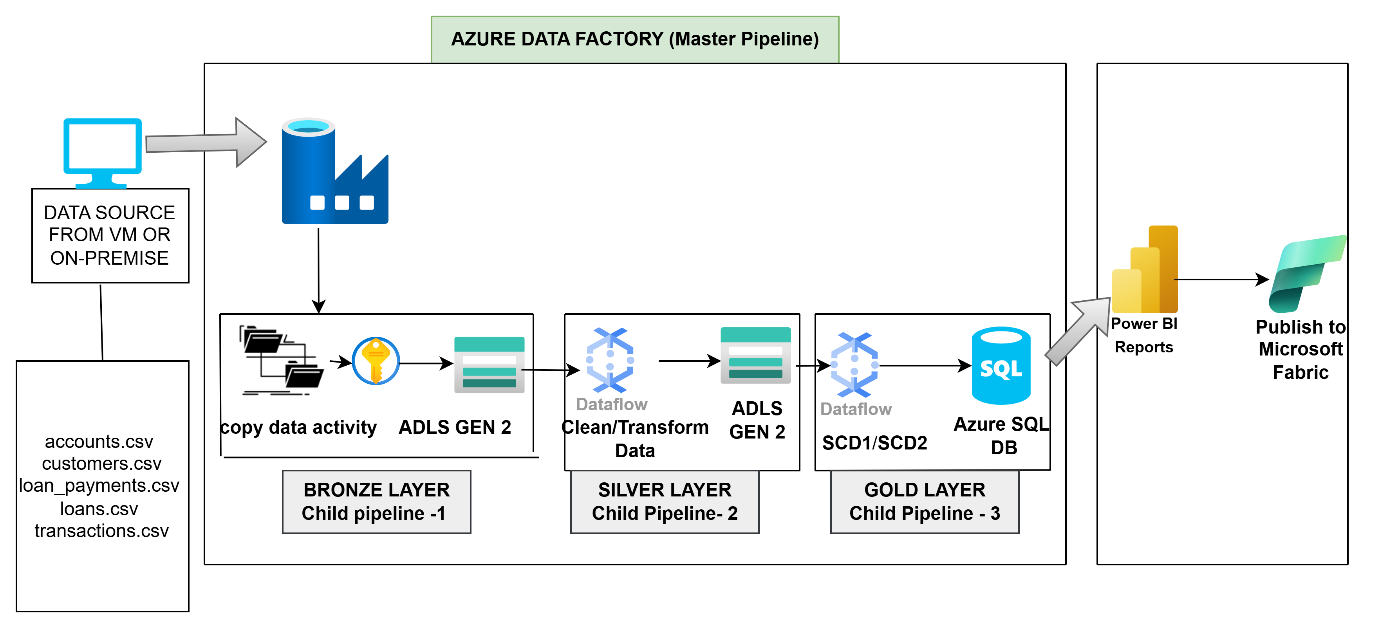
**Data Pipeline for Customer Account Analysis**

**Objective**

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

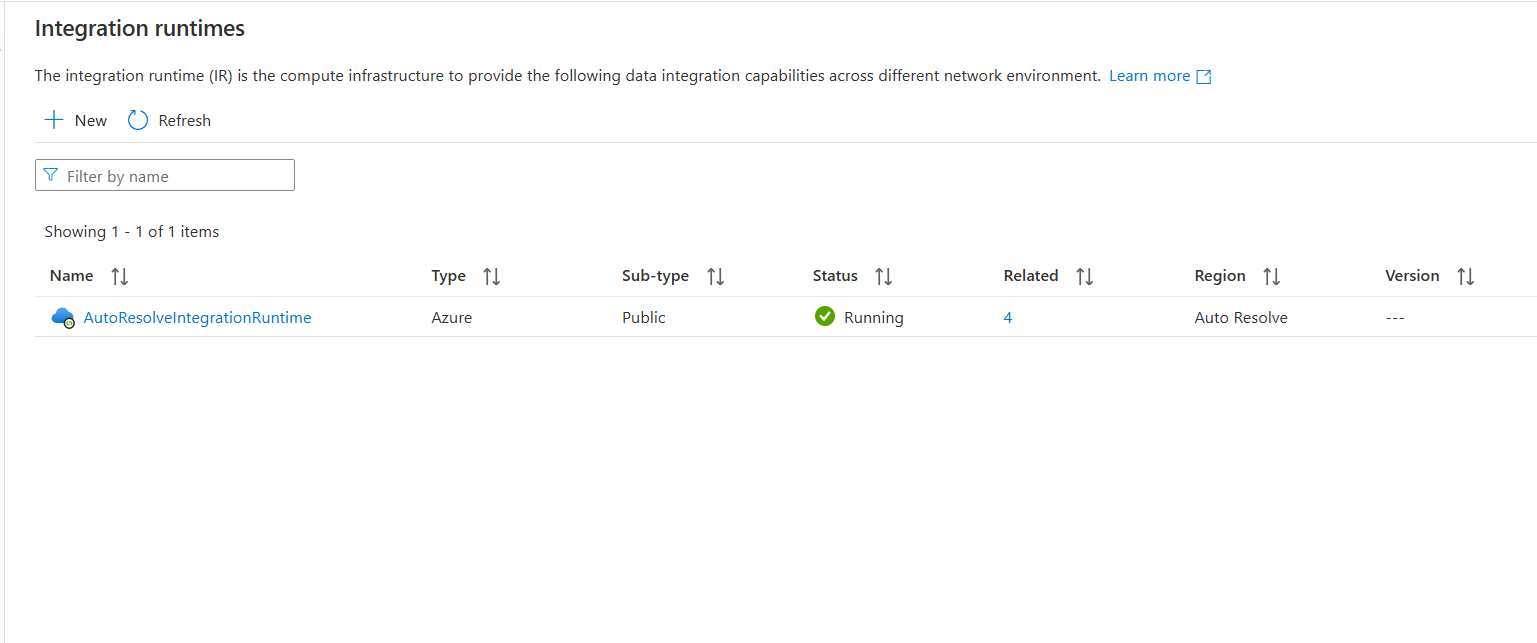
**Architecture**

****

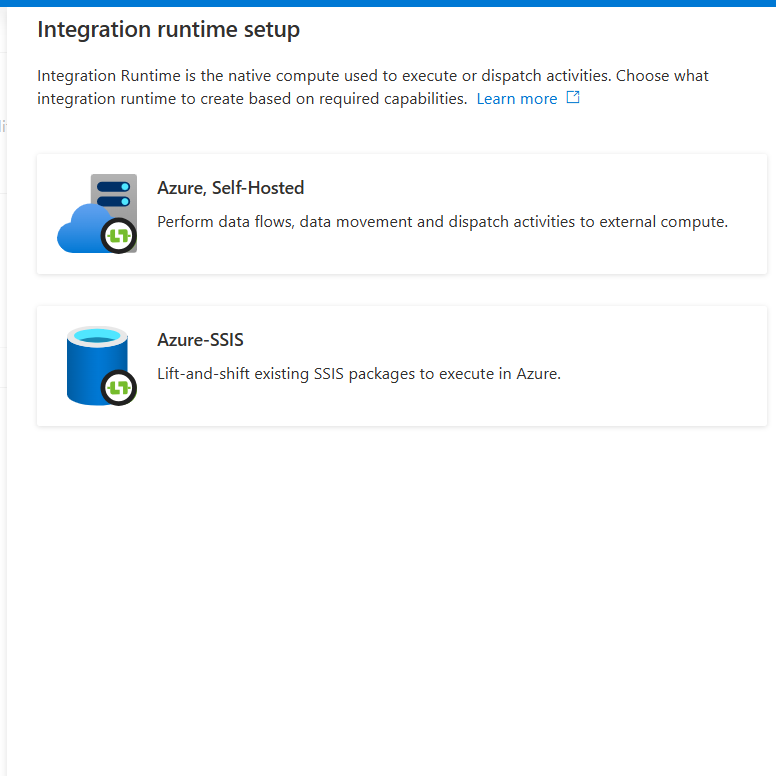
**Steps for Data Ingestion (Bronze Layer)**

Step 1: To run a self-hosted integration runtime download and then install it

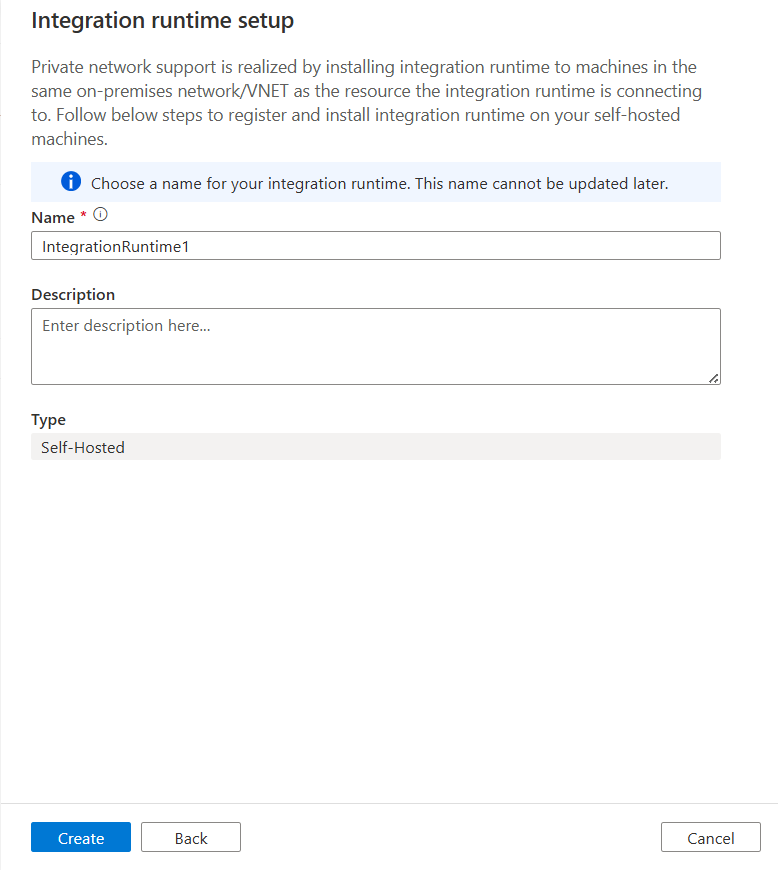
* First, we need to create a new Self hosted integration runtime



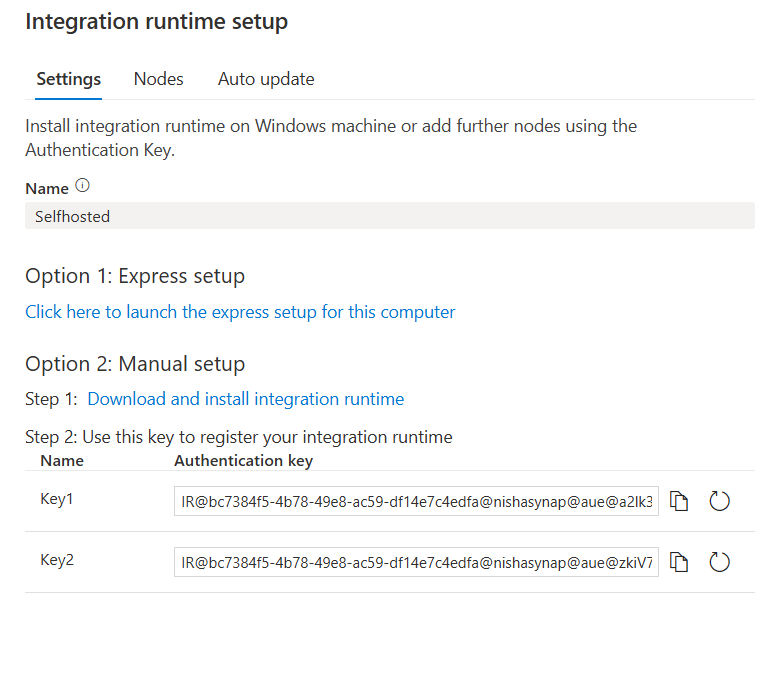
* In this step we need to choose azure self hosted



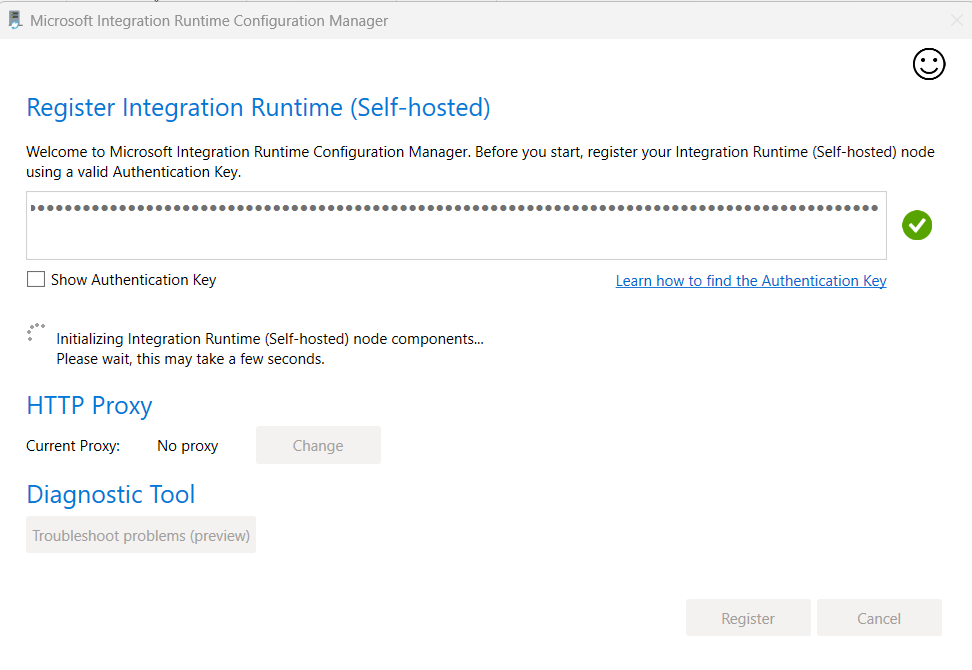
* In this as well lets choose self hosted and create it

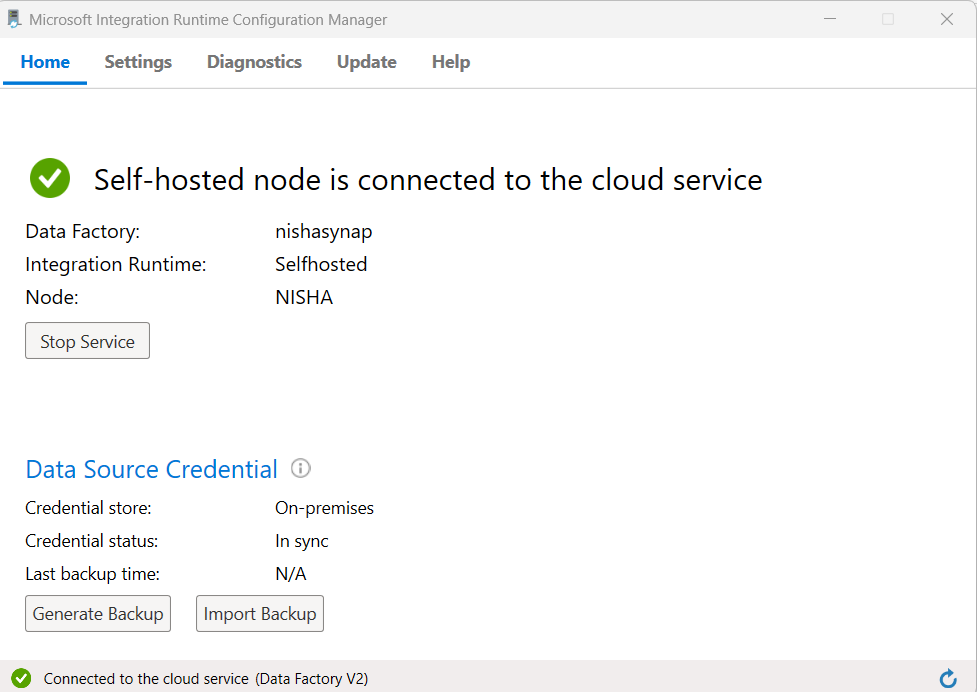


* After this it will generate two keys like mentioned below and do the following steps



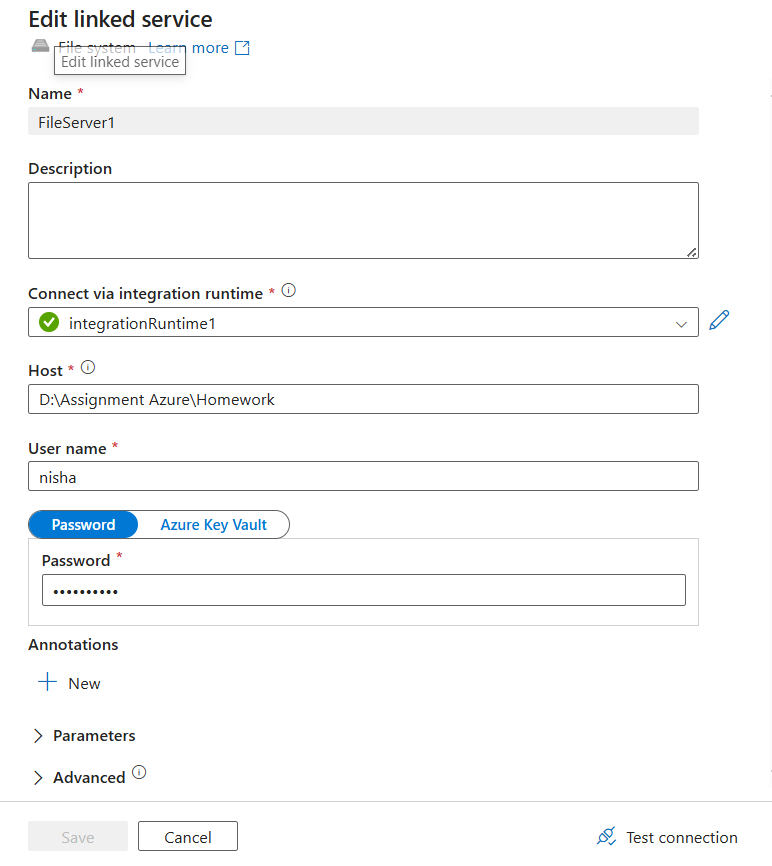
* This is the place where we need to paste our key





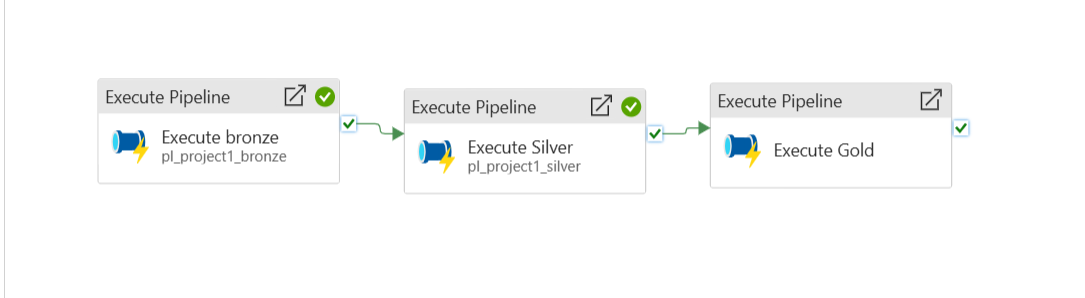
Step 2: After creating the Integration runtime, create a linked service for File Transfer

In the host tab, give the storage details where the file is located on our local machine, and for the username and password we need to provide the system login details

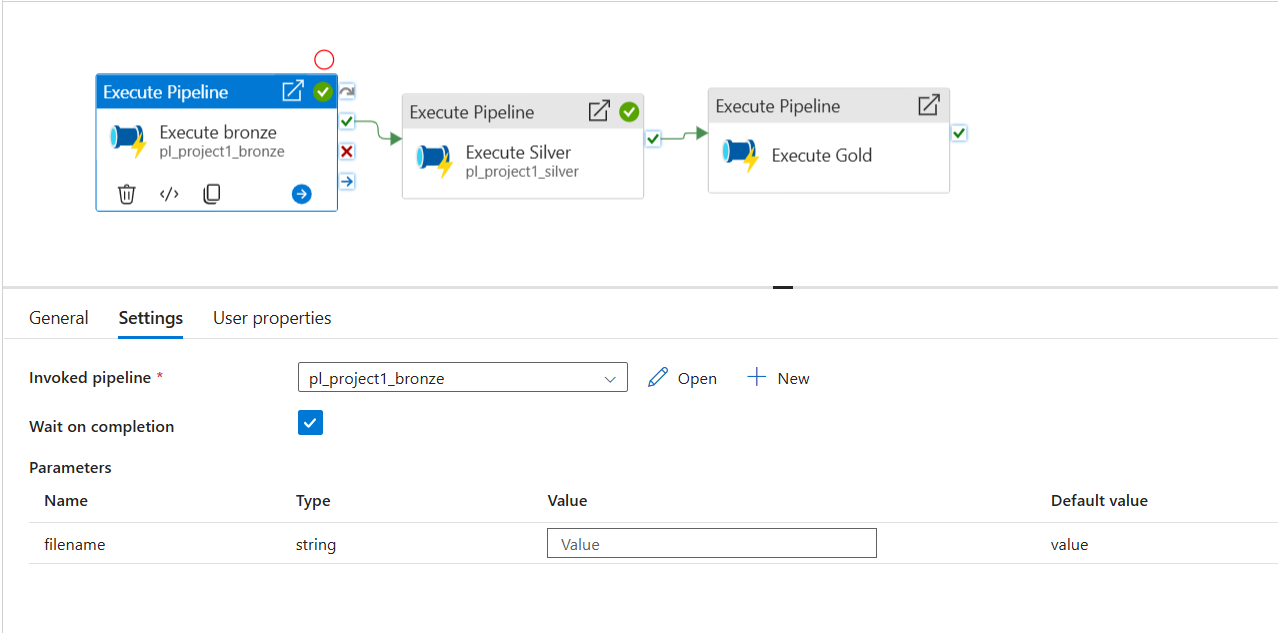


Step 3: Create a master pipeline with three child pipelines inside

* The Bronze Pipeline is used to load the data to the ADLS storage account
* The silver pipeline is used to cleanse the data
* The gold pipeline is used to use SCD transformation to the cleansed data

