Data Migration Pipeline from On-Prem SQL Server to AzureBlobStorage

Business Requirement

The Client requires a solution to migrate data from on-premises SQL Server Database to Azure Blob Storage containers using Azure Data Factory (ADF). The solution must automate the daily data copying process, delete existing files in the target containers before each run, and ensure the data is stored in the designated containers as per the defined mapping.

Prerequisites

SQL Server:

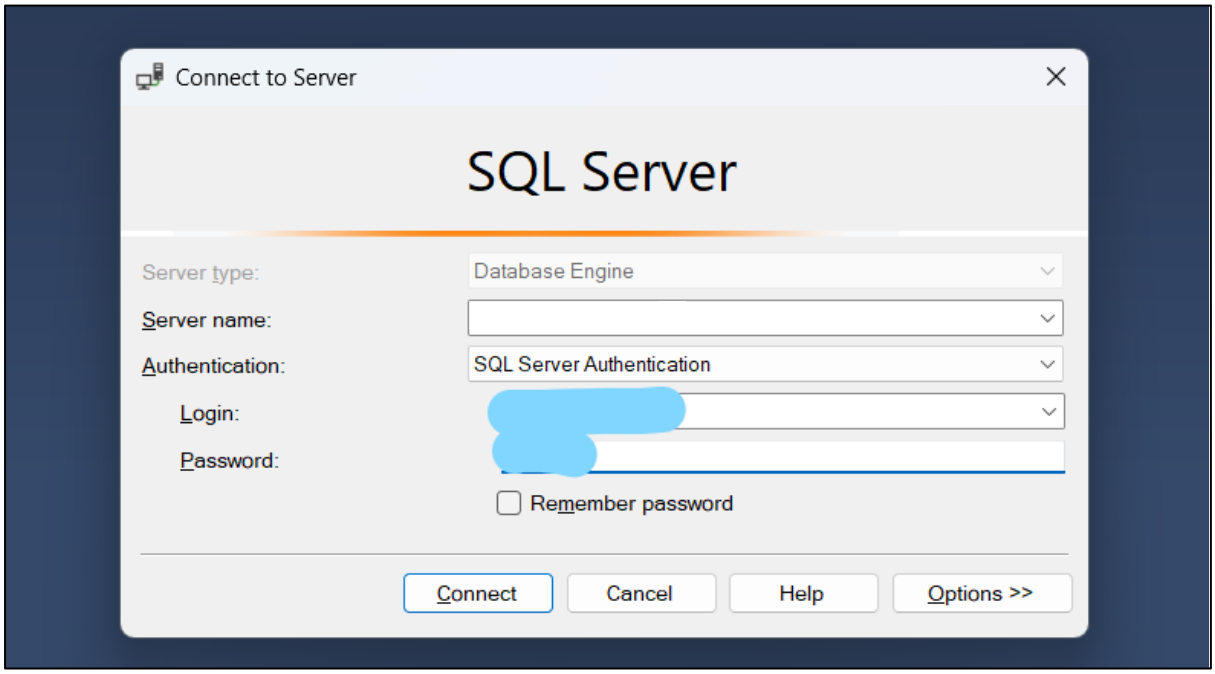
- Database: HealthcareDB
- Tables: Patients, Appointments, Hospital
- Authentication: SQL authentication
  - Username: give your desired username
  - Password: give password
- Integration Runtime: Self-Hosted Integration Runtime (IR) configured for on-premises SQL Server access.

Azure Blob Storage:

- Containers: patientblobstorage, appointmentsblobstorage, hospitalblobstorage
- Permissions: Contributor role on the storage account for read, write, and delete operations.

