

# Incremental Data Loading and Automated Notifications using Microsoft Fabric

—Sudhanshu Kharbanda

## Introduction

Modern organizations rely on timely and accurate data to drive decision-making, yet integrating information from diverse on-premises sources into centralized analytics platforms remains a challenge. Issues such as data quality, transformation consistency, and the need for automation highlight the importance of efficient end-to-end data pipelines. Microsoft Fabric provides a unified environment that addresses these challenges by combining ingestion, storage, transformation, and monitoring capabilities within a single platform.

This project demonstrates the development of an automated data pipeline on Microsoft Fabric using the AI Bank Dataset. The pipeline securely ingests structured data through the On-Prem Gateway, applies cleansing and transformations with Dataflow Gen1, and loads the results into a Fabric Warehouse. Slowly Changing Dimension (SCD) Type 1 logic is implemented via Fabric Notebooks to maintain data consistency, while automated scheduling and email notifications ensure reliable monitoring and stakeholder updates. The solution highlights Fabric's ability to streamline data workflows and deliver scalable, analytics-ready datasets.

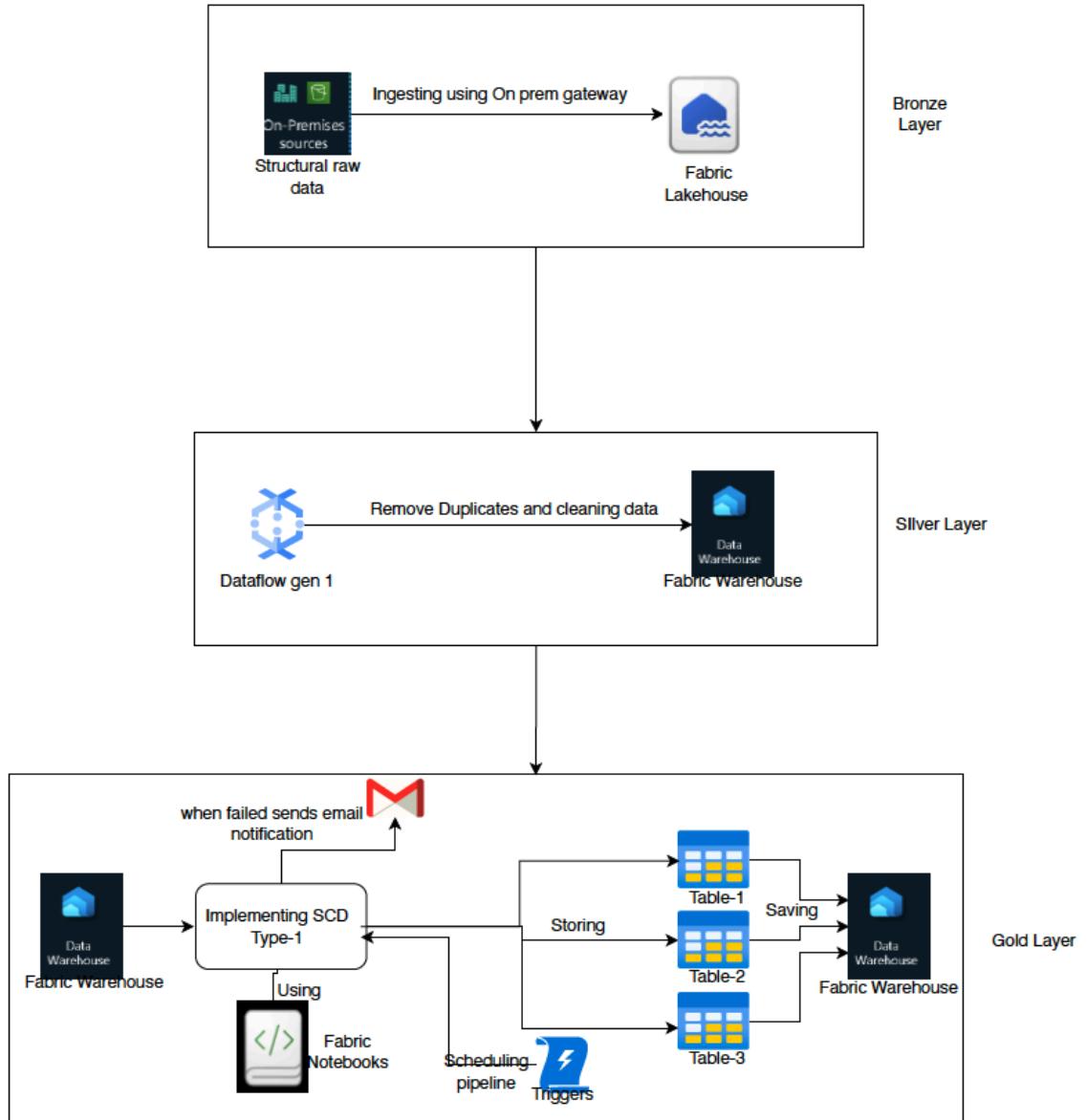
## Specifications

The primary objective of this project is to design and implement an end-to-end automated data pipeline on Microsoft Fabric that demonstrates secure ingestion, transformation, and monitoring of structured data. Specifically, the project aims to:

1. Ingest structured data from on-premises environments into a Fabric Lakehouse using the On-Prem Gateway.
2. Utilize the AI Bank Dataset as the source for pipeline implementation.
3. Apply data cleansing operations—such as joins, duplicate removal, and formatting—using Dataflow Gen1.
4. Load the cleansed data into a Fabric Warehouse for downstream analytics.
5. Implement Slowly Changing Dimension (SCD) Type 1 logic with Fabric Notebooks to maintain data consistency in dimension tables.

6. Schedule and monitor the pipeline while enabling automated email notifications to fail execution of the pipeline.

## Architecture Diagram



Click on the entralID for provide the permission for enabling fabric for the account.

