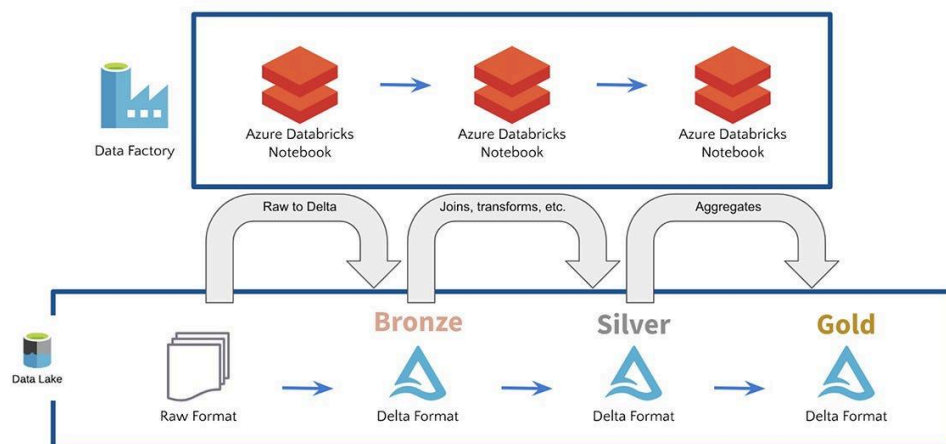
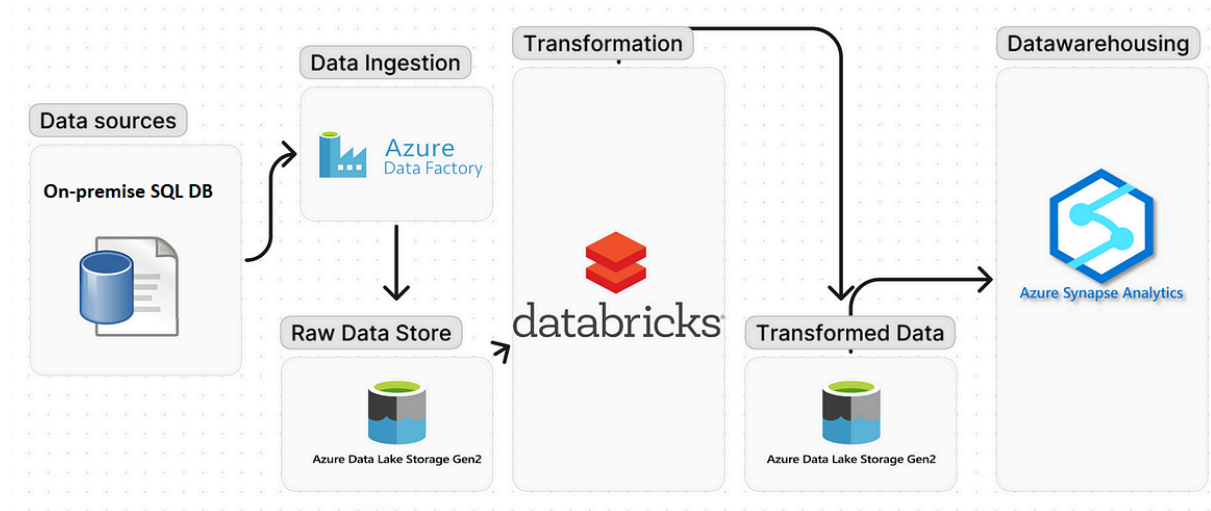


Project 2

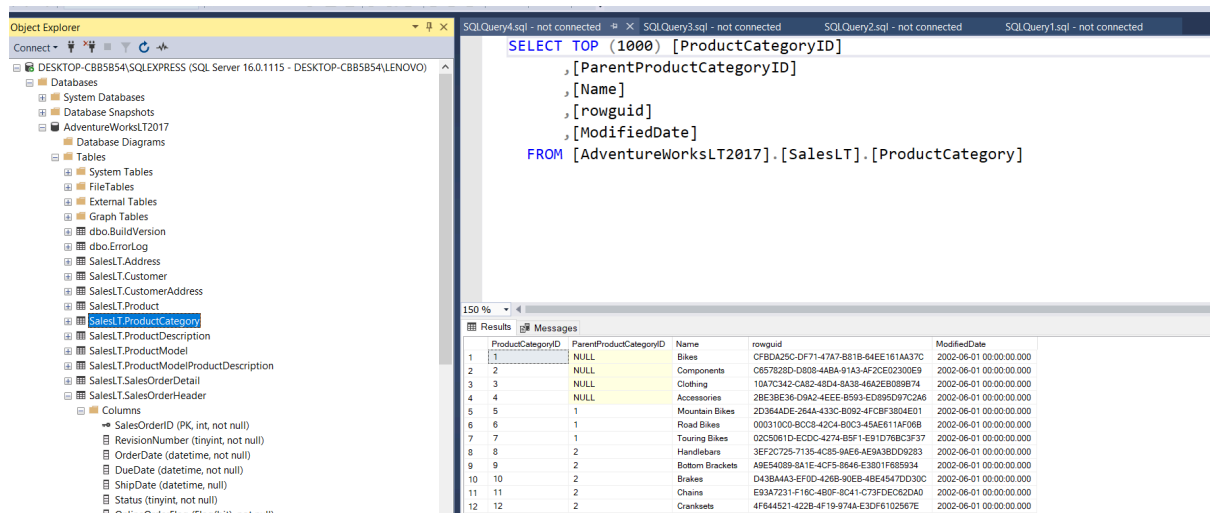
Project Architecture:



Part 1 - Dataset Overview:

AdventureWorks database supports standard online transaction processing scenarios for a fictitious bicycle manufacturer - Adventure Works Cycles. Scenarios include Manufacturing, Sales, Purchasing, Product Management, Contact Management, and Human Resources.

