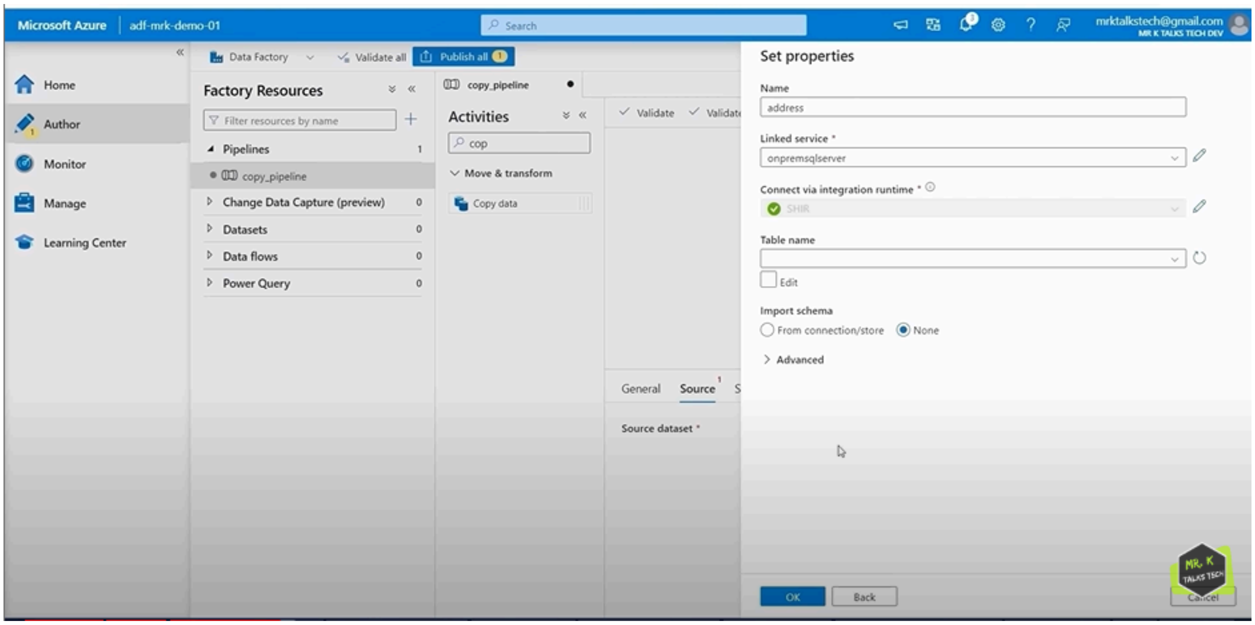
Data Ingestion using ADF - part1

1. Now we have successfully integrated our on perm SQL server to the Azure Data factory using self host integrationruntime



Set properties

Name: address

Linked service *: onpremsqlserver

Connect via integration runtime *: SHIR

Table name:

□ Edit

Import schema: ○ From connection/store ● None

> Advanced

2. Now we can choose the table names which are available in our onPerm

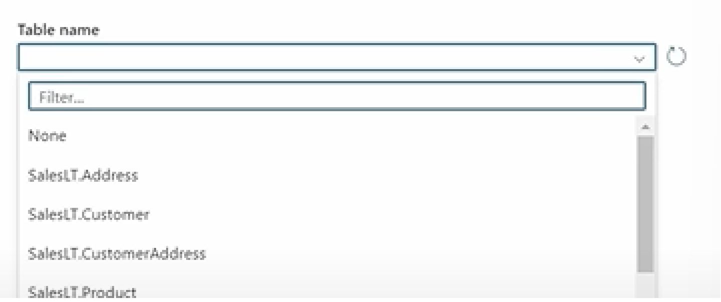


Table name

Filter...

None

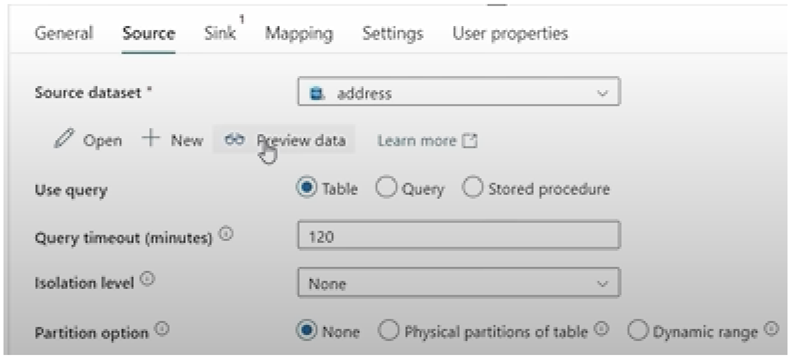
SalesLT.Address

SalesLT.Customer

SalesLT.CustomerAddress

SalesLT.Product

3. Now we'll just select address table for this demo..select it and then click OK
4. And now we can preview our data too



Source dataset *: address

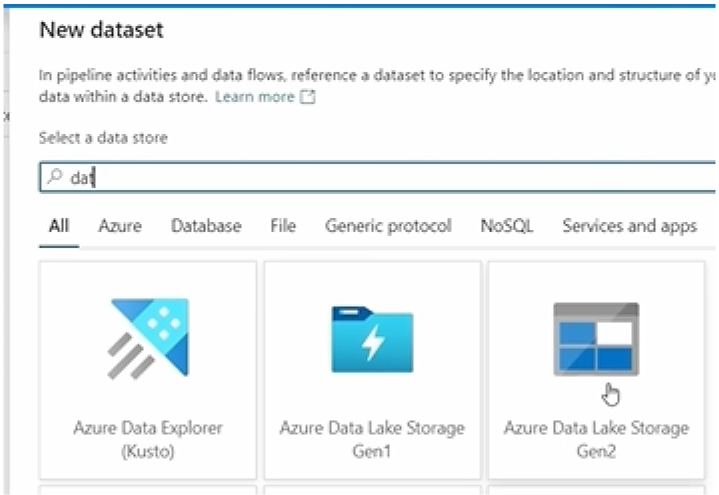
Use query: ● Table ○ Query ○ Stored procedure

Query timeout (minutes): 120

Isolation level: None

Partition option: ● None ○ Physical partitions of table ○ Dynamic range

5. Now lets configure the sink option…here our sink(dumping) would be ADL



New dataset

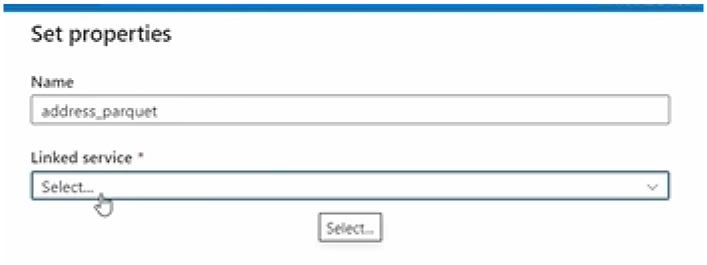
In pipeline activities and data flows, reference a dataset to specify the location and structure of y data within a data store. Learn more

Select a data store

dat

All
Azure
Database
File
Generic protocol
NoSQL
Services and apps
Azure Data Explorer (Kusto)
Azure Data Lake Storage Gen1
Azure Data Lake Storage Gen2

6. Then next we have to give name and linked)service

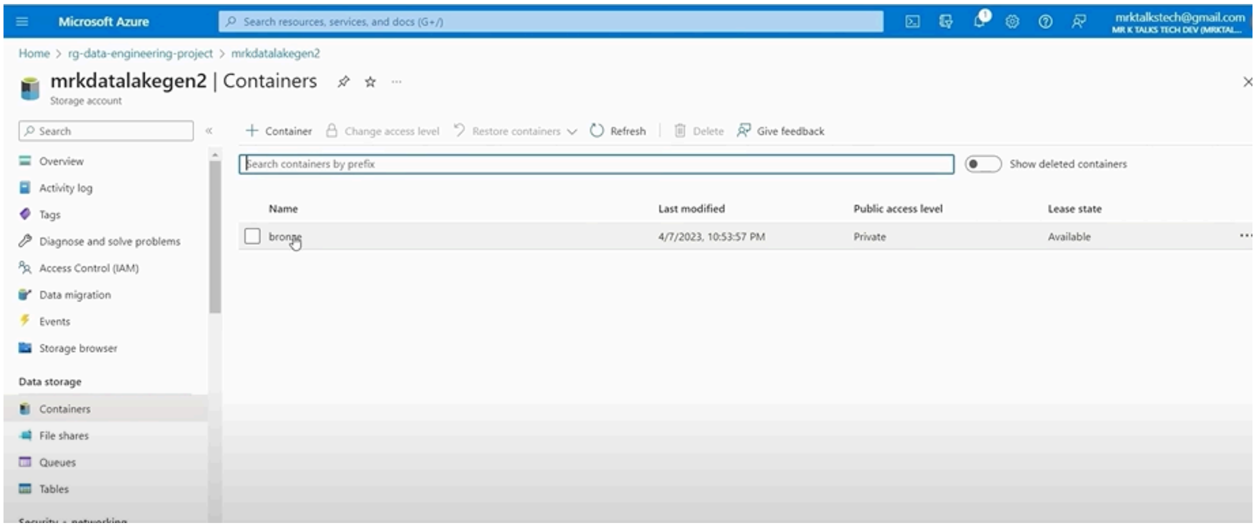


Set properties

Name: address_parquet

Linked service *: Select...