The screenshot shows the Azure Active Directory Users page. The left sidebar includes options like All users, Audit logs, Sign-in logs, Diagnose and solve problems, Deleted users, Password reset, User settings, Bulk operation results, and Bulk operation results (Preview). The main area displays a table with columns: Display name, User principal name, User type, On-premises sync status, Identities, Company name, and Creation type. Three users are listed: Roman Syed (Member, No, identities: jyotikharbanda23@gmail.com), Sudhanshu (Member, No, identities: jyotikharbanda23@gmail.com), and Syed Hasan (Member, No, identities: MicrosoftAccount). Sudhanshu's row is highlighted with a blue background, indicating it is selected.

Click on the assigned roles and add the fabric administrator role to the user

The screenshot shows the Azure Active Directory User Overview page for Sudhanshu. The left sidebar lists options like Overview, Audit logs, Sign-in logs, Diagnose and solve problems, Custom security attributes, Assigned roles, Administrative units, Groups, Applications, Licenses, Devices, Azure role assignments, Authentication methods, and New support request. The main area has tabs for Overview, Monitoring, and Properties. Under the Overview tab, there is a 'Basic info' section with Sudhanshu's profile picture, name, email (sudkhar@jyotikharbanda23@gmail.onmicrosoft.com), and member status. Below this are sections for User principal name, Object ID, Created date time, User type, Identities, and Agent ID. The 'Assigned roles' section shows 1 assigned role. At the bottom, there are 'My Feed' and 'Quick actions' sections, and a 'Quick actions' button.

Go to Azure and search for the fabric capacity. And create the fabric capacity.

[Home](#) > [sudhanshuboot](#) > [Marketplace](#) > [Microsoft Fabric](#) >

## Create Fabric capacity



**Welcome to Microsoft Fabric**

Fabric delivers an end-to-end analytics platform from the data lake to the business user.

[Find out more](#)

**\*Basics** Tags Review + create

Create Fabric capacity that you can use with your Fabric workspaces.

**Project details**

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all of your resources.

Subscription *	Azure subscription 1
Resource group *	sudhanshuboot
	<a href="#">Create new</a>

**Capacity details**

Name your Capacity and select a location.

Capacity name *	sudfabric
Region *	(Canada) Canada Central
Size	<b>F2</b> 2 Capacity units <a href="#">Change size</a>
Fabric capacity administrator *	sudkhar@jyotikharbanda233gmail.onmicrosoft.com
	<a href="#">Select</a>

---

[Review + create](#) [< Previous](#) [Next: Tags >](#)

The screenshot shows the Microsoft Azure portal interface. On the left, there's a sidebar with various options like Overview, Audit logs, Sign-in logs, Diagnose and solve problems, Custom security attributes, Assigned roles (which is selected), Administrative units, Groups, Applications, Licenses, Devices, Azure role assignments, Authentication methods, and New support request. The main content area shows 'Sudhanshu | Assigned roles' with a search bar, 'Add assignments', 'Remove assignments', 'Refresh', and 'Got feedback?' buttons. Below this is a section titled 'Administrative roles' with a note: 'Administrative roles can be used to grant access to Microsoft Entra ID and other Microsoft services.' It includes a search bar for 'Search by name or description', 'Add filters' button, and a table with columns 'Role' and 'Description'. The table is empty with the message 'No directory roles assigned.' To the right, there's a 'Copilot' button and a 'Directory roles' panel. This panel has a note: 'To assign custom roles to a user, your organization needs Microsoft Entra ID Premium P1 or P2.' It says 'Choose admin roles that you want to assign to this user' with a 'Learn more' link. A search bar shows 'fabric', a 'Role' dropdown set to 'Fabric Administrator', and a 'Description' field containing 'Can manage all aspects of Microsoft Fabric.'. An 'Add' button is at the bottom of the panel.

Create a container in the adls gen2 and upload the following csv files into it.

The screenshot shows the Azure Storage Explorer interface. The left sidebar lists 'Containers' and 'fabricdatapro' (Container). Under 'Overview', there are links for 'Diagnose and solve problems', 'Access Control (IAM)', and 'Settings'. The main content area shows a table of blobs. The table has columns: Name, Last modified, Access tier, Blob type, Size, and Lease state. The blobs listed are: accounts.csv, customers.csv, loan\_payments.csv, loans.csv, and transactions.csv. Each blob has a timestamp of '9/4/2025, 1:47:33 AM' and an 'Inferred' access tier. The blob types are 'Block blob', sizes range from 2.28 KiB to 3.43 KiB, and lease states are 'Available'. There are three-dot ellipsis icons next to each row.

Code for creating a watermark table for incremental loading

```
Create table watermarkDemo
(
    id INT,
    tablename varchar(100),
```

```

schemaname varchar(50) ,
filename varchar(20) ,
foldername varchar(30) ,
lastprocessedvalue varchar(20) ,
incrementalcolumn varchar(20)

);

insert into watermarkDemo values
(1,'accounts','dbo','accounts','bootcamp','0','accounts')

insert into watermarkDemo values
(2,'customers','dbo','customers','bootcamp','0','customers')

insert into watermarkDemo values
(3,'loan_payments','dbo','loan_payments','bootcamp','0','loan_payments')

insert into watermarkDemo values
(4,'loans','dbo','loans','bootcamp','0','loans')

insert into watermarkDemo values
(5,'transactions','dbo','transactions','bootcamp','0','transactions')

```

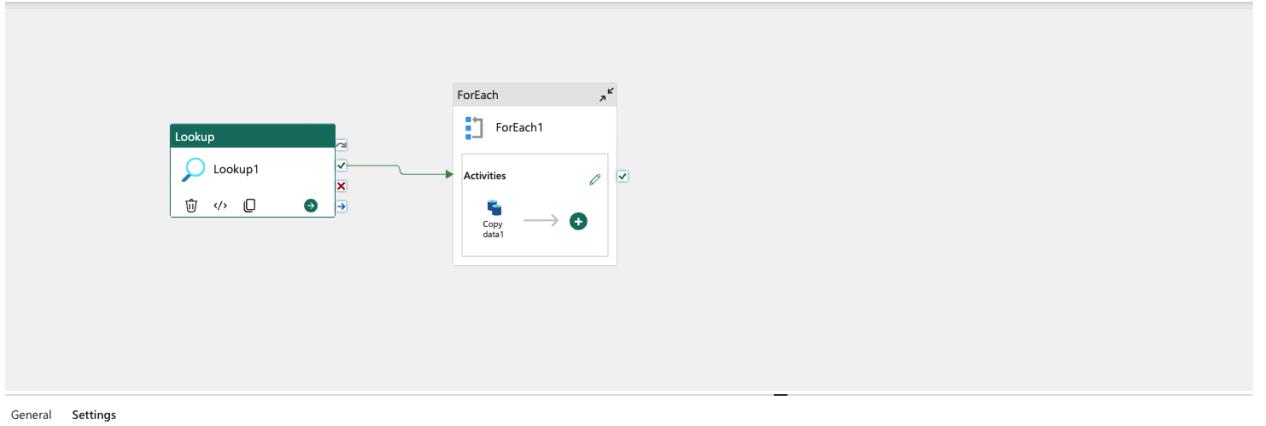
Select \* from watermarkDemo

DROP TABLE watermarkDemo

Watermark table for incremental loading

	id	tablename	schemaname	filename	foldername	lastprocessedvalue	incrementalcolumn
1	5	transactions	dbo	transactions	bootcamp	0	transactions
2	4	loans	dbo	loans	bootcamp	0	loans
3	3	loan_payments	dbo	loan_payments	bootcamp	0	loan_payments
4	1	accounts	dbo	accounts	bootcamp	0	accounts
5	2	customers	dbo	customers	bootcamp	0	customers

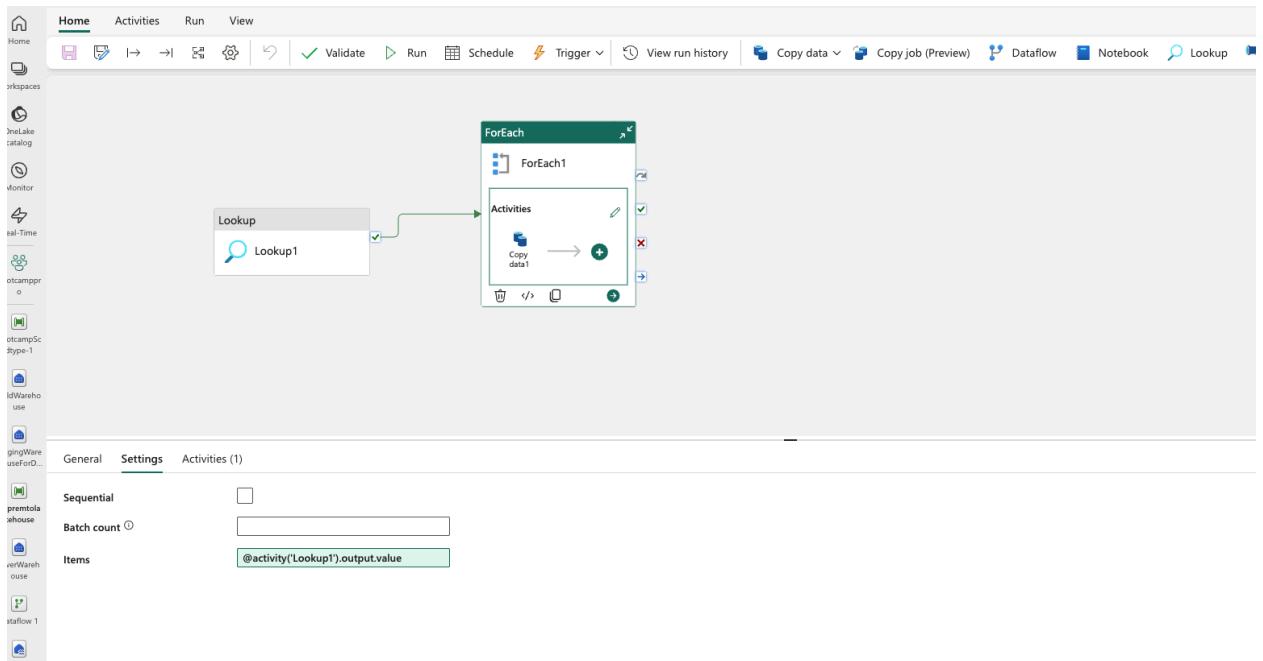
We are not doing incremental loading since we have to implement scd type-2 and updated rows will not come in the incremental loading, hence we are loading multiples files using lookup. This is how pipeline for loading data in the bronze layer will look like.



Adding for lookup activity, select the place where we have created the watermark table and selecting the name of the table.

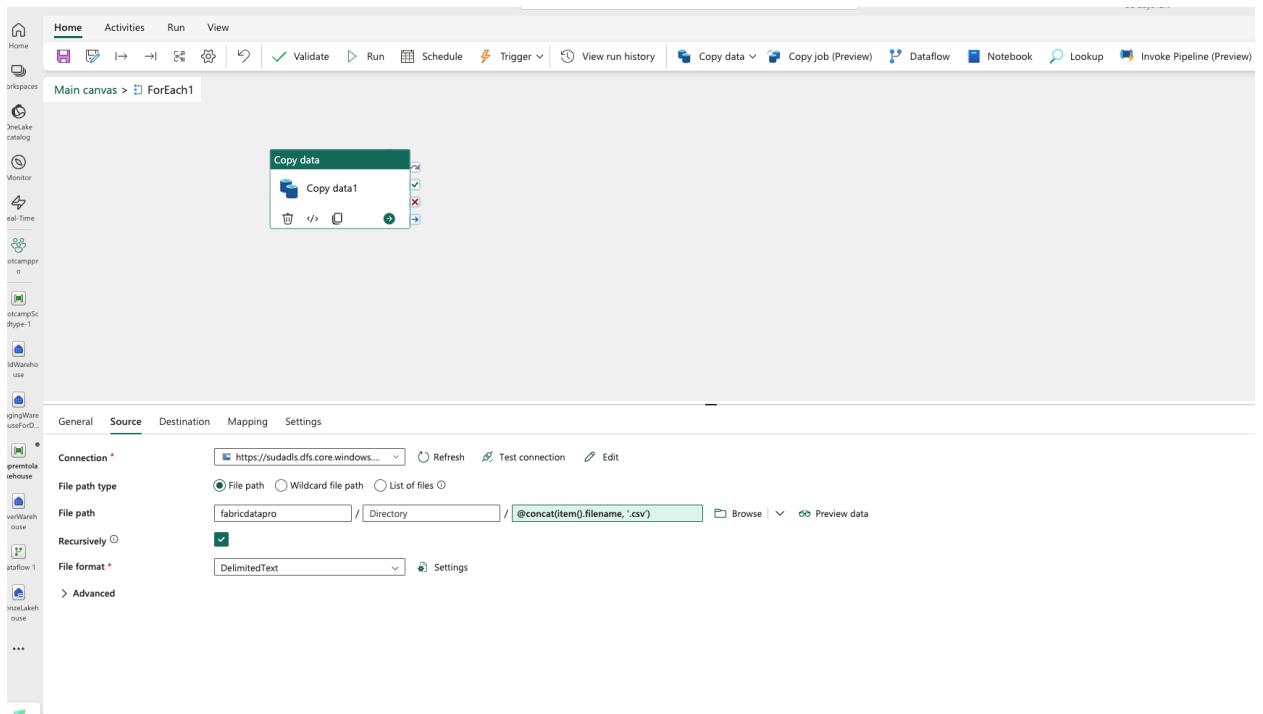
This screenshot shows the 'Settings' tab for the 'Lookup' activity within the pipeline. The 'Table' radio button is selected under the 'Use query' section. The 'Table' dropdown is set to 'dbo.watermarkDemo'. Other options like 'Query' and 'Stored procedure' are available but not selected. The pipeline structure above shows the 'Lookup' activity connected to an 'ForEach' loop, which contains a 'Copy data1' activity.

Adding the for each activity and adding the following expression.



```
@activity('Lookup1').output.value
```

Going inside the for each activity adding the copy data activity. In the copy data add the source path and the destination path.



Creating the connection in the copy data activity to connect ADLS Gen2 storage.

Get data

**Connect data source**

Azure Data Lake Storage Gen2

Azure

[Learn more](#)

**Connection settings**

URL \* ⓘ  
Example: https://contosoadls.dfs.core.windows.net...

**Connection credentials**

Connection: Create new connection ⓘ

Connection name: Connection

Data gateway: (none) ⓘ

Authentication kind: Organizational account ⓘ

You are currently signed in as:

S Sudhanshu sudkhar@jyotikharbanda233gmail.... Switch account

Privacy Level: None ⓘ

This connection can be used with on-premises data gateways and VNet data gateways.

Providing the URL, the structure can be found on the internet.

Get data

**Connect data source**

X

Azure Data Lake Storage Gen2

Azure

[Learn more](#)

**Connection settings**

URL \* ⓘ  
https://sudadls.dfs.core.windows.net/

**Connection credentials**

Connection: Create new connection ⓘ

Connection name: https://sudadls.dfs.core.windows.net/

Data gateway: (none) ⓘ

Authentication kind: Organizational account ⓘ

You are currently signed in as:

S Sudhanshu sudkhar@jyotikharbanda233gmail.... Switch account

Privacy Level: None ⓘ

This connection can be used with on-premises data gateways and VNet data gateways.

Back Connect

Providing the destination URL.

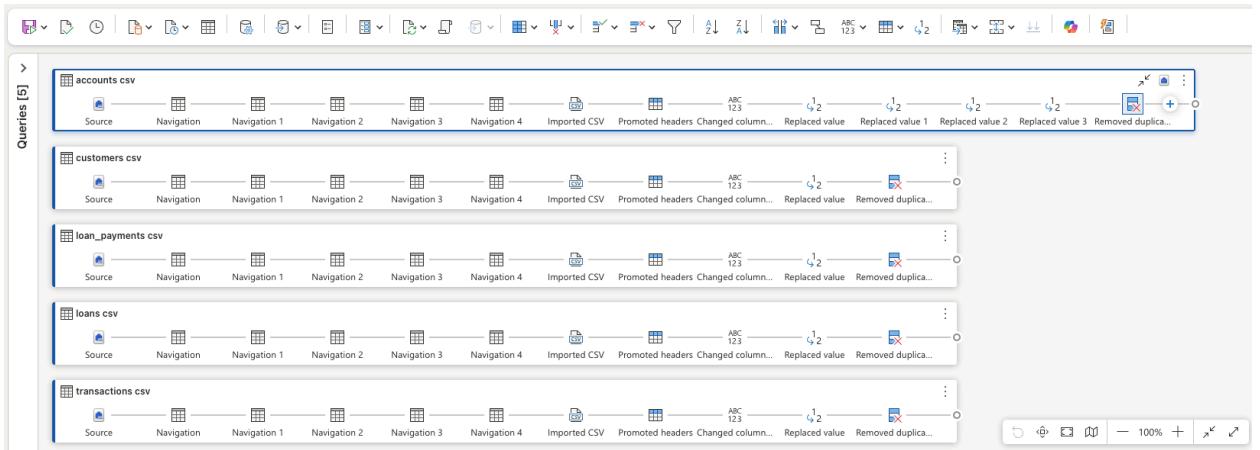
The screenshot shows the Azure Data Factory studio interface. A 'Copy data' activity is selected in the main canvas. Below it, the 'Destination' tab is active in the configuration pane. The 'Connection' dropdown is set to 'bronzeLakehouse'. The 'Root folder' section has 'Files' selected. The 'File path' field contains the expression '@concat(item().filename, '.csv')'. The 'File format' is set to 'DelimitedText'. There are tabs for 'General', 'Source', 'Mapping', and 'Settings'.

RUN THE pipeline, you can see the data is copied from ADLS Gen2 to Fabric Lake House.

