Project Report: Customer Account Analysis Pipeline

Project Title: Bootcamp Project 1 - Data Pipeline for Customer Account Analysis

Objective: To design and implement an end-to-end data pipeline using Azure Data Factory (ADF) to ingest, transform, and store customer account data from raw source files. The pipeline ensures scalable, clean, and analytics-ready data loaded into Azure SQL Database, supporting downstream Power BI reporting.

Source Files Used:[AI Bank Dataset]

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

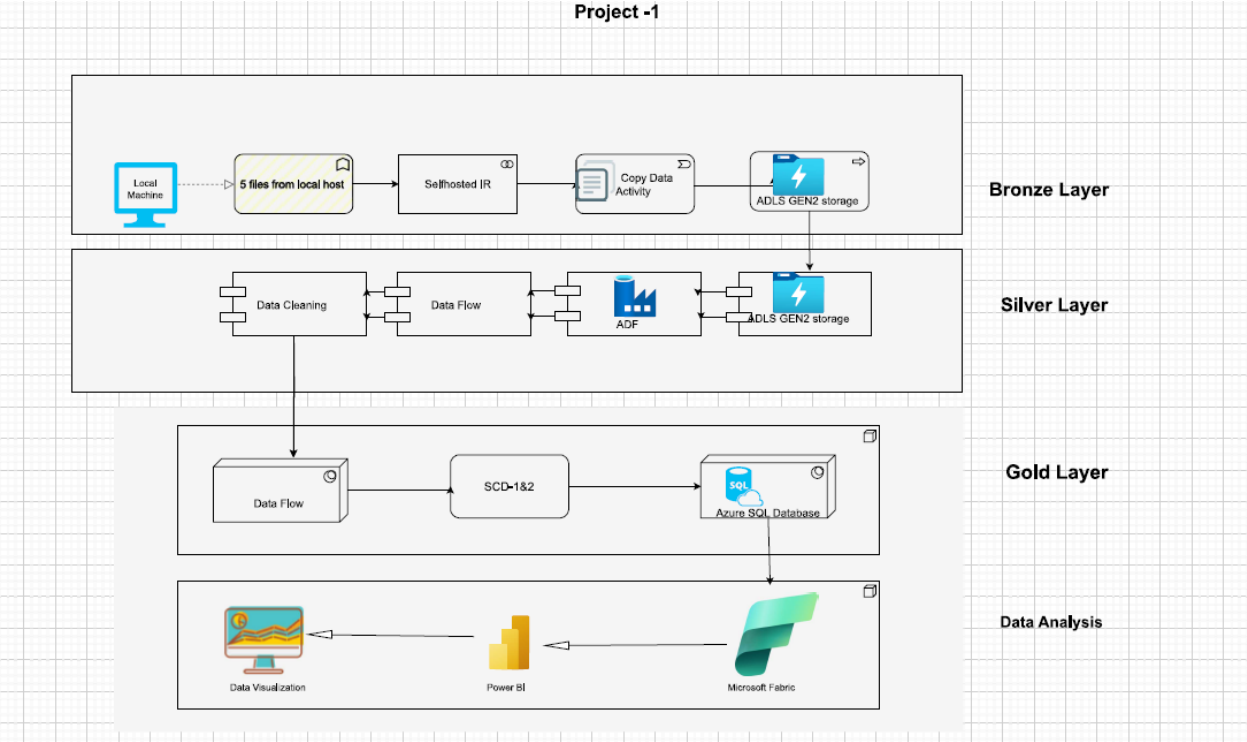
Storage Layers Implemented:

- Bronze: Raw ingestion from backend storage
- Silver: Cleaned and transformed data
- Gold: SCD-managed data models in SQL DB

Tools Used:

- Azure Data Factory
- Azure Data Lake Storage (Gen2)
- Azure SQL Database
- SQL Server Management Studio (SSMS)

❖ Architecture of project

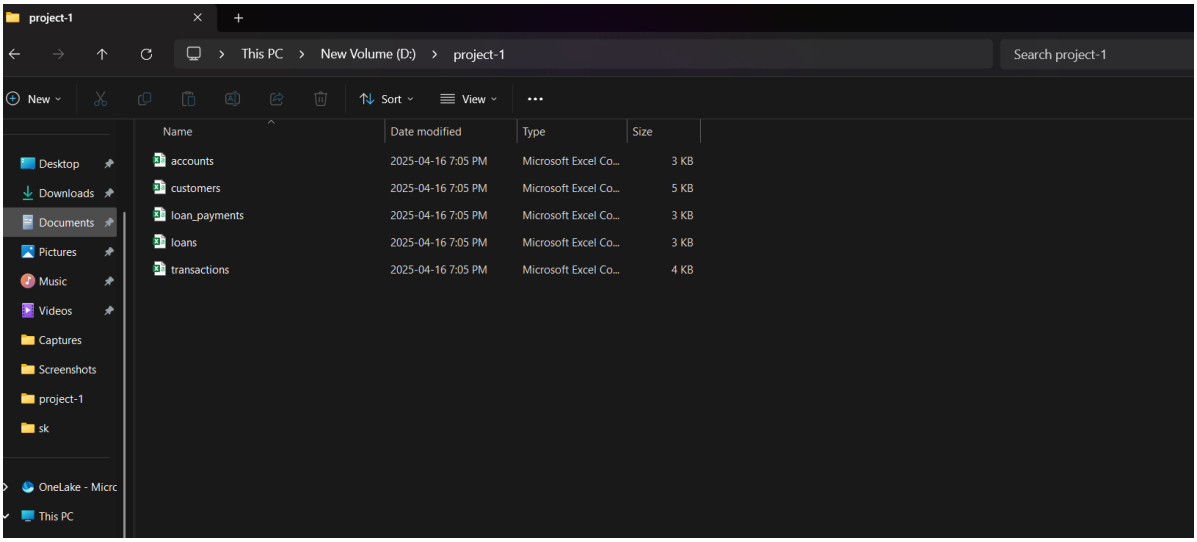


❖ Source Data

First we need the source data to use for this project so here is the link from where we will download the five files for this project.

Link :- [AI Bank Dataset]

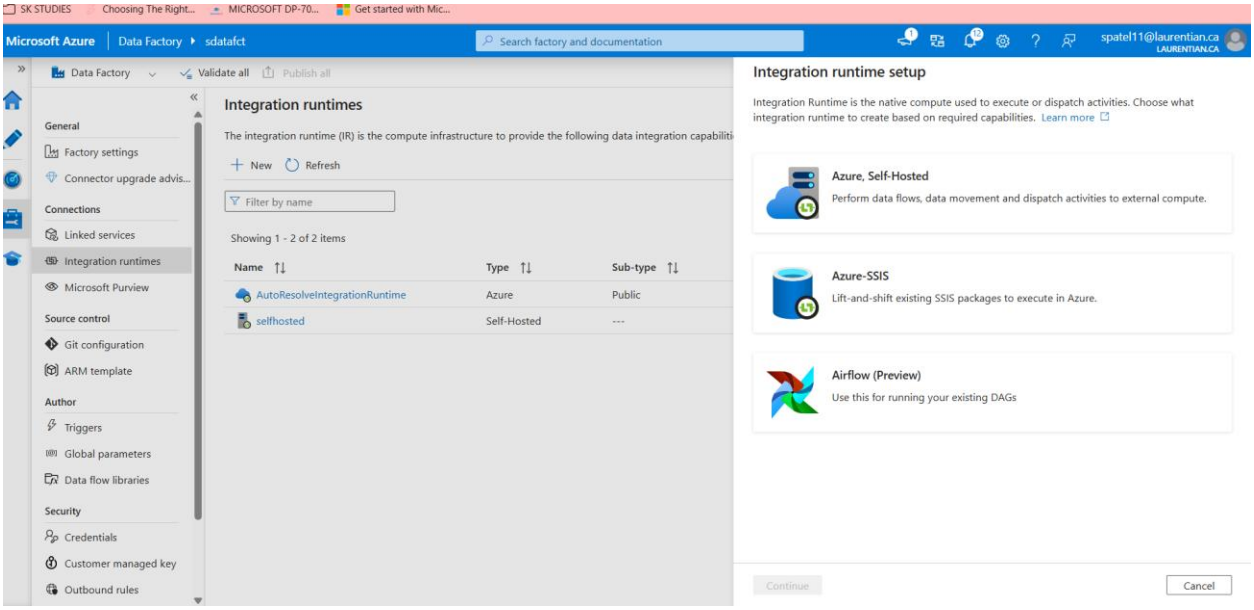
Please make sure that store these files in drive C and the folder name is project 1 to avoid unnecessary errors.



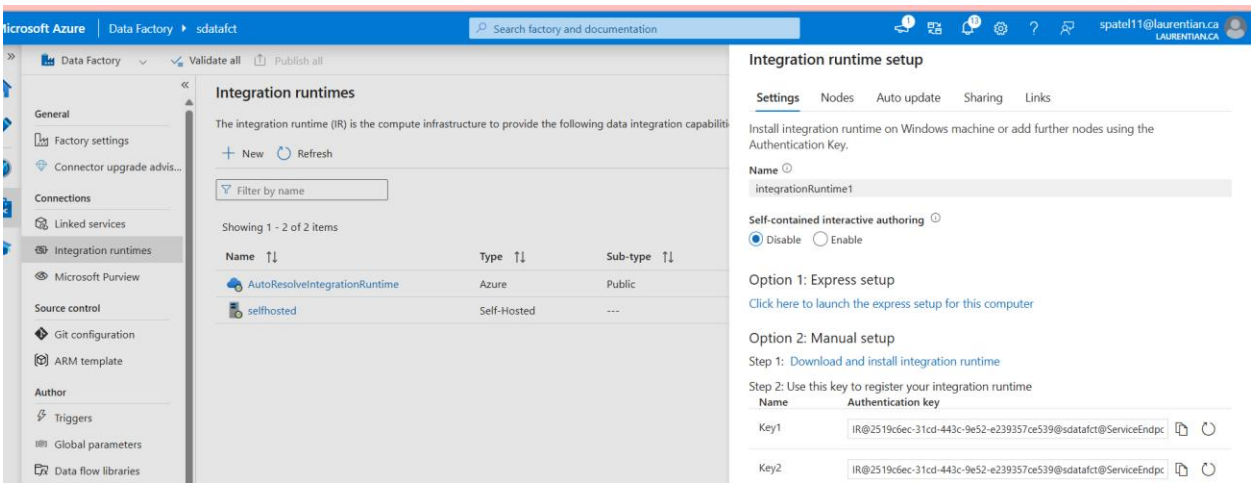
❖ **Bronze Layer** :- Data Ingestion

In this layer we need the self hosted integration runtime to transfer 5 files from local machine to Azure data lake Gen 2 storage account.

