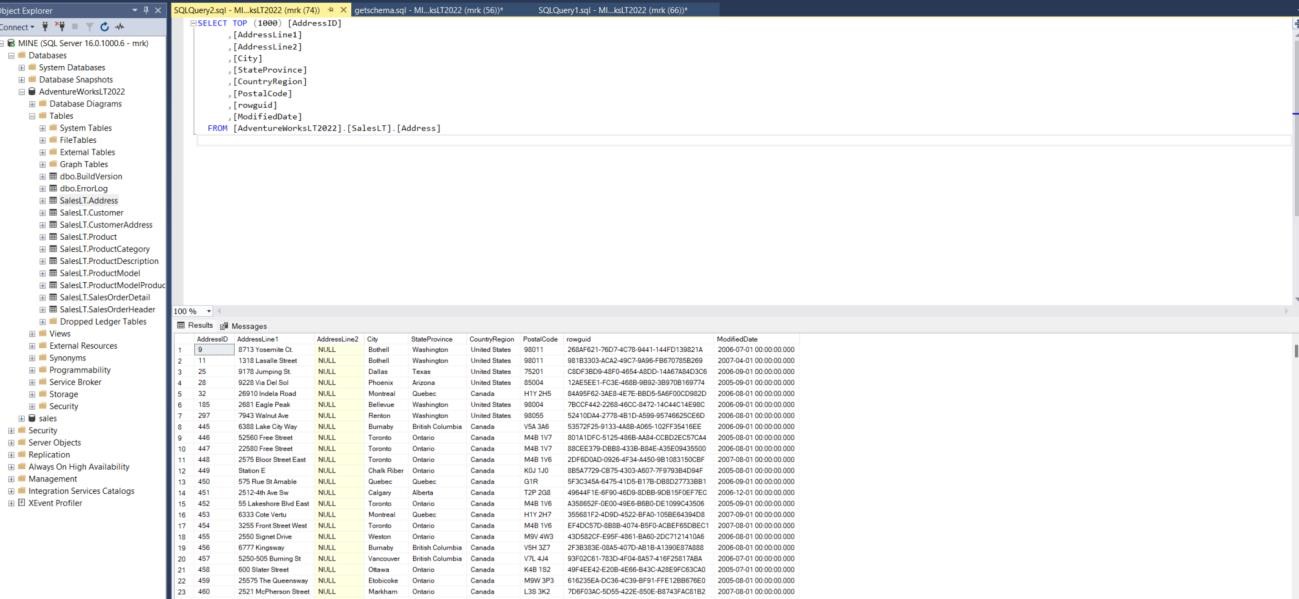
# Case Study: Data Pipeline for SQL to Azure

This document describes the step-by-step implementation of a data pipeline to extract data from an on-premises Microsoft SQL database, transfer it to Azure Data Lake Storage (ADLS) using Azure Data Factory (ADF), process and clean the data using Databricks, and finally load the processed data into a "gold" storage layer.

