

Project Overview:

The goal of this project is to design and implement a robust data pipeline for processing customer account data. The pipeline focuses on the following key components:

1. Data Ingestion:

- Copy data from the backend team's Azure Storage Account to a centralized location.

2. Data Transformation:

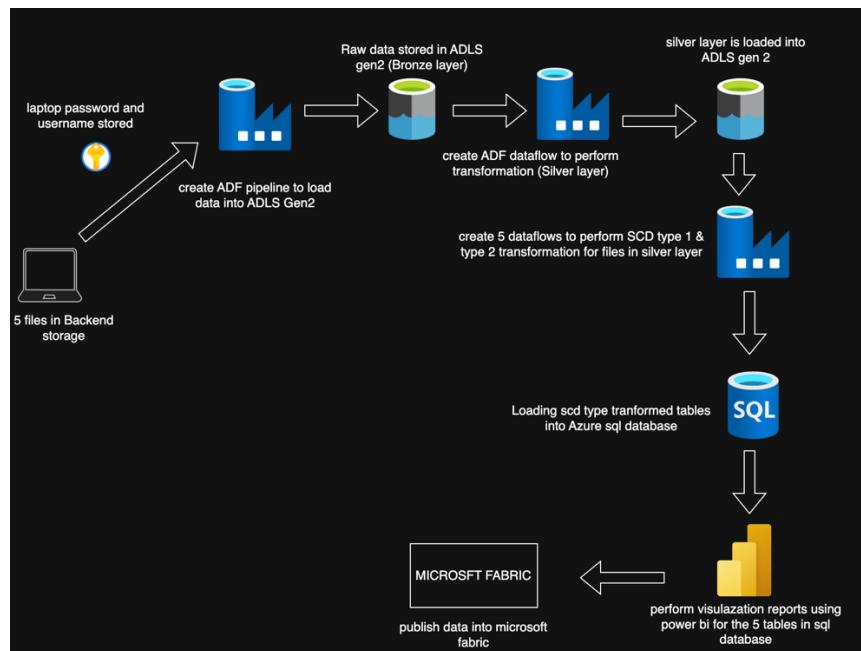
- Apply necessary data transformations using Azure Data Factory (ADF) to ensure data quality and consistency.

3. Data Upsertion:

- Upsert (insert or update) the transformed data from files stored in **Azure Data Lake Storage (ADLS)** into a **SQL database table**.

4. Objective:

- Ensure efficient, accurate, and scalable data processing to support downstream analytics and reporting requirements.

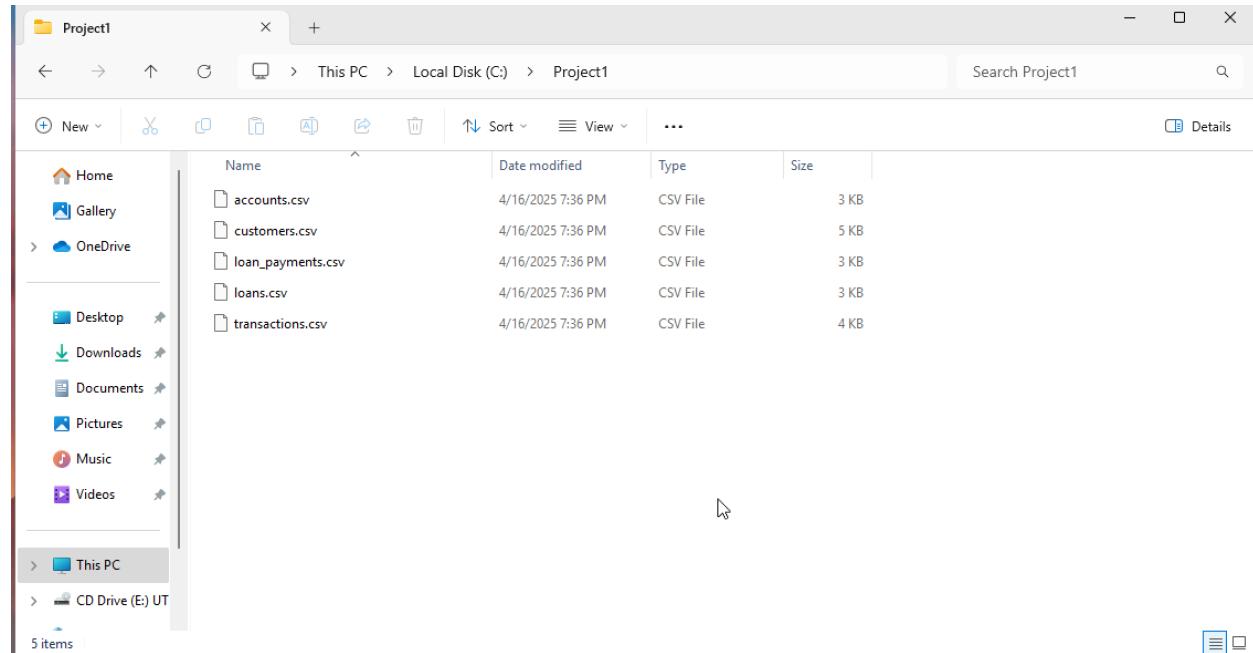


Architecture diagram for the project

Step_1_description:

Firstly we need data which was copied from the given link “

<https://www.kaggle.com/datasets/varunkumari/ai-bank-dataset>”. Now the files are saved in c drive in folder project 1.



Files downloaded from kaggle

A screenshot of the Microsoft Azure portal, specifically the Data Factory service. The left sidebar shows 'Data Factory', 'Connections', 'Integration runtimes', 'Microsoft Purview', 'Source control', 'Author', 'Triggers', 'Global parameters', 'Data flow libraries', 'Security', 'Credentials', 'Customer managed key', 'Outbound rules', 'Managed private endpoints', and 'Workflow orchestration'. The main area is titled 'Integration runtimes' and shows a table with one item: 'Name' (AutoResolveIntegrationRuntime), 'Type' (Azure), and 'Sub-type' (Public). To the right, there's a section titled 'Network environment' with a 'Azure' button and a 'Self-Hosted' button. Below that is a 'External Resources' section with a 'Linked Self-Hosted' button. At the bottom are 'Continue', 'Back', and 'Cancel' buttons.