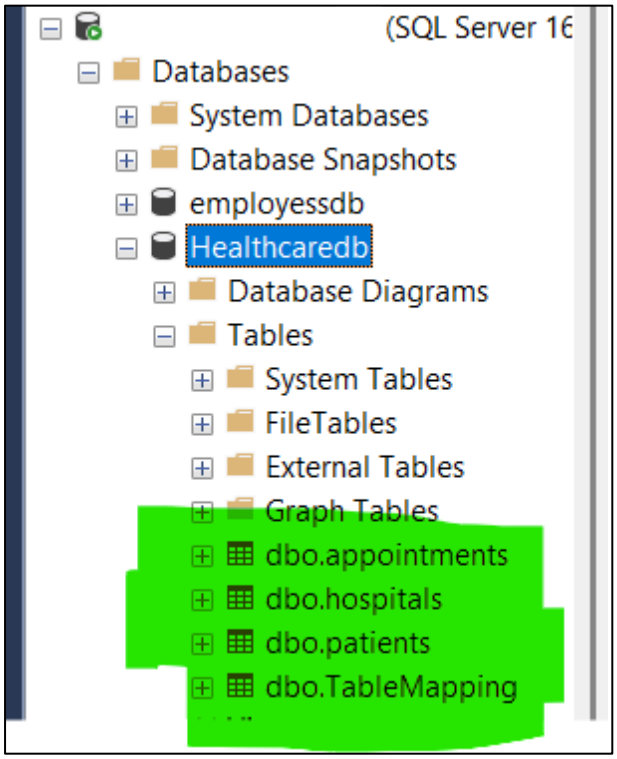
Implementation

1. In SQL server need to create server for self-hosted because we need to migrate data to Azure Storage.



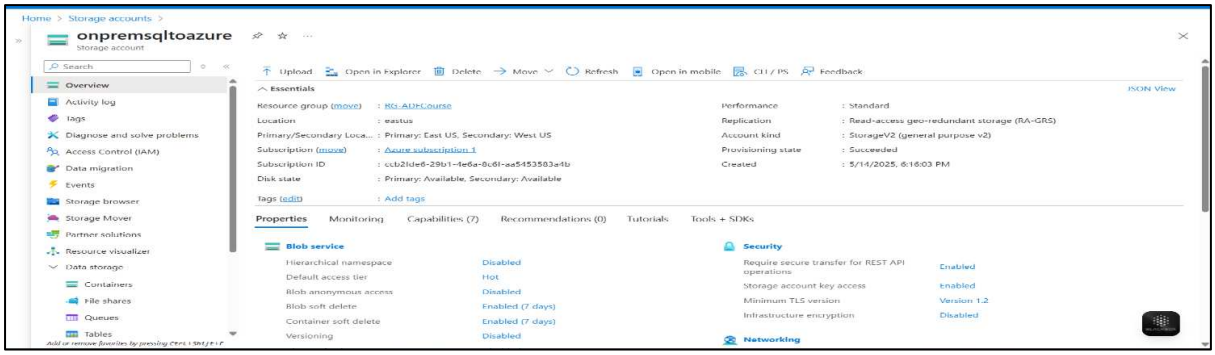
2. Create database and create tables in sql server
(patients,appoinments,hospitals,TabelMapping)

Note : TableMapping table used to map the tables with specific containers.

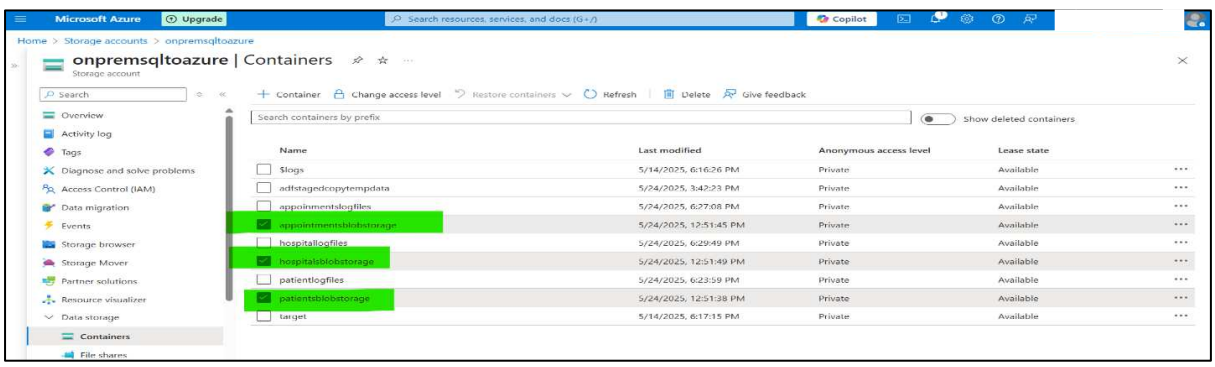


Check all the tables created successfully and fetching the data properly.

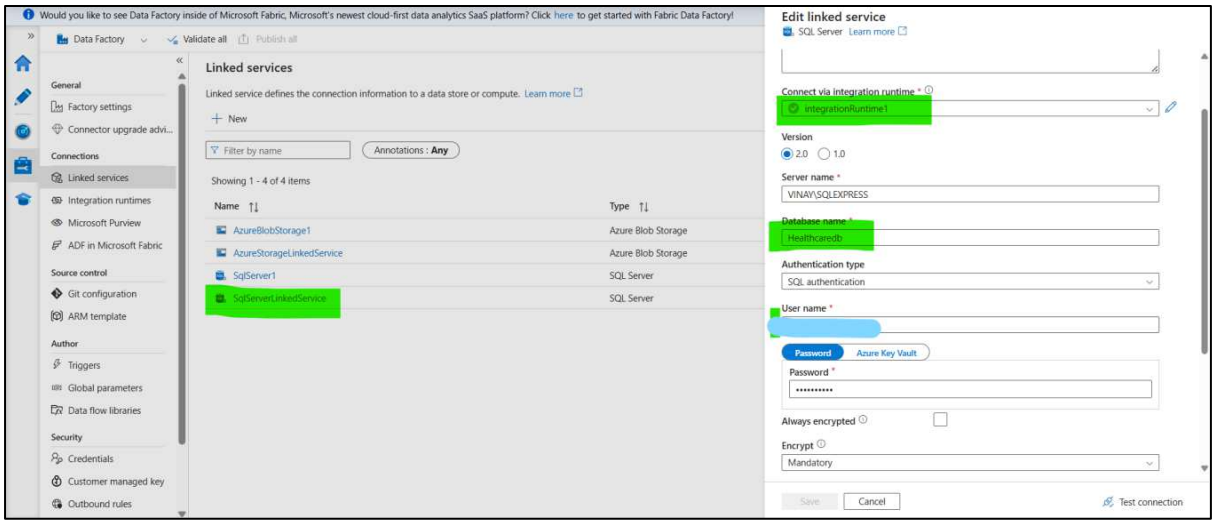
3. Then coming to Azure Data Factory create a Storage Account



