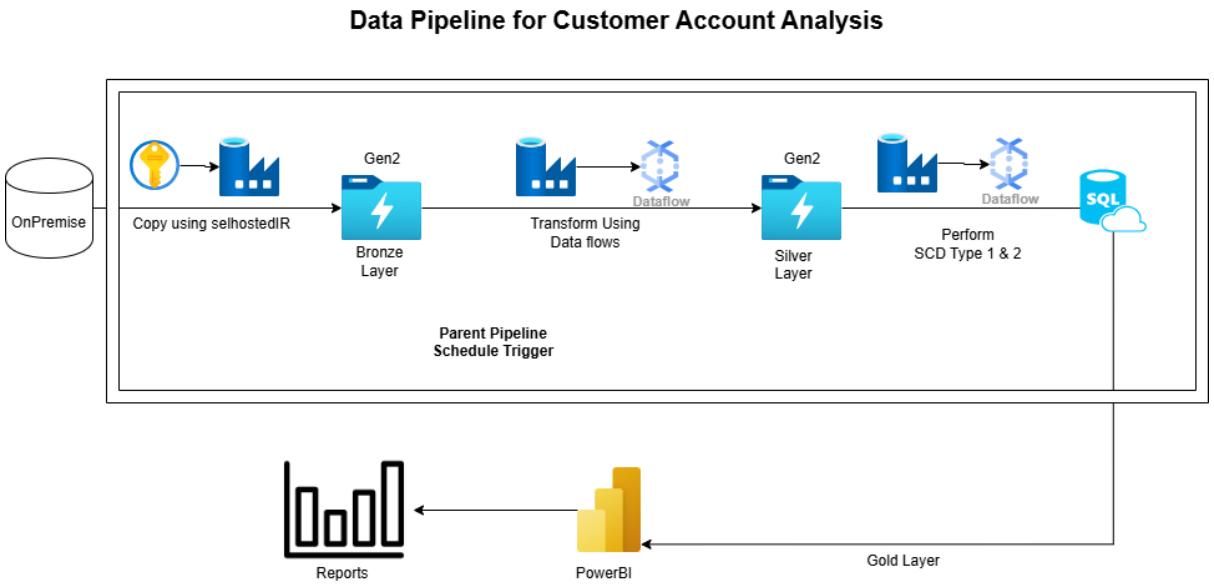
Project Title:

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.

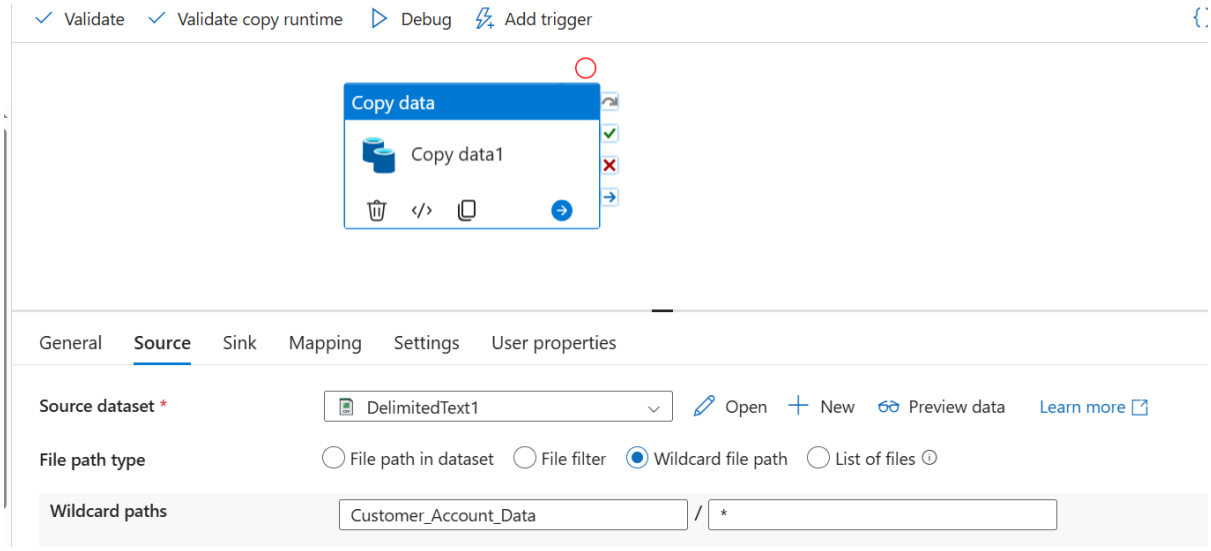
Table of Contents

Architecture Overview



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

Tool: Azure Data Factory (ADF) - Copy Activity

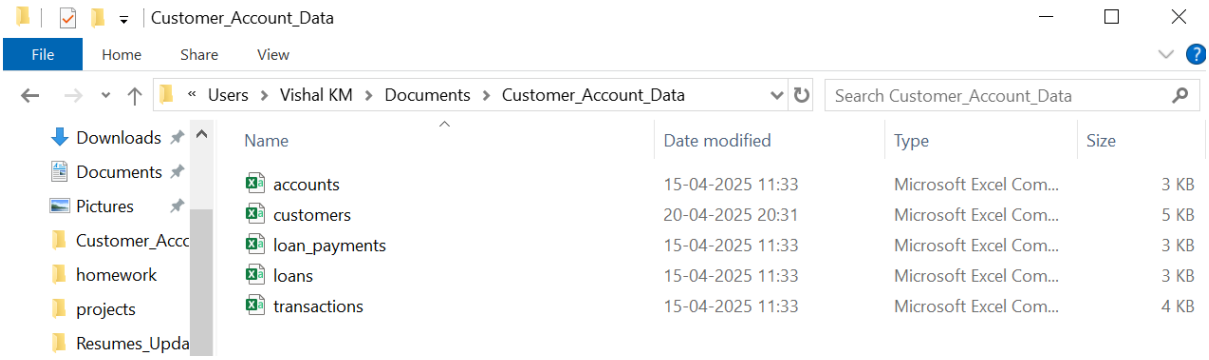


Action:

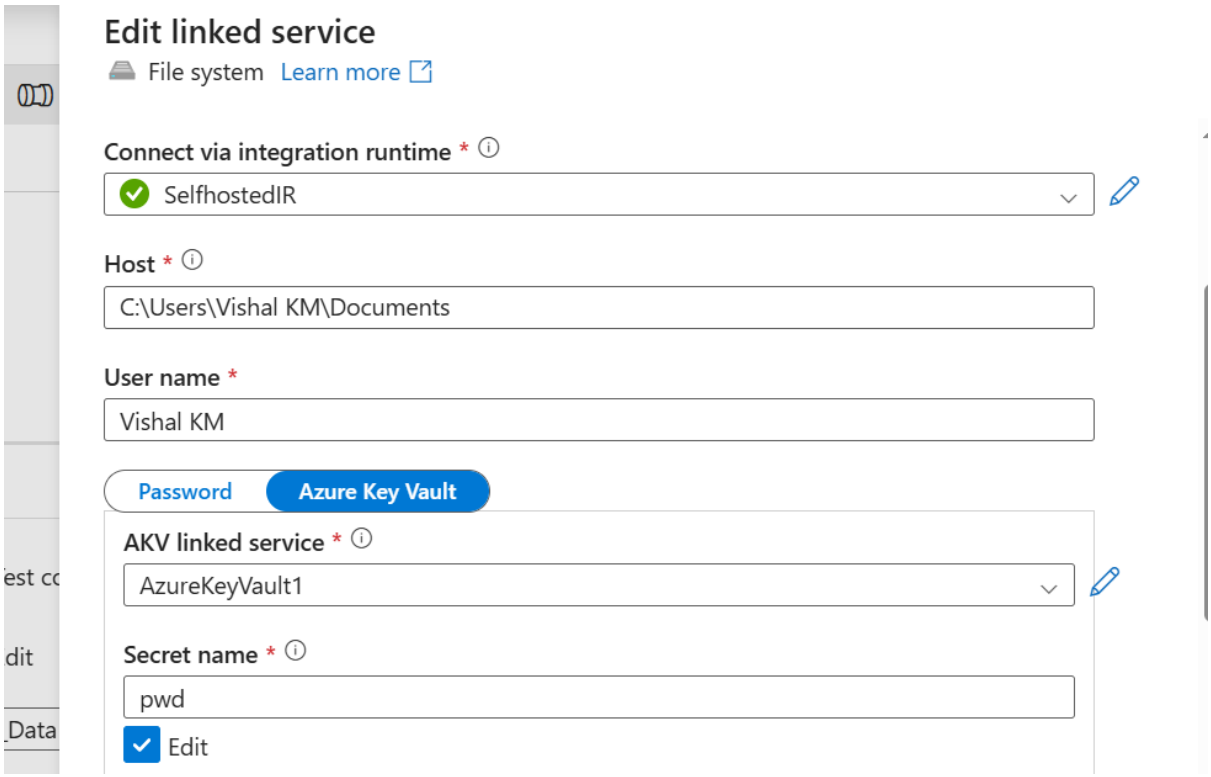
- Set up a copy activity in ADF to transfer data from the backend team's Azure Storage account to the Raw (Bronze) container in the Data Lake.
-

Source Files:

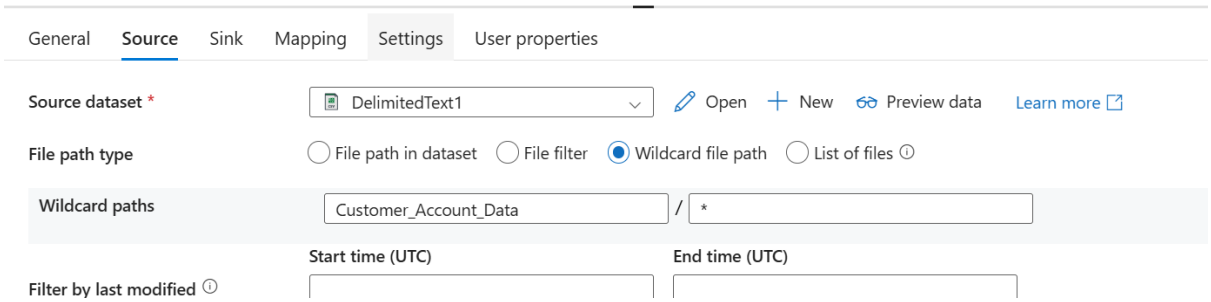
- accounts.csv
- customers.csv
- loan_payments.csv
- loans.csv
- transactions.csv



