Table of Contents

[**Problem Statement:** 2](#_heading=h.gjdgxs)

[**Data Model:** 2](#_heading=h.30j0zll)

[**Resource Group Creation :** 3](#_heading=h.1fob9te)

[**Azure Synapse Analytics :** 3](#_heading=h.3znysh7)

[**Azure Blob storage :** 4](#_heading=h.2et92p0)

[**Azure blob container:** 4](#_heading=h.tyjcwt)

[**Azure ADLS GEN2 :** 5](#_heading=h.3dy6vkm)

[**Data storage in respective containers :** 5](#_heading=h.1t3h5sf)

[**Creation of Linked Service :** 6](#_heading=h.4d34og8)

[**Approach For WH creation :** 6](#_heading=h.2s8eyo1)

[**Using Dedicated SQL Pool** 7](#_heading=h.17dp8vu)

[**Using Spark Pool :** 9](#_heading=h.3rdcrjn)

[**Jupyter Notebook script :** 10](#_heading=h.26in1rg)

[**Serverless SQL pool for analysis :** 10](#_heading=h.lnxbz9)

[**Final Analysis :** 11](#_heading=h.35nkun2)

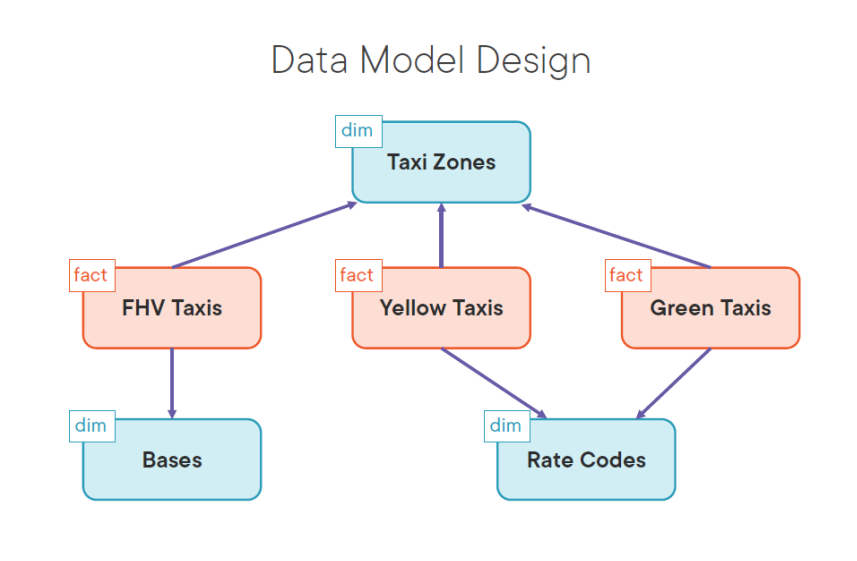
## **Problem Statement:**



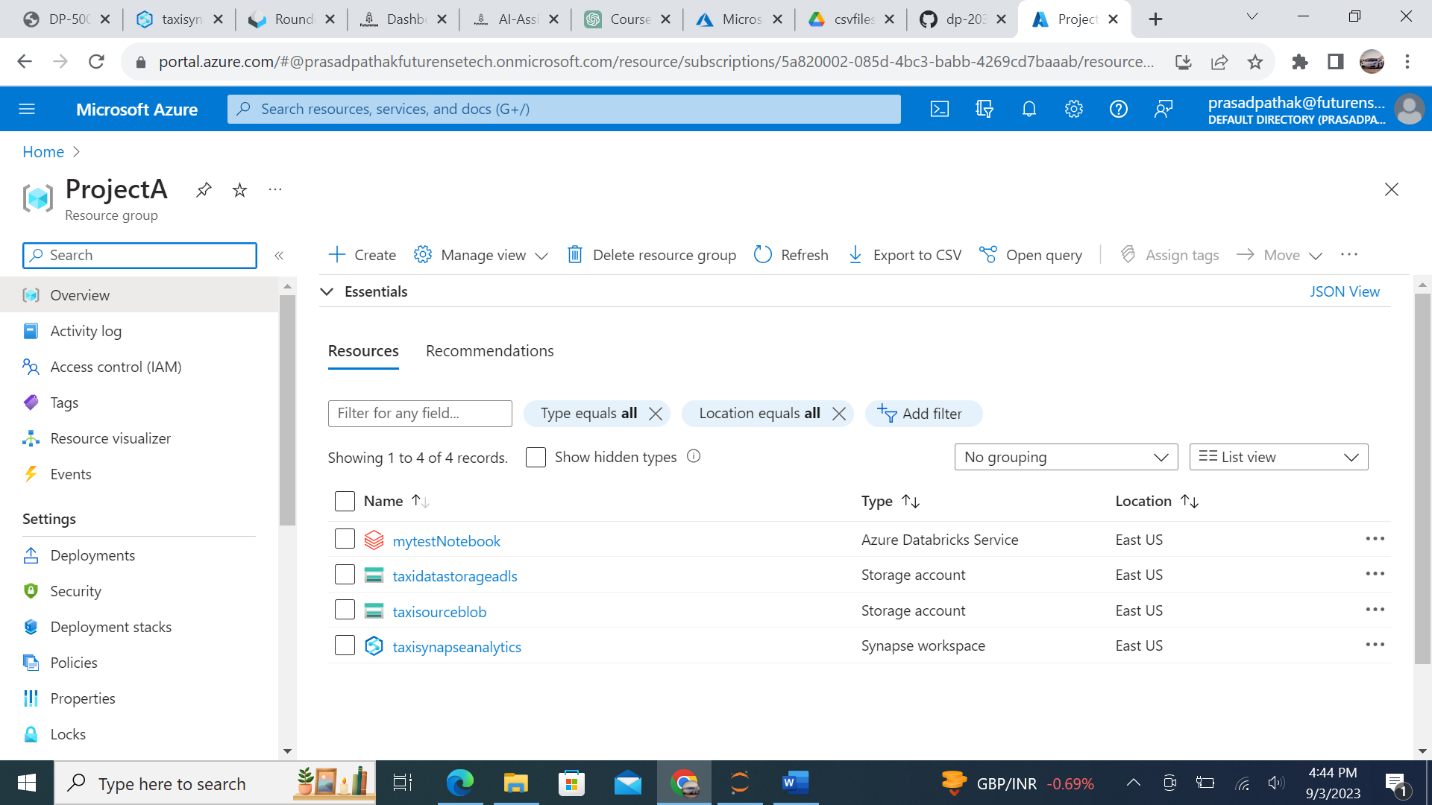
Assume that your organization is responsible for processing New York City taxi service data. There are three types of taxis in New York, yellow taxis, green taxis, and for-hire vehicles, or FHVs. FHVs include app-based taxis like Uber, Via, and Lyft, and each of them dispatches more than 10,000 trips per day, and all of them operate in different ways. The company collects ride-related information like pickup and drop time, details of pickup and drop location, trip distance, passenger count as reported by the driver, if it's a solo or a shared trip, as well as payment-related information. Initially, the system was working well, but now the company is seeing exponential increase in data volumes, and each taxi type uses different schema and shares data in different file formats like CSV, JSON, and Parquet. The new requirement is to compare taxis against one another and build granular and new aggregated reports. And the users are looking for faster turnaround time in terms of data processing and new requirement implementation. And of course, raw data must also be available for analysis. That's why to address these challenges your organization has decided to build a data lake house.

