Streamlining Analytics Pipeline with Azure Data Services

CLOUD COMPUTING – PROJECT




Hrushika Papani (16354873)

Viswanth Tammana (16352538)

Chandra Haas (16355595)

Dheeraj Golla (16348914)

10th May, 2024

Roles and Contribution:

- Hrushika had done the data transformation by creating dataflows in the azure data factory. Performed different transformation activities like Source, filter, select, pivot, lookup and sink, then created ADF pipeline for the above transformation.
- Viswa, ingested the data from HTTP(ECDC website).Created the control flow activities and the linked service parameters. Monitored pipelines and integration runtimes, created alerts for failures or successes, and visualized metrics using plots and Kusto Query Language in Azure.
- ChandraHaas had ingested the population data to the datalake using data factory. Performed copy activity, created linked services and datasets and pipeline. Worked on control flow activities like Validation, Get Metadata, If Condition, Web, Delete, Fail. Created trigger to ingest the data to the data lake.
- Dheeraj manages the integration of transformed data, including cases and deaths, hospital admissions, and testing data, into an Azure SQL database. He oversees the setup of SQL tables, data pipelines, and copy activities to enable efficient reporting and analysis.

Introduction and Purpose:

- Real-world data engineering workflow on Microsoft Azure, specifically targeting COVID-19 data analysis due to the pandemic's extensive data generation.
- Address the need for efficient COVID-19 data handling by enabling organizations to collect, integrate, and analyze data effectively through an end-to-end data pipeline on Azure
- Empower data engineers with scalable pipeline development skills, integrating CI/CD for automation and reliability, aiding real-time pandemic response decisions.

Background and Motive:

- The COVID-19 pandemic has inundated organizations with vast amounts of data from diverse sources, necessitating robust data engineering solutions to efficiently manage and analyze this information.
- Leveraging Microsoft Azure's suite of tools enables the development of a comprehensive data pipeline for COVID-19 data analysis, addressing the need for seamless data transfer, advanced analytics, and automation.
- By building an end-to-end data pipeline utilizing Azure Data Factory, Azure Data Lake Storage, and Databricks, the project aims to empower organizations with timely insights into COVID-19 data, facilitating informed decision-making for response and recovery strategies.
- This project serves to equip data engineers with the necessary skills to construct and manage scalable data pipelines tailored for COVID-19 data analysis, ultimately enabling organizations to leverage data-driven approaches for more effective pandemic response efforts.

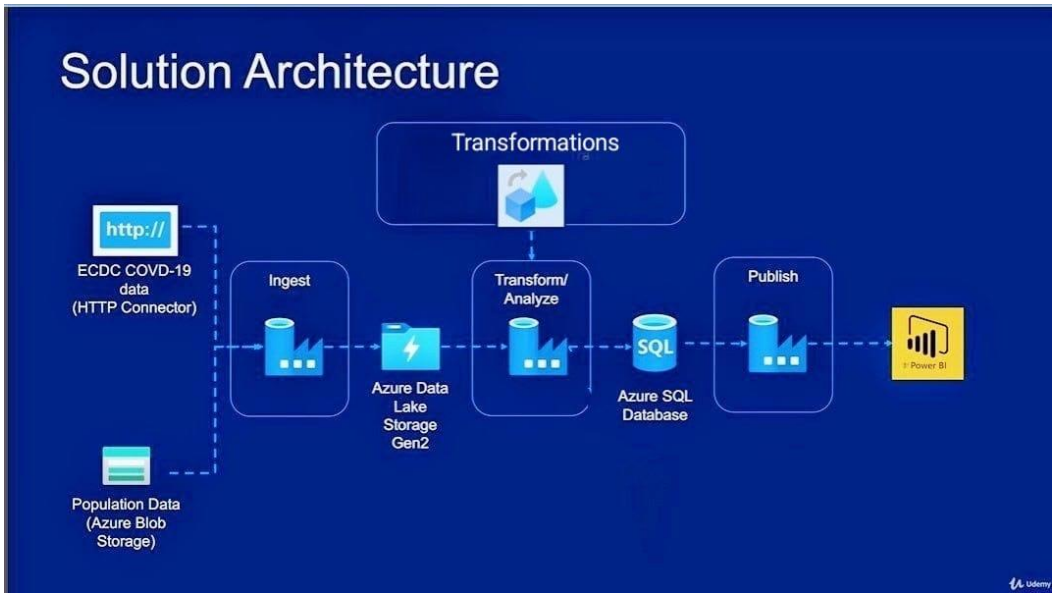
Project Implementation:

The data comprises population data ranging period from 2008 to 2019.

Data is based on country code and age (0-max).

Step 1 : We need to setup the environment

Solution Architecture



It includes Creating an Azure Data factory

In the home we could see “create a resource” click on it and create data factory

Create Data Factory ...

Changes on this step may reset later selections you have made. Review all options prior to deployment.

Basics | Git configuration | Networking | Advanced | Tags | Review + create

One-click to create data factory with sample pipeline and datasets. Try it

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * : Azure subscription 1

Resource group * : (New) covid-reporting-rg4

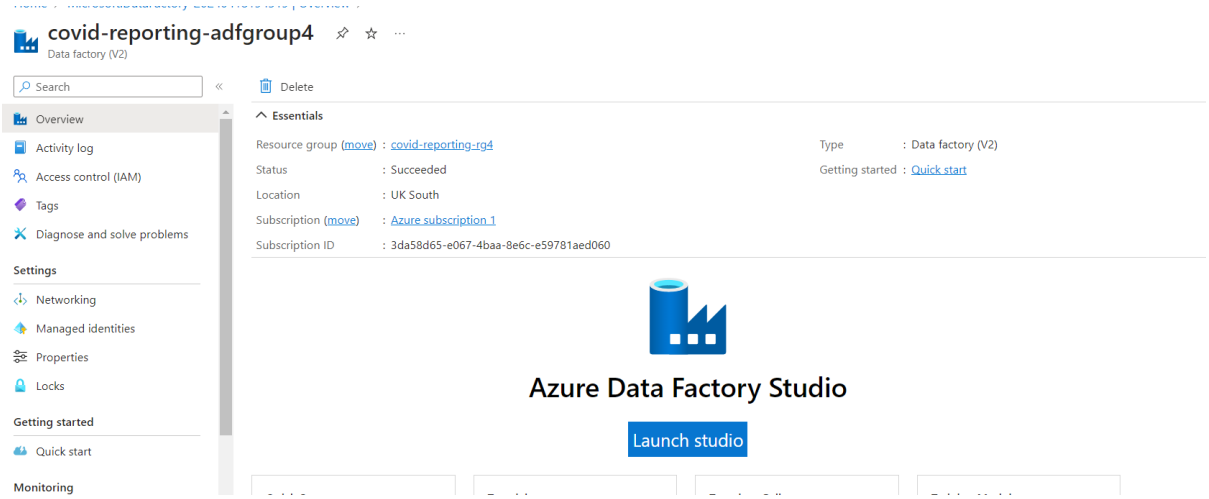
Create new

Instance details

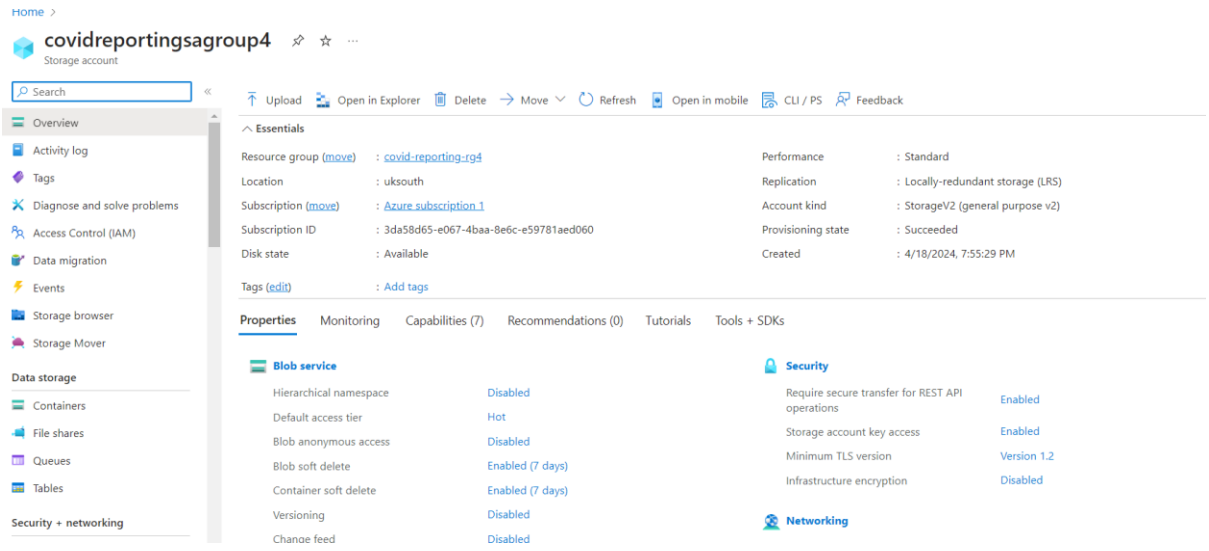
Name * : covid-reporting-adfgroup4

Region * : UK South

Version * : V2



Step 2: Also create azure storage account:

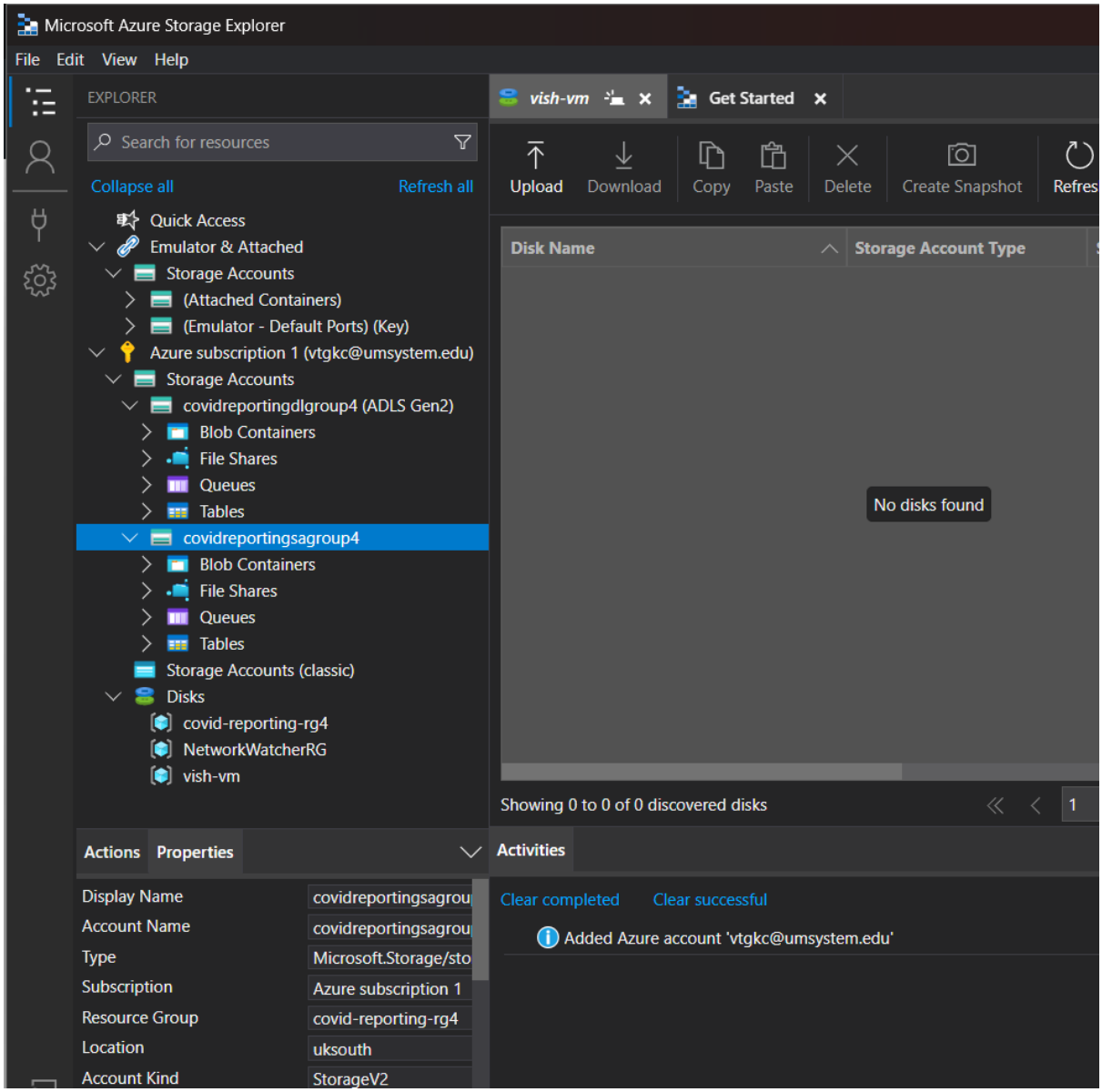


Step 3: Installing Azure Storage Explorer

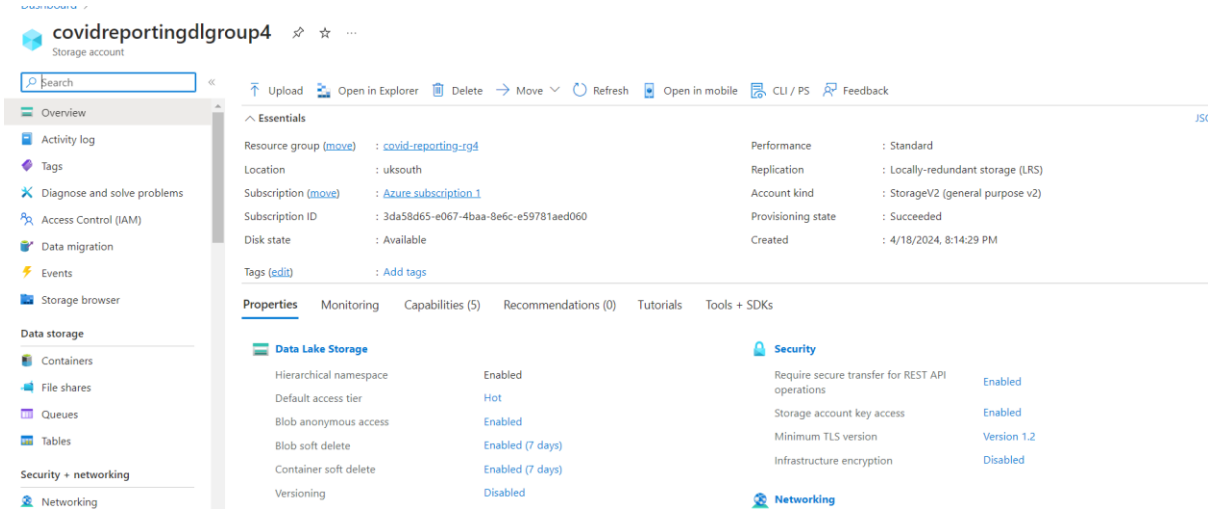
From the below link we can install the azure storage explorer.

https://azure.microsoft.com/en-us/products/storage/storage-explorer

After successful installation launch the explorer.



Step 4: Now creating Azure data lake storage:

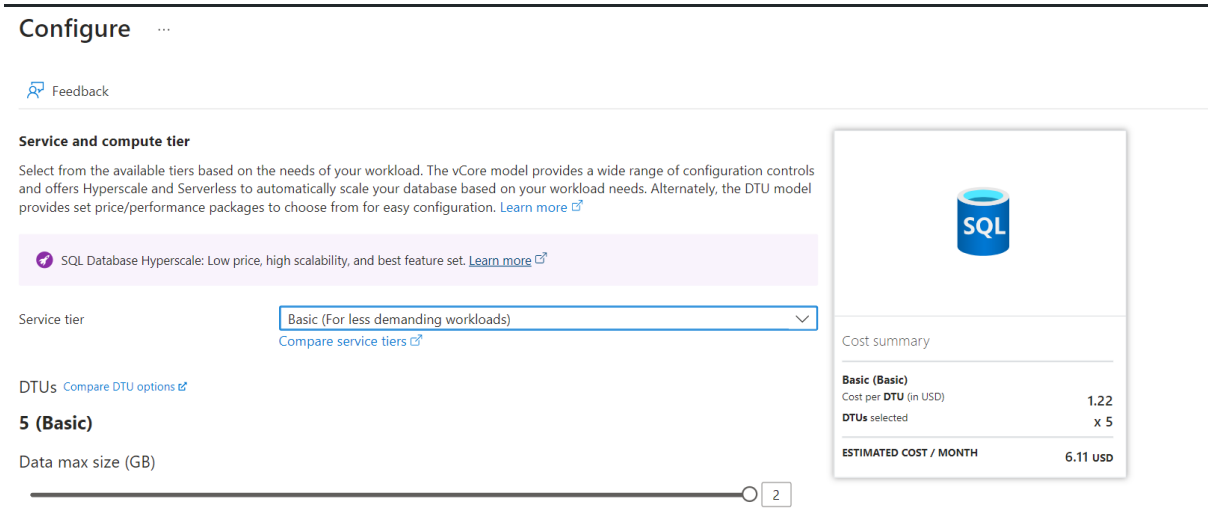


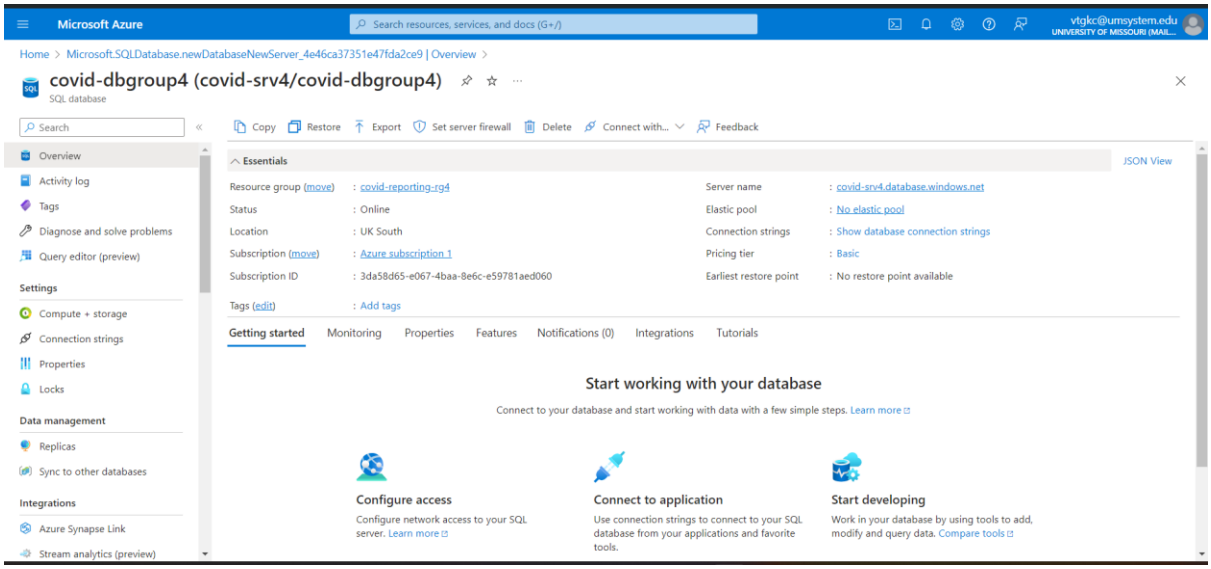
Creating Azure SQL Database

While doing so it allows us to configure Username and password.

Username: admingroup4

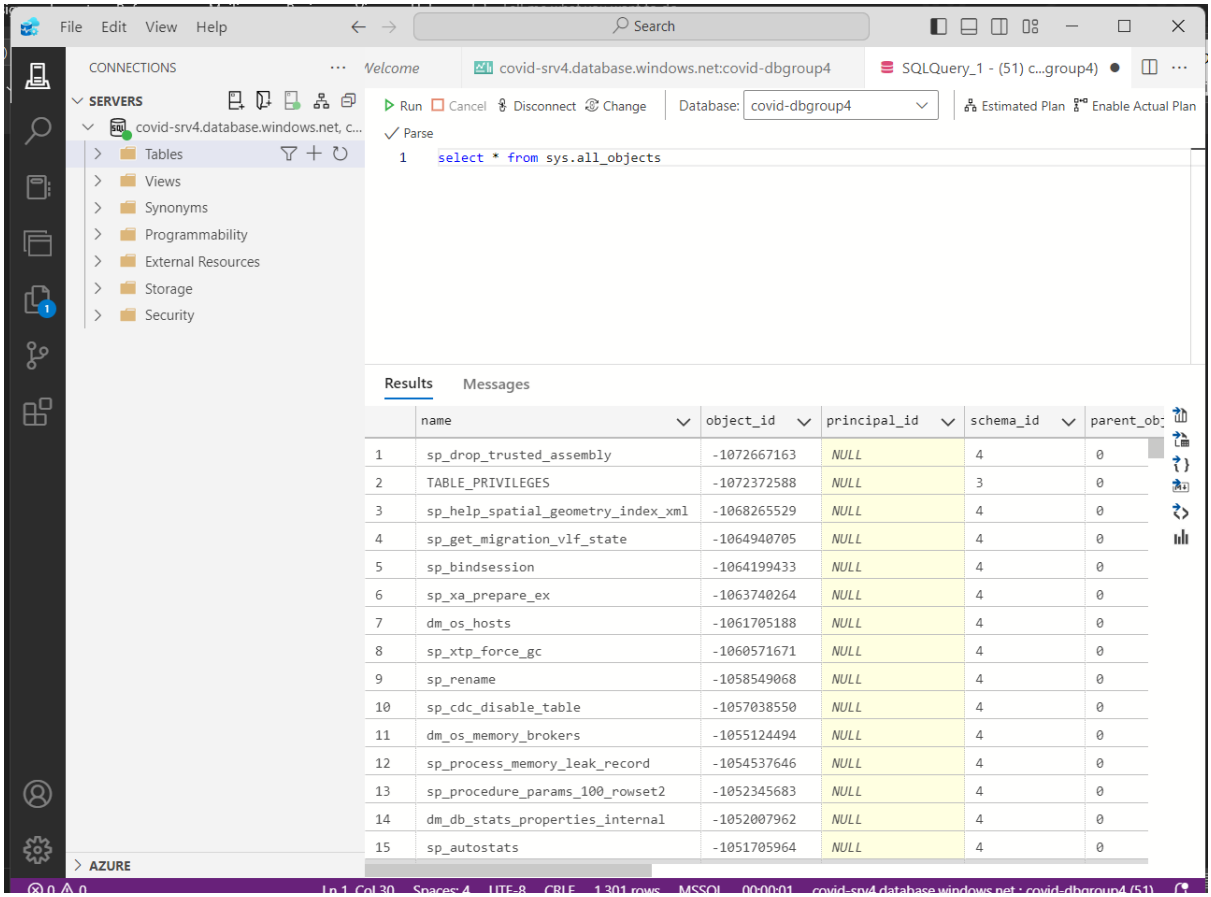
Pwd: CCgroup4





Installed Azure Data Studio and connected database:

Step 5: Now we need to link the database to Azure Date Studio



Step 6: Ingest population data (blob storage) to datalake using Azure Data factory.

Copy activity, Create Linked services, datasets, Execute Pipeline

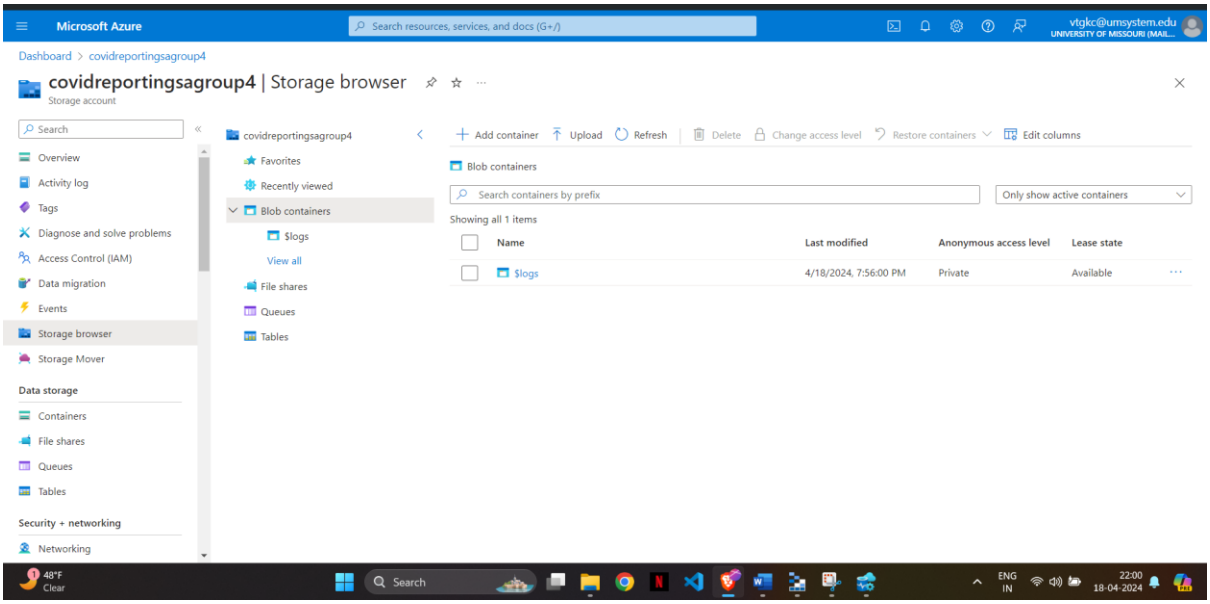
Control flow activities (Validation, if condition, web, get metadata, delete)

Triggers (to automate the workflow for the pipelines)

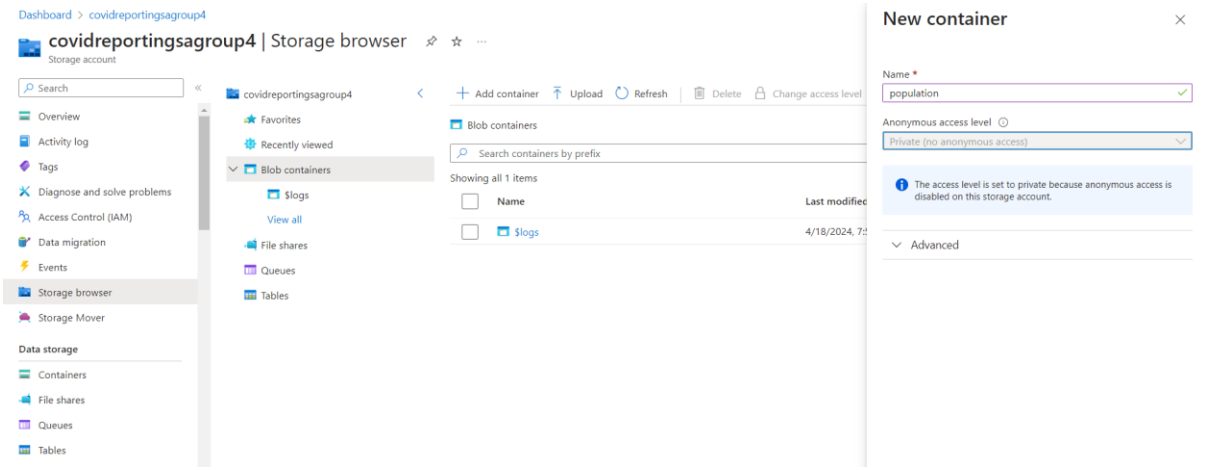
COPY ACTIVITY

Ingest population data (blob storage) to datalake using Azure Data factory.

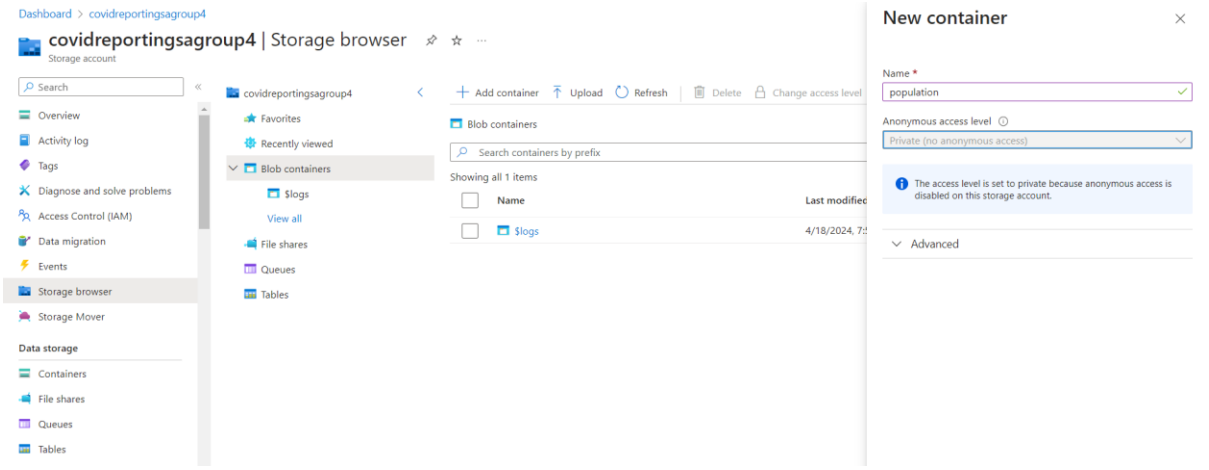
1. Add container in the blob storage > storage account

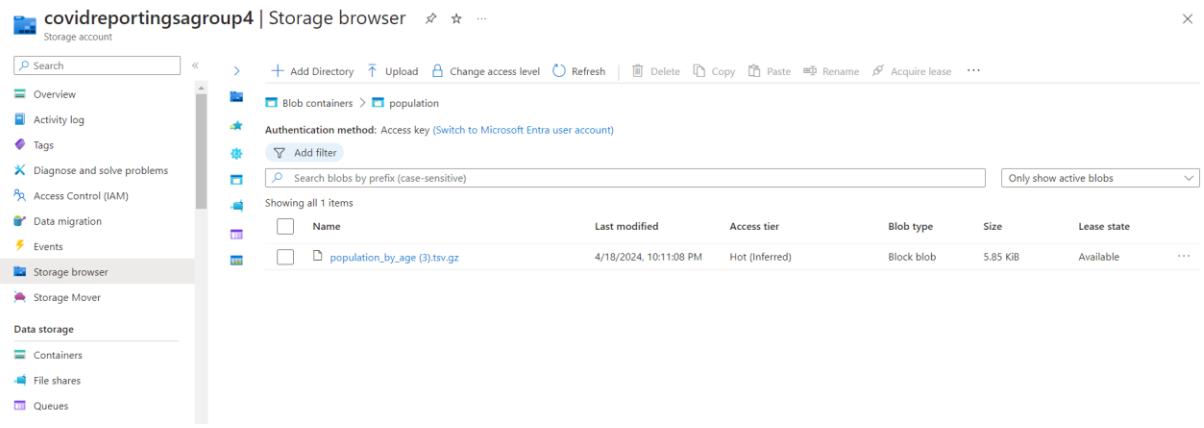


2. Create container



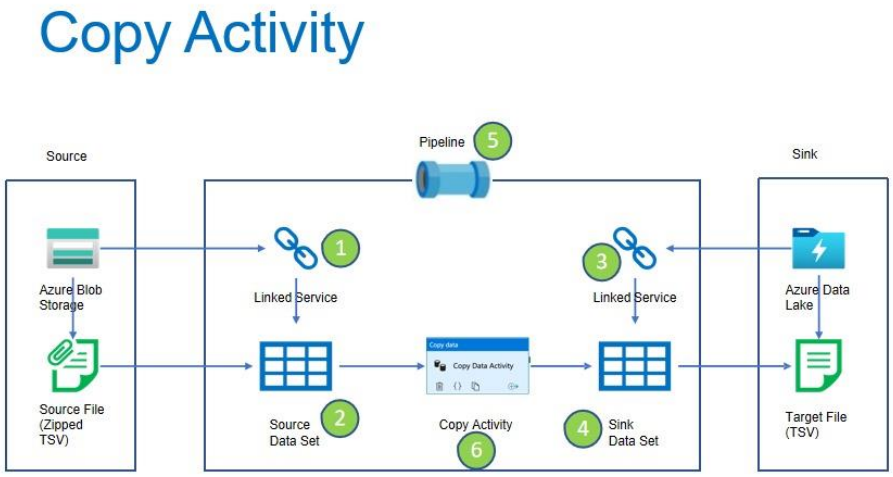
3. Upload files





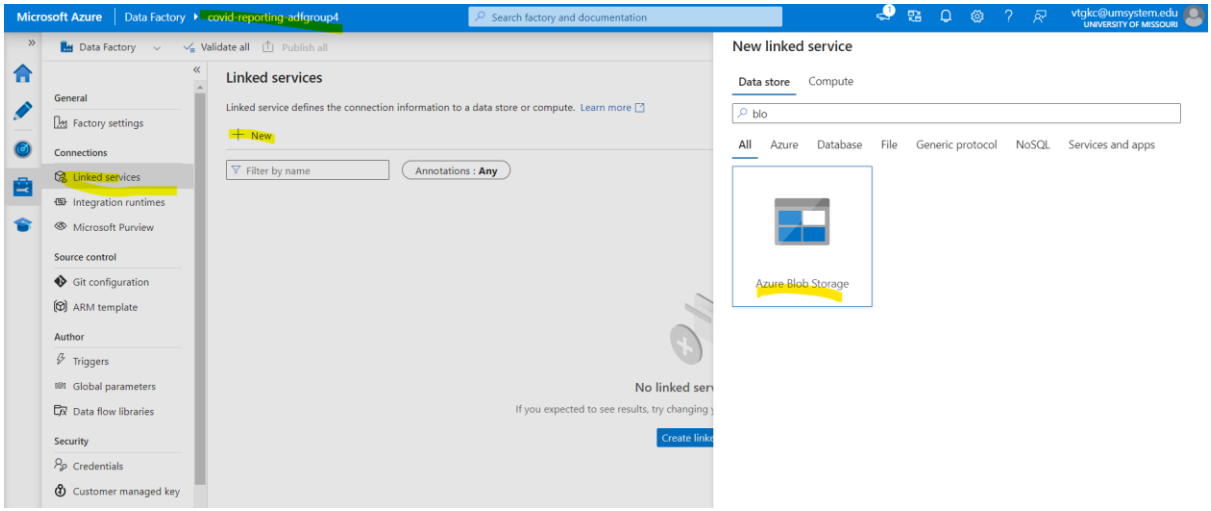
4. Create a raw container in the data lake.

   Naming standards

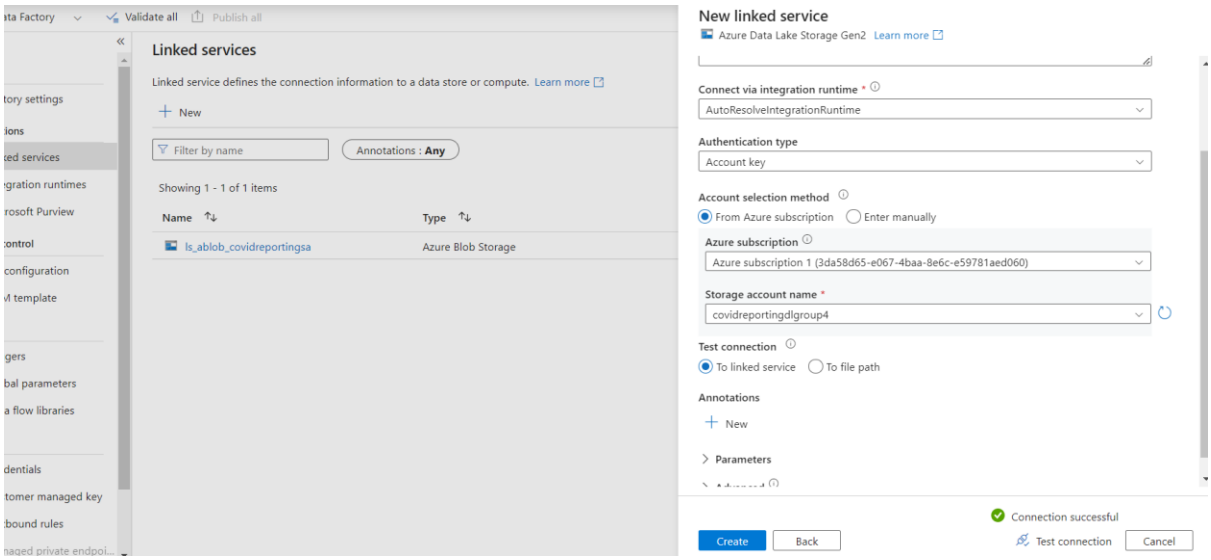


Linked services and datasets

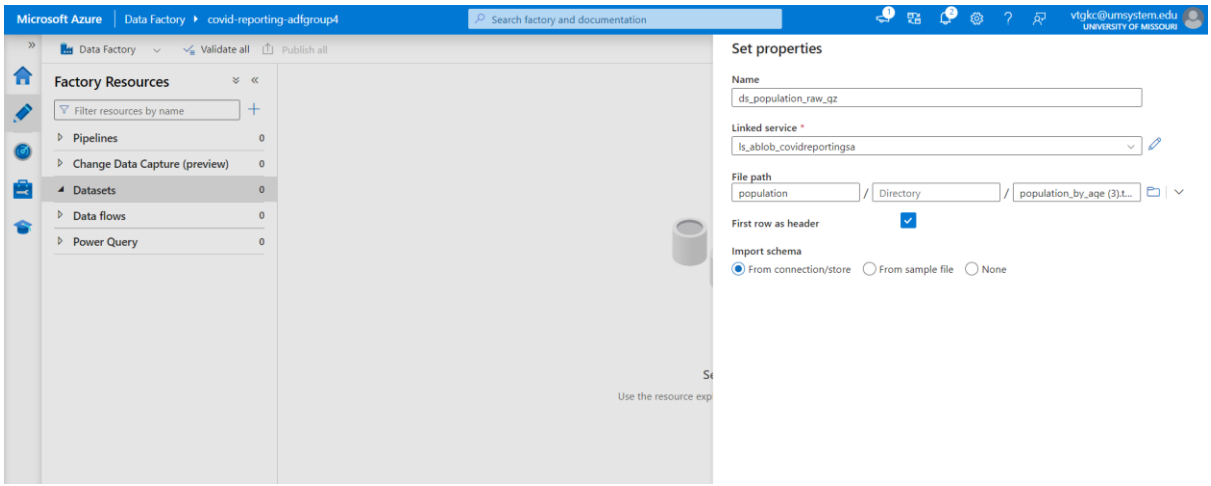
Create linked services for storage account

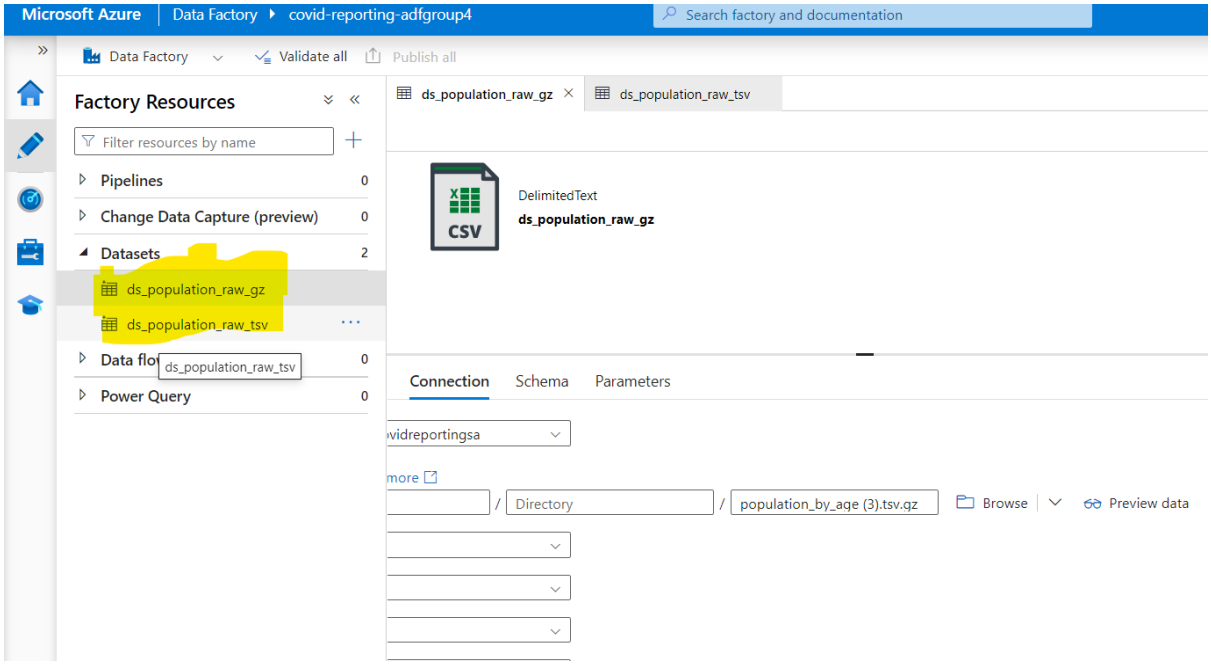


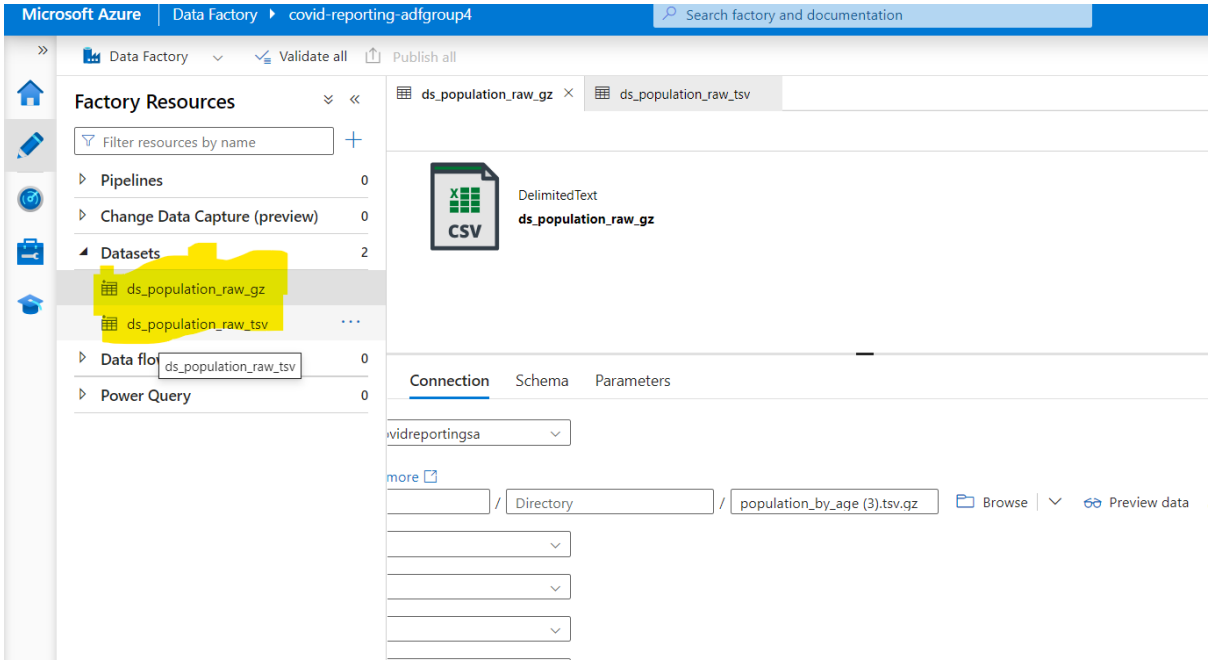
Create linked services for datalake storage gen 2