4. Create three containers for each table to store data in particular container

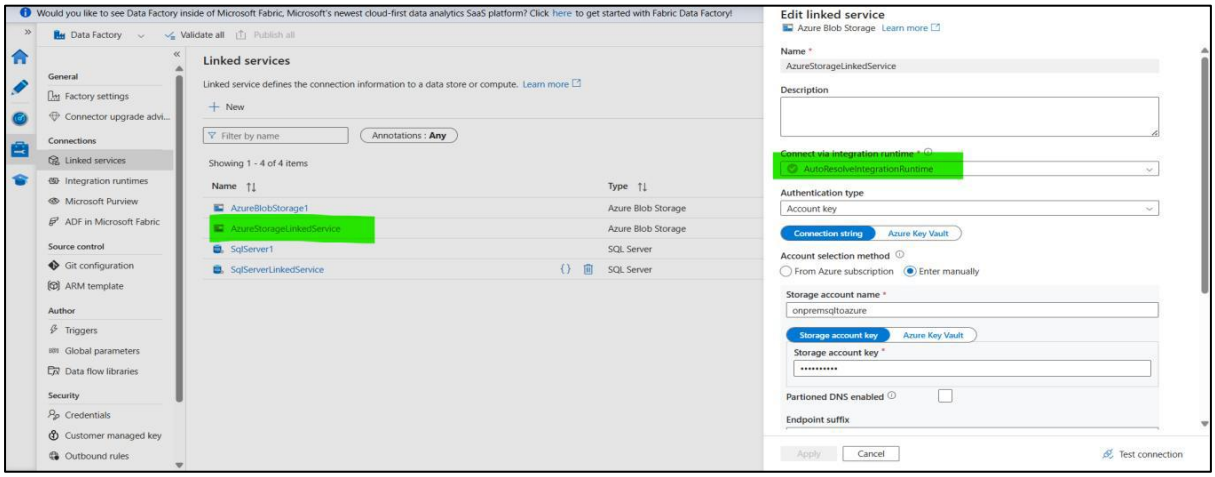


5. In Azure Data Factory studio, Create a Source Linked Service for On-Prem SQL Server



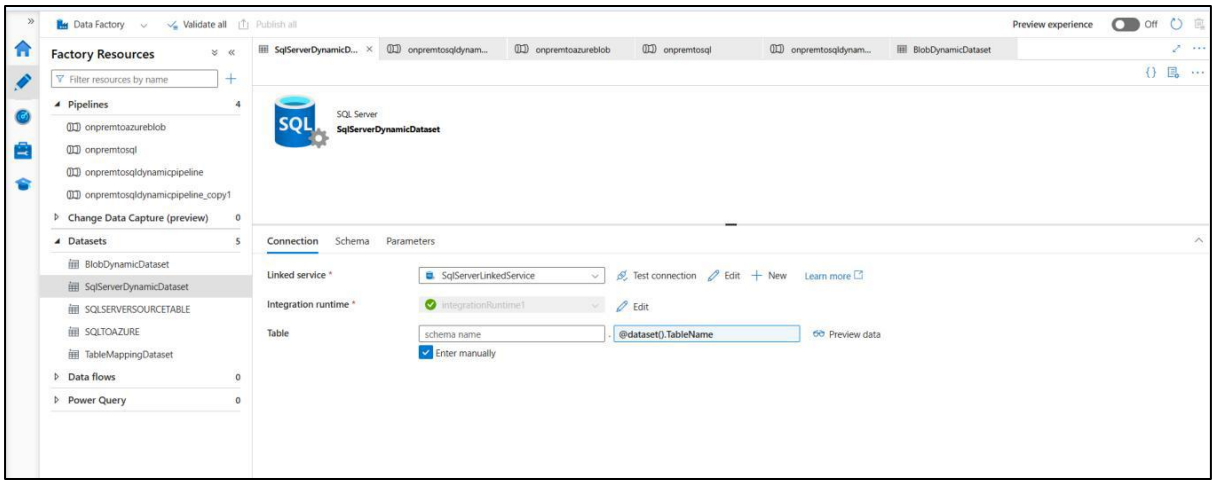
In Connect Via Integration runtime create self-hosted server because we need migrate data from on-prem to Azure Storage and give Database name, select SQL Authentication, give Username, Password which was there for on-prem SQL Server.