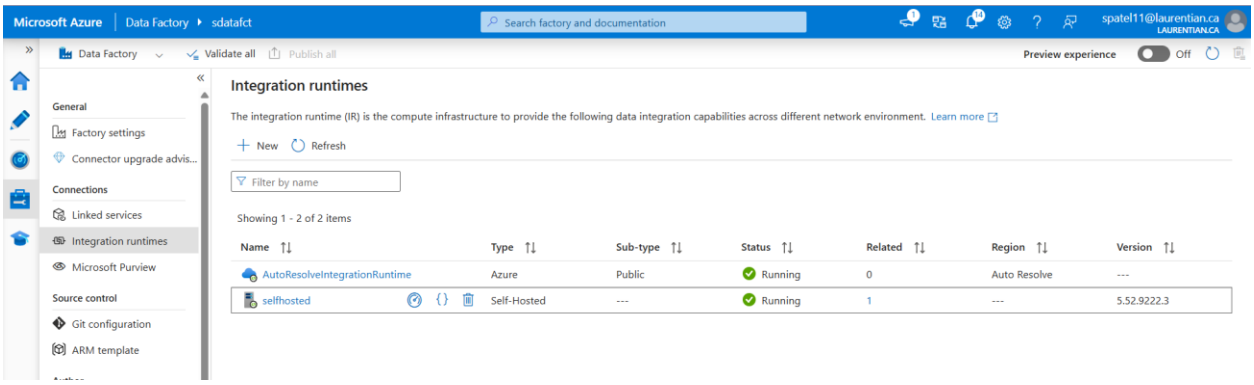
First, we have to create selfhosted IR in our Azure data Factory. For that go to ADF—manage -- integration runtime--create new integration—select azure selfhosted IR



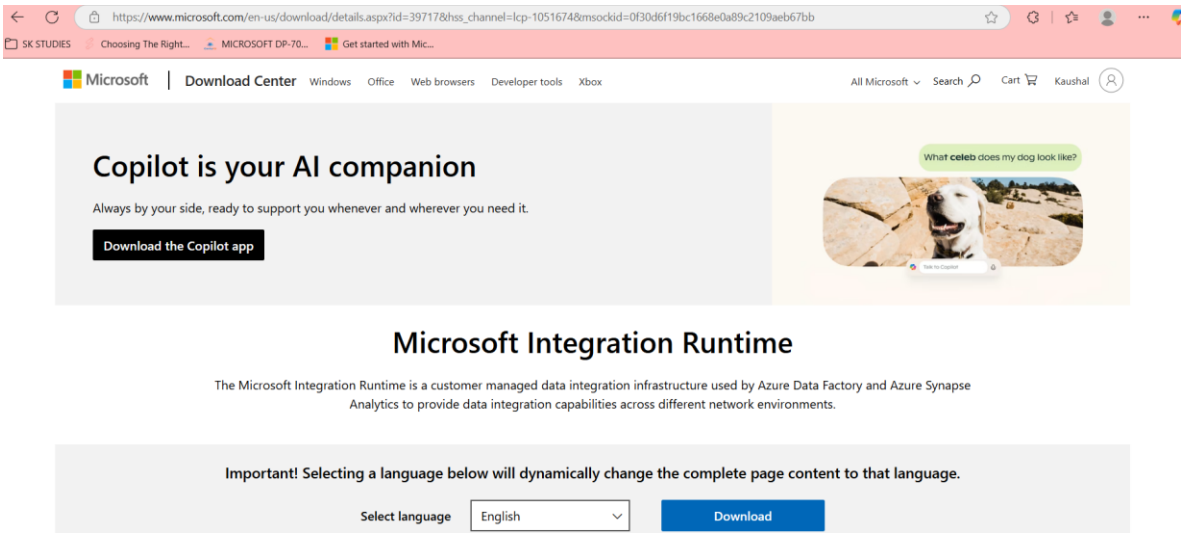
Once the self hosted IR created it will generate 2 links we have to copy anyone link that we have to provide after the installing self hosted IR in our local machine to connect ADF



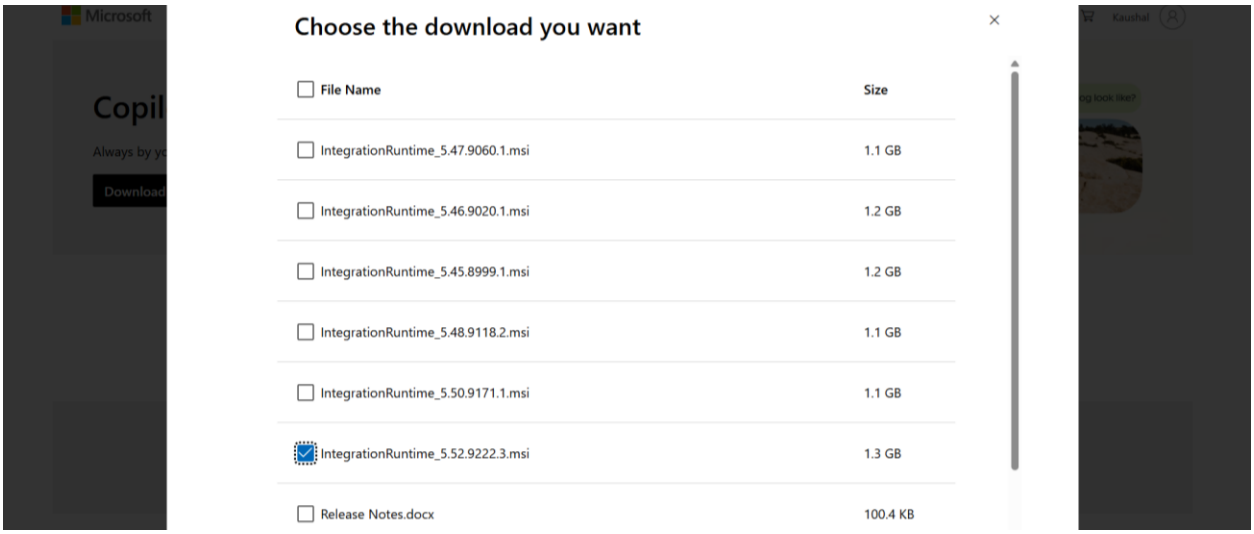
Once it is successfully created and connected to the local system that means our files server it will show as a running status.



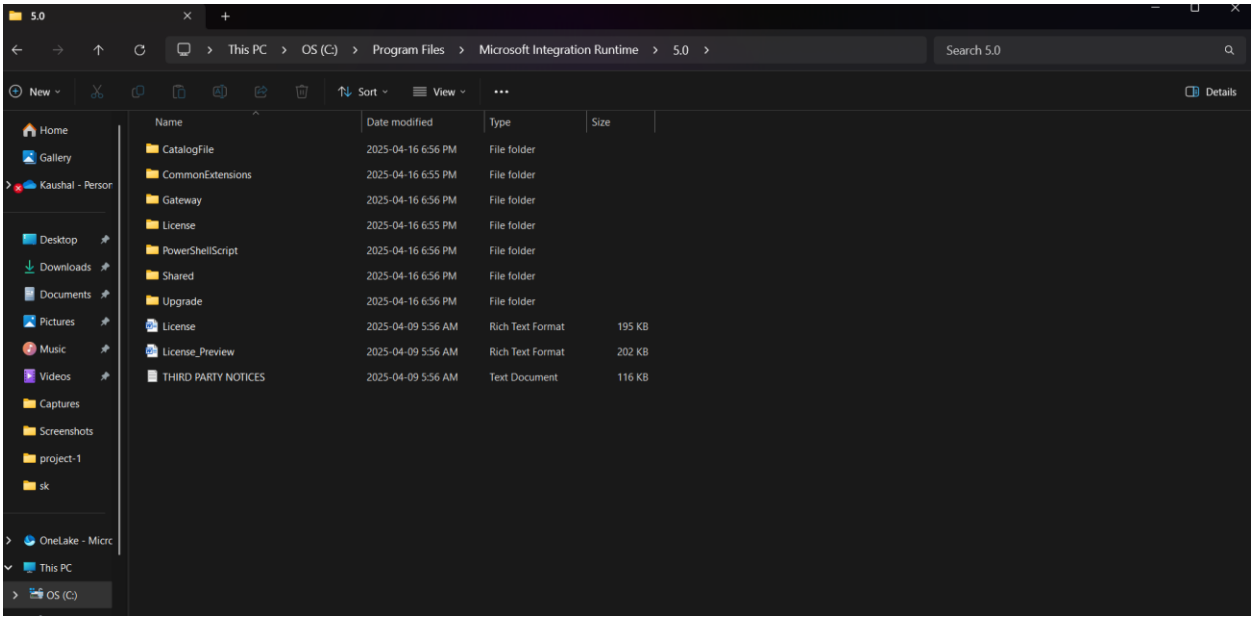
Now , we install the self hosted integration runtime in our local machine.For that Link :- [Download Microsoft Integration Runtime from Official Microsoft Download Center]



Then we click on download to see all the available version of self hosted integration runtime and here we select the latest version of integration runtime to download in our local machine.

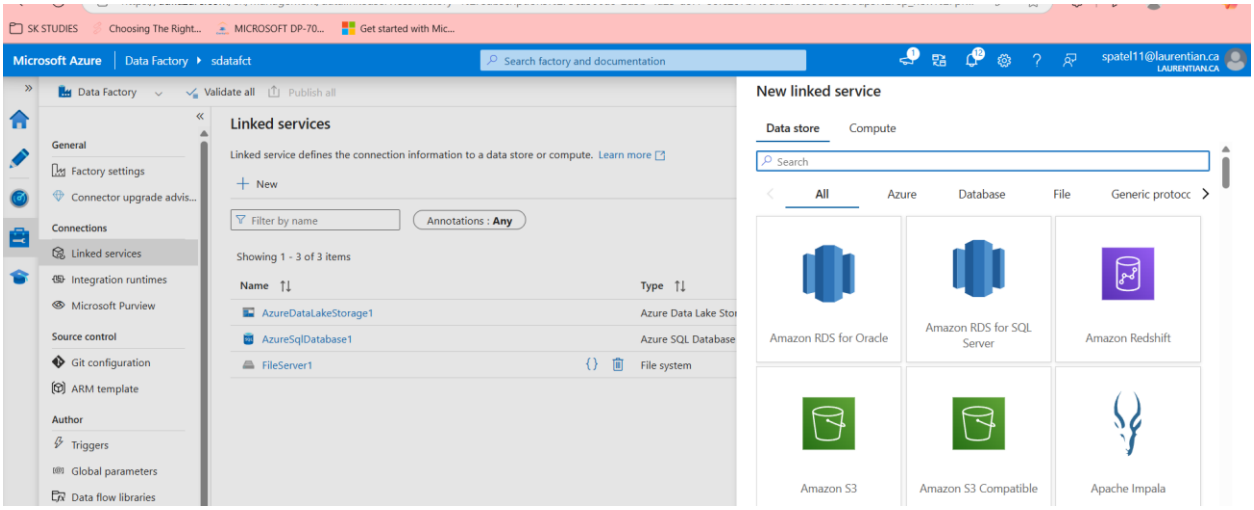


once it downloaded , we can see that self hosted IR is stored in drive C in program files.



now we have to create the link service between local machine and ADL SN2 storage account for that we will go to Azure data factory end select manage in that we will select link service in create on new link service.

To create a new link service we have to search for the file system.



To connect IR :- we have to select the self hosted

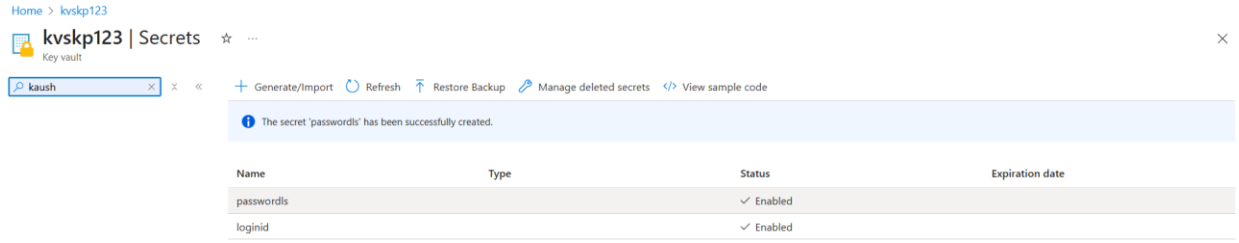
Host :- we have to provide the path which have all 5 files in our local system

Username :- we have to provide our laptop user name [use whoami in CM to know your use name]

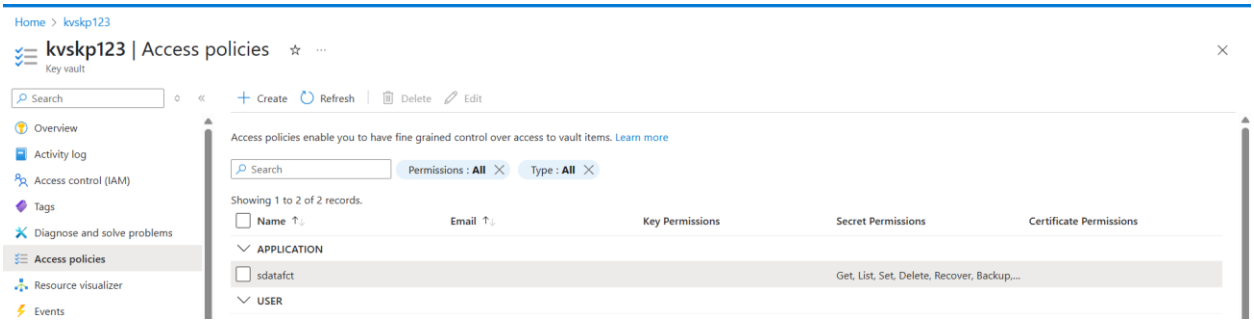
password :- we have to provide our Microsoft account password/laptop login password / the email id password which we have in our laptop accounts.

