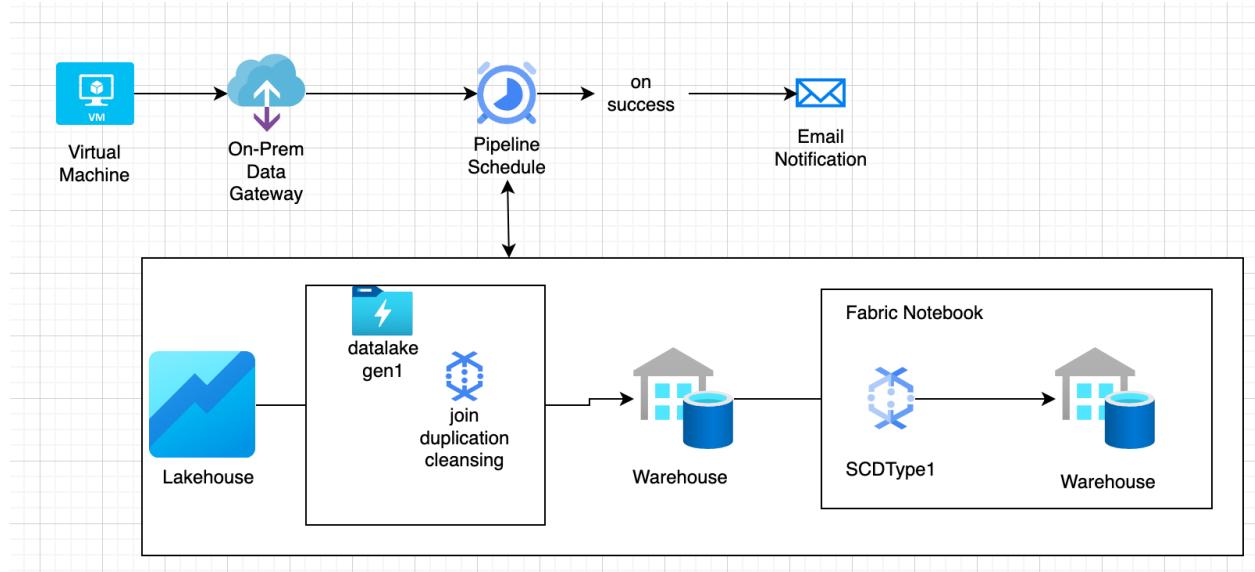


Bootcamp Project 4

Project Title: Incremental Data Loading and Automated Notifications using Microsoft Fabric

Problem Statement: In modern data ecosystems, organizations need to efficiently ingest, transform, and load data from various sources into centralized platforms for analytics, while also ensuring timely monitoring and notification upon successful data refreshes. This project addresses the challenge of incrementally loading data from on-premises sources to Microsoft Fabric Lakehouse, processing it through a structured transformation pipeline, and triggering automated notifications upon successful execution.

Architecture Diagram:



Tools & Technologies:

- Microsoft Fabric
- On-Premises Data Gateway
- Fabric Lakehouse and Warehouse
- Fabric Dataflow Gen 1
- Fabric Notebook
- Email Notification Task (in-built)
- Azure Key Vault (optional for secure credential management)

Data Ingestion from virtual Machine to Fabric Lakehouse using On Prem Data Gateway

? X



On-premises data gateway

You are signed in as shubha@vaishali2outlook.onmicrosoft.com and are ready to register the gateway.

New on-premises data gateway name *

shubha

Add to an existing gateway cluster [Learn more](#)

Recovery key (8 character minimum) *

① This key is needed to restore the gateway and can't be changed. Record it in a safe place.

Confirm recovery key *

We'll use this region to connect the gateway to cloud services: Canada Central [Change Region](#)

[Provide relay details \(optional\)](#) By default, Azure Relays are automatically provisioned



Configure

Cancel

The screenshot shows the 'On-premises data gateway' status page. The left sidebar has a 'Status' tab selected, along with other options like Service Settings, Diagnostics, Network, Connectors, and Recovery Keys. The main content area displays the following status information:

- Logic Apps, Azure Analysis Services**: Canada Central. Status: [Create a gateway in Azure](#)
- Power Apps, Power Automate**: Canada Central. Status: Ready
- Microsoft Fabric**: Default environment. Status: Ready

Now check if data gateway status is online in fabric

Fabric → setting → manage connection and gateway → gateways and check if it is online

Tenant administration

Manage Connections and Gateways

[Connections](#) [On-premises data gateways](#) [Virtual network data gateways](#) [Azure Key Vault references](#)

The data gateway acts as a bridge, providing quick and secure data transfer between on-premises data and Power BI, Microsoft Flow, Logic Apps, and PowerApps. [Learn more in this overview.](#)

Name ↑	Contact info	Users	Status	Gateways
shubha	(i) ... shubha@vaishali2outlook.onmicrosoft.com	shubha	Online	1

and create a connection(linked service)

Manage Connections and Gateways

Connections On-premises data gateways Virtual network data gateways Azure Key Vault references

Cloud and data gateway connections for artifacts. [Learn more about supported connections.](#)

Name ↑	Connection type	Users	Status	Gateway cluster name
FabricDataPipelines shubha	Fabric Data Pipelines	shubha	🕒	
Lakehouse	Lakehouse	shubha	🕒	
Lakehouse shubha	Lakehouse	shubha	🕒	
shubha	...	Folder	🕒	shubha
Warehouse	Warehouse	shubha	🕒	

Now create a pipeline to ingest data from Virtual Machine to Lake house

The screenshot shows the Azure Data Factory Pipeline designer interface. At the top, there is a preview of the pipeline structure:

```

graph LR
    GetMetadata1[Get Metadata] --> ForEach1[ForEach]
  
```

Below the preview, the "Settings" tab is selected for the "Get Metadata" activity. The configuration includes:

- Connection:** shubha
- File path:** Project1 / File name
- File format:** DelimitedText
- Field list:** New, Argument, Child items

On the right side, the "Pipeline expression builder" pane shows the expression: `@activity('Get Metadata1').output.childItems`.

At the bottom, the "Activities (1)" tab is selected for the pipeline, showing the configuration for the "ForEach" activity:

- Sequential:** Unchecked
- Batch count:** (empty input field)
- Items:** `@activity('Get Metadata1').output.chi...`

Main canvas > `ForEach1`

The screenshot shows the Azure Data Factory interface. At the top, there's a navigation bar with tabs: General, Source, Destination, Mapping, and Settings. The Destination tab is currently selected. Below the tabs, there's a form with the following fields:

- Connection ***: project4 (dropdown menu)
- Root folder**: Tables (radio button selected)
- Table**: `@concat(replace(item().name, '.csv', ''))` (text input field)
- Table action**: Overwrite (radio button selected)

Below the form is a toolbar with various icons for Home, Activities, Run, View, and other common operations like Validate, Run, Schedule, Add trigger, and preview.

Main canvas > `ForEach1`

The second part of the screenshot shows the same ForEach loop structure and configuration as the first one, indicating a successful operation.

General Source Destination Mapping Settings

Connection * project4 Refresh Open

Root folder Tables Files

Table `@concat(replace(item().name, '.csv', ''))`

Table action Append Overwrite

> Advanced

Home Activities Run View

Main canvas > `ForEach1`

Copy data1

Validate Run Schedule Add trigger

Main canvas > `ForEach1`

Copy data1

Validate Run Schedule Add trigger

General Source Destination Mapping Settings

Connection * shubha Refresh Test connection Edit

File path type File path File filter Wildcard file path List of files

Wildcard paths Wildcard folder path / `@item().name` Preview data

Recursively

File format * DelimitedText Settings

> Advanced

data got successfully loaded in lakehouse

project4 ▾

Search

59 days left

Lakehouse ▾

A SQL analytics endpoint for SQL querying was created with this item.

Explorer

accounts

Showing

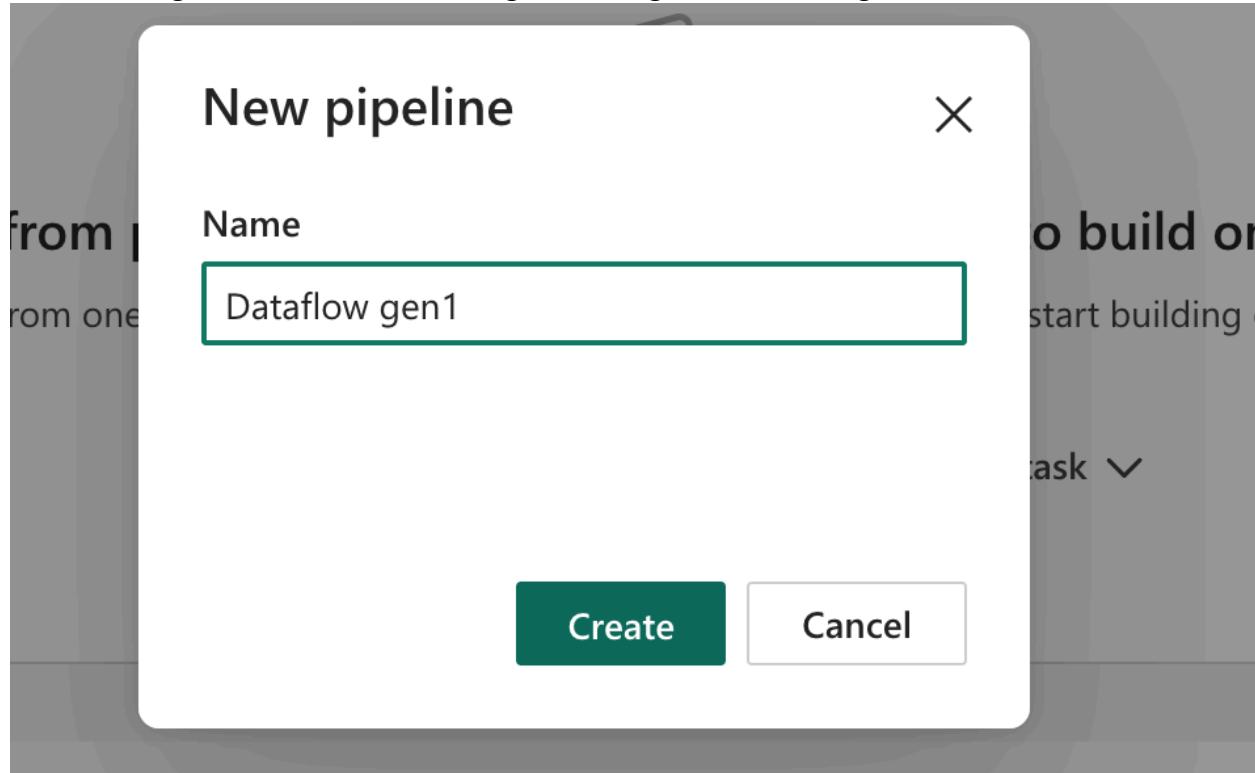
Search tables

project4

- Tables
 - accounts
 - customers
 - loan_payments
 - loans
 - transactions
- Files

	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	10	92	Checking	1800.50
11	11	3	Savings	1100.75
12	12	81	Checking	2700.00
13	13	29	Savings	1300.25

Data cleaning and transformation using dataflow gen1 and storing in warehouse



Dataflow gen1

Search Trial: 59 days left

Home Activities Run View

Validate Run Schedule Add trigger

Dataflow

Dataflow1

General Settings

Workspace * shubha Refresh

⚠ Selected workspace does not have any Dataflow.

Dataflow * Refresh New

New Dataflow Gen2

Name Dataflow 1

Enable Git integration, deployment pipelines and Public API scenarios (preview)

Create Cancel

General Settings

Workspace * shubha Refresh

⚠ Selected workspace does not have any Dataflow.

Dataflow * Refresh New

Dataflow 1

Power Query Dataflow 1 Dataflow saved

Search (Alt + Q)

Home Transform Add column View Help

Import from Excel Import from SQL Server Import from a Text/CSV file Import from dataflows

Get data from another source →

Import from a Power Query template

Import from

Now to get the source we need to create a connection , for that click new source → choose source

Get data
Choose data source

Search

New sources

View more →

Excel workbook File

SQL Server database Database

SharePoint folder File

Text/CSV File

Power BI dataflows (Leg...)

Dataflows Power Platform

Dataverse Power Platform

SharePoint Online list Online services

Recent (Preview)

View more →

Cancel

here we can search for any source that we want , I am looking for lakehouse so search it and select it

Once lakehouse is selected , we it create a connection automatically to all the available workspaces

Get data
Connect to data source

Lakehouse Microsoft Fabric

Create new connection

Connection name

Connection

Data gateway

(none)

Authentication kind

Organizational account

You are currently signed in as:

shubha
shubha@bhanumurali925@gmail.on...

Switch account

Back

Cancel

Next

Now we want to select which workspace lakehouse we want and what is the source from the lakehouse file or table

Get data

Choose data

The screenshot shows the Power Query Editor interface with the 'Choose data' step selected. At the top, there is a search bar and a 'Display options' dropdown. Below this, a tree view shows the project structure: 'DataflowsStagingLakehouse' > 'shubha_project1' > 'Files' > 'project4'. Inside 'project4', several CSV files are listed: 'queryinsights.exec_reque...', 'queryinsights.exec_sessio...', 'queryinsights.frequently...', 'queryinsights.long_runni...', 'sys.managed_delta_table...', and 'sys.managed_delta_table...'. To the right of the tree view is a table titled 'project4' showing the details of these files.

Table: project4

Content	Name	Extension	Date accessed	Date modified	Date created	Attributes	Folder Path
[Table]	accounts.csv	.csv	null	5/2/2025, 1:53:09 AM	null	[Record]	https://onelake.dfs.fabric.n
[Table]	customers.csv	.csv	null	5/2/2025, 1:53:02 AM	null	[Record]	https://onelake.dfs.fabric.n
[Table]	loan_payments....	.csv	null	5/2/2025, 1:53:10 AM	null	[Record]	https://onelake.dfs.fabric.n
[Table]	loans.csv	.csv	null	5/2/2025, 1:53:02 AM	null	[Record]	https://onelake.dfs.fabric.n
[Table]	transactions.csv	.csv	null	5/2/2025, 1:53:03 AM	null	[Record]	https://onelake.dfs.fabric.n

Below the table, the Power Query ribbon shows 'Power Query' > 'Dataflow 1' > 'Dataflow saved'. The main area displays the 'Queries [1]' pane, which lists 'project4' as the source. To the right, the 'Query settings' pane shows the 'Name' as 'project4' and the 'Applied steps' section, which includes 'Navigation 1' through 'Navigation 4'. The bottom pane shows the flow of data from 'Navigation 1' through various steps like 'Imported CSV' and 'Promoted headers' to 'Filtered rows'.

The screenshot shows the Power Query Editor interface. A context menu is open over a step labeled 'Filtered rows'. The menu is titled 'Search commands' and contains several options under the 'Reduce rows' section, including 'Keep top rows', 'Keep bottom rows', 'Keep range of rows', 'Keep duplicates', 'Keep errors', 'Remove top rows', 'Remove bottom rows', 'Remove alternate rows', 'Remove duplicates', 'Remove blank rows', 'Remove errors', 'Filter rows', 'Sort', and 'Sort ascending'. The 'Remove duplicates' option is highlighted.

Filter rows ?

Apply one or more filter conditions to the rows in this table.

Basic Advanced

Keep rows where

Column name	Operator	Value	...
account_id	does not equal	null	...
customer_id	does not equal	null	...

[Add clause](#)

[Cancel](#)

[OK](#)

The screenshot shows the Power Query Editor interface with a context menu open over a query named 'Removed duplicates'. The menu includes options like 'Collapse', 'Highlight related queries', 'Expand related queries', 'Collapse related queries', 'Copy', 'Paste', 'Delete', 'Rename', 'Enable staging', 'Incremental refresh', 'Require fast copy', 'Duplicate', 'Reference', 'Move to group', 'Create function...', 'Convert to parameter', 'Advanced editor', 'Properties...', 'Append queries', and 'Append queries as new'. The 'Append queries' option is highlighted.

Power Query | Dataflow 1 | Dataflow saved

Search (Alt + Q)

Home Transform Add column View Help

Queries [1] > project4 (3) 11 steps

Source Navigation Navigation 1 Navigation 2 Navigation 3 Navigation 4 Imported CSV Promoted headers Changed column...

accounts csv

Source Navigation Navigation 1 Navigation 2 Navigation 3 Navigation 4 Imported CSV

`Csv.Document(#"Navigation 4", [Delimiter = ",", Columns = 4, QuoteStyle = QuoteStyle.None])`

	Column1	Column2	Column3	Column4
1	account_id	customer_id	account_type	balance
2	1	45	Savings	1000.50
3	2	12	Checking	2500.75
4	3	78	Savings	1500.00
5	4	34	Checking	3000.25
6	5	56	Savings	500.00
7	6	23	Checking	1200.50

Completed (0.72 s) Columns: 4 Rows: 99+ Add default destination...

Source Navigation Navigation 1 Navigation 2 Navigation 3 Navigation 4 Imported CSV

accounts csv

Source Navigation Navigation 1 Navigation 2 Navigation 3 Navigation 4 Imported CSV

`Csv.Document(#"Navigation 4", [Delimiter = ",", Columns = 4, QuoteStyle = QuoteStyle.None])`

	Column1	Column2	Column3	Column4
1	account_id	customer_id	account_type	balance
2	1	45	Savings	1000.50
3	2	12	Checking	2500.75
4	3	78	Savings	1500.00
5	4	34	Checking	3000.25
6	5	56	Savings	500.00
7	6	23	Checking	1200.50

Completed (0.72 s) Columns: 4 Rows: 99+ Add default destination...

Search commands

Sort

- Sort ascending
- Sort descending

Combine

- Merge queries
- Merge queries as new
- Append queries
- Append queries as new

Transform table

- Group by
- Use first row as headers
- Use headers as first row

Transform any column

- Replace values
- Replace errors

Home Transform Add column View Help

Queries [2]

Source Navigation Navigation 1 Navigation 2 Navigation 3 Navigation 4

accounts csv

Source Navigation Navigation 1 Navigation 2 Navigation 3 Navigation 4

Table.TransformColumnTypes("#Promoted headers", {"account_id": Int, "customer_id": Int, "account_type": Categorical, "balance": Number})

1	2	3	4
1	1	45	Savings
2	2	12	Checking
3	3	78	Savings
4	4	34	Checking
5	5	56	Savings
6	6	23	Checking
7	7	89	Savings

Completed (0.81 s) Columns: 4 Rows: 99+ Add default destination...

Search commands Manage columns

- Choose columns
- Remove columns
- Remove other columns

Reduce rows

- Keep top rows
- Keep bottom rows
- Keep range of rows
- Keep duplicates
- Keep errors
- Remove top rows
- Remove bottom rows
- Remove alternate rows
- Remove duplicates
- Remove blank rows

Step

Filter rows ?

Apply one or more filter conditions to the rows in this table.

Basic Advanced

Keep rows where

Column name	Operator	Value
account_id	does not equal	null
and	customer_id	does not equal
		null

... Add clause

Cancel

OK

Queries [2]

Source Navigation Navigation 1 Navigation 2 Navigation 3 Navigation 4 Imported CSV Promoted headers Changed column... Removed duplicates Filtered rows

Table.SelectRows("#Removed duplicates", each [account_id] <> null and [customer_id] <> null)

1	2	3	4
39	39	74	Savings
40	40	19	Checking
41	41	51	Savings
42	42	36	Checking
43	43	83	Savings
44	44	13	Checking
45	45	68	Savings

Completed (2.14 s) Columns: 4 Rows: 100 Add default destination...

Step Publish

Dataflow 1 Dataflow saved

Power Query | Dataflow 1 | Dataflow saved

Home Transform Add column View Help

Queries [2]

Navigation Navigation 1 Navigation 2 Navigation 3 Navigation 4 Imported CSV Promoted headers Changed column...

Table.SelectRows(#"Removed duplicates", each [account_id] <> null and [customer_id] <> null)

account_id	customer_id	account_type	balance
39	39	Savings	225.75
40	40	Checking	4100
41	41	Savings	250.25
42	42	Checking	4300.5
43	43	Savings	275.75
44	44	Checking	4500
45	45	Savings	300.25

Completed (2.14 s) Columns: 4 Rows: 100 Add default destination...

26 Trial: 59 days left

Collapse

- Highlight related queries
- Expand related queries
- Collapse related queries
- Copy
- Paste
- Delete
- Rename
- Enable staging
- Incremental refresh
- Require fast copy
- Duplicate
- Reference
- Move to group
- Create function...
- Convert to parameter
- Advanced editor
- Properties...
- Append queries
- Append queries as new

Publish

The screenshot shows the Microsoft Power BI Dataflow interface. A context menu is open on the right side of the screen, listing various actions such as 'Collapse', 'Highlight related queries', 'Delete', and 'Append queries'. The main area displays a table with four columns: account_id, customer_id, account_type, and balance. The table contains 10 rows of data. At the bottom left, it says 'Completed (2.14 s)' and 'Columns: 4 Rows: 100'. At the top, there's a search bar and a trial status message 'Trial: 59 days left'.

add a destination for warehouse

Merge [?](#)



Select a table and matching columns to create a merged table.

accounts csv



1 ² 3 account_id	1 ² 3 customer_id	A ^B C account_type	1.2 balance	
2	12	Checking	2500.75	
3	78	Savings	1500	
4	34	Checking	3000.25	
5	56	Savings	500	
6	22	Checking	1000.5	

Right table for merge *

customers csv



1 ² 3 customer_id	A ^B C first_name	A ^B C last_name	A ^B C address	A ^B C city	A ^B C state	A ^B C
1	John	Doe	123 Elm St	Toronto	ON	M4

The selection matches 88 of 100 rows from the first table

Cancel

OK

Merge [?](#)



Select a table and matching columns to create a merged table.

1 ² 3 customer_id	A ^B C first_name	A ^B C last_name	A ^B C address	A ^B C city	A ^B C state	A ^B C
1	John	Doe	123 Elm St	Toronto	ON	M4
2	Jane	Smith	456 Maple Ave	Ottawa	ON	K1A
3	Michael	Johnson	789 Oak Dr	Montreal	QC	H1A
4	Emily	Davis	101 Pine Rd	Calgary	AB	T2A
5	David	Wilson	202 Birch Blvd	Vancouver	BC	V5B

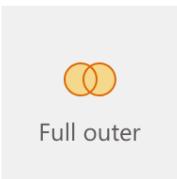
Join kind



Left outer



Right outer



Full outer



Inner



Left anti

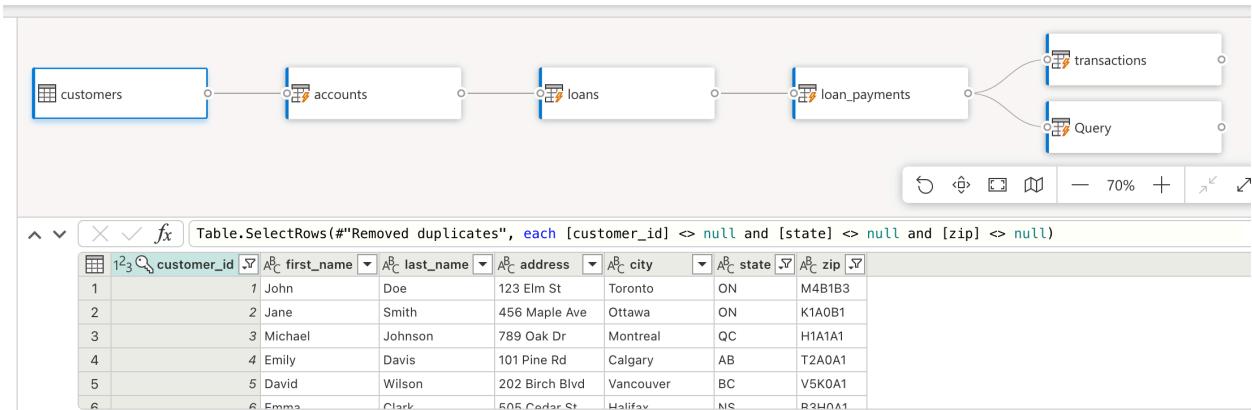


Right anti

The selection matches 88 of 100 rows from the first table

Cancel

OK



data got successfully loaded in data warehouse

Explorer pane shows the project structure:

- + Warehouses
- project
 - Schemas
 - dbo
 - Tables
 - > accounts
 - > customers
 - > jointables
 - > loan_payments
 - > loans
 - > transactions

Data preview - accounts table:

account_id	customer_id	account_type	balance
1	45	Savings	1000.50
2	12	Checking	2500.75
3	12	Checking	6500.00
4	2	Checking	8300.50
5	78	Savings	1500.00
6	3	Savings	1100.75
7	34	Checking	3000.25
8	4	Checking	7900.50
9	56	Savings	500.00
10	5	Checking	1600.50
11	23	Checking	1200.50
12	6	Checking	4900.00

Succeeded (2 sec 351 ms) Columns: 4 Rows: 100

Transforming data into SCD type 1 logic

To access the tables from warehouse create shortcut in lake house

Explorer pane shows the project structure:

- + project4
 - Tables
 - > accounts
 - > customers
 - > loan_payments
 - > loans
 - > transactions

New shortcut context menu options:

- Properties
- Sort tables by
- Refresh

Data preview - accounts table:

account_id	customer_id	account_type	balance
1	45	Savings	1000.50
2	12	Checking	2500.75
3	78	Savings	1500.00
4	34	Checking	3000.25
5	56	Savings	500.00
6	23	Checking	1200.50
7	89	Savings	800.75
8	67	Checking	2200.00
9	14	Savings	900.25
10	92	Checking	1800.50
11	3	Savings	1100.75

A SQL analytics endpoint for SQL querying was created with this item.

Explorer

Tables > Unidentified > dbo

Name	Date modified	Type	Size
accounts	5/5/2025, 8:20....	Folder	-
customers	5/5/2025, 8:20....	Folder	-
jointables	5/5/2025, 8:20....	Folder	-
loan_payments	5/5/2025, 8:20....	Folder	-
loans	5/5/2025, 8:20....	Folder	-
transactions	5/5/2025, 8:21....	Folder	-

SCDTYPE 1 logic using Fabric Notebook

Read Source Delta Table

Register it as a temporary view (source_view) for optional SQL querying.

```

1 df_source = spark.read.format("delta").load("Tables/dbo/accounts")
2 df_source.createOrReplaceTempView("source_view")
3 Spark jobs (1 of 1 succeeded)
[1] ✓ 1 sec - Command executed in 5 sec 927 ms by shubha on 12:07:10 AM, 5/06/25
PySpark (Python) ▾

```

> [Spark jobs \(1 of 1 succeeded\)](#) [Resources](#) [Log](#) ...

Create Target SCD Table (If Not Exists)

```

1 create_table_query = """
2 CREATE TABLE Account_SCD (
3     account_id int,
4     customer_id int,
5     account_type string,
6     balance float,
7     hash_key BIGINT,
8     created_by STRING,
9     created_date TIMESTAMP,
10    updated_by STRING,
11    updated_date TIMESTAMP
12 )
13 USING DELTA
14 LOCATION 'Tables/Gold_layer/Account_SCD'
15 """
16 # Execute the query to create the table
17 spark.sql(create_table_query)
[2] ✓ 20 sec - Command executed in 20 sec 613 ms by shubha on 12:08:38 AM, 5/06/25
PySpark (Python) ▾

```

> [Spark jobs \(1 of 1 succeeded\)](#) [Resources](#) [Log](#) ...

Load the Target Delta Table

```
1 from delta.tables import DeltaTable
2 target_path = "Tables/Gold_layer/Account_SCD"
3 delta_target = DeltaTable.forPath(spark, target_path)
[3] ✓ <1 sec - Command executed in 922 ms by shubha on 12:09:16 AM, 5/06/25
```

PySpark (Python) ▾

...

Generate Hash Key for Source Data

Markdown preview

```
1 from pyspark.sql.functions import *
2 df_src1= df_source.withColumn("hash_key",crc32(concat(*df_source.columns)))
[4] ✓ <1 sec - Command executed in 356 ms by shubha on 12:09:43 AM, 5/06/25
```

PySpark (Python) ▾

> Log

...

...

Find Changed or New Records

```
1 df_src1=df_src1.alias("src").join(delta_target.toDF().alias("tgt"),
2 ((col("src.account_id")==col("tgt.account_id"))&
3 (col("src.hash_key")==col("tgt.hash_key"))),"anti").select(col("src.*"))
4
[5] ✓ <1 sec - Command executed in 357 ms by shubha on 12:10:21 AM, 5/06/25
```

PySpark (Python) ▾

...

```
1 from pyspark.sql.functions import col
2
3 delta_target.alias("tgt").merge(df_src1.alias("src"),
4 "tgt.account_id = src.account_id")\
5     .whenMatchedUpdate(set={"tgt.account_id":"src.account_id",
6         "tgt.customer_id":"src.customer_id","tgt.account_type":"src.account_type",
7         "tgt.balance":"src.balance","tgt.hash_key":"src.hash_key",
8         "tgt.updated_date":current_timestamp(),"tgt.updated_by":lit("databricks_Updated"))}\\
9     .whenNotMatchedInsert(values={"tgt.account_id":"src.account_id",
10        "tgt.customer_id":"src.customer_id","tgt.account_type":"src.account_type",
11        "tgt.balance":"src.balance","tgt.hash_key":"src.hash_key",
12        "tgt.created_date":current_timestamp(),
13        "tgt.created_by":lit("databricks"),"tgt.updated_date":current_timestamp(),
14        "tgt.updated_by":lit("databricks")}).execute()
15
16 display(spark.read.format("delta").option("header","true").load(target_path))
[6] ✓ 14 sec - Command executed in 14 sec 100 ms by shubha on 12:12:35 AM, 5/06/25
```

PySpark (Python) ▾

> Spark jobs (20 of 20 succeeded) Resources Log

...

[6] 15 `display(spark.read.format("delta").option("header","true").load(target_path))`

16 `display(spark.read.format("delta").option("header","true").load(target_path))`

✓ 14 sec - Command executed in 14 sec 100 ms by shubha on 12:12:35 AM, 5/06/25

PySpark (Python) ▾

> Spark jobs (20 of 20 succeeded) Resources Log ...

Table view

Table

+ New chart

9 columns, 100 rows ▾

Download Search

Inspect

	123 account_id	123 customer_id	ABC account_type	12F balance	12L hash_key	ABC created_by	⌚ created_date	ABC updated_by	⌚ updated
1	45	Savings	1000.5	4261402674	databricks	2025-05-06 07:1...	databricks	2025-05-06	
2	12	Checking	2500.75	796571617	databricks	2025-05-06 07:1...	databricks	2025-05-06	
64	12	Checking	6500.0	4285205776	databricks	2025-05-06 07:1...	databricks	2025-05-06	
82	2	Checking	8300.5	1578107828	databricks	2025-05-06 07:1...	databricks	2025-05-06	
3	78	Savings	1500.0	3519015933	databricks	2025-05-06 07:1...	databricks	2025-05-06	
11	3	Savings	1100.75	744264830	databricks	2025-05-06 07:1...	databricks	2025-05-06	
4	34	Checking	3000.25	761699333	databricks	2025-05-06 07:1...	databricks	2025-05-06	
78	4	Checking	7900.5	2650882798	databricks	2025-05-06 07:1...	databricks	2025-05-06	
5	56	Savings	500.0	2866974084	databricks	2025-05-06 07:1...	databricks	2025-05-06	
0	18	5	Checking	1600.5	2895374357	databricks	2025-05-06 07:1...	databricks	2025-05-06

1 `create_table_query = """`

2 `CREATE TABLE Customer_SCD1 (`

3 `customer_id int,`

4 `first_name string,`

5 `last_name string,`

6 `address string,`

7 `city string,`

8 `state string,`

9 `zip string,`

10 `hash_key BIGINT,`

11 `created_by STRING,`

12 `created_date TIMESTAMP,`

13 `updated_by STRING,`

14 `updated_date TIMESTAMP`

15 `)`

16 `USING DELTA`

17 `LOCATION 'Tables/Gold_layer/Customer_SCD1'`

18 `""""`

19 `# Execute the query to create the table`

20 `spark.sql(create_table_query)`

✓ - Session ready in 12 sec 234 ms. Command executed in 24 sec 586 ms by shubha on 5:46:56 PM, 5/05/25

PySpark (Python) ▾

... DataFrame[]

```
1  create_table_query = """  
2  CREATE TABLE Customer_SCD1 (  
3      customer_id int,  
4      first_name string,  
5      last_name string,  
6      address string,  
7      city string,  
8      state string,  
9      zip string,  
10     hash_key BIGINT,  
11     created_by STRING,  
12     created_date TIMESTAMP,  
13     updated_by STRING,  
14     updated_date TIMESTAMP  
15    )  
16    USING DELTA  
17    LOCATION 'Tables/Gold_layer/Customer_SCD1'  
18    """  
19    # Execute the query to create the table  
20    spark.sql(create_table_query)  
[8] ✓ 6 sec - Command executed in 6 sec 213 ms by shubha on 12:15:42 AM, 5/06/25 PySpark (Pyth
```

```
1  from pyspark.sql.functions import *  
2  df_source = spark.read.format("delta").load("Tables/dbo/customers")  
3  df_source.createOrReplaceTempView("source_view")  
4  src_path="Tables/dbo/customers"  
5  print(src_path)  
[2] ✓ - Command executed in 1 sec 520 ms by shubha on 5:47:06 PM, 5/05/25 PySpark (Python) ▾  
  ↴ M ↵ Lx ↷ ↸ ...  
✓  
1  
2  df_src1=df_source.withColumn("hash_key",crc32(concat(*df_source.columns)))  
3  
[3] ✓ <1 sec - Command executed in 348 ms by shubha on 12:16:23 AM, 5/06/25 PySpark (Python) ▾  
..
```

```
1  
2  from delta.tables import DeltaTable  
3  tgt_path = "Tables/Gold_layer/Customer_SCD1"  
4  dbtable = DeltaTable.forPath(spark, tgt_path)  
5  dbtable.toDF().show()  
6  
[4] ✓ - Command executed in 3 sec 509 ms by shubha on 5:47:58 PM, 5/05/25 PySpark (Python) ▾  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
|customer_id|first_name|last_name|address|city|state|zip|hash_key|created_by|created_date|updated_by|updated_date|  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
|-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
|-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
|-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
|-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

1 df_src1=df_src1.alias("src").join(dbtable.toDF().alias("tgt"),
2 ((col("src.customer_id") == col("tgt.customer_id")) & (col("src.hash_key") == col("tgt.has
3 "anti")).select(col("src.*"))
4
5

] ✓ <1 sec - Command executed in 367 ms by shubha on 12:29:32 AM, 5/06/25 PySpark (Python) ▾

▶ | ▾ 1 dbtable.alias("tgt").merge(df_src1.alias("src"),"tgt.customer_id = src.customer_id")\n2 .whenMatchedUpdate(set={"tgt.customer_id":"src.customer_id","tgt.first_name":"src.firs\n3 .whenNotMatchedInsert(values={"tgt.customer_id":"src.customer_id","tgt.first_name"\n4\n5

[34] ✓ 2 sec - Command executed in 2 sec 462 ms by shubha on 12:30:34 AM, 5/06/25 PySpark (Python) ▾

> Spark jobs (9 of 9 succeeded) Resources Log ...

> | ▾ 1 display(spark.read.format("delta").option("header","true").load(tgt_path))
[7] ✓ - Command executed in 3 sec 541 ms by shubha on 5:50:14 PM, 5/05/25 PySpark (Python) ▾

Table view

customer_id	ABC first_name	ABC last_name	ABC address	ABC city	ABC state	ABC zip	12L hash_key	ABC created_by	created_date
	John	Doe	123 Elm St	Toronto	ON	M4B1B3	2373314683	databricks	2025-05-06 00:4...
	Jane	Smith	456 Maple A...	Ottawa	ON	K1A0B1	3911254336	databricks	2025-05-06 00:4...
	Michael	Johnson	789 Oak Dr	Montreal	QC	H1A1A1	4114944968	databricks	2025-05-06 00:4...
	Emily	Davis	101 Pine Rd	Calgary	AB	T2A0A1	165372536	databricks	2025-05-06 00:4...
	David	Wilson	202 Birch Blvd	Vancouver	BC	V5K0A1	3284019540	databricks	2025-05-06 00:4...
	Emma	Clark	505 Cedar St	Halifax	NS	B3H0A1	4268191859	databricks	2025-05-06 00:4...
	James	Martinez	606 Spruce Ln	Winnipeg	MB	R3C0A1	4278935663	databricks	2025-05-06 00:4...
	Olivia	Garcia	707 Fir St	Edmonton	AB	T5A0A1	443479193	databricks	2025-05-06 00:4...
	William	Lopez	808 Redwoo...	Victoria	BC	V8W0A1	4188371369	databricks	2025-05-06 00:4...

A SQL analytics endpoint for SQL querying was created with this item.

Explorer account_scd Showing 100 rows

#	123 account_id	123 customer_id	ABC account_type	12F balance	12L hash_key	ABC created_by	created_date	ABC upd...
1	1	45	Savings	1000.5	4261402674	databricks	5/6/2025 7:12:21...	databrick
2	2	12	Checking	2500.75	796571617	databricks	5/6/2025 7:12:21...	databrick
3	64	12	Checking	6500	4285205776	databricks	5/6/2025 7:12:21...	databrick
4	82	2	Checking	8300.5	1578107828	databricks	5/6/2025 7:12:21...	databrick
5	3	78	Savings	1500	3519015933	databricks	5/6/2025 7:12:21...	databrick
6	11	3	Savings	1100.75	744264830	databricks	5/6/2025 7:12:21...	databrick
7	4	34	Checking	3000.25	744264830	databricks	5/6/2025 7:12:21...	databrick
8	78	4	Checking	7900.5	2650882798	databricks	5/6/2025 7:12:21...	databrick
9	5	56	Savings	500	2866974084	databricks	5/6/2025 7:12:21...	databrick
10	18	5	Checking	1600.5	2895374357	databricks	5/6/2025 7:12:21...	databrick
11	6	23	Checking	1200.5	3079739098	databricks	5/6/2025 7:12:21...	databrick
12	48	6	Checking	4900	3216528439	databricks	5/6/2025 7:12:21...	databrick
13	7	89	Savings	800.75	74260363	databricks	5/6/2025 7:12:21...	databrick

Explorer <>

Search tables

- project4
 - Tables
 - > account_scd
 - > accounts
 - > customer_scd1
 - > customers
 - > loan_payment
 - > loan_payments
 - > loans
 - > loans1
 - > transactions

customer_scd1 Showing 86 rows

#	123 customer_id	ABC first_name	ABC last_name	ABC address	ABC city	ABC state	ABC zip	12L hash
1	1	John	Doe	123 Elm St	Toronto	ON	M4B1B3	2373314c
2	2	Jane	Smith	456 Maple Ave	Ottawa	ON	K1A0B1	3911254:
3	3	Michael	Johnson	789 Oak Dr	Montreal	QC	H1A1A1	4114944f
4	4	Emily	Davis	101 Pine Rd	Calgary	AB	T2A0A1	1653725:
5	5	David	Wilson	202 Birch Blvd	Vancouver	BC	V5K0A1	3284019!
6	6	Emma	Clark	505 Cedar St	Halifax	NS	B3H0A1	4268191t
7	7	James	Martinez	606 Spruce Ln	Winnipeg	MB	R3C0A1	4278935t
8	8	Olivia	Garcia	707 Fir St	Edmonton	AB	T5A0A1	4434791!
9	9	William	Lopez	808 Redwood Dr	Victoria	BC	V8W0A1	4188371:
10	10	Ava	Anderson	909 Cypress Ave	Quebec City	QC	G1A0A1	9807816k
11	11	Alexander	Thomas	1010 Willow Rd	St. John's	NL	A1A0A1	8230925:
12	12	Isabella	Lee	1111 Poplar St	Fredericton	NB	E3B0A1	2367321!
13	13	Daniel	Harris	1212 Ash Blvd	Charlottetown	PE	C1A0A1	2952121t

(● Succeeded (11 sec 265 ms)) Columns 12 Rows 86

Explorer <>

Search tables

- loan_payment
 - > accounts
 - > customer_scd1
 - > customers
 - > loan_payment
 - > loan_payments
 - > loans
 - > loans1
 - > transactions
 - > transactions1
 - > Unidentified
 - > Files

loan_payment Showing 104 rows

#	123 payment_id	123 loan_id	12F payment_da...	12F payment_a...	12L hash_key	ABC created_by	12F created_date	ABC upd...
1	1	45	1/1/2024 12:00:...	100	3466784654	databricks	5/6/2025 7:32:40...	databrick
2	99	1	4/8/2024 12:00:...	5000	1369522191	databricks	5/6/2025 7:32:40...	databrick
3	2	23	1/2/2024 12:00:...	150	1385209747	databricks	5/6/2025 7:32:40...	databrick
4	91	2	3/31/2024 12:00:...	4600	247445288	databricks	5/6/2025 7:32:40...	databrick
5	91	2	3/31/2024 12:00:...	4600	247445288	databricks	5/6/2025 7:32:40...	databrick
6	3	67	1/3/2024 12:00:...	200	2829437187	databricks	5/6/2025 7:32:40...	databrick
7	4	89	1/4/2024 12:00:...	250	2467606655	databricks	5/6/2025 7:32:40...	databrick
8	33	64	2/2/2024 12:00:...	1700	4018524585	databricks	5/6/2025 7:32:40...	databrick
9	33	64	2/2/2024 12:00:...	1700	4018524585	databricks	5/6/2025 7:32:40...	databrick
10	5	12	1/5/2024 12:00:...	300	3225416827	databricks	5/6/2025 7:32:40...	databrick
11	6	34	1/6/2024 12:00:...	350	359778397	databricks	5/6/2025 7:32:40...	databrick
12	82	3	3/22/2024 12:00:...	4150	3991820484	databricks	5/6/2025 7:32:40...	databrick
13	7	56	1/7/2024 12:00:...	400	2064246126	databricks	5/6/2025 7:32:40...	databrick

(● Succeeded (10 sec 102 ms)) Columns 9 Rows 104

● A SQL analytics endpoint for SQL querying was created with this item. X

Explorer <>

Search tables

- loans1
 - > accounts
 - > customer_scd1
 - > customers
 - > loan_payment
 - > loan_payments
 - > loans
 - > loans1
 - > transactions
 - > transactions1
 - > Unidentified
 - > Files

loans1 Showing 104 rows

loan_id	123 customer_id	12F interest_rate	123 loan_term	12F loan_amount	12L hash_key	ABC created_by	12F created_date	AI
	45	5.5	36	10000.5	1485553409	databricks	5/6/2025 7:35:29...	d
	12	4.5	48	20000.75	3355992476	databricks	5/6/2025 7:35:29...	d
	12	4.5	48	20000.75	3355992476	databricks	5/6/2025 7:35:29...	d
	12	3	24	30000	2213555697	databricks	5/6/2025 7:35:29...	d
	12	3	24	30000	2213555697	databricks	5/6/2025 7:35:29...	d
	78	6	60	15000	1716007356	databricks	5/6/2025 7:35:29...	d
	34	3.5	24	30000.25	2847880128	databricks	5/6/2025 7:35:29...	d
	2	4.5	48	20000.5	3935580499	databricks	5/6/2025 7:35:29...	d
	56	5	36	25000	951778087	databricks	5/6/2025 7:35:29...	d
	23	4	48	17500.5	3923234150	databricks	5/6/2025 7:35:29...	d
	3	6	60	10000.75	2179146733	databricks	5/6/2025 7:35:29...	d
	89	6.5	60	22500.75	1348082437	databricks	5/6/2025 7:35:29...	d
	67	3	24	27500	1488692577	databricks	5/6/2025 7:35:29...	d

(● Succeeded (12 sec 137 ms)) Columns 10 Rows 104

Explorer

transactions1

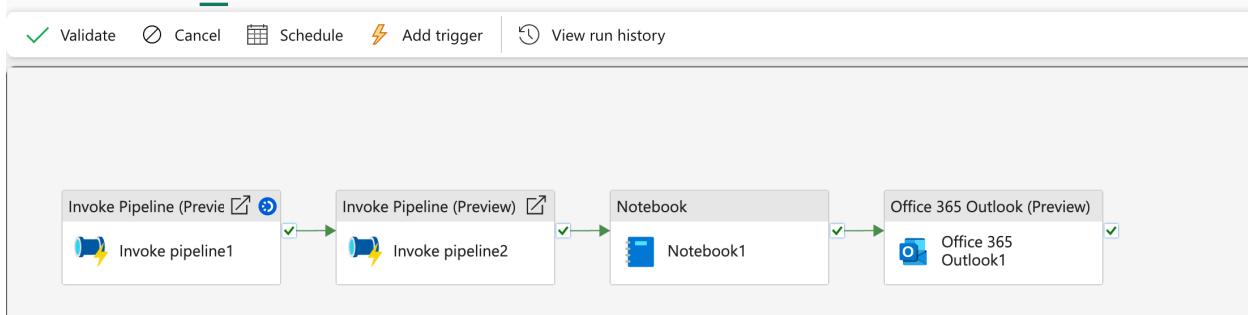
Showing 105 rows

Search tables

transaction_id account_id transaction_date transaction_amount transaction_type hash_key created_by created_at

	transaction_id	account_id	transaction_date	transaction_amount	transaction_type	hash_key	created_by	created_at
1	1	45	1/1/2024 12:00:00	100.5	Deposit	4252403073	databricks	5/6/2025
2	2	12	1/2/2024 12:00:00	200.75	Withdrawal	463284897	databricks	5/6/2025
3	64	12	3/4/2024 12:00:00	300.25	Withdrawal	3270191256	databricks	5/6/2025
4	88	1	3/28/2024 12:00:00	275.75	Withdrawal	777307740	databricks	5/6/2025
5	3	78	1/3/2024 12:00:00	150	Deposit	3567602057	databricks	5/6/2025
6	6	23	1/6/2024 12:00:00	175	Withdrawal	1611457425	databricks	5/6/2025
7	4	34	1/4/2024 12:00:00	300.25	Withdrawal	2754677836	databricks	5/6/2025
8	82	2	3/22/2024 12:00:00	200.75	Withdrawal	3196733283	databricks	5/6/2025
9	82	2	3/22/2024 12:00:00	200.75	Withdrawal	3196733283	databricks	5/6/2025
10	5	56	1/5/2024 12:00:00	250	Deposit	1545993203	databricks	5/6/2025
11	14	64	1/14/2024 12:00:00	300.25	Withdrawal	3926257497	databricks	5/6/2025
12	14	64	1/14/2024 12:00:00	300.25	Withdrawal	3926257497	databricks	5/6/2025
13	8	67	1/8/2024 12:00:00	275.75	Withdrawal	1426350130	databricks	5/6/2025

Succeeded (11 sec 13 ms) Columns 10 Rows 105



Parameters Variables Settings Output Library variables (preview)

Validate Cancel Schedule Add trigger View run history

Invoke Pipeline (Preview) Invoke Pipeline (Preview) Notebook Office 365 Outlook (Preview)

General Settings

Signed in as Shubhashini.Kurumanarsimhulu@student.ufv.ca Change account

To * Shubhashini.Kurumanarsimhulu@student.ufv.ca

Subject * Pipeline successful

Body *
 Hi,

Office 365 Outlook (Preview)
 Office 365 Outlook1