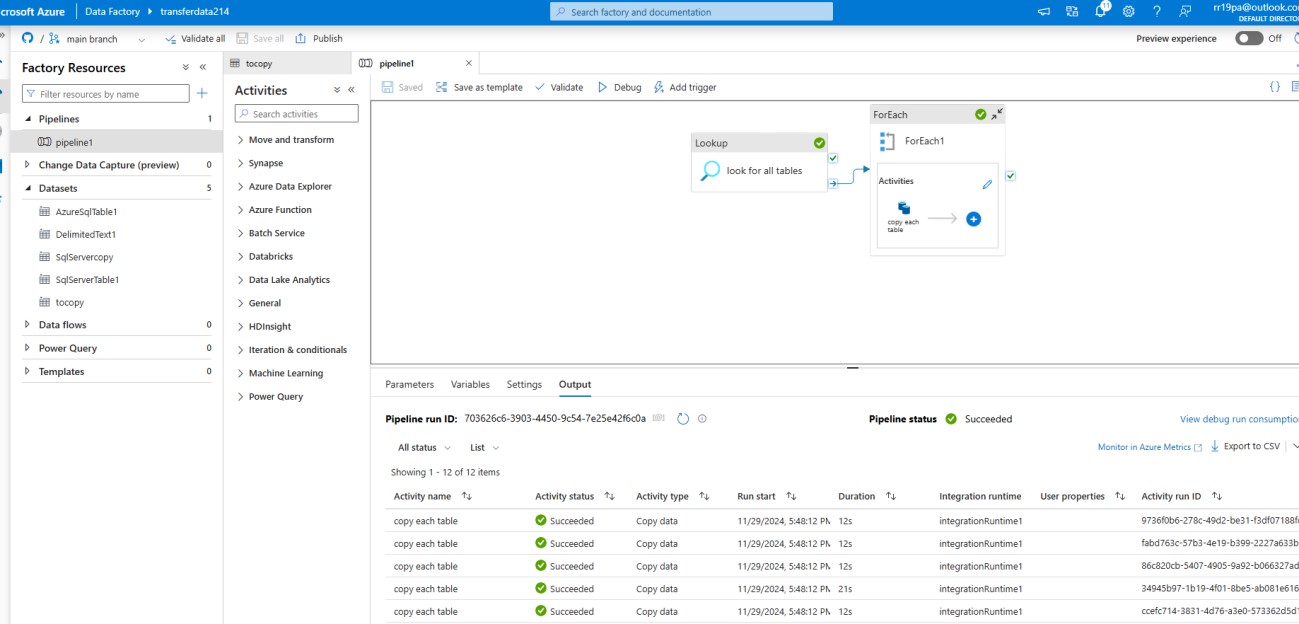
---

## Project Workflow Overview

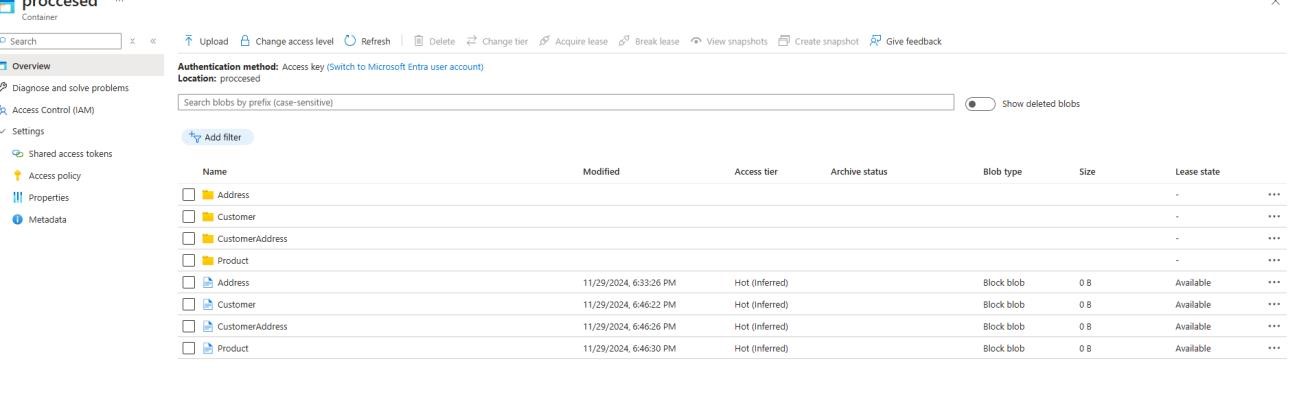
1. \*\*Data Extraction\*\*: Extract data from the on-premises Microsoft SQL database.



1. \*\*Data Transfer to Raw Zone\*\*: Use Azure Data Factory to move the raw data to the raw folder in Azure Data Lake Storage.



1. \*\*Data Cleaning and Processing\*\*: Utilize Databricks to clean and transform the data, storing the output in the processed folder.



1. \*\*ETL and Gold Zone Storage\*\*: Perform ETL operations and move the final transformed data to the gold folder in Azure Data Lake Storage.

---

## Prerequisites

* An Azure account with access to Azure Data Factory, Azure Data Lake Storage, and Databricks.
* On-premises Microsoft SQL database with necessary credentials.
* Installed and configured \*\*Azure Data Gateway\*\* to connect the on-premises SQL database to Azure.
* Databricks cluster configured and running.
* Python and/or PySpark knowledge for Databricks scripts.
* Proper \*\*Access Control\*\* setup to secure data access in Azure Data Lake Storage and Databricks.

