

Project Title:

Data Pipeline for Customer Account Analysis

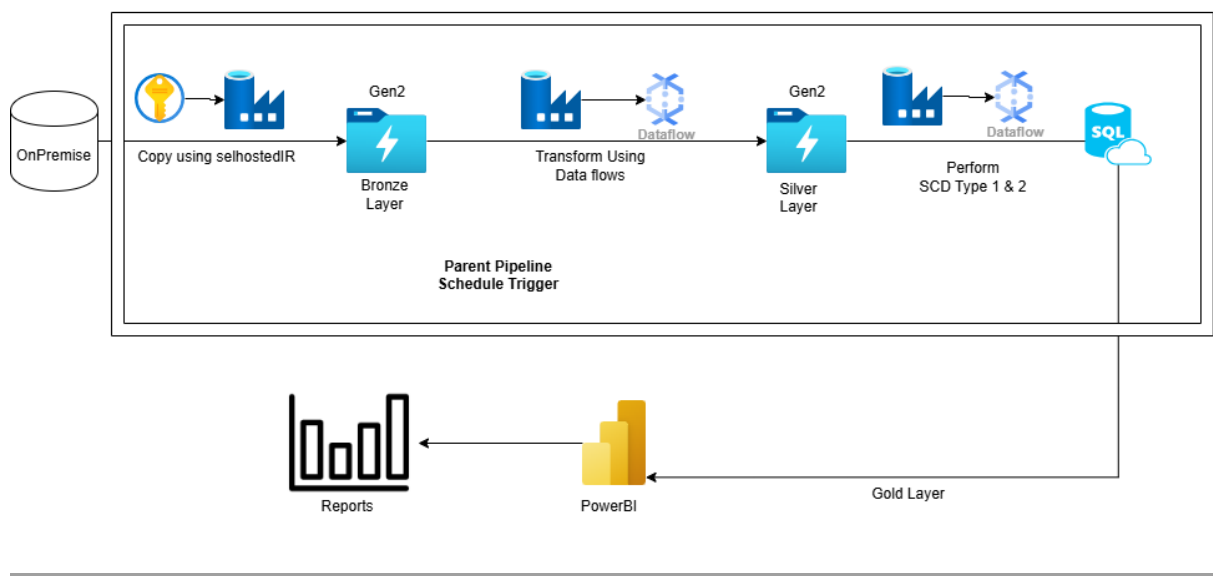
Objective: Design and implement a scalable, efficient, and robust data pipeline to process customer account data. This includes data ingestion, transformation using Azure Data Factory (ADF), and upsertion of cleansed data into a SQL database from Azure Data Lake Storage (ADLS) GOLD Layer. The pipeline will support downstream analytics and reporting.

Table of Contents

1. [Architecture Overview](#)
2. [Step 1: Data Ingestion \(Backend Storage to Raw/Bronze Layer\)](#)
3. [Step 2: Data Cleaning and Transformation \(Bronze Layer\)](#)
4. [Step 3: Load to SQL Database using SCD Techniques](#)
5. [Step 4: Data Visualization Using Power BI](#)
6. [Conclusion](#)