7. In linked_services…now our integration runtime would be AutoResolveIntegrationRuntime ..as we are moving data(within cloud) from ADF to ADL
8. We'll fill all the required things and click on test connection
9. Next we need to specify a location in ADL
10. So here we have created a container in ADL which is bronze layer



Home > rg-data-engineering-project > mrkdatalakegen2

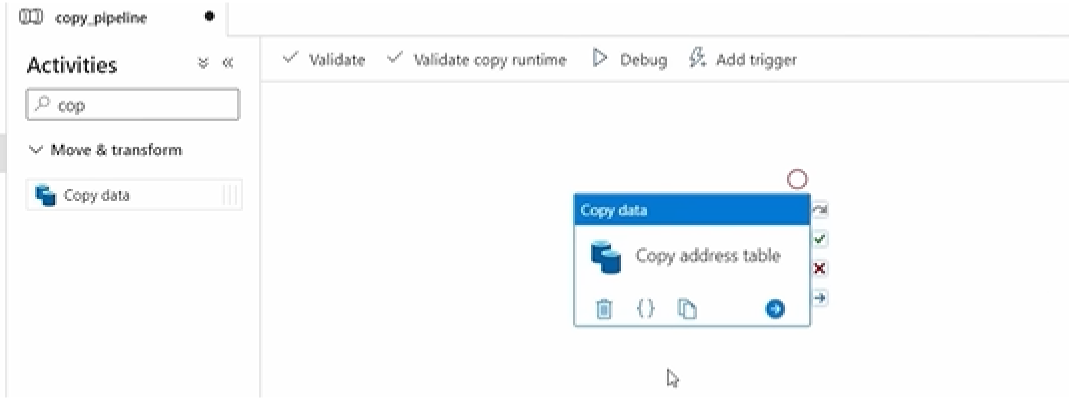
mrkdatalakegen2 | Containers

Storage account

	Name	Last modified	Public access level	Lease state
□	bronze	4/7/2023, 10:53:57 PM	Private	Available

11. So we'll just give this as the file path…and click ok
12. Now after that..we have successfully completed source and sink

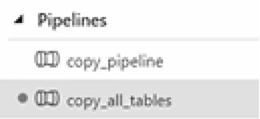
13. Next we'll run this demo pipeline using DEBUG



14. Now what this pipeline does is…copy the onPerm address table to ADL(bronze layer)

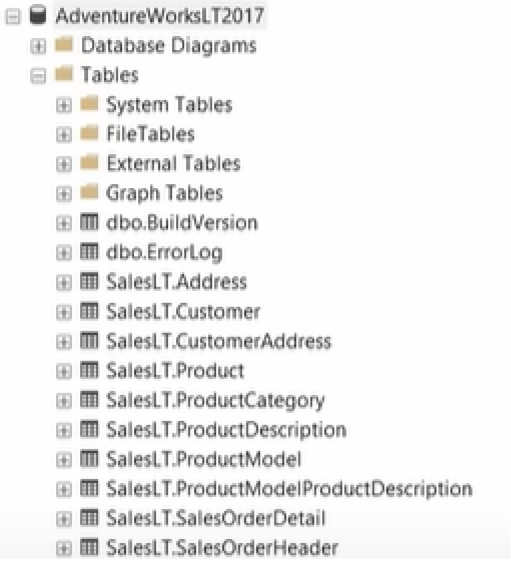
Data Ingestion using ADF - part2

1. Previously we have ingested only one table to show the demo
2. Now we ingest all the tables present in our OnPerm
3. We created a pipeline called copy all tables



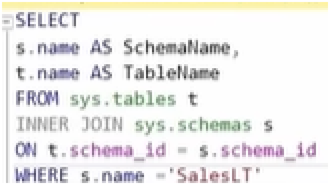
Pipelines
copy_pipeline
copy_all_tables

4. Now we are actually interested in SalesLT schemas tables

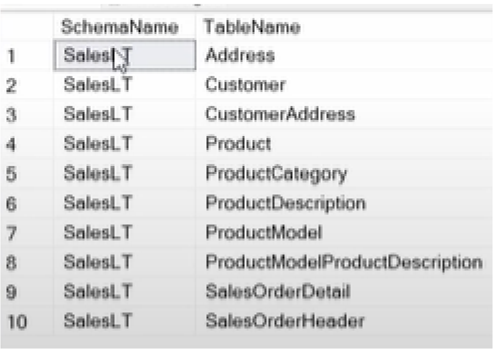


AdventureWorksLT2017
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.BuildVersion
dbo.ErrorLog
SalesLT.Address
SalesLT.Customer
SalesLT.CustomerAddress
SalesLT.Product
SalesLT.ProductCategory
SalesLT.ProductDescription
SalesLT.ProductModel
SalesLT.ProductModelProductDescription
SalesLT.SalesOrderDetail
SalesLT.SalesOrderHeader

5. SO we write a script to where we get schema name and table name



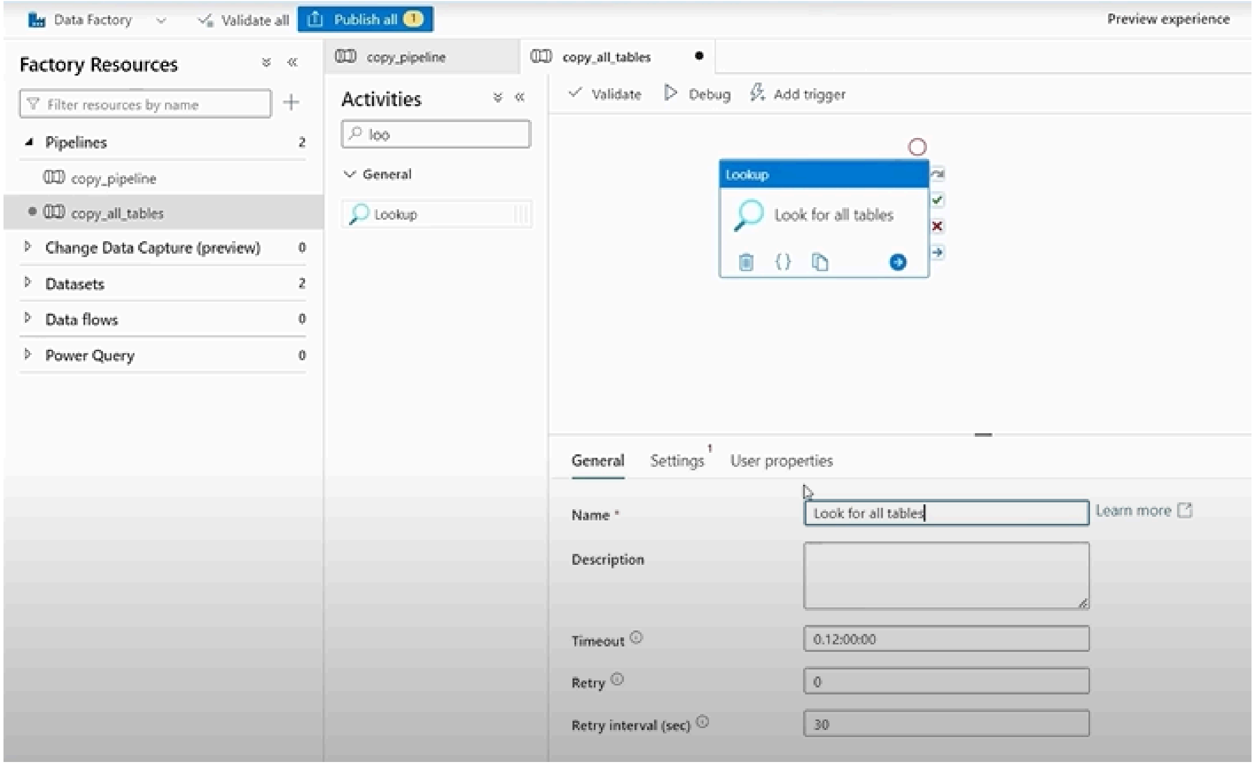
```
SELECT
s.name AS SchemaName,
t.name AS TableName
FROM sys.tables t
INNER JOIN sys.schemas s
ON t.schema_id = s.schema_id
WHERE s.name ='SalesLT'
```