-Proper login protection using azure key vault

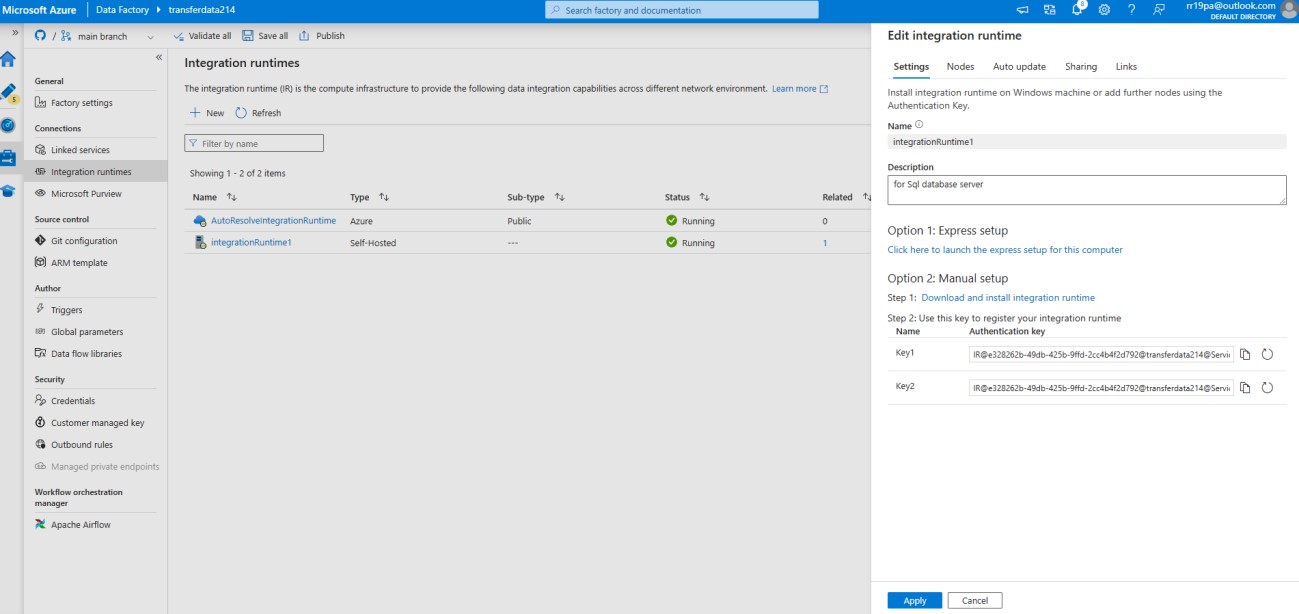
---

## Step-by-Step Implementation

### Step 1: Extract Data from On-Premises SQL Database

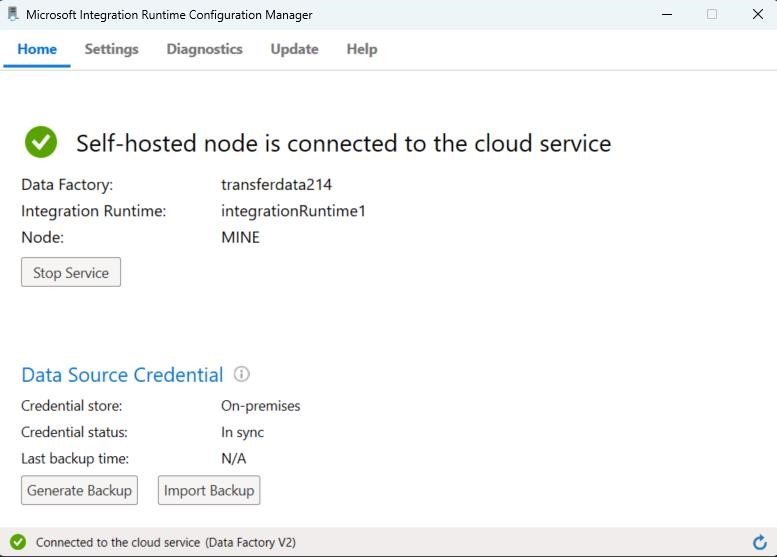
1. \*\*Set Up Integration Runtime in Azure Data Factory\*\*:

* Go to Azure Data Factory > Manage > Integration Runtimes.
* Set up a self-hosted integration runtime to connect to the on-premises SQL database.



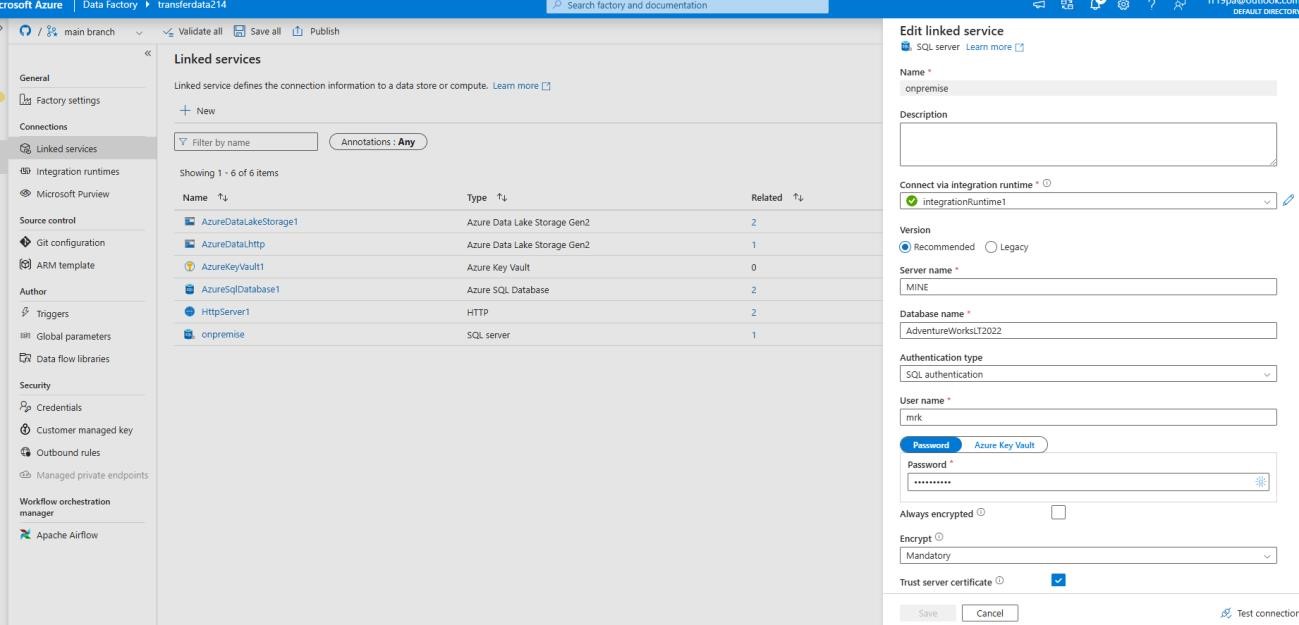
* Ensure the local runtime is installed on a machine with access to the on-premises SQL server.

