

Data Engineering Project-1:

Data Pipeline for Customer Account Analysis

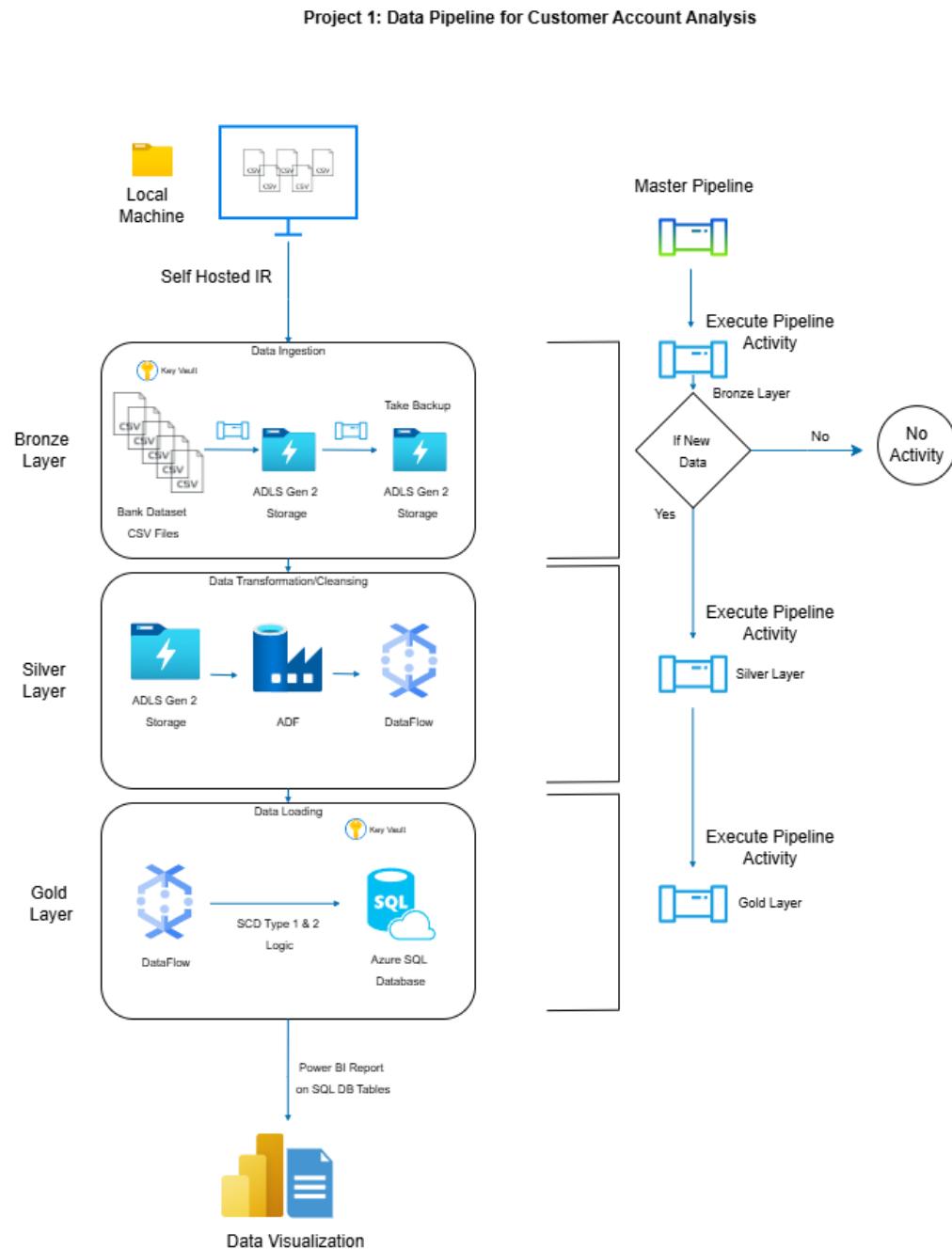
Table of Contents

Objective:	2
Project Flow Diagram:	2
Step 1: Data Ingestion (Backend Storage to Raw(Bronze) Container)	3
Step 2: Use ADF Dataflows to remove the duplicates	16
Step 3: Dataflows using SCD Type technique (SCD 1 and SCD 2)	45
Step 4: Use Power BI for Data Visualization	72
Step 5: GitHub Integration.....	74
Step 6: Conclusion.....	74
Points to remember.....	74

Objective:

The project aims to design and implement a robust data pipeline for processing customer account data. This includes copying data from a backend team's storage account, performing necessary transformations using ADF and upserts (inserting or updating) data from a file stored in Azure Data Lake Storage ADLS GOLD Storage into a SQL database table. The pipeline aims to ensure efficient, accurate, and scalable data processing to support downstream analytics and reporting needs.

Project Flow Diagram:



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

Azure Home -> Azure Data Factory -> Author Tab -> New Pipeline

Step 1.1: Get All Available Files (Dynamically discover all available files without hardcoding)

- o Use the Get Metadata Activity.
- o Source: Local folder via Self-Hosted Integration Runtime (SHIR).
- o Fetch: Child Items (all CSV file names from the folder).

Take the Get Metadata Activity first, and create a new data source file, **File System**, as we want to access the local storage folder.

Connection Schema Parameters

Linked service * Test connection + New Learn more

Integration runtime *

File path

Compression type

Column delimiter

Row delimiter

Encoding

Quote character

Escape character

First row as header

Settings tab in Get Metadata Activity:

Field list: Child item from drop-down options

General Settings User properties

Dataset * Open + New Learn more

Field list *
 Argument
 Child items

Filter by last modified Start time (UTC)
End time (UTC)

Skip line count

Step 1.2: Loop Through Each File (Process each file individually but dynamically)

Note: Inside the "Bronze Layer" pipeline, a ForEach activity is used to iterate through a list of your CSV files: 'accounts.csv', 'customers.csv', 'loan_payments.csv', 'loans.csv', and 'transactions.csv'.

This allows for processing multiple files in a single pipeline run.

Search ForEach Activity from the Activities tab and drag it to the design canvas

Write this expression in the ForEach Settings tab:

```
@activity('Get All Available Files').output.childItems
```

Click on the Edit icon of the ForEach activity

Step 1.3: Check File Freshness

- Get the Metadata Activity inside ForEach.
- Fetch the Last Modified timestamp of each file.

In the dataset, select a file system from all options and create a linked service using a **Self-hosted integration** runtime.

Give the path of the local folder where all the CSV files are stored.

Your username will be your Windows username. if you don't know, then execute the `whoami` command in the command prompt in Windows.

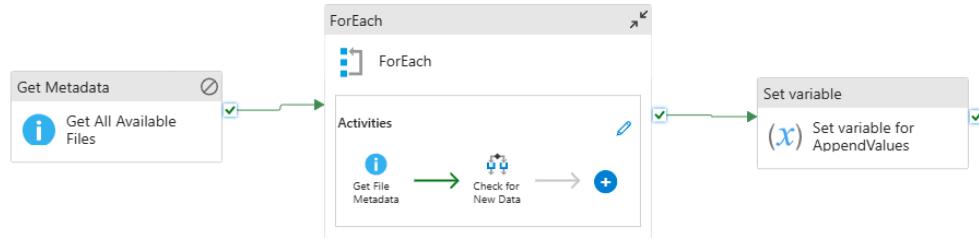
The password will be your account password attached to your account in Windows or your Windows login password, depending on your Windows user configurations.

We have stored the password in **Azure Key Vault** and used it in the linked service here in the Get Metadata activity.

[Dashboard](#) > [keyvault-harpal-vaghela](#)

The screenshot shows the Azure Key Vault interface for the 'keyvault-harpal-vaghela' vault. The left sidebar has links for Overview, Activity log, Access control (IAM), and Tags. The main area is titled 'keyvault-harpal-vaghela | Secrets'. It features a search bar and buttons for Generate/Import, Refresh, Restore Backup, and Manage deleted secrets. A table lists two secrets:

Name	Type
azuresqldb-password	
onprem-outlookuser-password	



Connect via integration runtime * ⓘ

 selfHostedIR

Host * ⓘ

 D:\DataEngineering\BankDataset

User name *

 harpa

Password

Azure Key Vault

AKV linked service * ⓘ

 ls_key_vault

Secret name * ⓘ

 onprem-outlookuser-password

Edit

Secret version ⓘ

 Latest version

Edit

Annotations

Create one parameter in the parameters tab:

Connection Schema Parameters

+ New | Delete

Name	Type	Default value
fileName	String	Value

Go to the connection tab:

Mention the filename parameter here as shown below:

Connection Schema Parameters

Linked service * ds_filesystem_csvfiles Test connection Edit + New Learn more

Integration runtime * self-HostedIR Edit

File path D:\DataEngineering\BankDataset / Directory / @dataset.fileName Browse | Preview data Detect format

Compression type No compression

Column delimiter Commas (,)

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

Encoding Default(UTF-8)

Quote character Double quote ("")

Escape character Backslash (\)

First row as header

Null value

In Field list options, select Last Modified from the drop down:

General Settings User properties

Dataset * ds_filesystem_csvfiles Open + New Learn more

Dataset properties

Name	Value
fileName	@item().name

Field list *

+ New | Delete

Argument

Last modified
Filter...

Start time

Exists

Item name

Item type

Last modified

Size

Finally Get Metadata Activity's settings tab will look like this:

General **Settings** User properties

Dataset * ds_filesystem_csvfiles Open New Learn more

Dataset properties

Name	Value
fileName	@item().name

Field list *

+ New | Delete

Argument
Last modified

Start time (UTC) End time (UTC)

Filter by last modified

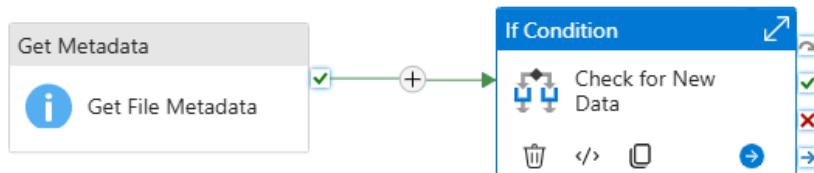
Skip line count

Step 1.4: If Condition (Check If Modified in Last 24 Hours)

Take if condition after the Get Metadata Activity and rename it

Write this expression in an if condition:

```
@greaterOrEquals(activity('Get File Metadata').output.lastModified, addDays(utcNow(), -1))
```



Inside the True part of the If Condition, take 3 activities as

Copy Data (for copying files from local to Bronze layer in ADLS Gen 2 Storage account)

Copy Data (for taking backup of the client's data)

Append Variable activity (for storing the name of files, as we will use this for a condition later on)

pl_project1_bronze_layer > ForEach > Check for New Data - True activities



Step 1.5: Copy Files to Bronze + Take Backup

- Copy Data Activity 1: Local → ADLS Gen2 (Bronze folder)
- Copy Data Activity 2: Local → ADLS Gen2 (Backup folder)
- Dynamic folder structure: Add a timestamp to the backup filename.

Use the same dataset that we have used for the Get Metadata activity

Source dataset *					
<input type="button" value="ds_filesystem_csvfiles"/>	Open				
New	Preview data				
Learn more					
Dataset properties <table border="1"> <tr> <th>Name</th> <th>Value</th> </tr> <tr> <td>fileName</td> <td>@item().name</td> </tr> </table>		Name	Value	fileName	@item().name
Name	Value				
fileName	@item().name				
File path type <input checked="" type="radio"/> File path in dataset <input type="radio"/> File filter <input type="radio"/> Wildcard file path <input type="radio"/> List of files					
Filter by last modified <input type="checkbox"/>					
Start time (UTC) <input type="text"/>					
End time (UTC) <input type="text"/>					
Recursively <input type="checkbox"/>					
Enable partitions discovery <input type="checkbox"/>					

For Sink, we will create a new dataset, as we have to store data in an ADLS Gen 2 storage account.

Link service:

Connect via integration runtime * ⓘ
 AutoResolveIntegrationRuntime

Authentication type
Account key

Account selection method ⓘ
 From Azure subscription Enter manually

URL *
https://adlsharopalvaghela.dfs.core.windows.net/

Storage account key Azure Key Vault

Storage account key *
.....

Test connection ⓘ
 To linked service To file path

Annotations
+ New
Parameters
Advanced ⓘ

Create one parameter in the parameters tab:

Connection	Schema	Parameters
		+ New Delete
Name	Type	Default value
FileName	String	Value Delete

Go to the connection tab:

Mention the filename parameter here as shown below:

Connection Schema Parameters

Linked service * Test connection Edit New Learn more

File path / /

Compression type

Column delimiter

Row delimiter

Encoding

Quote character

Escape character

First row as header

Null value

Here, select the first row as a header, as we have all CSV files with a header.

Select the Bronze folder from the ADLS Gen 2 account.

Finally sink part will look like this:

General Source **Sink** Mapping Settings User properties

Sink dataset * Open New Learn more

Dataset properties

Name	Value
FileName	<input type="text" value="@item().name"/>

Copy behavior

Max concurrent connections

Block size (MB)

Metadata

Quote all text

File extension

- **Copy Data Activity (For Taking Backup)**

Use the same source dataset as used in the first copy data activity

General **Source** Sink Mapping Settings User properties

Source dataset *

Dataset properties

Name	Value
FileName	@item().name

File path type File path in dataset Wildcard file path List of files

Filter by last modified

Recursively

Enable partitions discovery

Max concurrent connections

Skip line count

For Sink, create a new data source for the ADLS Gen 2 storage account.

Create one parameter in the parameters tab:

Connection Schema **Parameters**

Name	Type	Default value
fileName	String	<input type="text" value="Value"/> <input type="button" value="Delete"/>

General Source **Sink** Mapping Settings User properties

Sink dataset *

Dataset properties

Name	Value
fileName	@item().name

Copy behavior

Max concurrent connections

Block size (MB)

Metadata

Quote all text

File extension

Max rows per file

Go to the connection tab and select the clientbackupfiles folder by clicking the browse icon.