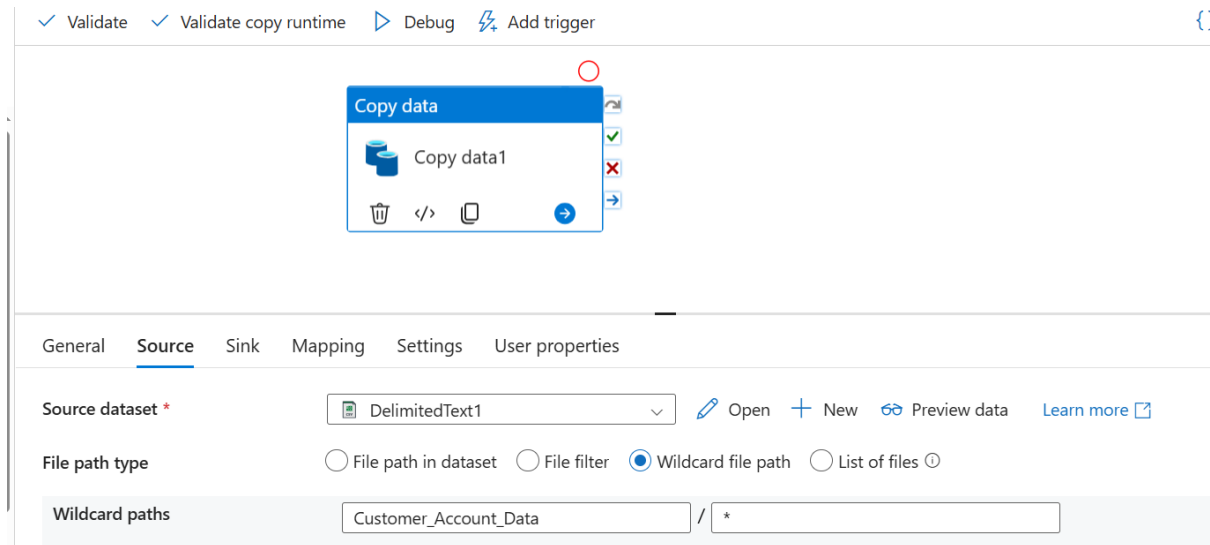
Architecture Overview

Data Pipeline for Customer Account Analysis



Step 1: Data Ingestion (Backend Storage to Raw/Bronze Layer)

Tool: Azure Data Factory (ADF) - Copy Activity

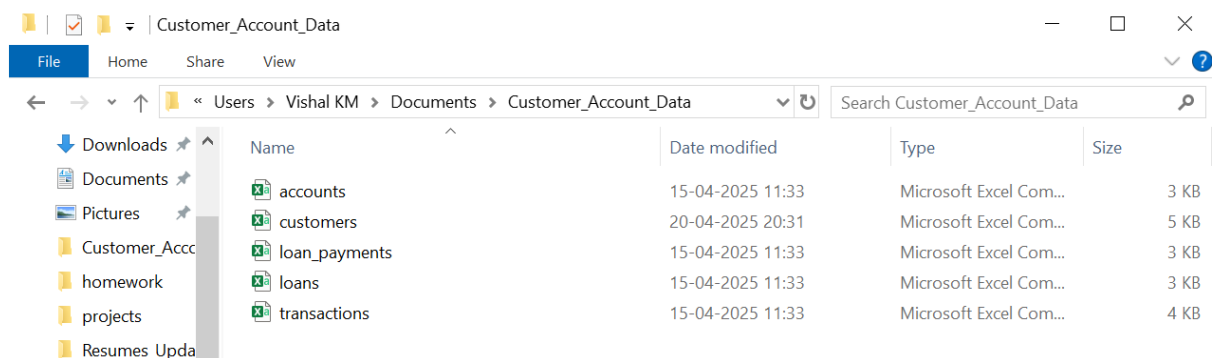


Action:

- Set up a copy activity in ADF to transfer data from the backend team's Azure Storage account to the Raw (Bronze) container in the Data Lake.
-

Source Files:

- accounts.csv
- customers.csv
- loan_payments.csv
- loans.csv
- transactions.csv



Create SelfhostedIR and Filesystem linked service

Edit linked service
File system [Learn more](#)

Connect via integration runtime * ⓘ
SelfhostedIR

Host * ⓘ
C:\Users\Vishal KM\Documents

User name *
Vishal KM

Password Azure Key Vault

AKV linked service * ⓘ
AzureKeyVault1

Secret name * ⓘ
pwd

☒ Edit

- Using KeyVault secrets we provide the password securely
- Using wildcard path , I have copied all the documents into my bronze folder in my container