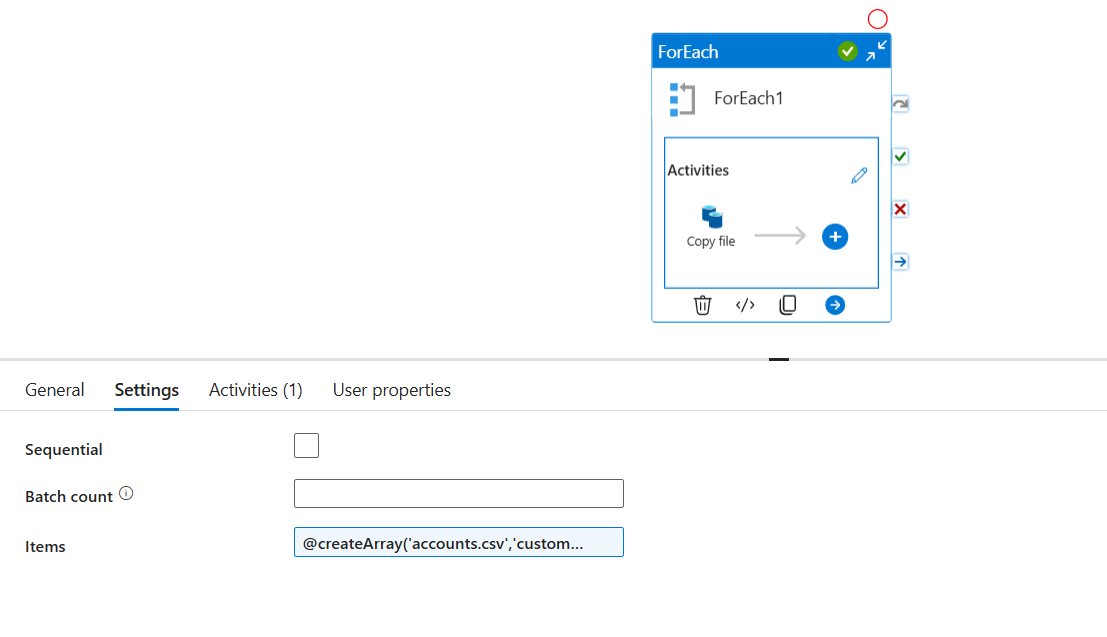


Inside the execute pipeline activity, we need to call the pipeline where we are doing the copy activity function

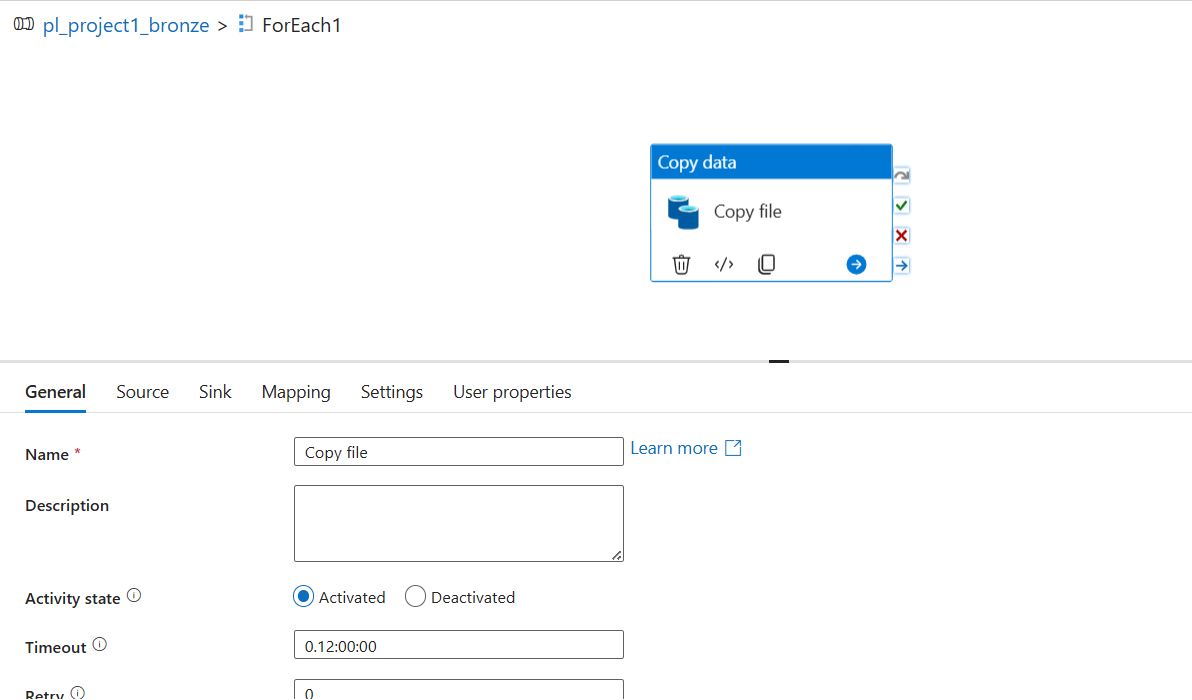


Step 4: Create a foreach activity so that it will help in iterating all files present inside a particular folder. Pass the below command inside items

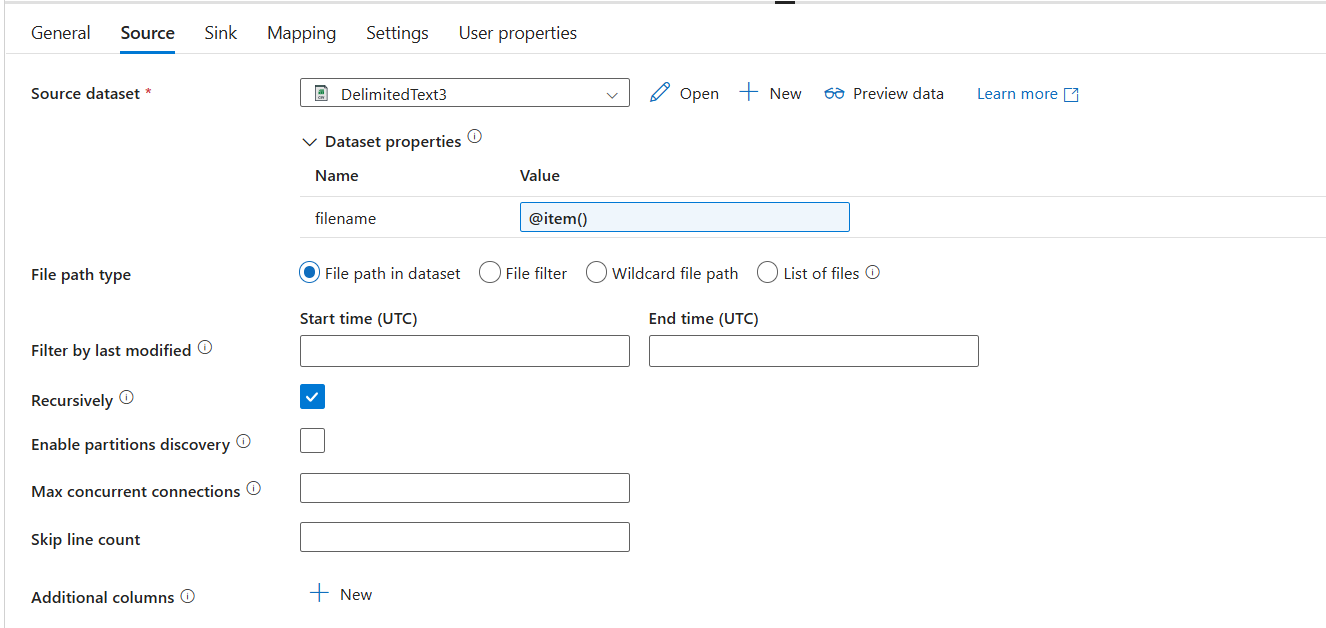
@createArray('accounts.csv','customers.csv','loan\_payments.csv','loans.csv','transactions.csv')



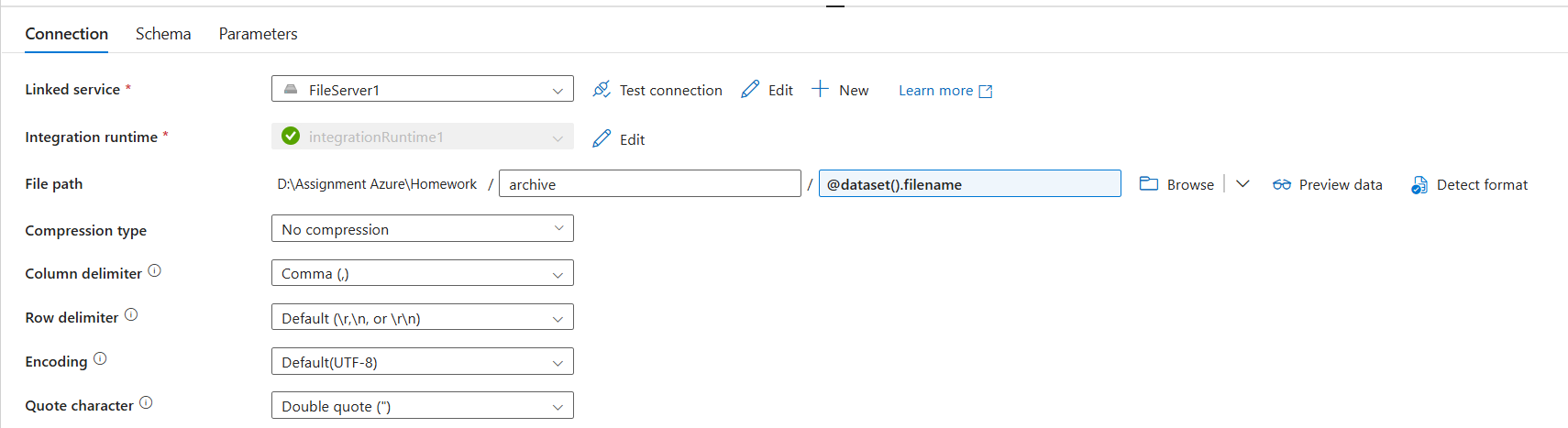
Step 5: Inside the foreach activity, open a copy activity, which is used to store the files inside ADLS storage account (Bronze Container)



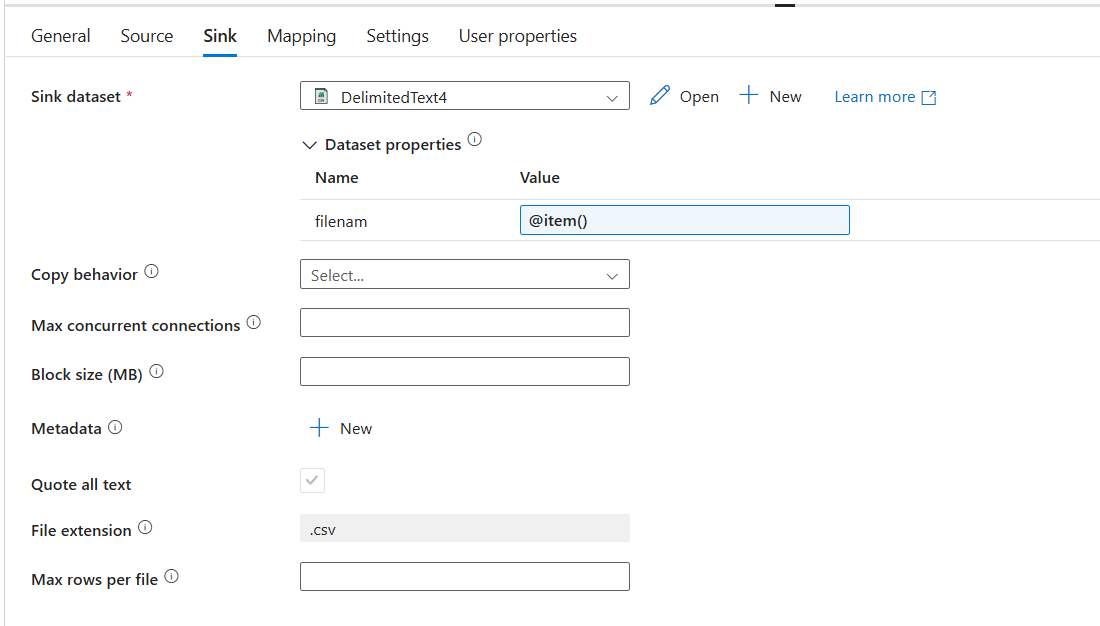
Step 6: Inside the source, choose the source dataset as Fileserver and pass the file name dynamically so that we don’t need to do the same procedure again for different files

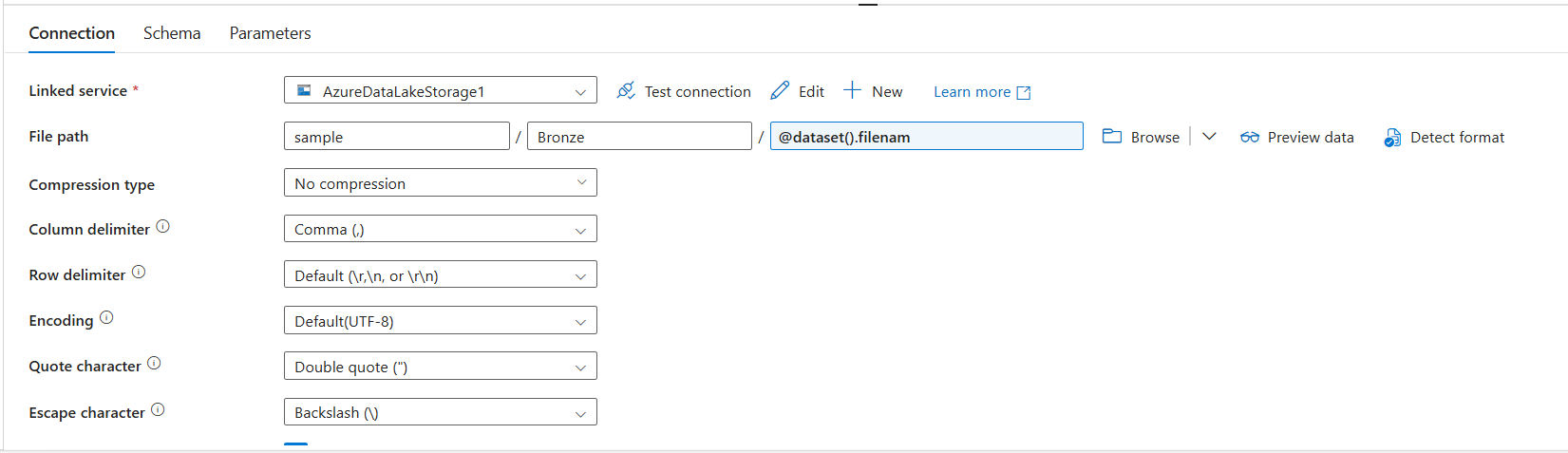


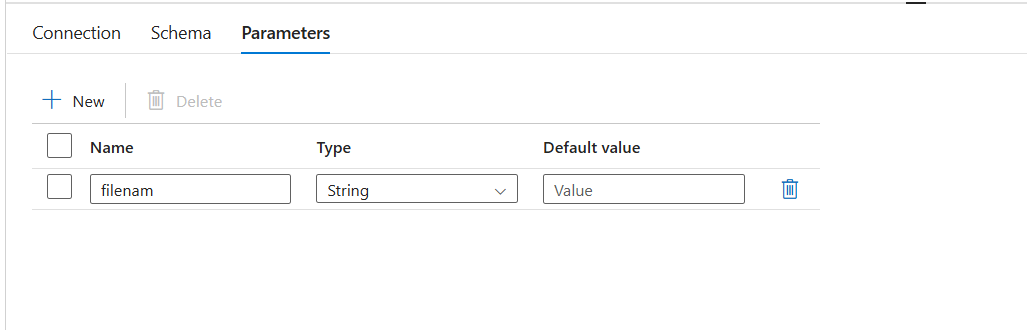
Create a parameter and pass the filename dynamically like below, this function will go to the particular folder which me mentioned and do the iteration for all the files which is present inside that



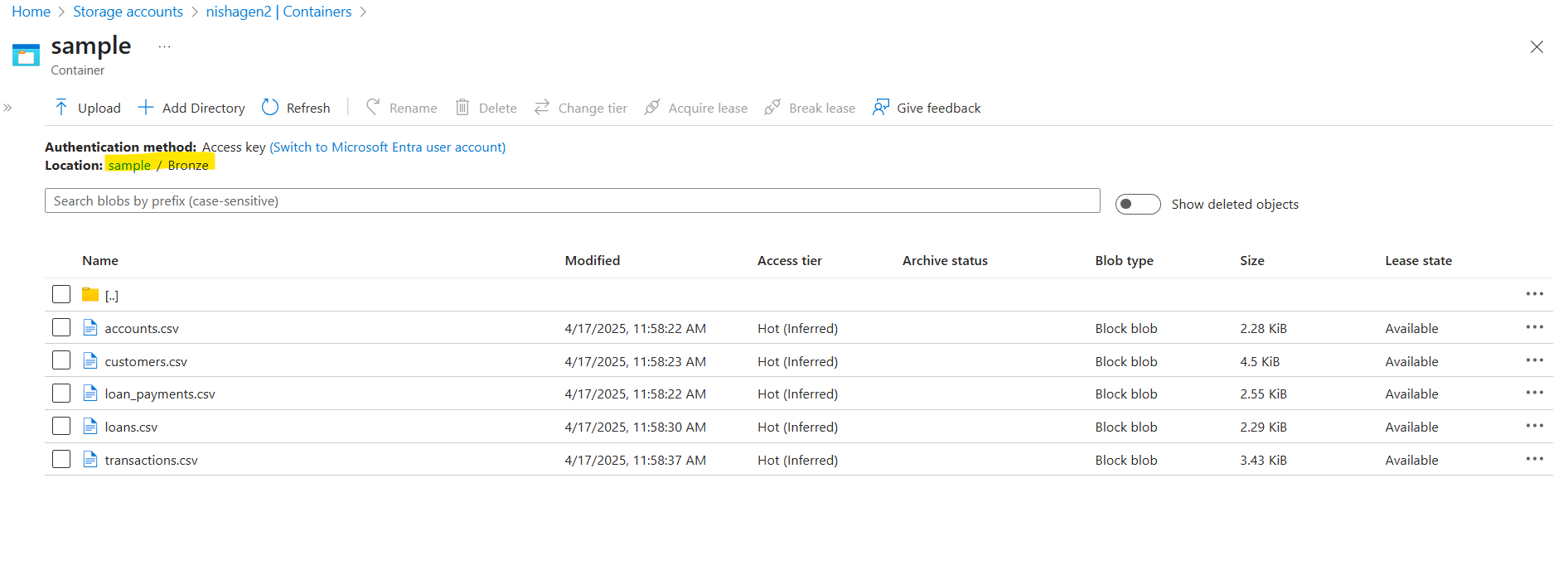
Step 7: Under the sink tab choose the sink dataset as ADLS and browse the Bronze container where we want to store the data, and give the parameter like earlier for this well so that it will run dynamically and store each file