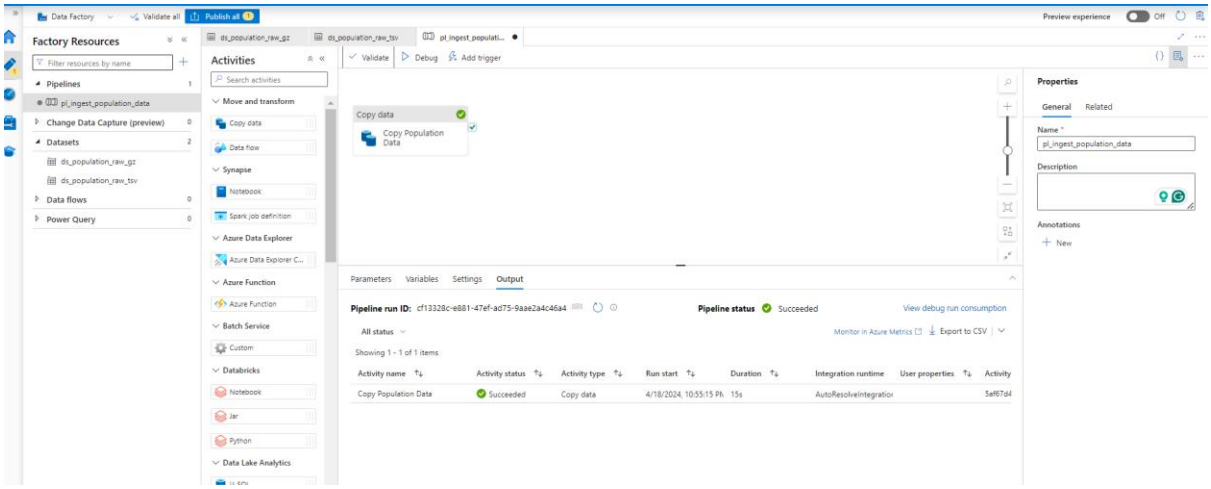
Create a corresponding dataset:

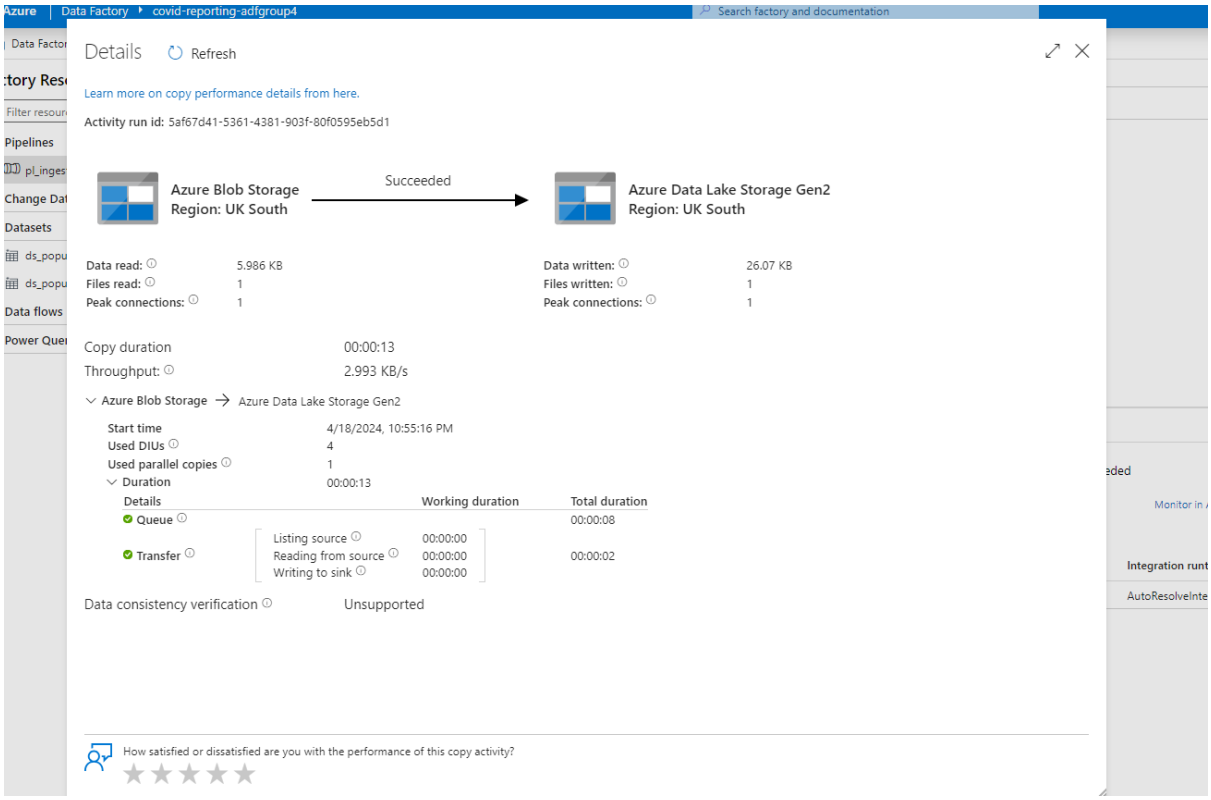




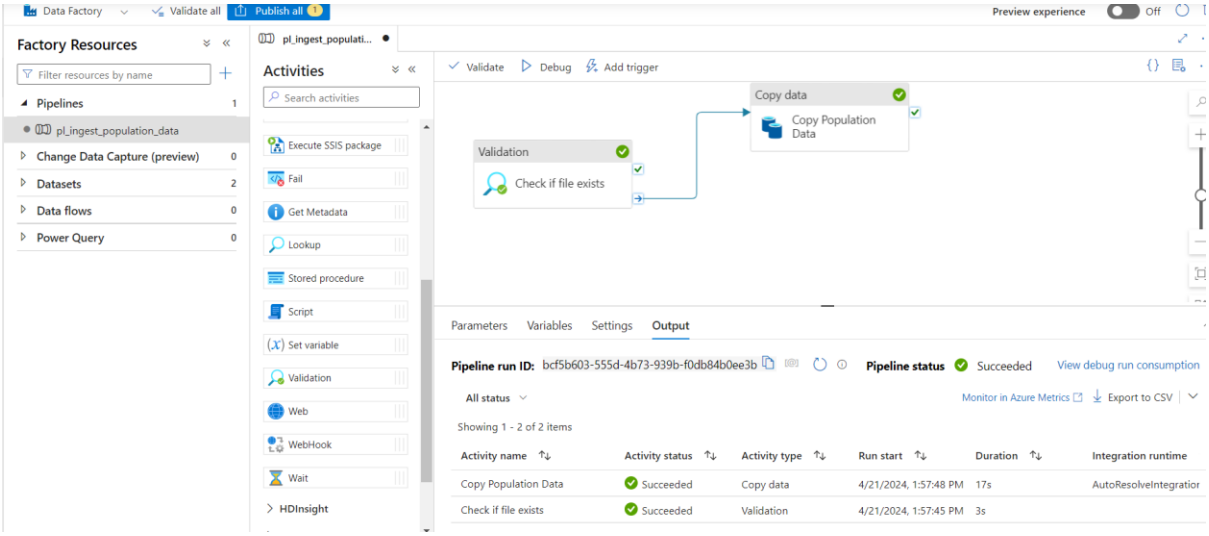


Creating pipeline ADF





Control Flow Activities 1 -Validation



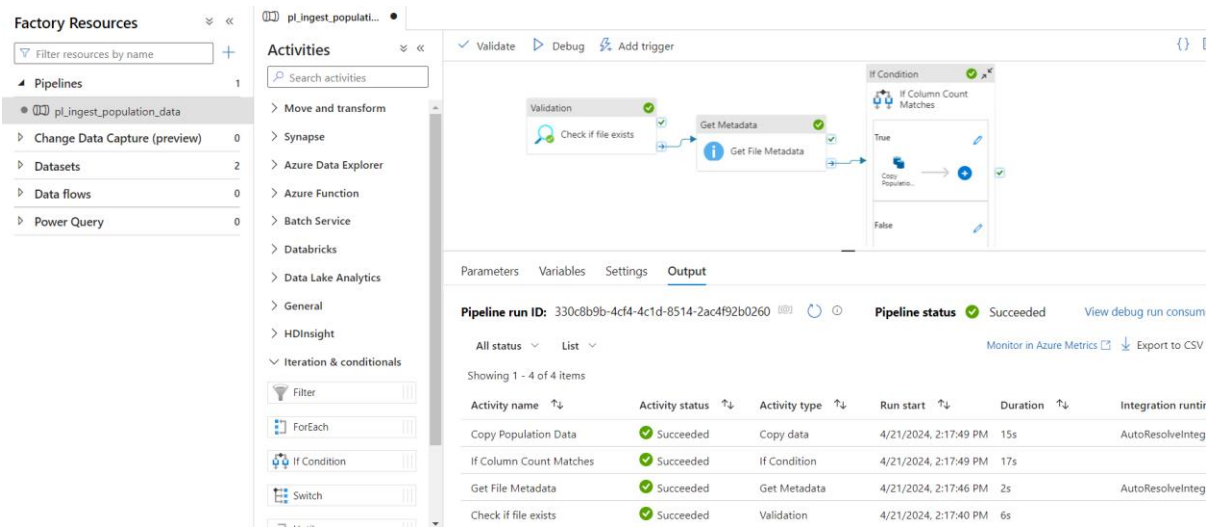
Control Flow Activities 2 – Get Metadata, If Condition, Web Activities

Get File Metadata copy

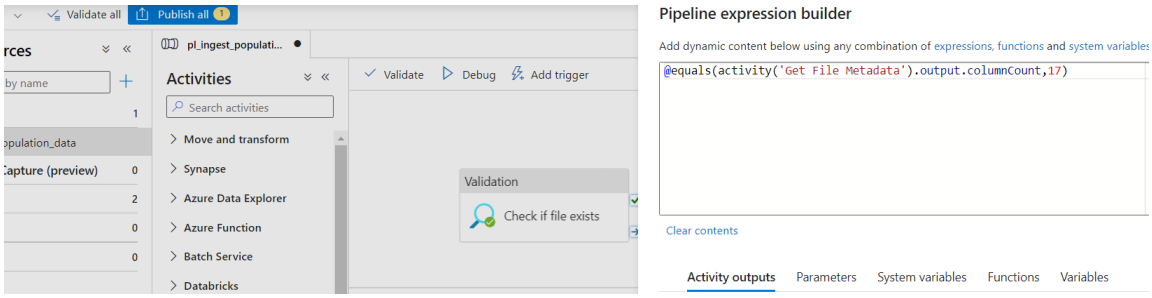

```
{
        "size": 5986,
        "exists": true,
        "columnCount": 13,
        "effectiveIntegrationRuntime": "AutoResolveIntegrationRuntime (UK South)",
        "executionDuration": 0,
        "durationInQueue": {
                "integrationRuntimeQueue": 1
        },
        "billingReference": {
                "activityType": "PipelineActivity",
                "billableDuration": [
```

```
                    {
                        "meterType": "AzureIR",
                        "duration": 0.016666666666666666,
                        "unit": "Hours"
                    }
                ]
            }
}
```

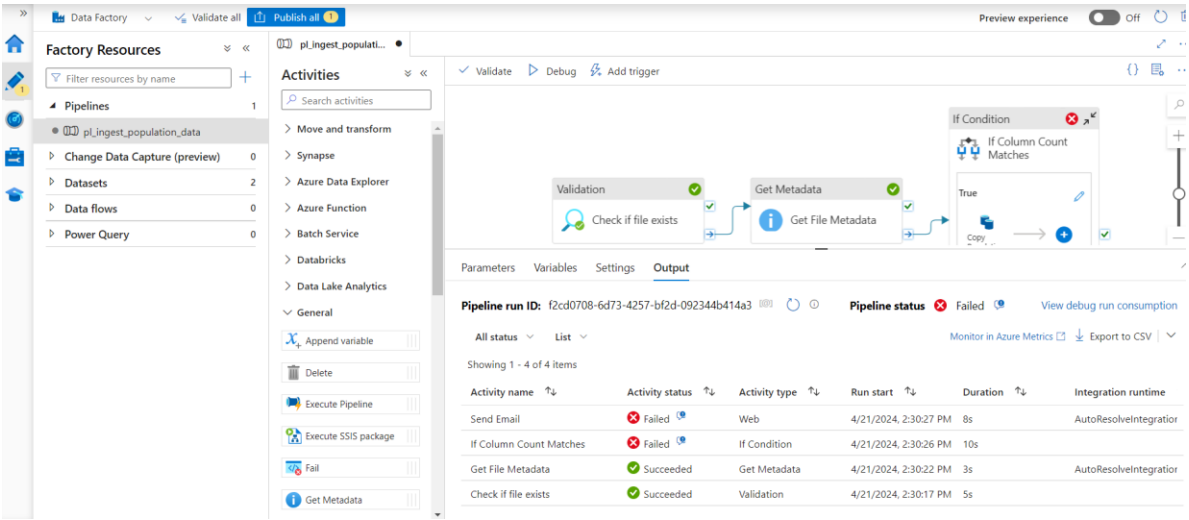

If condition



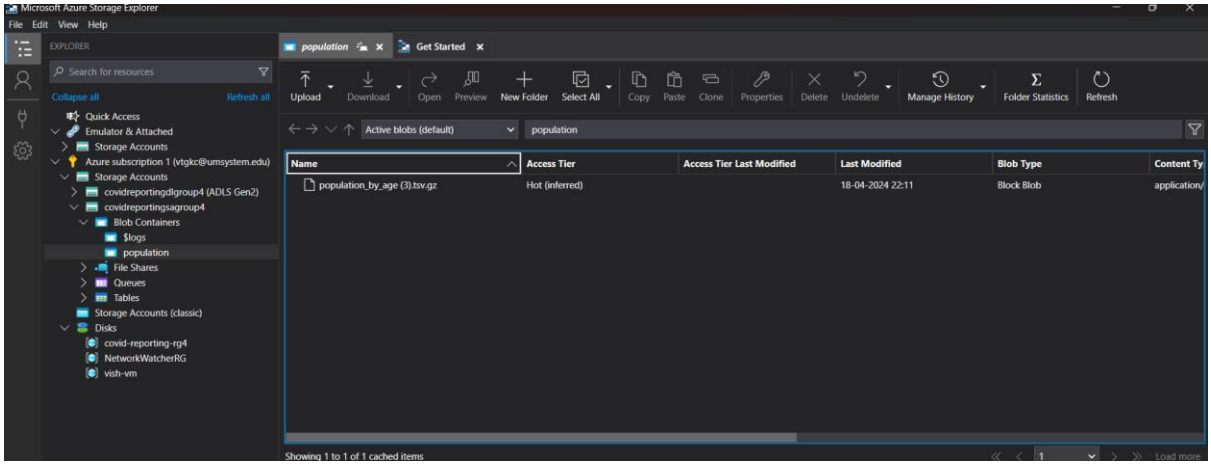
Testing the condition with 17, it returns false



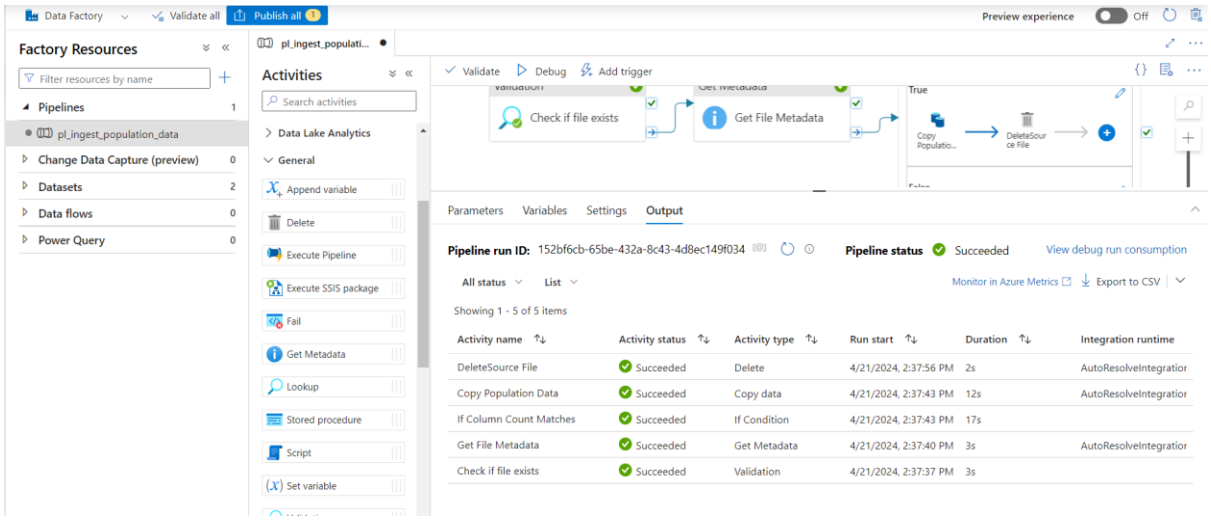
Count did not match, and send email failed since the request is not possible with url provided.



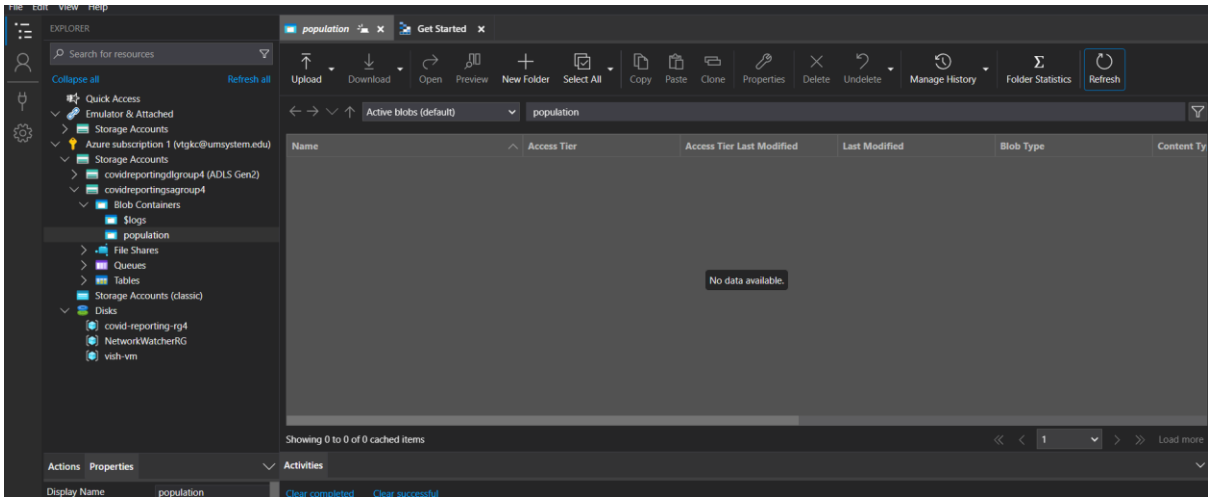
We could see the gz file in the corresponding path.



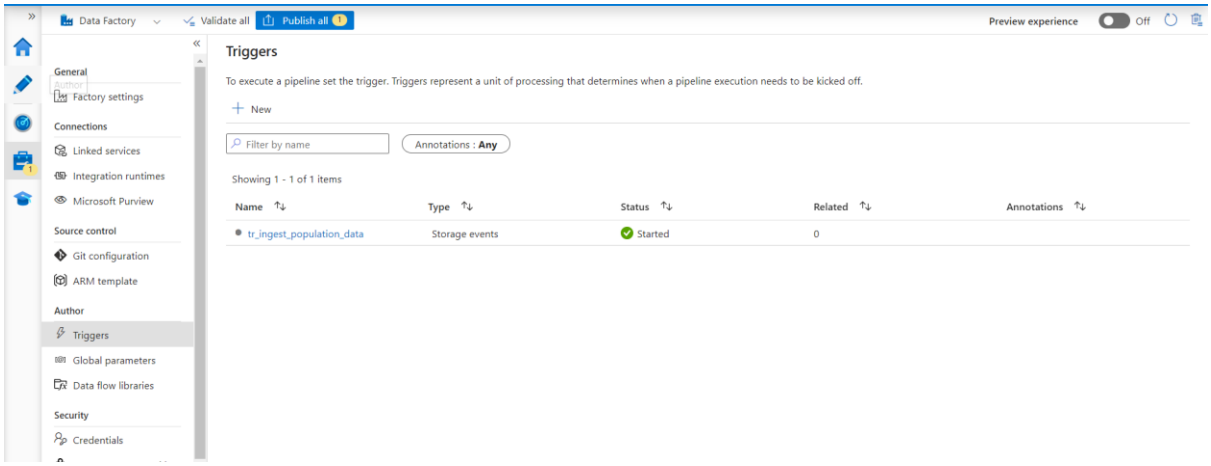
Delete file option successfully carried out



And now we could see the gz file got vanished



Created an event Trigger

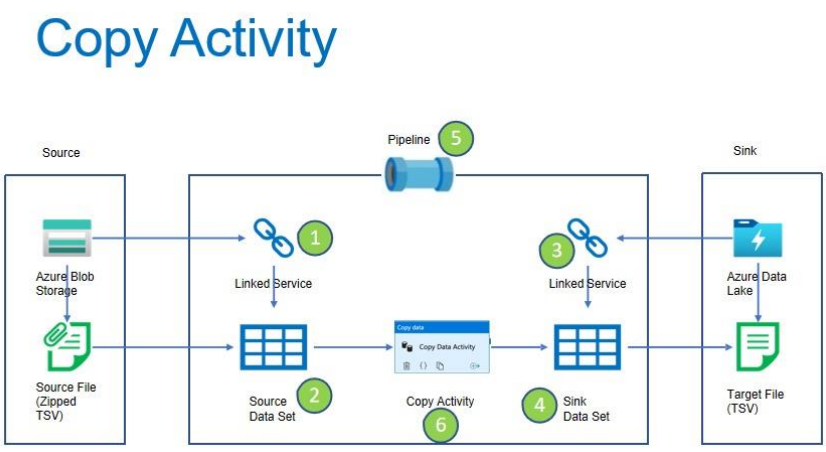


Data Ingestion from https

Data is from ECDC website

https://github.com/cloudboxacademy/covid19

Copy Activity – Case & Deaths Data



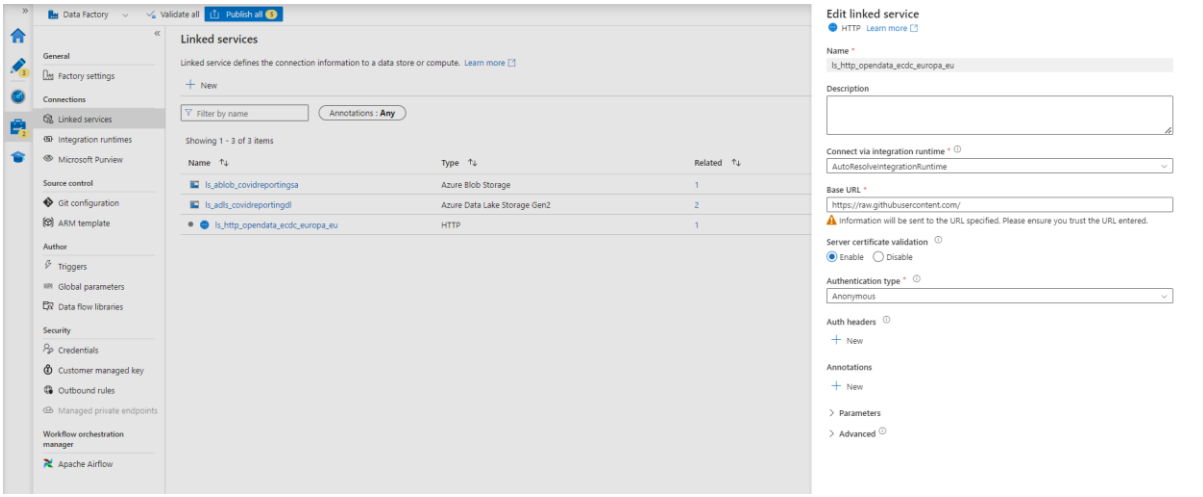
Create pipeline:

1. URL:

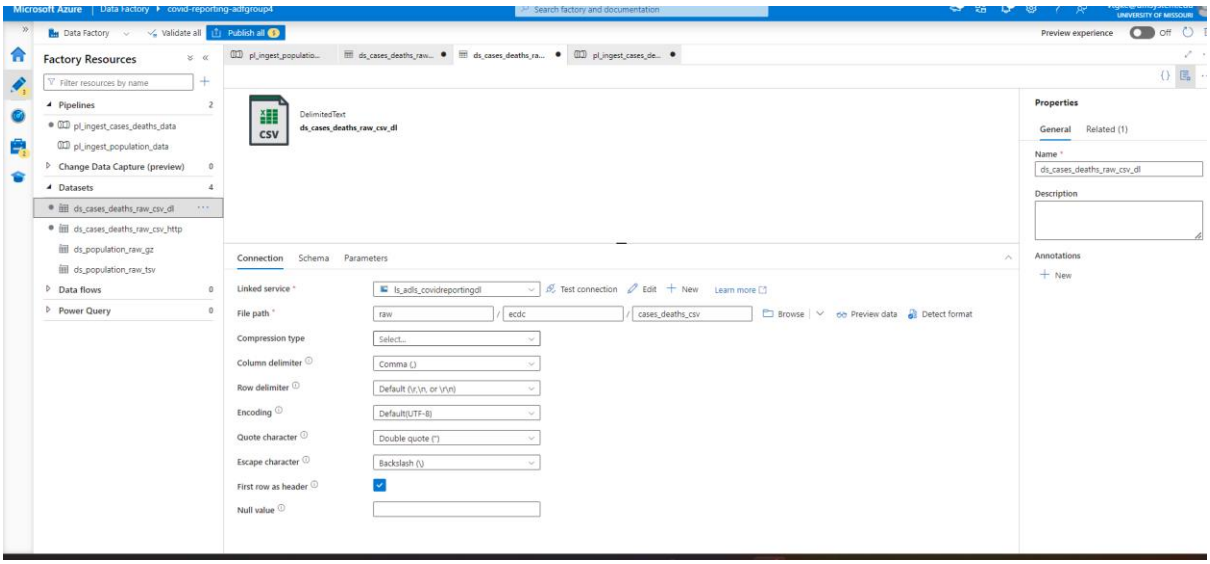
   https://raw.githubusercontent.com/cloudboxacademy/covid19/main/raw/main/ecdc_data/cases_deaths.csv

Linked service.

Base URL : https://raw.githubusercontent.com/



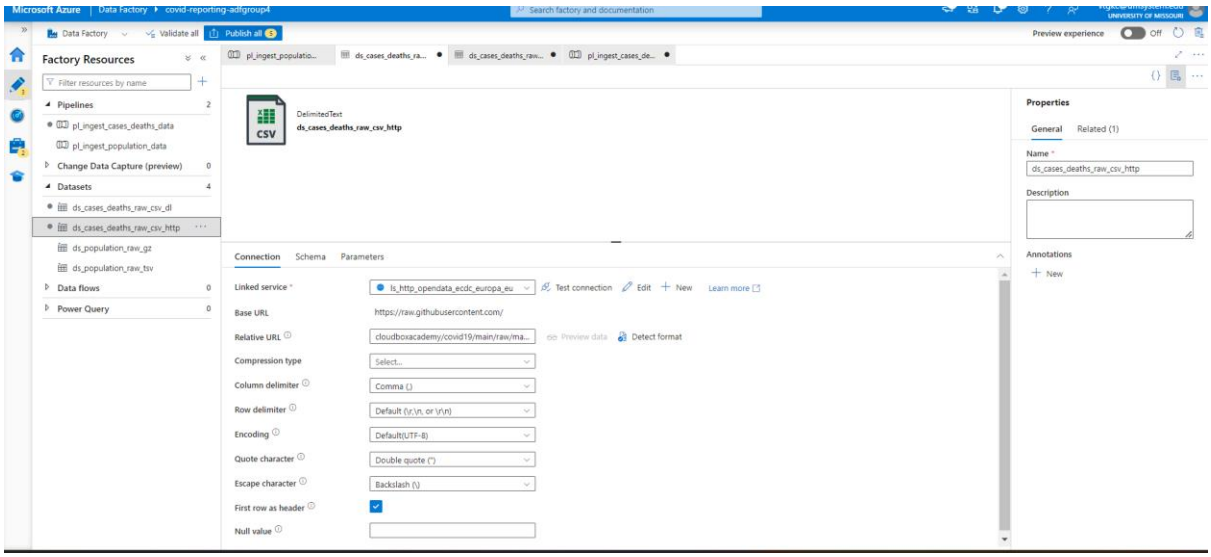
DL dataset



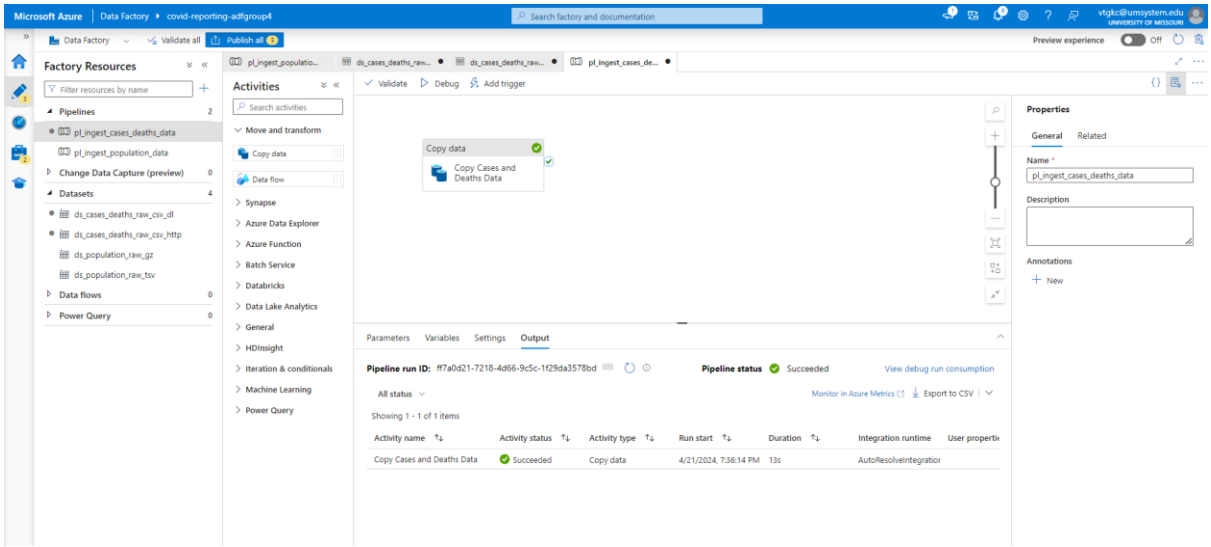
http dataset

Relative url:

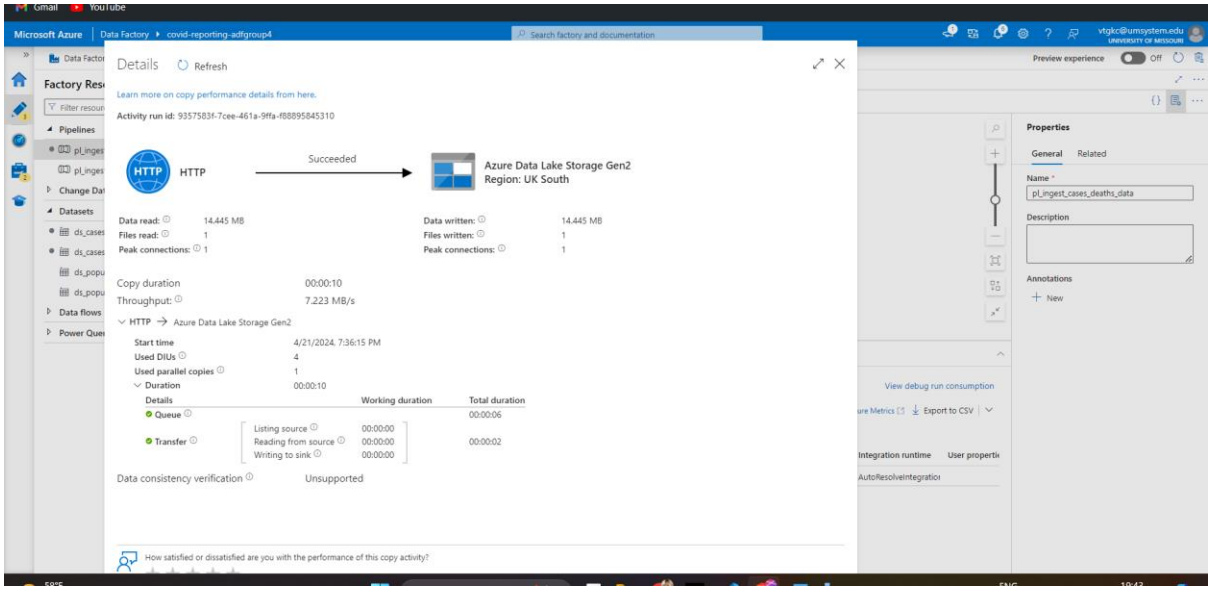
cloudboxacademy/covid19/main/raw/main/ecdc_data/cases_deaths.csv



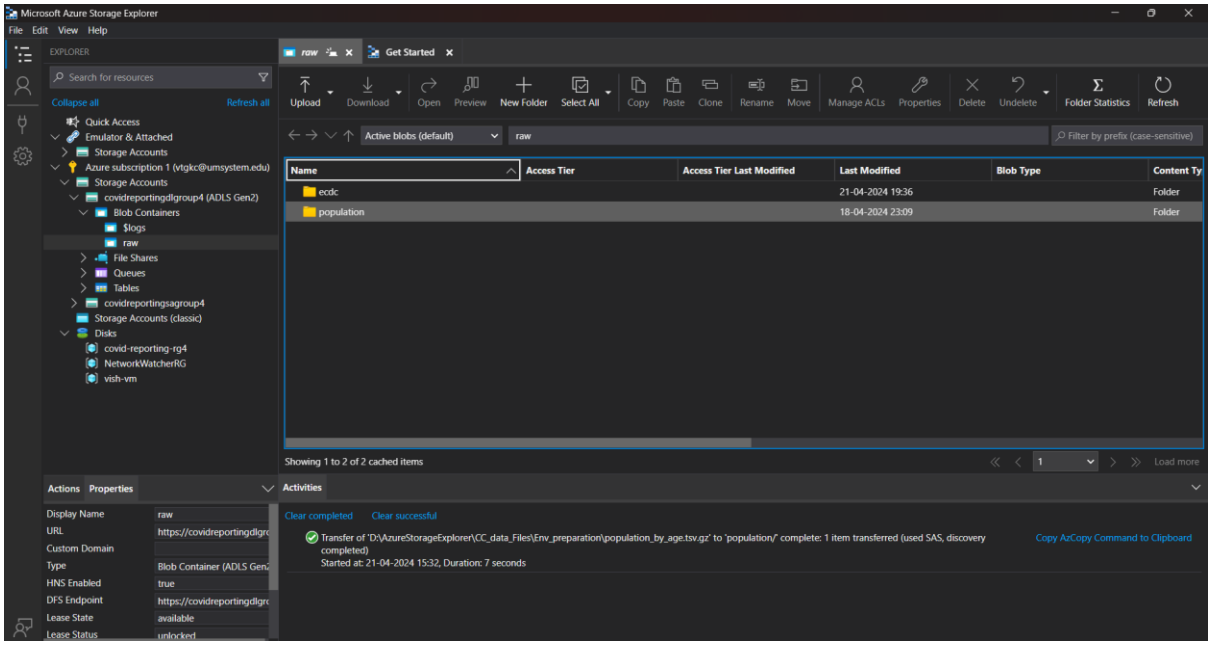
Pipeline



Details of http can be seen in the below screenshot

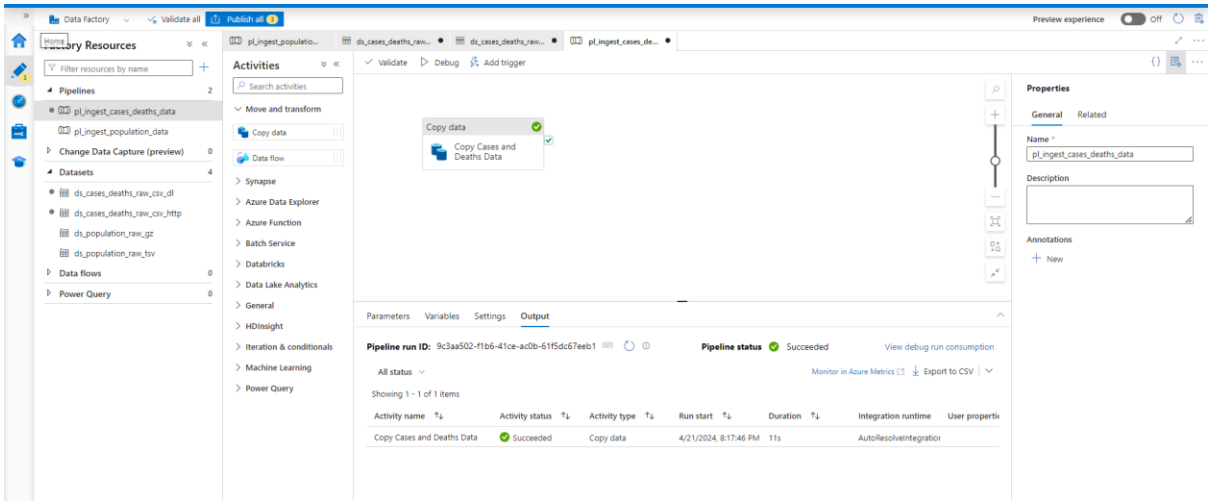


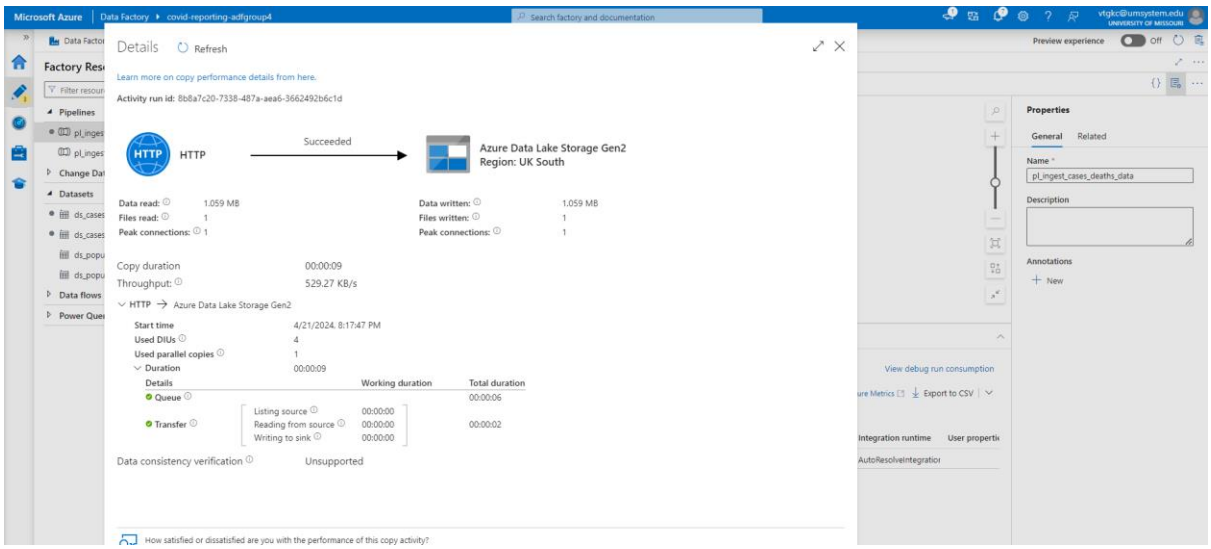
Now in the storage explorer you can see a new folder is created (ecdc).