Create SelfhostedIR and Filesystem linked service

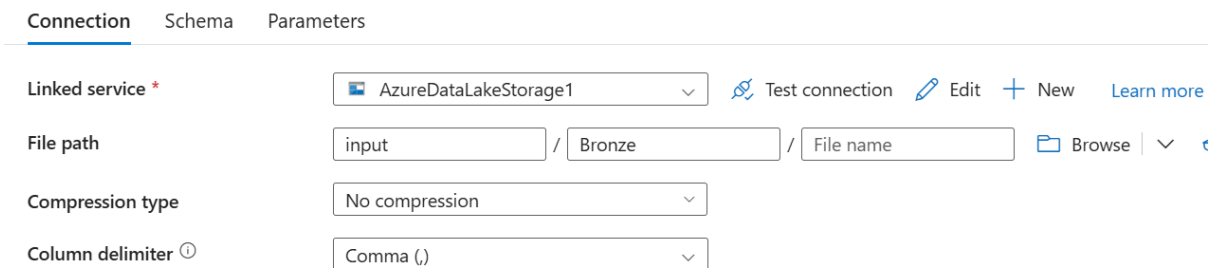


- Using KeyVault secrets we provide the password securely
- Using wildcard path , I have copied all the documents into my bronze folder in my container



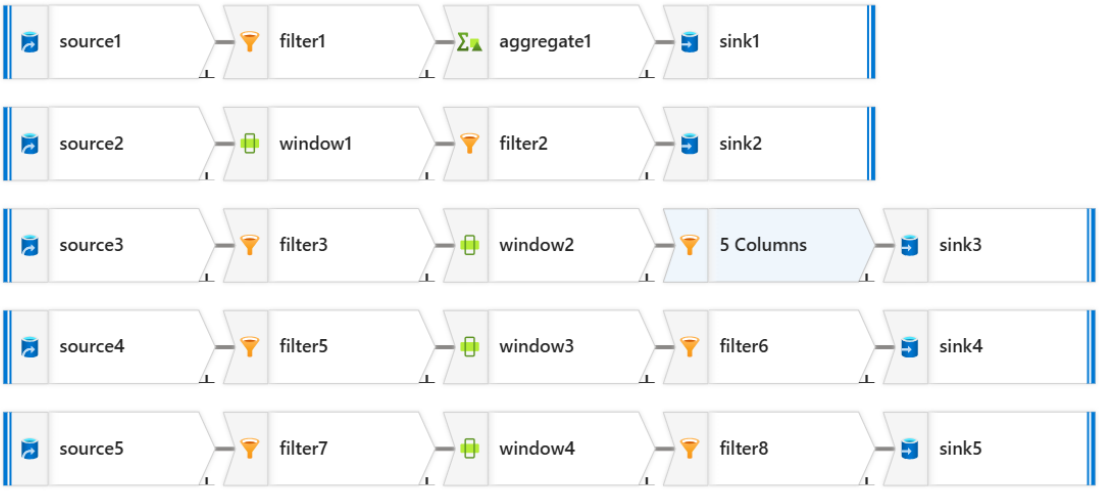
Sink:

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



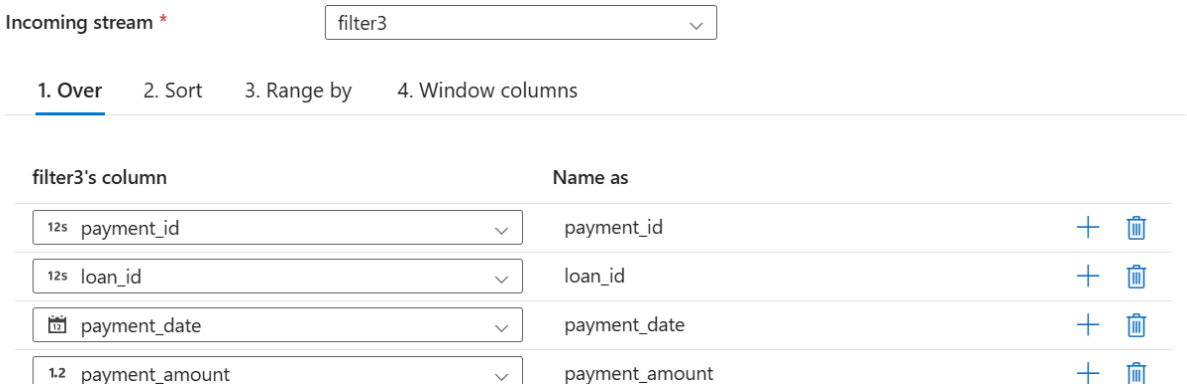
Step 2: Data Cleaning and Transformation (Bronze Layer)

Tool: Azure Data Factory (ADF) - Data Flows

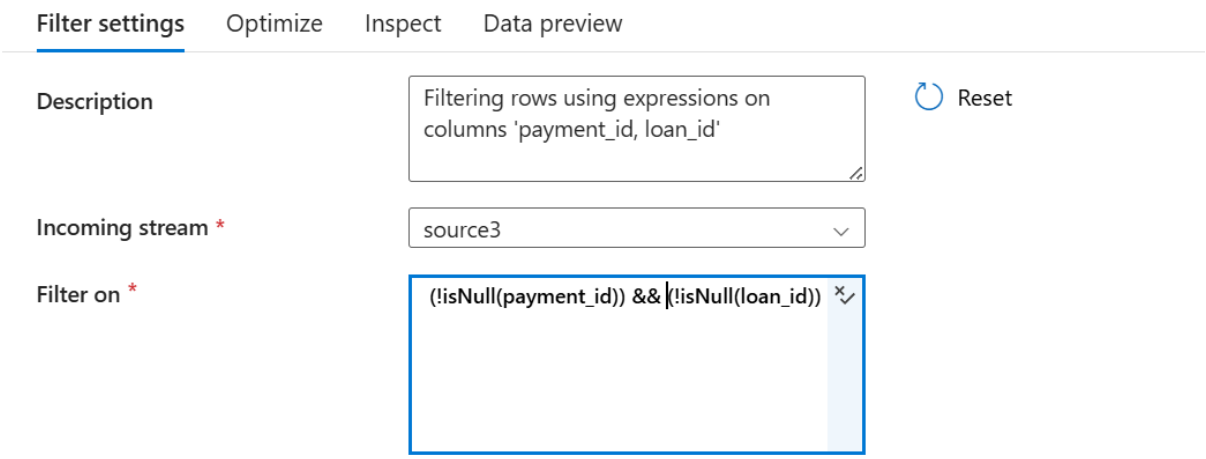


Sub-Steps:

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.