6. Create a Destination Azure Blob Storage we need to store data in Azure Blob Storage

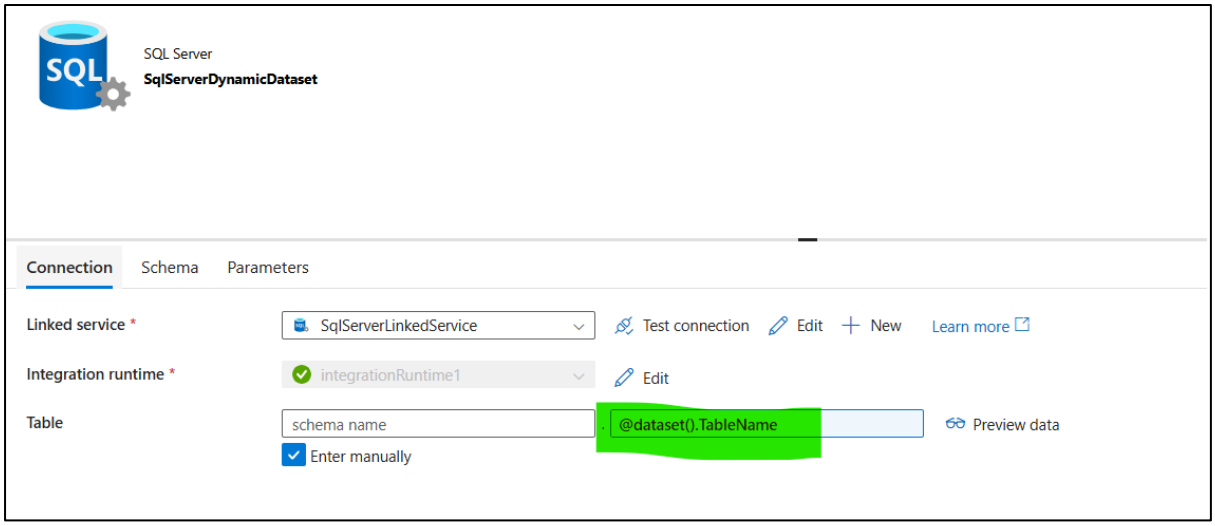


Note: After creating both Source and Destination Linked Service check the Test connection in order to check this works properly.

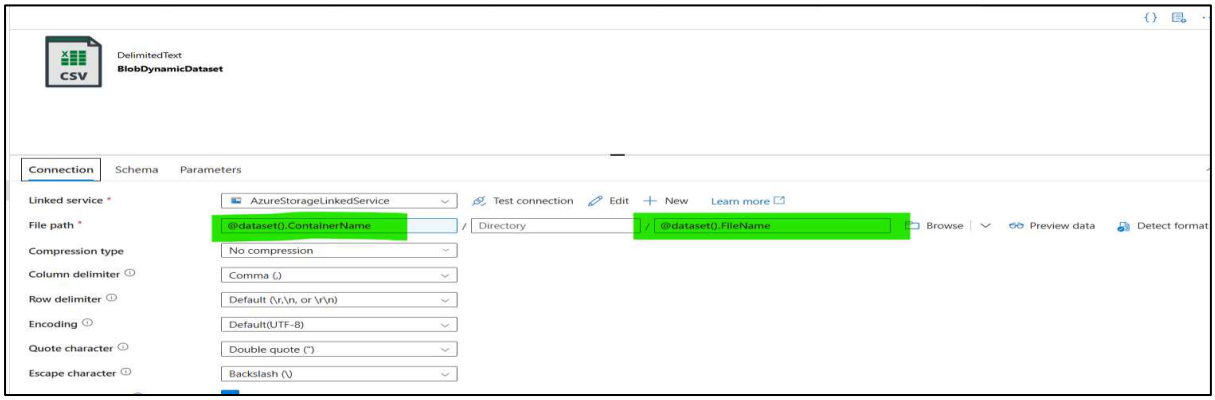
7. Goto Author create dataset for Source type of SQL Server



In dataset window create a parameter called table name, because we have three tables we need to set it dynamically