SelfHosted Integration runtime for the local machine

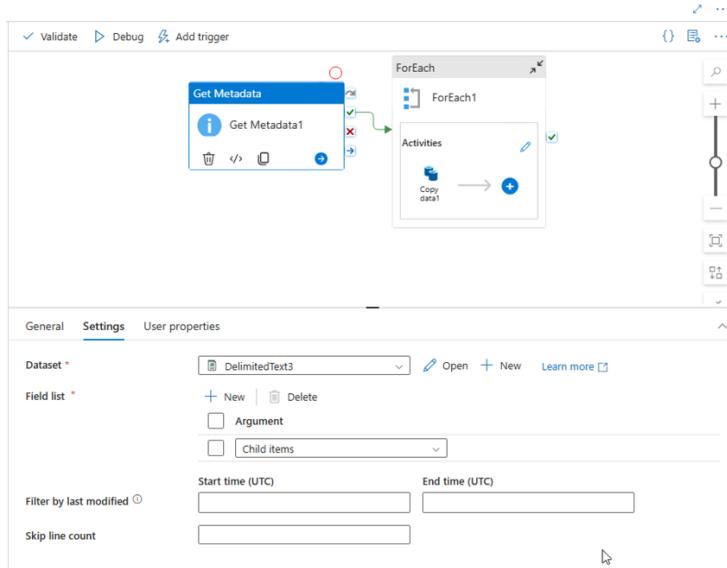
The screenshot shows the Azure Key Vault interface. At the top, there's a navigation bar with 'Home > Key vaults > harshakeyv'. Below it, the title 'harshakeyv | Secrets' is displayed with a 'Key vault' icon. A search bar and several action buttons ('Generate/Import', 'Refresh', 'Restore Backup', 'Manage deleted secrets', 'View sample code') are visible. On the left, a sidebar lists 'Overview', 'Activity log', 'Access control (IAM)', and 'Tags'. The main area is a table with columns 'Name', 'Type', 'Status', and 'Expiration date'. One row is shown: 'lappypassword' (Type: String), Status: 'Enabled', and no expiration date listed.

Secret key in keyvault for the laptop password

Using self-hosted IR and keyvault we have created a linked service to get the files from laptop to the azure account.

The screenshot shows the 'Edit linked service' dialog for a 'File system' linked service. The 'File system' tab is selected. The 'Name' field contains 'FileServer1'. The 'Description' field is empty. Under 'Connect via integration runtime', 'selfHostedIR' is selected. The 'Host' field contains 'C:\Project1'. The 'User name' field contains 'Harsha Vardhan'. The 'Password' tab is selected, while 'Azure Key Vault' is also present. In the 'AKV linked service' dropdown, 'AzureKeyVault1' is selected. The 'Secret name' dropdown contains 'lappypassword'. There is an 'Edit' checkbox next to it. The 'Secret version' dropdown contains '96c7c0eb3155425f8f9e1415f52f5397 (Current version)'. There is another 'Edit' checkbox next to it. At the bottom, there are 'Save' and 'Cancel' buttons, and a message 'Connection successful' with a checkmark icon.

Linked service to copy the backend data



I used metadata to get the data from my folder the child items will be the files in my local folder and will be passed to for each loop

Name	Value	Type
filename	@item().name	string

I set a parameter in the copy activity to give the file name. We have given @item().name to get the child items individual file names.

Connection Schema Parameters

Linked service * FileServer1 Test connection Edit + New Learn more

Integration runtime * selfHostedIR Edit

File path C:\Project1 / [Directory] / @dataset().filename Browse Preview data

Compression type No compression

Column delimiter Comma (,)

Row delimiter Default (\r\n, or \r\n)

Encoding Default(UTF-8)

Quote character Double quote ("")

Escape character Backslash (\)

Created parameters for the filename to be copied

Microsoft Azure | Data Factory > adofdharshas

Activities Validate all Publish all

Get Metadata1 ForEach1 Activities Copy data1

Parameters Variables Settings Output

Pipeline run ID: 441cd65d-b9e2-4f04-b086-2fd54a6b8e33 Pipeline status Succeeded View debug run consumption

Activity name	Activity state	Activity type	Run start	Duration	Integration runtime	User prop...	Activ
Copy data1	Succeeded	Copy data	4/20/2025, 9:42:38 PM	22s	selfHostedIR		174ci
Copy data1	Succeeded	Copy data	4/20/2025, 9:42:38 PM	15s	selfHostedIR		7842l
Copy data1	Succeeded	Copy data	4/20/2025, 9:42:38 PM	24s	selfHostedIR		3824i
Copy data1	Succeeded	Copy data	4/20/2025, 9:42:38 PM	25s	selfHostedIR		dbe6
Copy data1	Succeeded	Copy data	4/20/2025, 9:42:38 PM	25s	selfHostedIR		c904c
ForEach1	Succeeded	ForEach	4/20/2025, 9:42:38 PM	27s			a5ea'
Get Metadata1	Succeeded	Get Metadata	4/20/2025, 9:42:38 PM	14s	selfHostedIR		be33-

Pipeline to get the data from laptop to ADLS gen 2

Microsoft Azure Upgrade Search resources, services, and docs (G+)