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



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

Step 3: Load to SQL Database using SCD Techniques

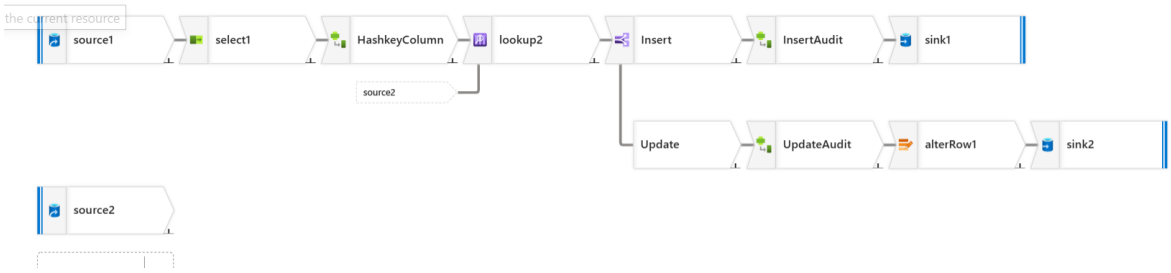
Tool: Azure Data Factory - Data Flows + Pipelines

Tasks:

1. Implement Slowly Changing Dimensions (SCD):

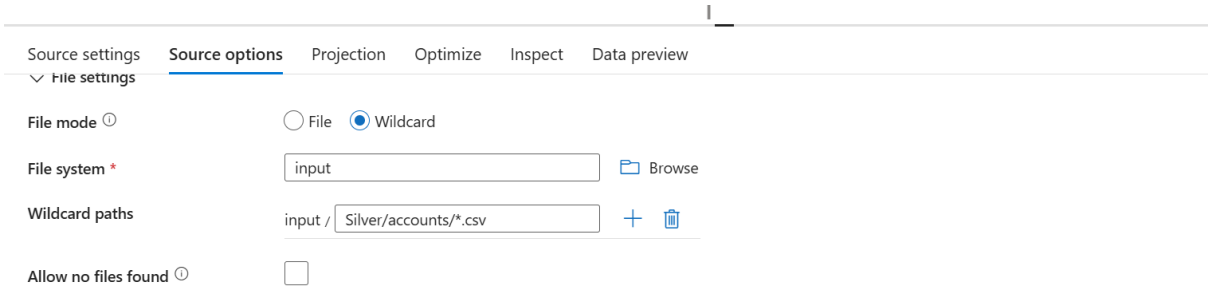
- **SCD Type 1:** Overwrite existing records
  - I have done the SCD Type-1 for the Accounts, loans, loan-payments, transactions