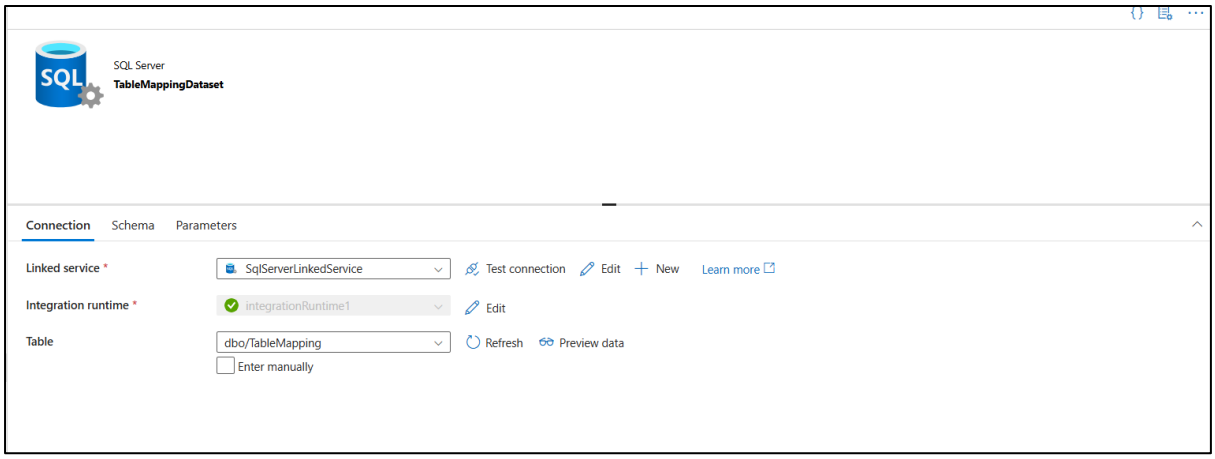


8. Create a Sink dataset for Destination type of Blob Storage



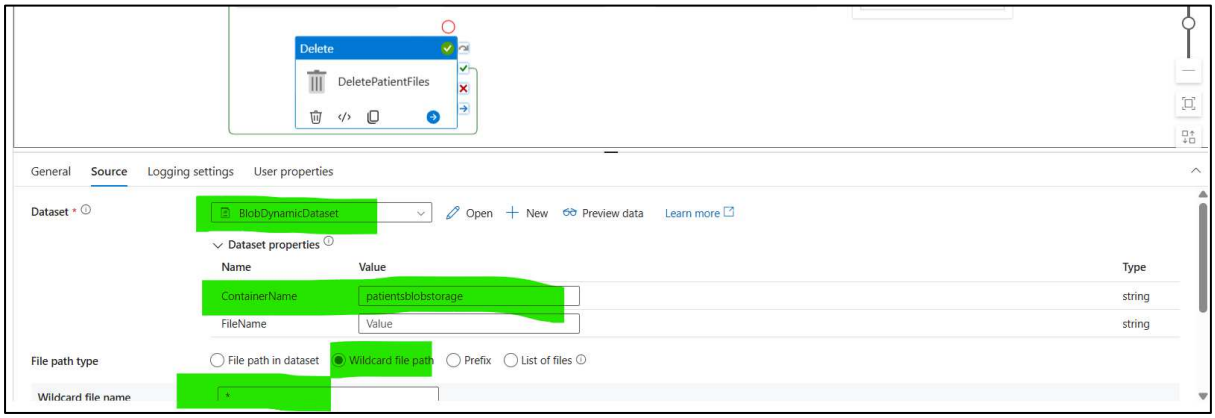
Create Two Parameter called ContainerName and FileName

