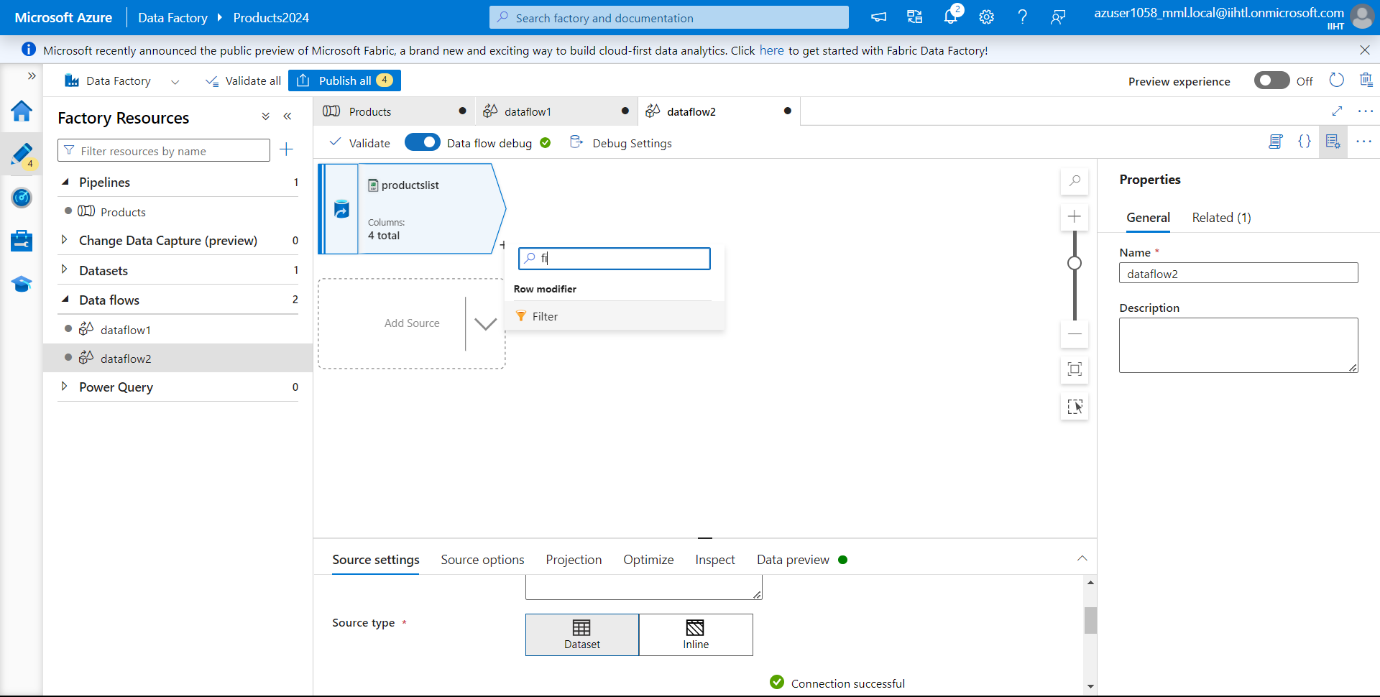
Now go to Data Preview tab of the source transformation and click Refresh to preview and verify the transformation data is correct or not.



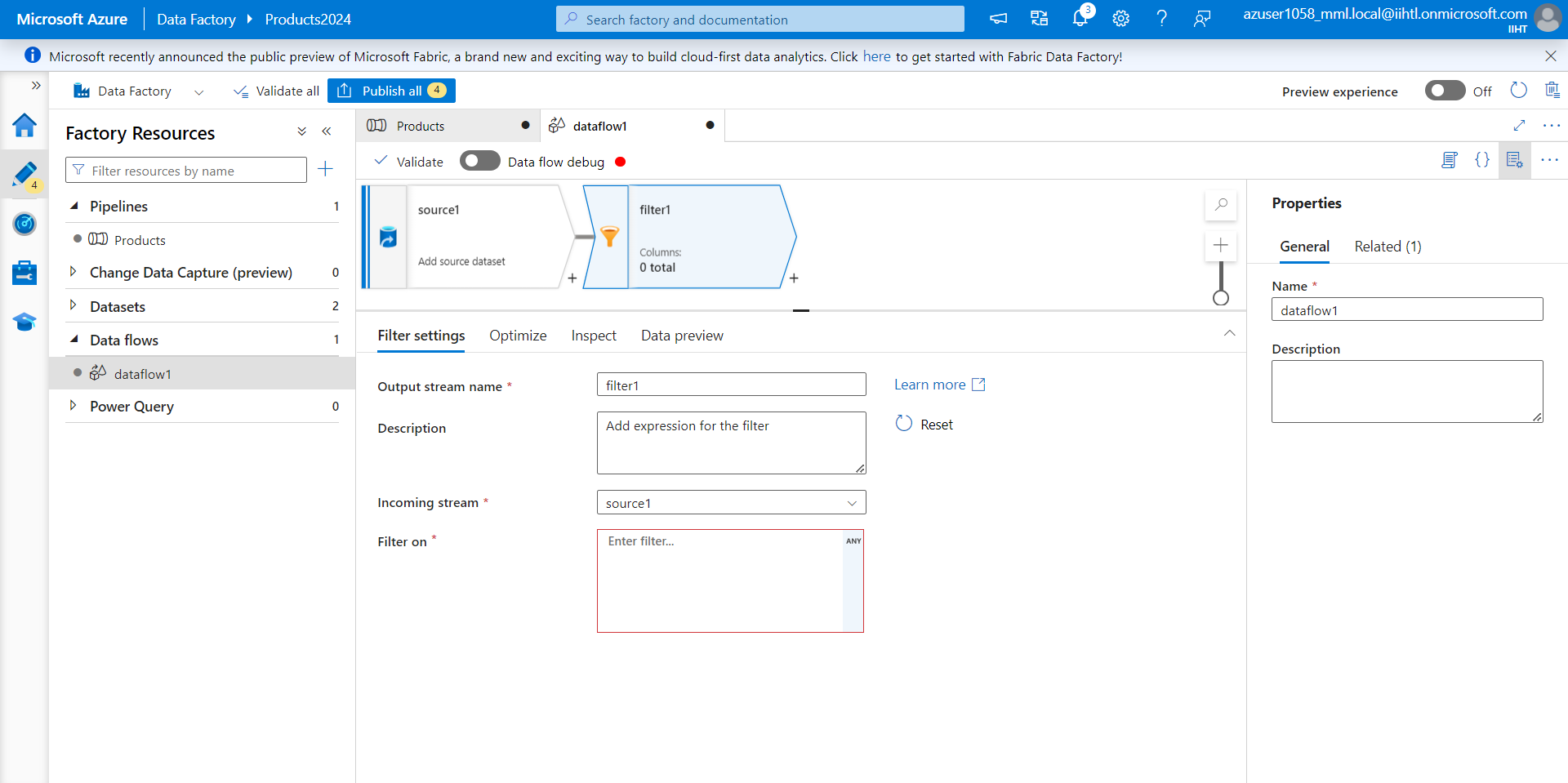
Now go to Source settings and test the connection. Here we have the connection successful.



Now next to your source node on the data flow canvas, click on + icon to add a new transformation. Search for filter because we are performing first transformation by adding a Filter.



Now Click on the expression box next to Filter on to open the expression builder. Here we will specify our filtering condition.