Follow the Azure documentation to configure it properly.



2. \*\*Create a Linked Service for SQL Database\*\*:

* In ADF, create a new linked service pointing to the on-premises SQL database.
* Test the connection to ensure proper integration.

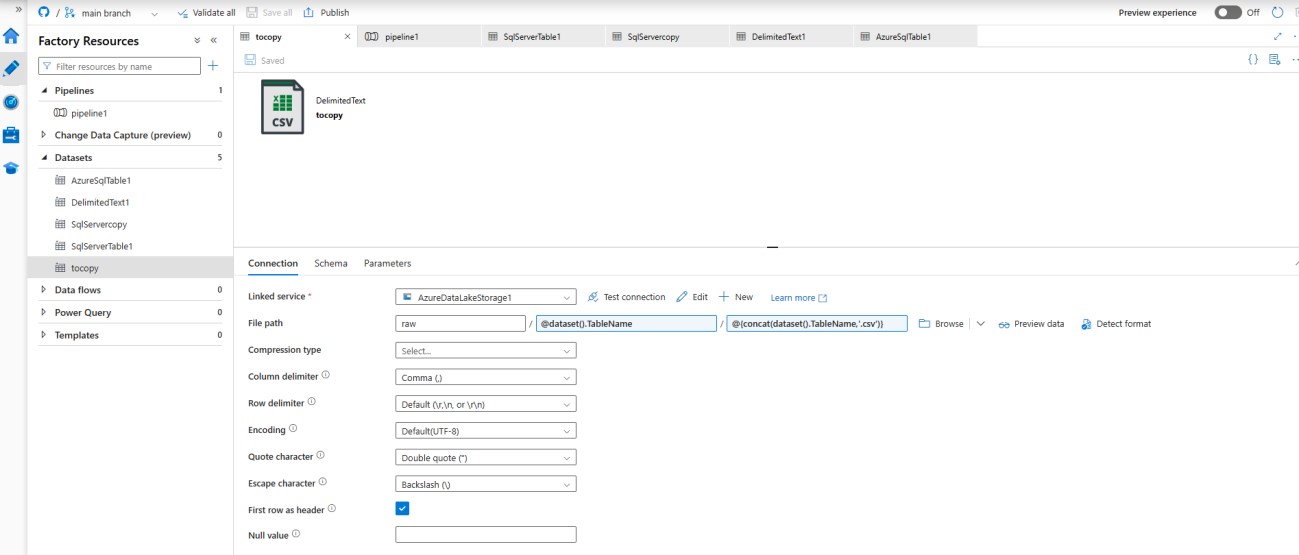
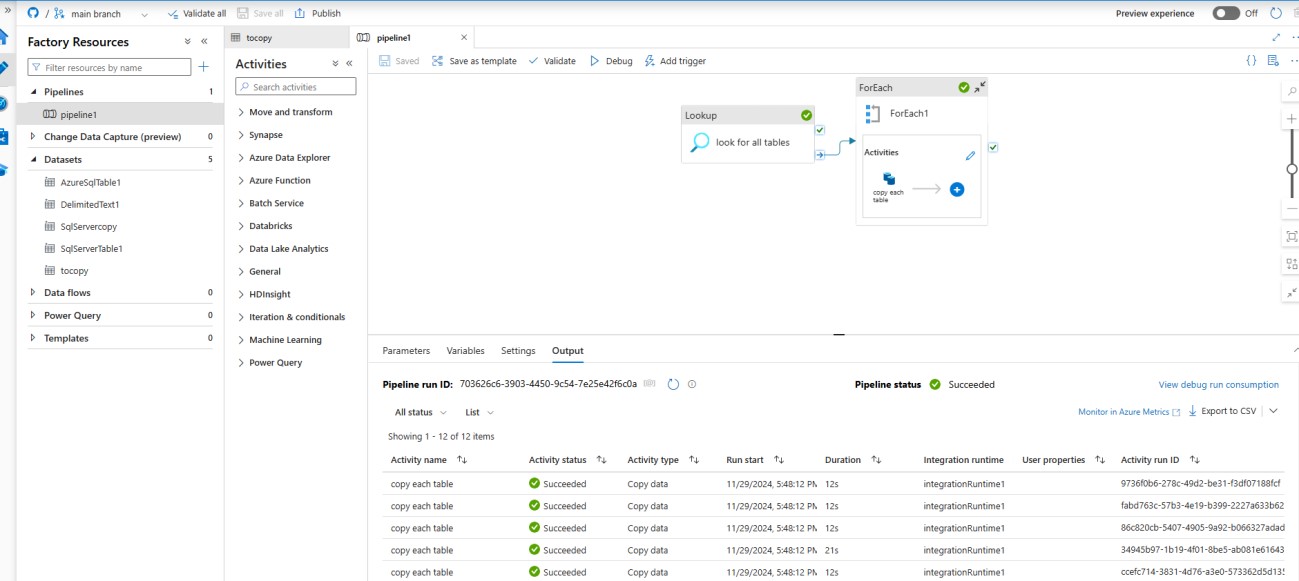


3. \*\*Define Access Control\*\*:

* Ensure the self-hosted runtime has the necessary permissions to access the on-premises database.
* Configure secure credentials, such as a managed identity or key vault, to protect sensitive information.

