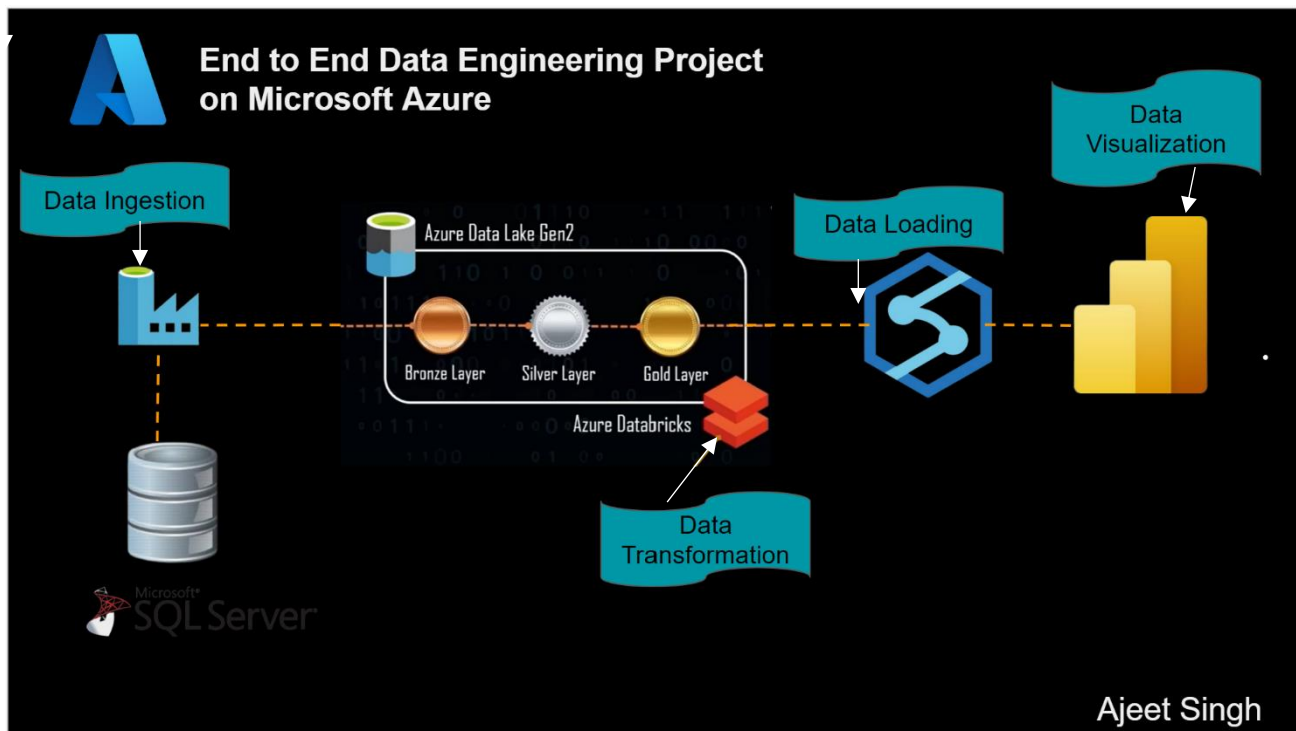


# End-to-End Data Engineering Project

## Objective:

The objective of this project of this project is to build a data pipeline to store data in Data Lake from a local computer, perform different types of transformation, load the data cloud-based database, and connect the data visualization tool to the cloud-based database to perform data visualization and perform end to end data pipeline testing.



## Different steps involved in this project:

- I. Environment Setup
- II. Data Ingestion
- III. Data Transformation
- IV. Data Loading
- V. Data Reporting
- VI. End-to-end Pipeline Testing

## Different tools and technology used in this project:

### MS SQL Server:

MS SQL Server is a Relational Database Management System (RDBMS) developed by Microsoft. A Relational database is based on a Relational Model architecture. The data is

organized in tables(relations), and the tables are related to each other. Each table has rows and columns(attributes). MS SQL Server is a software product used to administer the database and retrieve information.

### **Self-Hosted Integration Runtime:**

A self-hosted integration runtime (SHIR) is a secure, managed, and serverless computing infrastructure that allows data integration between on-premises systems and the Azure cloud. It is a component of Azure Integration Runtime (AIR) that is used when an on-premises data source cannot be accessed directly from the cloud. SHIR serves as a bridge between on-premises and Azure services.