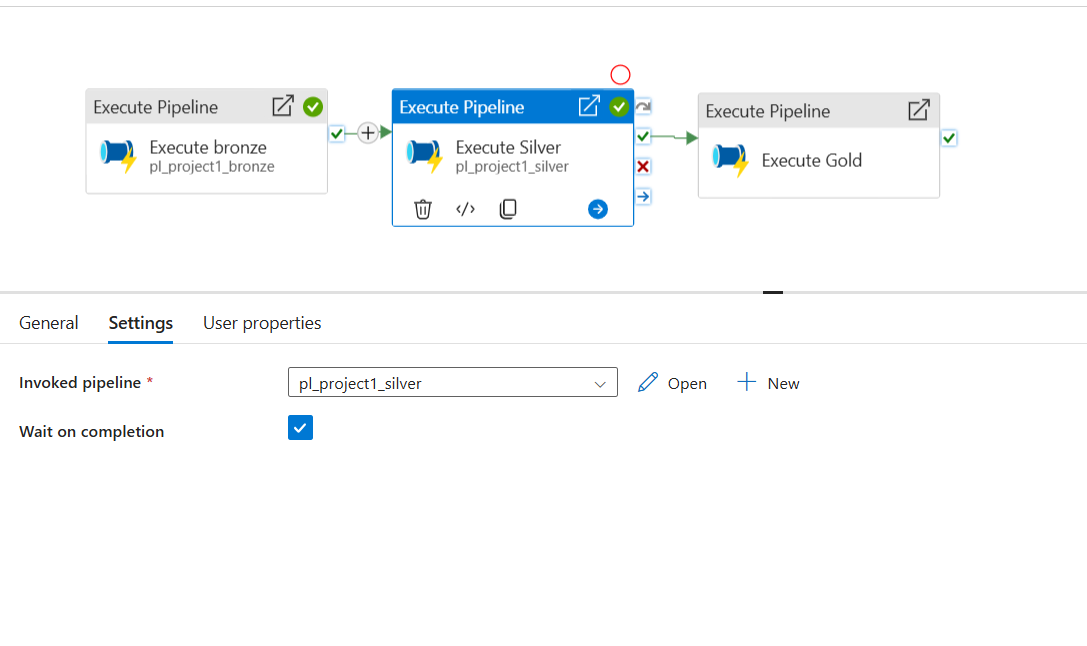


Step 8: Inside the bronze container the files will look the below

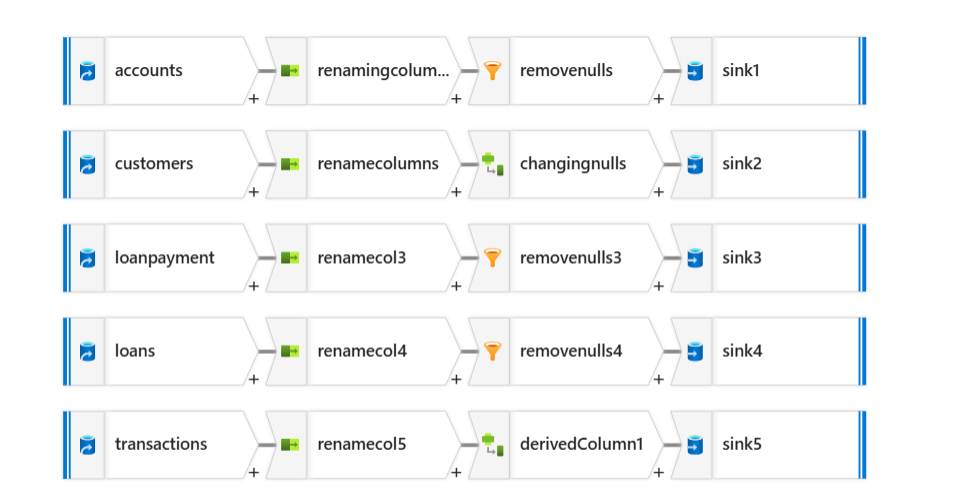


**Steps for Cleaning/transforming the data (Silver Layer)**

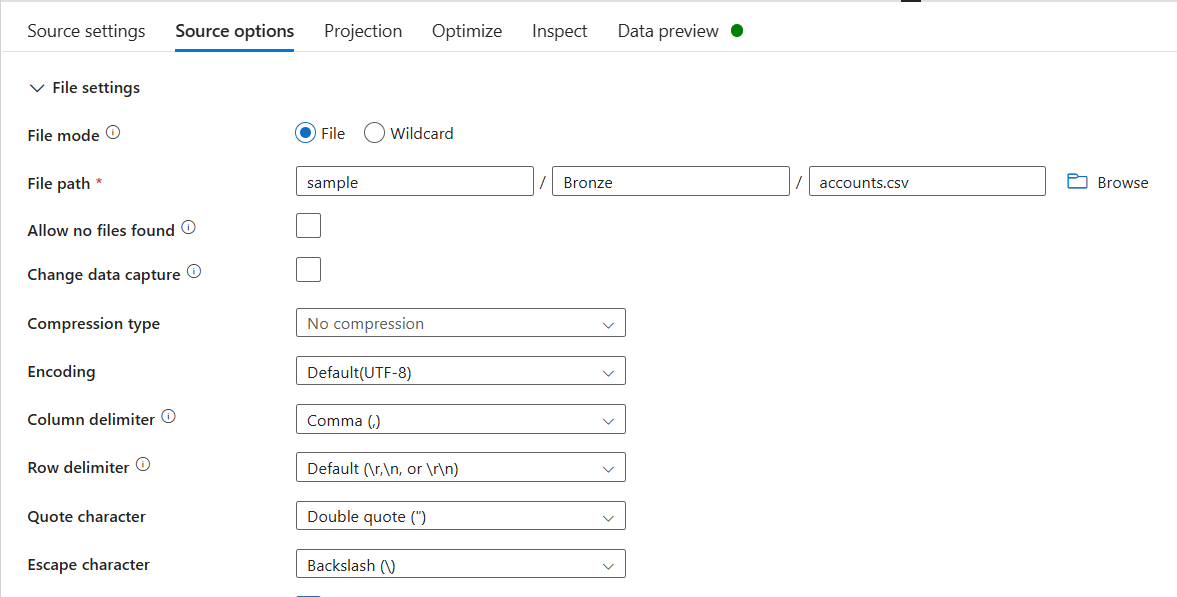
Step 1: Now, first child pipeline is done let us start working on the silver layer, call the silver pipeline inside the second execute pipeline function

****

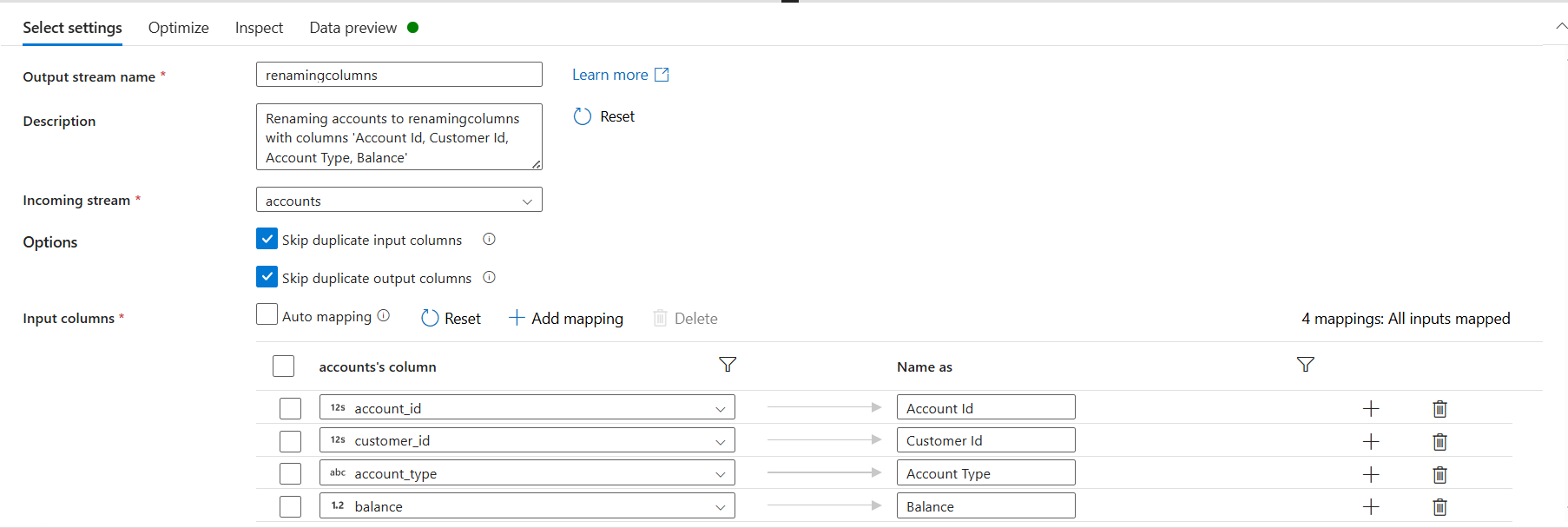
Step 2: Create five separate sources and sinks inside one data flow activity like below

****

Step 3: Inside the source options, choose the file from the bronze container

****

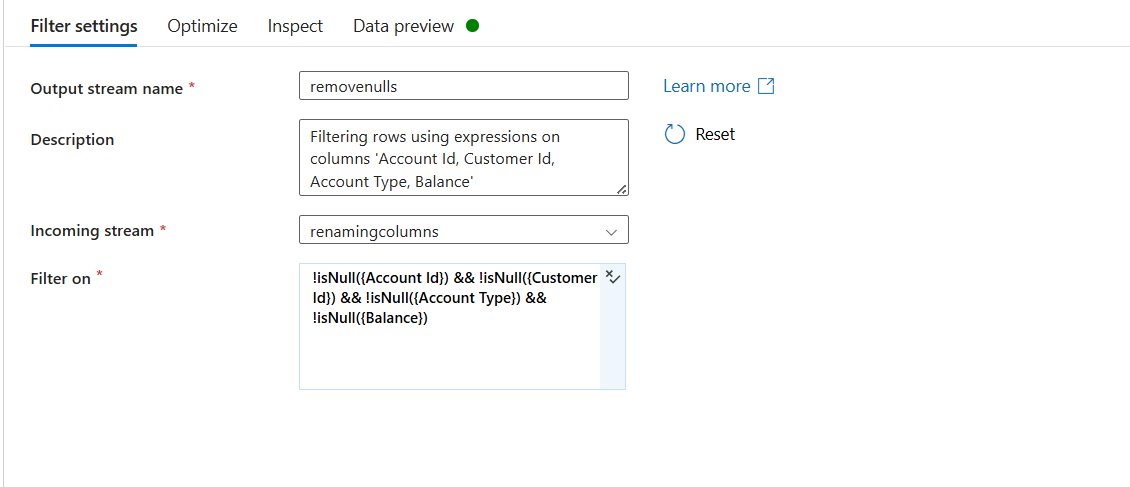
Step 4: Rename the column names using the Select setting activity after the Source

****

Step 5: Next, we need to remove nulls

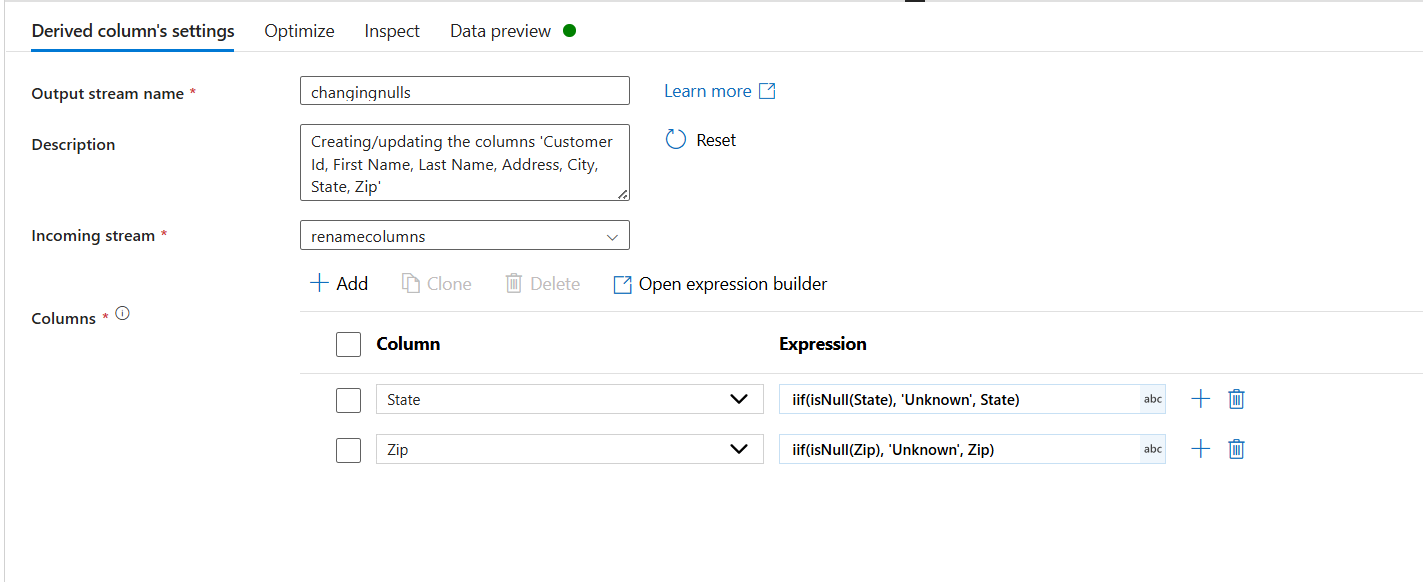
**1st Method:** Use Filter settings and remove the null values from the file by passing the below command inside Filter on condition

!isNull({Account Id}) && !isNull({Customer Id}) && !isNull({Account Type}) && !isNull({Balance})

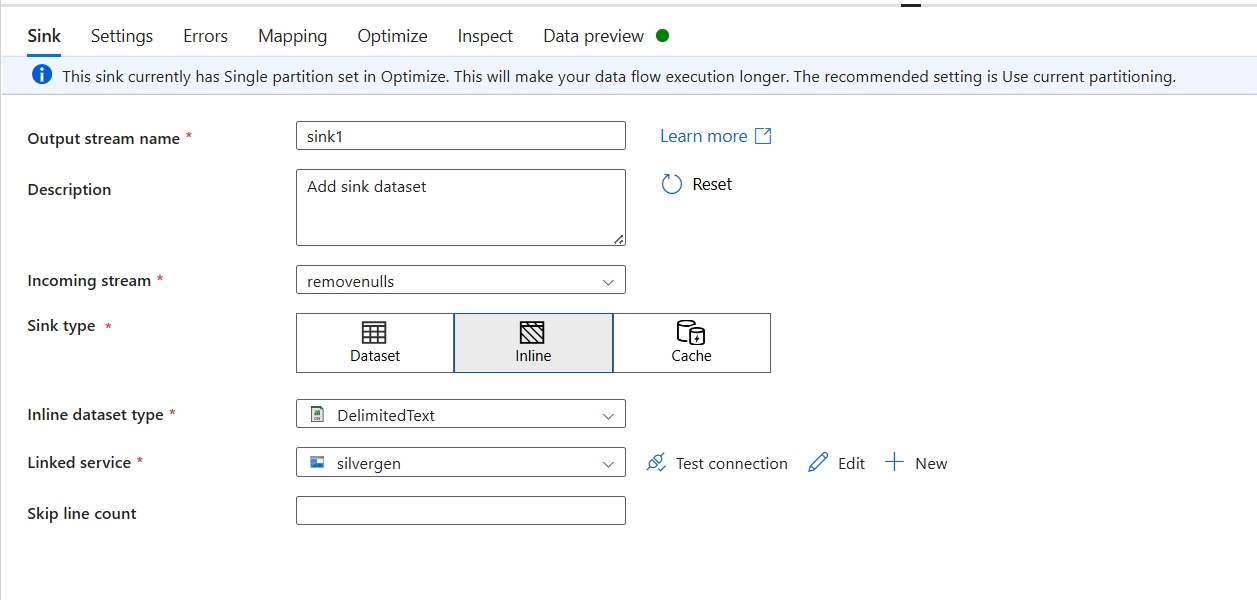
****

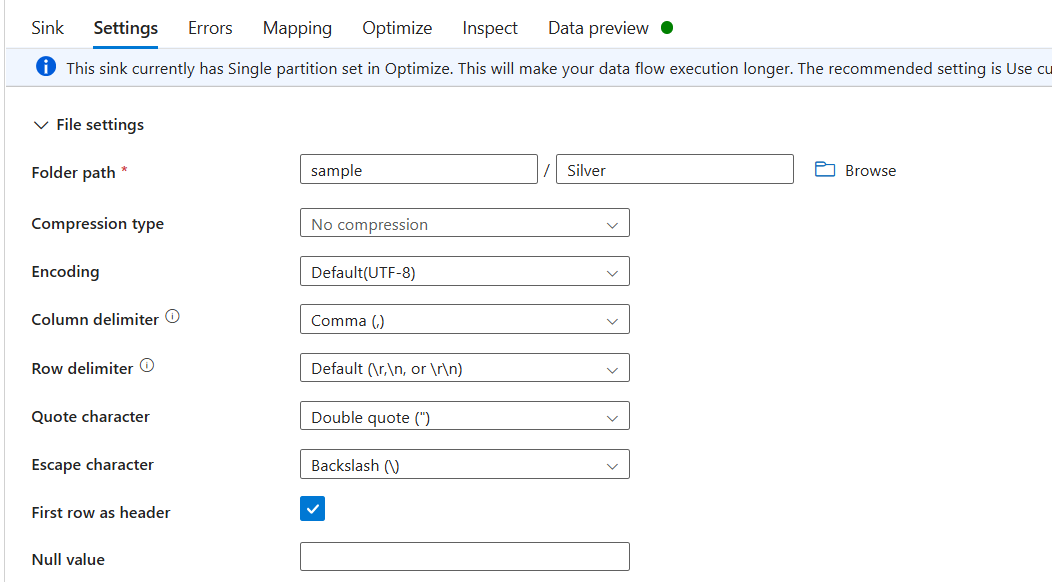
**2nd method:** Use a derived column setting and replace the null values with a dummy value i.e Unknown or NA using the below command

iif(isNull(Zip), 'Unknown', Zip)

****

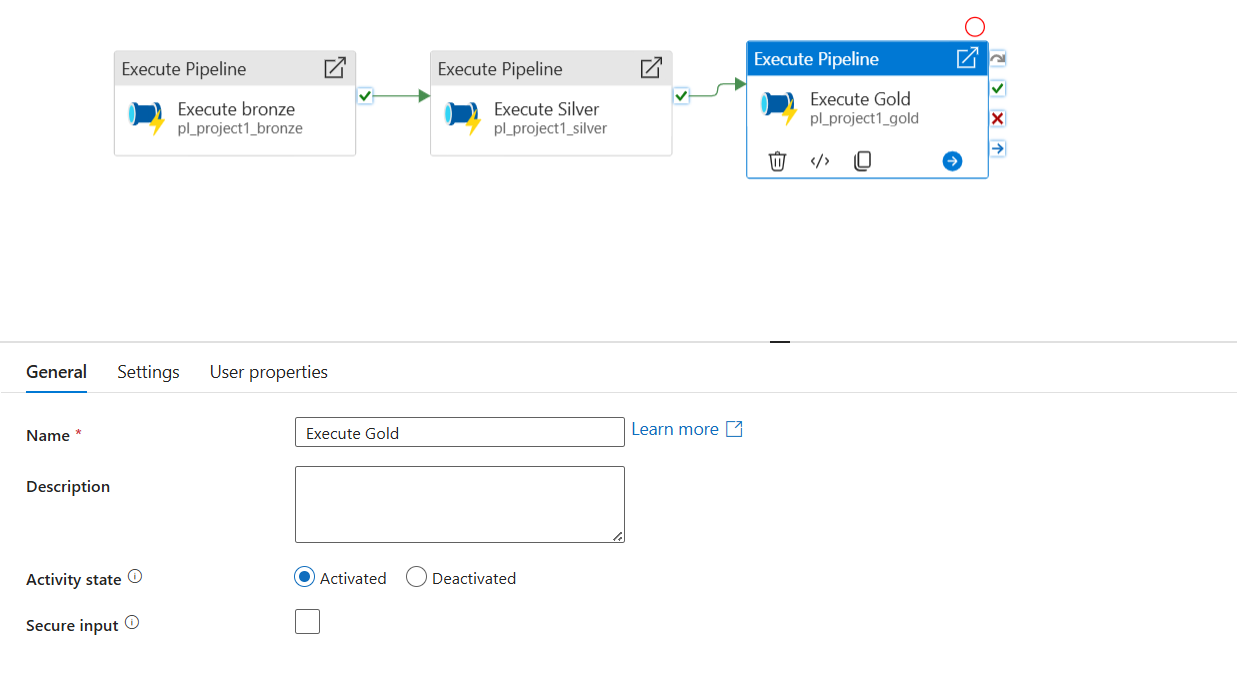
Step 6: Sink the cleansed data into an ADLS Gen 2 Silver container