4. \*\*Build a Pipeline to Extract Data\*\*:

* Create a new pipeline in ADF.
* Add a "Copy Data" activity to pull data from the SQL database.
* Define the source as the SQL database linked service.



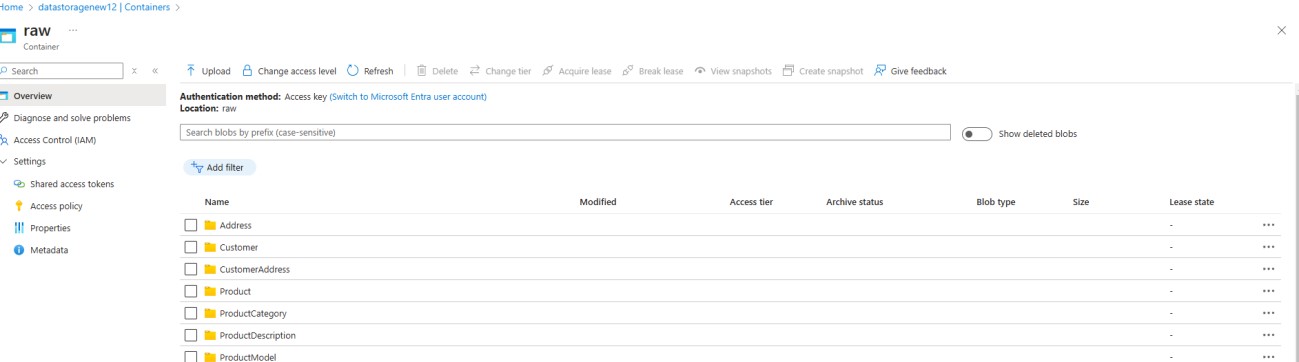
### Step 2: Transfer Data to Raw Folder in Azure Data Lake Storage

1. \*\*Set Up Azure Data Lake Linked Service\*\*:

- In ADF, create a linked service for Azure Data Lake Storage.

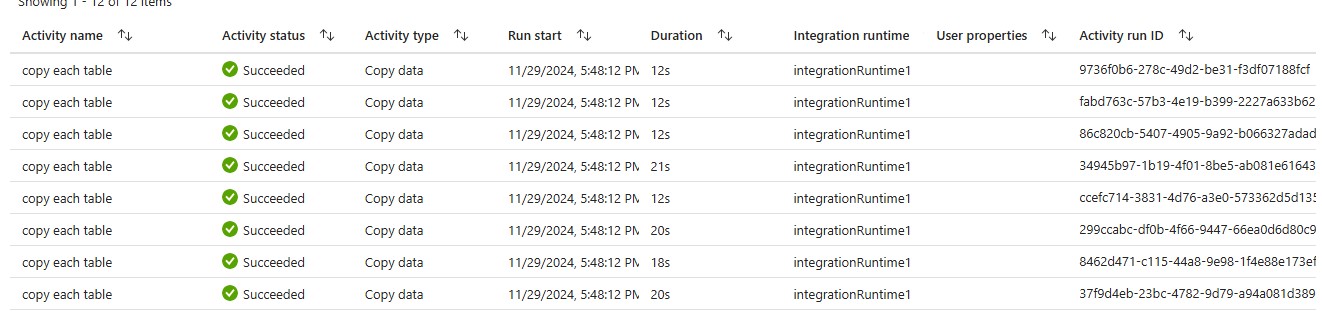
2. \*\*Configure the Destination in Copy Activity\*\*:

* In the "Copy Data" activity, set the destination as the raw folder in ADLS.
* Define the folder structure to store the raw data



3. \*\*Trigger the Pipeline\*\*:

* Run the pipeline manually or schedule it for periodic runs.
* Monitor pipeline execution to ensure successful data transfer.