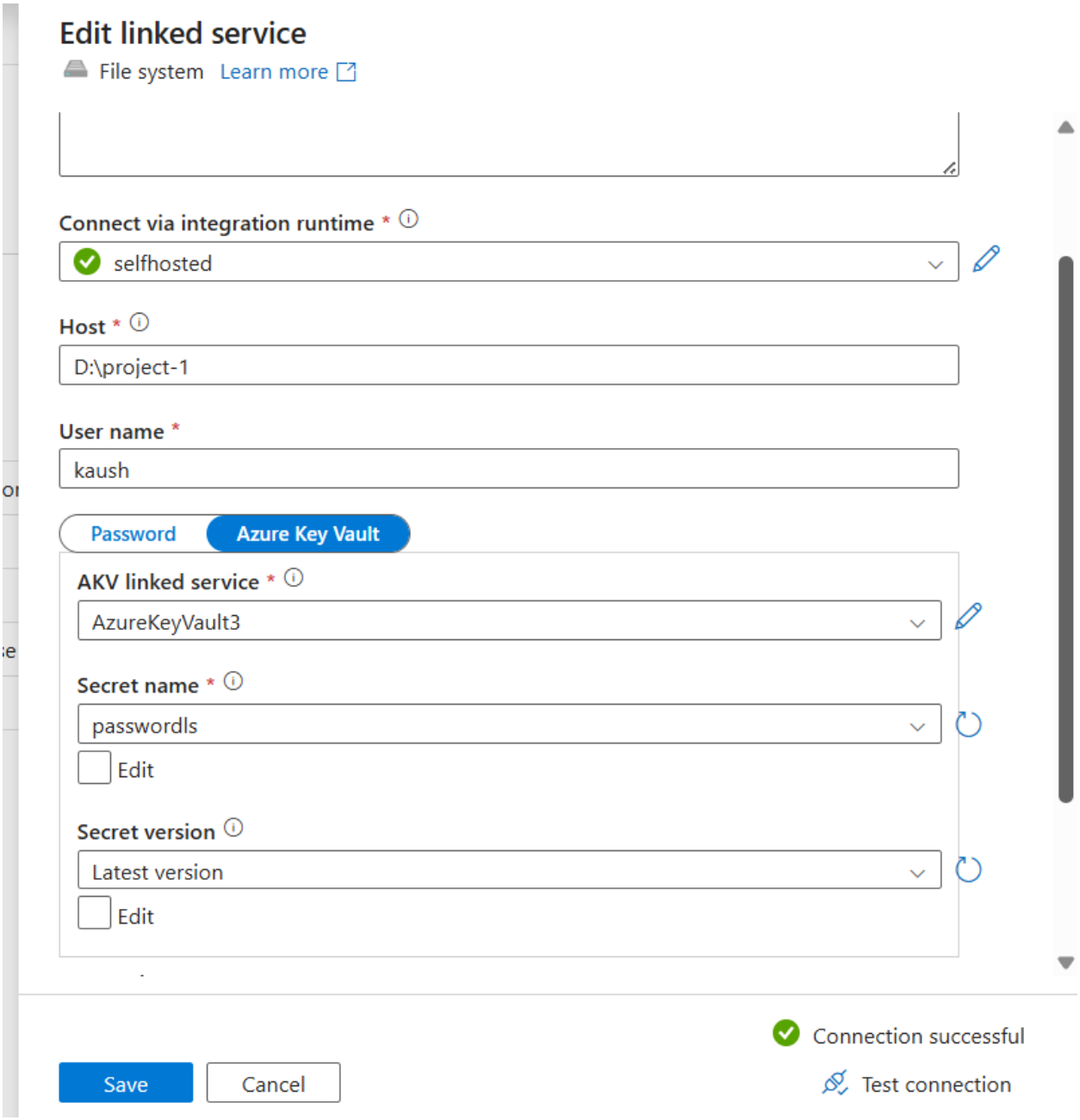
For me I use Microsoft account password because that account I also used to login into my laptop account.

Save the usename and password in Key Vault



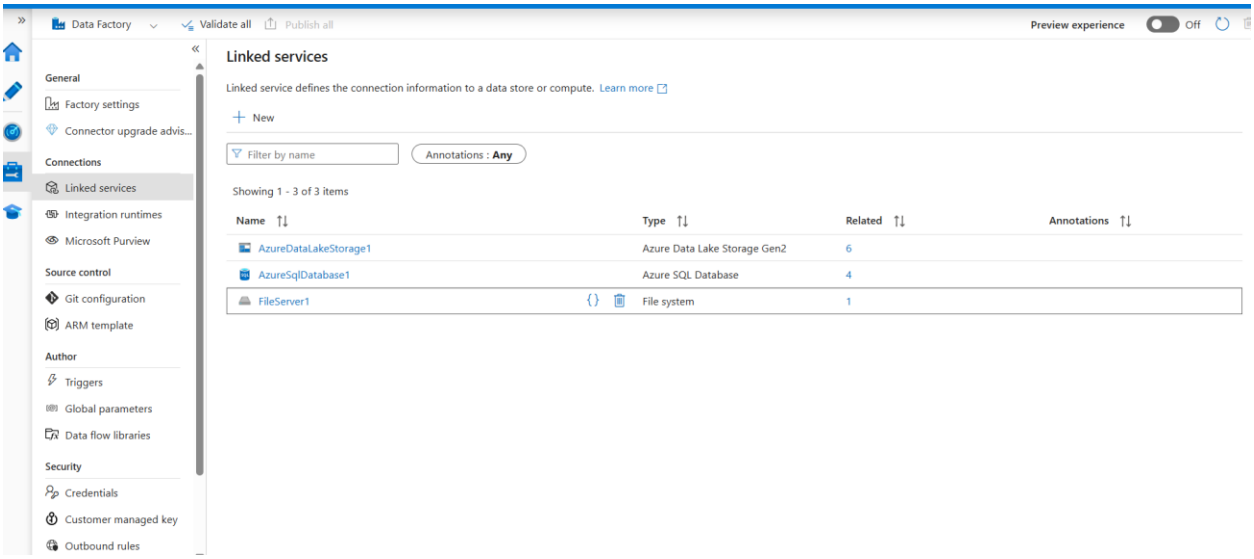
After this go to Access Control ---select get and list from Secret ---service principle—select ---azure data factory to provide permissions to access azure key vault secrets.



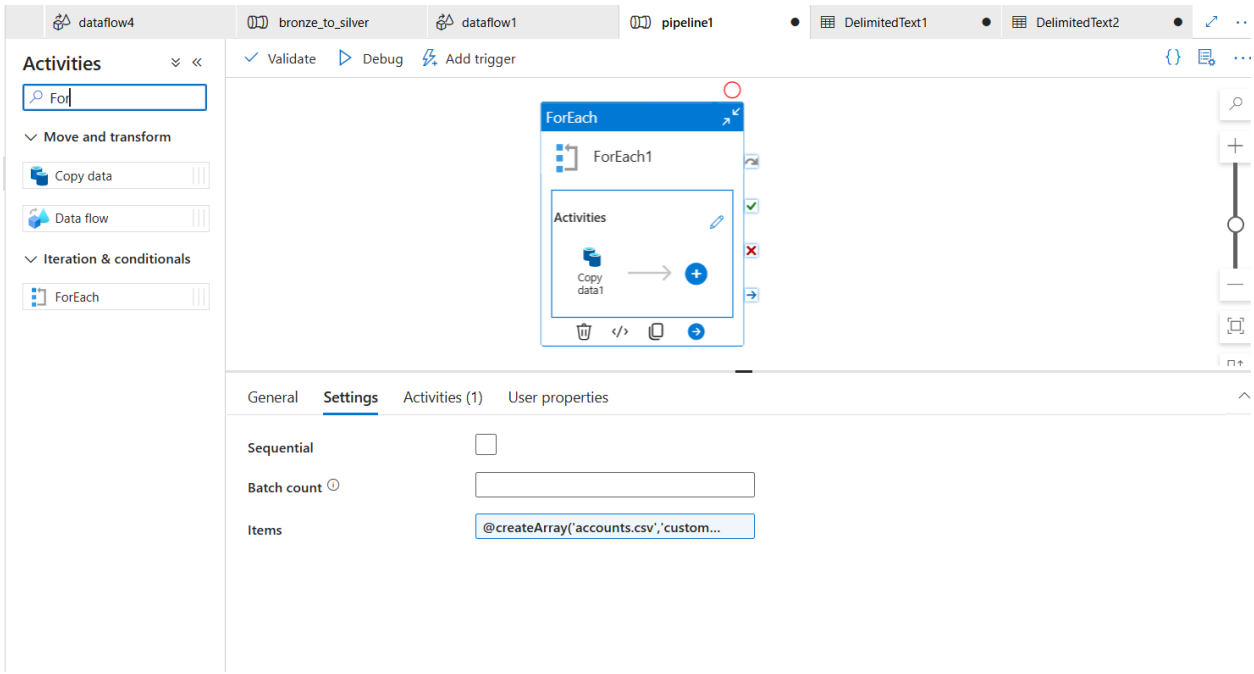


We have to select our password for secret name.

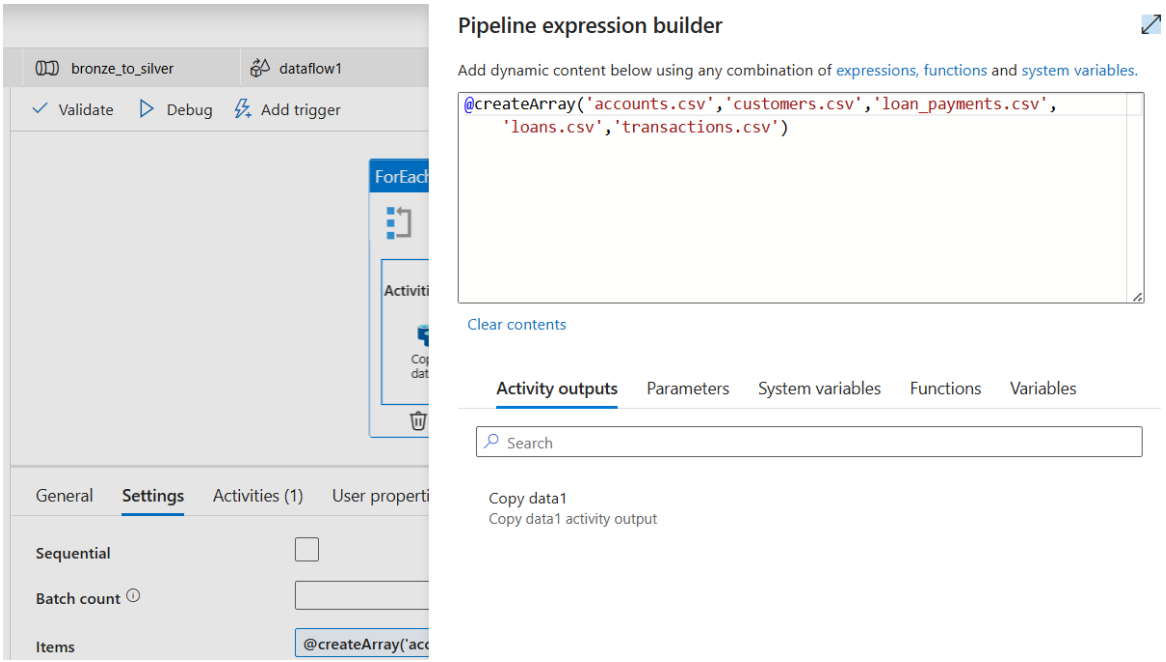
Now we can see that file server linked service created successfully