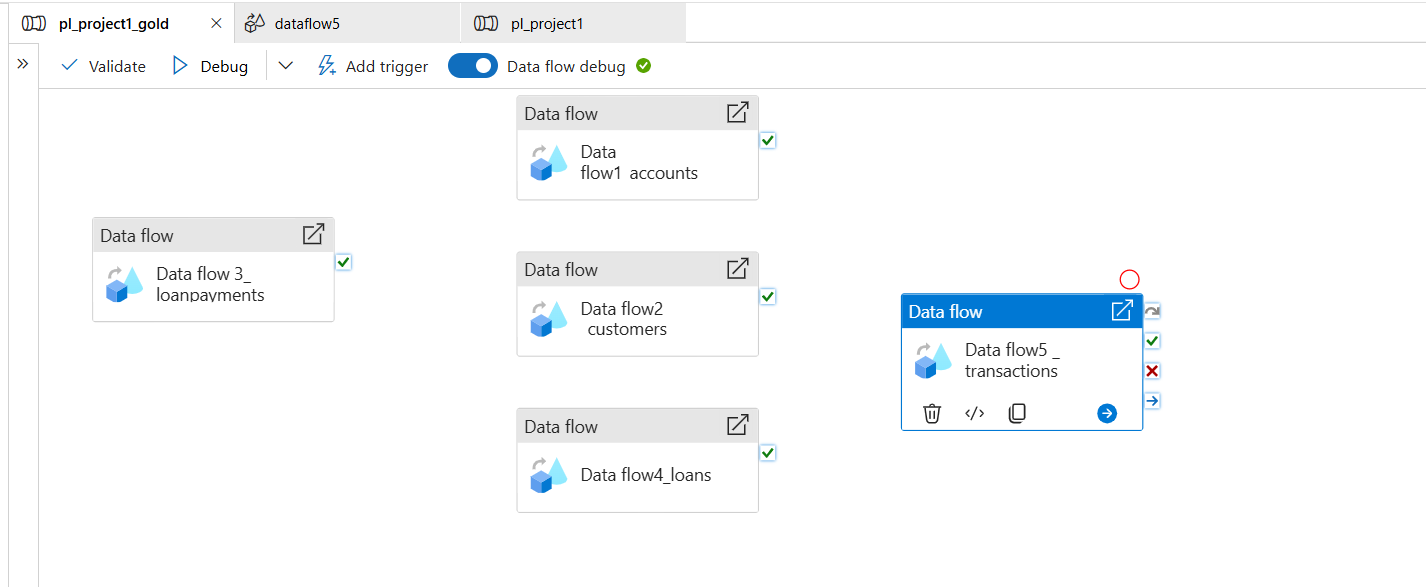
****

****

Step 7: Repeat the same set of process for all the four more files

**Steps for SCD1/ SCD2 transformation(Gold Layer)**

****

****

**SCD 1 (Accounts, Customers,Loans)**

**Database Table Creation Script**

* Creating a table in SQL DB

CREATE TABLE accounts\_SCDTYPE1

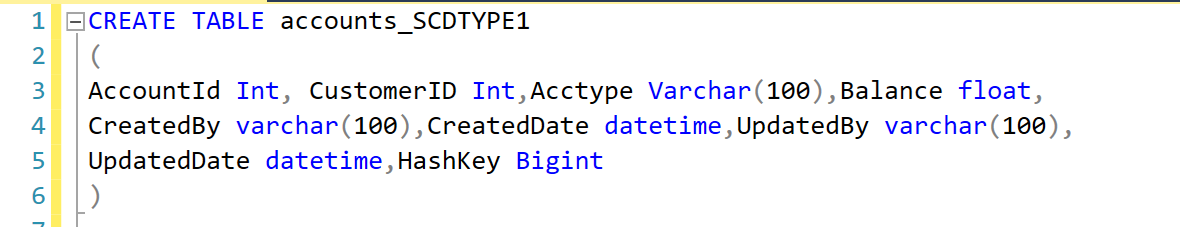
(

AccountId Int, CustomerID Int,Acctype Varchar(100),Balance float,

CreatedBy varchar(100),CreatedDate datetime,UpdatedBy varchar(100),

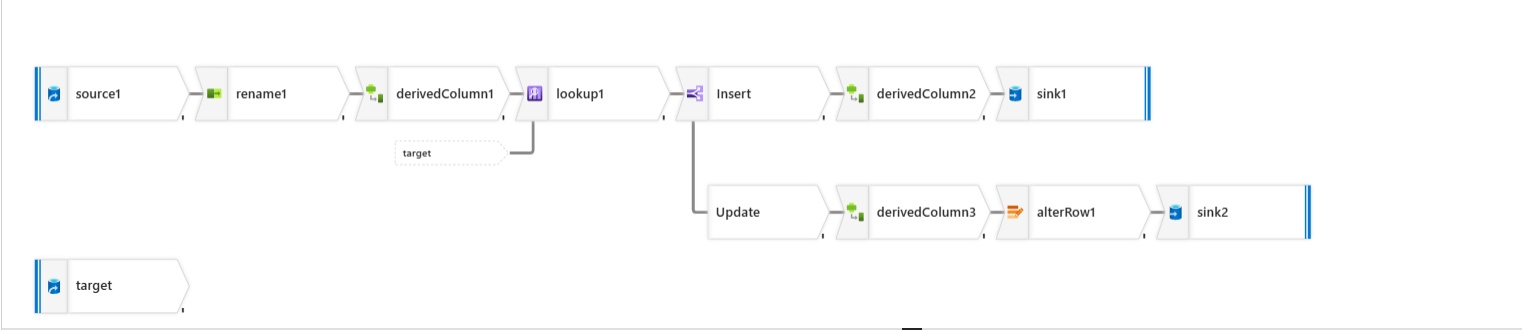
UpdatedDate datetime,HashKey Bigint

)

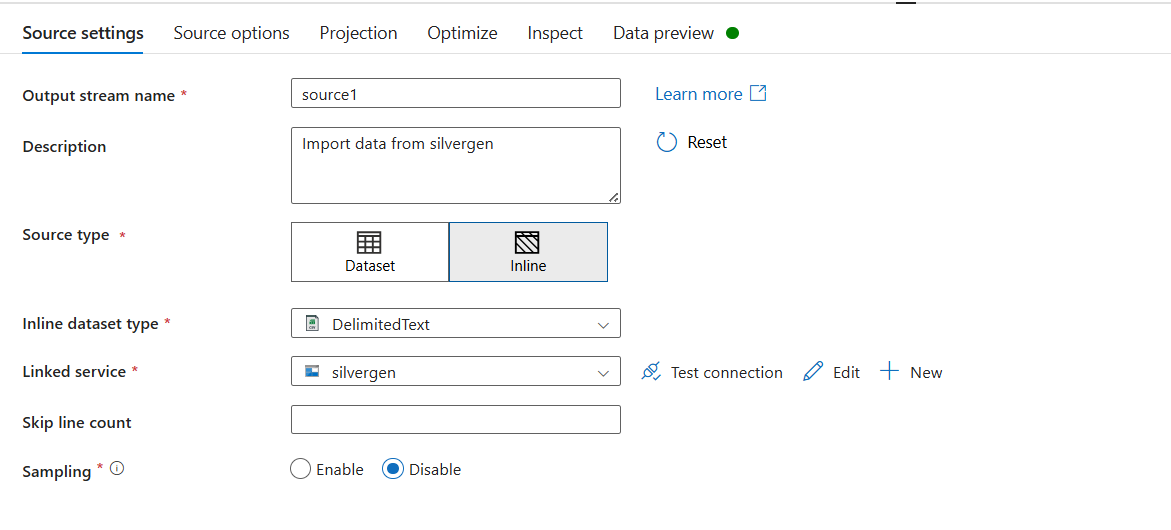


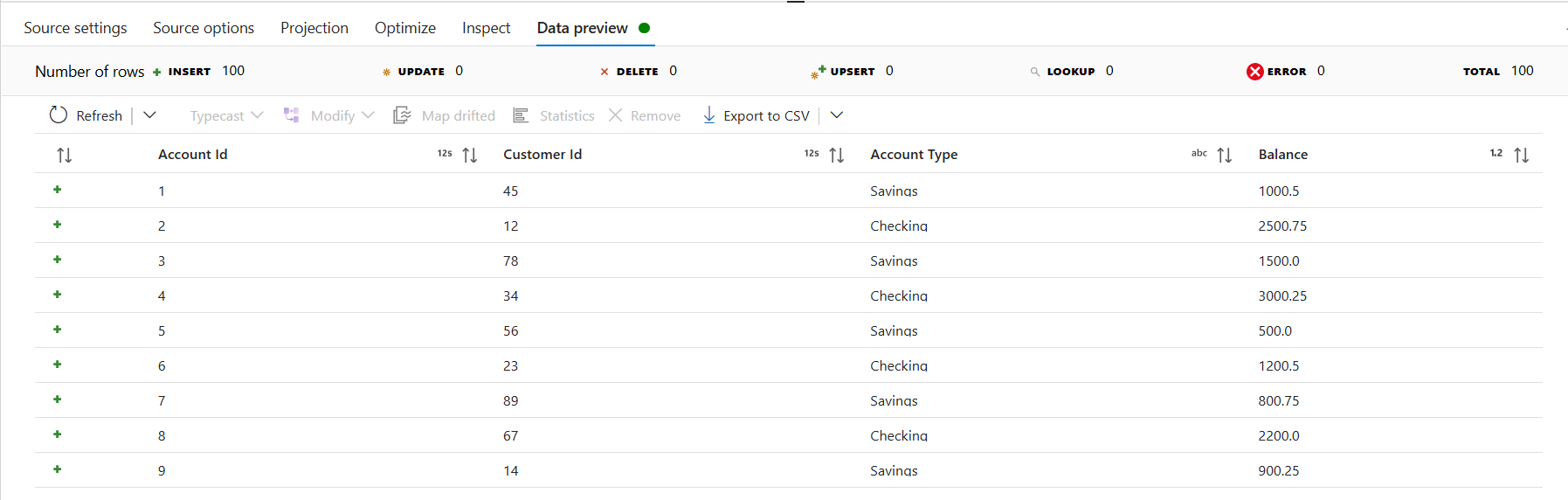
**Pipeline Creation and Optimization**

\* Practiced creating pipelines in Azure Synapse Analytics, focusing on:

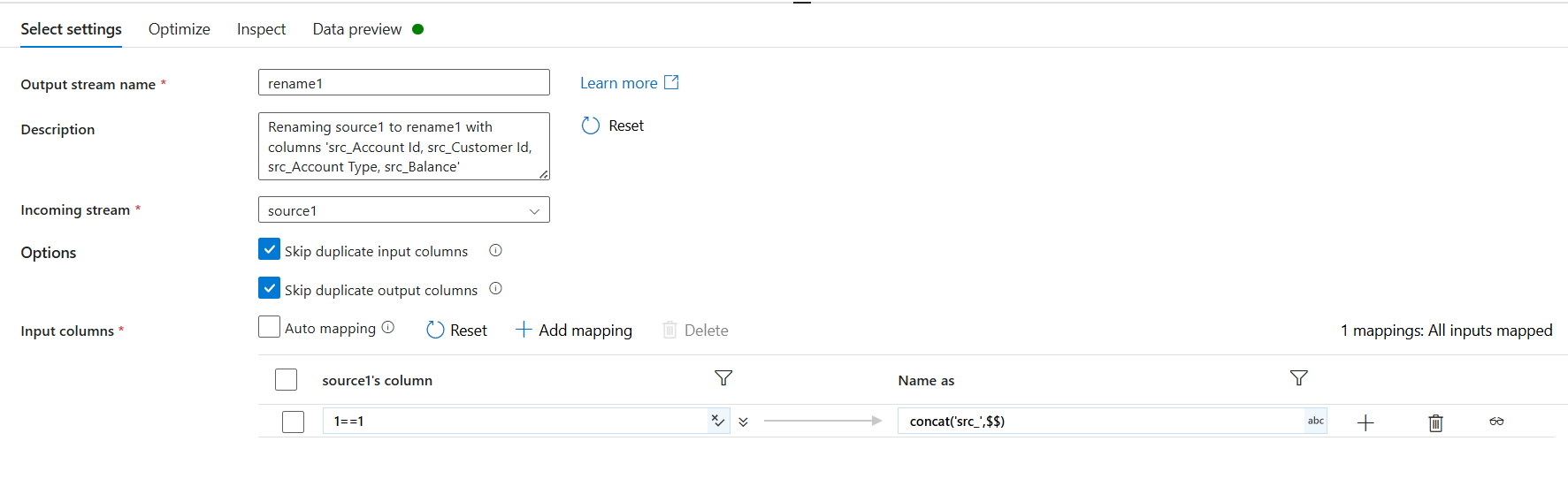


\* Source configuration, column renaming, and hash key generation.

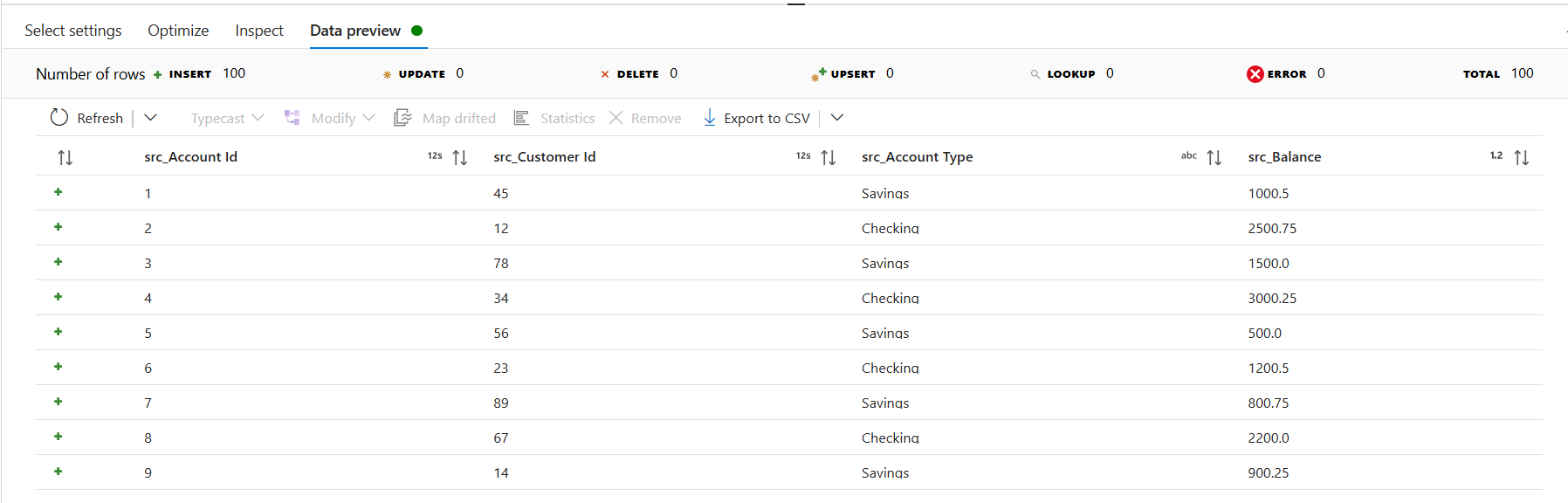
* In Source we are choosing ADLS Gen 2 as inline data type, and in file path we are selecting the CSV file which has data consisting of details

****

* Secondly, we are using a select settings to rename the columns

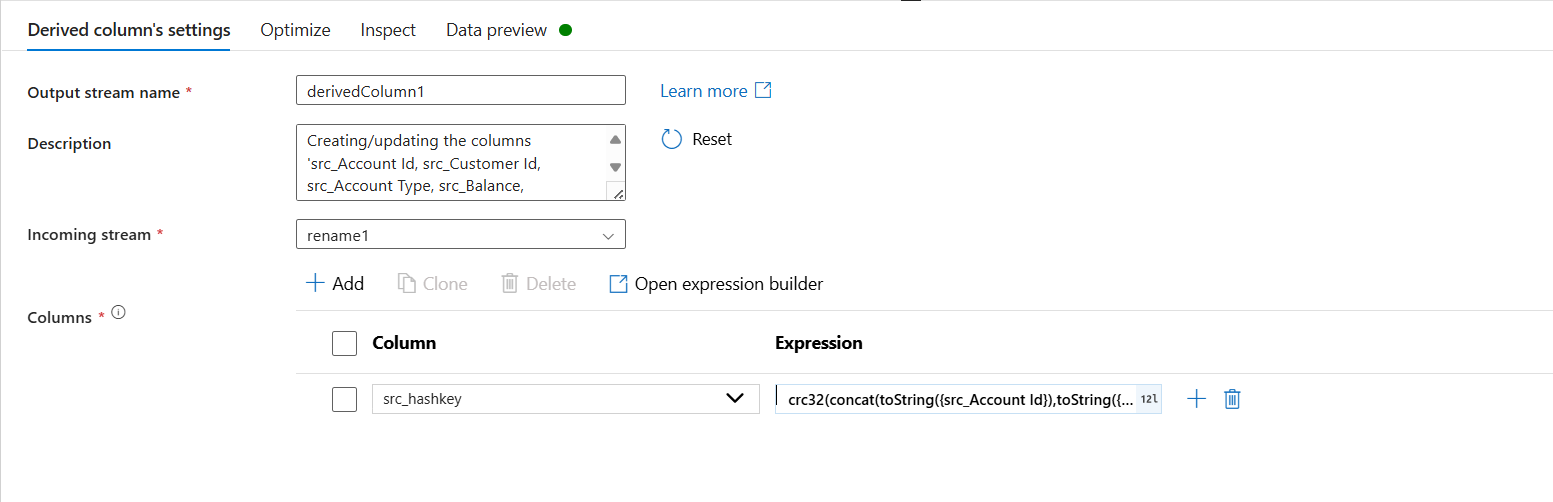


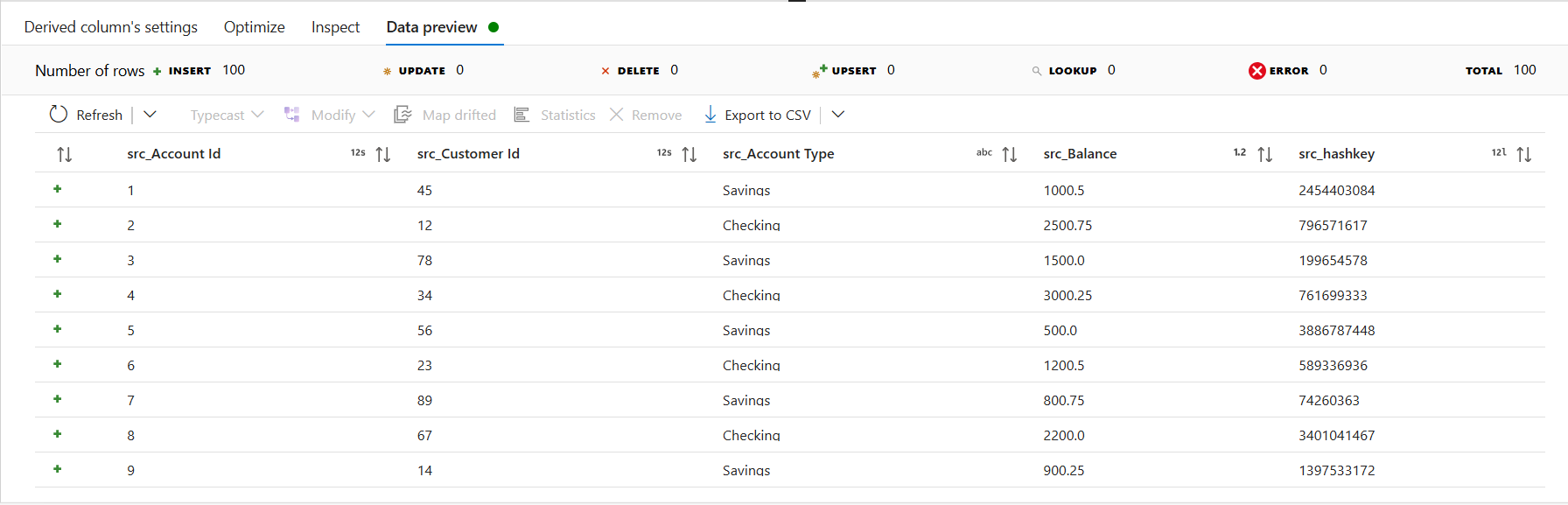
* The renamed column will look like below



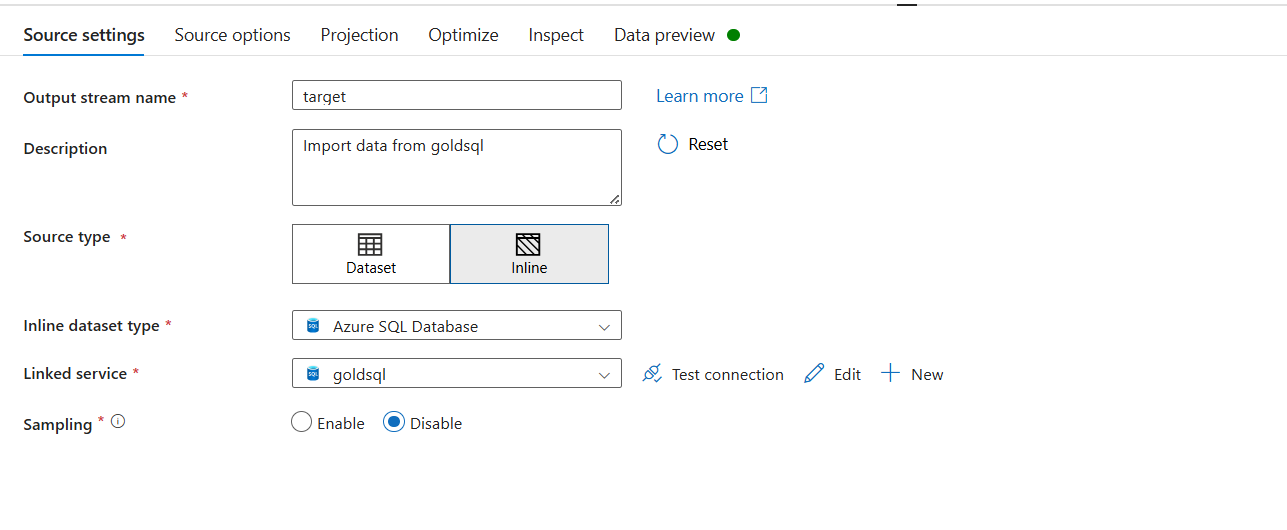
* Next, a derived column setting is used to generate unique hashkey for all the data which are in the CSV file. The hashkey will be updated if any changes is made to the existing data
* The below expression is used to generate the hashkey

crc32(concat(toString({src\_Account Id}),toString({src\_Customer Id}),{src\_Account Type},toString(src\_Balance)))