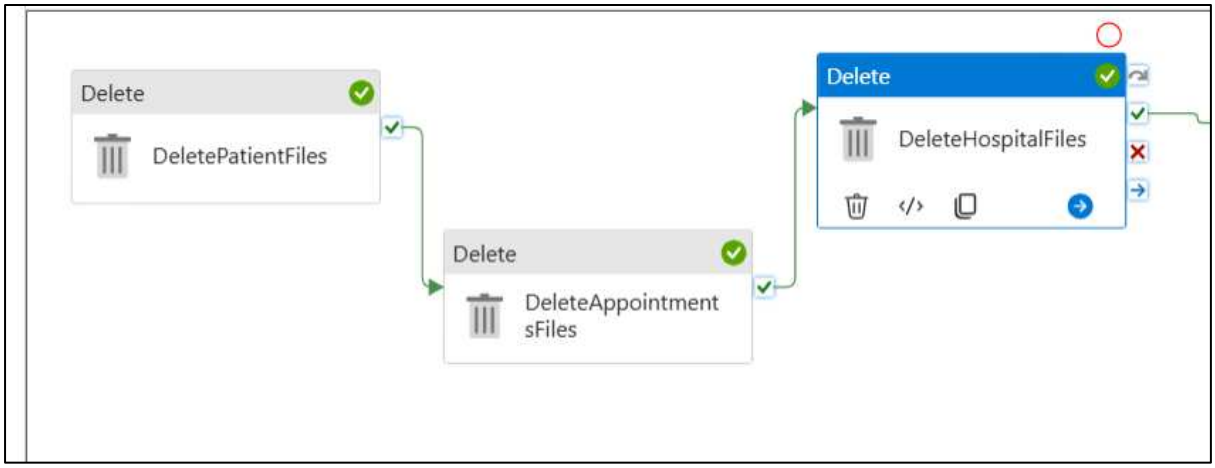
9. Create a dataset for TableMapping



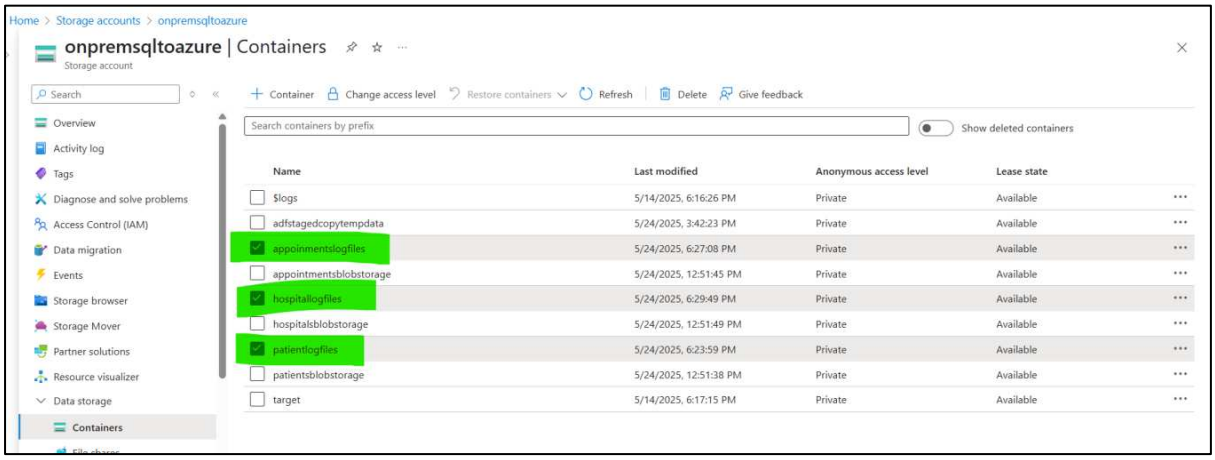
After creating both Linked Services and Datasets for Source and Destination please cross check the connection by clicking the Test Connection and preview data.

10. **Now, create a pipeline where it will dynamically migrate the data from on-prem to Azure Storage**
11. **Create a three Delete Activities because Since the Delete activity in ADF operates on a single dataset at a time, and each dataset instance can only point to one container at a time, you need separate activities to handle each container.**
12. **Create DeletePatientFiles, DeleteAppointmentsFiles, DeleteHospitalFiles for each storage.**
13. **In every particular delete activity, in source select dataset. Then in container name specify every container name for example, for patient give patinetblobstorage, for appointment give appoinmentblobstorage, for hospital give hospitalblobstorage.**
14. **In every delete activity select Wildcard file path and give wildcard name as (*).**





15. **Connect every activity with each other**
16. **In every container select enable logging because we need to log files for each activity and for that create a three log files in Azure container so that it can store in particular log storage.**



In this container whenever a delete activity will occur that deleted log files will store in this particular container.

17. Add Lookup Activity to canvas

Settings Tab:

- Source Dataset: Select TableMappingDataset.
- Use Query: Select Query.
- Query: SELECT TableName, ContainerName FROM TableMapping
- First row only: Uncheck (to get all rows).

18. Add ForEach Activity:

A. Drag a ForEach activity to the canvas and connect it to the Lookup activity (green arrow from GetTableList to ForEach).

B. Configure

- Name: IterateTables
- Settings Tab:
- Items: @activity('GetTableList').output.value (this references the Lookup output, an array of table-container pairs).
- Sequential: Check (to process tables one at a time).

19. Add Copy Data Activity Inside ForEach:

A. Inside the ForEach activity, drag a Copy Data activity.

B. Configure: Name: CopyTableToBlob

- Source Tab:
- Source Dataset: Select SqlServerDynamicDataset.
- Dataset Parameters:
- TableName: @item().TableName (uses the current table name from the ForEach loop).