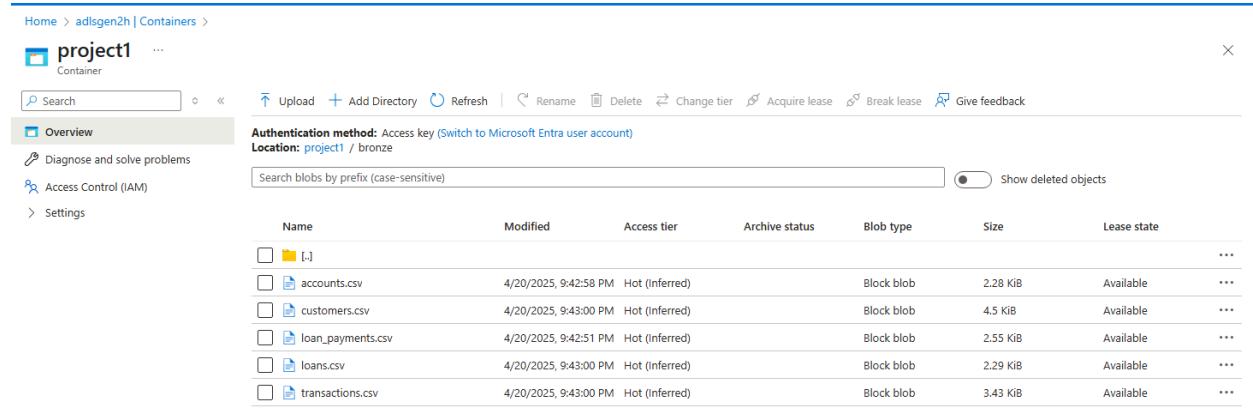
Home > adlsgen2h | Containers > project1 Container

Overview Authentication method: Access key Switch to Microsoft Entra user account Location: project1

Search blobs by prefix (case-sensitive) Show deleted objects

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
bronze	4/18/2025, 4:32:22 PM				-	***
silver	4/18/2025, 4:32:28 PM				-	***

Containers created inside the ADLS to store the files



The screenshot shows the Azure Storage Explorer interface. At the top, it displays 'Home > adlsgen2h | Containers >' and the container name 'project1'. Below the header are standard navigation and management buttons: Upload, Add Directory, Refresh, Rename, Delete, Change tier, Acquire lease, Break lease, and Give feedback. A message indicates the 'Authentication method: Access key (Switch to Microsoft Entra user account)' and the 'Location: project1 / bronze'. On the left, there's a sidebar with 'Overview', 'Diagnose and solve problems', 'Access Control (IAM)', and 'Settings'. The main area is titled 'Search blobs by prefix (case-sensitive)' and includes a checkbox for 'Show deleted objects'. A table lists the blobs in the container:

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
[...]						...
accounts.csv	4/20/2025, 9:42:58 PM	Hot (Inferred)		Block blob	2.28 KiB	Available
customers.csv	4/20/2025, 9:43:00 PM	Hot (Inferred)		Block blob	4.5 KiB	Available
loan_payments.csv	4/20/2025, 9:42:51 PM	Hot (Inferred)		Block blob	2.55 KiB	Available
loans.csv	4/20/2025, 9:43:00 PM	Hot (Inferred)		Block blob	2.29 KiB	Available
transactions.csv	4/20/2025, 9:43:00 PM	Hot (Inferred)		Block blob	3.43 KiB	Available

These are files loaded by successful run of the pipeline.

Challenges in Step 1:

- The system is not connecting while test connection because of backend permission we must give command to bypass the permissions. The command is

"Solution:

There is this 'dmgcmd.exe' at location "C:\Program Files\Microsoft Integration Runtime\5.0\Shared".

In powershell change directory to this location and run the following command:

```
\dmgcmd.exe -DisableLocalFolderPathValidation
```

- Initially connecting keyvault presented a challenge. It is failing to connect for the linked service. For this case we must go to the access policy and give us GET to read the secrets in keyvault.

Step_2_Description:

In step 2 we must use the data in the bronze layer, create a dataflow and remove null values duplicates etc. for further analysis.



Here is the dataflow design for step_2

The screenshot shows the Azure Data Factory pipeline run details for step_2. It includes the pipeline run ID (37a74090-960b-4e45-ac72-34bb07b408cf), pipeline status (Succeeded), and a table of activity logs. The table lists one activity, Data flow1, which succeeded with a duration of 1m 23s and was run on 4/19/2025, 7:03:21 PM. The table also includes columns for Activity name, Activity st..., Activit..., Run start, Duration, Integration runtime, User prop..., and Acti... (partially cut off).

Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime	User prop...	Acti...
Data flow1	Succeeded	Data flow	4/19/2025, 7:03:21 PM	1m 23s	AutoResolveIntegrationRuntime (East US)		54e

The pipeline run for step 2

Source settings Source options Projection Optimize Inspect Data preview

Output stream name * accounts [Learn more](#)

Description Import data from AzureDataLakeStorage1 [Reset](#)

Source type * Dataset Inline

Inline dataset type * DelimitedText

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

Skip line count

Sampling * Enable Disable

We are giving the source from our ADLS gen 2 from bronze container

Source settings **Source options** Projection Optimize Inspect Data preview

File settings

File mode File Wildcard

File path * project1 / bronze / accounts.csv [Browse](#)

Allow no files found

Change data capture

Compression type No compression

Encoding Default(UTF-8)

Column delimiter Comma (,)

Row delimiter Default (\r,\n, or \v\n)

Quote character Double quote ("")

Escape character Backslash (\)

Selecting the accounts.csv file from bronze layer.

Source settings Source options **Projection** Optimize Inspect Data preview

[Import schema](#) [Clear schema](#) [Schema options](#)

Column name	Type	Format
account_id	12s short	Specify format
customer_id	12s short	Specify format
account_type	abc string	Specify format
balance	1.2 double	Specify format

Then import schema for the file

Filter settings Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description

Incoming stream *

Filter on *

Now use filter option to check main field like account_id & customer_id if they are null we remove the entire row

Derived column's settings Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description

Incoming stream *

[Add](#) [Clone](#) [Delete](#) [Open expression builder](#)

Columns [①](#)

Column	Expression
<input type="checkbox"/> account_type	<input type="text" value="iif(isNull(account_type), 'NA', account_type)"/> abc + ✖

Then we give some values to the missing values in the row we are using derived column function for that with expression “`iif(isNull(account_type), 'NA', account_type)`”.

