Data Lake Analytics



Project Overview

In today's data-driven world, the ability to efficiently process and analyse large volumes of data is crucial for businesses to gain insights and make informed decisions. This project aims to leverage the power of Azure Databricks and PySpark to perform data analytical tasks, including Extract, Transform, and Load (ETL) operations, on massive datasets.

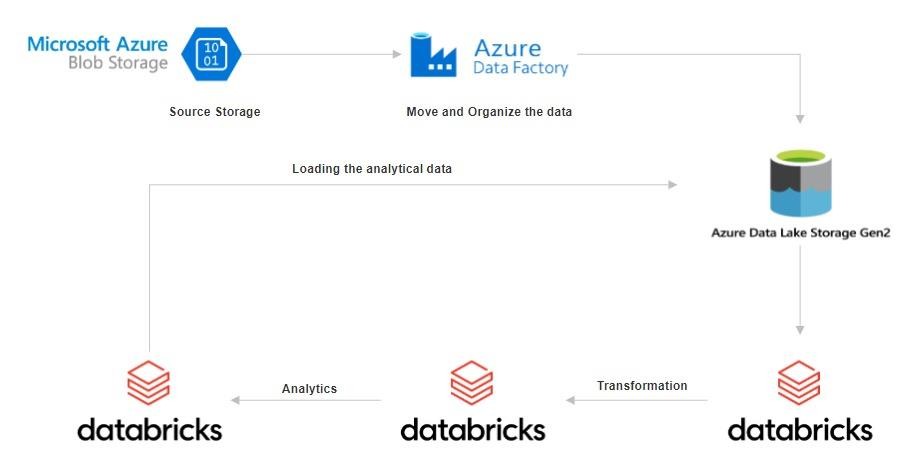
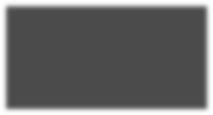


About Project

This project revolves around the seamless orchestration of data movement, transformation, and analytics by integrating Azure Data Factory (ADF) and Azure Databricks. The workflow involves moving data from Azure Blob Storage to Azure Data Lake Storage Gen2 (ADLS Gen2) using ADF, mounting the ADLS Gen2 within Databricks notebooks for data transformations and analytics, and finally, persisting the analytical results back to ADLS Gen2.



Architectural Diagram



Key-Components/Requirements of the projects

# Azure Databricks:

* + Azure Databricks provides a cloud-based platform for big data analytics and machine learning. It offers a collaborative environment for data engineers, data scientists, and analysts to work together seamlessly.
  + Databricks provides managed Spark clusters, eliminating the need for infrastructure management and allowing teams to focus on data processing tasks.

# PySpark**:**

* + PySpark is the Python API for Apache Spark, a powerful open-source framework for distributed data processing. PySpark simplifies development tasks by providing a Python interface to Spark's capabilities.
  + With PySpark, developers can write concise and expressive code to perform complex data transformations, aggregations, and analytics on large datasets.

# Azure Data Factory:

* + Data Factory (ADF) allows users to create, schedule, and manage data pipelines that can move data between various supported data stores. ADF provides a scalable, fully managed platform for orchestrating and automating data workflows.
  + It has It has many features such as data orchestration, seamless integration with Azure services, hybrid data integration, security, scalability, monitoring, metadata management, and cost management, enabling users to create, automate, and manage data pipelines for diverse data workflows.



Azure Resources Used for this Project

# Azure Blob Storage

* + This is where the raw data is stored. Azure Blob Storage integral to Microsoft Azure's storage service, is a cloud-based solution tailored for managing vast amounts of unstructured data, encompassing both text and binary data. Termed "Blob" for "Binary Large Object," it signifies a compilation of binary data treated as a singular entity within a database.

# Azure Data Factory

* + Here, we use azure data factory for moving and organizing the data from blob storage to azure data lake gen 2. It provides essential tools to create pipelines for the movement of data.
  + It also provides Cloud-based service for orchestrating, automating data workflows, enabling seamless movement, transformation, and integration across diverse sources.

# Azure Data Lake Storage Gen2

* + This is where the raw data is loaded by ADF and then the analytical data is also moved and organized here. Azure Data Lake Storage Gen2 provides a scalable and secure platform for storing large volumes of data. It enables us to manage, access, and analyze data effectively.

# Azure Databricks

* + Azure Databricks are used to develop essential notebooks that contains the pyspark code which perform transformation and analytical operations on the data based on the business requirement.

# Databricks Cluster

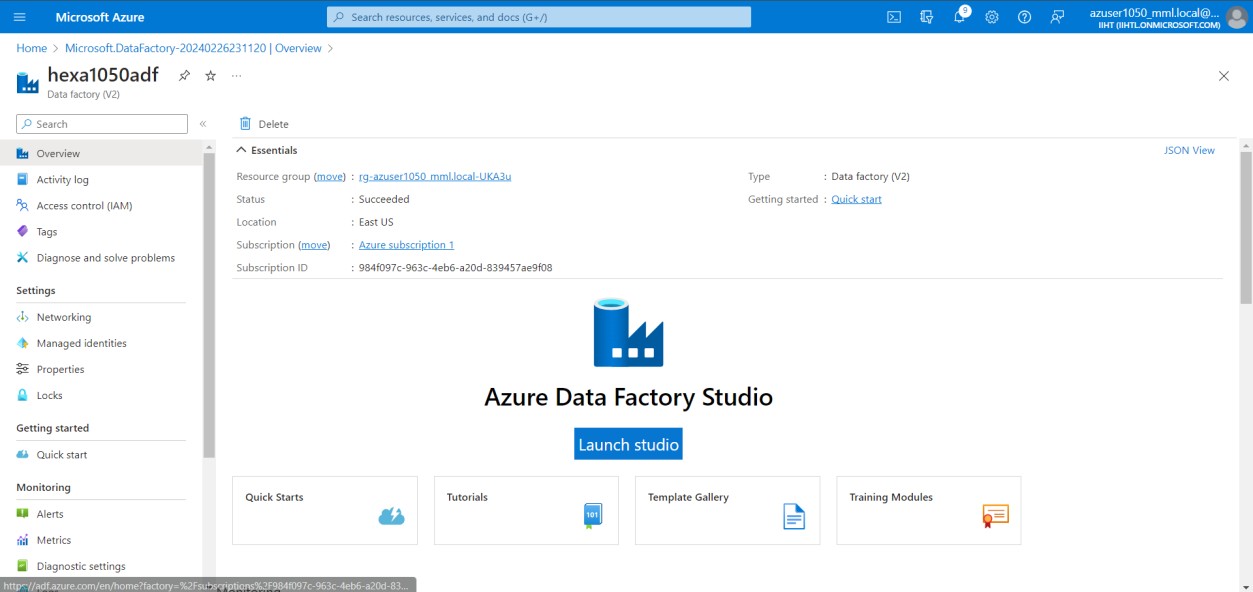
* + An Azure Databricks cluster process the data depending on the user instructions in the Azure Notebook. It serves as a computational resource facilitating the processing of extensive data and execution of analytics workloads through the Apache Spark platform within the Microsoft Azure cloud.
  + Azure Databricks are used to develop essential notebooks that contains the pyspark code which perform transformation and analytical operations on the data based on the business requirement.



How It works

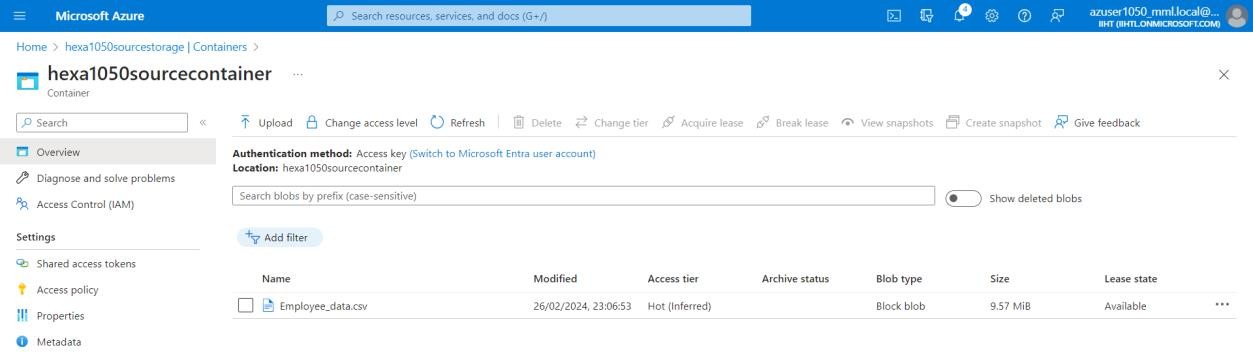
# Setting Up Azure Databricks Factory:

* + Sign in to the Azure portal and create an Azure Data Factory. Configure the settings, including pricing tier, region, and workspace name.