	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

This gives list of all tables belongs to SalesLT schema

6. Now we'll copy this 10 tables to ADF
7. Next in our pipeline we'll add new activity which is LookUp activity



Name * Look for all tables

Description

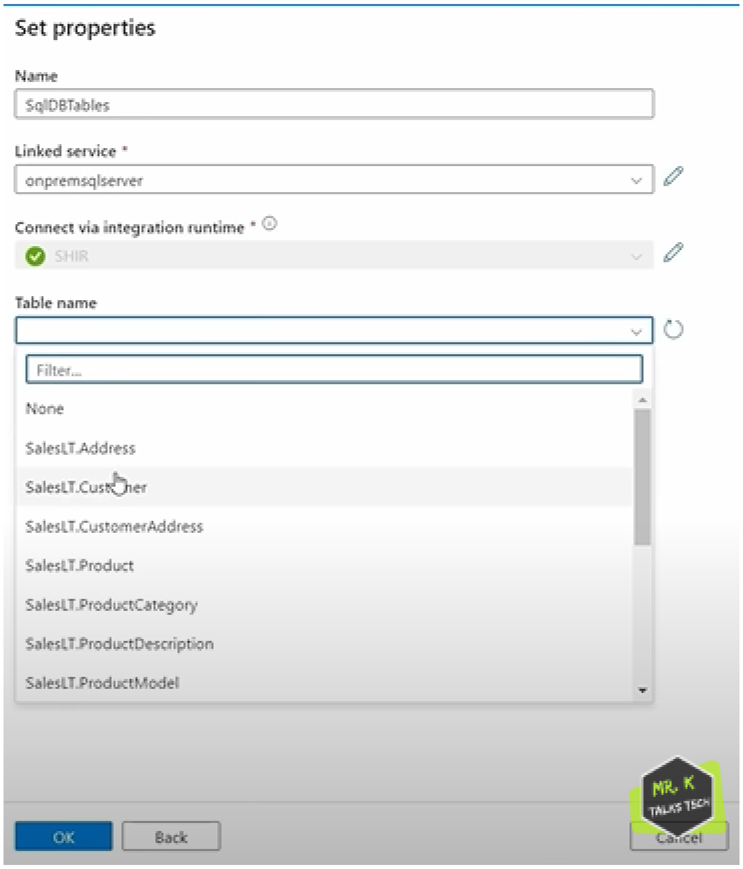
Timeout 0.12:00:00

Retry 0

Retry interval (sec) 30

8. Next in the settings tab..we'll create a new Source dataset called SqlDBTables and choose our linked service which we created earlier and in the table name section..we'll

not choose any table and we click ok



Set properties

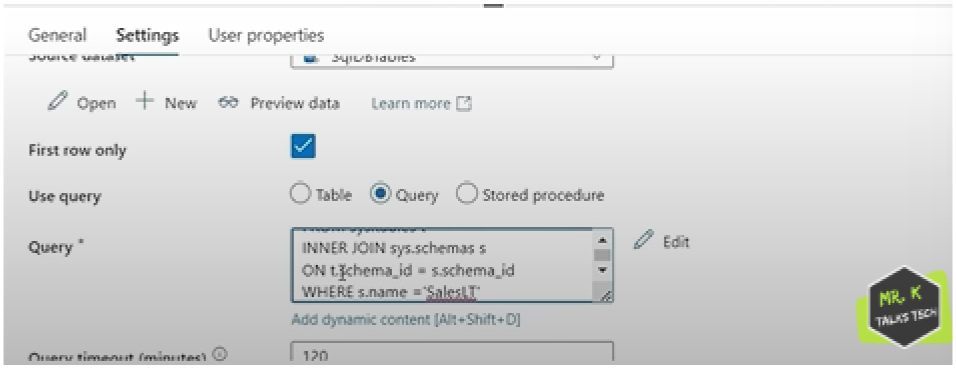
Name
SqlDBTables

Linked service *
onpremsqlserver

Connect via integration runtime *
SHIR

Table name

9. Now to retrieve the SalesLT schema's table in ADF..we use Query and give our query to run



First row only ☑

Use query ○ Table ● Query ○ Stored procedure

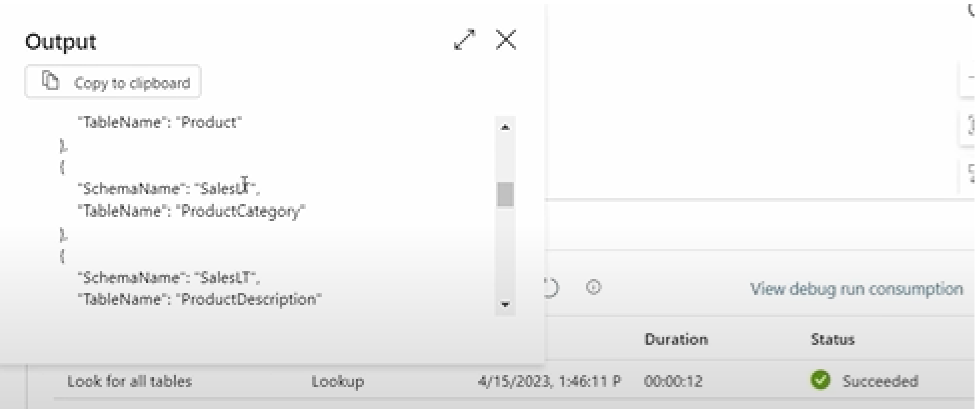
Query *

```
INNER JOIN sys.schemas s
ON t.schema_id = s.schema_id
WHERE s.name = 'SalesLT'
```

Query timeout (minutes) 120

10. Now if we debug our copyalltables pipeline..then it gives us Input and output after it successfully debugged

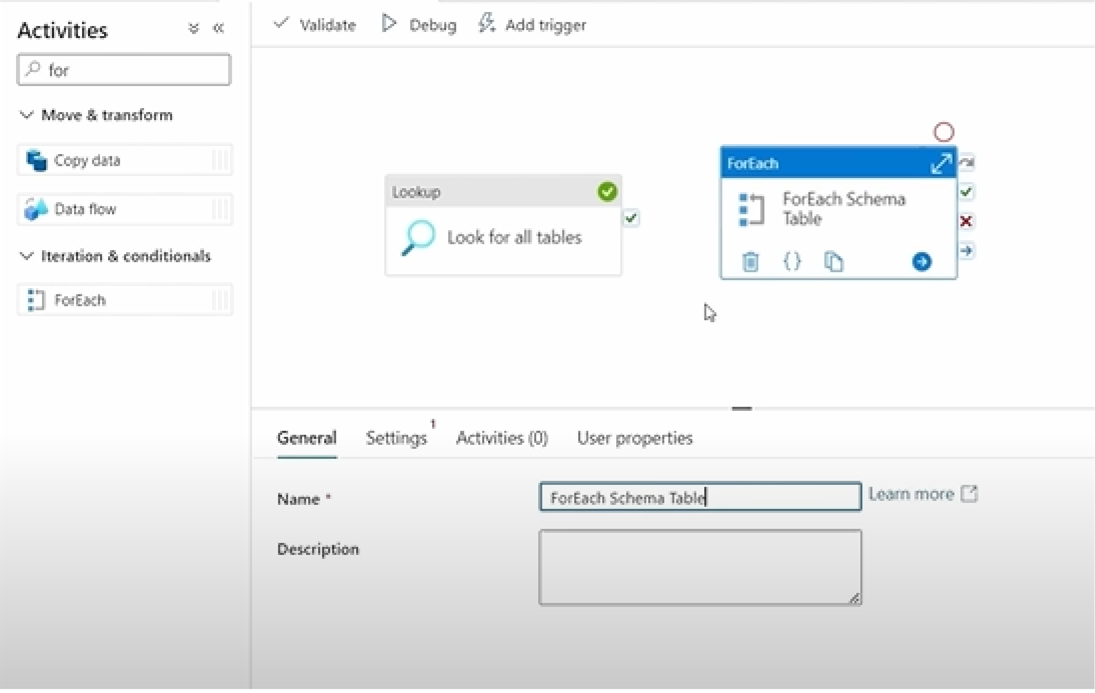
11. Here using this json structured output ..we'll copy the tables from source



```
    "TableName": "Product"
},
{
    "SchemaName": "SalesLT",
    "TableName": "ProductCategory"
},
{
    "SchemaName": "SalesLT",
    "TableName": "ProductDescription"
```