Mahesh Raut - Project - 31 May 2024



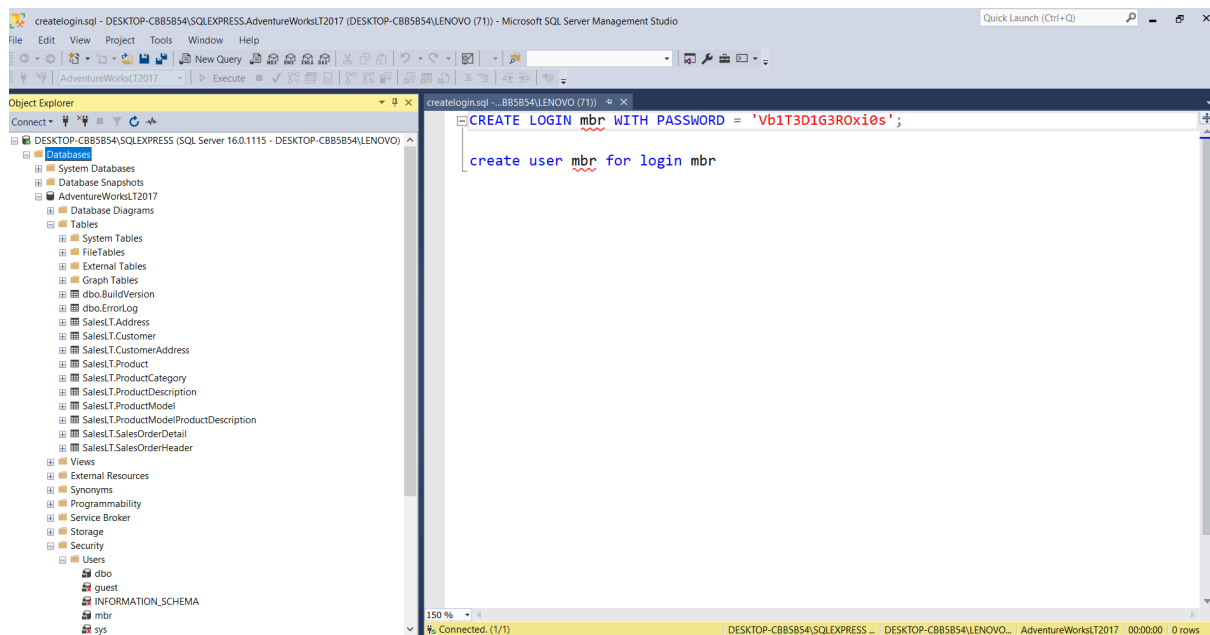
The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for AdventureWorksLT2017. The main pane shows a SQL query and its results.

```
SELECT TOP (1000) [ProductCategoryID]
, [ParentProductCategoryID]
, [Name]
, [rowguid]
, [ModifiedDate]
FROM [AdventureWorksLT2017].[SalesLT].[ProductCategory]
```

	ProductCategoryID	ParentProductCategoryID	Name	rowguid	ModifiedDate
1	1	NULL	Bikes	CF8D425C-DF71-47A7-881B-64EE161AA37C	2002-06-01 00:00:00.000
2	2	NULL	Components	C657B28D-D809-48BA-91A3-AF2CE02300E9	2002-06-01 00:00:00.000
3	3	NULL	Clothing	10A7C342-CAB2-48D4-8A38-46A2EB089874	2002-06-01 00:00:00.000
4	4	NULL	Accessories	28E3BE36-D9A2-4EEF-B593-ED895D97C2A6	2002-06-01 00:00:00.000
5	5	1	Mountain Bikes	2D364ADE-264A-433C-B092-4FCBF3804E01	2002-06-01 00:00:00.000
6	6	1	Road Bikes	000310C0-8CC8-42C4-80C3-45AE611AF06B	2002-06-01 00:00:00.000
7	7	1	Touring Bikes	02C5081D-ECDC-4274-B9F1-4B91D78BC937	2002-06-01 00:00:00.000
8	8	2	Handlebars	3EF2C725-7135-4C85-86C5-AE9A38D090B3	2002-06-01 00:00:00.000
9	9	2	Bottom Brackets	A6E54089-8A1E-4CF5-8846-E3801F685934	2002-06-01 00:00:00.000
10	10	2	Brakes	D4384A43-EF0D-426B-90E8-4BE4547DD30C	2002-06-01 00:00:00.000
11	11	2	Chains	E93A7231-F16C-4B0F-8C41-C73FDEC82DAD	2002-06-01 00:00:00.000
12	12	2	Cranksets	4F644521-422B-4F19-974A-E3DF6102567E	2002-06-01 00:00:00.000

Create Database login

```
CREATE LOGIN mbr WITH PASSWORD = 'Vb1T3D1G3R0xi0s';
createuser mbr for login mbr
```

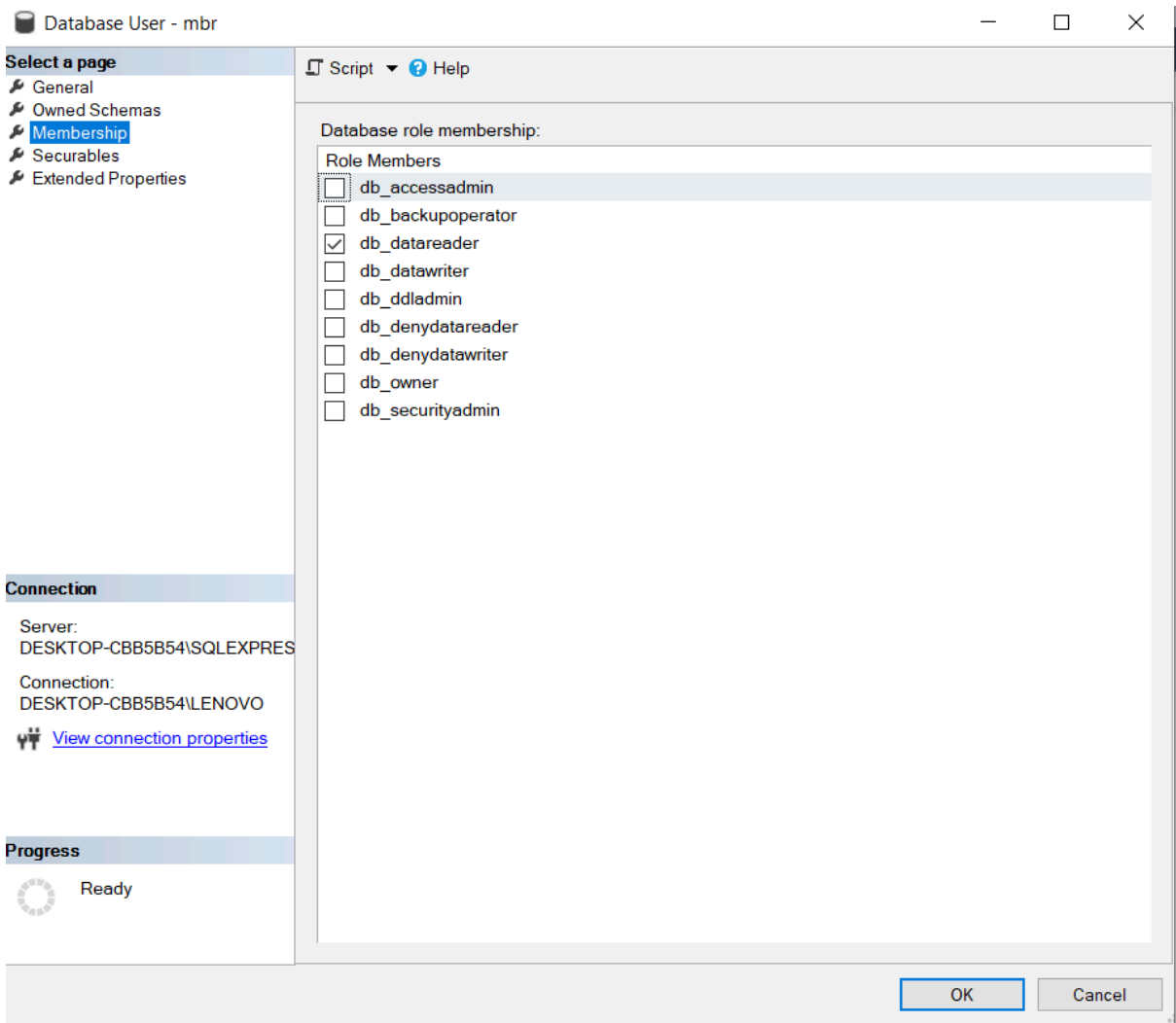


The screenshot shows the SQL Server Enterprise Manager interface with a script executed. The script creates a login and a user.