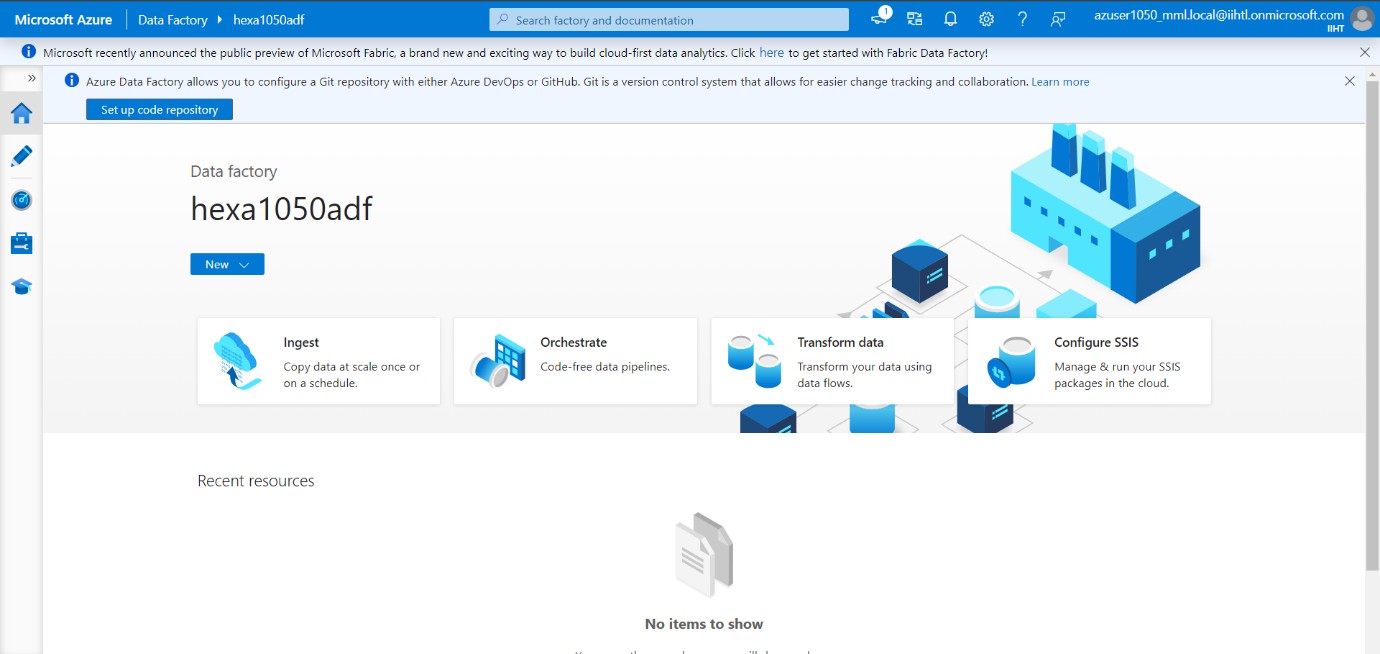


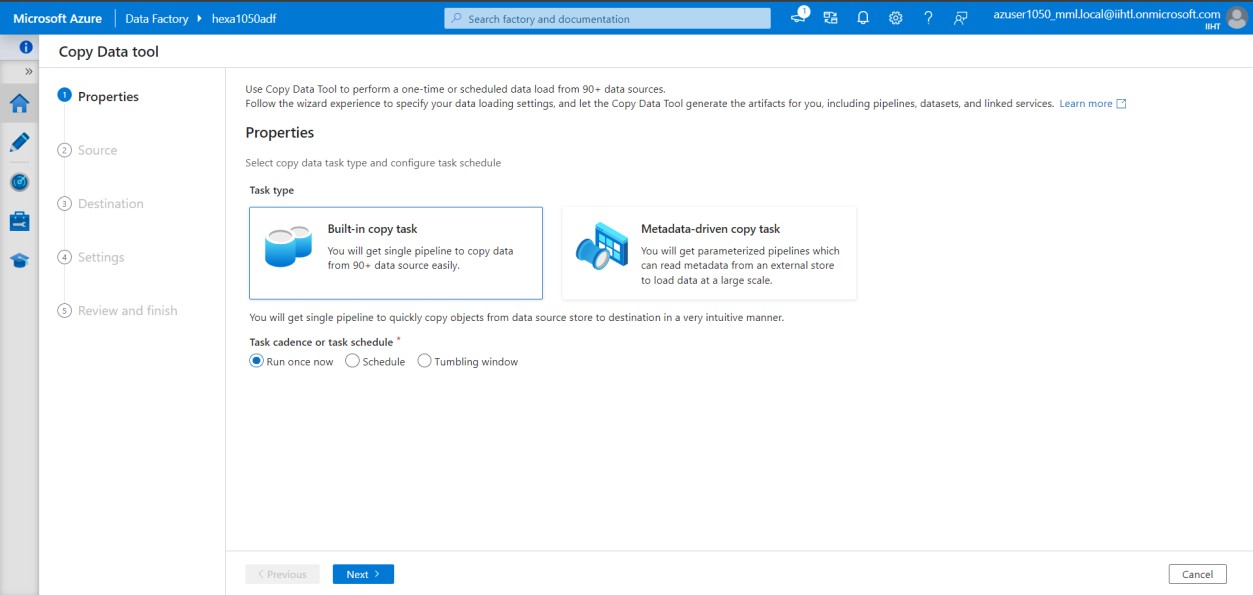
# Movement of data:

* + Raw data is stored in source storage (Azure Blob Storage)

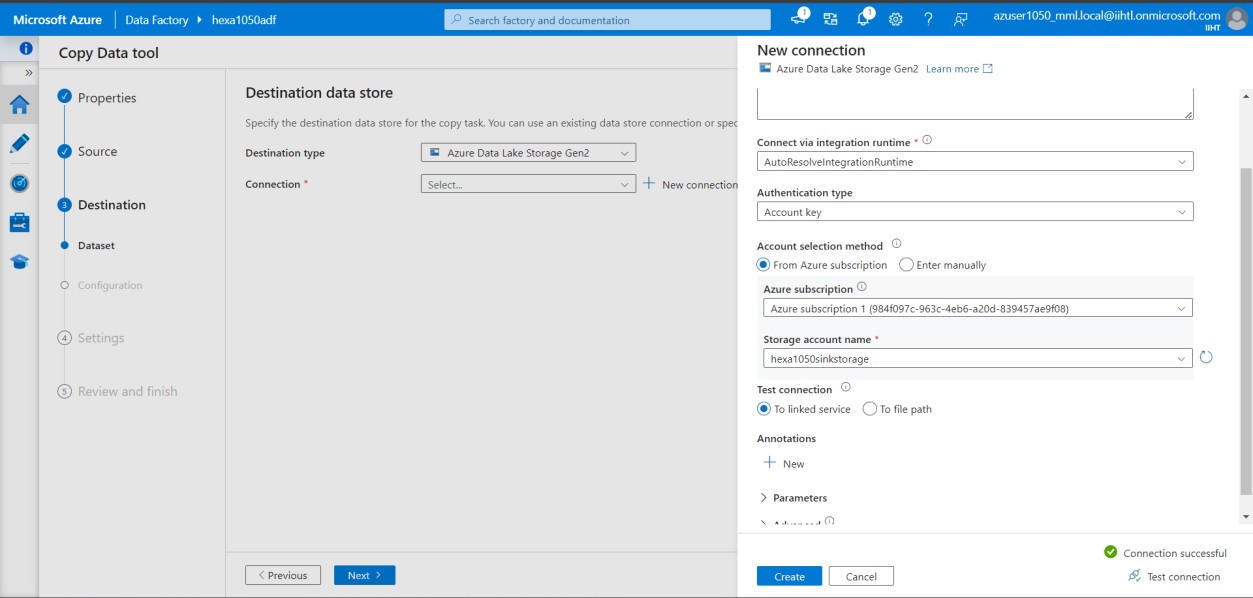


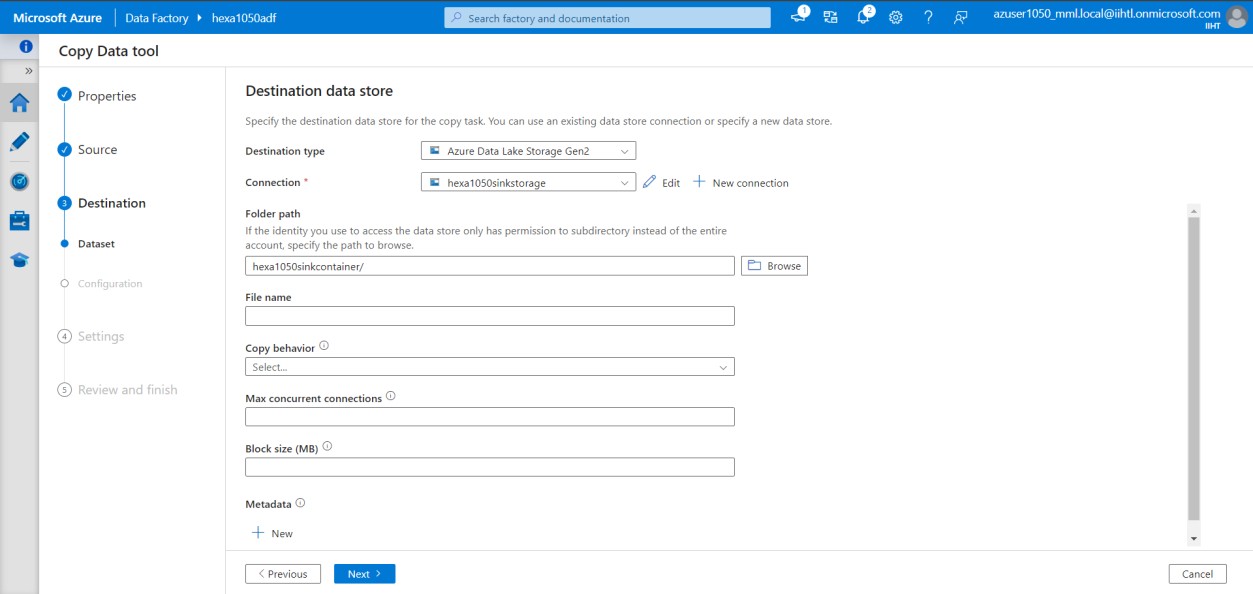
* + Ingest the data from the source storage(blob storage) to sink storage(Azure data lake gen 2) by creating a pipeline in the azure data factory with the following properties.