			Duration	Status
Look for all tables	Lookup	4/15/2023, 1:46:11 P	00:00:12	Succeeded

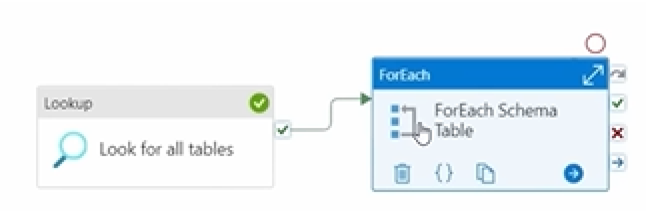
12. Next we'll add new activity called for each



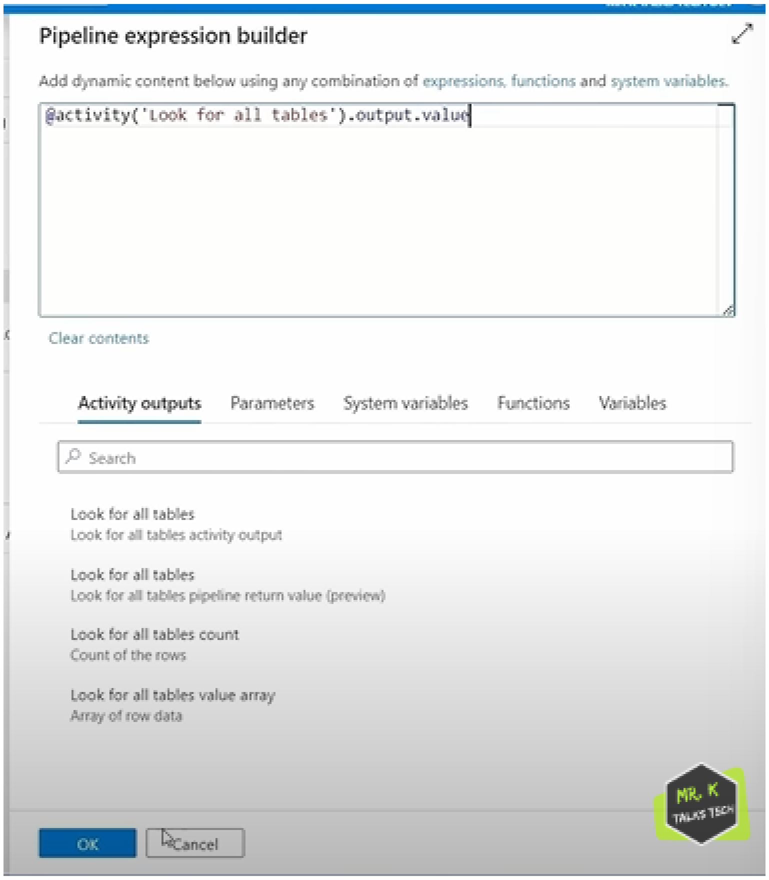
Name * ForEach Schema Table

Description

13. Next we'll connect our lookup activity with for each



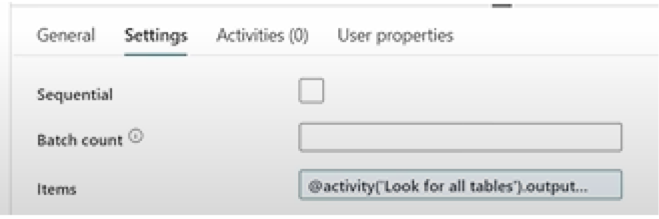
14. Now in the setting tabs of for each activity..we have to give items(dynamic content) and we give output of lookup activity



Pipeline expression builder

Add dynamic content below using any combination of expressions, functions and system variables.

```
@activity('Look for all tables').output.value
```

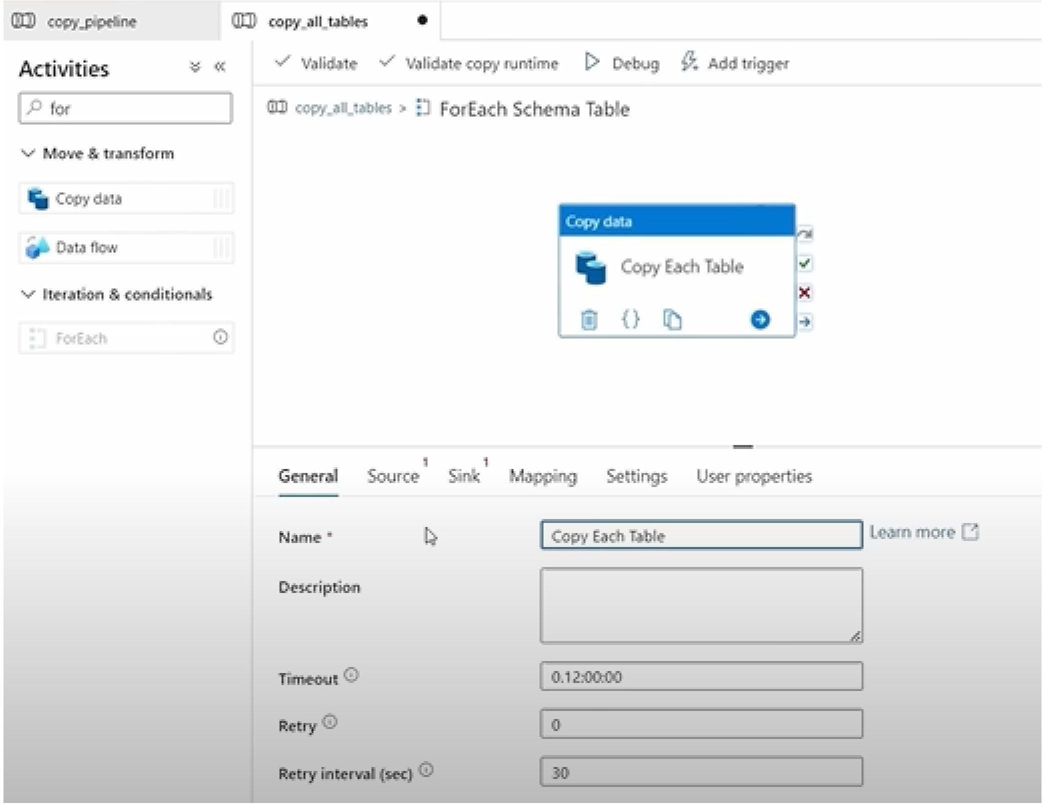


Sequential □

Batch count

Items @activity('Look for all tables').output...