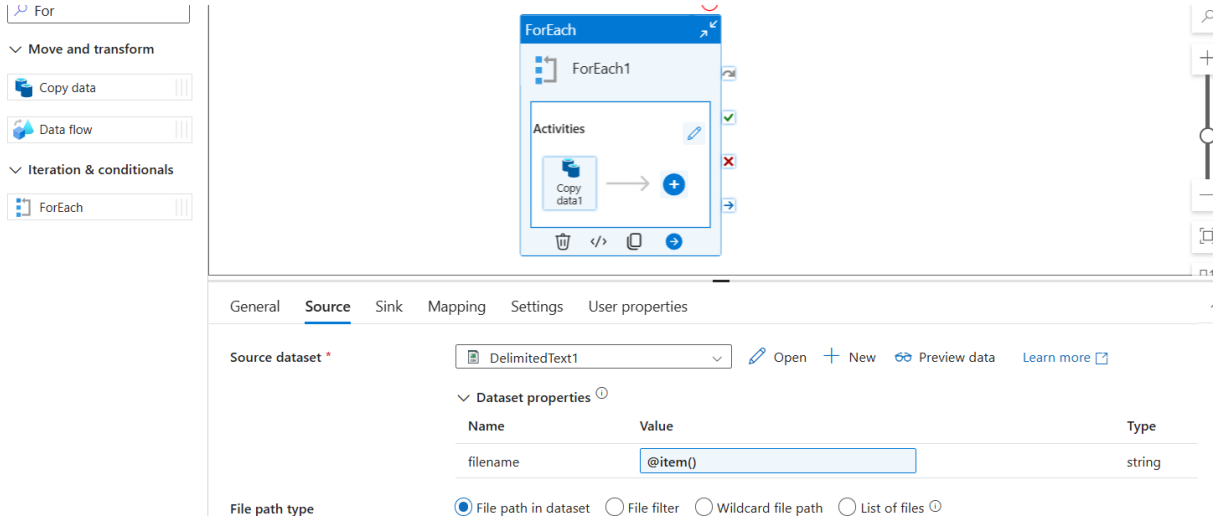
the next step is we must create a pipeline to transfer the data from our local machine to ADLS. For that Select Foreach activity ---inside it ---copy data

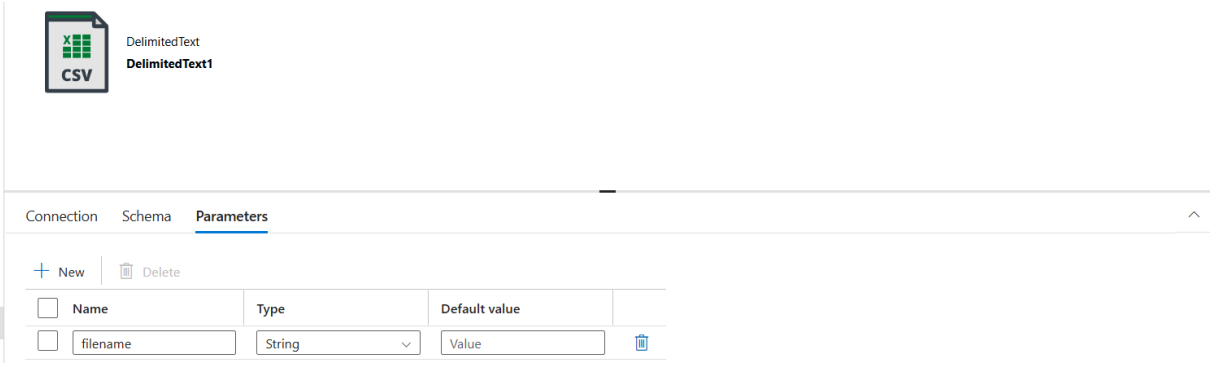


We passed array as a input of foreach loop which is 5 file name .

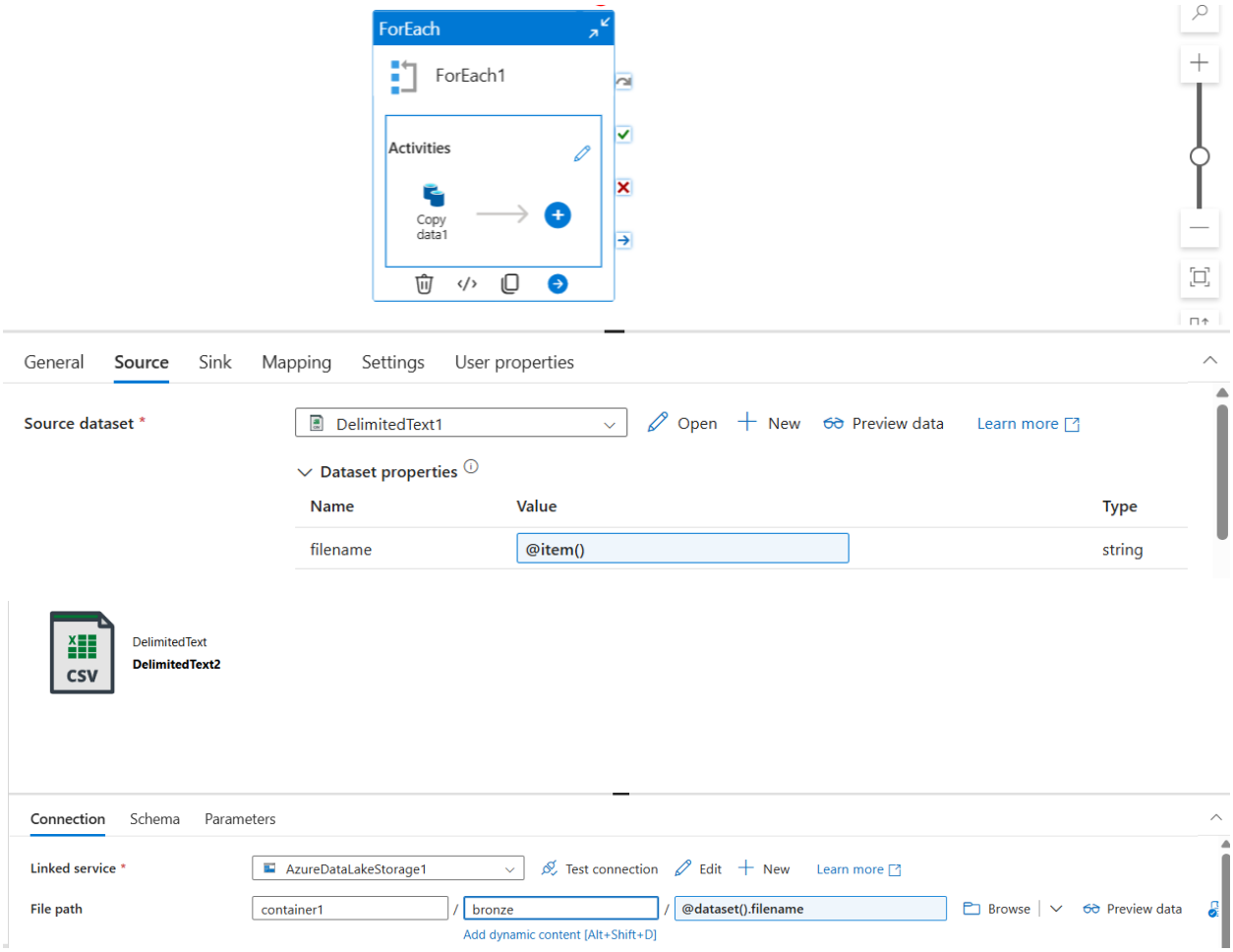


As a source we have to select file server and csv file types. And also provide filename dynamically.

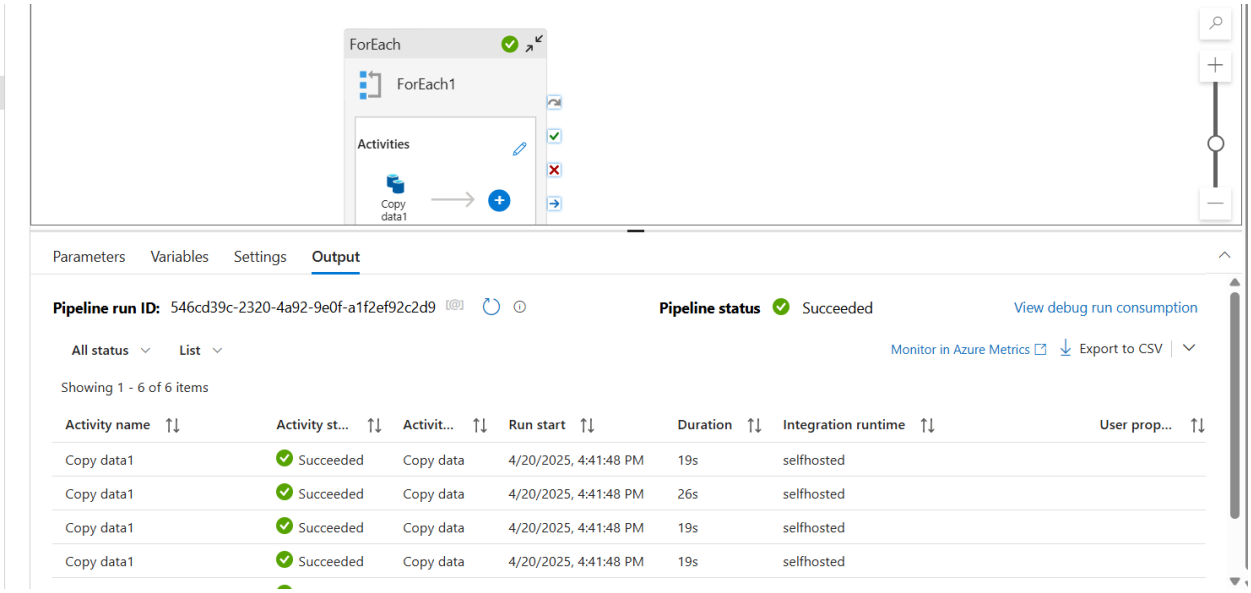


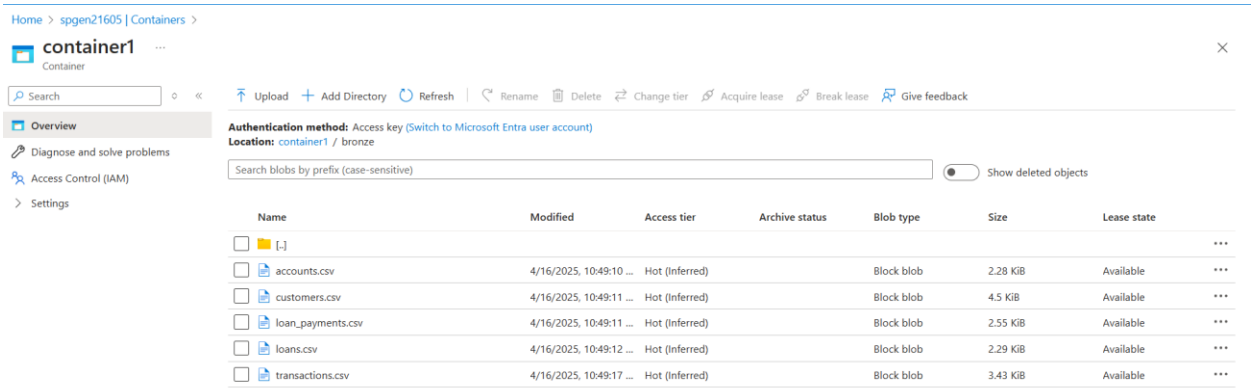


Same a sink we have to select ADLS gen 2 and provide csv file type and provide folder name as bronze.



Pipeline run successfully.Our Bronze layer completed .



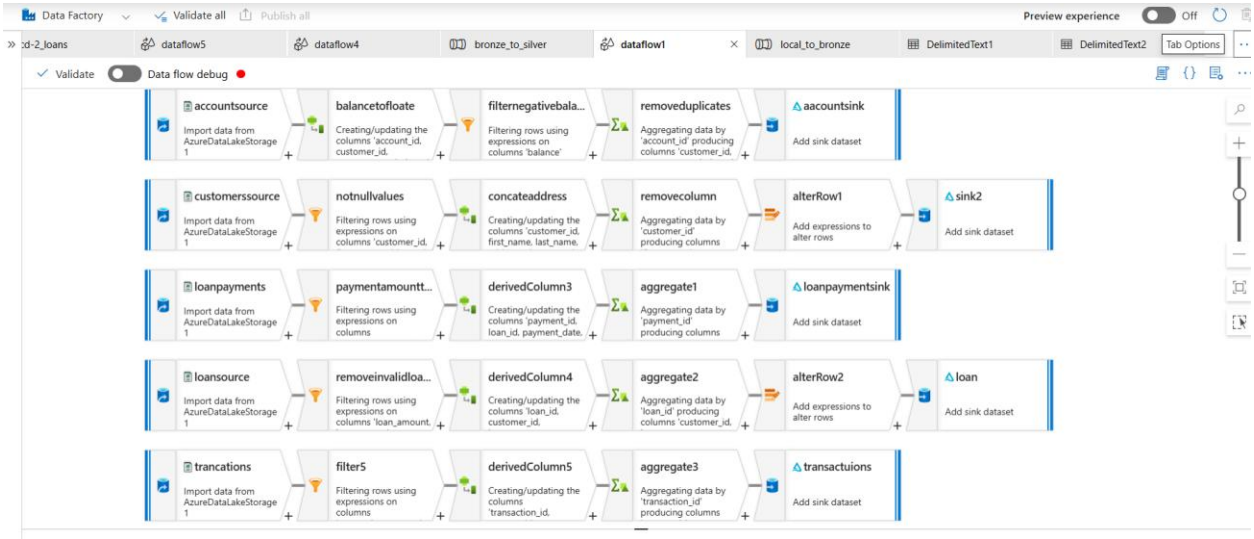


❖ Silver Layer :- Data Cleaning

Objective: Remove duplicates, nulls, and apply schema transformations.