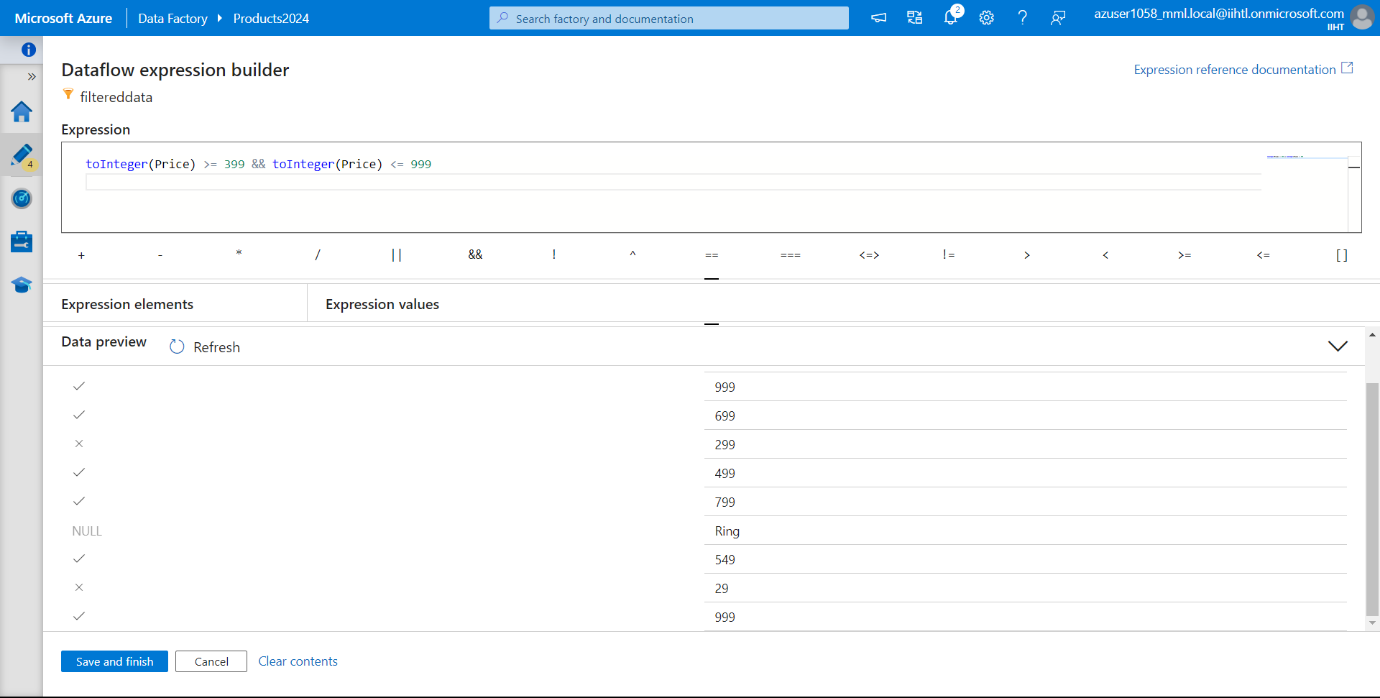
Now we will filter between the prices 399 and 999. As Price is currently a string, you need to convert it to an integer using the toInteger() function. Here we used greater than or equals to (>=) and than or equals to (<=) operators. Union these expressions together with the and (&&) operator.

The Expression is written as:

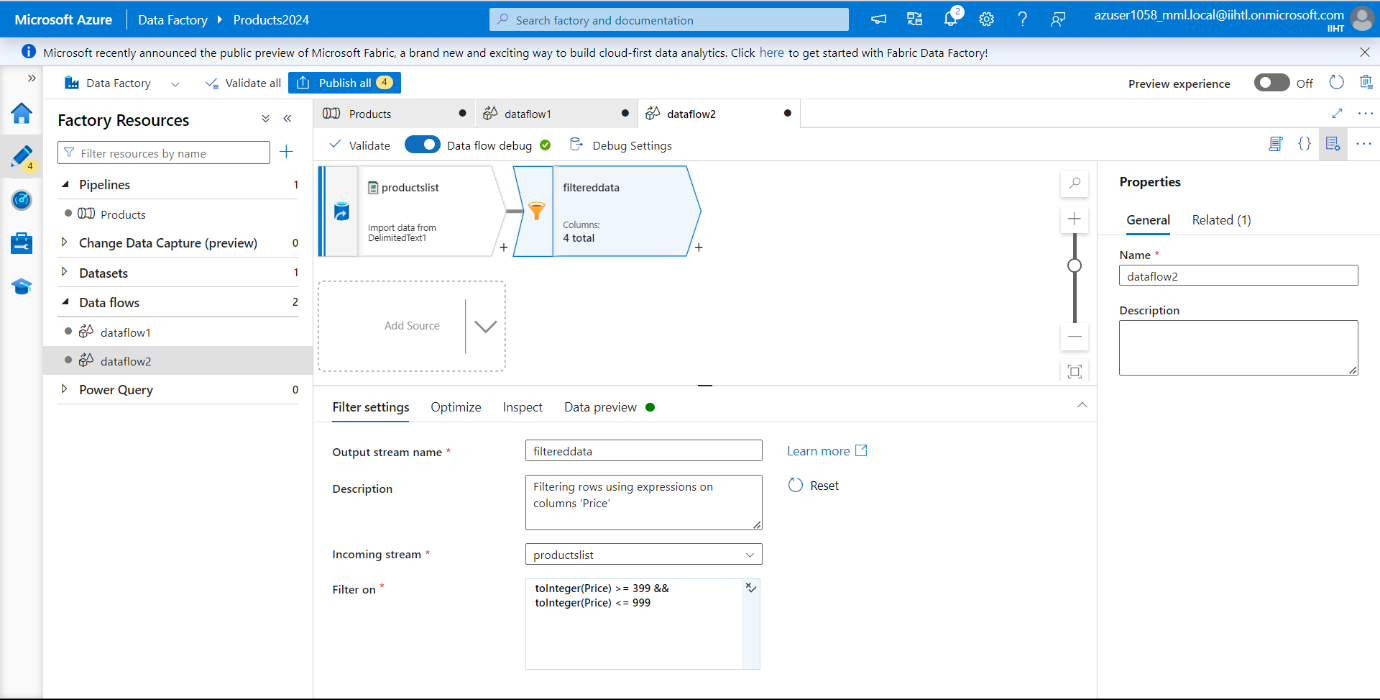
**“toInteger(Price) >= 399 && toInteger(Price) <= 999”**

you can verify your logic by clicking Refresh to see expression output compared to the inputs used.

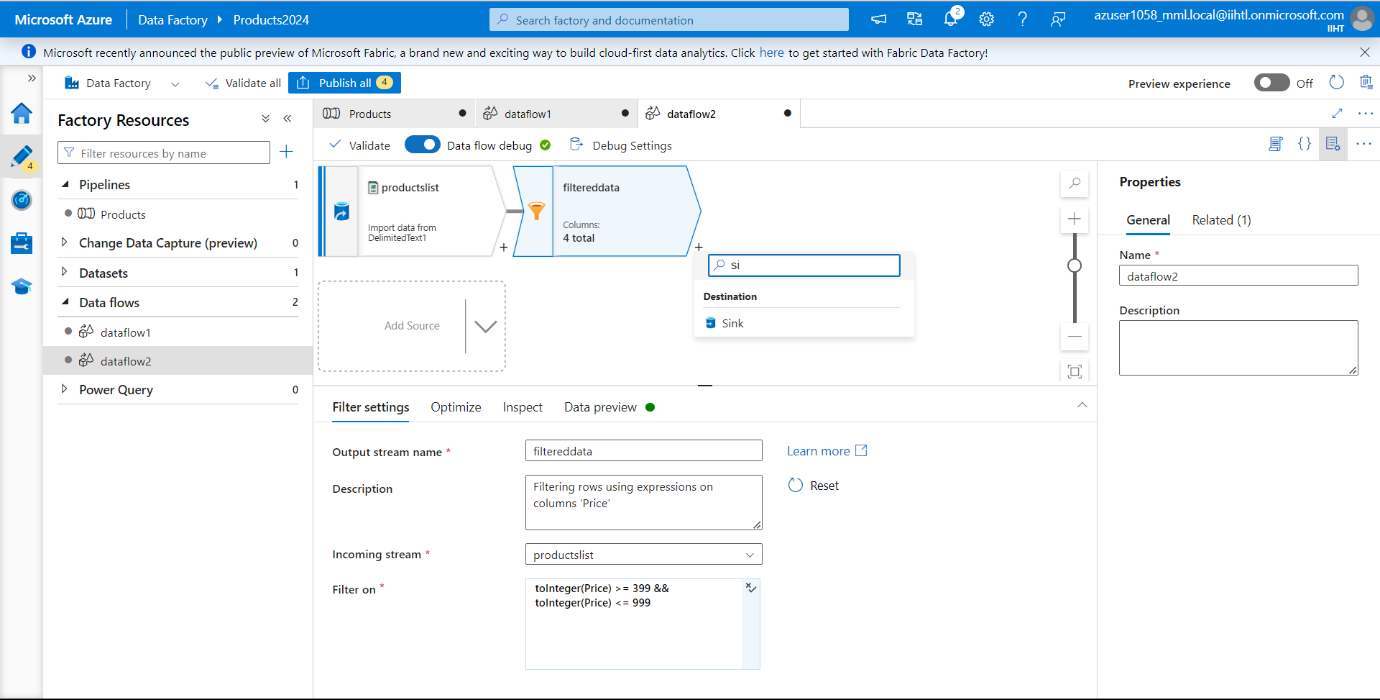
Next click on Save and Finish.