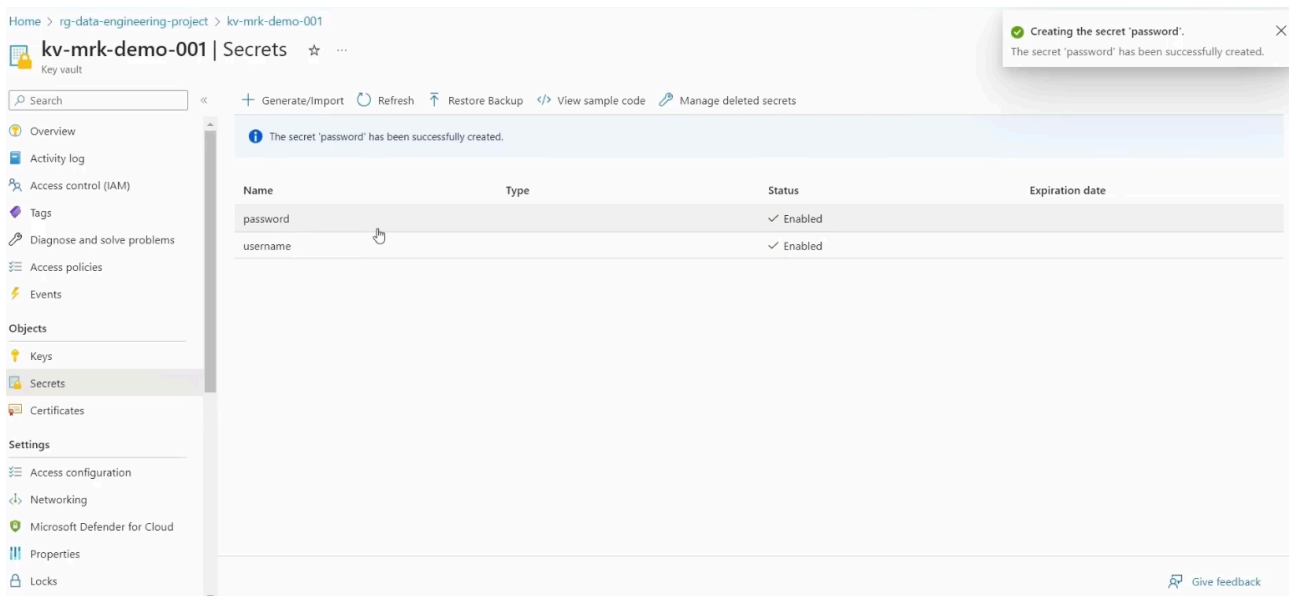
```
CREATE LOGIN mbr WITH PASSWORD = 'Vb1T3D1G3R0xi0s';
create user mbr for login mbr
```

The status bar at the bottom indicates the script was executed successfully, connecting to the AdventureWorksLT2017 database.

Give User access:



Store secrets in Azure Key Vault:



Part 2 - Data Ingestion: From On-premise to Azure Data Lake Storage (Bronze container) (SQL DB to Parquet Format)

getschema.sql

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'AdventureWorksLT2017' database is expanded, showing its tables. On the right, a query window titled 'getschema.sql - loc...ksLT2017 (mrk (65))' contains the following SQL query:

```
SELECT
s.name AS SchemaName,
t.name AS TableName
FROM sys.tables t
INNER JOIN sys.schemas s
ON t.schema_id = s.schema_id
WHERE s.name = 'SalesLT'
```

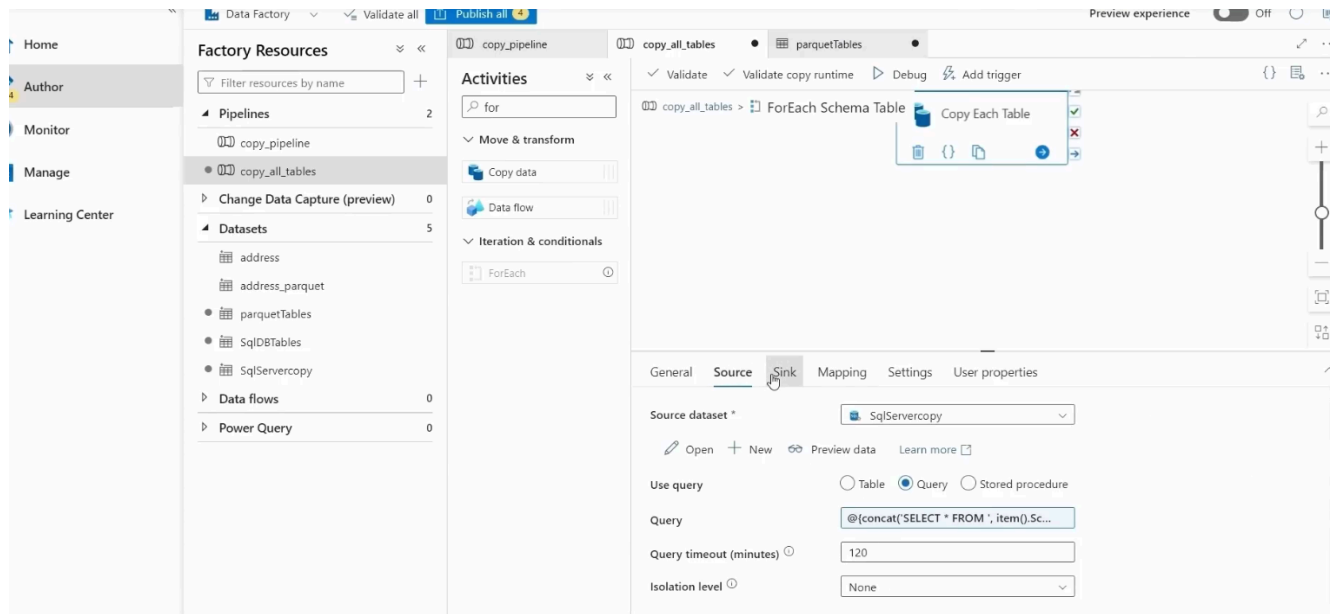
Below the query window, the 'Results' tab displays the output of the query as a table with 10 rows and 2 columns: SchemaName and TableName.