The screenshot shows the Stream Analytics window settings interface. At the top, there are tabs: Window settings (selected), Optimize, Inspect, and Data preview. Below these are input fields for Output stream name (removeDuplicate1), Description (Aggregates data based on a window and joins with original data), and Incoming stream (derivedColumn1). A navigation bar at the bottom includes tabs 1. Over, 2. Sort, 3. Range by, and 4. Window columns. The 1. Over tab is currently active. The main area displays a table titled 'derivedColumn1's column' with four rows: account_id, customer_id, account_type, and balance. To the right of each column is a 'Name as' field and a '+' button to add more columns.

For removing duplicate values, I am using window function. In window there are 3 main field to fill

- Over: over is used as order by in SQL. We give every column of the table here.
- Sort: we also give every column of the table here as well to sort.
- Windows column: we give the expression here. I am using the expression rownumber () to count all repeated values here.

The screenshot shows the Stream Analytics window settings interface with the 'Sort' tab selected. The interface is identical to the one above, but the 'Sort' tab is highlighted. The main area displays a table titled 'derivedColumn1's column' with four rows: account_id, customer_id, account_type, and balance. To the right of each column is an 'Order' dropdown set to 'Ascending' and a checkmark indicating it is sorted. There is also a 'Nulls first' checkbox and a '+' button to add more columns.

Sort in window function

Window settings Optimize Inspect Data preview

Output stream name * ? Help Learn more [🔗](#)

Description ⏪ Reset

Incoming stream * ▾

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

+ Add Clone Delete [Open expression builder](#)

Column	Expression
<input type="checkbox"/>	<input type="text" value="rowcol"/> ▾ <input type="text" value="rowNumber()"/> 123 + ↗ Delete ↗

Window column expression of window function

Filter settings Optimize Inspect Data preview

Output stream name * Learn more [🔗](#)

Description ⏪ Reset

Incoming stream * ▾

Filter on * [X](#)

Now to remove the duplicate values we use filtering function if select rowcol==1 and only let the fields that pass-through filter.

Select settings Optimize Inspect Data preview

Output stream name * removingcol1 [Learn more](#)

Description Renaming filter1 to removingcol1 with columns 'account_id, customer_id, account_type, balance'

Incoming stream * filter1

Options

- Skip duplicate input columns
- Skip duplicate output columns

Input columns *

<input type="checkbox"/>	filter1's column	▼	Name as
<input type="checkbox"/>	12s account_id	▼	account_id
<input type="checkbox"/>	12s customer_id	▼	customer_id
<input type="checkbox"/>	abc account_type	▼	account_type
<input type="checkbox"/>	12 balance	▼	balance

4 mappings: 1 column(s) fr

[Auto mapping](#) [Reset](#) [+ Add mapping](#) [Delete](#)

We use select function and remove rowcol and let only original rows of the table to pass

Alter row settings Optimize Inspect Data preview

Output stream name * alterRow1 [Learn more](#)

Description Add expressions to alter rows

Incoming stream * removingcol1

Alter row conditions * [①](#)

 Upsert if	1==1	+	-
----------------------------	------	-------------------	-------------------

We give alter row function and give condition upsert to create delta format file.

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

Folder path * project1 / concat('silver','accounts') [abc](#) [Browse](#)

Compression type No compression

Vacuum 0

Table action None Overwrite Truncate

Update method [①](#)

- Allow insert
- Allow delete
- Allow upsert
- Allow update

We give the path to set the file to be saved.

Name	Modified	Access tier	Archive status	Blob type	Size	Lease state
[...]						...
_delta_log	4/19/2025, 7:03:48 PM					...
_temporary	4/20/2025, 7:21:23 PM					...
part-00000-27d76ef7-0127-4194-a13...	4/19/2025, 7:03:49 PM	Hot (Inferred)		Block blob	2.59 KiB	Available
part-00000-3fb4761b-c966-4119-978...	4/20/2025, 7:21:26 PM	Hot (Inferred)		Block blob	601 B	Available
part-00000-4447bcfe-48d4-44cd-a6a...	4/20/2025, 7:30:36 PM	Hot (Inferred)		Block blob	601 B	Available
part-00000-6198b761-3615-471b-89...	4/20/2025, 7:30:36 PM	Hot (Inferred)		Block blob	2.13 KiB	Available
part-00000-7d900ba7-b057-4789-9c...	4/20/2025, 7:21:27 PM	Hot (Inferred)		Block blob	2.13 KiB	Available

Here is the saved file

The delta format files created inside each separate file with their names.

The design goes like this :