Connection Schema Parameters

Linked service * Test connection Edit + New Learn more

File path / / Browse | Preview data Detect format

Compression type

Column delimiter

Row delimiter

Encoding

Quote character

Escape character

First row as header

Null value

Use this expression for the filename as shown below:

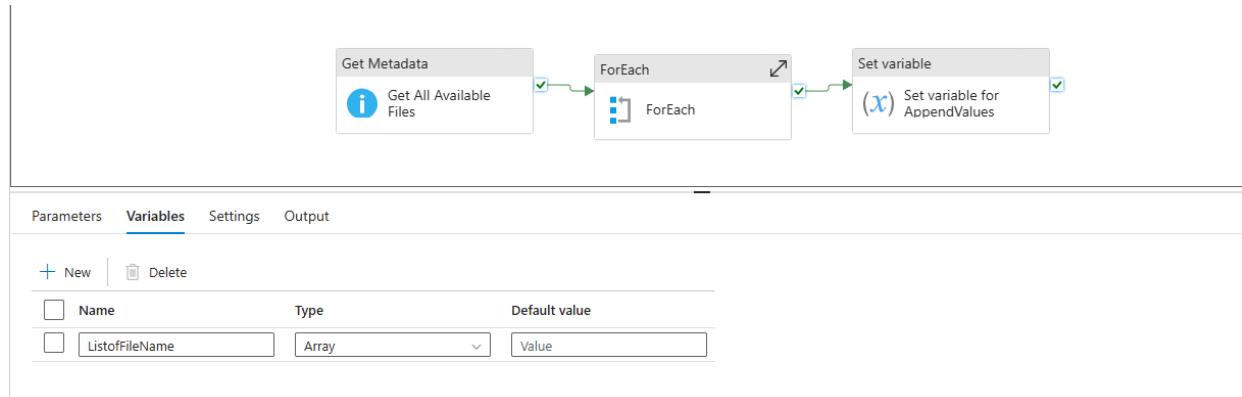
```
@concat(replace(dataset().fileName, '.csv', ''), '/', replace(dataset().fileName, '.csv', ''), '_', utcNow(), '.csv')
```

This will create a folder in the clientbackupfiles folder as per our file (accounts, transactions, loans, etc)

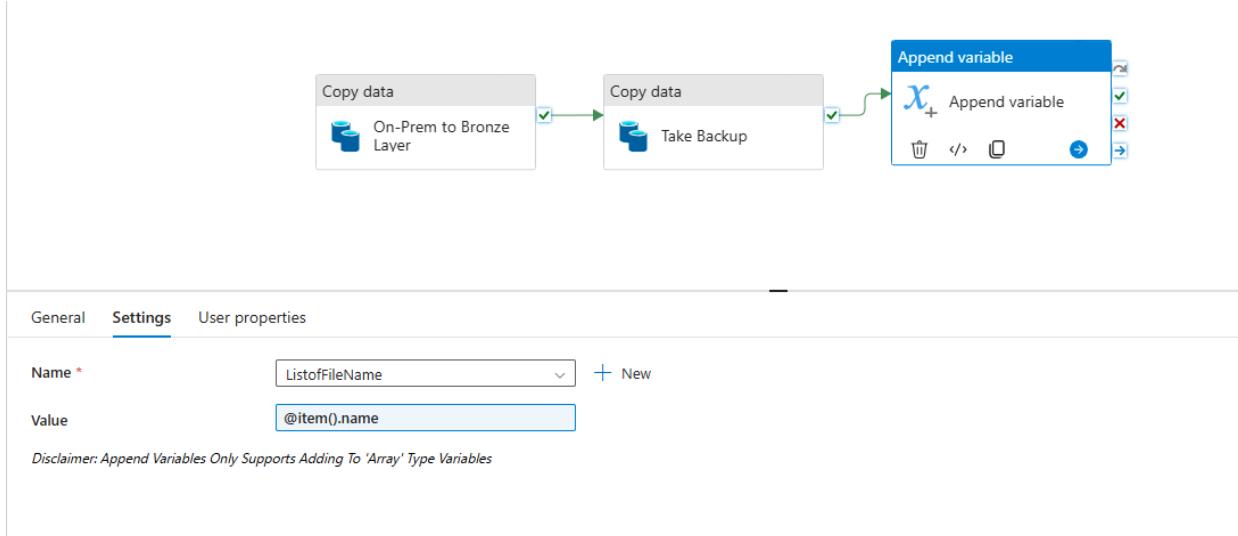
Step 1.6: Capture File Names

- Append Variable Activity: Collects names of modified files for downstream steps
- Inform Silver and Gold layers which files to process.

First of all, create a pipeline level variable as shown below



Now, go inside the ForEach -> If Condition activity



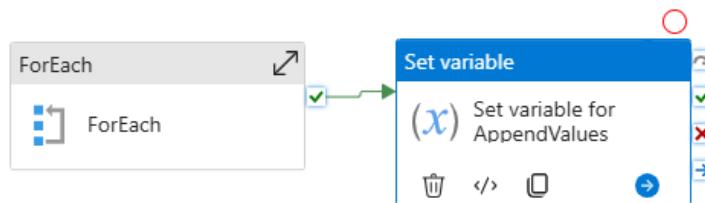
Select that variable here, and in value mention @item().name

So it will store all file names if they are modified or the client has added new data inside them.

Step 1.7: Return Values to Master Pipeline

- Set Variable Activity

Go outside of the ForEach activity and take the Set Variable activity



Create a new variable, and the type should be Pipeline Return Value

Name: `ReturnFileNameValue`

Type: `Expression`

Value: `@variables('ListofFileName')`

So, now we have designed our bronze layer.

Step 1.8: Create Master Pipeline

- Master Pipeline

Create a new pipeline, let's call it Master Pipeline, from which we will execute all layers. (Bronze, Silver and Gold Layers)

Take the Execute Pipeline activity and drag it to the canvas.

In Execute Pipeline Settings, select our Bronze layer pipeline, as shown below

The screenshot shows the 'Execute Pipeline' settings interface. At the top, there are tabs for 'General', 'Settings' (which is selected), and 'User properties'. Below these, under 'Invoked pipeline', a dropdown menu is open, showing 'pl_project1_bronze_layer' as the selected option. To the right of the dropdown are edit and cancel icons. Under 'Wait on completion', a checked checkbox is followed by a blue checkmark icon.

Step1.9: Set Variable Activity in Master Pipeline

Now, take the Set variable activity after an Execute pipeline activity

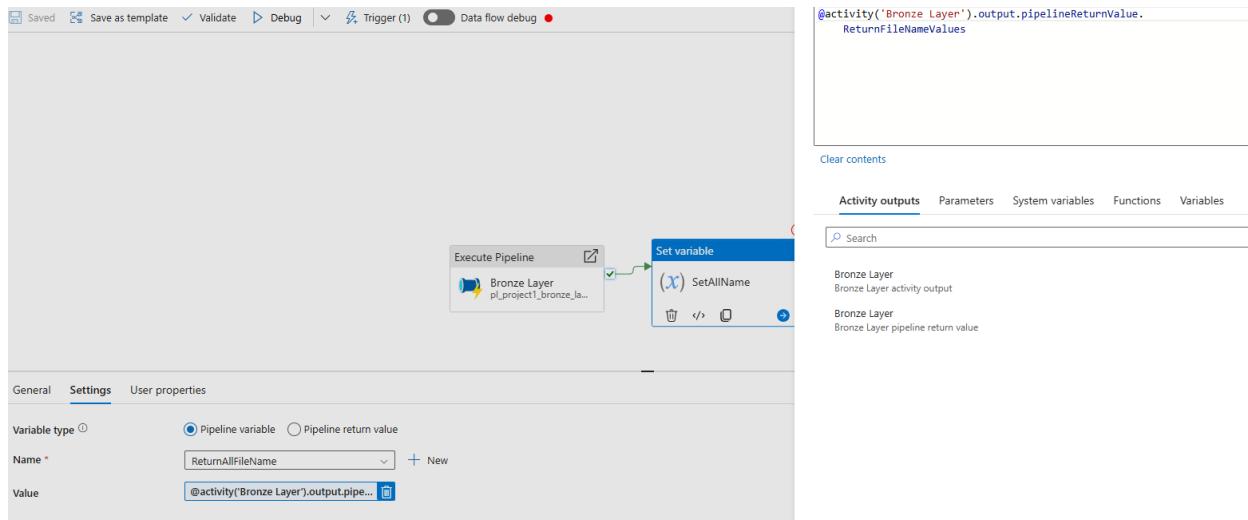


Take the Pipeline level variable in the master pipeline

Name	Type	Default value
ReturnAllFileName	Array	

The type should be an Array as we need an array of file names.

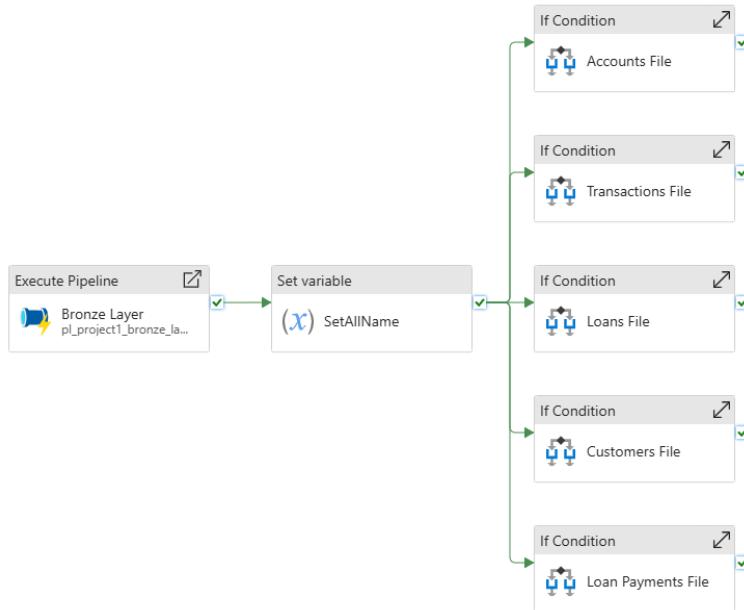
Write this expression in Set Variable -> Settings -> Value



`@activity('Bronze Layer').output.pipelineReturnValue.ReturnFileNameValues`

Step 1.9: If Condition in Master Pipeline

- If Condition: IF Condition activities allow you to create branching logic in your pipeline flow. Based on a dynamic condition (True/False), you can control which activities or pipelines should execute, making your process efficient and intelligent.



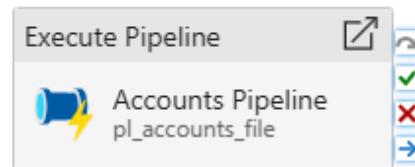
Rename each IF Condition activity appropriately based on the dataset it is intended to control (as shown in the reference image). Within the True path of each IF Condition, add an Execute Pipeline activity.

Each Execute Pipeline activity should be configured to call a dedicated pipeline designed specifically for that file, where both Silver Layer (data transformation) and Gold Layer (SCD processing and loading) are implemented.

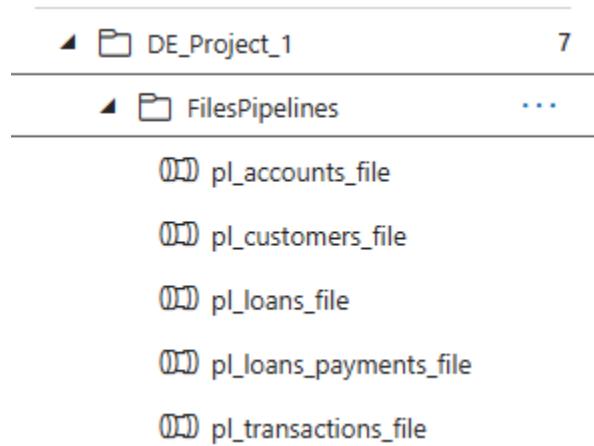
This structure ensures that only the relevant Silver and Gold processes are triggered for each dataset individually, based on the file modification status.

For Example:

 pl_project1_master > Accounts File - True activities



As shown below, create 5 pipelines for all 5 CSV files to be processed in the silver and gold layers.



Step 2: Use ADF Dataflows to remove the duplicates

Step 2.1: Accounts File Pipeline Design (Silver Layer Processing)

The Silver Layer aims to cleanse, standardize, and deduplicate the raw ingested data to prepare it for downstream processing in the Gold Layer.

Dataflow Activity: Drag and drop a new Dataflow into the canvas area.



First, we will go through the silver layer in all pipelines.

Here is the entire flow:



Step-by-Step Process:

- Source Transformation:

Source Settings:

Source settings Source options Projection Optimize Inspect Data preview

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

Description Import data from ls_adls_csvfiles [Reset](#)

Source type * Dataset Inline

Inline dataset type * DelimitedText

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

Skip line count

Sampling * Enable Disable

Source Options:

Source Dataset: Point to the accounts.csv file located in the Bronze Layer (ADLS Gen2 storage).

Specify the file path. Enable the checkbox "First row as header" since the file contains column names.

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 \n)

Quote character Double quote ("")

Escape character Backslash (\)

First row as header

Projection:

- Projection allows you to define or modify the column structure of the incoming dataset in ADF Dataflows.
- Click Import Schema to automatically detect the file structure.

- Perform a Data Preview to validate imported columns.

Column name	Type	Format
account_id	short	Specify format
customer_id	short	Specify format
account_type	string	Specify format
balance	double	Specify format

- **Filter Transformation**

- Add a Filter Transformation immediately after the Source.
- Filter Transformation allows you to include or exclude rows based on logical conditions.

Use this filtering condition: `!isNull(account_id) || !isNull(customer_id)`

Output stream name *

[Learn more](#)

Description

Filtering rows using expressions on columns 'account_id, customer_id'

 Reset

Incoming stream *

Filter on *

`!isNull(account_id) ||
!isNull(customer_id)`

- **Aggregate Transformation(Remove Duplicate)**

- Choose appropriate columns (account_id, customer_id) to identify duplicates.
- Ensure only unique records move forward for consistency and data quality.
- Remove Duplicate identifies and removes rows that have identical values across selected key columns.

Aggregate settings Optimize Inspect Data preview

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

Description Aggregating data by 'account_id' producing columns 'customer_id', 'account_type', 'balance'

Incoming stream * FilterAccounts

Group by **Aggregates**

Columns	Name as
12s account_id	account_id

Incoming stream * FilterAccounts

Group by **Aggregates**

Grouped by: account_id

+ Add Clone Delete Open expression builder

Column	Expression
Each column that matches name='account_id'	creates 1 column(s)
\$\$	abc first(\$\$) ANY

Inspect Tab:

Aggregate settings Optimize **Inspect** Data preview

Schema **Input** Output

Number of columns New * 3		Dropped 3	Unchanged 1	
Order ↑	Column ↑	Type ↑	Aggregated as ↑	Based on ↑
1	account_id	12s short	Group by	account_id
2	customer_id	12s short	Aggregate	customer_id
3	account_type	abc string	Aggregate	account_type
4	balance	1.2 double	Aggregate	balance

- **Select Transformation**

- Add a Select Transformation to rename columns for standardization.
- Example: Rename account_id to AccountID, customer_id to CustomerID, etc.
- Select Transformation is used to rename, reorder, or drop columns in ADF Dataflows.

Select settings Optimize Inspect Data preview

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

Description

Incoming stream *

Options Skip duplicate input columns [\[?\]](#)
 Skip duplicate output columns [\[?\]](#)

Input columns * Auto mapping [\[?\]](#) 4 mappings: All inputs mapped

RemoveDuplicate's column	Name as
12s account_id	AccountID
12s customer_id	CustomerID
abc account_type	AccountType
12 balance	AccountBalance

- **Alter Row Transformation**

- Insert an Alter Row Transformation.
- Configure it to always Upsert records using the condition: $1==1$
- Alter Row Transformation assigns actions (Insert, Update, Delete, Upsert) at the row level based on specified conditions.