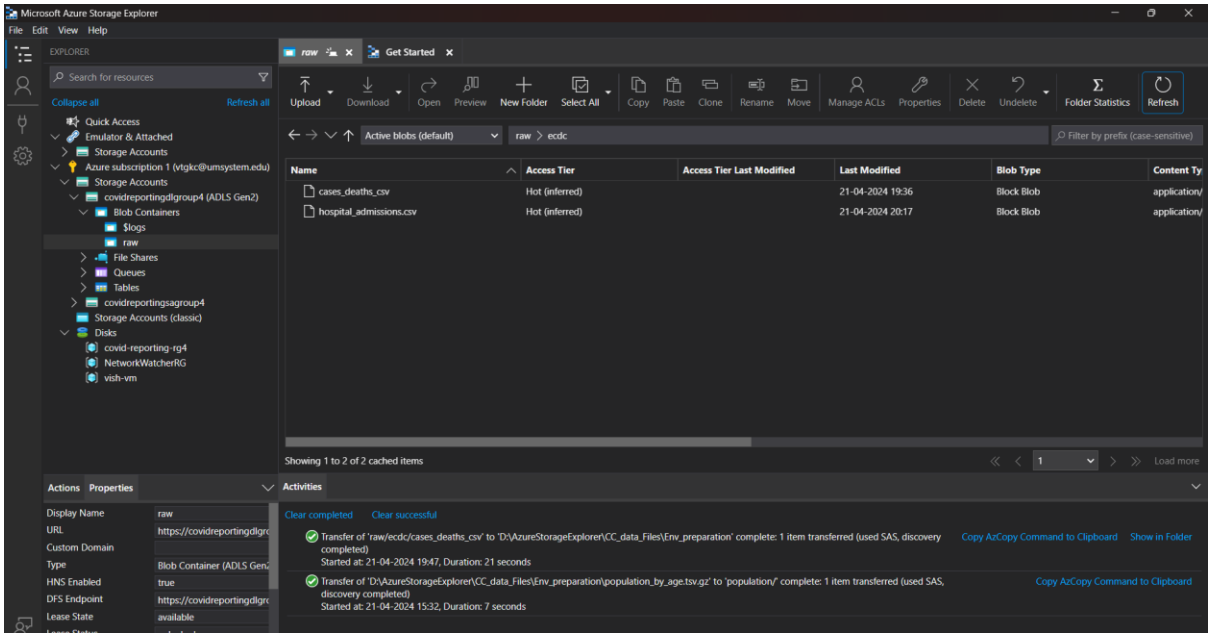
Pipeline variable

Pipeline source-sink hospital





This is hospital info and the file has been added

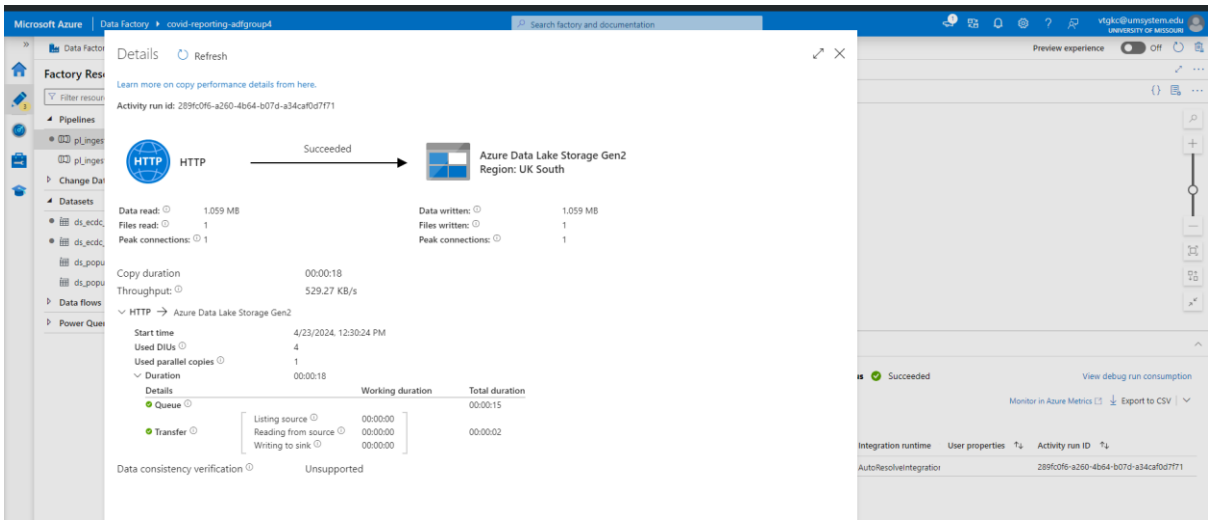


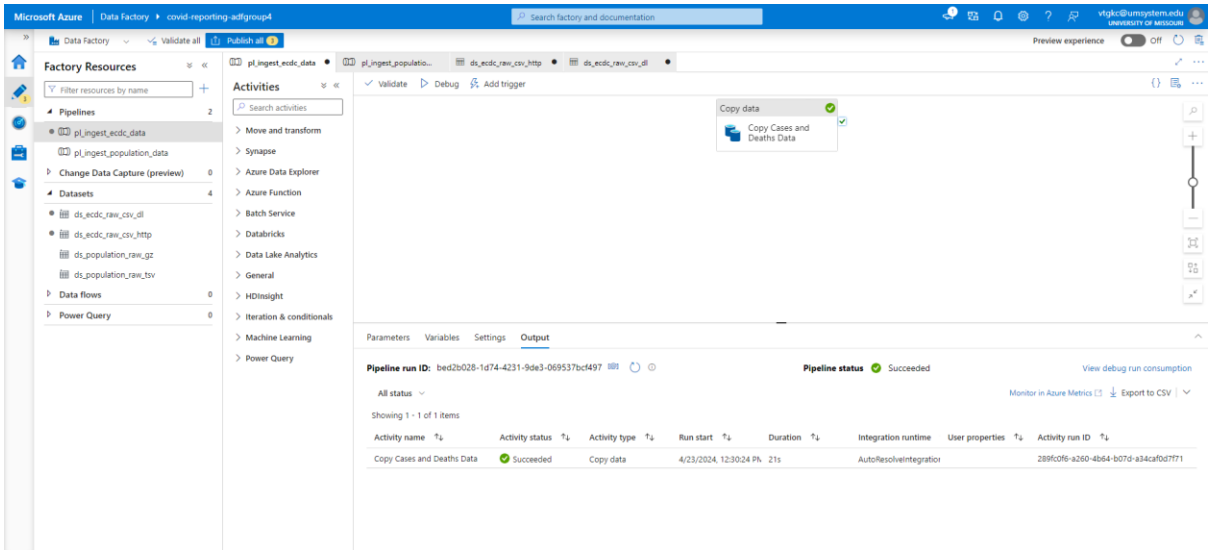
Pipeline parameters and schedule trigger

Hospital admissions link

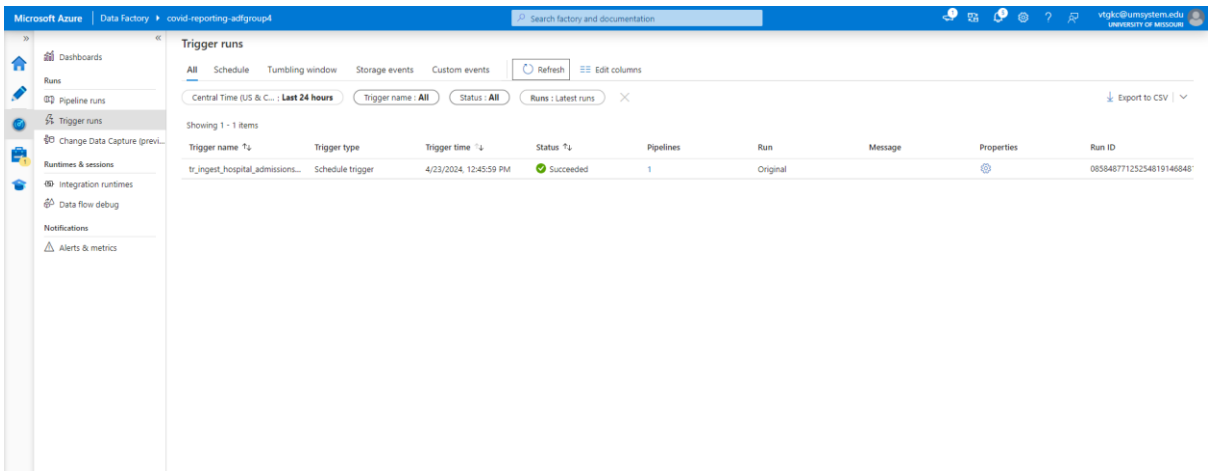
https://raw.githubusercontent.com/cloudboxacademy/covid19/main/raw/main/ecdc_data/hospital_admissions.csv

Relative url : cloudboxacademy/covid19/main/raw/main/ecdc_data/hospital_admissions.csv

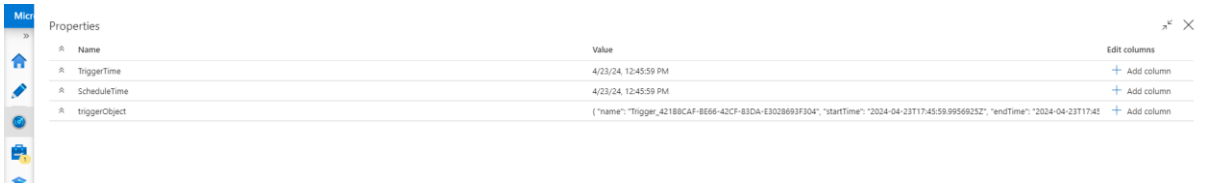


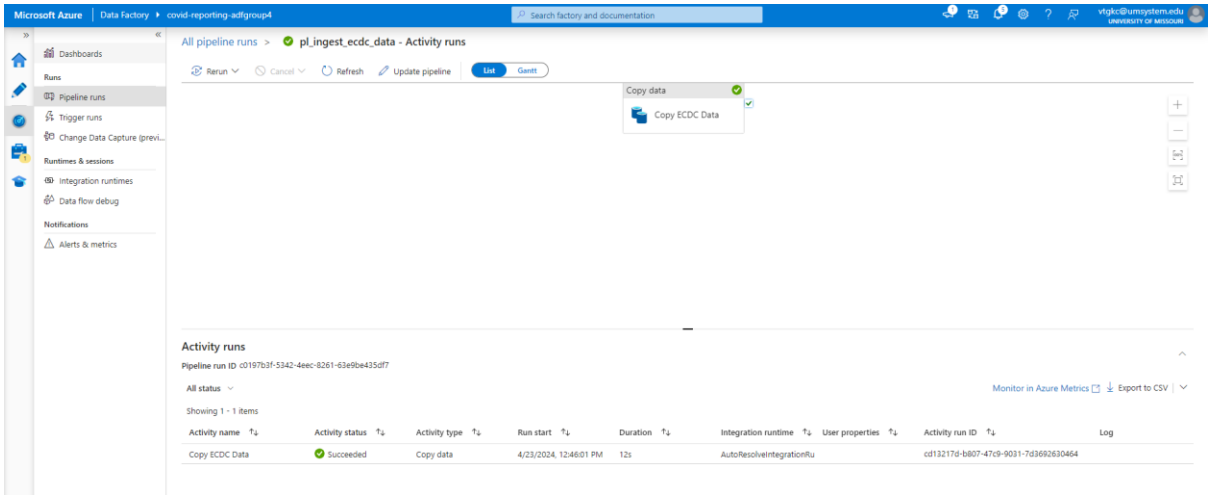


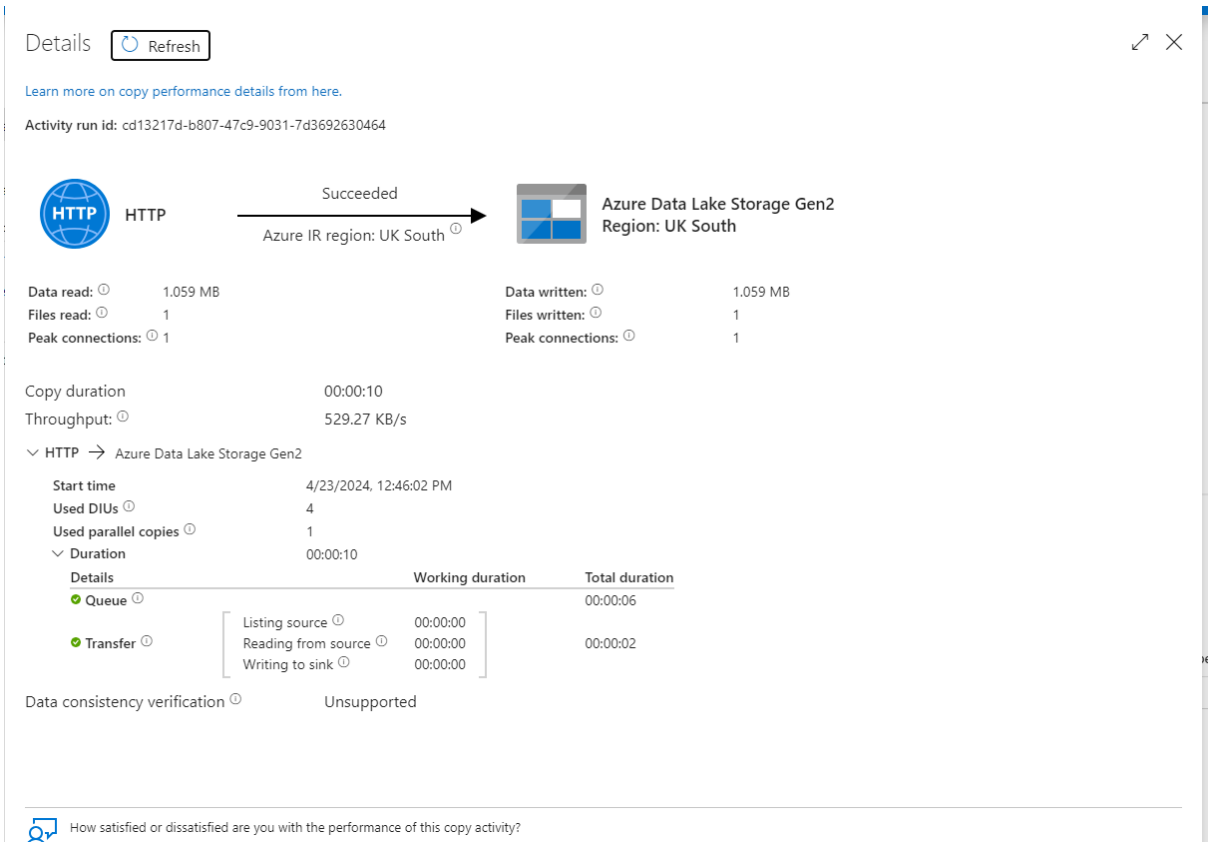
Hospital trigger run



Goto settings



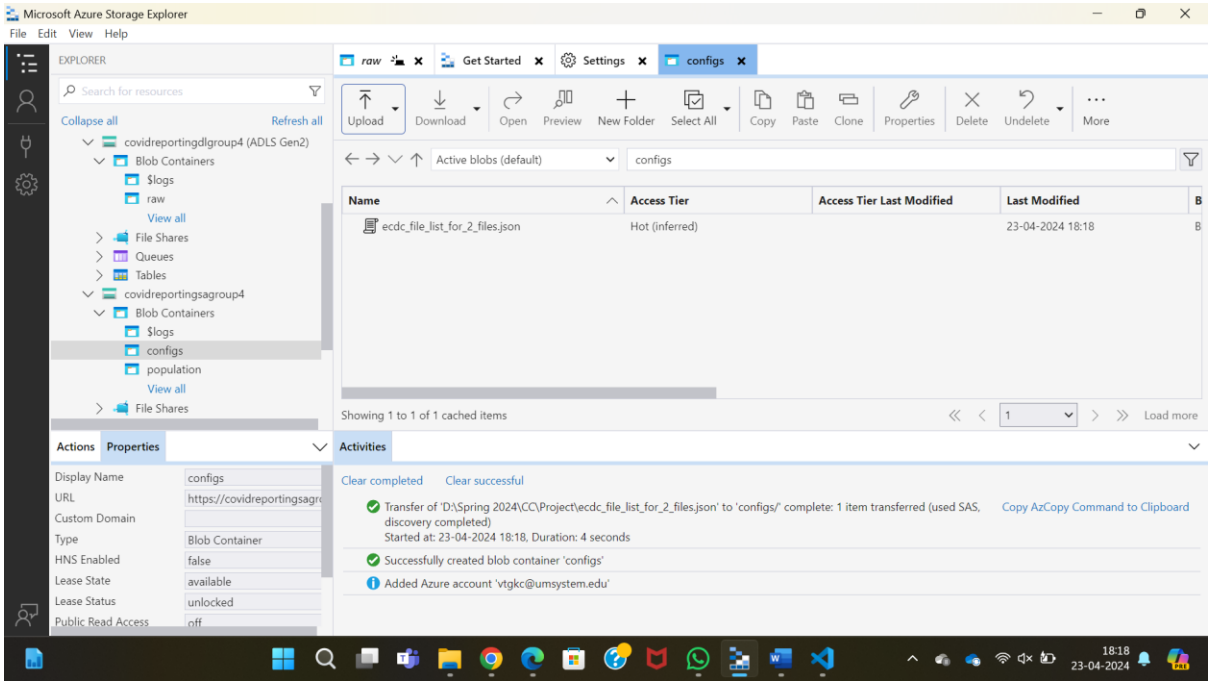




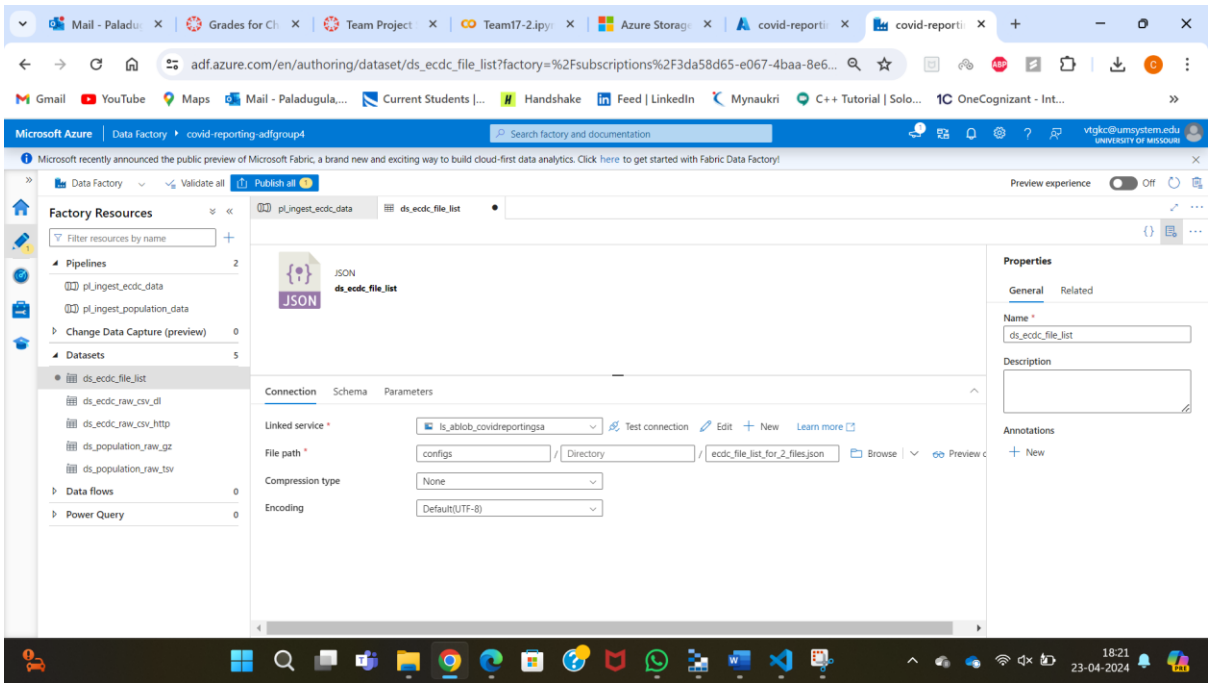
Control flow Activities:

Json file: ecdc_file_list_for_2_files

Upload the file in the storage account in configs.



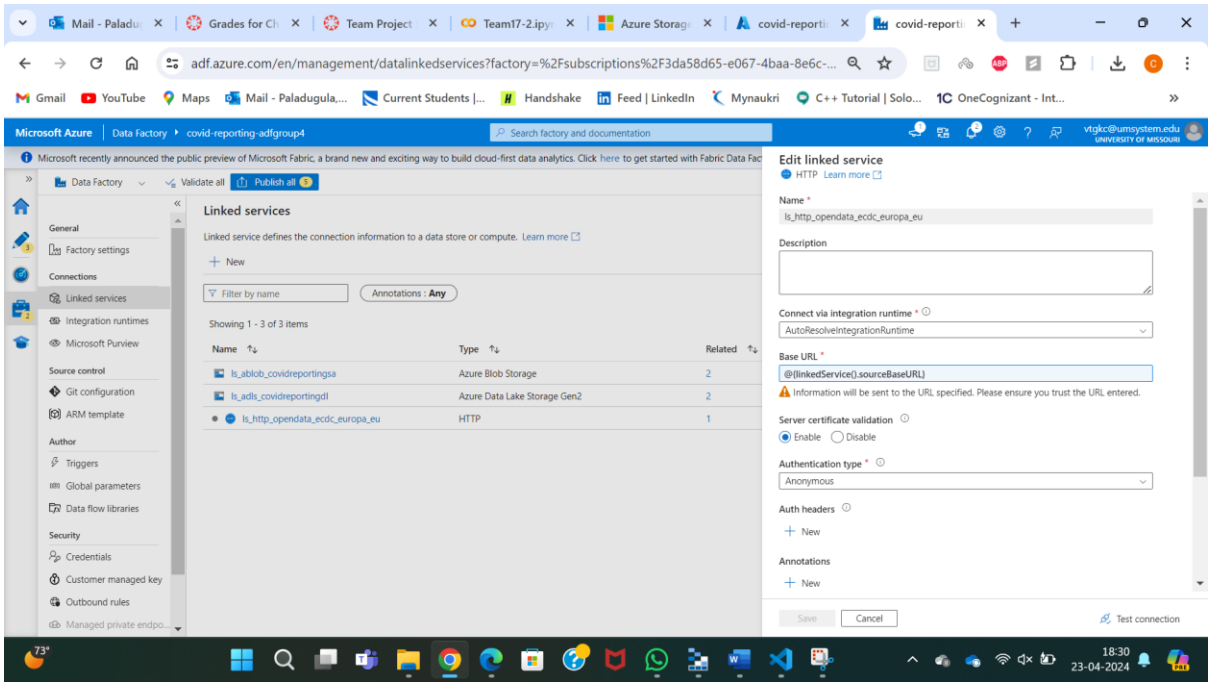
Upload the same json in the datasets in data factory.



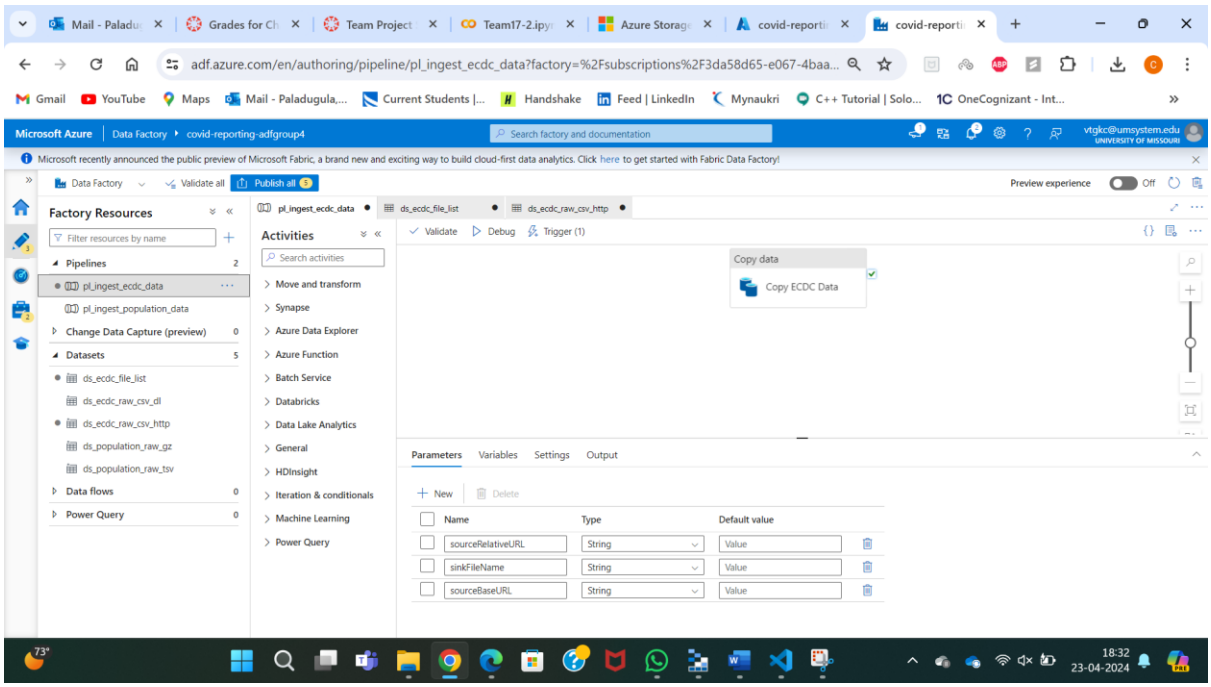
Make a new pipeline the lookup activity.

5:39 mins

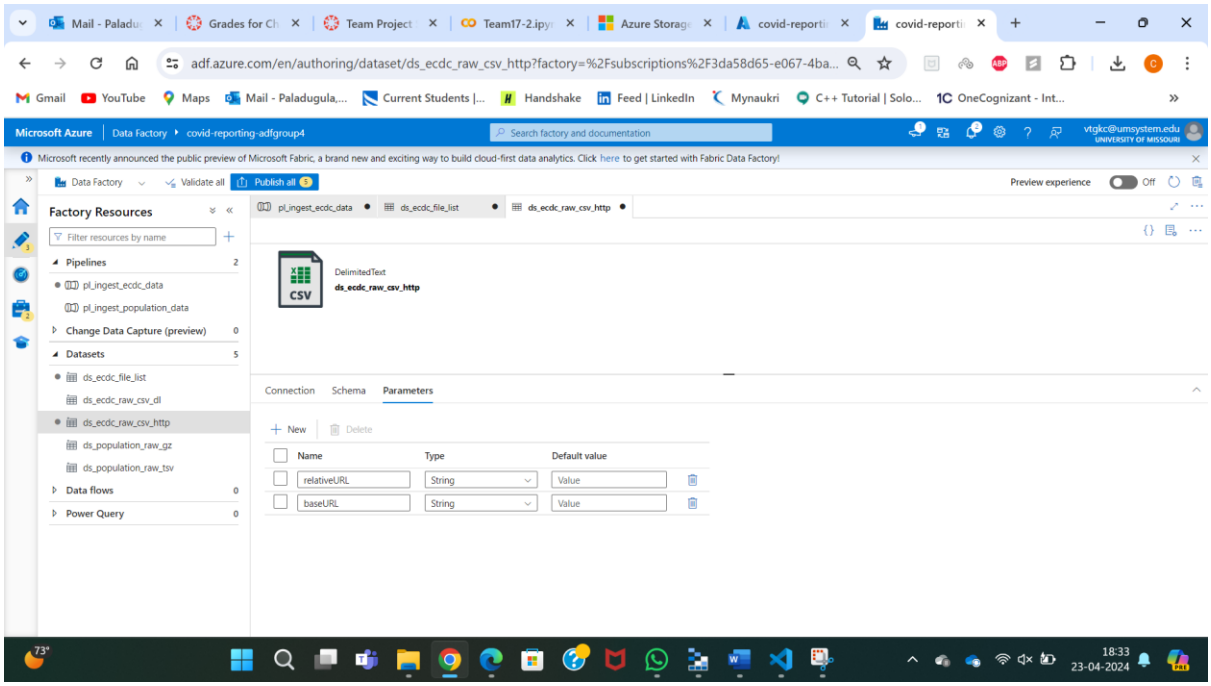
In manage, Linked services update the base url,

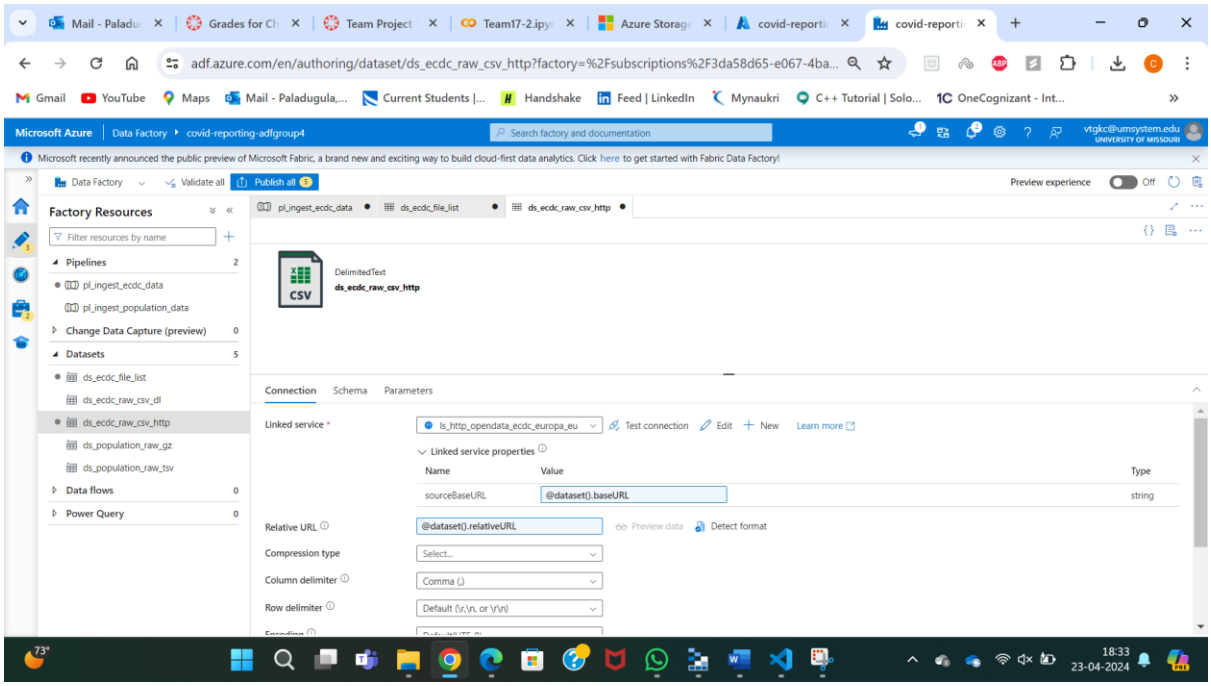


In pipelines, create the parameters for the base url.



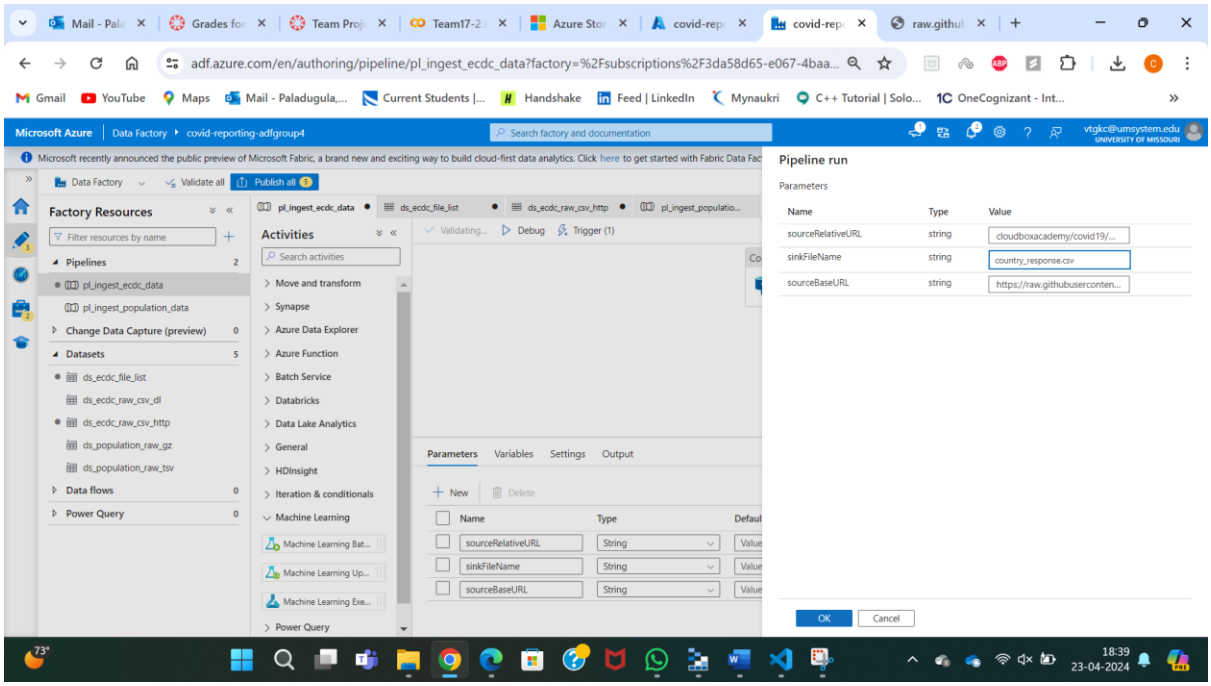
In datasets, create the parameters for the base url.



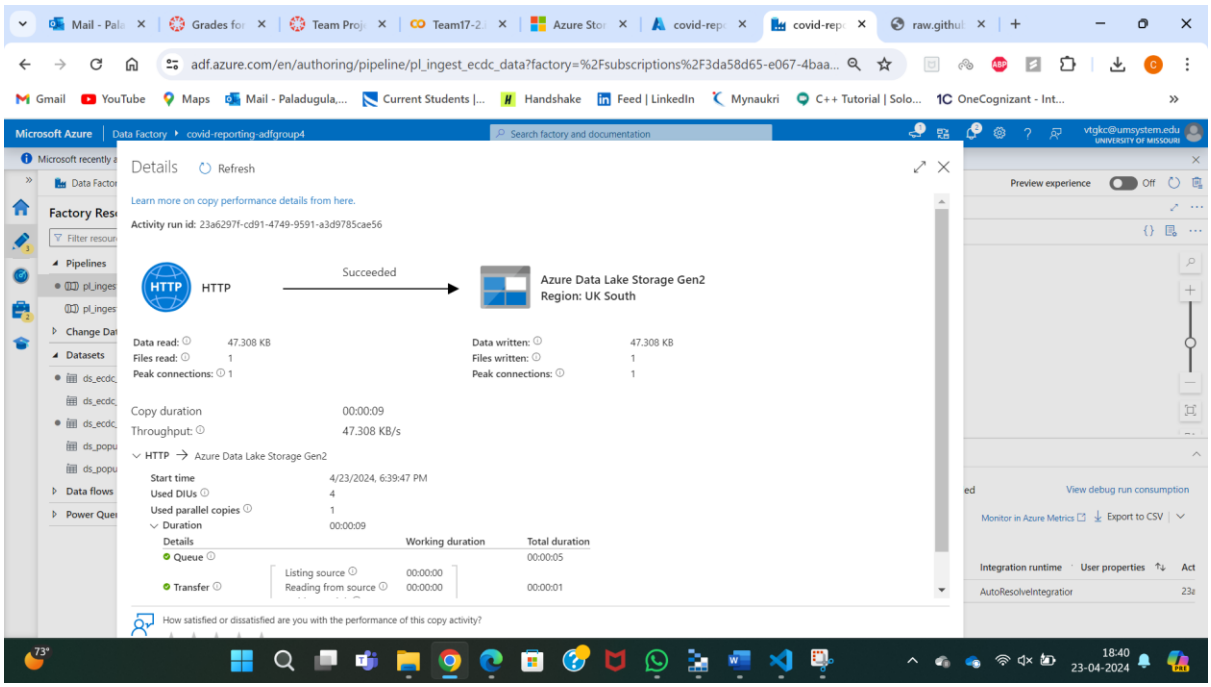


Relative url: cloudboxacademy/covid19/main/raw/main/ecdc_data/country_response.csv

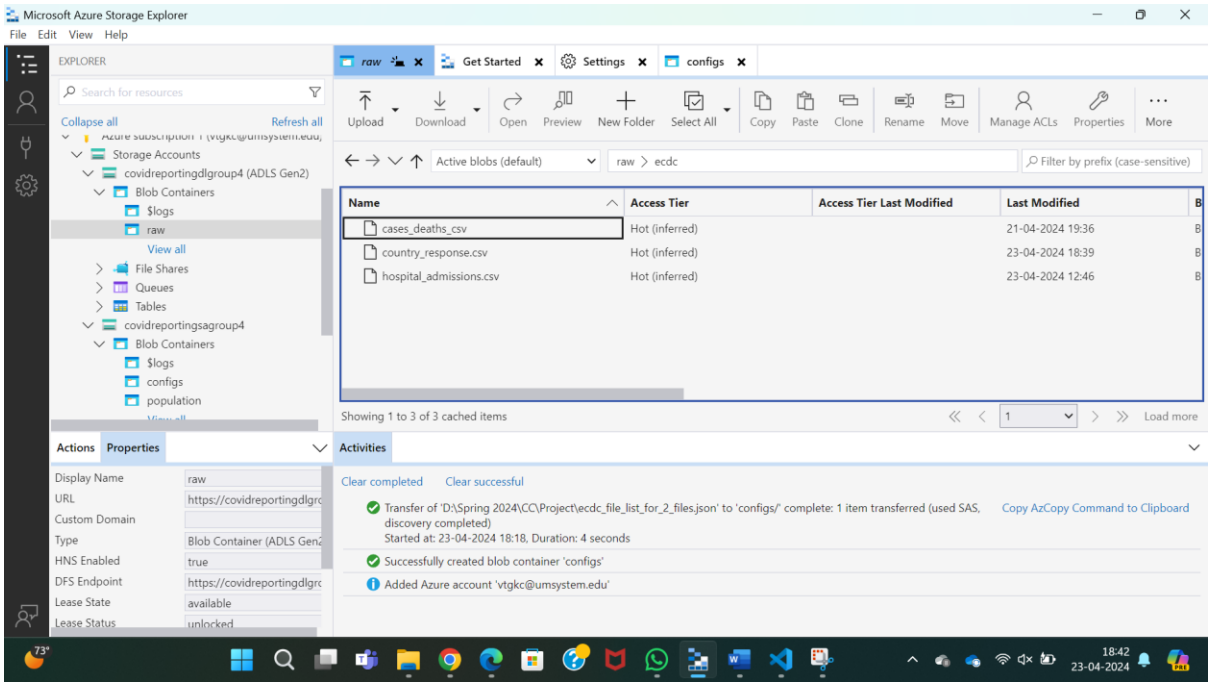
Base url: https://raw.githubusercontent.com/



Copied the country response from ecdc website to data lake.



We have verified the same thing in the explorer.



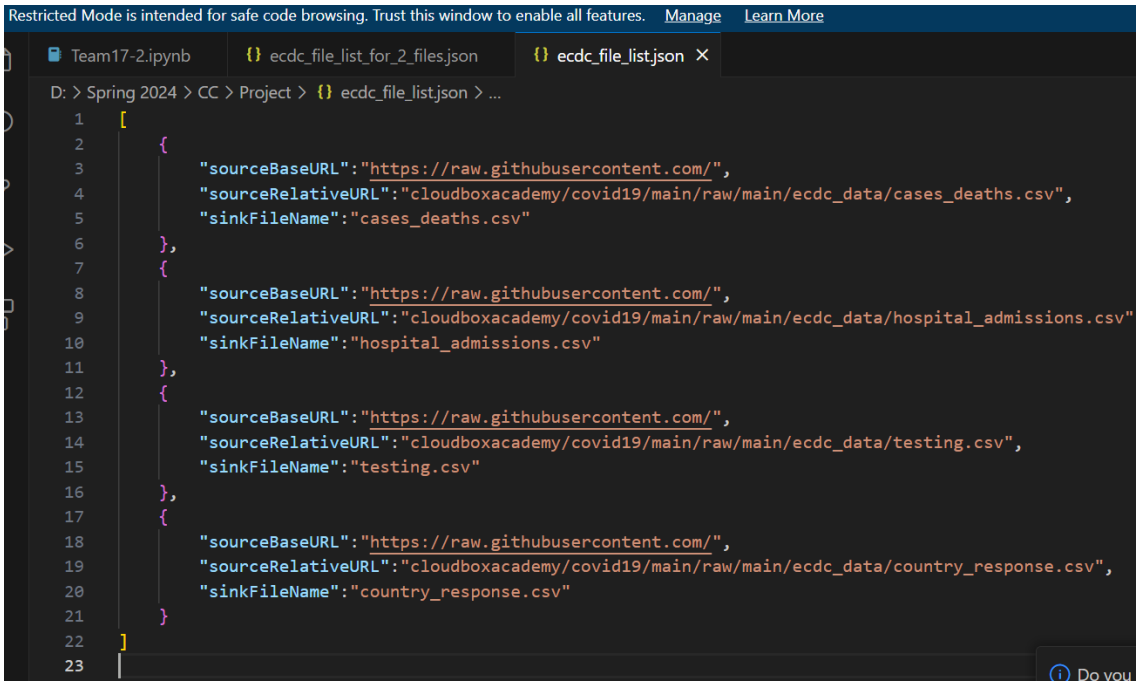
We publish it to save the changes.