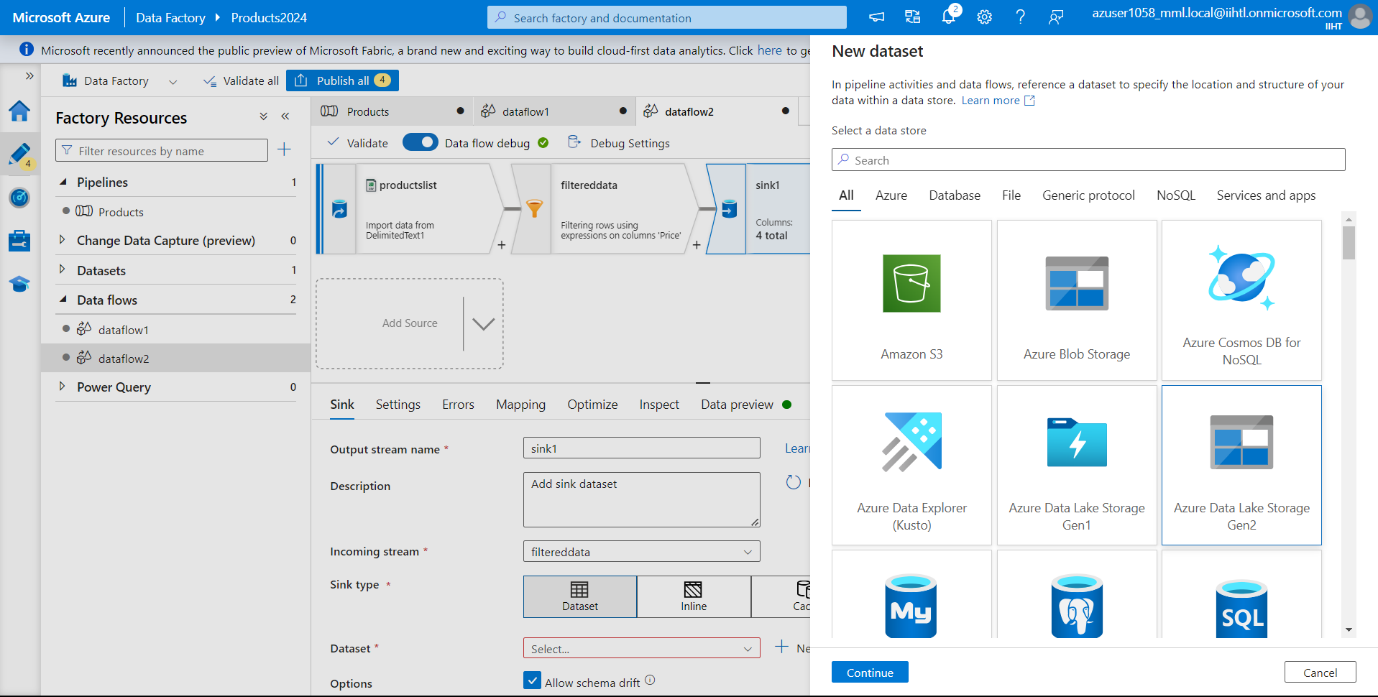
Now you can see in Filter On, the expression which we have written appear here.



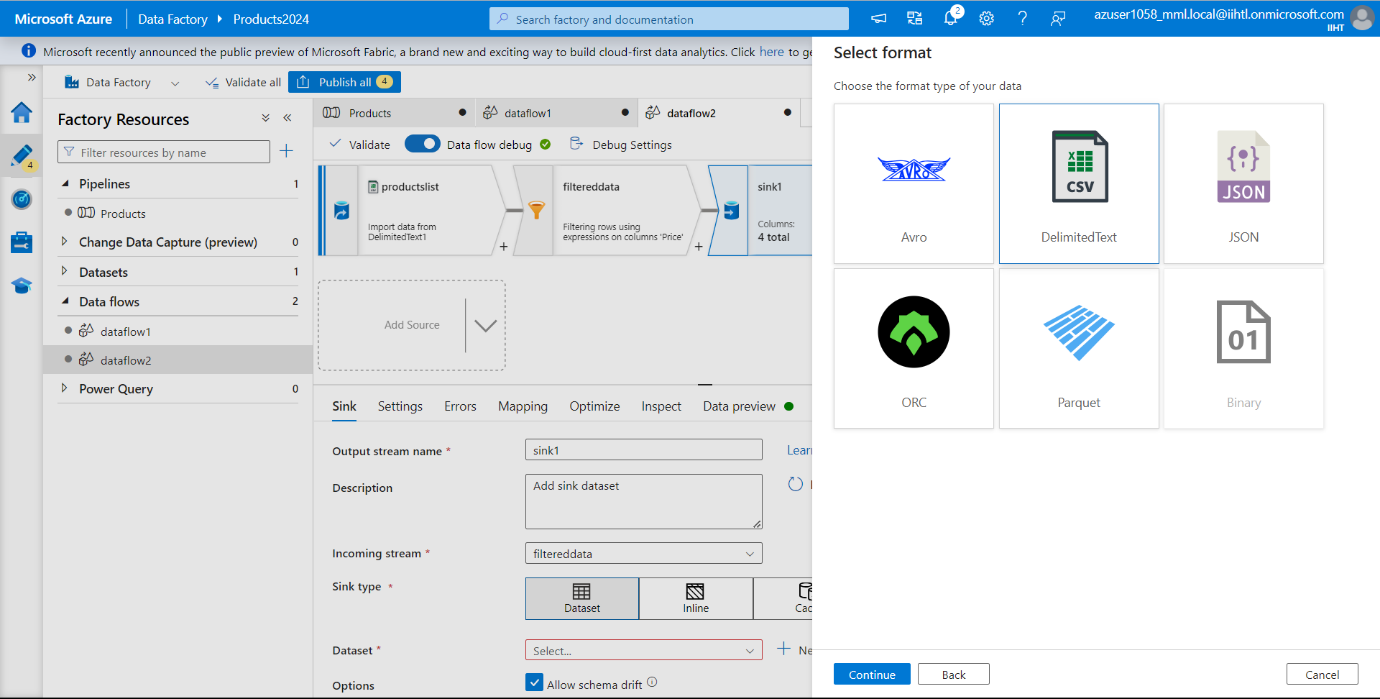
Now we need to add Sink transformation under Destination.



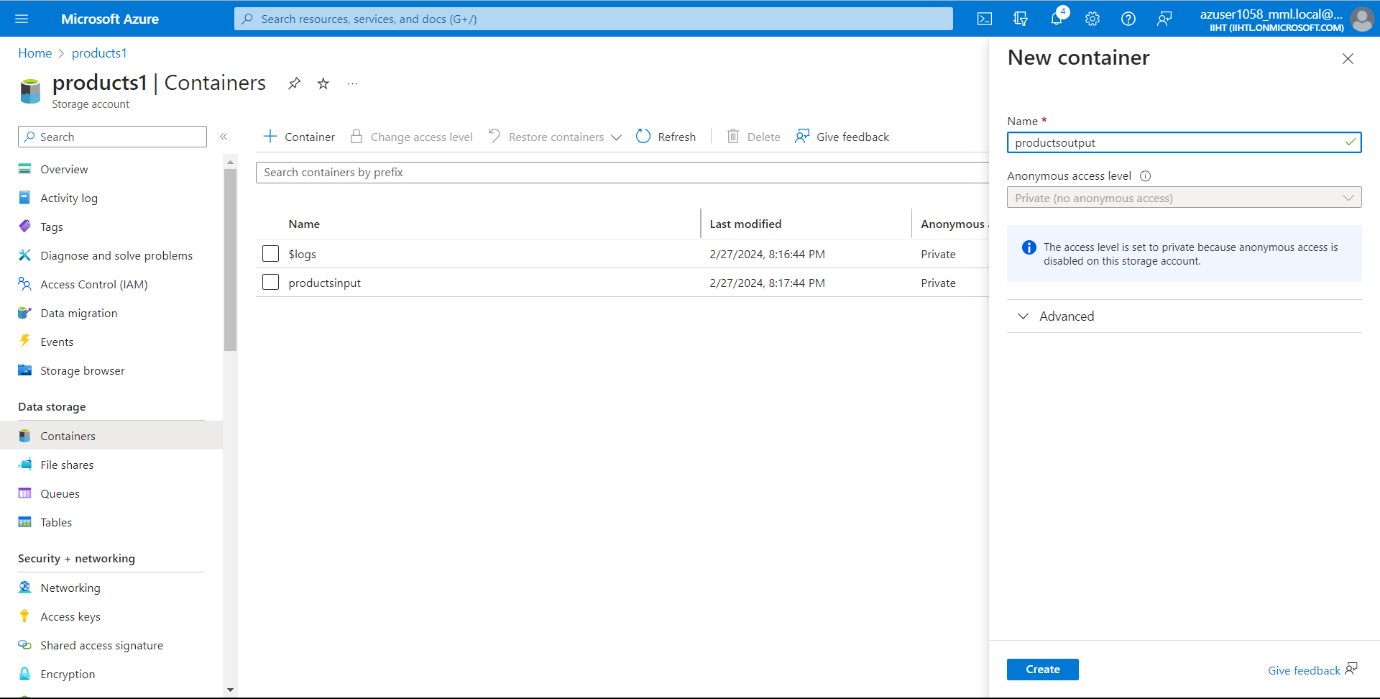
We have named Sink as **“sink1”.** Click New to create your sink dataset and select Azure Data Lake Storage Gen2 as New dataset and click on continue.