## **Data Model:**

We have three facts, one for FHV taxis, second for yellow taxis, and third for green taxis. FHV taxi's fact is linked to bases dimension, which tells us the dispatch-based location of the taxi. On the other hand, yellow and green tax facts are linked to the dimension rate codes, which tells us if the trip is a shared trip, solo trip, or trip to any specific airport. And all the three facts are linked to taxi zones dimension, which will tell us the pickup location and drop location details for each trip. Implement the data warehouse using both spark pool and SQL pool and list the Pros and Cons of each implementation.

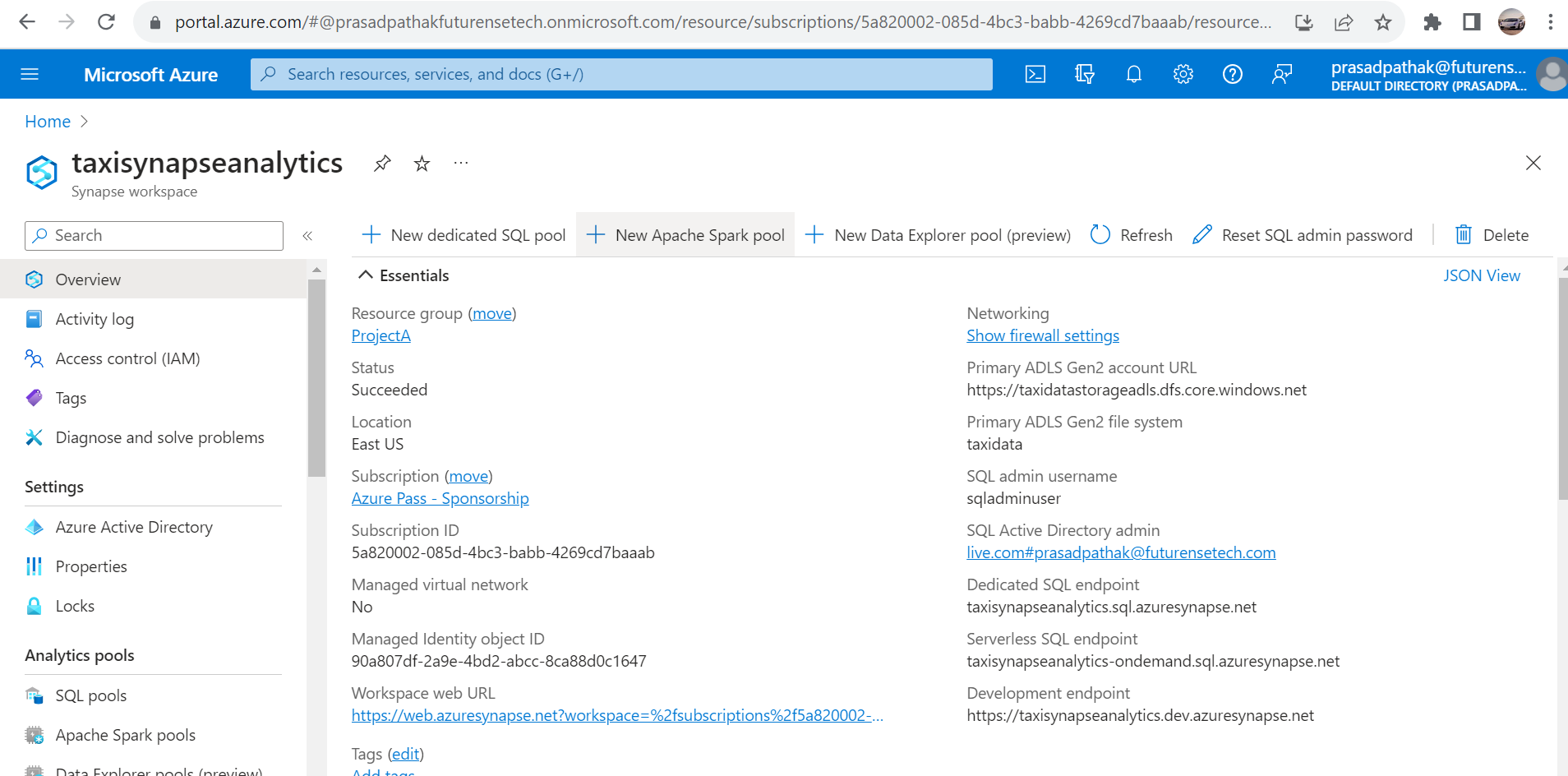


## **Resource Group Creation :**

Created A resource group for PROJECT-A with name ProjectA:

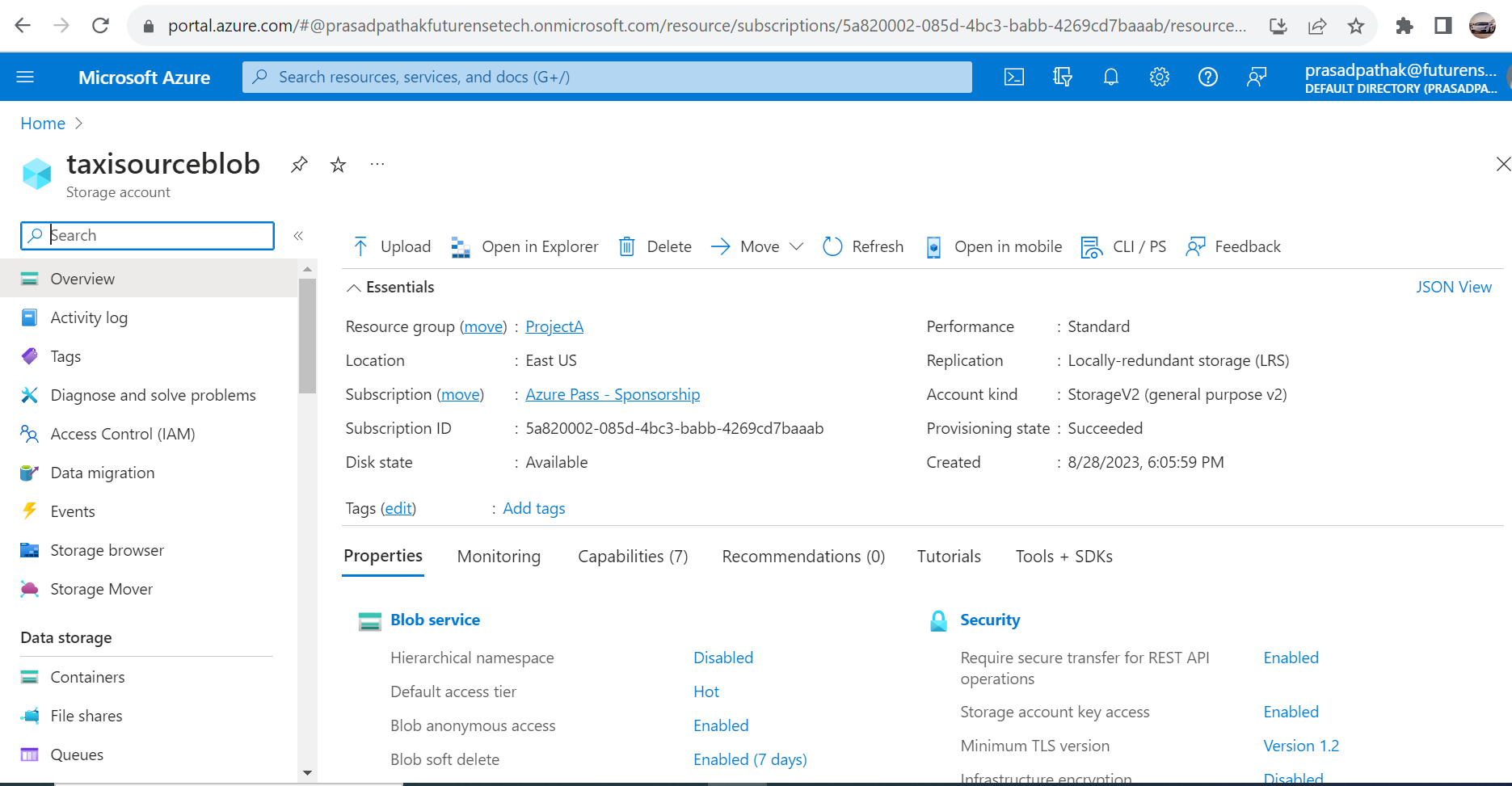
## **Azure Synapse Analytics :**

Created Azure Synapse Analytics instance :