General **Source** Sink Mapping Settings User properties

Source dataset *
DelimitedText1 [Open](#) [New](#) [Preview data](#) [Learn more](#)

File path type
☐ File path in dataset ☐ File filter ☒ Wildcard file path ☐ List of files ⓘ

Wildcard paths
Customer_Account_Data / *

Start time (UTC) End time (UTC)

Filter by last modified ⓘ

Sink:

- Target: Azure Data Lake Storage (ADLS)
- Container: input/bronze

Connection Schema Parameters

Linked service *
AzureDataLakeStorage1 [Test connection](#) [Edit](#) [New](#) [Learn more](#)

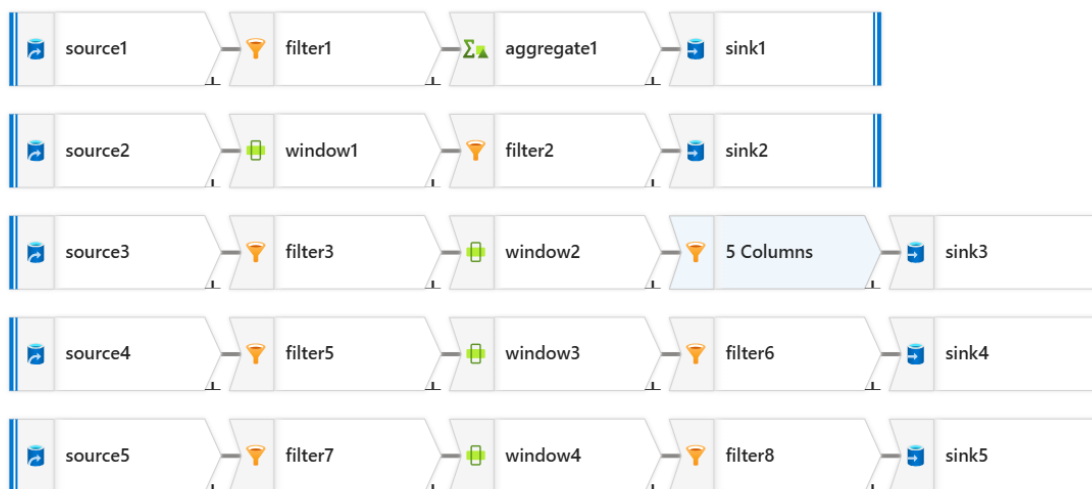
File path
input / Bronze / File name [Browse](#)

Compression type
No compression

Column delimiter ⓘ
Comma (,)

Step 2: Data Cleaning and Transformation (Bronze Layer)

Tool: Azure Data Factory (ADF) - Data Flows



Sub-Steps:

1. **Read Data:** Load the five datasets from the Bronze layer using five sources
2. **Remove Duplicates:** Used Aggregate/Window transformations to detect and eliminate duplicate rows.

Incoming stream * filter3

1. Over 2. Sort 3. Range by 4. Window columns

filter3's column	Name as
12s payment_id	payment_id
12s loan_id	loan_id
payment_date	payment_date
1.2 payment_amount	payment_amount

3. **Data Cleaning:** Remove hanging/null rows using filter transformations.

Filter settings


Optimize

Inspect

Data preview

Description

Filtering rows using expressions on columns 'payment_id, loan_id'



 Reset

Incoming stream *

source3

Filter on *

(!isNull(payment_id)) && (!isNull(loan_id))