20. Sink Tab:

A. Sink Dataset: Select BlobDynamicDataset.

B. Dataset Parameters:

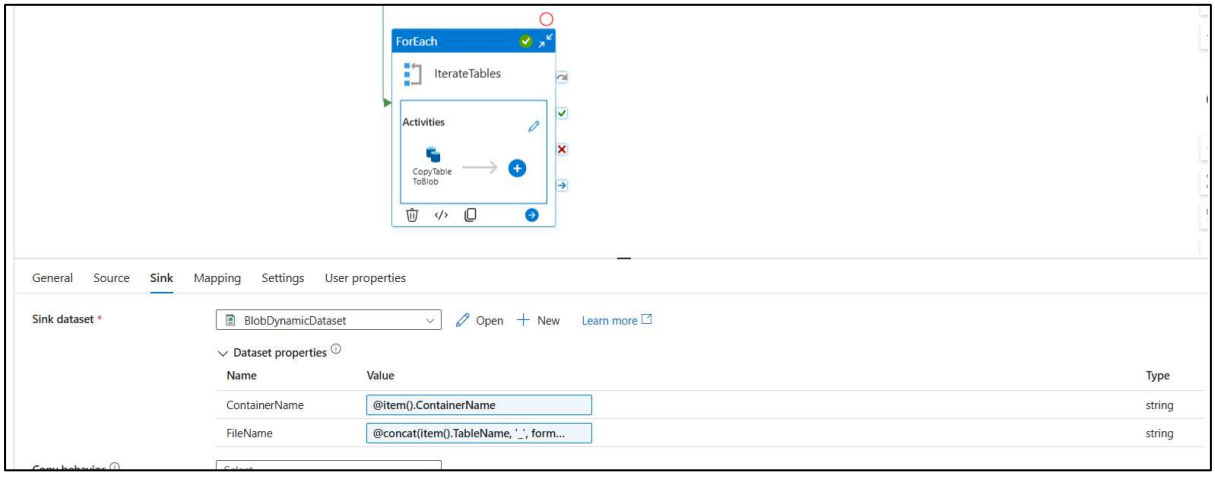
- ContainerName: @item().ContainerName (uses the current container name from the ForEach loop).
- FileName:@concat(item().TableName,'_', formatDateTime(utcNow(),'yyyyMMdd_HHmmss'), '.csv')
- fileformat

    a. Settings Tab

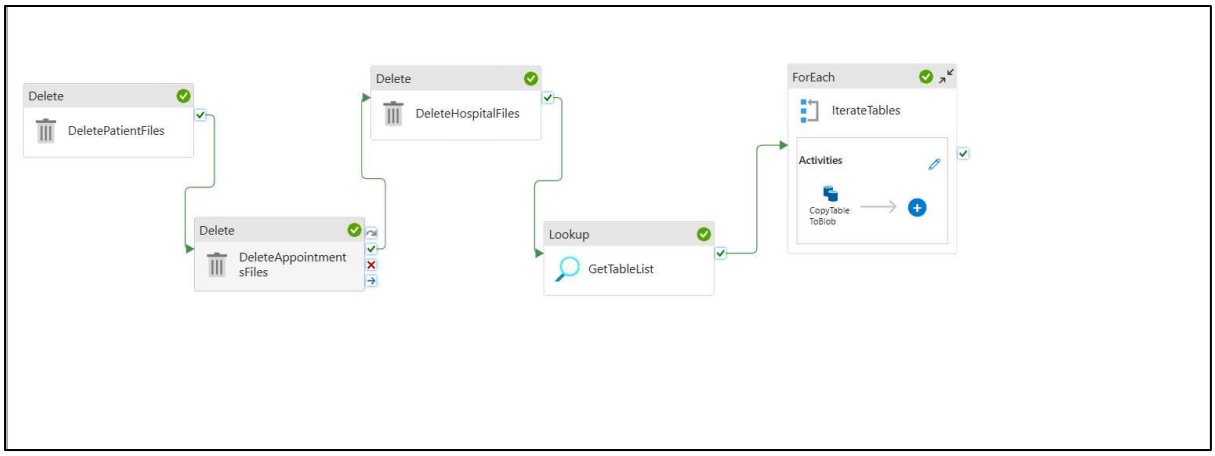
    b. Enable staging: Check (for better performance with self-hosted IR).

    c. Staging storage account: Select your Blob Storage account via AzureStorageLinkedService.

    d. Staging folder: Specify a folder (e.g., staging).



21. Final pipeline is here



22. Save the pipeline and click on publish all and then Debug check for container whether they are working as per requirement.