	SchemaName	TableName
1	SalesLT	Address
2	SalesLT	Customer
3	SalesLT	CustomerAddress
4	SalesLT	Product
5	SalesLT	ProductCategory
6	SalesLT	ProductDescription
7	SalesLT	ProductModel
8	SalesLT	ProductModelProductDescription
9	SalesLT	SalesOrderDetail
10	SalesLT	SalesOrderHeader

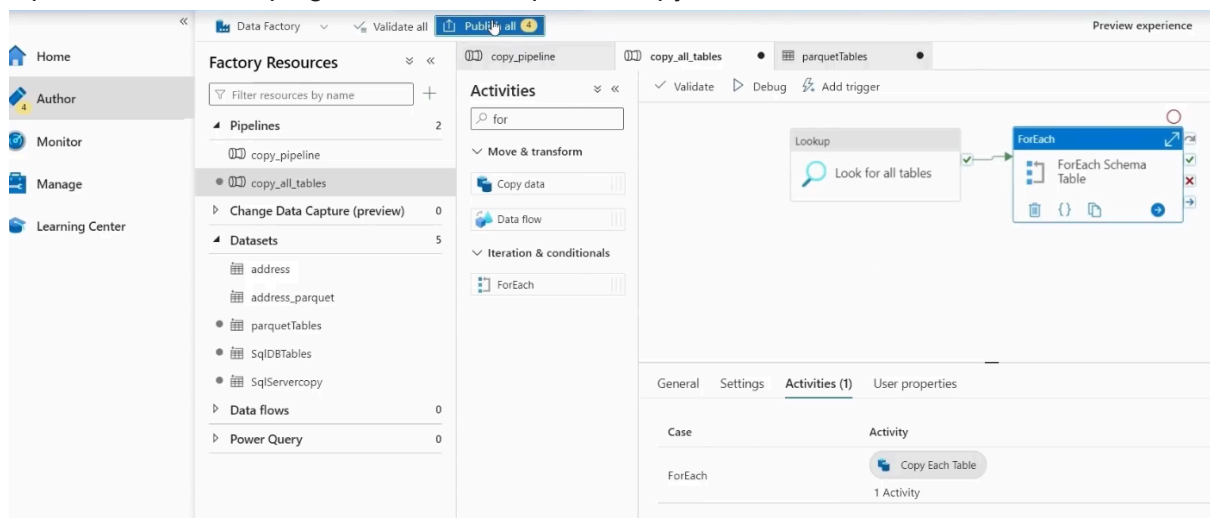
Folder Structure in Data Lake:

bronze/Schema/Tablename/Tablename.parquet
Ex. bronze/SalesLT/Address/Address.parquet

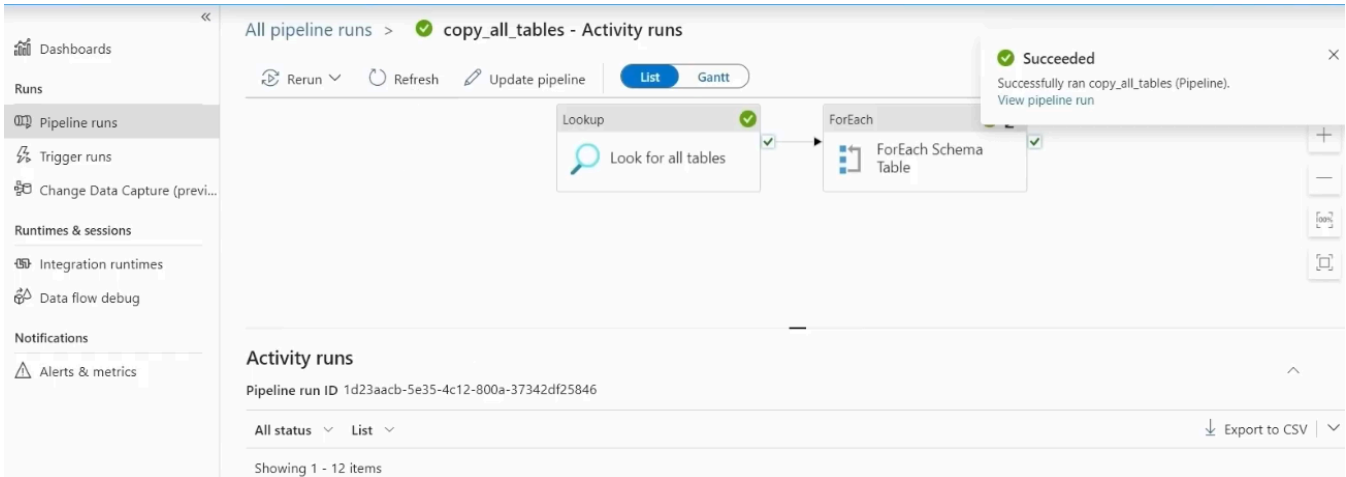
Source dataset - sql db



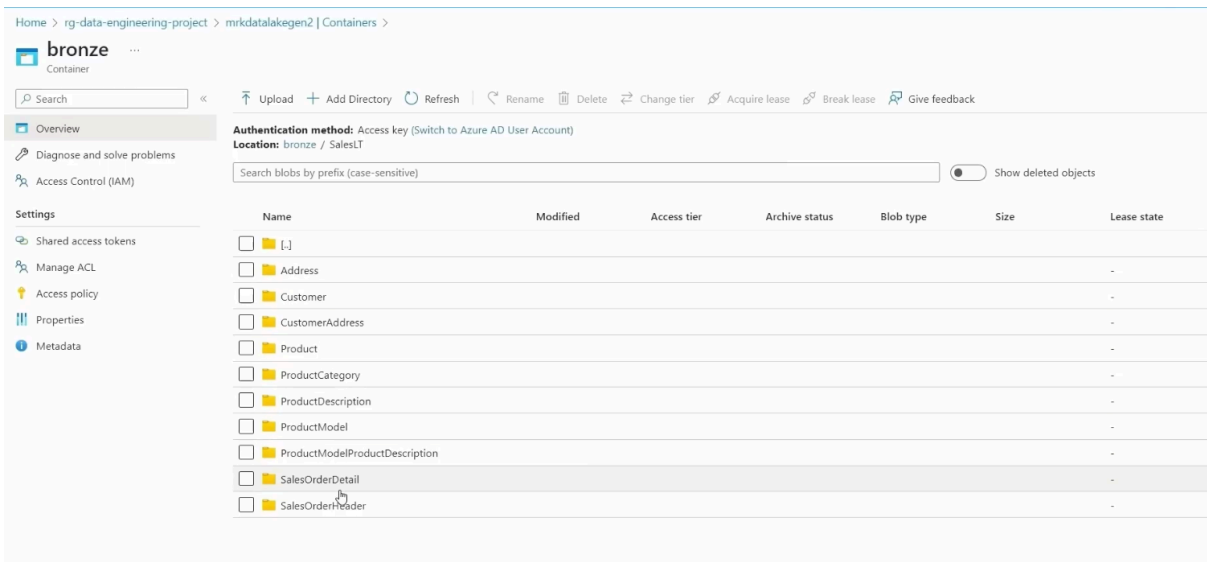
Pipeline build for looping each table in sql db to copy all tables in bronze container