### **Azure Data Factory:**

Azure Data Factory (ADF) is a cloud-based data integration service that allows users to create workflows for data transformation and movement. It's serverless and fully managed and can simplify hybrid data integration at an enterprise scale. ADF is available in over 25 regions and offers a pay-as-you-go service that scales on demand.

### **Azure Data Lake Gen2:**

Azure Data Lake Storage Gen2 (ADLS Gen2) is a service that stores and analyzes big data in its original file formats. It's built on Azure Blob Storage and combines the capabilities of Azure Data Lake Storage Gen1 with Azure Blob Storage. ADLS Gen2 can store text, CSV, JSON, XML, image, and video files. It's designed to work with Hadoop and frameworks that use the Apache Hadoop Distributed File System (HDFS).

### **Azure Synapse Analyst:**

Azure Synapse Analytics is a cloud-based enterprise analytics service that combines SQL technologies, big data, and Azure Data Explorer to speed up time to insight across data warehouses and big data systems. It offers cloud data warehousing, dashboarding, and machine learning analytics. Azure Synapse Analytics is designed for data mining and data exploration and can handle high volumes of data and very complex queries and aggregation. It also supports up to 128 concurrent queries.

### **Azure Key Vault:**

Azure Key Vault is a cloud service that provides a secure store for secrets. You can securely store keys, passwords, certificates, and other secrets. Azure key vaults may be created and managed through the Azure portal. In this QuickStart, you create a key vault and then use it to store a secret.

### **Power BI:**

Microsoft Power BI is a collection of apps, services, and connectors that help users visualize and connect to data from different sources. Power BI can help users make better decisions, and can be used to: Uncover insights with AI, Turn data into visuals, Establish a governed source of truth, Unify enterprise scale and self-service, and Infuse data experiences everywhere.

## Project Explanation:

### Step:1 (Download the Data from Microsoft Learn)

Link of Data: <https://learn.microsoft.com/en-us/sql/samples/adventureworks-install-configure?view=sql-server-ver16&tabs=ssms>

## Download backup files





















Use these links to download the appropriate sample database for your scenario.

- **OLTP** data is for most typical online transaction processing workloads.
- **Data Warehouse (DW)** data is for data warehousing workloads.
- **Lightweight (LT)** data is a lightweight and pared down version of the **OLTP** sample.

If you're not sure what you need, start with the OLTP version that matches your SQL Server version.

OLTP	Data Warehouse	Lightweight
<a href="#">AdventureWorks2022.bak ↗</a>	<a href="#">AdventureWorksDW2022.bak ↗</a>	<a href="#">AdventureWorksLT2022.bak ↗</a>
<a href="#">AdventureWorks2019.bak ↗</a>	<a href="#">AdventureWorksDW2019.bak ↗</a>	<a href="#">AdventureWorksLT2019.bak ↗</a>
<a href="#">AdventureWorks2017.bak ↗</a>	<a href="#">AdventureWorksDW2017.bak ↗</a>	<a href="#">AdventureWorksLT2017.bak ↗</a>
<a href="#">AdventureWorks2016.bak ↗</a>	<a href="#">AdventureWorksDW2016.bak ↗</a>	<a href="#">AdventureWorksLT2016.bak ↗</a>

### Different tables in the dataset:

-   SalesLT.Address
-   SalesLT.Customer
-   SalesLT.CustomerAddress
-   SalesLT.Product
-   SalesLT.ProductCategory
-   SalesLT.ProductDescription
-   SalesLT.ProductModel
-   SalesLT.ProductModelProductDescription
-   SalesLT.SalesOrderDetail
-   SalesLT.SalesOrderHeader

Give permission to the user for read the data:

Select a page

General

Owned Schemas

Membership

Securables

Extended Properties

Script

Help

Database role membership:

Role Members

☐

db\_accessadmin

☐

db\_backupoperator

☒

db\_datareader☐☐☐☐☐☐

## Setup the Self-hosted integration runtime:

Microsoft Integration Runtime Configuration Manager

Home

Settings

Diagnostics

Update

Help

✓

Self-hosted node is connected to the cloud service

Data Factory:

Integration Runtime:

Node:

SelfHostedIntegrationRuntime1

AJEET\_SINGH

Stop Service

Data Source Credential ⓘ

Credential store:

Credential status:

Last backup time:

On-premises

In sync

N/A

Generate Backup

Import Backup

✓ Connected to the cloud service (Data Factory V2)

## Step:2 Data Ingestion


## Edit linked service

 SQL server [Learn more](#) 

Name \*

OnPremSqlServer

Description

Connect via integration runtime \* 



SelfHostedintegrationRuntime1



Connection string

Azure Key Vault

Server name \*

AJEET\_SINGH\SQLEXPRESS

Database name \*

AdventureWorksLT2017

Authentication type

SQL authentication




User name \*

ajeet


Password

Azure Key Vault

AKV linked service \* 

AzureKeyVault1



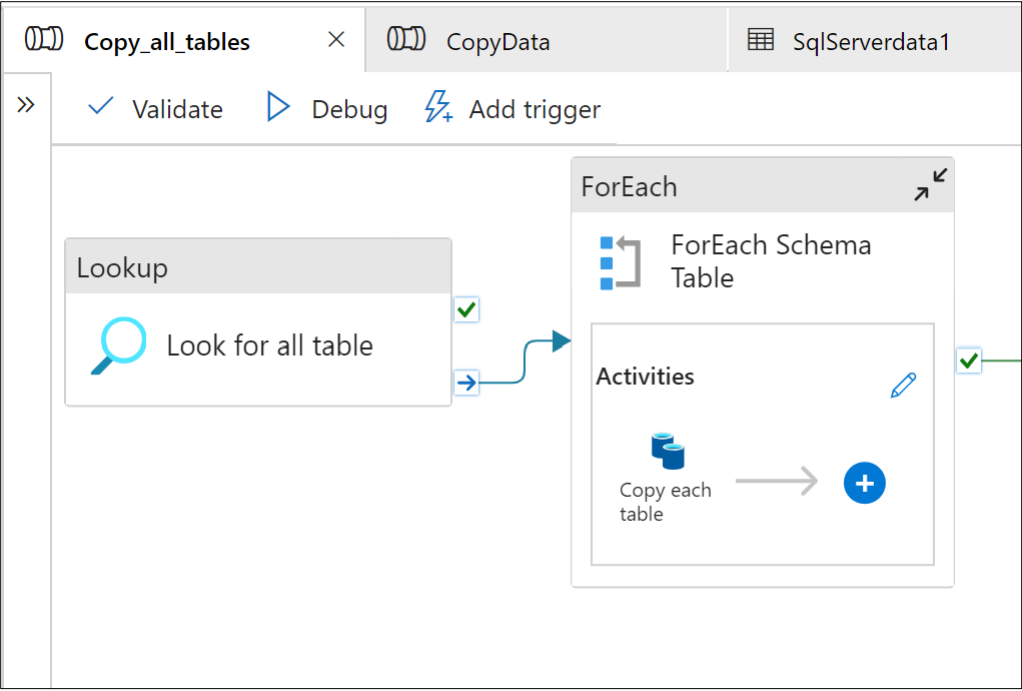
Secret name \* 

password



Edit

Secret version 



+ Container    Change access level    Restore containers    Refresh    Delete    Give feedback

Search containers by prefix    Show deleted containers

Name	Last modified	Anonymous access level	Lease state
<input type="checkbox"/> \$logs	3/4/2024, 2:33:53 AM	Private	Available ***
<input type="checkbox"/> bronze	3/4/2024, 2:40:29 AM	Private	Available ***
<input type="checkbox"/> gold	3/4/2024, 11:04:40 AM	Private	Available ***
<input type="checkbox"/> silver	3/4/2024, 11:04:18 AM	Private	Available ***

bronze    Container

Search    Upload    Add Directory    Refresh    Rename    Delete    Change tier    Acquire lease    Break lease    Give feedback

Overview

Diagnose and solve problems

Access Control (IAM)

Settings

Shared access tokens

Manage ACL

Access policy

Properties

Metadata

Authentication method: Access key (Switch to Microsoft Entra user account)

Location: bronze / SalesLT

Search blobs by prefix (case-sensitive)    Show deleted objects

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
<input type="checkbox"/> [-]						
<input type="checkbox"/> Address						-
<input type="checkbox"/> Customer						-
<input type="checkbox"/> CustomerAddress						-
<input type="checkbox"/> Product						-
<input type="checkbox"/> ProductCategory						-
<input type="checkbox"/> ProductDescription						-
<input type="checkbox"/> ProductModel						-
<input type="checkbox"/> ProductModelProductDescription						-
<input type="checkbox"/> SalesOrderDetail						-
<input type="checkbox"/> SalesOrderHeader						-

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
<input type="checkbox"/> [-]						...
<input type="checkbox"/> Address.parquet	3/16/2024, 4:17:35 PM	Hot (Inferred)		Block blob	34.75 KiB	Available ...