Create Data Flows in ADF

- Go to Author > Data Flows > + New Data Flow



---

1. accounts.csv

Schema

account_id, customer_id, account_type, balance

Data Flow 1 Steps in ADF

1. Source

- Dataset: bronze/accounts.csv
- Projection: Set schema manually

2. Derived Column

- Add: balance_float = toFloat(balance)

3. Filter

- Condition:
- toFloat(balance) >= 0

4. Aggregate

- Group by: account_id
- Aggregates:
- customer_id = first(customer_id)
- account_type = first(account_type)
- balance = first(balance_float)

5.Sink

- Path: silver/accounts/
- Format: Delta

---

2. customers.csv

Schema

customer_id, first_name, last_name, address, city, state, zip

---

Data Flow Steps in ADF

1. Source

- Dataset: bronze/customers.csv

2. Filter

- Condition:
- !isNull(customer_id) && !isNull(first_name) && !isNull(address) && !isNull(zip)

3. Derived Column

- Add:
- full_address = address + ', ' + city + ', ' + state + ' ' + zip

4. Aggregate

- Group by: customer_id
- Aggregates:
- first_name = first(first_name)
- last_name = first(last_name)
- full_address = first(full_address

5. Alter Row

- Condition: update if : 1==1

6. Sink

- Path: silver/customers/
- Format: Delta

---

3. loan_payments.csv

Schema

payment_id, loan_id, payment_date, payment_amount

Data Flow Steps in ADF

1. Source

- Dataset: bronze/loan_payments.csv

2. Filter

- Condition:
- toFloat(payment_amount) > 0

3. Derived Column

- Add:
- amount_float = toFloat(payment_amount)

4. Aggregate

- Group by: payment_id

- Aggregates:
- loan_id = first(loan_id)
- payment_date = first(payment_date)
- payment_amount = first(amount_float)

5. Sink

- Path: silver/loan_payments/
- Format: Parquet or Delta

---

4. loans.csv

Schema

loan_id, customer_id, loan_amount, interest_rate, loan_term

Data Flow Steps in ADF

1. Source

- Dataset: bronze/loans.csv

2. Filter

- Condition:
- toFloat(loan_amount) > 0 && toFloat(interest_rate) > 0

3. Derived Column

- Add:
- clean_amount = toFloat(loan_amount)
- clean_rate = toFloat(interest_rate)

4. Aggregate

- Group by: loan_id
- Aggregates:
- customer_id = first(customer_id)
- loan_amount = first(clean_amount)
- interest_rate = first(clean_rate)

- loan_term = first(loan_term)

5. Alter Row

- Condition: update if : 1==1

6. Sink

- Path: silver/loans/
- Format: Delta

---

5. transactions.csv

Schema

transaction_id, account_id, transaction_date, transaction_amount, transaction_type

---

Data Flow Steps in ADF

1. Source

- Dataset: bronze/transactions.csv

2. Filter

- Condition:
- toFloat(transaction_amount) > 0

3. Derived Column

- Add:
- clean_amount = toFloat(transaction_amount)
- clean_type = upper(transaction_type)

4. Aggregate

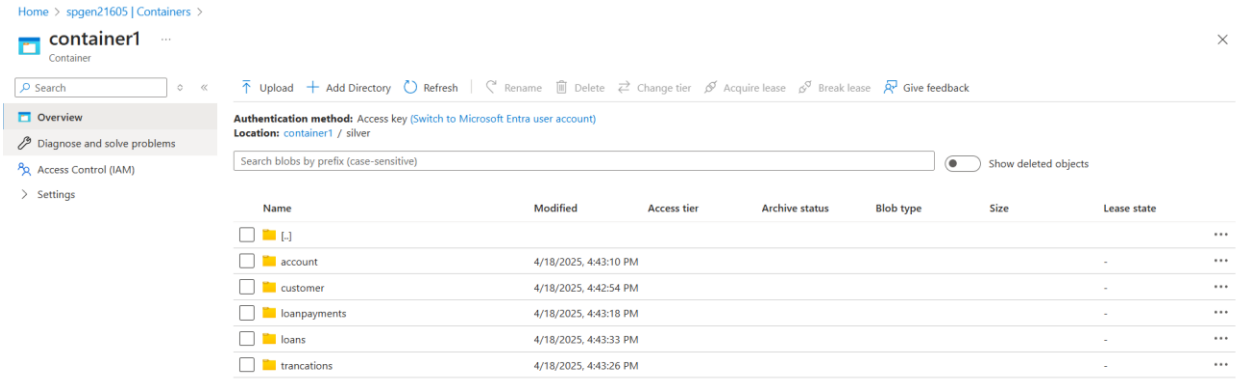
- Group by: transaction_id
- Aggregates:
- account_id = first(account_id)
- transaction_date = first(transaction_date)
- transaction_amount = first(clean_amount)

- transaction_type = first(clean_type)

5. Sink

- Path: silver/transactions/
- Format: Parquet or Delta

All cleaned and transformed data is stored in silver file in ADLS gen2.



❖ Gold Layer:- Upsert Logic with SCD Type 1 and 2 – Silver to Gold Layer

Objective: Load data into Azure SQL DB with slowly changing dimension logic.

Setup SQL Database

- Create SQL tables for each dataset

```
CREATE TABLE dbo.Account (
    AccountID INT PRIMARY KEY,
    CustomerID INT,
    AccountType VARCHAR(100),
    AccountBalance FLOAT,
    CreatedBy VARCHAR(100),
    CreatedDate DATETIME,
    UpdatedBy VARCHAR(100),
    UpdatedDate DATETIME,
    HashKey BIGINT
);

CREATE TABLE dbo.Customer (
```

110 %

Results | Messages

AccountID	CustomerID	AccountType	AccountBalance	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey

CustomerID	First_Name	Last_Name	Full_Address	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey	IsActive

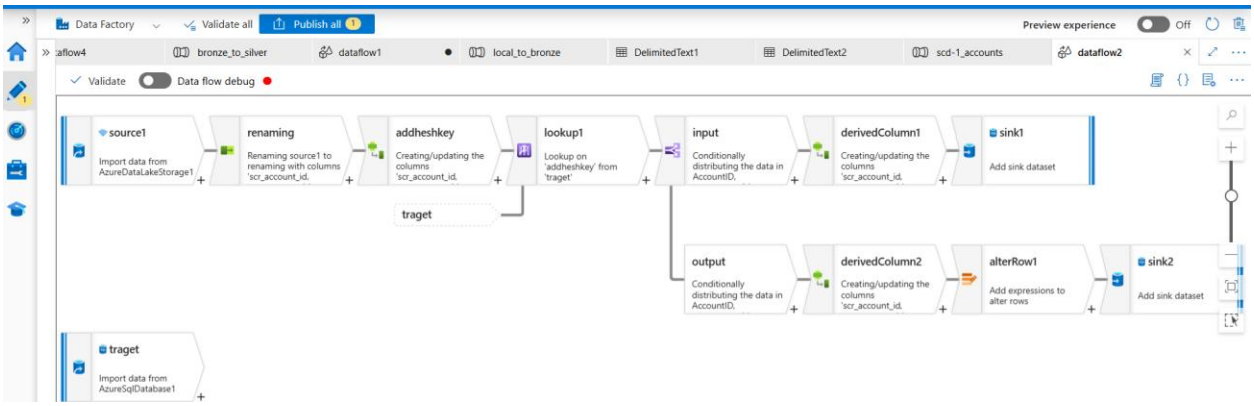
PaymentID	LoanID	PaymentDate	PaymentAmount	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey

LoanID	CustomerID	LoanAmount	InterestRate	LoanTerm	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey	IsActive

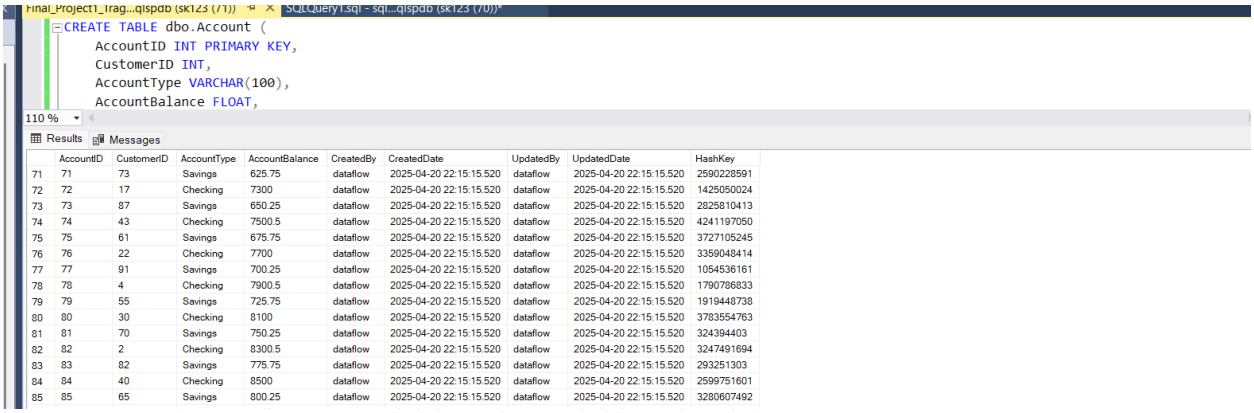
TransactionID	AccountID	TransactionDate	TransactionAmount	TransactionType	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey

SCD Type 1 (Overwrite Strategy)

- **Accounts**: Maintain only the latest account balance and type per account ID



Data inserted in target table :



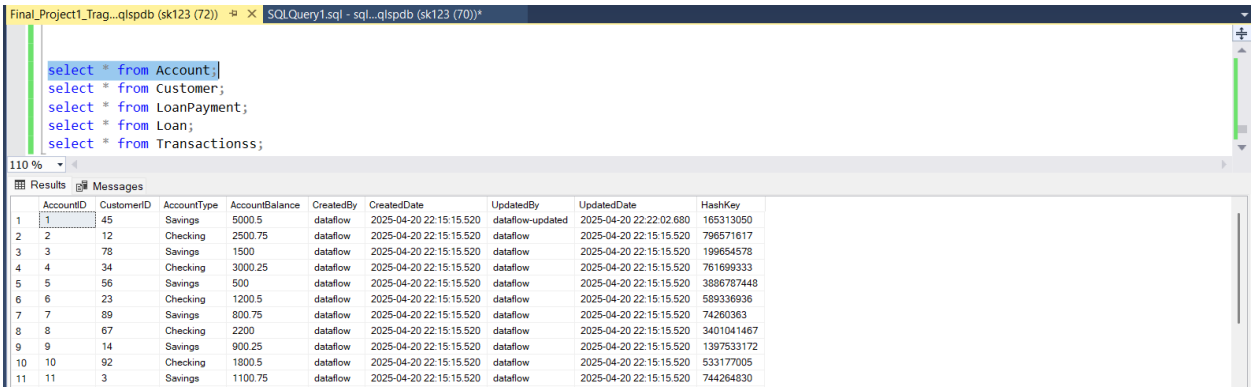
```
CREATE TABLE dbo.Account (
    AccountID INT PRIMARY KEY,
    CustomerID INT,
    AccountType VARCHAR(100),
    AccountBalance FLOAT,
```

110 %

Results | Messages

	AccountID	CustomerID	AccountType	AccountBalance	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey
71	71	73	Savings	625.75	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	2590228591
72	72	17	Checking	7300	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	1425050024
73	73	87	Savings	650.25	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	2825810413
74	74	43	Checking	7500.5	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	4241197050
75	75	61	Savings	675.75	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	3727105245
76	76	22	Checking	7700	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	3359048414
77	77	91	Savings	700.25	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	1054536161
78	78	4	Checking	7900.5	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	1790786833
79	79	55	Savings	725.75	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	1919448738
80	80	30	Checking	8100	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	3783554763
81	81	70	Savings	750.25	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	324394403
82	82	2	Checking	8300.5	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	3247491694
83	83	82	Savings	775.75	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	293251303
84	84	40	Checking	8500	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	2599751601
85	85	65	Savings	800.25	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	3280607492

After updates the target table also updated as per below mentioned



```
select * from Account;
select * from Customer;
select * from LoanPayment;
select * from Loan;
select * from Transactionss;
```

110 %

Results | Messages

	AccountID	CustomerID	AccountType	AccountBalance	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey
1	1	45	Savings	5000.5	dataflow	2025-04-20 22:15:15.520	dataflow-updated	2025-04-20 22:22:02.680	165313050
2	2	12	Checking	2500.75	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	796571617
3	3	78	Savings	1500	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	199654578
4	4	34	Checking	3000.25	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	761699333
5	5	56	Savings	500	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	3886787448
6	6	23	Checking	1200.5	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	589336936
7	7	89	Savings	800.75	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	74260363
8	8	67	Checking	2200	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	3401041467
9	9	14	Savings	900.25	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	1397533172
10	10	92	Checking	1800.5	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	533177005
11	11	3	Savings	1100.75	dataflow	2025-04-20 22:15:15.520	dataflow	2025-04-20 22:15:15.520	744264830

EXPLANATION OF EACH STEP:

1. source1 – Import Raw Data

- Reads data from the silver/accounts path (Delta or Parquet)
- Includes account_id, customer_id, account_type, account_balance, etc.