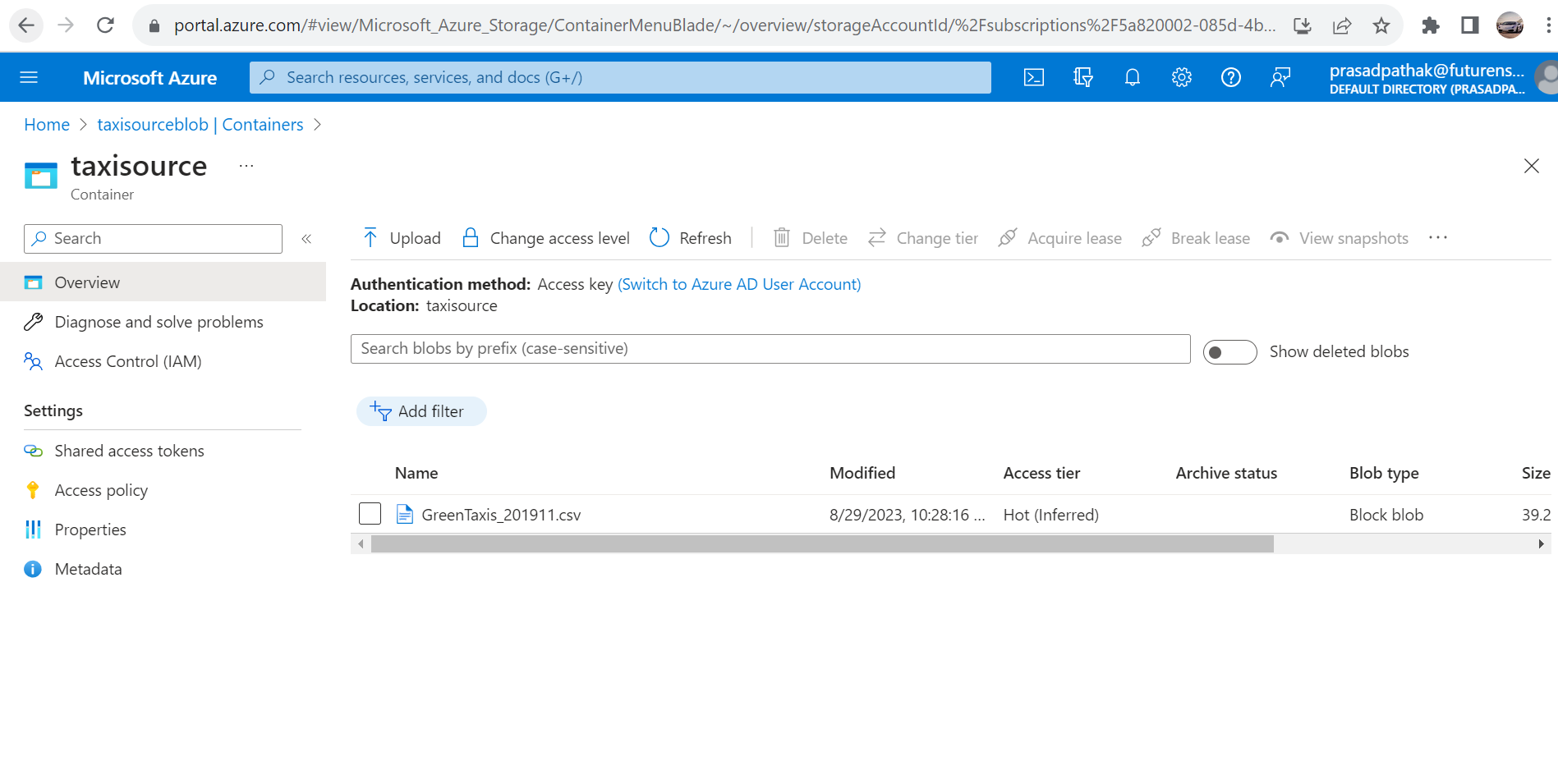
## **Azure Blob storage :**

Created Blob Storage account for Blob object store :



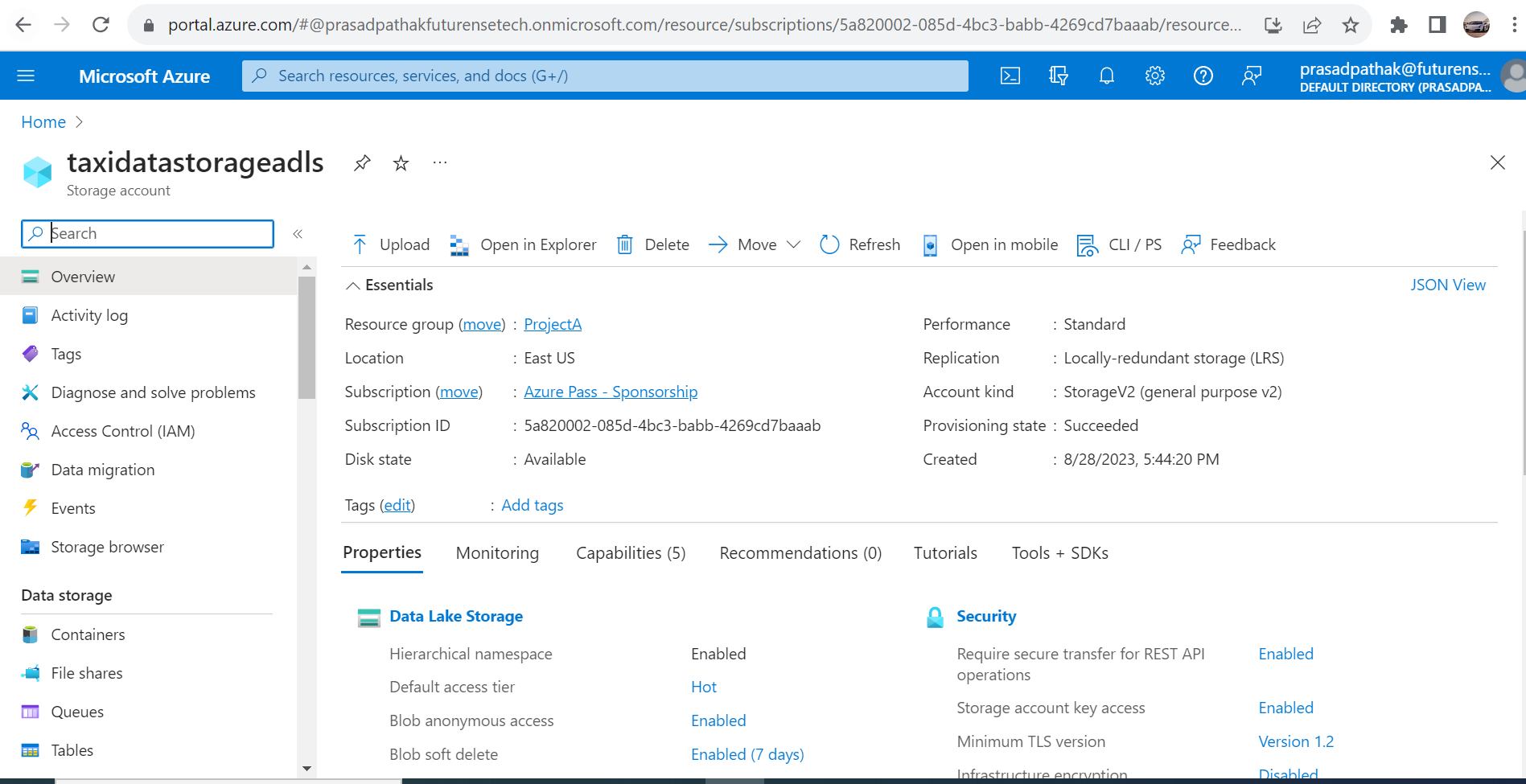
### **Azure blob container:**

Created Container inside Blob storage to store the data file :