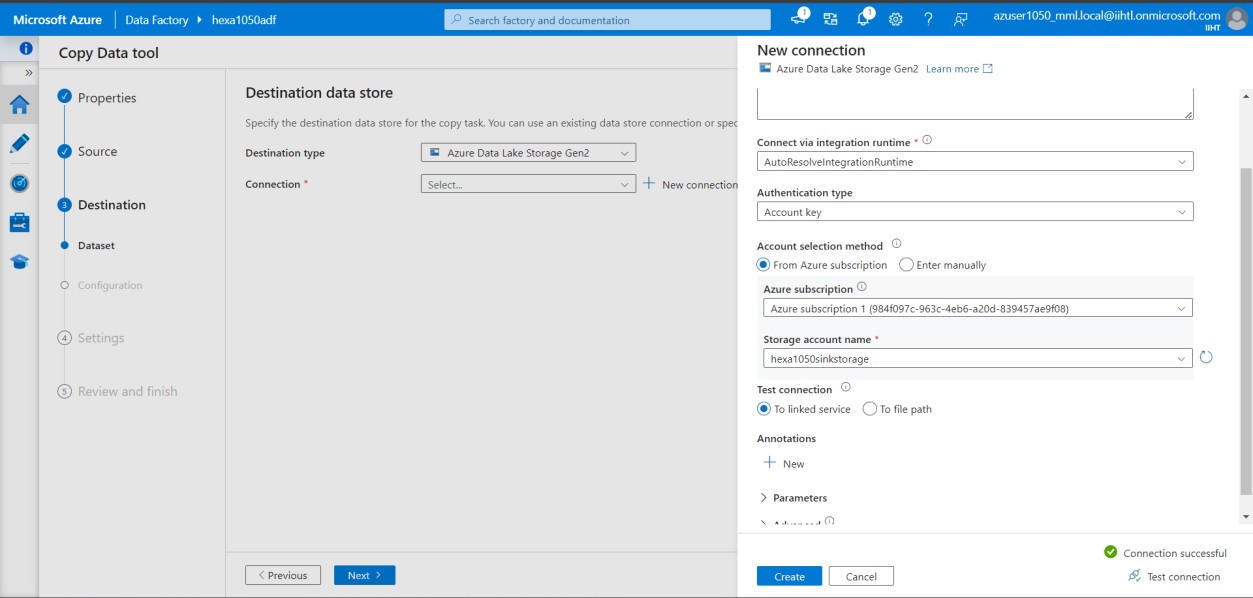


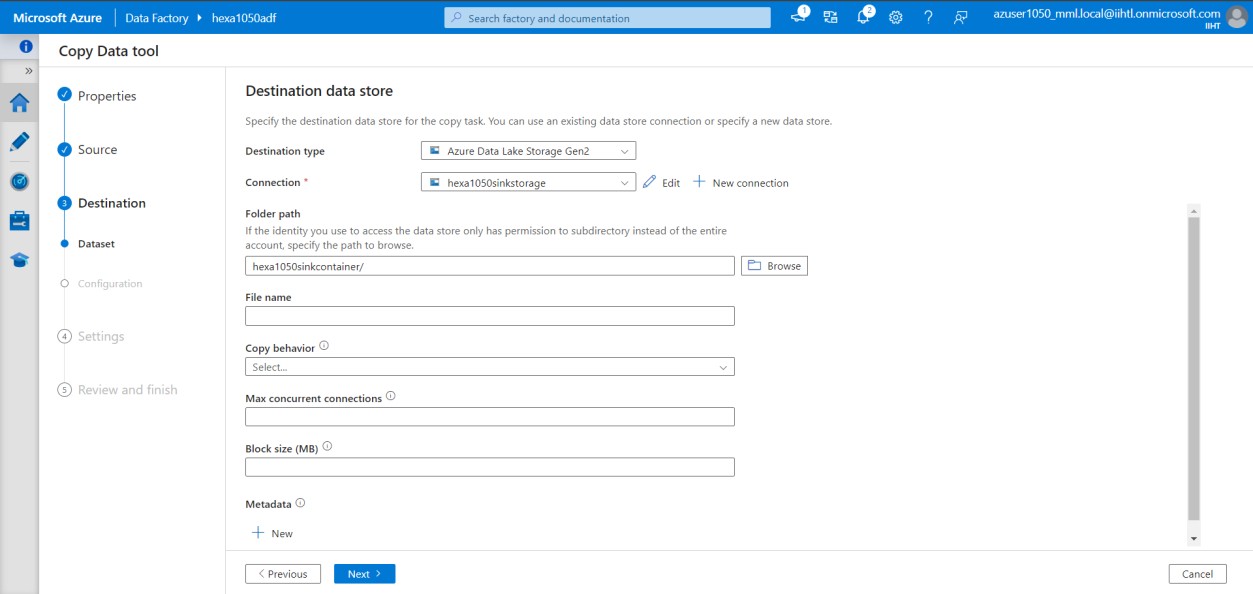
* + Create the connection with the source storage and mention the right file path of the raw data.



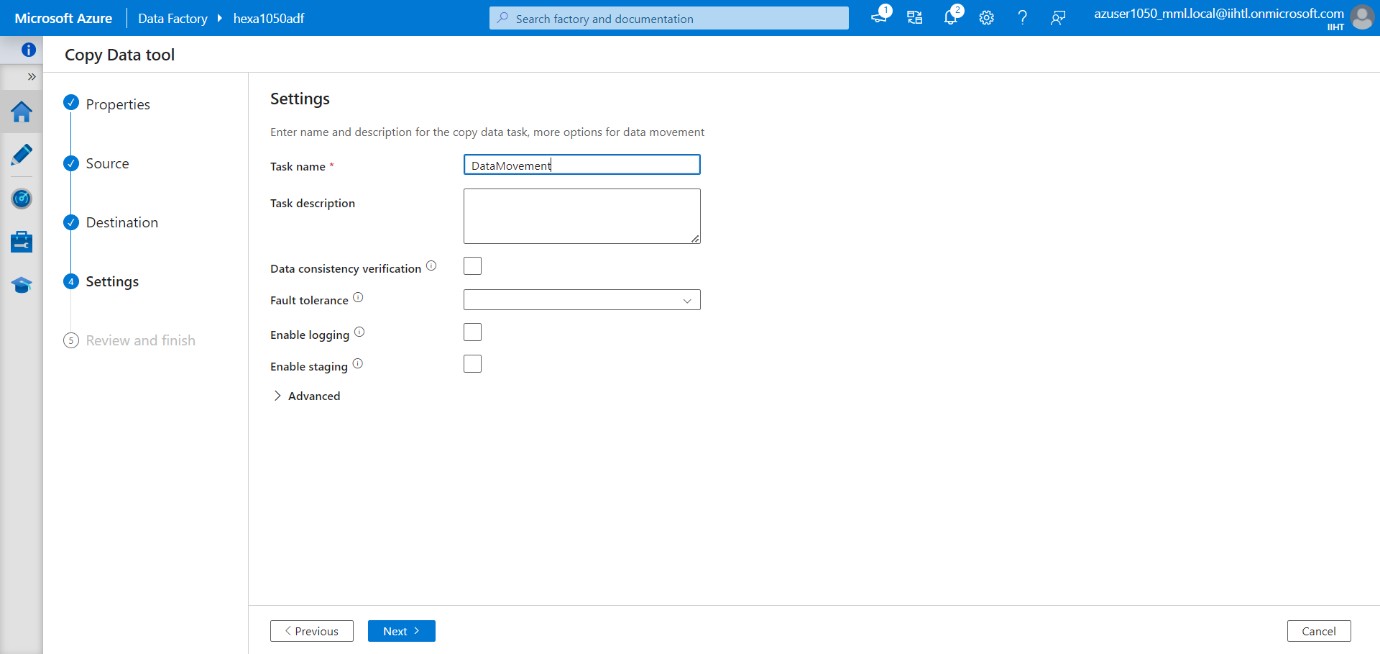


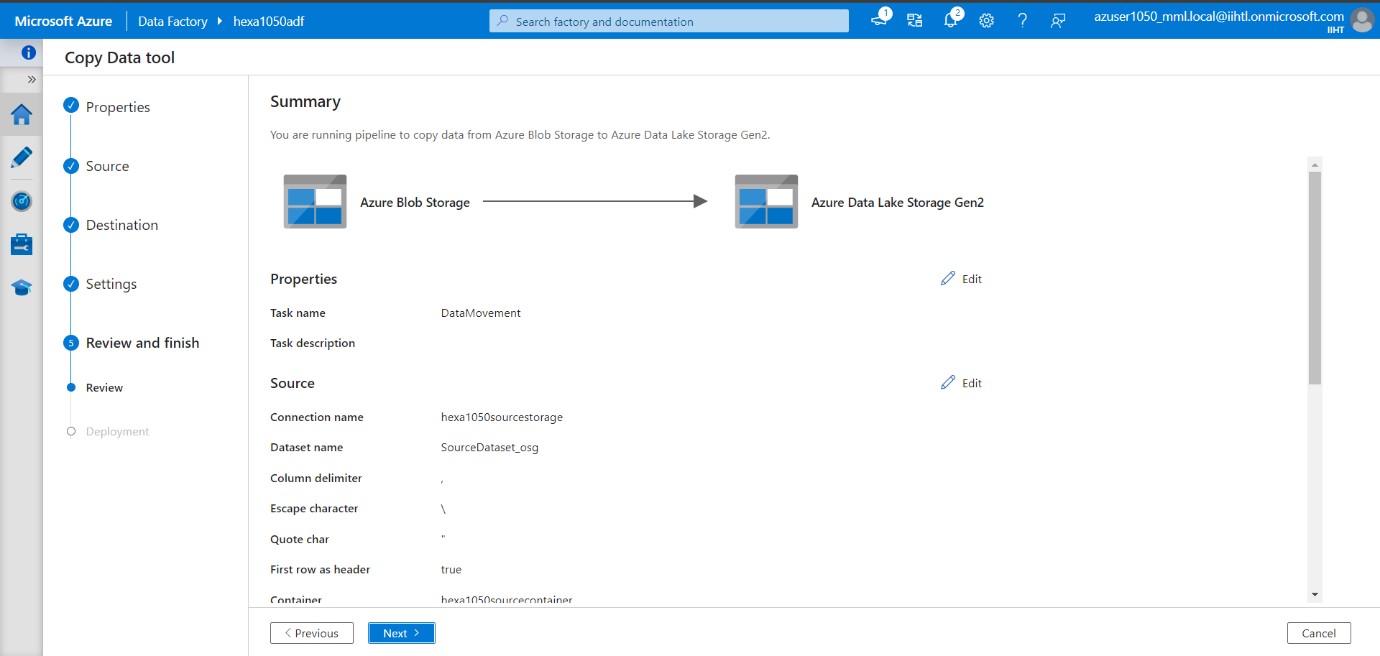
* + Create connection with sink storage and mention the path or container destination where the data has to be copy from source storage.