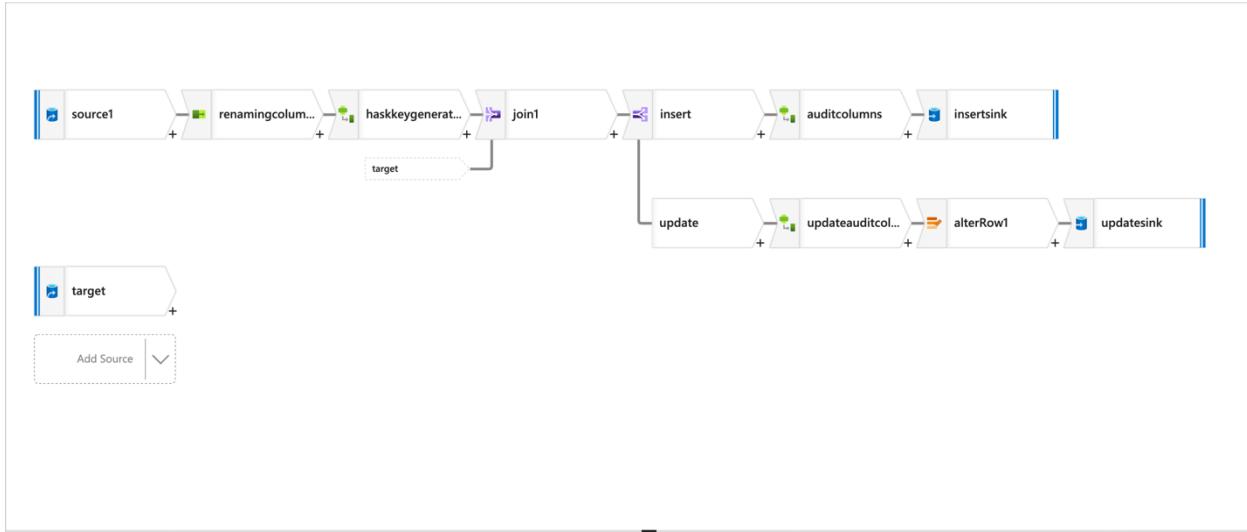
- Removing rows if primary key field are null
- Derived column for placing missing values
- Window function to check number of duplicate values and assign row count
- Filtering row count>1
- Removing row count column
- Checking necessary condition to create delta file format in alter row Upsert.
- Sink: Placing the file into destination as a delta format file.

Challenges in Step_2:

- Firstly we needed to figure out what data inside the files is more important. If they are null removed the entire columns. For ex: In accounts.csv file if account_id is null we have remove the full row. But we should not remove if amount field is null simply put it as "0".
- Mapping issue: after finishing the pipeline the mapping is changing for some reason and have to check each time before publishing it.

Step_3_Description:

Now we have to take the delta format files as source and perform SCD type 1 transformation and SCD type 2 transformation and place them in Azure sql database.



SCD type 1 transformation

Source settings	Source options	Projection	Optimize	Inspect	Data preview
Output stream name * source1 Learn more ↗	Description Import data from AzureDataLakeStorage1 Reset				
Source type * <input checked="" type="radio"/> Dataset <input type="radio"/> Inline					
Inline dataset type * Δ Delta					
Linked service * AzureDataLakeStorage1 Test connection ↗ Edit + New					
Sampling * ⓘ <input type="radio"/> Enable <input checked="" type="radio"/> Disable					

The source details are here we are selecting the delta file format from ADLS gen 2

Source settings **Source options** Projection Optimize Inspect Data preview

Folder path * / [Browse](#)

Allow no files found

Compression type

Time travel * Disable Query by timestamp Query by version

Setting the folder path

Select settings Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description

Incoming stream *

Options Skip duplicate input columns [①](#) Skip duplicate output columns [①](#)

Input columns * Auto mapping [①](#) [Reset](#) [+ Add mapping](#) [Delete](#) 1 mappings: All inputs mapped

	source1's column	Name as
<input type="checkbox"/>	1==1	concat("src_",\$\$)

Renaming the source columns to differentiate them from the table. We are giving the function concat("src_",\$\$). \$\$ gets the columns of the table.

Derived column's settings Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description

Incoming stream *

[+ Add](#) [Clone](#) [Delete](#) [Open expression builder](#)

Columns * [①](#)

<input type="checkbox"/> Column	Expression
<input type="checkbox"/> src_hashkey	crc32(concat(toString(src_account_id),toString(sr... 121))

Here we are giving crc32 has function by giving all the columns and converting non-string datatypes to string.

Source settings Source options Projection Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description [Reset](#)

Source type * Dataset Inline

Inline dataset type *

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

Sampling * Enable Disable

Here is the target file from the table already created in sql database.

Source settings **Source options** Projection Optimize Inspect Data preview

Input Table Query Stored procedure

Query * [①](#) [Edit](#)

Incremental column [①](#)

Isolation level [①](#)

We give this query to select the account_id & haskey from the table.

Join settings Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description

Left stream *

Right stream *

Join type * Full outer Inner Left outer Right outer Custom (cross)

Use fuzzy matching

Join conditions *

[+](#) [Edit](#)

We are using left join to join the source and target

Conditional split settings Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description

Incoming stream *

Split on First matching condition All matching conditions

Split condition

Stream names	Condition
<input type="text" value="insert"/>	<input type="text" value="isNull(account_id)"/> +
<input type="text" value="update"/>	<input type="text" value="src_account_id==account_id && src_hashkey!=hashkey"/> +

We are giving the conditions to set the insert and update functions

For insert we are giving : isnull(account_id)

For update we are giving: src_account_id==account_id && src_hashkey!=hashkey

Derived column's settings Optimize Inspect Data preview

Output stream name * auditcolumns Learn more

Description Creating/updating the columns
'src_account_id, src_customer_id,
src_account_type, src_balance,'

Incoming stream * split1@insert

+ Add Clone Delete Open expression builder

Columns *

<input type="checkbox"/> Column	Expression
<input type="checkbox"/> src_created_by	"dataflow"
<input type="checkbox"/> src_created_date	currentTimestamp()
<input type="checkbox"/> src_updated_by	"dataflow"
<input type="checkbox"/> src_updated_date	currentTimestamp()

Now we are seeing the derived columns functions on insert side

Derived column's settings Optimize Inspect Data preview

Output stream name * updateauditcolumns Learn more

Description Creating/updating the columns
'src_account_id, src_customer_id,
src_account_type, src_balance,'

Incoming stream * split1@update

+ Add Clone Delete Open expression builder

Columns *

<input type="checkbox"/> Column	Expression
<input type="checkbox"/> src_updated_by	"dataflow-updated"
<input type="checkbox"/> src_updated_date	currentTimestamp()

This is derived columns on the update side

Sink Settings Errors Mapping Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description [Reset](#)

Incoming stream *

Sink type *  Dataset  **Inline**  Cache

Inline dataset type *  Azure SQL Database

Linked service *  AzureSQLDatabase1 [Test connection](#) [Edit](#) [New](#)

Options

- Allow schema drift ①
- Validate schema ①

Here is the sink settings we are going to save the scd type 1 transformed data to sql database.