---

2. renaming – Rename Columns (Optional but Recommended)

- Likely renames columns like account_id → scr_account_id
- Helps avoid column name conflicts later during joins or merges

---

3. addhashkey

- used for compare hash of all values to detect row changes
- addhashkey :crc32(concat(column names))

---

4. target – Import Existing SQL Table

- Reads from the SQL table dbo.Account
- Should contain current state of accounts

---

5. lookup1 – Lookup SQL Table by Account ID

- Joins source1 and target records on account_id
- Brings in current record to compare with incoming record

---

6. input – Conditional Split (New vs Existing vs Unchanged)

- Probably splits data into:
    - **Insert (new account)** → goes to upper path
    - **Update (existing but changed)** → goes to lower path
    - **Unchanged** → may be filtered out

logic is:

isNull(target.account_id) → INSERT

source.account_id ==target.account.id and source.hashkey!=target.hashkey → UPDATE

---

7. derivedColumn1 – Prepares Data for Insert (NEW rows)

- add audit fields like CreatedDate , updatedby...

---

8. sink1 – SQL Sink for INSERTS

- Inserts new accounts

- Should **NOT** allow updates

Sink settings:

- Insert: yes
- Update: no
- Upsert: no

---

9. derivedColumn2 – Prepares Data for UPDATE

- Adds UpdatedDate = currentUTC()
- remove unchanged columns such as crediteddate...

---

10. alterRow1 – Define Upsert Rules

- Tells ADF which records should be updated

Expression should be:

Update: true() → (all records from update stream go here)

---

11. sink2 – SQL Sink for UPDATEs

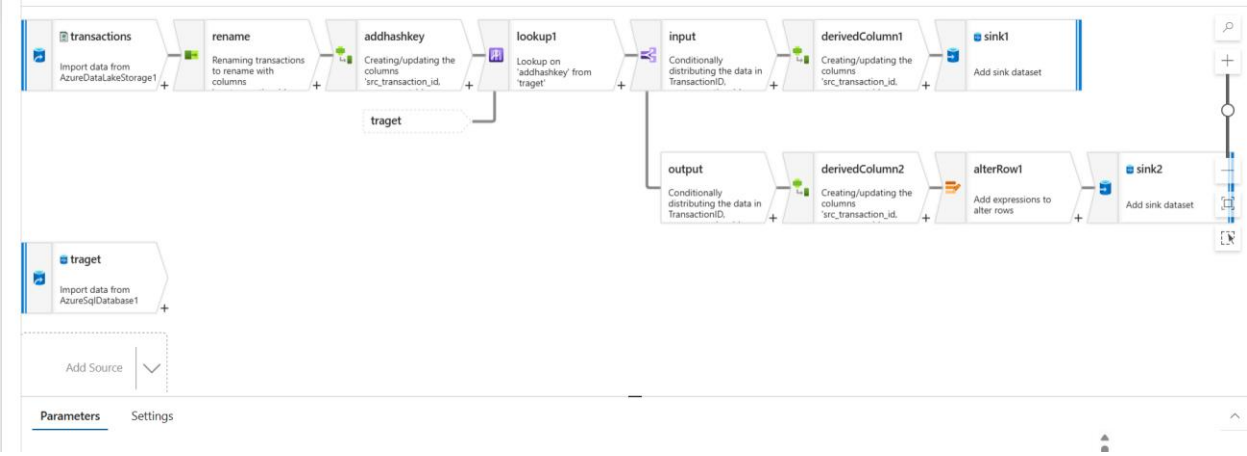
- This is your SCD-1 update target

Sink settings:

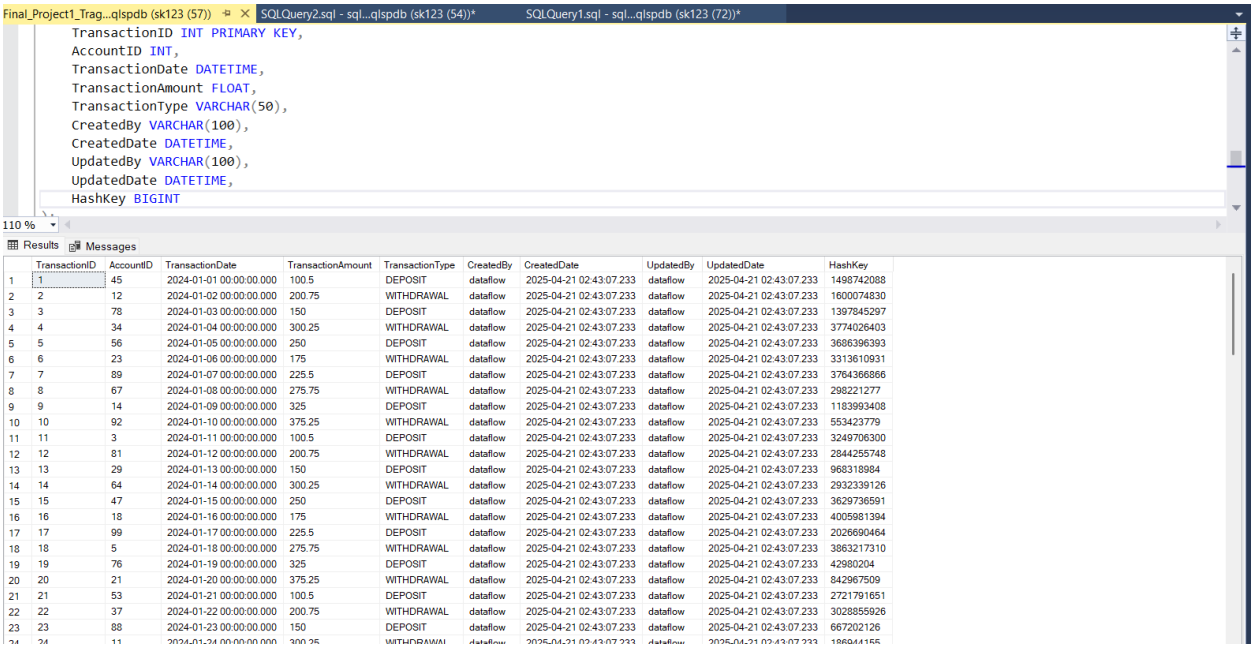
- Insert: no
- Update: yes
- Upsert: no
- Key Column: AccountID

Transactions SCD-1

- **Transactions**: Event log; updated by TransactionID when changes occur



Data inserted into targeted table



	TransactionID	AccountID	TransactionDate	TransactionAmount	TransactionType	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey
1	1	45	2024-01-01 00:00:00.000	100.5	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	1498742088
2	2	12	2024-01-02 00:00:00.000	200.75	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	1600074830
3	3	78	2024-01-03 00:00:00.000	150	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	1397845297
4	4	34	2024-01-04 00:00:00.000	300.25	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3774026403
5	5	56	2024-01-05 00:00:00.000	250	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3686396393
6	6	23	2024-01-06 00:00:00.000	175	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3313610931
7	7	89	2024-01-07 00:00:00.000	225.5	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3764366866
8	8	67	2024-01-08 00:00:00.000	275.75	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	298221277
9	9	14	2024-01-09 00:00:00.000	325	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	1183993408
10	10	92	2024-01-10 00:00:00.000	375.25	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	553423779
11	11	3	2024-01-11 00:00:00.000	100.5	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3249706300
12	12	81	2024-01-12 00:00:00.000	200.75	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	2844255748
13	13	29	2024-01-13 00:00:00.000	150	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	968318984
14	14	64	2024-01-14 00:00:00.000	300.25	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	2932339126
15	15	47	2024-01-15 00:00:00.000	250	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3629736591
16	16	18	2024-01-16 00:00:00.000	175	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	4005981394
17	17	99	2024-01-17 00:00:00.000	225.5	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	2026690464
18	18	5	2024-01-18 00:00:00.000	275.75	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3863217310
19	19	76	2024-01-19 00:00:00.000	325	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	42980204
20	20	21	2024-01-20 00:00:00.000	375.25	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	842967509
21	21	53	2024-01-21 00:00:00.000	100.5	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	2721791651
22	22	37	2024-01-22 00:00:00.000	200.75	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3028855926
23	23	88	2024-01-23 00:00:00.000	150	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	667202126

EXPLANATION OF EACH STEP:

Upper Flow (INSERT NEW Records)

1. **transactions (Source)**
    - Loads transaction data from **Azure Data Lake Storage**.
2. **rename**
    - Renames columns (probably to match schema with target table).
3. **addhashkey**
    - Adds a hash key (src_transaction_id) to track changes or for comparison purposes.
4. **lookup1**

- Performs a **lookup** against the **target SQL table** (traget) to see if each record already exists.

5. **input (Conditional Split)**
    - Filters records where **lookup doesn't find a match** (i.e., new transactions that are not yet in the target).
6. **derivedColumn1**
    - Updates or sets metadata like src_transaction_id for new records.
7. **sink1**
    - Inserts new records into the target table.

Lower Flow (UPDATE Existing Records)

1. **traget (Source)**
    - Loads existing records from **Azure SQL Database**.
2. **output (Conditional Split)**
    - Filters records where **lookup finds a match**, but data **has changed** (this is the update condition).
3. **derivedColumn2**
    - Prepares updated values or columns.
4. **alterRow1**
    - Specifies **which rows to update** (using UpdateIf condition).
5. **sink2**
    - Performs **update** operation on the target table.