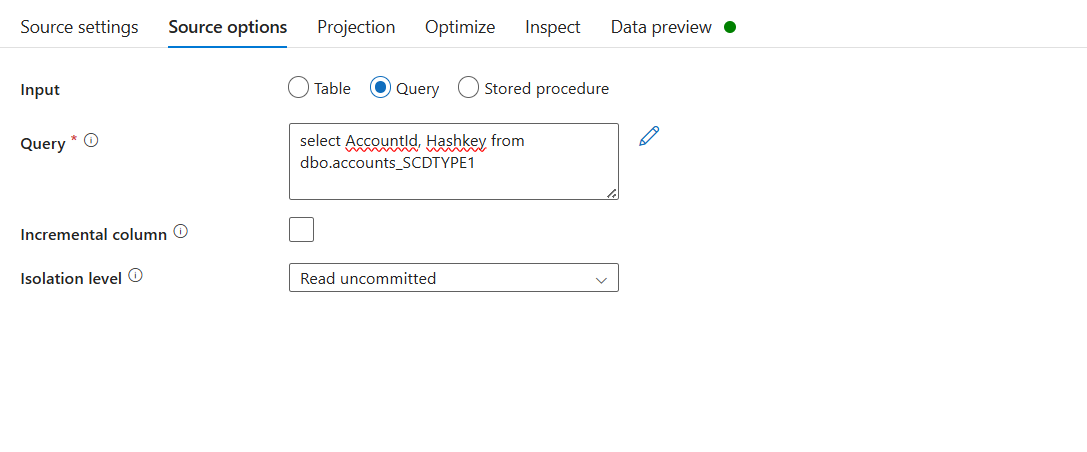


* Now we are using a source settings to take two columns from Database to compare what values are there in the database and CSV file

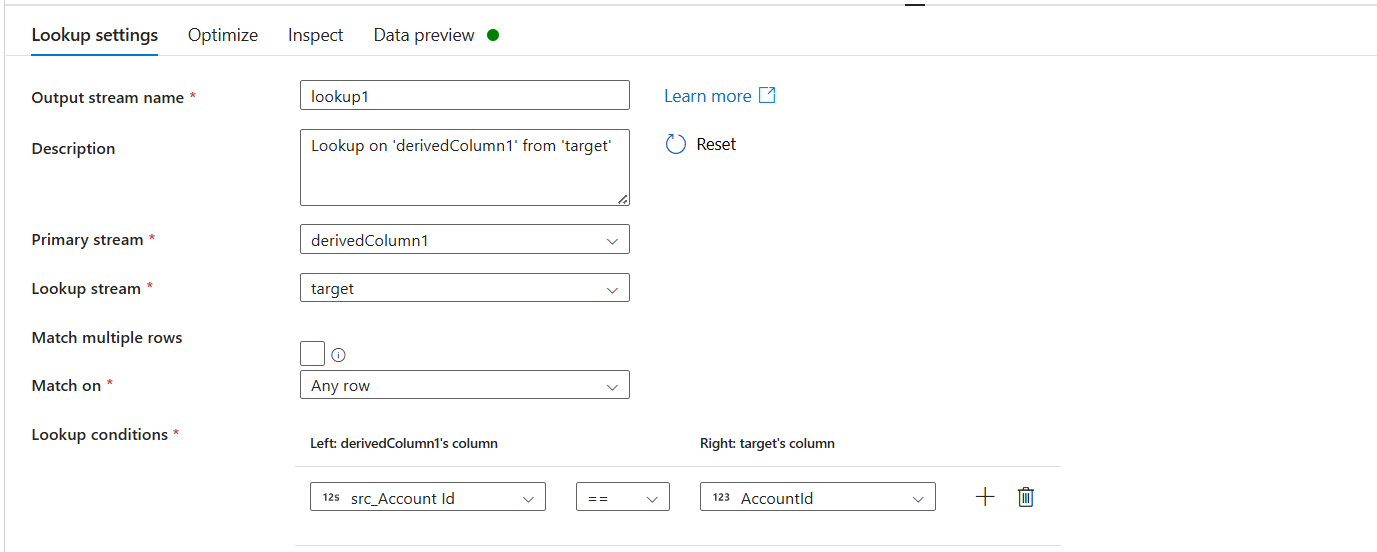


* Under source operation we need to select query and write the below query

select AccountId, Hashkey from dbo.accounts\_SCDTYPE1



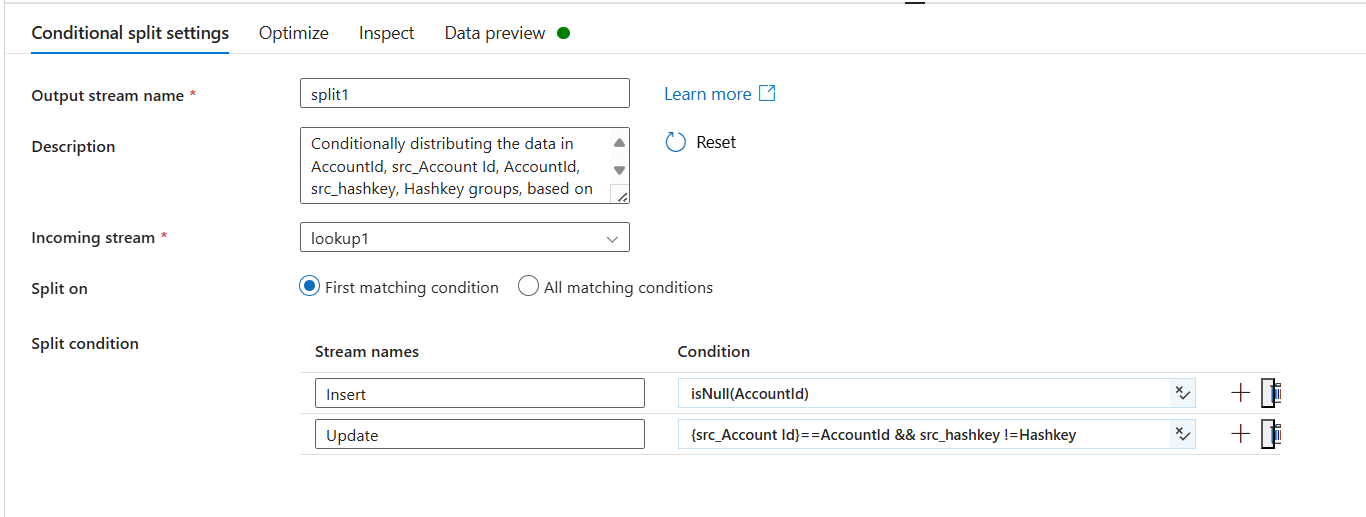
* Following to that a lookup transformation is used in which the primary stream will point to Hashkey and lookup stream should connect to Target



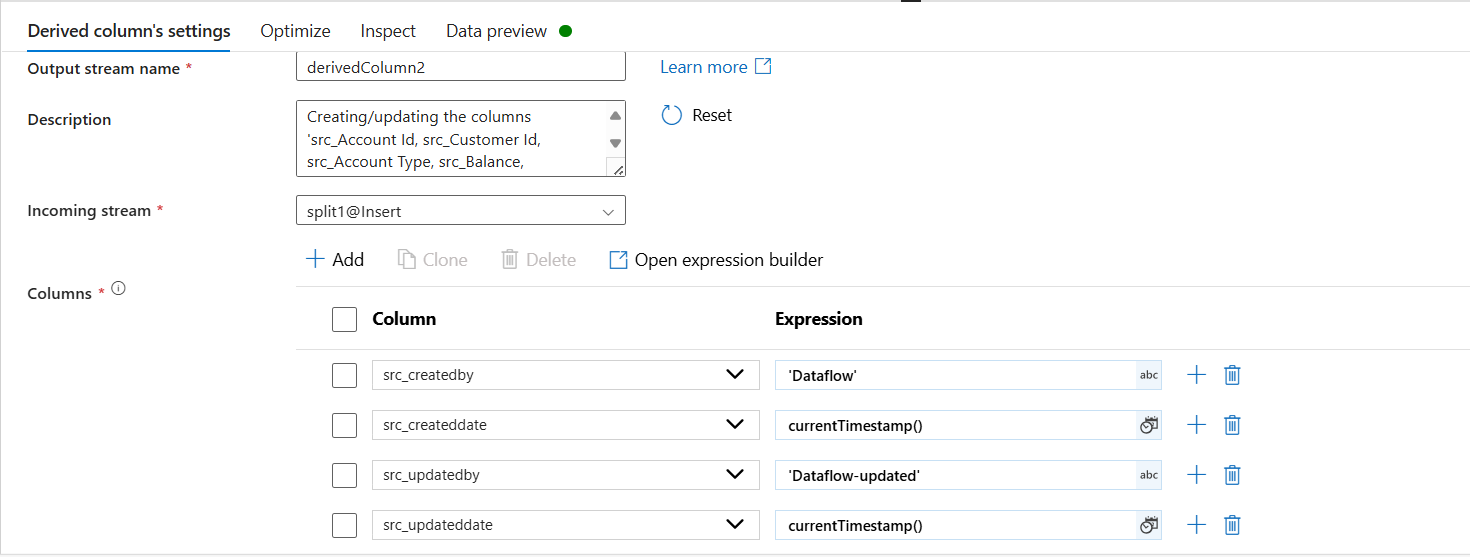
* A conditional split is used to split Insert and Update details, the below expression is used

Insert – isNull(AccountId)

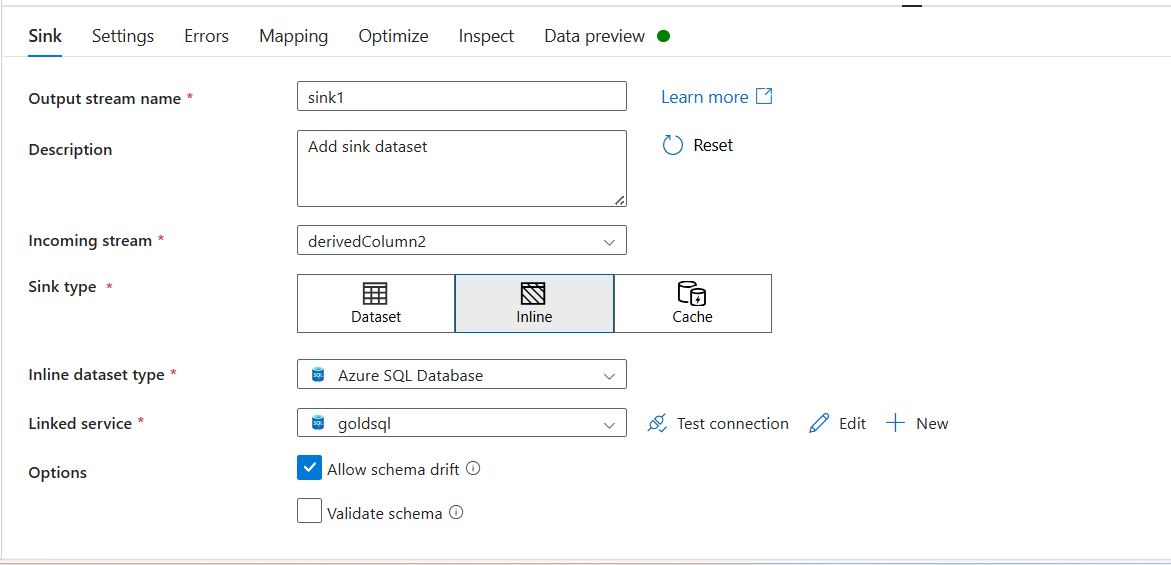
Update -- {src\_Account Id}==AccountId && src\_hashkey !=Hashkey



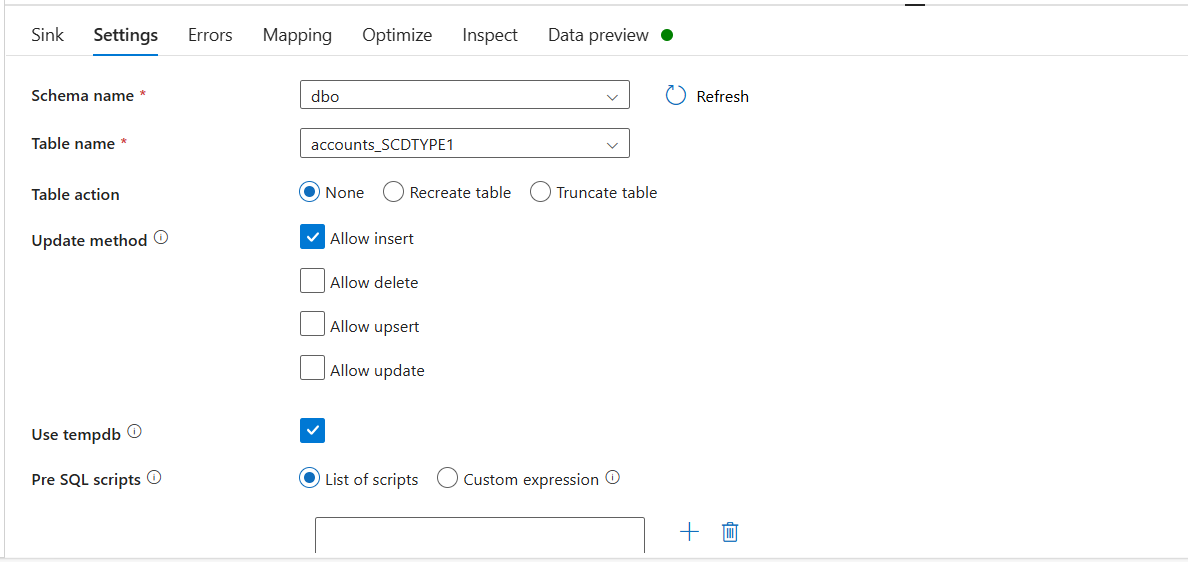
* Next, a audit column has been added to Insert to add createdby, createddate, updatedby, updateddate



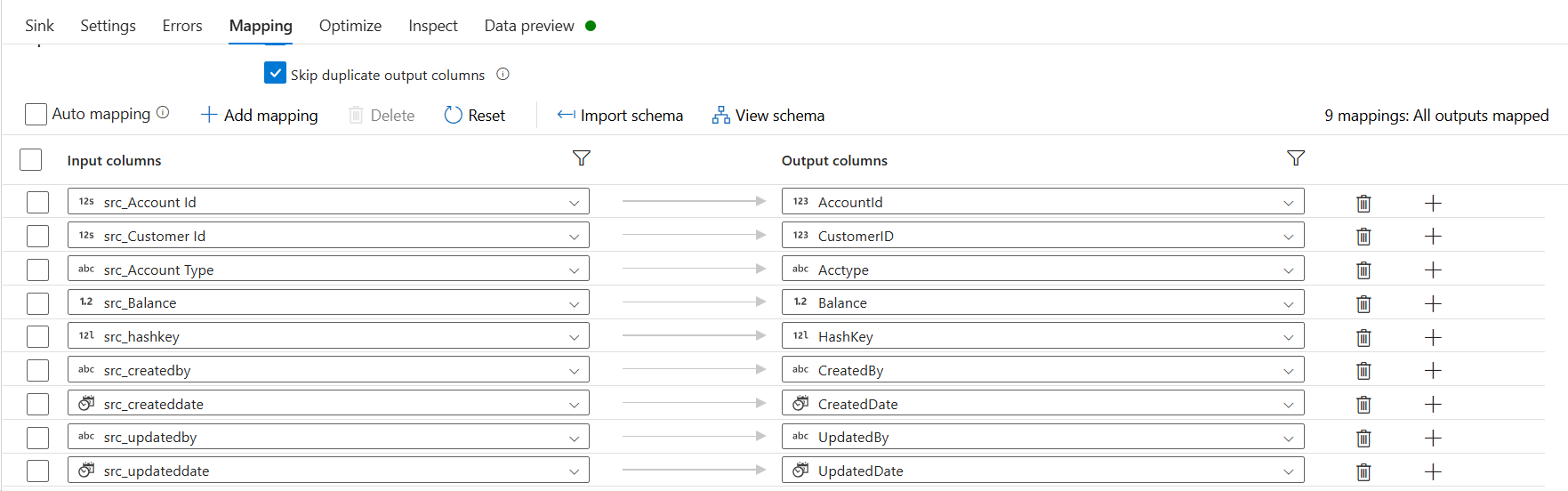
* Use a sink at end to push all the data to the database



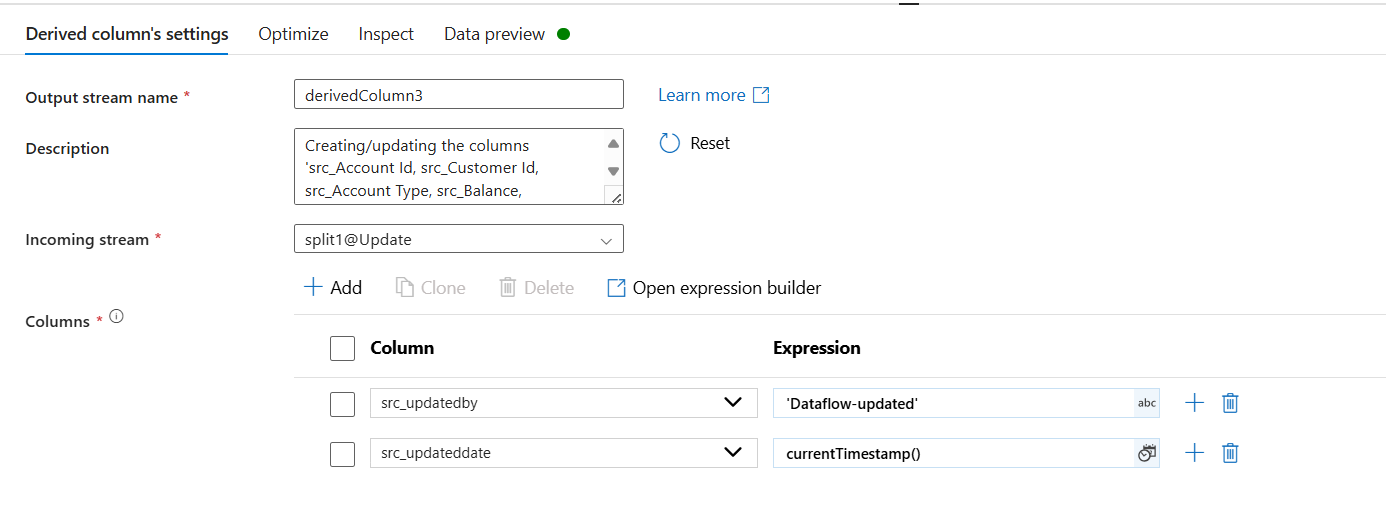
* In source setting we need to select allow insert in update method



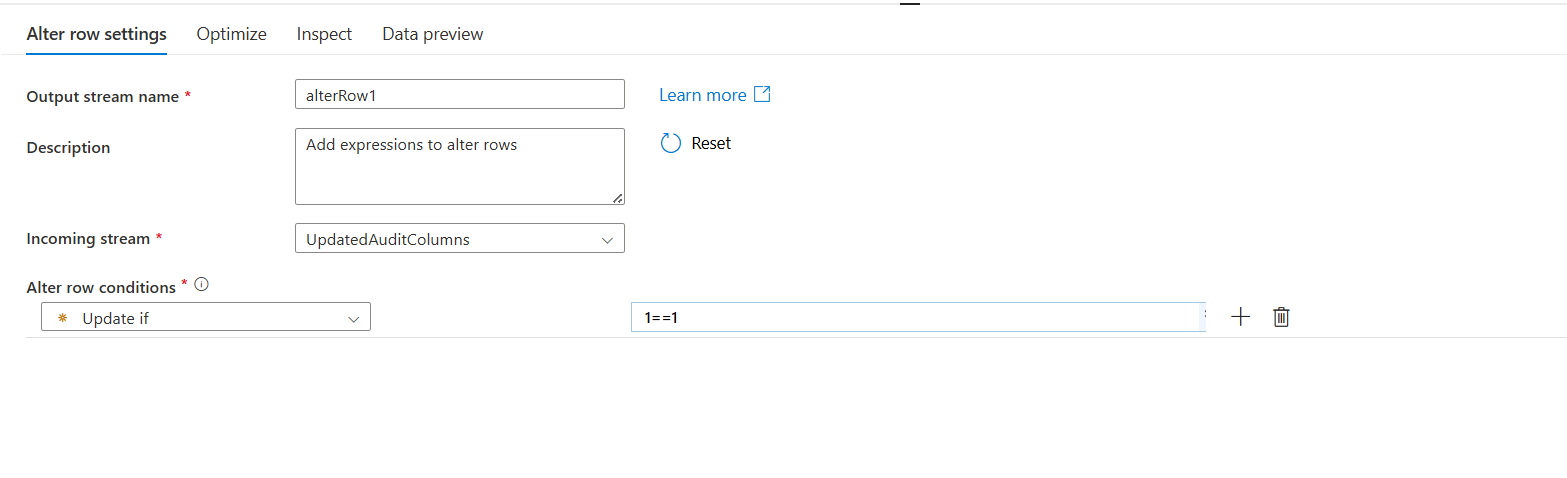
* In mapping we need to map all the columns with their respective columns



