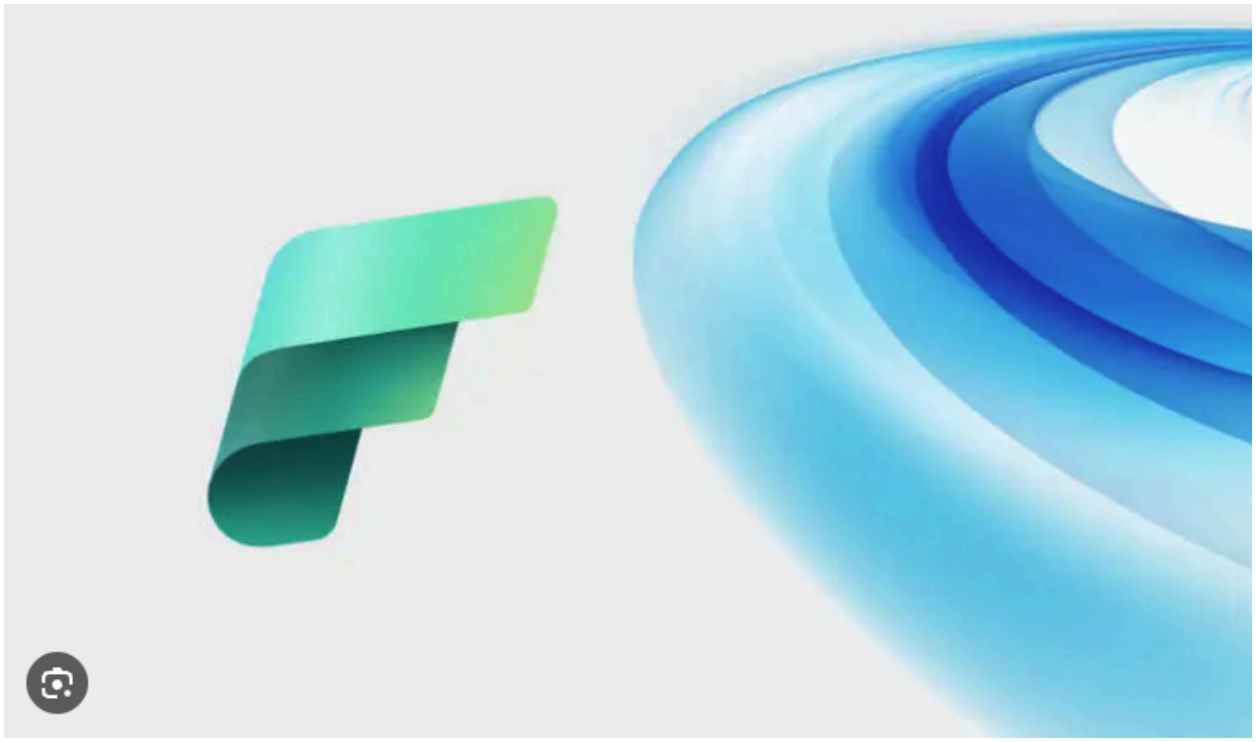


Incremental Data Loading and Automated Notifications using Microsoft Fabric

—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.

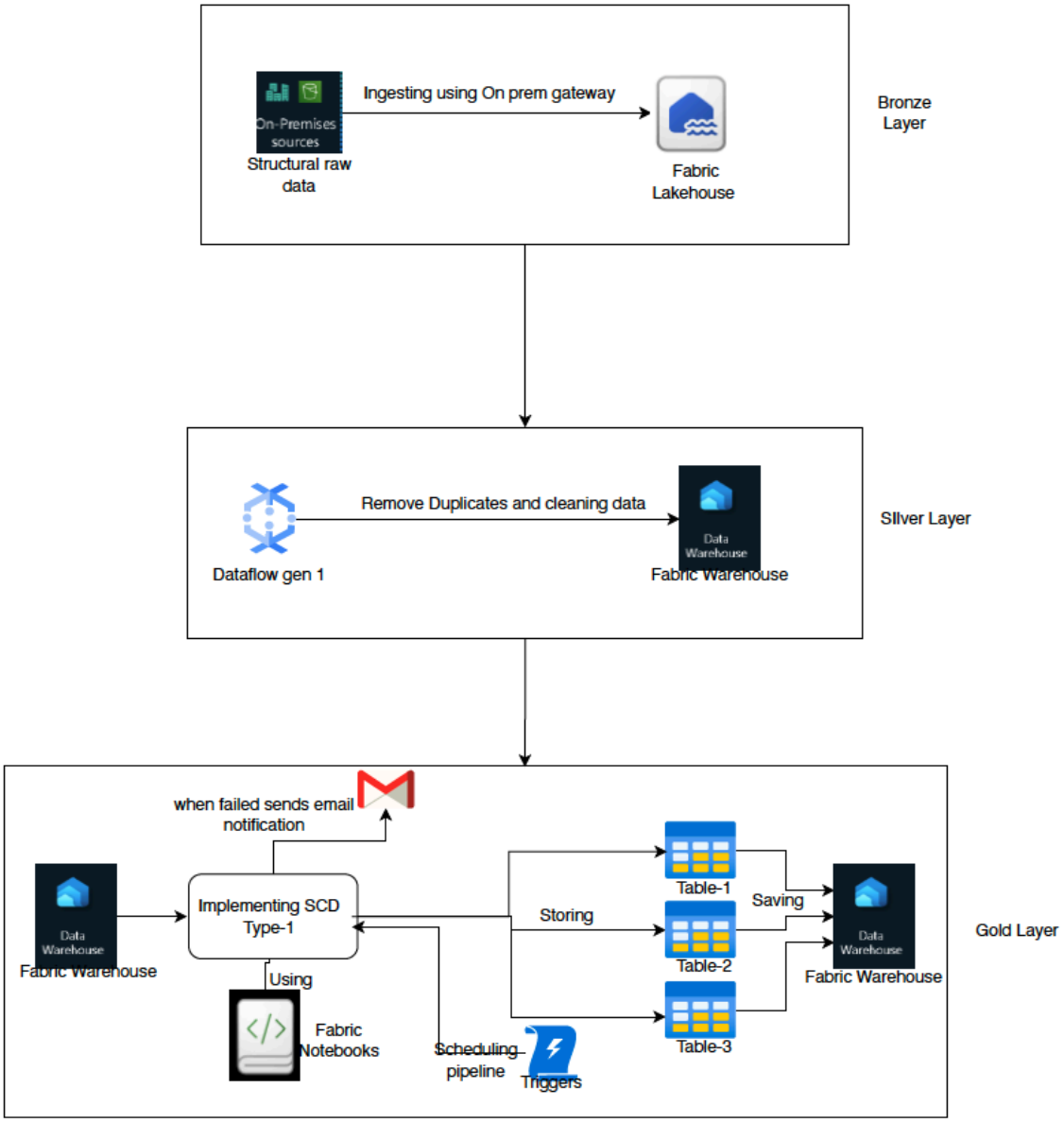
3. Apply data cleansing operations—such as joins, duplicate removal, and formatting—using Dataflow Gen1.

4. Load the cleansed data into a Fabric Warehouse for downstream analytics.

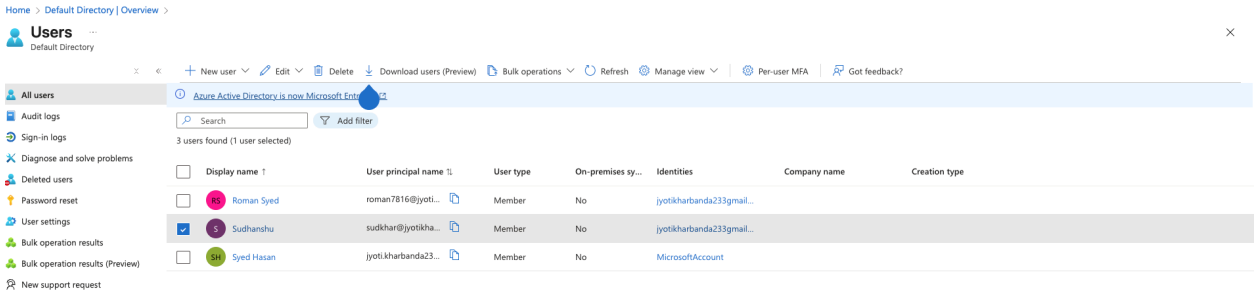
5. Implement Slowly Changing Dimension (SCD) Type 1 logic with Fabric Notebooks to maintain data consistency in dimension tables.