Meta data driven pipeline:

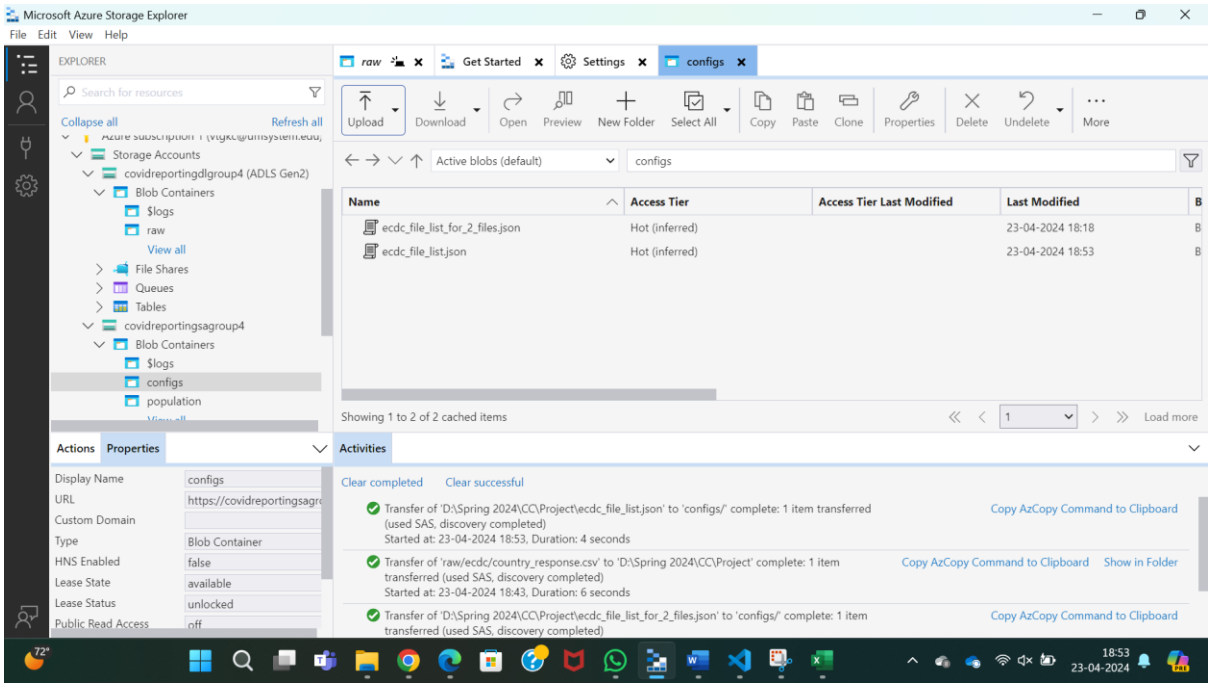
File name : ecdc_file_list

Relative [URL: cloudboxacademy/covid19/main/raw/main/ecdc_data/testing.csv](cloudboxacademy/covid19/main/raw/main/ecdc_data/testing.csv)

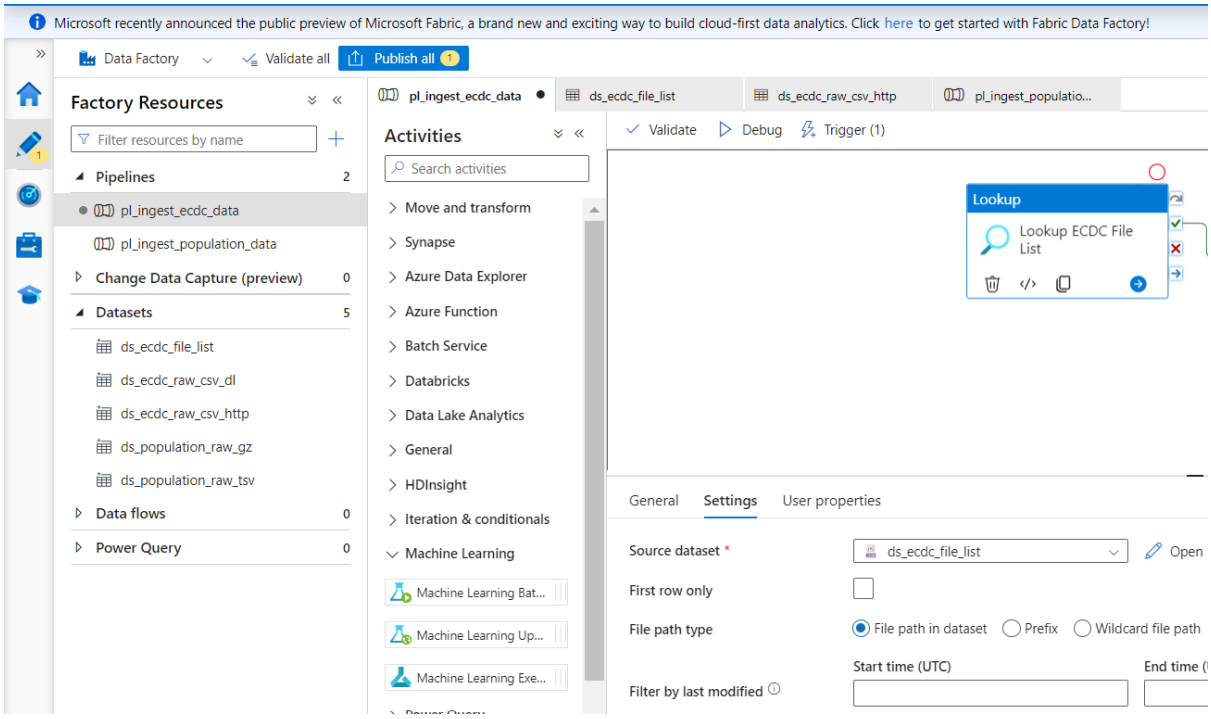
Base URL: https://raw.githubusercontent.com/



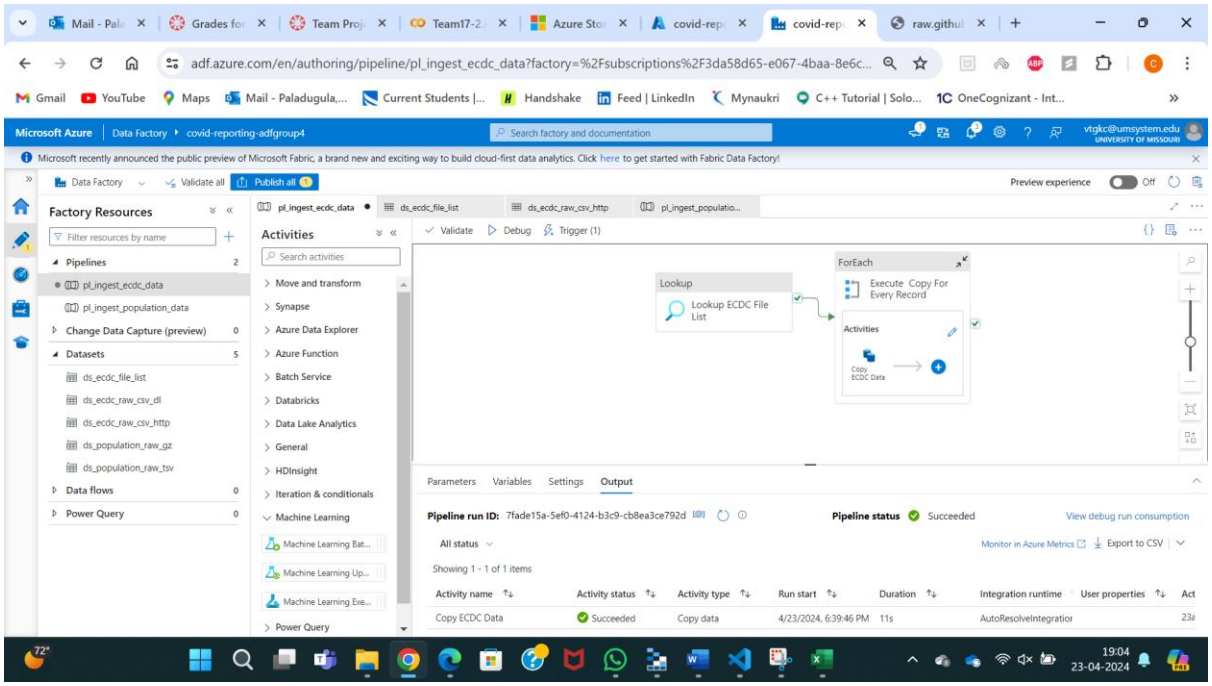
Upload the file in Explorer



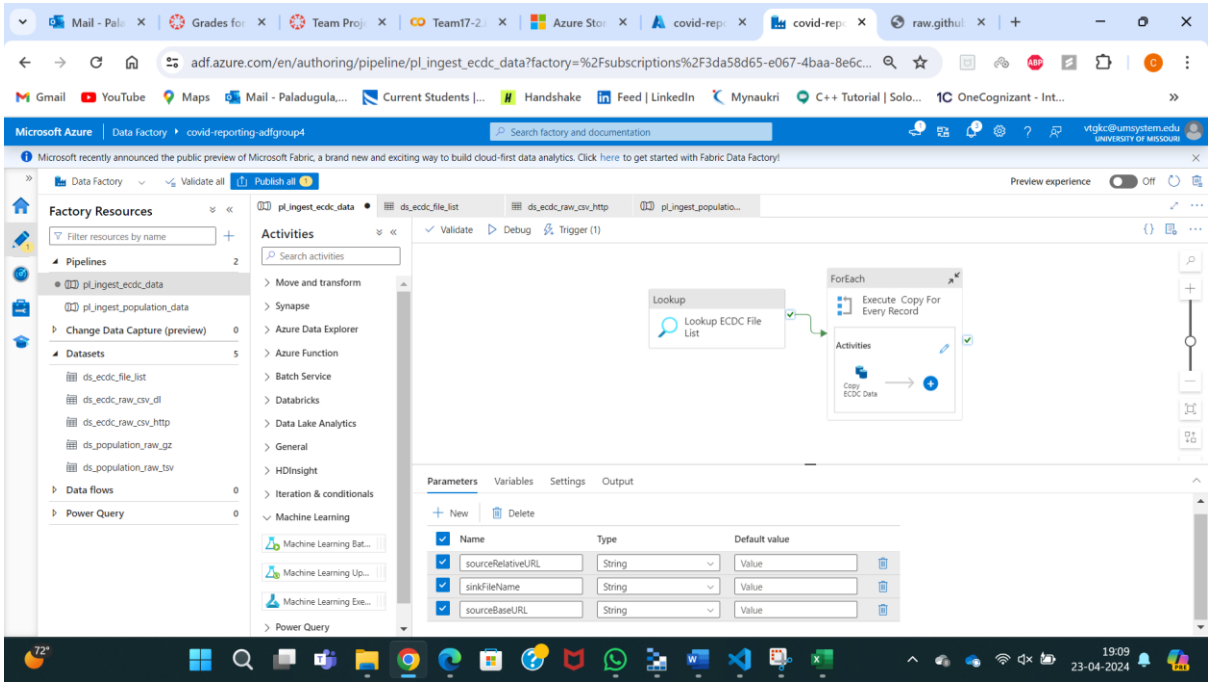
Creating a lookup activity in the ecdc pipeline,



Create a for each activity and connect to the lookup activity, then move the copy data to from the pipeline to for each activity.

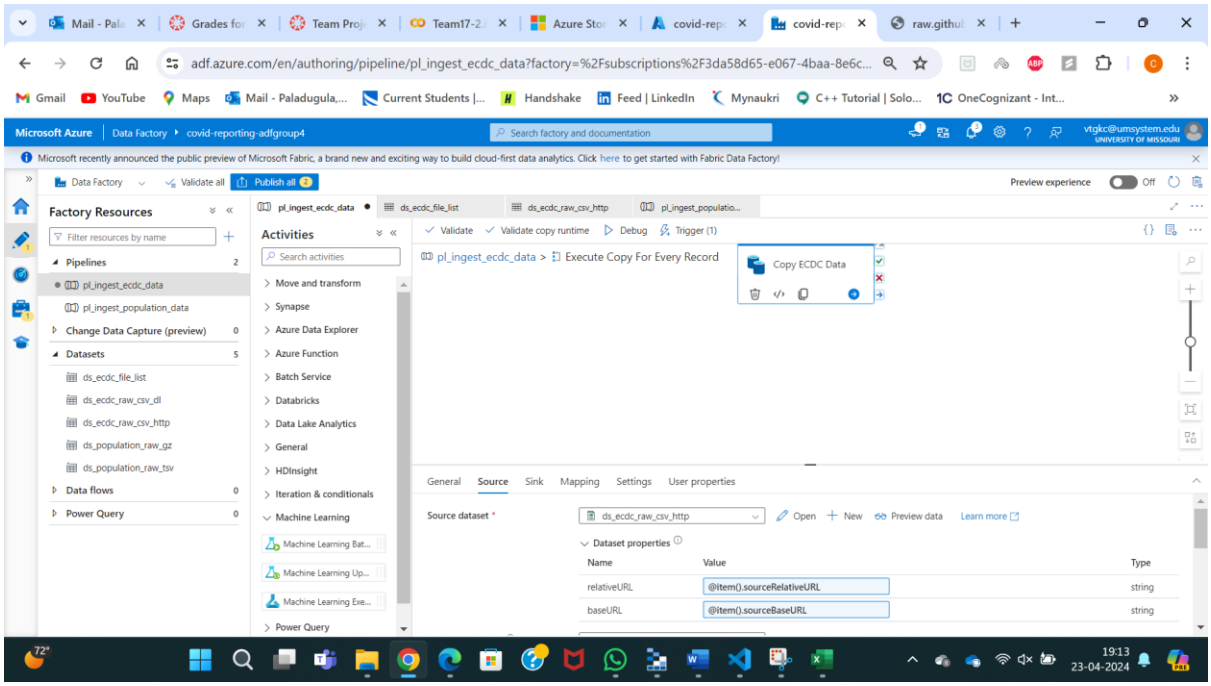


Then delete the parameters in the pipeline,

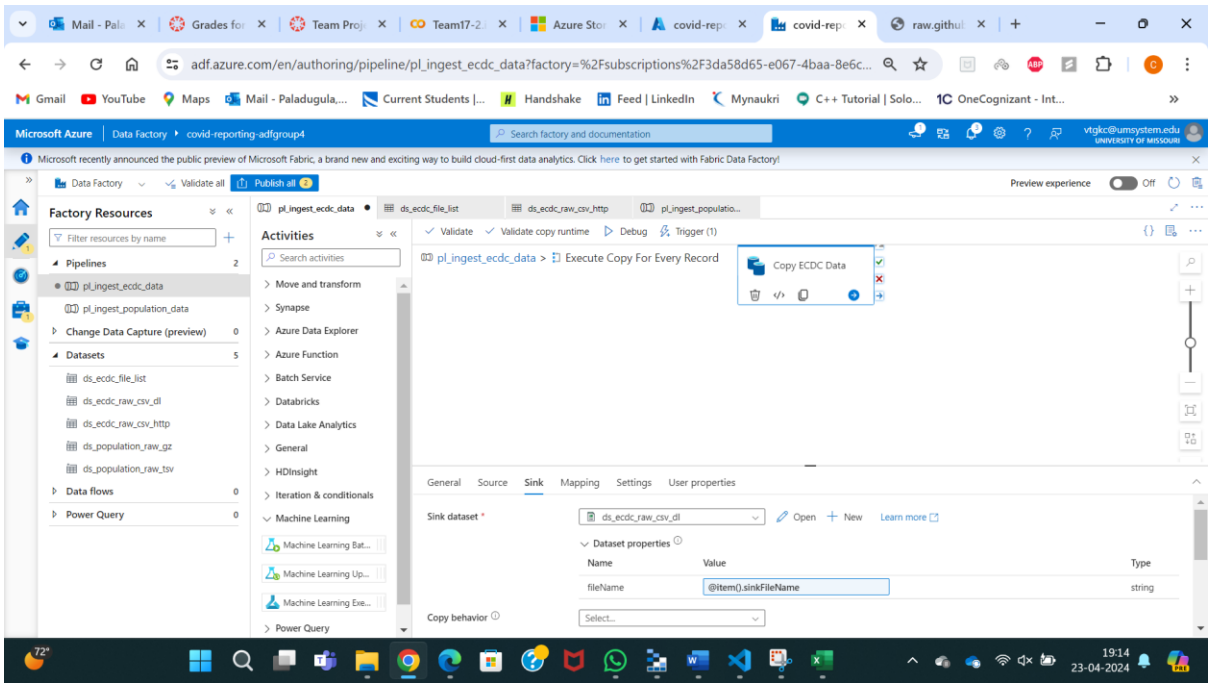


Open activity in the for each activity, Open the copy data.

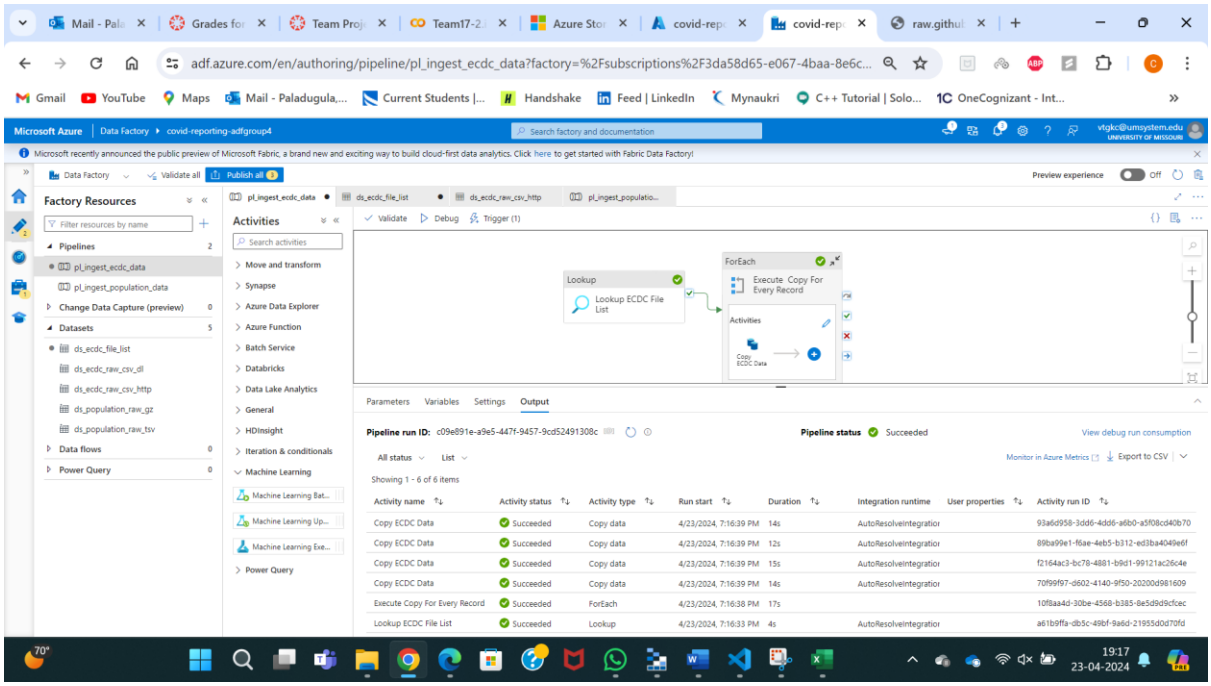
Update the relative url and base url, in the source



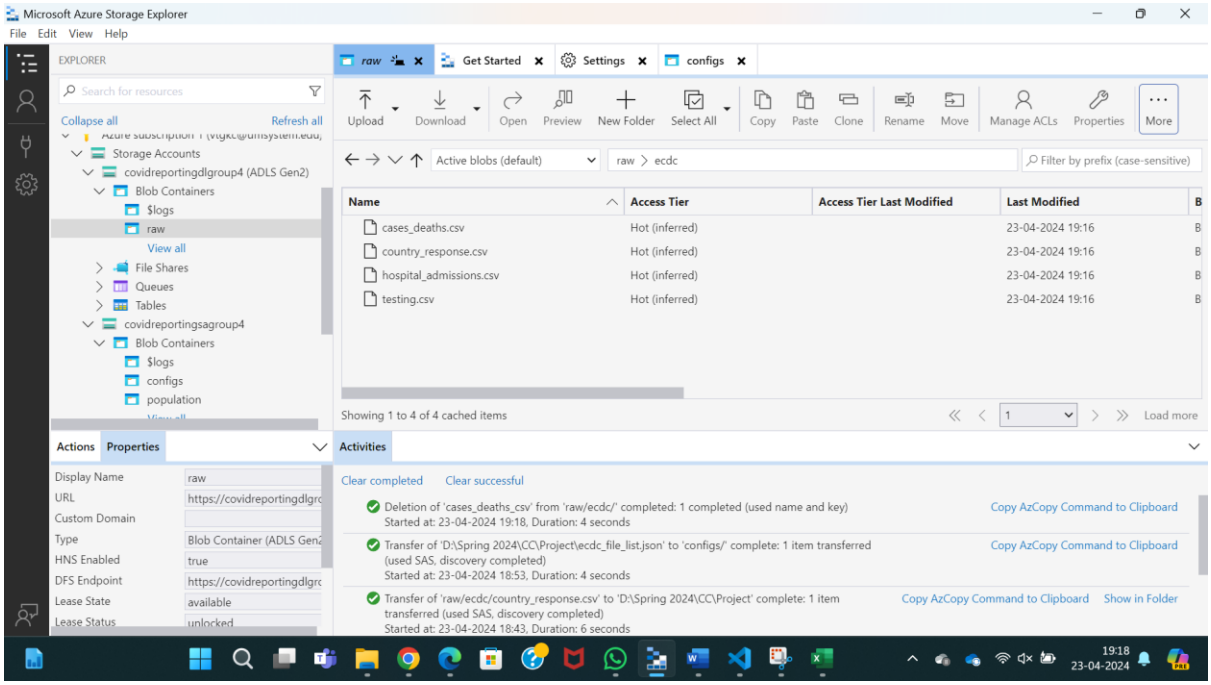
In sink, update the sink filename



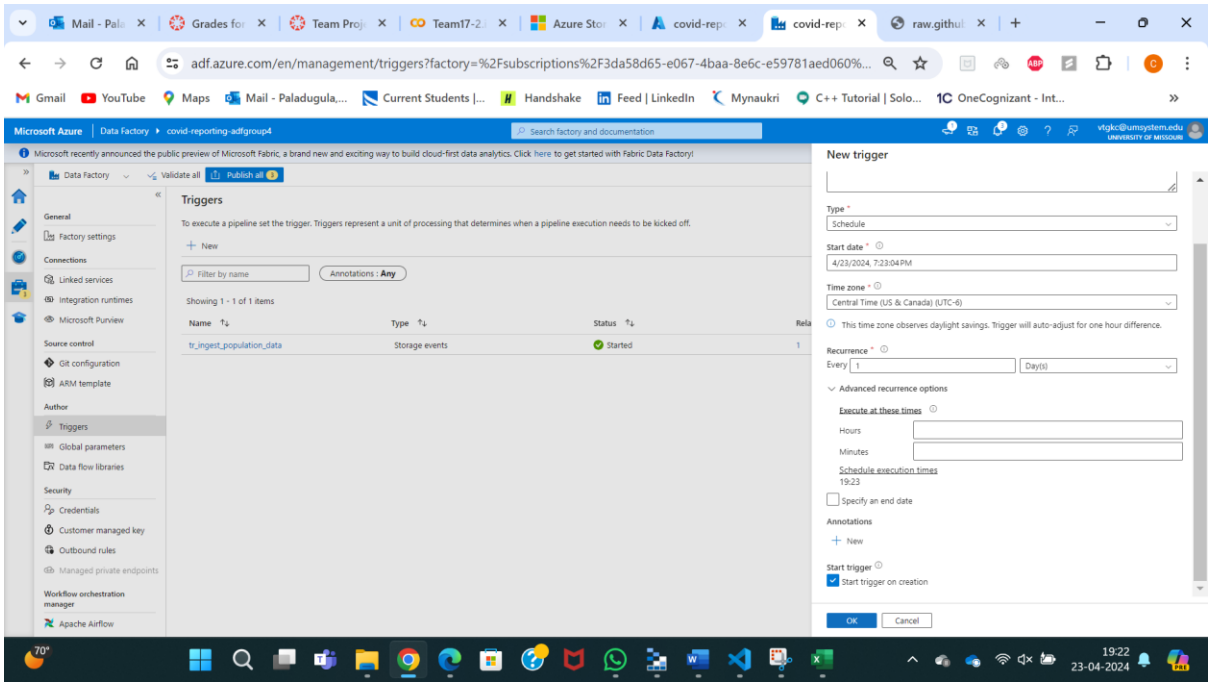
The pipeline is executed successfully.



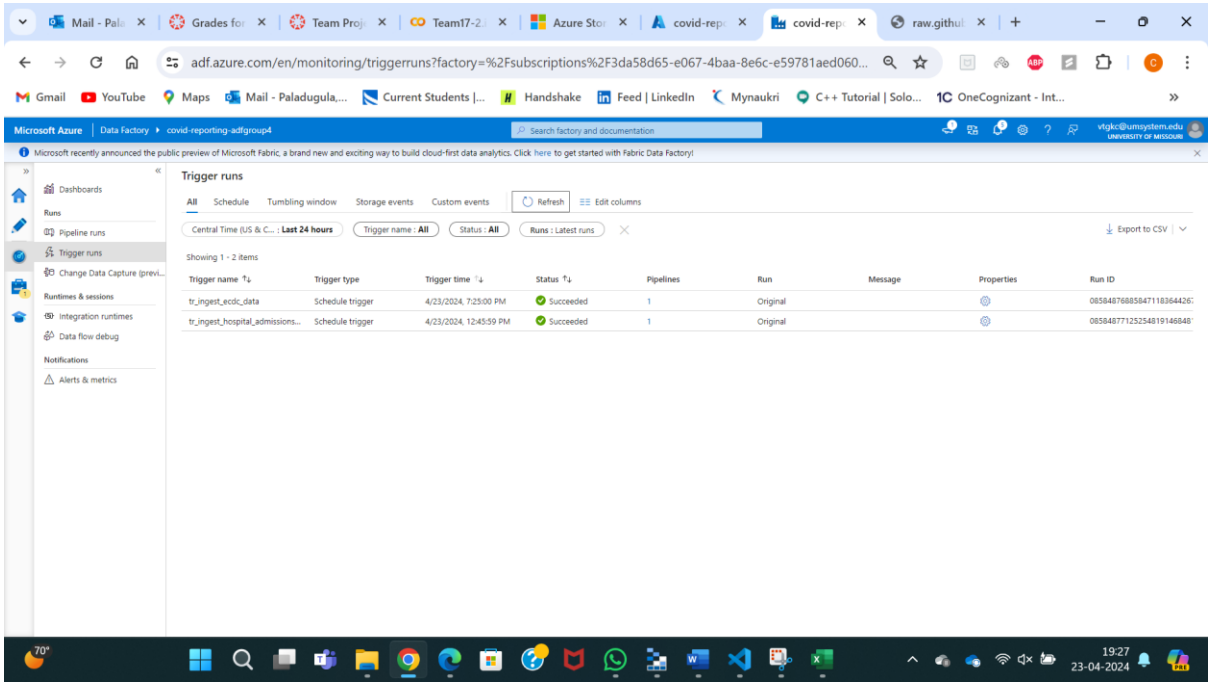
Now we can check the same in the explorer.



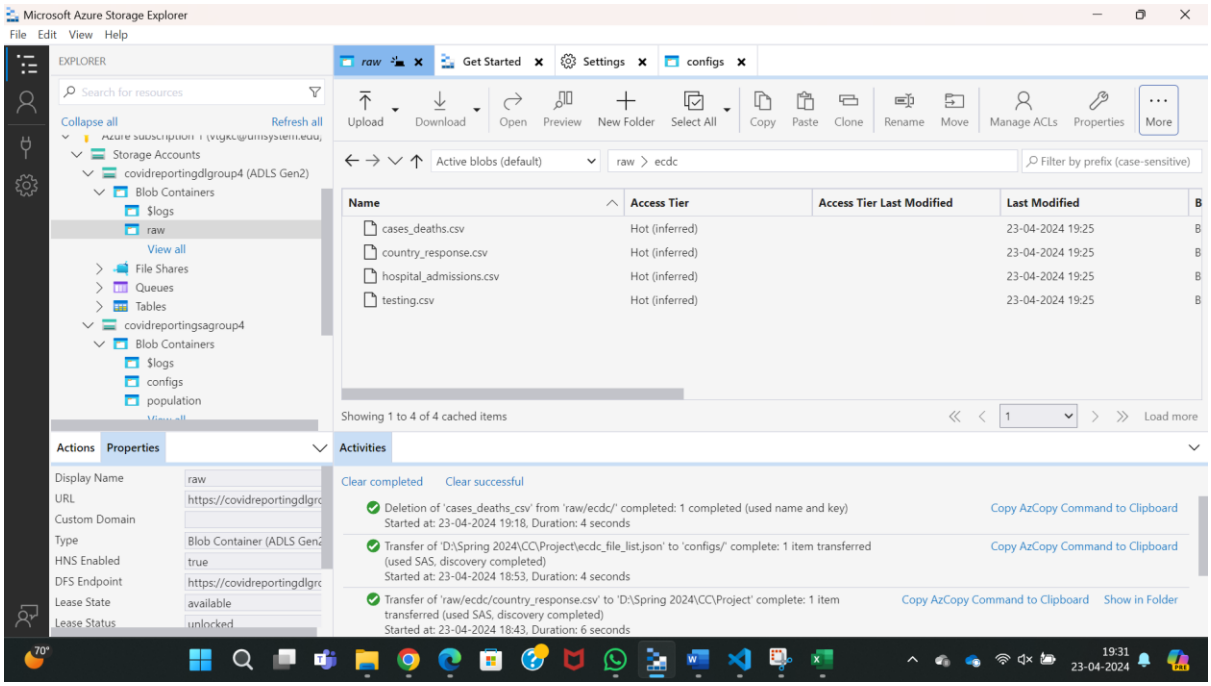
Create a new trigger, To update all the data automatically



Add the trigger in the pipeline and publish all.

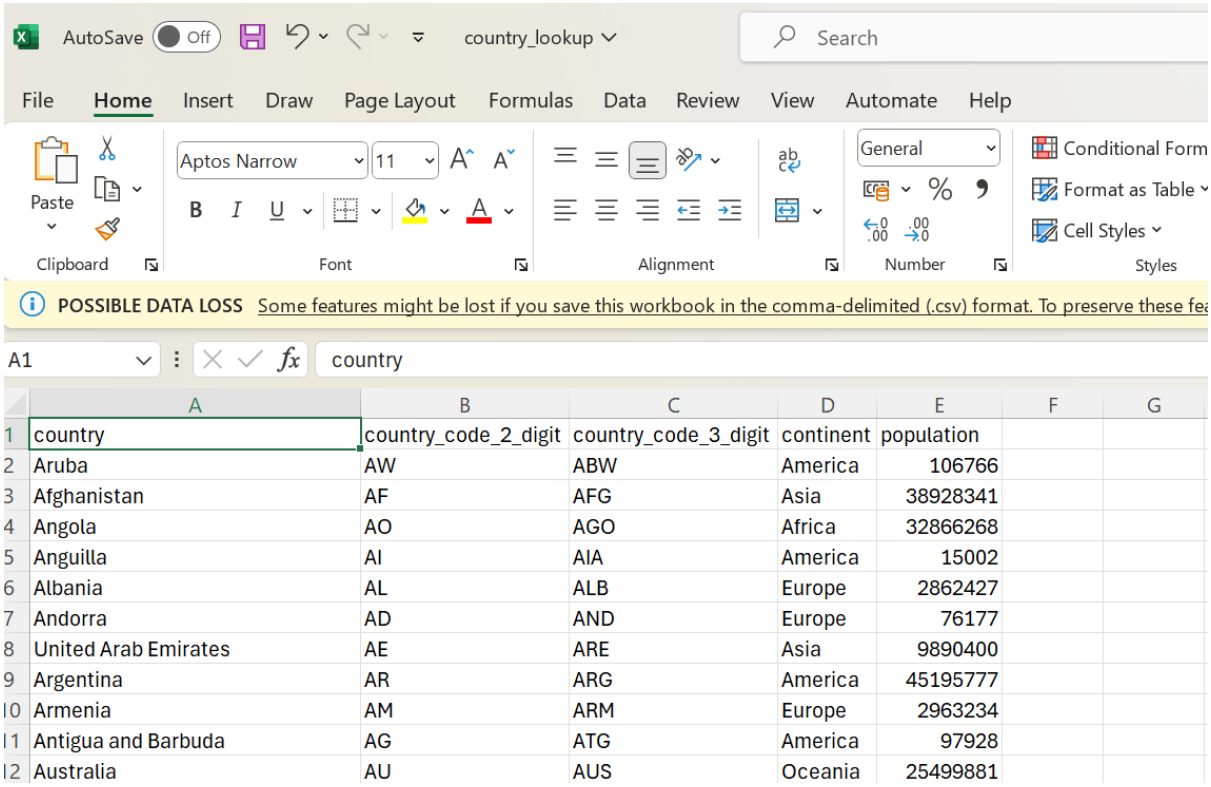


In the file explorer, we can see the data is updated.



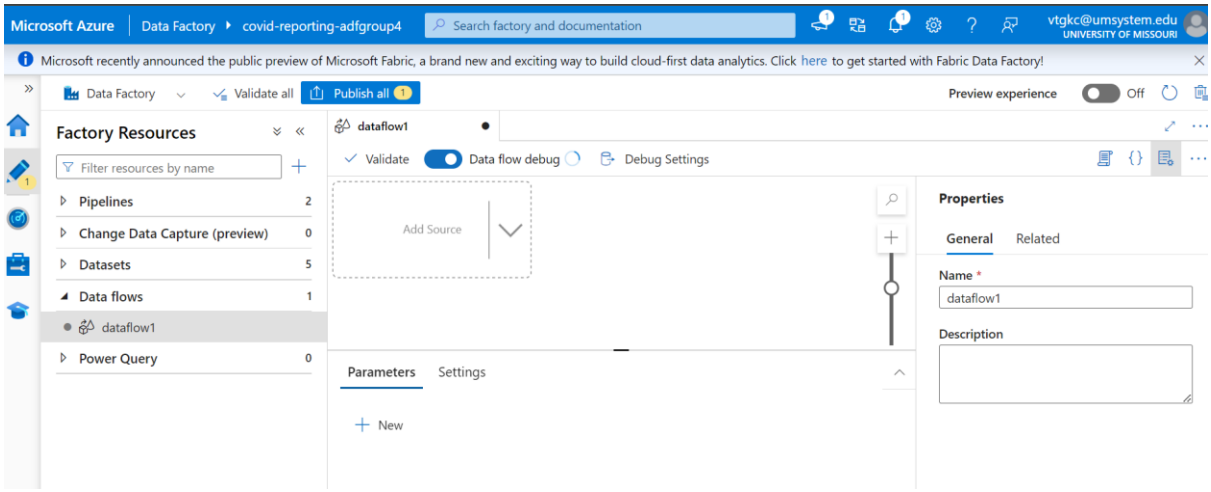
Data – flow: Data transformation

Transform cases and deaths data, only performing for country Europe, Created country lookup file by doing the lookup activity on cases deaths and testing file, to bring two digit and three-digit country code's together.

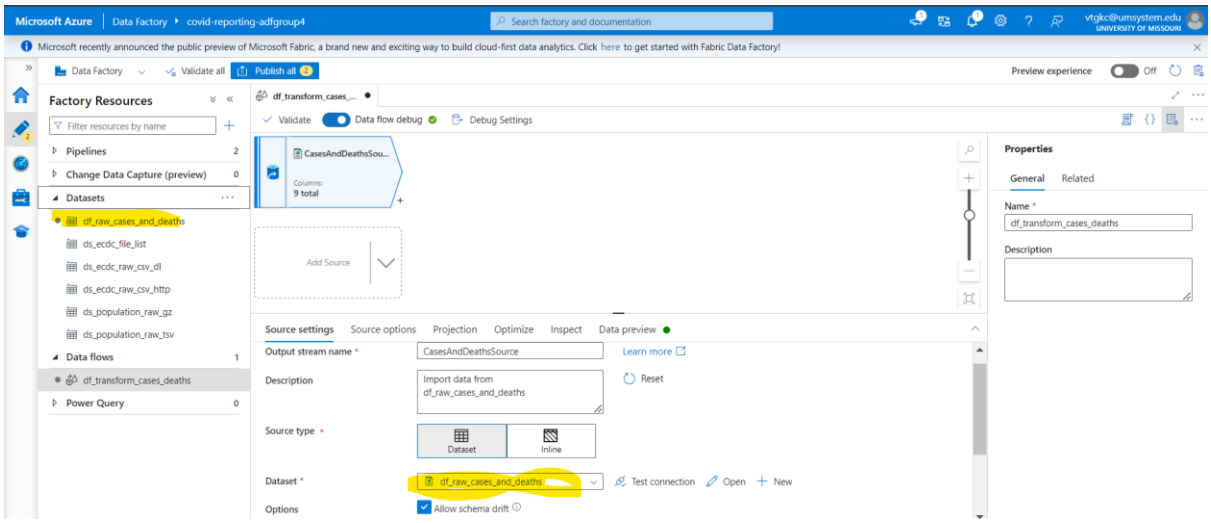


	A	B	C	D	E	F	G
1	country	country_code_2_digit	country_code_3_digit	continent	population		
2	Aruba	AW	ABW	America	106766		
3	Afghanistan	AF	AFG	Asia	38928341		
4	Angola	AO	AGO	Africa	32866268		
5	Anguilla	AI	AIA	America	15002		
6	Albania	AL	ALB	Europe	2862427		
7	Andorra	AD	AND	Europe	76177		
8	United Arab Emirates	AE	ARE	Asia	9890400		
9	Argentina	AR	ARG	America	45195777		
10	Armenia	AM	ARM	Europe	2963234		
11	Antigua and Barbuda	AG	ATG	America	97928		
12	Australia	AU	AUS	Oceania	25499881		

Create a new data flow, Enable the data flow debug,

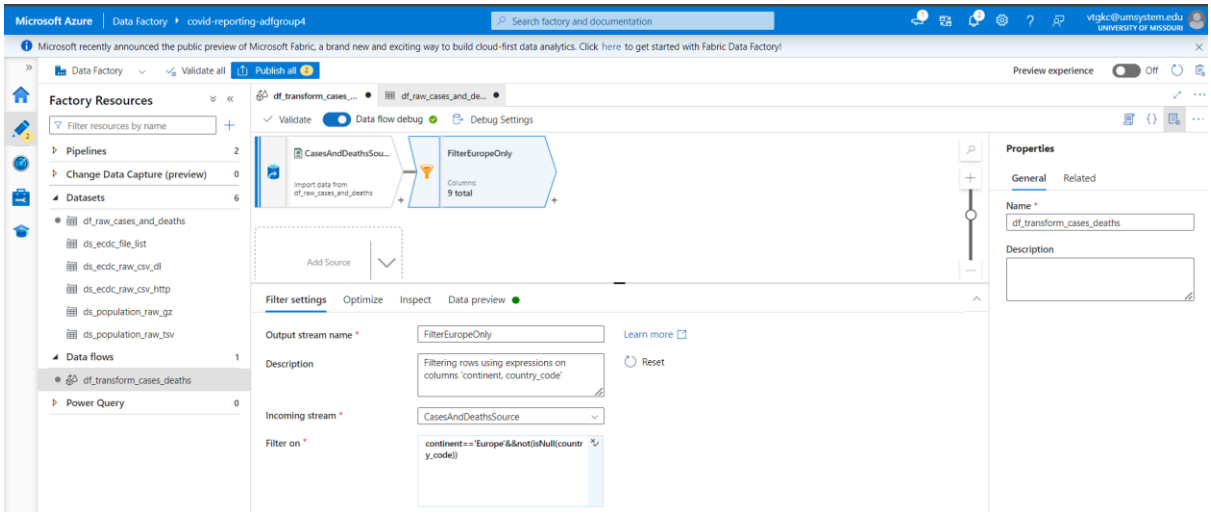


Create a source and give the cases death file as source, you can see the new file in the datasets



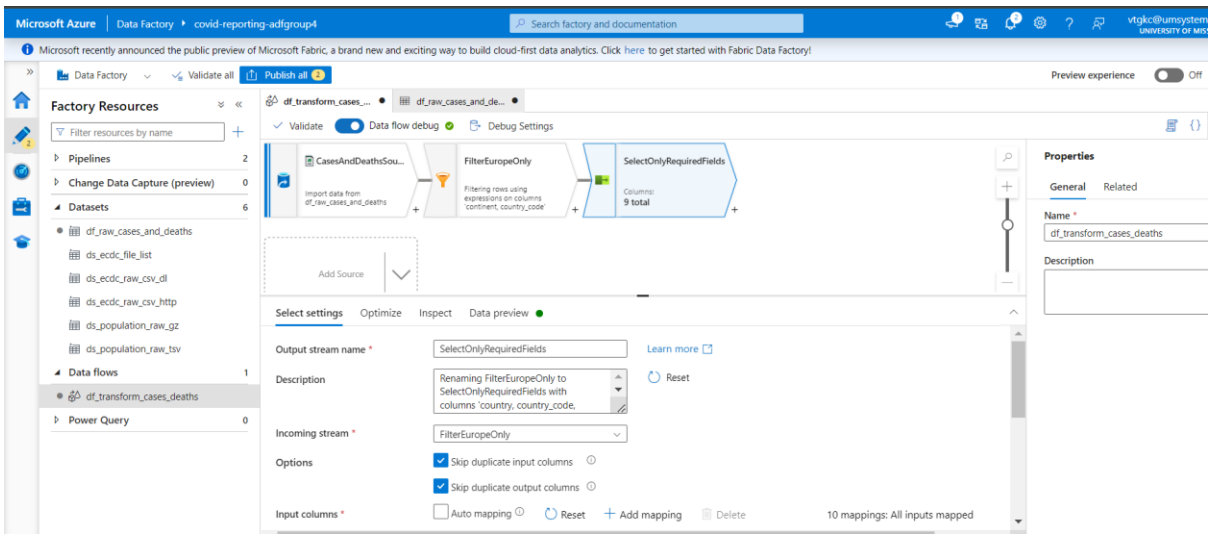
Filter transformation:

Add the filter to the source using the + icon, filter condition as Europe country. Also went through other options in Filter to explore.