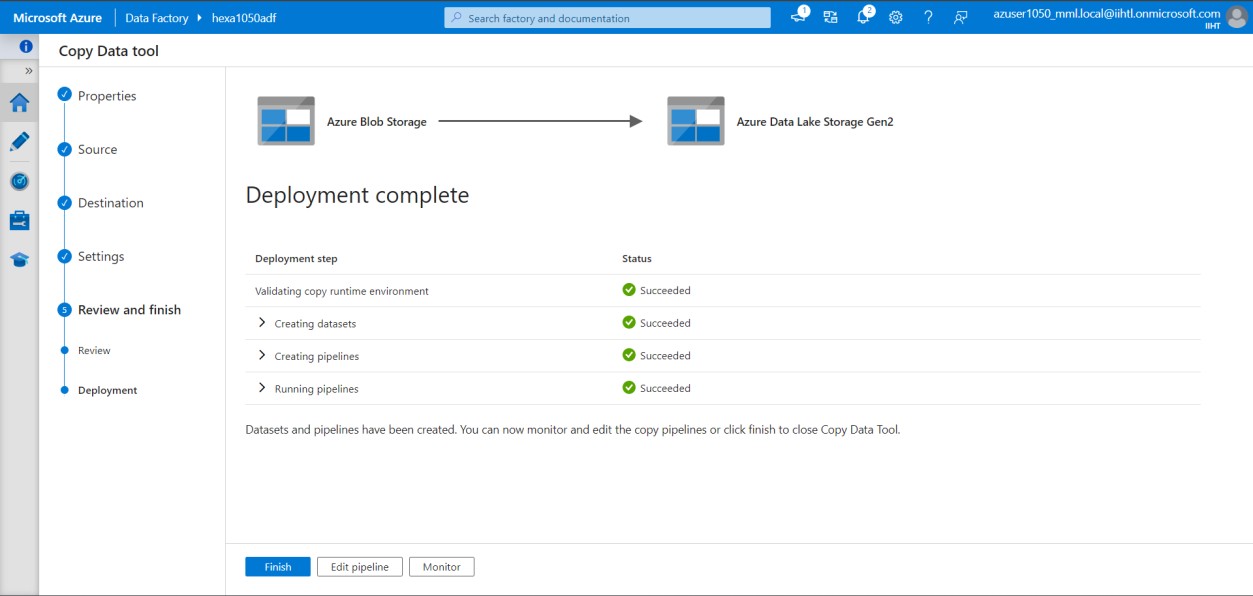




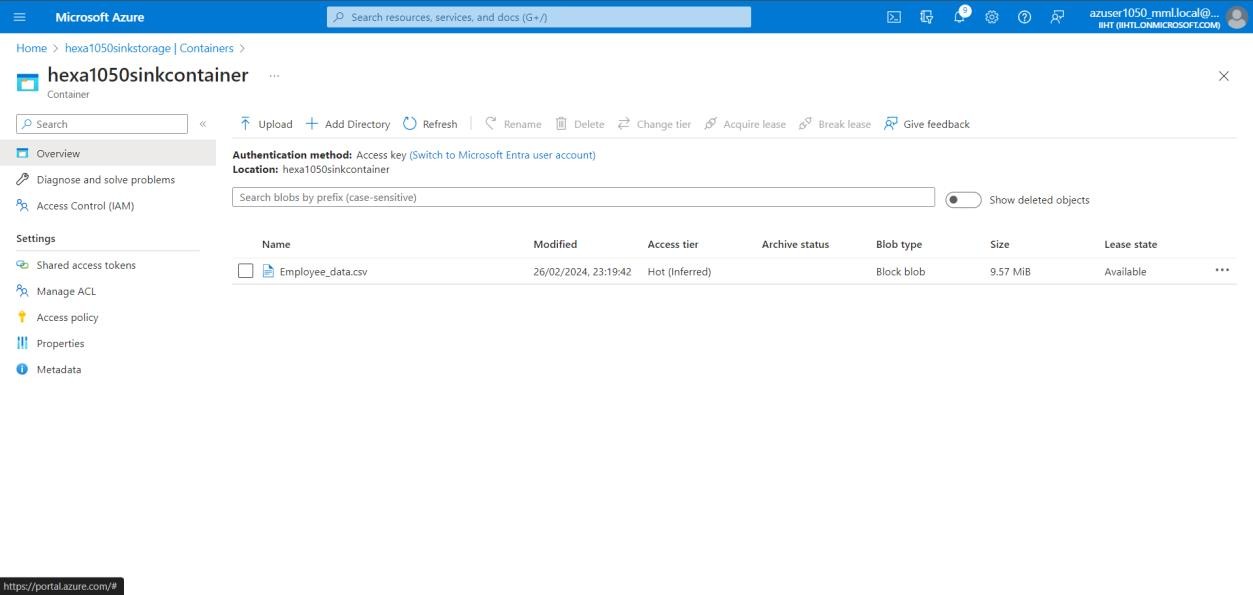
* + Give pipeline details, review and start the copy process







* + After the process gets finish we can see that the data in the source storage (blob storage) has copied to sink storage (Azure data lake)

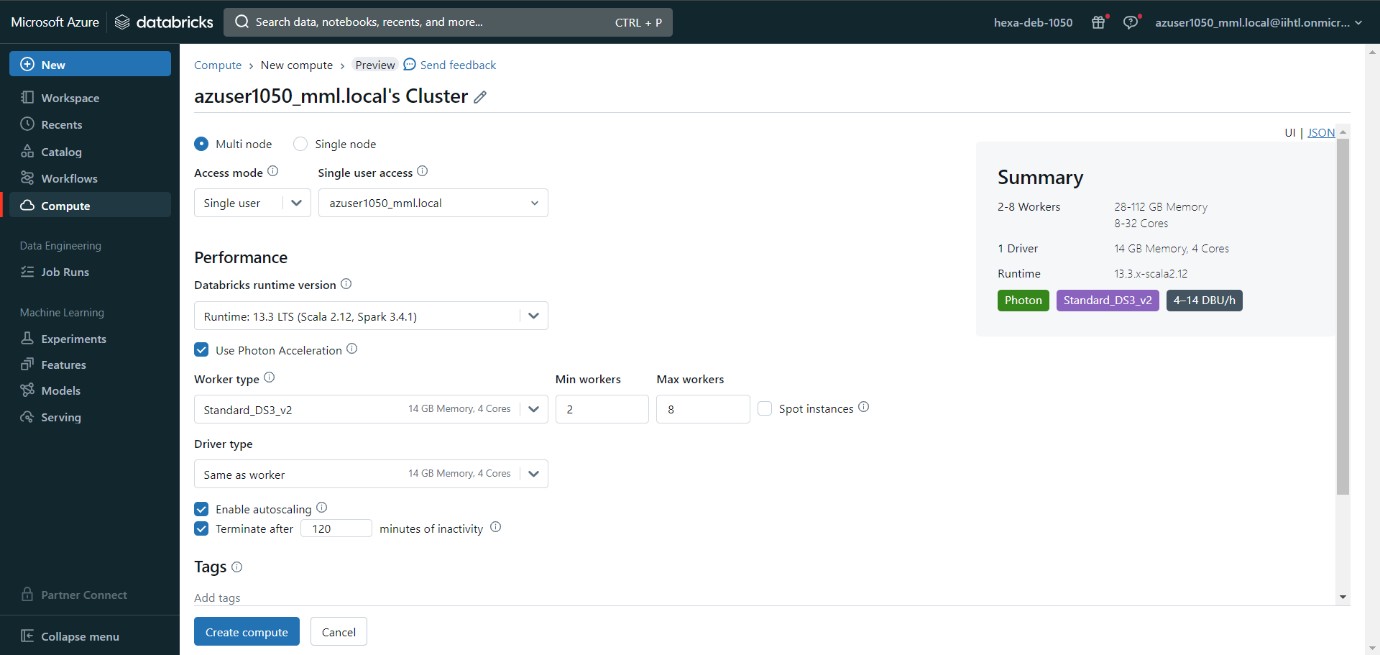


# Developing Pyspark Notebook

* + Create a new PySpark notebook within the Databricks workspace. Begin writing PySpark code to perform ETL operations, data transformations, and data analytical tasks

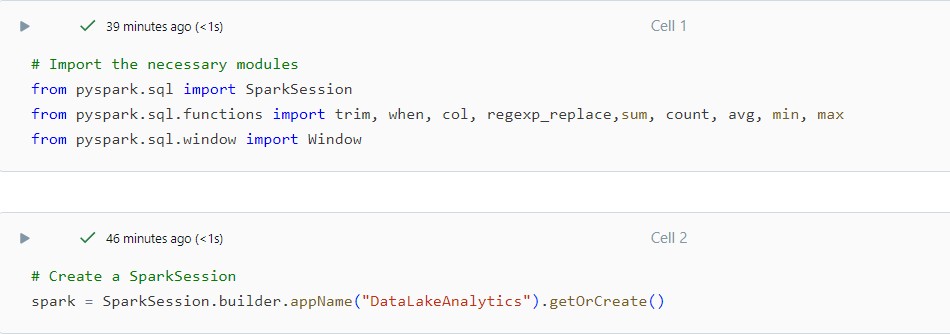
# Create cluster and connecting it to notebook

* + The cluster is created with autoscaling is enabled which automatically adjust cluster size to accommodate changes in workload demand, allowing for seamless scalability without manual intervention.



# Importing Necessary libraries and Creating Spark Session

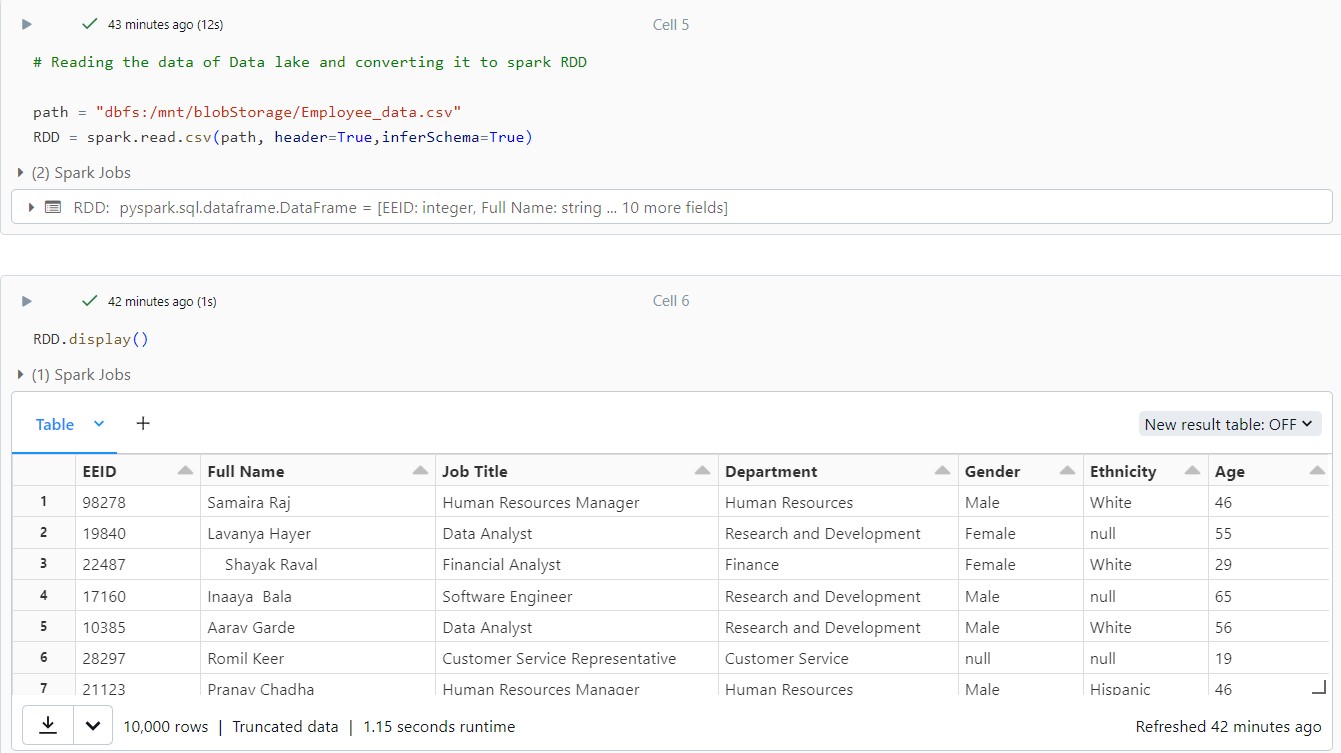
* + Use SparkSession.builder to configure and create a SparkSession. specify the application name using .appName() and configure any additional Spark options using .config(). Finally, call .getOrCreate() to either create a new SparkSession