Accounts Dataflow



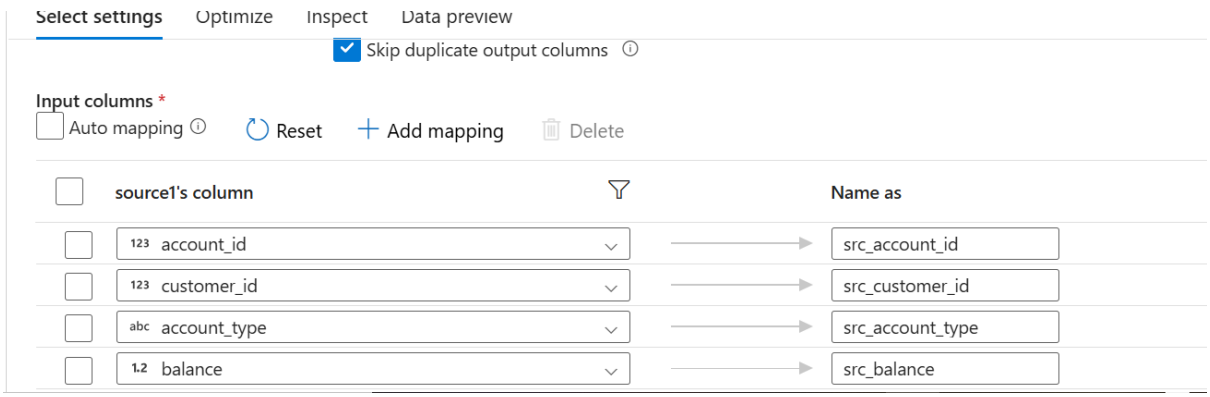
Source

From Silver.accounts folder using wildcard path



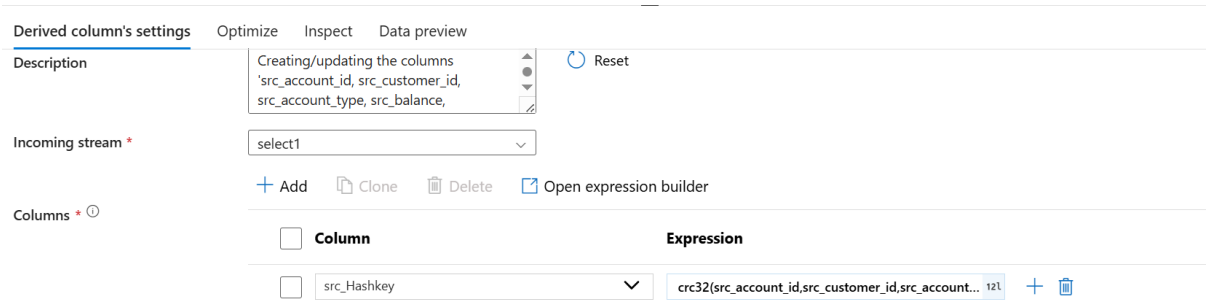
Select

To rename columns



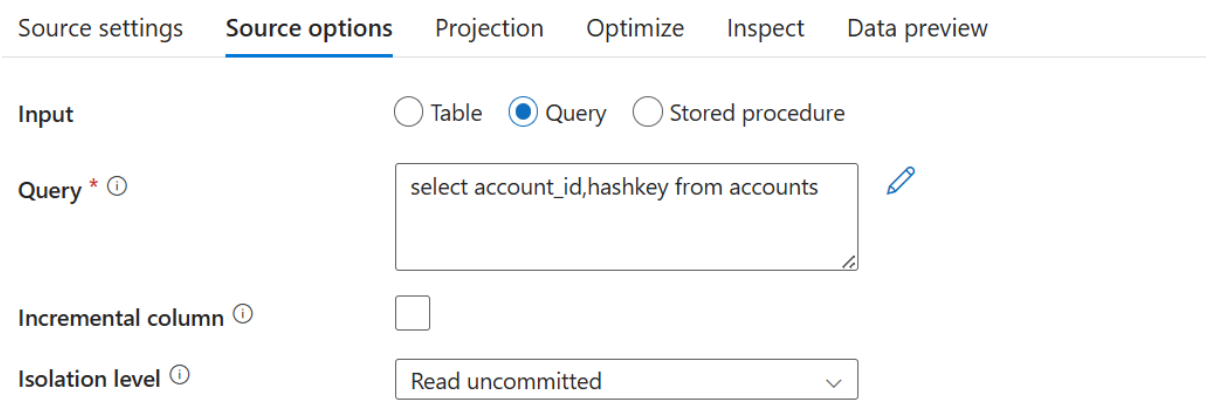
Hashkey derived Column Activity

crc32(src_account_id,src_customer_id,src_account_type,src_balance)



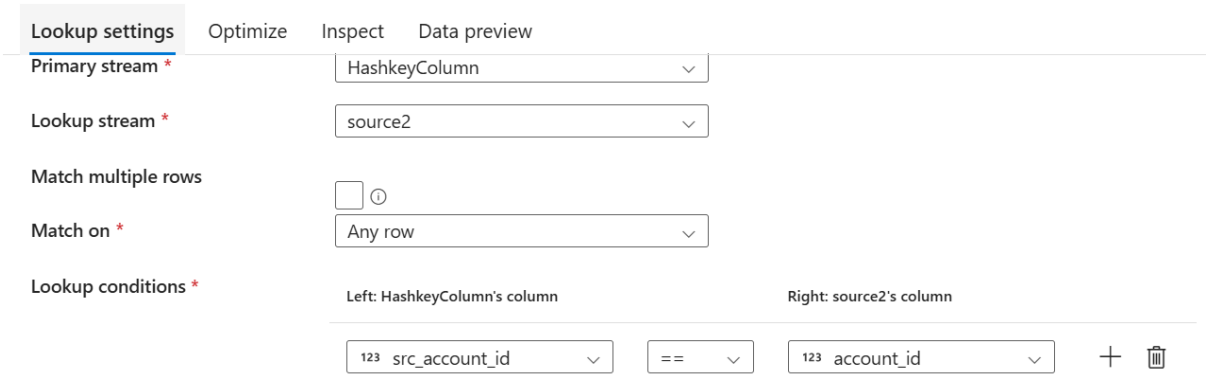
Source2

From sql table for lookup new records



LookUp

Lookup based on 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 settings | Optimize | Inspect | Data 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 settings | Optimize | Inspect | Data preview

Columns *

Add | Clone | Delete | Open expression builder

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

UpdateAudit derived column activity

Derived column's settings | Optimize | Inspect | Data preview

Columns *

Add | Clone | Delete | Open expression builder

Column	Expression
src_createdBy	"Dataflow"
src_createdDate	currentTimestamp()
src_updatedBy	"Dataflow-Updated"
src_updatedDate	currentTimestamp()

Sink1 for insert

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

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

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

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

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

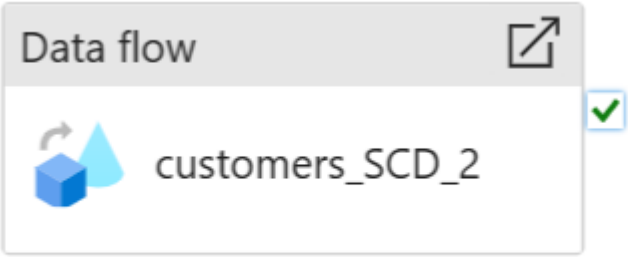
```
select * from accounts where account_id = 100
```

110 %

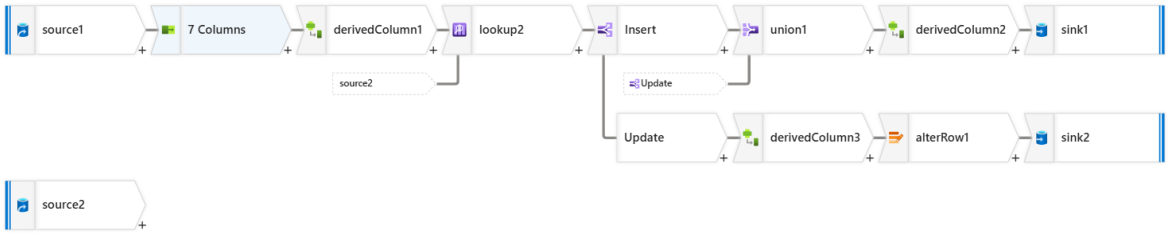
Results | Messages

	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 settings | Optimize | Inspect | Data preview | ← Previ