Step:-3 Data Transformation:

# Pick up where you left off

[Recents](#)[Favorites](#)**silver to gold**

Notebook · 22 days ago

**bronze to silver**

Notebook · 22 days ago

**storageMount**

Notebook · 22 days ago

## mounting

▶

Last execution failed

1

Python

+

⌵

⋮

```
1  configs = {
2      "fs.azure.account.auth.type": "CustomAccessToken",
3      "fs.azure.account.custom.token.provider.class": spark.conf.get("spark.databricks.passthrough.adls.gen2.tokenProviderClassName")
4  }
5
6  # Optionally, you can add <directory-name> to the source URI of your mount point.
7  dbutils.fs.mount(
8      source = "abfss://bronze@salesproject1storage.dfs.core.windows.net/",
9      mount_point = "/mnt/bronze",
10     extra_configs = configs)

```

ⓘ

> java.rmi.RemoteException: java.lang.IllegalArgumentException: requirement failed: Directory already mounted: /mnt/bronze; nested exception is:

🔍 Diagnose error

▶

Last execution failed

3

Python

+

⌵

⋮

```
1  configs = {
2      "fs.azure.account.auth.type": "CustomAccessToken",
3      "fs.azure.account.custom.token.provider.class": spark.conf.get("spark.databricks.passthrough.adls.gen2.tokenProviderClassName")
4  }
5
6  # Optionally, you can add <directory-name> to the source URI of your mount point.
7  dbutils.fs.mount(
8      source = "abfss://silver@salesproject1storage.dfs.core.windows.net/",
9      mount_point = "/mnt/silver",
10     extra_configs = configs)

```

ⓘ

> java.rmi.RemoteException: java.lang.IllegalArgumentException: requirement failed: Directory already mounted: /mnt/silver; nested exception is:

🔍 Diagnose error

▶

Last execution failed

4

Python

+

⌵

⋮

```
1  configs = {
2      "fs.azure.account.auth.type": "CustomAccessToken",
3      "fs.azure.account.custom.token.provider.class": spark.conf.get("spark.databricks.passthrough.adls.gen2.tokenProviderClassName")
4  }
5
6  # Optionally, you can add <directory-name> to the source URI of your mount point.
7  dbutils.fs.mount(
8      source = "abfss://gold@salesproject1storage.dfs.core.windows.net/",
9      mount_point = "/mnt/gold",
10     extra_configs = configs)

```

ⓘ

> java.rmi.RemoteException: java.lang.IllegalArgumentException: requirement failed: Directory already mounted: /mnt/gold; nested exception is:

🔍 Diagnose error

## Silver level transformation



✓ 3/13/2024 (22s)

11

```
from pyspark.sql.functions import from_utc_timestamp, date_format
from pyspark.sql.types import TimestampType

# Assuming you have a list of table names
#table_names = [...]

for i in table_name:
    path = '/mnt/bronze/SalesLT/' + i + '/' + i + '.parquet'

    # Read the Parquet file
    df = spark.read.format('parquet').load(path)

    # Get column names
    columns = df.columns

    # Process date columns
    for col in columns:
        if "Date" in col or "date" in col:
            df = df.withColumn(col, date_format(from_utc_timestamp(df[col].cast(TimestampType()), "UTC"), "yyyy-MM-dd"))

    # Write the processed DataFrame to Delta format
    output_path = '/mnt/silver/SalesLT/' + i + '/'
    df.write.format('delta').mode("overwrite").save(output_path)
```

▶ (50) Spark Jobs

▶ df: pyspark.sql.dataframe.DataFrame = [SalesOrderID: integer, RevisionNumber: integer ... 20 more fields]



✓ 3/13/2024 (< 1s)

12

display(df)