Apply filter on major columns so that we can avoid null records

Step 3: Load to SQL Database using SCD Techniques

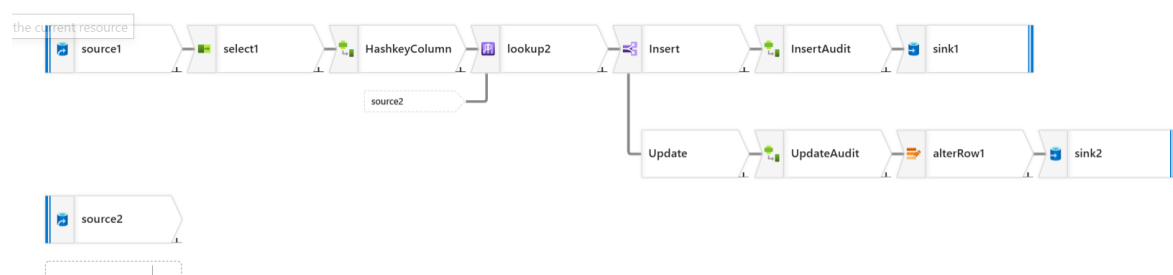
Tool: Azure Data Factory - Data Flows + Pipelines

Tasks:

1. Implement Slowly Changing Dimensions (SCD):
 - **SCD Type 1:** Overwrite existing records
 - I have done the SCD Type-1 for the Accounts, loans, loan-payments, transactions



Accounts Dataflow



Source

From Silver.accounts folder using wildcard path

Source settings	Source options	Projection	Optimize	Inspect	Data preview
File settings					
File mode	<input type="radio"/> File <input checked="" type="radio"/> Wildcard				
File system	input Browse				
Wildcard paths	input / Silver/accounts/*.csv + -				
Allow no files found	<input type="checkbox"/>				

Select

To rename columns

Select settings	Optimize	Inspect	Data preview
<input checked="" type="checkbox"/> Skip duplicate output columns			
Input columns *			
<input type="checkbox"/> Auto mapping Reset + Add mapping - Delete			
<input type="checkbox"/>	source1's column		Name as
<input type="checkbox"/>	123 account_id	→	src_account_id
<input type="checkbox"/>	123 customer_id	→	src_customer_id
<input type="checkbox"/>	abc account_type	→	src_account_type
<input type="checkbox"/>	1.2 balance	→	src_balance

Hashkey derived Column Activity

crc32(src_account_id,src_customer_id,src_account_type,src_balance)

Derived column's settings	Optimize	Inspect	Data preview				
Description	Creating/updating the columns 'src_account_id', 'src_customer_id', 'src_account_type', 'src_balance',						
Incoming stream *	select1						
Columns * ⓘ	<div>+ Add Clone Delete Open expression builder</div> <table><thead><tr><th>Column</th><th>Expression</th></tr></thead><tbody><tr><td>src_Hashkey</td><td>crc32(src_account_id,src_customer_id,src_account... 121</td></tr></tbody></table>			Column	Expression	src_Hashkey	crc32(src_account_id,src_customer_id,src_account... 121
Column	Expression						
src_Hashkey	crc32(src_account_id,src_customer_id,src_account... 121						

Source2

From sql table for lookup new records

Source settings	Source options	Projection	Optimize	Inspect	Data preview
Input	<input type="radio"/> Table <input checked="" type="radio"/> Query <input type="radio"/> Stored procedure				
Query * ⓘ	select account_id,hashkey from accounts				
Incremental column ⓘ	<input type="checkbox"/>				
Isolation level ⓘ	Read uncommitted				

LookUp

Lookup based on ID

Lookup settings	Optimize	Inspect	Data preview
Primary stream *	HashkeyColumn		
Lookup stream *	source2		
Match multiple rows	<input type="checkbox"/> ⓘ		
Match on *	Any row		
Lookup conditions *	Left: HashkeyColumn's column		
	Right: source2's column		
	123 src_account_id == 123 account_id		

Conditional Split activity

Insert condition: !isNull(src_account_id) && isNull(account_id)

Update condition: account_id == src_account_id && src_Hashkey!=hashkey

Conditional split settingsOptimizeInspectData preview

Description

Conditionally distributing the data in src_account_id, account_id, account_id, src_account_id, src_Hashkey, hashkey