Trigger pipeline once - copy all tables

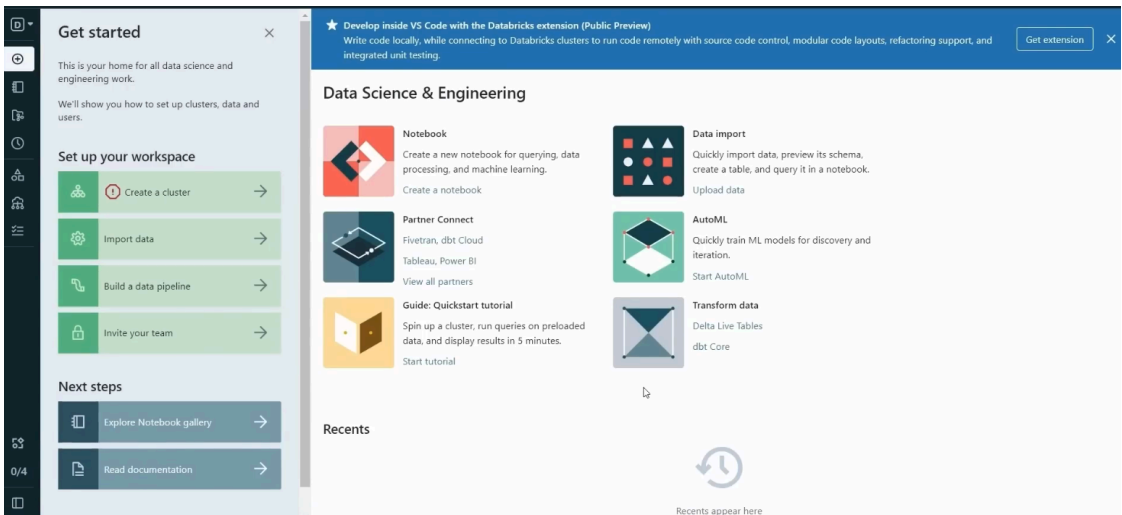


Check bronze container




Part 3 - Data Transformation


Azure Databricks Overview





Create Compute Cluster

[Clusters / New Compute](#) [UI preview](#) [Provide feedback](#)

data_transformation 

Standard_DS3_v214 GB Memory, 4 Cores

☒ Terminate after minutes of inactivity


Tags 

Add tags

Add


> Automatically added tags

▼ Advanced options

Azure Data Lake Storage credential passthrough 

☒ Enable credential passthrough for user-level data access

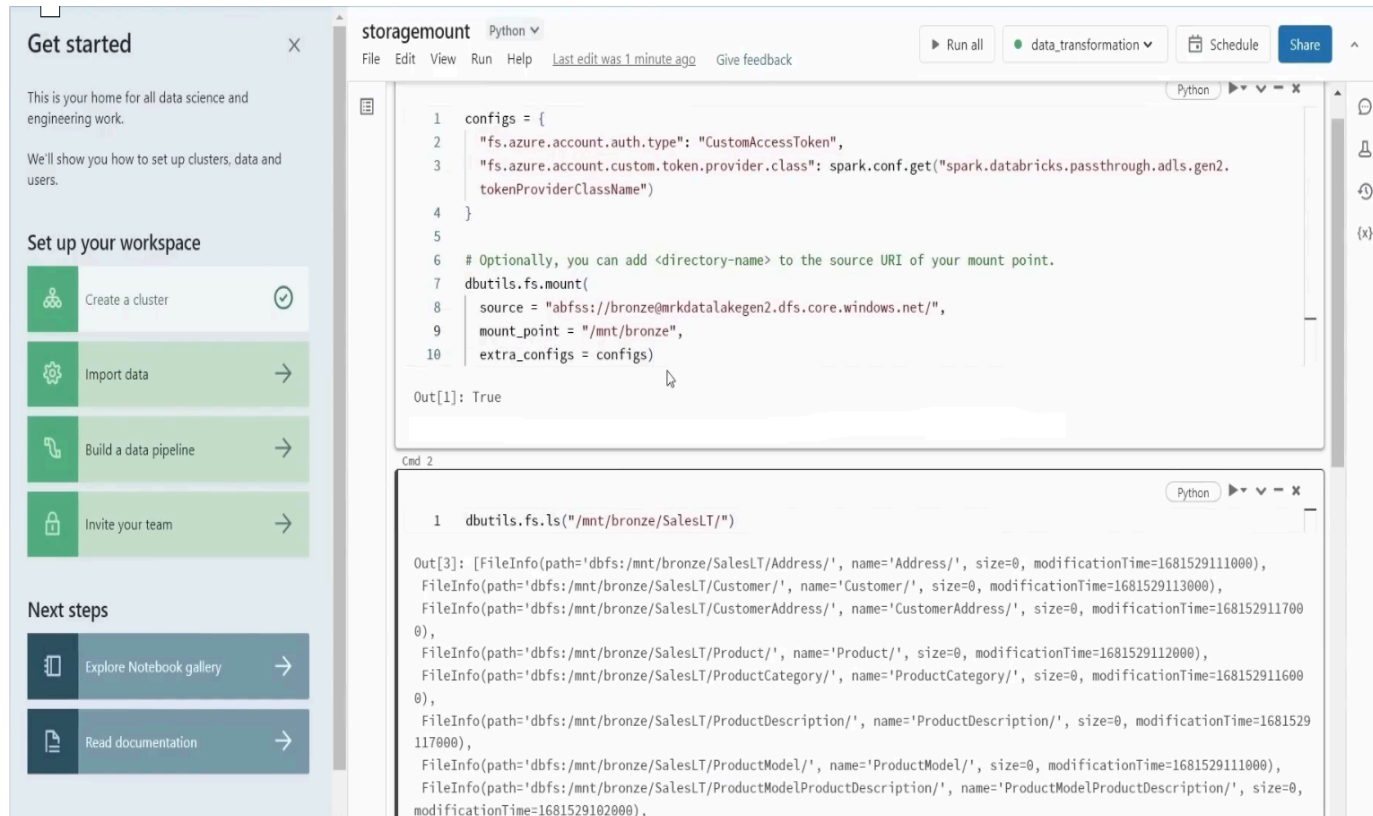
SparkLoggingInit Scripts

Spark config 

```
spark.master local[*, 4]
spark.databricks.cluster.profile singleNode
```

Create ClusterCancel

Storagemount notebook - bronze, silver, gold



Get started

This is your home for all data science and engineering work.

We'll show you how to set up clusters, data and users.