After the update

```
    TransactionDate DATETIME,
    TransactionAmount FLOAT,
    TransactionType VARCHAR(50),
    CreatedBy VARCHAR(100),
    CreatedDate DATETIME,
    UpdatedBy VARCHAR(100),
    UpdatedDate DATETIME,
    HashKey BIGINT
```

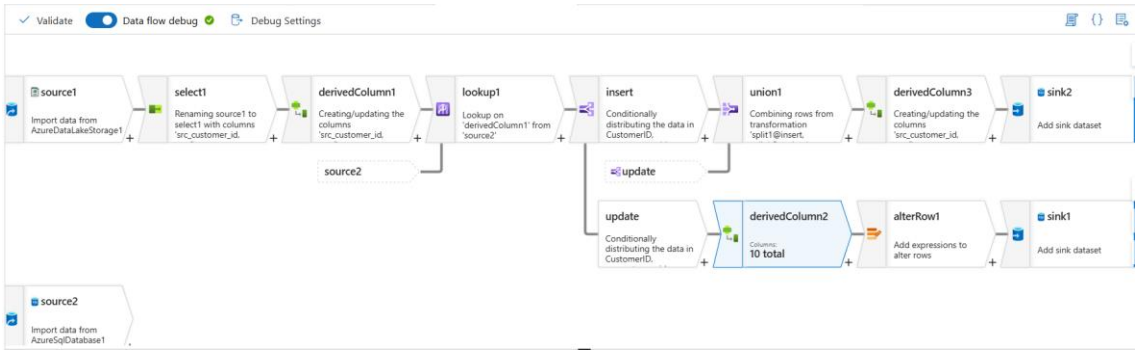
110 %

Results | Messages

	TransactionID	AccountID	TransactionDate	TransactionAmount	TransactionType	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey
80	80	30	2024-03-20 00:00:00.000	375.25	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3615416596
81	81	70	2024-03-21 00:00:00.000	100.5	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3391522849
82	82	2	2024-03-22 00:00:00.000	200.75	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	4199045516
83	83	82	2024-03-23 00:00:00.000	150	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	220750129
84	84	40	2024-03-24 00:00:00.000	300.25	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3904058926
85	85	65	2024-03-25 00:00:00.000	250	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	2768369402
86	86	21	2024-03-26 00:00:00.000	175	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	93555657
87	87	93	2024-03-27 00:00:00.000	225.5	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3692481788
88	88	1	2024-03-28 00:00:00.000	275.75	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	1788270771
89	89	54	2024-03-29 00:00:00.000	325	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	2144814873
90	90	38	2024-03-30 00:00:00.000	375.25	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3335414391
91	91	77	2024-03-31 00:00:00.000	100.5	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3331151726
92	92	44	2024-04-01 00:00:00.000	200.75	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	33991143
93	93	79	2024-04-02 00:00:00.000	150	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3195884329
94	94	39	2024-04-03 00:00:00.000	300.25	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3101055717
95	95	60	2024-04-04 00:00:00.000	250	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	405995127
96	96	48	2024-04-05 00:00:00.000	175	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3744672639
97	97	90	2024-04-06 00:00:00.000	225.5	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	2364519947
98	98	49	2024-04-07 00:00:00.000	275.75	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	520924924
99	99	80	2024-04-08 00:00:00.000	325	DEPOSIT	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	1673093766
100	100	50	2024-04-09 00:00:00.000	375.25	WITHDRAWAL	dataflow	2025-04-21 02:43:07.233	dataflow	2025-04-21 02:43:07.233	3783938867
101	101	21	2024-01-02 00:00:00.000	230.75	Withdrawal	dataflow	2025-04-21 03:01:37.447	dataflow	2025-04-21 03:01:37.447	182343008
102	102	34	2024-01-03 00:00:00.000	15	Deposit	dataflow	2025-04-21 03:01:37.447	dataflow	2025-04-21 03:01:37.447	3881605382

SCD Type 2 (Versioning Strategy)

- **Customers**: Insert new rows when address or personal info changes
    - Fields: IsActive, EffectiveStartDate, EffectiveEndDate



Final_Project1_Trag...qlspdb (sk123 (73))* | SQLQuery2.sql - sql...qlspdb (sk123 (54))* | SQLQuery1.sql - sql...qlspdb (sk123 (72))*

```
CREATE TABLE dbo.Customer (
    CustomerID INT,
    First_Name VARCHAR(255),
    Last_Name VARCHAR(255),
    Full_Address VARCHAR(255),
    CreatedBy VARCHAR(100),
    CreatedDate DATETIME,
    UpdatedBy VARCHAR(100),
    UpdatedDate DATETIME,
    HashKey BIGINT,
```

110 %

Results | Messages

	CustomerID	First_Name	Last_Name	Full_Address	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey	IsActive
1	1	John	Doe	123 Elm St, Toronto, ON M4B1B3	dataflow	2025-04-21 03:58:33.837	dataflow	2025-04-21 03:58:33.837	2254139456	1
2	2	Jane	Smith	456 Maple Ave, Ottawa, ON K1A0B1	dataflow	2025-04-21 03:58:33.837	dataflow	2025-04-21 03:58:33.837	2411602810	1
3	3	Michael	Johnson	789 Oak Dr, Montreal, QC H1A1A1	dataflow	2025-04-21 03:58:33.837	dataflow	2025-04-21 03:58:33.837	3021238274	1
4	4	Emily	Davis	101 Pine Rd, Calgary, AB T2A0A1	dataflow	2025-04-21 03:58:33.837	dataflow	2025-04-21 03:58:33.837	1730737493	1
5	5	David	Wilson	202 Birch Blvd, Vancouver, BC V5K0A1	dataflow	2025-04-21 03:58:33.837	dataflow	2025-04-21 03:58:33.837	1470512336	1
6	6	Emma	Clark	505 Cedar St, Halifax, NS B3H0A1	dataflow	2025-04-21 03:58:33.837	dataflow	2025-04-21 03:58:33.837	2471443894	1
7	7	James	Martinez	606 Spruce Ln, Winnipeg, MB R3C0A1	dataflow	2025-04-21 03:58:33.837	dataflow	2025-04-21 03:58:33.837	396866029	1
8	8	Olivia	Garcia	707 Fir St, Edmonton, AB T5A0A1	dataflow	2025-04-21 03:58:33.837	dataflow	2025-04-21 03:58:33.837	4015790471	1
9	9	William	Lopez	808 Redwood Dr, Victoria, BC V8W0A1	dataflow	2025-04-21 03:58:33.837	dataflow	2025-04-21 03:58:33.837	2484685838	1
10	10	Ava	Anderson	909 Cypress Ave, Quebec City, QC G1A0A1	dataflow	2025-04-21 03:58:33.837	dataflow	2025-04-21 03:58:33.837	2040526662	1

Explanation Of Each Steps

1.source1 – Source (Incoming Data from Silver Layer)

- Reads customer data (cleaned) from the Silver Layer (ADLS)
- Fields: customer_id, full_name, address, city, postal_code, etc.

2.select1 – Rename Columns

- Renames fields from customer_id → src_customer_id etc.

3.derivedColumn1 – add hashkey

- Adds a hash key (src_transaction_id) to track changes or for comparison purposes.

4.source2 – Lookup Source (Existing SQL Table)

- Reads data from target SQL table dbo.Customers
- Should only return active records:
- SELECT * FROM dbo.Customers WHERE IsActive = 1

5.lookup1 – Join Source1 to Source2

- Match records based on customer_id
- Retrieves the **current active record** from SQL for comparison

---

6.insert – Conditional Split for Inserts

Condition:

isNull(lookup1.customer_id)

If no match is found → insert this as a **new customer**

---

update – Conditional Split for Updates

Condition:

Src.customer_id != Target.customer_id || src.hashkey!=Target.hashkey

If values differ → mark as **update needed**
This is your **SCD Type 2 core logic**

7.derivedColumn2 – Deactivate Existing Record

Add fields like:

IsActive = 0

updateddate= currentUTC()

This closes the previous version of the customer record.

8.alterRow1 – Mark for Update

Expression:

iif(IsActive == 0, 'Update')

This tells ADF Sink to **update** the existing customer record to make it inactive.

9.sink1 – SQL Sink (Update Existing Record)

- Updates the **previous customer version**
- Set Sink settings:
    - Update: YES
    - Insert: NO
    - Upsert: Optional
    - Key: CustomerID (or composite key with HashKey)

---

10. union1 – Combine New & Updated Records

Combines:

- New inserts (insert path)
- New version of updated records (update path)

---

11. **derivedColumn3 – Set Insert Fields**

This sets:

- IsActive = 1

- createdDate = currentUTC()

12.sink2 – SQL Sink (Insert New Records)

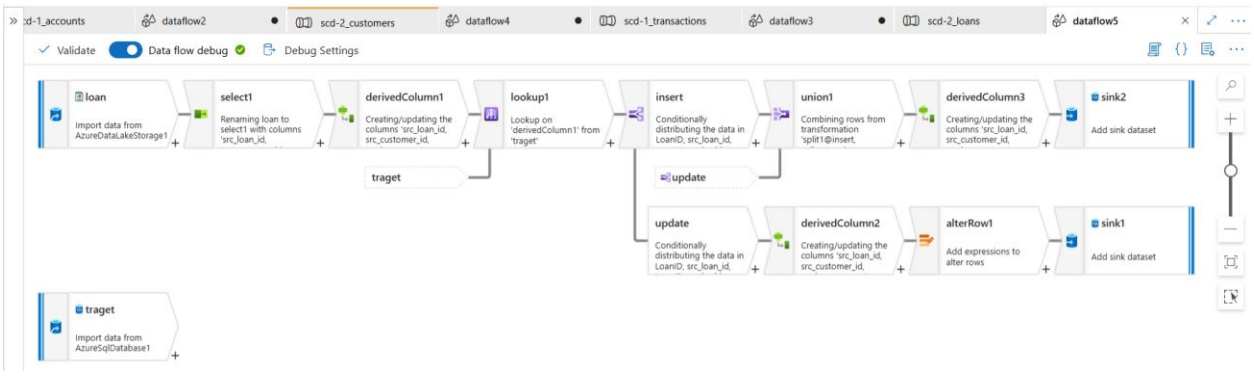
- Writes both:
    - Completely new records
    - New versions of changed records
- Sink settings:
    - Insert: YES
    - Update: NO
    - Key column: CustomerID

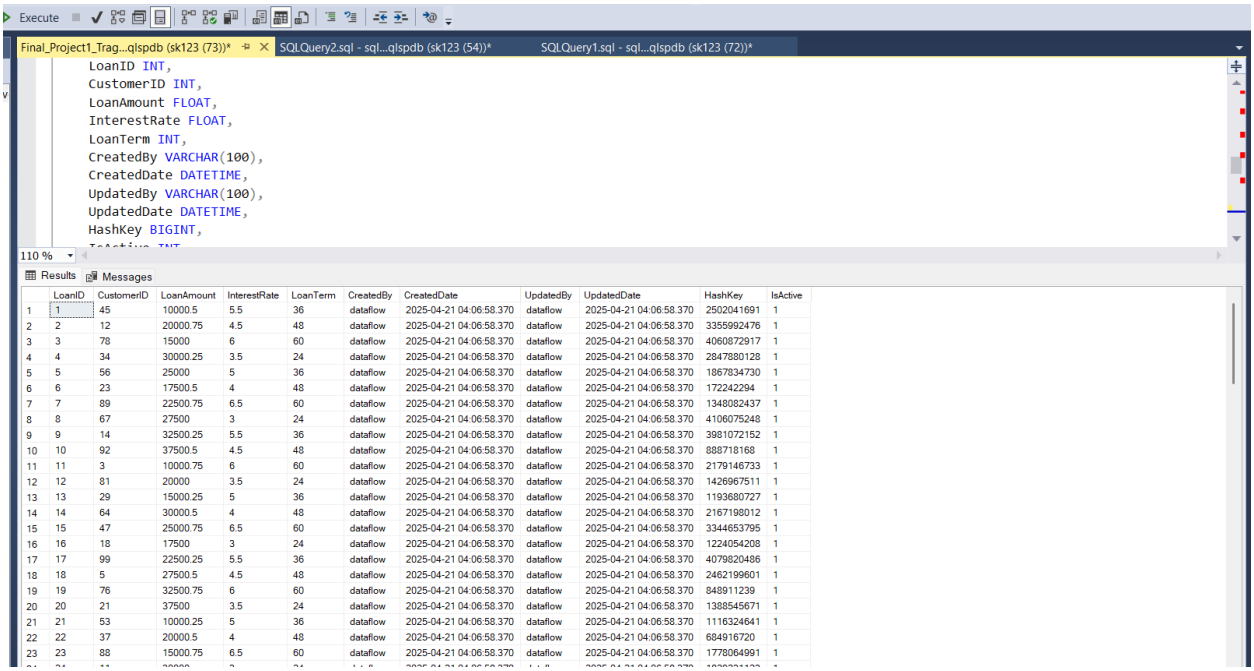
After updates the data , we can see that new data and old data also in table so we can keep old data as well.