Reset

Incoming stream *lookup2

Split on

☒ First matching condition☐ All matching conditions

Split condition

Stream names

Condition

Insert

!isNull(src_account_id) && isNull(account_id)

+

Update

account_id == src_account_id && src_Hashkey!=hashkey

+

InsertAudit derived column activity

Derived column's settingsOptimizeInspectData preview

+ Add

Clone

Delete

Open expression builder

Columns * ⓘ

☐

Column

Expression

☐

src_createdBy

▼

"Dataflow"

abc

+

☐

src_createdDate

▼

currentTimestamp()

+

☐

src_updatedBy

▼

"Dataflow"

abc

+

☐

src_updatedDate

▼

currentTimestamp()

+

UpdateAudit derived column activity

Derived column's settingsOptimizeInspectData preview

+ Add

Clone

Delete

Open expression builder

Columns * ⓘ

☐

Column

Expression

☐

src_createdBy

▼

"Dataflow"

abc

+

☐

src_createdDate

▼

currentTimestamp()

+

☐

src_updatedBy

▼

"Dataflow-Updated"

abc

+

☐

src_updatedDate

▼

currentTimestamp()

+

Sink1 for insert

SinkSettingsErrorsMappingOptimizeInspectData preview

Schema name *

dbo

▼

Refresh

Table name *

accounts

▼

Table action

☒ None☐ Recreate table☐ Truncate table

Update method ⓘ

☒ Allow insert

☐ Allow delete

☐ Allow upsert

☐ Allow update

Sink2 for Update

Sink Settings Errors Mapping Optimize Inspect Data preview

Schema name *

dbo

Refresh

Table name *

accounts

Table action

☒ None ☐ Recreate table ☐ Truncate table

Update method ⓘ

☐ Allow insert

☐ Allow delete

☐ Allow upsert

☒ Allow update

Skip writing key columns ⓘ

☐

Key columns * ⓘ

☒ List of columns ☐ Custom expression ⓘ

123 account_id

+

🗑

- Similarly for the loans, loan_payments and Transaction file dataflows

Output Validation

Initial record

```
99      98,49,Checking,9900.5
100     99,80,Savings,975.75
101     100,50,Checking,10100.0
102
```

Initial load in database

```
select * from accounts where account_id = 100
```

110 %

Results Messages

	account_id	customer_id	account_type	balance	created_date	created_by	updated_date	updated_by	hashkey
1	100	50	Checking	10100.00	2025-04-21 03:25:00.980	Dataflow	2025-04-21 03:25:00.980	Dataflow	3368568676

Day2 record

```
99      98,49,Checking,9900.5
100     99,80,Savings,975.75
101     100,50,Savings,10100.0
102
```

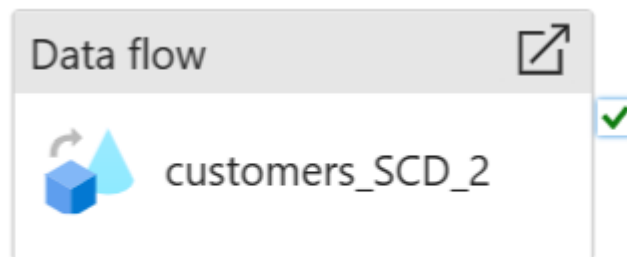
```
select * from accounts where account_id = 100
```