15. Now inside these for each activity ..we'll add copy data activity



Name * Copy Each Table

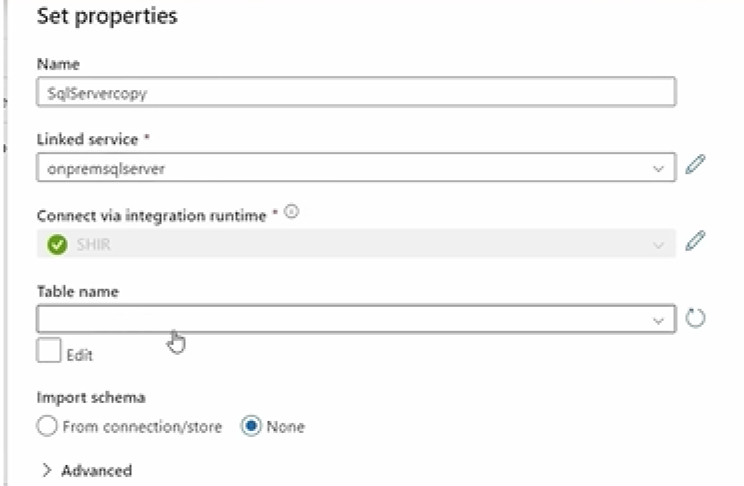
Description

Timeout 0.12:00:00

Retry 0

Retry interval (sec) 30

16. Next we have configure source and sink
17. First we set properties for source



Set properties

Name
SqlServercopy

Linked service *
onpremsqlserver

Connect via integration runtime *
SHIR

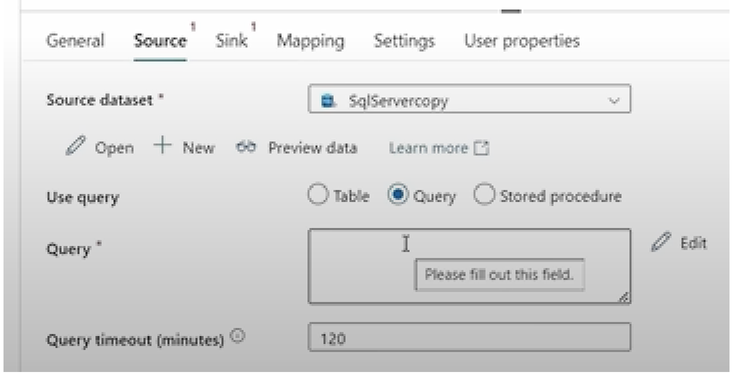
Table name

□ Edit

Import schema
○ From connection/store ● None

> Advanced

18. Now in the query window of source..we'll add dynamic content



Source dataset * SqlServercopy

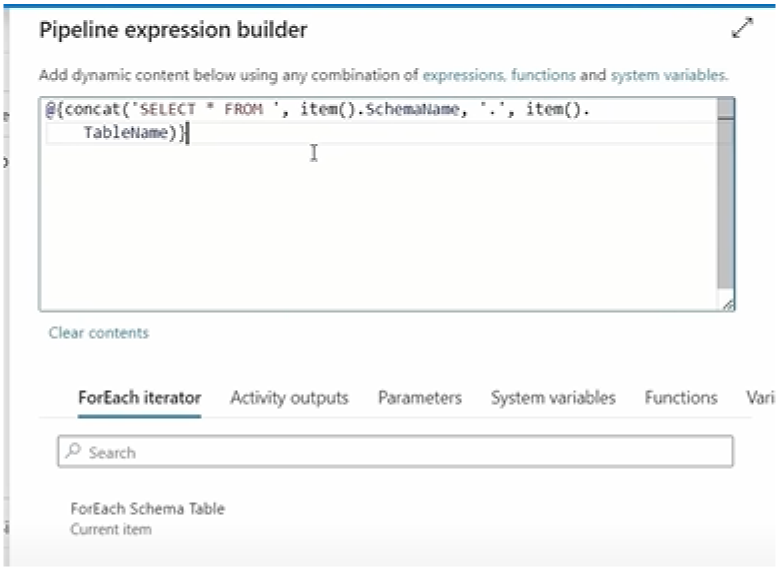
Use query ○ Table ● Query ○ Stored procedure

Query *

Please fill out this field.

Query timeout (minutes) 120

19. Our dynamic content is as follows

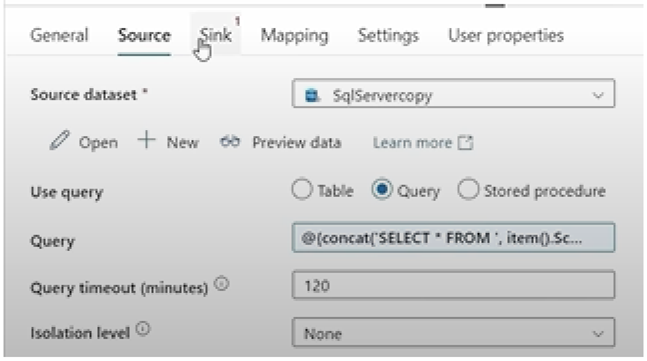

Pipeline expression builder

Add dynamic content below using any combination of expressions, functions and system variables.

```
@{concat('SELECT * FROM ', item().SchemaName, '.', item().
    TableName)}
```

ForEach Schema Table
Current item

..here we'll be getting the output from lookup activity which is the outer loop's output



Source dataset * SqlServercopy

Use query ○ Table ● Query ○ Stored procedure

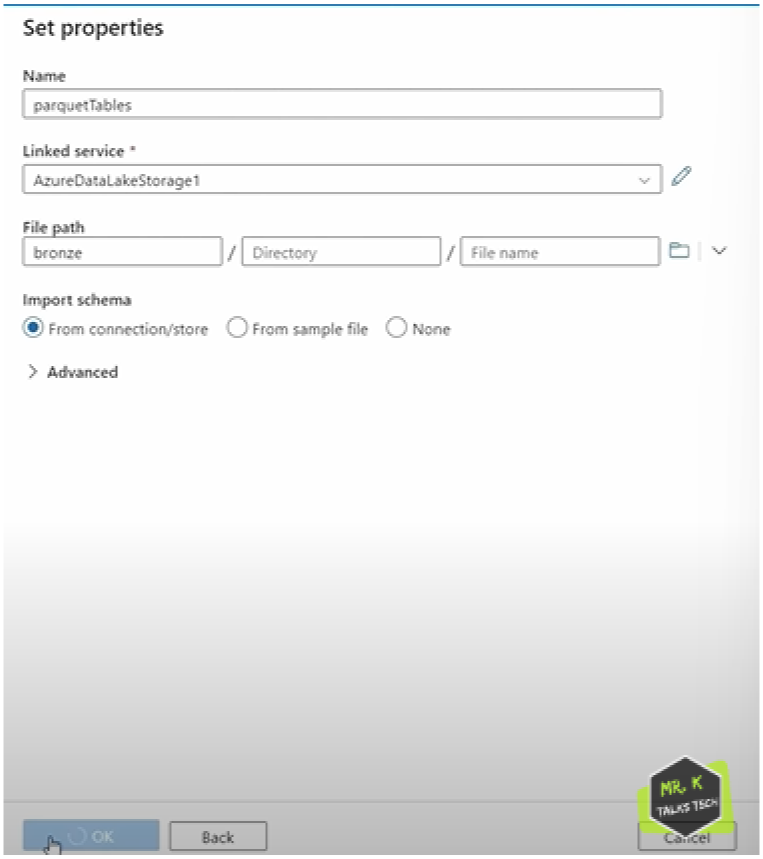
Query @{concat('SELECT * FROM ', item().Sc...