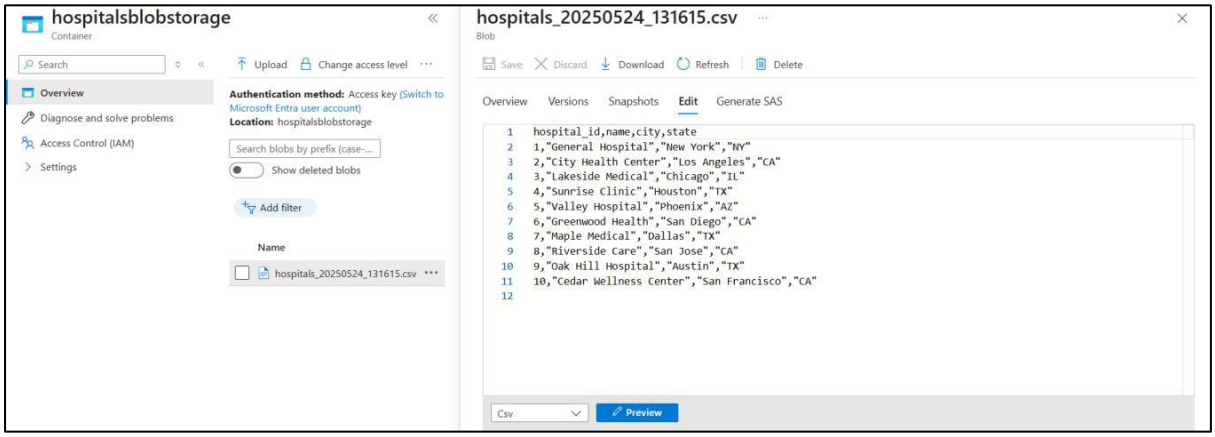
23. As mentioned in requirement that we need to to delete the data at mentioned intervals so for that we need to add trigger

24. Open CopyTableToBlobPipeline

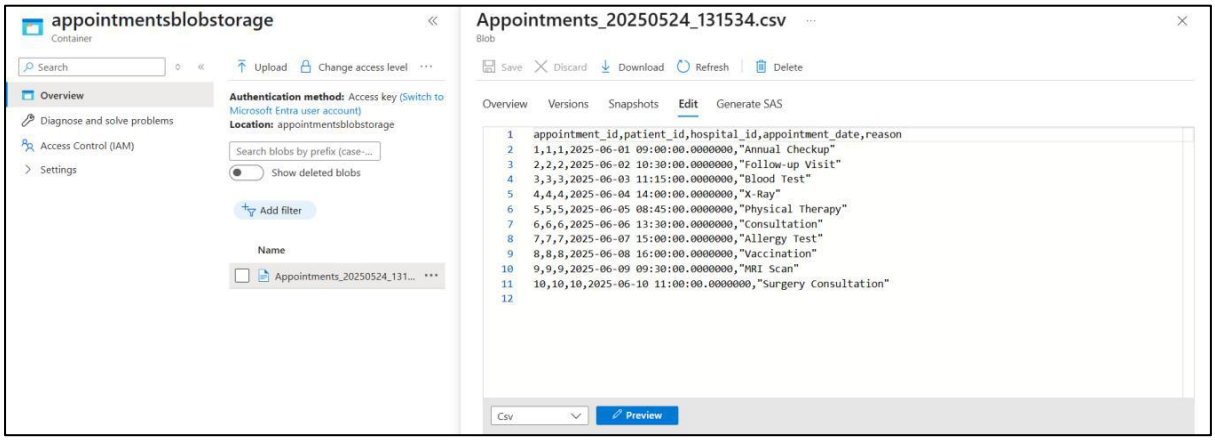
- Go to the Trigger tab and click New/Edit.
- In the popup, click Choose trigger > New.
- Configure the trigger:
- Name: DailyTrigger
- Type: Tumbling Window Trigger
- Start Date: 2025-05-25T00:00:00Z (May 25, 2025, 00:00 UTC, which is 05:30 AM IST) just for example
- Recurrence: Every 1 Day (set Interval to 1, Frequency to Day)
- End Date: Optional (leave blank for no end date, or set a future date like 2026-05-25T00:00:00Z)
- Window Size: 1 Day (default)
- Advanced Settings:
- Time Zone: UTC (default). Since the pipeline's file names use IST (adjusted with addMinutes(utcNow(), 330)), the trigger time will be in UTC, but the file names will reflect IST.
- Activated: Yes
- Click OK to create the trigger. And then Click Publish All to save and activate the trigger.

Containers

Patient Container



Hospital Container



Appointment Container

Validate Output:

1. In Azure Portal, navigate to your Blob Storage account.

2. Before the run, note the existing files in each container.

3. After the run:

- Confirm the old files are deleted.
- Check for new files with the current timestamp (e.g., Patients_20250524_181200.csv for a run at 06:12 PM IST).
- Verify the files are still in CSV format with headers.

4. Verify that each day

- Old files are deleted.
- New files are created with the current date and time (e.g., Patients_20250525_000000.csv).

Meets Requirement

- **Delete Files Daily**: The Delete activities (DeletePatientFiles, DeleteAppointmentsFiles, DeleteHospitalFiles) remove all files in the containers at the start of each pipeline run, ensuring the containers only contain the current day's data.
- **New Data Daily**: The pipeline copies fresh data from the SQL Server tables (Patients, Appointments, Hospital) to the containers each day, with file names reflecting the current date and time (e.g., Patients_20250525_000000.csv).
- **Scheduled Runs**: The Tumbling Window Trigger runs the pipeline daily, automating the process without manual intervention.
- **CSV Format**: The files remain in CSV format, as confirmed by the BlobDynamicDataset settings.