😊 Cache Sink Implementation

In Azure Data Factory, the Cache Sink is a component used within Mapping Data Flows to optimize data processing performance by caching intermediate data during transformations. Here's a brief overview of its implementation:

1. **Purpose:** The Cache Sink is used to store intermediate results of data transformations within a Mapping Data Flow. By caching data, it reduces the need to reprocess the same data multiple times, improving overall performance and reducing data movement.
2. **Configuration:** To implement a Cache Sink in Azure Data Factory, you add it to your Mapping Data Flow and configure it to cache the data at a specific point in your data transformation logic. You specify the columns to cache and the cache storage settings.
3. **Usage:** Once configured, the Cache Sink temporarily stores the data during the execution of the data flow. Subsequent transformation steps can then access this cached data without needing to reprocess the original source data.
4. **Benefits:** Using a Cache Sink can significantly improve performance, especially in scenarios where multiple transformations are applied to the same dataset. By caching intermediate results, it reduces the computational overhead and data movement within the data flow, leading to faster processing times.
5. **Considerations:** While Cache Sink can enhance performance, it's essential to use it judiciously, as caching large datasets can consume additional storage resources. It's also crucial to manage cache expiration and refresh policies to ensure that the cached data remains up-to-date.

😄 Use cases of Cache Sink:

The Cache Sink in Azure Data Factory's Mapping Data Flows finds application in various scenarios where optimizing data processing performance is crucial. Here are some common use cases:

1. **Complex Transformations:** When implementing complex data transformations involving multiple steps or iterations, the Cache Sink can be used to store intermediate results. This prevents redundant processing of the same data, reducing computational overhead and improving overall efficiency.
2. **Aggregations and Joins:** In scenarios where aggregations or joins are performed on large datasets, caching intermediate results using the Cache Sink can significantly improve performance. It allows subsequent transformations to access pre-computed results instead of recalculating them, leading to faster processing times.
3. **Repetitive Data Access:** If certain data subsets are accessed multiple times within a data flow, caching them using the Cache Sink can eliminate the need for repeated data reads from the source. This is particularly useful in scenarios where source data is located in remote or slow-access systems.
4. **Iterative Development:** During the development and testing phases of a data integration project, the Cache Sink can be used to temporarily store intermediate data

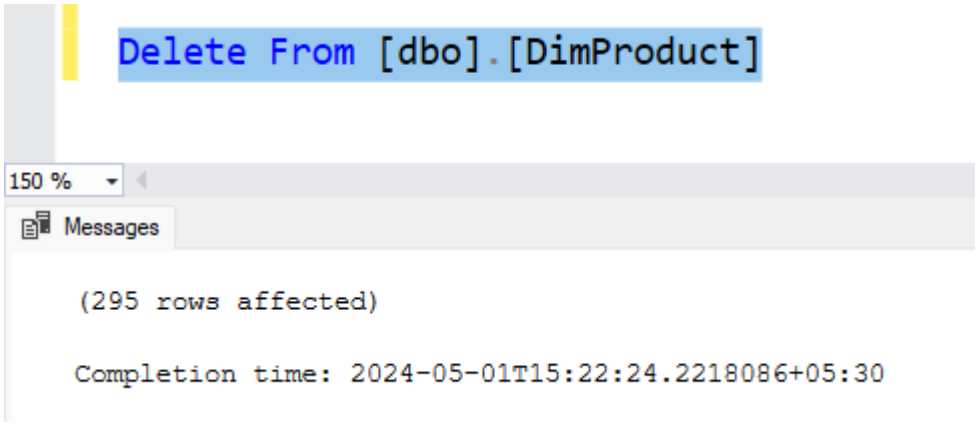
for debugging and validation purposes. It allows developers to inspect and analyze intermediate results without rerunning the entire data flow repeatedly.

5. **Real-Time Data Processing:** In real-time data processing pipelines, where low-latency processing is essential, caching intermediate results using the Cache Sink can help meet stringent performance requirements. It enables faster data processing by reducing the computational overhead associated with complex transformations.
6. **Batch Processing Optimization:** When dealing with batch processing of large volumes of data, caching intermediate results using the Cache Sink can streamline processing and reduce overall execution times. It ensures that previously processed data is readily available for subsequent transformations, minimizing redundant work.
7. **Data Deduplication:** In scenarios where duplicate records need to be identified and removed from datasets, caching intermediate results using the Cache Sink can facilitate efficient deduplication processes. By storing unique identifiers or hash values of processed records, it enables faster comparison and elimination of duplicates.
8. **Incremental Loading:** When loading data incrementally into a target system, caching intermediate results using the Cache Sink can improve performance by reducing the need to reprocess unchanged data. It allows for efficient delta processing by caching previously processed records and identifying new or modified ones.

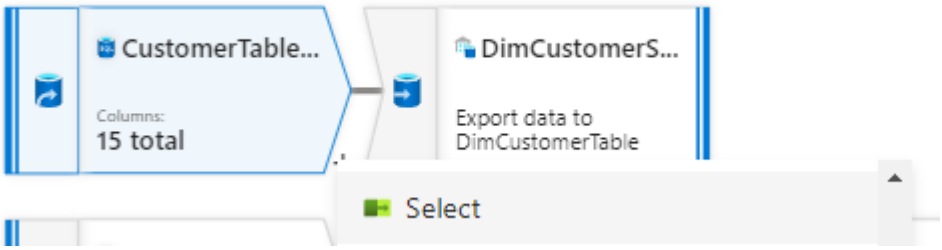
In this guide, we're implementing the Cache Sink feature in Azure Data Factory's Mapping Data Flows to optimize data processing performance. The end goal is to reduce computational overhead and improve efficiency by caching intermediate results during data transformations. By following the provided steps, users can configure Cache Sink, apply it to various use cases such as complex transformations and aggregations, and ultimately streamline their data integration processes for faster and more efficient results.

😄 To begin with the Lab:

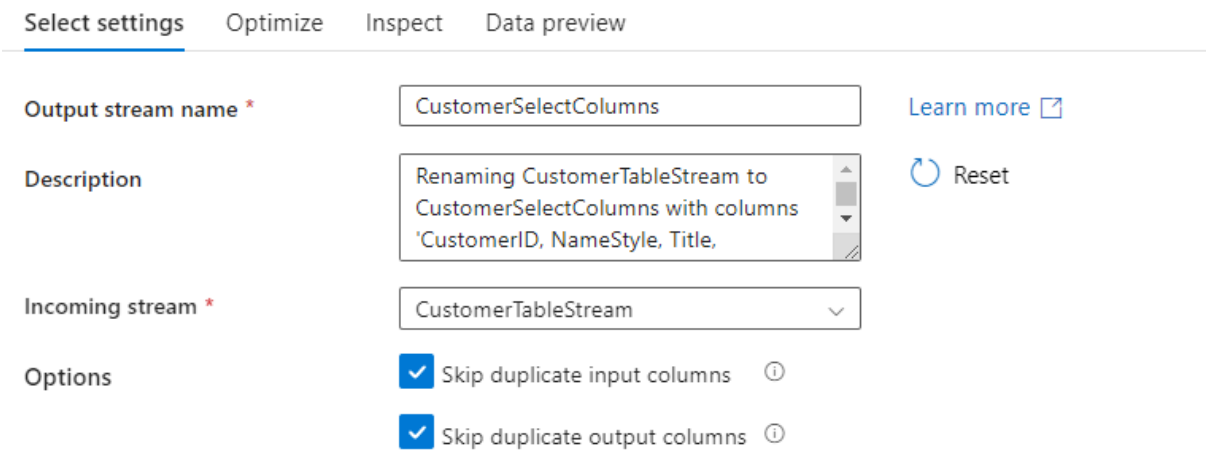
1. In the last lab we added surrogate keys to our Dimension Product Table.
2. This time we will be doing the same for the Dimension Customer Table.
3. First, we are going to drop our customer table and create a new one with the Product SK column in it.
4. At the same time since we are going to run our pipeline again, we have to delete the data from the product table also.



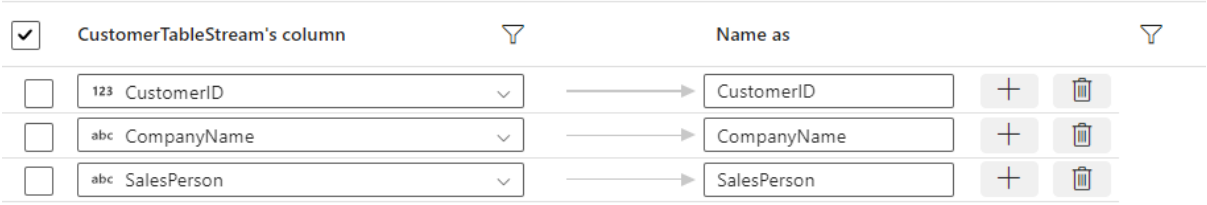
5. Now we are going to do the same implementation as before but this time it will be our customer table.
6. So, head towards Data Factory Wizard and open your dimension dataflow.
7. Now choose your customer table stream and then click on the plus icon and choose select column modifier just to select the columns of our interest.