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 settings | Optimize | Inspect | Data 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 settings | Optimize | Inspect | Data preview

Columns * : + Add | Clone | Delete | Open expression builder

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 settings | Optimize | Inspect | Data preview

src_last_name, src_address, src_city,

Incoming stream * : Insert@Update

Columns * : + Add | Clone | Delete | Open expression builder

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 * alterRow1 Learn more

Description Add expressions to alter rows Reset

Incoming stream * derivedColumn3

Alter row conditions *

Update if 1==1

Sink 1 for Insert

Sink | Settings | Errors | Mapping | Optimize | Inspect | Data preview

Schema name * dbo Refresh

Table name * customers

Table action ( ) None ( ) Recreate table ( ) Truncate table

Update method

[x] Allow insert

[ ] Allow delete

[ ] Allow upsert

[ ] Allow update

Sink 2 for Update

Sink | Settings | Errors | Mapping | Optimize | Inspect | Data preview

Schema name * dbo Refresh

Table name * customers

Table action ( ) None ( ) Recreate table ( ) Truncate table

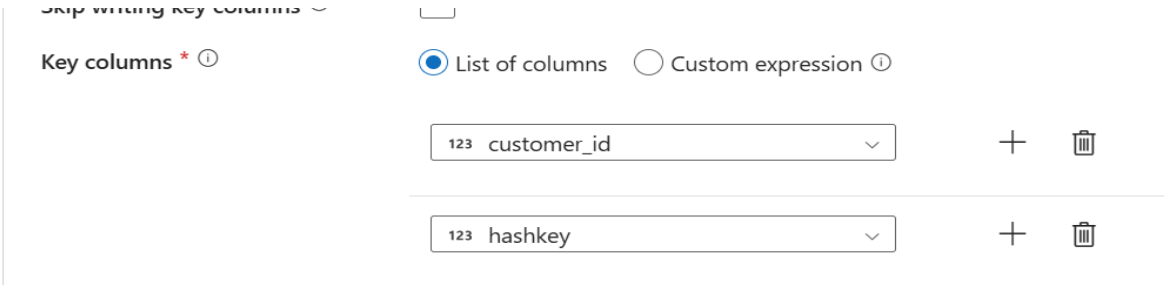
Update method

[ ] Allow insert

[ ] Allow delete

[ ] Allow upsert

[x] Allow update



Output Validation

Initial data record

```
87,William,McDonald,8686 Maple Ave,Haileybury,,,1
88,vishal,CardinalHealth,222,sad,ON,POJ0A1,1
```

Day2 Updated record

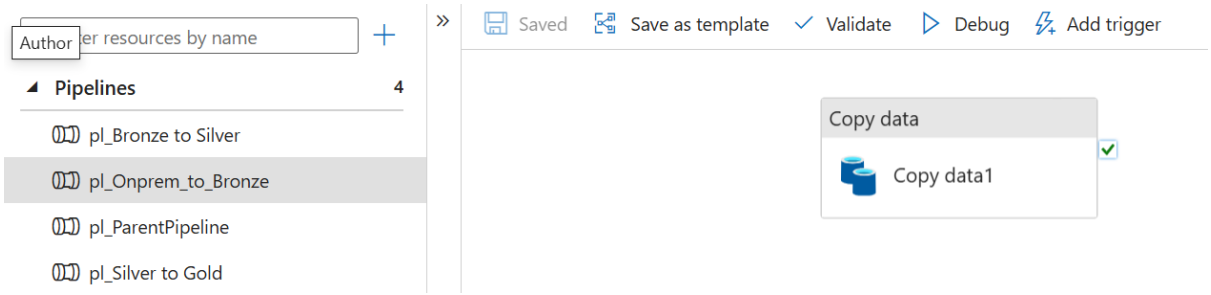
```
87,William,McDonald,8686 Maple Ave,Haileybury,,,1
88,vishal,CH,222,sad,ON,POJ0A1,1
```

After SCD-2

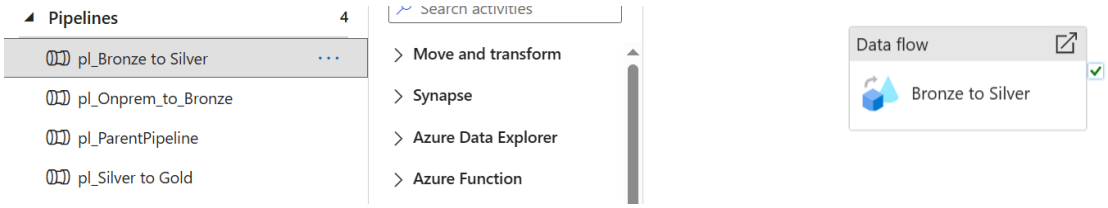
91	88	vishal	cardinal...	222	sad	ON	POJ0A1	2025-04-21 02:02:56.313	Dataflow	2025-04-21 02:40:53.920	Dataflow-...	2813940578	0
92	88	vishal	CH	222	sad	ON	POJ0A1	2025-04-21 02:43:43.337	Dataflow	2025-04-21 02:43:43.337	Dataflow	1366610954	1