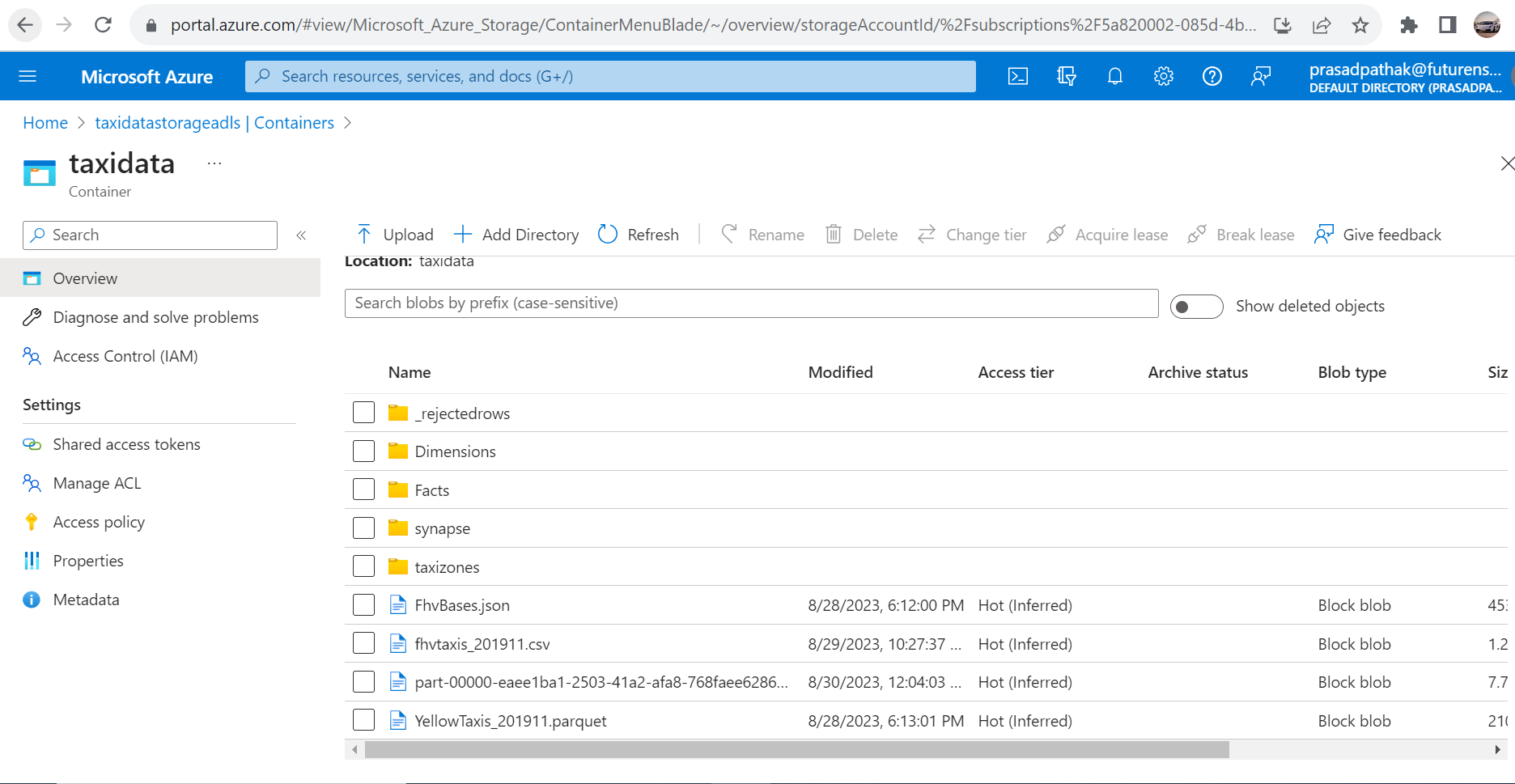
## **Azure ADLS GEN2 :**

Created Data lake storage account for storing Hierarchical data:



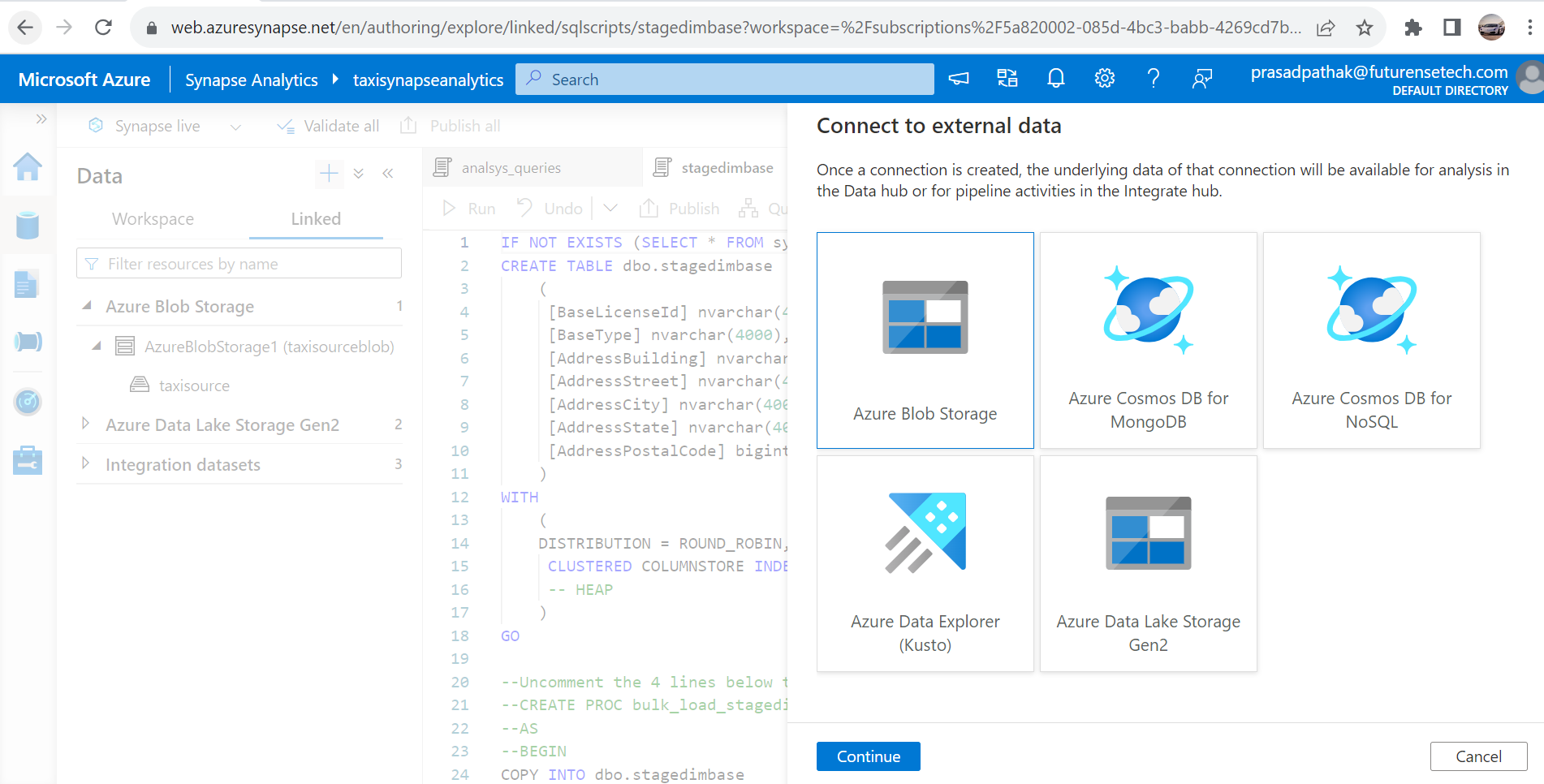
### **Data storage in respective containers :**