# Extracting Data from Source storage

* + Connecting data source (Azure Blob Storage) by mounting it to the Databricks File System (DBFS) to simplify data access
  + It helps to retrieve raw data for processing and analysis within the PySpark environment



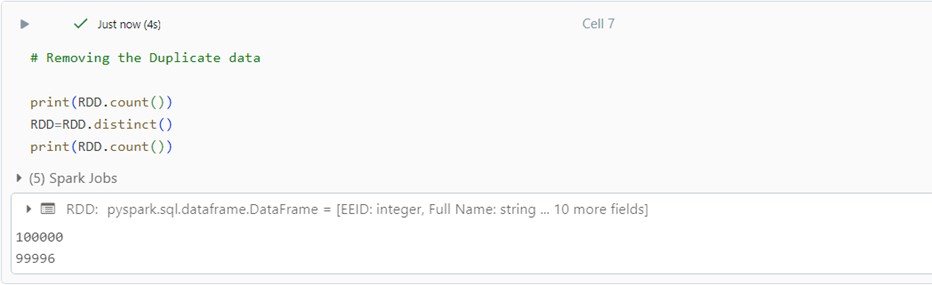


# Transforming the raw data

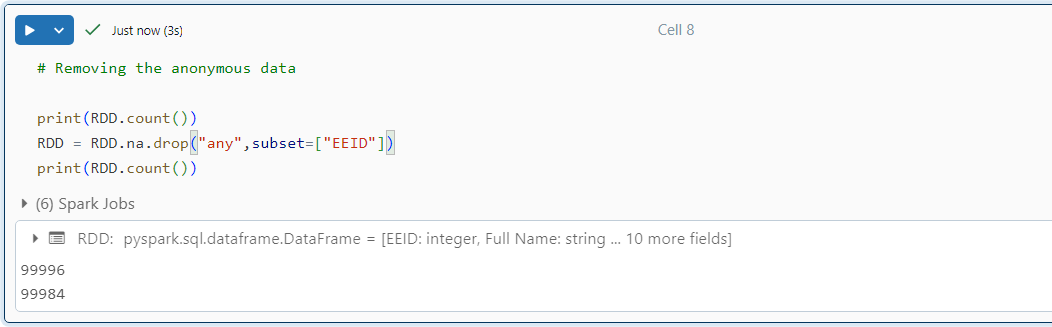
* + Utilize PySpark DataFrame transformations and functions to cleanse, transform, and prepare the data for analysis.
  + Implement business logic and data processing steps to transform raw dataset up to mark for data analysis purpose.

# Transformations done:-

* Removing the Duplicate records



# Handling anonymous data



* Removing Extra spaces and filling the null data with proper messages



# Handling the numerical columns and renaming USA to US to make dataset consistent

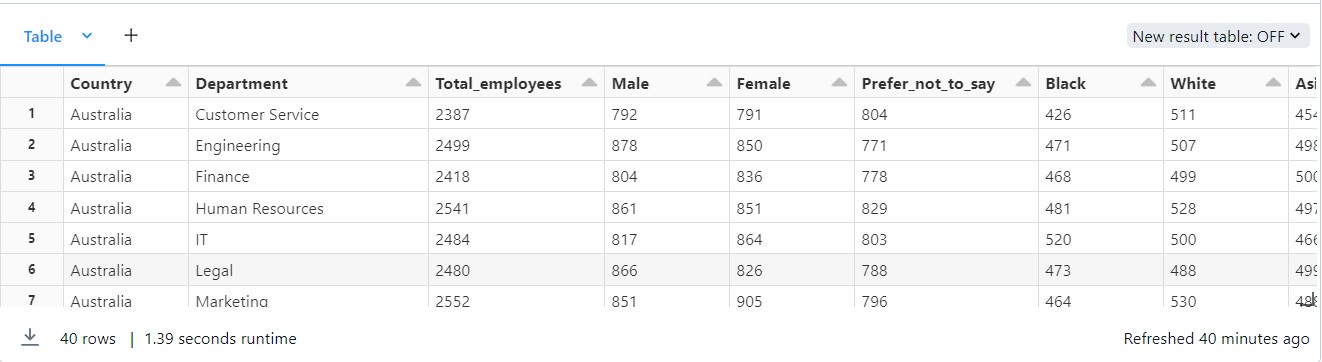


1. Data Analytics
   * Data analytics is the collection, transformation, and organization of data in order to draw conclusions, make predictions, and drive informed decision making.

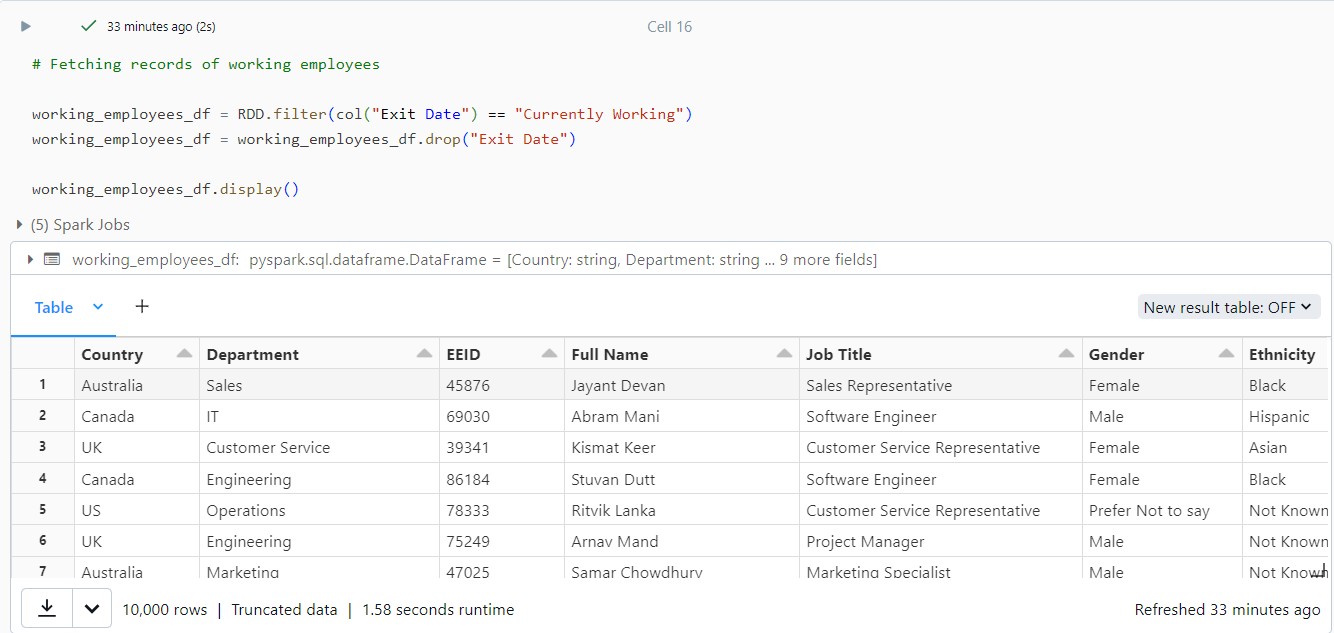
# Analytics Performed

* Grouping the data based on country and department (count\_df)