The screenshot shows the 'Explorer' view in the Azure Data Factory studio. The 'bronzeLakehouse' folder is expanded, revealing its contents: 'Tables' and 'Files'. The 'Files' folder is also expanded, showing the 'bootcamp' folder. Inside 'bootcamp', five CSV files are listed: 'accounts.csv', 'customers.csv', 'loan\_payments.csv', 'loans.csv', and 'transactions.csv'. The table includes columns for 'Name', 'Date modified', 'Type', and 'Size'.

Name	Date modified	Type	Size
accounts.csv	9/4/2025, 3:23:11 PM	csv	2 KB
customers.csv	9/4/2025, 3:23:12 PM	csv	4 KB
loan_payments.csv	9/4/2025, 3:23:11 PM	csv	2 KB
loans.csv	9/4/2025, 3:23:11 PM	csv	2 KB
transactions.csv	9/4/2025, 3:23:11 PM	csv	3 KB

Going to the dataflow gen2 and creating the dataflow for cleaning the CSV file.



Importing the csv file.

The screenshot shows the Power BI Dataflows interface with the same five queries, but now each has "Imported CSV" selected. Below the queries, a preview pane displays a table with the following data:

	account_id	customer_id	account_type	balance
1	1	45	Savings	1000.50
2	2	12	Checking	2500.75
3	3	78	Savings	1500.00
4	4	34	Checking	3000.25
5	5	56	Savings	500.00
6	6	23	Checking	1200.50
7	7	89	Savings	800.75
8	8	67	Checking	2200.00
9	9	14	Savings	900.25
10				

Adding the first row as columns.

Queries [5]

accounts.csv

customers.csv

loan\_payments.csv

loans.csv

transactions.csv

Table.PromoteHeaders(#"Imported CSV", [PromoteAllScalars = true])

	ABC_account_id	ABC_customer_id	ABC_account_type	ABC_balance
1	1	45	Savings	1000.50
2	2	12	Checking	2500.75
3	3	78	Savings	1500.00
4	4	34	Checking	3000.25
5	5	56	Savings	500.00
6	6	23	Checking	1200.50
7	7	89	Savings	800.75
8	8	67	Checking	2200.00
9	9	14	Savings	900.25
10	10	92	Checking	1800.50

Columns: 4 Rows: 99+ Add default destination...

Changing the datatype of columns

Power Query

Home Transform Add column View Help

Search (Alt + Q)

Queries [5]

accounts.csv

customers.csv

loan\_payments.csv

loans.csv

transactions.csv

Table.TransformColumnTypes(#"Promoted headers", {{"account\_id", Int64.Type}, {"customer\_id", Int64.Type}, {"account\_type", type text}, {"balance", type number}})

	123_account_id	123_customer_id	ABC_account_type	1.2_balance
1	1	45	Savings	1000.5
2	2	12	Checking	2500.75
3	3	78	Savings	1500
4	4	34	Checking	3000.25
5	5	56	Savings	500
6	6	23	Checking	1200.5
7	7	89	Savings	800.75
8	8	67	Checking	2200
9	9	14	Savings	900.25
10	10	92	Checking	1800.5

Completed (1.00 s) Columns: 4 Rows: 99+ Add default destination...

Changing the values where there is null value to 0.

Changing the values where there is null value to 0.

Completed (0.93 s) Columns: 4 Rows: 99+ Add default destination...

1.2 account_id	1.2 customer_id	#&_account_type	1.2 balance
1	1	45 Savings	1000.5
2	2	12 Checking	2500.75
3	3	78 Savings	1500
4	4	34 Checking	3000.25
5	5	56 Savings	500
6	6	23 Checking	1200.5
7	7	89 Savings	800.75
8	8	67 Checking	2200
9	9	14 Savings	900.25
10	10	92 Checking	1800.5

Completed (0.93 s) Columns: 4 Rows: 99+ Add default destination...

1.2 account_id	1.2 customer_id	#&_account_type	1.2 balance
1	1	45 Savings	1000.5
2	2	12 Checking	2500.75
3	3	78 Savings	1500
4	4	34 Checking	3000.25
5	5	56 Savings	500
6	6	23 Checking	1200.5
7	7	89 Savings	800.75
8	8	67 Checking	2200
9	9	14 Savings	900.25
10	10	92 Checking	1800.5

Removing duplicates in the dataflow and adding the sink as data warehouse in each dataflow. Run the dataflow.

	1.2 account_id	1.2 customer_id	1.2 account_type	1.2 balance
1	1	45	Savings	1000.5
2	2	12	Checking	2500.75
3	3	78	Savings	1500
4	4	34	Checking	3000.25
5	5	56	Savings	500
6	6	23	Checking	1200.5
7	7	89	Savings	800.75
8	8	67	Checking	2200
9	9	14	Savings	900.25
10	10	92	Checking	1800.5

Here we can see all the tables have been created for the respective csv file where the cleaned data is stored.

	12f account_id	12f customer_id	12f account_type	12f balance
1	1	45	Savings	1000.5
2	2	12	Checking	2500.75
3	3	78	Savings	1500
4	4	34	Checking	3000.25
5	5	56	Savings	500
6	6	23	Checking	1200.5
7	7	89	Savings	800.75
8	8	67	Checking	2200
9	9	14	Savings	900.25
10	10	92	Checking	1800.5
11	11	3	Savings	1100.75
12	12	81	Checking	2700
13	13	29	Savings	1300.25
14	14	64	Checking	3200.5
15	15	47	Savings	700.75
16	16	18	Checking	1400
17	17	16	Savings	600.25
18	18	5	Checking	1600.5
19	19	76	Savings	400.75
20	20	21	Checking	2000
21	21	53	Savings	300.25
22	22	37	Checking	2400.5
23	23	88	Savings	200.75
24	24	11	Checking	2600
25	25	66	Savings	100.25
26	26	25	Checking	2800.5
27	27	94	Savings	50.75

Now we are creating an empty target table, which will be used for storing SCD type-1 data.

```

CREATE TABLE dbo.TarAccounts

(
    account_id      FLOAT           NOT NULL,
    customer_id     FLOAT           NOT NULL,
    account_type    VARCHAR(50)     NULL,
    balance         FLOAT           NULL,
    createdBy        VARCHAR(50)     NULL,
    createdOn       DATETIME2(3)   NULL,
    updatedBy       VARCHAR(50)     NULL,
    updatedOn       DATETIME2(3)   NULL
);

SELECT * FROM dbo.TarAccounts

SELECT * FROM silverWarehouse.dbo.accounts AS source_accounts

SELECT account_id FROM source_accounts;

```

Here is the source table.