Alter row settings Optimize Inspect Data preview

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

Description

Incoming stream *

Alter row conditions * [\[?\]](#)

- **Sink Transformation**

- Add a Sink to write the cleansed data.
- Sink Type: Inline dataset.
- Dataset Type: Delta (high-performance, ACID-compliant format).
- Linked Service: ADLS Gen2 storage account.
- Settings Tab:
 - Enable "Auto Create Table" if needed.
 - Review mappings under the Mapping Tab to ensure correct column flow.
- Sink is the destination where transformed data is written, such as files, tables, or databases.

Sink Settings Errors Mapping Optimize Inspect Data preview

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

Description Add sink dataset [Reset](#)

Incoming stream * alterRow1

Sink type * Dataset **Inline** Cache

Inline dataset type * Delta

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

Options Allow schema drift [①](#) Validate schema [①](#)

Settings:

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

Folder path * project1 / silver/accounts [Browse](#)

Compression type No compression

Vacuum 0

Table action None Overwrite [①](#) Truncate [①](#)

Update method Allow insert
 Allow delete
 Allow upsert
 Allow update

[Δ Delta options](#)

Inspect:

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

Schema [Input](#) **Output**

Number of columns Updated* 0		Dropped 0	Unchanged 4	To
Order ↑↓	Column ↑↓	Type ↑↓	Updated ↑↓	Input column ↑↓
1	AccountID	12s short		AccountID
2	CustomerID	12s short		CustomerID
3	AccountType	abc string		AccountType
4	AccountBalance	12 double		AccountBalance

That's it for the silver layer accounts dataflow.

Similarly, we have to design for the other 4 files.

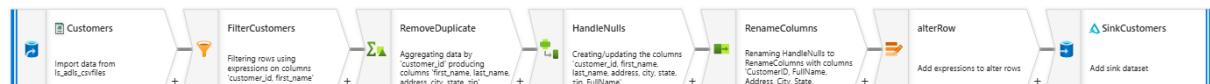
Step 2.2: Customers File Pipeline (Silver Layer Processing)

Step-by-Step Process:

Drag and drop two dataflows in the canvas area



Here is the entire flow:



- Source Transformation**

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

Skip line count

Sampling * Enable Disable

Source Options:

Set the file path pointing to the Bronze folder. Enable the "First row as header" option since the file includes headers.

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

File mode File Wildcard

File path * / /

Allow no files found

Change data capture

Compression type

Encoding

Column delimiter

Row delimiter

Quote character

Escape character

First row as header

Projection:

- Click Import Schema to automatically generate the schema.
- Perform a Data Preview to verify columns and data types.

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

Column name	Type	Format
customer_id	12s short	Specify format
first_name	abc string	Specify format
last_name	abc string	Specify format
address	abc string	Specify format
city	abc string	Specify format
state	abc string	Specify format
zip	abc string	Specify format

- **Filter Transformation**

Add a Filter Transformation after the Source. Use this expression: `!isNull(customer_id) || !isNull(first_name)`

Filter settings Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description

Incoming stream *

Filter on *

- Aggregate Transformation (Remove Duplicate):

Aggregate settings Optimize Inspect Data preview ●

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

Description Aggregating data by 'customer_id' producing columns 'first_name, last_name, address, city, state, zip'

Incoming stream * FilterCustomers [Reset](#)

[Group by](#) [Aggregates](#)

Columns	Name as
12s customer_id	customer_id

Aggregate settings Optimize Inspect Data preview ●

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

Description Aggregating data by 'customer_id' producing columns 'first_name, last_name, address, city, state, zip'

Incoming stream * FilterCustomers [Reset](#)

[Group by](#) [Aggregates](#)

Grouped by: customer_id

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

Column	Expression
Each column that matches name!=customer_id	creates 1 column(s)
\$\$	first(\$\$)

Data Preview Tab:

customer_id	first_name	last_name	address	city	state	zip
1	John	Doe	123 Elm St	Toronto	ON	M4B1B3
2	Jane	Smith	456 Maple Ave	Ottawa	ON	K1A0B1
3	Michael	Johnson	789 Oak Dr	Montreal	QC	H1A1A1
4	Emily	Davis	101 Pine Rd	Calgary	AB	T2A0A1
5	David	Wilson	202 Birch Blvd	Vancouver	BC	V5K0A1
6	Emma	Clark	505 Cedar St	Halifax	NS	B3H0A1
7	James	Martinez	606 Spruce Ln	Winnipeg	MB	R3C0A1
8	Olivia	Garcia	707 Fir St	Edmonton	AB	T5A0A1
9	William	Lopez	808 Redwood Dr	Victoria	BC	V8W0A1
10	Ava	Anderson	909 Cypress Ave	Quebec City	QC	G1A0A1
11	Alexander	Thomas	1010 Willow Rd	St. John's	NL	A1A0A1
12	Isabella	Lee	1111 Poplar St	Fredericton	NB	E3B0A1
13	Daniel	Harris	1212 Ash Blvd	Charlottetown	PE	C1A0A1

- **Derived Column Transformation:**

Column	Expression
FullName	concat(first_name, ' ', last_name)
zip	iifNull(zip, 'Unknown', zip)
state	iifNull(state, 'Unknown', state)
city	iifNull(city, 'Unknown', city)

Derived column's settings Optimize Inspect Data preview ●

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

Description Creating/updating the columns 'customer_id, first_name, last_name, address, city, state, zip, FullName'

Incoming stream * RemoveDuplicate

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

Columns * ⓘ

<input type="checkbox"/> Column	Expression
<input type="checkbox"/> FullName	concat(first_name, ' ', last_name)
<input type="checkbox"/> zip	iifNull(zip, 'Unknown', zip)
<input type="checkbox"/> state	iifNull(state, 'Unknown', state)
<input type="checkbox"/> city	iifNull(city, 'Unknown', city)

Data Preview Tab:

Number of rows		INSERT 88	UPDATE 0	DELETE 0	UPSERT 0	LOOKUP 0	ERROR 0								
↻	Refresh	Typecast	Modify	Map drifted	Statistics	Remove	Export to CSV								
↑↓	customer_id	↑↓	first_name	abc ↑↓	last_name	abc ↑↓	address	abc ↑↓	city	abc ↑↓	state	abc ↑↓	zip	abc ↑↓	FullName
+	1		John		Doe		123 Elm St		Toronto		ON		M4B1B3		John Doe
+	2		Jane		Smith		456 Maple Ave		Ottawa		ON		K1A0B1		Jane Smith
+	3		Michael		Johnson		789 Oak Dr		Montreal		QC		H1A1A1		Michael Johnson
+	4		Emily		Davis		101 Pine Rd		Calgary		AB		T2A0A1		Emily Davis
+	5		David		Wilson		202 Birch Blvd		Vancouver		BC		V5K0A1		David Wilson
+	6		Emma		Clark		505 Cedar St		Halifax		NS		B3H0A1		Emma Clark
+	7		James		Martinez		606 Spruce Ln		Winnipeg		MB		R3C0A1		James Martinez
+	8		Olivia		Garcia		707 Fir St		Edmonton		AB		T5A0A1		Olivia Garcia
+	9		William		Lopez		808 Redwood Dr		Victoria		BC		V8W0A1		William Lopez
+	10		Ava		Anderson		909 Cypress Ave		Quebec City		QC		G1A0A1		Ava Anderson
+	11		Alexander		Thomas		1010 Willow Rd		St. John's		NL		A1A0A1		Alexander Thomas
+	12		Isabella		Lee		1111 Poplar St		Fredericton		NB		E3B0A1		Isabella Lee

- Select Transformation:**

Rename the column name here

The screenshot shows the configuration for the 'RenameColumns' transformation. It includes fields for 'Output stream name' (set to 'RenameColumns'), 'Description' (containing 'Renaming HandleNulls to RenameColumns with columns 'CustomerID, FullName, Address, City,''), 'Incoming stream' (set to 'HandleNulls'), and 'Options' (with checkboxes for 'Skip duplicate input columns' and 'Skip duplicate output columns' both checked). Under 'Input columns', there is a mapping table showing the renaming of columns from their original names to new names: 'customer_id' to 'CustomerID', 'FullName' to 'FullName', 'address' to 'Address', 'city' to 'City', 'state' to 'State', and 'zip' to 'PostalCode'. A note at the bottom right indicates '6 mappings: 2 column(s) from the inputs left unmapped'.

Data Preview Tab:

The screenshot shows the data preview tab with a table of 88 rows. The columns are CustomerID, FullName, Address, City, State, and PostalCode. The data includes various Canadian cities and provinces like Toronto, Ottawa, Montreal, Calgary, Vancouver, Halifax, NS, MB, and AB.

CustomerID	FullName	Address	City	State	PostalCode
1	John Doe	123 Elm St	Toronto	ON	M4B1B3
2	Jane Smith	456 Maple Ave	Ottawa	ON	K1A0B1
3	Michael Johnson	789 Oak Dr	Montreal	QC	H1A1A1
4	Emily Davis	101 Pine Rd	Calgary	AB	T2A0A1
5	David Wilson	202 Birch Blvd	Vancouver	BC	V5K0A1
6	Emma Clark	505 Cedar St	Halifax	NS	B3H0A1
7	James Martinez	606 Spruce Ln	Winnipeg	MB	R3C0A1

- Alter Row Transformation:**

Select Upsert if condition from the drop down, and $1==1$ expression

The screenshot shows the configuration for the 'alterRow1' transformation. It includes fields for 'Output stream name' (set to 'alterRow1'), 'Description' (containing 'Add expressions to alter rows'), 'Incoming stream' (set to 'select1'), and 'Alter row conditions' (set to 'Upsert if' with the expression '1==1').

- Sink Transformation:**

Sink type: Inline

Inline dataset type: Delta

Linked service: ADLS Gen 2 storage account

Sink Settings Errors Mapping Optimize Inspect Data preview

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

Description Add sink dataset [Reset](#)

Incoming stream * alterRow

Sink type * Dataset **Inline** Cache

Inline dataset type * Delta

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

Options Allow schema drift [①](#)
 Validate schema [①](#)

Settings:

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

Folder path * project1 / silver/customers/ [Browse](#)

Compression type No compression

Vacuum [①](#) 0

Table action None Overwrite [①](#) Truncate [①](#)

Update method [①](#) Allow insert
 Allow delete
 Allow upsert
 Allow update

Key columns * [①](#) List of columns Custom expression [①](#)

CustomerID [+](#) [-](#)

Data Preview Tab:

CustomerID	FullName	Address	City	State	PostalCode
1	John Doe	123 Elm St	Toronto	ON	M4B1B3
2	Jane Smith	456 Maple Ave	Ottawa	ON	K1A0B1
3	Michael Johnson	789 Oak Dr	Montreal	QC	H1A1A1
4	Emily Davis	101 Pine Rd	Calgary	AB	T2A0A1
5	David Wilson	202 Birch Blvd	Vancouver	BC	V5K0A1
6	Emma Clark	505 Cedar St	Halifax	NS	B3H0A1
7	James Martinez	606 Spruce Ln	Winnipeg	MB	R3C0A1
8	Olivia Garcia	707 Fir St	Edmonton	AB	T5A0A1
9	William Lopez	808 Redwood Dr	Victoria	BC	V8W0A1
10	Ava Anderson	909 Cypress Ave	Quebec City	QC	G1A0A1
11	Alexander Thomas	1010 Willow Rd	St. John's	NL	A1A0A1
12	Isabella Lee	1111 Poplar St	Fredericton	NB	E3B0A1
13	Daniel Harris	1212 Ash Blvd	Charlottetown	PE	C1A0A1

That's it for the silver layer customers' data flow.

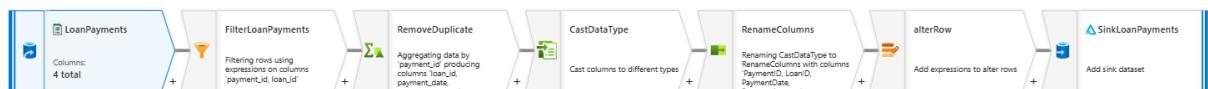
- **Loan Payments File Pipeline**

Drag and drop two dataflows in the canvas area



First, we will go through the silver layer in this.

Here is the entire flow:



- **Source:**

Source Settings:

Source settings [Source options](#) [Projection](#) [Optimize](#) [Inspect](#) [Data preview](#) ●

Output stream name * [Learn more](#)

Description [Reset](#)

Source type *  Dataset  Inline

Inline dataset type *  DelimitedText [▼](#)

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

Skip line count

Sampling * [①](#) Enable Disable

Source Options:

File path for the loan_payments.csv file from the bronze folder in ADLS Gen 2

Tick the first row as a header

[Source settings](#) **Source options** [Projection](#) [Optimize](#) [Inspect](#) [Data preview](#) ●

File settings

File mode File Wildcard

File path * / / [Browse](#)

Allow no files found

Change data capture

Compression type

Encoding

Column delimiter

Row delimiter

Quote character

Escape character

First row as header

Projection:

Click on Import Schema, and then click on Data Preview

The screenshot shows the 'Projection' tab of a data pipeline configuration. It lists four columns: 'payment_id', 'loan_id', 'payment_date', and 'payment_amount'. Each column has a 'Type' dropdown and a 'Format' dropdown. The types are set to '12s short' for payment_id, loan_id, and payment_amount, and 'date' for payment_date. The formats are all set to 'Specify format'.

Column name	Type	Format
payment_id	12s short	Specify format
loan_id	12s short	Specify format
payment_date	date	Specify format
payment_amount	12s short	Specify format

Filter transformation:

Click on the plus icon at the bottom of the source, and take a filter transformation

Use this expression: `!isNull(payment_id) || !isNull(loan_id)`

The screenshot shows the 'Filter settings' tab for a filter transformation. The output stream name is 'FilterLoanPayments'. The incoming stream is 'LoanPayments'. The filter expression is `!isNull(payment_id) || !isNull(loan_id)`.

Output stream name *	FilterLoanPayments	Learn more
Description	Filtering rows using expressions on columns 'payment_id, loan_id'	Reset
Incoming stream *	LoanPayments	
Filter on *	<code>!isNull(payment_id) !isNull(loan_id)</code>	

- **Aggregate Transformation (Remove Duplicate)**

The screenshot shows the 'Aggregate settings' tab for an aggregate transformation. The output stream name is 'RemoveDuplicate'. The incoming stream is 'FilterLoanPayments'. The group by column is 'payment_id'. The aggregate column is also 'payment_id'.