Stored the data as per requirement in separate container :



## **Creation of Linked Service :**

Created Linked service for Blob storage account for getting the data from blob storage :

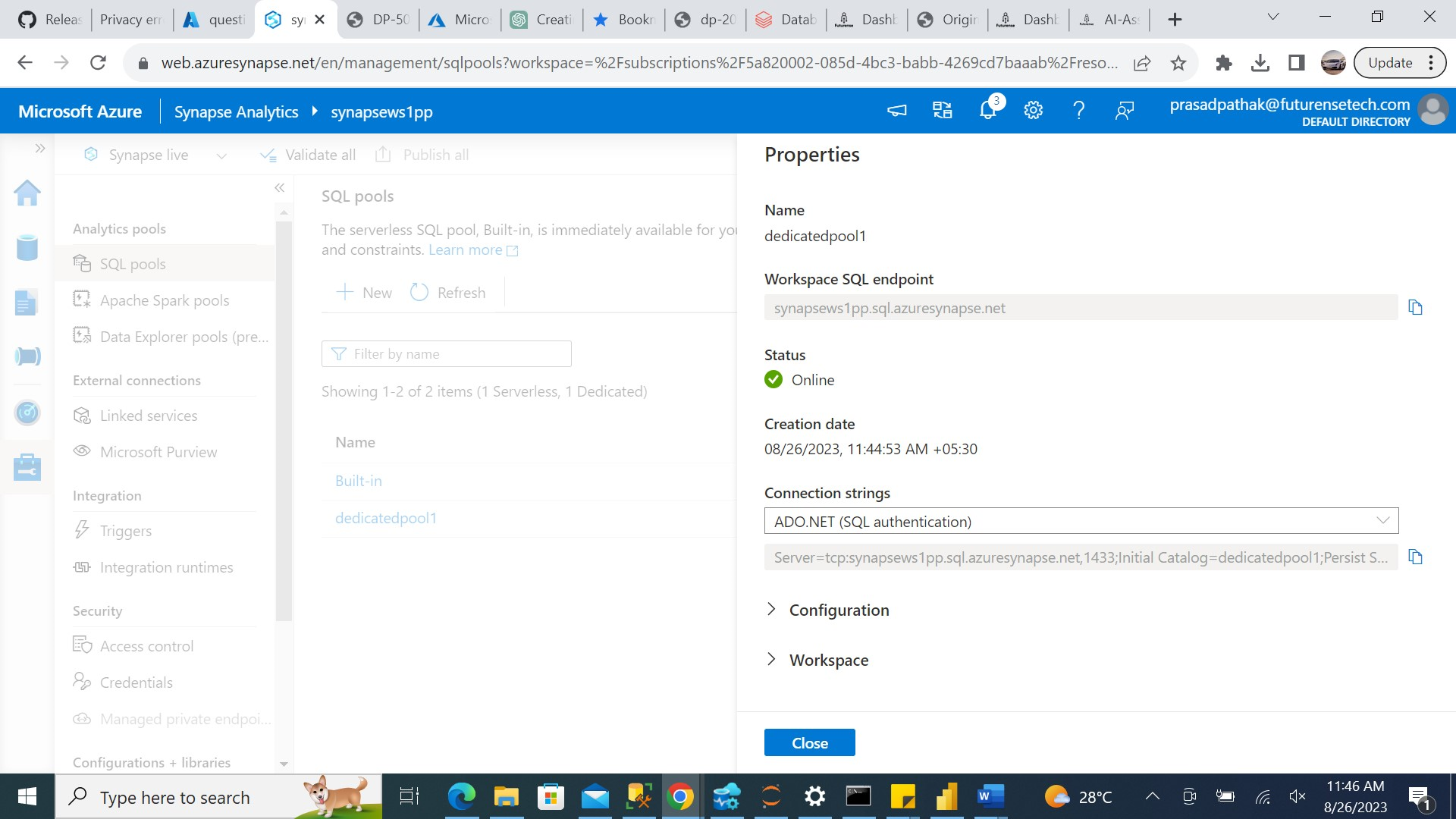


## **Approach For WH creation :**

There are 2 approach that we can follow to create the Data warehouse and design Fact and Dimension table :

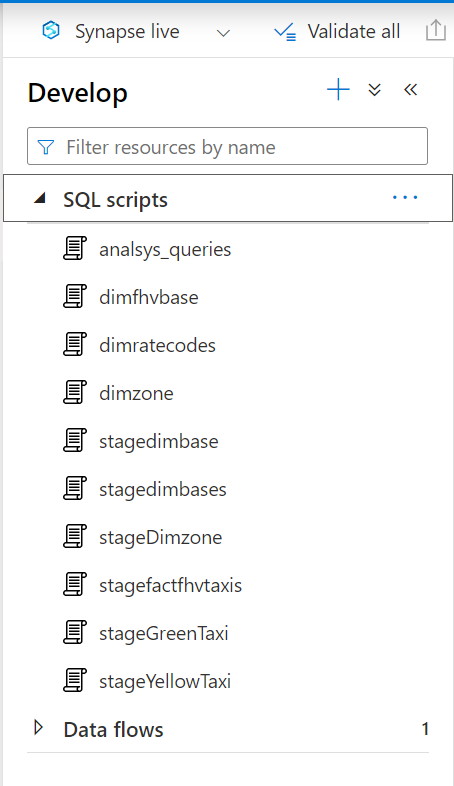
### **Using Dedicated SQL Pool:**

Created Dedicated SQL pool for creating Fact and Dimension tables



The Approach that I used is loading the data initially into staging table and then loading the data into main table .