Set up your workspace

- Create a cluster
- Import data
- Build a data pipeline
- Invite your team

Next steps

- Explore Notebook gallery
- Read documentation

storagemount Python

File Edit View Run Help Last edit was 1 minute ago Give feedback

Run all data_transformation Schedule Share

```
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@mrkdatalakegen2.dfs.core.windows.net/",
9     mount_point = "/mnt/bronze",
10    extra_configs = configs)

Out[1]: True
```

Cmd 2

```
1 dbutils.fs.ls("/mnt/bronze/SalesLT/")

Out[3]: [FileInfo(path='dbfs:/mnt/bronze/SalesLT/Address/', name='Address/', size=0, modificationTime=1681529111000),
FileInfo(path='dbfs:/mnt/bronze/SalesLT/Customer/', name='Customer/', size=0, modificationTime=1681529113000),
FileInfo(path='dbfs:/mnt/bronze/SalesLT/CustomerAddress/', name='CustomerAddress/', size=0, modificationTime=1681529117000),
FileInfo(path='dbfs:/mnt/bronze/SalesLT/Product/', name='Product/', size=0, modificationTime=1681529112000),
FileInfo(path='dbfs:/mnt/bronze/SalesLT/ProductCategory/', name='ProductCategory/', size=0, modificationTime=1681529116000),
FileInfo(path='dbfs:/mnt/bronze/SalesLT/ProductDescription/', name='ProductDescription/', size=0, modificationTime=1681529117000),
FileInfo(path='dbfs:/mnt/bronze/SalesLT/ProductModel/', name='ProductModel/', size=0, modificationTime=1681529110000),
FileInfo(path='dbfs:/mnt/bronze/SalesLT/ProductModelProductDescription/', name='ProductModelProductDescription/', size=0, modificationTime=1681529102000),
```

```
configs = {
    "fs.azure.account.auth.type": "CustomAccessToken",
    "fs.azure.account.custom.token.provider.class":
spark.conf.get("spark.databricks.passthrough.adls.gen2.tokenProviderClass
sName")
}

dbutils.fs.mount(
    source = "abfss://bronze@mrkdatalakegen2.dfs.core.windows.net/",
    mount_point = "/mnt/bronze",
    extra_configs = configs
)

dbutils.fs.ls("/mnt/bronze/SalesLT/")
```

Silvermount

Cmd 3

```
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@mrkdatalakegen2.dfs.core.windows.net/",
9     mount_point = "/mnt/silver",
10    extra_configs = configs)
```


Goldmount

```
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@mrkdatalakegen2.dfs.core.windows.net/",
9      mount_point = "/mnt/gold",
10     extra_configs = configs)
```

Out[5]: True

Bronze to silver notebook

Convert ModifiedDate column with Date Column

ModifiedDate		ModifiedDate
2002-06-01 00:00:00.000		2002-06-01
2002-06-01 00:00:00.000		2002-06-01
2002-06-01 00:00:00.000		2002-06-01
2002-06-01 00:00:00.000		2002-06-01
2002-06-01 00:00:00.000		2002-06-01
2002-06-01 00:00:00.000		2002-06-01
2002-06-01 00:00:00.000		2002-06-01
2002-06-01 00:00:00.000		2002-06-01
2002-06-01 00:00:00.000		2002-06-01
2002-06-01 00:00:00.000		2002-06-01
2002-06-01 00:00:00.000		2002-06-01
2002-06-01 00:00:00.000		2002-06-01
2002-06-01 00:00:00.000		2002-06-01
2002-06-01 00:00:00.000		2002-06-01

----- to -----

```
dbutils.fs.ls('mnt/bronze/SalesLT/')
```

```
dbutils.fs.ls('mnt/silver/')
```

```
input_path = '/mnt/bronze/SalesLT/Address/Address.parquet'
```

```
df = spark.read.format('parquet').load(input_path)
```

```
display(df)
```

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

```
from pyspark.sql.types import TimestampType
```

```
df = df.withColumn("ModifiedDate",
```

```
date_format(from_utc_timestamp(df["ModifiedDate"].cast(TimestampType()),
"UTC"), "yyyy-MM-dd"))
```

Doing transformation for all tables

```
table_name = []
for i in dbutils.fs.ls('mnt/bronze/SalesLT/'):
    table_name.append(i.name.split('/')[0])

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

# Loop through table names
for i in table_name:
    # Construct path to the Parquet file
    path = '/mnt/bronze/SalesLT/' + i + '/' + i + '.parquet'

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

    # Get list of columns in the DataFrame
    column = df.columns

    # Loop through each column
    for col in column:
        # Check if the column name contains "Date" or "date"
        if "Date" in col or "date" in col:
            # Convert the column to timestamp format (assuming it's not
            already a timestamp)
            df = df.withColumn(col,
date_format(from_utc_timestamp(df[col].cast(TimestampType()), "UTC"),
"yyyy-MM-dd"))

    # Construct path to write the transformed data
    output_path = '/mnt/silver/SalesLT/' + i + '/'

    # Write the transformed DataFrame to Delta format in overwrite mode
    df.write.format('delta').mode("overwrite").save(output_path)

display(df)
```

Mahesh Raut - Project - 31 May 2024

bronze to silverPython

FileEditViewRunHelpLast edit was 7 minutes agoGive feedback

Run alldata_transformationScheduleShare

(60) Spark Jobs

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

Command took 39.77 seconds -- by mrktalkstech@gmail.com at 4/17/2023, 12:23:00 AM on data_transformation

Cmd 13

Python

1display(df)

(1) Spark Jobs

Table

	SalesOrderID	RevisionNumber	OrderDate	DueDate	ShipDate	Status	OnlineOrderFlag	SalesOrderNumber	PurchaseOrderNumber	AccountNumber
1	71774	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71774	PO348186287	10-4020-000609
2	71776	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71776	PO19952192051	10-4020-000106
3	71780	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71780	PO19604173239	10-4020-000340
4	71782	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71782	PO19372114749	10-4020-000582
5	71783	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71783	PO19343113609	10-4020-000024
6	71784	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71784	PO19285135919	10-4020-000448
7	71796	2	2008-06-01	2008-06-13	2008-06-08	5	false	SO71796	PO17052159664	10-4020-000420

32 rows | 0.22 seconds runtime

Refreshed now

Updated column:

ModifiedDate

2008-06-08

2008-06-08

2008-06-08

2008-06-08

2008-06-08

2008-06-08

Refreshed now

Silver container

silverContainer

Search

UploadAdd DirectoryRefreshRenameDeleteChange tierAcquire leaseBreak leaseGive 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 Azure AD User Account)