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

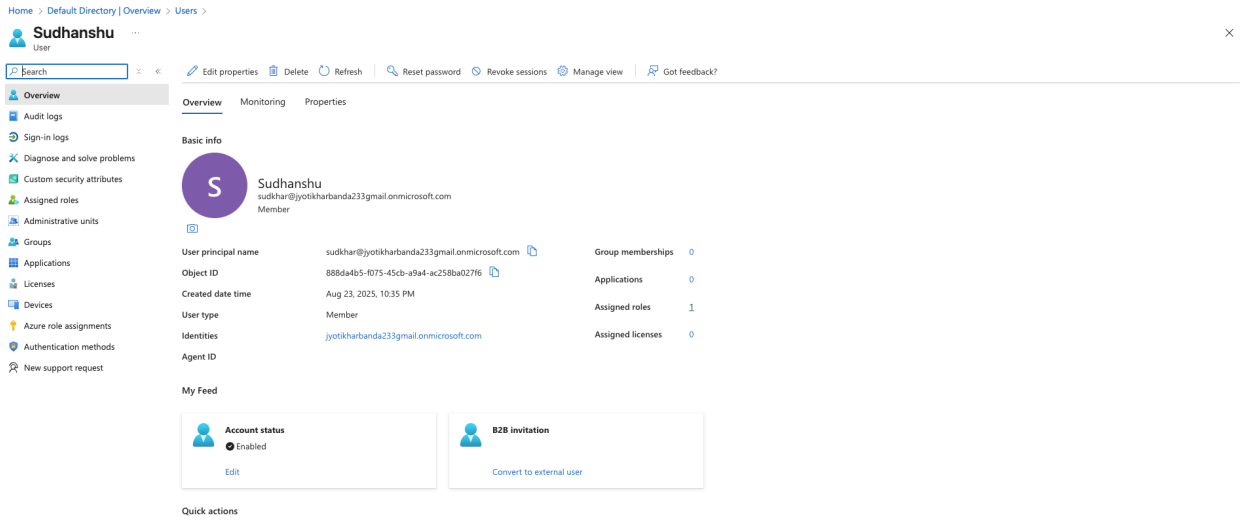
Architecture Diagram



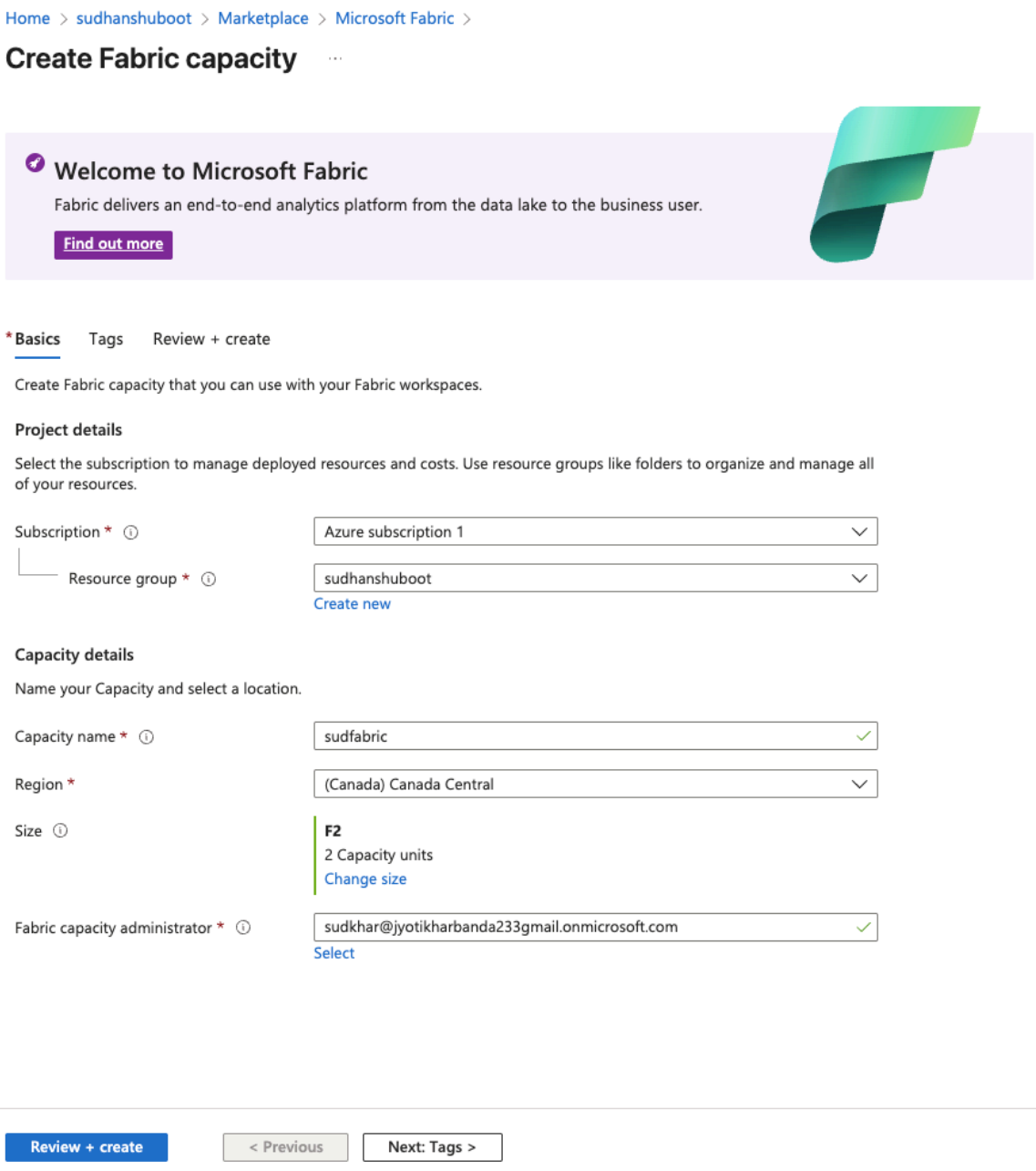
Click on the entraID for provide the permission for enabling fabric for the account.

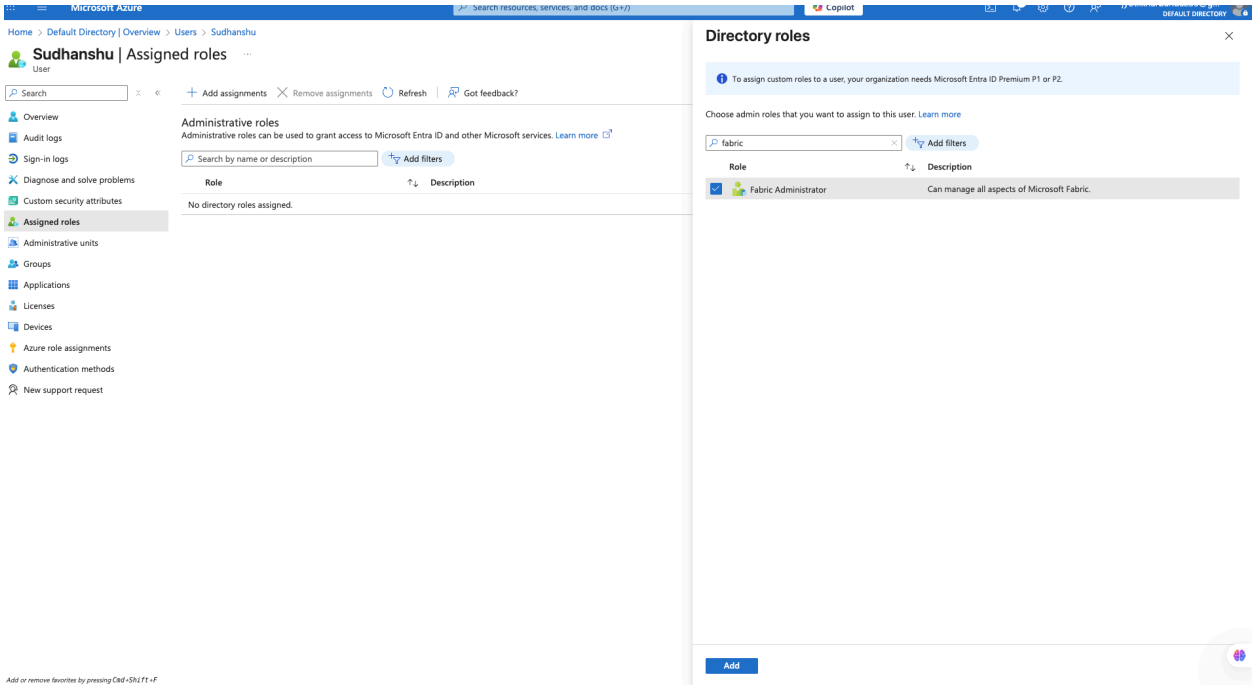


Click on the assigned roles and add the fabric administrator role to the user

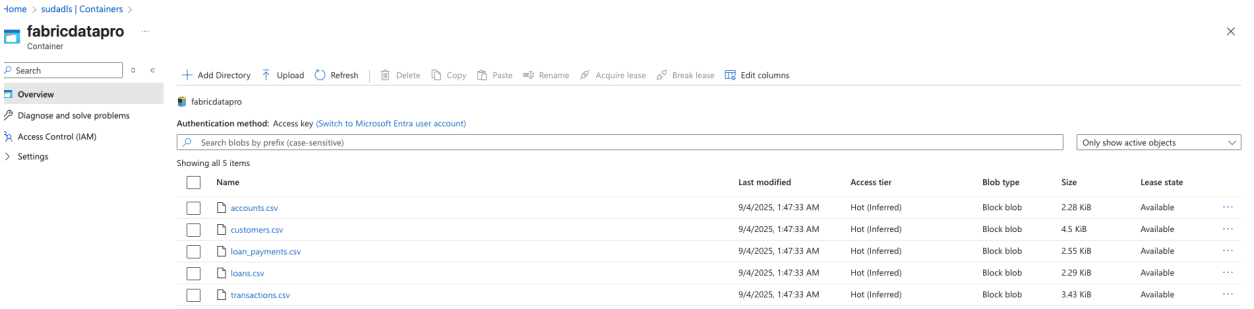


Go to Azure and search for the fabric capacity. And create the fabric capacity.