Aggregate settings	Optimize	Inspect	Data preview
Output stream name *	RemoveDuplicate	Learn more	
Description	Aggregating data by 'payment_id' producing columns 'loan_id, payment_date, payment_amount'	Reset	
Incoming stream *	FilterLoanPayments		
<input checked="" type="radio"/> Group by <input type="radio"/> Aggregates			
Columns	Name as		
12s payment_id	payment_id		

Incoming stream * FilterLoanPayments

Grouped by: payment_id

Add Clone Delete Open expression builder

Column	Expression
Each column that matches name!=payment_id	creates 1 column(s) first(\$\$)

- Cast Transformation:**

Cast settings Optimize Inspect Data preview

Output stream name * CastDataType

Description Cast columns to different types

Incoming stream * RemoveDuplicate

Columns *

Column name	Type	Format
payment_date	timestamp	yyyy-MM-dd'T'HH:mm:ss

Assert type check

Data Preview:

Number of rows		INSERT 100	UPDATE 0	DELETE 0	UPSERT 0	LOOKUP 0	ERROR 0
↻ Refresh ↴	Typecast ↴	Modify ↴	Map drifted ↴	Statistics ↴	Remove ↴	Export to CSV ↴	
↑↓ payment_id	12s ↑↓ loan_id	12s ↑↓ payment_date	12s ↑↓ payment_amount				
+ 1	45	2024-01-01 00:00:00.000	100				
+ 2	23	2024-01-02 00:00:00.000	150				
+ 3	67	2024-01-03 00:00:00.000	200				
+ 4	89	2024-01-04 00:00:00.000	250				
+ 5	12	2024-01-05 00:00:00.000	300				
+ 6	34	2024-01-06 00:00:00.000	350				
+ 7	56	2024-01-07 00:00:00.000	400				
+ 8	78	2024-01-08 00:00:00.000	450				
+ 9	90	2024-01-09 00:00:00.000	500				
+ 10	11	2024-01-10 00:00:00.000	550				
+ 11	22	2024-01-11 00:00:00.000	600				

- Select Transformation:**

Rename the column name here

Select settings Optimize Inspect Data preview ●

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

Description Renaming CastDataType to RenameColumns with columns 'PaymentID, LoanID, PaymentDate, PaymentAmount'

Incoming stream * CastDataType

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

Input columns * Auto mapping [ⓘ](#) [Reset](#) [+](#) Add mapping [Delete](#)

CastDataType's column	Name as
12s payment_id	PaymentID
12s loan_id	LoanID
PaymentDate	PaymentDate
12s payment_amount	PaymentAmount

Data Preview Tab:

Number of rows		INSERT 100	UPDATE 0	DELETE 0	UPSERT 0	LOOKUP 0	ERROR 0
Refresh	▼	Typecast	Modify	Map drifted	Statistics	Remove	Export to CSV
↑↓	PaymentID	12s ↑↓	LoanID	12s ↑↓	PaymentDate	↑↓	PaymentAmount
+	1		45		2024-01-01 00:00:00.000		100
+	2		23		2024-01-02 00:00:00.000		150
+	3		67		2024-01-03 00:00:00.000		200
+	4		89		2024-01-04 00:00:00.000		250
+	5		12		2024-01-05 00:00:00.000		300
+	6		34		2024-01-06 00:00:00.000		350
+	7		56		2024-01-07 00:00:00.000		400
+	8		78		2024-01-08 00:00:00.000		450
+	9		90		2024-01-09 00:00:00.000		500
+	10		11		2024-01-10 00:00:00.000		550
+	11		22		2024-01-11 00:00:00.000		600

- Alter Row Transformation:

Select Upsert if condition from the drop down and $1==1$ expression

Alter row settings Optimize Inspect Data preview

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

Description Add expressions to alter rows

Incoming stream * select1

Alter row conditions * [ⓘ](#) Upsert if [1==1](#) [+](#) [Delete](#)

- Sink Transformation:

Sink type: Inline

Inline dataset type: Delta

Linked service: ADLS Gen 2 storage account

Sink Settings Errors Mapping Optimize Inspect Data preview ●

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

Description Add sink dataset [Reset](#)

Incoming stream * alterRow

Sink type * Dataset **Inline** Cache

Inline dataset type * Delta

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

Options Allow schema drift [①](#)
 Validate schema [①](#)

Settings:

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

Folder path * project1 / silver/loanpayments/ [Browse](#)

Compression type No compression

Vacuum [①](#) 0

Table action None Overwrite [①](#) Truncate [①](#)

Update method [①](#) Allow insert
 Allow delete
 Allow upsert
 Allow update

Key columns * [①](#) List of columns Custom expression [①](#)

123 PaymentID [+](#) [-](#)

Data Preview Tab:

Number of rows		+ INSERT 0	+ UPDATE 0	X DELETE 0	+ UPSERT 0	X LOOKUP 0	X ERROR 0
1	↓	PaymentID	↑↓	LoanID	↑↓	PaymentDate	↑↓
1	↑	1		45		2024-01-01 00:00:00.000	
2	↑	2		23		2024-01-02 00:00:00.000	
3	↑	3		67		2024-01-03 00:00:00.000	
4	↑	4		89		2024-01-04 00:00:00.000	
5	↑	5		12		2024-01-05 00:00:00.000	
6	↑	6		34		2024-01-06 00:00:00.000	
7	↑	7		56		2024-01-07 00:00:00.000	
8	↑	8		78		2024-01-08 00:00:00.000	
9	↑	9		90		2024-01-09 00:00:00.000	
10	↑	10		11		2024-01-10 00:00:00.000	
11	↑	11		22		2024-01-11 00:00:00.000	
12	↑	12		33		2024-01-12 00:00:00.000	

That's it for the silver layer loan payments' data flow.

- **Loan File Pipeline (Silver Layer)**

Drag and drop two dataflows in the canvas area



First, we will go through the silver layer in this.

Here is the entire flow:



Source:

Source Settings:

Source settings Source options Projection Optimize Inspect Data preview ●

Output stream name * [Learn more](#)

Description [Reset](#)

Source type * Dataset Inline

Inline dataset type * DelimitedText

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

Skip line count

Sampling * Enable Disable

Source Options:

File path for loans.csv file from the bronze folder in ADLS Gen 2

Tick the first row as a header

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

File settings

File mode File Wildcard

File path * / / [Browse](#)

Allow no files found

Change data capture

Compression type

Encoding

Column delimiter

Row delimiter

Quote character

Escape character

First row as header

Projection: Click on Import Schema

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

Import schema Clear schema Schema options

Column name	Type	Format
loan_id	short	Specify format
customer_id	short	Specify format
loan_amount	double	Specify format
interest_rate	double	Specify format
loan_term	short	Specify format

Click on the plus icon at the bottom of the source, and take a filter transformation

Use this expression: `!isNull(loan_id) || !isNull(customer_id)`

Incoming stream *

Loans

Filter on *

`!isNull(loan_id) ||
!isNull(customer_id)`

- Aggregate Transformation (Remove Duplicate):

Aggregate settings Optimize Inspect Data preview

Output stream name *

RemoveDuplicate

Description

Aggregating data by 'loan_id' producing columns 'customer_id, loan_amount, interest_rate, loan_term'

Incoming stream *

FilterLoans

Group by Aggregates

Columns Name as

loan_id loan_id

Grouped by: loan_id

Add Clone Delete Open expression builder

Column Expression

Each column that matches `name != 'loan_id'` creates 1 column(s)

`first($$)`

- Select Transformation:**

Rename the column name here

Screenshot of the Column Mappings section in a data transformation tool. It shows five input columns: loan_id, customer_id, loan_amount, interest_rate, and loan_term, each mapped to a new name: LoanID, CustomerID, LoanAmount, InterestRate, and LoanTerm respectively. There are checkboxes for skipping duplicate input and output columns, and buttons for auto mapping, reset, add mapping, and delete.

Data Preview Tab:

Data Preview						
	LoanID	CustomerID	LoanAmount	InterestRate	LoanTerm	
1	1	45	10000.5	5.5	36	
2	2	12	20000.75	4.5	48	
3	3	78	15000.0	6.0	60	
4	4	34	30000.25	3.5	24	
5	5	56	25000.0	5.0	36	
6	6	23	17500.5	4.0	48	
7	7	89	22500.75	6.5	60	
8	8	67	27500.0	3.0	24	
9	9	14	32500.25	5.5	36	
10	10	92	37500.5	4.5	48	
11	11	3	10000.75	6.0	60	
12	12	81	20000.0	3.5	24	

- Alter Row Transformation:**

Select Upsert if condition from the drop down and $1==1$ expression

Screenshot of the Alter Row Transformation settings. It shows the 'Output stream name' set to 'alterRow1', the 'Incoming stream' set to 'select1', and the 'Alter row conditions' dropdown set to 'Upsert if'. The condition '1==1' is entered in the expression field. There are tabs for Alter row settings, Optimize, Inspect, and Data preview.

- Sink Transformation:**

Sink type: Inline

Inline dataset type: Delta

Linked service: ADLS Gen 2 storage account

Sink Settings Errors Mapping Optimize Inspect Data preview ●

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

Description Add sink dataset [Reset](#)

Incoming stream * alterRow

Sink type * Dataset **Inline** Cache

Inline dataset type * Delta

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

Options Allow schema drift [①](#) Validate schema [①](#)

Settings:

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

Folder path * project1 / silver/loans/ [Browse](#)

Compression type No compression

Vacuum ① 0

Table action None Overwrite [①](#) Truncate [①](#)

Update method ① Allow insert Allow delete Allow upsert Allow update

Key columns * ① List of columns Custom expression [①](#)

123 LoanID [+](#) [-](#)

Partition Pruning ①

Data Preview Tab:

		LoanID	CustomerID	LoanAmount	InterestRate	LoanTerm
1	1	45		10000.5	5.5	36
2	2	12		20000.75	4.5	48
3	3	78		15000.0	6.0	60
4	4	34		30000.25	3.5	24
5	5	56		25000.0	5.0	36
6	6	23		17500.5	4.0	48
7	7	89		22500.75	6.5	60
8	8	67		27500.0	3.0	24
9	9	14		32500.25	5.5	36
10	10	92		37500.5	4.5	48
11	11	3		10000.75	6.0	60
12	12	81		20000.0	3.5	24
13	13	29		15000.25	5.0	36

That's it for the silver layer loans' data flow.

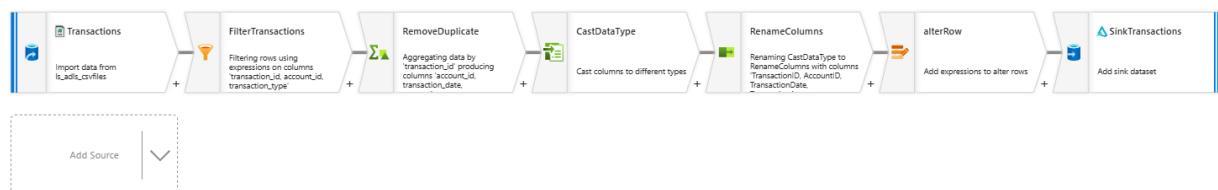
- Transactions File Pipeline

Drag and drop two dataflows in the canvas area



First, we will go through the silver layer in this.

Here is the entire flow:



Source:

Source Settings:

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

Output stream name * [Learn more](#)

Description [Reset](#)

Source type * Dataset **Inline**

Inline dataset type * DelimitedText

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

Skip line count

Sampling * Enable Disable

Source Options:

File path for the transactions.csv file from the bronze folder in ADLS Gen 2

Tick the first row as a header

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

File settings

File mode File Wildcard

File path * / / [Browse](#)

Allow no files found

Change data capture

Compression type

Encoding

Column delimiter

Row delimiter

Quote character

Escape character)

First row as header

Projection: Click on Import Schema

Source settings	Source options	Projection	Optimize	Inspect	Data preview
		Import schema	Clear schema	Schema options	
Column name		↑↓ Type		↑↓ Format	
transaction_id		12s short		Specify format	▼
account_id		12s short		Specify format	▼
transaction_date		date		Specify format	▼
transaction_amount		12 double		Specify format	▼
transaction_type		abc string		Specify format	▼

- **Filter transformation**

Click on the plus icon at the bottom of the source, and take a filter transformation

Use this expression: `!isNull(transaction_id) || !isNull(account_id) || !isNull(transaction_type)`

Filter settings	Optimize	Inspect	Data preview
Output stream name *	FilterTransactions	Learn more	
Description	Filtering rows using expressions on columns 'transaction_id, account_id, transaction_type'	Reset	
Incoming stream *	Transactions		
Filter on *	<pre>!isNull(transaction_id) !isNull(account_id) !isNull(transaction_type)</pre>		

- **Aggregate Transformation (Remove Duplicate):**

Incoming stream *	FilterTransactions
Columns	Name as
12s transaction_id	transaction_id

Incoming stream * FilterTransactions

Group by **Aggregates**

Grouped by: transaction_id

+ Add Clone Delete Open expression builder

Column	Expression
<input type="checkbox"/> Each column that matches name != 'transaction_id'	<input type="text"/> abc <input type="text"/> first(\$\$) ANY

- **Cast Transformation:**

Cast settings Optimize Inspect Data preview ●

Output stream name * CastDataType Learn more ↗

Description Cast columns to different types Reset

Incoming stream * RemoveDuplicate

Columns *

Column name	Type	Format
transaction_date	timestamp	yyyy-MM-ddTHH:mm:ss

Assert type check

- **Select Transformation:**

Rename the column name here

Skip duplicate input columns
 Skip duplicate output columns

Auto mapping Reset Add mapping Delete 5 mappings: All inputs mapped

CastDataType's column	Name as
12s_transaction_id	TransactionID
12s_account_id	AccountID
transaction_date	TransactionDate
12_transaction_amount	TransactionAmount
abc_transaction_type	TransactionType

Data Preview Tab:

↑↓	TransactionID	↑↓	AccountID	↑↓	TransactionDate	↑↓	TransactionAmount	↑↓	TransactionType
+	1		45		2024-01-01 00:00:00.000		100.5		Deposit
+	2		12		2024-01-02 00:00:00.000		200.75		Withdrawal
+	3		78		2024-01-03 00:00:00.000		150.0		Deposit
+	4		34		2024-01-04 00:00:00.000		300.25		Withdrawal
+	5		56		2024-01-05 00:00:00.000		250.0		Deposit
+	6		23		2024-01-06 00:00:00.000		175.0		Withdrawal
+	7		89		2024-01-07 00:00:00.000		225.5		Deposit
+	8		67		2024-01-08 00:00:00.000		275.75		Withdrawal
+	9		14		2024-01-09 00:00:00.000		325.0		Deposit
+	10		92		2024-01-10 00:00:00.000		375.25		Withdrawal
+	11		3		2024-01-11 00:00:00.000		100.5		Deposit
+	12		81		2024-01-12 00:00:00.000		200.75		Withdrawal
+	13		29		2024-01-13 00:00:00.000		150.0		Deposit

- Alter Row Transformation:

Select Upsert if condition from the drop down and $1==1$ expression

The screenshot shows the 'Alter row settings' configuration screen. It includes the following fields:

- Output stream name:** alterRow1
- Description:** Add expressions to alter rows
- Incoming stream:** select1
- Alter row conditions:** Upsert if $1==1$

- Sink:

Sink type: Inline

Inline dataset type: Delta

Linked service: ADLS Gen 2 storage account

Sink Settings Errors Mapping Optimize Inspect Data preview ●

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

Description Add sink dataset [Reset](#)

Incoming stream * alterRow

Sink type * Dataset **Inline** Cache

Inline dataset type * Delta

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

Options

- Allow schema drift ⓘ
- Validate schema ⓘ

Settings:

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

Folder path * project1 / silver/transactions/ [Browse](#)

Compression type No compression

Vacuum ⓘ 0

Table action None Overwrite ⓘ Truncate ⓘ

Update method ⓘ

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

Key columns * ⓘ

- List of columns Custom expression ⓘ

123 TransactionID [+](#) [-](#)

Data Preview Tab:

TransactionID	AccountID	TransactionDate	TransactionAmount	TransactionType
1	45	2024-01-01 00:00:00.000	100.5	Deposit
2	12	2024-01-02 00:00:00.000	200.75	Withdrawal
3	78	2024-01-03 00:00:00.000	150.0	Deposit
4	34	2024-01-04 00:00:00.000	300.25	Withdrawal
5	56	2024-01-05 00:00:00.000	250.0	Deposit
6	23	2024-01-06 00:00:00.000	175.0	Withdrawal
7	89	2024-01-07 00:00:00.000	225.5	Deposit
8	67	2024-01-08 00:00:00.000	275.75	Withdrawal
9	14	2024-01-09 00:00:00.000	325.0	Deposit
10	92	2024-01-10 00:00:00.000	375.25	Withdrawal
11	3	2024-01-11 00:00:00.000	100.5	Deposit
12	81	2024-01-12 00:00:00.000	200.75	Withdrawal

That's it for the silver layer transactions' data flow.