Query timeout (minutes) 120

Isolation level None

now we have configured the source successfully..next we'll configure the sink

20. FOr our sink we’ll choose ADL with parquet format..and we set these properties



Set properties

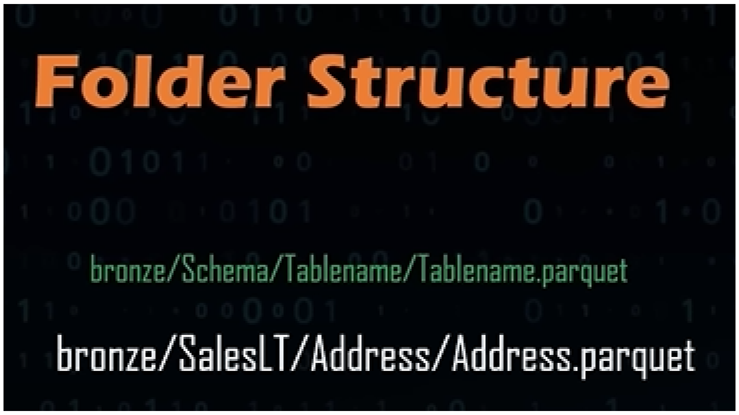
Name: parquetTables

Linked service *: AzureDataLakeStorage1

File path: bronze / Directory / File name

Import schema: ● From connection/store ○ From sample file ○ None

21. So inside the bronze layer..we’ll need this type of folder structure

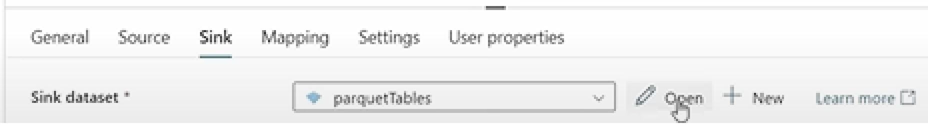


Folder Structure

bronze/Schema/Tablename/Tablename.parquet

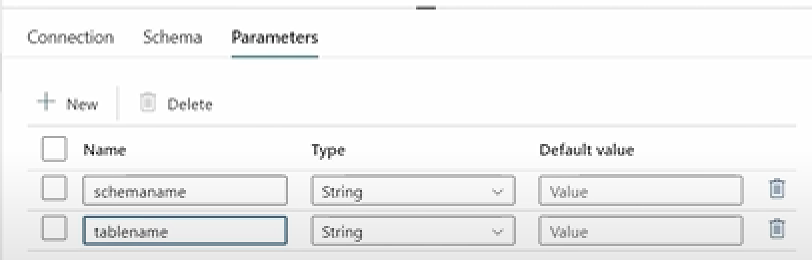
bronze/SalesLT/Address/Address.parquet

22. To add this type of structure..we go to sink dataset and open it



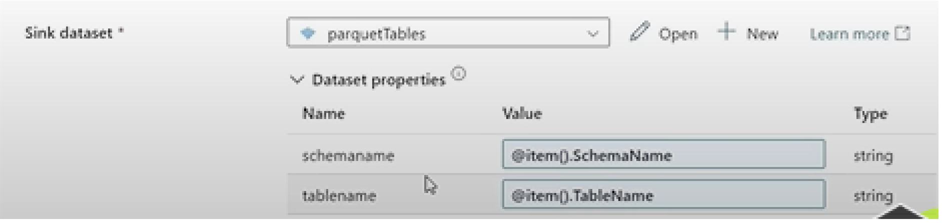
Sink dataset *: parquetTables

23. We’ll add two parameters .. after that we can give dynamic content values



Name	Type	Default value
schemaname	String	Value
tablename	String	Value

24. We give this values..which are coming from lookup activity

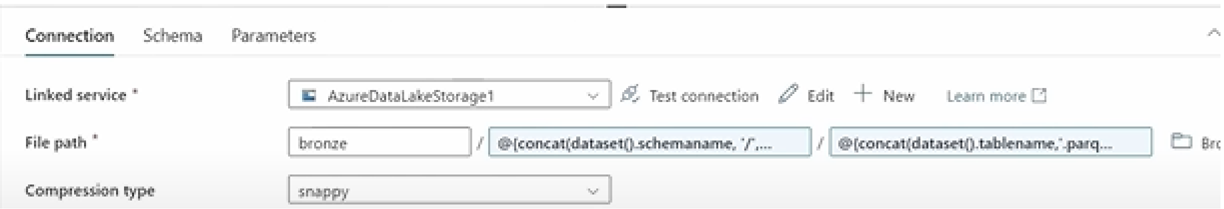


Sink dataset * parquetTables

Dataset properties

Name	Value	Type
schemaname	@item().SchemaName	string
tablename	@item().TableName	string

25. Now in the connection tab..we’ll give the folder structure using the parameters

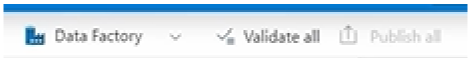


Linked service * AzureDataLakeStorage1

File path * bronze / @{concat(dataset().schemaname, '/',... / @{concat(dataset().tablename,'.parq...

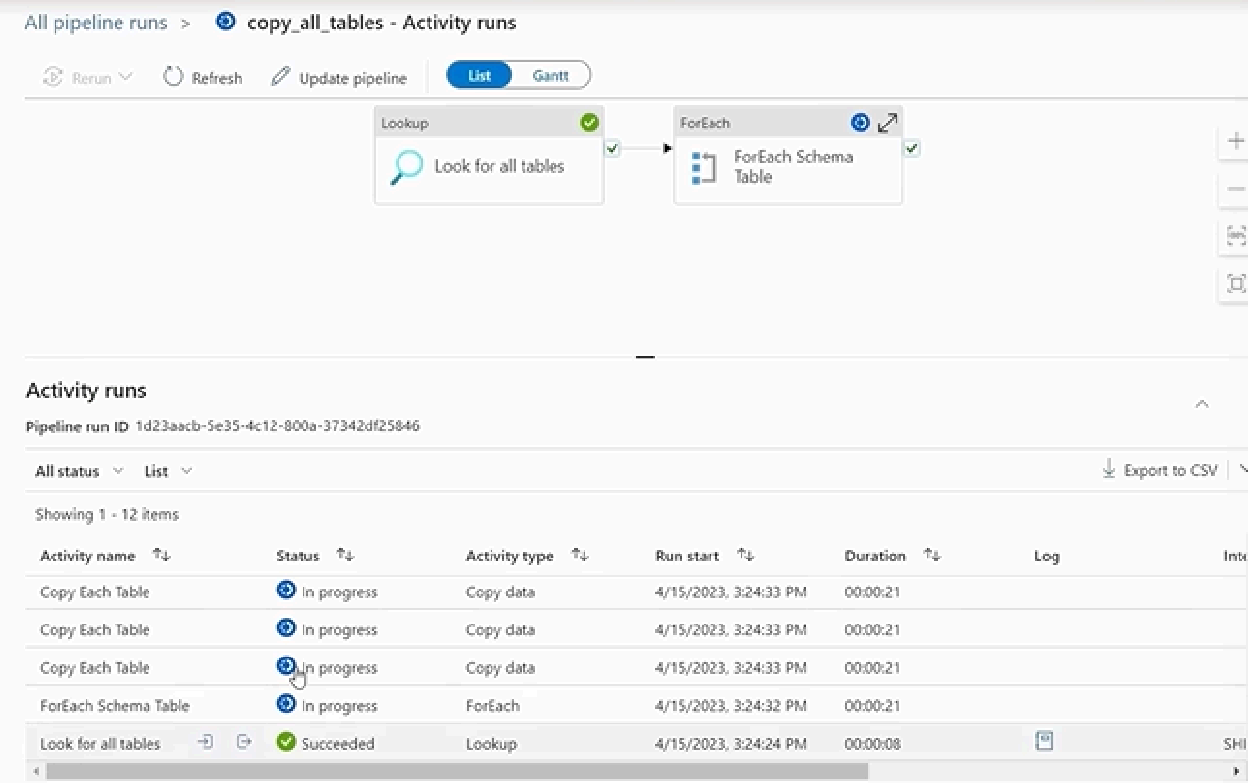
Compression type snappy

26. Now we have completed our pipeline and we click on publish



27. We’ll run our pipeline by using add trigger(trigger now)

28. Our pipeline is in progress



Activity runs

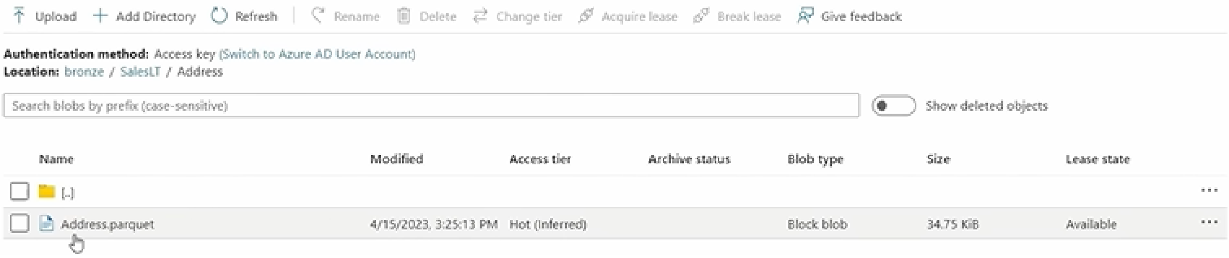
Pipeline run ID 1d23aacb-5e35-4c12-800a-37342df25846

All status | List | Export to CSV

Showing 1 - 12 items

Activity name	Status	Activity type	Run start	Duration	Log	Inte
Copy Each Table	In progress	Copy data	4/15/2023, 3:24:33 PM	00:00:21		
Copy Each Table	In progress	Copy data	4/15/2023, 3:24:33 PM	00:00:21		
Copy Each Table	In progress	Copy data	4/15/2023, 3:24:33 PM	00:00:21		
ForEach Schema Table	In progress	ForEach	4/15/2023, 3:24:32 PM	00:00:21		
Look for all tables	Succeeded	Lookup	4/15/2023, 3:24:24 PM	00:00:08		SHI

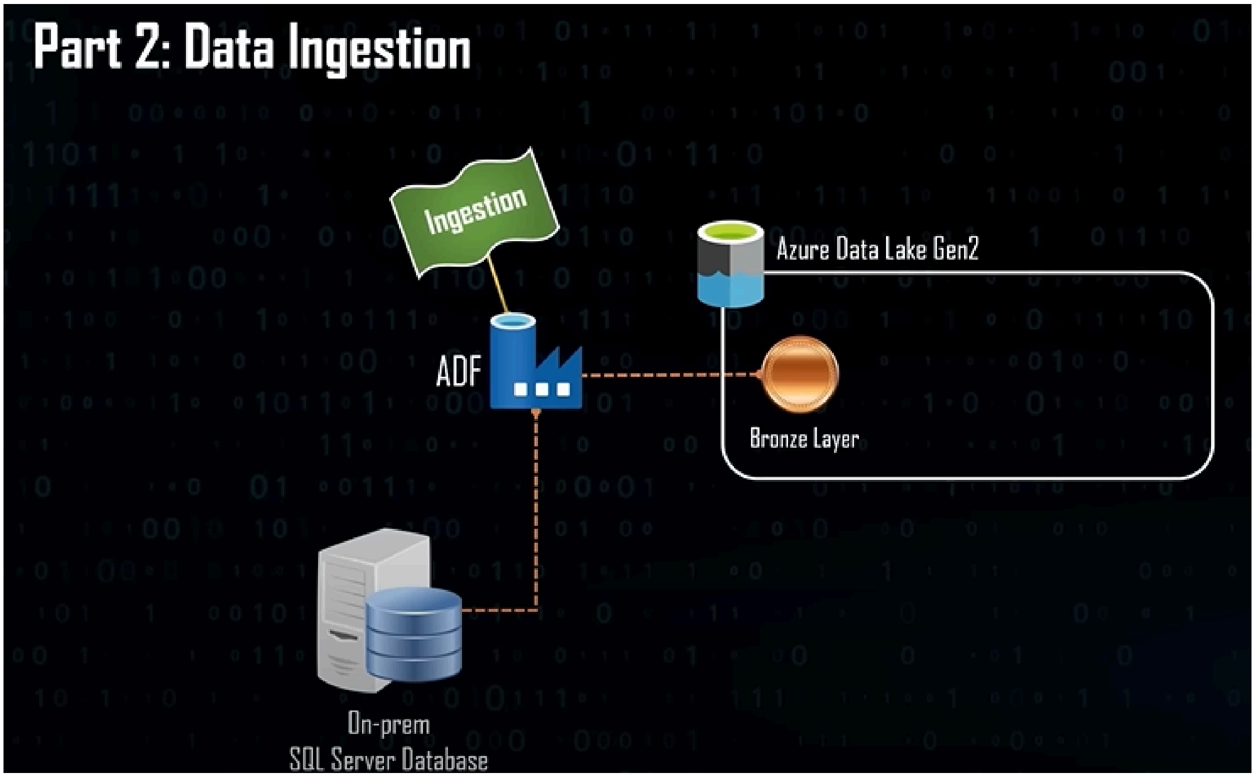
29. Now if we refresh our datalake..then we can see the data in the folders as we structured