Results | Messages

	CustomerID	First_Name	Full_Address	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey	IsActive	Last_Name
1	1	John	123 Elm St, Toronto, ON M4B1B3	dataflow	2025-04-20 03:13:30.443	dataflow-updated	2025-04-20 03:37:40.477	2254139456	0	Doe
2	1	John	304 ramsey view sudbury ON P3C2E6	dataflow	2025-04-20 03:36:57.850	dataflow	2025-04-20 03:36:57.850	2322987022	1	Doe
3	2	Jane	456 Maple Ave, Ottawa, ON K1A0B1	dataflow	2025-04-20 03:13:30.443	dataflow-updated	2025-04-20 03:37:40.477	2411602810	0	Smith
4	2	Jane	12 havens Ottawa ON K1A0A2	dataflow	2025-04-20 03:36:57.850	dataflow	2025-04-20 03:36:57.850	3301853113	1	Smith
5	3	Michael	789 Oak Dr, Montreal, QC H1A1A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	3021238274	1	Johnson
6	4	Emily	101 Pine Rd, Calgary, AB T2A0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	1730737493	1	Davis
7	5	David	202 Birch Blvd, Vancouver, BC V5K0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	1470512336	1	Wilson
8	6	Emma	505 Cedar St, Halifax, NS B3H0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	2471443894	1	Clark
9	7	James	606 Spruce Ln, Winnipeg, MB R3C0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	396866029	1	Martinez
10	8	Olivia	707 Fir St, Edmonton, AB T5A0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	4015790471	1	Garcia
11	9	William	808 Redwood Dr, Victoria, BC V8W0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	2484685838	1	Lopez
12	10	Ava	909 Cypress Ave, Quebec City, QC G1A0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	2040526662	1	Anderson
13	11	Alexander	1010 Willow Rd, St. John's, NL A1A0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	2918641721	1	Thomas
14	12	Isabella	1111 Poplar St, Fredericton, NB E3B0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	1422967276	1	Lee
15	13	Daniel	1212 Ash Blvd, Charlottetown, PE C1A0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	3230835177	1	Harris
16	14	Sophia	1313 Beech Dr, Yellowknife, NT X1A0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	157402343	1	Young
17	15	Matthew	1414 Cedar Ln, Whitehorse, YT Y1A0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	3072665139	1	King
18	16	Charlotte	1515 Elm St, Iqaluit, NU X0A0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	2376223294	1	Scott
19	17	Joseph	1616 Maple Ave, Regina, SK S4P0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	824856101	1	Green
20	18	Amelia	1717 Oak Dr, Saskatoon, SK S7K0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	2087862254	1	Adams
21	19	Christopher	1818 Pine Rd, Thunder Bay, ON P7A0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	4198701270	1	Baker
22	20	Mia	1919 Birch Blvd, London, ON N6A0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	2511601197	1	Nelson
23	21	Andrew	2020 Spruce Ln, Hamilton, ON L8P0A1	dataflow	2025-04-20 03:13:30.443	dataflow	2025-04-20 03:13:30.443	2491279256	1	Mitchell

- **Loans**: Maintain history when loan amount, rate, or terms change
    - Same SCD2 tracking structure

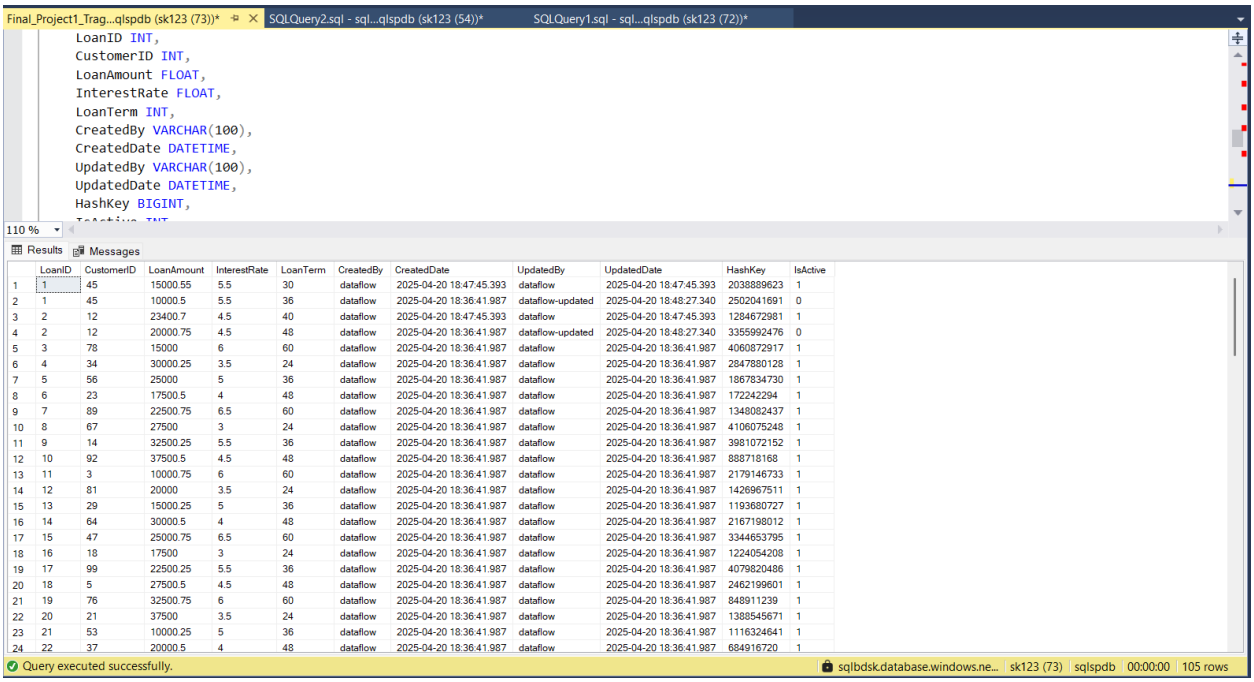




```
    LoanID INT,
    CustomerID INT,
    LoanAmount FLOAT,
    InterestRate FLOAT,
    LoanTerm INT,
    CreatedBy VARCHAR(100),
    CreatedDate DATETIME,
    UpdatedBy VARCHAR(100),
    UpdatedDate DATETIME,
    HashKey BIGINT,
```

	LoanID	CustomerID	LoanAmount	InterestRate	LoanTerm	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey	IsActive
1	1	45	10000.5	5.5	36	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	2502041691	1
2	2	12	20000.75	4.5	48	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	3355992476	1
3	3	78	15000	6	60	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	4060872917	1
4	4	34	30000.25	3.5	24	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	2847880128	1
5	5	56	25000	5	36	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	1867834730	1
6	6	23	17500.5	4	48	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	172242294	1
7	7	89	22500.75	6.5	60	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	1348082437	1
8	8	67	27500	3	24	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	4106075248	1
9	9	14	32500.25	5.5	36	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	3981072152	1
10	10	92	37500.5	4.5	48	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	888718168	1
11	11	3	10000.75	6	60	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	2179146733	1
12	12	81	20000	3.5	24	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	1426967511	1
13	13	29	15000.25	5	36	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	1193680727	1
14	14	64	30000.5	4	48	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	2167198012	1
15	15	47	25000.75	6.5	60	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	3344653795	1
16	16	18	17500	3	24	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	1224054208	1
17	17	99	22500.25	5.5	36	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	4079820486	1
18	18	5	27500.5	4.5	48	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	2462199601	1
19	19	76	32500.75	6	60	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	848911239	1
20	20	21	37500	3.5	24	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	1388545671	1
21	21	53	10000.25	5	36	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	1116324641	1
22	22	37	20000.5	4	48	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	684916720	1
23	23	88	15000.75	6.5	60	dataflow	2025-04-21 04:06:58.370	dataflow	2025-04-21 04:06:58.370	1778064991	1

After the updates the new data and old data also in same table to keep historical data we use SCD type-2.



```
    LoanID INT,
    CustomerID INT,
    LoanAmount FLOAT,
    InterestRate FLOAT,
    LoanTerm INT,
    CreatedBy VARCHAR(100),
    CreatedDate DATETIME,
    UpdatedBy VARCHAR(100),
    UpdatedDate DATETIME,
    HashKey BIGINT,
```

	LoanID	CustomerID	LoanAmount	InterestRate	LoanTerm	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey	IsActive
1	1	45	15000.55	5.5	30	dataflow	2025-04-20 18:47:45.393	dataflow	2025-04-20 18:47:45.393	2038889623	1
2	1	45	10000.5	5.5	36	dataflow	2025-04-20 18:36:41.987	dataflow-updated	2025-04-20 18:48:27.340	2502041691	0
3	2	12	23400.7	4.5	40	dataflow	2025-04-20 18:47:45.393	dataflow	2025-04-20 18:47:45.393	1284672981	1
4	2	12	20000.75	4.5	48	dataflow	2025-04-20 18:36:41.987	dataflow-updated	2025-04-20 18:48:27.340	3355992476	0
5	3	78	15000	6	60	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	4060872917	1
6	4	34	30000.25	3.5	24	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	2847880128	1
7	5	56	25000	5	36	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	1867834730	1
8	6	23	17500.5	4	48	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	172242294	1
9	7	89	22500.75	6.5	60	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	1348082437	1
10	8	67	27500	3	24	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	4106075248	1
11	9	14	32500.25	5.5	36	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	3981072152	1
12	10	92	37500.5	4.5	48	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	888718168	1
13	11	3	10000.75	6	60	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	2179146733	1
14	12	81	20000	3.5	24	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	1426967511	1
15	13	29	15000.25	5	36	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	1193680727	1
16	14	64	30000.5	4	48	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	2167198012	1
17	15	47	25000.75	6.5	60	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	3344653795	1
18	16	18	17500	3	24	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	1224054208	1
19	17	99	22500.25	5.5	36	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	4079820486	1
20	18	5	27500.5	4.5	48	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	2462199601	1
21	19	76	32500.75	6	60	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	848911239	1
22	20	21	37500	3.5	24	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	1388545671	1
23	21	53	10000.25	5	36	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	1116324641	1
24	22	37	20000.5	4	48	dataflow	2025-04-20 18:36:41.987	dataflow	2025-04-20 18:36:41.987	684916720	1

SCD Type 2 for Loans – Step-by-Step in ADF

1: Import Source Loan Data (Silver Layer)

- **Activity:** loan (Source)
- **Source:** ADLS Gen2 (e.g., Delta or Parquet)

- **Contains Fields:** LoanID, CustomerID, LoanAmount, LoanTerm, InterestRate, etc.

STEP 2: Rename Columns

- **Activity:** select1
- **Purpose:** Rename columns to avoid clashes (e.g., LoanID → src_loan_id)

3: Add Metadata Columns

- **Activity:** derivedColumn1
- **Add Columns: Addhashkey**

4: Load Existing Target Table (Active Records Only)

- **Activity:** target (Source2)
- **Query:**
- SELECT Loan_id , Hashkey FROM dbo.Loans WHERE IsActive = 1

5: Join Source to Target

- **Activity:** lookup1
- **Join Condition:** src_loan_id == target.LoanID
- **Purpose:** Match source record with active existing record

---

STEP 6: Identify New Inserts

- **Activity:** insert (Conditional Split)
- **Condition:**
- isNull(target.LoanID)
- **Meaning:** Loan is completely new → Insert

7: Identify Changes in Existing Loans

- **Activity:** update (Conditional Split)
- **Condition:** Source.Loan_id==Target.Source_id &&Source.Hashkey !=target.hashkey

- **Meaning:** Existing loan has changed → Update version

8: Close Old Version

- **Activity:** derivedColumn2
- **Set Fields:**
- IsActive = 0
- updateddate= currentUTC()

9: Mark for Update

- **Activity:** alterRow1
- **Expression:**
- iif(IsActive == 0, 'Update')

10: Update Existing Record in SQL (Deactivate)

- **Activity:** sink1
- **Sink Settings:**
    - Mode: **Update**
    - Key: LoanID

11: Combine New + New Versions

- **Activity:** union1
- **Combines:** Rows from insert and update paths

12: Finalize New Version

- **Activity:** derivedColumn3
- **Set Fields:**
- IsActive = 1
- CreatedDate = currentUTC()
- Updateddate = currentUTC()

13: Insert New Version into SQL Table

- **Activity:** sink2
- **Sink Settings:**
    - Mode: **Insert Only**
    - Key: LoanID

Master Pipeline:

❖ challenges faced in each layer of project:

Bronze Layer (Raw Ingestion)

- **Dynamic file name handling**: Required parameterized file ingestion for multiple CSVs
- **Schema inference issues**: Needed manual schema projection to prevent type mismatches
- **Storage path consistency**: Maintaining structured paths for each file source

---

Silver Layer (Transformations)

- **Data quality inconsistencies**: Had to filter nulls, invalid amounts, and standardize casing
- **Data type conversions**: String to Float/Int caused transformation failures if not cast properly
- **Deduplication logic**: Needed proper grouping and aggregation (e.g., first by ID)
- **Null-safe expressions**: Needed iif(isNull(...)) in derived columns and filters

---

Gold Layer (SCD Logic)

- **SCD Type 1 updates not triggering**: Caused by join mismatches or incorrect alter row expressions
- **SCD Type 2 versioning complexity**: Managing two parallel streams (insert + update)
- **Lookup join failures**: Due to data type mismatches or column naming conflicts
- **Sink configuration**: Forgetting to separate insert-only and update-only sinks correctly
- **Missing or inconsistent IsActive, EffectiveDate values** in logic

---