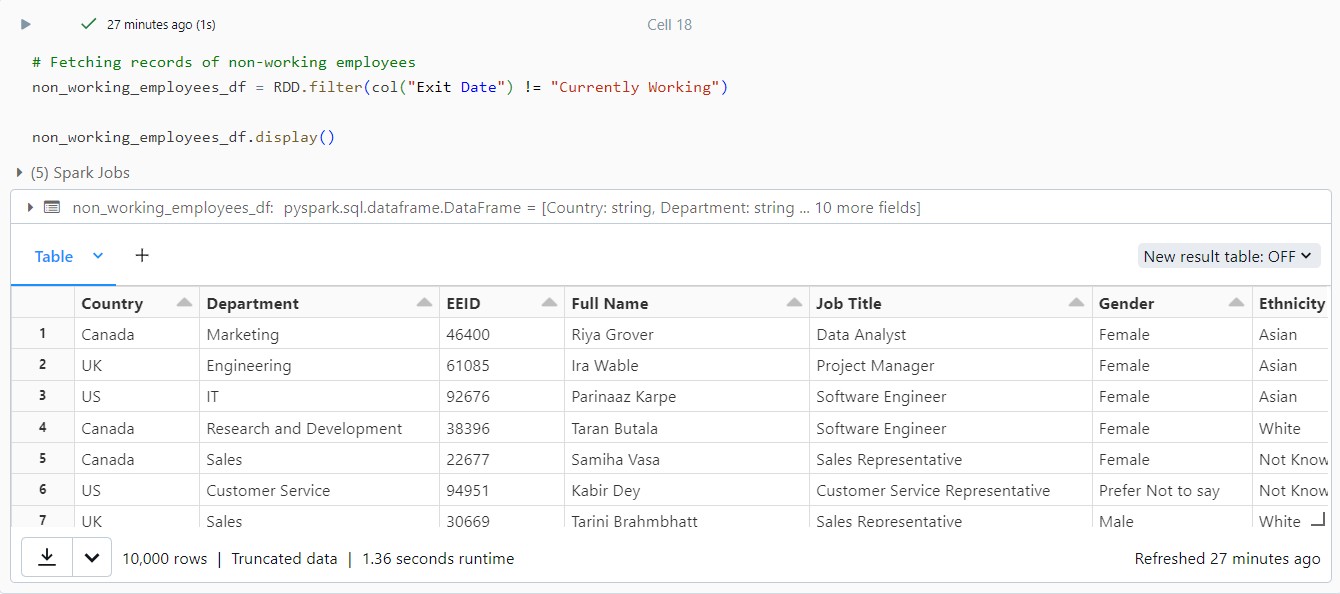




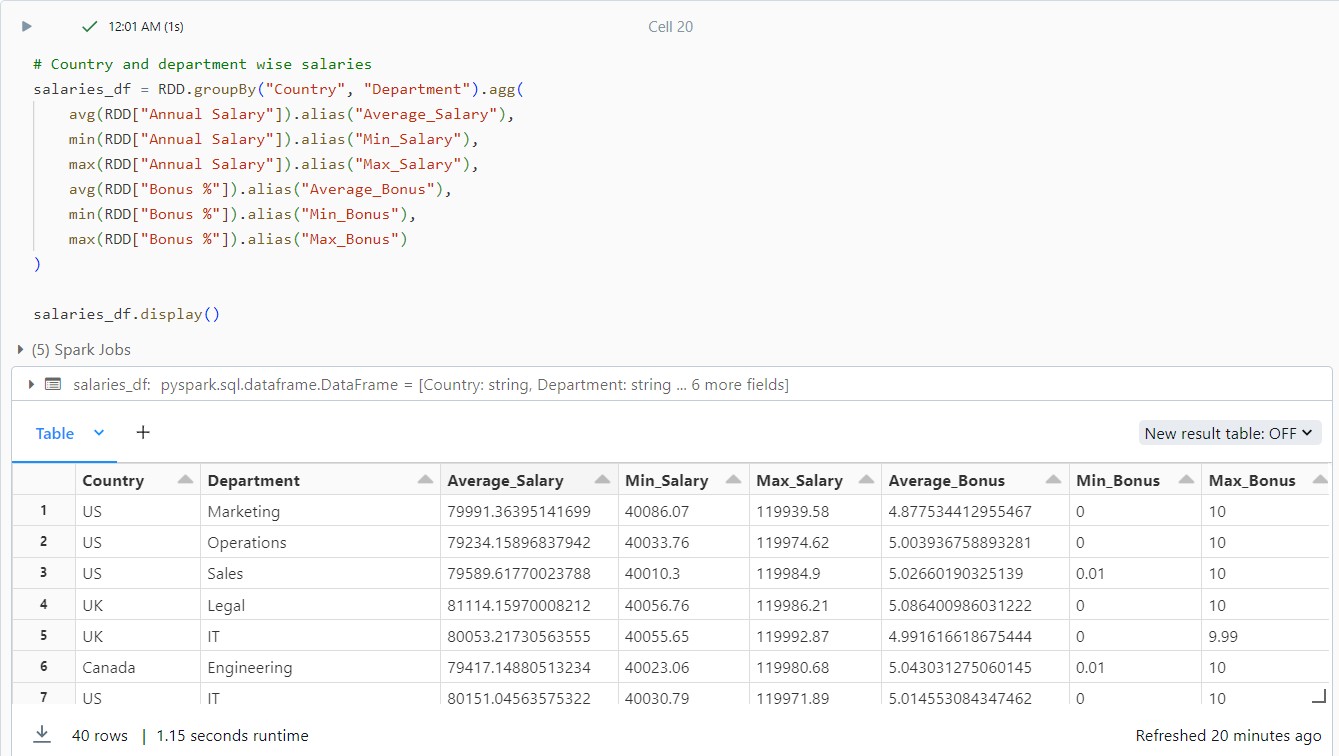
# Sorting the data by considering only working employees (working\_employees\_df)



* Sorting the data by considering only those employees who left the company (non\_working\_employees\_df)



* Making the statistical data of Annual salary and Bonus percentage in each country based on department (salaries\_df)

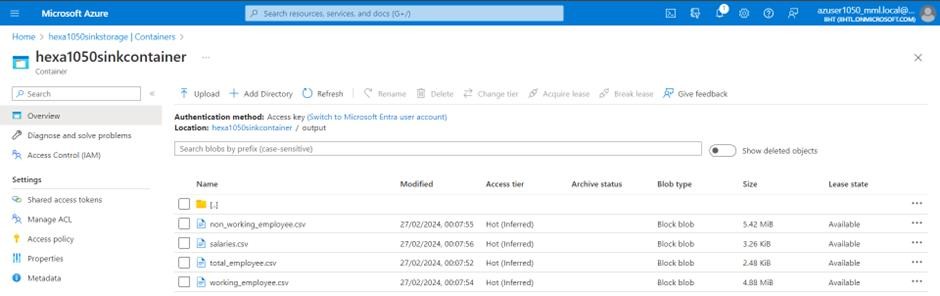


# Loading and organizing the analytical data to the azure data lake

* + By using the mount point load and organize the data into the data lake
  + Use the appropriate folder path in the container for better organizing of analytical data and also use the appropriate names for the files



* + We can see the analytical data files has been generated and organized in the appropriate output path in the azure data lake