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

Schema name * [Refresh](#)

Table name * [Refresh](#)

Table action None Recreate table Truncate table

Update method ①

- Allow insert
- Allow delete
- Allow upsert
- Allow update

Use tempdb

Interim table schema [Refresh](#)

Pre SQL scripts ①

- List of scripts Custom expression ①

Post SQL scripts ①

- List of scripts Custom expression ①

Select the dbo schema and select the table required and click on allow insert for insertion

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

Skip duplicate input columns [?](#)

Skip duplicate output columns [?](#)

Auto mapping [?](#) [+](#) Add mapping [Delete](#) [Reset](#) [← Import schema](#) [View schema](#) 9 mappings: All outputs mapped

Input columns	Output columns
src_account_id	account_id
src_customer_id	customer_id
src_account_type	account_type
src_balance	balance
src_hashkey	hashkey
src_created_by	created_by
src_created_date	created_date
src_updated_by	updated_by
src_updated_date	updated_date

And then set the mapping for the sql table to allow data to be inserted properly

Alter row settings Optimize Inspect Data preview

Output stream name * alterRow1 [Learn more](#) [?](#)

Description Add expressions to alter rows [Reset](#)

Incoming stream * updateauditcolumns

Alter row conditions * [?](#)

* Update if [1==1](#) [+](#) [Delete](#)

Update condition for updates in scd type 1

Sink Settings Errors Mapping Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description [Reset](#)

Incoming stream *

Sink type *  Dataset  Inline  Cache

Inline dataset type *  Azure SQL Database

Linked service *  AzureSqlDatabase1 [Test connection](#) [Edit](#) [New](#)

Options

- Allow schema drift ⓘ
- Validate schema ⓘ

N

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

Skip writing key columns ⓘ

Key columns * ⓘ List of columns Custom expression ⓘ

[+](#) [!\[\]\(6ccbcca048ccf38c2ac7a58dfa8a21f9_img.jpg\)](#)

...

Here is the update side sink details

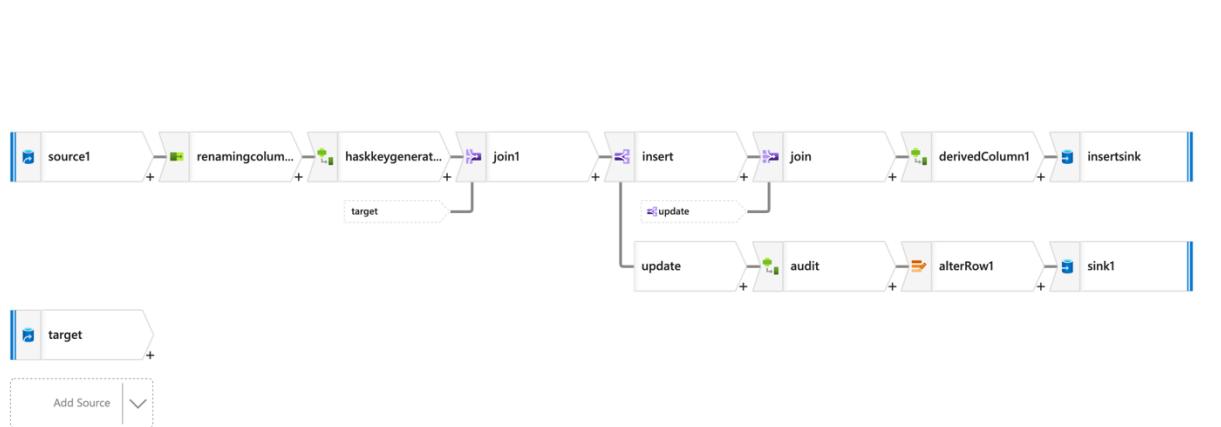
Sink Settings Errors **Mapping** Optimize Inspect Data preview

Options Skip duplicate input columns Skip duplicate output columns

Auto mapping Add mapping Delete Reset Import schema View schema 7 mappings: 2 column(s) from the output schema left unmapped

Input columns		Output columns	
12s src_account_id	▶	12s account_id	▶
12s src_customer_id	▶	12s customer_id	▶
abc src_account_type	▶	abc account_type	▶
1.2 src_balance	▶	1.2 balance	▶
12l src_hashkey	▶	12l hashkey	▶
abc src_updated_by	▶	abc updated_by	▶
⌚ src_updated_date	▶	⌚ updated_date	▶

Update side mapping



SCD type 2 transformation

Firstly we need to import the schema from source and then create tables in sql database table. Which will then be added into the target to perform SCD transformations.

Source settings **Source options** Projection Optimize Inspect Data preview

Input Table Query Stored procedure

Query * ⓘ

```
SELECT customer_id, hashkey, isActive
FROM dbo.customersscdtype2
WHERE isActive = 1
```

Incremental column ⓘ

Isolation level ⓘ

Read uncommitted

Here is the target side source setting we select the customer_id,hashkey, isActive where isActive ==1. We are doing this because we are only editing data which is active. We don't care about unactive data

Union settings Optimize Inspect Data preview

Output stream name *

[Learn more](#)

Description

Combining rows from transformation
'split1@insert, split1@update'

[Reset](#)

Incoming stream *

split1@insert

Union by * ⓘ

Name Position

Union with *

split1@update

The above scd type 1 transformation up to conditional split is same. But after conditional split in scd type transformation we use derived columns to set the created_by updated_by and all. But here first we perform union between insert and update and then proceed to derived columns.