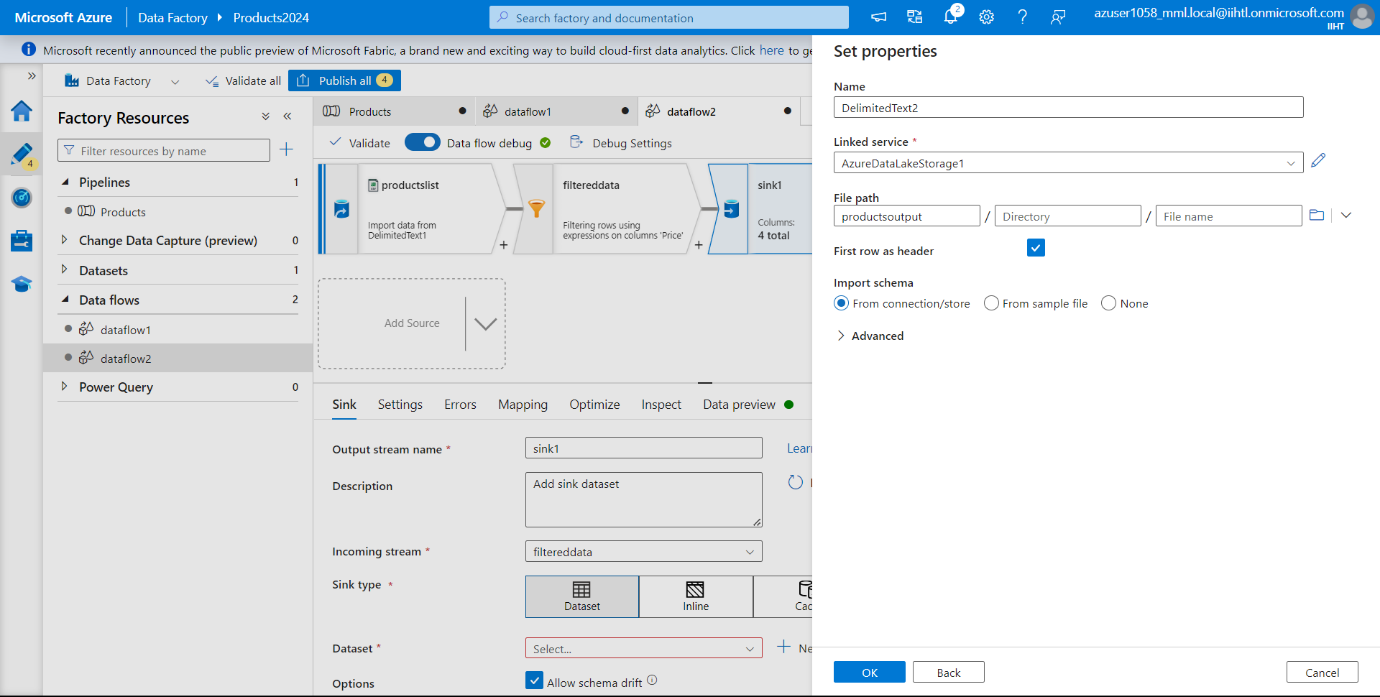
Select format as CSV File and click on Continue.



Now we need to create another storage for destination to store the New Output File. We have named the Destination Container name as **“productsoutput”** and click Create.

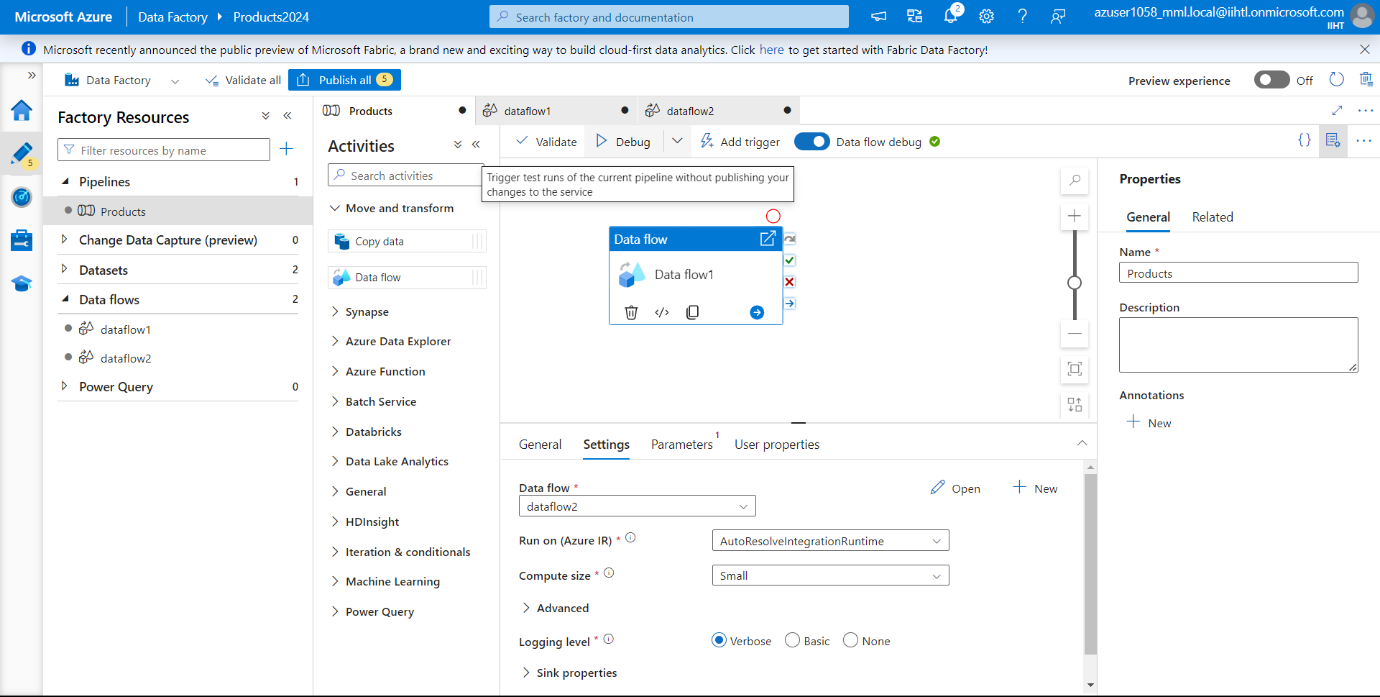


In set properties, select the destination path which we created dynamically and browse the file path and select the Import Schema as None and click on ok.

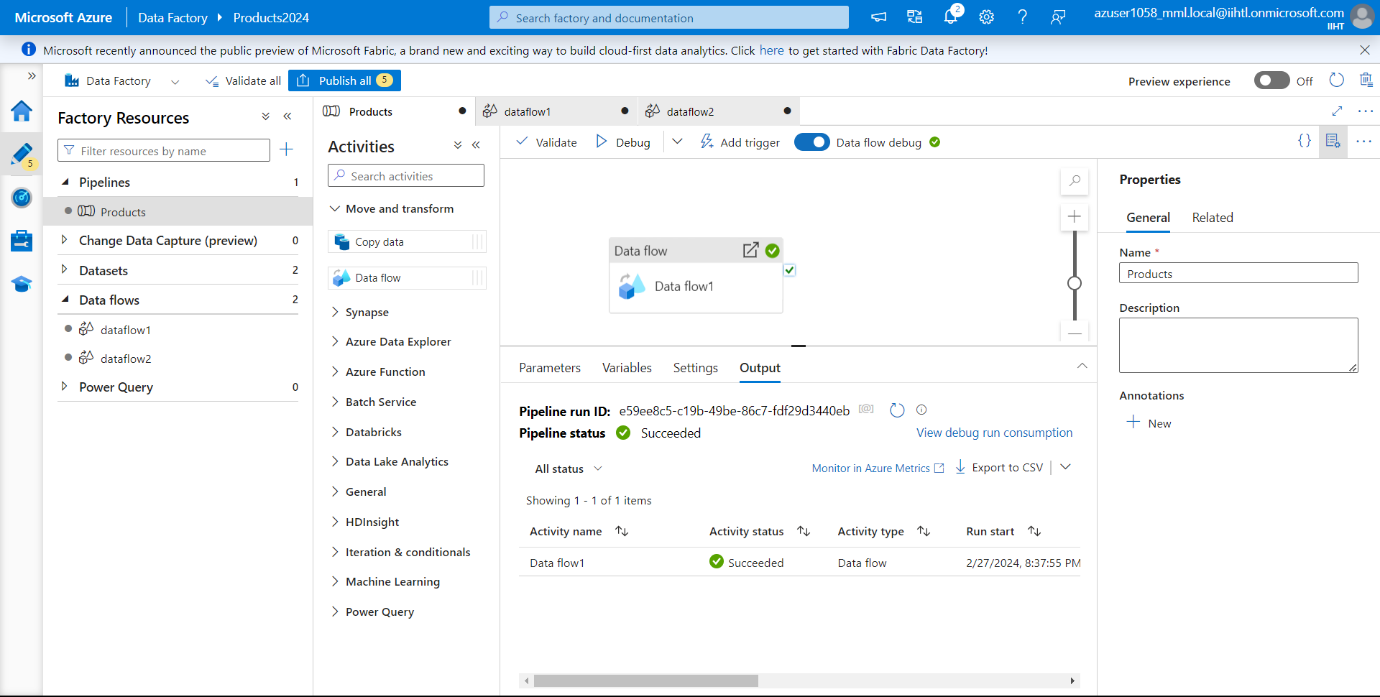


We have finished building the data flow process. Now we are ready to run the Pipeline.

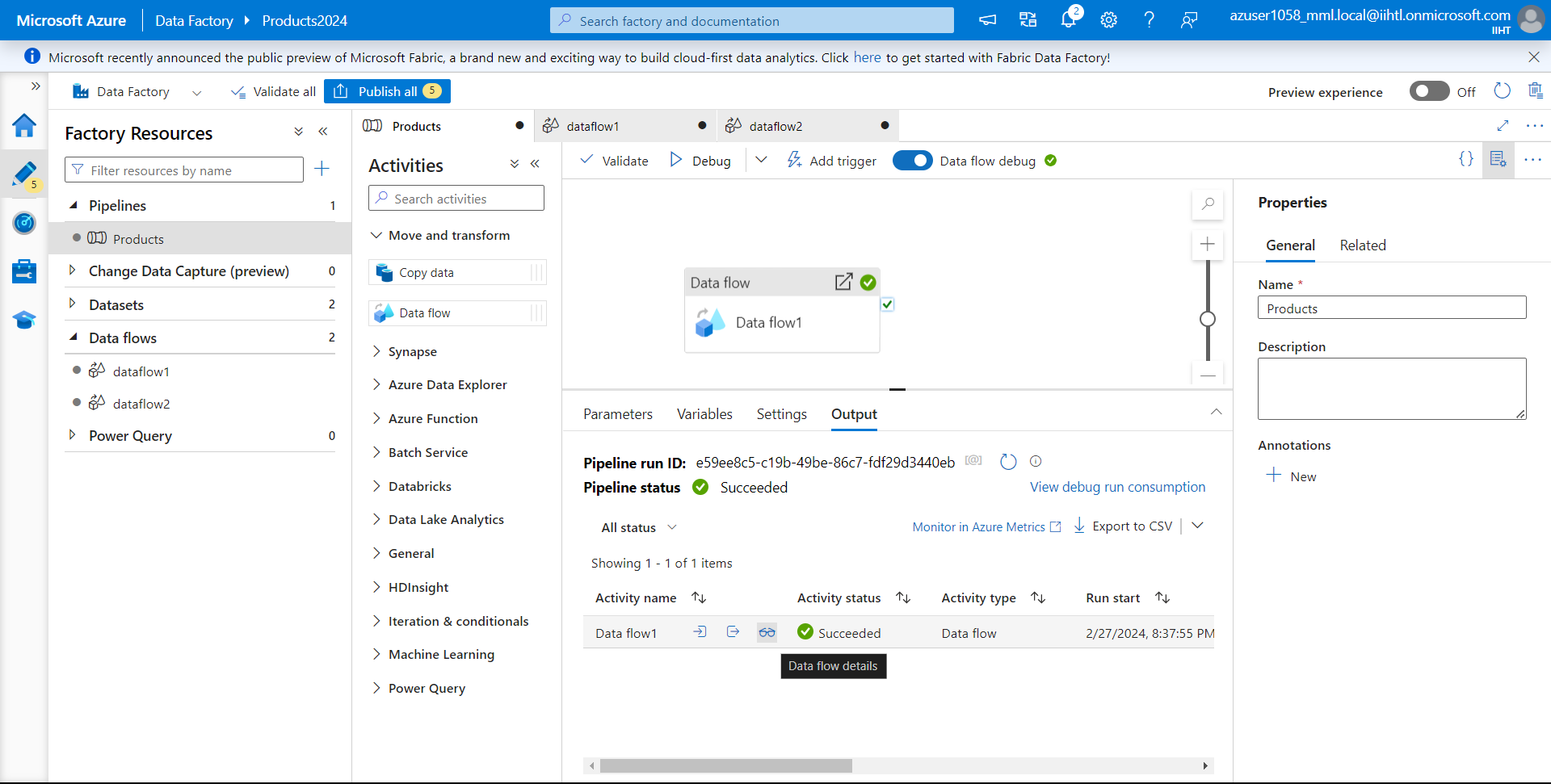
Go to Pipeline and click on Debug.



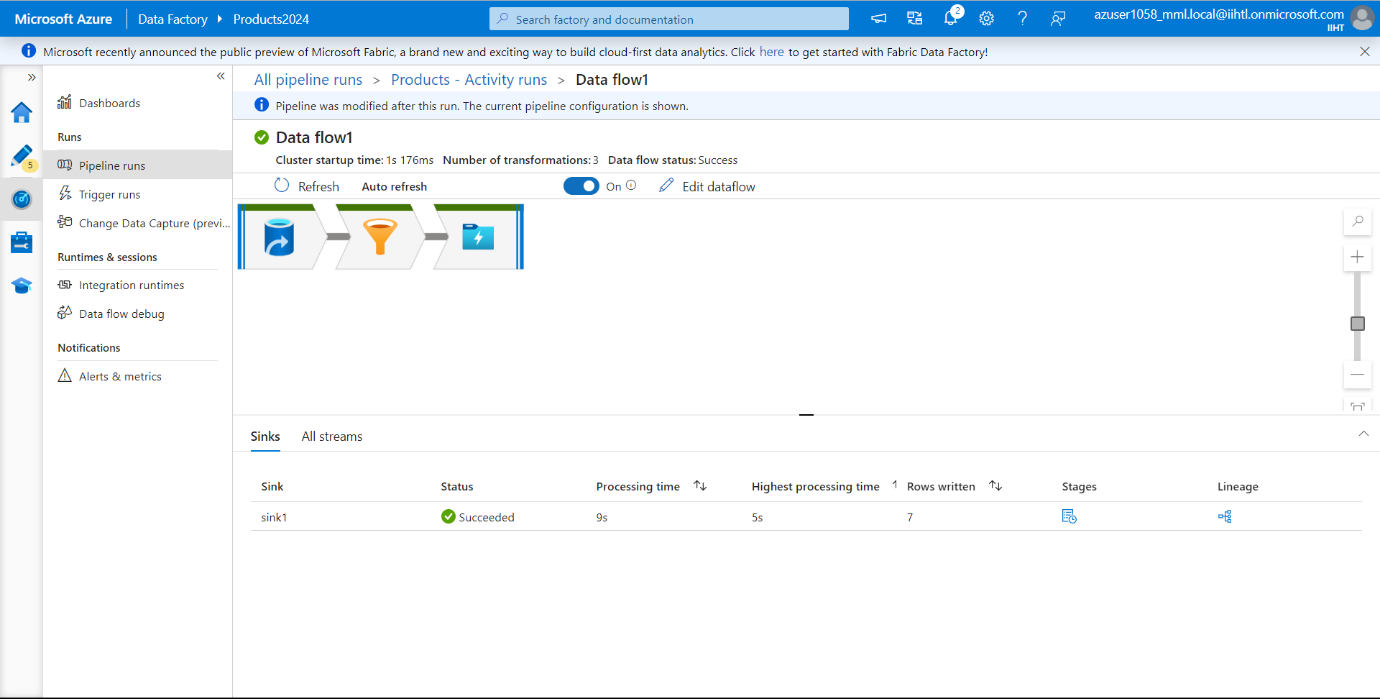
It takes few mins to initialize. We can track the progress in Output Tab, and here we can see that the run in successful in the output tab.



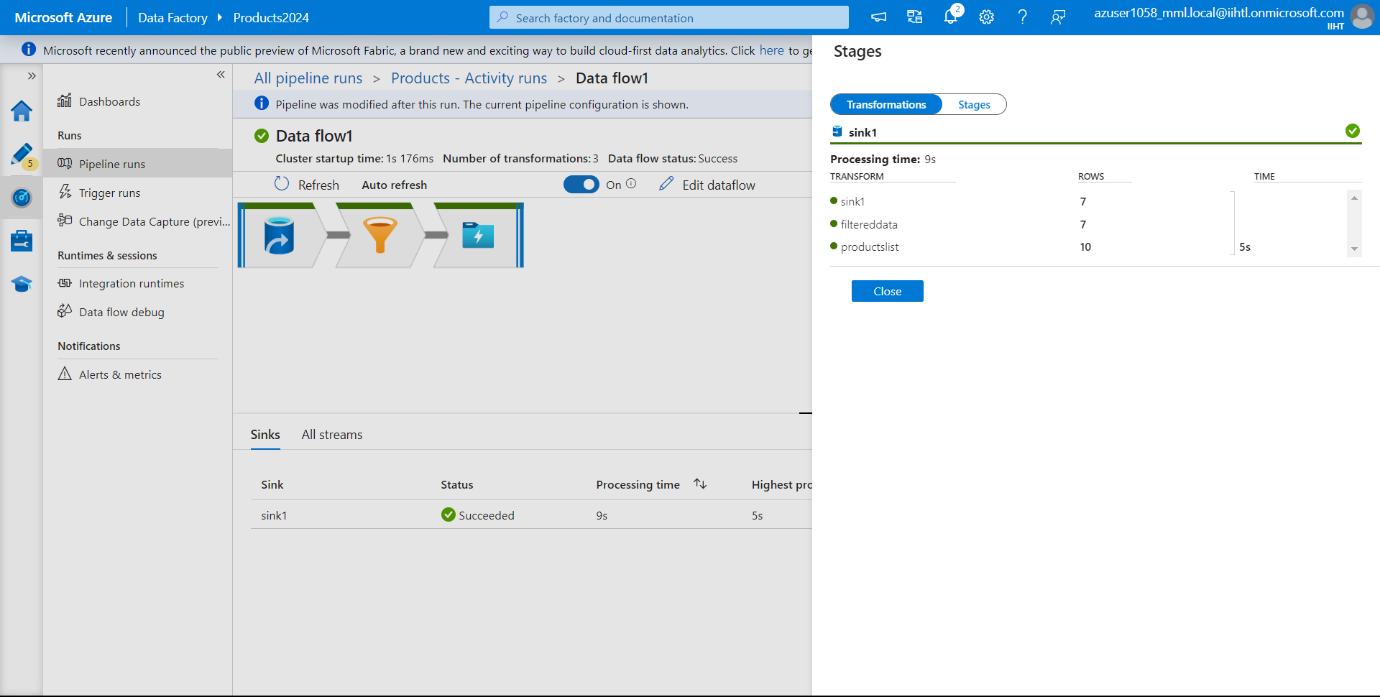
click on the eyeglasses icon to open the monitoring data.



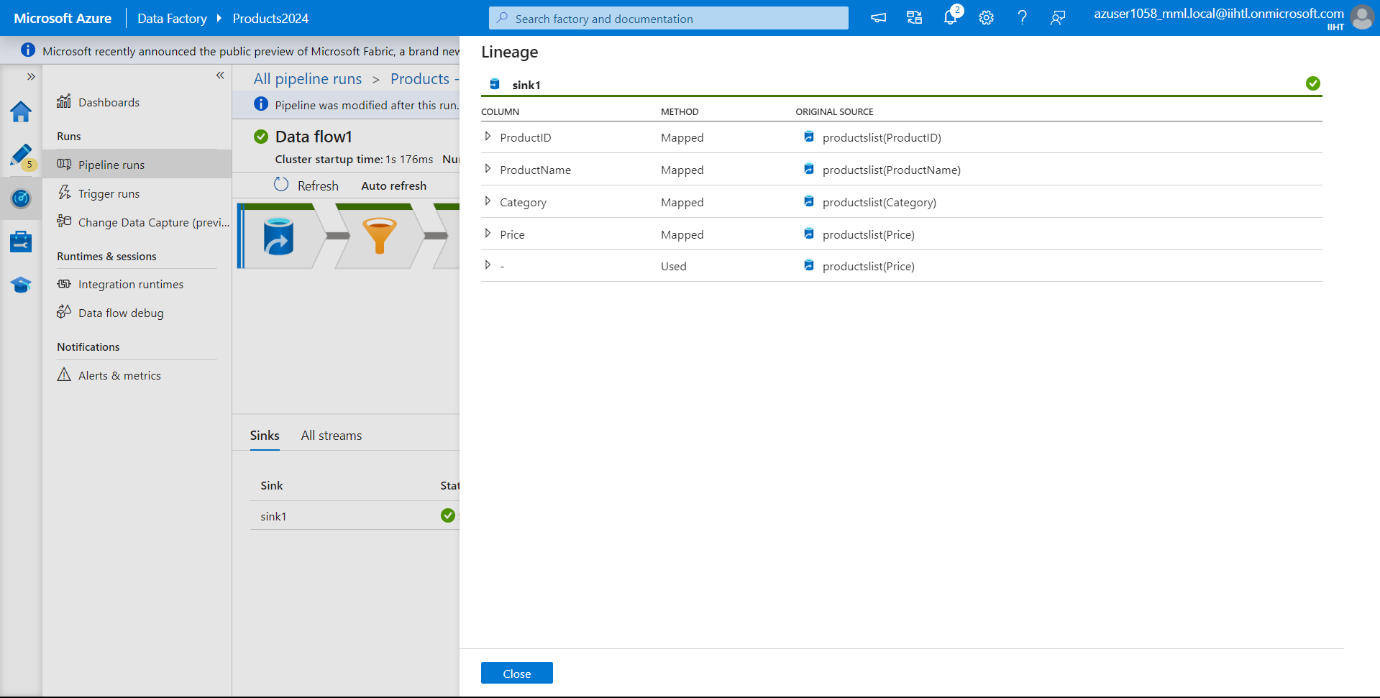
Here we can see the number of rows, processing time etc



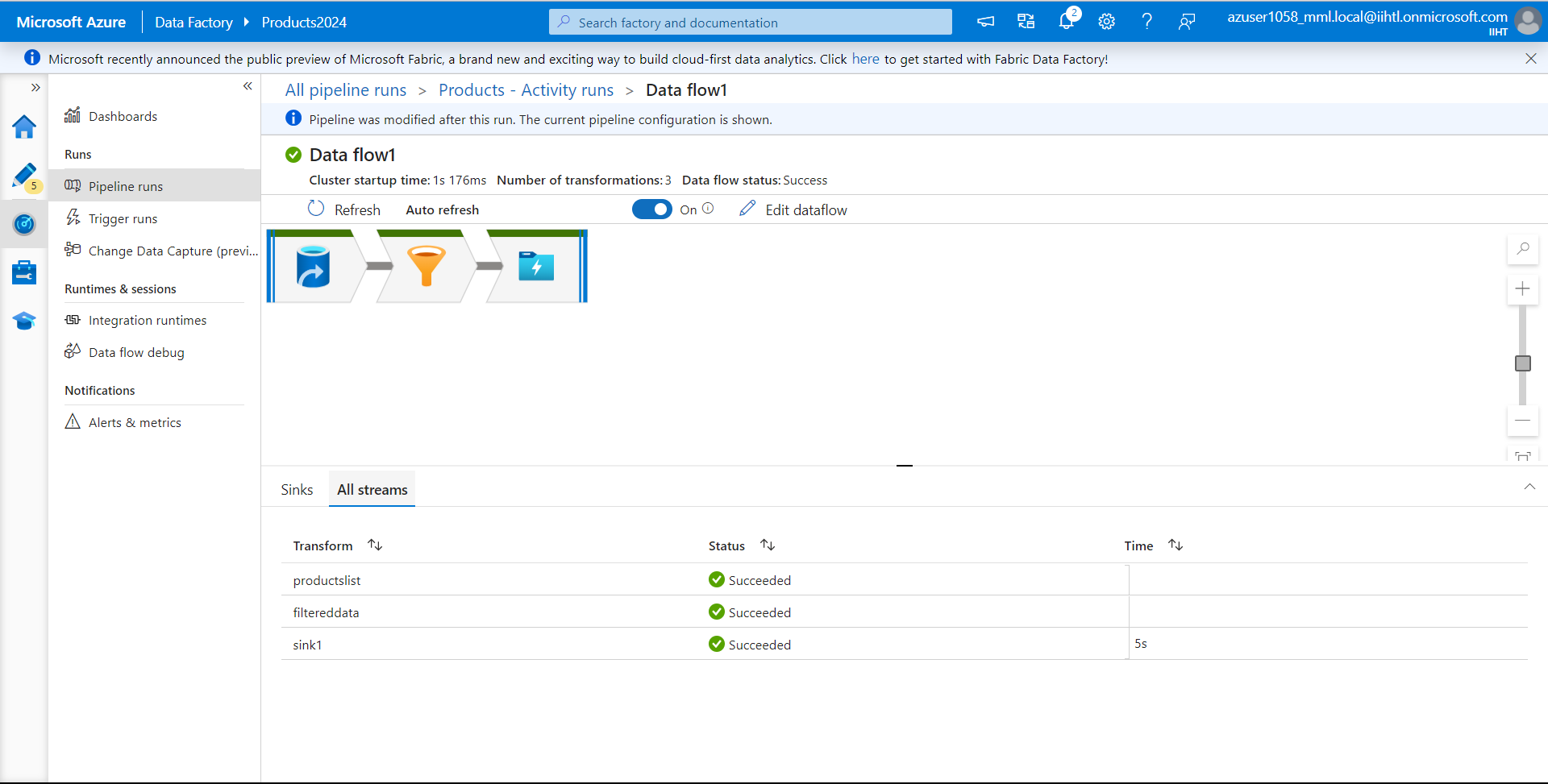
Click on stages to view the stages present in the data flow.



Now go to Linkage to observe the number of observations done.

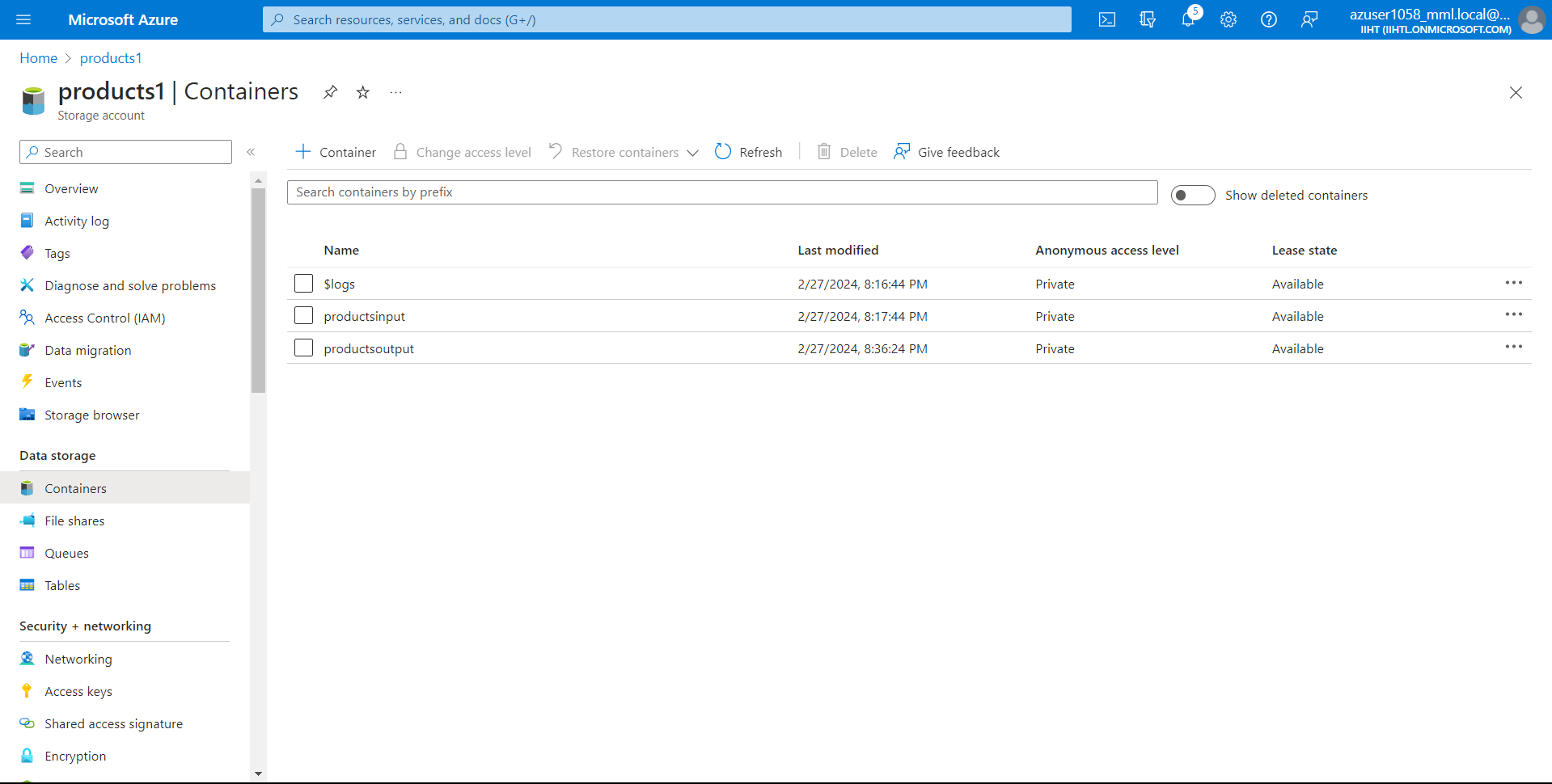


Go to All streams where it displays all activities which we have performed.

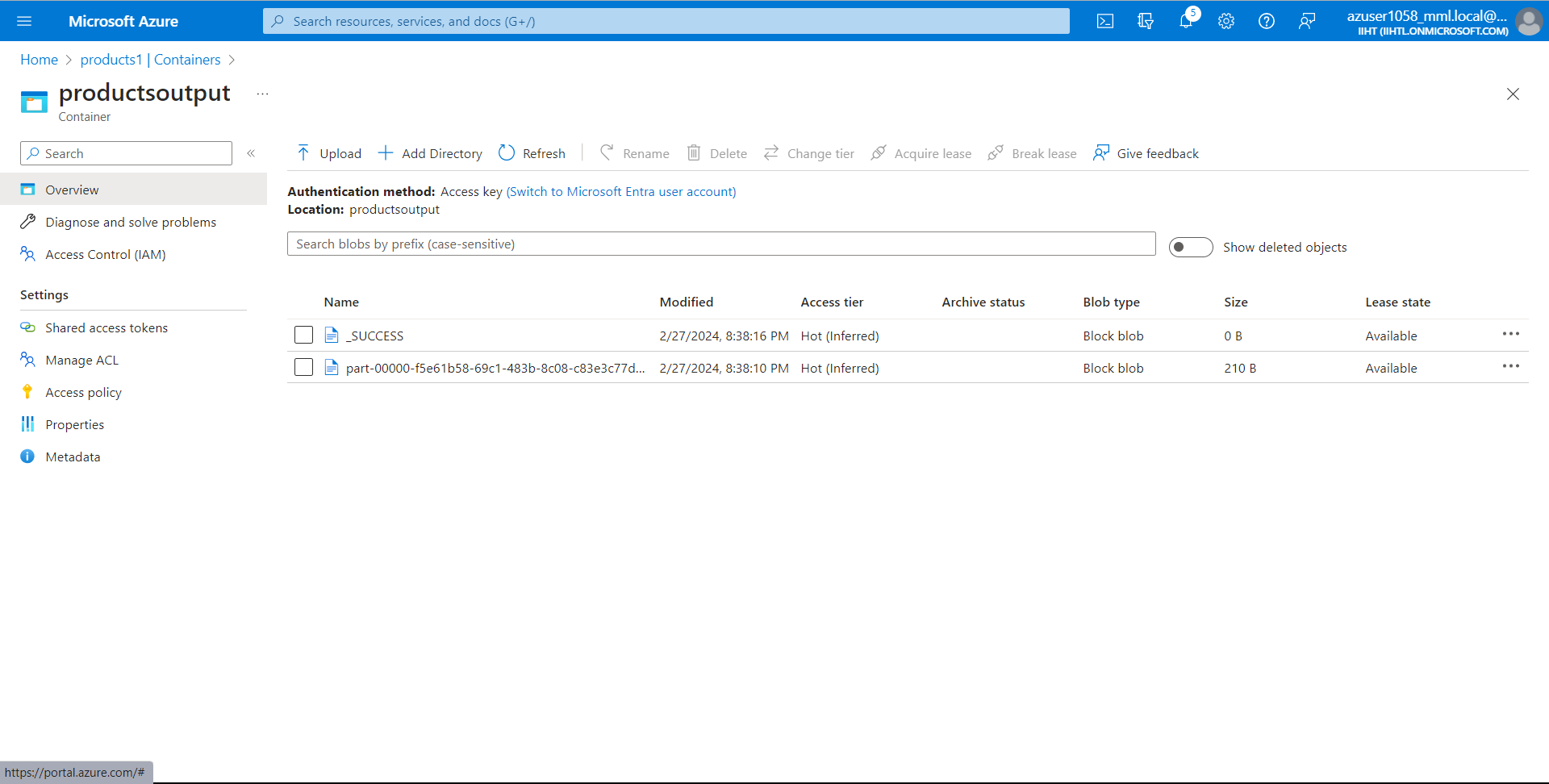


Now to check whether the Output file is stored in the Destination Path, we can verify the data is correct by checking in the blob storage.