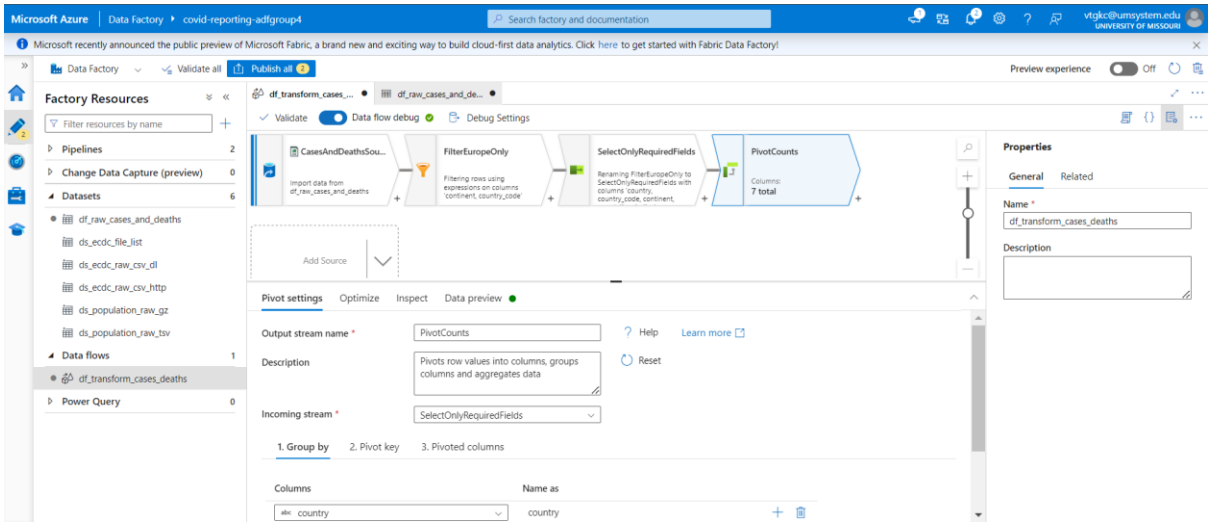
Select transformation:

create a select transform, using + icon. Also went through other options in select to explore.

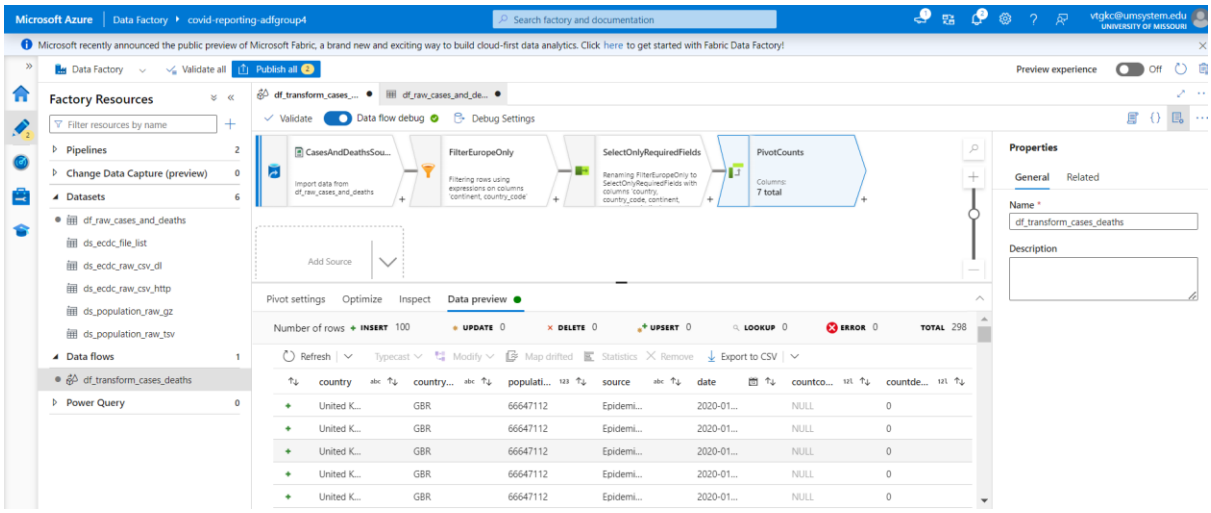


Pivot transformation:

create a pivot transform, using + icon.

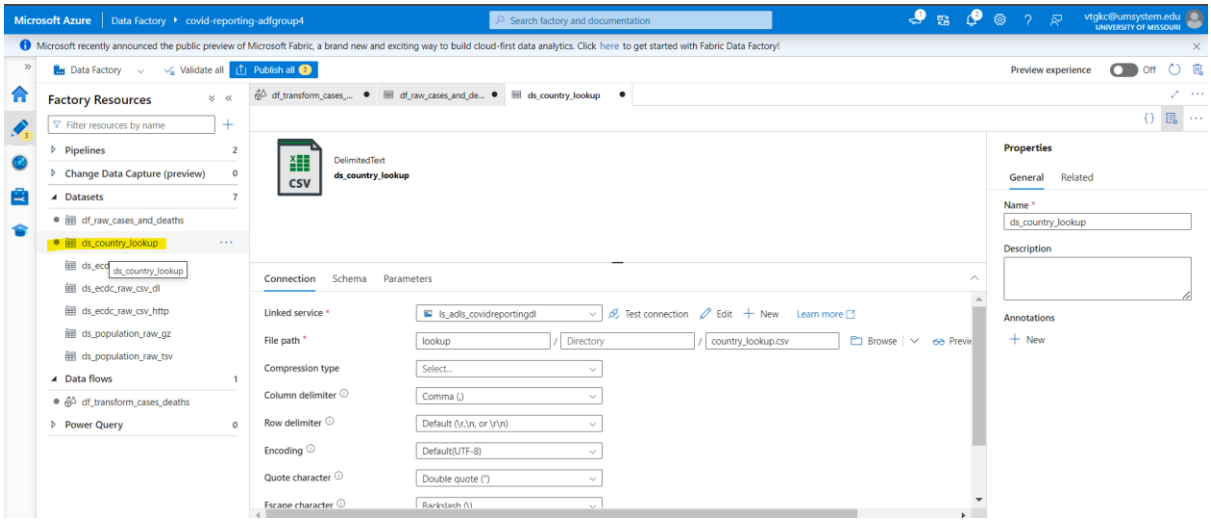


In Data preview tab, two new columns as countdeaths and countconfirmed cases

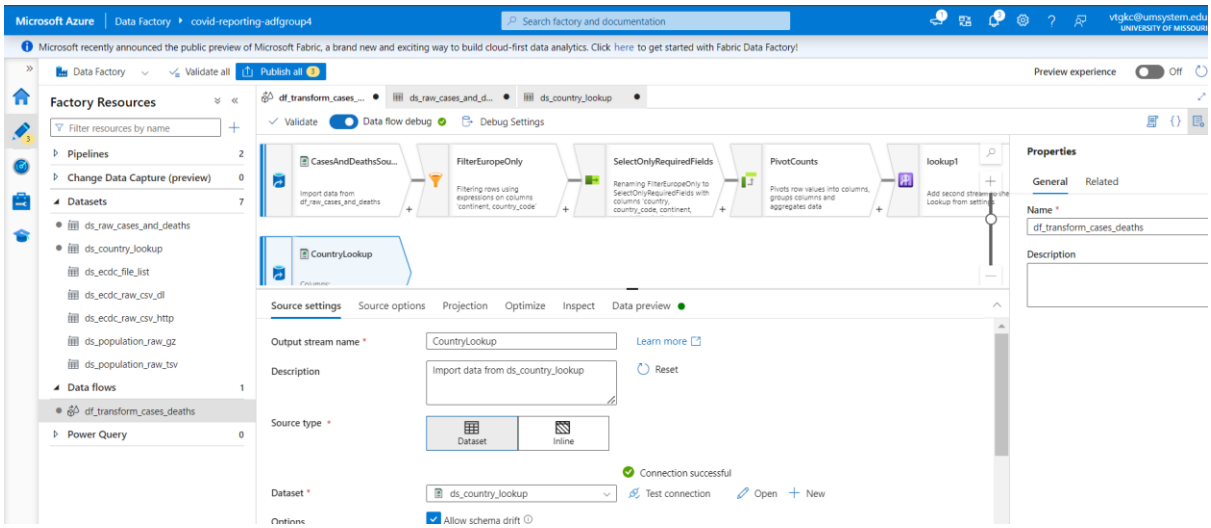


Lookup transformation:

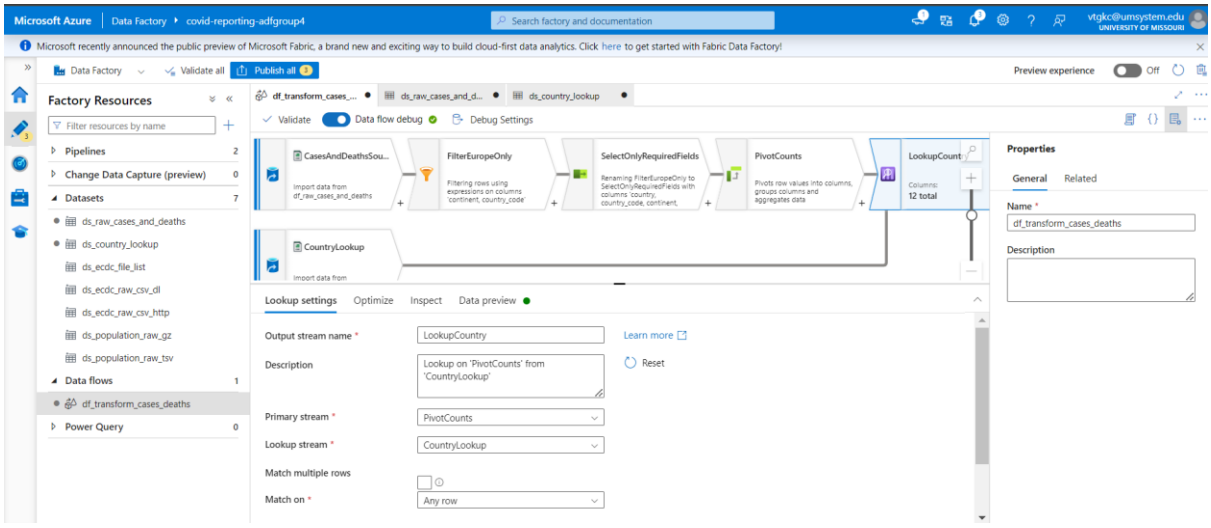
Create a new dataset for lookup, --- countrylookup



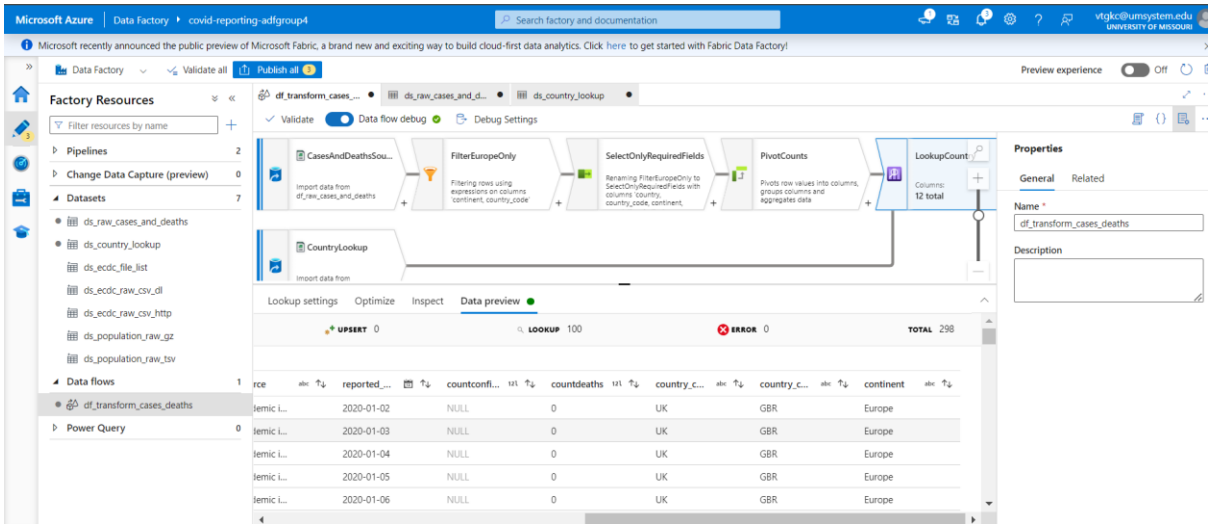
Create a new source for the country lookup,



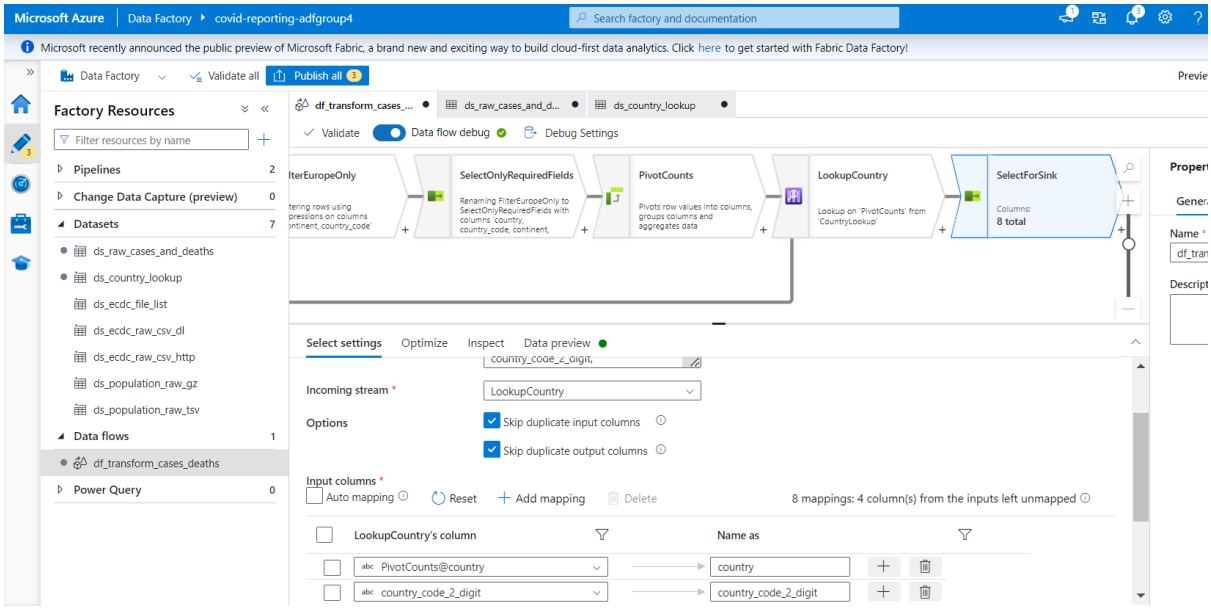
Join the lookup with the countrylookup source



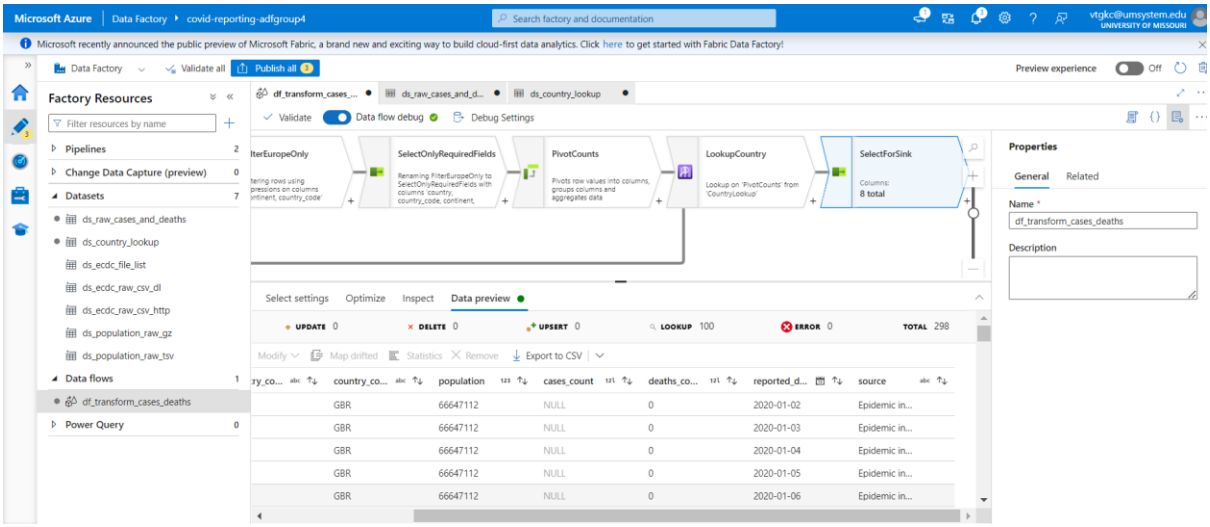
In data preview you can see two new columns two digit and three digit country code’s



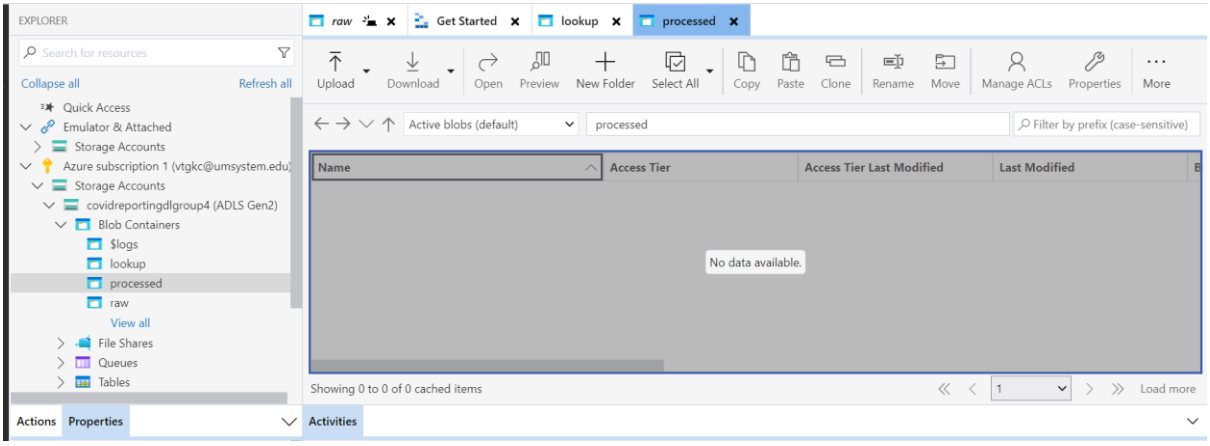
Create the select, after lookup as selectforsink,



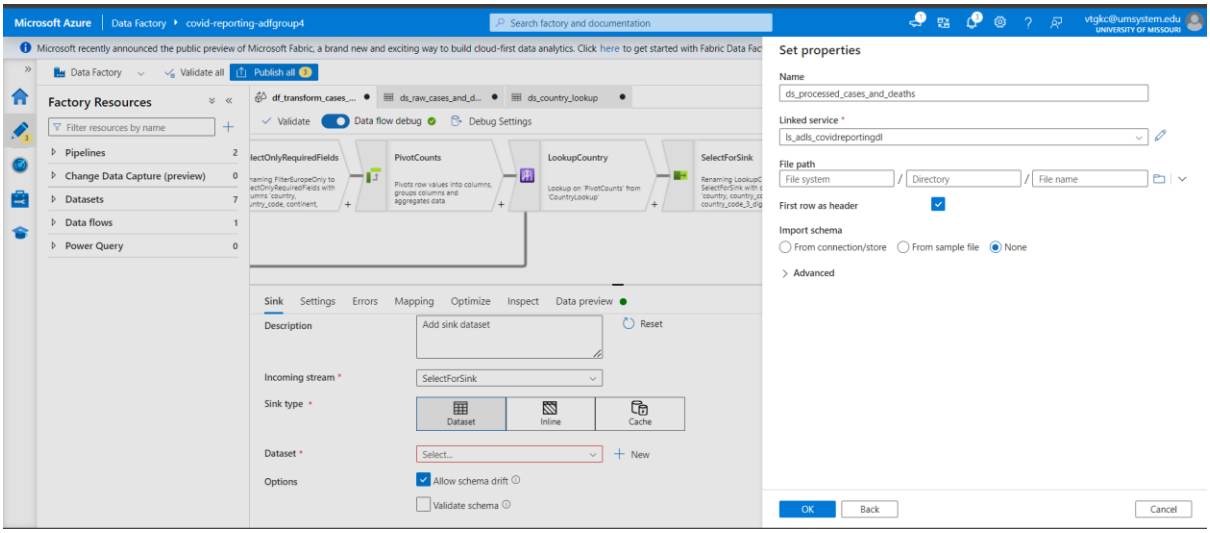
In data preview, we can see the columns,



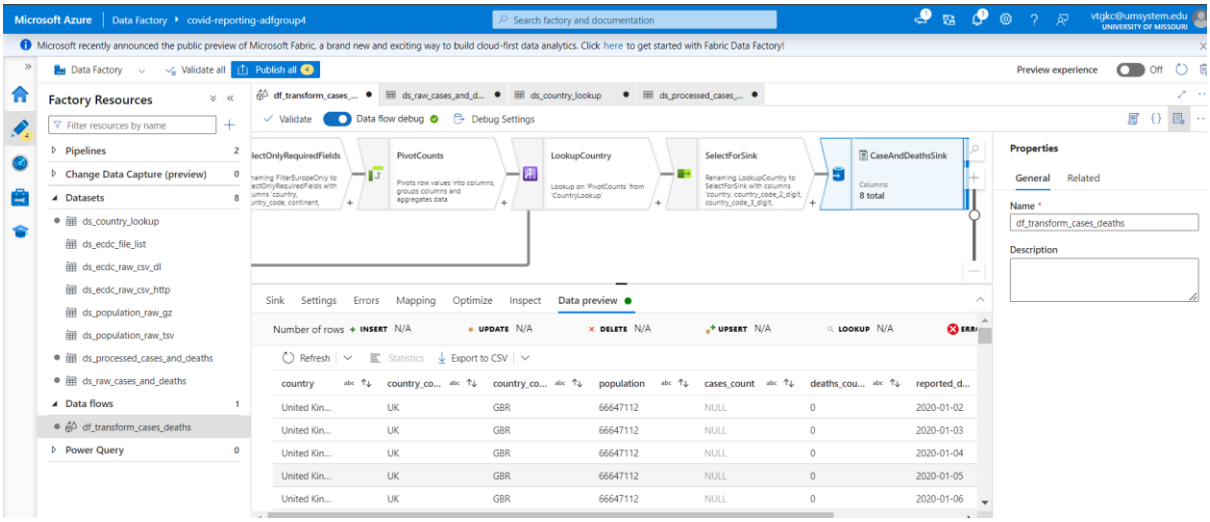
Create a Sink after the select, create a new dataset in the explorer with name processed,



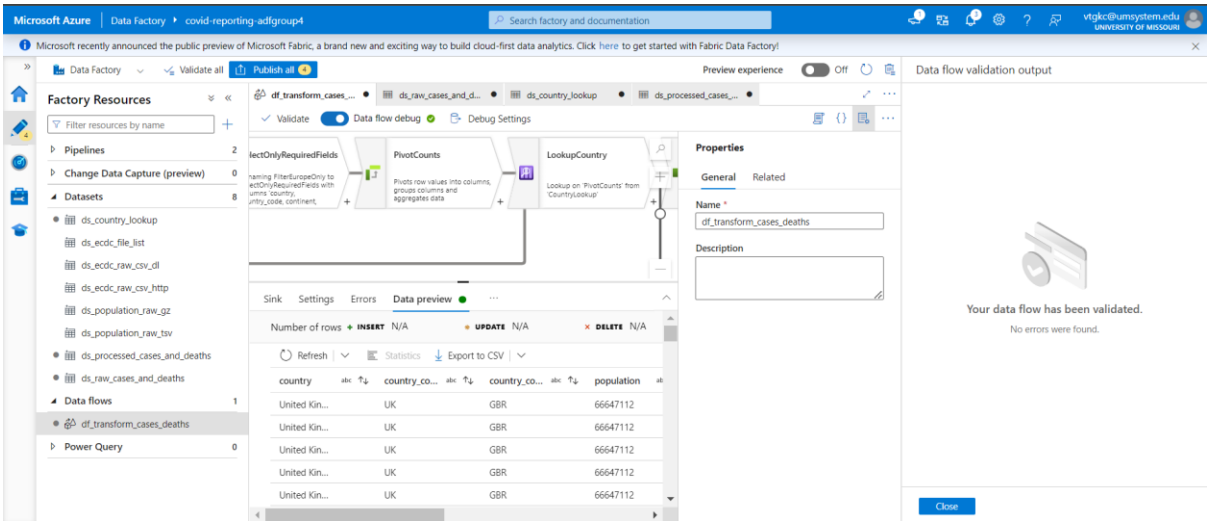
In the sink create a new dataset,

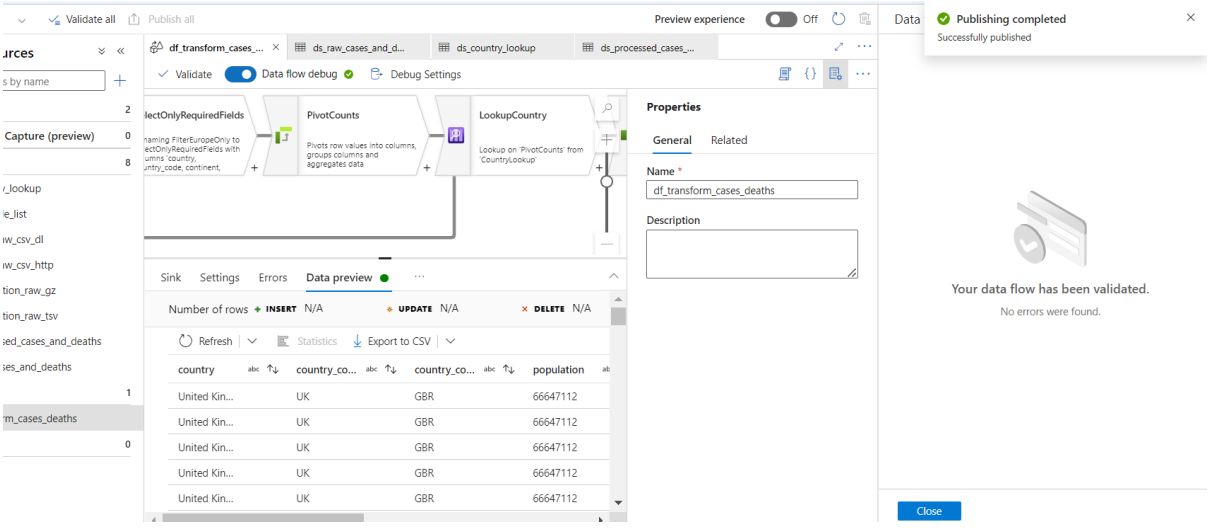


The data preview in sink,

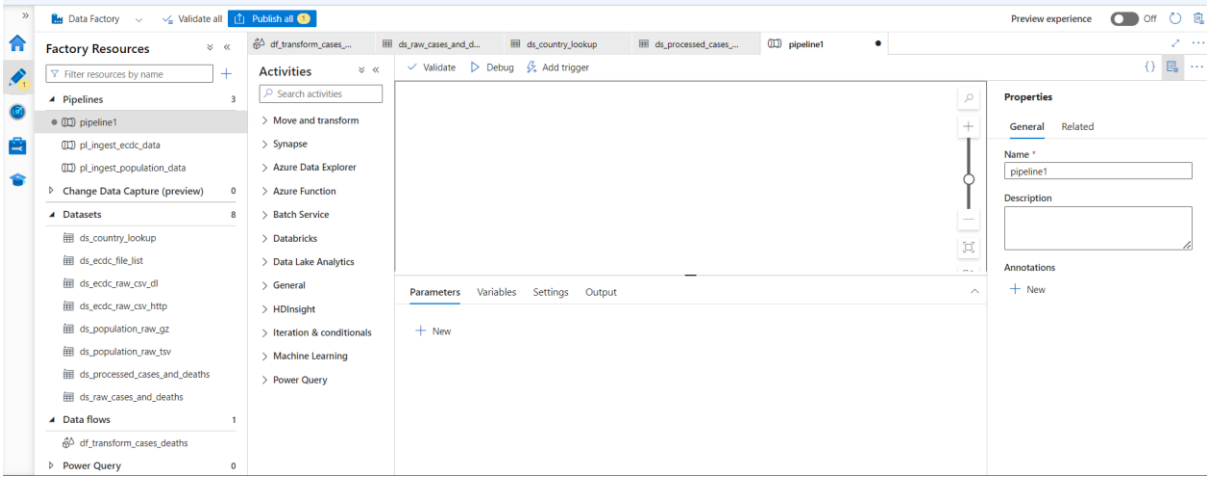


Now give validate and publish,

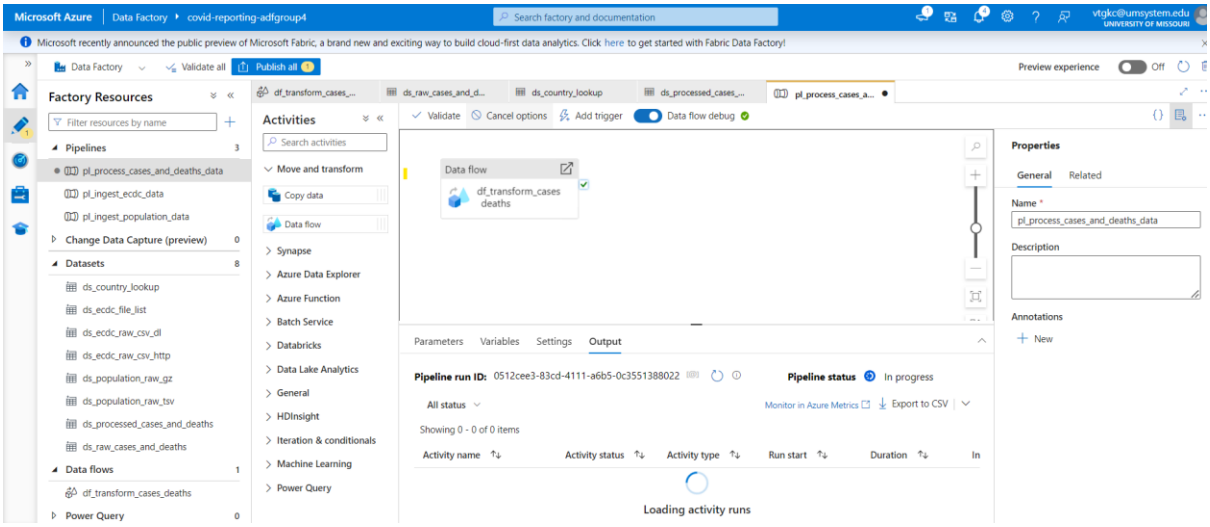




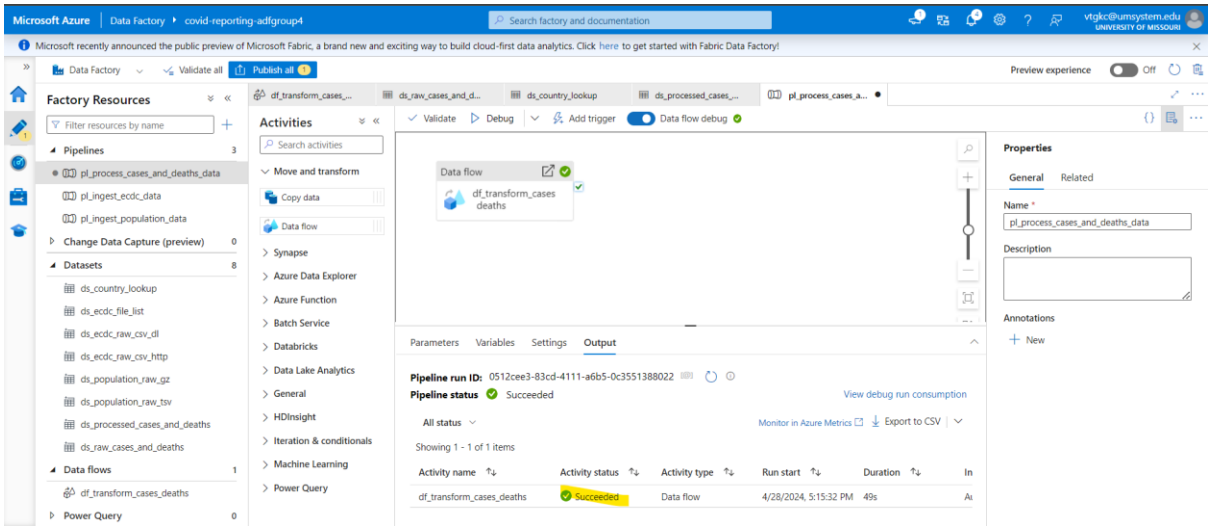
Create ADF pipeline:



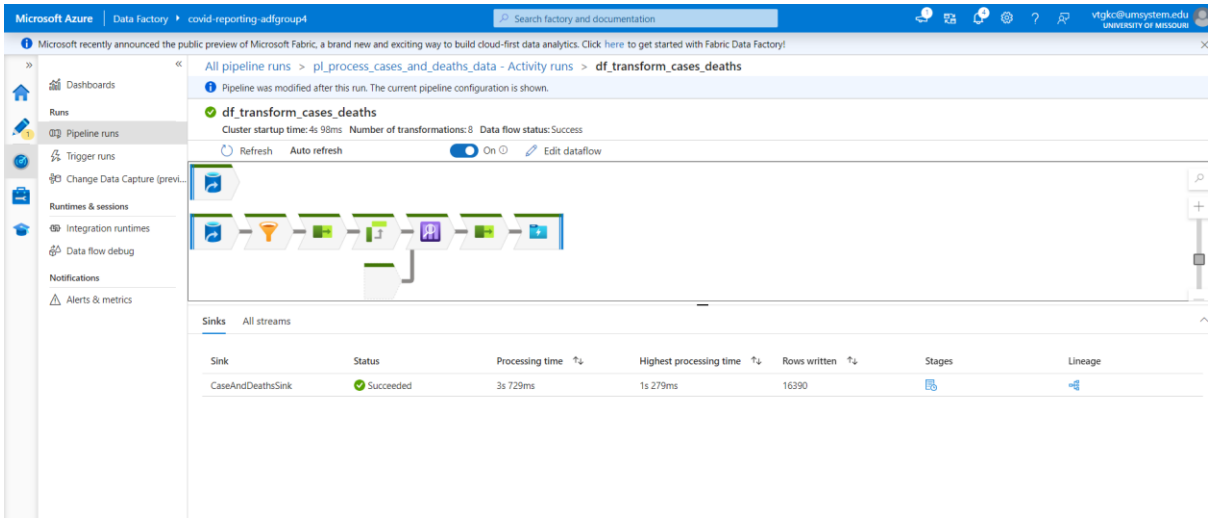
In activity, add data flow, select the data flow and hit debug



Wait till success,

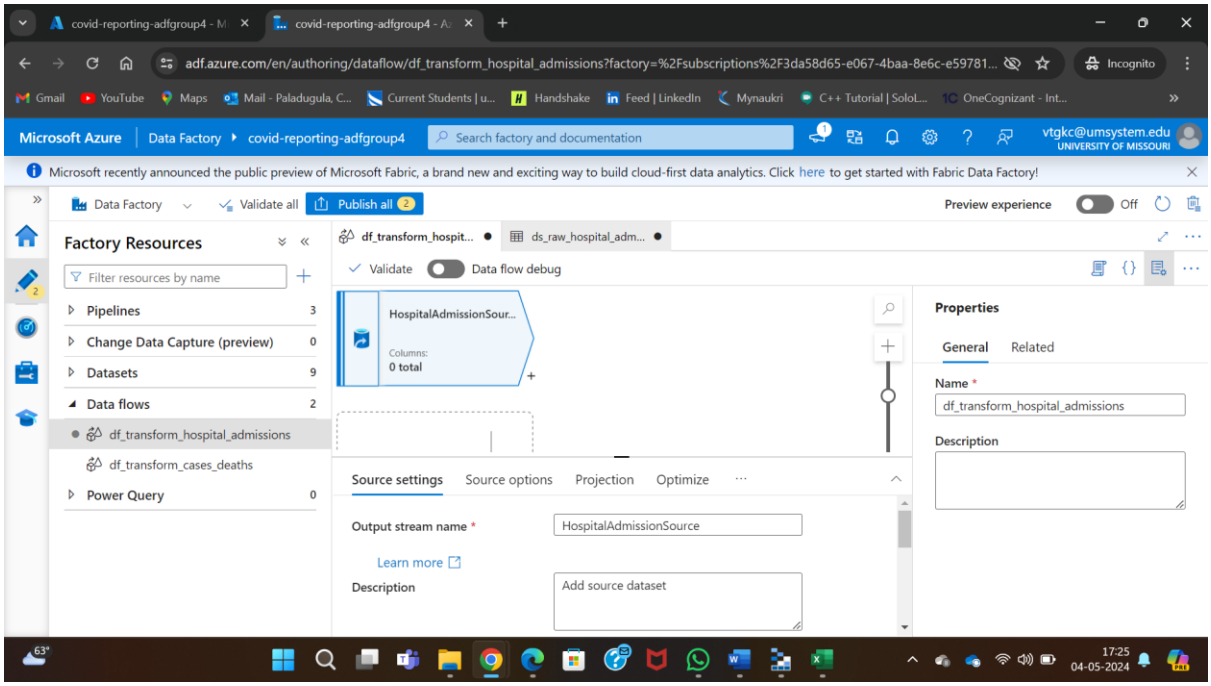


Pipeline preview,

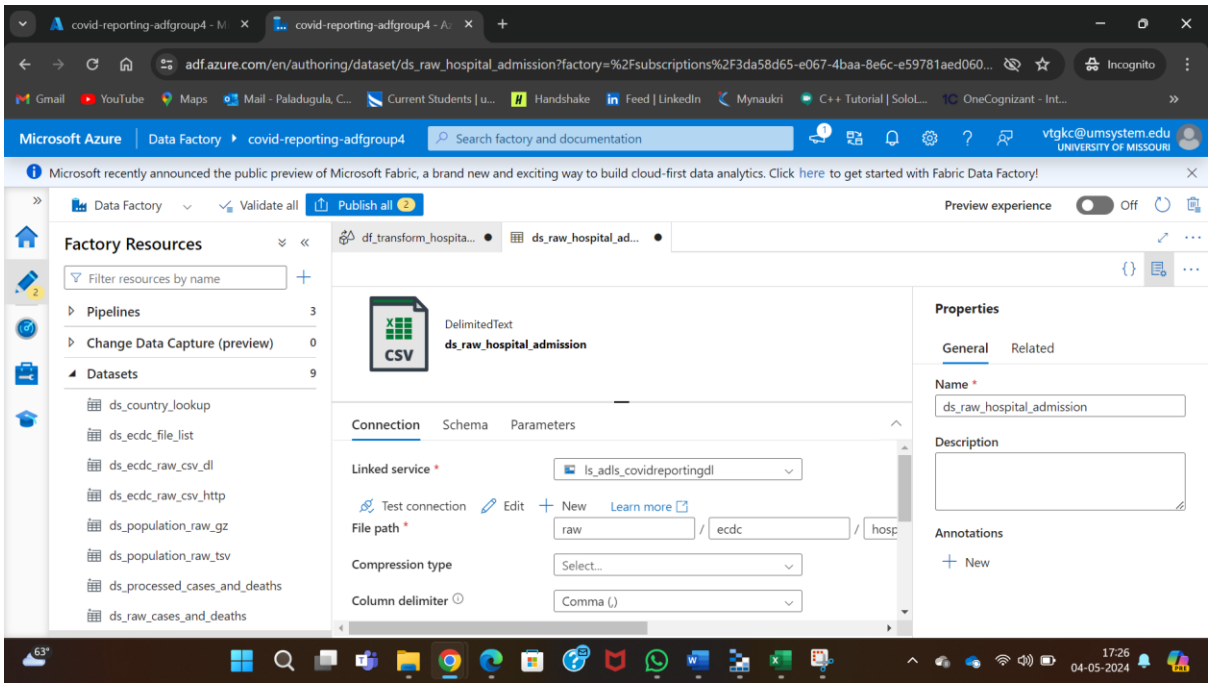


Data Flow – hospital admissions and data transformation

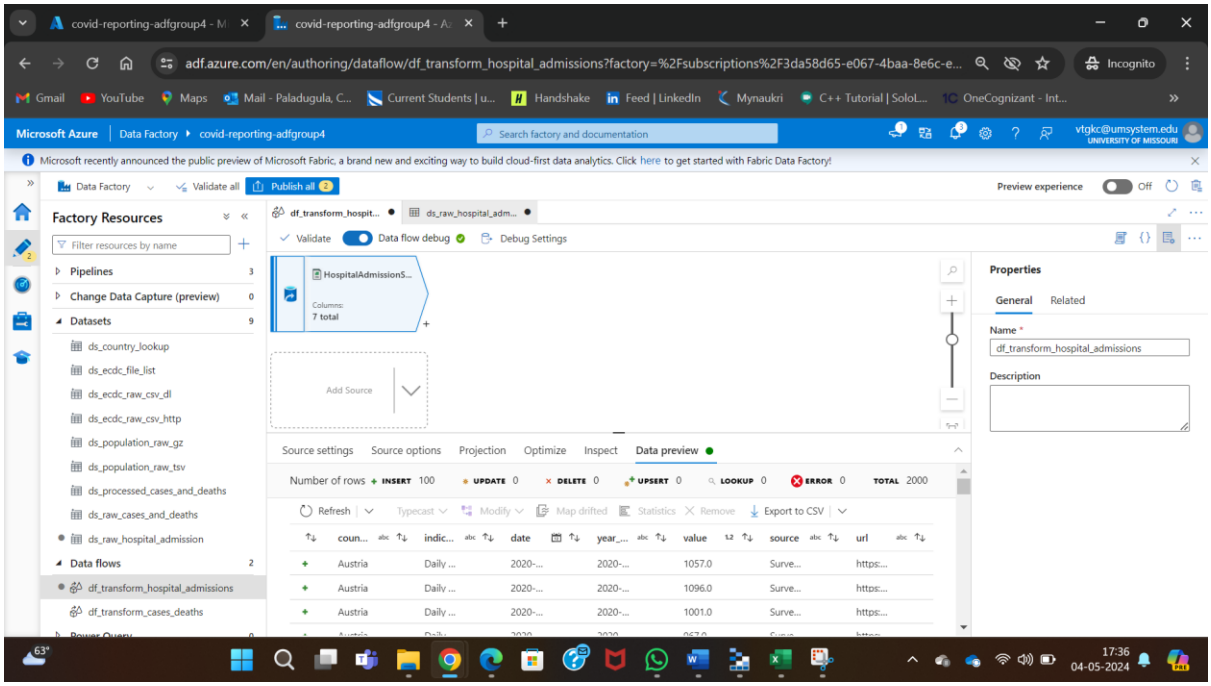
Create a new DataFlow, name – df_transform_hospital_admissions