▶ (1) Spark Jobs

Table

New result table: ON

Search

	<div><div></div><div>SalesOrderID</div></div>	<div><div></div><div>RevisionNumber</div></div>	<div><div></div><div>OrderDate</div></div>	<div><div></div><div>DueDate</div></div>	<div><div></div><div>ShipDate</div></div>	<div><div></div><div>Status</div></div>	<div><div></div><div>OnlineOrderFlag</div></div>	<div><div></div><div>SalesC</div></div>
12	71832	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71832
13	71845	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71845
14	71846	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71846
15	71856	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71856
16	71858	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71858
17	71863	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71863
18	71867	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71867
19	71885	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71885
20	71895	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71895
21	71897	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71897
22	71898	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71898
23	71899	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71899
24	71902	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71902
25	71915	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71915

32 rows | 0.20 seconds runtime

Refreshed 25 days ago

## Gold level transformation





▶ 3/13/2024 (<1s)

4

```
df = spark.read.format('delta').load('/mnt/silver/SalesLT/Address/')
```

▼ df: pyspark.sql.dataframe.DataFrame

Schema Details History

```
AddressID: integer
AddressLine1: string
AddressLine2: string
City: string
StateProvince: string
CountryRegion: string
PostalCode: string
rowguid: string
ModifiedDate: string
```

▶ 3/13/2024 (<1s)

7

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, regexp_replace

# Assuming you have a SparkSession created
#spark = SparkSession.builder.appName("example").getOrCreate()

# Assuming you have a DataFrame named df
# If not, you need to read your data first using df = spark.read.

# Get the list of column names
column_names = df.columns

for old_col_name in column_names:
    # Convert column name from ColumnName to Column_Name format
    new_col_name = "".join(["_" + char if char.isupper() and not old_col_name[i - 1].isupper() else char for i, char in enumerate(old_col_name)]).lstrip("_")
    # Change the column name using withColumnRenamed and regexp_replace
    df = df.withColumnRenamed(old_col_name, new_col_name)
```

▶

3/13/2024 (<1s)

8

Python

display(df)

▶ (1) Spark Jobs

Table

+

New result table: ON

Search

🔍

📄

	Address_ID	Address_Line1	Address_Line2	City	State_Province	Country_Region	Postal_C
54	491	20225 Lansing Ave	null	Montreal	Quebec	Canada	H1Y 2H7
55	492	99954 Boul. Laurier, Local 060, Place	null	Sainte-Foy	Quebec	Canada	G1W
56	493	6th Floor 5250 Main Street	null	Winnipeg	Manitoba	Canada	R3
57	494	Box 99354 300 Union Street	null	Saint John	Brunswick	Canada	E2P 1E3
58	495	Suite 800 2530 Slater Street	null	Ottawa	Ontario	Canada	K4B 1T7
59	496	Suite 500 995 W. 11th Avenue	null	Mississauga	Ontario	Canada	L5A 1H6
60	497	#9900 2700 Production Way	null	Burnaby	British Columbia	Canada	V5A 4X1
61	498	25537 Hillside Avenue	null	Victoria	British Columbia	Canada	V8V
62	499	Suite 2502 410 Albert Street	null	Waterloo	Ontario	Canada	N2V
63	500	5700 Explorer Drive	null	Mississauga	Ontario	Canada	L4W 5J3
64	501	965 De La Gauchetiere West	null	Montreal	Quebec	Canada	H1Y 2H8
65	502	No. 25400 10665 Jasper Avenue	null	Edmonton	Alberta	Canada	T5
66	503	P.O. Box 44000	null	Winnipeg	Manitoba	Canada	R3
67	504	Suite 99320 255 - 510th Avenue S.W.	null	Calgary	Alberta	Canada	T2P 2G8

450 rows | 0.44 seconds runtime

Refreshed 25 days ago

▶

3/13/2024 (17s)

12

Python

```

print(path)
# Read the Delta file
df = spark.read.format('delta').load(path)

# Get list of column names
columns_names = df.columns

# Process date columns
for old_col_name in columns_names:
    # Convert column name from to Column_Name format
    new_col_name = "".join(["_" + char if char.isupper() and not old_col_name[i - 1].isupper() else char for i, char in enumerate(old_col_name)]).lstrip("_")
    # Update column name
    df = df.withColumnRenamed(old_col_name, new_col_name)