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

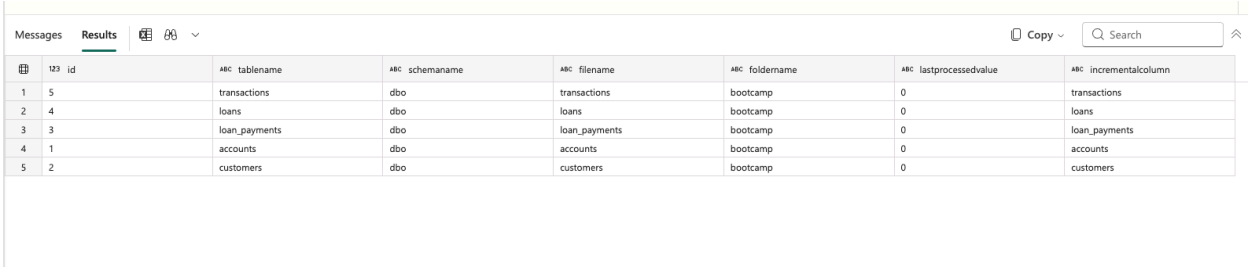


Code for creating a watermark table for incremental loading

```
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

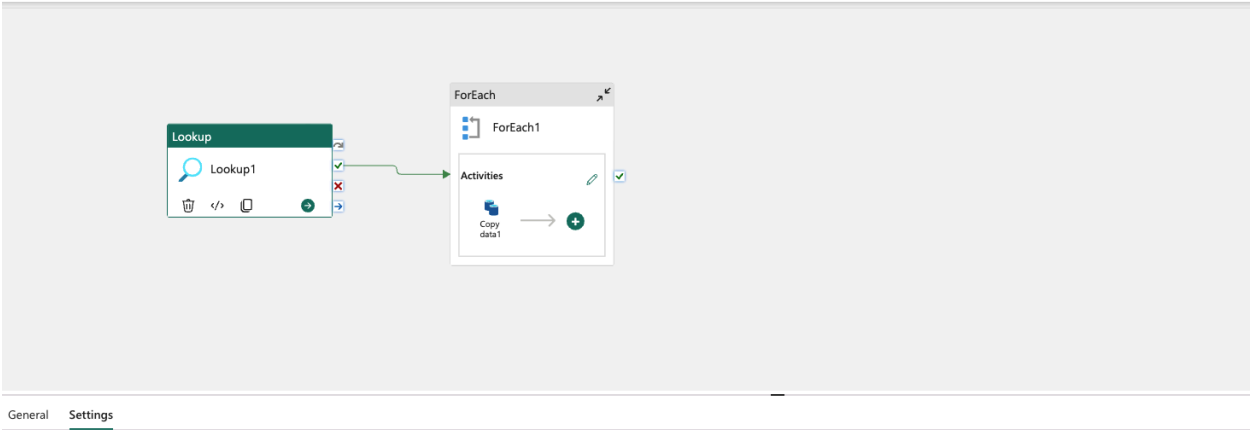


Messages Results

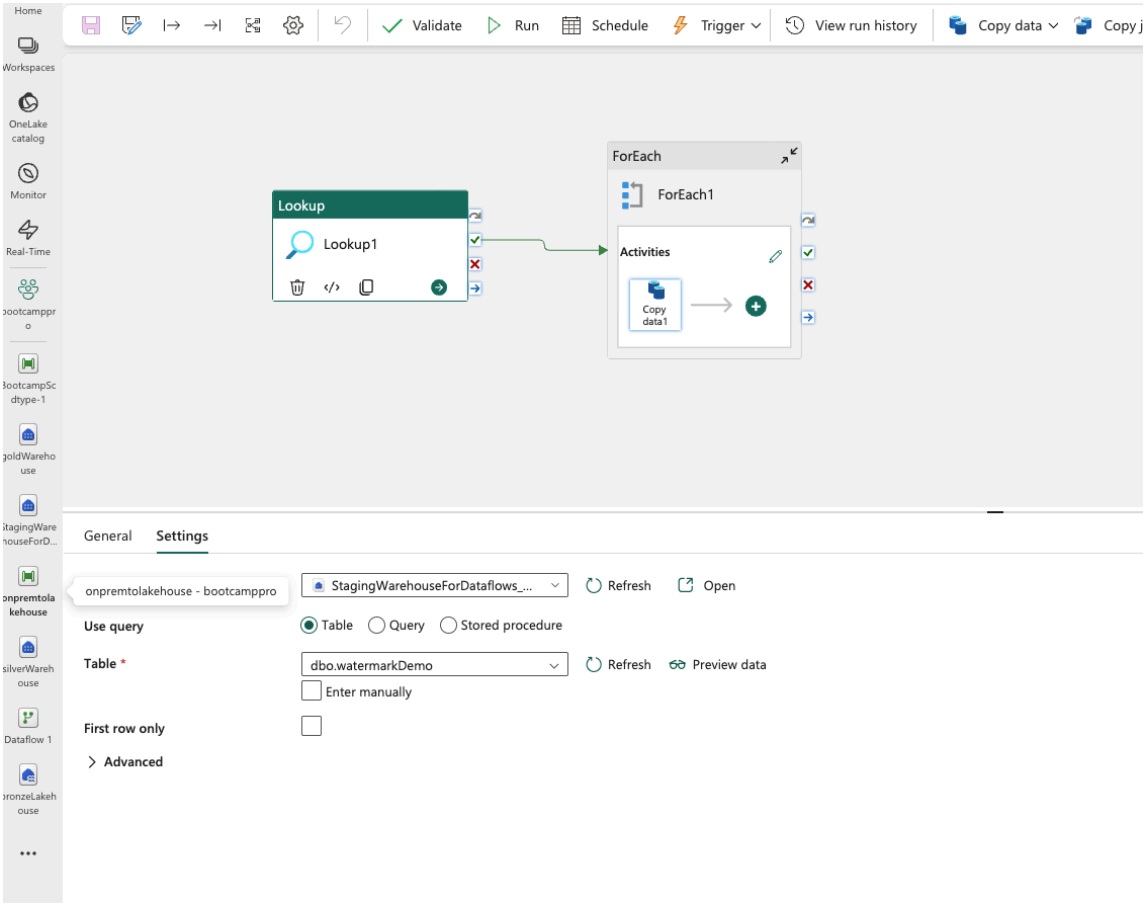
Copy Search

	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

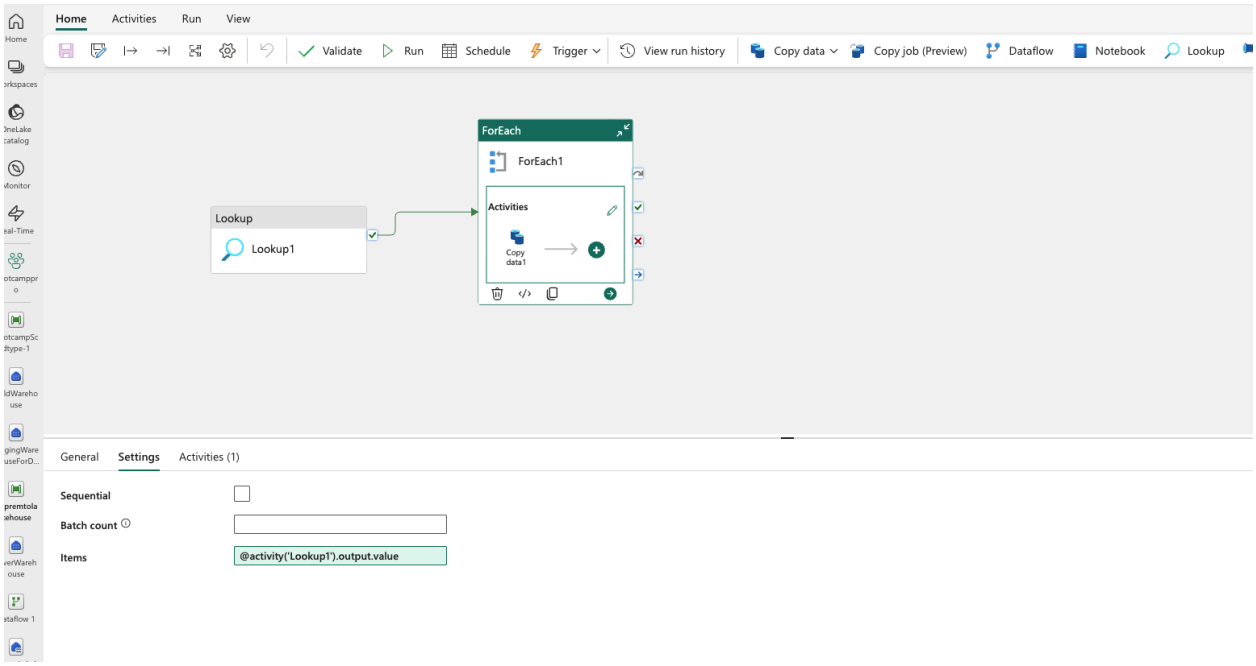
We are not doing incremental loading since we have to implement scd type-2 and updated rows will not come in the incremental loading, hence we are loading multiples files using lookup. This is how pipeline for loading data in the bronze layer will look like.



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

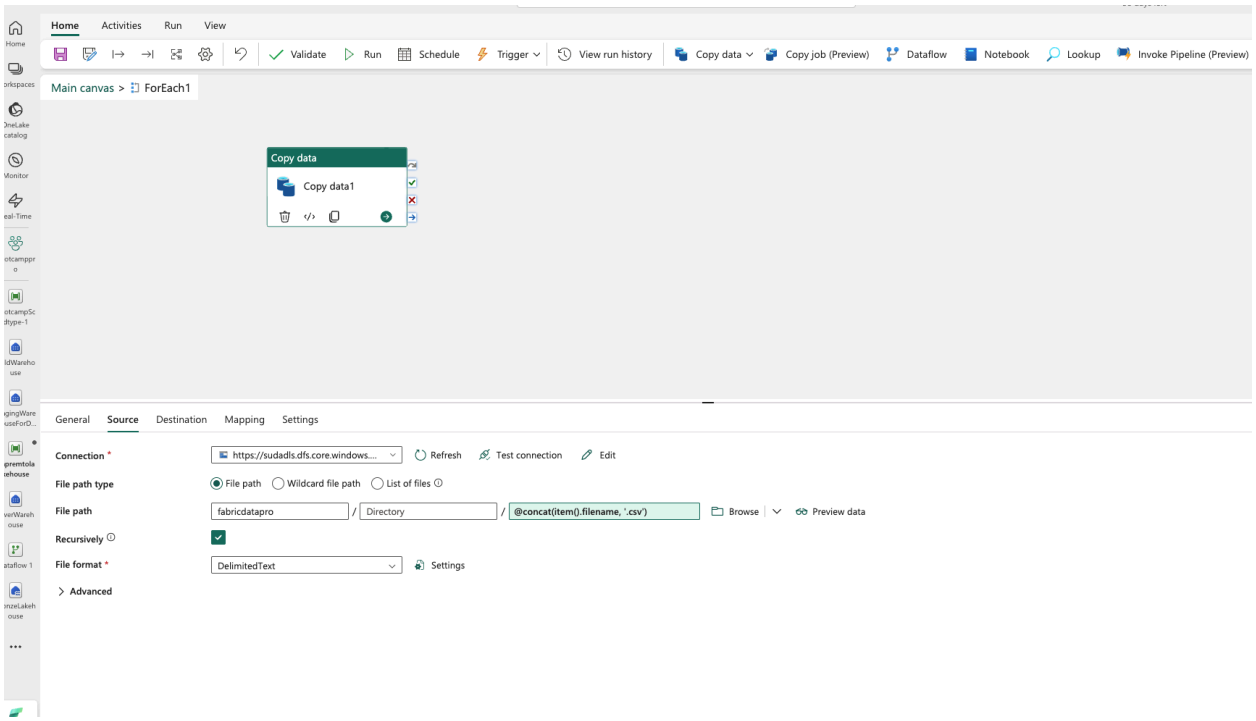


Adding the for each activity and adding the following expression.

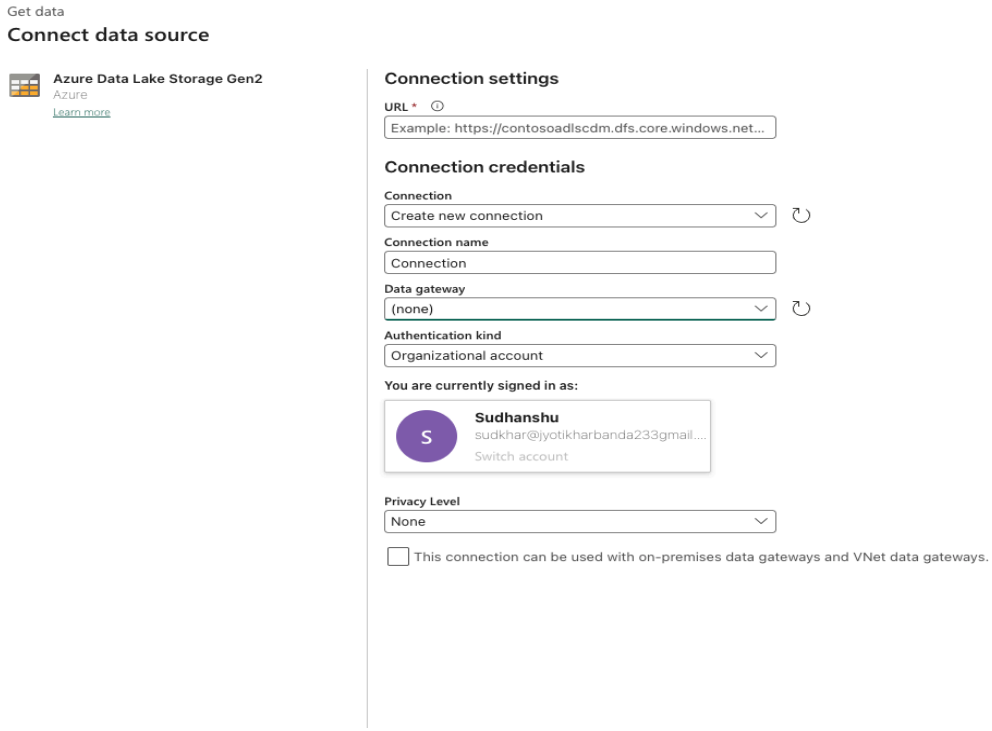



```
@activity('Lookup1').output.value
```

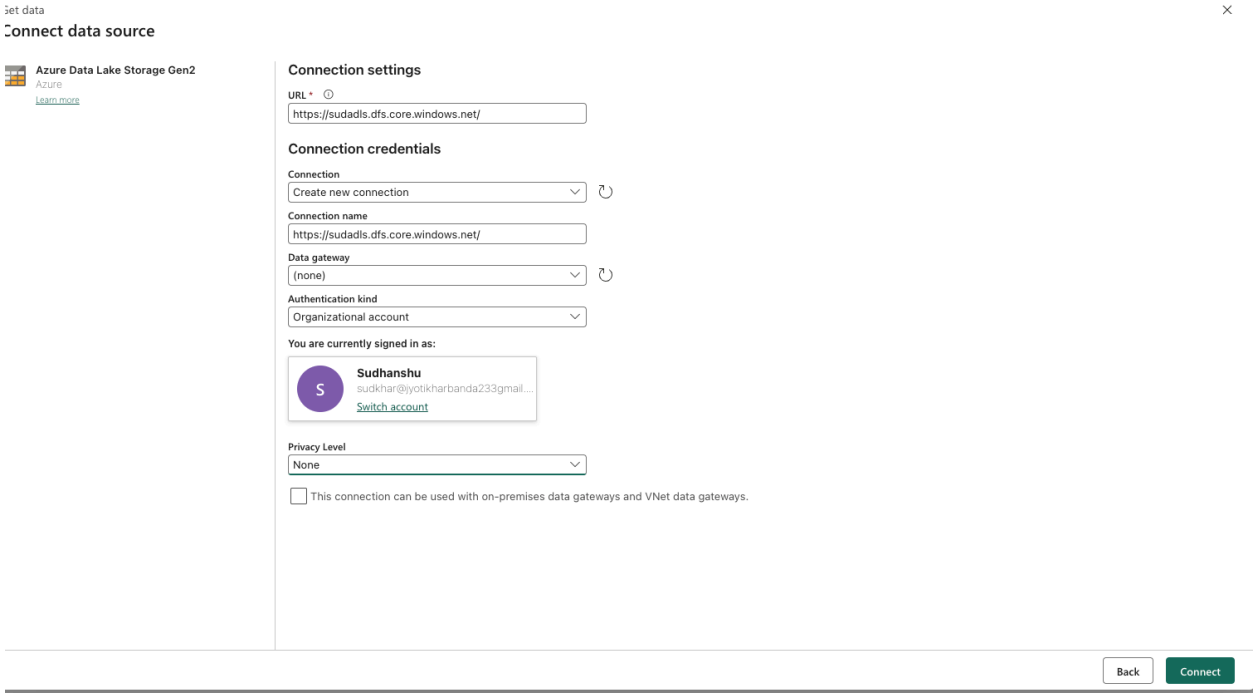
Going inside the for each activity adding the copy data activity. In the copy data add the source path and the destination path.



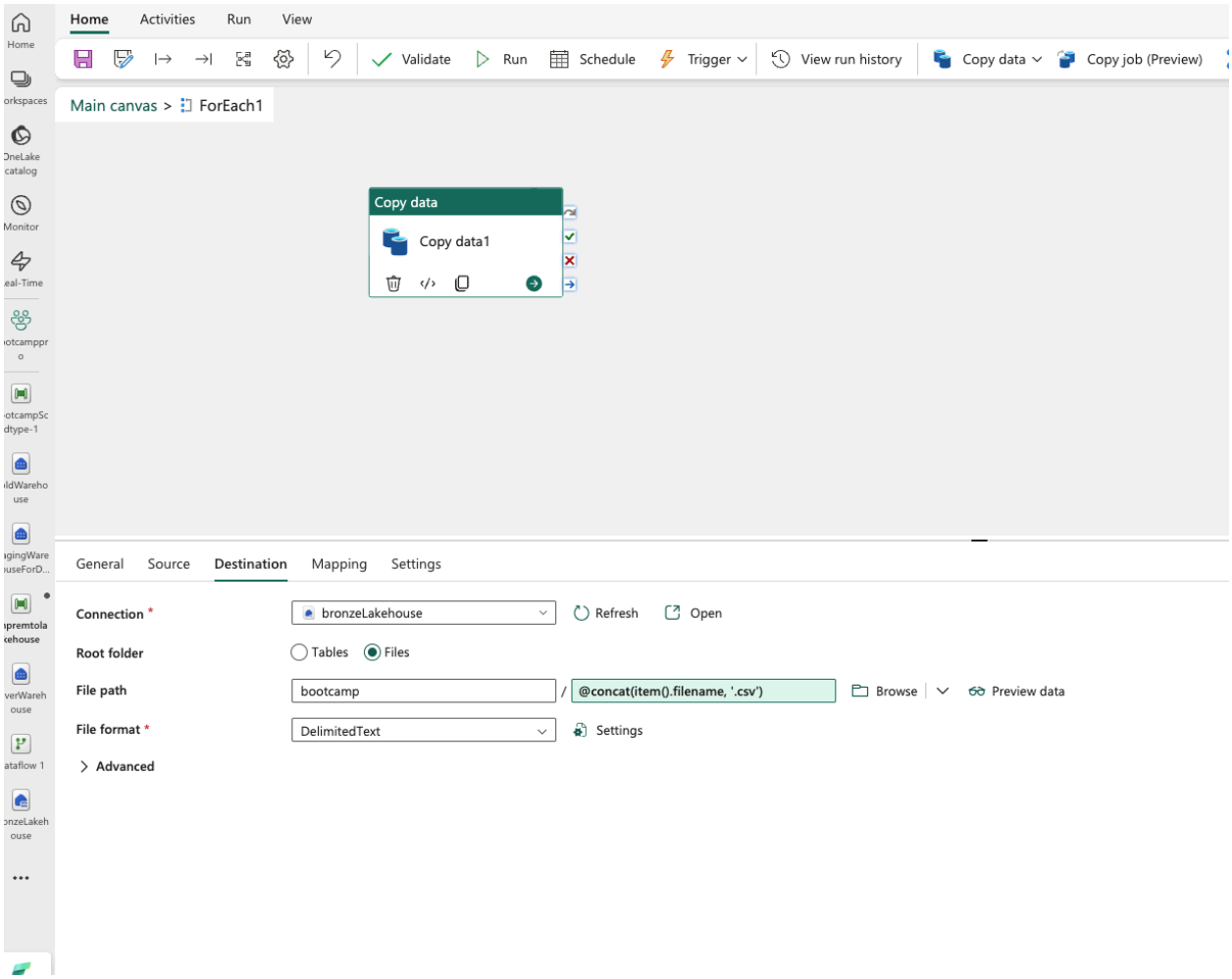
Creating the connection in the copy data activity to connect ADLS Gen2 storage.



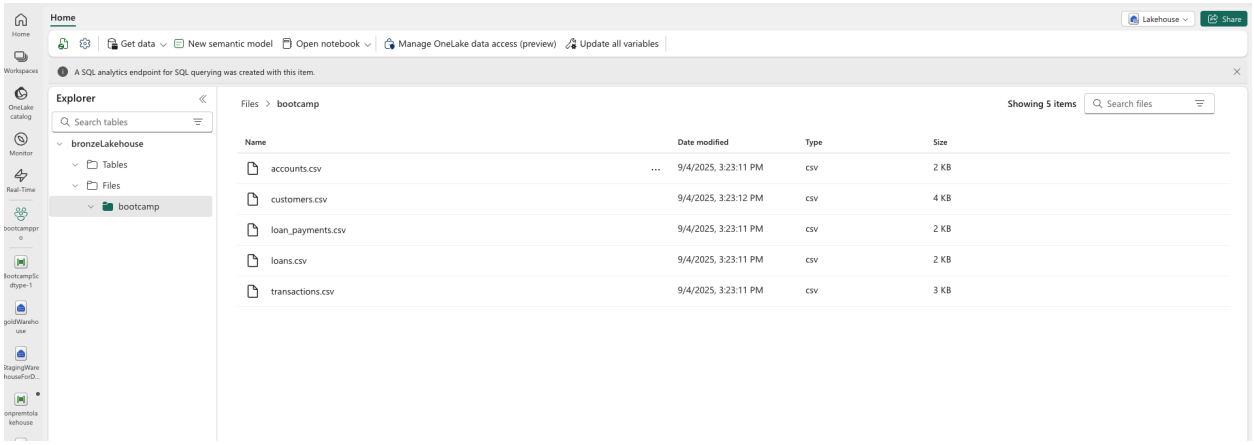
Providing the URL, the structure can be found on the internet.



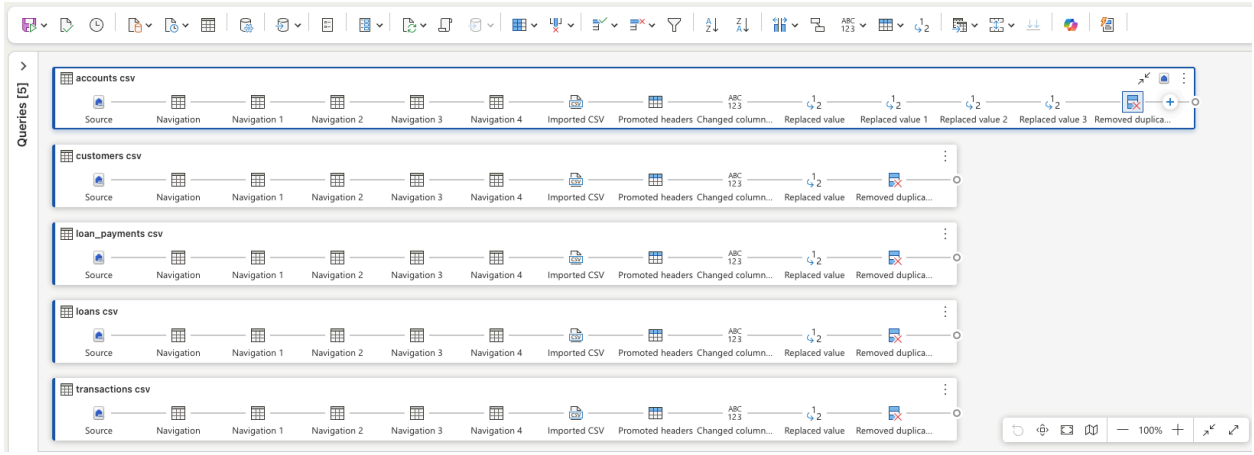
Providing the destination URL.



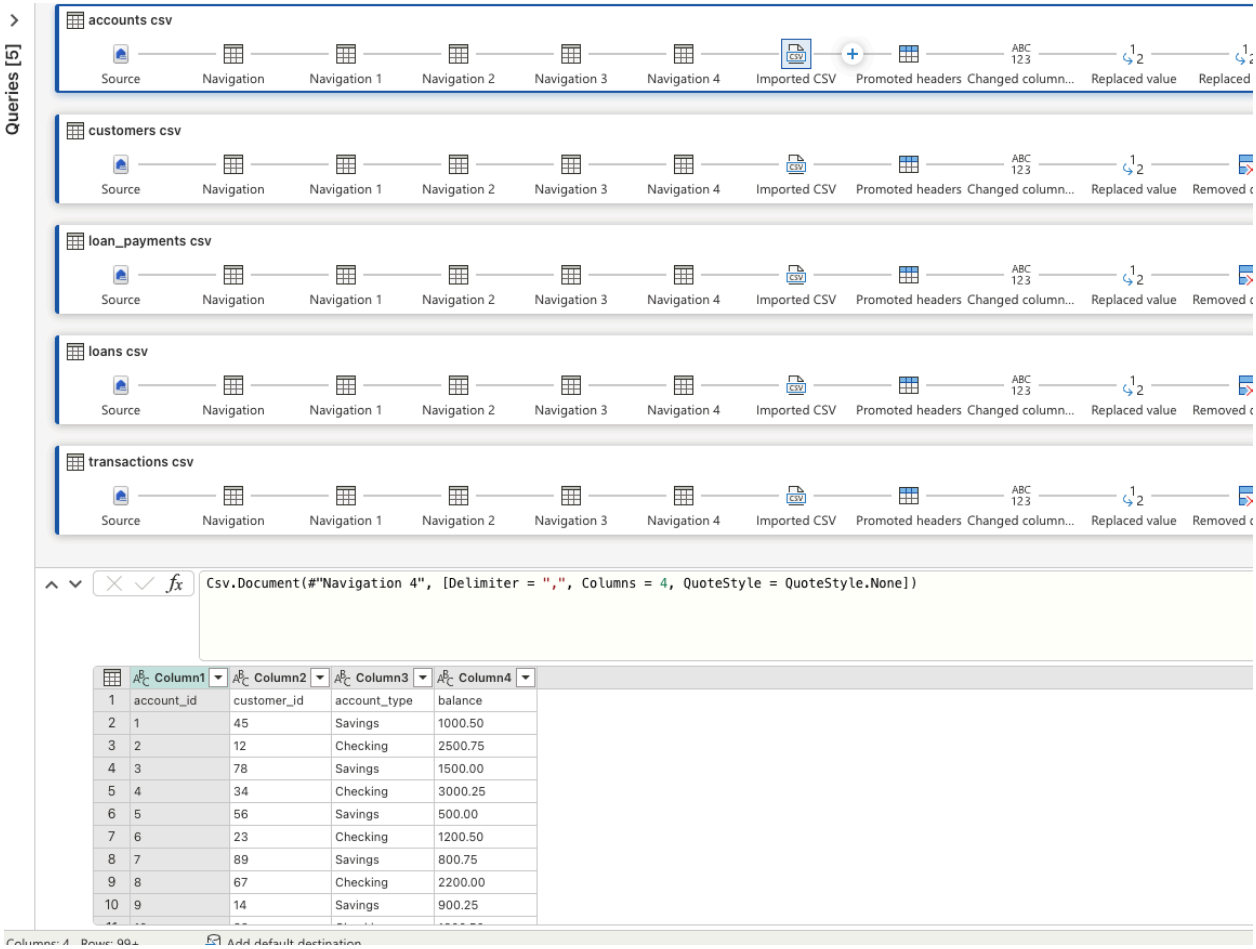
RUN THE pipeline, you can the data is copied from ADLS Gen2 to Fabric Lake House.



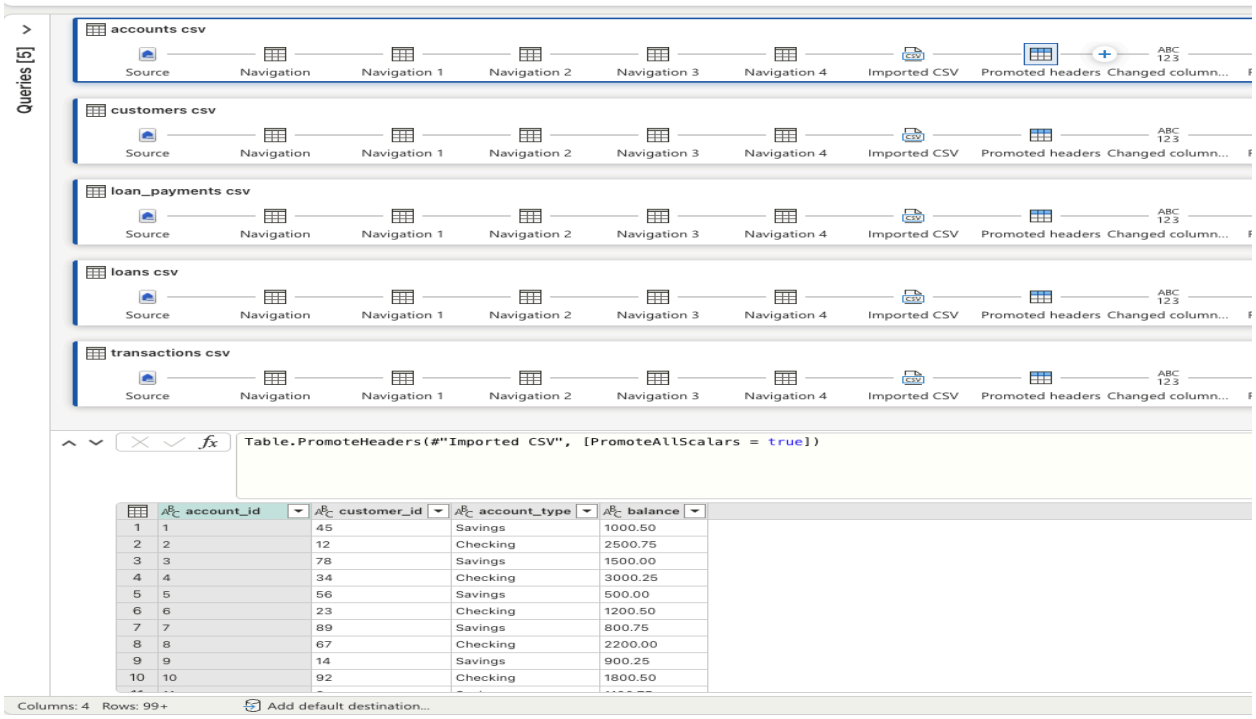
Going to the dataflow gen2 and creating the dataflow for cleaning the CSV file.



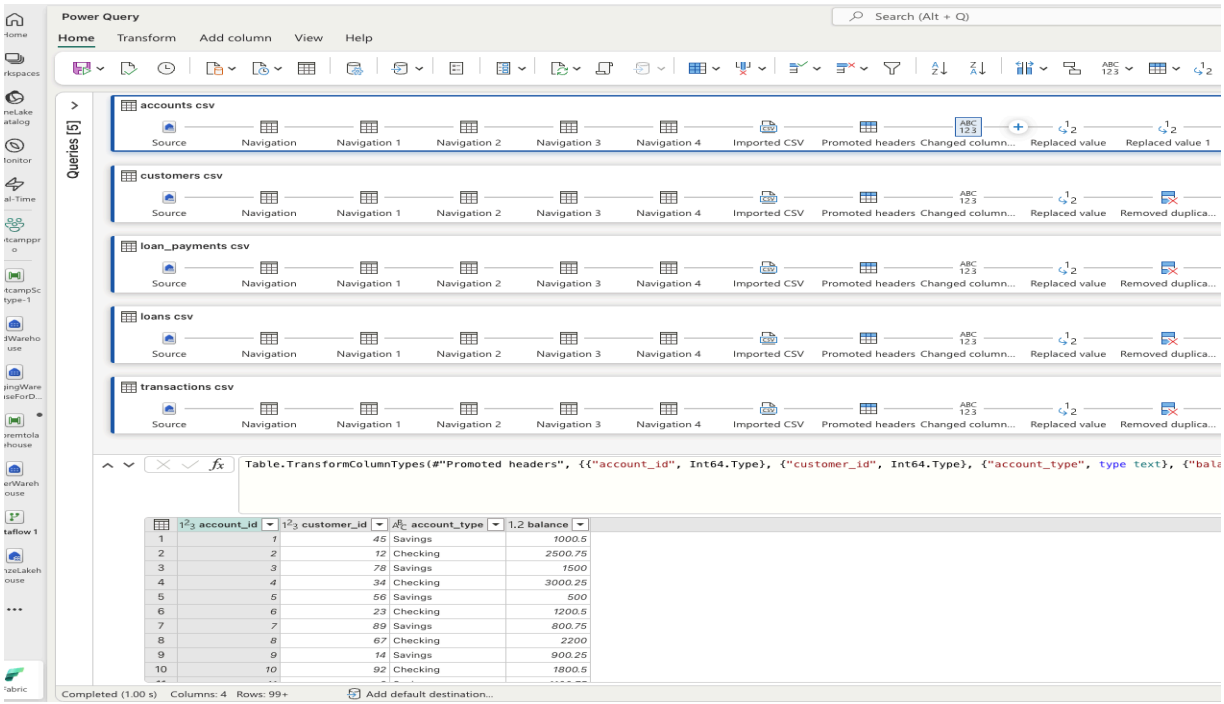
Importing the csv file.



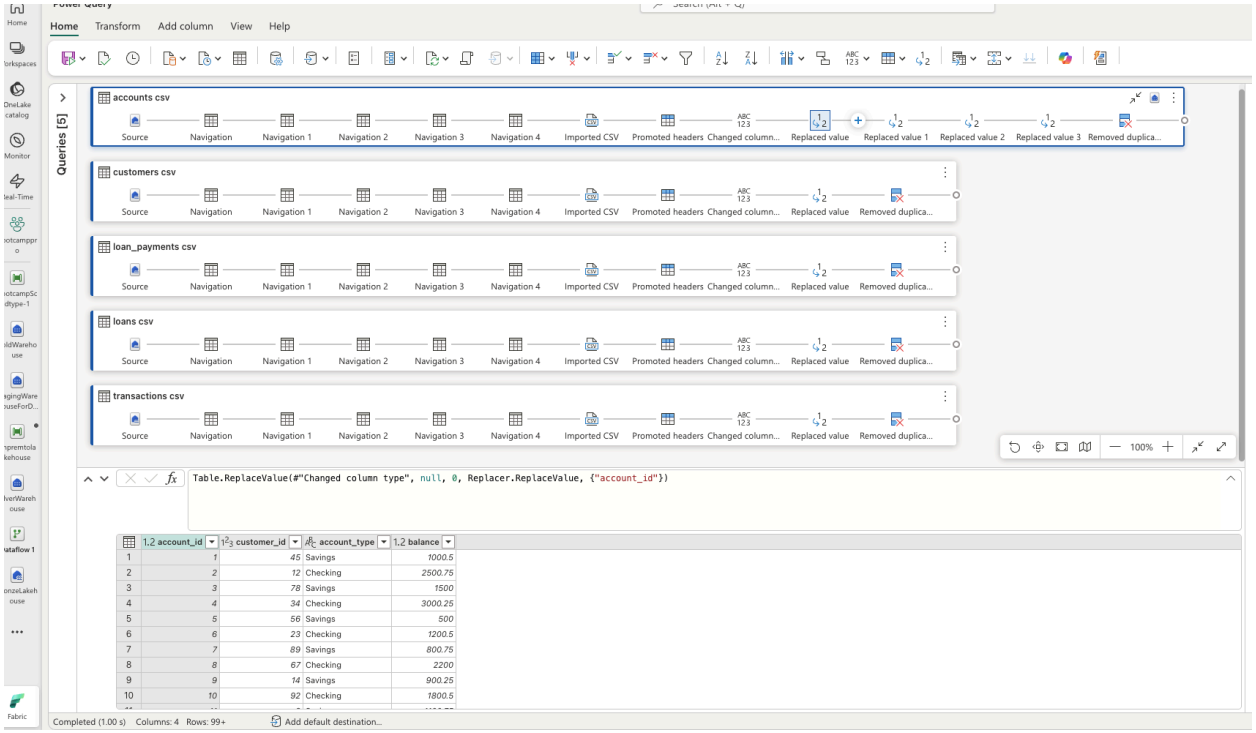
Adding the first row as columns.

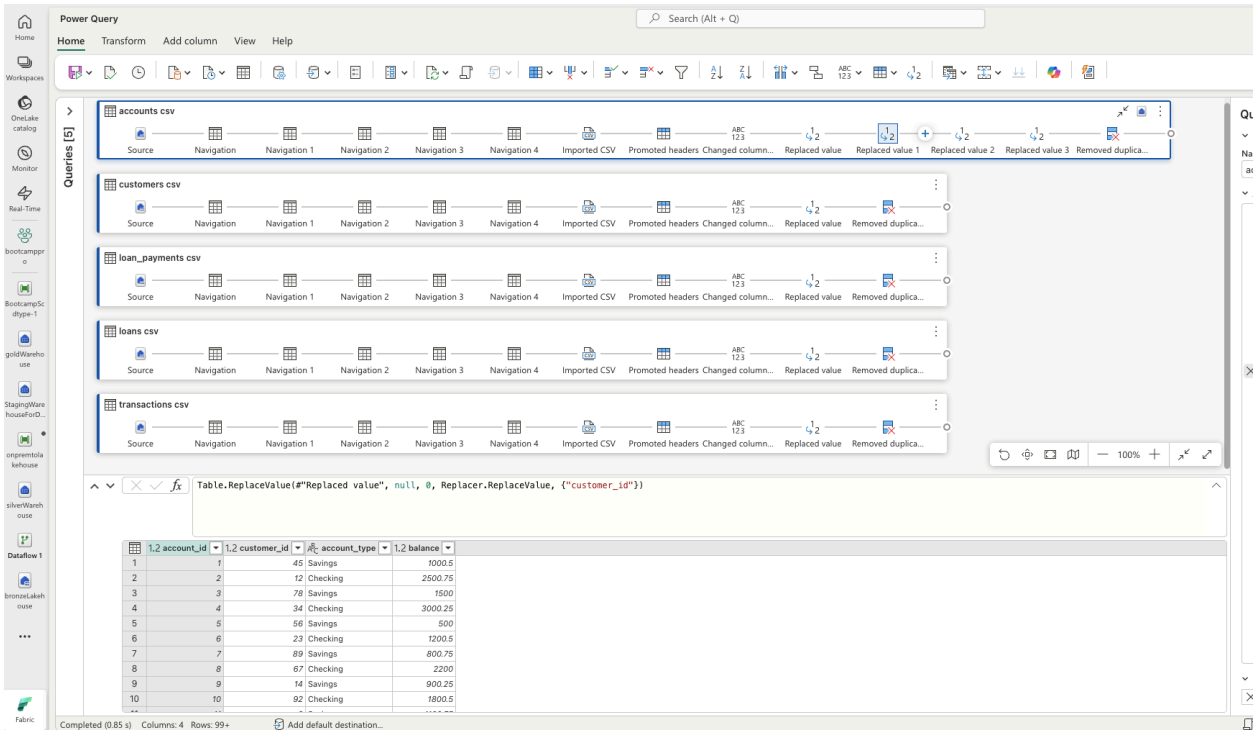


Changing the datatype of columns

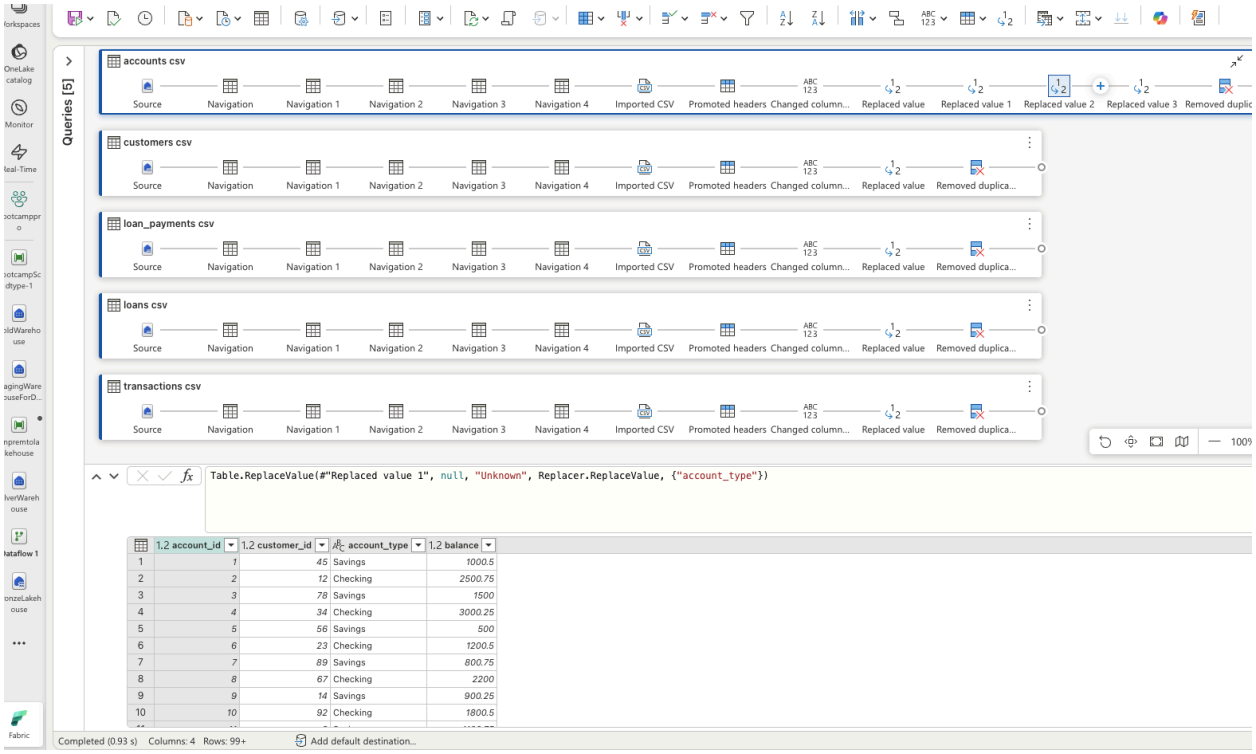


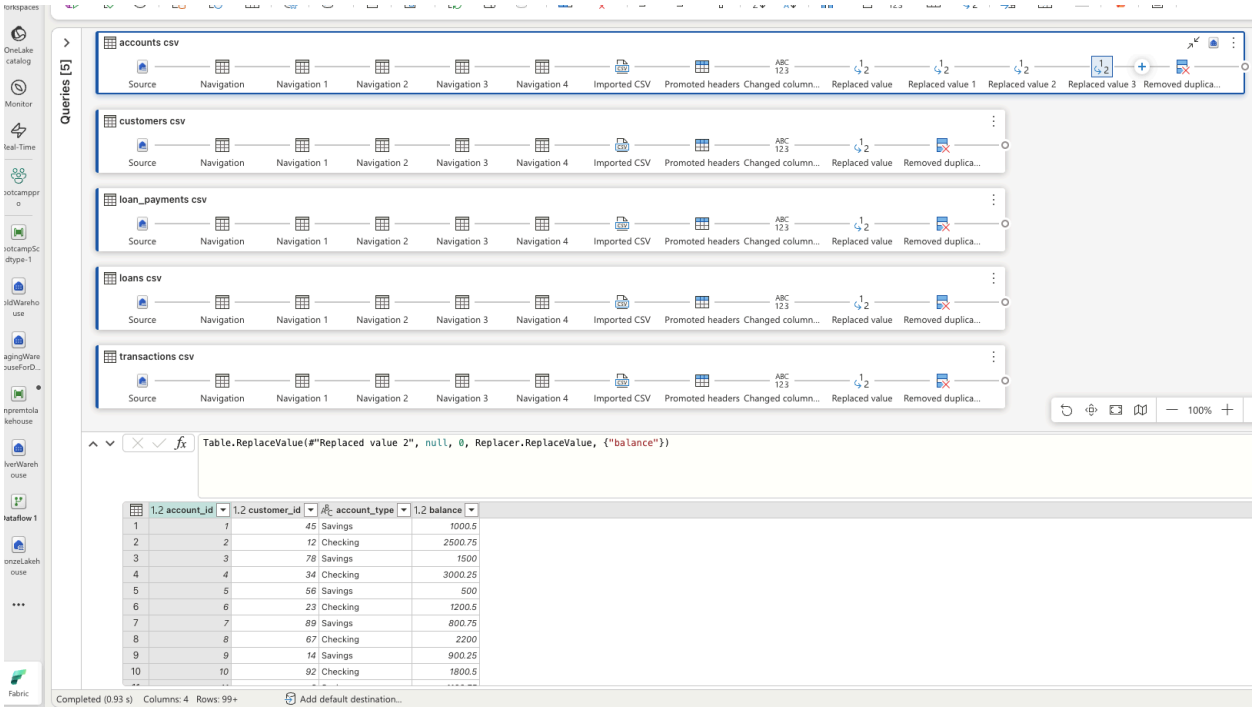
Changing the values where there is null value to 0.



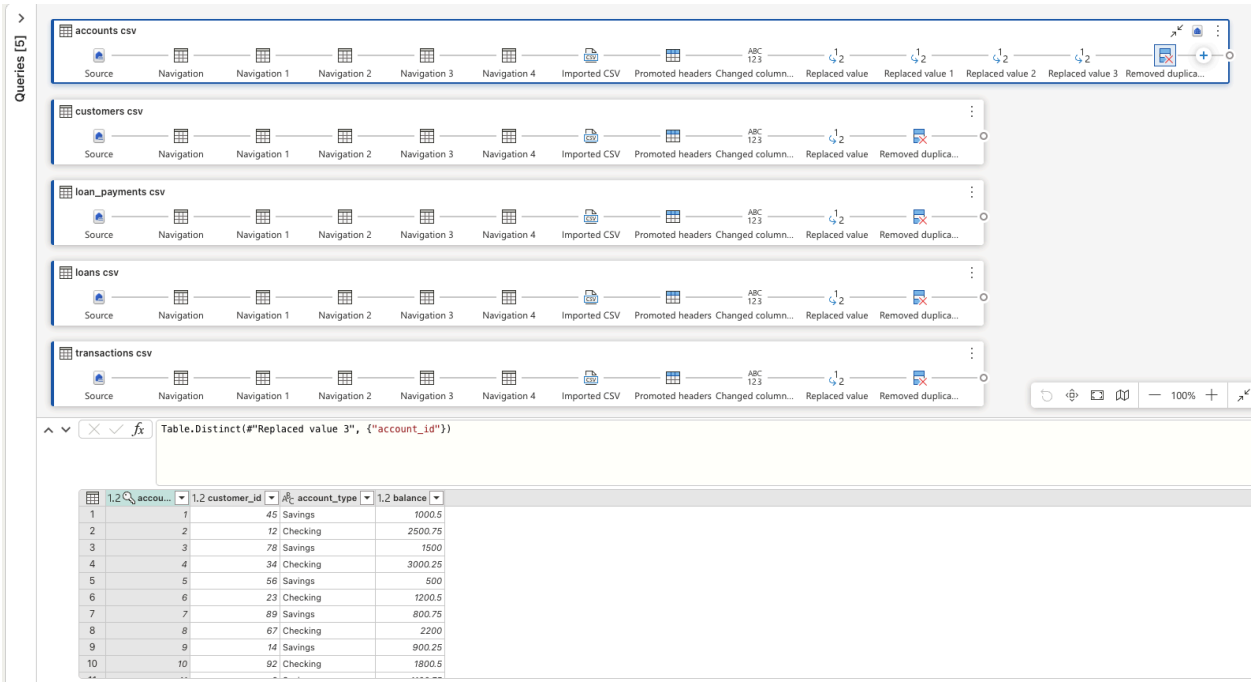


Changing the values where there is null value to 0.

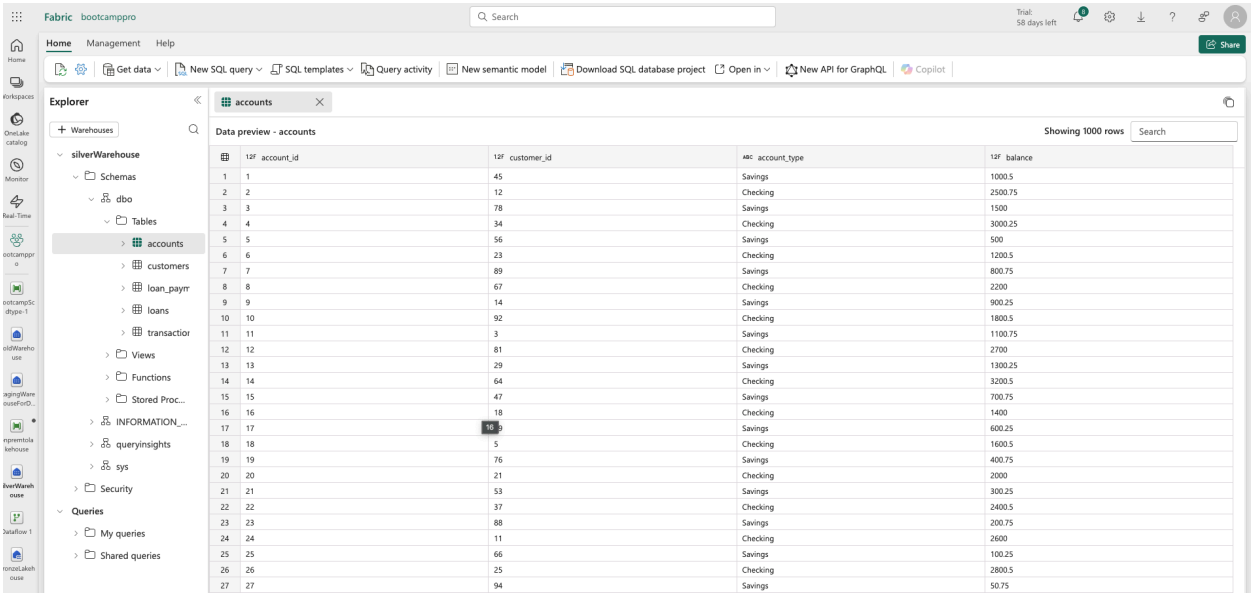




Removing duplicates in the dataflow and adding the sink as data warehouse in each dataflow. Run the dataflow.



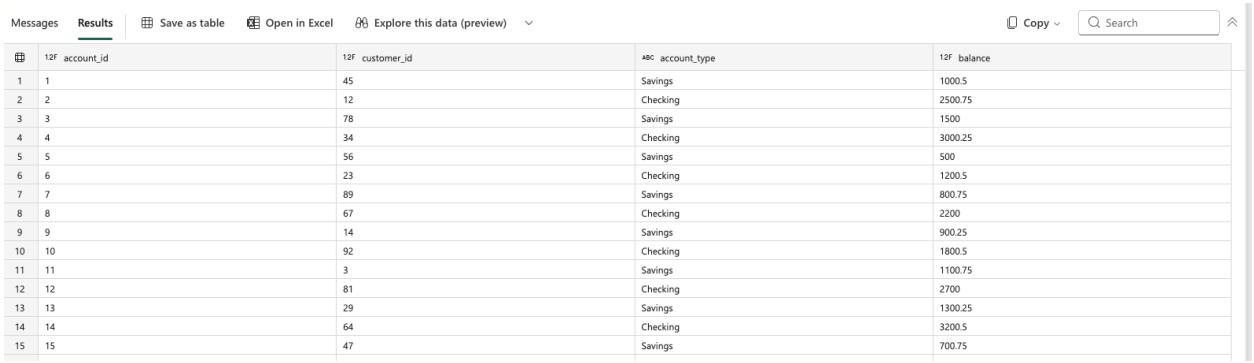
Here we can see all the tables have been created for the respective csv file where the cleaned data is stored.



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.



Messages | Results | Save as table | Open in Excel | Explore this data (preview) | Copy | Search

	account_id	customer_id	account_type	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

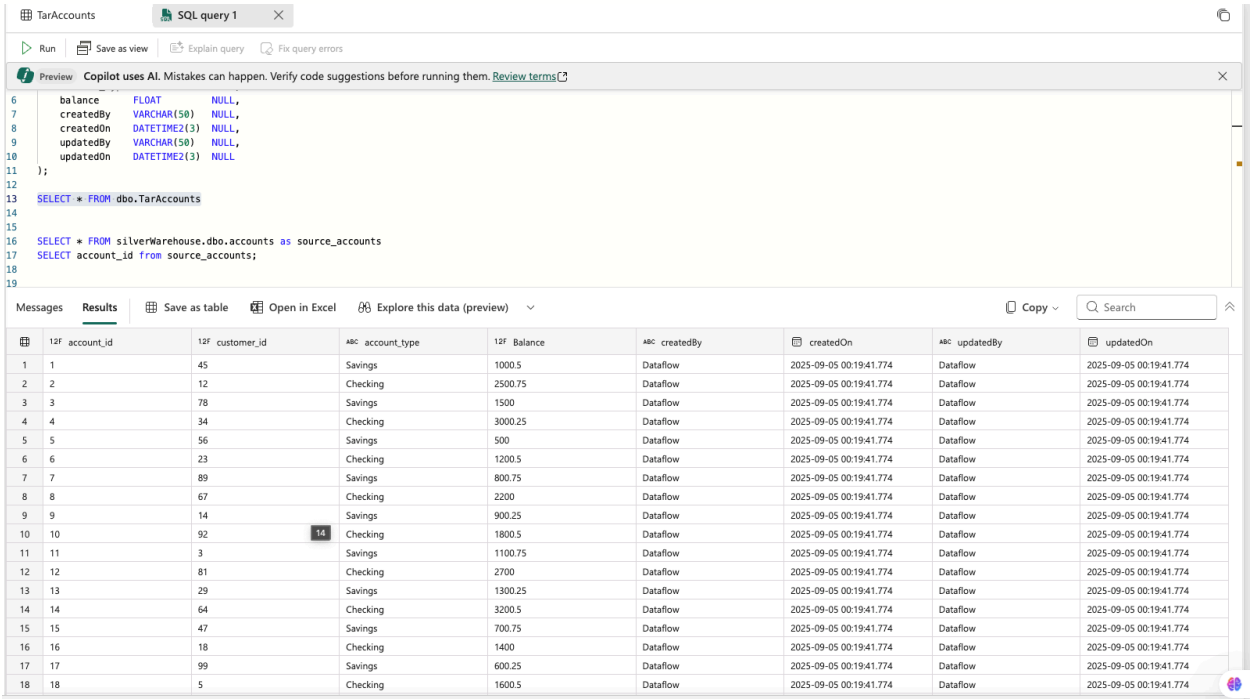
Creating a procedure for implementing SCD type-1

```
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
```

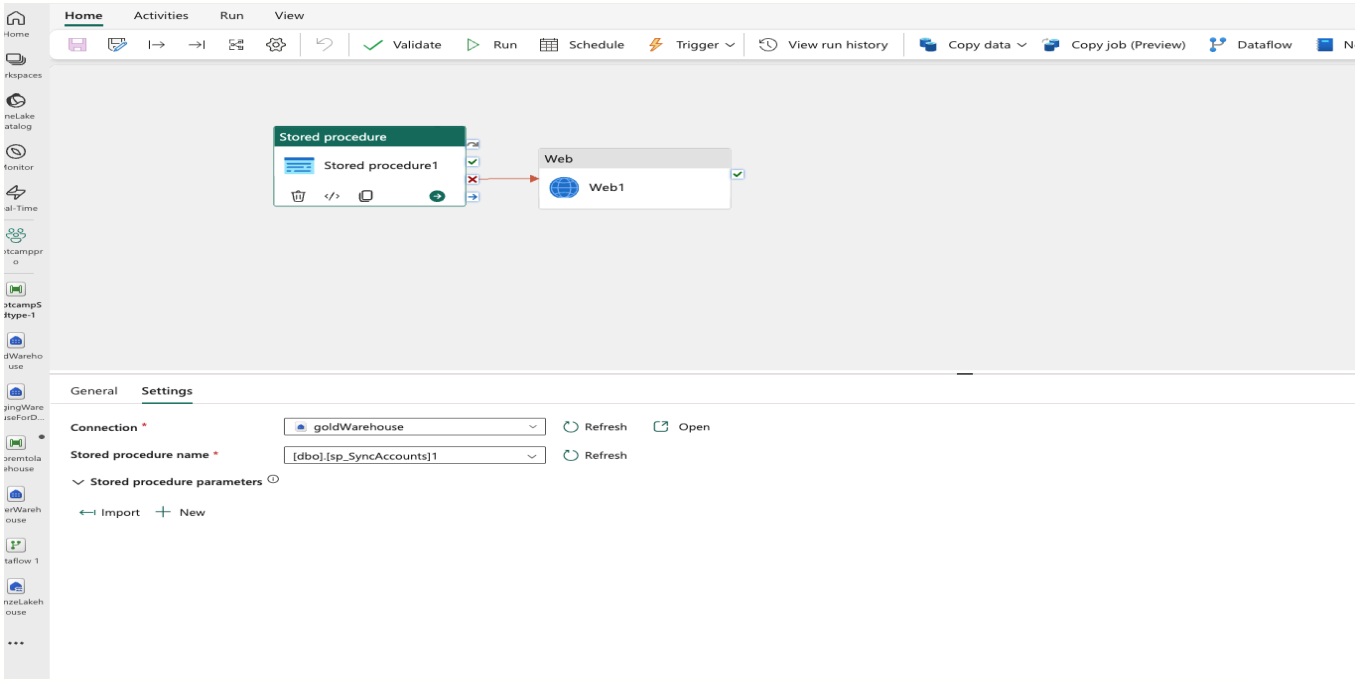
```
        WHERE TARGETS.account_id IS NULL;
    END;
    GO


    EXEC dbo.sp_SyncAccounts;
```

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

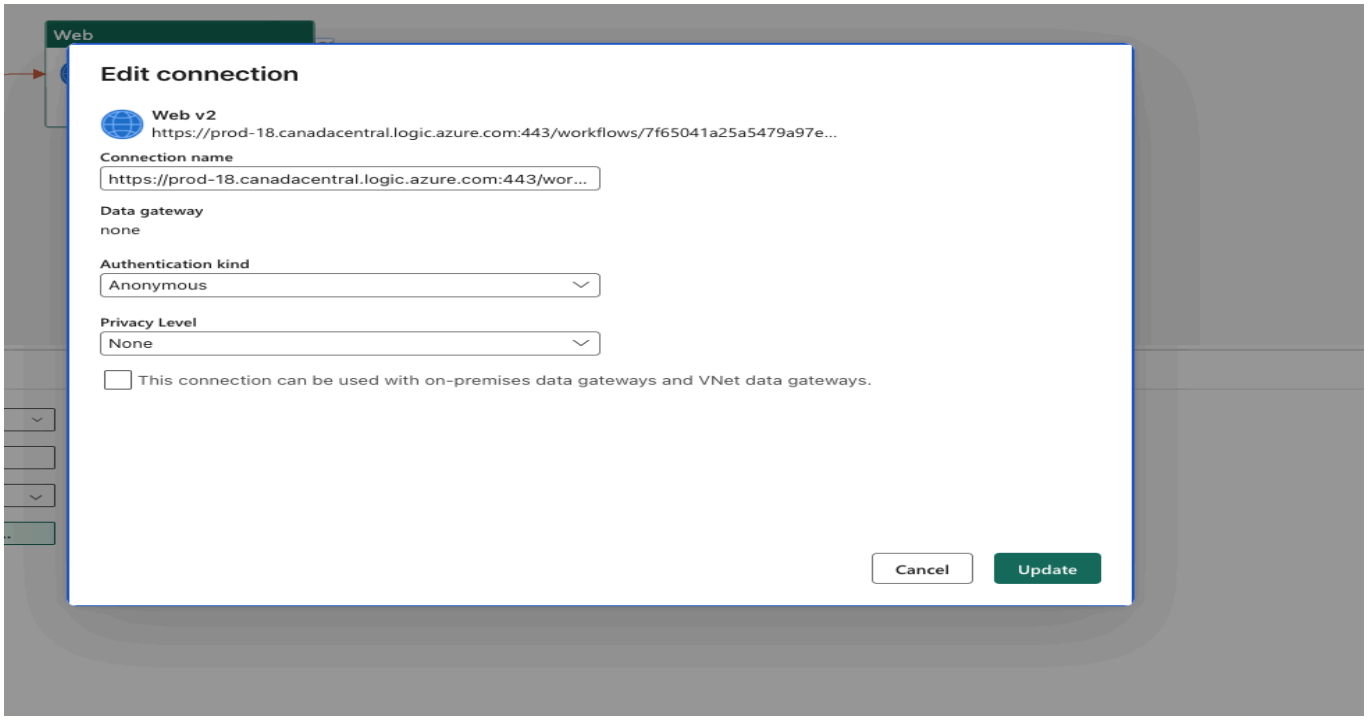


	account_id	customer_id	account_type	Balance	createdBy	createdOn	updatedBy	updatedOn
1	1	45	Savings	1000.5	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
2	2	12	Checking	2500.75	Dataflow	2025-09-05 00:19:41.774	Dataflow	2025-09-05 00:19:41.774
3	3	78	Savings	1500	Dataflow	2025-09-05 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

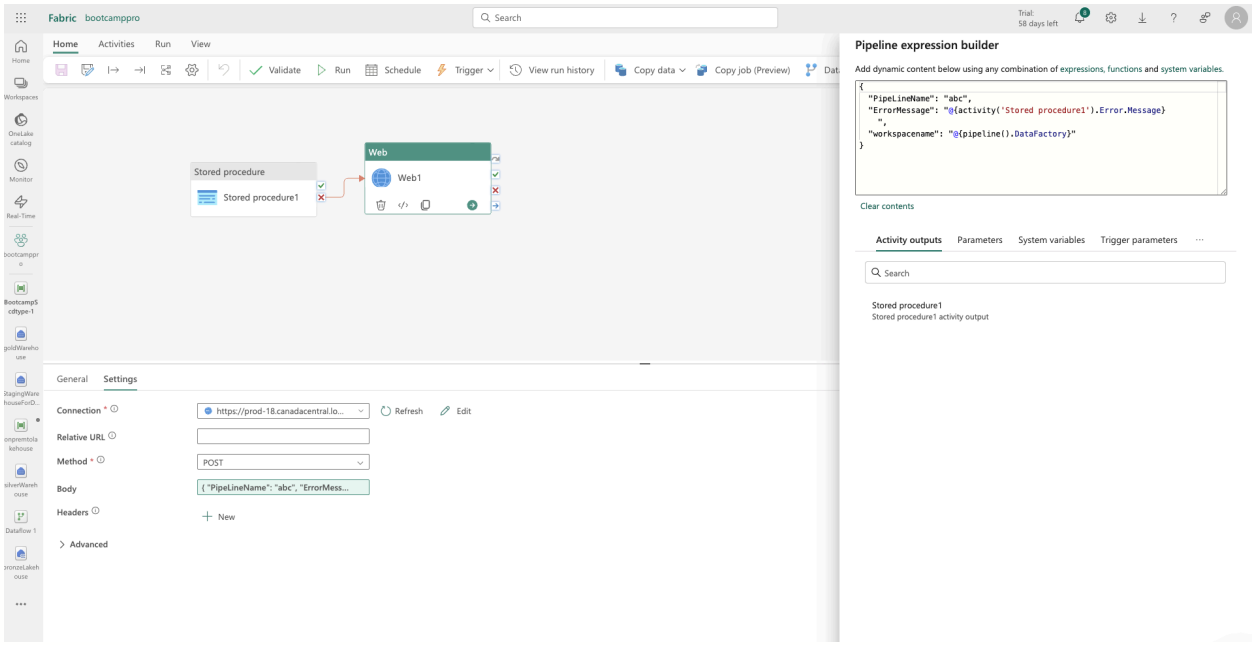
Now we are creating a new pipeline that will trigger a mail when scdtype-1 implementation fails. Add a stored procedure activity in the pipeline and do the following configuration.



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

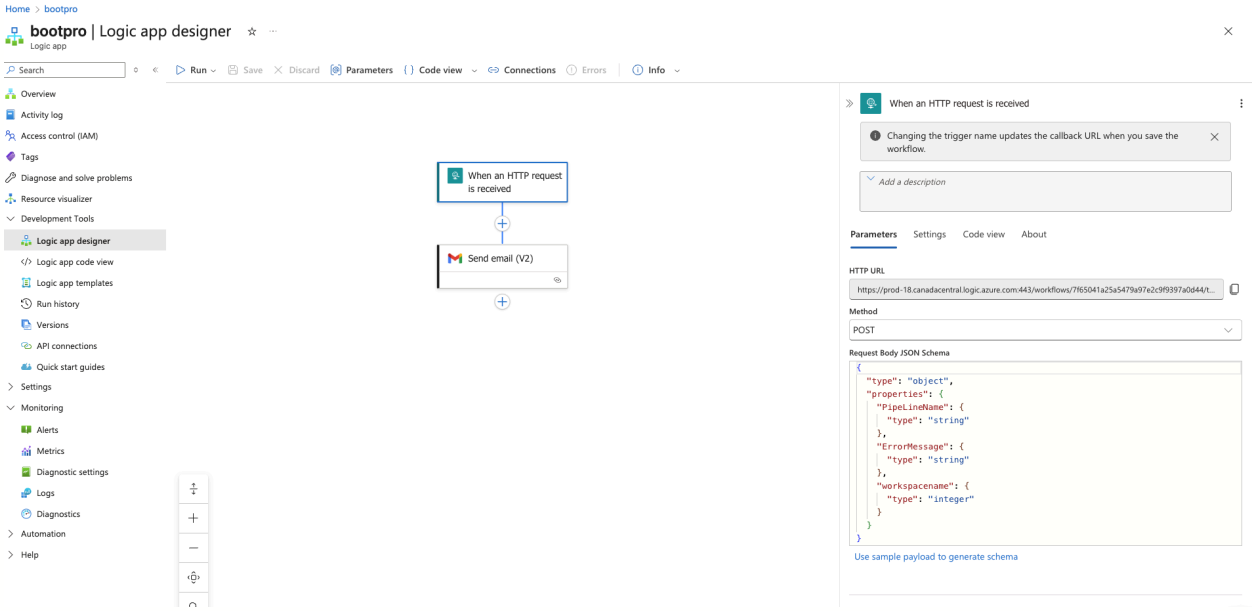


Provide the following expression in the expression builder.

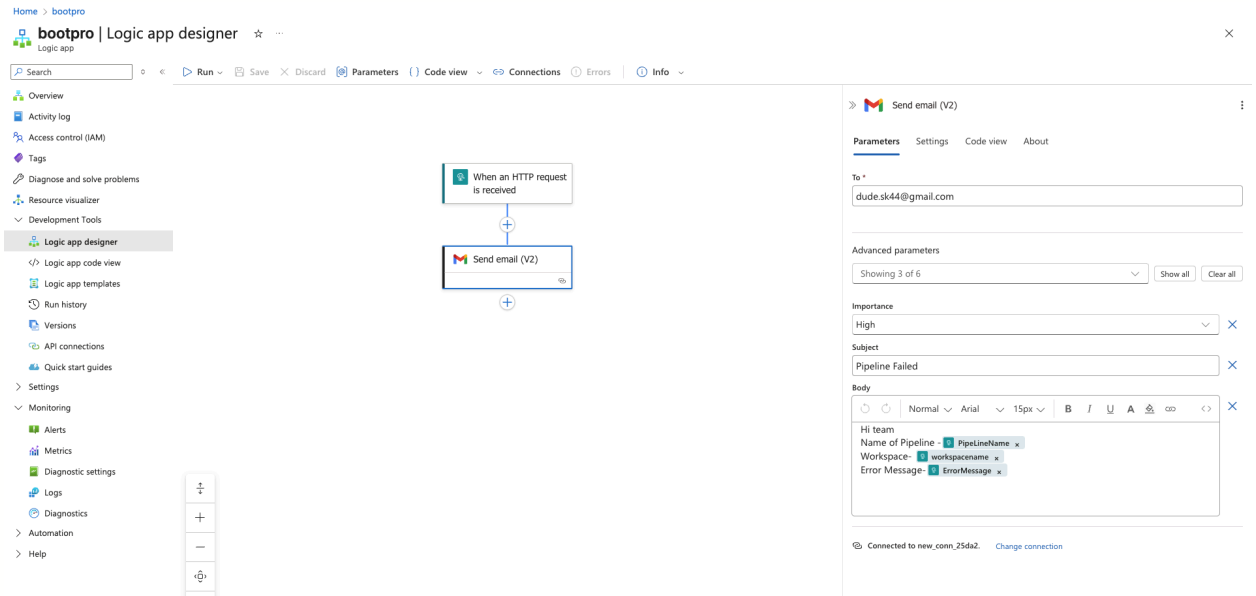



```
{
 "type": "object",
 "properties": {
    "PipeLineName": {
        "type": "string"
    },
    "ErrorMessage": {
        "type": "string"
    },
    "workspacename": {
        "type": "integer"
    }
 }
}
```

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.



Provide your email ID and provide the following configuration.



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