Create a new data set,

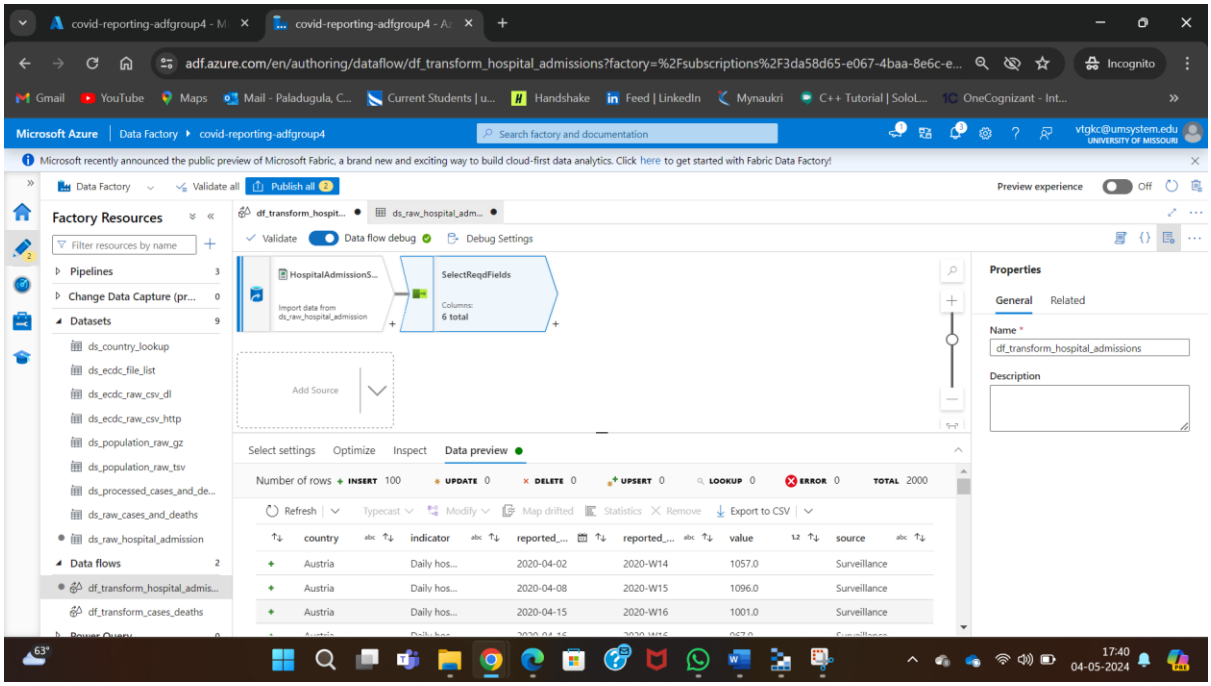


Create a source transformation, you can see the Data preview,

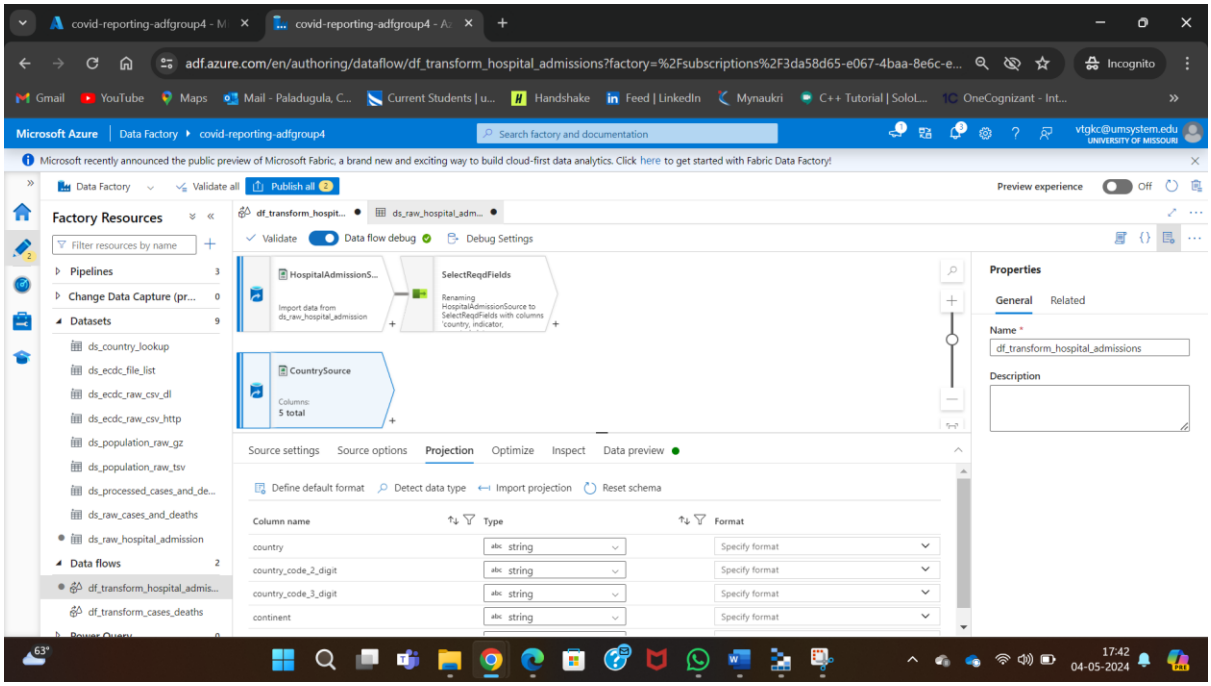


Create a select transformation to remove some fields, also to rename some fields,

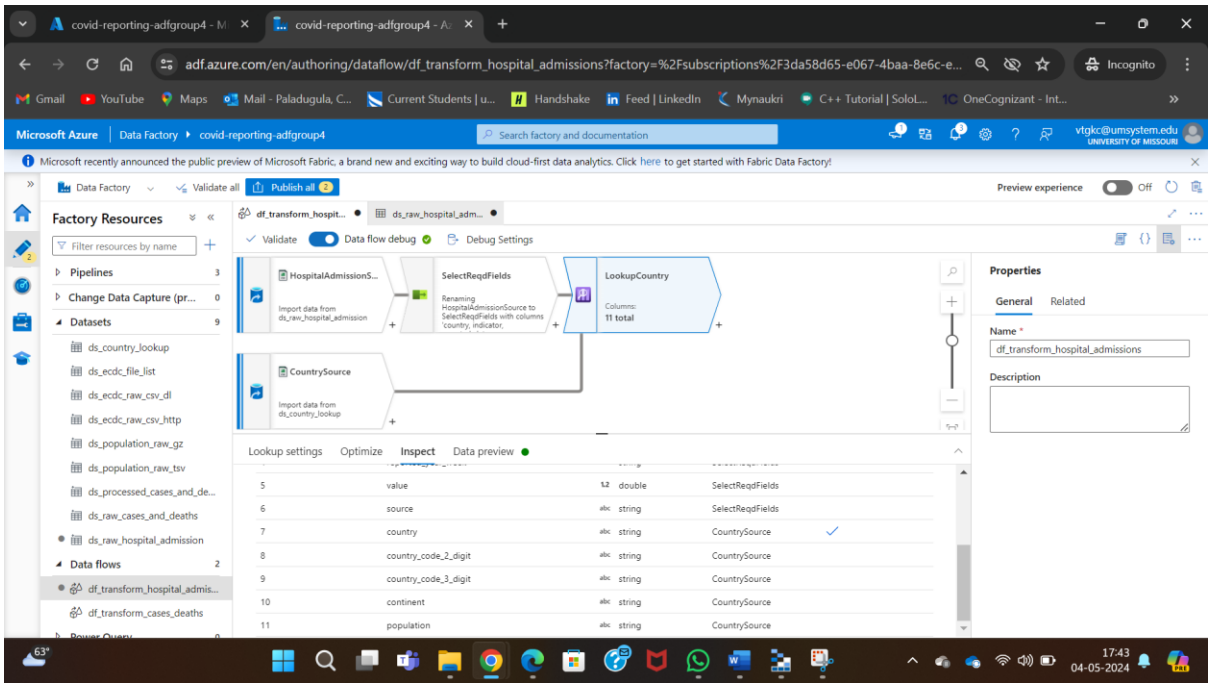
You can see the data preview the new fields and name changes



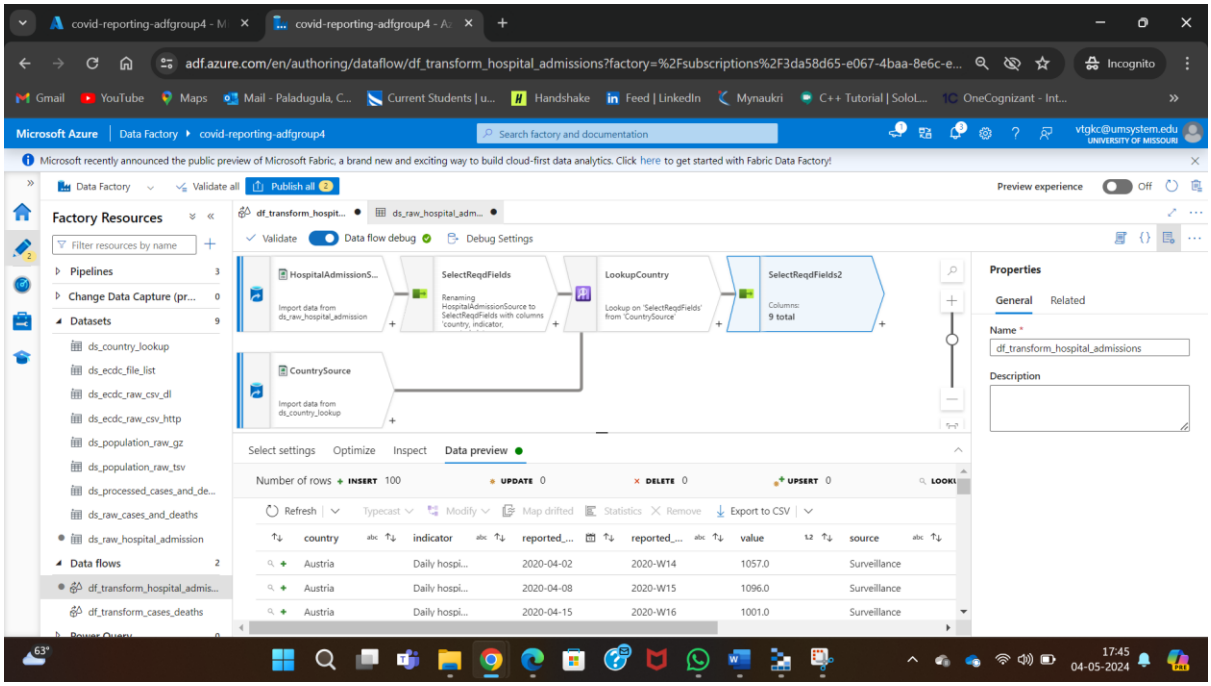
Create new source for the lookup ,



Now create the lookup,



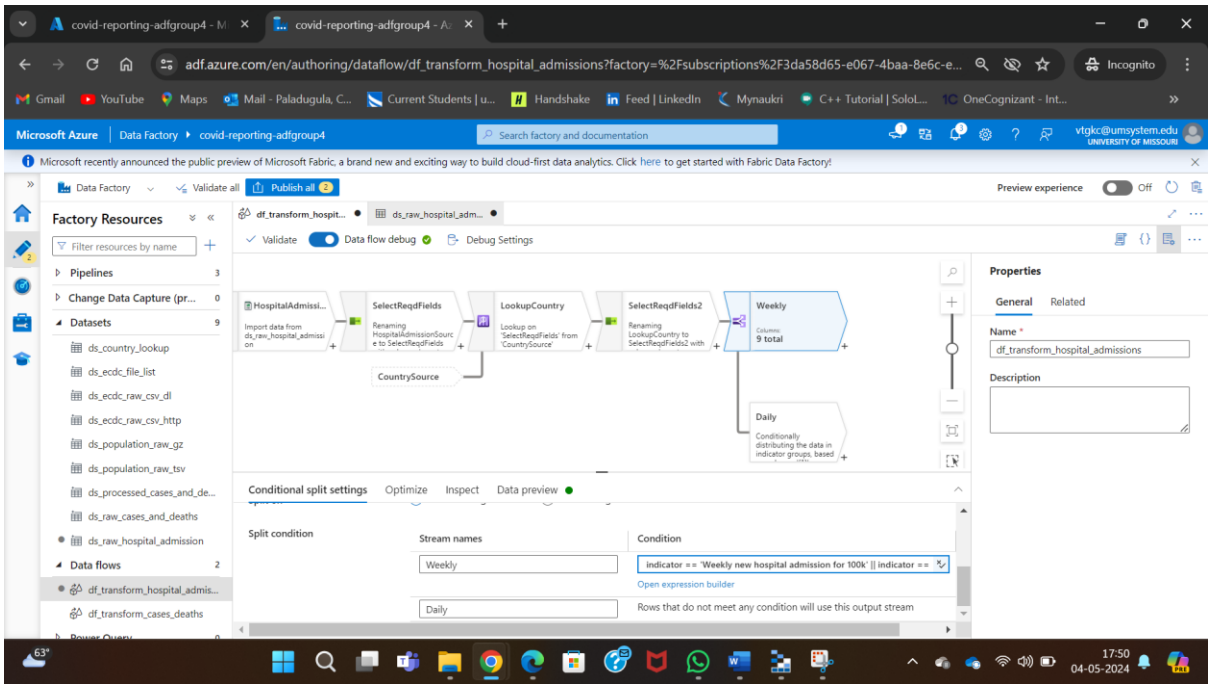
Create a new select to remove some fields, to remove the duplicate columns and continents



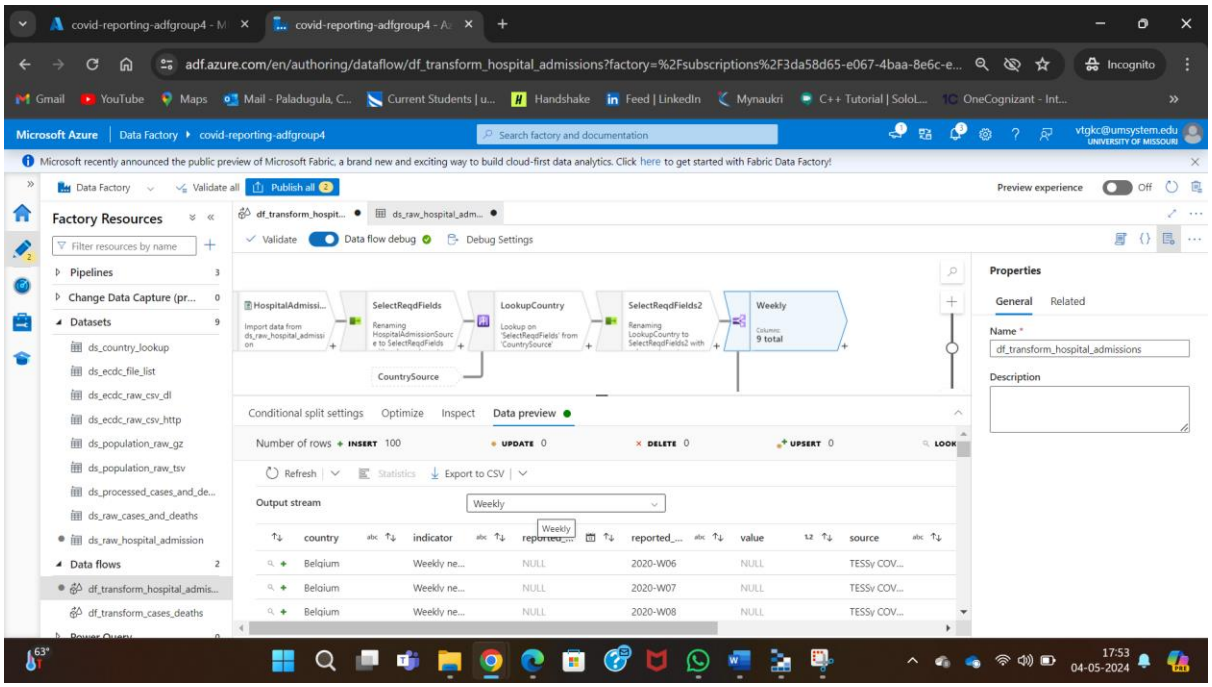
Now we split the data into 2 streams, one for the daily info and the other one for weekly,

We create a conditional split, here is the condition given for the split.

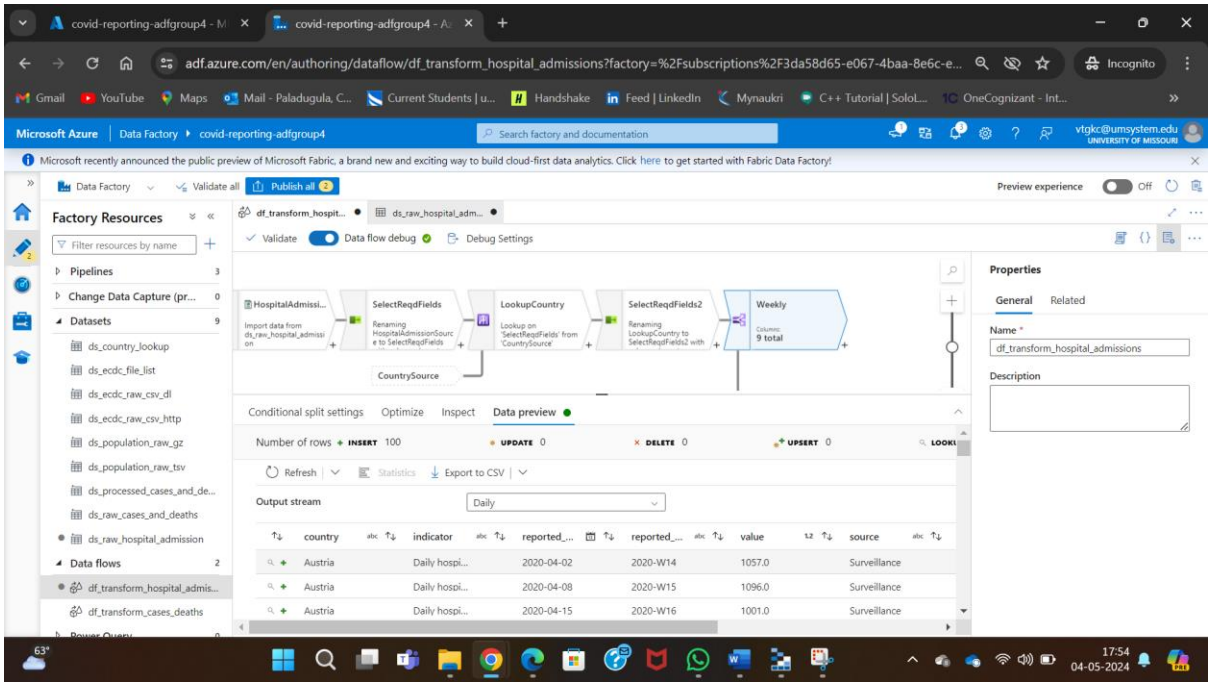
Weekly condition = “indicator == 'Weekly new hospital admissions per 100k' || indicator == 'Weekly new ICU admissions per 100k'”



Here is the data preview, for the week

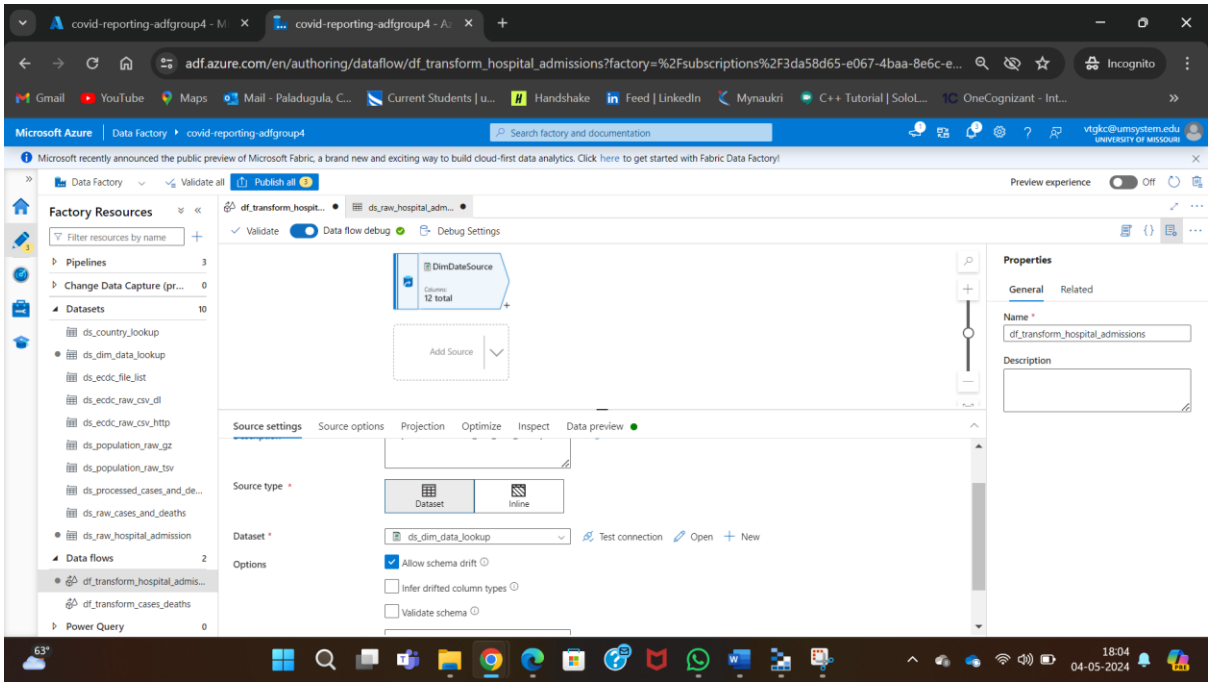


For the daily,

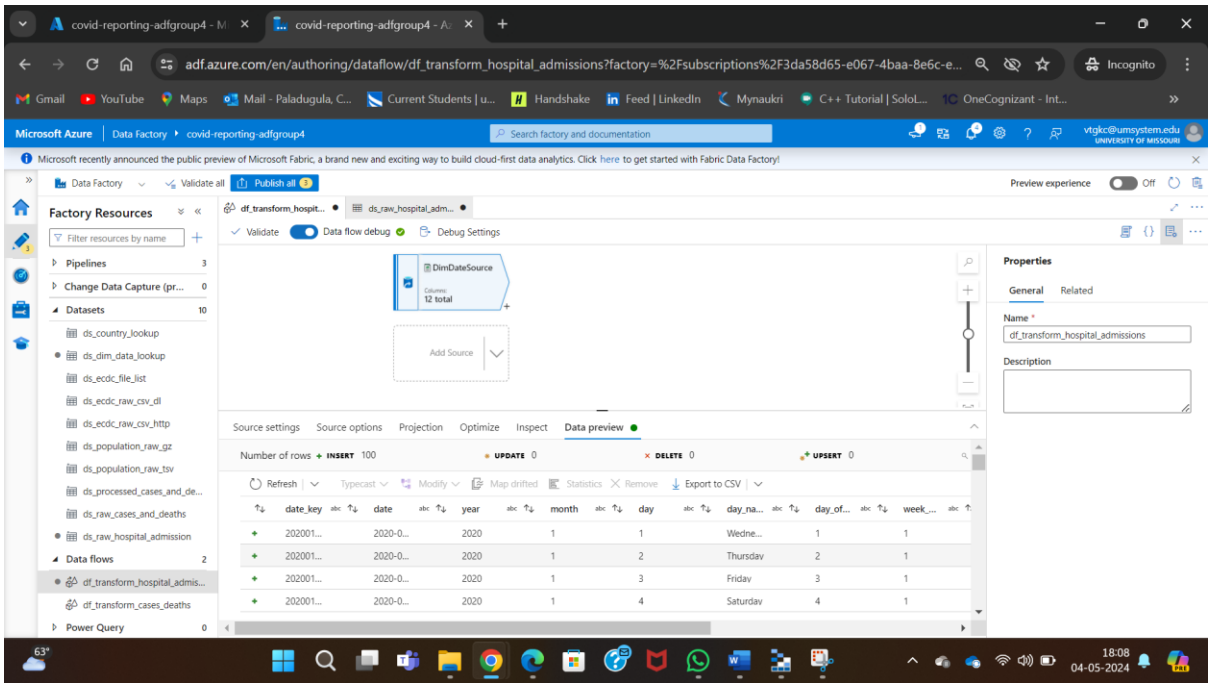


Create a Source transformation, for dim date

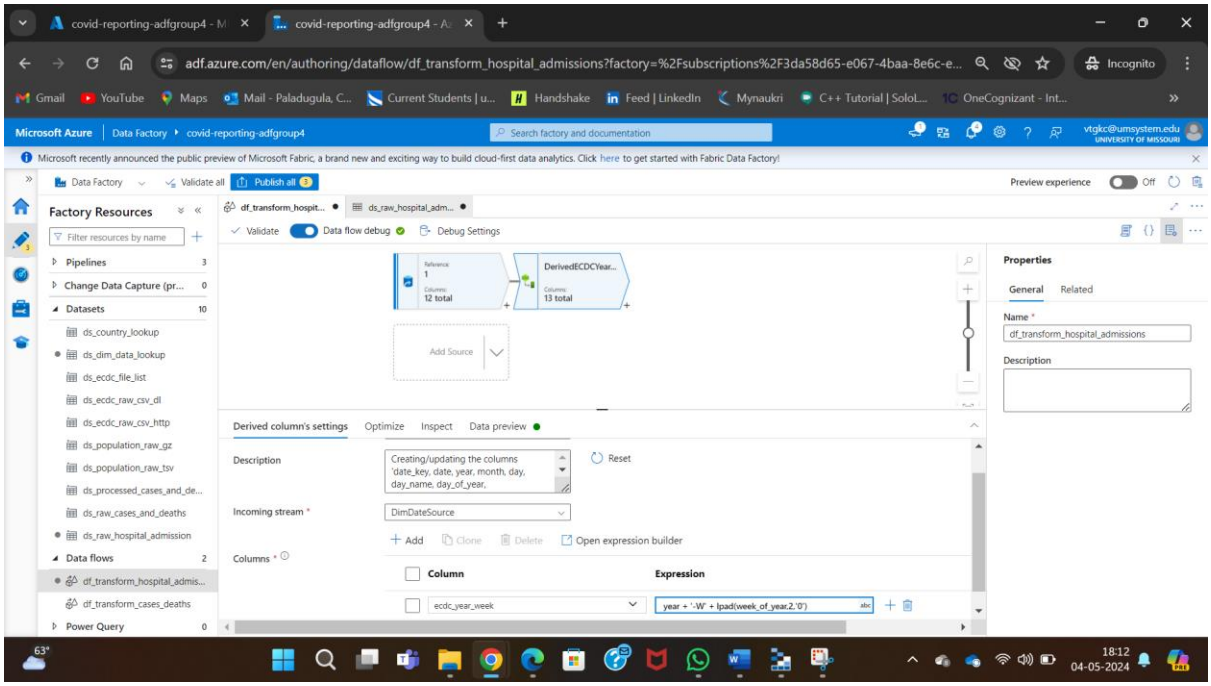
Create a new Source , and add a new dataset as well



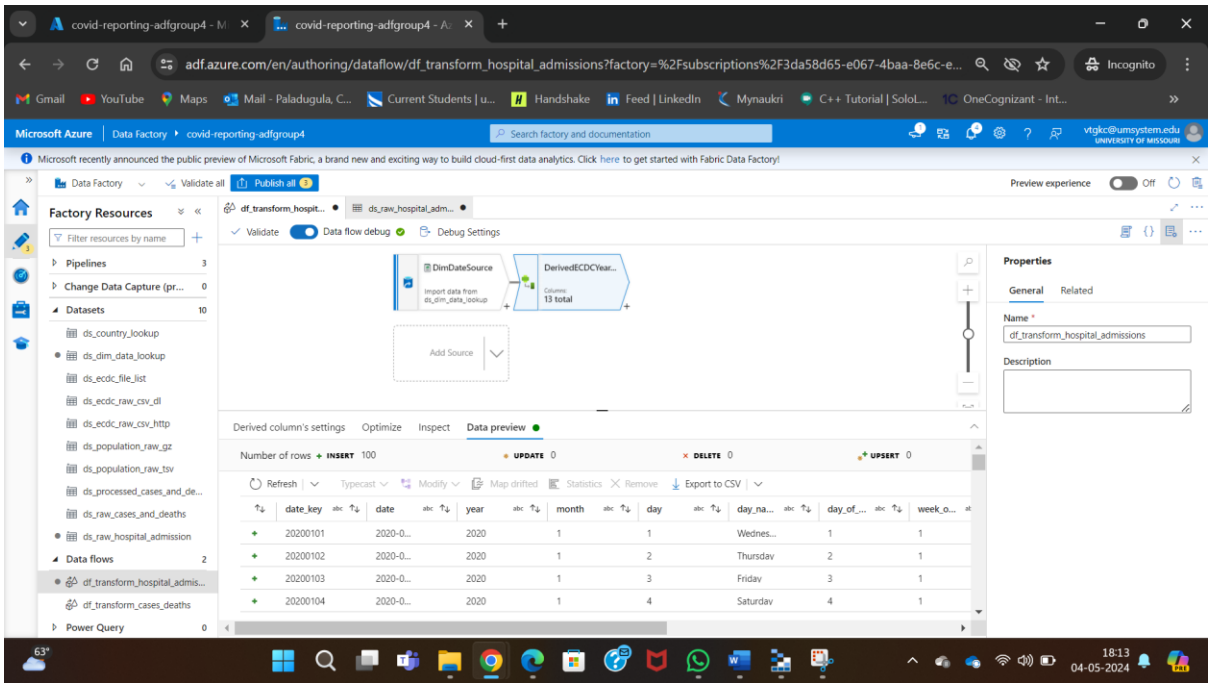
Here is the data preview,



Now we combine two columns (year , week of the year as ecdc_year_week) using derive transformation, you see the condition for combining them

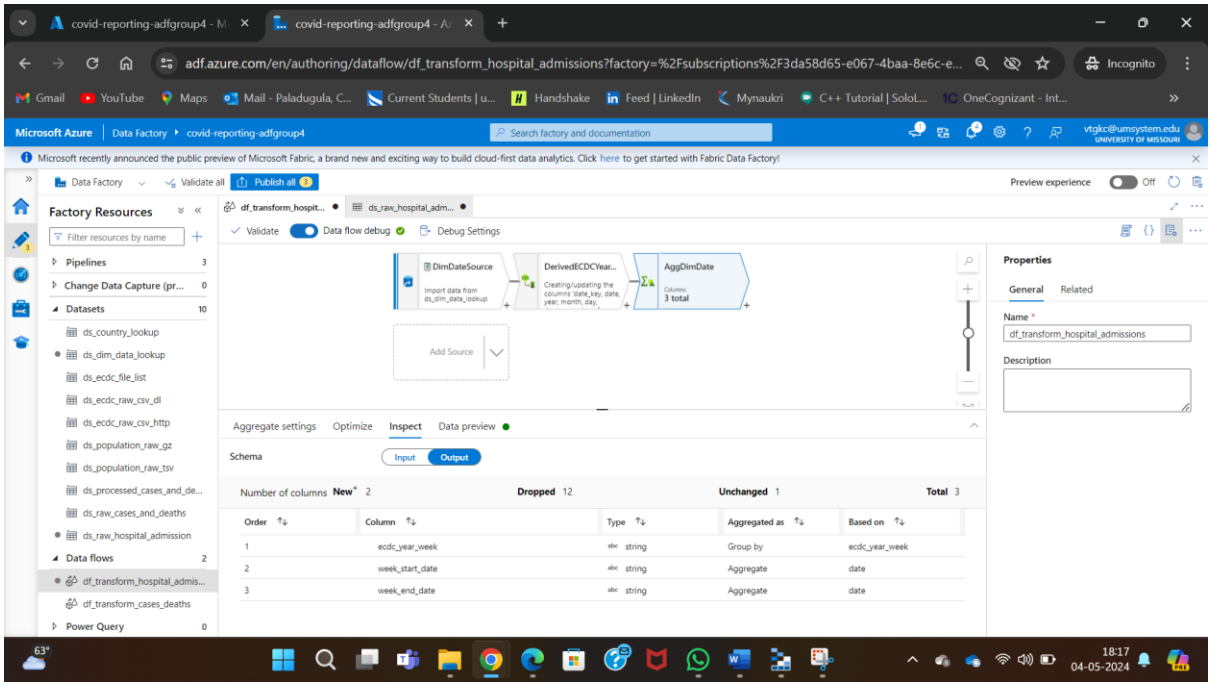


Here is the data preview,



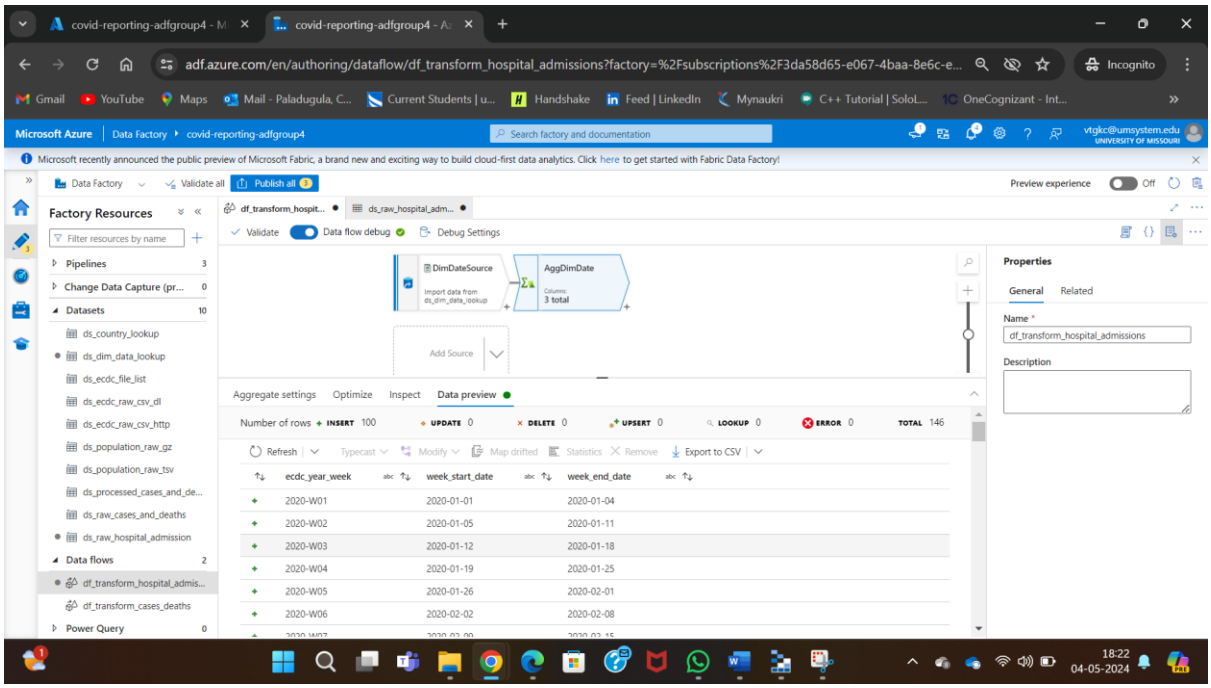
Now we create a single record to get one record for week,

Create a aggregate transformation, We used both group by and aggregate conditions,



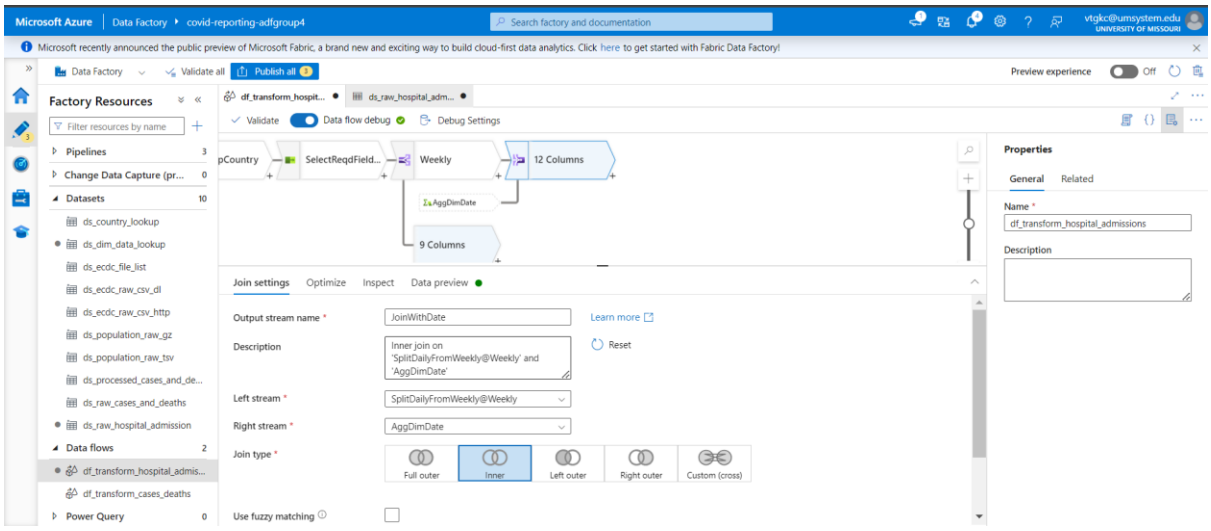
Here is the data preview, after performing the derived column condition in the aggregate condition, year + '-W' + lpad(week_of_year,2,'0')