Go to Destination container. Here we have already created a container named **“productsoutput”.**

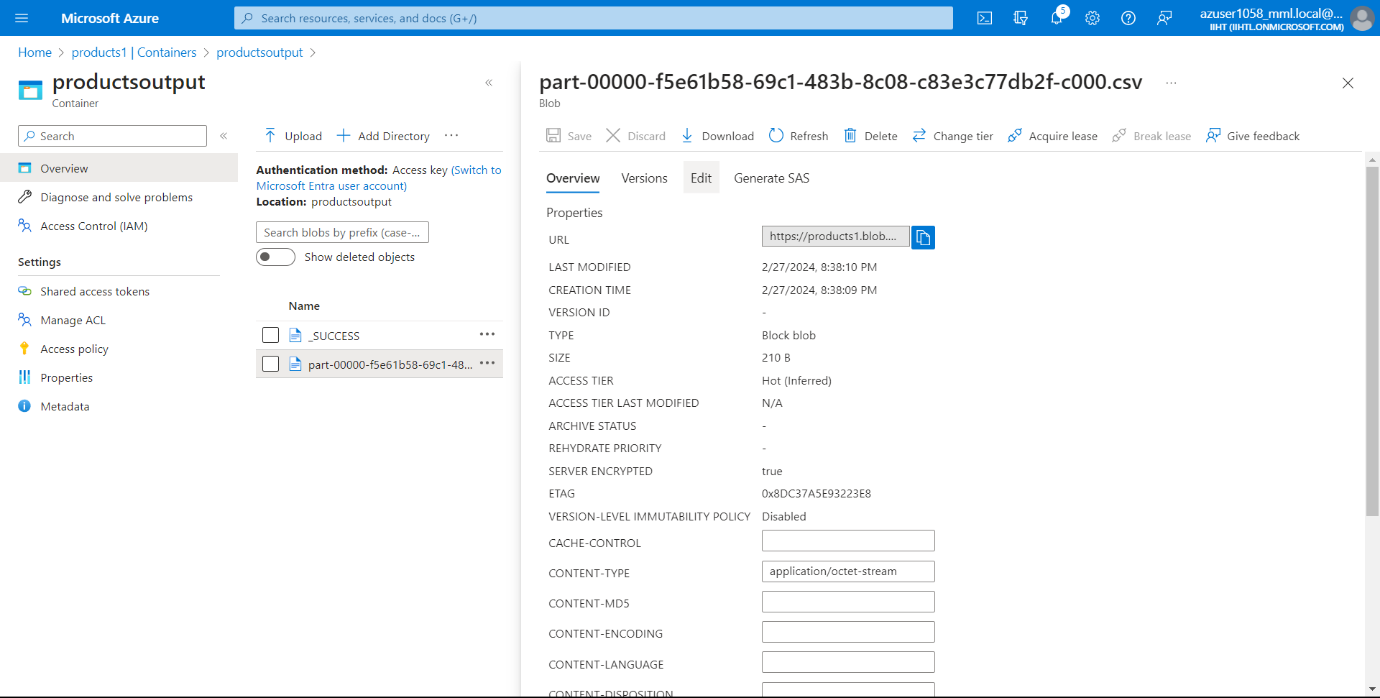


Go to productsoutput and here we can see two files have been added.



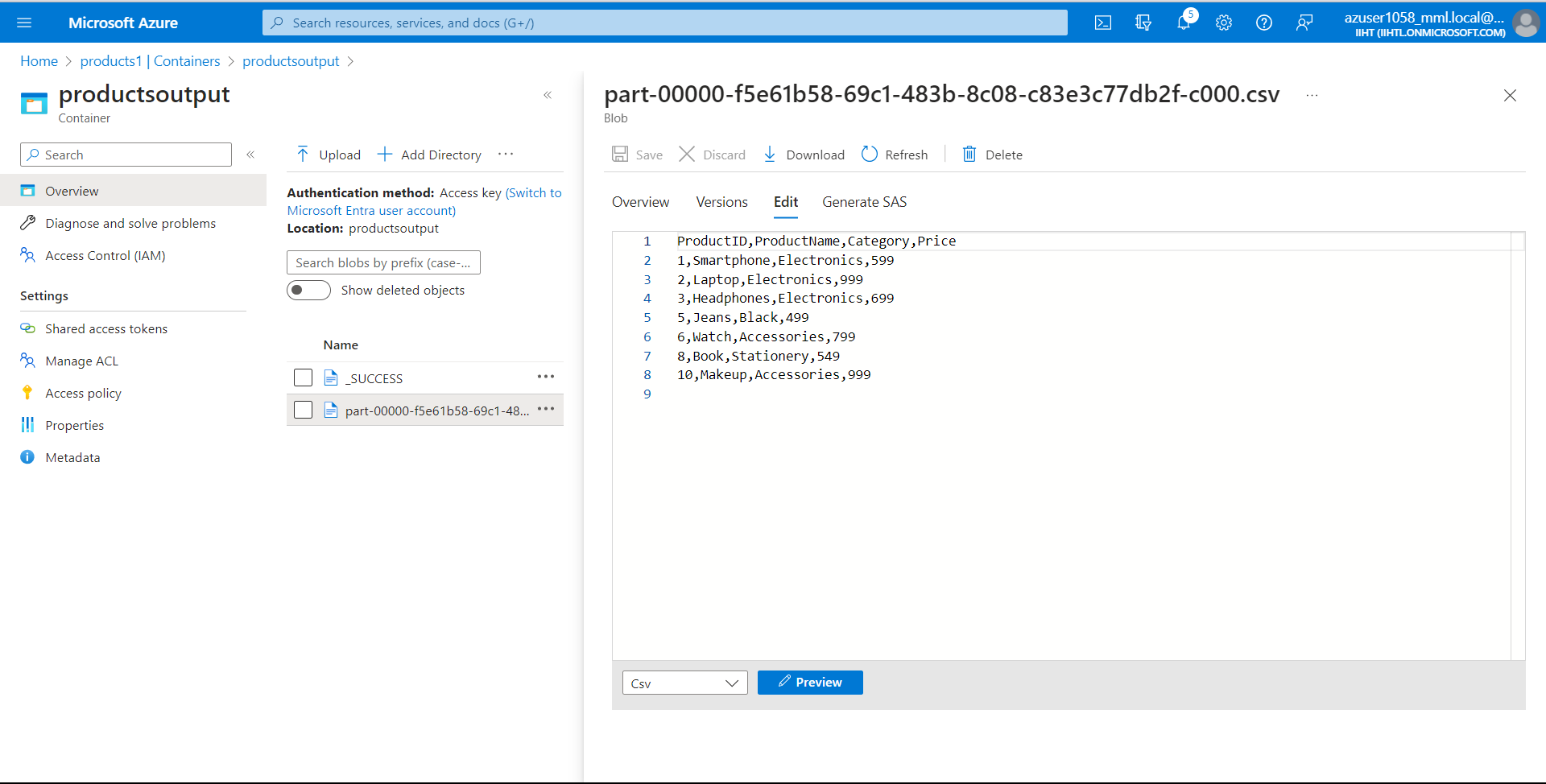
Open the second file which is present in this container.

Now click on Edit.



Here we can see the output data, for which we performed a filter expression where the Prices are Greater than or equal to 399 and less than or equal to 999.

**FINAL OUTPUT:**



**Conclusion:**

By leveraging Mapping Data Flow with Azure Data Factory activities, organizations can streamline their data transformation workflows, improve productivity, and accelerate insights delivery from raw data to actionable insights. This project provides a comprehensive overview of how Mapping Data Flow can be utilized within Azure Data Factory pipelines to achieve efficient data processing and transformation tasks in a cloud-based environment.