Authentication method: Access key (Switch to Azure AD User Account)
Location: bronze / SalesLT / Address

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
[..]						
Address.parquet	4/15/2023, 3:25:13 PM	Hot (Inferred)		Block blob	34.75 KiB	Available

30. Now we have officially completed the data ingestion process



Part 2: Data Ingestion

Ingestion

Azure Data Lake Gen2

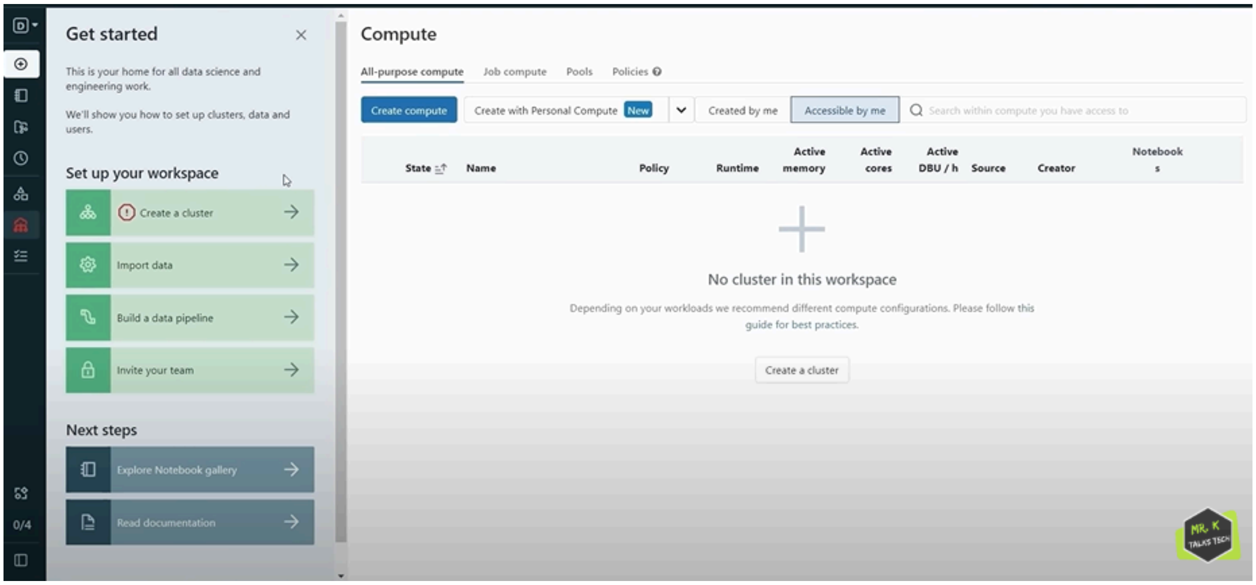
ADF

Bronze Layer

On-prem
SQL Server Database

Data Transformation using Databricks Part1

1. Now we'll use our databricks resource to make transformations on our data
2. So we need to create a compute cluster ..to work on notebooks
3. Click on create compute



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 →

Compute

All-purpose compute | Job compute | Pools | Policies

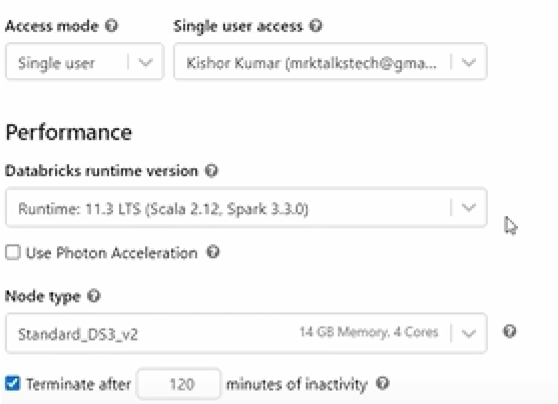
Create compute | Create with Personal Compute New | Created by me | Accessible by me | Search within compute you have access to

State	Name	Policy	Runtime	Active memory	Active cores	Active DBU / h	Source	Creator	Notebooks

No cluster in this workspace

Depending on your workloads we recommend different compute configurations. Please follow this guide for best practices.

Create a cluster



Access mode: Single user

Single user access: Kishor Kumar (mrktalkstech@gma...

Performance

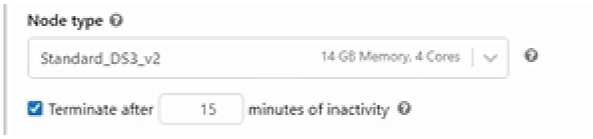
Databricks runtime version: Runtime: 11.3 LTS (Scala 2.12, Spark 3.3.0)

☐ Use Photon Acceleration

Node type: Standard_DS3_v2 — 14 GB Memory, 4 Cores

☑ Terminate after 120 minutes of inactivity

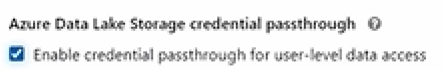
4. We give default specifications for our cluster



Node type: Standard_DS3_v2 — 14 GB Memory, 4 Cores

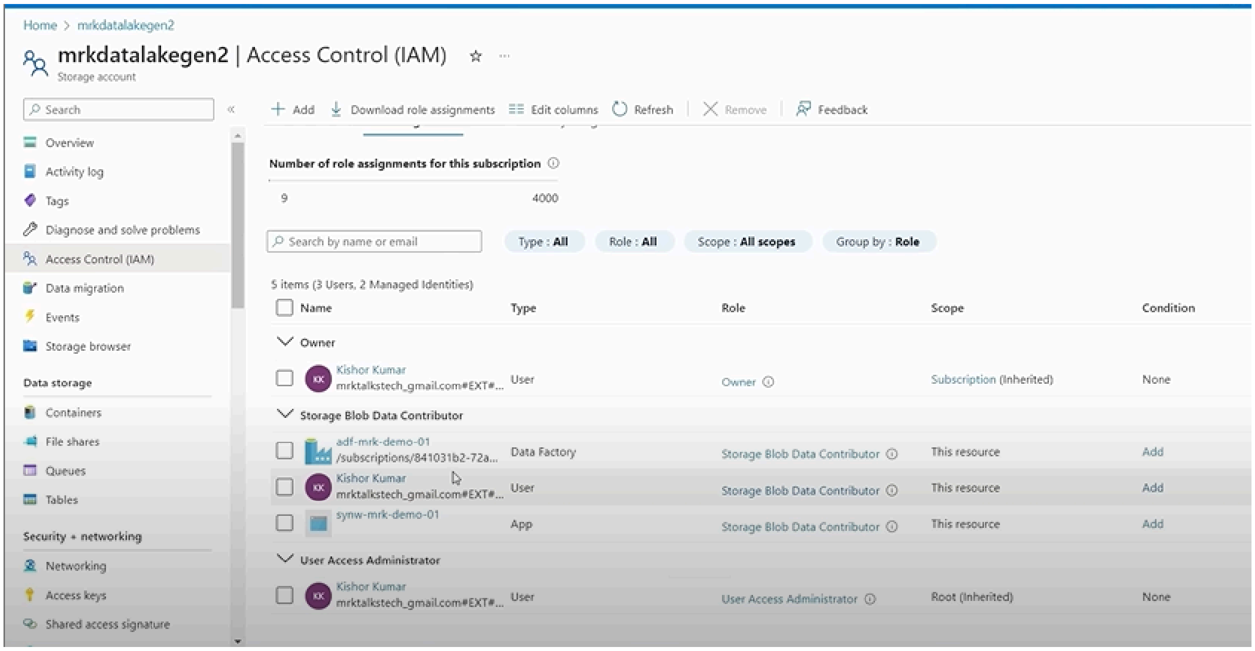
☑ Terminate after 15 minutes of inactivity