Following are the scripts used for creation of Dimension and Fact table in dedicated SQL pool warehouse:



#### Staging table Script :



#### Dimension table Script :

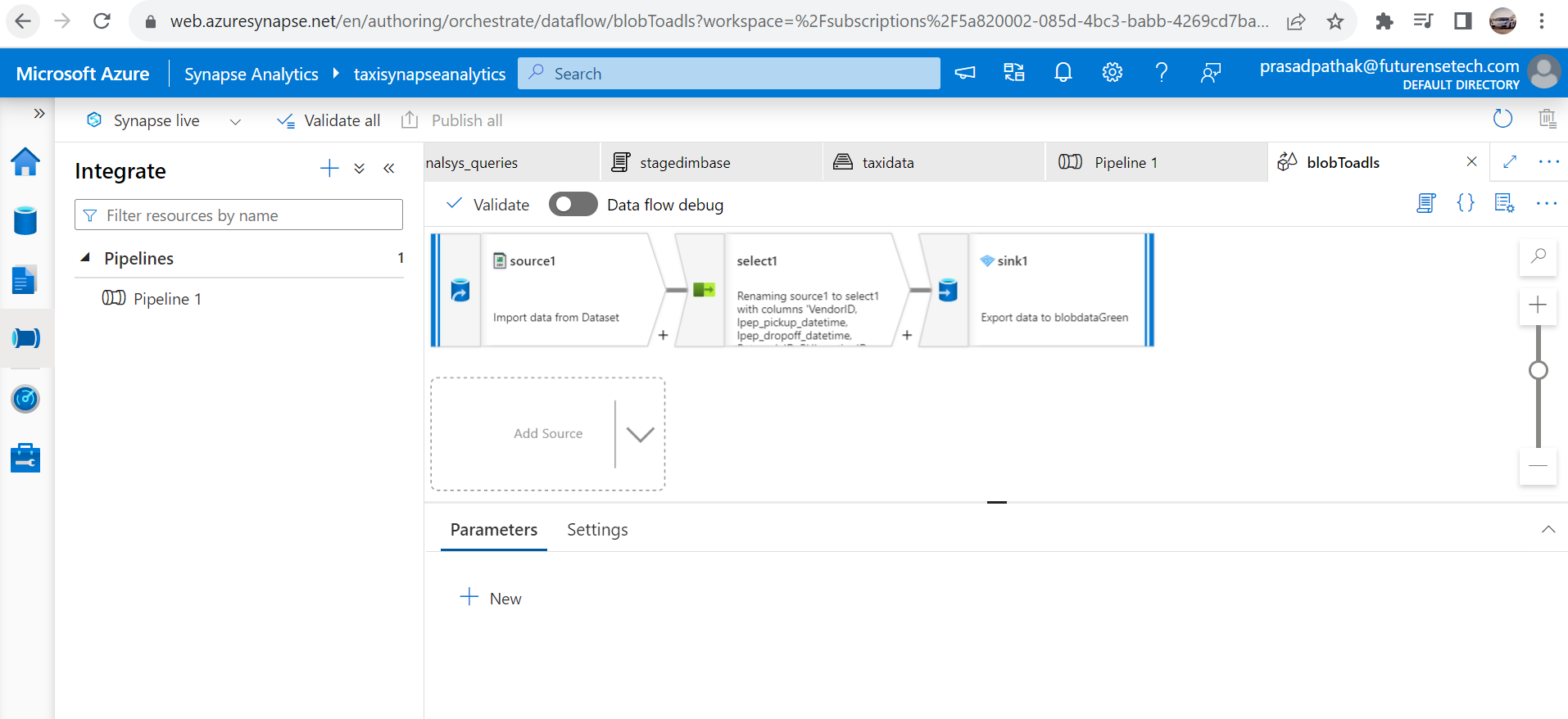


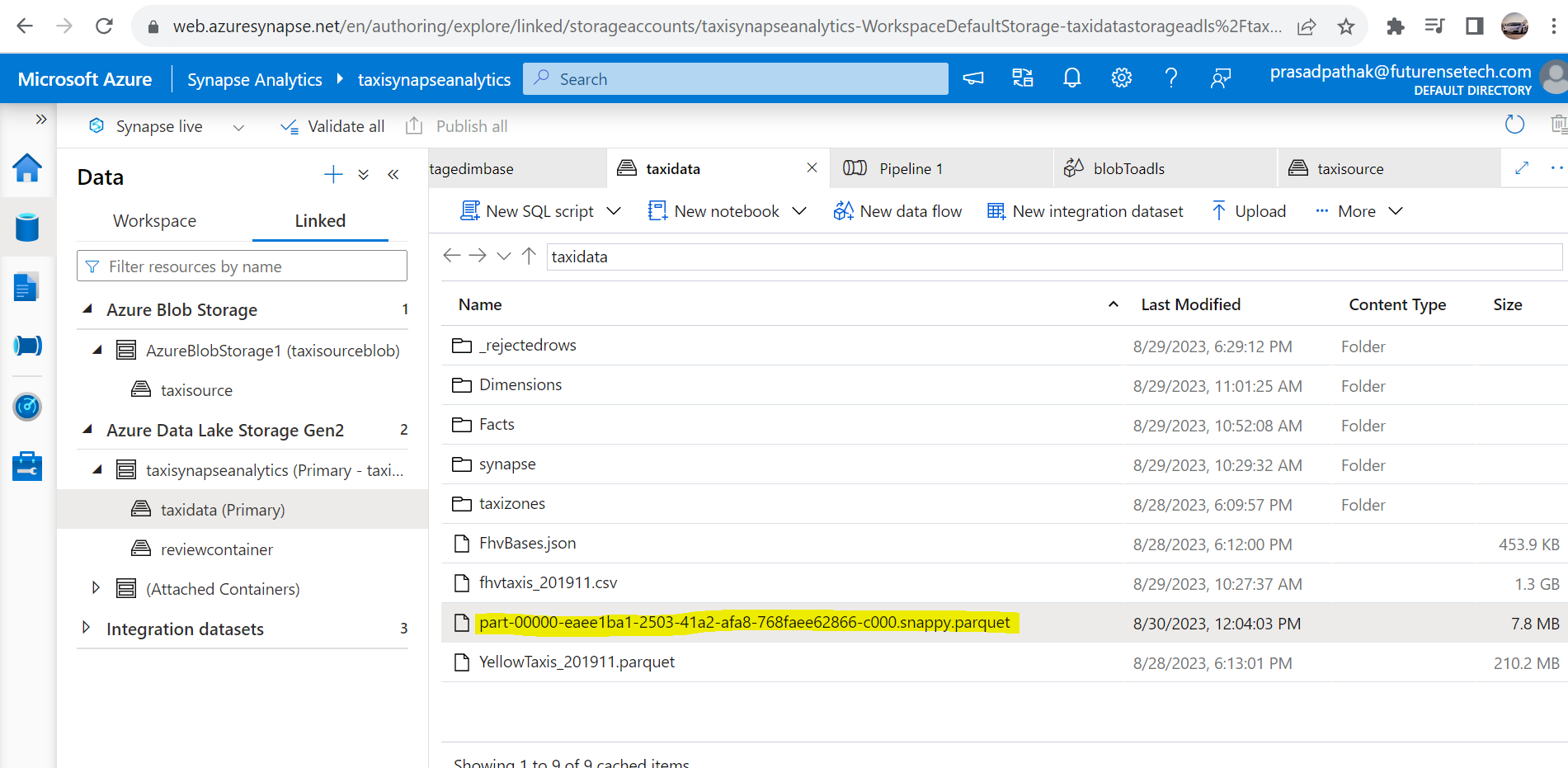
#### Fact Table Script :