We get the same results even after deleting the derived column,

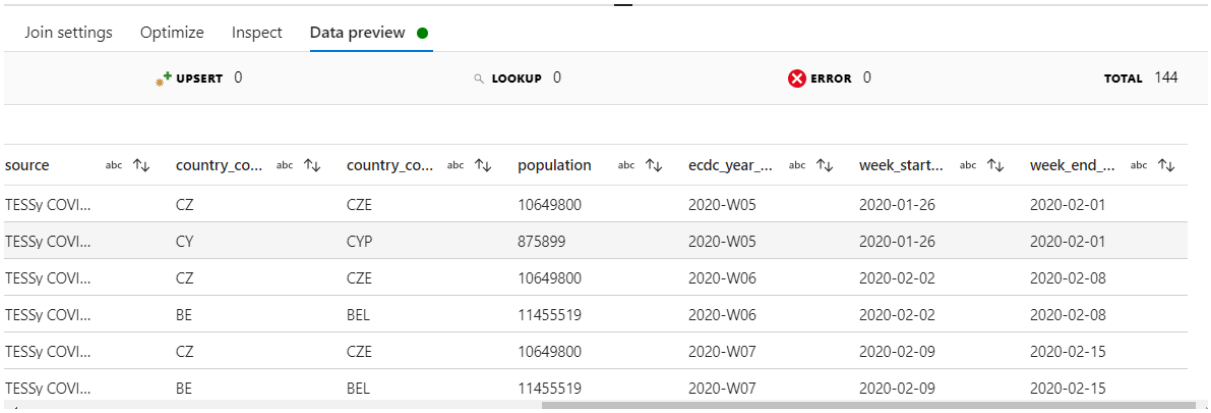


Now we join the data from hospital weekly and dimdata source stream

Now create join transformation



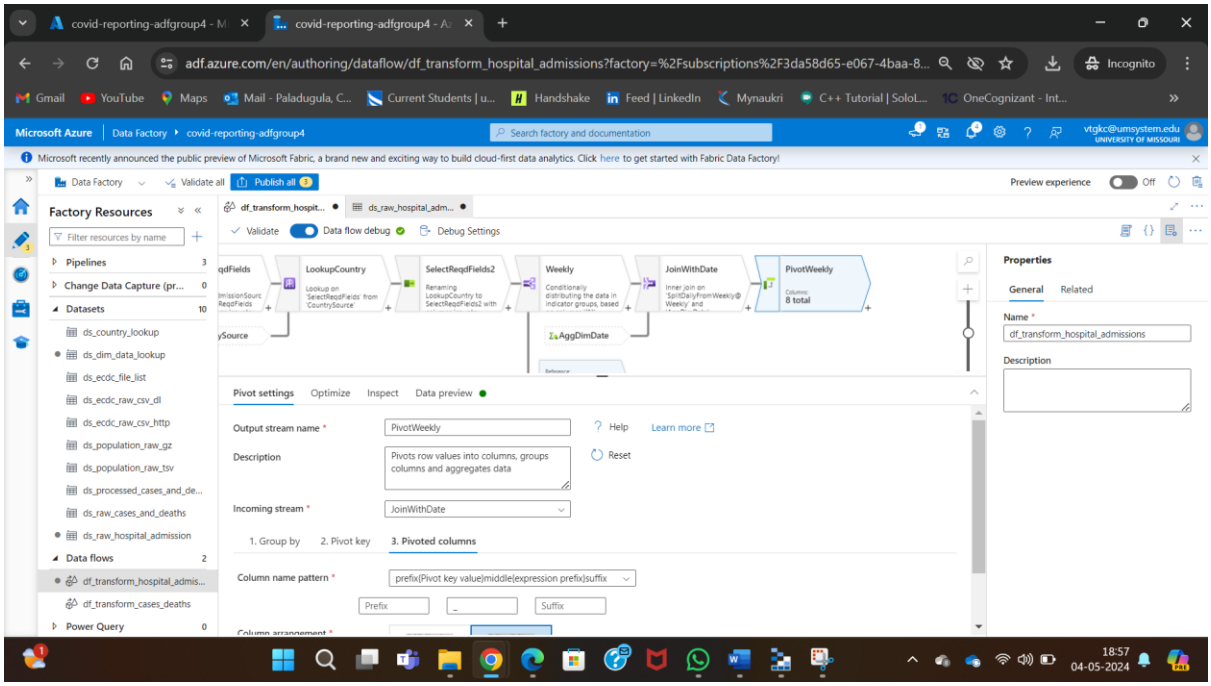
data preview



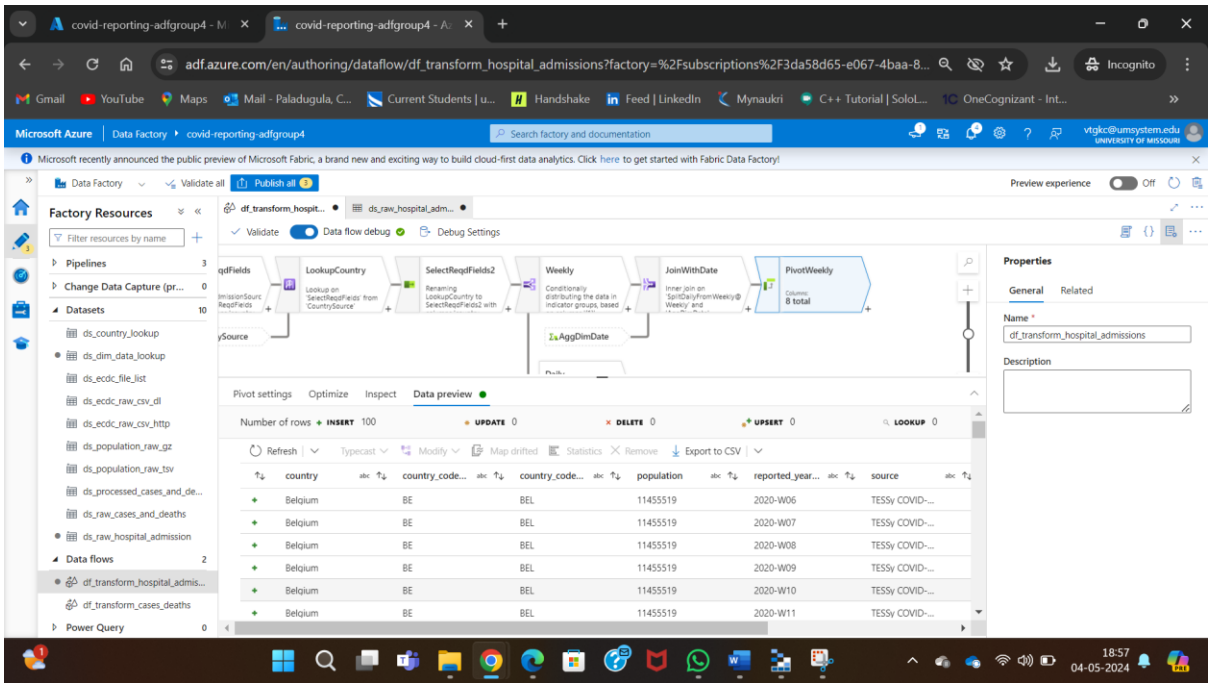
source	country_co...	country_co...	population	ecdc_year_...	week_start...	week_end_...
TESSy COVI...	CZ	CZE	10649800	2020-W05	2020-01-26	2020-02-01
TESSy COVI...	CY	CYP	875899	2020-W05	2020-01-26	2020-02-01
TESSy COVI...	CZ	CZE	10649800	2020-W06	2020-02-02	2020-02-08
TESSy COVI...	BE	BEL	11455519	2020-W06	2020-02-02	2020-02-08
TESSy COVI...	CZ	CZE	10649800	2020-W07	2020-02-09	2020-02-15
TESSy COVI...	BE	BEL	11455519	2020-W07	2020-02-09	2020-02-15

Now we perform pivot transformation, to combine daily and weekly streams

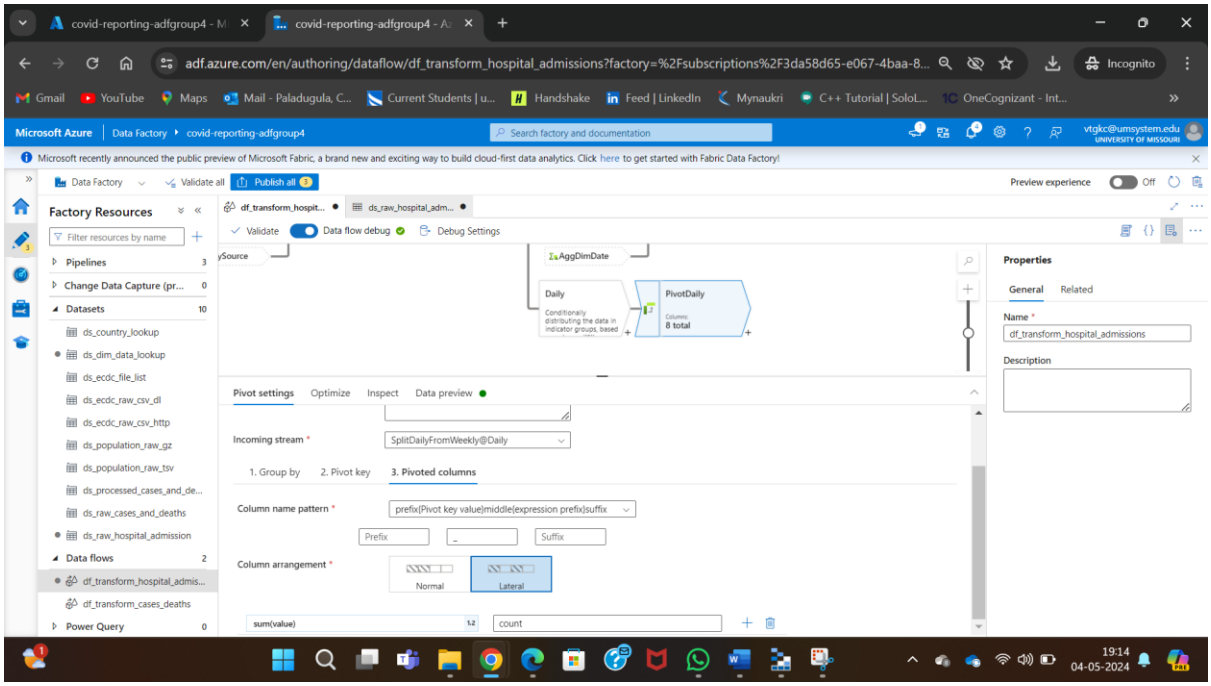
To get the hospital admission count from indicator and value.



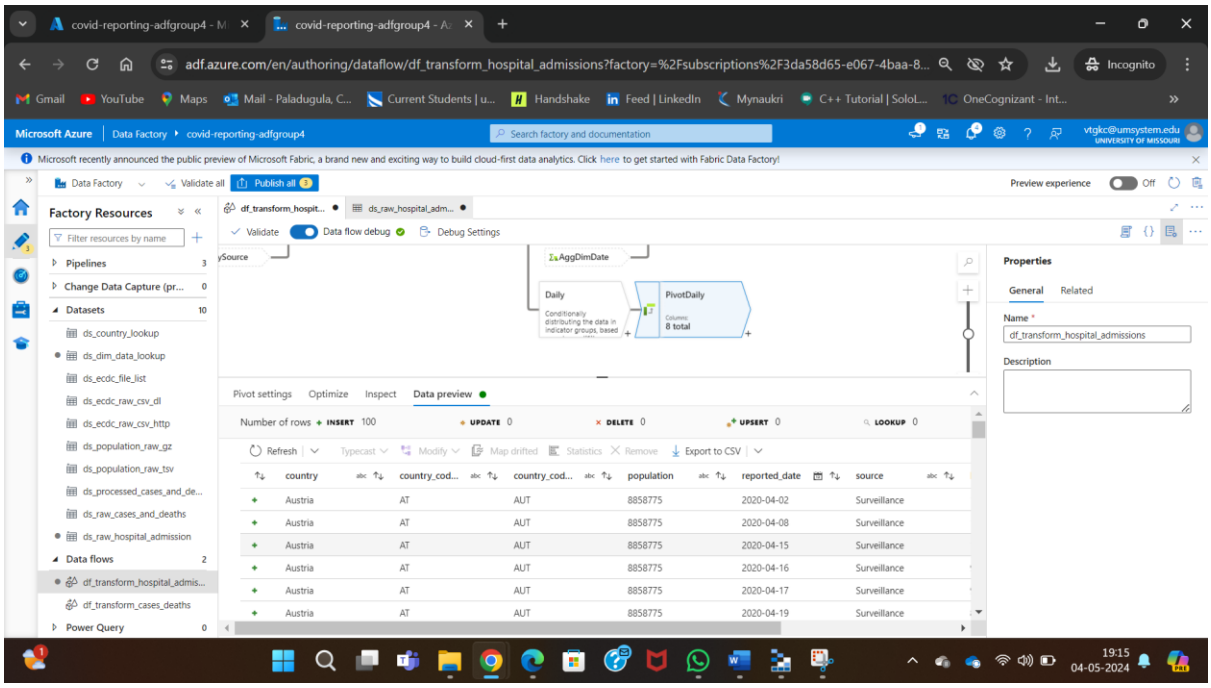
Here is the data preview,



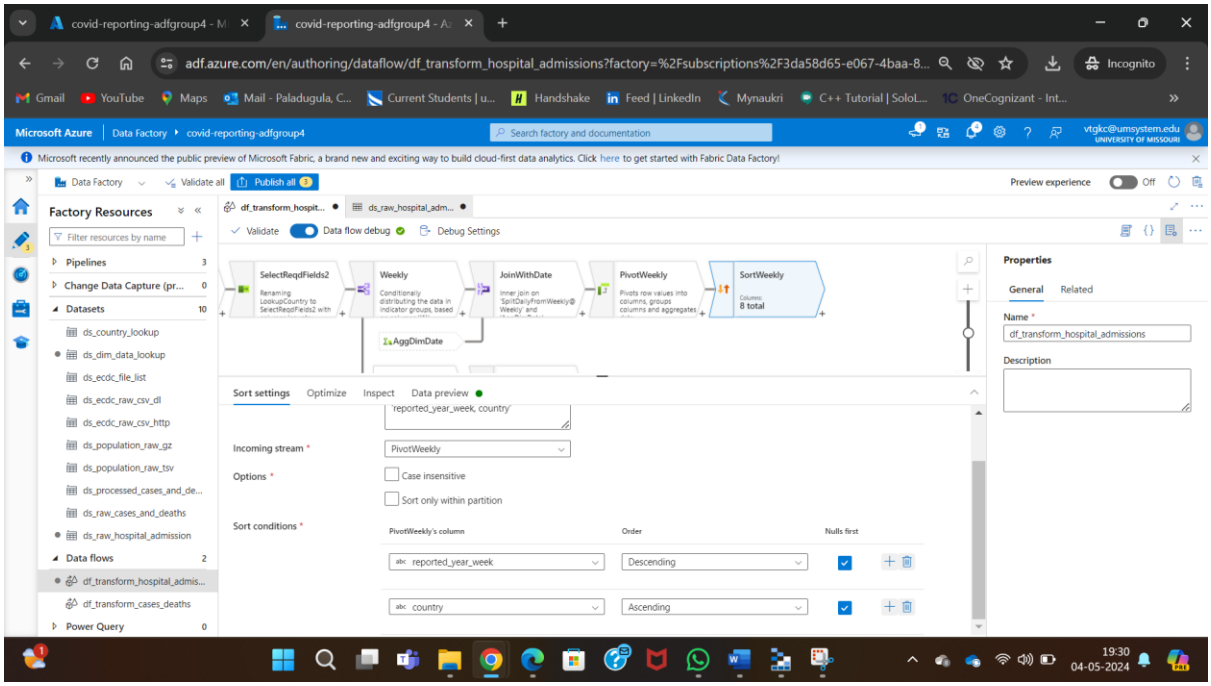
Now we create another pivot for daily,



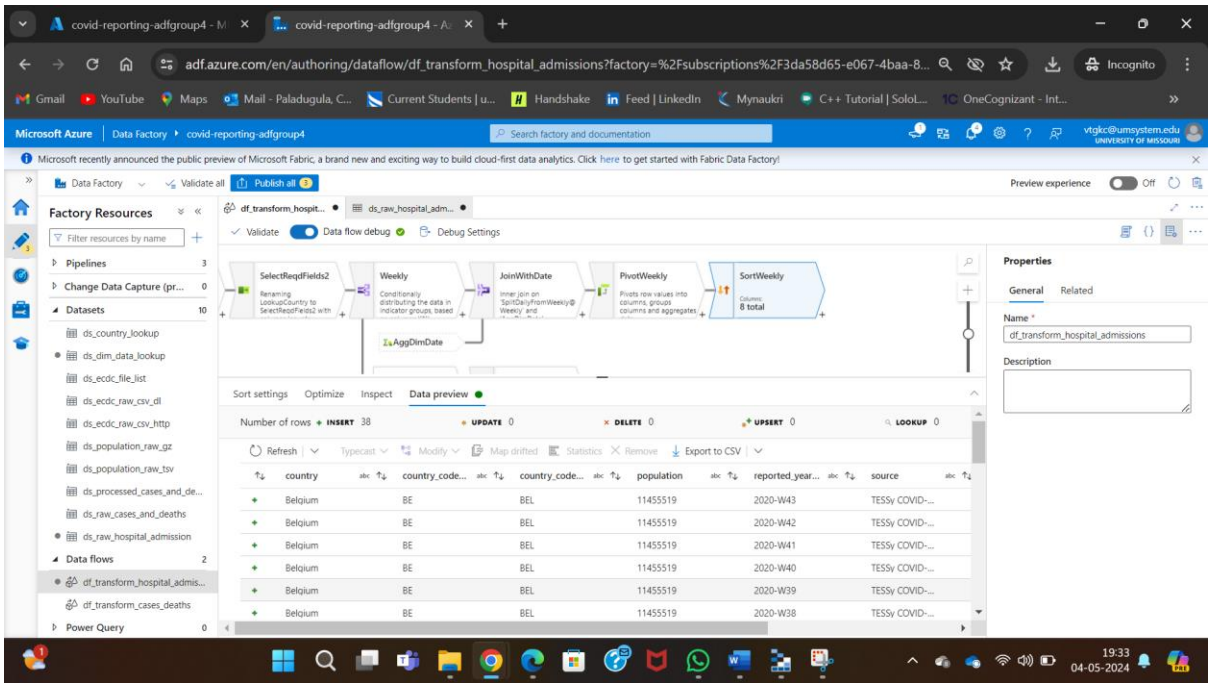
Here is the data preview,



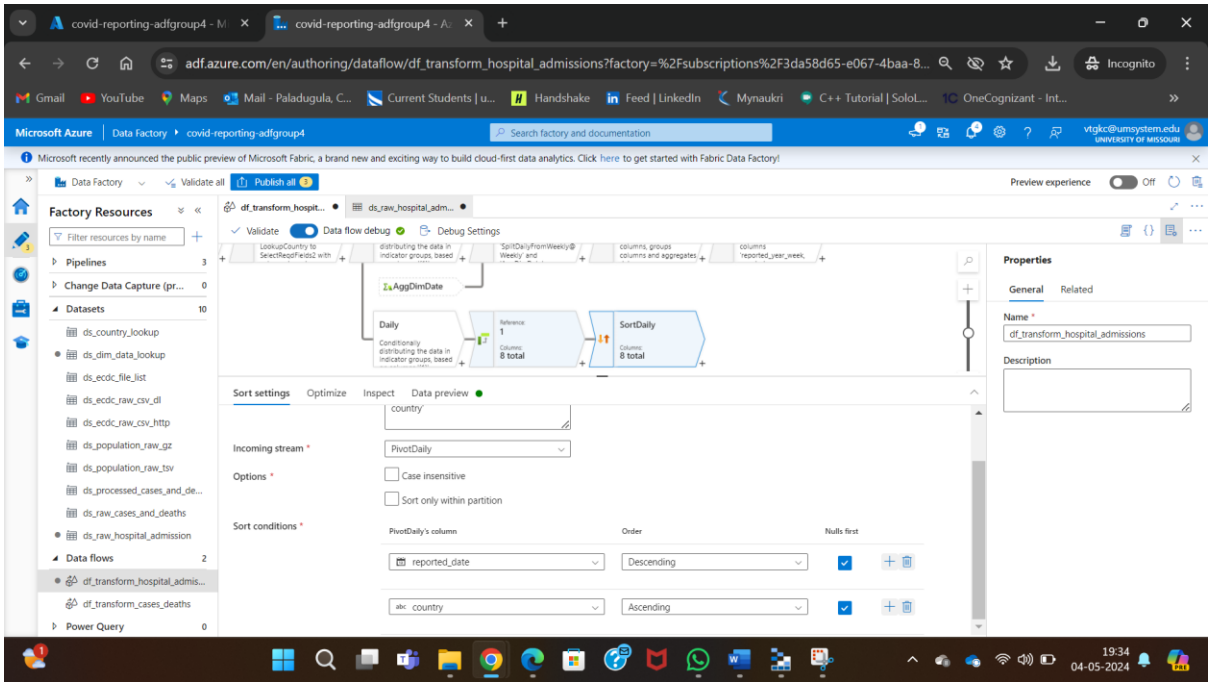
Now we create a sort transformation, To sort the records with country Ascending and reported year week descending, to recent date on the top.



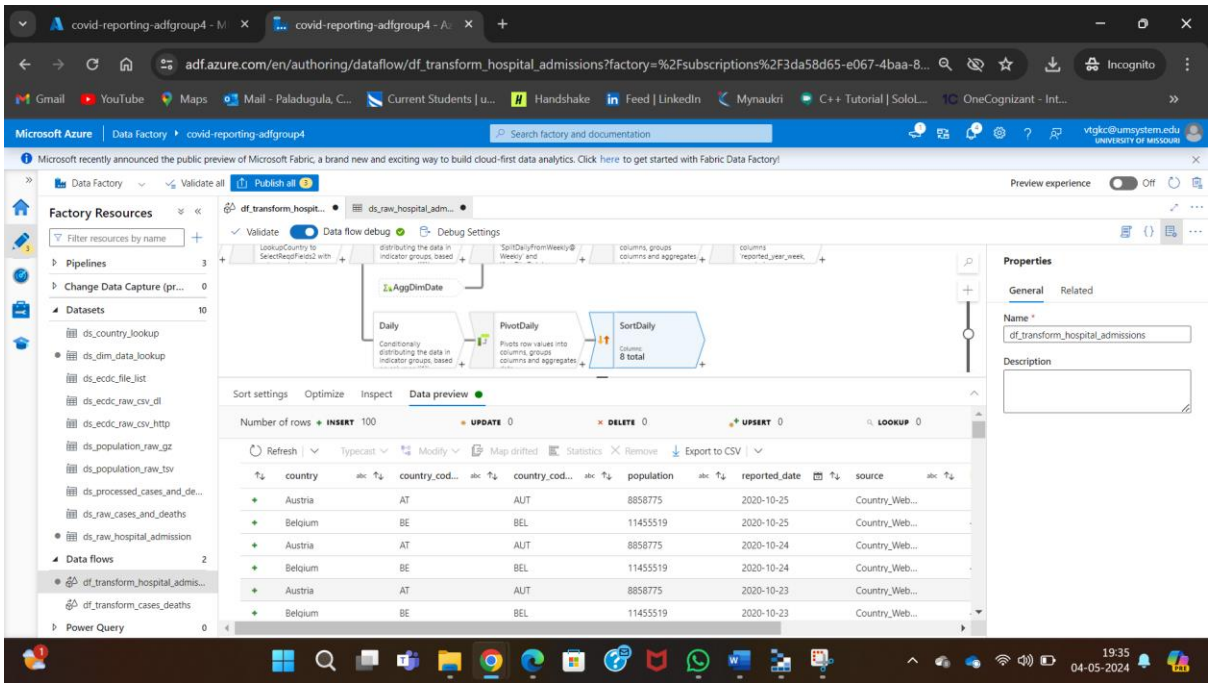
Here is the data preview,



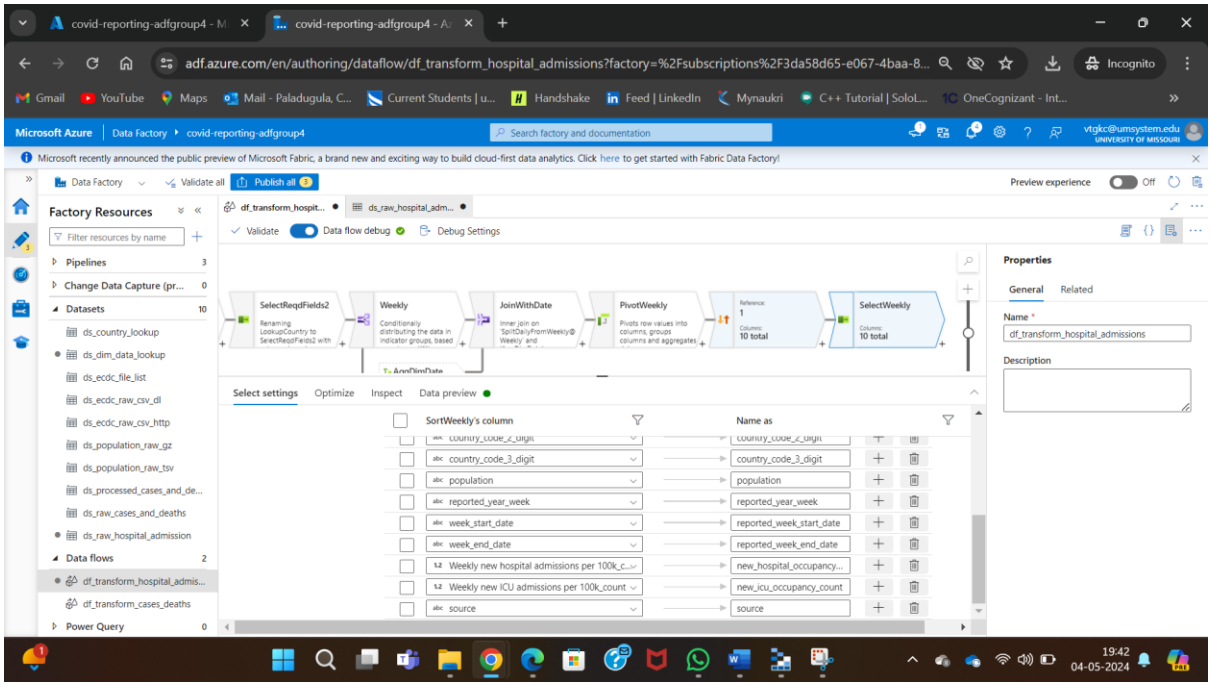
Now we perform the same for Daily,



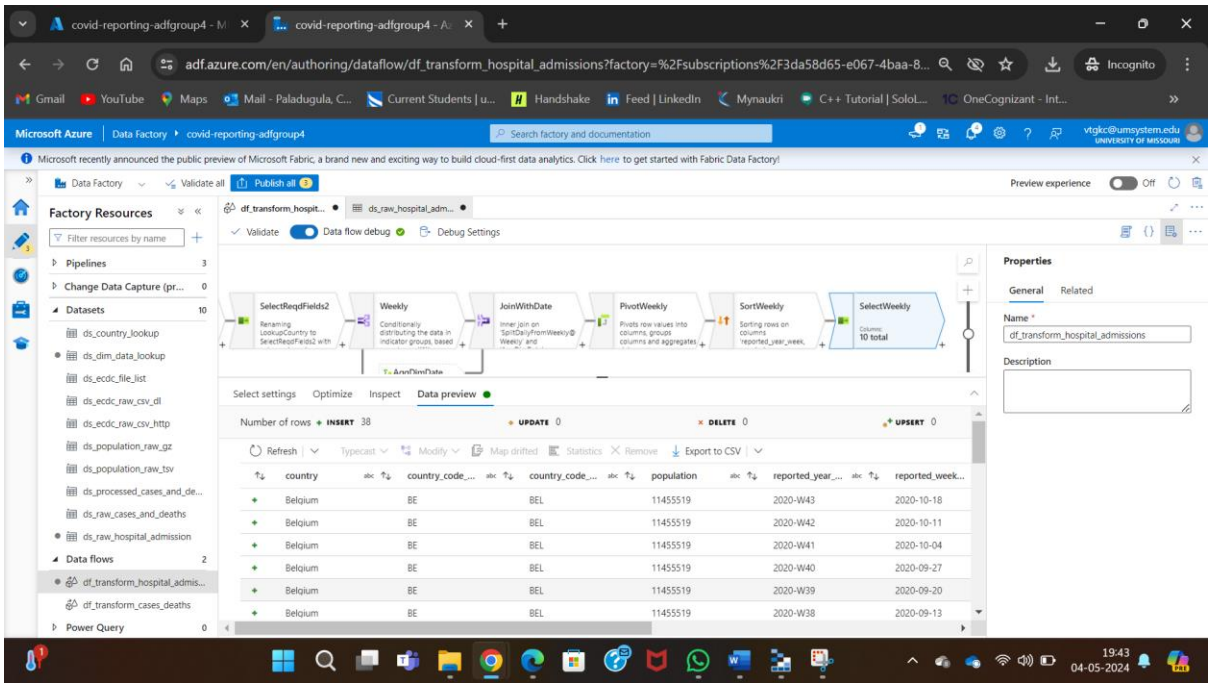
Here is the data preview,



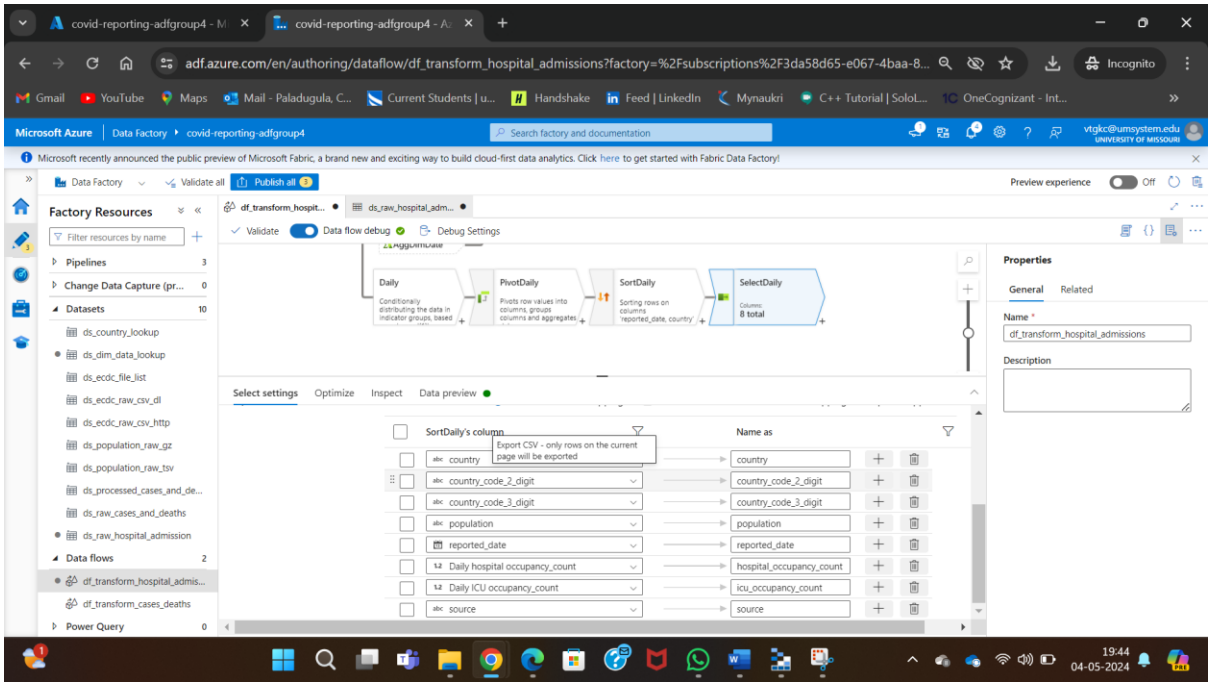
Now we perform Select and sink transformation,



Here is the data preview,

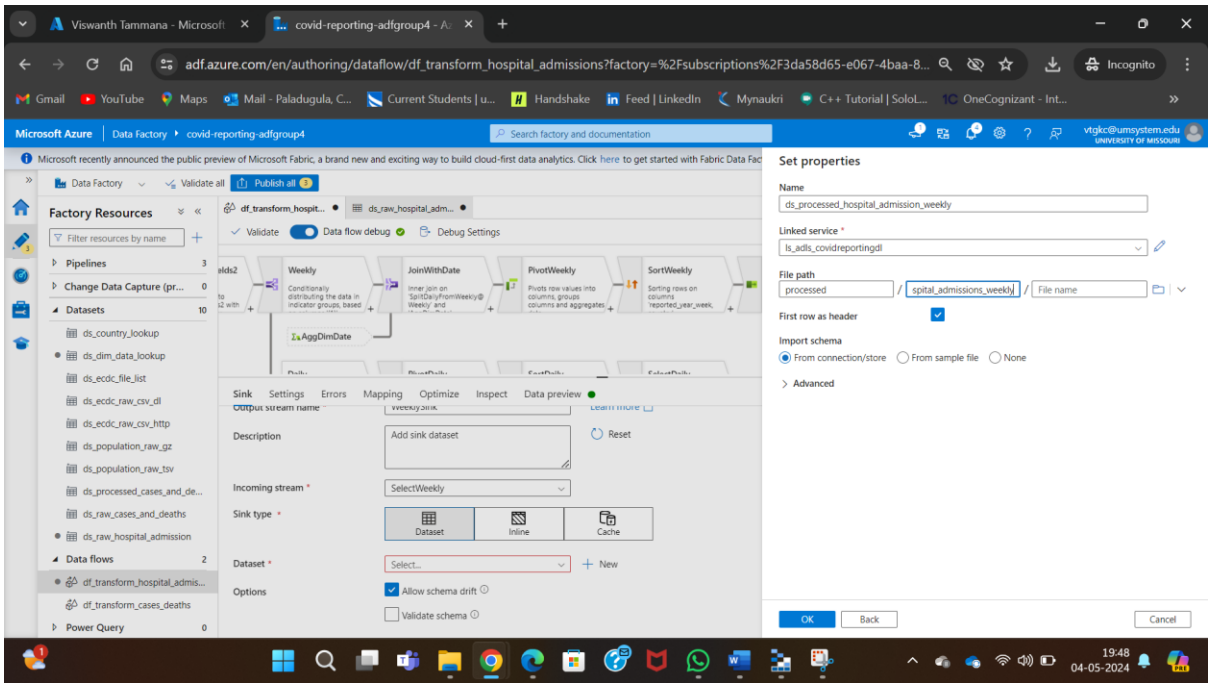


Now create the same select for daily records,

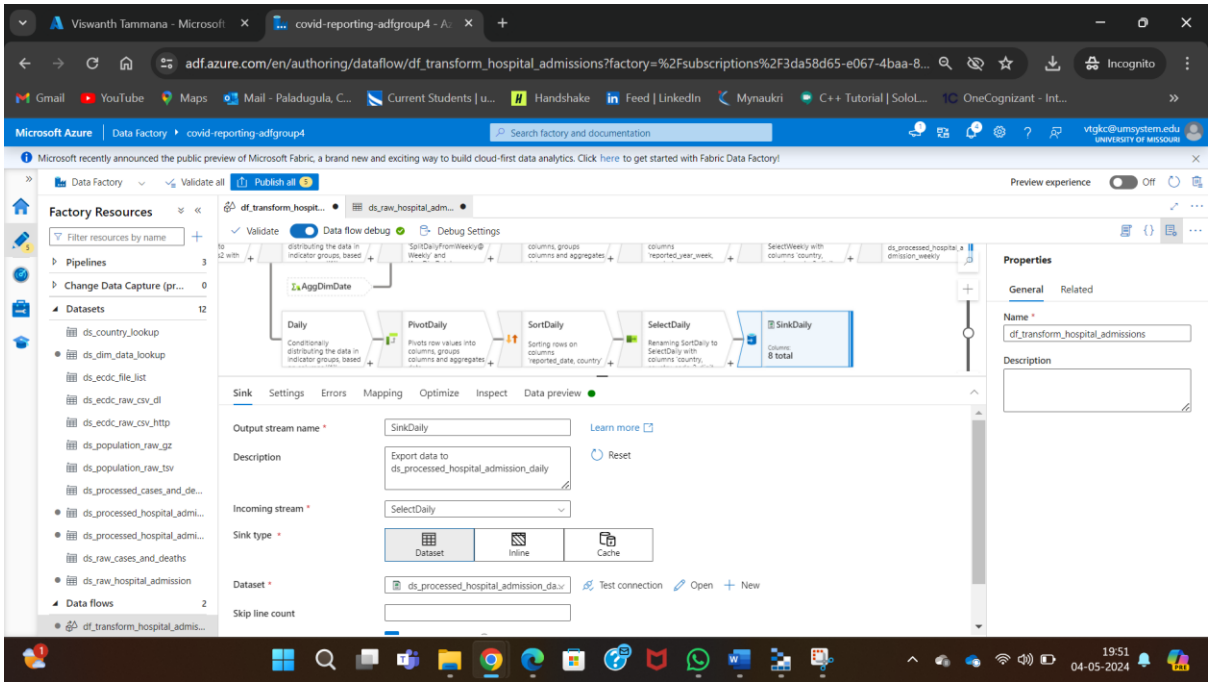


Now we create a new sink, along with a new dataset, dataset-

'ds_processed_hospital_admission_weekly'