	12F account_id	12F customer_id	abc account_type	12F balance
1	1	45	Savings	1000.5
2	2	12	Checking	2500.75
3	3	78	Savings	1500
4	4	34	Checking	3000.25
5	5	56	Savings	500
6	6	23	Checking	1200.5
7	7	89	Savings	800.75
8	8	67	Checking	2200
9	9	14	Savings	900.25
10	10	92	Checking	1800.5
11	11	3	Savings	1100.75
12	12	81	Checking	2700
13	13	29	Savings	1300.25
14	14	64	Checking	3200.5
15	15	47	Savings	700.75

## Creating a procedure for implementing SCD type-1

```

CREATE OR ALTER PROCEDURE dbo.sp_SyncAccounts
AS
BEGIN
    SET NOCOUNT ON;

```

```

-- Update existing records

UPDATE TARGETS SET

TARGETS.[account_id] = SOURCE.[account_id],
TARGETS.[customer_id] = SOURCE.[customer_id],
TARGETS.[account_type] = SOURCE.[account_type],
TARGETS.[Balance] = SOURCE.[balance],
TARGETS.updatedBy = 'DataFlow_Updates',
TARGETS.updatedOn = SYSDATETIME()

FROM goldWarehouse.dbo.TarAccounts AS TARGETS

JOIN silverWarehouse.dbo.accounts AS SOURCE

ON TARGETS.[account_id]=SOURCE.[account_id]

WHERE TARGETS.[account_id] <> SOURCE.[account_id] OR
TARGETS.[customer_id] <> SOURCE.[customer_id] OR
TARGETS.[account_type] <> SOURCE.[account_type] OR
TARGETS.[Balance] <> SOURCE.[balance];

-- Insert new records

INSERT INTO goldWarehouse.dbo.TarAccounts (

[account_id], [customer_id], [account_type], [Balance],
createdBy, createdOn, updatedBy, updatedOn

)

SELECT

SOURCE.[account_id], SOURCE.[customer_id], SOURCE.[account_type],
SOURCE.[balance],
'Dataflow', SYSDATETIME(),
'Dataflow', SYSDATETIME()

FROM silverWarehouse.dbo.accounts AS SOURCE

LEFT JOIN goldWarehouse.dbo.TarAccounts AS TARGETS

ON TARGETS.account_id = SOURCE.account_id

```

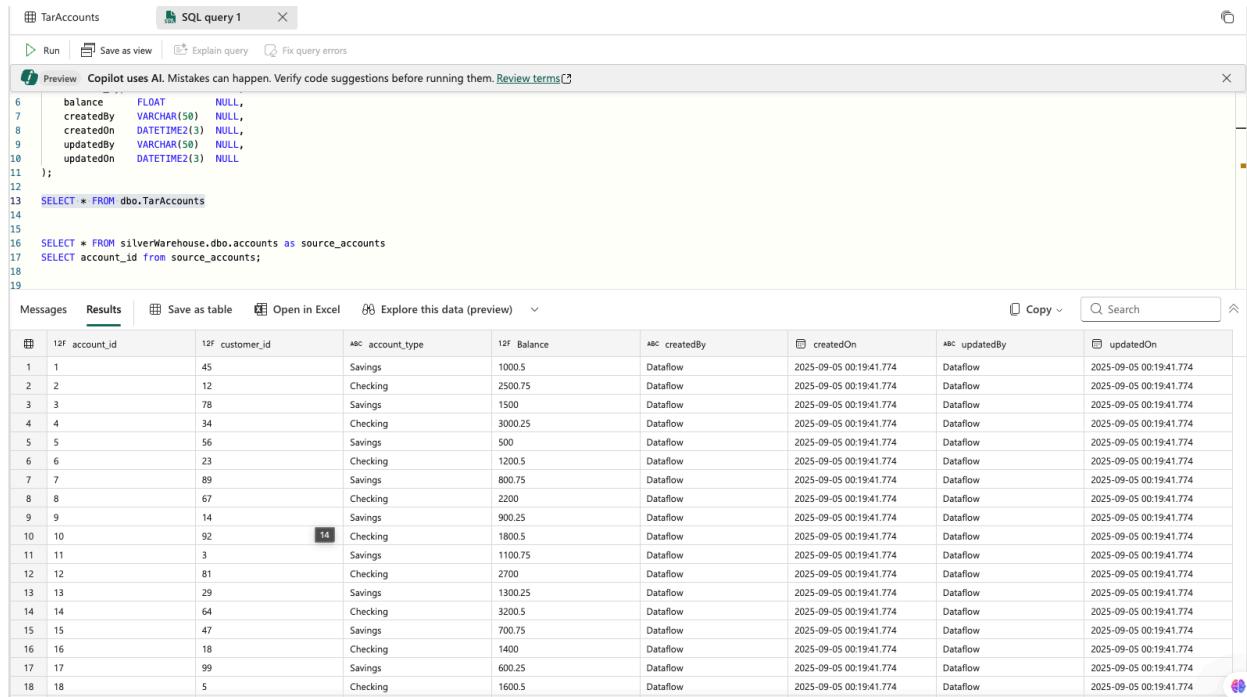
```
WHERE TARGETS.account_id IS NULL;
```

```
END;
```

```
GO
```

```
EXEC dbo.sp_SyncAccounts;
```