Step 3: Dataflows using SCD Type technique (SCD 1 and SCD 2)

Now, we will design a dataflow for the gold layer for all 5 files.

Select a new dataflow

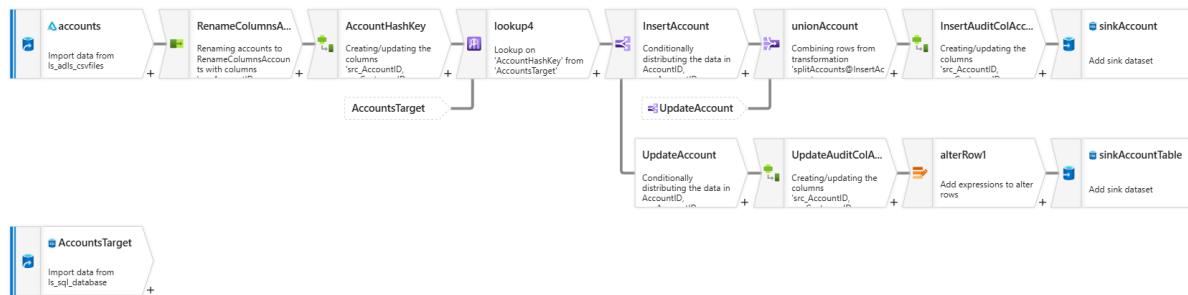
The screenshot shows the Azure Data Factory interface with two data flows:

- DF_Accounts_Silver**: A data flow with a blue icon and a white background.
- DF_Accounts_Gold_SCD2**: A data flow with a blue icon and a white background, containing several transformation steps.

Below the data flows, there is a settings panel with the following configuration:

- Data flow ***: df_gold_accounts_scd2
- Run on (Azure IR) ***: AutoResolveIntegrationRuntime
- Compute size ***: Small

Here is the entire flow of the SCD 2 type accounts file



We will design the accounts file.

- **Source:**

Source settings:

Source settings Source options Projection Optimize Inspect Data preview ●

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

Description Import data from ls_adls_csvfiles [Reset](#)

Source type * Dataset Inline

Inline dataset type * Delta

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

Sampling * Enable Disable

Source options:

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

Folder path * project1 / silver/accounts [Browse](#)

Allow no files found

Compression type No compression

Time travel * Disable Query by timestamp Query by version

Projection:

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

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

Column name	Type
AccountID	12s short
CustomerID	12s short
AccountType	abc string
AccountBalance	1.2 double

- **Select Transformation:**

Use this expression: `concat('src_', $$)`

Output stream name * Learn more

Description

Incoming stream *

Options Skip duplicate input columns Skip duplicate output columns

Input columns * Auto mapping accounts's column Name as 1=1

- **Derived column transformation**

To create Hashkey

Expression used:

```
crc32(concat(toString(src_AccountID),toString(src_CustomerID),src_AccountType,toString(src_AccountBalance)))
```

Derived column's settings Optimize Inspect Data preview

Output stream name * Learn more

Description

Incoming stream *

Add Clone Delete Open expression builder

Columns * Column Expression src_hashkey

Create Target, take source and name it as Accounts Target:

Inline Dataset: Azure SQL Database

Linked Service: Azure SQL Database

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

Output stream name *	AccountsTarget	Learn more ↗
Description	Import data from ls_sql_database	⟳ Reset
Source type *	<input checked="" type="radio"/> Dataset <input type="radio"/> Inline	
Inline dataset type *	<input checked="" type="radio"/> Azure SQL Database	
Linked service *	<input checked="" type="radio"/> ls_sql_database	
Sampling *	<input type="radio"/> Enable <input checked="" type="radio"/> Disable	

Query:

```
SELECT AccountID, HashKey FROM dbo.Accounts where IsActive = 1
```

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

Input	<input type="radio"/> Table <input checked="" type="radio"/> Query <input type="radio"/> Stored procedure
Query *	<pre>SELECT AccountID, HashKey FROM dbo.Accounts where IsActive = 1</pre>
Incremental column	<input type="checkbox"/>
Isolation level	Read uncommitted

Projection:

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

Import schema Clear schema Schema options Overwrite schema

Column name	Type
AccountID	123 integer
HashKey	121 long

- **Lookup transformation:**

Primary Stream: AccountHashkey

Lookup Stream: AccountTarget

Lookup settings Optimize Inspect Data preview ●

Output stream name *	lookup4	Learn more
Description	Lookup on 'AccountHashKey' from 'AccountsTarget'	Reset
Primary stream *	AccountHashKey	
Lookup stream *	AccountsTarget	
Match multiple rows	<input type="checkbox"/> ⓘ	
Match on *	Any row	
Lookup conditions *	Left: AccountHashKey's column Right: AccountsTarget's column 12s src_AccountID = 123 AccountID	
	+	

- **Conditional Split:**

InsertAccount

Condition: `isNull(AccountID)`

UpdateAccount

Condition: `src_AccountID == AccountID && src_hashkey != HashKey`

Conditional split settings Optimize Inspect Data preview ●

Output stream name *	splitAccounts	Learn more
Description	Conditionally distributing the data in AccountID, src_AccountID, AccountID, src_hashkey, HashKey groups, based on	Reset
Incoming stream *	lookup4	
Split on	<input checked="" type="radio"/> First matching condition <input type="radio"/> All matching conditions	
Split condition	Stream names Condition InsertAccount <code>isNull(AccountID)</code> UpdateAccount <code>src_AccountID == AccountID && src_hashkey != HashKey</code>	

Here, we will complete the insert part.



- **Union Transformation:**

Union settings Optimize Inspect Data preview

Output stream name *	unionAccount	Learn more ↗
Description	Combining rows from transformation 'splitAccounts@InsertAccount', 'splitAccounts@UpdateAccount'	
Incoming stream *	splitAccounts@InsertAccount	
Union by * ⓘ	<input checked="" type="radio"/> Name <input type="radio"/> Position	
Union with *	splitAccounts@UpdateAccount + Delete	

- **Derived Column Transformation:**

Derived column's settings Optimize Inspect Data preview

Output stream name *	InsertAuditColAccounts	Learn more ↗												
Description	Creating/updating the columns 'src_AccountID', 'src_CustomerID', 'src_AccountType', 'src_AccountBalance', 'src_createddate', 'src_createdby', 'src_updateddate', 'src_updatedby', 'src_isActive'													
Incoming stream *	unionAccount													
Columns * ⓘ	+ Add Clone Delete Open expression builder <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left; padding: 2px;">Column</th> <th style="text-align: left; padding: 2px;">Expression</th> </tr> </thead> <tbody> <tr> <td style="padding: 2px;"><input type="checkbox"/> src_createddate</td> <td style="padding: 2px;">currentTimestamp()</td> </tr> <tr> <td style="padding: 2px;"><input type="checkbox"/> src_createdby</td> <td style="padding: 2px;">'Harpal'</td> </tr> <tr> <td style="padding: 2px;"><input type="checkbox"/> src_updateddate</td> <td style="padding: 2px;">currentTimestamp()</td> </tr> <tr> <td style="padding: 2px;"><input type="checkbox"/> src_updatedby</td> <td style="padding: 2px;">'Harpal'</td> </tr> <tr> <td style="padding: 2px;"><input type="checkbox"/> src_isActive</td> <td style="padding: 2px;">1</td> </tr> </tbody> </table>		Column	Expression	<input type="checkbox"/> src_createddate	currentTimestamp()	<input type="checkbox"/> src_createdby	'Harpal'	<input type="checkbox"/> src_updateddate	currentTimestamp()	<input type="checkbox"/> src_updatedby	'Harpal'	<input type="checkbox"/> src_isActive	1
Column	Expression													
<input type="checkbox"/> src_createddate	currentTimestamp()													
<input type="checkbox"/> src_createdby	'Harpal'													
<input type="checkbox"/> src_updateddate	currentTimestamp()													
<input type="checkbox"/> src_updatedby	'Harpal'													
<input type="checkbox"/> src_isActive	1													

Column	Expression	Purpose
src_createddate	currentTimestamp()	Set current datetime when a new row is inserted
src_createdby	'Harpal'	Hardcoding the value to who inserted (can replace dynamically too later)
src_updateddate	currentTimestamp()	For new inserted records, updated date = created date (same initially)
src_updatedby	'Harpal'	Hardcoding updated by also (again, can improve later with parameters)
srcIsActive	1	Mark new inserted record as active (important for SCD2 tracking)

- **SinkAccount:**

Sink Settings Errors Mapping Optimize Inspect Data preview ●

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

Description Add sink dataset [Reset](#)

Incoming stream * InsertAuditColAccounts

Sink type * Dataset Inline Cache

Inline dataset type * Azure SQL Database

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

Options Allow schema drift [?](#) Validate schema [?](#)

Settings:

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

Use tempdb

Pre SQL scripts List of scripts Custom expression

Mapping:

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

Options Skip duplicate input columns
 Skip duplicate output columns

Auto mapping Reset | Import schema View schema

10 mappings:

Input columns	Output columns
src_AccountID	AccountID
src_CustomerID	CustomerID
src_AccountType	AccountType
src_AccountBalance	AccountBalance
src_hashkey	HashKey
src_createdate	CreatedDate
src_createdby	CreatedBy
src_updatedate	UpdatedDate
src_updatedby	UpdatedBy
src_isActive	IsActive

Now, let's do update part:



- **Derived column transformation:**

Derived column's settings Optimize Inspect Data preview ●

Output stream name * Learn more

Description

Incoming stream *

Add Clone Delete Open expression builder

Columns * ⓘ

<input type="checkbox"/> Column	<input type="checkbox"/> Expression
<input type="checkbox"/> src_updatedBy	'Harpal-Updated'
<input type="checkbox"/> src_updateDate	currentTimestamp()
<input type="checkbox"/> src_isActive	0

Column	Expression	Purpose
src_updatedBy	'Harpal-Updated'	Mark who updated the record (hardcoded for now, can be dynamic too)
src_updateDate	currentTimestamp()	Capture the update timestamp when record is expired (for SCD2)
src_isActive	0	Mark the old record as inactive (critical for SCD2 design)

- **Alter Row Transformation:**

Alter row settings Optimize Inspect Data preview ●

Output stream name * Learn more

Description

Incoming stream *

Alter row conditions * ⓘ

*

- SinkAccountTable:**

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 *

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

Options Allow schema drift [①](#) Validate schema [①](#)

Settings:

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 [①](#)

Skip duplicate output columns [①](#)

Auto mapping [①](#) [+ Add mapping](#) [Delete](#) [Reset](#) [Import schema](#) [View schema](#)

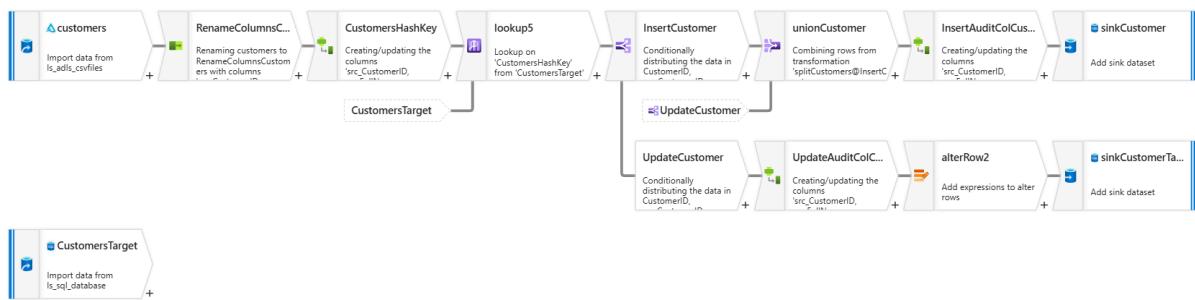
Input columns		Output columns	
<input type="checkbox"/> 123 AccountID	<input type="checkbox"/> 123 AccountID	<input type="checkbox"/> abc UpdatedBy	<input type="checkbox"/> abc UpdatedBy
<input type="checkbox"/> abc src_updatedBy	<input type="checkbox"/> abc src_updatedBy	<input type="checkbox"/> xyz UpdatedDate	<input type="checkbox"/> xyz UpdatedDate
<input type="checkbox"/> xyz src_UpdateDate	<input type="checkbox"/> xyz src_UpdateDate	<input type="checkbox"/> 123 HashKey	<input type="checkbox"/> 123 HashKey
<input type="checkbox"/> 123 IsActive	<input type="checkbox"/> 123 IsActive	<input type="checkbox"/> 123 IsActive	<input type="checkbox"/> 123 IsActive

5 mappings: 5 column(s) from the output schema [①](#)

That's it for the SCD type 2 dataflow design for the accounts file.

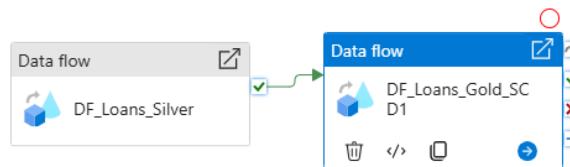
Similarly, we have to do for customers, as we will do 2 files as SCD 2 type and 3 files as SCD 1 type.