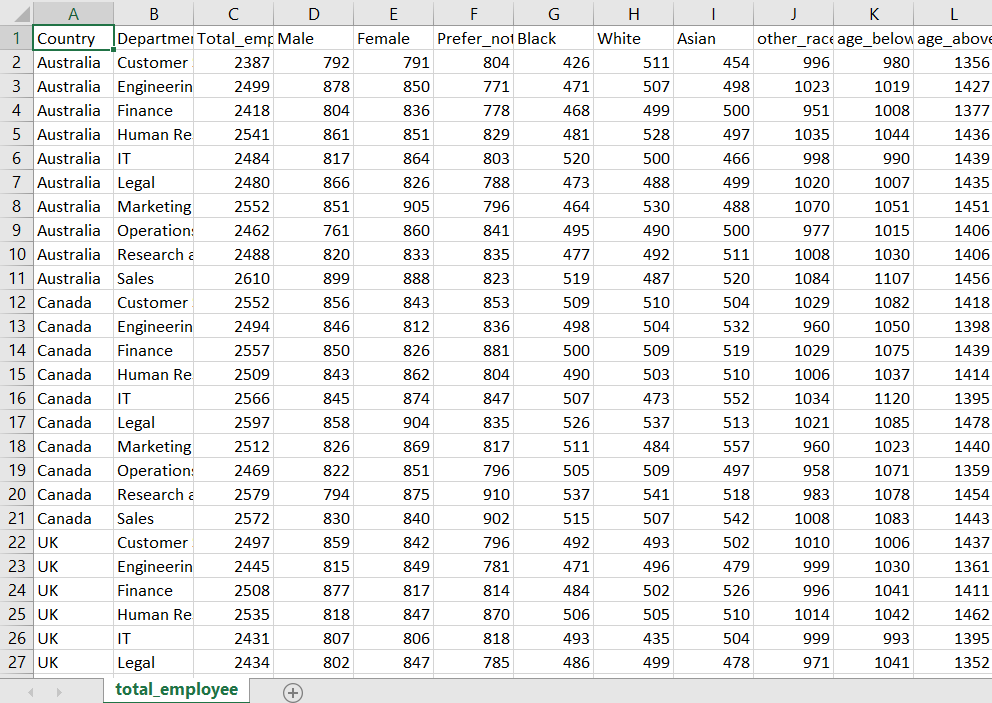
# Unmounting the azure data lake



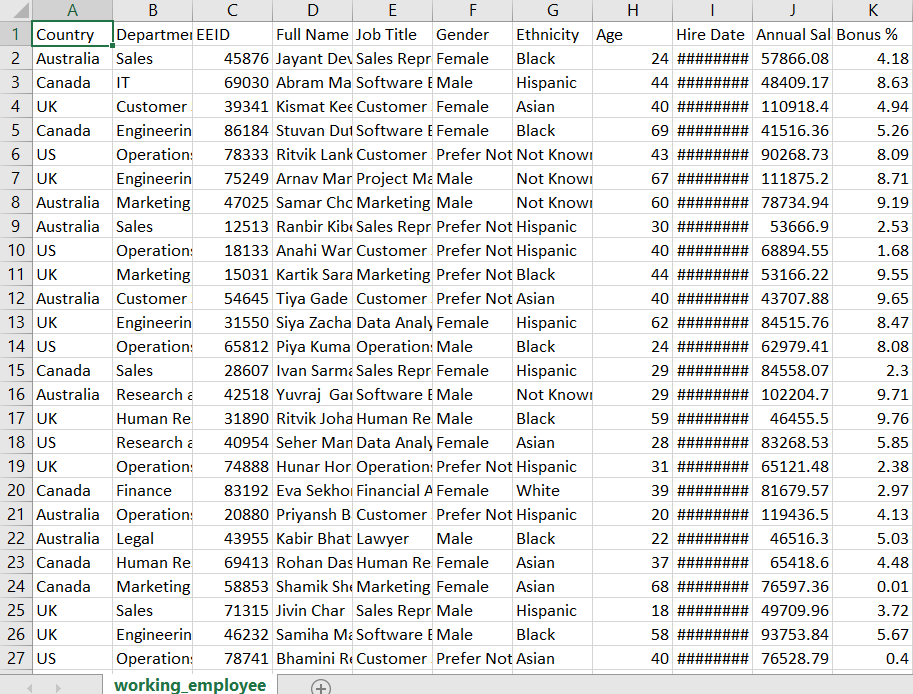


Analytical data stored in Sink storage

1. Grouped data based on country and department



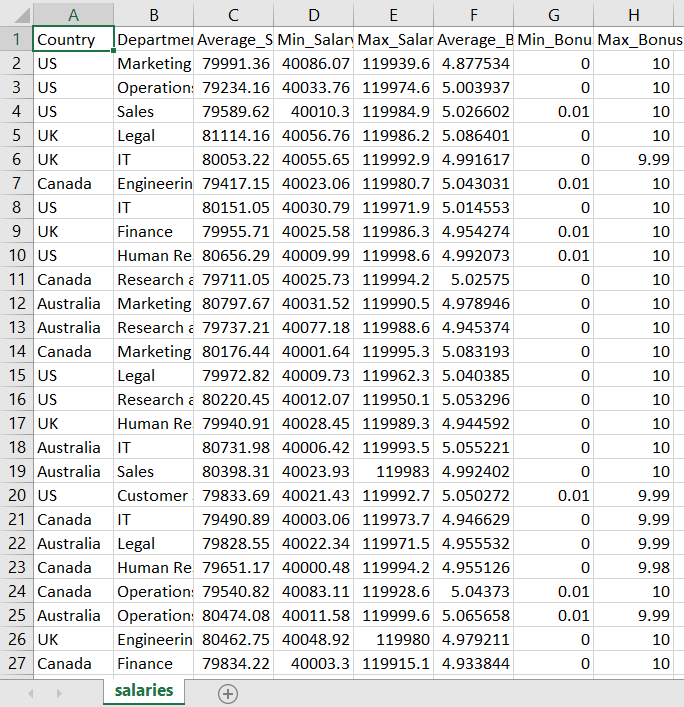
# Currently Working employees data



1. Non-Working employees data



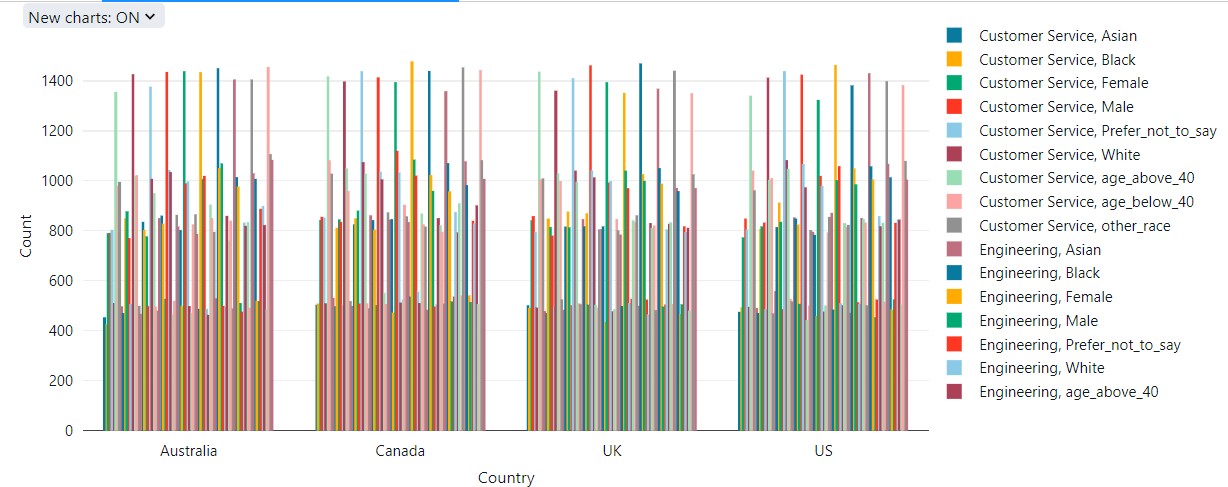
# Statistical data of Annual salary and Bonus percentage in each country based on department



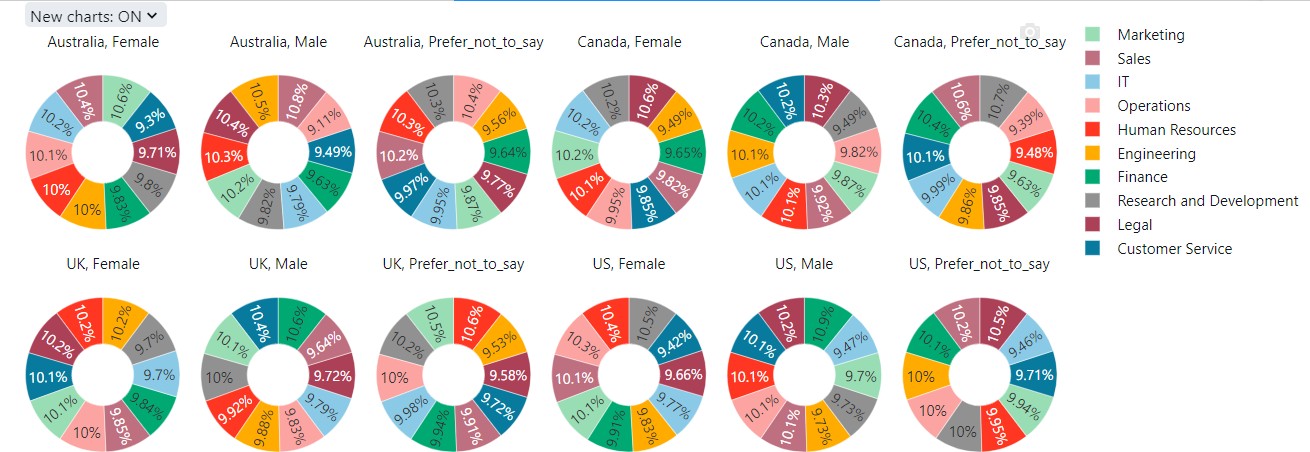


Data visualization on analytical data (truncated data is used)

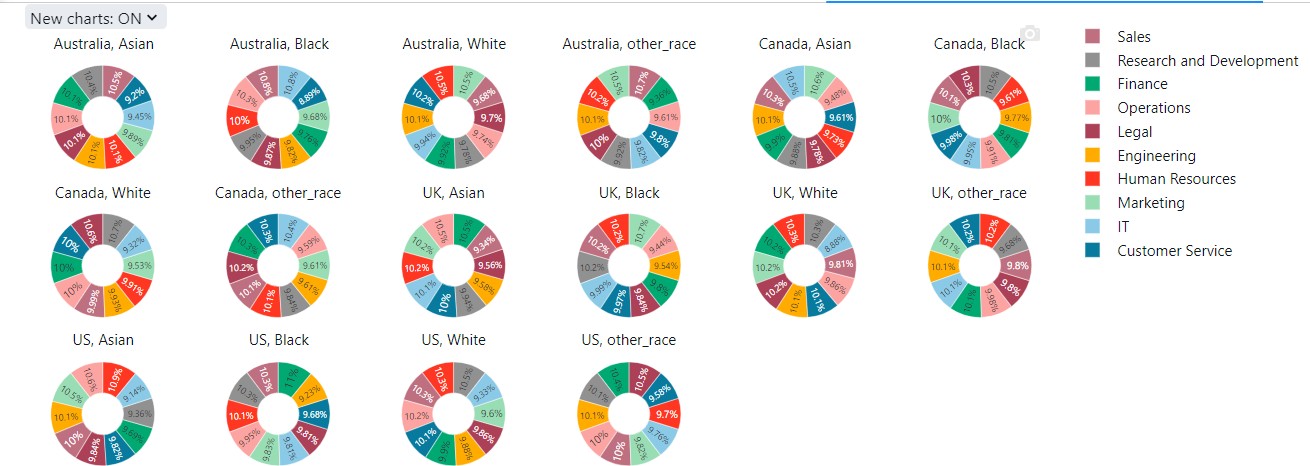
1. Count\_df:
   * Bar graph of count\_df with country at x-axis and count at y-axis



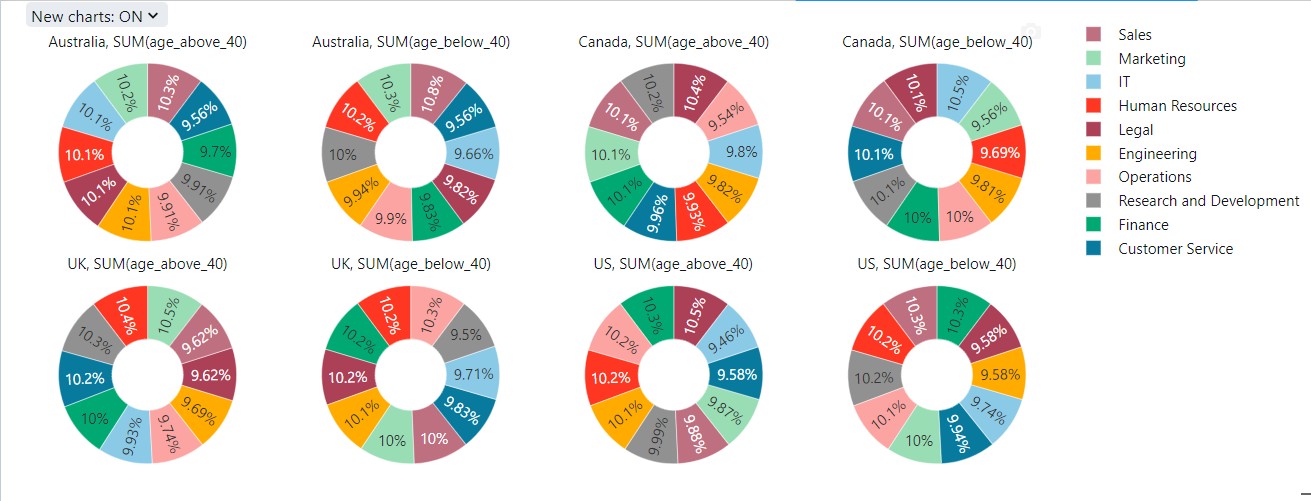
* + Pie chart of Departmental Statistical data based on gender