Location: silver / SalesLT

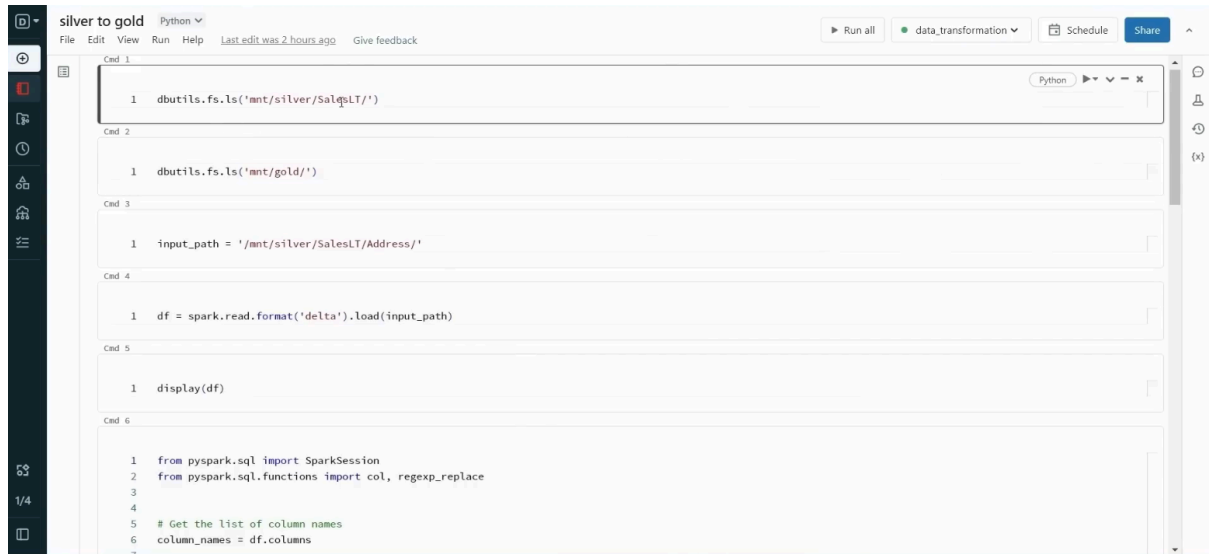
Search blobs by prefix (case-sensitive)

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						...

Silver to gold notebook

Transform the column names for all tables (ex. ProductId to Product_Id)



```
for name in table_name:
    path = '/mnt/silver/SalesLT/' + name
    print(path)
    df = spark.read.format('delta').load(path)

    # 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)]).rstrip("_")

        # Change the column name using withColumnRenamed and regexp_replace
        df = df.withColumnRenamed(old_col_name, new_col_name)

    output_path = '/mnt/gold/SalesLT/' + name + '/'
    df.write.format('delta').mode("overwrite").save(output_path)
```

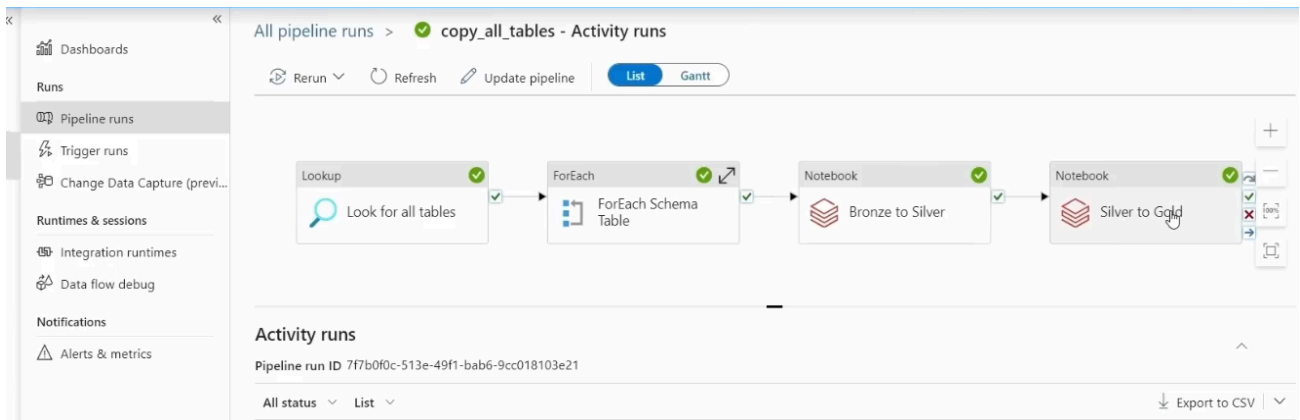
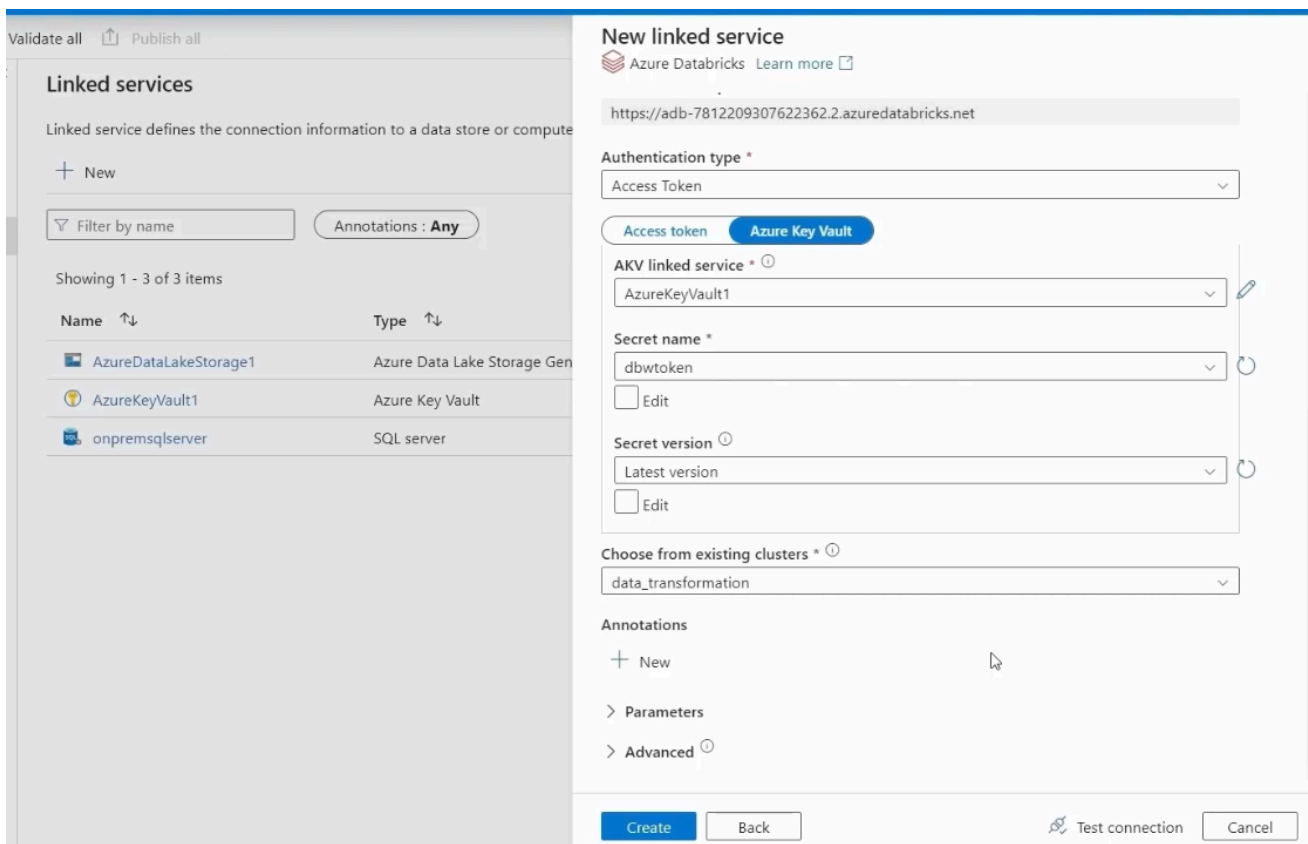
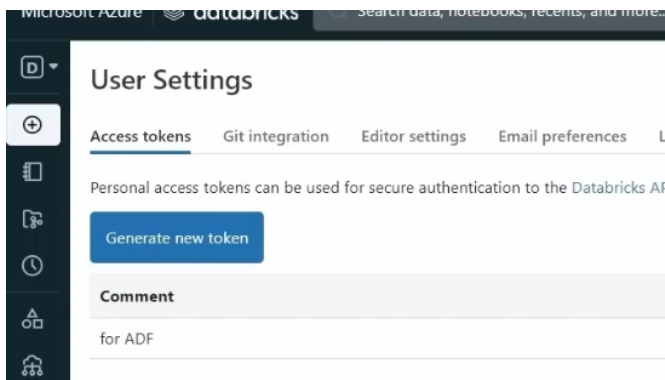
```
table_name = []
for i in dbutils.fs.ls('mnt/silver/SalesLT/'):
    table_name.append(i.name.split('/')[0])
# Get list of column names in column_names
column_names = df.columns

# Loop through each column name
for old_col_name in column_names:
    # Convert column name from CamelCase to snake_case format
    new_col_name = "".join([word.lower() if word.isupper() and not i == 0
else word for i, word in enumerate(old_col_name)]).rstrip("_")

    # Change the column name using withColumnRenamed
    df = df.withColumnRenamed(old_col_name, new_col_name)

# Print transformed DataFrame
print(df)
```