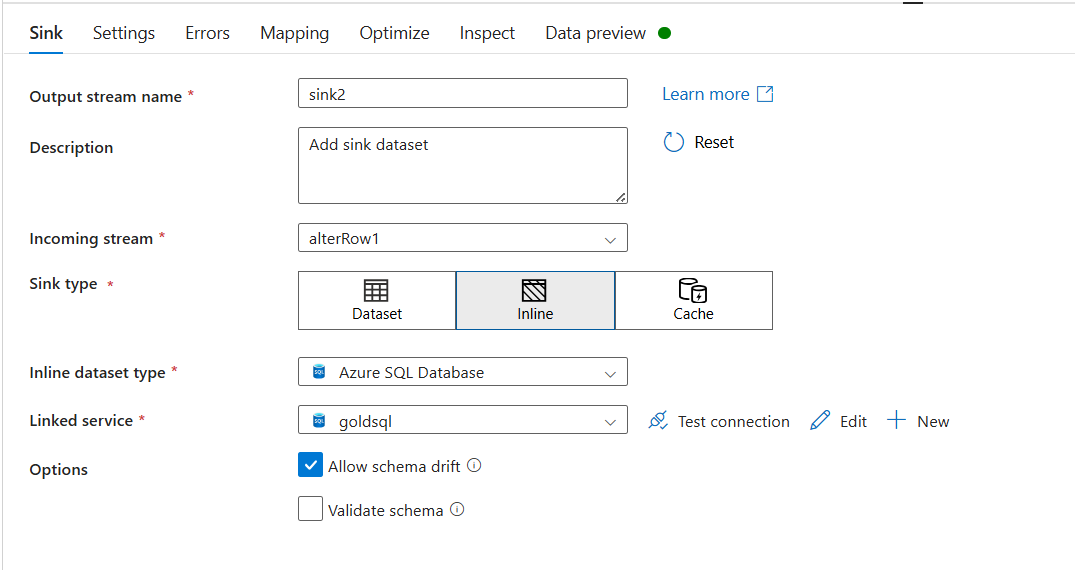
* Next, a audit column has been added to Update to add updatedby, updateddate
* In updatedby we need to give “dataflow” and in updated date we need to give currentTimestamp()



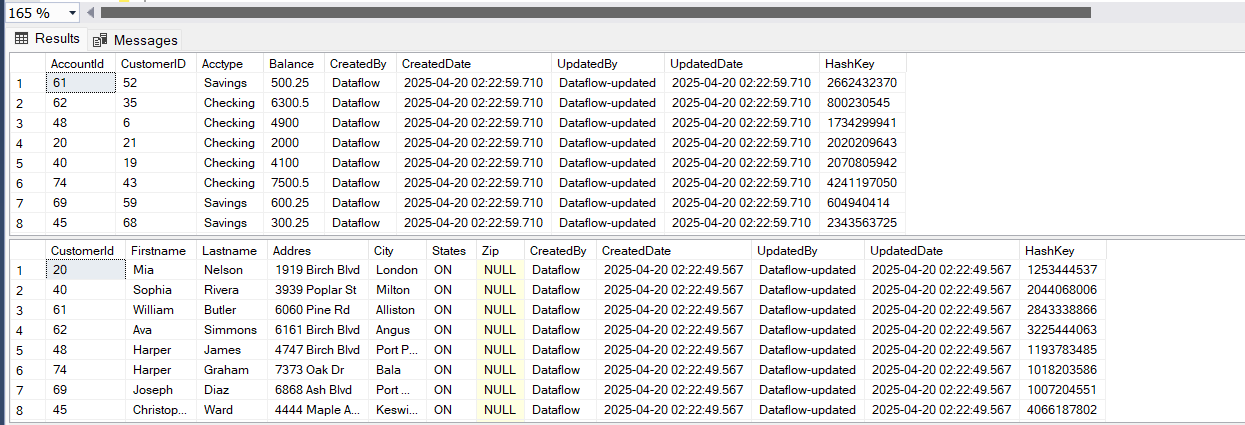
* An alter row has been used to pass a condition



* Use a sink at end to push all the data to the database



* In source setting we need to select allow insert in update method
* In mapping we need to map all the columns with their respective columns
* The final output will look like below in Azure SQL Database



**SCD 2(Loan payments, Transactions)**

* Create a table in SQL Database

CREATE TABLE LoanPayments\_SCDTYPE2

(

PaymentId Int, LoanId Int,PaymentDate datetime,Paymentamt Int,

CreatedBy varchar(100),CreatedDate datetime,UpdatedBy varchar(100),

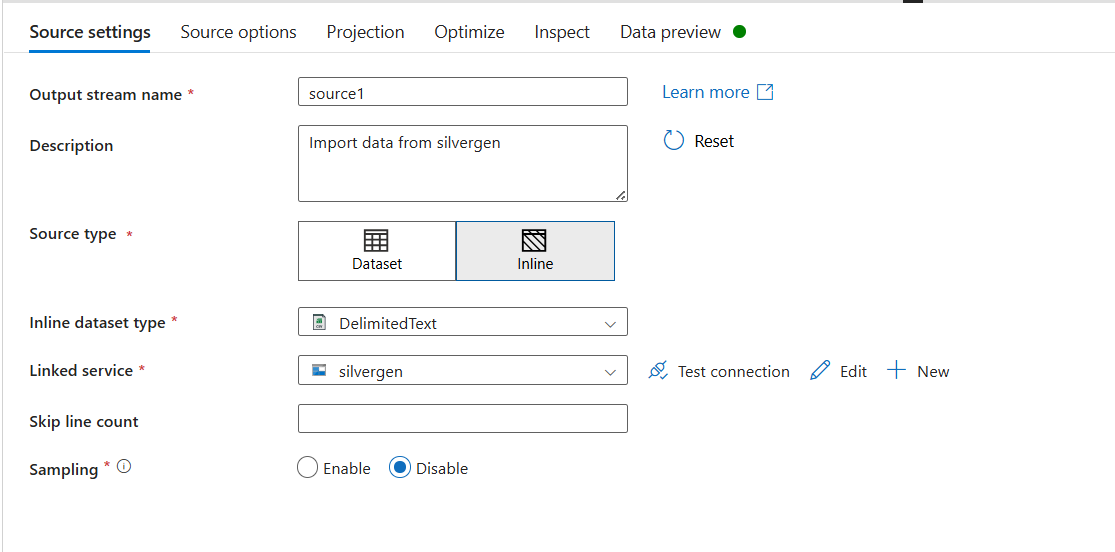
UpdatedDate datetime,HashKey Bigint, Isactive INT

)

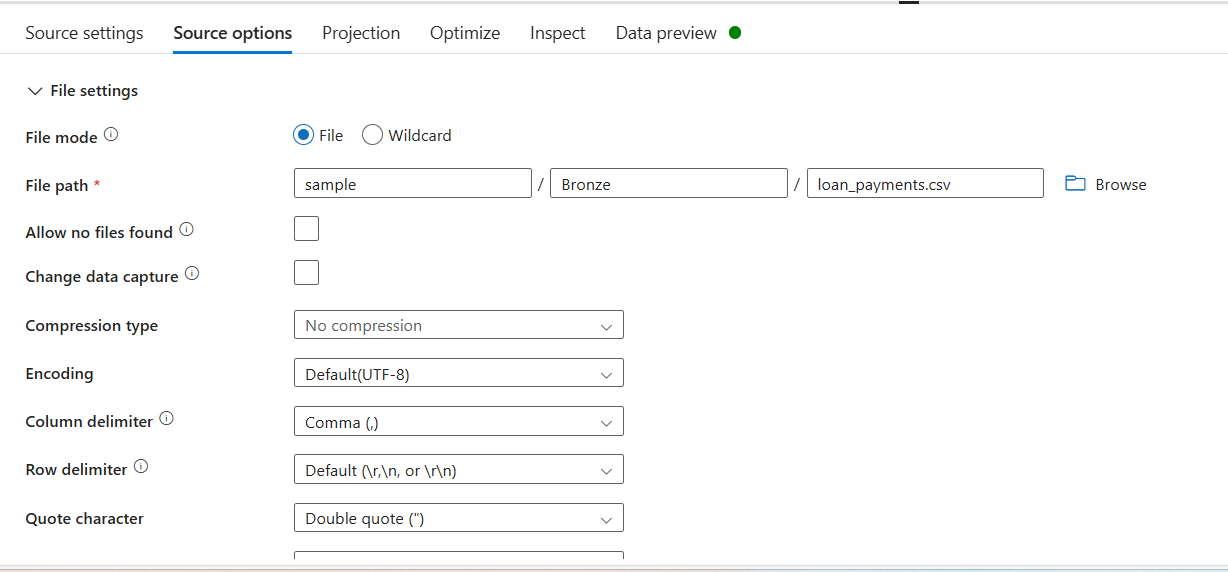
* Create a new pipeline for SCD Type 2 activity in azure synapse analytics, open a dataflow

*Open the data flow and start creating the activity*

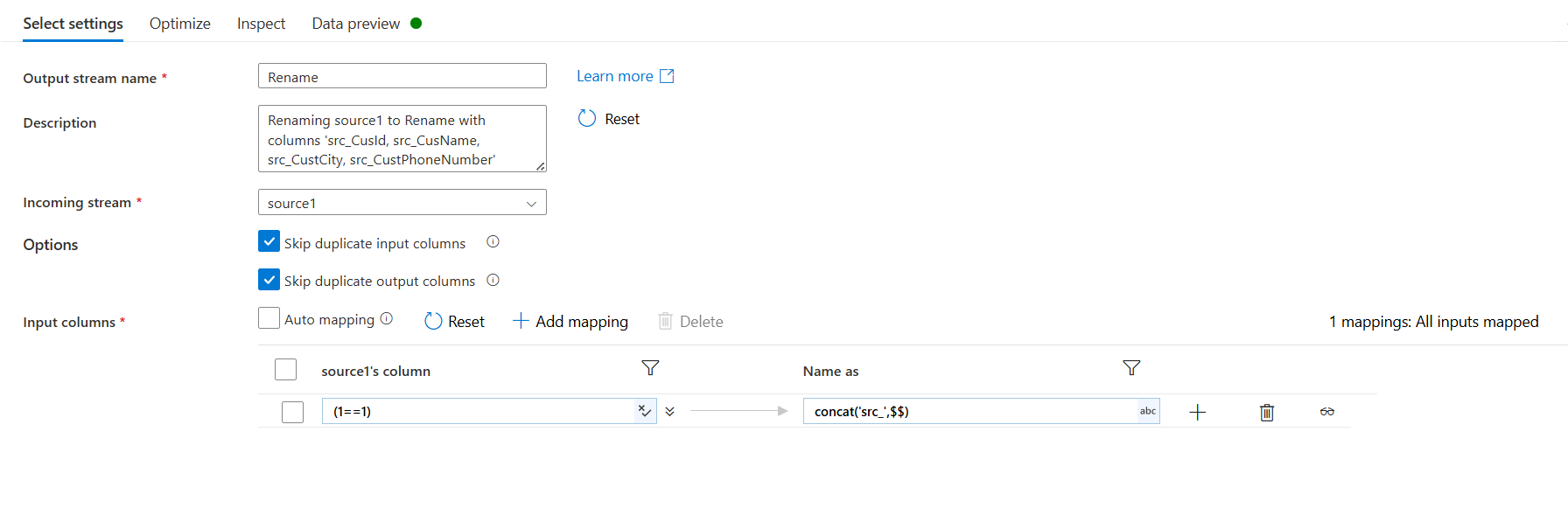
* Create a Source inside the dataflow activity, **the source type** should be **Inline** and **Inline data type** should be **Delimited Text**



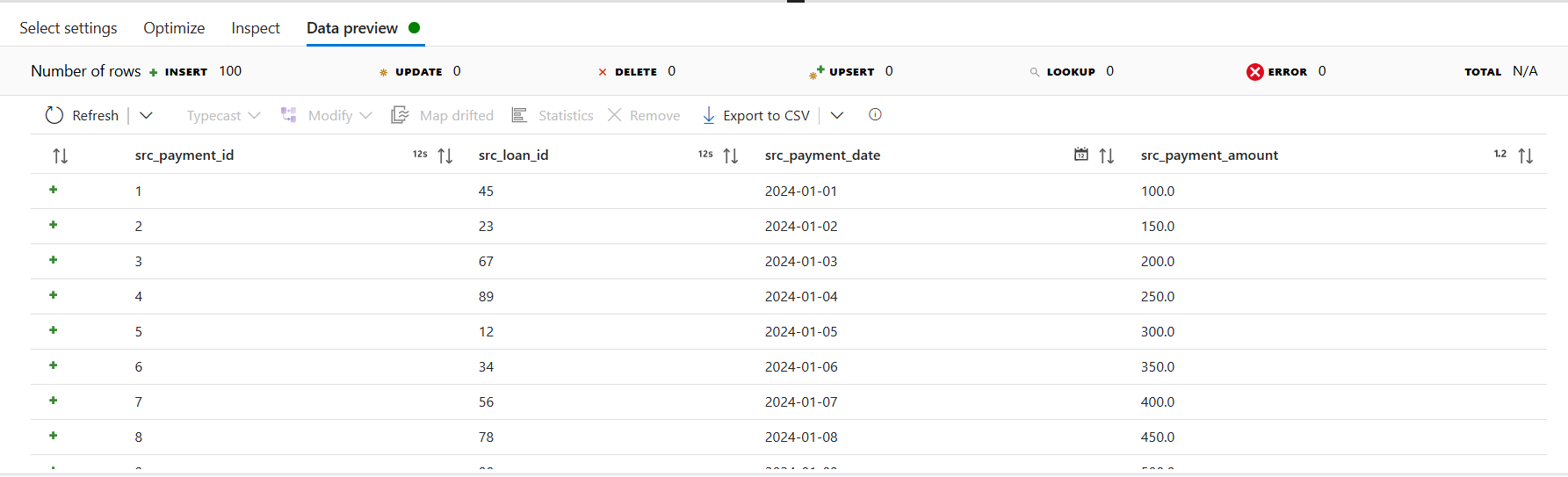
* In the source options choose the file and **select** the **first row as header**



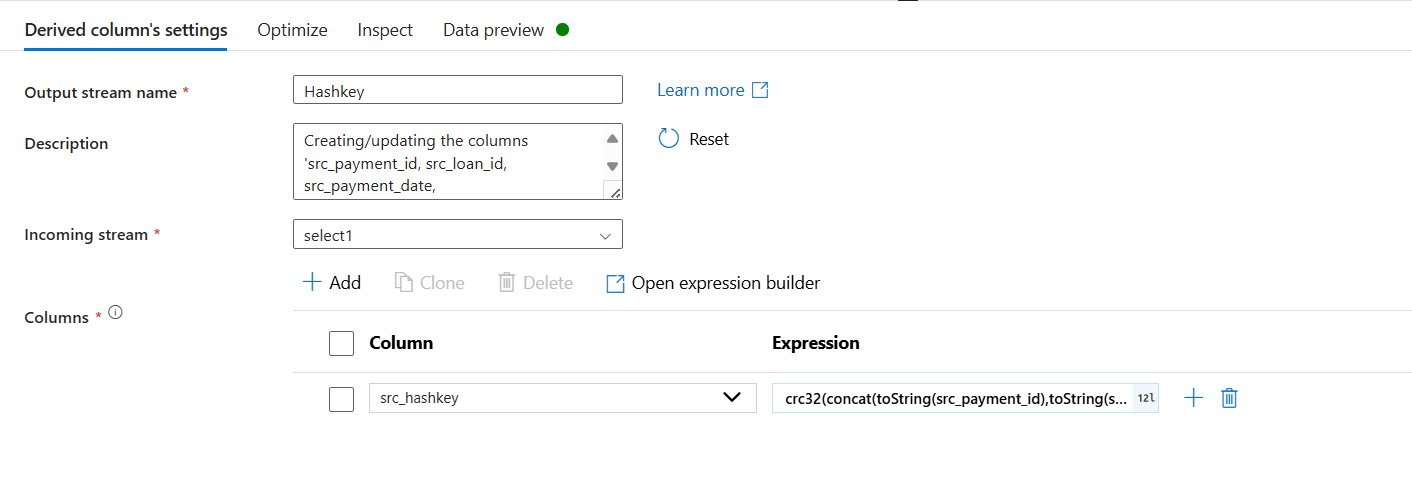
* Add select settings and rename it “Rename”, to rename the columns first delete all the columns and open **Rule based mapping** in **source** field give **(1==1)** and in **expression** field **concat(‘src\_’,$$)**

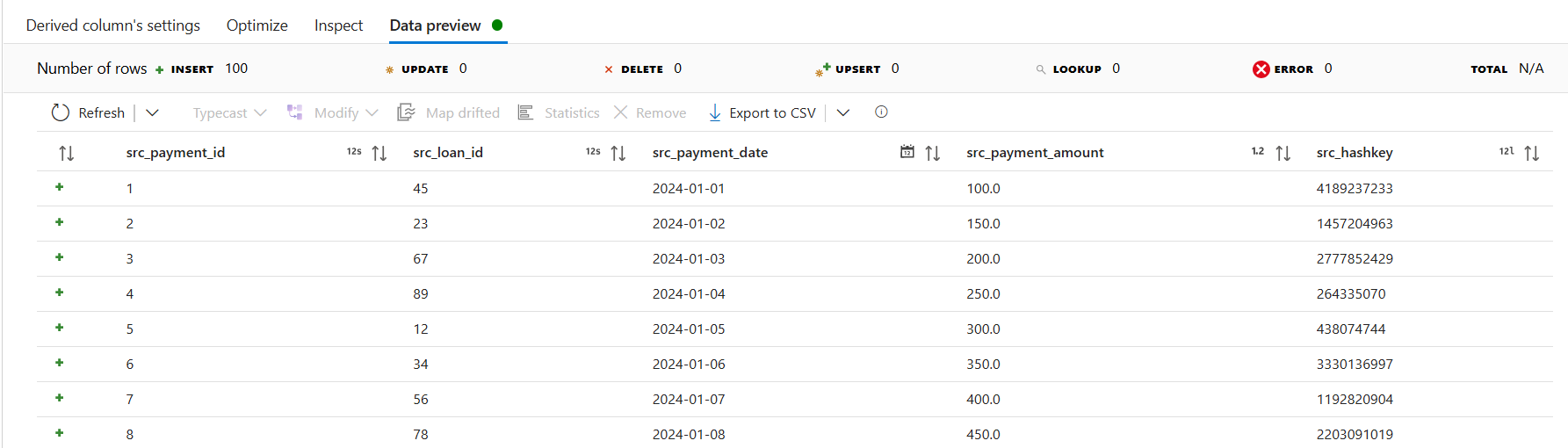


* And preview the data to check whether the column name has been changed or not

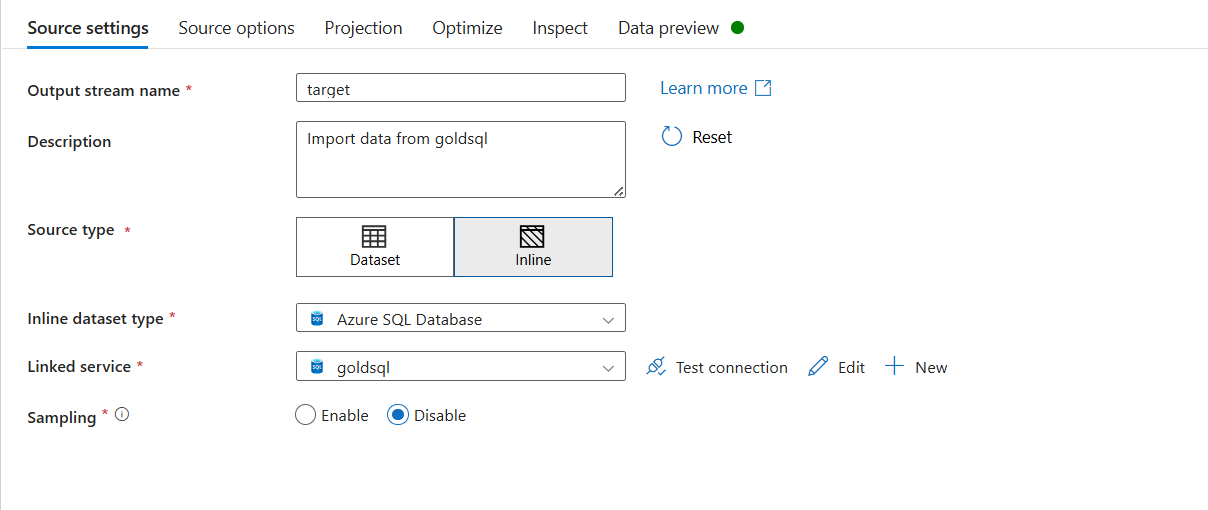


* Add derived column settings and rename it to “Hashkey” in columns field give src\_hashkey and in expression field give crc32(concat(toString(src\_payment\_id),toString(src\_loan\_id),toString(src\_payment\_date),toString(src\_payment\_amount))) and preview the data