Derived column's settings Optimize Inspect Data preview

Output stream name * audit [Learn more](#)

Description Creating/updating the columns
'src_customer_id, src_first_name,
src_last_name, src_address, src_city,'

Incoming stream * split1@update

[+ Add](#) [Clone](#) [Delete](#) [Open expression builder](#)

Columns * [①](#)

<input type="checkbox"/> Column	Expression	+	Delete
<input type="checkbox"/> src_updated_by	"dataflow-updated"	abc	+
<input type="checkbox"/> src_updated_date	currentTimestamp()	⌚	+
<input type="checkbox"/> src_isActive	0	123	+



Here is the update side derived column

Derived column's settings Optimize Inspect Data preview

Output stream name * derivedColumn1 [Learn more](#)

Description Creating/updating the columns
'src_customer_id, src_first_name,
src_last_name, src_address, src_city,'

Incoming stream * join

[+ Add](#) [Clone](#) [Delete](#) [Open expression builder](#)

Columns * [①](#)

<input type="checkbox"/> Column	Expression	+	Delete
<input type="checkbox"/> src_createdby	"dataflow"	abc	+
<input type="checkbox"/> src_createddate	currentTimestamp()	⌚	+
<input type="checkbox"/> src_updatedby	"dataflow"	abc	+
<input type="checkbox"/> src_updateddate	currentTimestamp()	⌚	+
<input type="checkbox"/> src-isactive	1	123	+



This is the insert side derived columns currenttimestamp() gives todays date and time at the time of insertion

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

Skip duplicate input columns Skip duplicate output columns

Auto mapping 13 mappings: All outputs mapped

Input columns		Output columns	
12s src_customer_id	→	12s customer_id	<input type="button" value="Delete"/> +
abc src_first_name	→	abc first_name	<input type="button" value="Delete"/> +
abc src_last_name	→	abc last_name	<input type="button" value="Delete"/> +
abc src_address	→	abc address	<input type="button" value="Delete"/> +
abc src_city	→	abc city	<input type="button" value="Delete"/> +
abc src_state	→	abc state	<input type="button" value="Delete"/> +
abc src_zip	→	abc zip	<input type="button" value="Delete"/> +
12l src_hashkey	→	12l hashkey	<input type="button" value="Delete"/> +
abc src_createdby	→	abc created_by	<input type="button" value="Delete"/> +
src_createddate	→	src_created_date	<input type="button" value="Delete"/> +
abc src_updatedby	→	abc updated_by	<input type="button" value="Delete"/> +
src_updateddate	→	src_updated_date	<input type="button" value="Delete"/> +
123 src-isactive	→	123 isActive	<input type="button" value="Delete"/> +

This is the mapping on the insert side of the pipeline

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

Schema name *

Table name *

Table action None Recreate table Truncate table

Update method Allow insert Allow delete Allow upsert Allow update

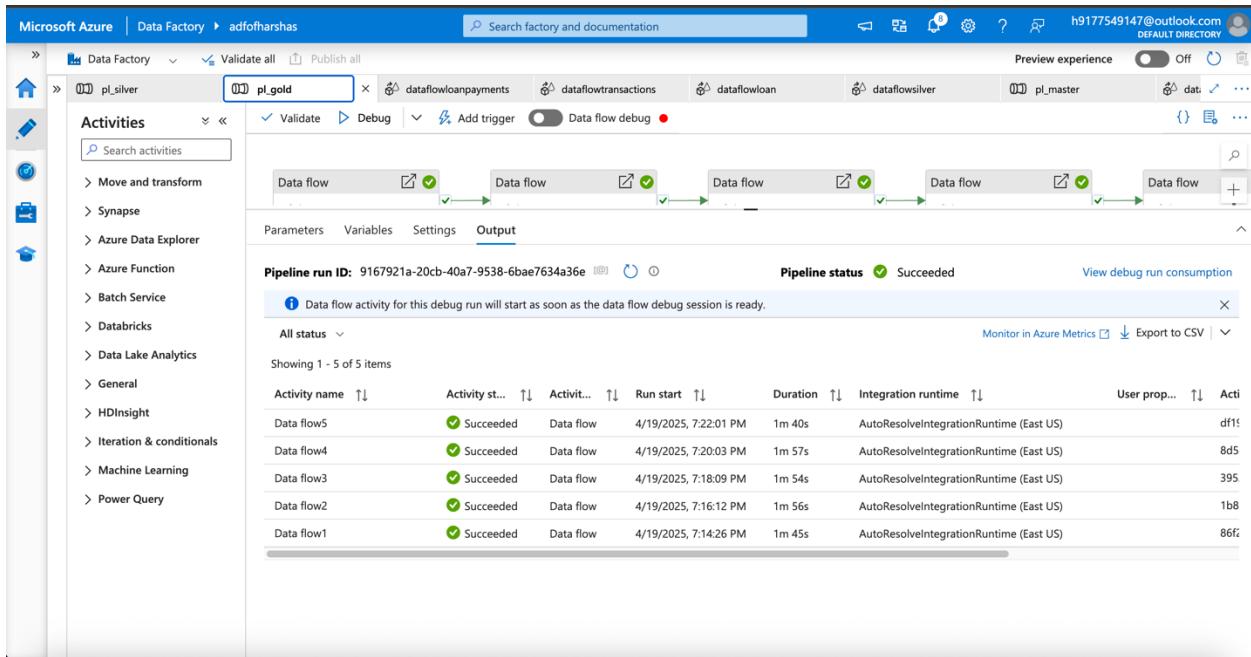
We are allowing the insert function for the insert side of the pipeline

The screenshot shows the 'Mapping' tab of a data integration interface. At the top, there are several options: 'Auto mapping' (unchecked), 'Add mapping' (+), 'Delete' (-), 'Reset' (refresh), 'Import schema' (import icon), and 'View schema' (view icon). A status message indicates '5 mappings: 8 column(s) from the output schema left unmapped'. Below this, there are two columns: 'Input columns' and 'Output columns'. The 'Input columns' list contains five items: 'src_customer_id', 'src_hashkey', 'src_updated_by', 'src_updated_date', and 'src_isActive'. The 'Output columns' list also contains five items: 'customer_id', 'hashkey', 'updated_by', 'updated_date', and 'isInactive'. Each input item has a dropdown arrow pointing to its corresponding output item.