Make connection with cluster in ADF



Activity runs

Pipeline run ID 7f7b0f0c-513e-49f1-bab6-9cc018103e21

All status ▾ List ▾

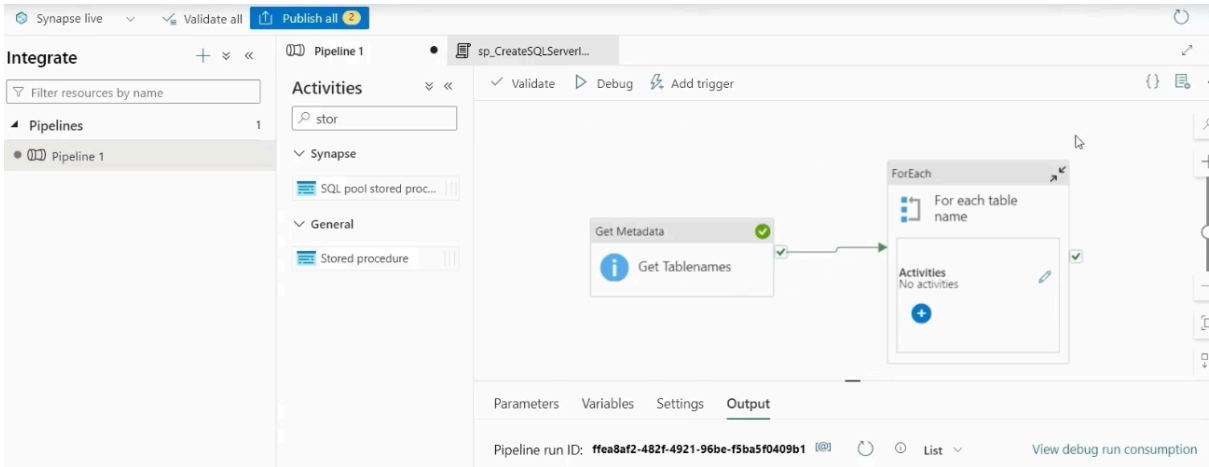
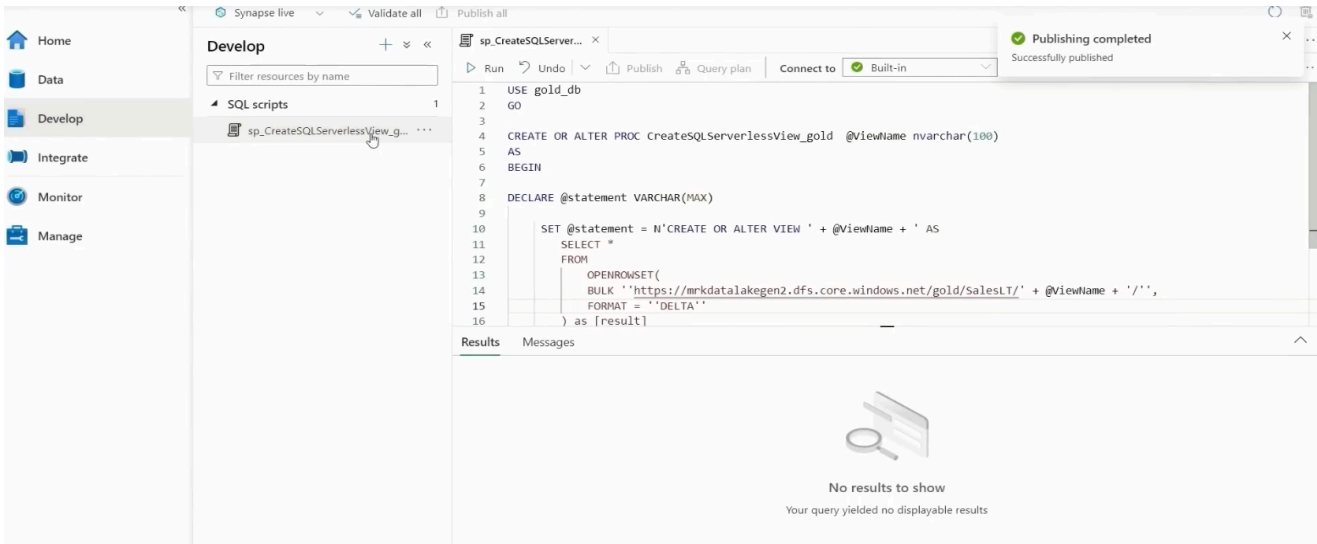
Showing 1 - 14 items

Activity name ↑↓	Status ↑↓	Activity type ↑↓
Silver to Gold	✓ Succeeded	Notebook
Bronze to Silver	✓ Succeeded	Notebook
Copy Each Table	✓ Succeeded	Copy data
Copy Each Table	✓ Succeeded	Copy data
Copy Each Table	✓ Succeeded	Copy data
Copy Each Table	✓ Succeeded	Copy data

Part 4 - Data Loading - Gold container to Azure Synapse Serverless SQL Pool

```
USE gold_db
GO
CREATE OR ALTER PROC CreateSQLServerlessView_gold @ViewName nvarchar(100)
AS
BEGIN
DECLARE @statement VARCHAR (MAX)
SET @statement = N'CREATE OR ALTER VIEW ' + @ViewName + N' AS
                SELECT *
                FROM                                OPENROWSET (
                                                        BULK
''https://mrkdatalakegen2.dfs.core.windows.net/gold/SalesLT/' + @ViewName + '/',
                                                        FORMAT =
''DELTA''
                                                        ) as [result]'

EXEC (@statement)
END
GO
```



After trigger:

