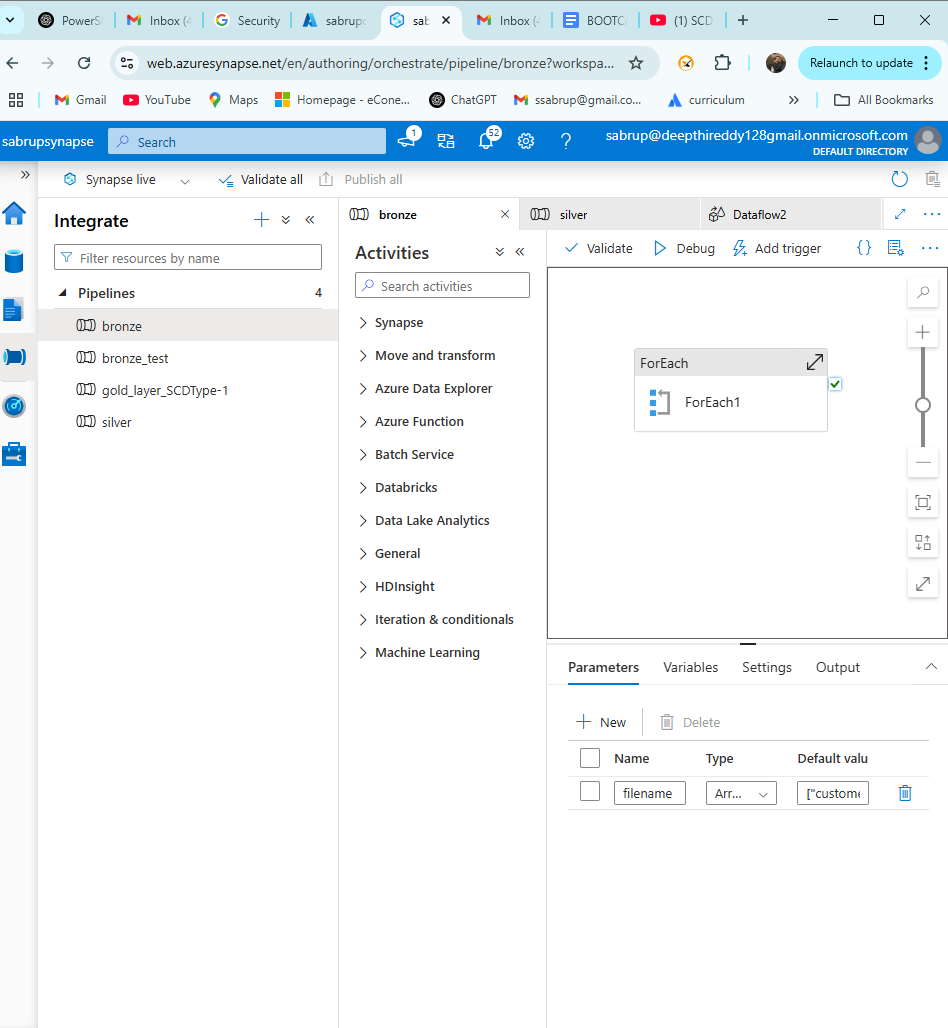
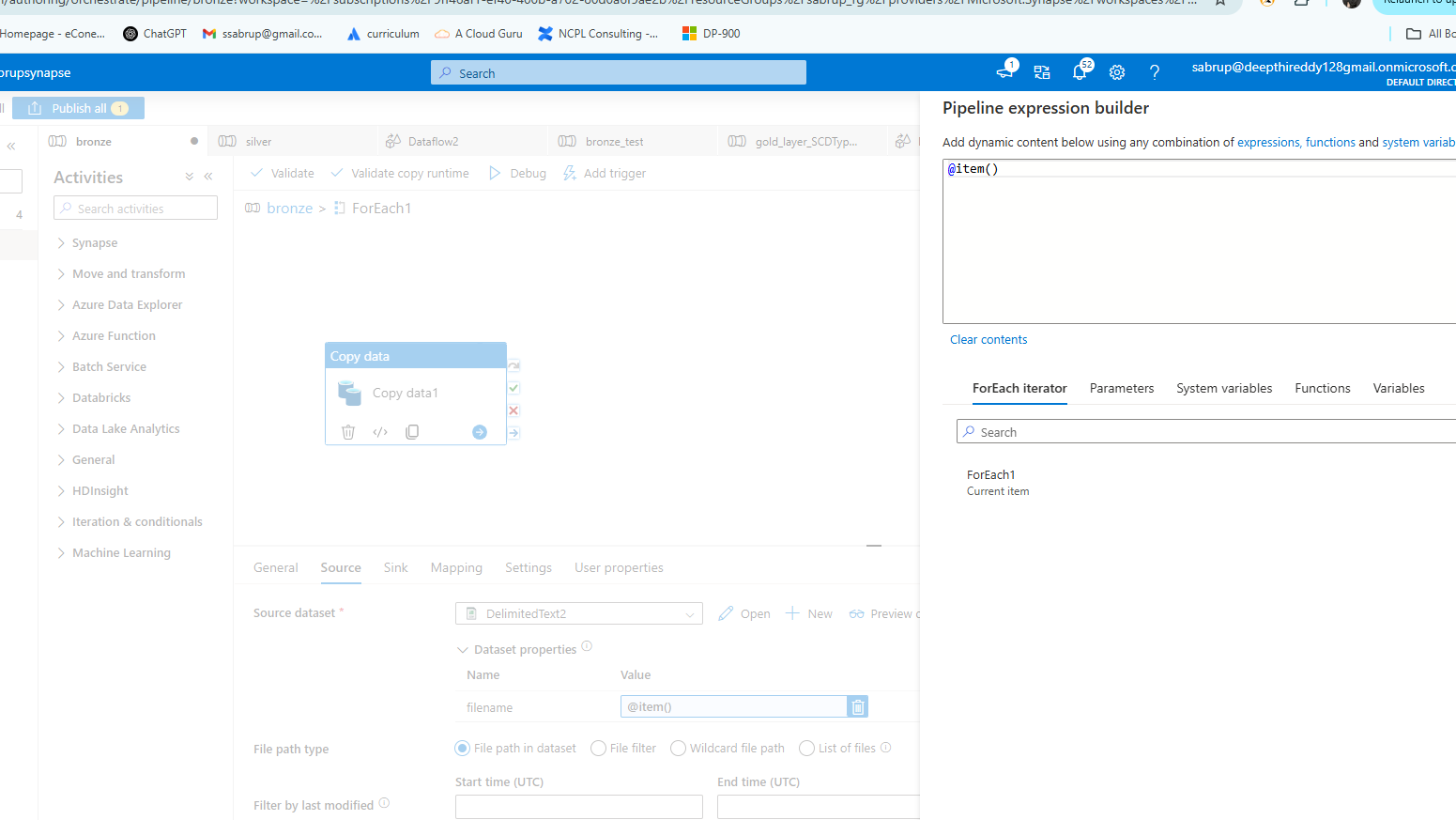
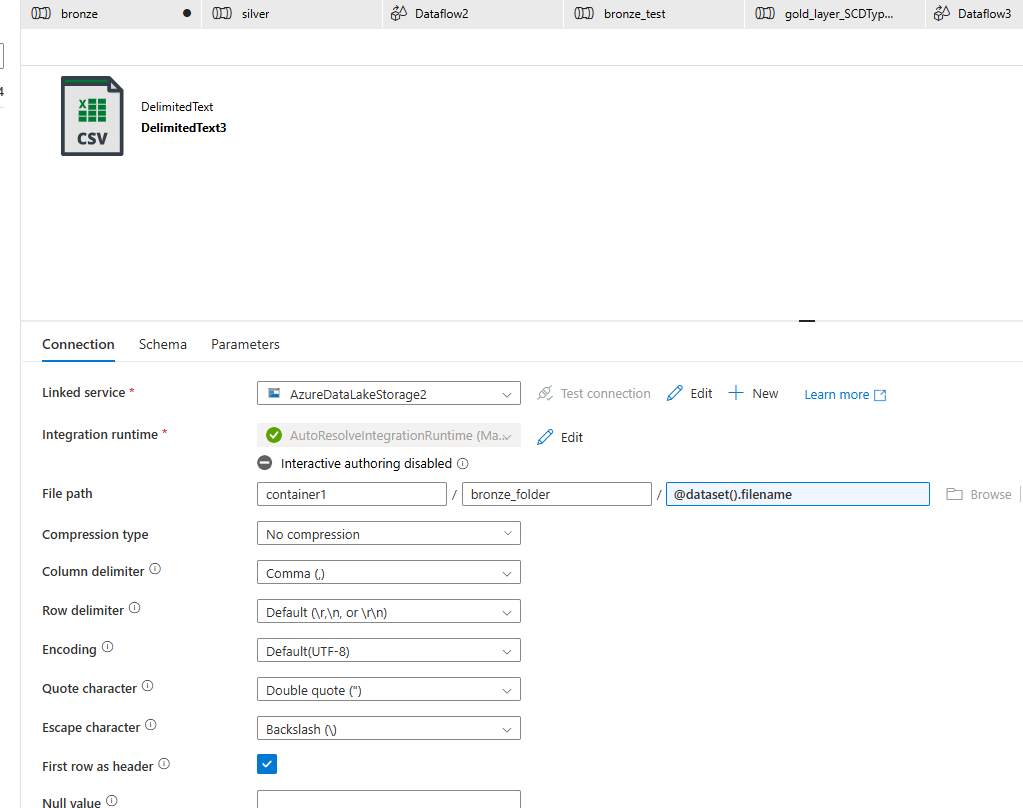
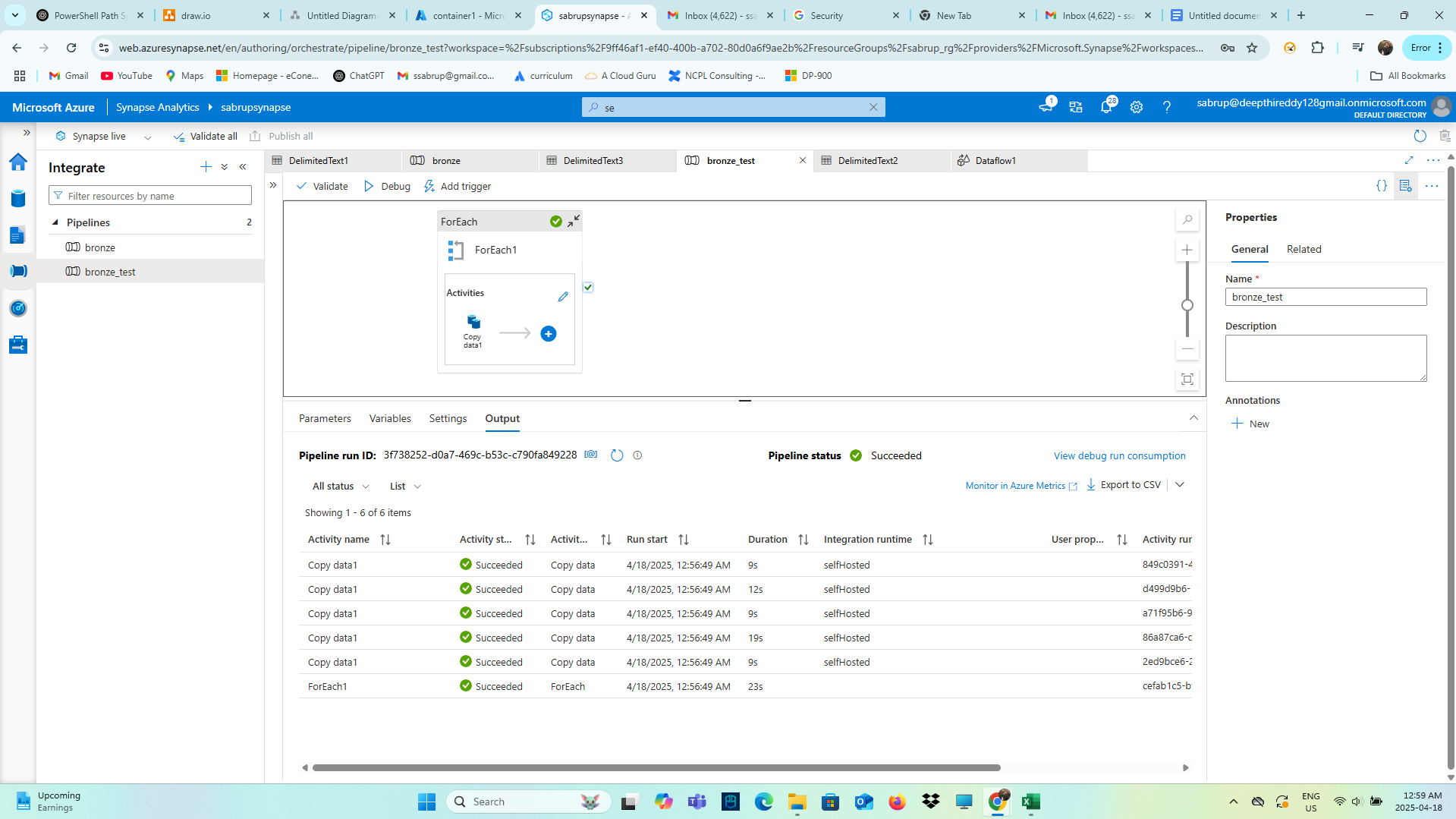
**Bootcamp Project 1**

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

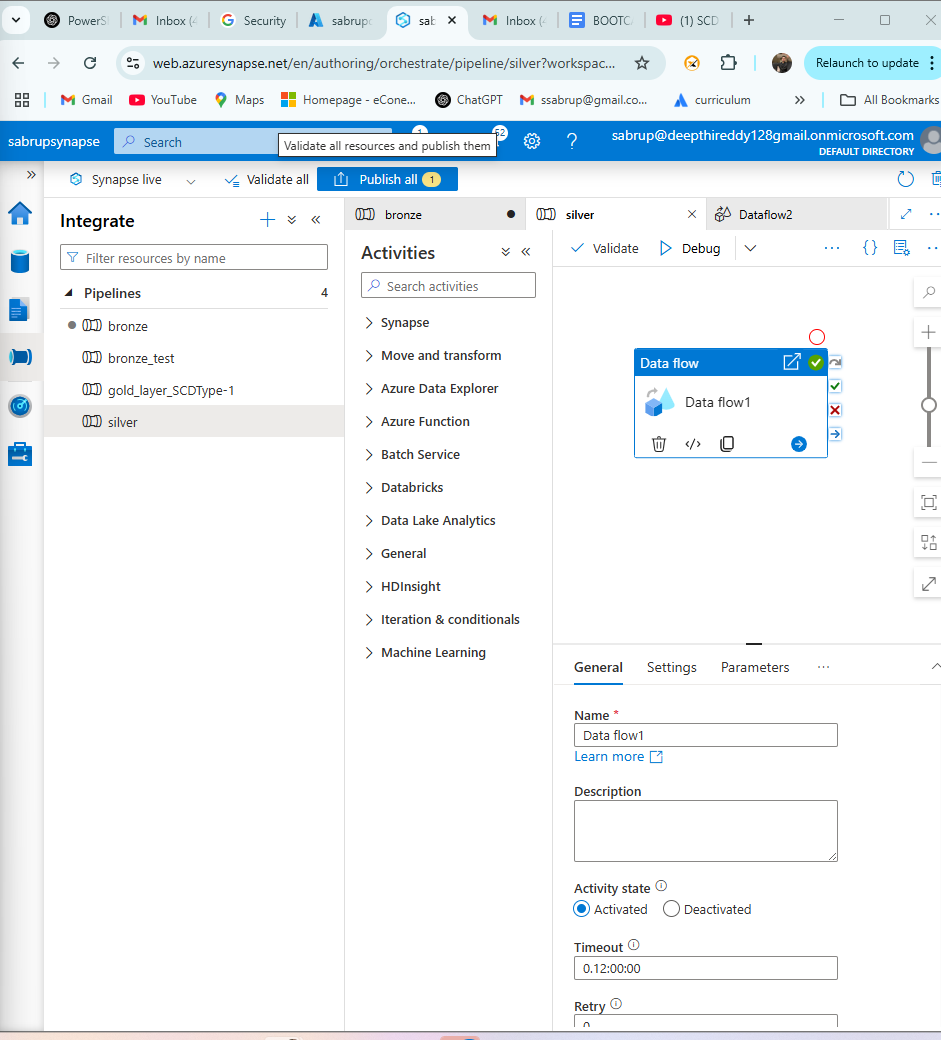
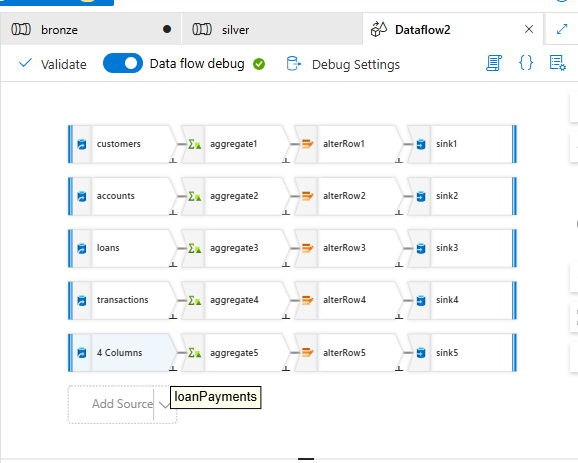
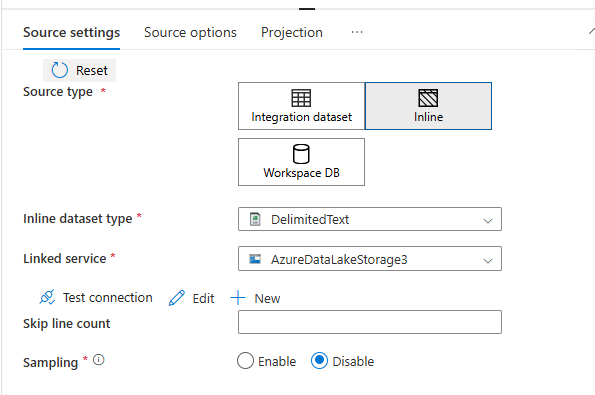
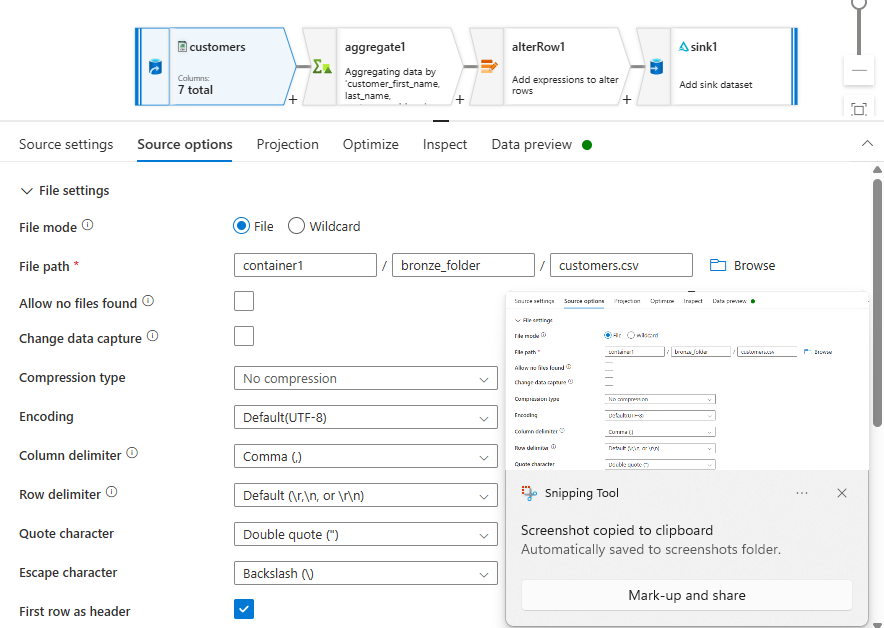
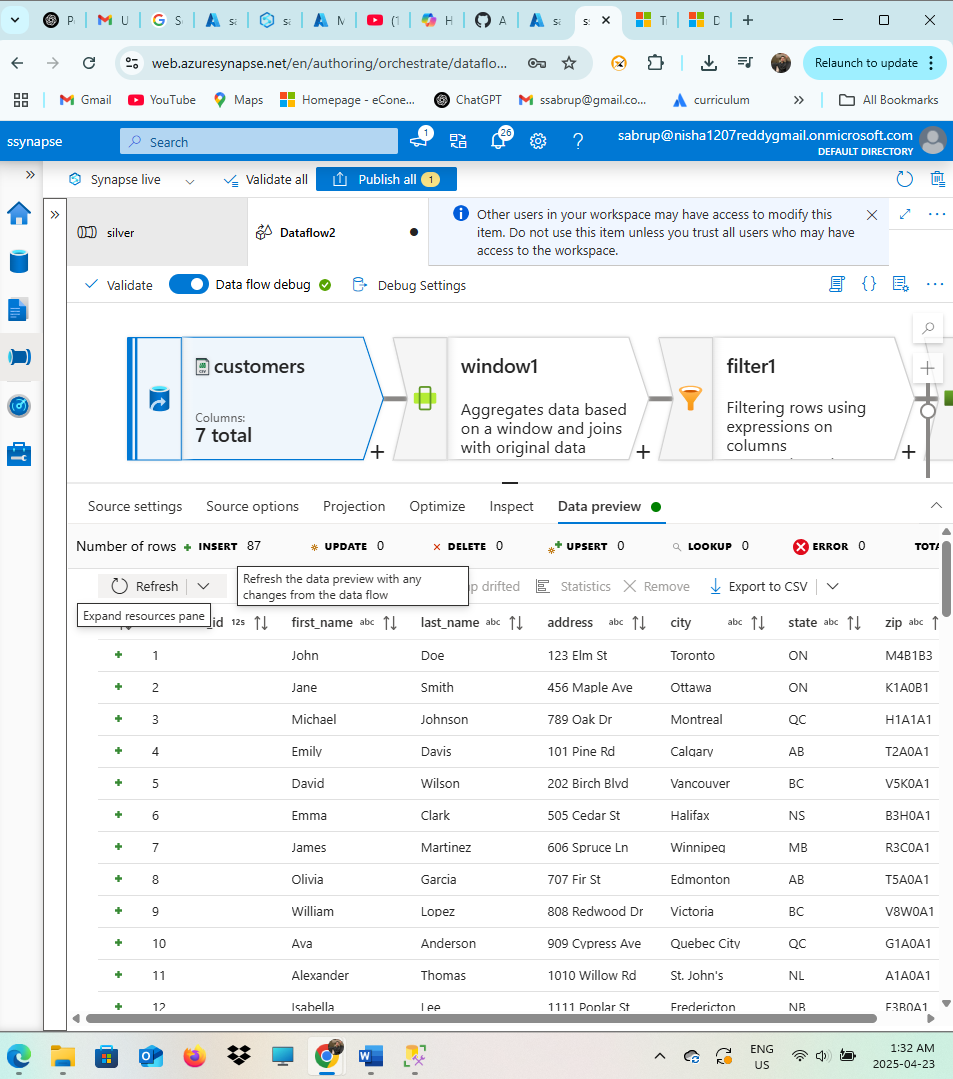
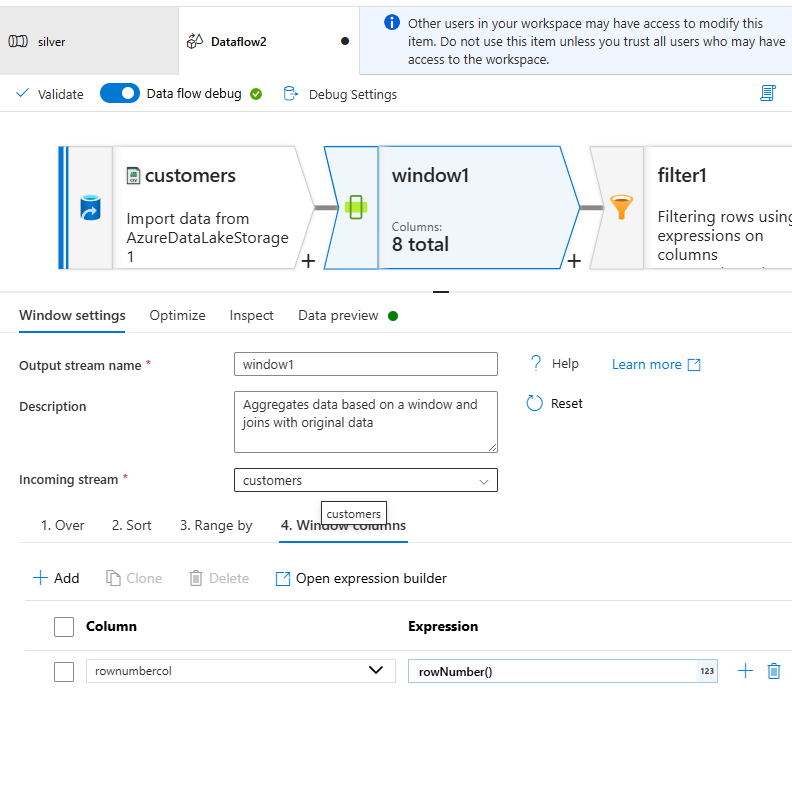
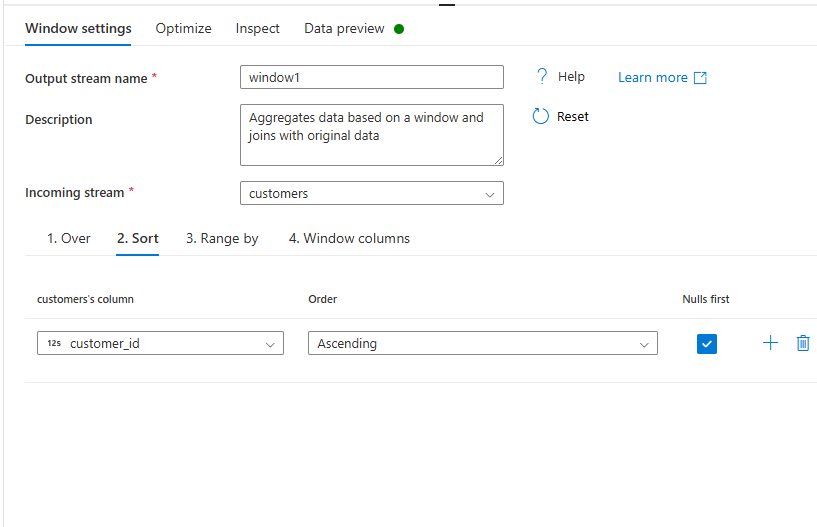
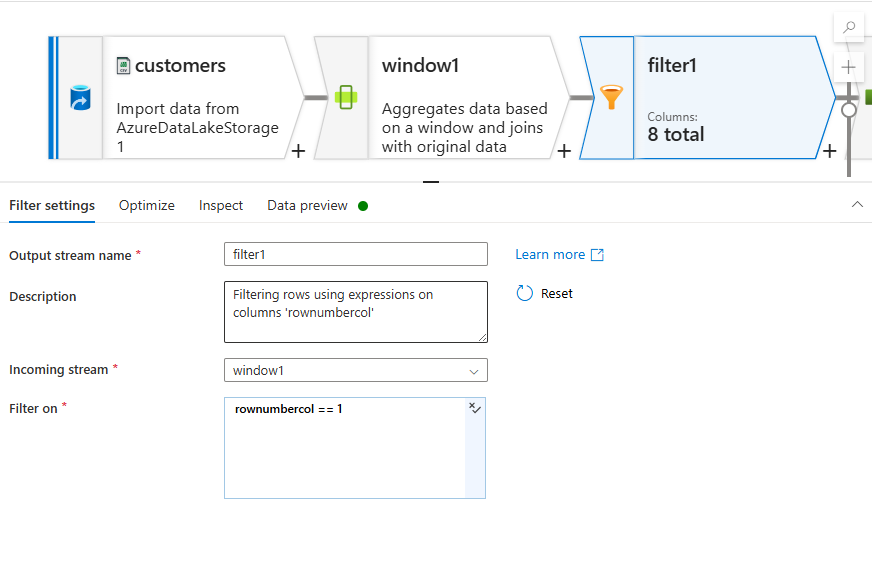
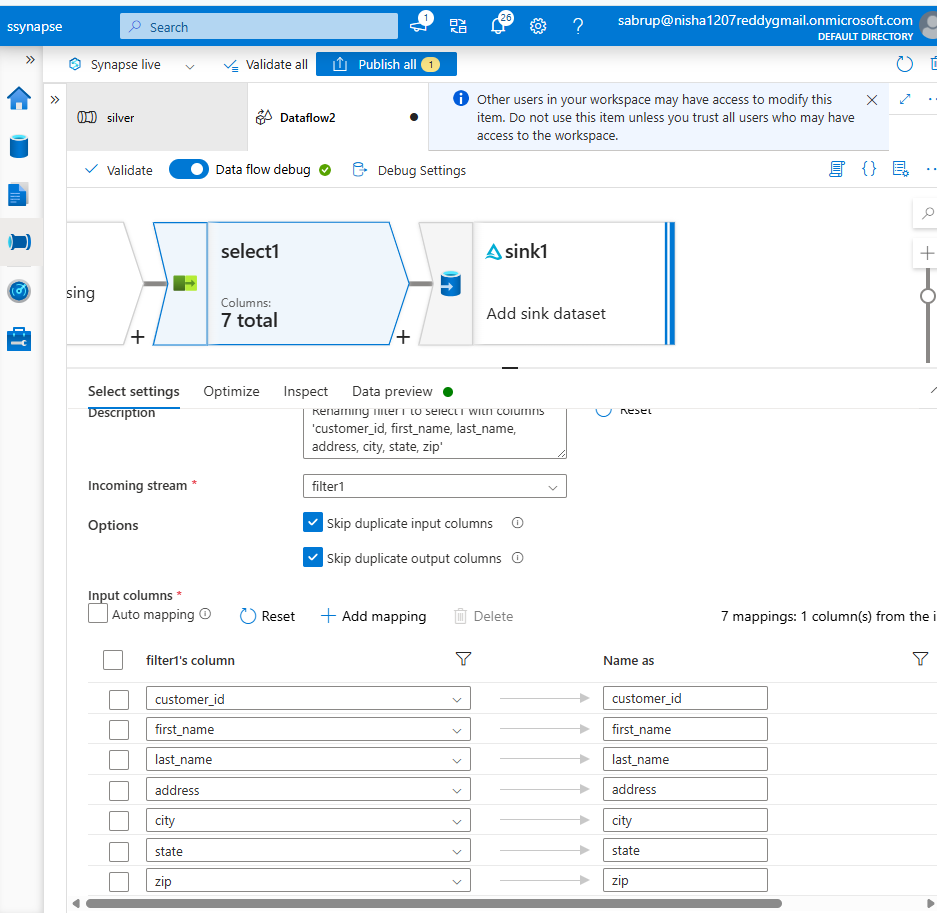
* To get the all 5 files from my laptop I created a pipeline called “bronze” with parmeter type Array that holds all the names of the files.  
  
* In the bronze file, there is ForEach Activity that iterates through each of the value of thepipeline parameter.
* In ForEach there is copydata Activity where dataset is DelimitedText2 and has a parameter “filename” who’s value is currentItem of ForEach.  
  
* For sink the target is ADLS Gen2 to store all the raw files.  
  To store I have created a directory called “bronze\_folder”

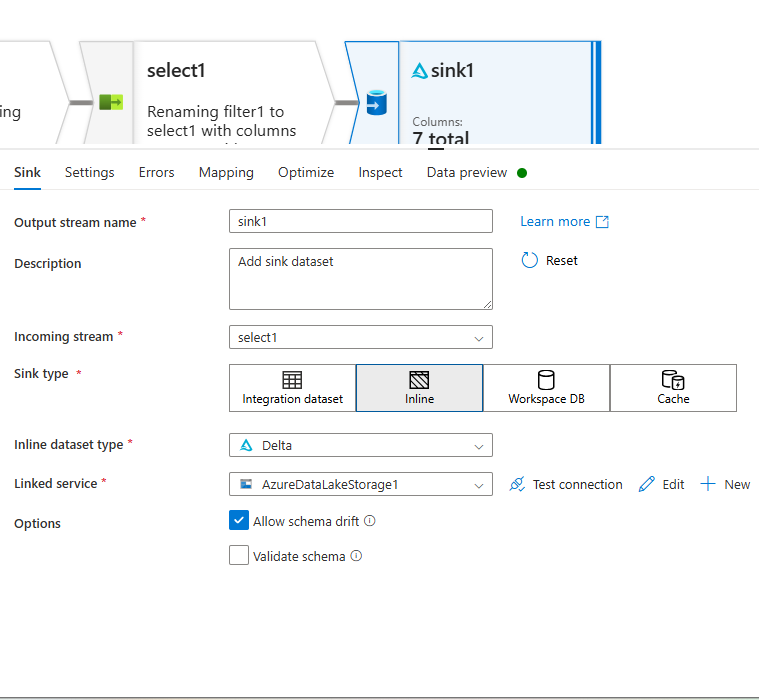
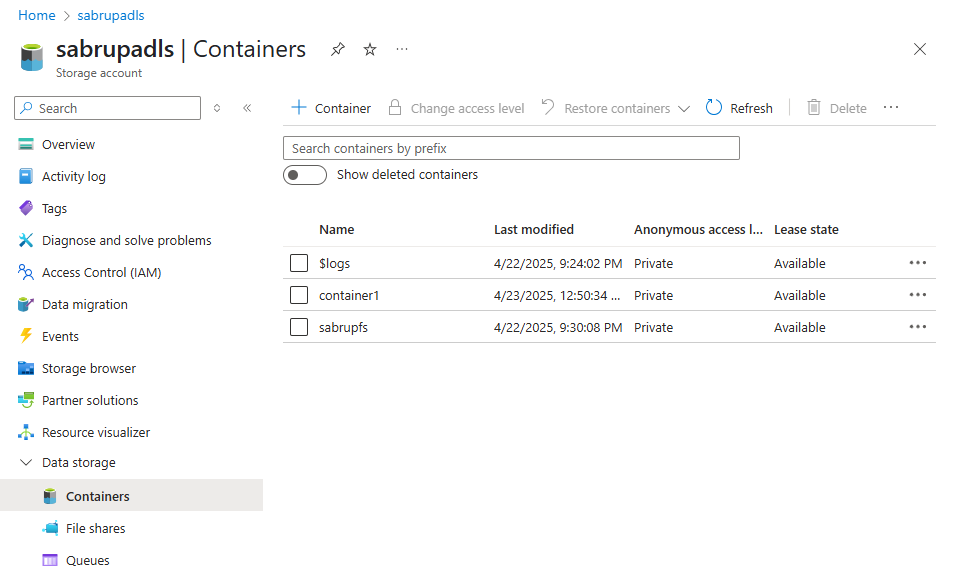
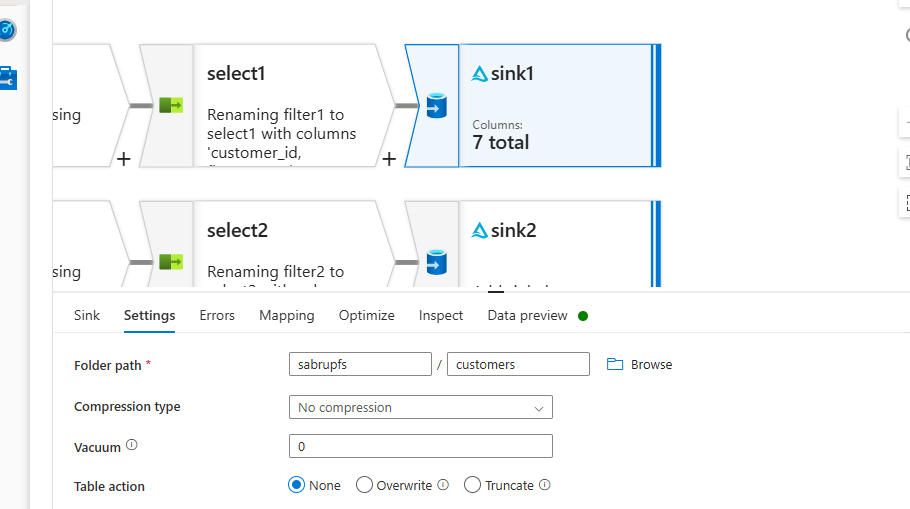
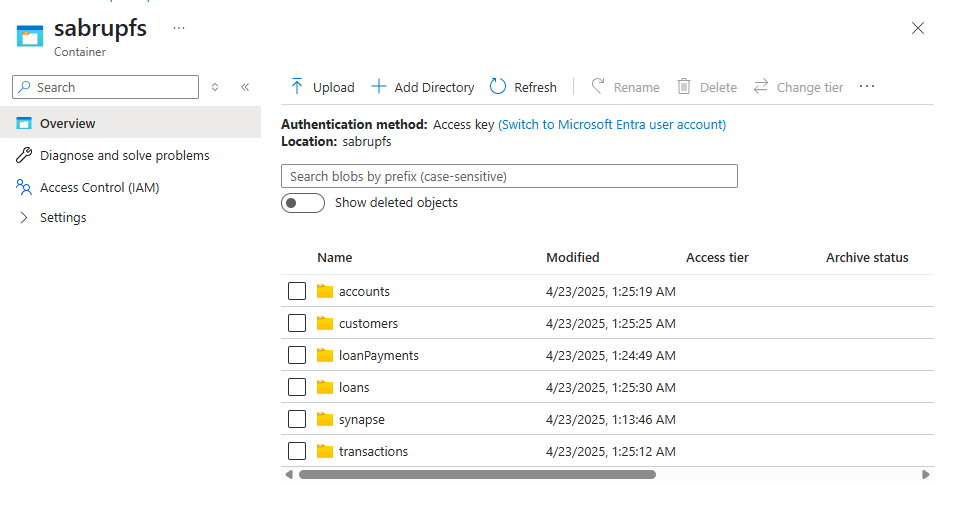
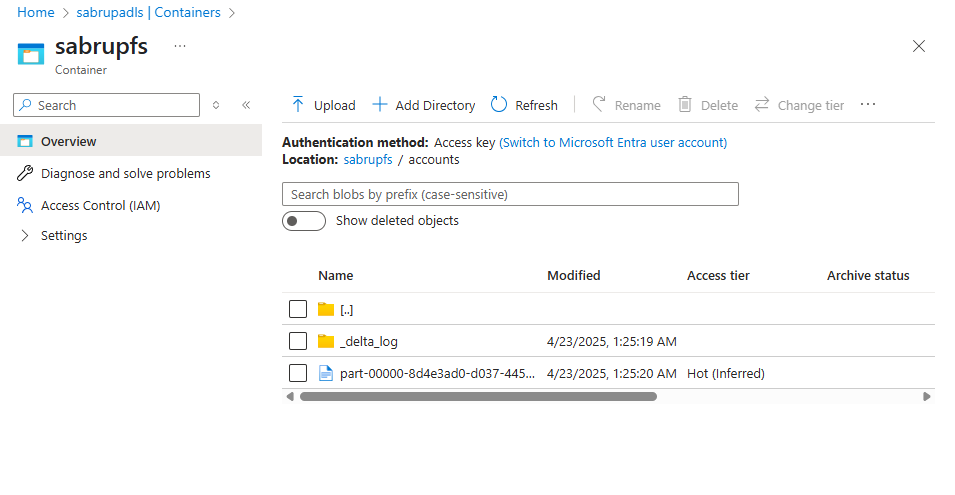
  
Again sink has parameter to fetch the value of currentItem from ForEach and the name of the file would be whatever it is.

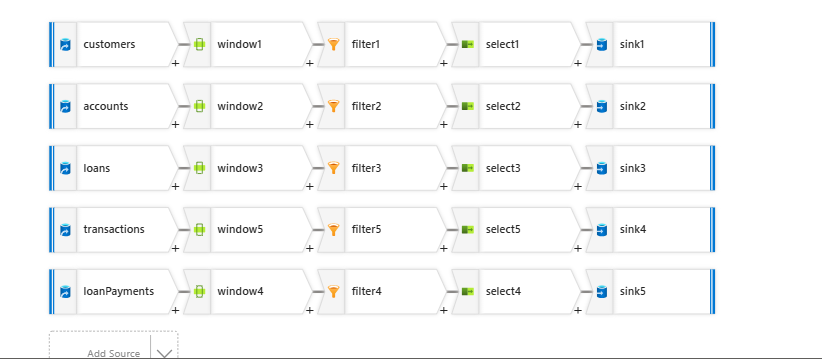
The pipeline running successfully:



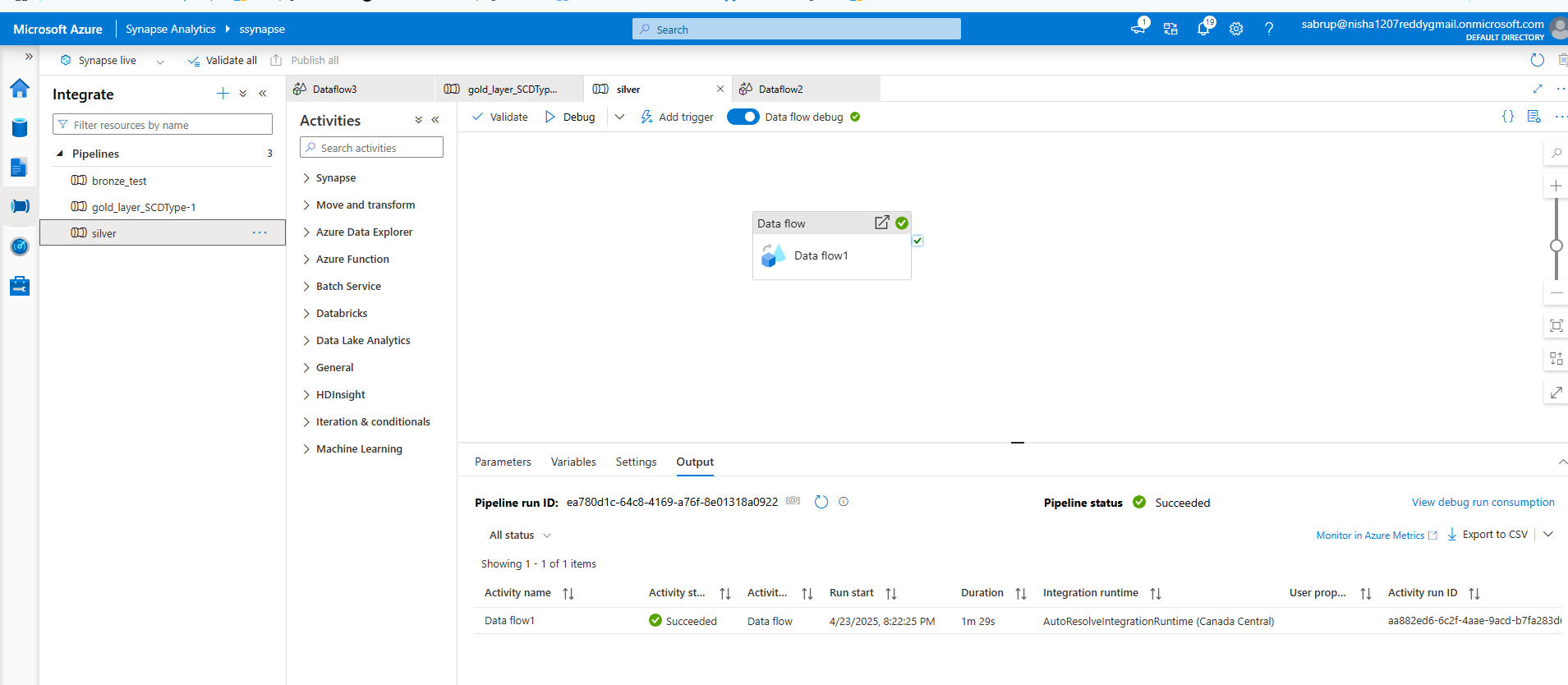
**Step 2: Use ADF Dataflows to remove the duplicates**

* To clean the data there is pipeline called “silver” that has a Dataflow that performs aggregate tranformation to clean the data.  
  
* Since there are 5 files that I am taking from the ADLS Gen2 so there are 5 dataflows to perform transformations on each of the files.  
  
* **Explaining for Customers:**1. I have added a source of dataset type DelimitedText and the linked service is AzureDatalakeStorage3 to establish a connection with ADLS  
  
* **Source option:**  
  Here the file path is customers.csv which is in bronze\_folder of container1.
* **Data preview:  
  **  
  The condition aggregates all the columns except the “customer\_id”.
* **Window transformation:**   
    
  Window transformation I used to remove all the duplicates.  
    
  **Filter Transformation:  
  **
* Then there is select transformation **to remove the rownumbercol**

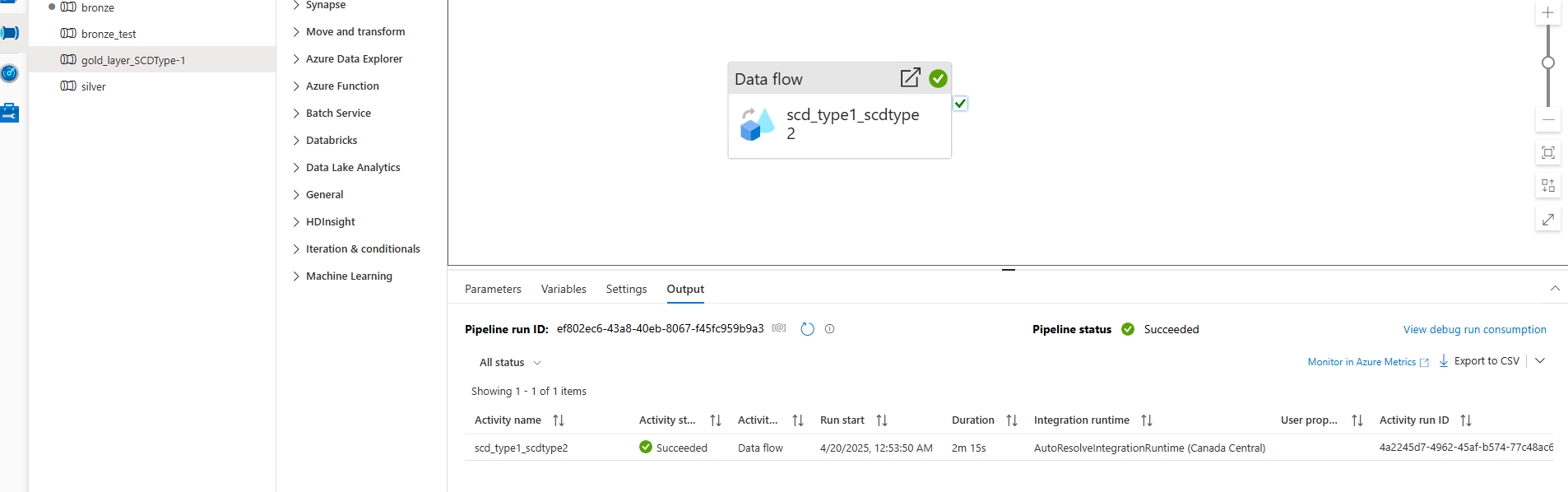
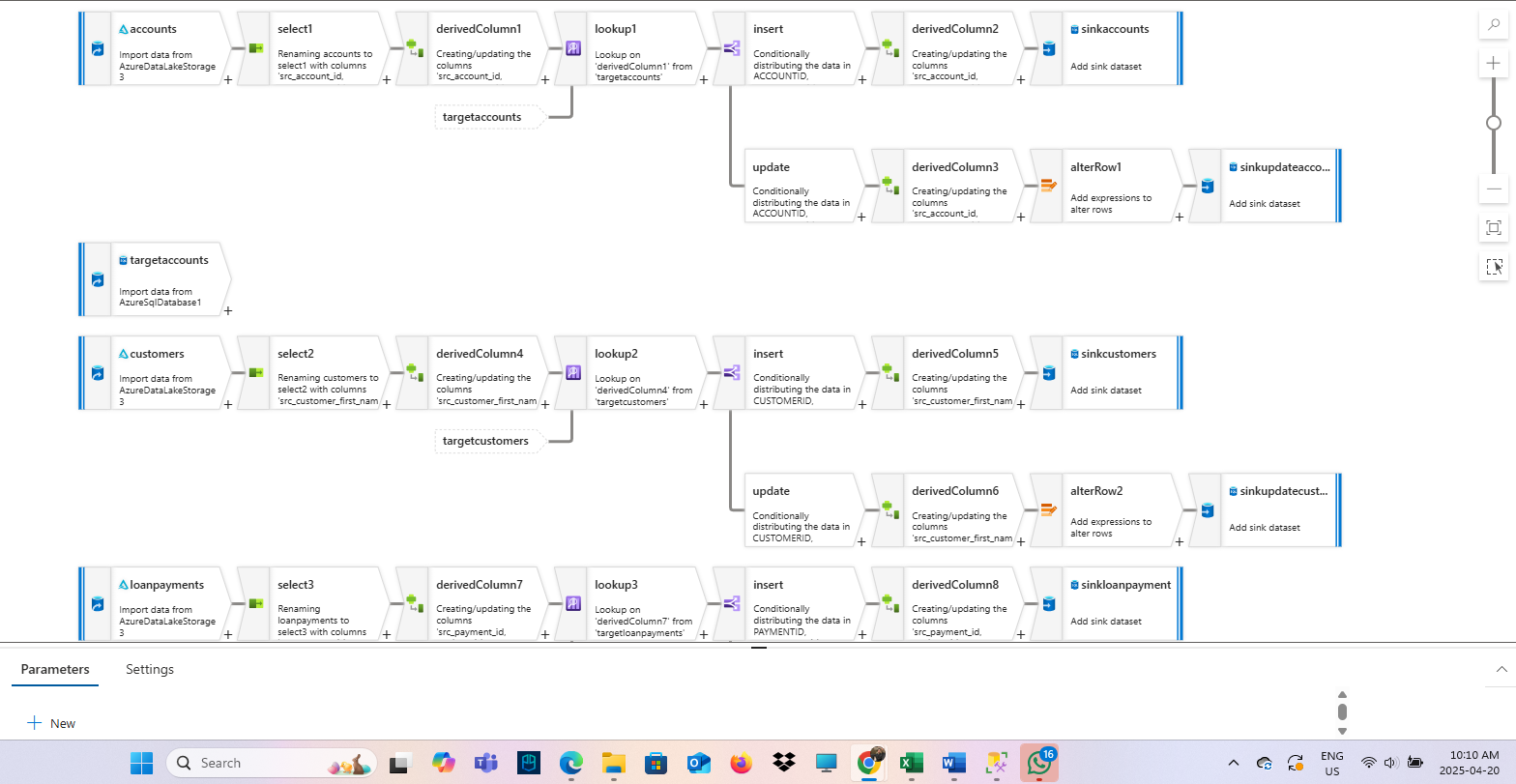
* Finally to store this clean data there is Sink transformation that stores file in “Delta” format in ADLS using “AzureDataLakeStorage3” linked service.  
  
* To store these clean files I have created a separate container “sabrupfs” .  
  
* To store a clean file there is respective folder.  
  For example here, to store customers file I have created customers folder.  
  
* **The below are all the folders for respective file.**
* The below are the files along with the modified files:  
  
* **The same steps I performed for all the other 4 files.**

****

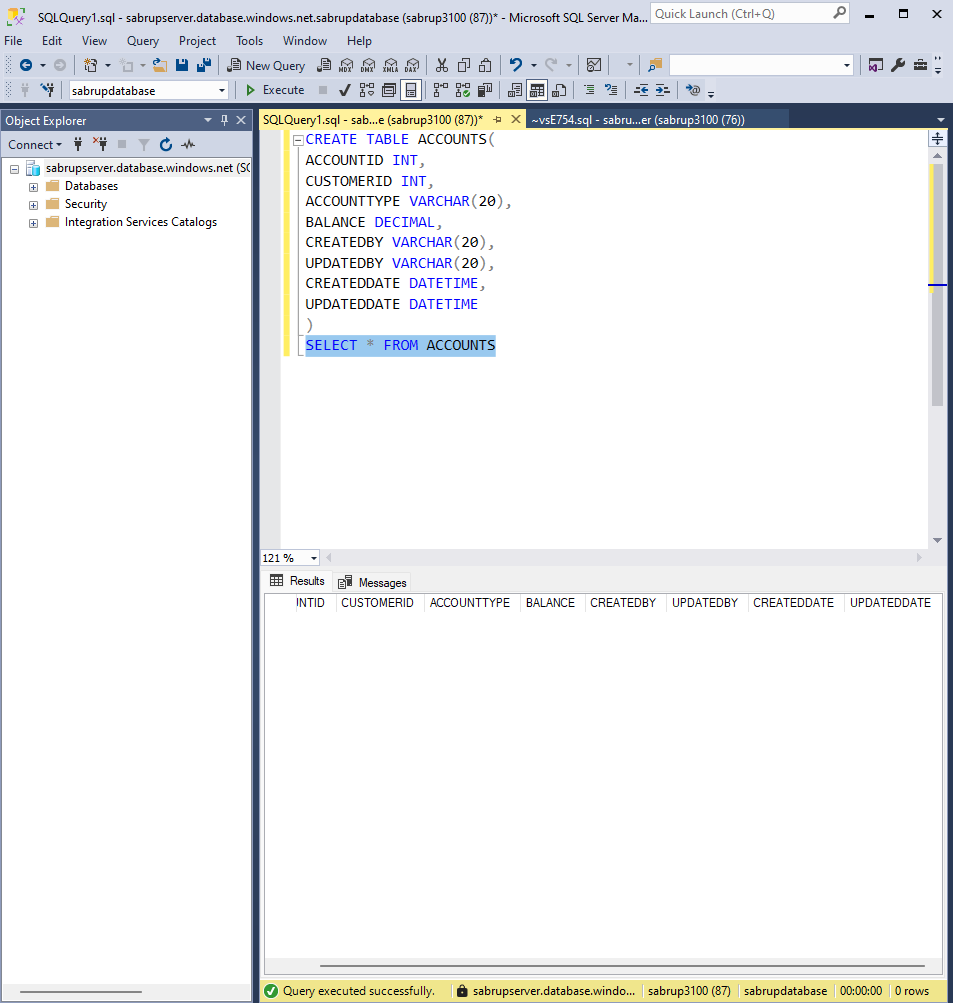
**The successfully running silver pipeline’s dataflow:**



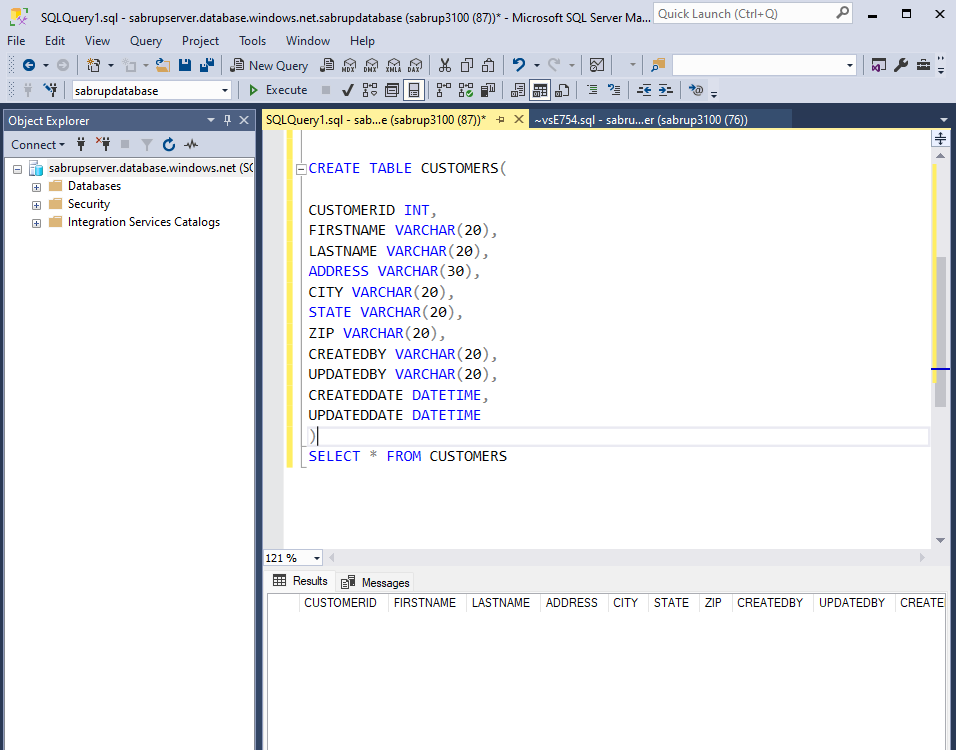
**Step 3: Dataflows using SCD Type technique (SCD 1 and SCD 2)**

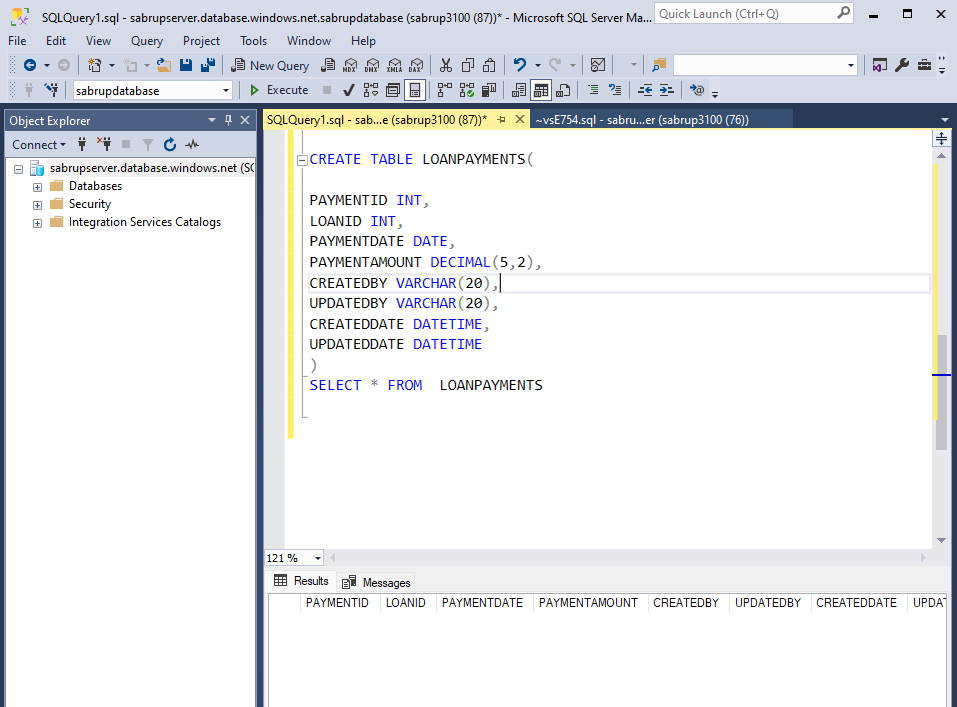
* **So far the data is in Delta format and in ADLS Gen2.**Next step is to store all the five files into Azure SQL Database in SCDType-1 and SCDType-2.
* For that there is a pipeline-”gold\_layer\_scdtype1\_scdtype2” which has a Dataflow.  
  
* I have store three of the files, accounts, customers and loanpayments using SCDType1 and the rest 2 with SCDType2.  
  
* **SCDTYPE-1:**

First and foremost is to create SCDType1 table that I already created with added createdby, updatedby, createddate, updateddate and hashkey columns for all three files:

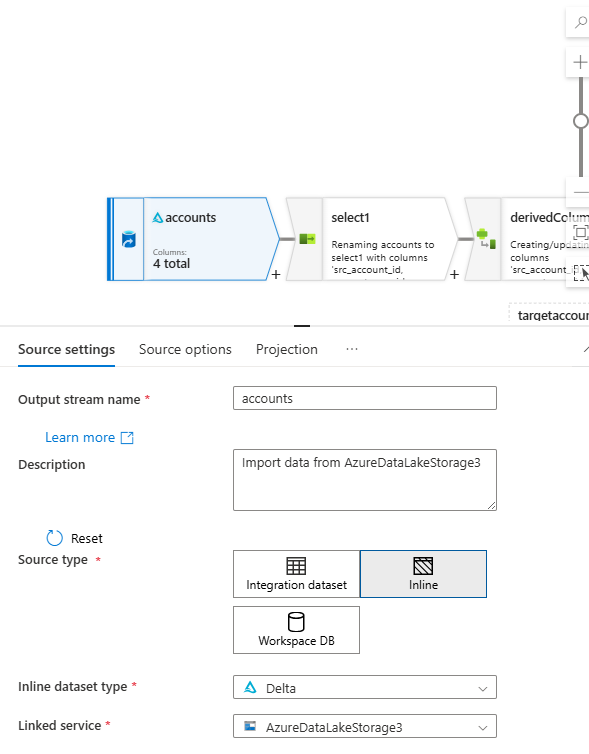


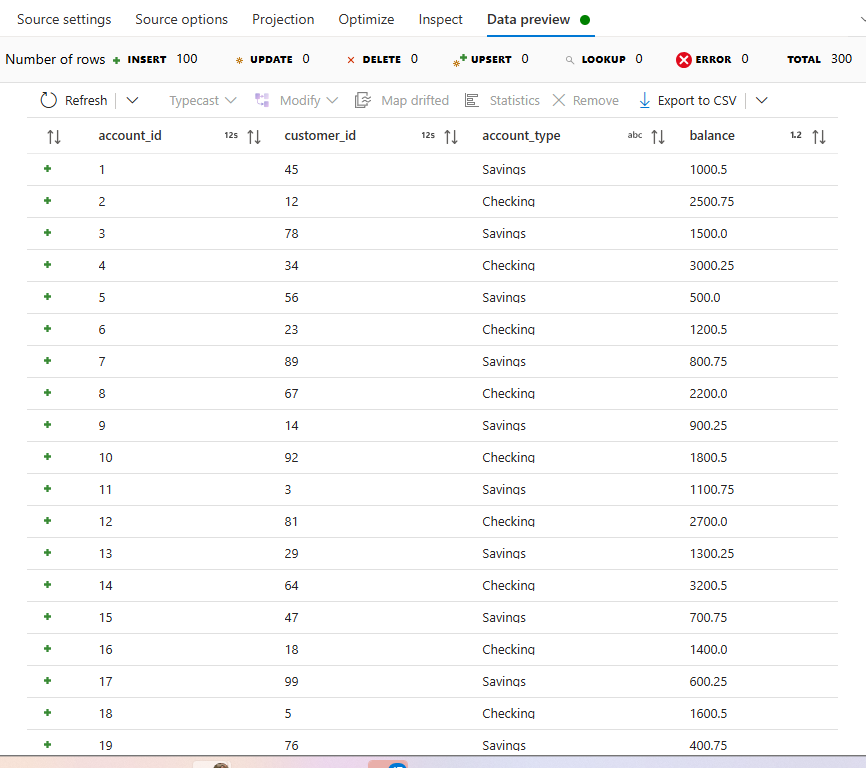
**And same with customers and loan\_payments**



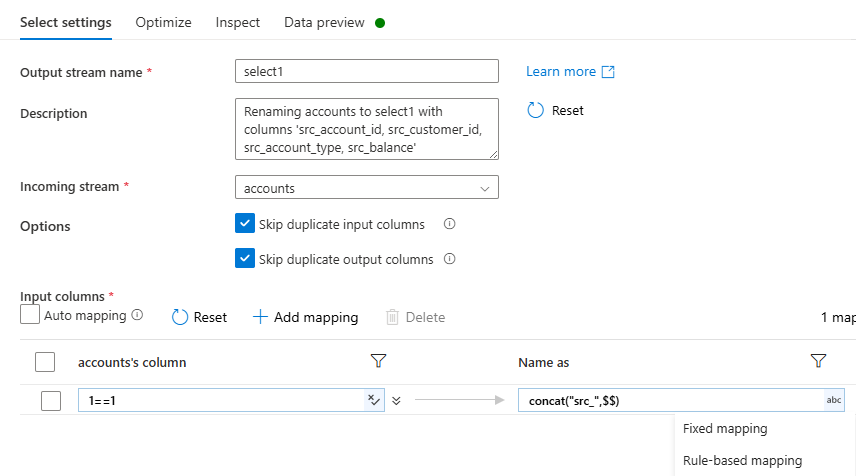


**1.** The first step is to add a source which is accounts folder that I stored in silvercontainer as Delta Format. For that the dataset type is “Delta” and the linked service is AzureDatalakeStorage3 since the delta files are in ADLS.

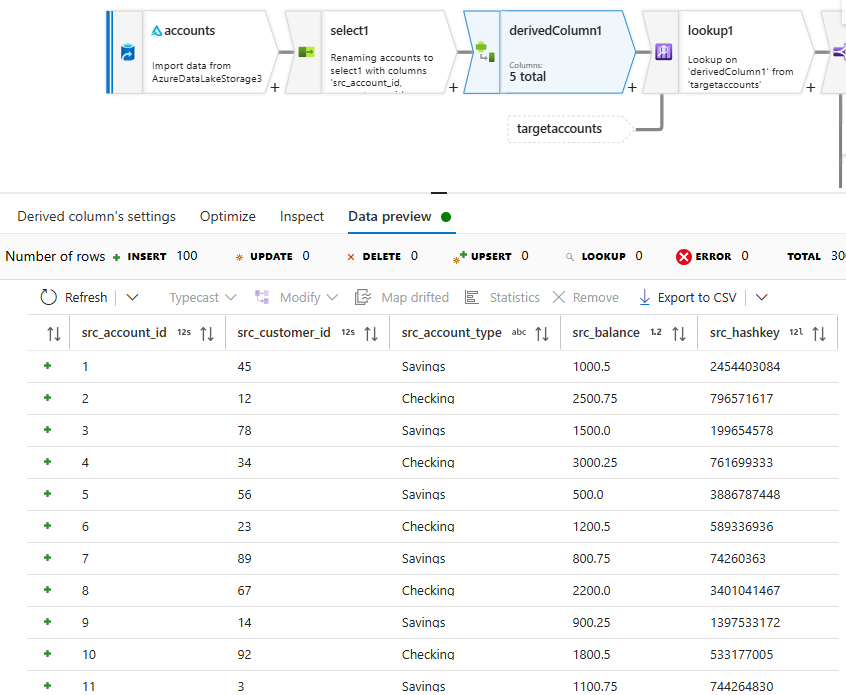
  
2. The source path is silvercontainer/accounts which I selected from the Browse option.  


3.After importing schema, the Data preview is below:  
  


4. Then I added select transformation just to rename the columns’ names to make them more clear.  
For that I added **Rule-Based mapping** and use **concat(“src\_”,$$)** to add src\_ as prefix with the source columns’ names.

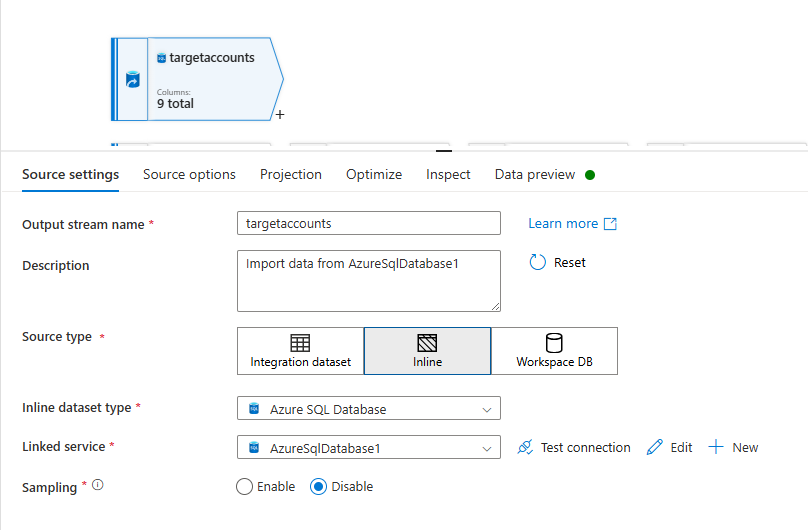


5. Then I used **derived column** transformation to add a new column named **src\_hashkey** which is created by the combination of number of columns and is integer type.  


6. The Data preview shows the src\_hashkey:  


7. Then I added another source basically it’s target. Here it’s ACCOUNTS SCDType 1 table in Azure SQL which I created to store the delta files to it.

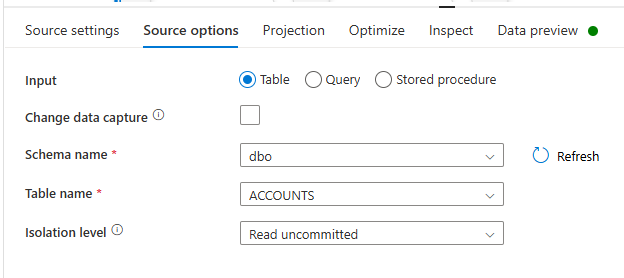
I named it as **targetaccounts.**



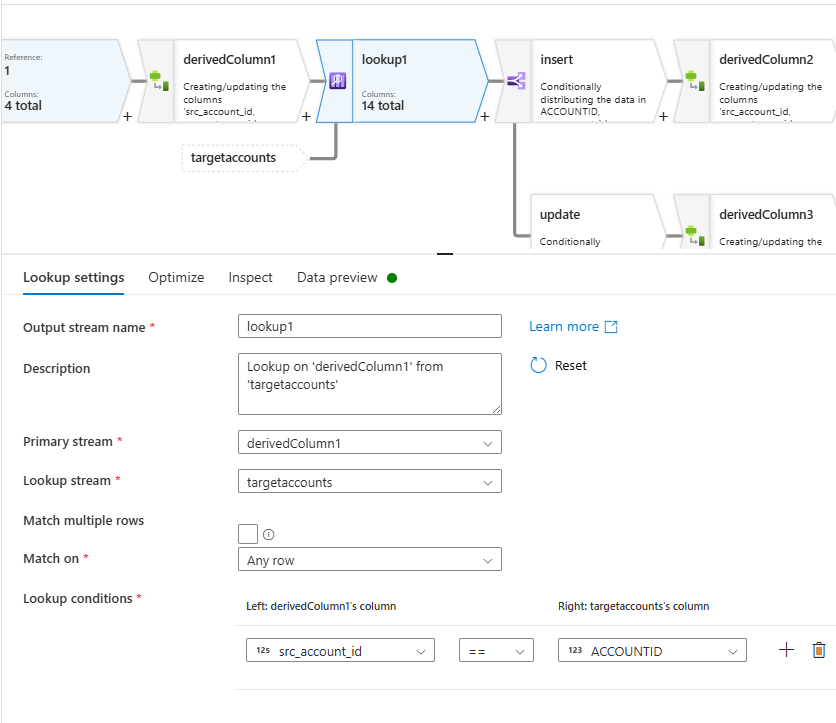
Here the dataset type is Azure SQL Database and the Linked service AzureSqlDatabase1.

**Schema: dbo**

**Table Name: ACCOUNTS**



8. After that I added lookup transformation which **works like left-join**.

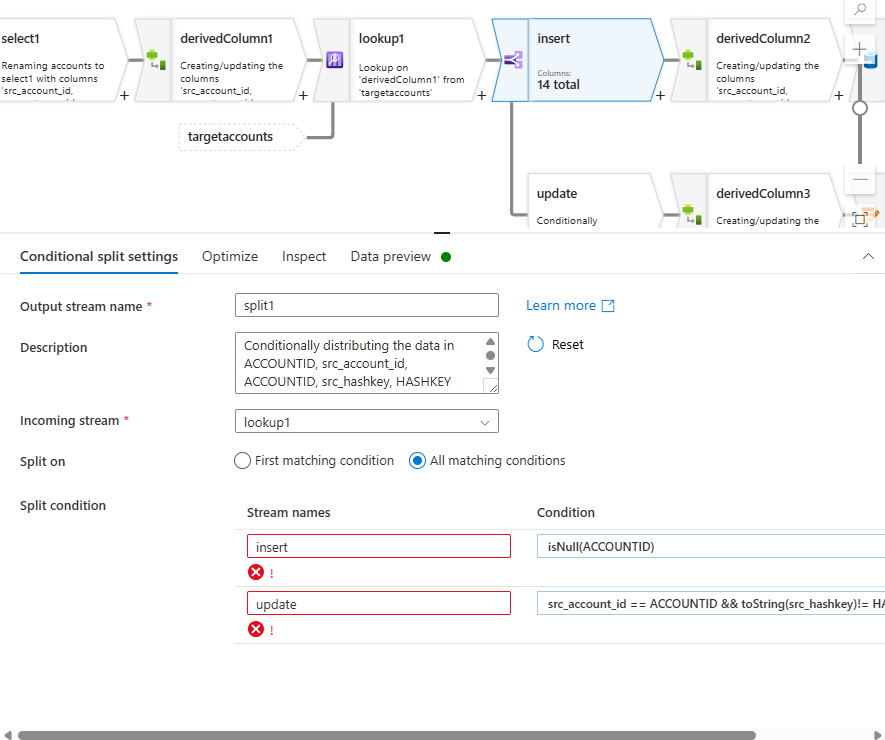


Here the output of derivedcolumn is its input and is joining with the another source(target). The condition satisfies if **src\_account\_id == ACCOUNTID.**

9. Ninth step for SCDType-1 is to add **conditional split** transformation.

**Insert if the target ACCOUNTID is null that means record doesn’t exist and will perform insertion.**

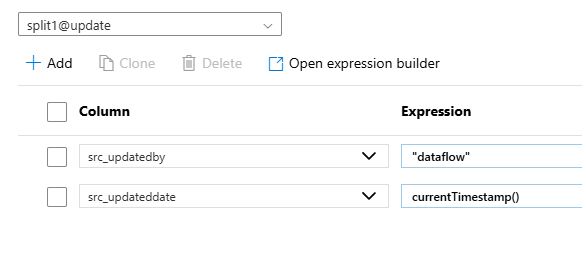
**Perform update if the Target ACCOUNTID matches with source account\_id but the hashkeys doesn’t.**



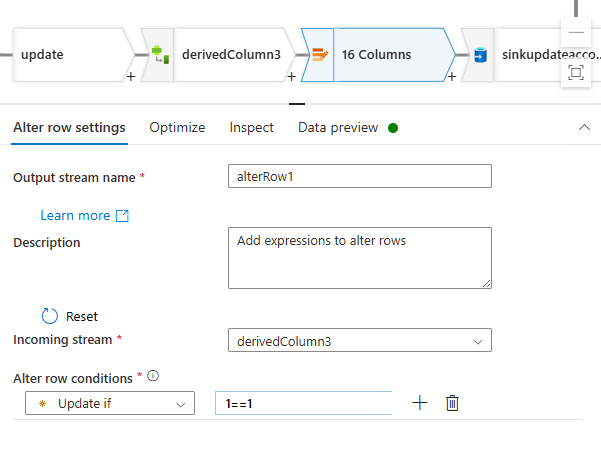
10. After conditional split, I added derived column transformation to add new columns.  
For insertion: I have added **createdby** and **createddate** with values **“dataflow”**  and **currentTimestamp()**  also I added **updatedby**  and  **updateddate** with values **“dataflow”** and  **currentTimeStamp()**.

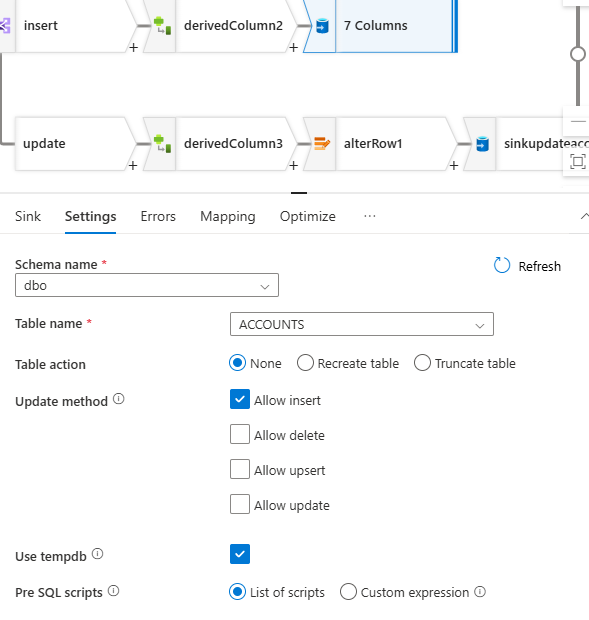


Similarly for update: I added **updatedby**  and  **updateddate** with values **“dataflow”** and  **currentTimeStamp()**  respectively.

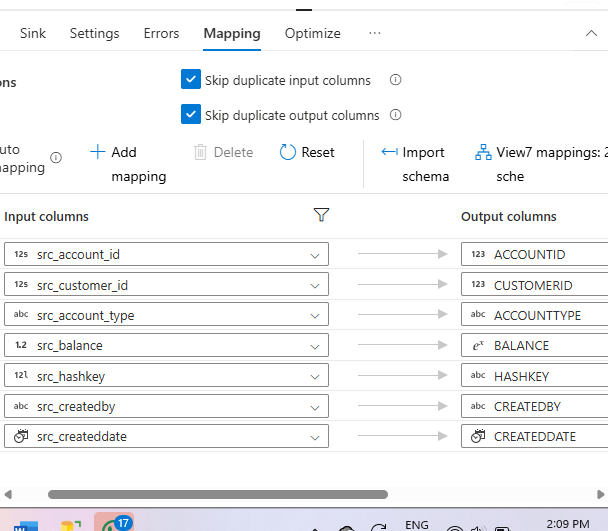


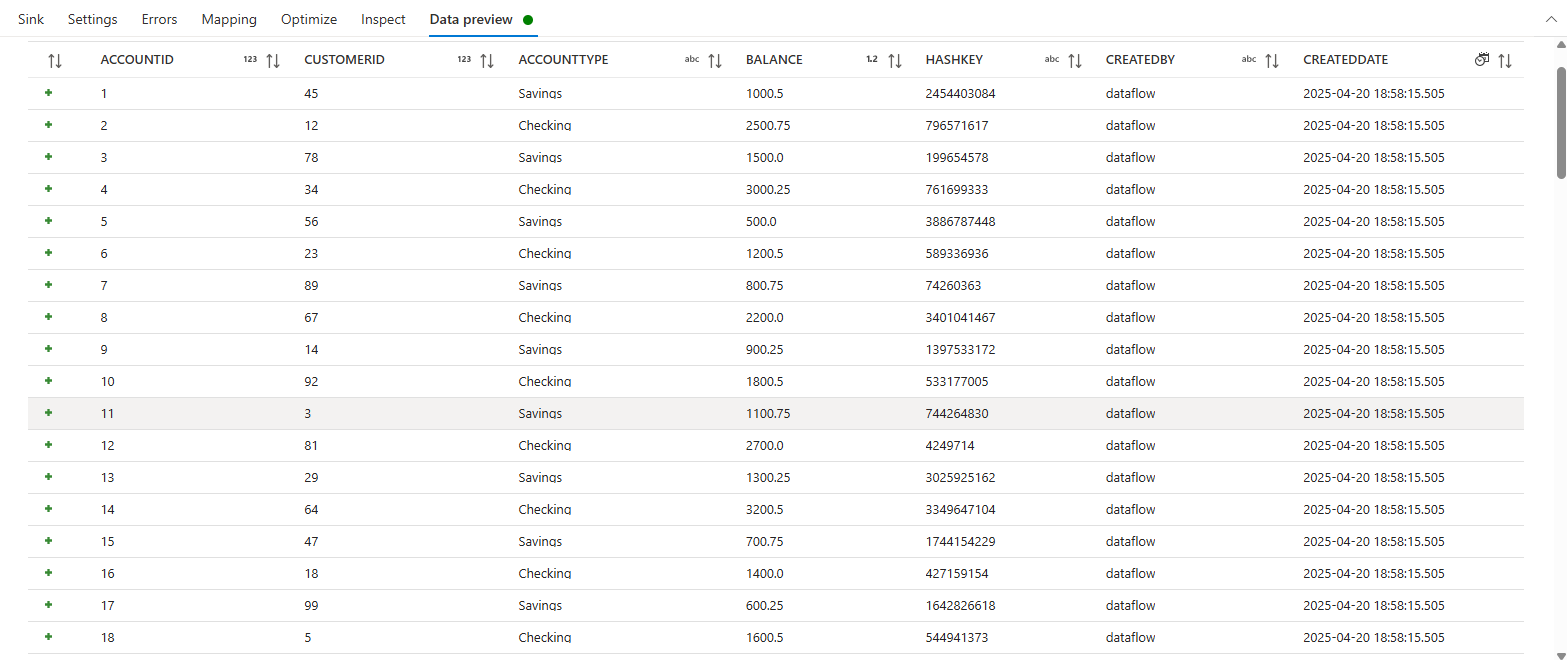
11. On the side of updation, I added alterrow transformation with condition always **true** .

  
12. Lastly, the sink transformation stores the data in the tables created.  
Sink to Azure Sql -> schema name: dbo and Table: ACCOUNTS



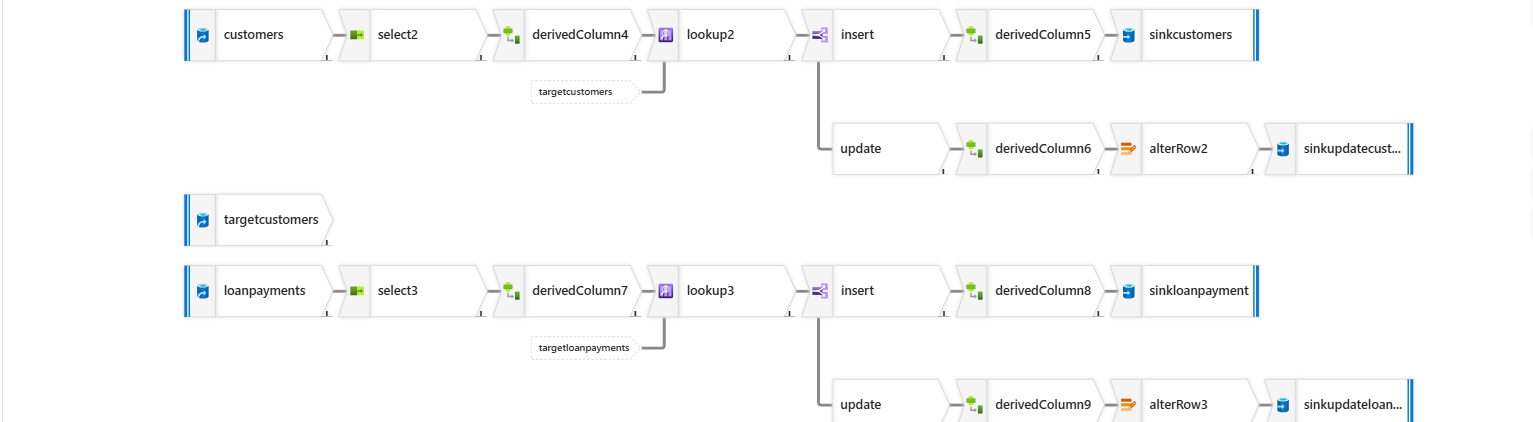
13. Here the mapping plays crucial role to get correct output.

I disabled the **Auto-Mapping** -> Import schema. After deleting unnecessary columns and matching the formats the mapping looks like:  


Here’s the Data Preview for Insertion Sink:  


Similarly, the sink does same thing for updation but updates updatedby and updateddate column.

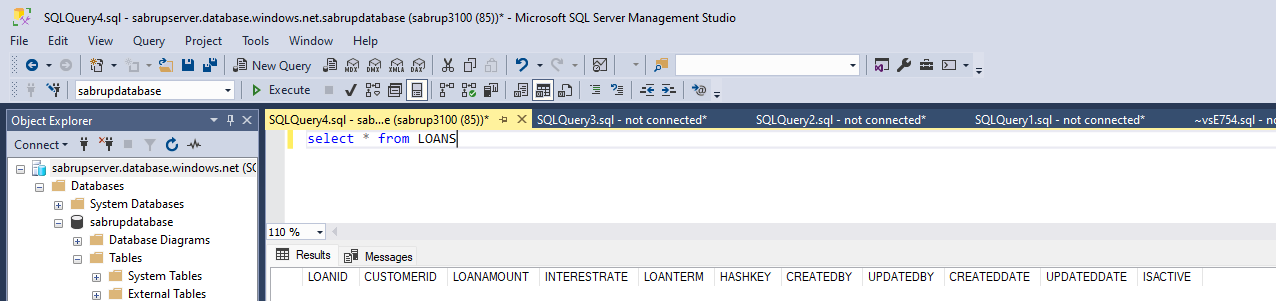
14. For another two files-customers and loan\_payments the same steps I repeated.

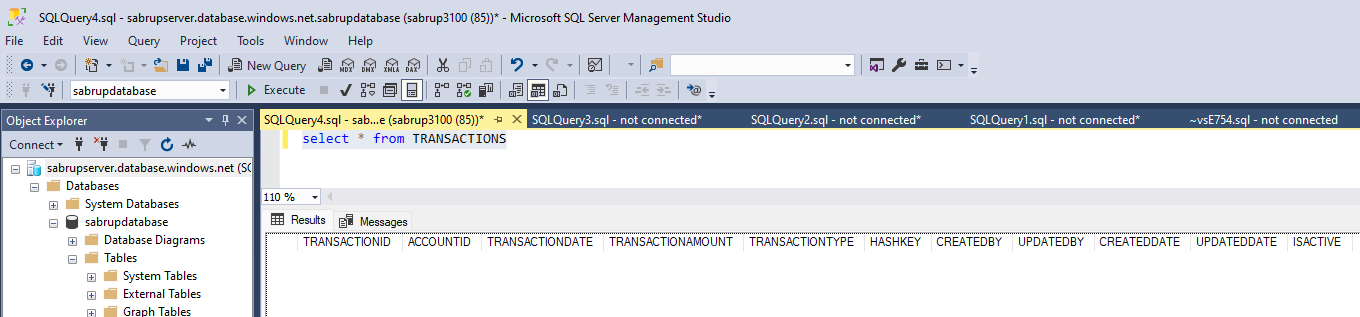


* **SCD TYPE 2**

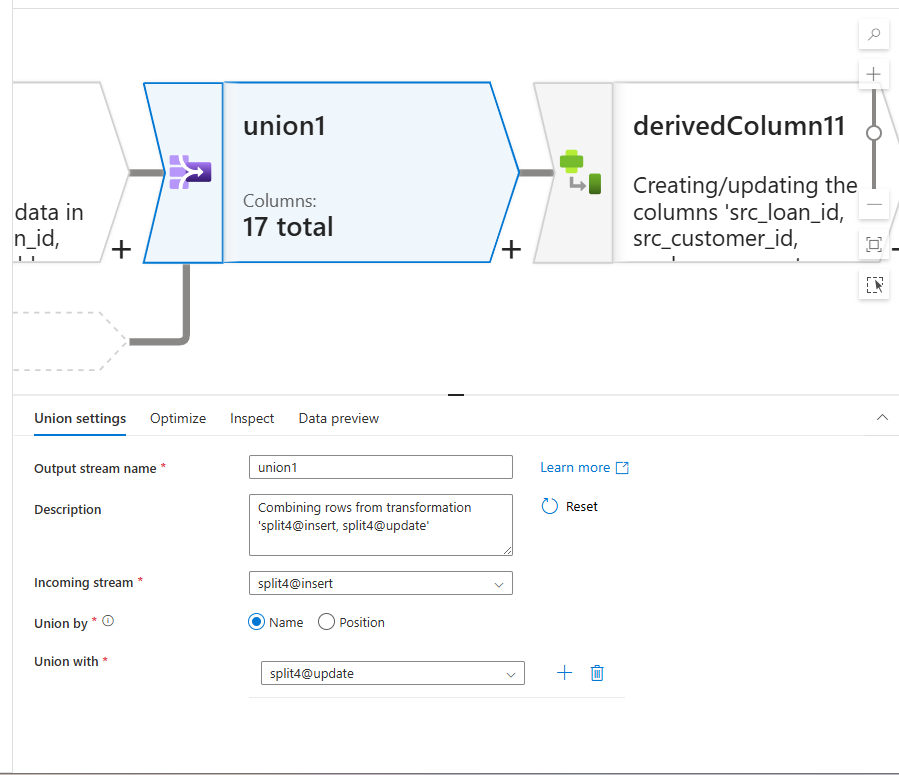
**For loans and transactions** I used SCD type 2 to store the data in Azure Sql.

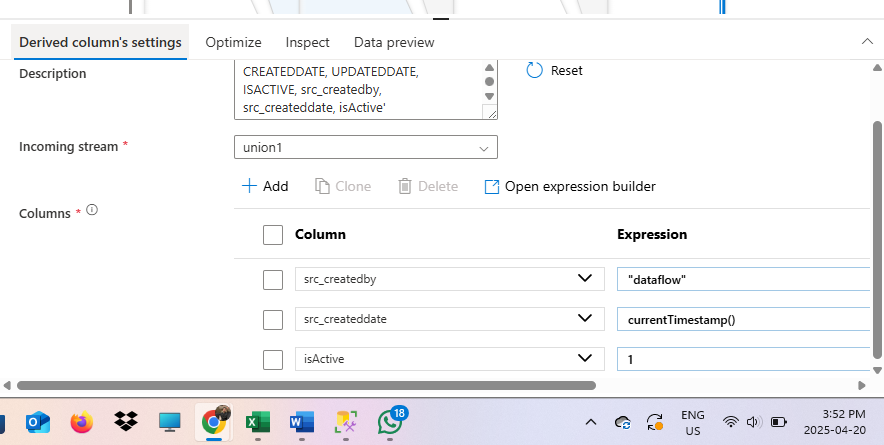
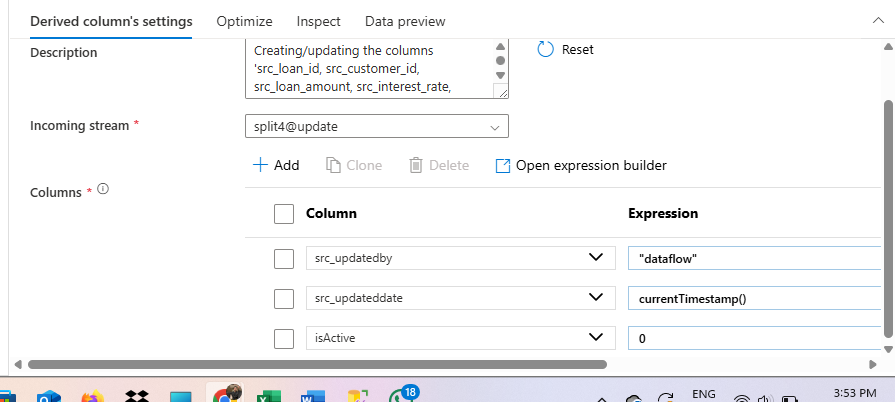
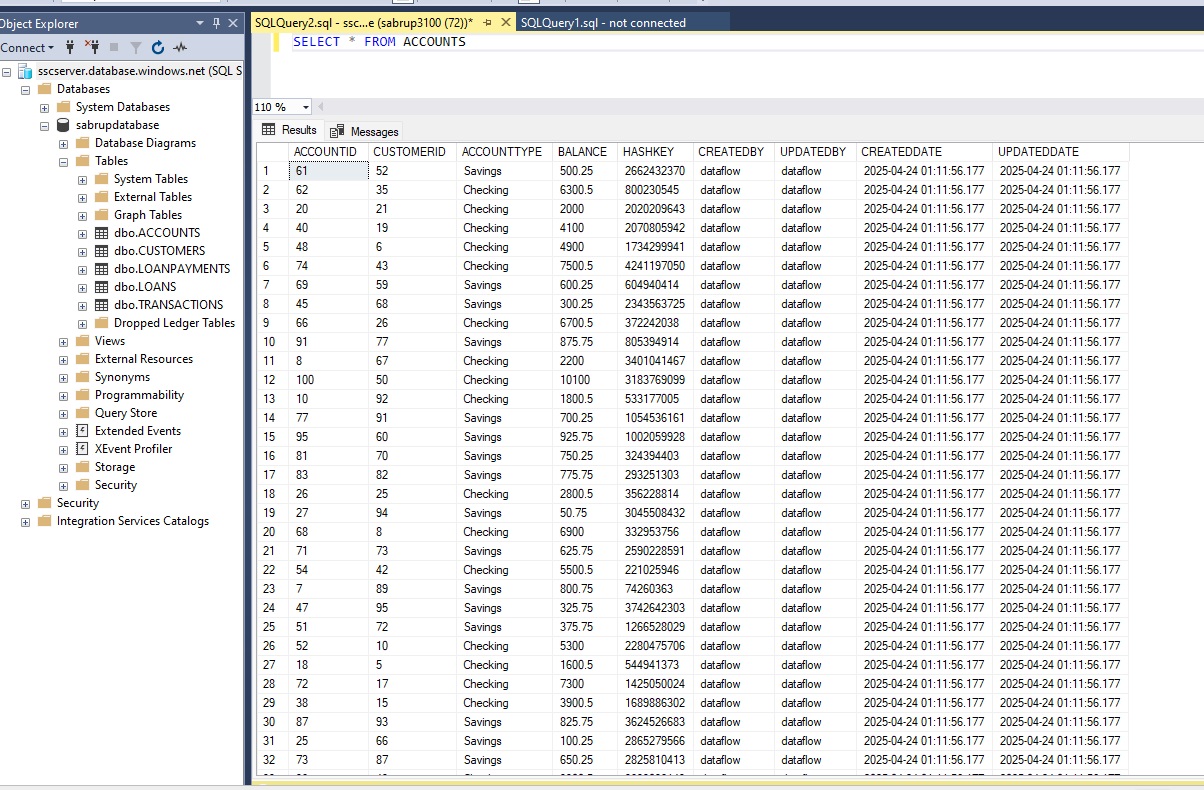
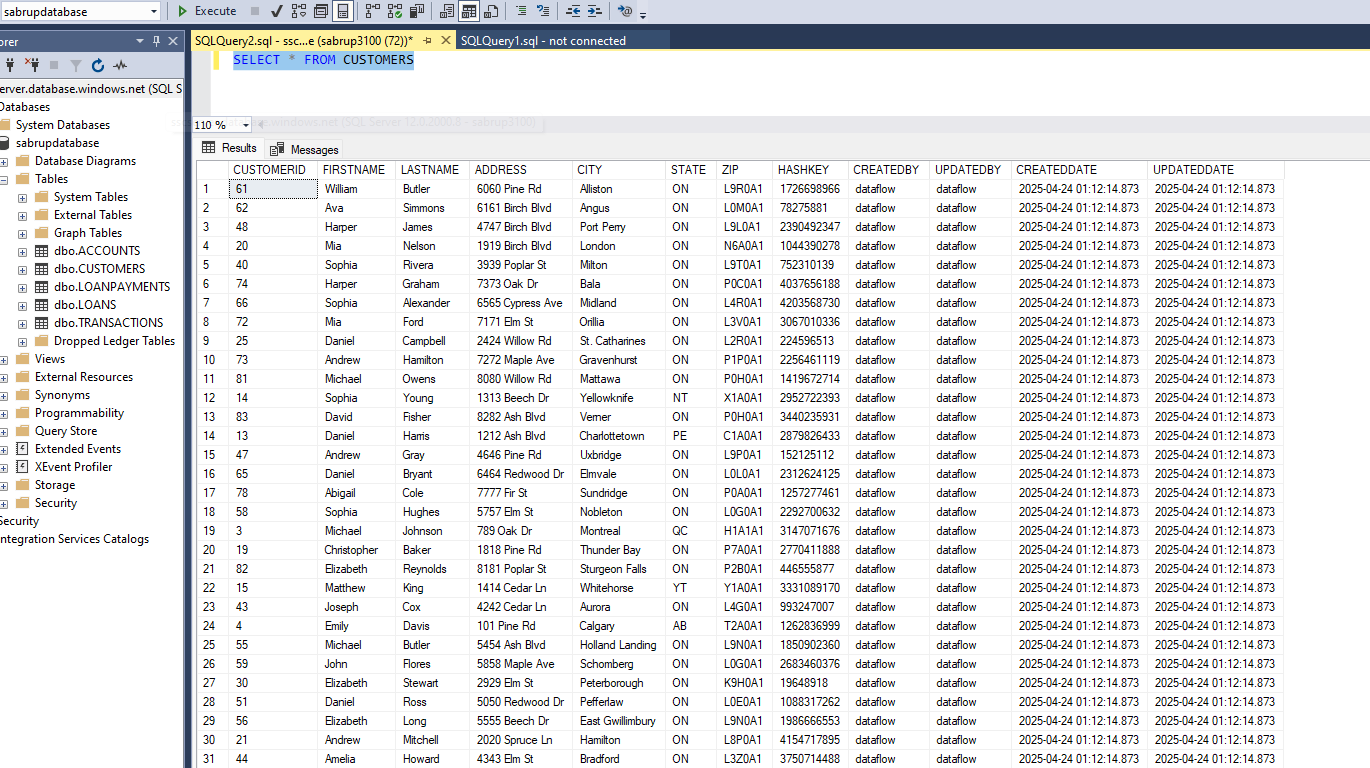
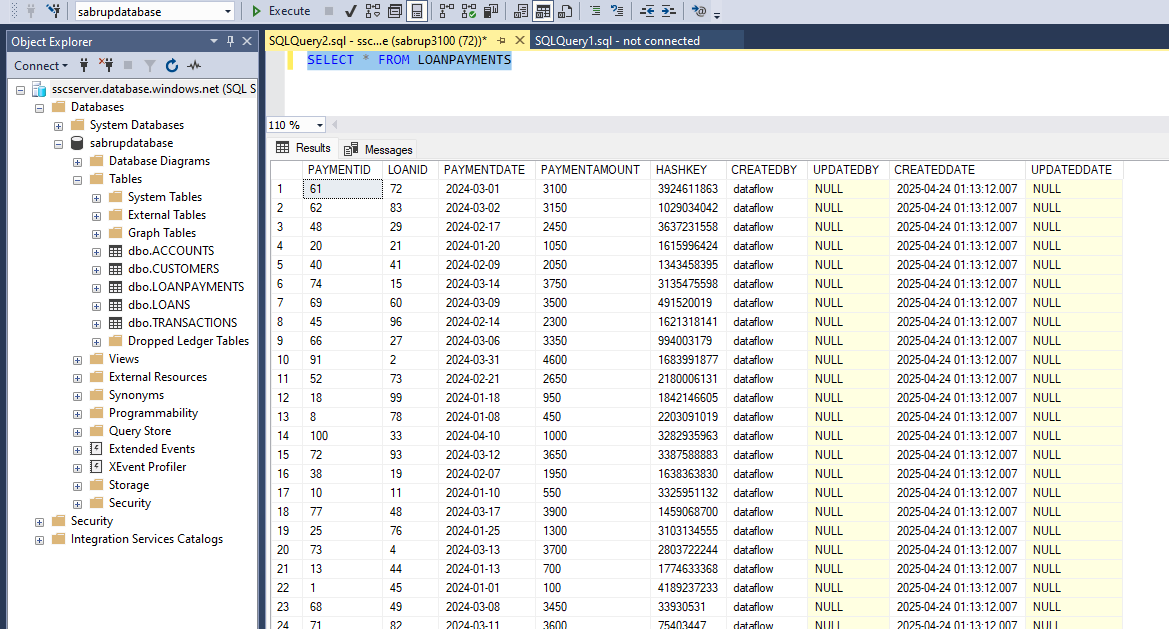
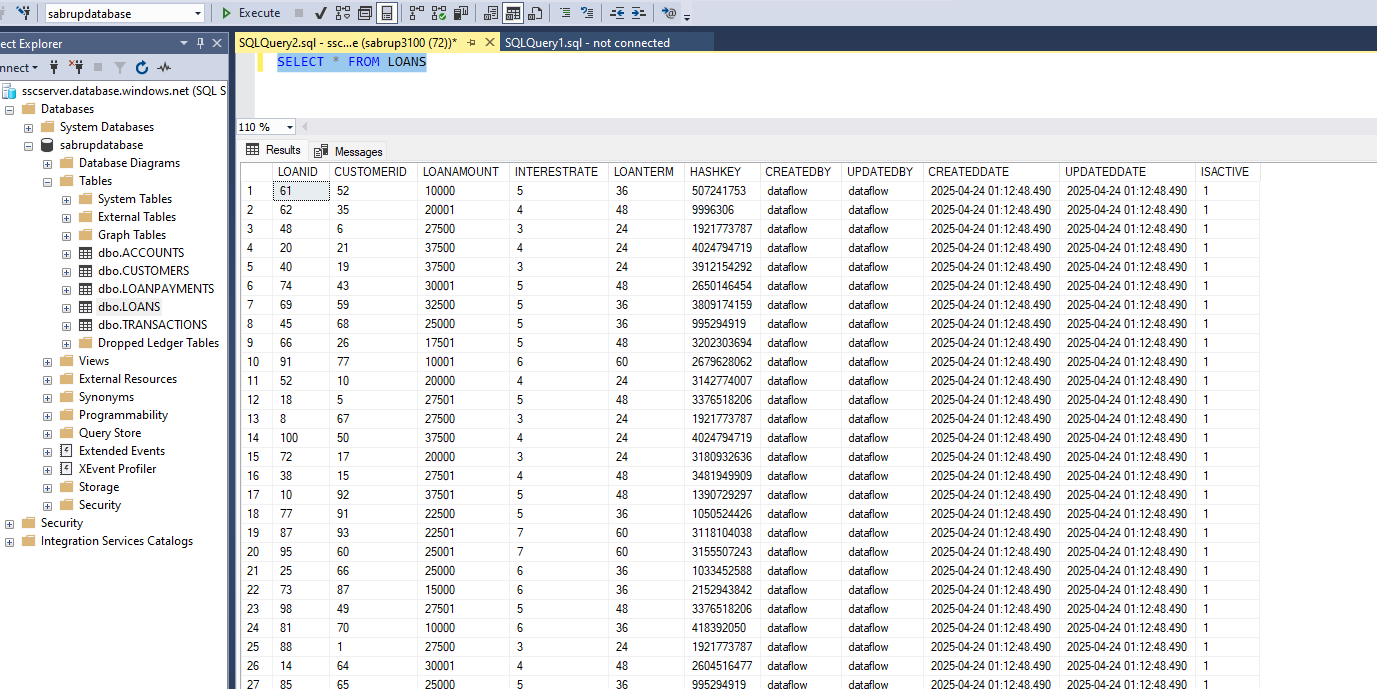
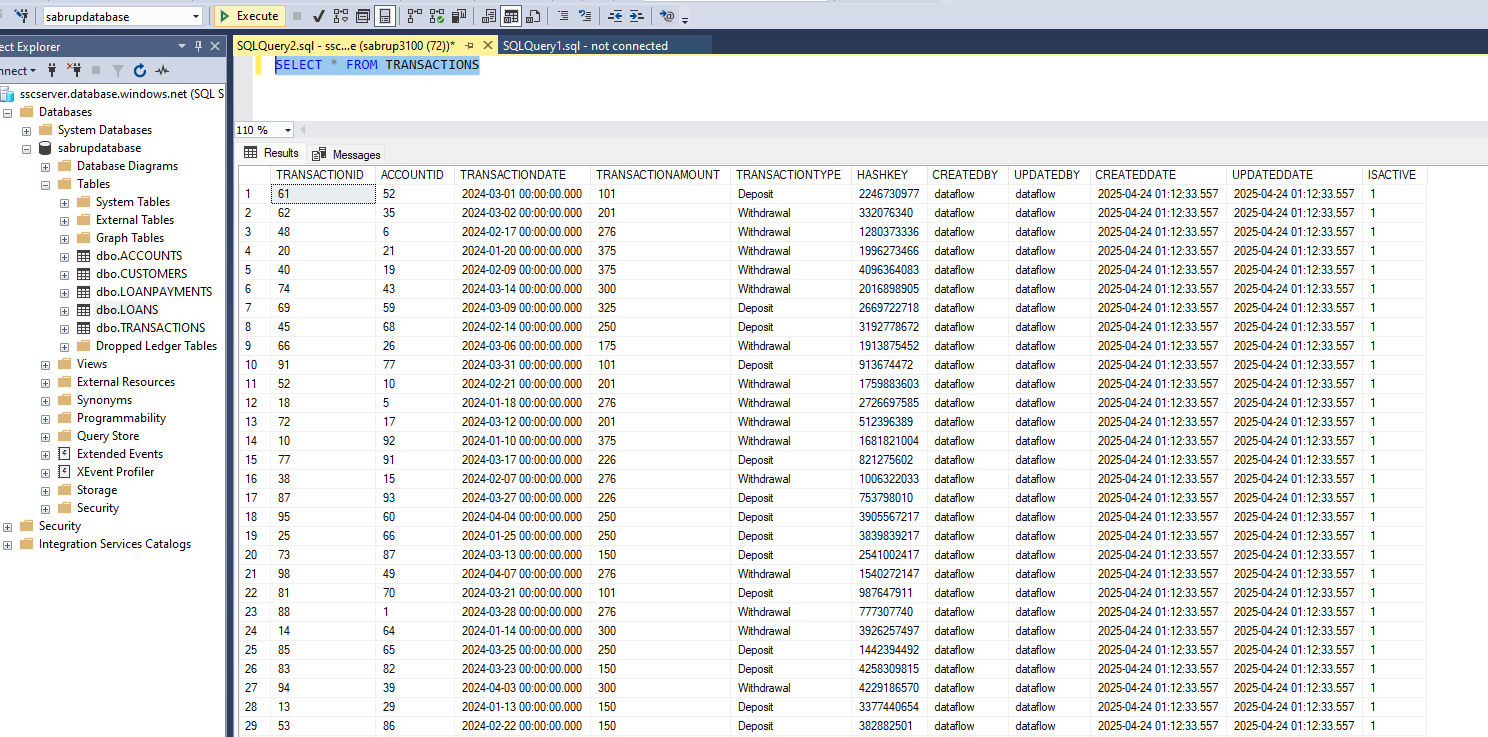
1. For SCD Type 2 I created two more tables in Azure Sql with same CREATEDBY, CRFEATEDDATE, UPDATEDBY, UPDATEDDATE, HASHKEY aliong with ISACTIVE columns.
2. ISACTIVE is INT type stores 0 or 1 to reveal the current state of that particular row.

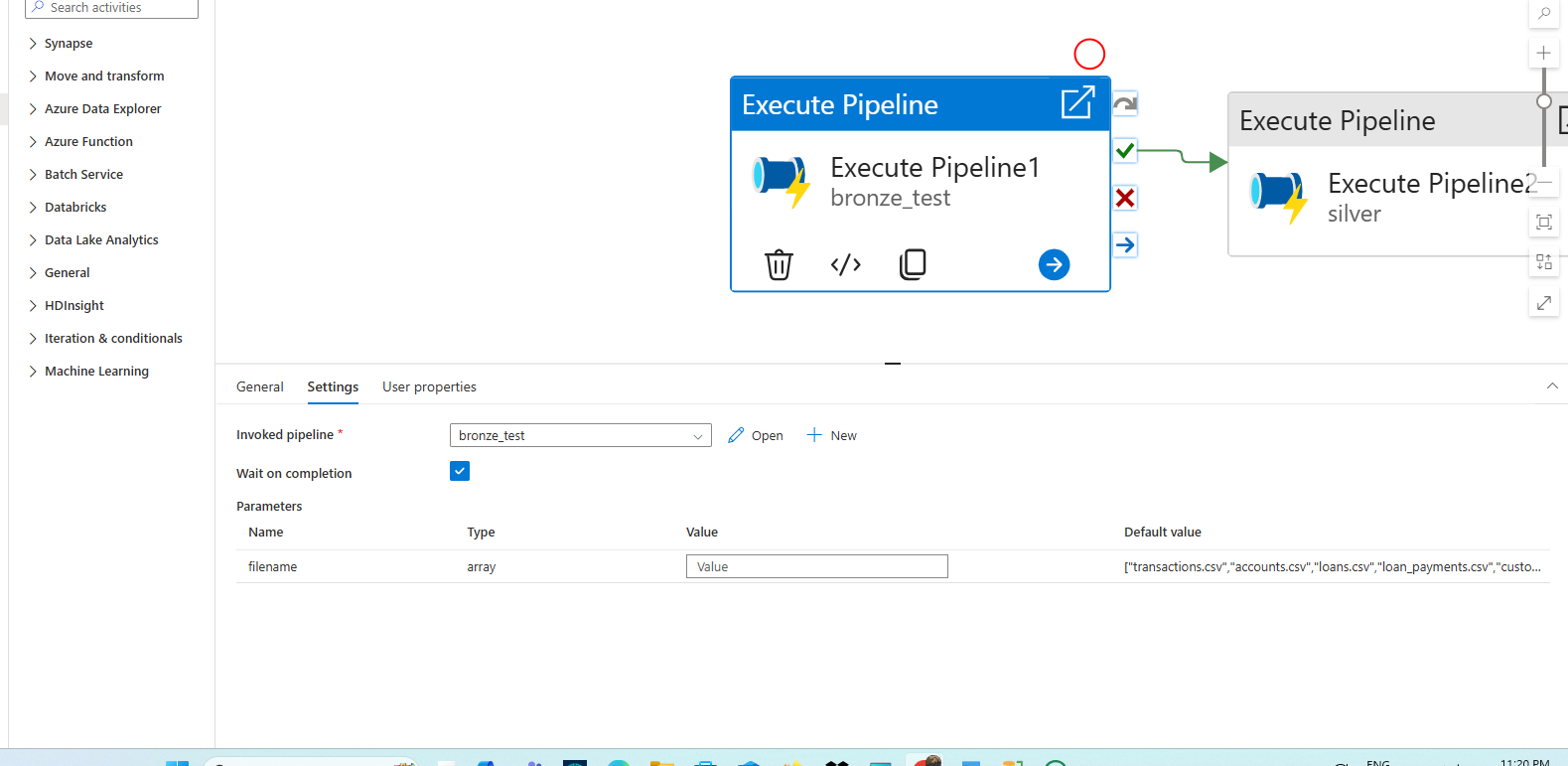
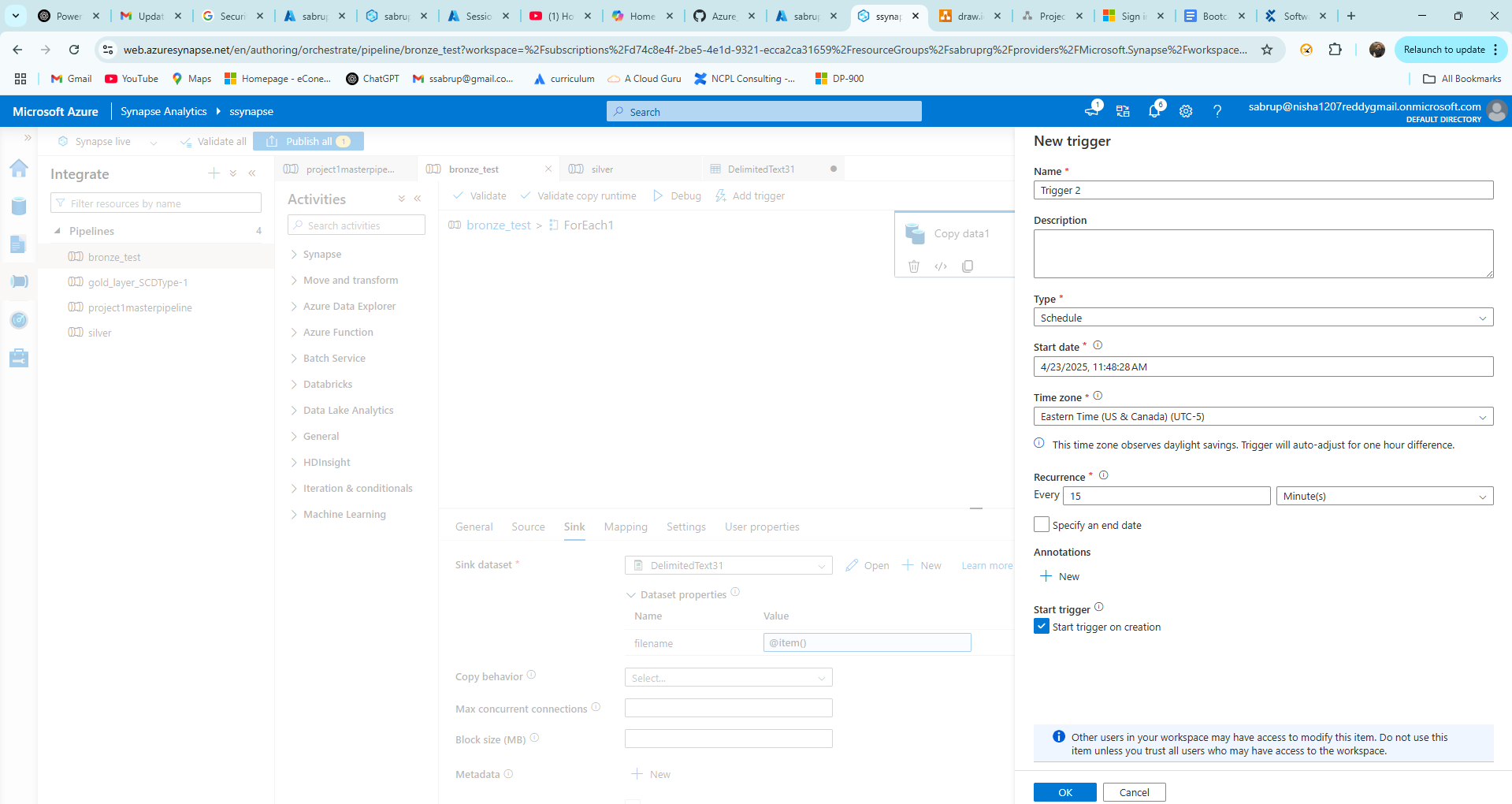
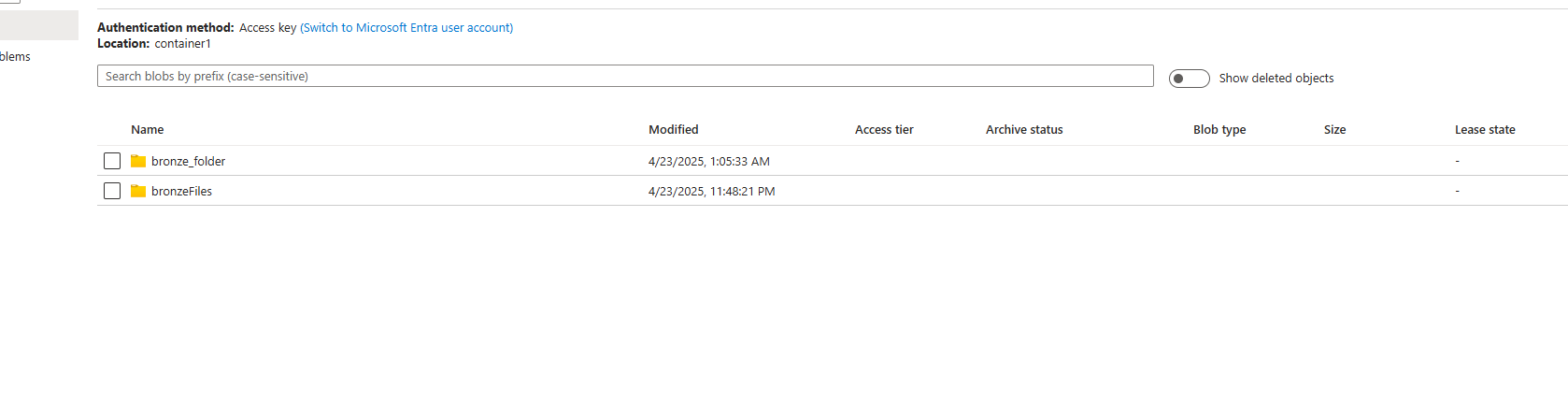
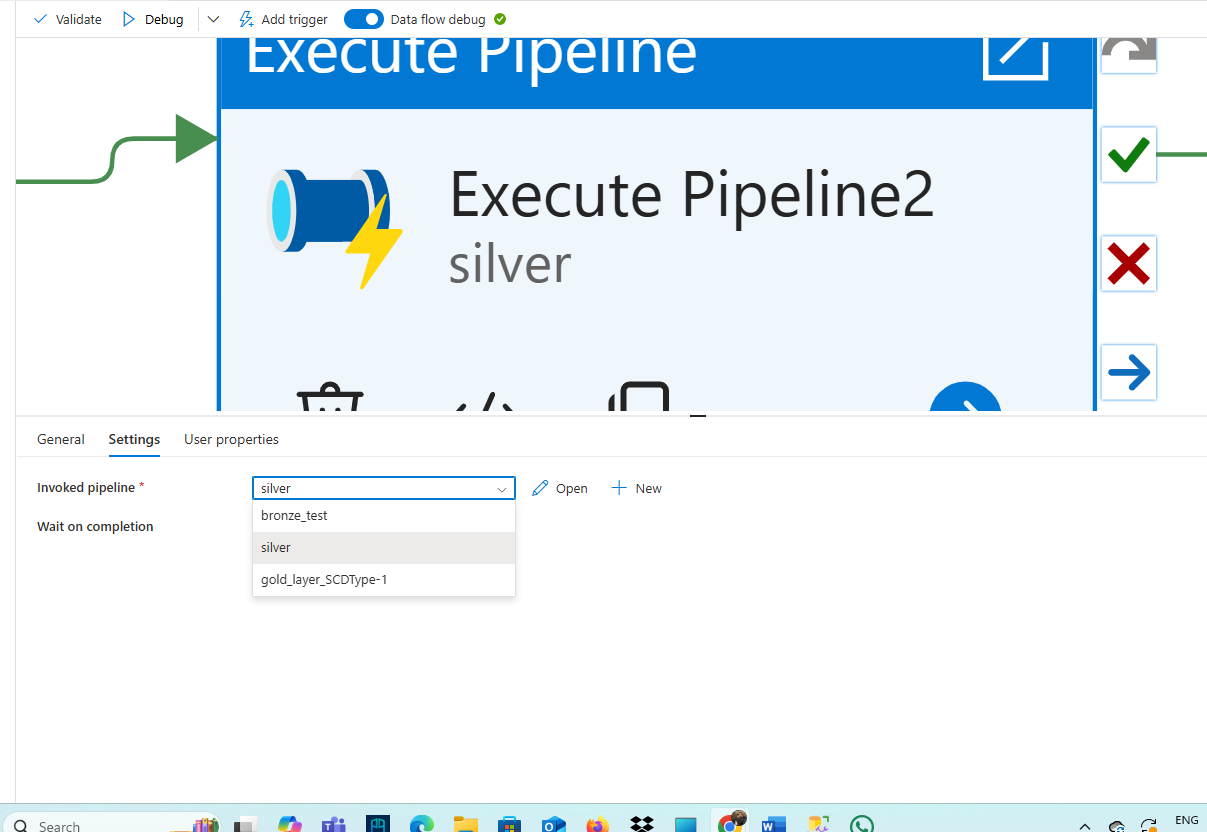
If theb ISACTIVE0 that means this row got updated and a new updated row has been inserted with ISACTIVE value 1.  


TRANSACTIONS SCDTYPE 2 TABLE:  


1. SCD Type 2 is quite similar to SCD Type 1 but Type 2 keeps the historical data as well.   
   Even if we update a particular row, it doesn’t overwrite it, instead adds new row (with new data) and make the older one’s state **0.**
2. The rest evering is same like the derivedcolumn transformation to add new column, **src\_hashkey** , lookup transformation to join the source and the target.
3. Then conditional split again same. But after conditional split, there is **union transformation.** As here even while updating a new row is created, for that there is union of insert split with update.



1. Again the derived column transformation same but a new “isActive” column has been added for both update and insert.
2. In insert the isActive value is 1 since it’s a new record even while updating.  
   
3. In update side the isActive value is 0. As this row would be the history of the newly added. So , it’s state would be 0  
     
   
4. The successfully running Dataflow to store SCDType 1 and SCDType2  
   
5. The **ACCOUNTS SCDType 1** table with data:  
   
6. CUSTOMERS SCDType1:  
   
7. LOANPAYMENTS SCDType1 table:  
   
8. Loans table( SCD Type 2):  
   
9. TRANSACTIONS TABLE(SCD Type 2):  
   

* **Master Pipeline:**1. Create a new pipeline, named **project1masterpipeline  
  2.** Add **Executive Pipeline**Under settings select the first pipeline, **bronze test  
    
  3.**  Again repeat the same steps for silver and gold layer  
    
    
    
    
    
    
    
    
    
    
    
    
    
    
    
    
    
    
    
    
  **Trigger:  
  1.**  To schedule the pipeline, we have to use Trigger.  
    
  As in the above SC I have created a new trigger to schedule the bronze pipeline to a particular time and date. At the given time the pipeline will automatically gets debuged.  
    
  It automatically generates the folder.

**Key Vault:**Here I have created two secrets “systemusername” and “systempassword”, store the credentials to connect the self-hosted integration run time.  