Pipelines

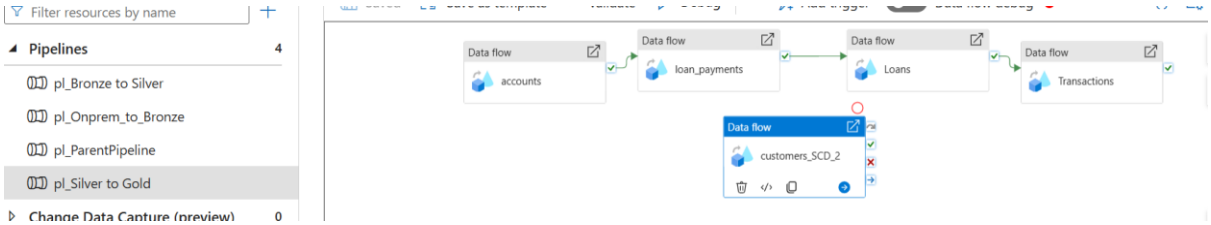
- **Local to Bronze Layer**



- **Bronze to Silver Layer**

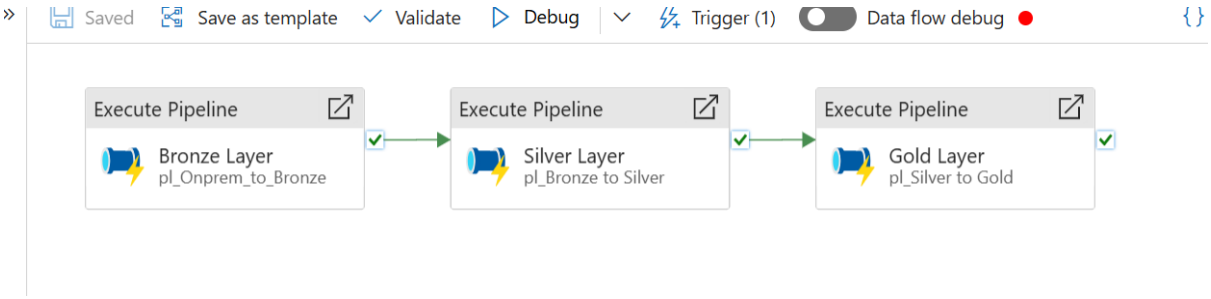


- **Silver to Gold Layer**

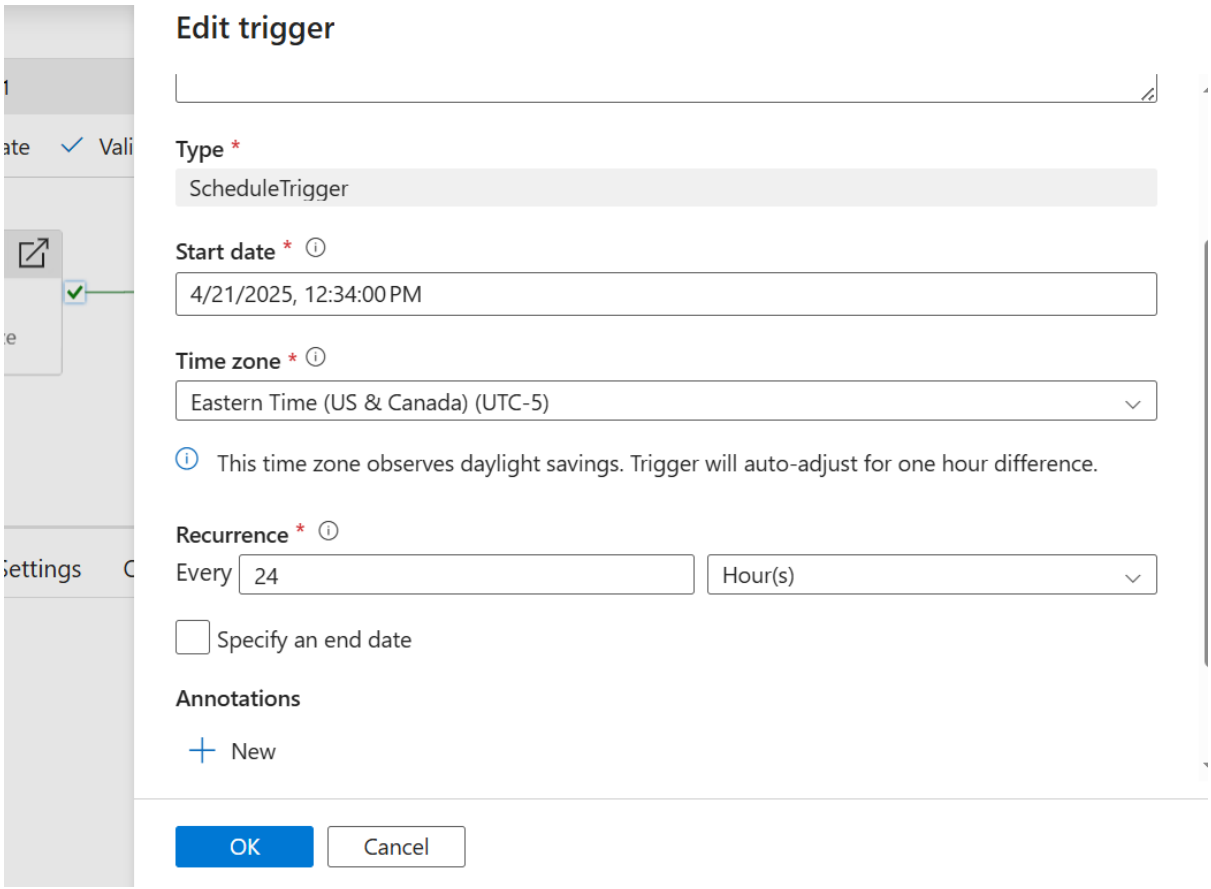


Master Pipeline Creation:

- Use Execute Pipeline Activity to trigger child pipelines in sequence



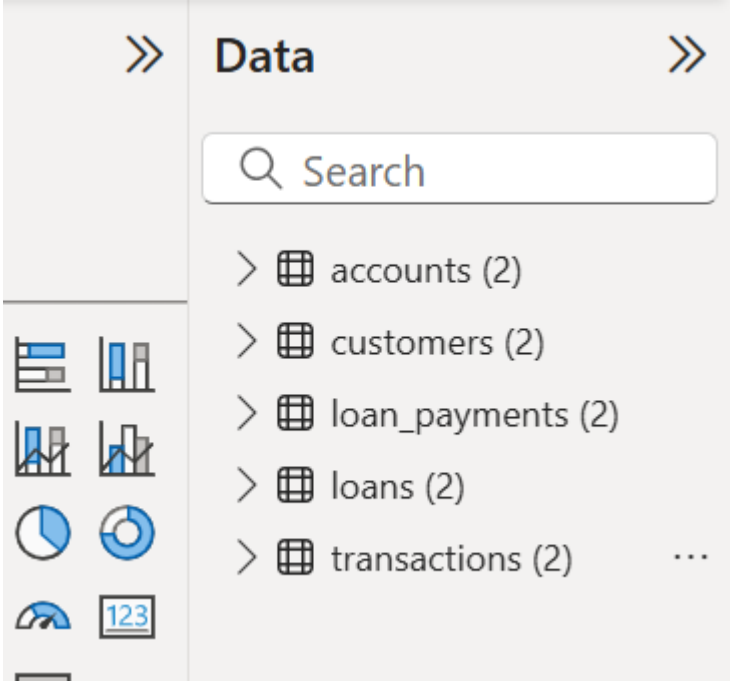
Scheduled Trigger



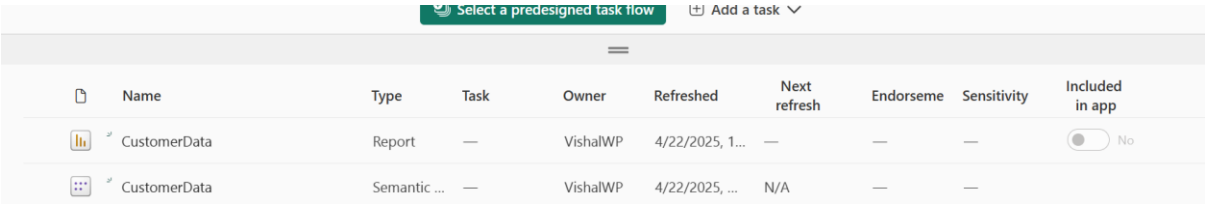
Step 4: Data Visualization Using Power BI

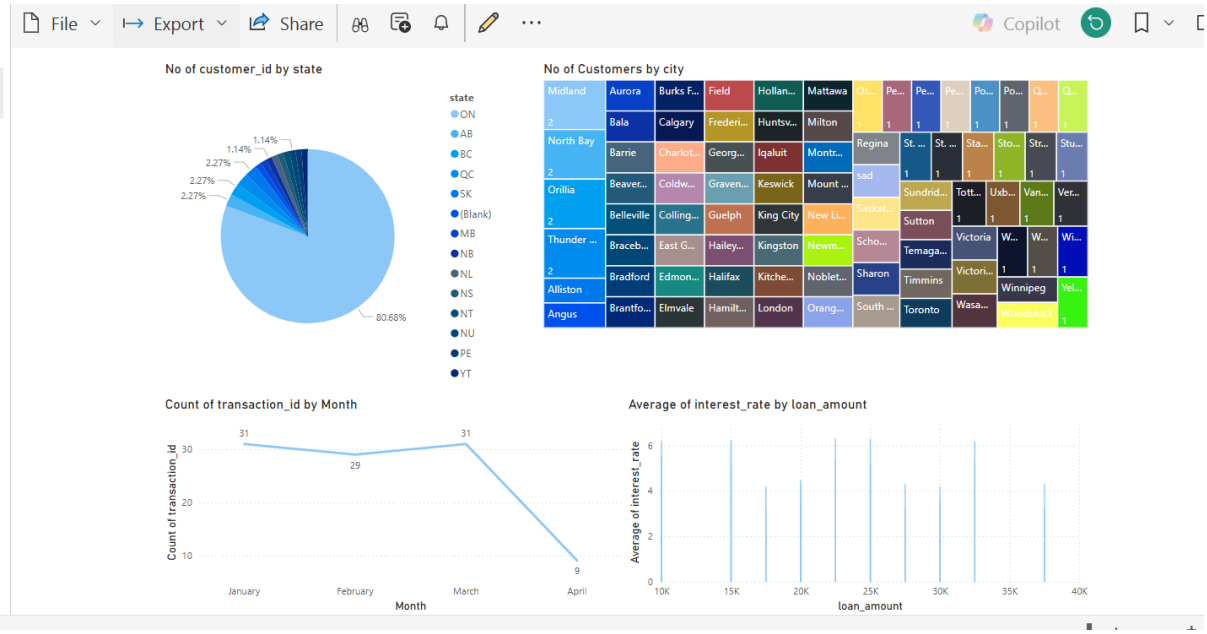
Tasks:

1. Connect Power BI to SQL Database tables



2. Publish reports to Microsoft Fabric Workspace





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