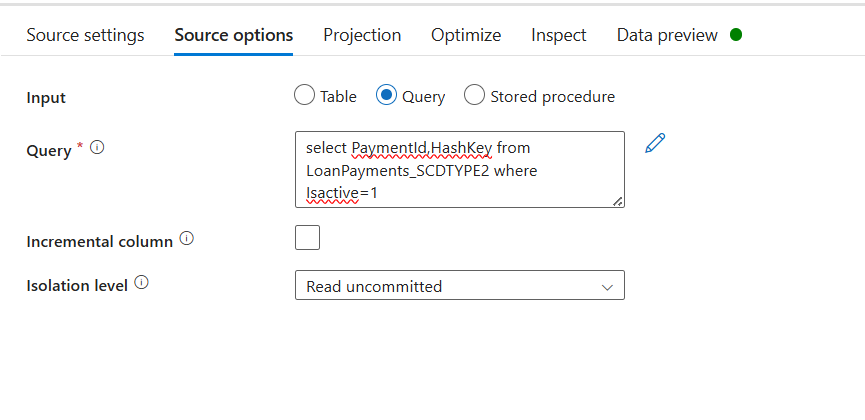




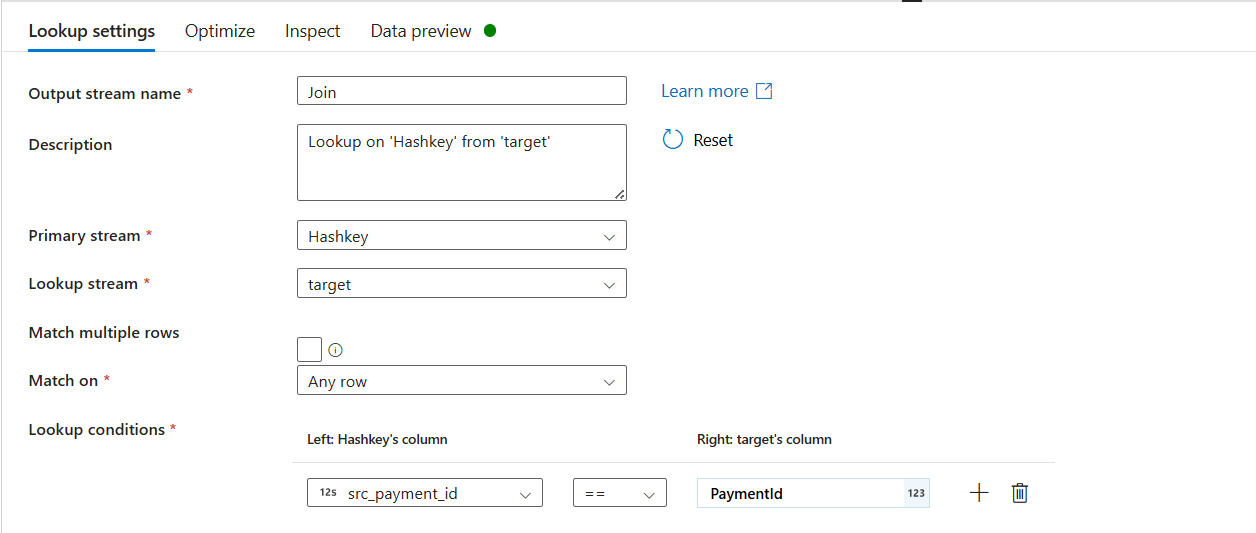
* Add a source and rename it to target and choose the **datatype** as **inline** and the **inline dataset type** as **Azure SQL Database**



* In **source options** choose **query** as input and write the below query in the query field select PaymentId,HashKey from LoanPayments\_SCDTYPE2 where Isactive=1



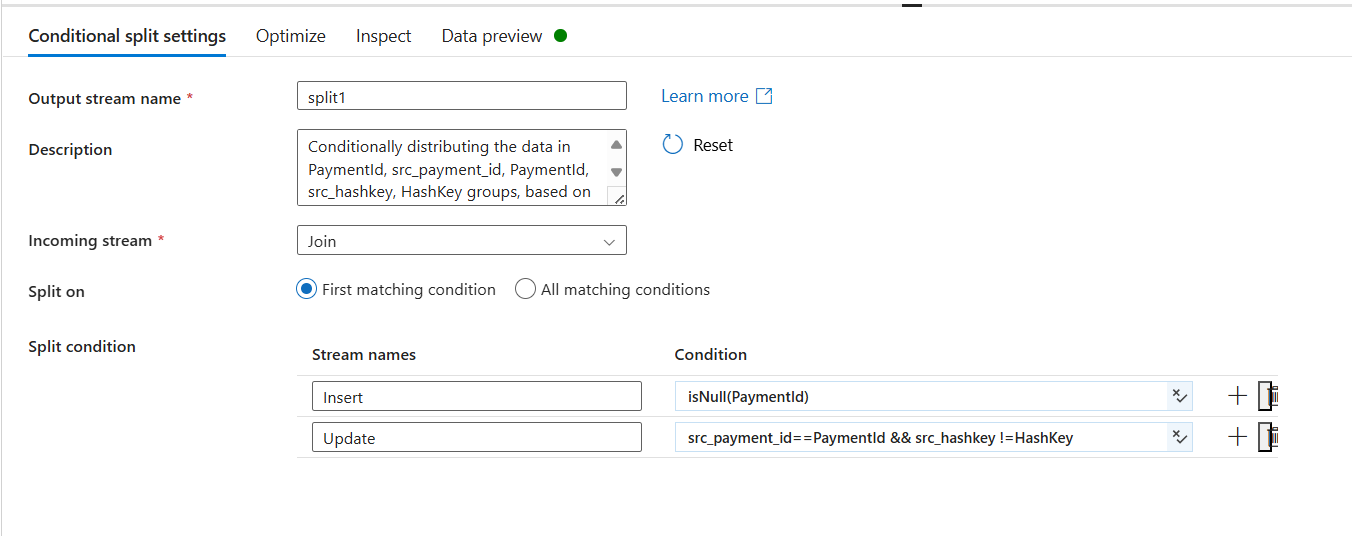
* Import the schema and preview the data
* Next add Lookup and rename it to “Join”, in **Primary** stream choose **Hashkey** and in **Lookup** Stream choose **target**
* In hashkey column choose src\_paymentid and in target column choose paymentid and preview the data it should have Null values in paymentid and Hashkey



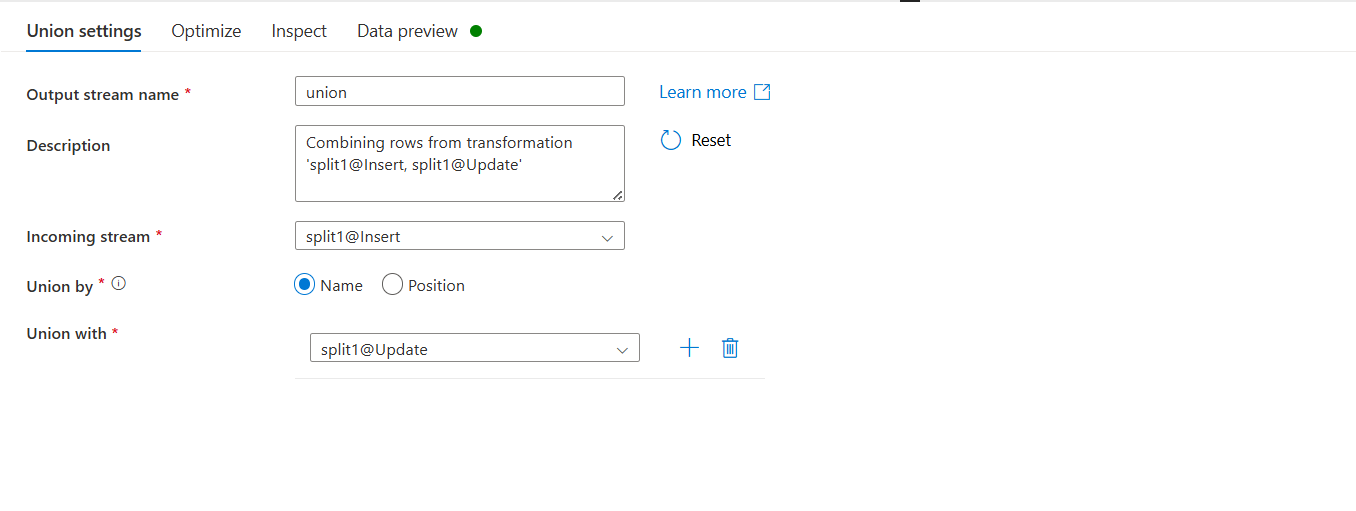
* Add conditional split and keep the stream names as Insert and Update

For Insert the Condition is isNull(PaymentId)

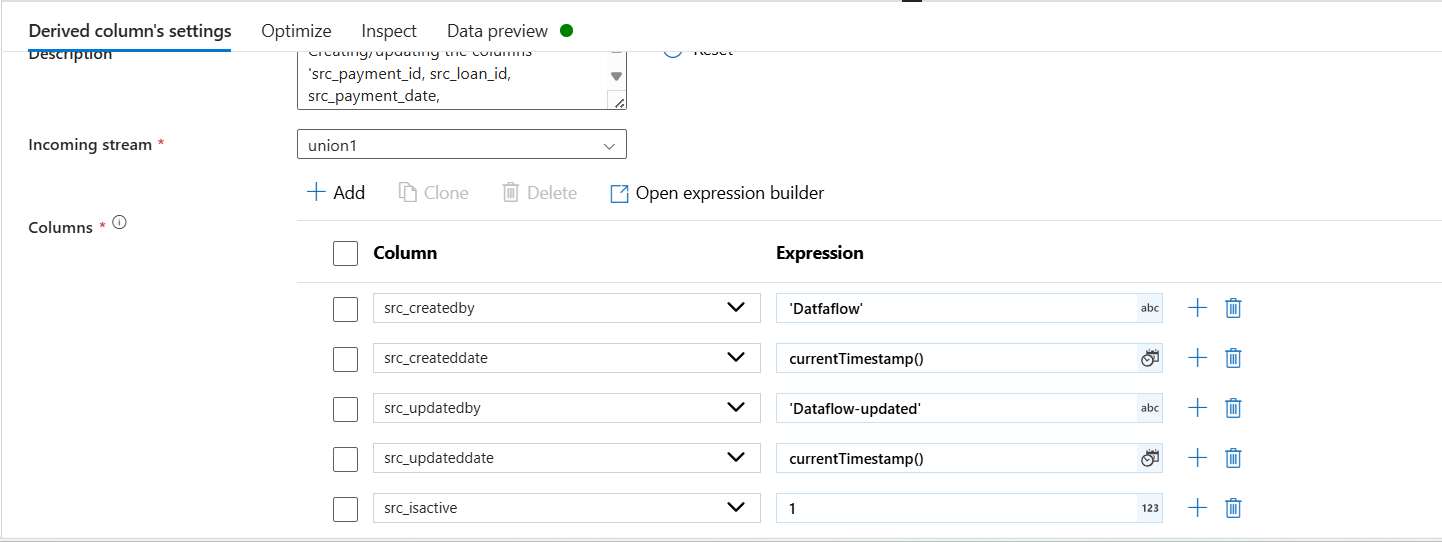
For Update the Condition is src\_ PaymentId ==CustId && src\_hashkey !=HashKey



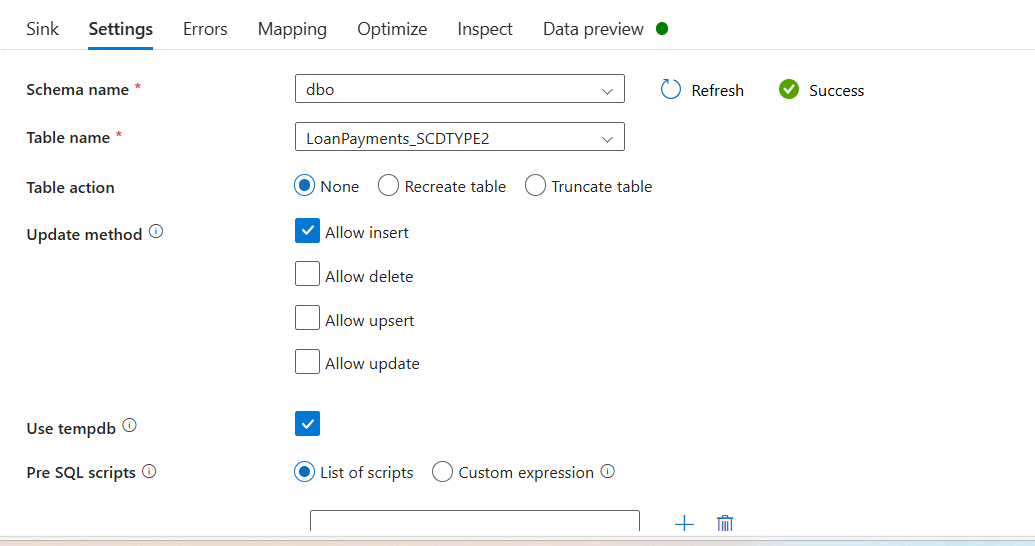
* A union transformation has been added to the insert to merge into a single flow



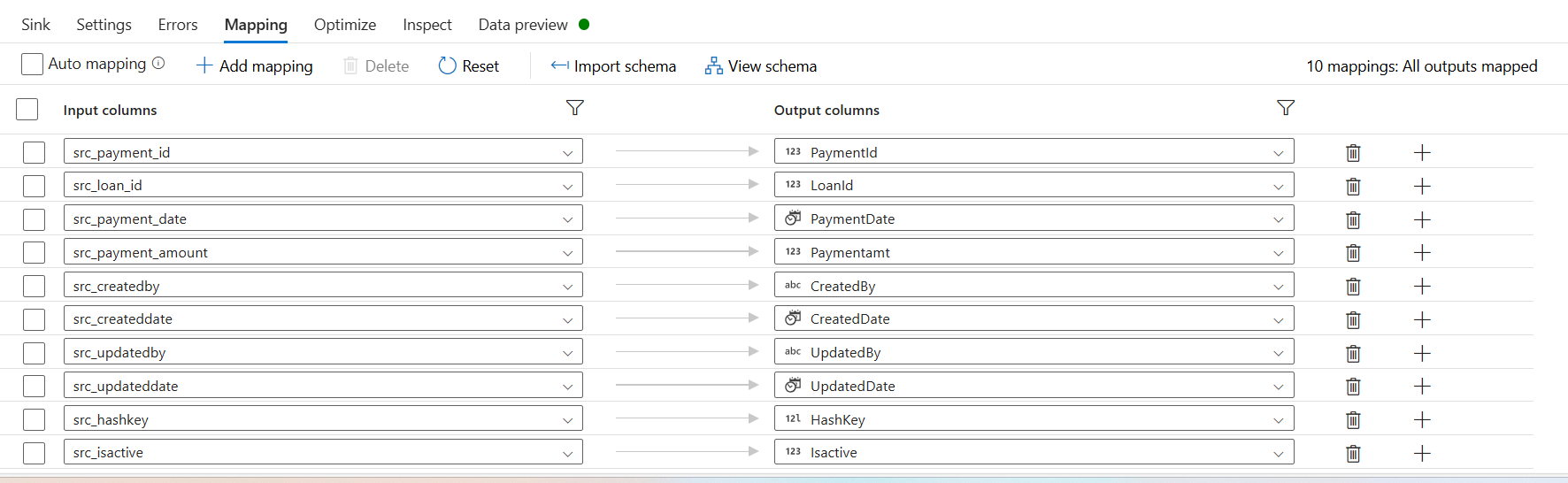
* Add a derived column and rename it to Insert Audit and give the Column name and expression as below

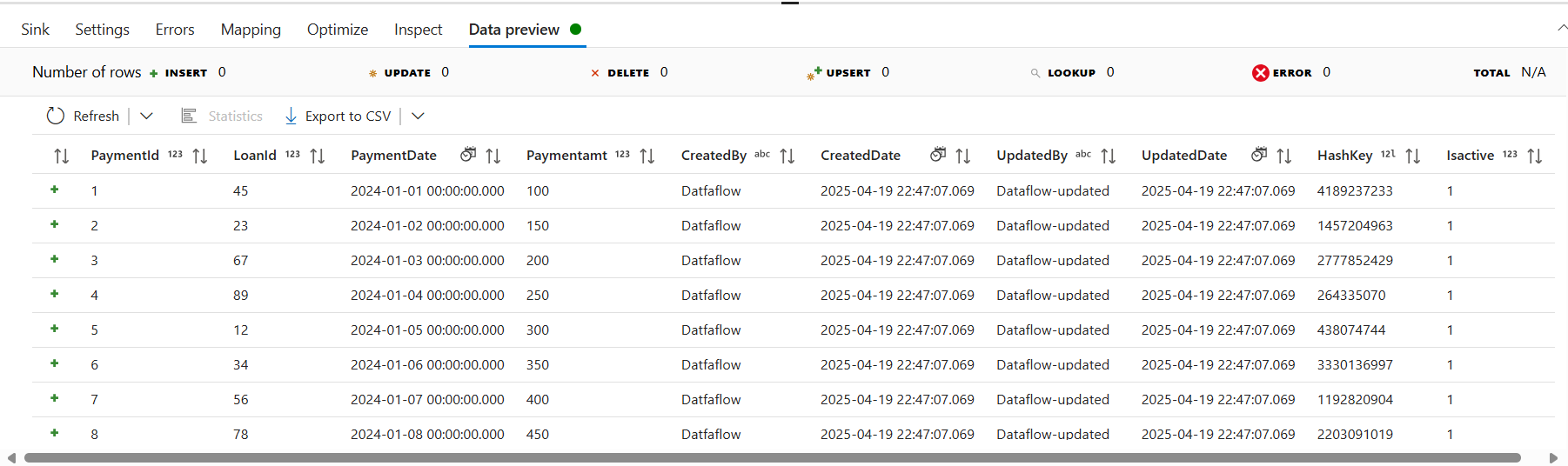


* In settings choose the correct table and keep the update method as follows

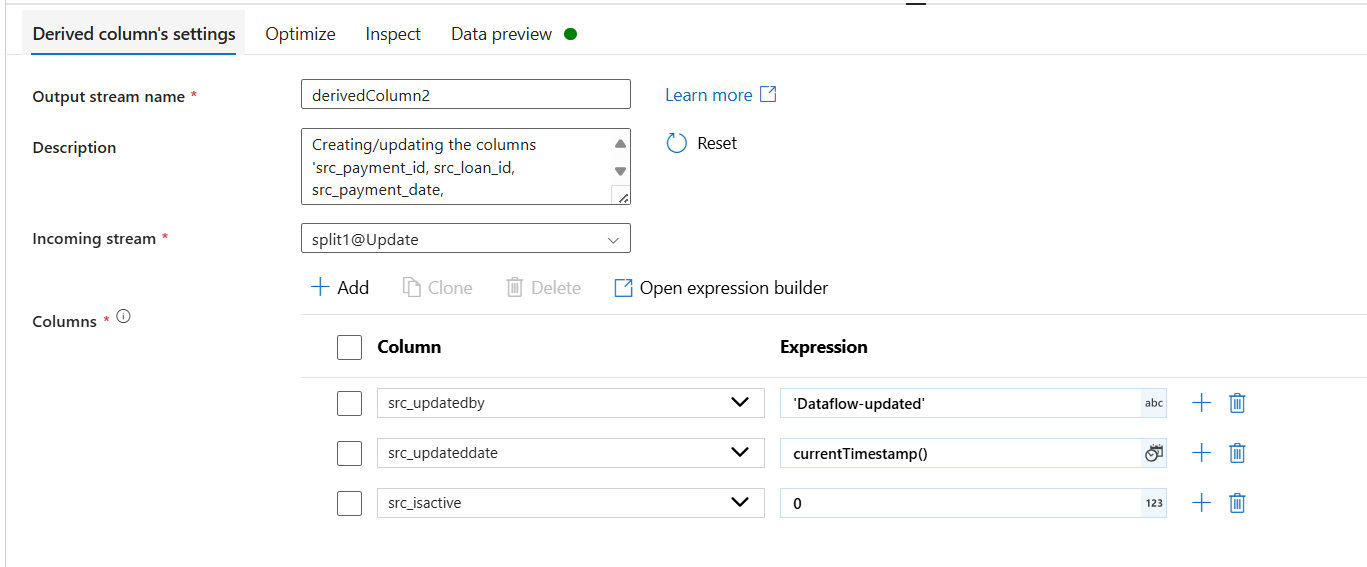


* In mapping just choose and map the columns which will be updated

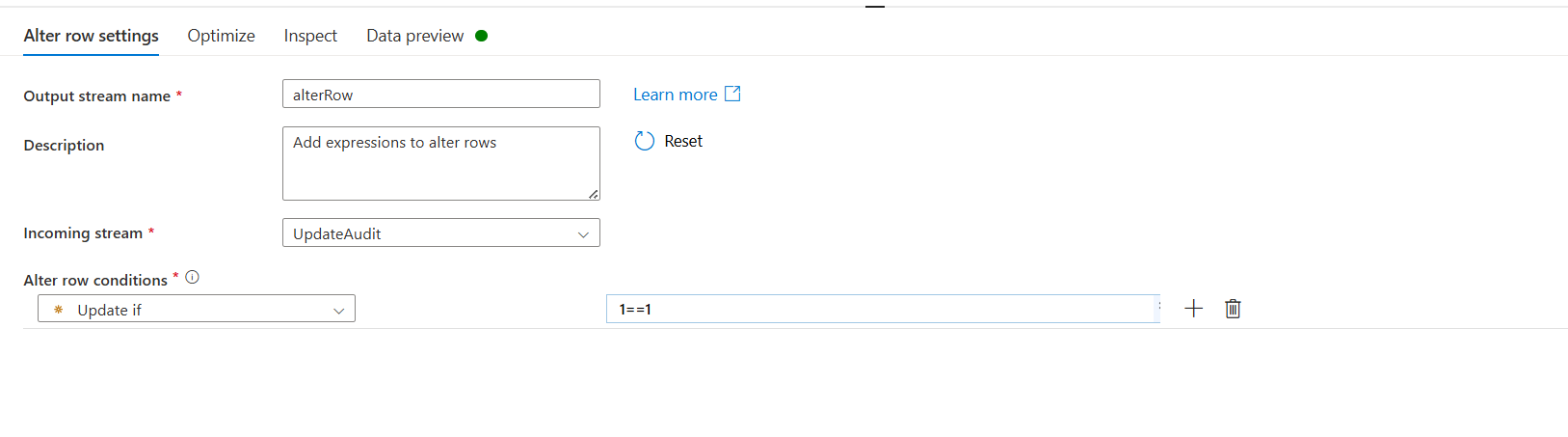




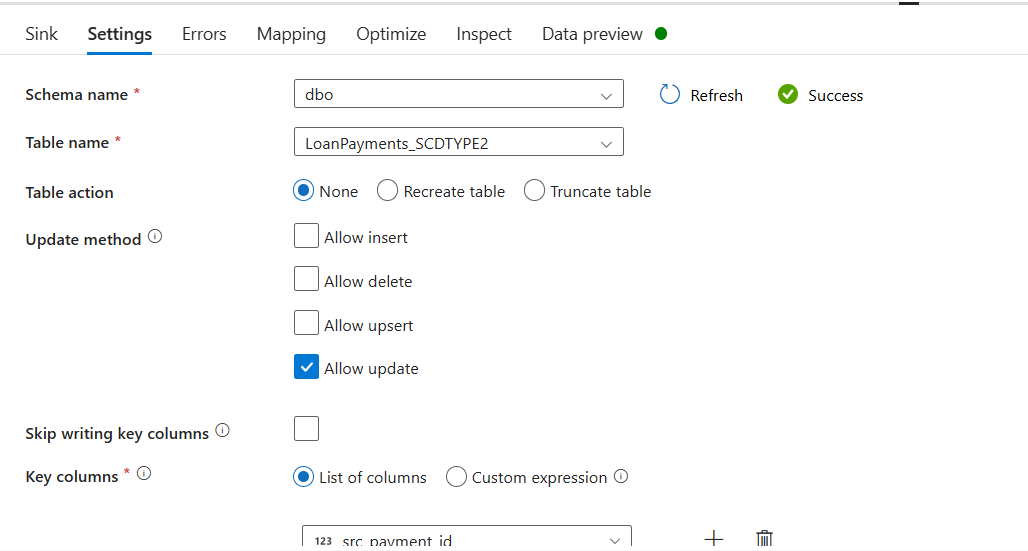
* In Update side add a derived column and rename it to update Audit and give the Column name and expression as below