We only update haskey updated_by updated_date and isactive fields only in the update side

The screenshot shows the 'Settings' tab of a data integration interface. It includes fields for 'Schema name' (set to 'dbo'), 'Table name' (set to 'customersscdtype2'), and 'Table action' (set to 'None'). Under 'Update method', the 'Allow update' checkbox is checked. In the 'Skip writing key columns' section, the checkbox is unchecked. The 'Key columns' section is set to 'List of columns' and lists 'customer_id' and 'hashkey' as key columns.

We are allowing update and are giving customer_id and hashkey for the key columns



Pipelines successfully run

Run Cancel query Save query Export data as Show only Editor

```
1  SELECT TOP (1000) * FROM [dbo].[transactionscdtype2]
```

Results Messages

Search to filter items...

transaction_id	account_id	transaction_date	transaction_amount	transaction_type	hash
65	69	2024-03-05T00:00:00.0000000	250.00	Deposit	3733E
53	86	2024-02-22T00:00:00.0000000	150.00	Deposit	3828E
85	65	2024-03-25T00:00:00.0000000	250.00	Deposit	1442E
31	71	2024-01-31T00:00:00.0000000	100.50	Deposit	4779E
78	4	2024-03-18T00:00:00.0000000	275.75	Withdrawal	2544E

SCD type 2 data stored successfully in SQL database

Query 1 × Query 4 ×

Cancel query Show only Editor

```
1  SELECT TOP (1000) * FROM [dbo].[accountsscdtype1]
```

I

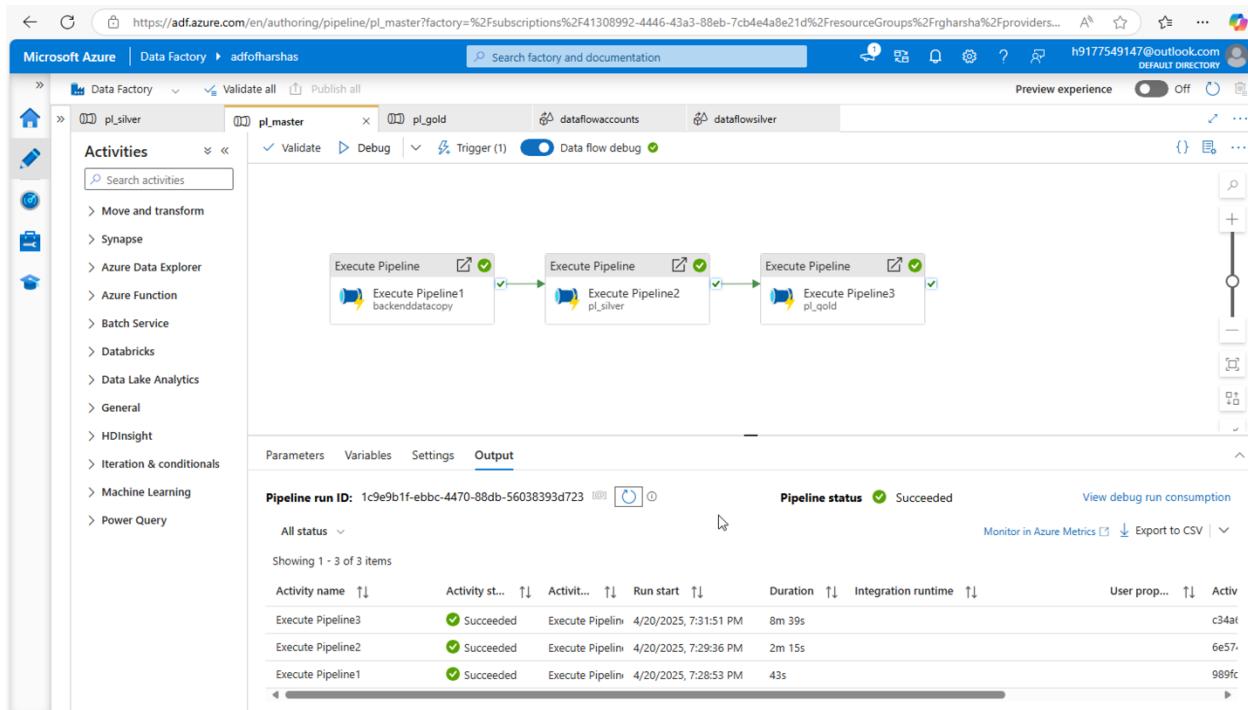
Results Messages

hashkey	created_by	created_date	updated_by	updated_date
2662522274	dataflow	2025-04-20T02:12:37.6510000	dataflow-updated	2025-04-21T02:32:33.1570000
2464207871	dataflow	2025-04-20T02:12:37.6510000	dataflow-updated	2025-04-21T05:04:01.8550000
1122301726	dataflow	2025-04-20T02:12:37.6510000	dataflow-updated	2025-04-21T02:32:33.1570000
630820887	dataflow	2025-04-20T02:12:37.6510000	dataflow-updated	2025-04-21T02:32:33.1570000

SCD type 1 data successfully stored in SQL database.

Challenges in step_3:

- In scd type 2 we while creating table we should not give any primary key



Master pipeline where all the 3 pipelines are connected and automatically triggered by a trigger.

In the above pipeline we have set a trigger

The screenshot shows a 'Triggers' section with the following details:

- Name:** pl_master_trigger
- Type:** Schedule
- Status:** Started
- Related:** 1

This trigger is enabled and the master pipeline is executing pipelines.

Execute pipeline is an activity where we select the child pipelines to run in the master pipeline. Master pipeline run activity:

- Firstly backend copy activity will be running (bronze layer)
- The pl_silver pipeline will be running to create delta format files. (Silver layer)
- PL_gold where the pipeline created gold layer in sql database.

While running the master pipeline it failed and the reason is “failed to connect to sql database”. After giving it sometime the pipeline successfully executed.