#### DataFlow pipeline for JSON file :

While establishing the Green taxi table, the data stored in Blob storage was initially transferred to ADLS Gen2 to facilitate its loading into the data warehouse. Subsequently, ADF was employed to transform the data within a dataflow, ultimately saving it as a parquet file.

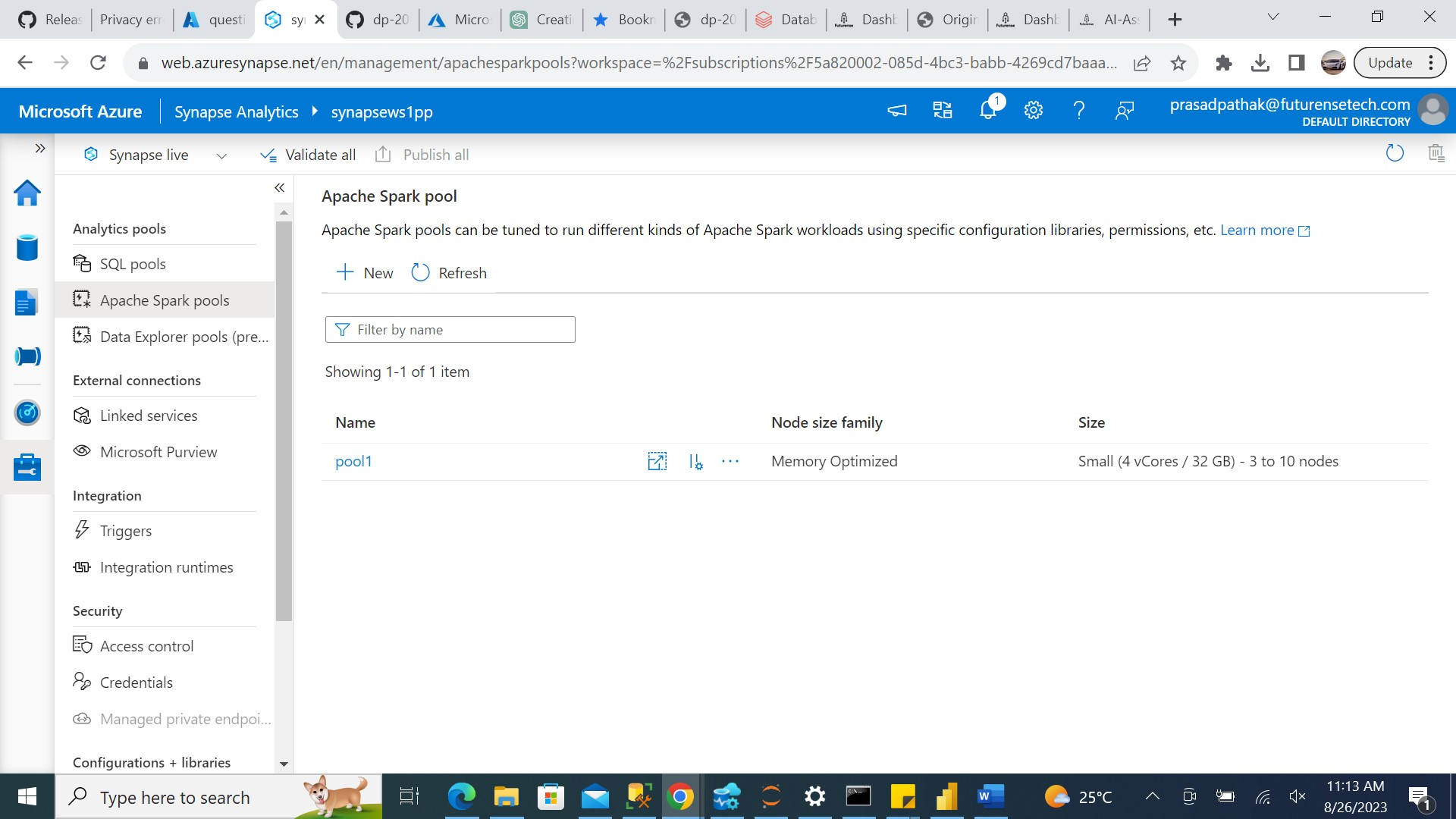




## **Using Spark Pool :**

1. Using Spark pool to load the data warehouse :

For this created the spark Pool



### **Jupyter Notebook script :**

Utilize the Spark pool within this notebook to conduct an analysis of the data. Within the notebook, establish the lake database and store both fact and dimension tables within it to enable further analysis.Utilize the Spark pool within this notebook to conduct an analysis of the data. Within the notebook, establish the lake database and store both fact and dimension tables within it to enable further analysis.



### **Serverless SQL pool for analysis :**

Subsequently, employed the integrated **serverless SQL pool** to analyze the data residing in the Fact and Dimension tables. Below is the SQL script utilized for analytical purposes.



## **Final Analysis :**

I've experimented with both approaches, employing both Spark Pool and Dedicated SQL Pool.

When using the Dedicated SQL Pool, I found it necessary to also utilize ADF for loading Blob data. Additionally, the Dedicated pool incurs charges on an hourly basis, making it a more costly option.

In contrast, utilizing the Spark Pool proved to be straightforward for loading various types of data from all storage accounts. The cost associated with the Spark Pool is determined by the queries executed on the Spark engine.

Considering these factors, it is advisable to opt for the Spark Pool when creating Fact and Dimension tables in the data warehouse. Moreover, the option of using the cost-effective Serverless Pool is available within the Spark Pool, making it a compelling choice in comparison to other alternatives.