110 %

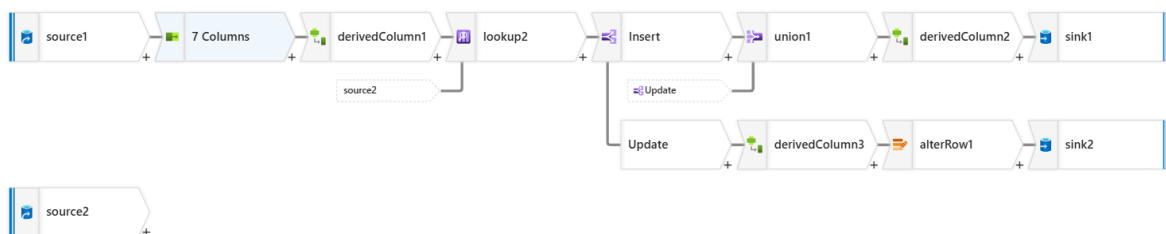
Results Messages

	account_id	customer_id	account_type	balance	created_date	created_by	updated_date	updated_by	hashkey
1	100	50	Savings	10100.00	2025-04-21 03:25:00.980	Dataflow	2025-04-22 06:45:26.117	Dataflow-Updated	3003657656

SCD Type 2: Preserve history of changes using effective start and end dates



Performed SCD 2 for customers data



All the activities are similar but we use Union activity to insert a new record and make it active

Conditional split activity

Conditional split settingsOptimizeInspectData preview

Output stream name *

Insert

Learn more

Description

Conditionally distributing the data in
src_customer_id, customer_id,
customer_id, src_customer_id,

Reset

Incoming stream *

lookup2

Split on

☒ First matching condition ☐ All matching conditions

Split condition

Stream names	Condition
Insert	!isNull(src_customer_id) && isNull(customer_id)
Update	customer_id == src_customer_id && src_Hashkey!=hashkey

Union Activity

Union insert split with update split

Union settingsOptimizeInspectData preview

Output stream name *

union1

Learn more

Description

Combining rows from transformation
'Insert@Insert, Insert@Update'

Reset

Incoming stream *

Insert@Insert

Union by *

☒ Name ☐ Position

Union with *

Insert@Update

+

Insert audit derived column activity

Derived column's settingsOptimizeInspectData preview

+ Add

Clone

Delete

Open expression builder

Columns *

Column	Expression
src_createdBy	"Dataflow"
src_createdDate	currentTimestamp()
src_updatedBy	"Dataflow"
src_updatedDate	currentTimestamp()
src_isActive	1

Update Audit derived column activity

Derived column's settingsOptimizeInspectData preview

src_last_name, src_address, src_city,

Incoming stream *

Insert@Update

+ Add

Clone

Delete

Open expression builder

Columns *

Column	Expression
src_updatedBy	"Dataflow-Updated"
src_updatedDate	currentTimestamp()
src_isActive	0

Alter row activity for Update

Alter row settings Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description [Reset](#)

Incoming stream *

Alter row conditions * ⓘ

+

Sink 1 for Insert

Sink Settings Errors Mapping Optimize Inspect Data preview

Schema name * [Refresh](#)

Table name *

Table action ☒ None ☐ Recreate table ☐ Truncate table

Update method ⓘ ☒ Allow insert
☐ Allow delete
☐ Allow upsert
☐ Allow update

Sink 2 for Update

Sink Settings Errors Mapping Optimize Inspect Data preview

Schema name * [Refresh](#)

Table name *

Table action ☒ None ☐ Recreate table ☐ Truncate table

Update method ⓘ ☐ Allow insert
☐ Allow delete
☐ Allow upsert
☒ Allow update

Pipelines

Local to Bronze Layer

Filter resources by name

Pipelines4

pl_Bronze to Silver

pl_Onprem_to_Bronze

pl_ParentPipeline

pl_Silver to Gold

»

Saved Save as template Validate Debug Add trigger

Copy data

Copy data1

Bronze to Silver Layer

Pipelines4

pl_Bronze to Silver

pl_Onprem_to_Bronze

pl_ParentPipeline

pl_Silver to Gold

Search activities

Move and transform

Synapse

Azure Data Explorer

Azure Function

Data flow

Bronze to Silver

Silver to Gold Layer

Filter resources by name

Pipelines4

pl_Bronze to Silver

pl_Onprem_to_Bronze

pl_ParentPipeline

pl_Silver to Gold

Change Data Capture (review)0

»