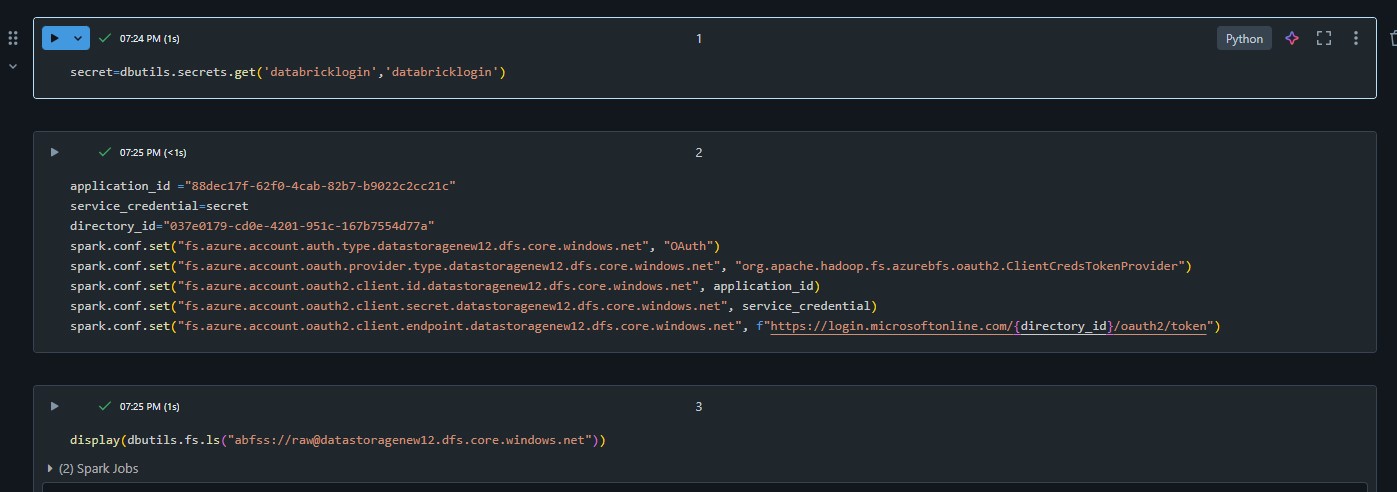


### Step 3: Clean and Process Data with Databricks

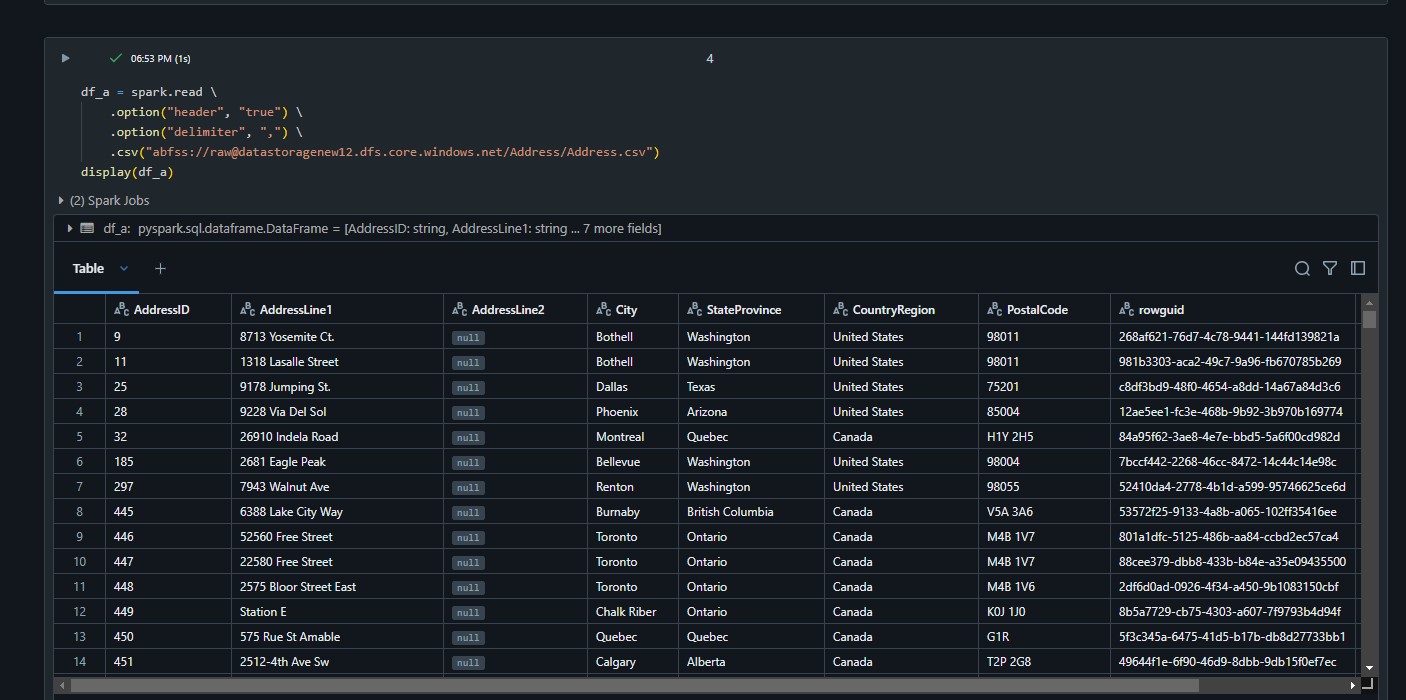
1. ADLS in Databricks\*\*:

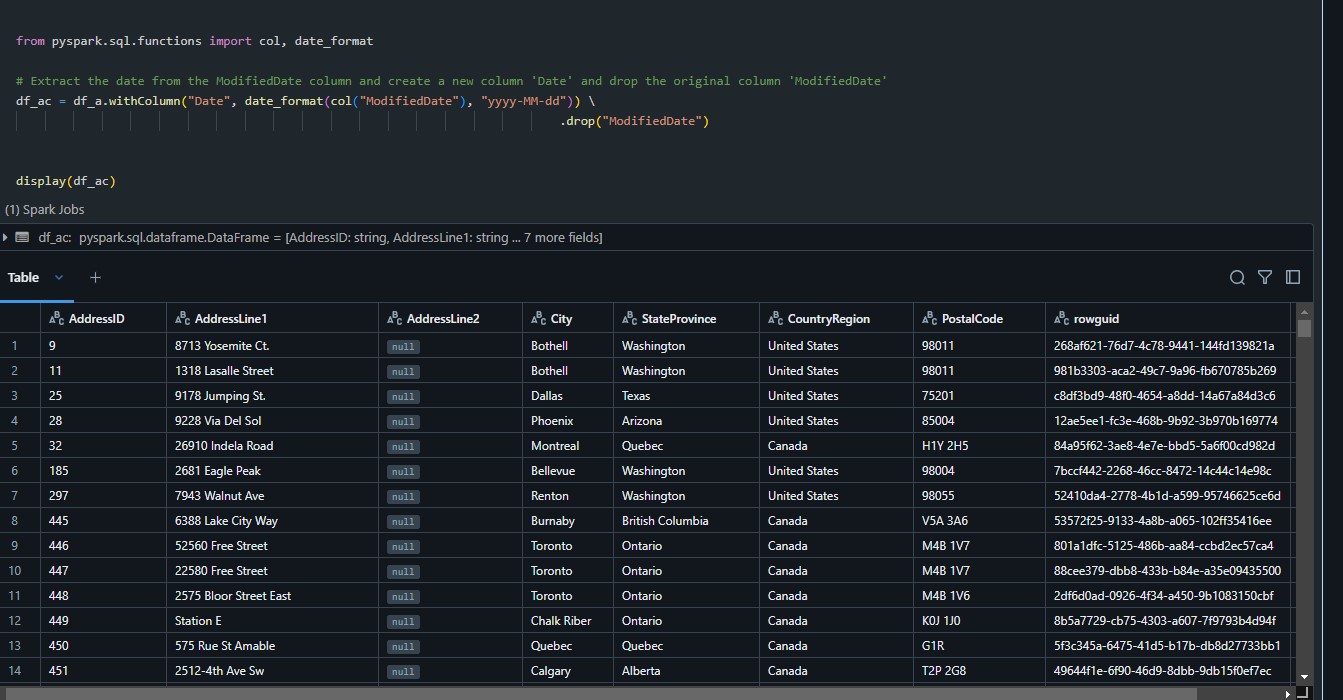
- Use Databricks forADLS raw folder:

1. \*\*Load Raw Data in Databricks Notebook\*\*:



1. \*\*Perform Data Cleaning and Transformation\*\*:



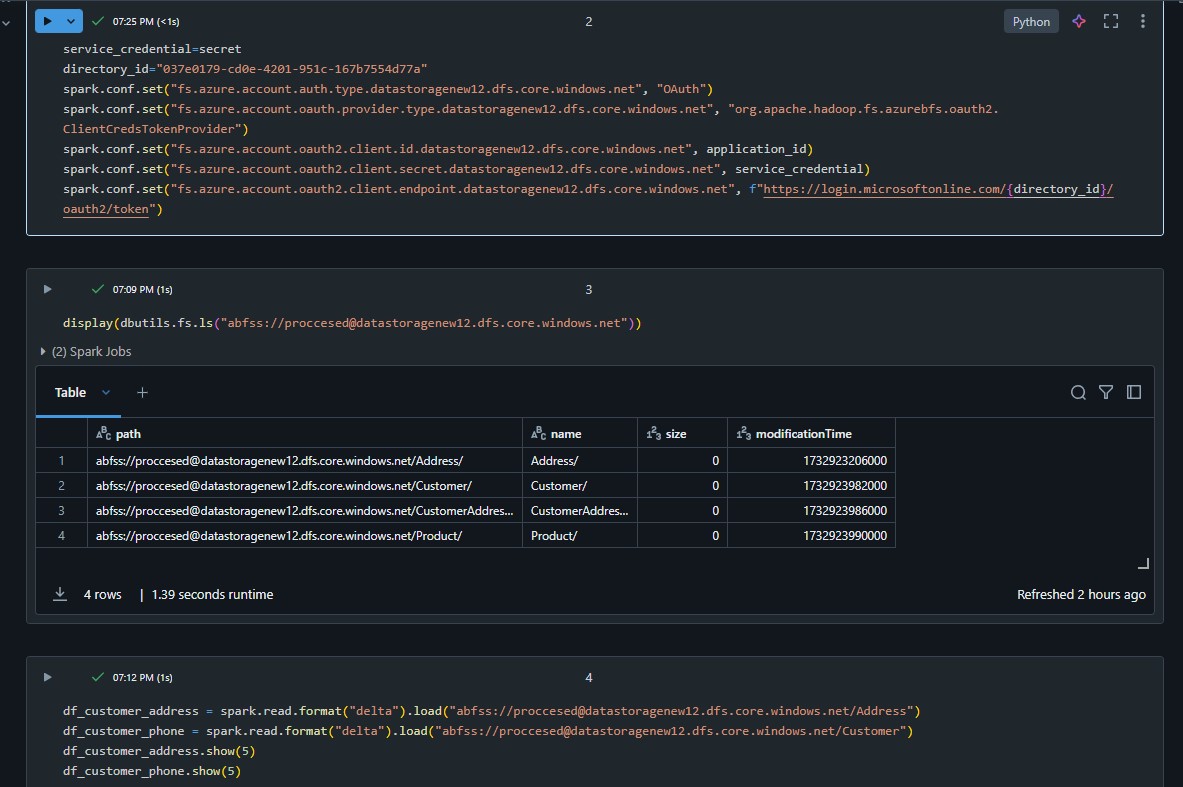


1. \*\*Save Cleaned Data to Processed Folder\*\*:



### Step 4: ETL and Load Data into Gold Zone

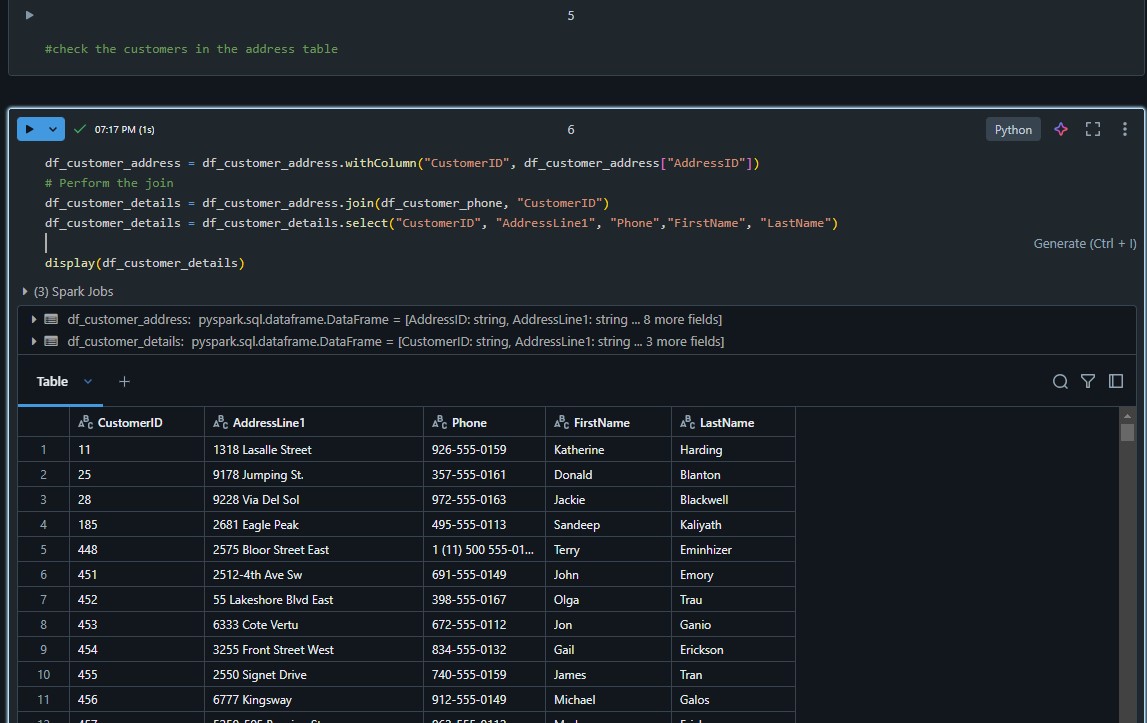
1. \*\*Load Processed Data\*\*:



1. \*\*Perform ETL Transformations\*\*:

- Apply final transformations, aggregations, or calculations for analytics.

--getting the phonenumber , address and name of the customers



3. \*\*Write Final Data to Gold Folder\*\*:

- Save the transformed data to the gold folder:



## Monitoring and Maintenance

1. \*\*Pipeline Monitoring\*\*:

- Use Azure Data Factory's monitoring dashboard to track pipeline execution.

2. \*\*Databricks Job Scheduling\*\*:

- Schedule Databricks notebooks as jobs for periodic processing.

3. \*\*Error Handling\*\*:

- **Data Loading:**

* Wrapped read operations in try-except blocks to handle missing files or permission issues.

**Transformations:**

* Validated the presence of required columns before proceeding.
* Used conditionals to manage unexpected nulls or data type mismatches.

**Writing Data:**

* Checked storage paths' availability and permissions.
* Logged successful writes and flagged failures.

---

## Conclusion

This pipeline enables seamless extraction, transformation, and loading of data from an onpremises SQL database to Azure Data Lake Storage. By leveraging Azure Data Factory and Databricks, the process is automated, scalable, and suitable for advanced analytics and reporting.