# Write the processed DataFrame to Delta format
output_path = '/mnt/gold/SalesLT/' + name + '/'
df.write.format('delta').mode("overwrite").save(output_path)

```

3/13/2024 (<1s)

13

display(df)

(1) Spark Jobs

Table

New result table: ON

Search

	<sup>1</sup> <sub>3</sub> Sales_Order_ID	<sup>1</sup> <sub>3</sub> Revision_Number	<sup>A</sup> <sub>C</sub> Order_Date	<sup>A</sup> <sub>C</sub> Due_Date	<sup>A</sup> <sub>C</sub> Ship_Date	<sup>1</sup> <sub>3</sub> Status	<sup>≡</sup> Online_Order_Flag	
12	71832	2	2008-06-01	2008-06-13	2008-06-08	5	false	St
13	71845	2	2008-06-01	2008-06-13	2008-06-08	5	false	St
14	71846	2	2008-06-01	2008-06-13	2008-06-08	5	false	St
15	71856	2	2008-06-01	2008-06-13	2008-06-08	5	false	St
16	71858	2	2008-06-01	2008-06-13	2008-06-08	5	false	St
17	71863	2	2008-06-01	2008-06-13	2008-06-08	5	false	St
18	71867	2	2008-06-01	2008-06-13	2008-06-08	5	false	St
19	71885	2	2008-06-01	2008-06-13	2008-06-08	5	false	St
20	71895	2	2008-06-01	2008-06-13	2008-06-08	5	false	St
21	71897	2	2008-06-01	2008-06-13	2008-06-08	5	false	St
22	71898	2	2008-06-01	2008-06-13	2008-06-08	5	false	St
23	71899	2	2008-06-01	2008-06-13	2008-06-08	5	false	St
24	71902	2	2008-06-01	2008-06-13	2008-06-08	5	false	St
25	71915	2	2008-06-01	2008-06-13	2008-06-08	5	false	St

32 rows | 0.35 seconds runtime

Refreshed 25 days ago