Here is the customers' dataflow for reference, we will not cover the steps here as it's similar to accounts (SCD 2 type)



Now, let's do one file as **SCD Type 1**:

Let's take the loans file and design a dataflow for it.



General **Settings** **Parameters** **User properties**

Data flow * Open + New

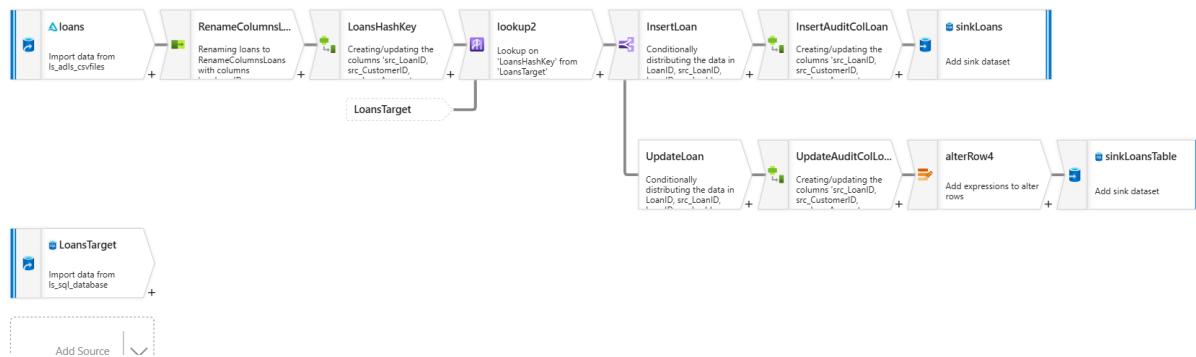
Run on (Azure IR) * AutoResolveIntegrationRuntime

Compute size *

Logging level * Verbose Basic None

Sink properties

Dataflow design:



Let's design one by one transformation:

- **Source:**

Source Settings:

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

Source Options:

Source settings		Source options	Projection	Optimize	Inspect	Data preview
Folder path *	project1 / silver/loans Browse					
Allow no files found	<input type="checkbox"/>					
Compression type	<input type="button"/> No compression					
Time travel *	<input checked="" type="radio"/> Disable <input type="radio"/> Query by timestamp <input type="radio"/> Query by version					

- **Select Transformation**

Expression used:

`concat('src_',$$)`

Select settings Optimize Inspect Data preview

Output stream name * Learn more

Description

Incoming stream *

Options Skip duplicate input columns
 Skip duplicate output columns

Input columns * Auto mapping

Column	Name as
<input type="checkbox"/> loans's column	
<input type="checkbox"/> 1=1	concat['src_,\$\$']

- **Derived Column Transformation:**

`crc32(concat(toString(src_LoanID), toString(src_CustomerID), toString(src_LoanAmount),
toString(src_InterestRate),toString(src_LoanTerm)))`

Derived column's settings Optimize Inspect Data preview

Output stream name * Learn more

Description

Incoming stream *

Add Clone Delete

Columns *

Column	Expression
<input type="checkbox"/> src_hashkey	crc32(concat(toString(src_LoanID), toString(src_C...'))

- **LoanTraget:**

Source settings:

Source settings Source options Projection Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description [Reset](#)

Source type * Dataset Inline

Inline dataset type * Azure SQL Database [Test connection](#)

Linked service * ls_sql_database [Edit](#) [New](#)

Sampling * Enable Disable

Source options:

SELECT LoanID, HashKey FROM dbo.loans

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

Input Table Query Stored procedure

Query * ① [Edit](#)

Incremental column ①

Isolation level ① Read uncommitted

Projection:

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

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

Column name	Type
LoanID	123 integer
HashKey	12l long

- **Lookup settings:**

Primary Stream: LoansHashKey

Lookup Stream: LoansTarget

Lookup settings Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description [Reset](#)

Primary stream *

Lookup stream *

Match multiple rows

Match on *

Lookup conditions *

Left: LoansHashKey's column	Right: LoansTarget's column		
<input type="text" value="12s src_LoanID"/>	<input type="text" value="=="/>	<input type="text" value="123 LoanID"/>	+ Delete

- **Conditional Split:**

Conditional split settings Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description [Reset](#)

Incoming stream *

Split on First matching condition All matching conditions

Split condition

Stream names	Condition
<input type="text" value="InsertLoan"/>	<input type="text" value="isNull(LoanID)"/> Delete Edit
<input type="text" value="UpdateLoan"/>	<input type="text" value="src_LoanID == LoanID && src_hashkey != HashKey"/> Delete Edit

Stream Name	Condition	What it means
InsertLoan	<code>isNull(LoanID)</code>	If the record from the target doesn't exist (LoanID is NULL) → it's a new record → So INSERT it
UpdateLoan	<code>src_LoanID == LoanID && src_hashkey != HashKey</code>	If record exists in target (LoanID matches) but there is some change in data (hash mismatch) → Then Update (Expire old + Insert new)

- **Derived Column (Insert)**

Derived column's settings Optimize Inspect Data preview

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

Description Creating/updating the columns
'src_LoanID, src_CustomerID,
src_LoanAmount, src_InterestRate,
[Reset](#)

Incoming stream * splitLoans@InsertLoan

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

Columns * [①](#)

<input type="checkbox"/> Column	Expression
<input type="checkbox"/> src_createddate	currentTimestamp()
<input type="checkbox"/> src_createdby	'Harpal'
<input type="checkbox"/> src_updateddate	currentTimestamp()
<input type="checkbox"/> src_updatedby	'Harpal'

Column Name	Expression	Meaning
src_createddate	currentTimestamp()	Always puts current time when inserting/updating.
src_createdby	'Harpal'	Hardcoded as "Harpal" for every record.
src_updateddate	currentTimestamp()	Always puts current time again.
src_updatedby	'Harpal'	Hardcoded as "Harpal" for every record.

- **SinkLoans:**

Sink Settings Errors Mapping Optimize Inspect Data preview

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

Description Add sink dataset

[Reset](#)

Incoming stream * InsertAuditColLoan

Sink type *

Dataset	Inline	Cache

Inline dataset type * Azure SQL Database

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

Options

Allow schema drift [①](#)

Validate schema [①](#)

Settings:

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

Use tempdb

Pre SQL scripts List of scripts Custom expression

Mapping:

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

Options skip duplicate input columns
 skip duplicate output columns

Auto mapping

Input columns	Output columns
src_LoanID	LoanID
src_CustomerID	CustomerID
src_LoanAmount	LoanAmount
src_InterestRate	InterestRate
src_LoanTerm	LoanTerm
src_createdby	CreatedBy
src_createddate	CreatedDate
src_updatedby	UpdatedBy
src_updateddate	UpdatedDate
src_hashkey	HashKey

Update Part design:

- **Derived Column(Update):**

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"/> src_updatedBy	'Harpal-Updated'
<input type="checkbox"/> src_updateDate	currentTimestamp()

Column Expression

Column	Expression
src_updatedBy	'Harpal-Updated'
src_updateDate	currentTimestamp()

- **Alter Row Settings:**

Alter row settings

Optimize Inspect Data preview

Output stream name * [Learn more](#)

Description

Incoming stream *

Alter row conditions * [①](#)

* Update if	1==1
-------------	------

- **SinkLoansTable:**

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 *

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

Options Allow schema drift [①](#)

Settings:

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

Schema name * [Refresh](#)

Table name *

Table action None Recreate table Truncate table

Update method [①](#) Allow insert
 Allow delete
 Allow upsert
 Allow update

Skip writing key columns [①](#)

Key columns * [①](#) List of columns Custom expression [①](#)

[+](#) [-](#)

Mapping:

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

Options

- Skip duplicate input columns ⓘ
- Skip duplicate output columns ⓘ

Auto mapping ⓘ

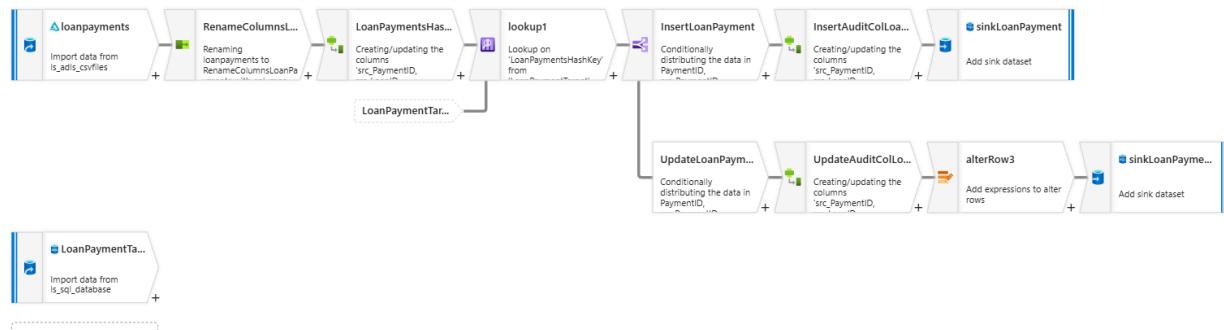
Input columns	Output columns
12s src_LoanID	123 LoanID
12s src_CustomerID	123 CustomerID
1.2 src_LoanAmount	1.2 LoanAmount
1.2 src_InterestRate	1.2 InterestRate
12s src_LoanTerm	123 LoanTerm
abc src_updatedBy	abc UpdatedBy
⌚ src_updateDate	⌚ UpdatedDate
12l src_hashkey	12l HashKey

That's it for the Loans SCD 2 design in the gold layer.

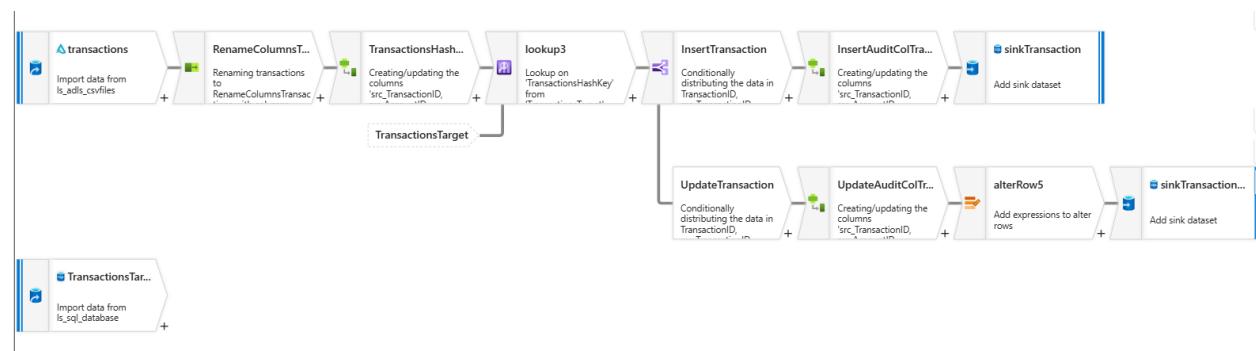
Similarly, we have to design for loan payments and transaction files.

Here I am attaching the flow for reference:

Loan Payments Gold Layer Data Flow:



Transactions Gold Layer Data Flow:



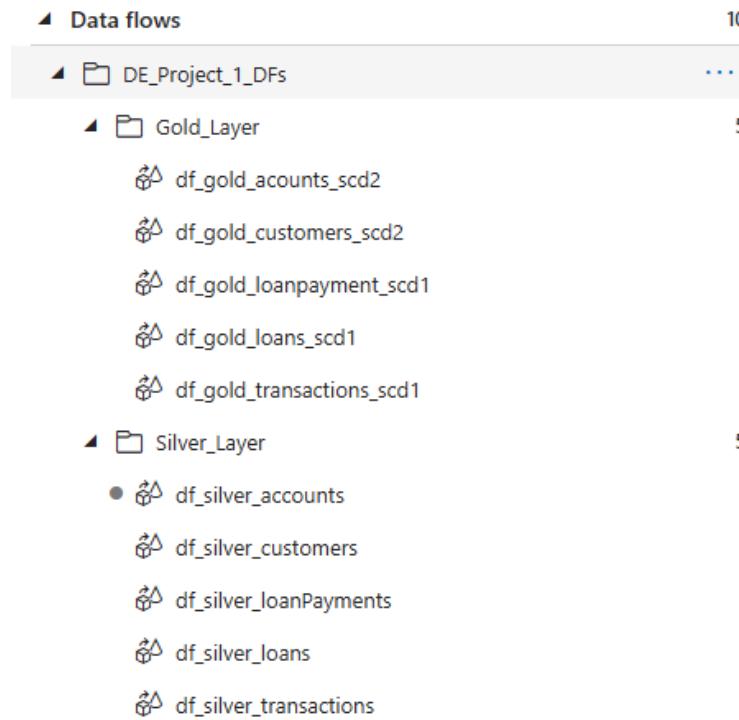
Folder Structure of pipelines:

▲ Pipelines	7
▲ DE_Project_1	7
▲ FilesPipelines	5
① pl_accounts_file	
① pl_customers_file	
① pl_loans_file	
① pl_loans_payments_file	
① pl_transactions_file	
▲ MainPipelines	2
① pl_project1_bronze_layer	
① pl_project1_master	

Folder Structure of the Dataset used:

▲ Datasets	...
① ds_adls	
① ds_adls_csvfiles	
① ds_filesystem_csvfiles	

Folder Structure of all Dataflows used:



Let's run the pipeline:

The screenshot shows the Azure Data Factory pipeline editor interface. The pipeline is named "df_gold_acounts_scd2". The main pane displays the pipeline structure with activities: "Execute Pipeline", "Set variable (SetAllName)", and five "If Condition" blocks. Each "If Condition" block is connected to a specific file: "Accounts File", "Transactions File", "Loans File", "Customers File", and "Loan Payments File". The pipeline status is listed as "Succeeded". Below the pipeline view, a table provides detailed logs for the execution of various activities, including their status, start time, duration, and unique identifiers.

Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime	User prop...	Activity run ID
Loans Pipeline	Succeeded	Execute Pipeline	4/20/2025, 8:24:05 PM	5m 13s			33a744d9-55f4-4f45-b8ef-e5697e
Accounts Pipeline	Succeeded	Execute Pipeline	4/20/2025, 8:24:05 PM	4m 55s			7268e546-32e0-4663-a7af-826162
Loan_Payments Pipeline	Succeeded	Execute Pipeline	4/20/2025, 8:24:05 PM	5m 3s			0e167dc7-f327-4cde-85ce-02ff771
Customers Pipeline	Succeeded	Execute Pipeline	4/20/2025, 8:24:05 PM	5m 17s			b42c02cb-cd8b-44db-85ad-b92c0
Transactions Pipeline	Succeeded	Execute Pipeline	4/20/2025, 8:24:05 PM	5m 5s			40bd7bce-4ec4-49bb-9479-86fb3
Accounts File	Succeeded	If Condition	4/20/2025, 8:24:05 PM	4m 56s			8a91ca31-f5d8-40c3-8e92-1051f7
Loan Payments File	Succeeded	If Condition	4/20/2025, 8:24:05 PM	5m 4s			ee96b96f-fa11-47b9-8fed-fe4a6ff
Loans File	Succeeded	If Condition	4/20/2025, 8:24:05 PM	5m 15s			65ebfb1f-1903-407c-b80d-f83a42
Customers File	Succeeded	If Condition	4/20/2025, 8:24:05 PM	5m 18s			a883df19-81d1-4d91-b8ab-60c29
Transactions File	Succeeded	If Condition	4/20/2025, 8:24:05 PM	5m 6s			db3f2c1c-28e9-4da0-b9ef-196717
SetAllName	Succeeded	Set variable	4/20/2025, 8:24:04 PM	Less than 1s			68ba15dc-8fab-41b4-be4a-cbfa6a
Bronze Layer	Succeeded	Execute Pipeline	4/20/2025, 8:23:02 PM	1m 2s			c695ca89-9715-40dc-a657-e9e69c

Check Bronze Layer Files in ADLS Gen 2 storage account:

Authentication method: Access key ([Switch to Microsoft Entra user account](#))

Location: project1 / bronze

Search blobs by prefix (case-sensitive)

Name	
<input type="checkbox"/> accounts.csv	▲
<input type="checkbox"/> customers.csv	▲
<input type="checkbox"/> loan_payments.csv	▲
<input type="checkbox"/> loans.csv	▲
<input type="checkbox"/> transactions.csv	▲

Silver Layer folder in ADLS Gen 2 Storage account:

Upload Add Directory Refresh Rename Delete Change tier

Authentication method: Access key ([Switch to Microsoft Entra user account](#))

Location: project1 / silver

Search blobs by prefix (case-sensitive)

Name	
<input type="checkbox"/> accounts	▲
<input type="checkbox"/> customers	▲
<input type="checkbox"/> loanpayments	▲
<input type="checkbox"/> loans	▲
<input type="checkbox"/> transactions	▲

Check account folder, its parquet file as we have selected delta format.

Authentication method: Access key ([Switch to Microsoft Entra user account](#))

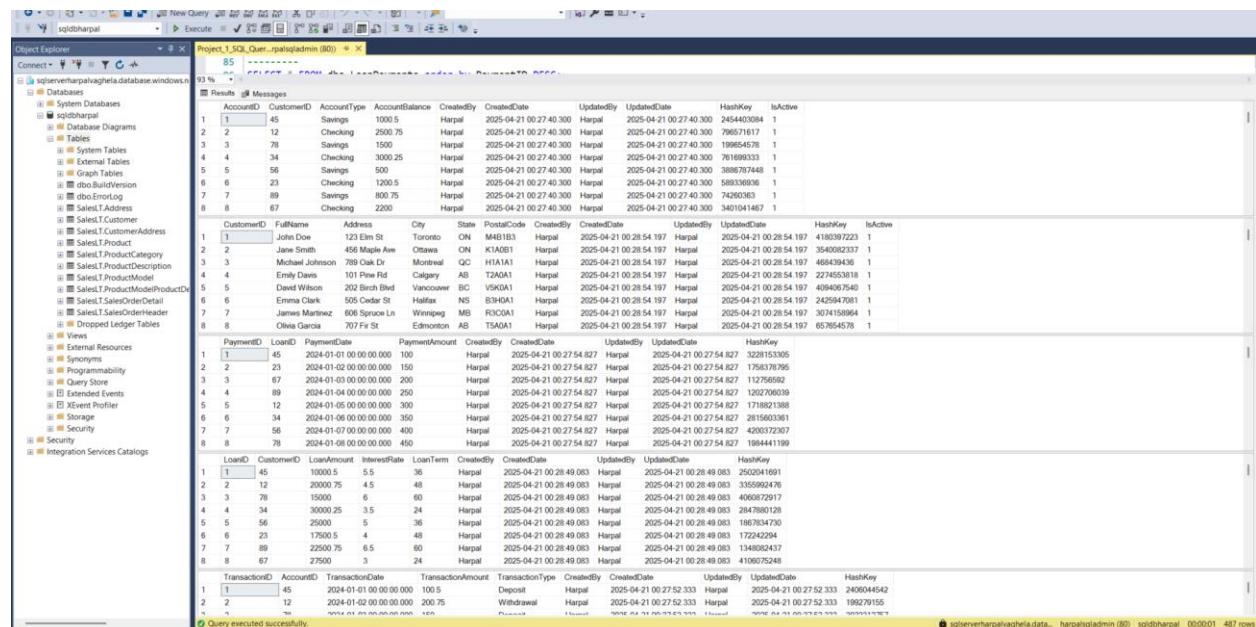
Location: project1 / silver / accounts

Search blobs by prefix (case-sensitive)

Name

-  _delta_log
-  part-00000-23b90dd2-06a0-4b1c-9e77-cb8e01d4dc3f-c000.snappy.parquet

Check the output in SSMS (Azure SQL Database Tables):



CustomerID	FirstName	LastName	Address	City	State	PostalCode	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey	IsActive
1	John	Doe	123 Elm St	Toronto	ON	M4B183	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	245440084	1
2	Jane	Smith	456 Maple Ave	Ottawa	ON	K1A0B1	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	796571617	1
3	Michael	Johnson	789 Oak Dr	Montreal	QC	H1A1A1	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	199865478	1
4	Emily	Davis	101 Pine Rd	Calgary	AB	T2A0A1	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	761699333	1
5	David	Wilson	202 Birch Blvd	Vancouver	BC	V5K0A1	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	2023077448	1
6	Emma	Clark	505 Cedar St	Halifax	NS	B3H0A1	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	242947081	1
7	James	Martinez	606 Spruce Ln	Winnipeg	MB	R3C0A1	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	3074059864	1
8	Olivia	Garcia	701 Fr St	Edmonton	AB	T5K0A1	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	857654578	1

PaymentID	LoanID	PaymentDate	PaymentAmount	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey	IsActive
1	1	2024-01-01 00:00:00 100	100	Harpal	2025-04-21 00:27:54.197	Harpal	2025-04-21 00:27:54.197	418020723	1
2	2	2024-01-01 00:00:00 150	150	Harpal	2025-04-21 00:27:54.197	Harpal	2025-04-21 00:27:54.197	354092337	1
3	3	2024-01-01 00:00:00 200	200	Harpal	2025-04-21 00:27:54.197	Harpal	2025-04-21 00:27:54.197	466439436	1
4	4	2024-01-01 00:00:00 250	250	Harpal	2025-04-21 00:27:54.197	Harpal	2025-04-21 00:27:54.197	2274553818	1
5	5	2024-01-01 00:00:00 300	300	Harpal	2025-04-21 00:27:54.197	Harpal	2025-04-21 00:27:54.197	4094607540	1
6	6	2024-01-01 00:00:00 350	350	Harpal	2025-04-21 00:27:54.197	Harpal	2025-04-21 00:27:54.197	242947081	1
7	7	2024-01-01 00:00:00 400	400	Harpal	2025-04-21 00:27:54.197	Harpal	2025-04-21 00:27:54.197	3074059864	1
8	8	2024-01-01 00:00:00 450	450	Harpal	2025-04-21 00:27:54.197	Harpal	2025-04-21 00:27:54.197	857654578	1

TransactionID	AccountId	LoanAmount	InterestRate	LoanTerm	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey	IsActive
1	1	1000.5	5.5	36	Harpal	2025-04-21 00:28:49.083	Harpal	2025-04-21 00:28:49.083	2520241691	1
2	2	2000.75	4.5	48	Harpal	2025-04-21 00:28:49.083	Harpal	2025-04-21 00:28:49.083	335599247	1
3	3	1500.6	6	60	Harpal	2025-04-21 00:28:49.083	Harpal	2025-04-21 00:28:49.083	40060072917	1
4	4	3000.25	3.5	24	Harpal	2025-04-21 00:28:49.083	Harpal	2025-04-21 00:28:49.083	2847980128	1
5	5	2500.5	5	36	Harpal	2025-04-21 00:28:49.083	Harpal	2025-04-21 00:28:49.083	1887634720	1
6	6	1750.5	4	48	Harpal	2025-04-21 00:28:49.083	Harpal	2025-04-21 00:28:49.083	17224294	1
7	7	2250.05	4.5	60	Harpal	2025-04-21 00:28:49.083	Harpal	2025-04-21 00:28:49.083	1340002457	1
8	8	2750.0	3	24	Harpal	2025-04-21 00:28:49.083	Harpal	2025-04-21 00:28:49.083	410675248	1

Now, let's suppose the client updates a few records and inserts a new record in the files below on the next day.

Customer Data Updated by client:

79	78	Abigail	Cole	7777 Fir St	Sundridge	ON	P0A0A1
30	79	James	West	7878 Redwood Dr	South River	ON	P0A0A1
31	80	Emily	Jordan	7979 Cypress Ave	North Bay	ON	P1B0A1
32	81	Michael	Owens	8080 Willow Rd	Mattawa	ON	P0H0A1
33	82	Elizabeth	Reynolds	8181 Poplar St	Sturgeon Falls	ON	P2B0A1
34	83	David	Fisher	8282 Ash Blvd	Verner	ON	P0H0A1
35	84	Sophia	Ellis	8383 Beech Dr	Field	ON	P0H0A1
36	85	John	Harrison	8484 Cedar Ln	Temagami	ON	P0H0A1
37	86	Olivia	Gibson	8585 Elm St	New Liskeard	ON	P0J0A1
38	87	William	McDonald	11 Maple Ave	NorthYork		
39	88	Harpal	Vaghela	Durham St	Oshawa	ON	L1J 5R3
40							
41							
42							
43							
44							
45							

Account Data Updated by client

95	94	39	Checking	9500.5
96	95	60	Savings	925.75
97	96	48	Checking	9700
98	97	90	Savings	950.25
99	98	49	Checking	9900.5
100	99	80	Savings	975.75
101	100	50	Checking	1111
102	101	51	Checking	1111
103				
104				
105				
106				
107				
108				

Run the pipeline, or if we have created a trigger it will run automatically on next day at 8 AM (as we have set 8:00 AM trigger timing.)

Pipeline run ID: 147c17a9-8234-496b-89f2-6102c7e71612

Pipeline status: Succeeded

Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime	User prop...	Activity run ID
Customers Pipeline	Succeeded	Execute Pipeline	4/20/2025, 8:34:38 PM	5m 28s			ba80d4f3-57de-4a95-9a72-21340e
Loan Payments Pipeline	Succeeded	Execute Pipeline	4/20/2025, 8:34:38 PM	5m 15s			1b8fb0fa-4c98-4fac-b404-87fae2d
Accounts Pipeline	Succeeded	Execute Pipeline	4/20/2025, 8:34:38 PM	3m 34s			ea2b2744-e2ee-4026-893f-119fc5
Transactions Pipeline	Succeeded	Execute Pipeline	4/20/2025, 8:34:38 PM	3m 57s			7c7d00c3-daf4-43aa-8db5-07da25
Loans Pipeline	Succeeded	Execute Pipeline	4/20/2025, 8:34:38 PM	5m 8s			80b4cfaf-0def-48a4-ae83-8f4ced8
Loans File	Succeeded	If Condition	4/20/2025, 8:34:37 PM	5m 9s			558770ad-0d07-4f7f-8684-53465f
Accounts File	Succeeded	If Condition	4/20/2025, 8:34:37 PM	3m 35s			0b2429e2-7f79-4bca-8846-b3cf8d
Customers File	Succeeded	If Condition	4/20/2025, 8:34:37 PM	5m 29s			6860d7c7-a1c4-42f0-b9f5-091972
Transactions File	Succeeded	If Condition	4/20/2025, 8:34:37 PM	3m 58s			2d2e5997-e0f2-46cc-929c-2a7086
Loan Payments File	Succeeded	If Condition	4/20/2025, 8:34:37 PM	5m 16s			7254554f-1fd9-449f-b9f8-afcd59t
SetAllName	Succeeded	Set variable	4/20/2025, 8:34:37 PM	Less than 1s			31dad324-338c-4265-a56b-47e2et
Bronze Layer	Succeeded	Execute Pipeline	4/20/2025, 8:33:27 PM	1m 9s			52393eb8-75ae-4613-a416-2ddfd3

Output in SSMS:

	Results											Messages	
	AccountID	CustomerID	AccountType	AccountBalance	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey	IsActive			
1	101	51	Checking	1111	Harpal	2025-04-21 00:37:41.347	Harpal	2025-04-21 00:37:41.347	1516189302	1			
2	100	50	Checking	1111	Harpal	2025-04-21 00:37:41.347	Harpal	2025-04-21 00:37:41.347	1677862621	1			
3	100	50	Checking	10100	Harpal	2025-04-21 00:27:40.300	Harpal-Updated	2025-04-21 00:37:57.397	3183769099	0			
4	99	80	Savings	975.75	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	3104537685	1			
5	98	49	Checking	9900.5	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	3893038149	1			
6	97	90	Savings	950.25	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	270206465	1			
7	96	48	Checking	9700	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	3750651349	1			
8	95	60	Savings	925.75	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	1002059928	1			
9	94	39	Checking	9500.5	Harpal	2025-04-21 00:27:40.300	Harpal	2025-04-21 00:27:40.300	1159360084	1			
	CustomerID	FullName	Address	City	State	PostalCode	CreatedBy	CreatedDate	UpdatedBy	UpdatedDate	HashKey	IsActive	
1	88	Harpal Vaghela	Durham St	Oshawa	ON	L1J 5R3	Harpal	2025-04-21 00:39:24.650	Harpal	2025-04-21 00:39:24.650	2511028016	1	
2	87	William McDonald	11 Maple Ave	North York	Unknown	Unknown	Harpal	2025-04-21 00:39:24.650	Harpal	2025-04-21 00:39:24.650	3581366593	1	
3	87	William McDonald	8686 Maple Ave	Haileybury	Unknown	Unknown	Harpal	2025-04-21 00:28:54.197	Harpal-Updated	2025-04-21 00:39:52.387	2616928843	0	
4	86	Olivia Gibson	8585 Elm St	New Liskeard	ON	P0J0A1	Harpal	2025-04-21 00:28:54.197	Harpal	2025-04-21 00:28:54.197	1323039564	1	
5	85	John Harrison	8484 Cedar Ln	Temagami	ON	P0H0A1	Harpal	2025-04-21 00:28:54.197	Harpal	2025-04-21 00:28:54.197	170127391	1	
6	84	Sophia Ellis	8383 Beech Dr	Field	ON	P0H0A1	Harpal	2025-04-21 00:28:54.197	Harpal	2025-04-21 00:28:54.197	2900792910	1	
7	83	David Fisher	8282 Ash Blvd	Verner	ON	P0H0A1	Harpal	2025-04-21 00:28:54.197	Harpal	2025-04-21 00:28:54.197	3958902989	1	
8	82	Elizabeth Reyno...	8181 Poplar St	Sturgeon F...	ON	P2B0A1	Harpal	2025-04-21 00:28:54.197	Harpal	2025-04-21 00:28:54.197	1762925461	1	
9	81	Michael Owens	8080 Willow Rd	Mattawa	ON	P0H0A1	Harpal	2025-04-21 00:28:54.197	Harpal	2025-04-21 00:28:54.197	3753478506	1	
10	80	Emily Jordan	7979 Cypress...	North Bay	ON	P1B0A1	Harpal	2025-04-21 00:28:54.197	Harpal	2025-04-21 00:28:54.197	2185190130	1	
11	79	James West	7878 Redwoo...	South River	ON	P0A0A1	Harpal	2025-04-21 00:28:54.197	Harpal	2025-04-21 00:28:54.197	4091079027	1	
12	78	Ahinsul Cole	7777 Fir St	Sundridge	ON	P0A0A1	Harpal	2025-04-21 00:28:54.197	Harpal	2025-04-21 00:28:54.197	4080494707	1	