5. Now we'll enable our which gives us access to the ADL using our email credentials ..if we using the same email which is present in the shared blob data contributor



Azure Data Lake Storage credential passthrough

☑ Enable credential passthrough for user-level data access



mrkdatalakegen2 | Access Control (IAM)

Storage account

Number of role assignments for this subscription: 9 / 4000

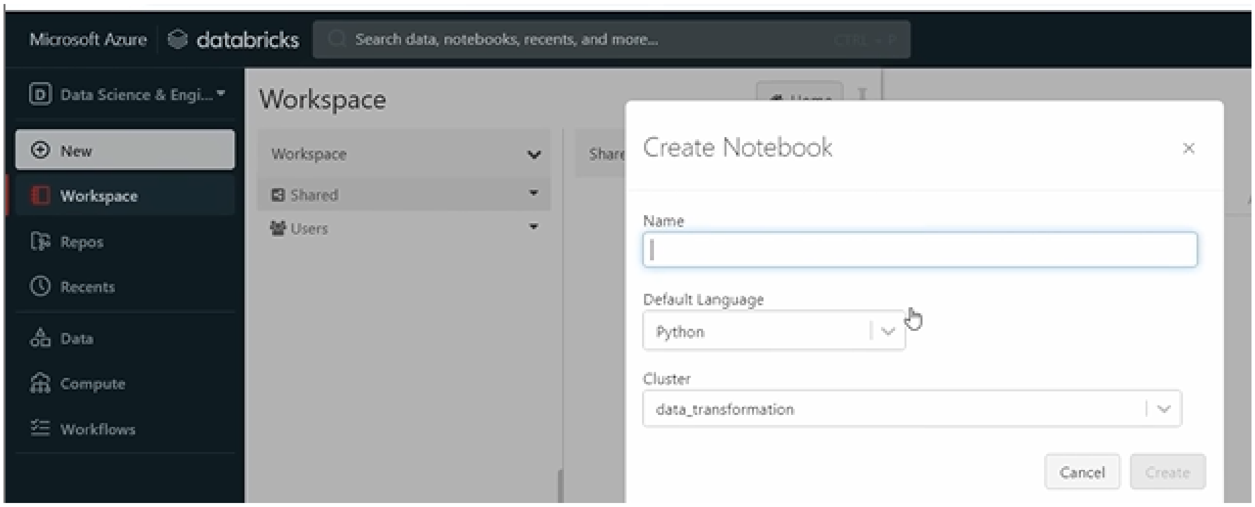
Type : All | Role : All | Scope : All scopes | Group by : Role

5 items (3 Users, 2 Managed Identities)

Name	Type	Role	Scope	Condition
Owner				
Kishor Kumar mrktalkstech_gmail.com#EXT#...	User	Owner	Subscription (Inherited)	None
Storage Blob Data Contributor				
adf-mrk-demo-01 /subscriptions/841031b2-72a...	Data Factory	Storage Blob Data Contributor	This resource	Add
Kishor Kumar mrktalkstech_gmail.com#EXT#...	User	Storage Blob Data Contributor	This resource	Add
synw-mrk-demo-01	App	Storage Blob Data Contributor	This resource	Add
User Access Administrator				
Kishor Kumar mrktalkstech_gmail.com#EXT#...	User	User Access Administrator	Root (Inherited)	None

6. Next we'll click create cluster

7. Now lets create notebook



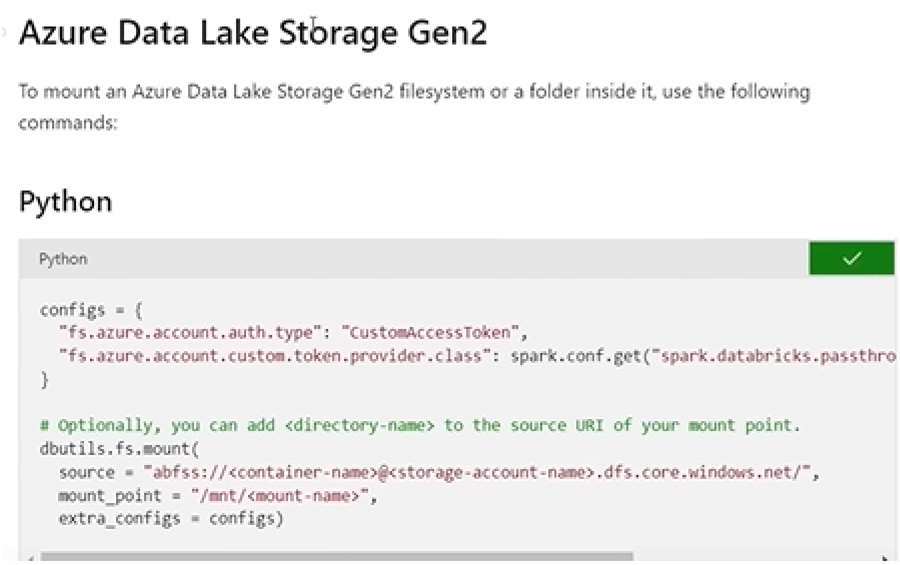
Create Notebook

Name

Default Language: Python

Cluster: data_transformation

8. Now we'll create a notebook for mounting our ADL to the cluster



Azure Data Lake Storage Gen2

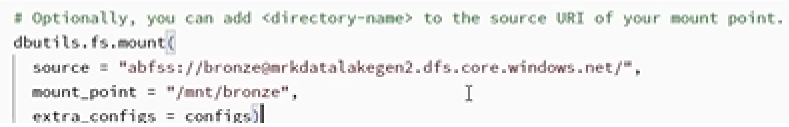
To mount an Azure Data Lake Storage Gen2 filesystem or a folder inside it, use the following commands:

Python

```python
configs = {
  "fs.azure.account.auth.type": "CustomAccessToken",
  "fs.azure.account.custom.token.provider.class": spark.conf.get("spark.databricks.passthro
}

# Optionally, you can add <directory-name> to the source URI of your mount point.
dbutils.fs.mount(
  source = "abfss://<container-name>@<storage-account-name>.dfs.core.windows.net/",
  mount_point = "/mnt/<mount-name>",
  extra_configs = configs)
```

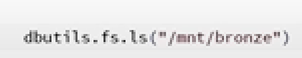
9. Now we'll change the source and the mount point



```python
# Optionally, you can add <directory-name> to the source URI of your mount point.
dbutils.fs.mount(
  source = "abfss://bronze@mrkdatalakegen2.dfs.core.windows.net/",
  mount_point = "/mnt/bronze",
  extra_configs = configs)
```

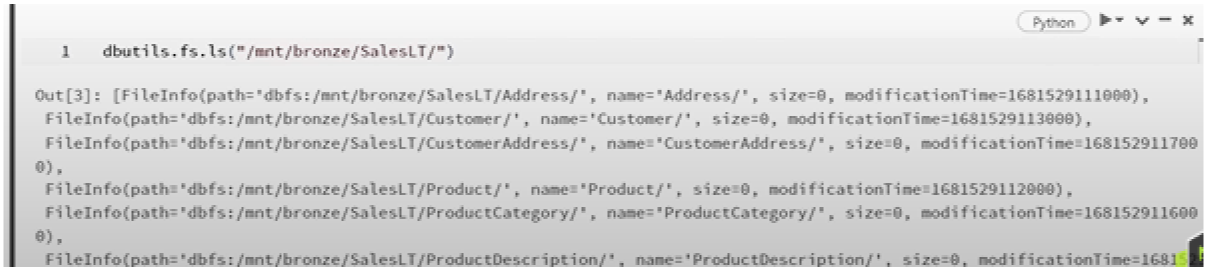
next we'll execute this code

10. So what this means is ..we can access all the data in our bronze container ..using the mount point



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

11. To see the tables we use



```
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=168152911700
0),
 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=168152911600
0),
 FileInfo(path='dbfs:/mnt/bronze/SalesLT/ProductDescription/', name='ProductDescription/', size=0, modificationTime=1681
```

12. Now we’ll create mounts for silver and for gold too
13. *Important point : Since we have used credential passthru..it is not always mandatory to mount the container..instead..we can just use full datalake path..to access the data inside a container
14. Now we’ll get the data from the bronze layer and do some transformations then load to silver