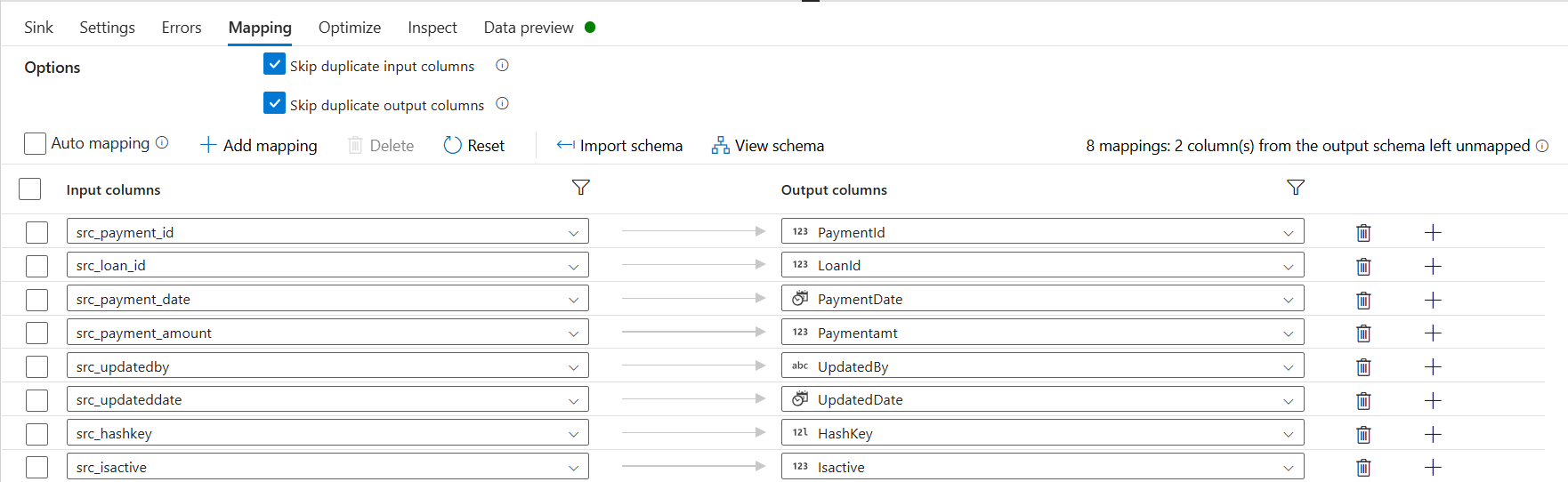
* Add alter row to the Derived column and set the conditions as below

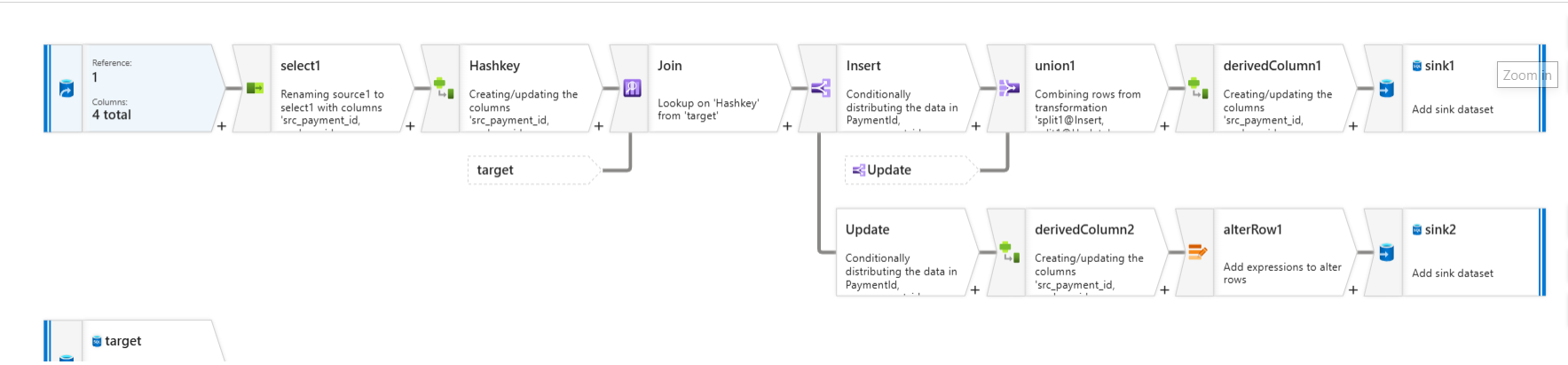


* Add a sink to the above one and rename it to “sinkupdate” and choose the sink type as Inline and Inline dataset type as Azure SQL Database in Sink Tab
* In settings choose the correct table and keep the update method as follows and in key columns give custid and hashkey

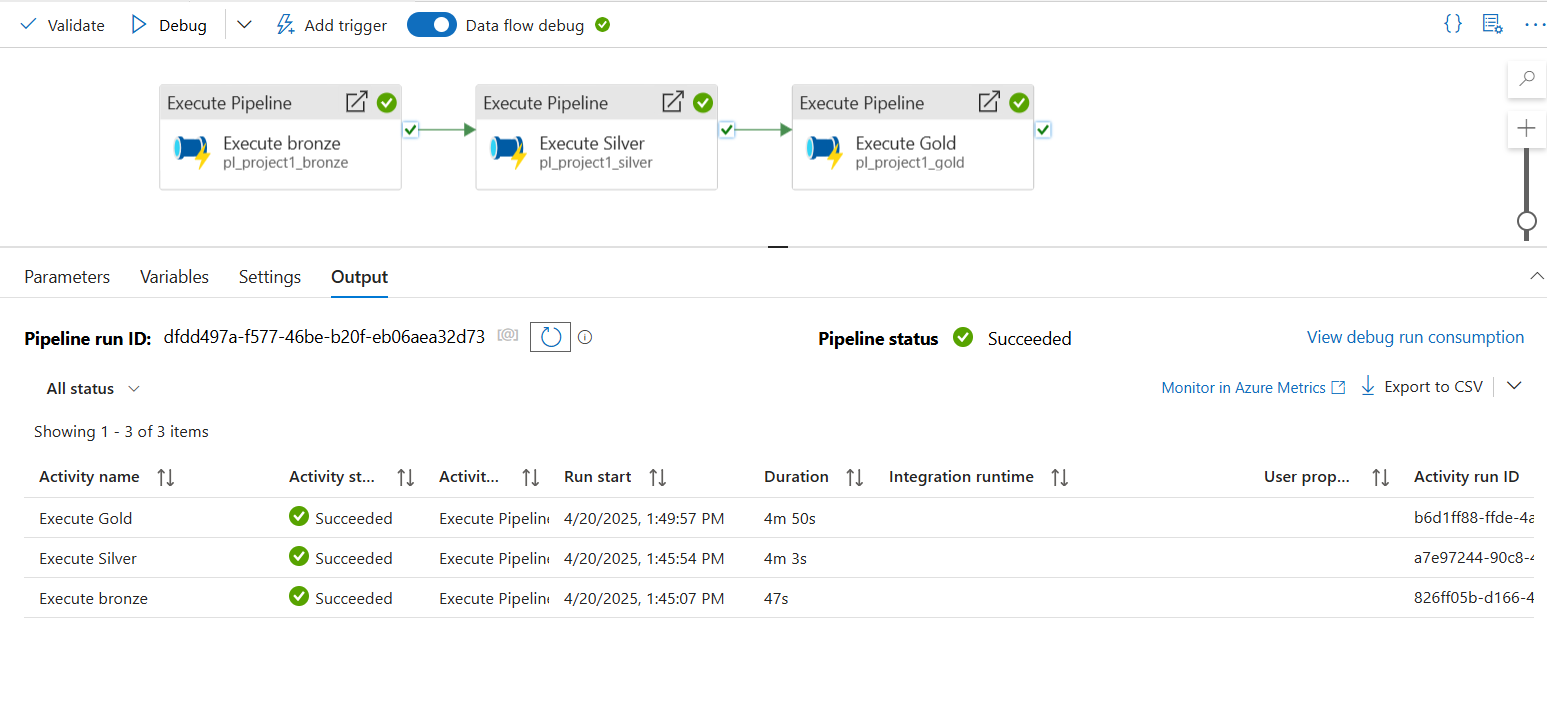


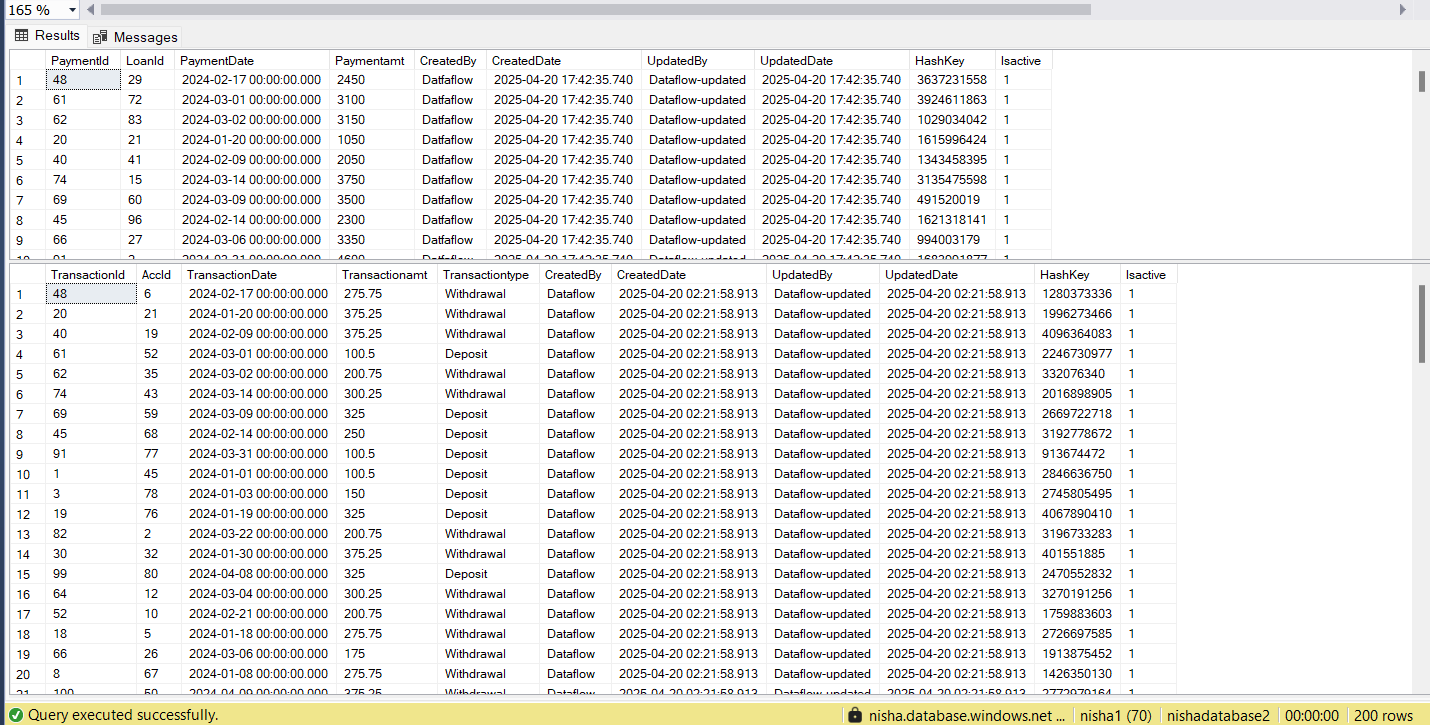
* In mapping just choose and map the columns which will be updated





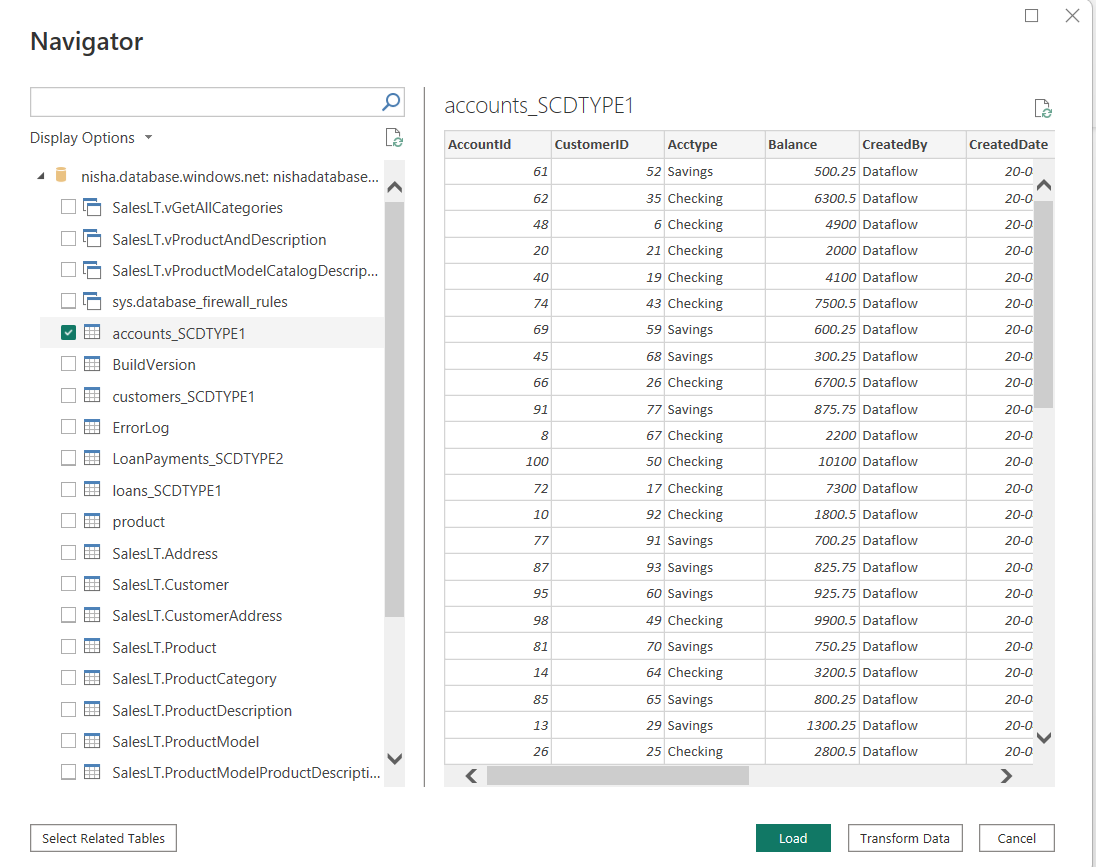
* Below is the output

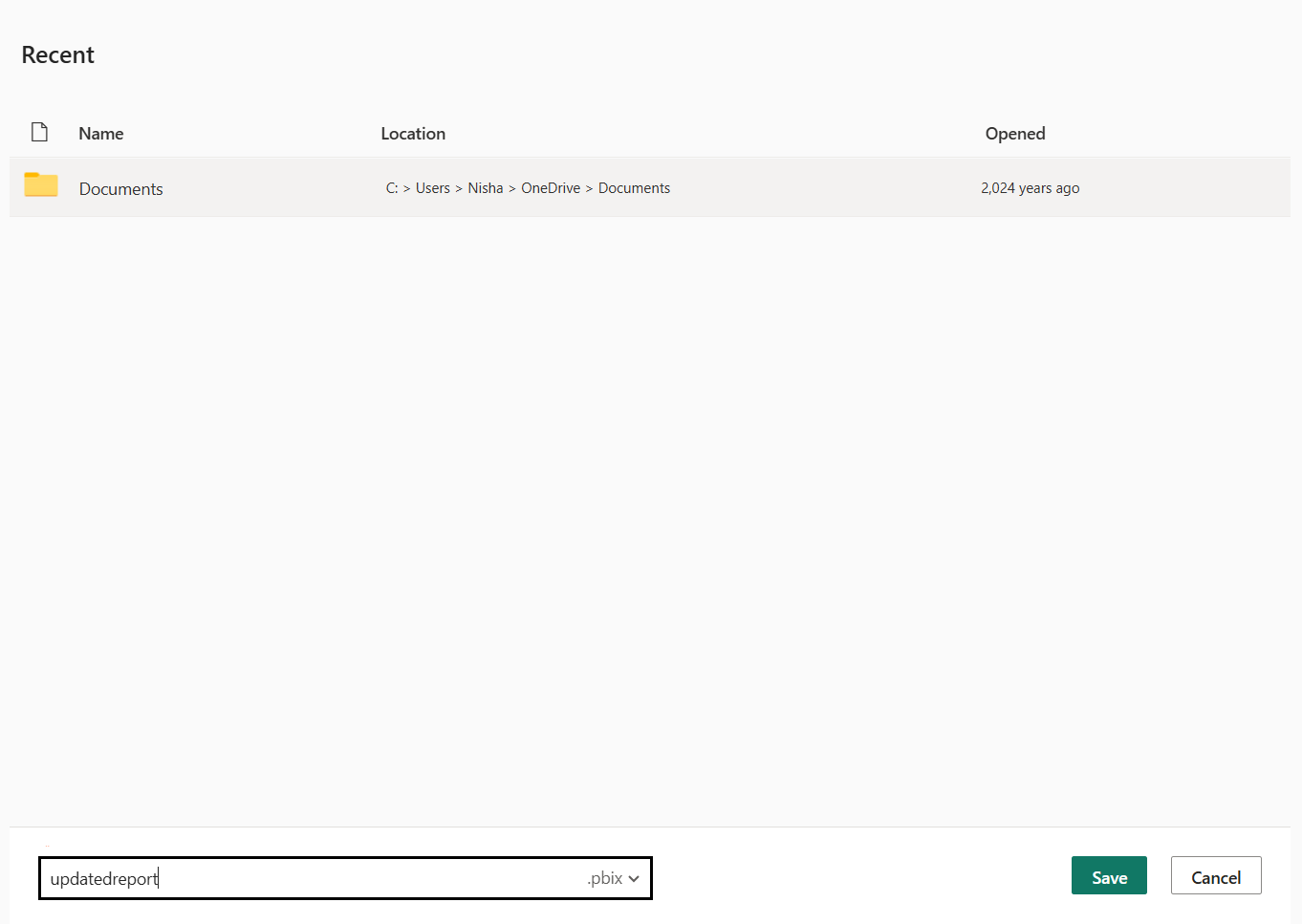
****

****

**Reporting using Power BI desktop**

Step 1: Download Power BI desktop and choose SQL server as data source and select the files for that we need to create reports





Step 2: Publish that to fabric account and check over there whether we got the report