We can create a trigger on the main pipeline by clicking on this -> New/Edit

The screenshot shows the 'New trigger' configuration dialog. At the top, there are tabs for 'Validate', 'Debug' (with a dropdown arrow), 'Trigger (1)', and 'Data flow debug'. Below the tabs, there's a section for 'Trigger now' and a link 'New/Edit (1)'. The main form fields include:

- Name ***: Project_1_Trigger_Everyday_8_AM
- Description**: (empty text area)
- Type ***: Schedule
- Start date * ①**: 4/21/2025, 12:41:48 AM
- Time zone * ①**: Eastern Time (US & Canada) (UTC-5)

① This time zone observes daylight savings. Trigger will auto-adjust for one hour difference.
- Recurrence * ①**: Every 1 Day(s)

① This time zone observes daylight savings. Trigger will auto-adjust for one hour difference.
- Advanced recurrence options**:
 - Execute at these times ①**:
 - Hours: 8 (X)
 - Minutes: 0 (X)
 - Schedule execution times**: 08:00
 - Specify an end date**: checked
- End On * ①**: 4/22/2025, 12:41:48 AM
- Annotations**: + New

At the bottom are 'OK' and 'Cancel' buttons.

We can see the trigger has been created in the running in the manage section of Azure Data Factory:

Triggers

To execute a pipeline set the trigger. Triggers represent a unit of processing that determines when a pipeline execution needs to be kicked off.

+ New	Refresh				
Filter by name		Annotations : Any			
Showing 1 - 1 of 1 items					
Name ↑↓	Type ↑↓	Status ↑↓	Related ↑↓	Annotations ↑↓	
● Project_1_Trigger_Everyday_8_AM	Schedule	Started	1		

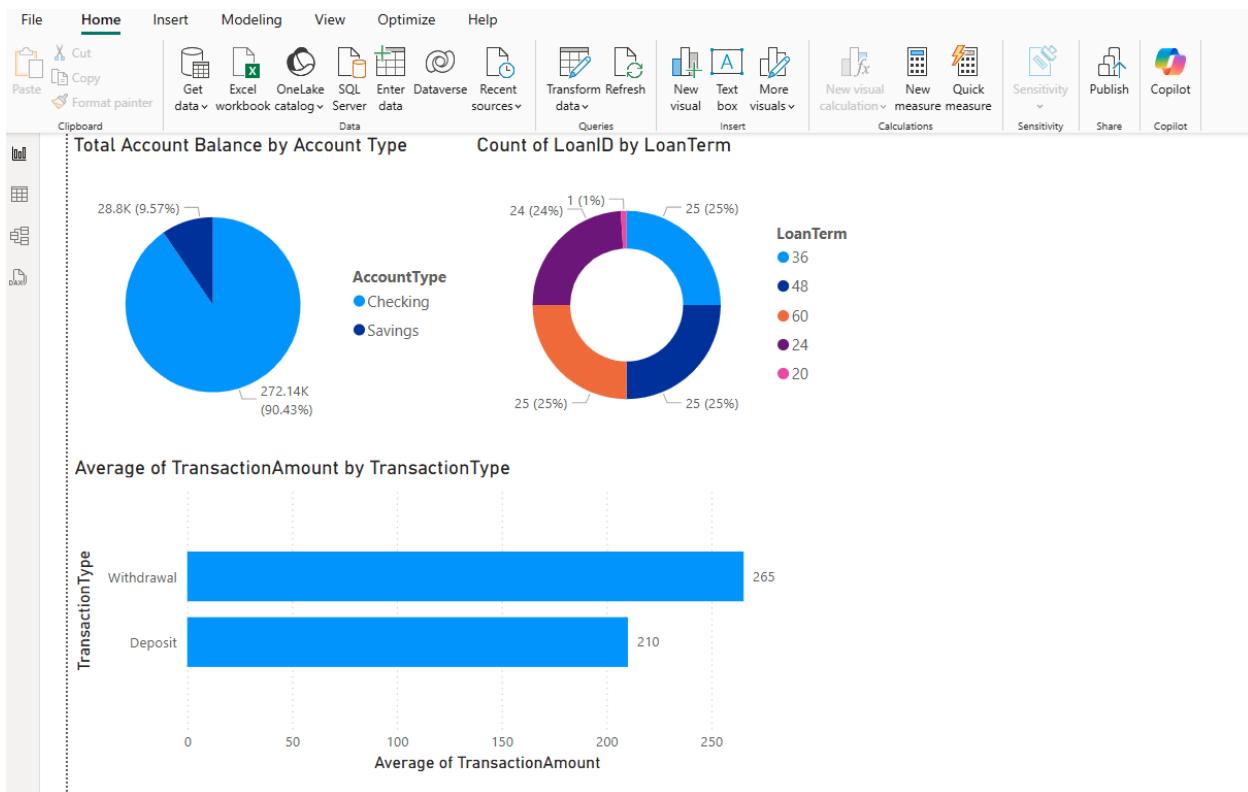
Step 4: Use Power BI for Data Visualization

Select Import from Azure SQL Database

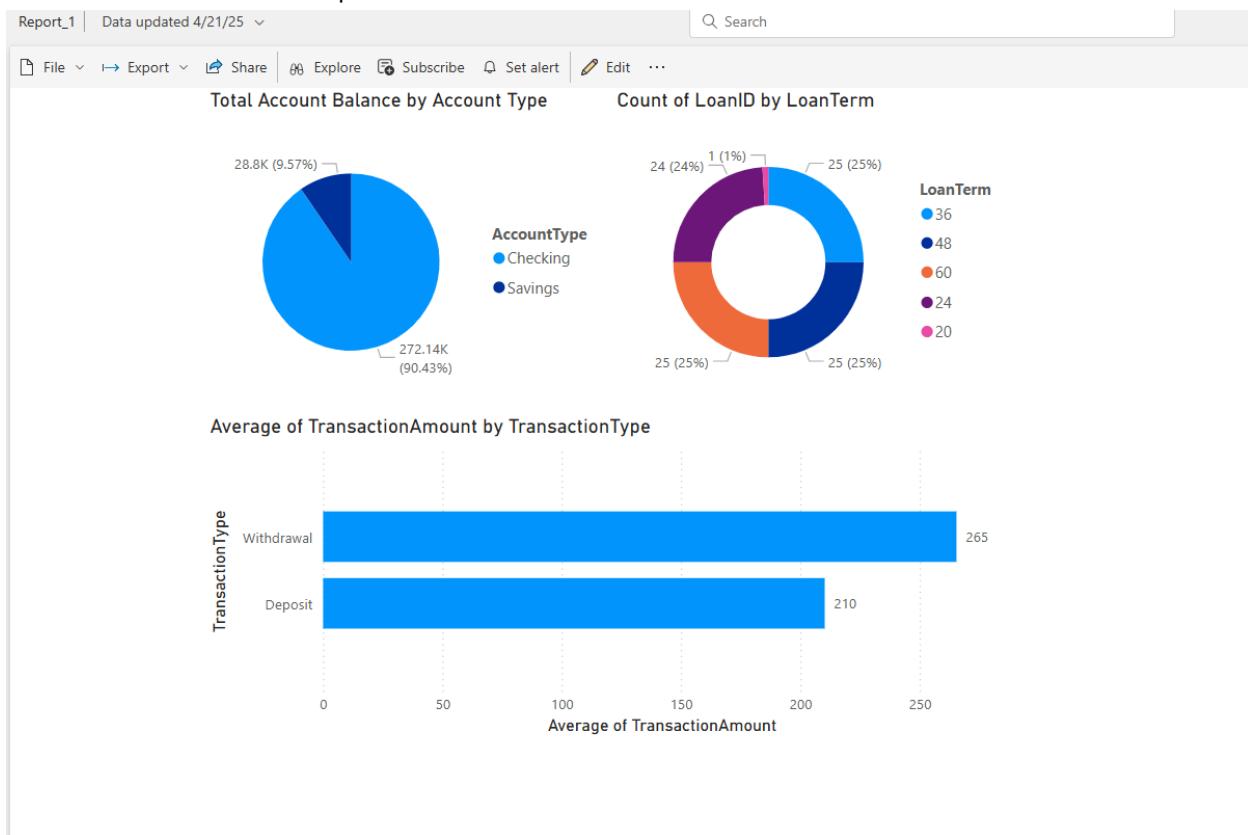
Give the Username and password

Select 5 tables that we need to make a visualization on

The screenshot shows the Microsoft Power BI desktop application. The ribbon menu is visible at the top with options like Home, Insert, Modeling, View, Optimize, Help, and various data import and transformation tools. On the left, the Navigator pane lists available tables in the data source, including SalesLT.ProductAndDescription, SalesLT.ProductModelCatalogDescription, sysdatabase_firewall_rules, Accounts, BuildVersion, Customers, ErrorLog, LoanPayments, Loans, SalesLT.Address, SalesLT.Customer, SalesLT.CustomerAddress, SalesLT.Product, SalesLT.ProductCategory, SalesLT.ProductDescription, SalesLT.ProductModel, SalesLT.ProductModelProductDescription, SalesLT.SalesOrderDetail, SalesLT.SalesOrderHeader, and Transactions. A specific table, 'Transactions', is selected and highlighted in blue. A data preview window is open, showing columns TransactionID, AccountID, TransactionDate, TransactionAmount, and Tran. The preview contains 73 rows of transaction data. To the right of the preview, the 'Visualizations' pane is open, showing various chart and report templates. At the bottom of the preview window, there are buttons for 'Load', 'Transform Data', and 'Cancel'.



Publish it to the Fabric Workspace



Step 5: GitHub Integration

Project Link: [harpalvaghela/DE-Project-1](https://github.com/harpalvaghela/DE-Project-1)

The screenshot shows the GitHub repository 'DE-Project-1' which is public. It has 1 branch and 0 tags. The commit history shows 17 commits from 'harpalvaghela' dated 4-21-2025. The commits are listed below:

- adf-harpalvaghela: ARM template and parameters deployed on 4-21-2025 1:43:... yesterday
- dataflow: Updating trigger: Project_1_Trigger_Everyday_8_AM yesterday
- dataset: Updating pipeline: pl_project1_master 10 hours ago
- factory: Adding linkedService: ls_filesystem_csvfiles yesterday
- integrationRuntime: Adding linkedService: ls_filesystem_csvfiles yesterday
- linkedService: Adding linkedService: ls_filesystem_csvfiles yesterday
- pipeline: Updating pipeline: pl_project1_master 10 hours ago
- trigger: Adding linkedService: ls_filesystem_csvfiles yesterday
- Project_1_Diagram.pdf**: (no date)
- Project_1_SQL_Queries.sql**: SQL Queries (Gold Layer) 2 minutes ago
- README.md**: Update README.md yesterday
- publish_config.json**: Update publish_config.json yesterday

The 'README' file contains the following content:

```
Project 1 – Data Pipeline for Customer Account Analysis
Objective Build a robust Azure Data Factory (ADF) pipeline to process customer account data.
```

Step 6: Conclusion

This project successfully demonstrates a real-world Data Engineering Pipeline on Azure Data Factory, handling ingestion, transformation (SCD 1 & SCD 2), and loading into Azure SQL DB, all automated and optimized. The smart use of dynamic variables, triggers, parallel pipelines, and delta formats showcases enterprise-grade best practices. With Power BI and GitHub integration, this project forms a complete end-to-end modern data platform solution ready for real-world production environments.

Points to remember

- Self-Hosted Integration Runtime: It allows secure copying of local on-premises files (CSV) into Azure without exposing public endpoints.

- Get Metadata to Fetch Last Modified: Used to check which files were recently modified, helping decide which files need to be processed.
- IF Condition for Data Freshness: Ensures that only new or updated files trigger a copy operation, saving compute and storage costs.
- Dual Copy Activities in Bronze Layer: One copy moves files into the Bronze layer, another takes a backup with timestamped filenames for audit purposes.
- Dynamic File Path with Parameters: Reduces hardcoding, making the pipeline flexible even if new files are added later.
- Append Variable inside ForEach: Smartly collects all modified filenames into a list for decision-making later in the master pipeline.
- Return Value from Child Pipeline: Passes back which files were touched so that the master pipeline can act only on relevant datasets.
- Master Pipeline Orchestration: Central controller that manages the full lifecycle from Bronze → Silver → Gold in the correct sequence.
- Dedicated Pipelines for Each File (Parallelism Ready): Designing separate pipelines for each dataset allows running them independently and faster.
- 5 IF Conditions for File-Specific Processing: Instead of running everything blindly, each IF checks if its related file was modified, improving efficiency.
- Delta Format in Silver Layer: Provides faster reads and updates compared to CSV and supports ACID transactions for analytics workloads.
- Filter, Remove Duplicate, and Derived Columns: Enforces basic data quality and enrichment before moving data further downstream.
- Hash Key Creation for SCD Operations: Hashes multiple fields to create a unique fingerprint of the row, enabling accurate change detection.
- SCD Type 1 Handling (Overwrite Updates): Simply updates existing records without tracking history when a change is detected.
- SCD Type 2 Handling (Maintain History): Marks old records as inactive and inserts new records, preserving the historical state of the data.
- Audit Columns Automation (CreatedBy, UpdatedBy): Dynamically tracks who inserted or updated data along with timestamps, to ensure good governance.
- Alter Row Transformation for Row-Level Control: Based on conditions, controls whether a row should be inserted, updated, or ignored during sink operations.
- Use TempDB for Small Interim Operations: For smaller data loads, using SQL Server Tempdb speeds up processing without physically creating tables.
- GitHub Integration for Version Control: All pipeline artifacts (datasets, dataflows, pipelines) are version-controlled and backed up automatically.
- Daily Trigger for Full Automation: A scheduled trigger at 8:00 AM ensures the pipeline executes automatically without human intervention.