8. Then you need to give it name and after that scroll down to the mappings.

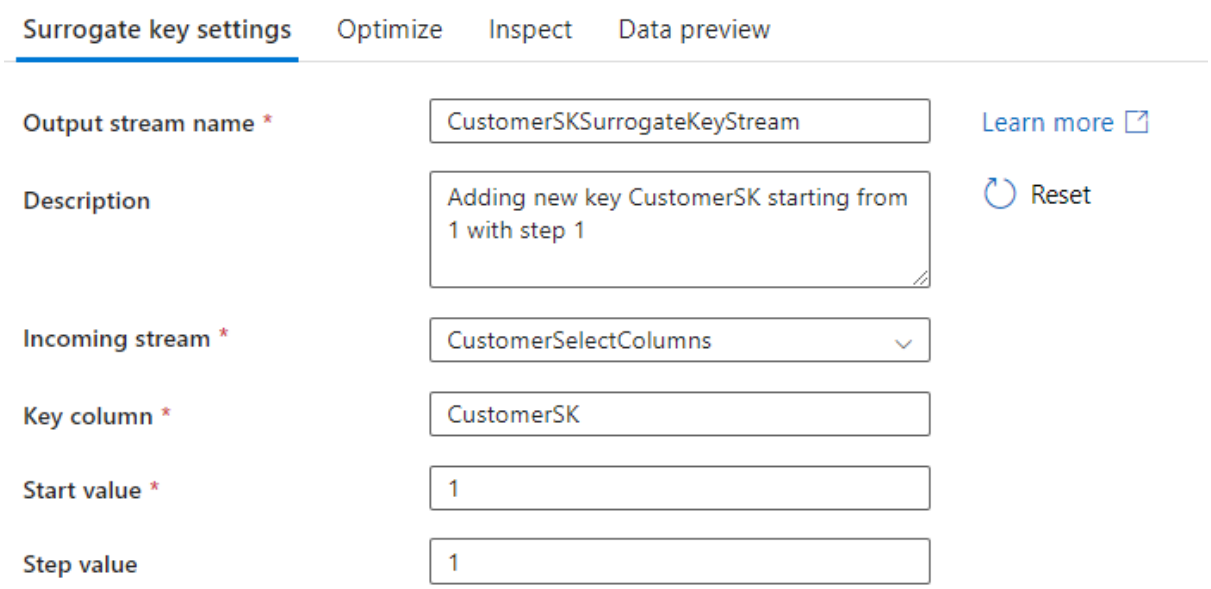


9. Here we just need the customer ID, company name, and salesperson. You have to remove the other mappings.

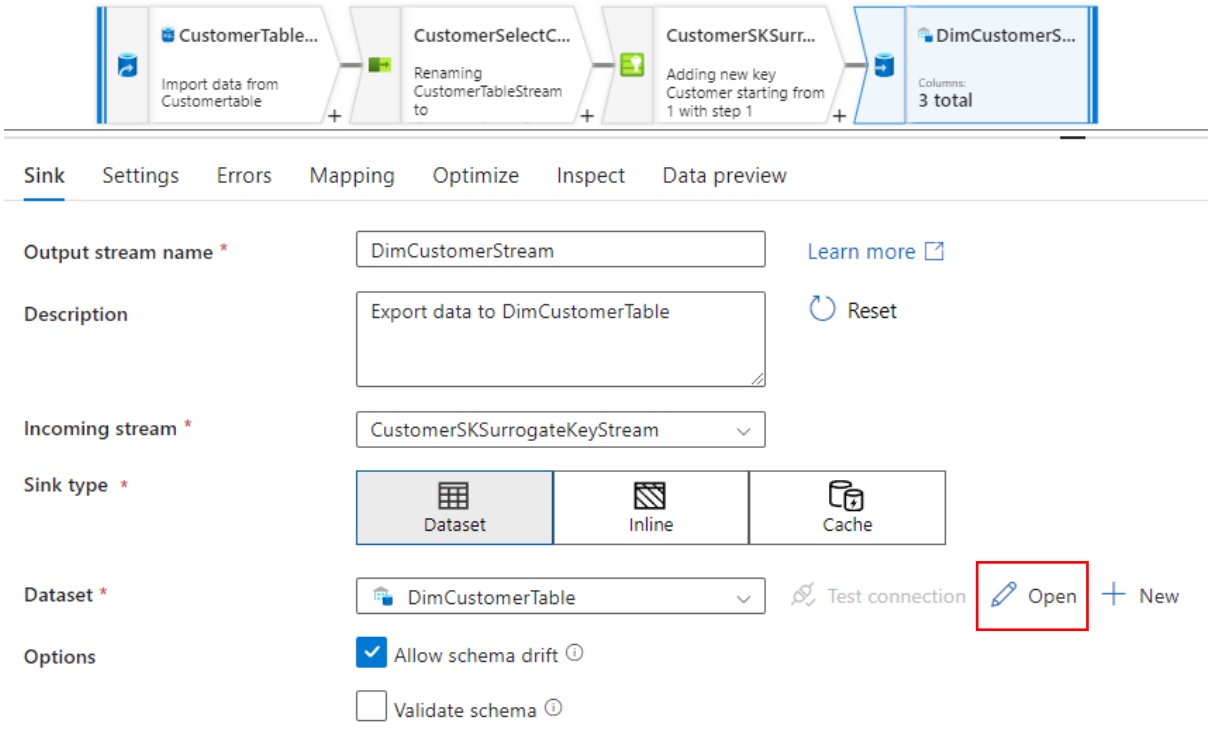


10. Then we need to add the surrogate key. For the customer select columns click on plus icon and choose surrogate key.

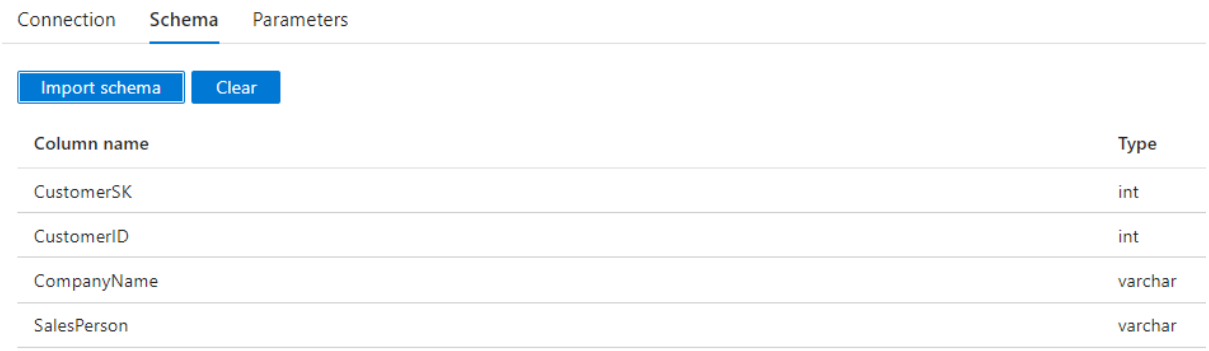
11. Now you need to give your stream a name then choose customer select columns as your incoming stream and in the key column give Customer SK.



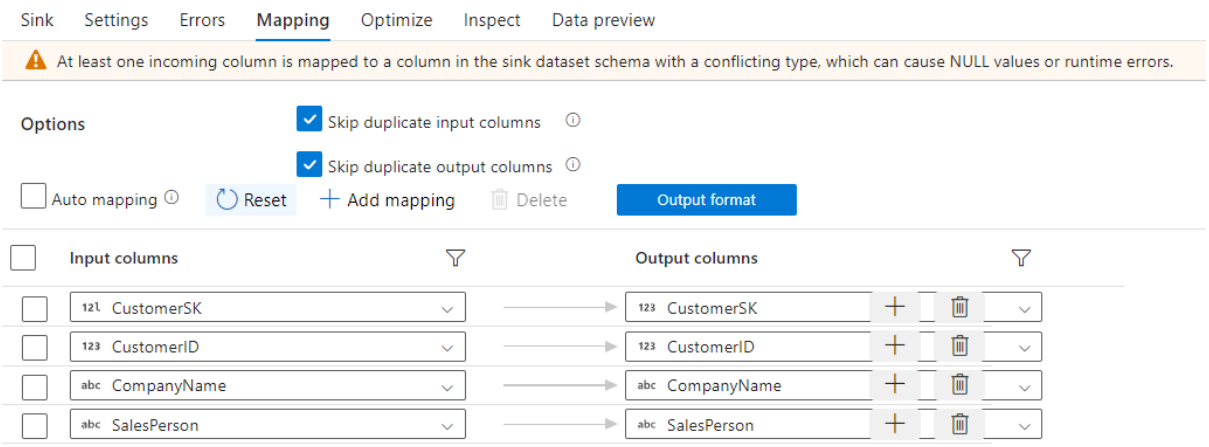
12. Now you need to go to the sink and click on open.



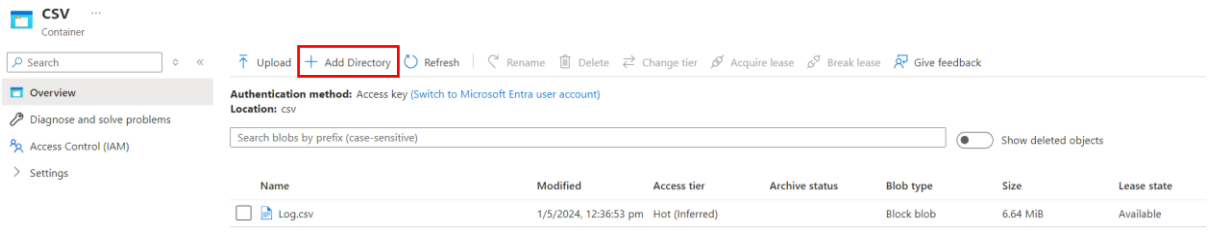
13. Then go to schema and click on import schema.



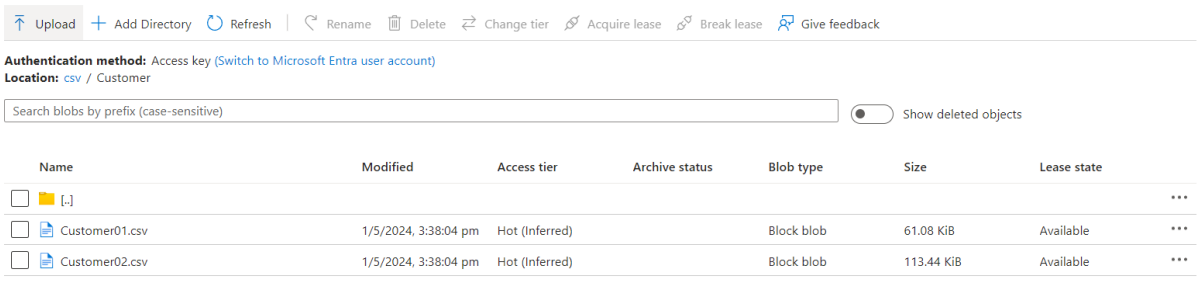
14. After that go to mappings and click on reset.



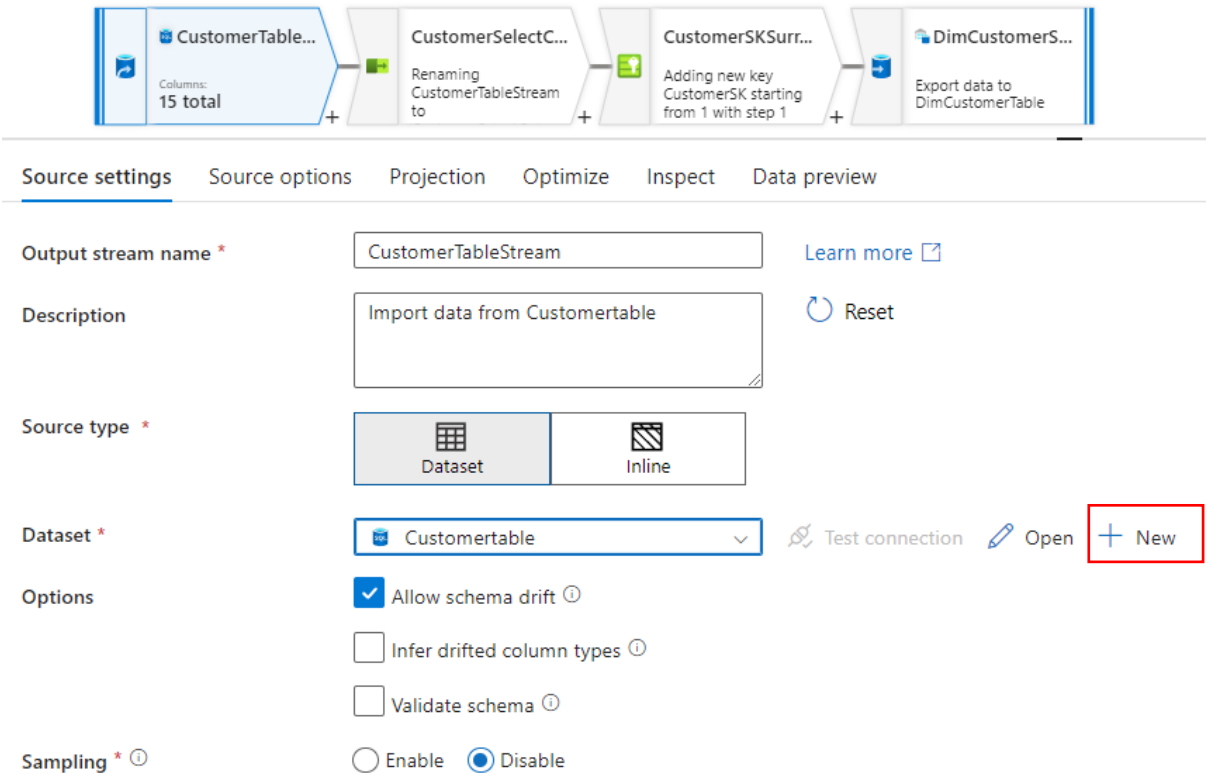
15. Then you have to click on Publish All.
16. Now you have to go to your CSV container in your storage account. Here you have to click on Add directory. Then name it and save it.



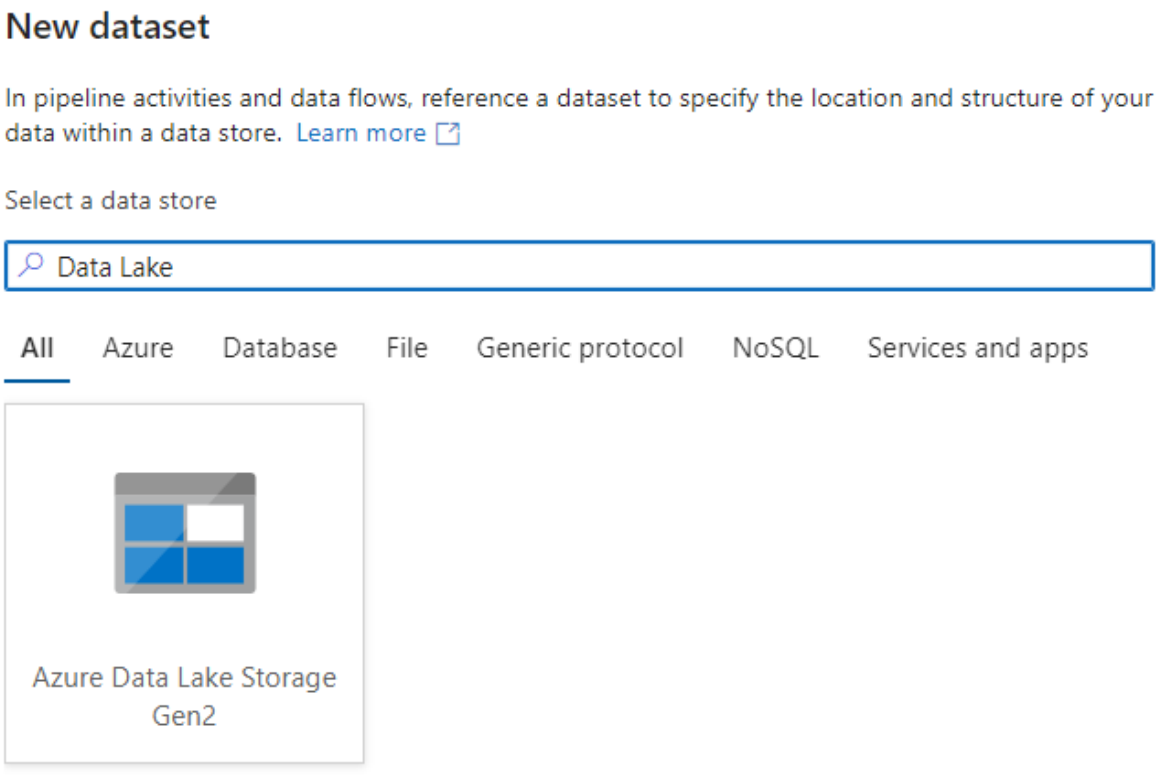
17. After that you have to go inside that directory and upload two files. So, these two files have the data from the customer table which we have seen in our dimension table. The data is divided into both of them.



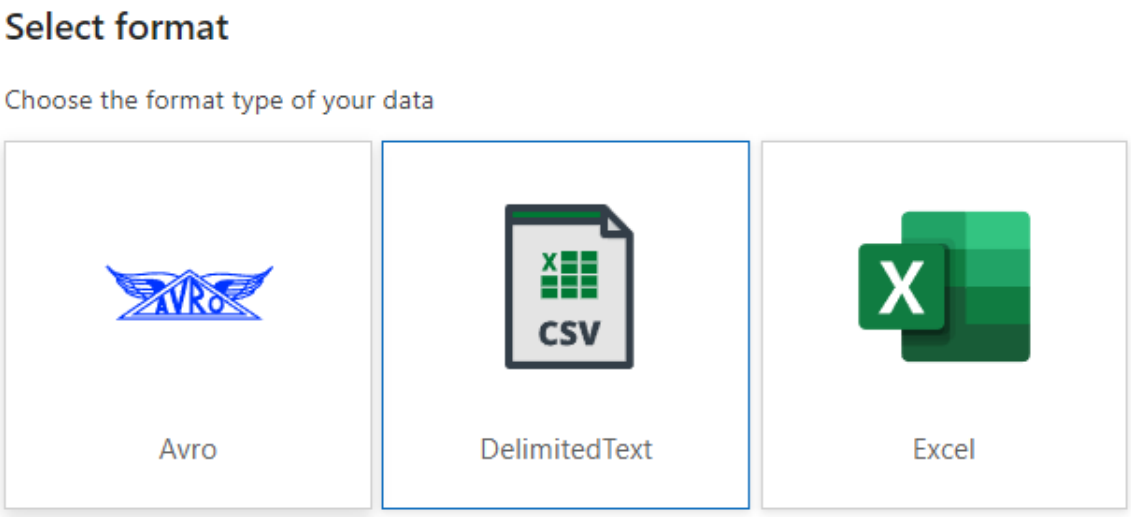
18. Now if you see in your customer table stream the dataset is pointing towards the customer table now we are going to create a new dataset for it. For that click on new.



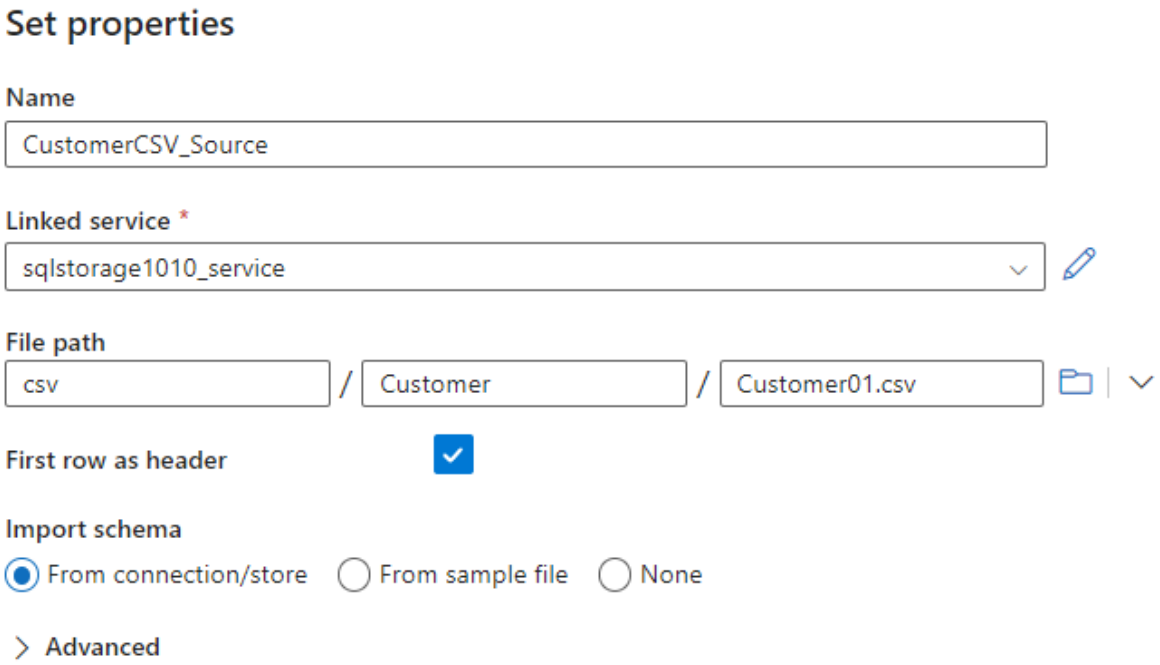
19. Then you have to choose Azure data lake gen2 storage account.



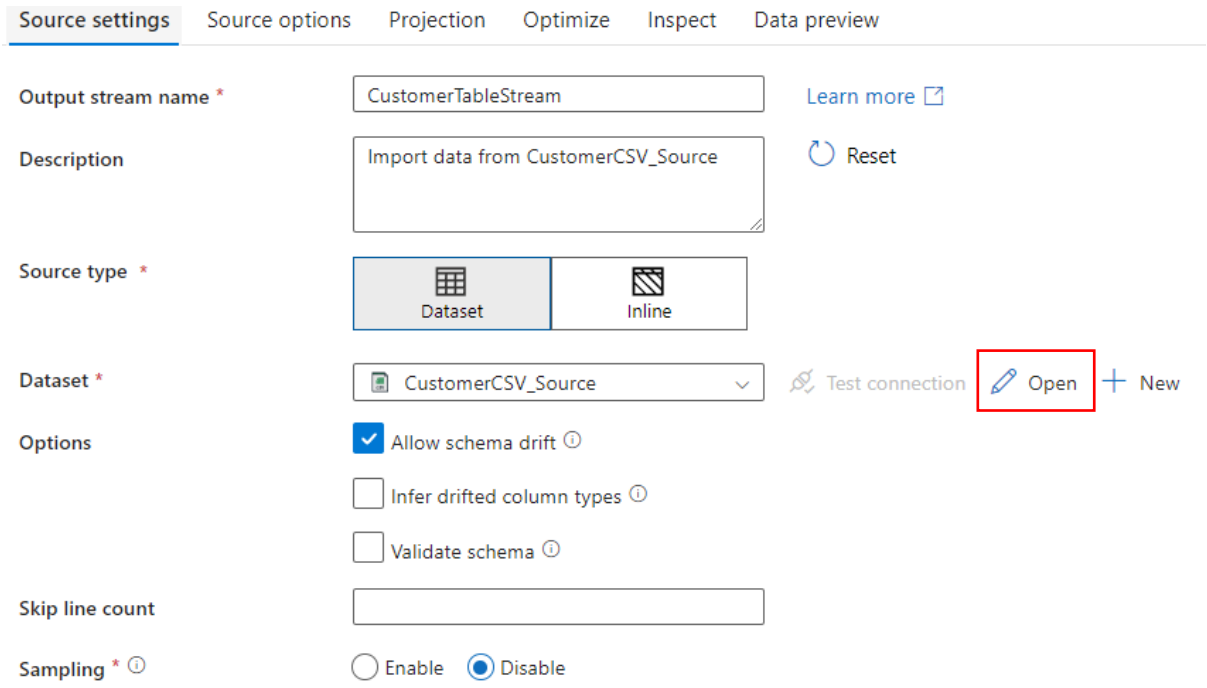
20. After that in the format choose delimited text.

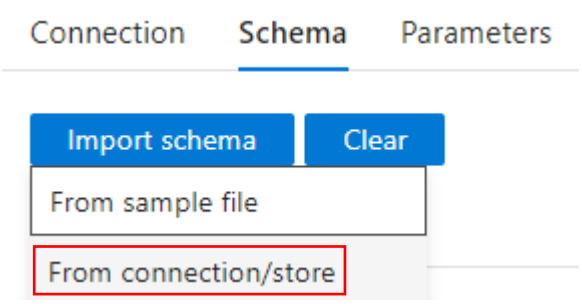


21. Then in the set properties option you have to give it a name, browse for the CSV container, in that the new directory, and then choose Customer01.csv

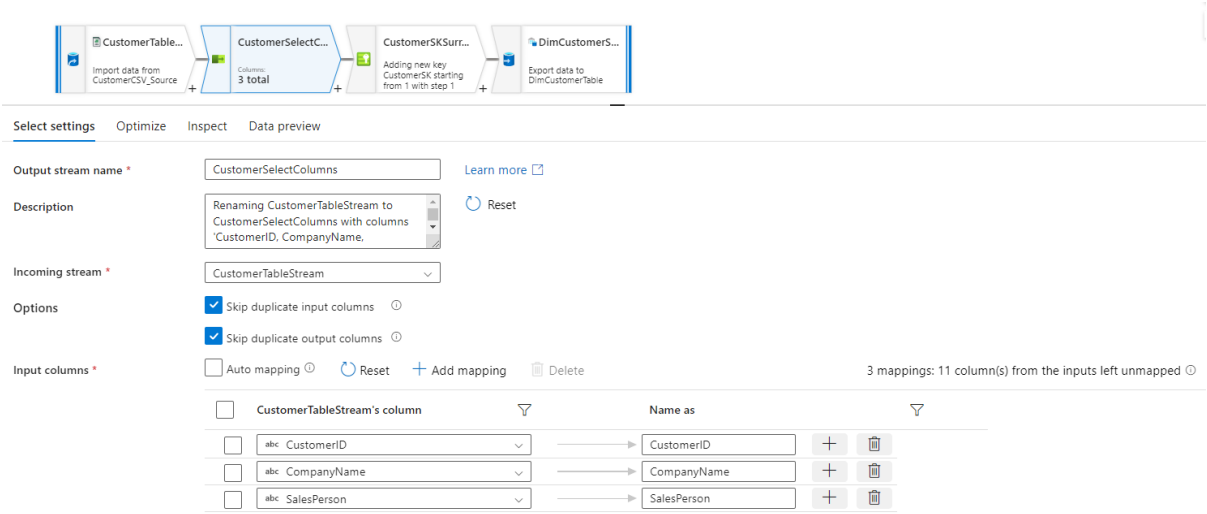


22. Once it is uploaded, you have to click on open, then move to schemas click on import schemas, and choose import schemas from the connection store.

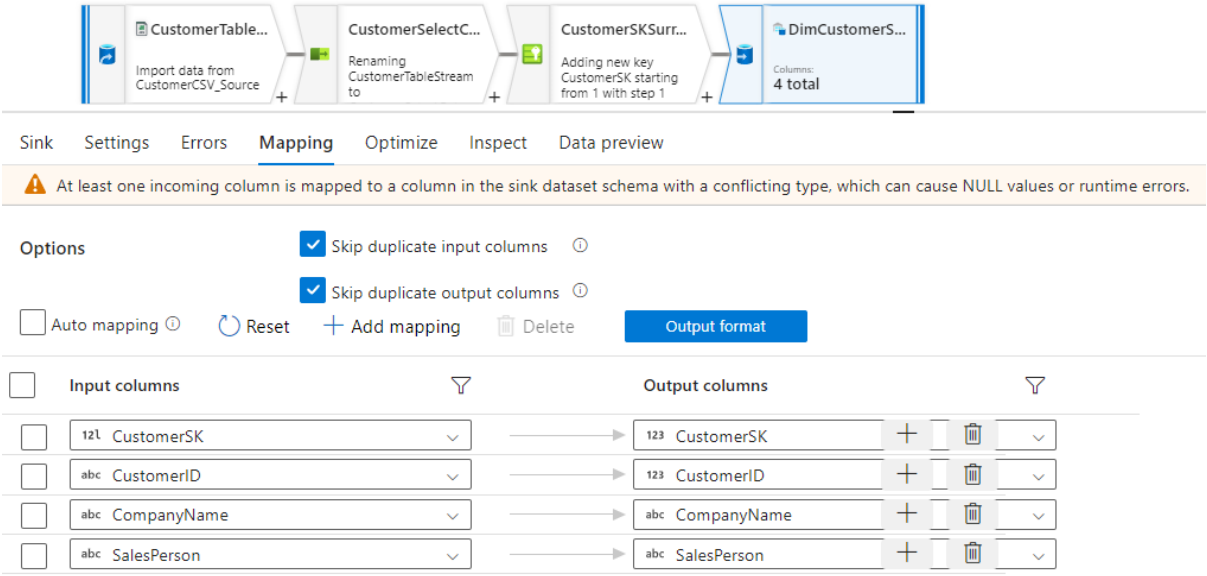




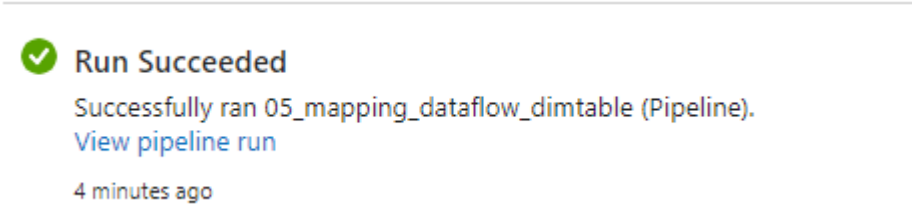
23. Now go back to customer select column stream then scroll down and reset your schema. You should have the same schema as shown below.



24. Then go back to sink and in the mapping click on reset. You should have the same mappings as shown below. Then click on publish all.



25. After that go back to your pipeline and trigger it to run your pipeline. Below you can see that your pipeline run was successful.



26. And below you can see that when we run the select statement with order, we can see the data accordingly. Here we get the entries added till customer ID 468 because our CSV file has this much information.

```
Select * from [dbo].[DimCustomer]
Order by [CustomerSK]
```

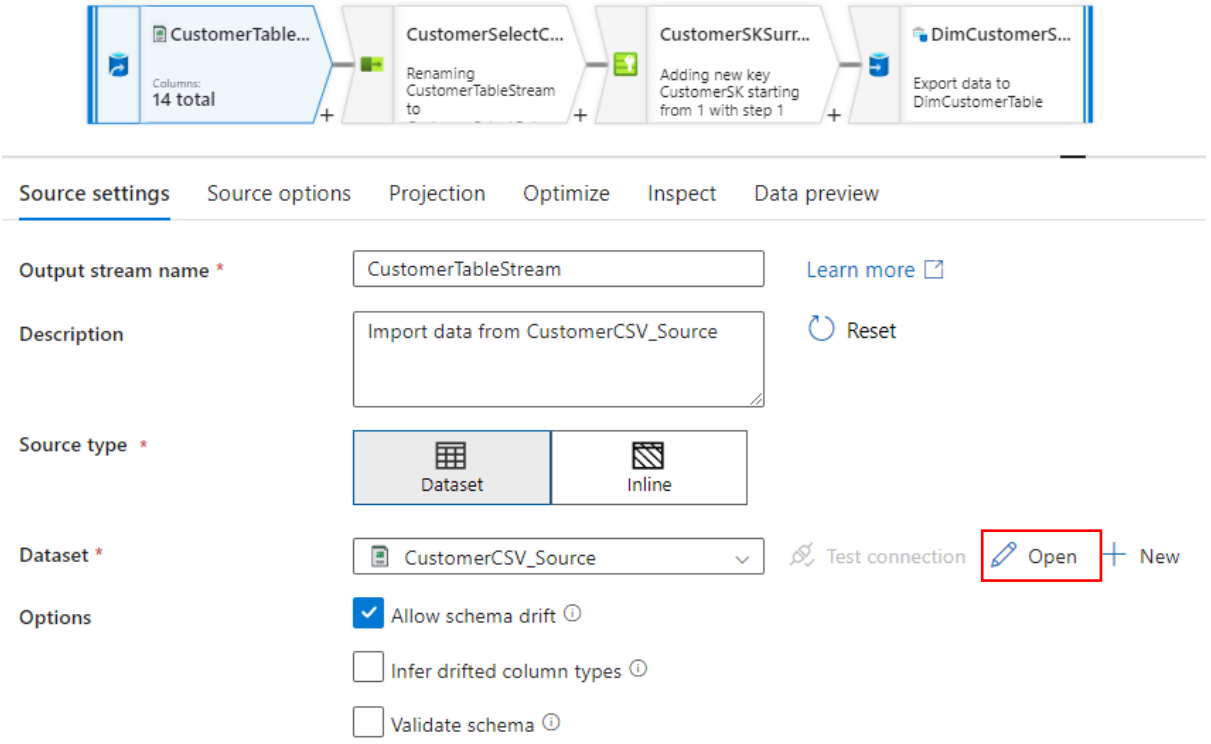
150 %

Results | Messages

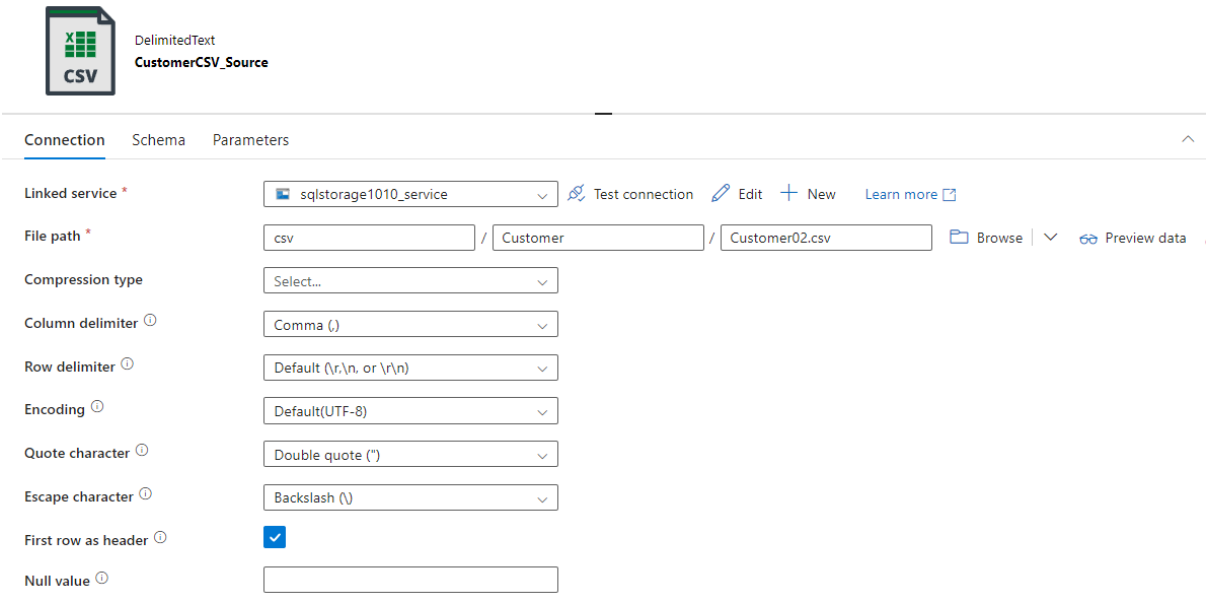
	CustomerSK	CustomerID	CompanyName	SalesPerson
271	271	421	Fitness Cycling	adventure-works\shu0
272	272	424	Bikes for Two	adventure-works\josé1
273	273	425	Bikes for Kids and Adults	adventure-works\josé1
274	274	426	Custom Accessories Company	adventure-works\garrett1
275	275	430	Uttermost Bike Shop	adventure-works\jae0
276	276	431	Small Cycle Store	adventure-works\michael9
277	277	433	Thorough Parts and Repair ...	adventure-works\pamela0
278	278	434	Road-Way Mart	adventure-works\david8
279	279	435	Satin Finish Company	adventure-works\jillian0
280	280	436	Sheet Metal Manufacturing	adventure-works\jillian0
281	281	437	Professional Cycle Store	adventure-works\shu0
282	282	438	Remarkable Bike Store	adventure-works\linda3
283	283	439	Regional Manufacturing	adventure-works\shu0
284	284	442	Original Bicycle Supply Com...	adventure-works\josé1
285	285	443	Courteous Bicycle Specialists	adventure-works\josé1
286	286	444	Finer Cycle Shop	adventure-works\garrett1
287	287	448	Action Bicycle Specialists	adventure-works\jae0
288	288	451	Roadway Bike Emporium	adventure-works\pamela0
289	289	452	A Cycle Shop	adventure-works\david8
290	290	453	Unique Bikes	adventure-works\jillian0
291	291	454	Sleek Bikes	adventure-works\jillian0
292	292	455	Timely Shipping Service	adventure-works\shu0
293	293	456	Riding Excursions	adventure-works\linda3
294	294	457	Retail Sporting Equipment	adventure-works\shu0
295	295	460	Price-Cutter Discount Bikes	adventure-works\josé1
296	296	461	Active Life Toys	adventure-works\josé1
297	297	462	General Riding Supplies	adventure-works\garrett1
298	298	466	Central Bicycle Specialists	adventure-works\jae0
299	299	468	Blue Bicycle Company	adventure-works\michael9

Query executed successfully.

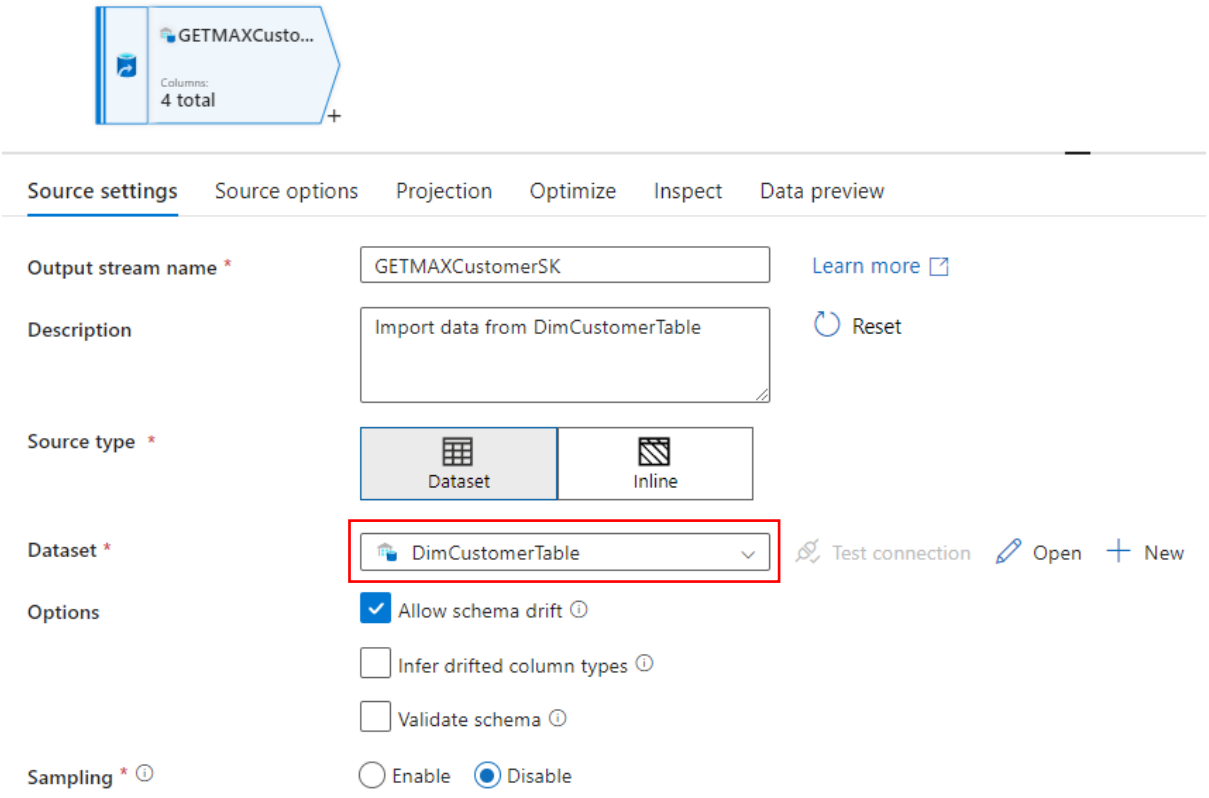
27. Now come back to our customer table Stream and click on open the dataset.



28. Here in the connection, we need to change the location for the CSV file. This time we will choose Customer02.csv to get the other half of the data.



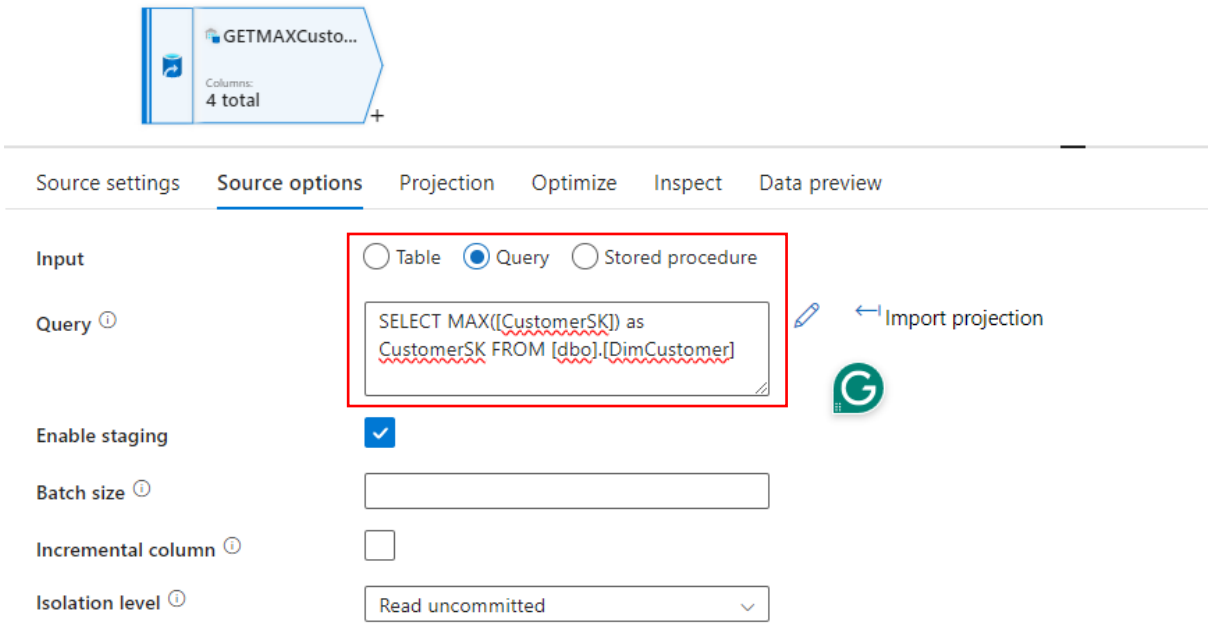
29. Now to our existing dimension data flow we are going to add a new source.
30. First give it a name then for the dataset we are going to choose our Dimension Customer Table which we created in the dataset for our customer sink.



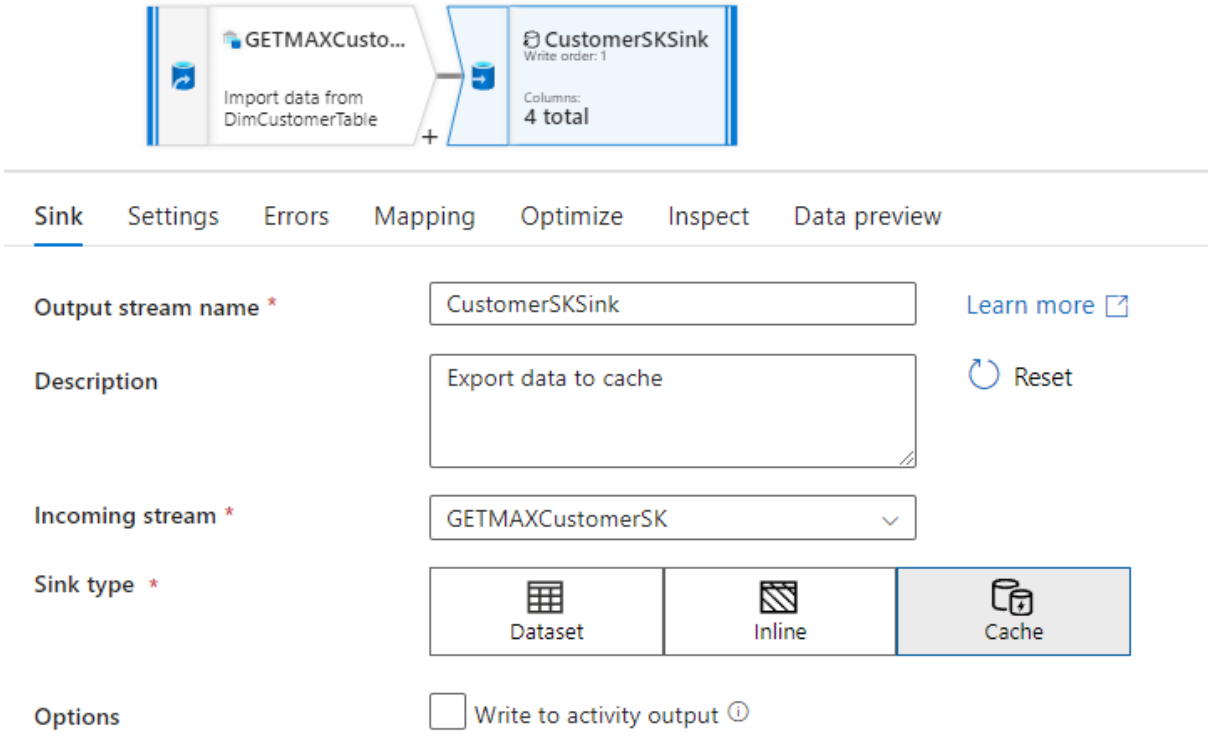
31. Then move to the source option choose the query option then paste the query as shown below.

This SQL query selects the maximum value from the column "CustomerSK" in the table "DimCustomer" from the database schema "dbo" (which typically stands for "database owner"). Here's a breakdown:

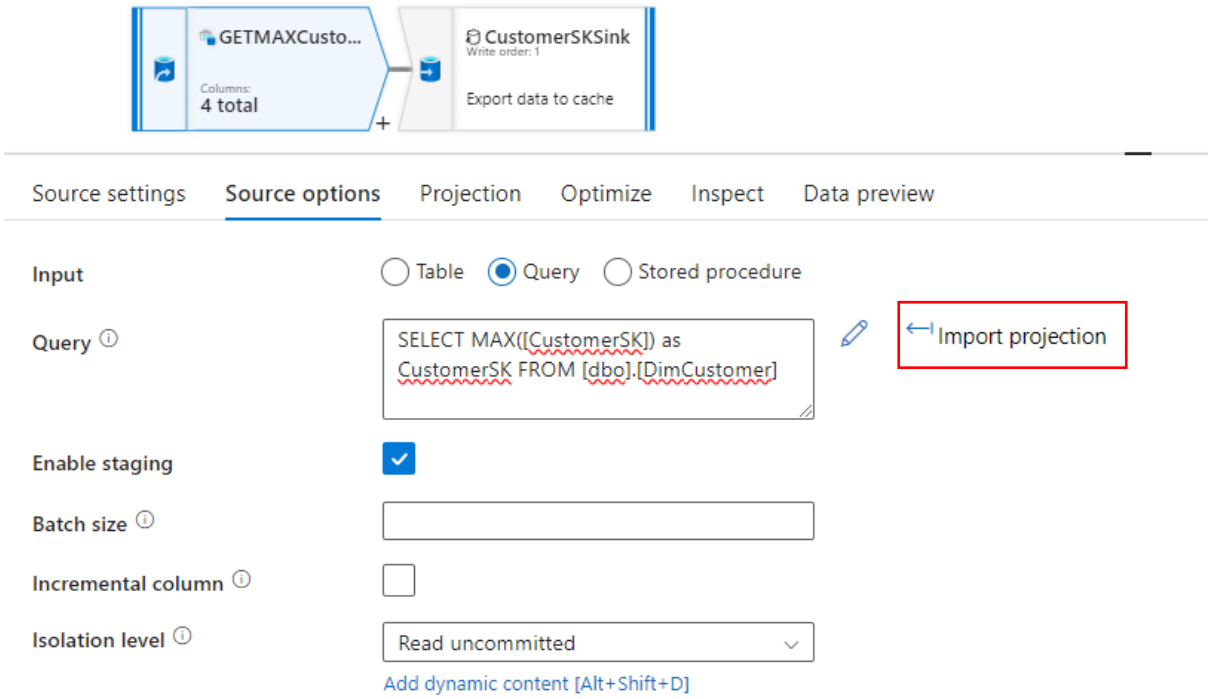
SELECT: This keyword is used to specify the columns that you want to retrieve data from.
MAX([CustomerSK]): This function calculates the maximum value of the column "CustomerSK". The MAX function is an aggregate function that returns the highest value in a set of values.
as CustomerSK: This part of the query renames the result of the MAX([CustomerSK]) calculation to "CustomerSK" in the output.
FROM [dbo].[DimCustomer]: This specifies the table from which the data is being retrieved. [dbo].[DimCustomer] indicates the schema (dbo) and table (DimCustomer) in the database.
In summary, this query retrieves the highest value of the "CustomerSK" column from the "DimCustomer" table in the specified database schema. This type of query is often used to find the maximum or minimum value in a column for analytical or reporting purposes.



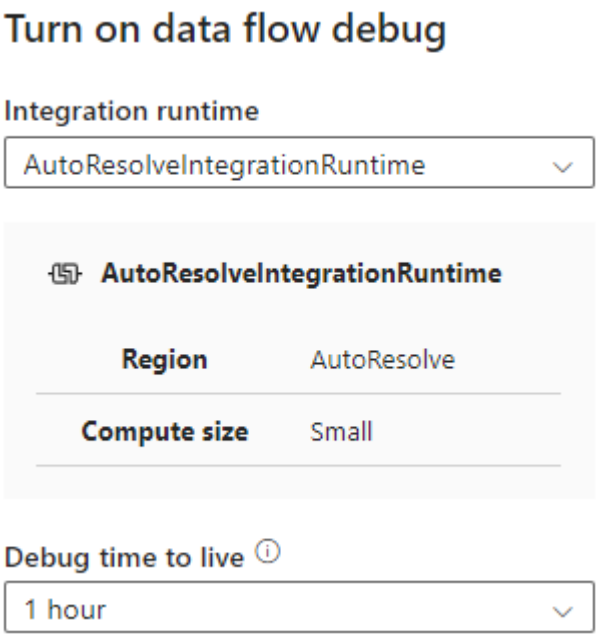
32. Now from our get max customer Stream we need to click on the plus icon and choose sink.
33. Then in the Sink first you need to give it a name and the incoming stream is get max customer SK. After that, you need to choose the Cache option.



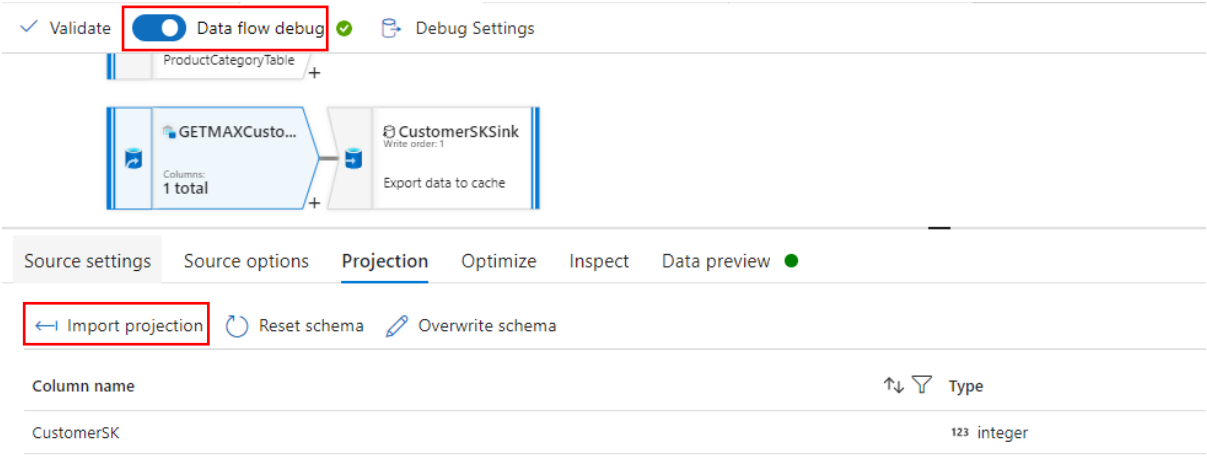
34. Then come back to get max customer stream and click on import projection.



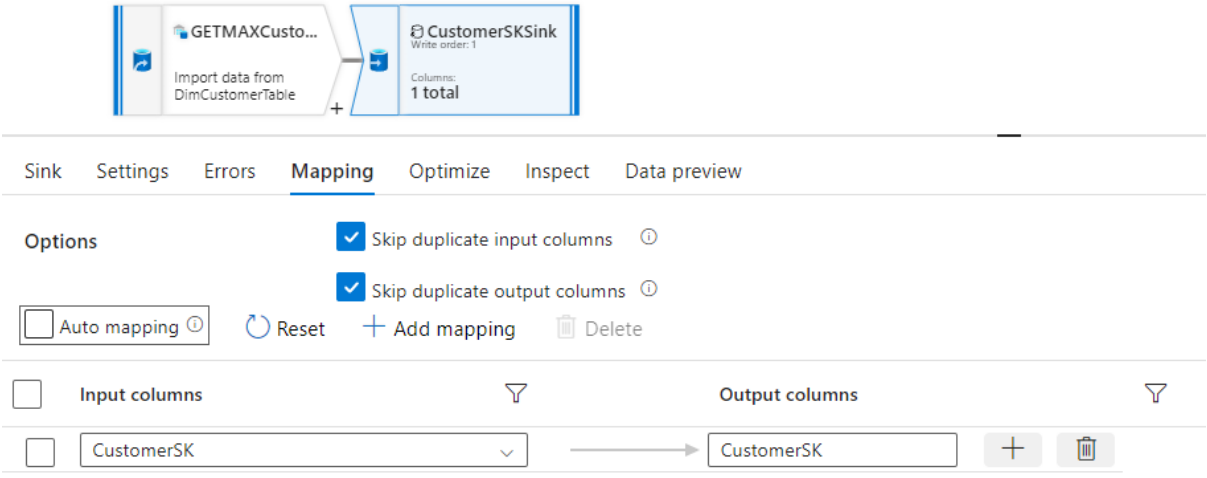
35. It is going to turn on data flow debug.



36. After that you need to come to projections and click on import projections and this will give you the below projection. Then you just need to turn off the data flow debug option.

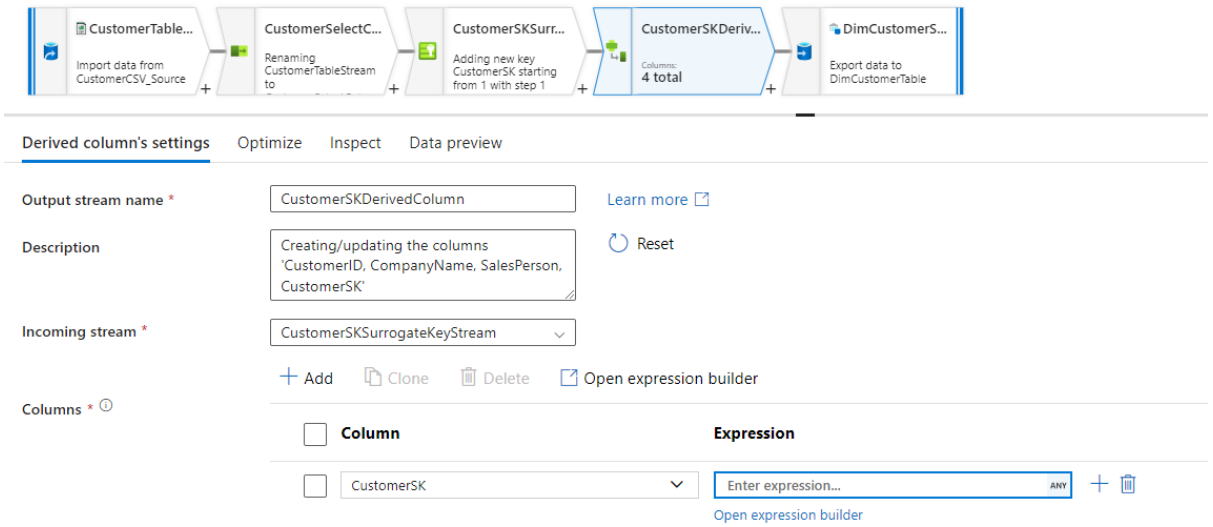


37. Then go back to sink and go to mappings then turn off the auto-mapping feature. Then you will see that latest mapping in place.

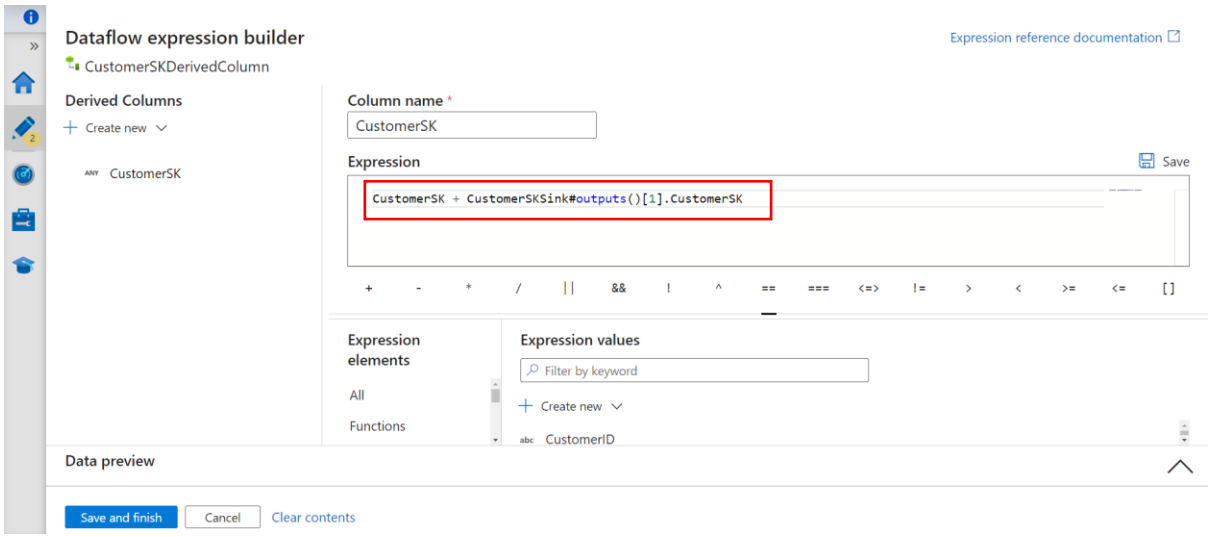


38. Then from your dataflow diagram you need select customer SK surrogate key and click on the plus icon from it. Then choose derived column.

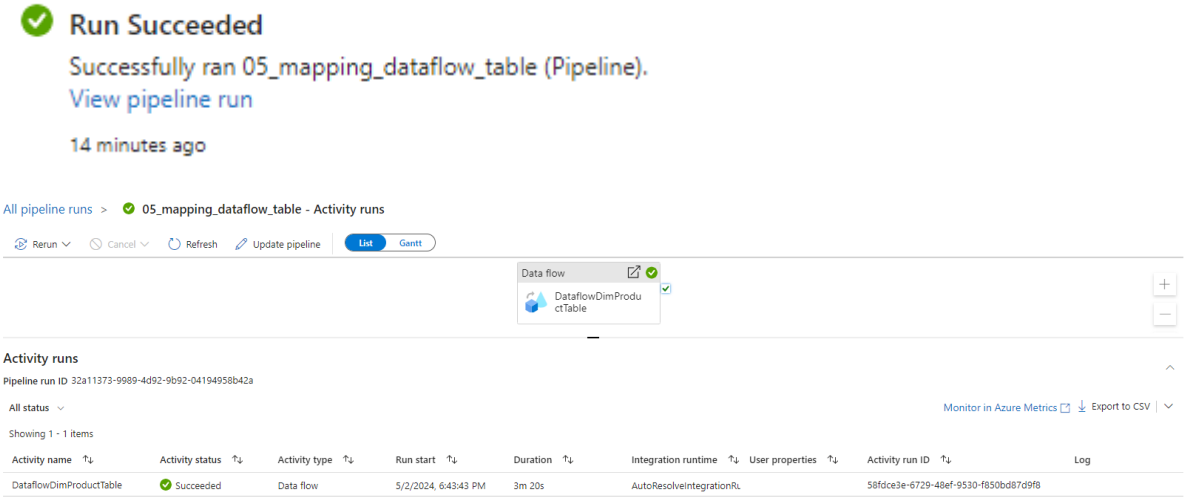
39. First give a name to your derived column then in the columns option choose Customer SK as your column and for the Expression you need to open the expression builder.



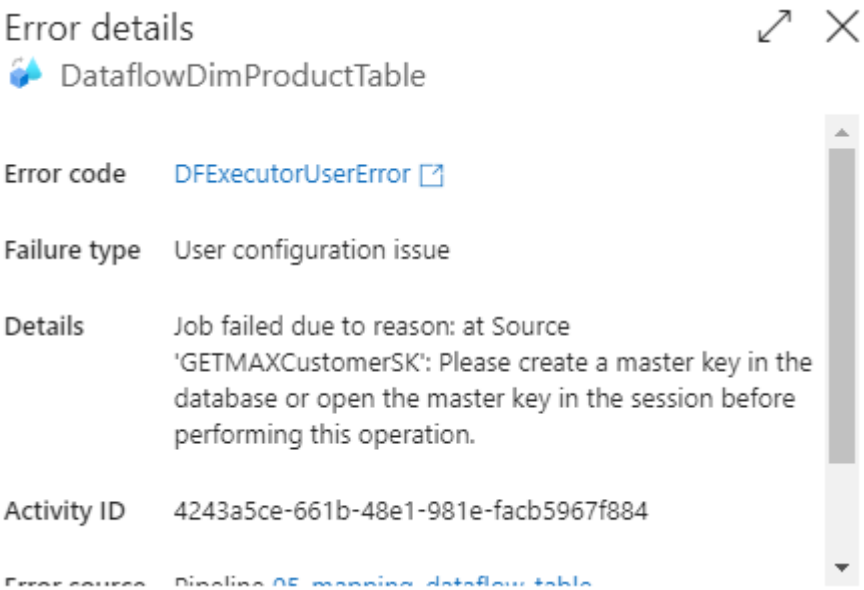
40. Here you have to write this expression as shown below and click on save and finish.



41. Once you are done with all this click on publish and then trigger your pipeline again.
42. Below you can see that our pipeline has ran successfully.



43. Now if in case you are getting an error like this below then you just have to create a password encryption key in you SSMS where you were creating tables all along.



44. You have to use the below statement to create an encryption key in your dedicated pooldb.

```
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'Password@1234';
```

45. Now you have to run the select command for the Customer table. This time you'll be able to see the data below customer id 468.