Delta format:

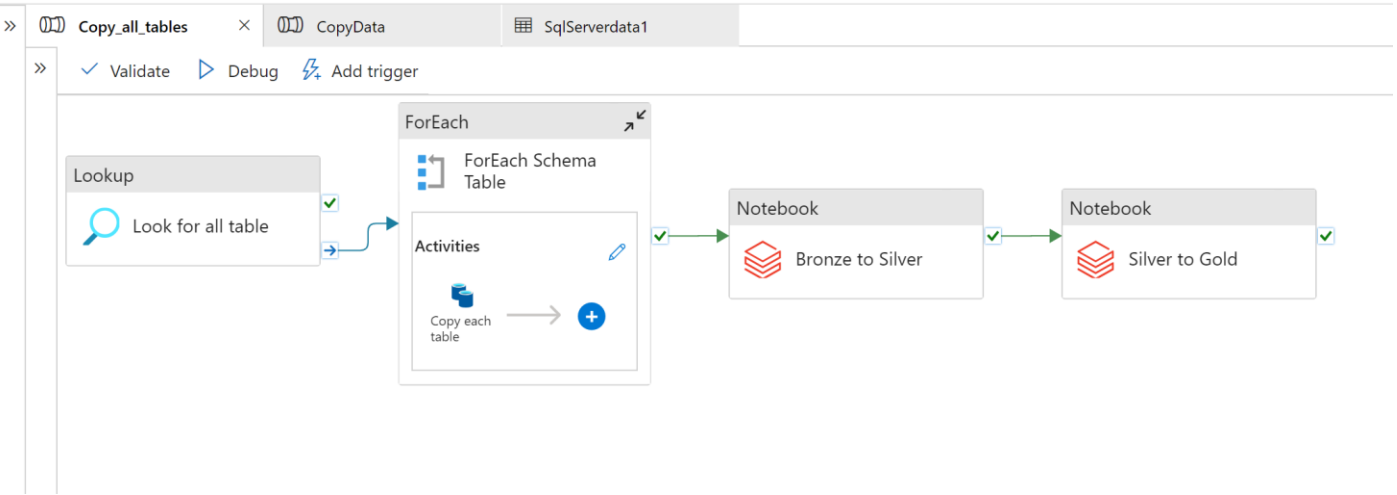
UploadAdd DirectoryRefreshRenameDeleteChange tierAcquire leaseBreak leaseGive feedback

Authentication method: Access key (Switch to Microsoft Entra user account)

Location: gold / SalesLT / Address

Search blobs by prefix (case-sensitive)Show deleted objects

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
<input type="checkbox"/> [.]						
<input type="checkbox"/> _delta_log						-
<input type="checkbox"/> part-00000-40a7f9b5-edea-4e6a-b292-e0af768a5282.c00...	3/16/2024, 4:18:39 PM	Hot (Inferred)		Block blob	34.33 KiB	Available
<input type="checkbox"/> part-00000-c2551fbd-4ecd-426c-b765-08b492df3de9.c00...	3/13/2024, 11:54:47 PM	Hot (Inferred)		Block blob	34.33 KiB	Available
<input type="checkbox"/> part-00000-d6bcbcbd-489a-466b-a427-112a070df1d2.c00...	3/16/2024, 3:02:19 PM	Hot (Inferred)		Block blob	34.33 KiB	Available
<input type="checkbox"/> part-00000-ef95bbd9-8a37-4308-90a2-12d972bbc8e3.c00...	3/13/2024, 11:12:30 PM	Hot (Inferred)		Block blob	34.33 KiB	Available

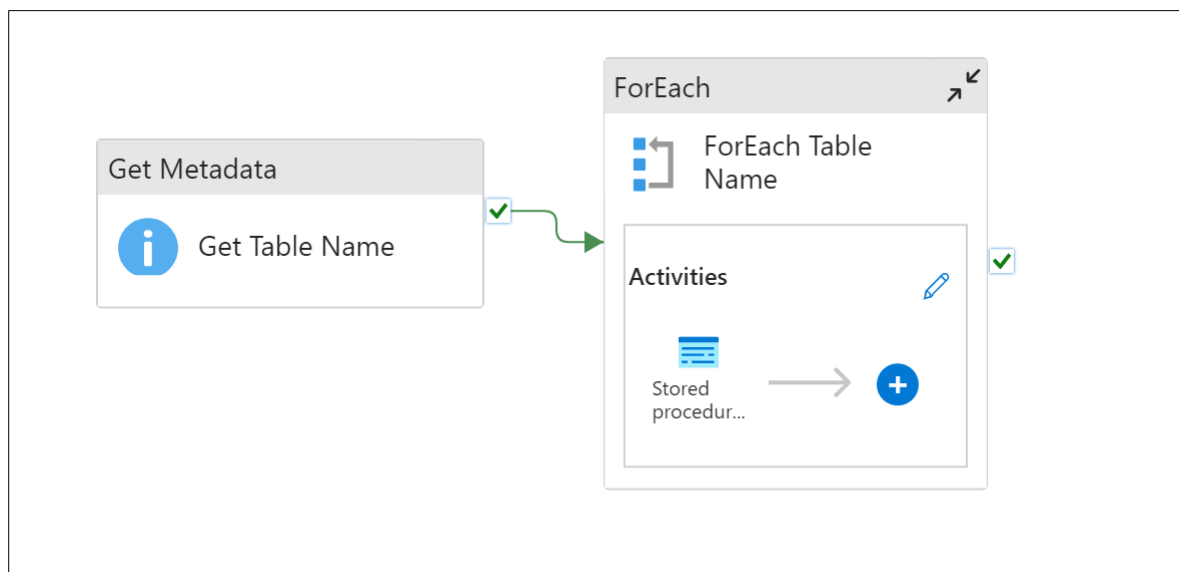


## Step:4 Data Loading

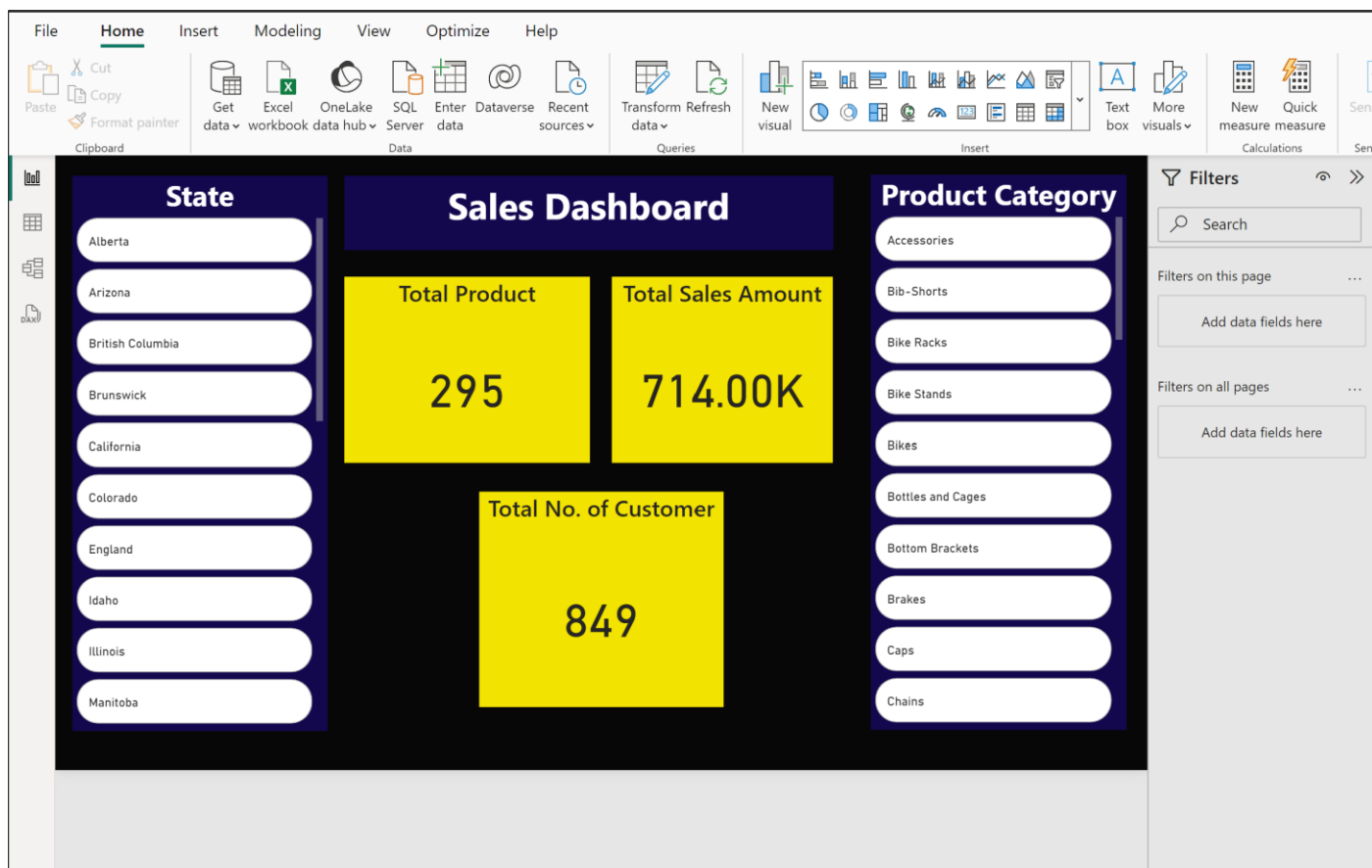