Saved Save as template Validate Debug Trigger (1) Data flow debug

Data flow

accounts

loan_payments

Loans

Transactions

customers_SCD_2

Master Pipeline Creation:

Use Execute Pipeline Activity to trigger child pipelines in sequence

»

Saved Save as template Validate Debug Trigger (1) Data flow debug

Execute Pipeline

Bronze Layer
pl_Onprem_to_Bronze

Execute Pipeline

Silver Layer
pl_Bronze to Silver

Execute Pipeline

Gold Layer
pl_Silver to Gold

Scheduled Trigger

Edit trigger

Type *

ScheduleTrigger

Start date * ⓘ

4/21/2025, 12:34:00 PM

Time zone * ⓘ

Eastern Time (US & Canada) (UTC-5)

ⓘ This time zone observes daylight savings. Trigger will auto-adjust for one hour difference.

Recurrence * ⓘ

Every Hour(s)

☐ Specify an end date

Annotations

+ New

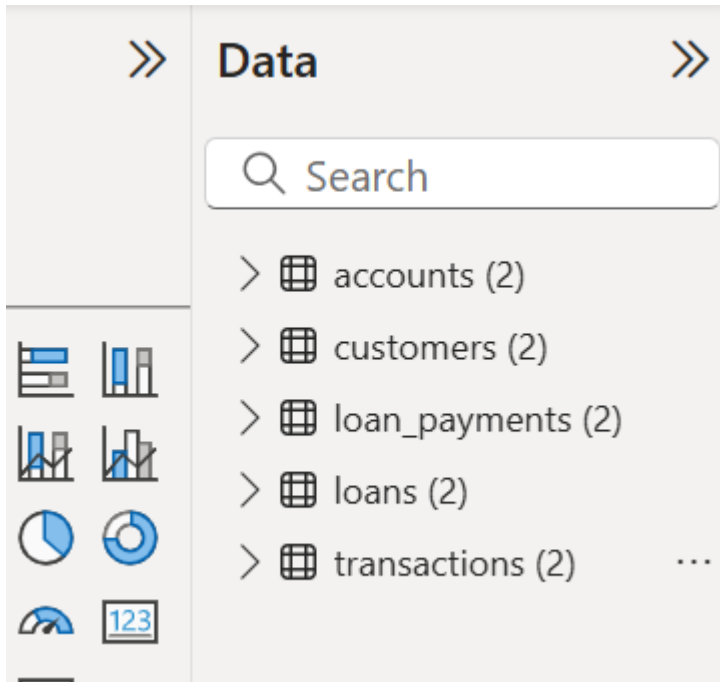
OK

Cancel



Step 4: Data Visualization Using Power BI

Tasks:

1. Connect Power BI to SQL Database tables



2. Publish reports to Microsoft Fabric Workspace

Select a predesigned task flow Add a task									
	Name	Type	Task	Owner	Refreshed	Next refresh	Endorseme	Sensitivity	Included in app
	CustomerData	Report	—	VishalWP	4/22/2025, 1...	—	—	—	<input type="checkbox"/> No
	CustomerData	Semantic ...	—	VishalWP	4/22/2025, ...	N/A	—	—	



Code Repository:

https://github.com/VishalKanaka/DataMigration_SCD_Type1_2.git

Conclusion

This project successfully demonstrates the implementation of a modern data pipeline using Azure Data Factory, Azure Data Lake Storage, and SQL Database integration. The structured approach from data ingestion to transformation and finally to visualization enables efficient and scalable analytics. Automation using ADF pipelines and security through Azure Key Vault ensure a production-ready solution. By delivering clean, well-modeled data to Power BI, this pipeline supports powerful insights into customer account behaviors and loan activities. The project lays a strong foundation for future enhancements and enterprise-grade deployments.

Prepared by: Vishal Kanakamamidi