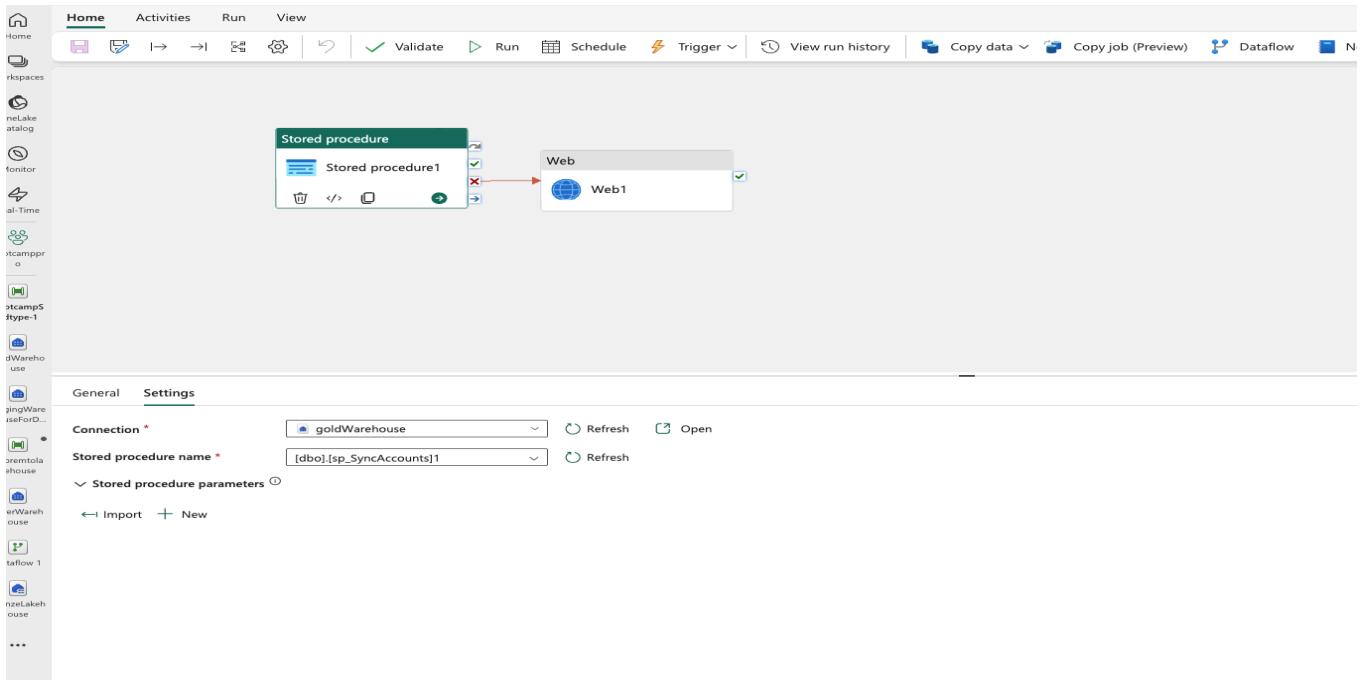
After executing scd type-1, here is the output.



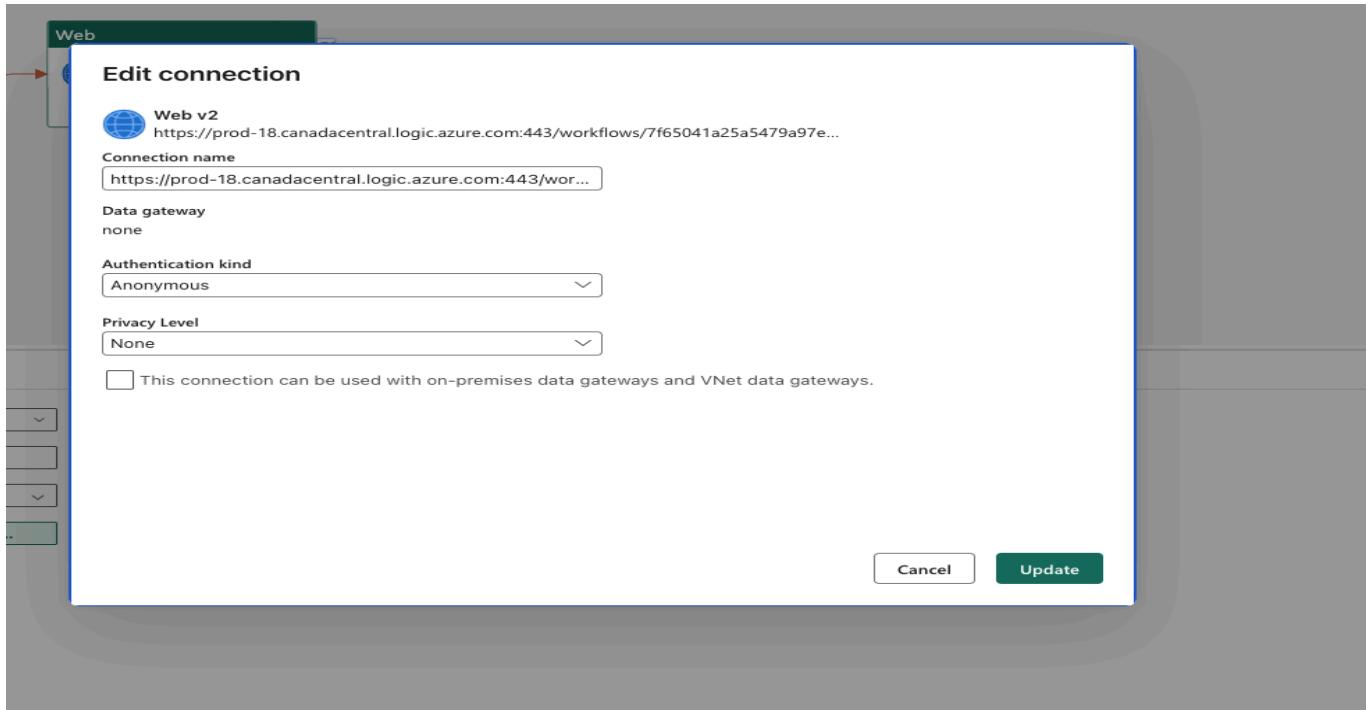
The screenshot shows a SQL query results table titled 'TarAccounts' in a 'SQL query 1' window. The table contains 18 rows of account data. The columns are: account\_id, customer\_id, account\_type, Balance, createdBy, createdOn, updatedBy, and updatedOn. The data is as follows:

	account_id	customer_id	account_type	Balance	createdBy	createdOn	updatedBy	updatedOn
1	1	45	Savings	1000.5	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
2	2	12	Checking	2500.75	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
3	3	78	Savings	1500	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
4	4	34	Checking	3000.25	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
5	5	56	Savings	500	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
6	6	23	Checking	1200.5	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
7	7	89	Savings	800.75	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
8	8	67	Checking	2200	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
9	9	14	Savings	900.25	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
10	10	92	Checking	1800.5	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
11	11	3	Savings	1100.75	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
12	12	81	Checking	2700	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
13	13	29	Savings	1300.25	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
14	14	64	Checking	3200.5	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
15	15	47	Savings	700.75	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
16	16	18	Checking	1400	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
17	17	99	Savings	600.25	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
18	18	5	Checking	1600.5	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774

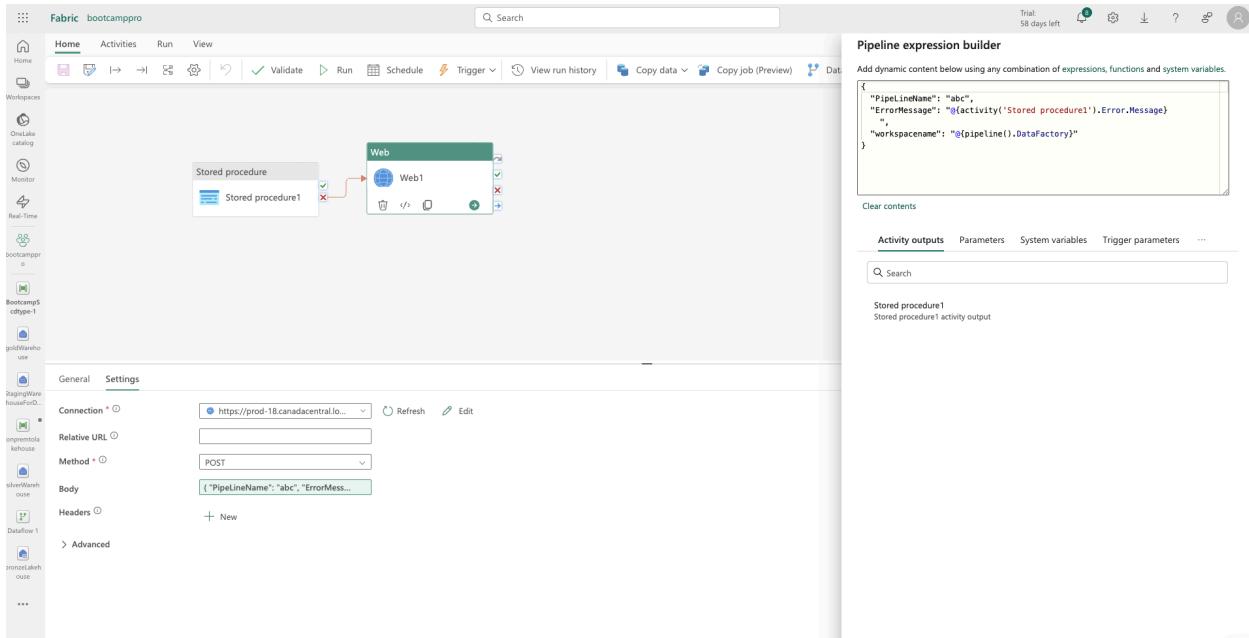
Now we are creating a new pipeline that will trigger a mail when scdtype=1 implementation fails. Add a stored procedure activity in the pipeline and do the following configuration.



In the web activity, creating a connection with logic app that we will create in the azure.



Provide the following expression in the expression builder.



{

```

    "type": "object",
    "properties": {
        "PipeLineName": {
            "type": "string"
        },
        "ErrorMessage": {
            "type": "string"
        },
        "workspacename": {
            "type": "integer"
        }
    }
}
```

Create a logic app, add a when an http received and send email v2 activity. Do the following configuration. Copy the HTTP URL and paste into the web activity.

The screenshot shows the Logic App Designer interface. On the left, the navigation menu is visible with the 'Logic app designer' option selected. The main workspace contains a 'When an HTTP request is received' trigger and a 'Send email (V2)' action connected by a blue line. The 'Send email (V2)' action has a plus sign next to it, indicating it can be expanded. To the right of the workspace, there is a detailed configuration pane for the 'Send email (V2)' action. It includes sections for 'Parameters', 'Settings', 'Code view', and 'About'. Under 'Parameters', the 'HTTP URL' is set to <https://prod-18.canadacentral.logic.azure.com:443/workflows/7f65041a25a5479a97e2c9f9397a0d44/t>. The 'Method' is set to 'POST'. The 'Request Body JSON Schema' section contains the following JSON:

```
{
  "type": "object",
  "properties": {
    "PipelineName": {
      "type": "string"
    },
    "ErrorMessage": {
      "type": "string"
    },
    "workspaceName": {
      "type": "integer"
    }
  }
}
```

[Use sample payload to generate schema](#)

Provide your email ID and provide the following configuration.

The screenshot shows the Logic App Designer interface with the 'Send email (V2)' action expanded. The 'Parameters' tab is selected. The 'To' field is set to 'dude.sk44@gmail.com'. In the 'Advanced parameters' section, 'Showing 3 of 6' items are listed: 'Importance' (High), 'Subject' (Pipeline Failed), and 'Body'. The 'Body' section contains a rich text editor with placeholder text: 'Hi team', 'Name of Pipeline-  **PipelineName** ', 'Workspace-  **workspaceName** ', and 'Error Message-  **ErrorMessage** '. A note at the bottom states 'Connected to new\_conn\_25da2. Change connection'.

Run the pipeline when the stored procedure fails, you will receive the mail.