The screenshot displays the SQL Server Data Tools (SSDT) interface. On the left, the 'Data' pane shows the 'Workspace' tab with a tree view of the 'myGold\_db (SQL)' database. The 'Views' folder is expanded, showing a list of tables including 'dbo.ADDRESS', 'dbo.Customer', 'dbo.CustomerAddress', 'dbo.Product', 'dbo.ProductCategory', 'dbo.ProductDescription', 'dbo.ProductModel', 'dbo.ProductModelProductD...', 'dbo.SalesOrderDetail', 'dbo.SalesOrderHeader', 'System views', 'Schemas', and 'Security'. The main editor shows a SQL script for creating a view named 'CreateSQLServerlessview\_gold'. The script is as follows:

```
1 USE myGold_db
2 GO
3 CREATE OR ALTER PROC CreateSQLServerlessview_gold @viewName nvarchar(100)
4 AS
5 BEGIN
6
7 DECLARE @statement VARCHAR(MAX)
8 SET @statement = N'CREATE OR ALTER VIEW ' + @viewName + ' AS
9     SELECT *
10    FROM OPENROWSET(
11        BULK 'https://salesproject1storage.dfs.core.windows.net/gold/SalesLT/' + @viewName + '/',
12        FORMAT = 'DELTA'
13    ) AS [result]'
14
15 EXEC (@statement)
16
17 END
18 GO
19
```

The script is executed against the 'myGold\_db' database. A warning message at the top states: 'Other users in your workspace may have access to modify this item. Do not use this item unless you trust all users who may have access to the workspace.'



## Step-5 Data Reporting



## Step- 6 Pipeline Testing

If I add a new row in the customer table the Power BI report will show the total no. of customers increased. From 848 to 849