* + Pie chart of Departmental Statistical data based on ethnicity

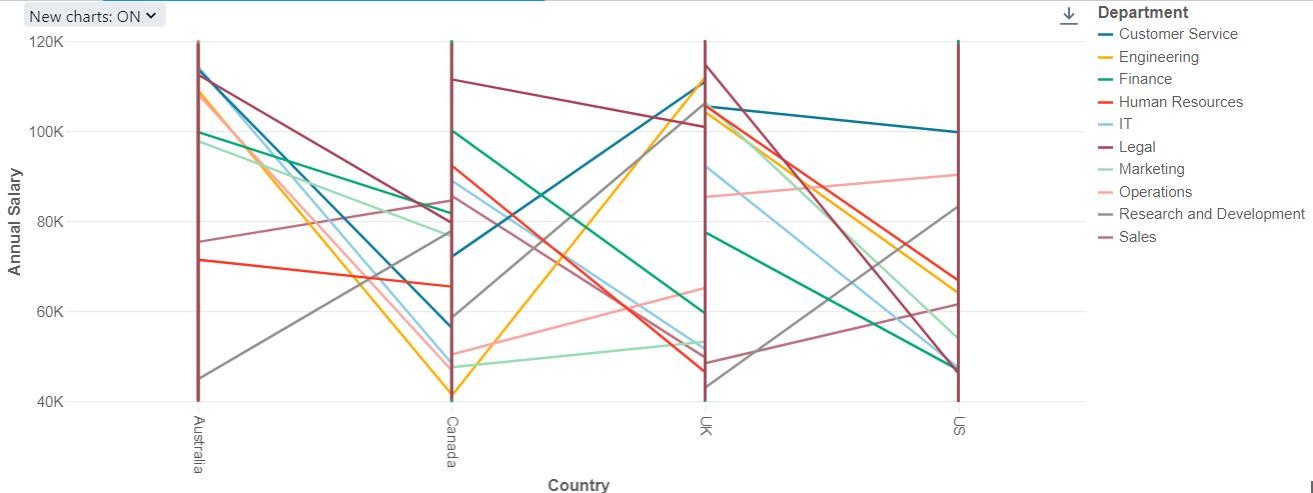


* + Pie chart of Departmental Statistical data based on age

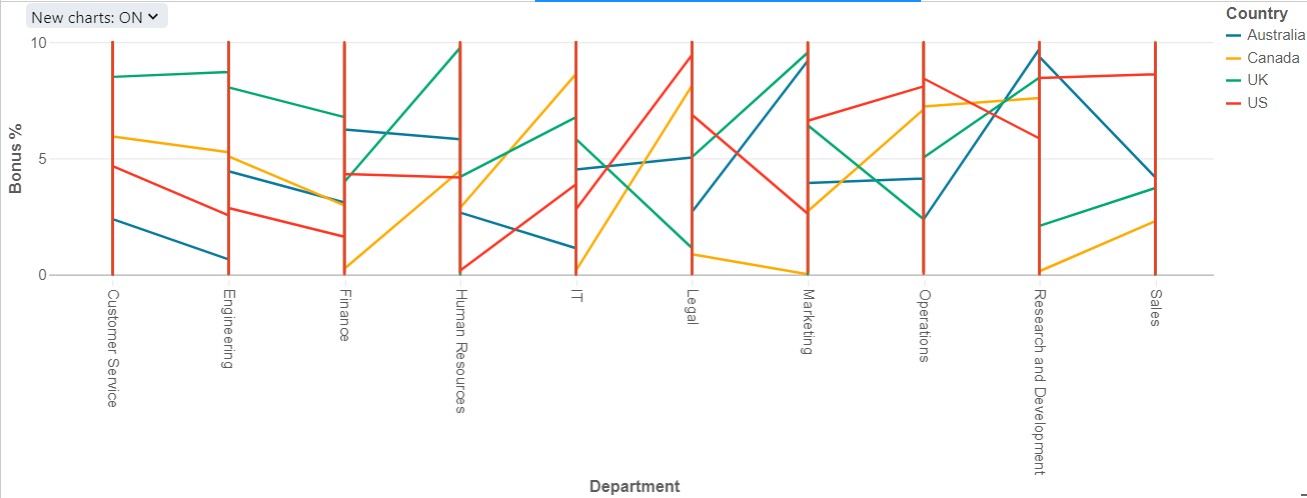


# working\_employee\_df

* + Representation of Departmental wise Annual Salary for each country

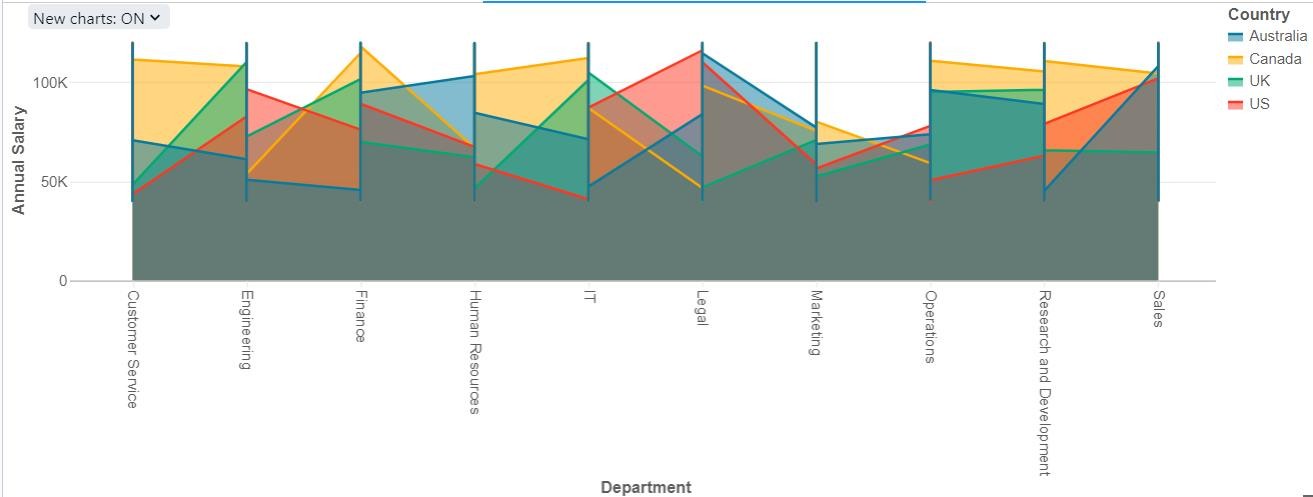


* + Representation of Departmental wise Bonus percentage for each country

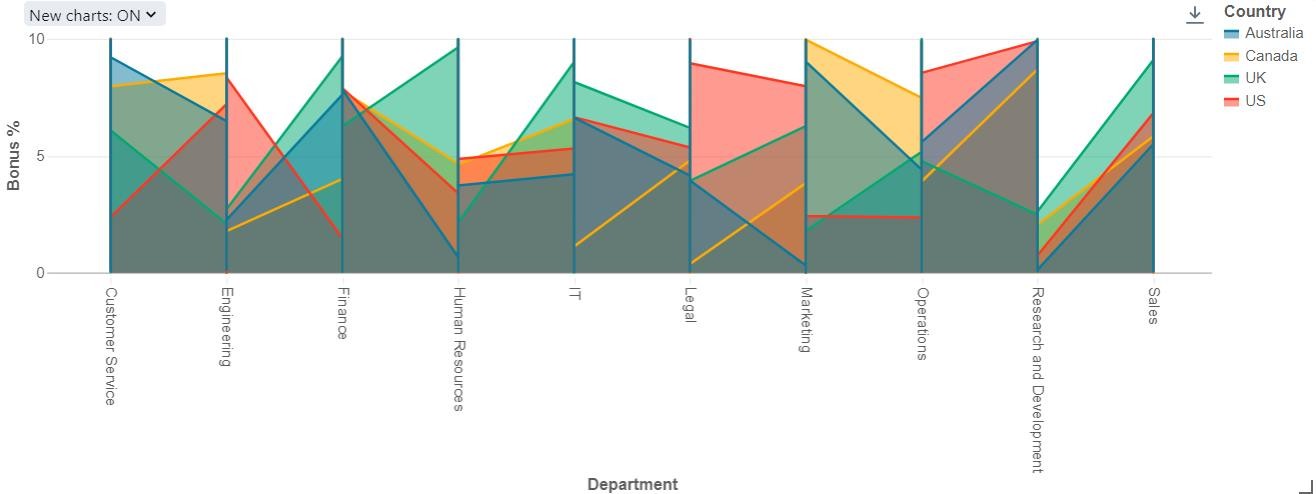


# non\_working\_employee\_df

* + Representation of Departmental wise Annual Salary for each country

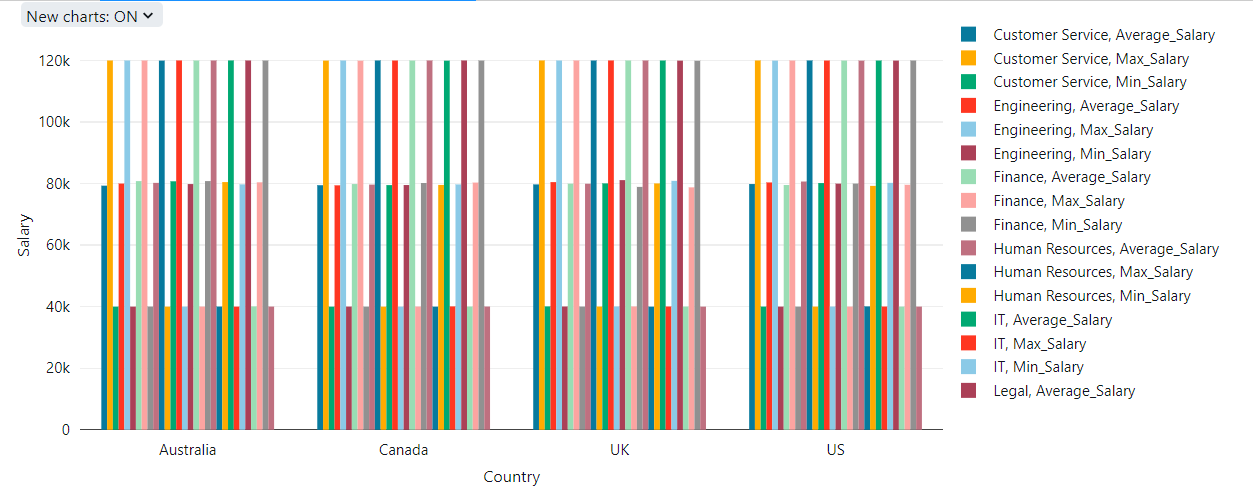


* + Representation of Departmental wise Bonus percentage for each country

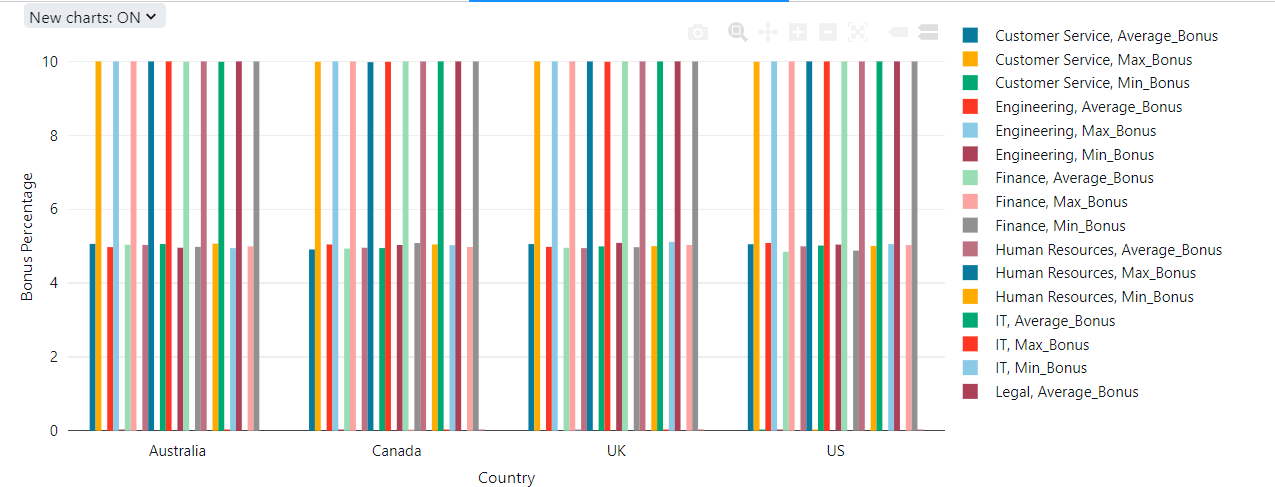


# salaries\_df

* + Representation of Department wise Annual Salary for each country



* + Representation of Department wise Bonus percentage for each country





Conclusion

In conclusion, this project showcases a cohesive data processing pipeline leveraging Azure Data Factory and Azure Databricks, seamlessly moving data from Azure Blob Storage to Azure Data Lake Storage Gen2 then processing the data by performing transformation and analytical operations through azure databricks, then loading and organizing the data back to Azure data lake storage.