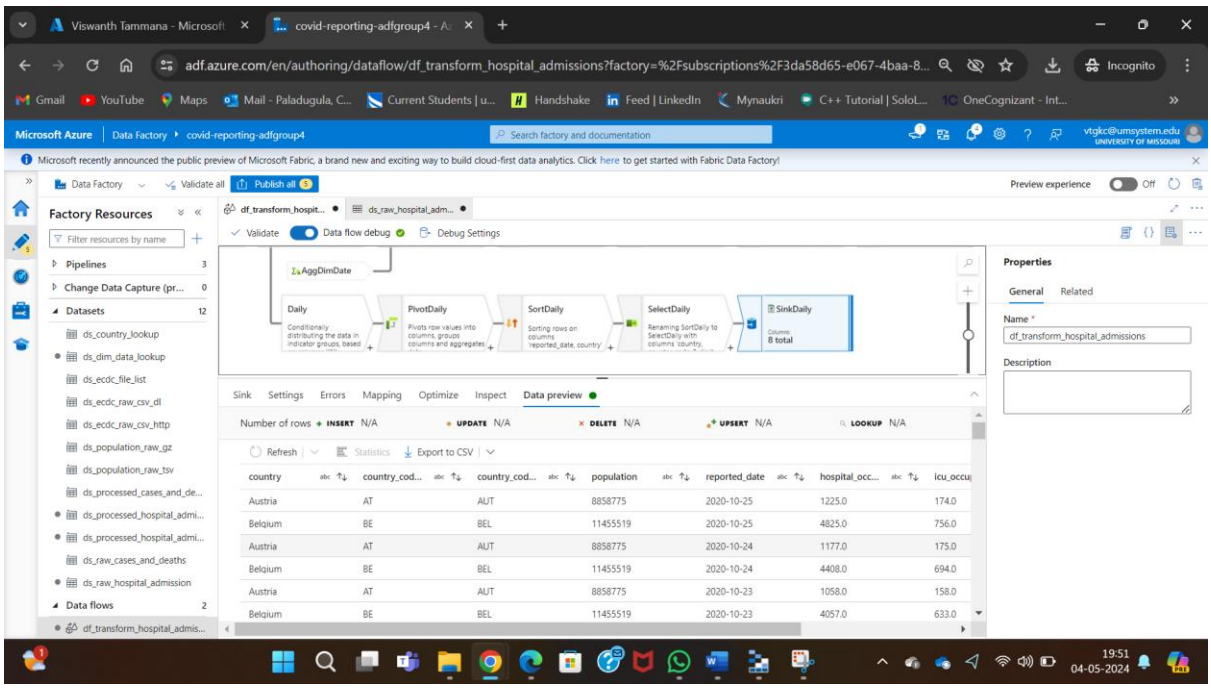


Create the same for daily,

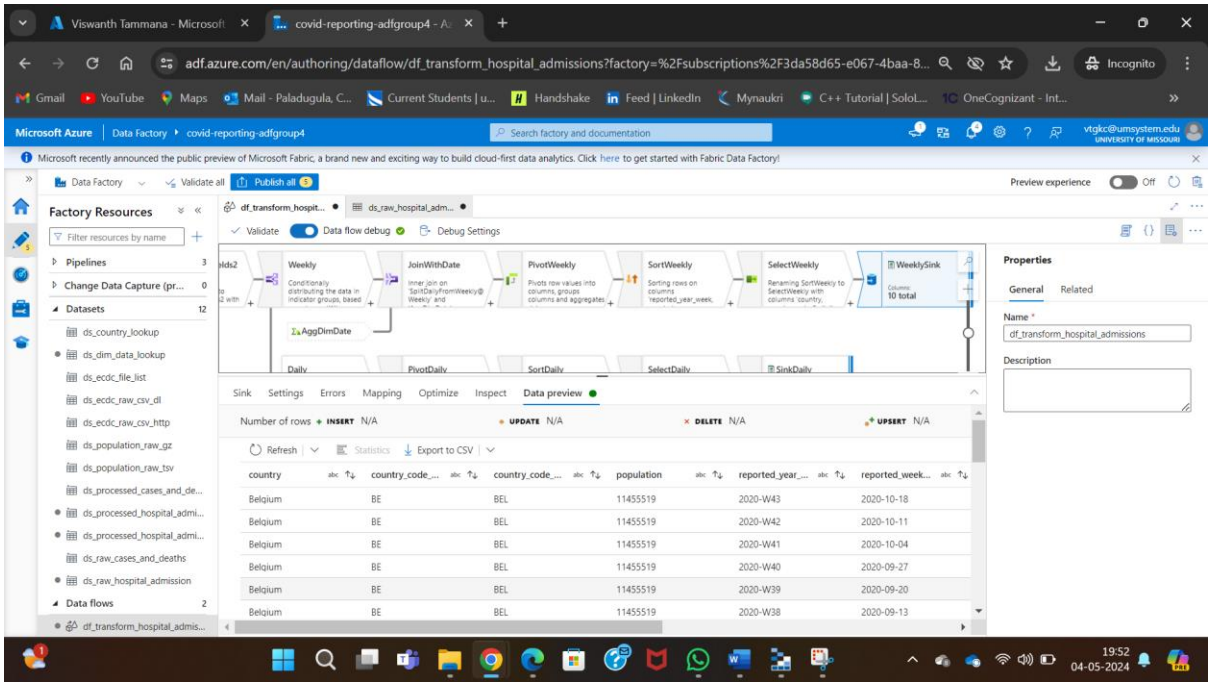


Here are the data preview for both weekly and daily,

Daily-



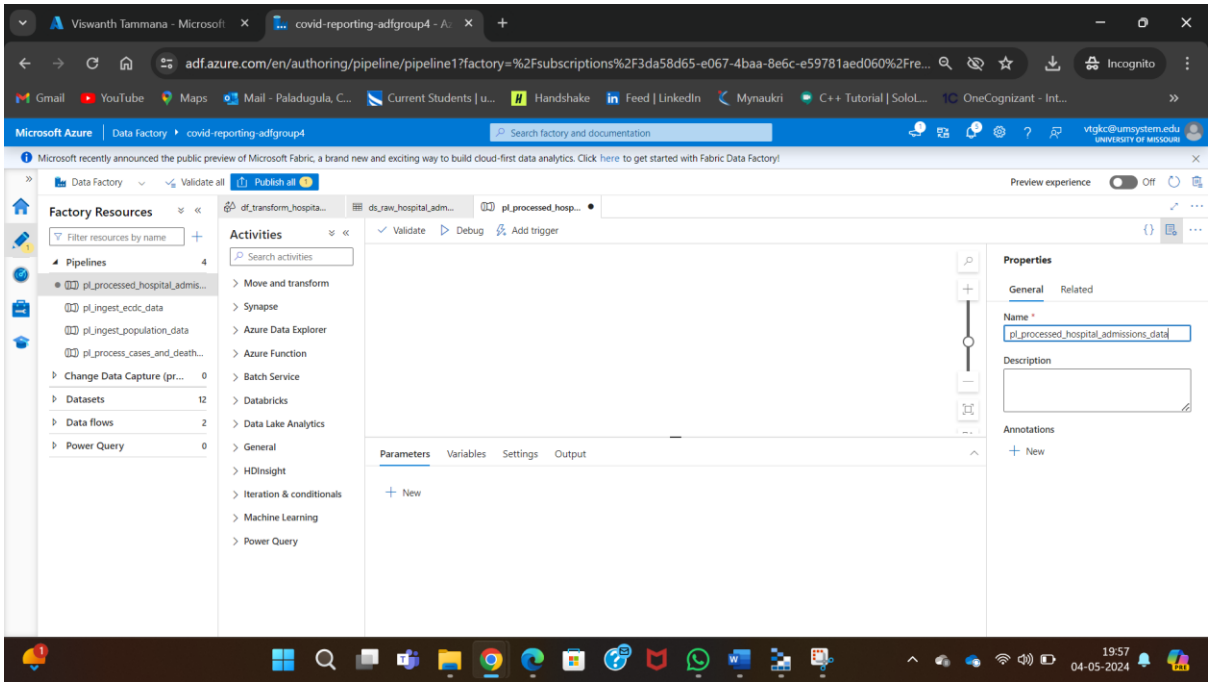
Weekly-



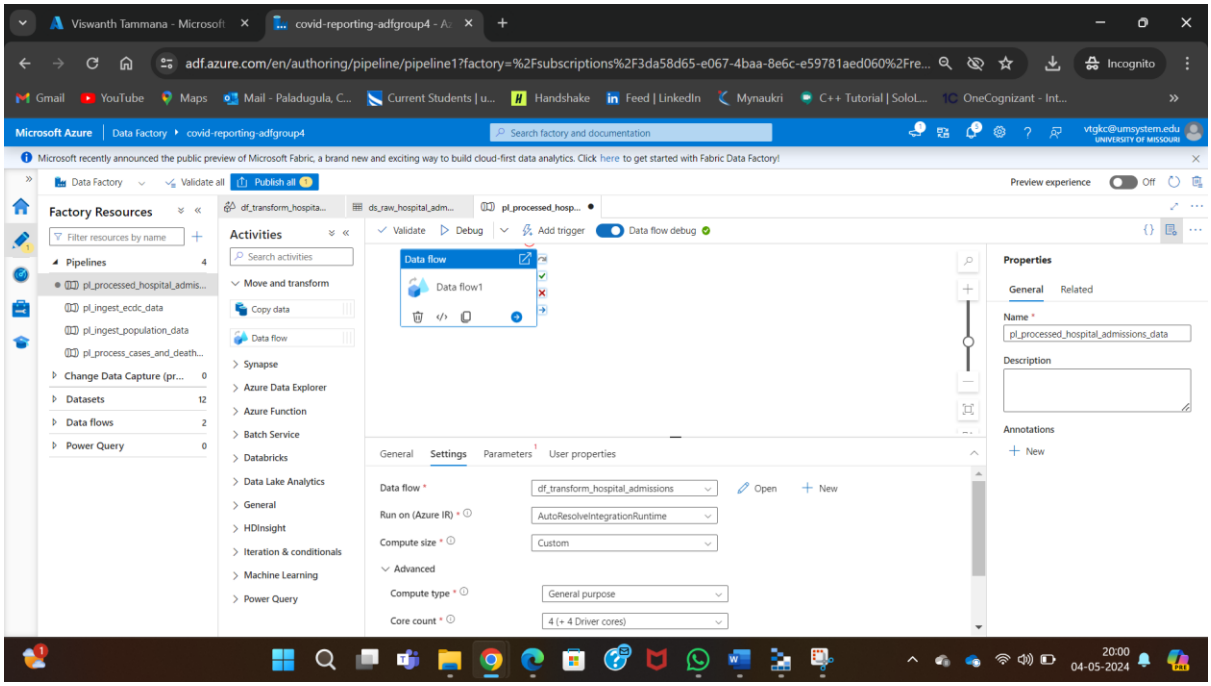
Now publish all the changes,

Now we have successfully created a dataflow we execute it from a pipeline,

New pipeline - pl_processed_hospital_admissions_data



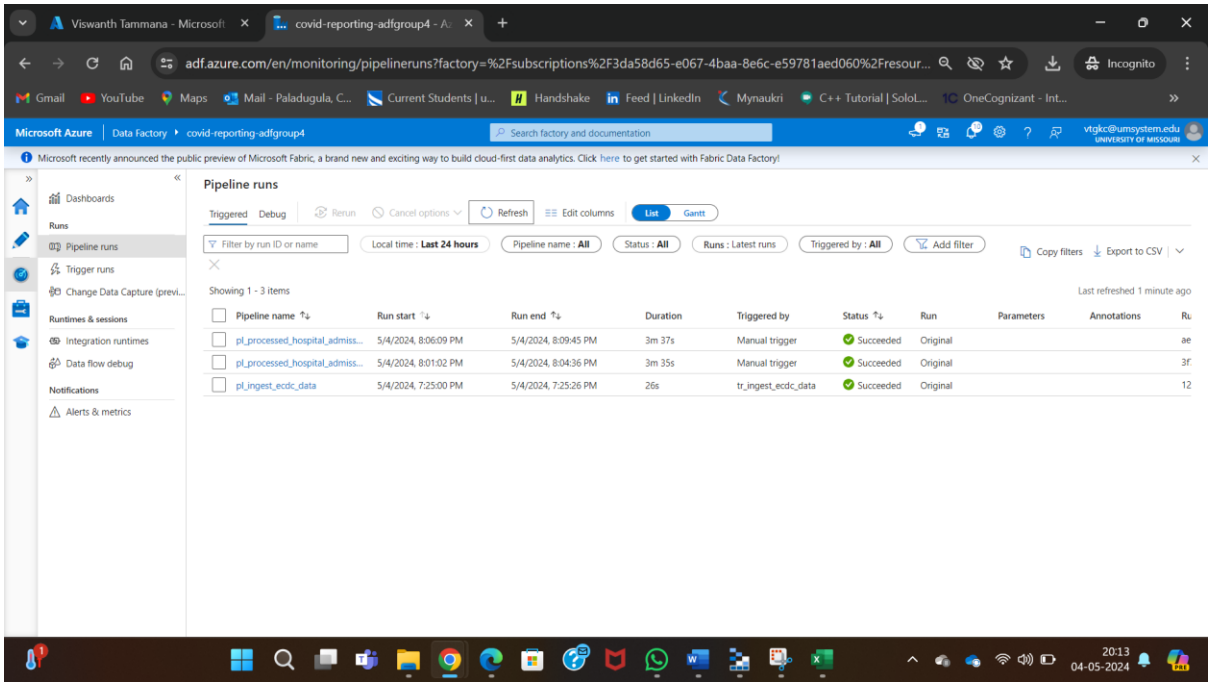
Use the existing dataflow,

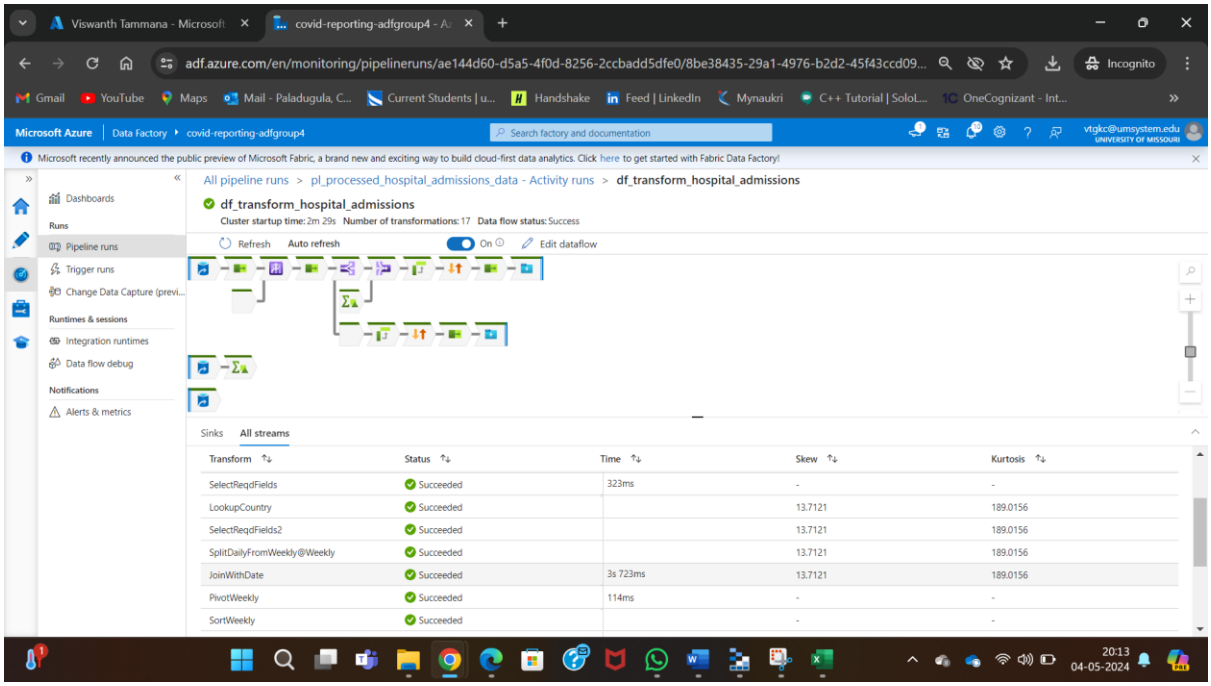


Now publish the pipeline,

Then we do the manual trigger by add trigger→ trigger now.

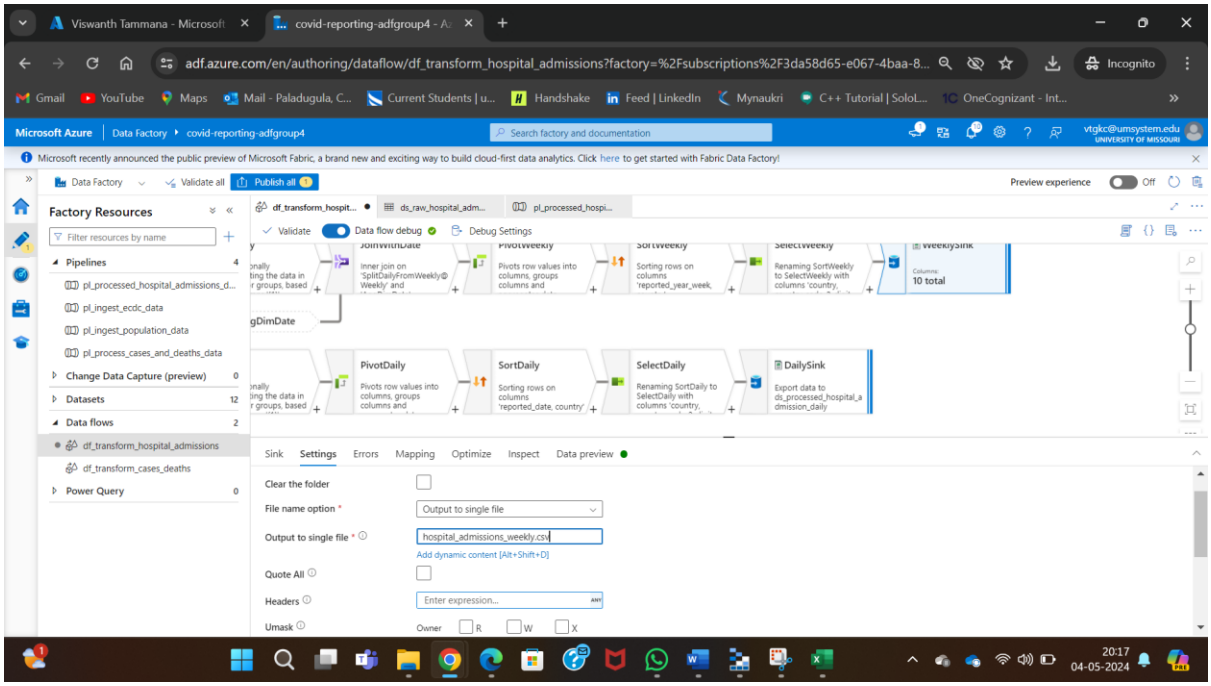
Now navigate to monitor to see changes,



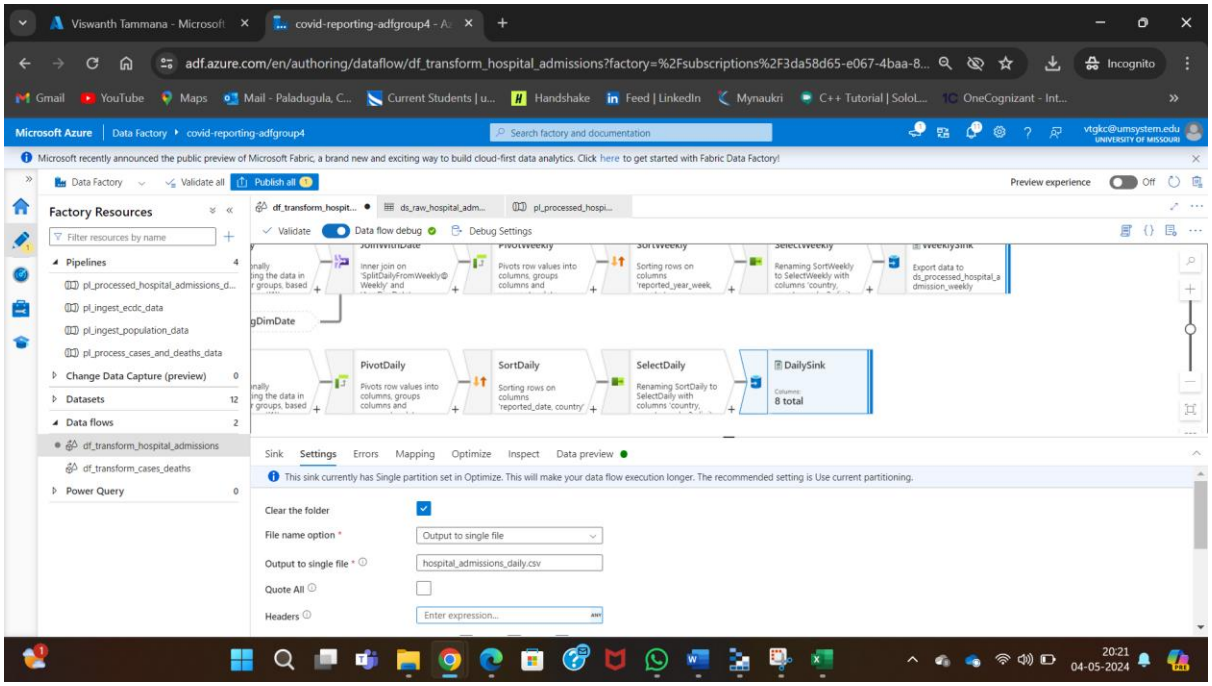


Now we use single partition in the sink to get the output file as a single file ,

Weekly sink file - hospital_admissions_weekly.csv



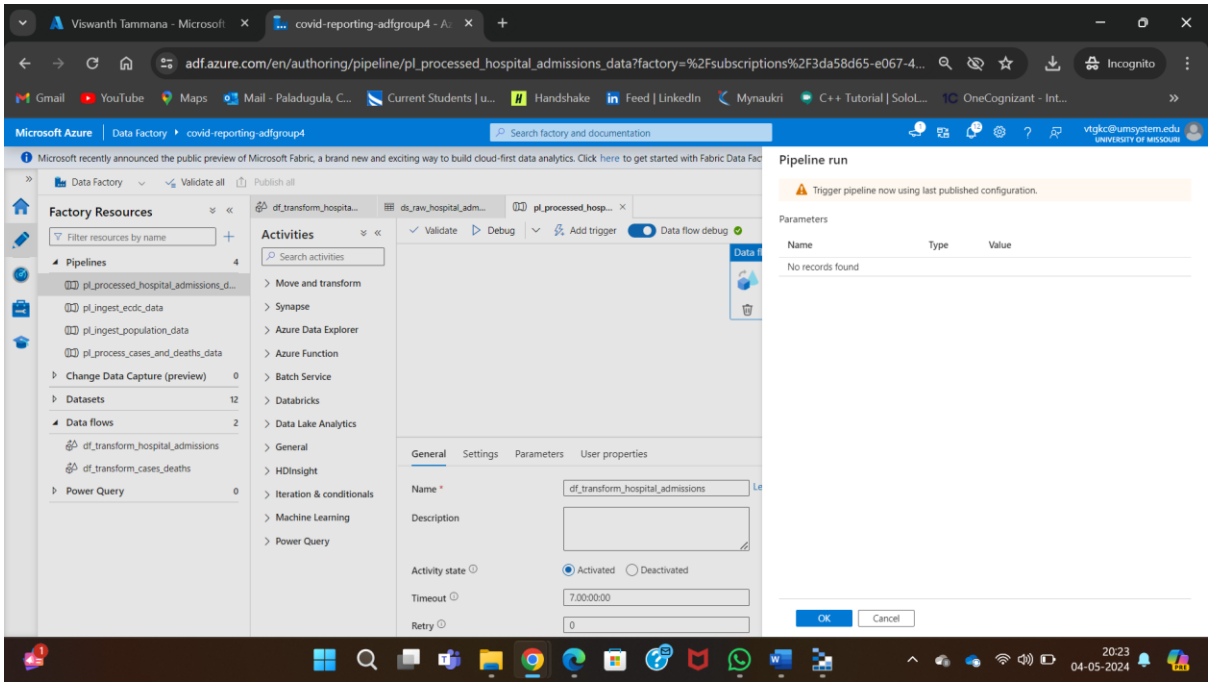
Now we perform the same for daily sink as well,



Select single partition in both sortweekly and sortdaily,

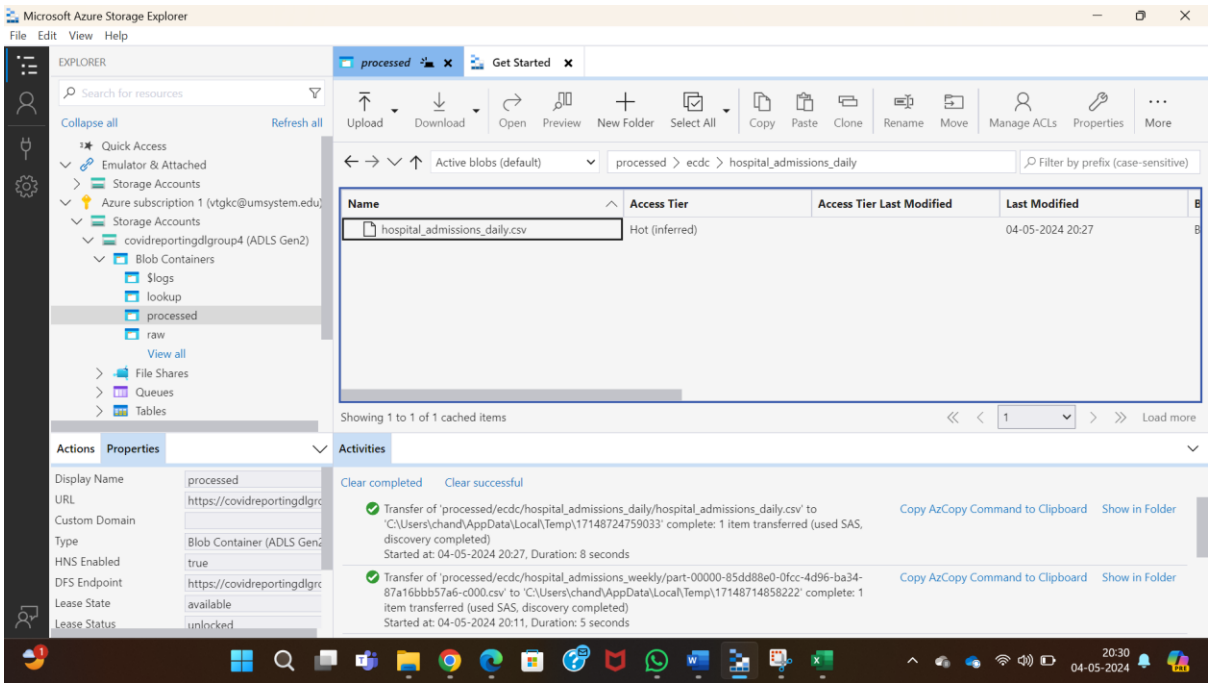
Now we publish all the changes,

Then we manually trigger the pipeline,

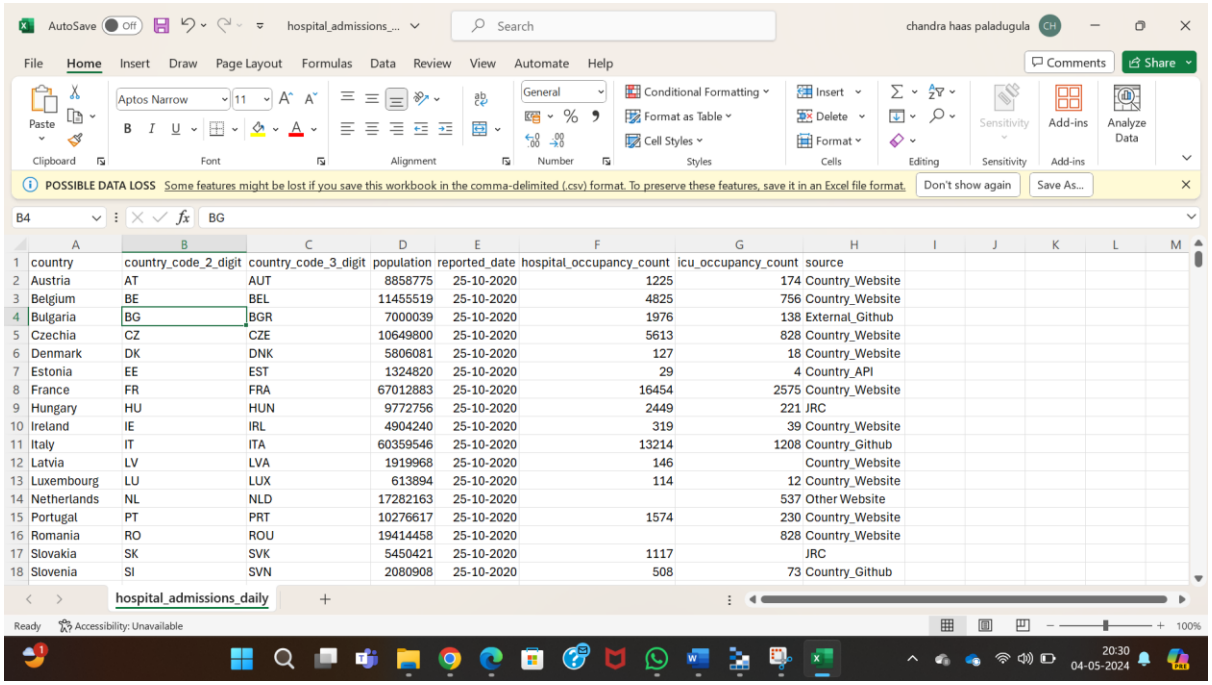


Now we can see the files in the file explorer,

This is the daily records file,

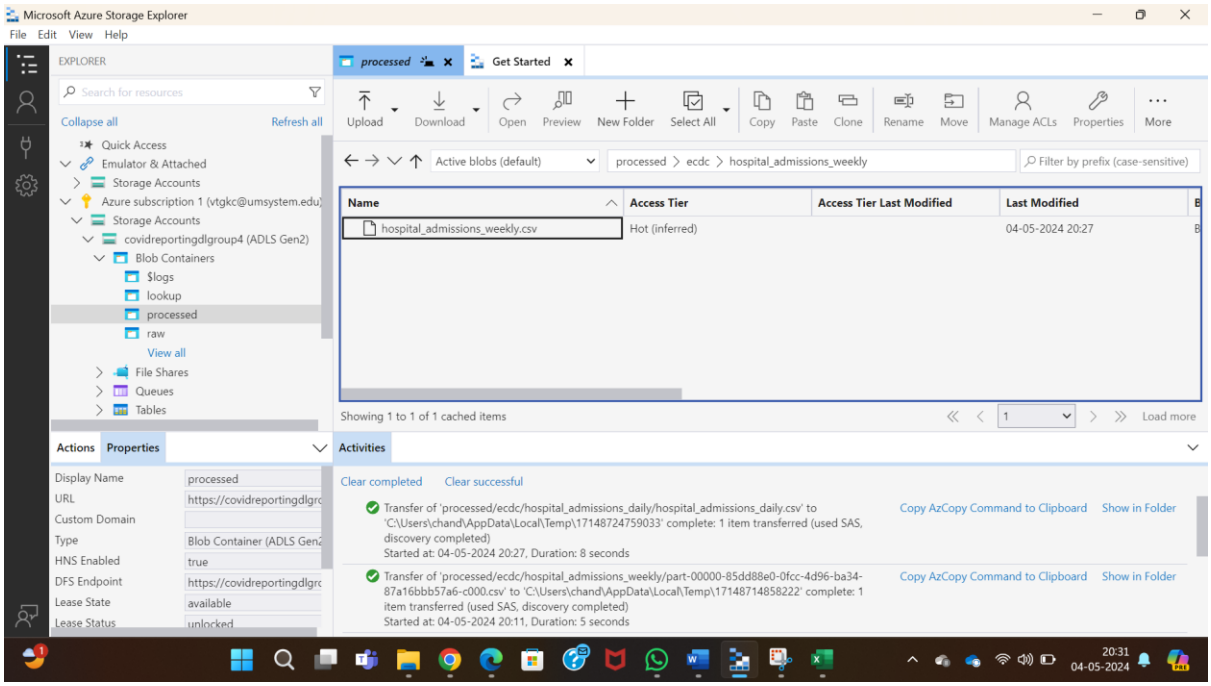


Here is the csv file,

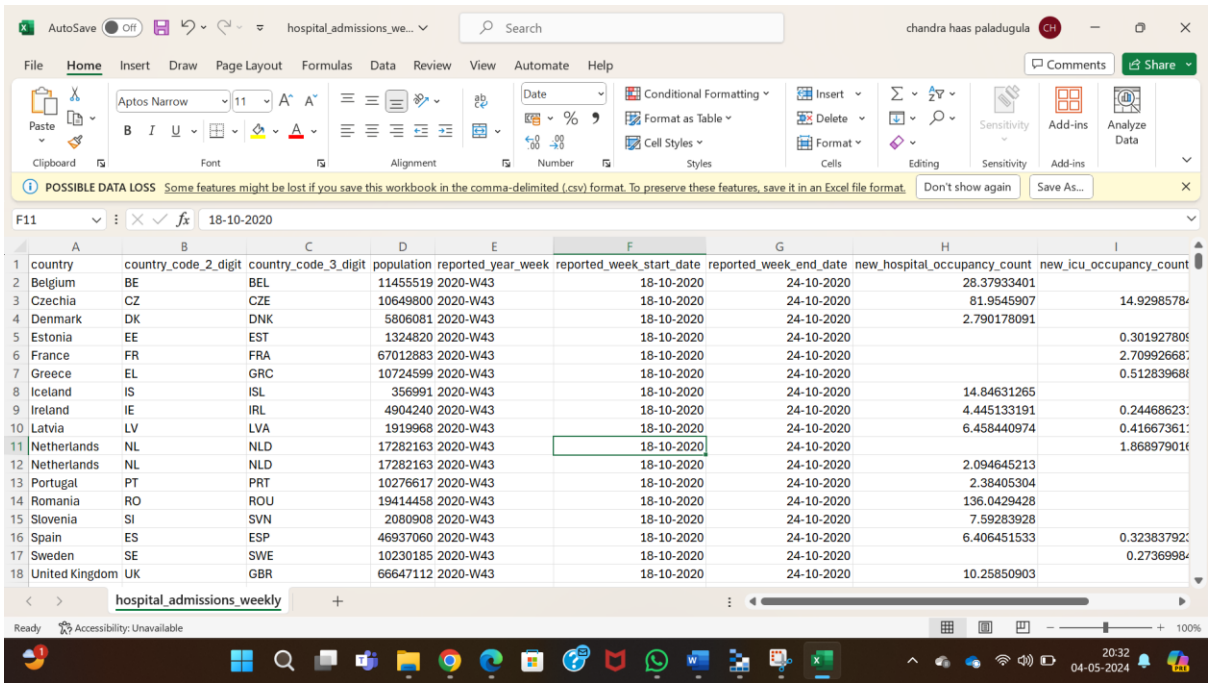


	A	B	C	D	E	F	G	H
1	country	country_code_2_digit	country_code_3_digit	population	reported_date	hospital_occupancy_count	icu_occupancy_count	source
2	Austria	AT	AUT	8858775	25-10-2020	1225	174	Country_Website
3	Belgium	BE	BEL	11455519	25-10-2020	4825	756	Country_Website
4	Bulgaria	BG	BGR	7000039	25-10-2020	1976	138	External_Github
5	Czechia	CZ	CZE	10649800	25-10-2020	5613	828	Country_Website
6	Denmark	DK	DNK	5806081	25-10-2020	127	18	Country_Website
7	Estonia	EE	EST	1324820	25-10-2020	29	4	Country_API
8	France	FR	FRA	67012883	25-10-2020	16454	2575	Country_Website
9	Hungary	HU	HUN	9772756	25-10-2020	2449	221	JRC
10	Ireland	IE	IRL	4904240	25-10-2020	319	39	Country_Website
11	Italy	IT	ITA	60359546	25-10-2020	13214	1208	Country_Github
12	Latvia	LV	LVA	1919968	25-10-2020	146		Country_Website
13	Luxembourg	LU	LUX	613894	25-10-2020	114	12	Country_Website
14	Netherlands	NL	NLD	17282163	25-10-2020		537	Other Website
15	Portugal	PT	PRT	10276617	25-10-2020	1574	230	Country_Website
16	Romania	RO	ROU	19414458	25-10-2020		828	Country_Website
17	Slovakia	SK	SVK	5450421	25-10-2020	1117		JRC
18	Slovenia	SI	SVN	2080908	25-10-2020	508	73	Country_Github

This is the weekly records file,



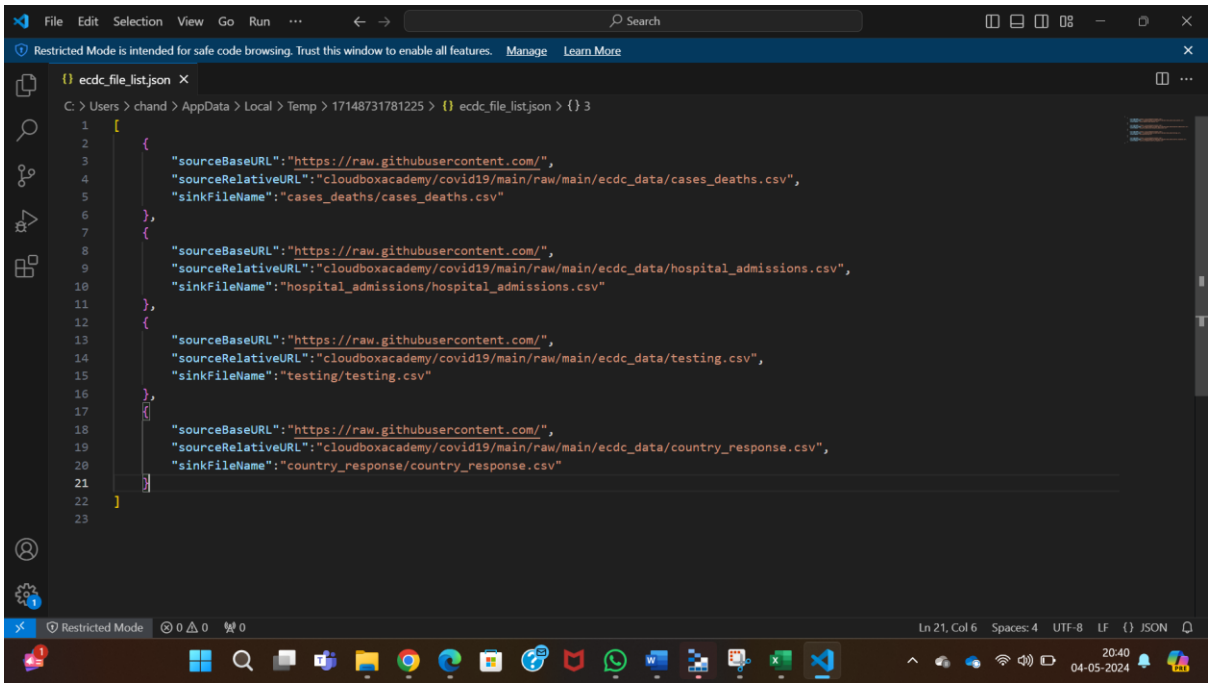
Here is the csv file,



Prepare data for HD insights and Databricks

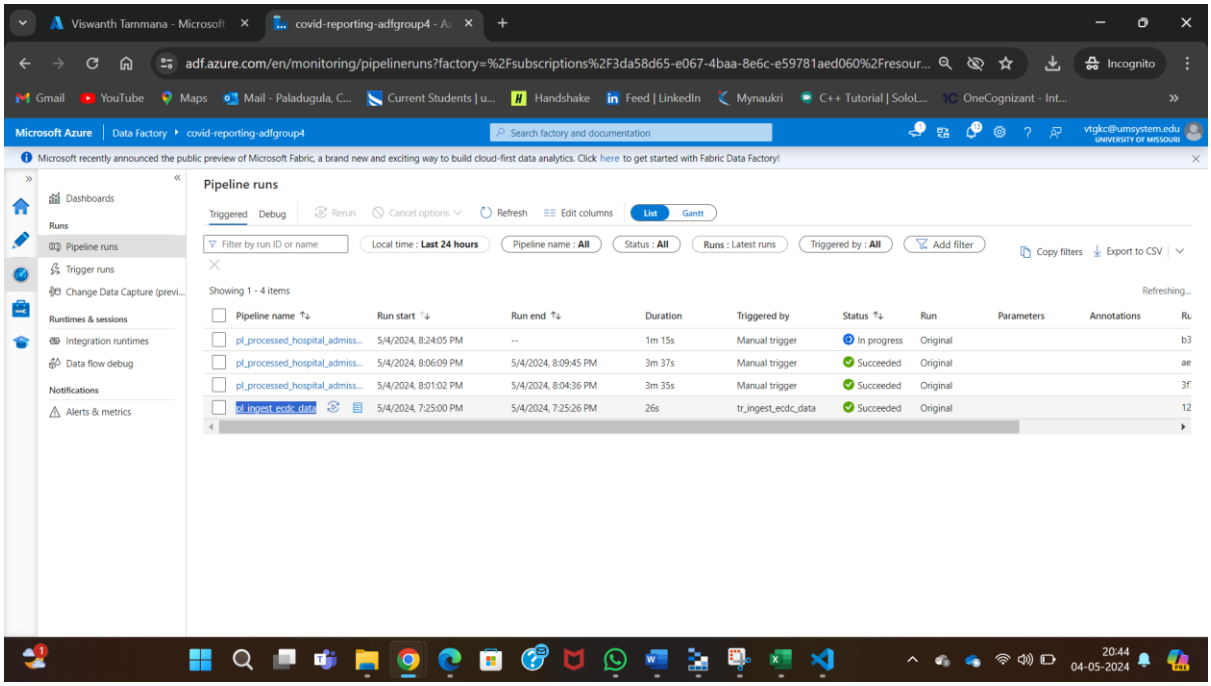
As we futher do transformations in HDInsights and Spark we need the data to be in a folder, As they works on distributed computing. We need to make some changes in the Json files,

File path – covidreportingsagroup4 → blob containers → configs → ecdc_file_list.json

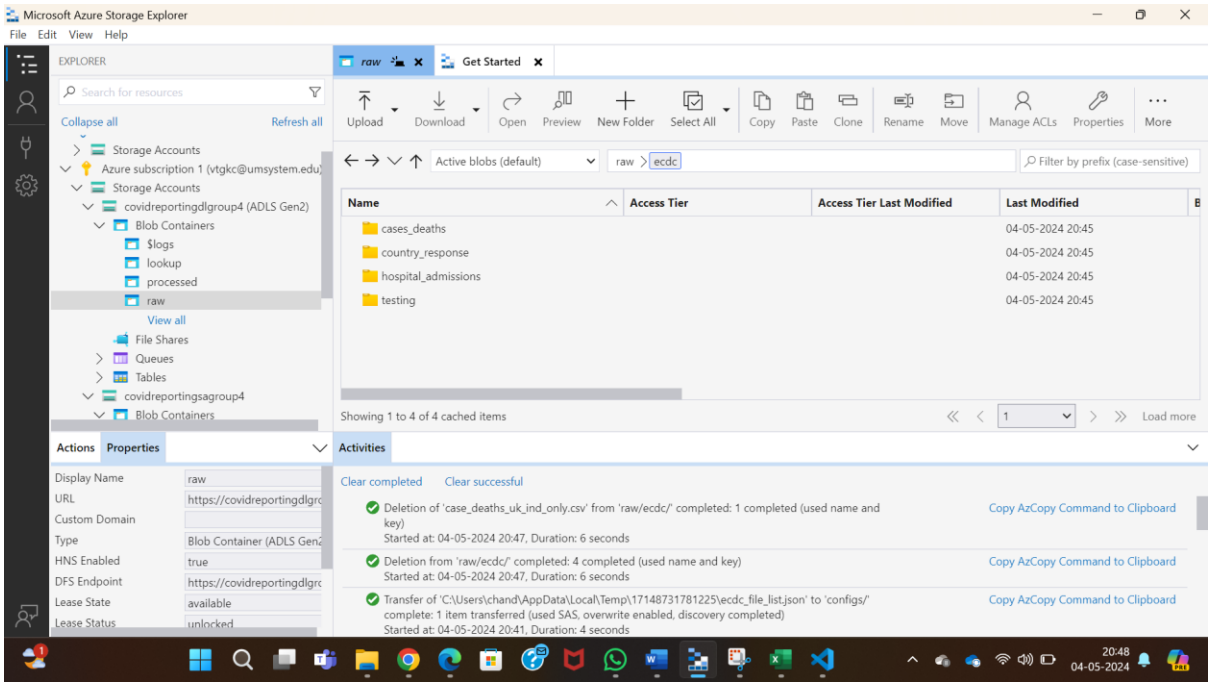


We change the file path here.

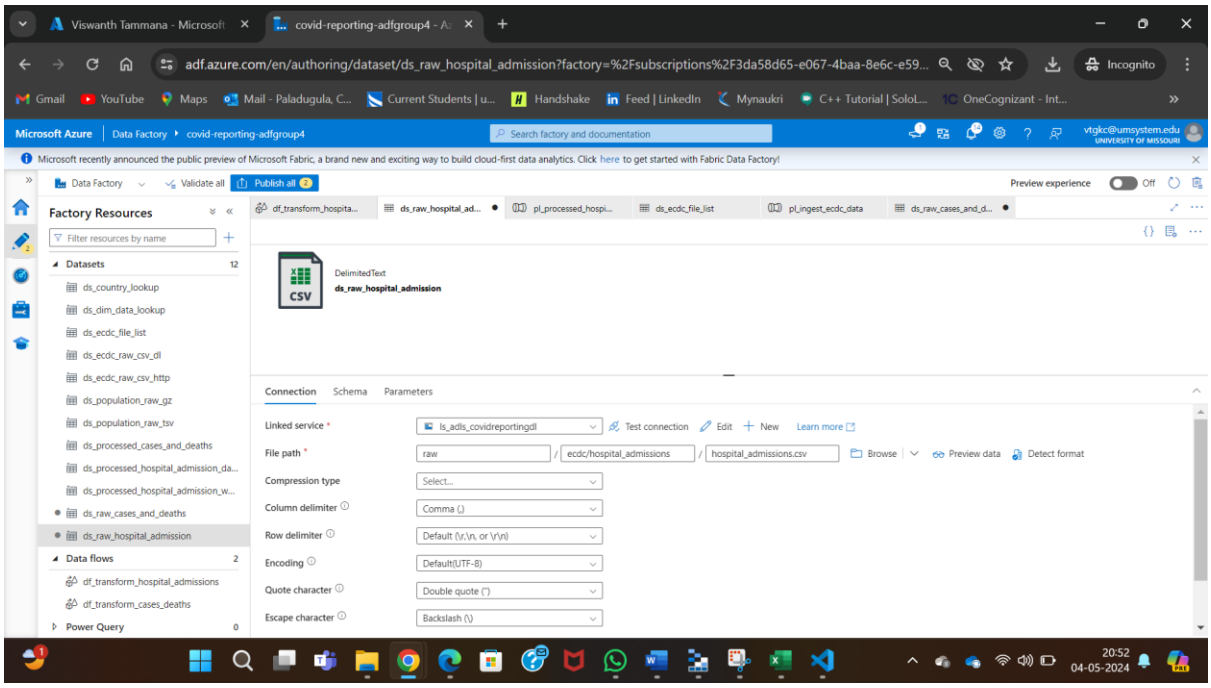
Then we rerun the pipeline from the monitor tab, name - pl_ingest_ecdc_data



Now in explorer we see the changes in the ecdc folder,



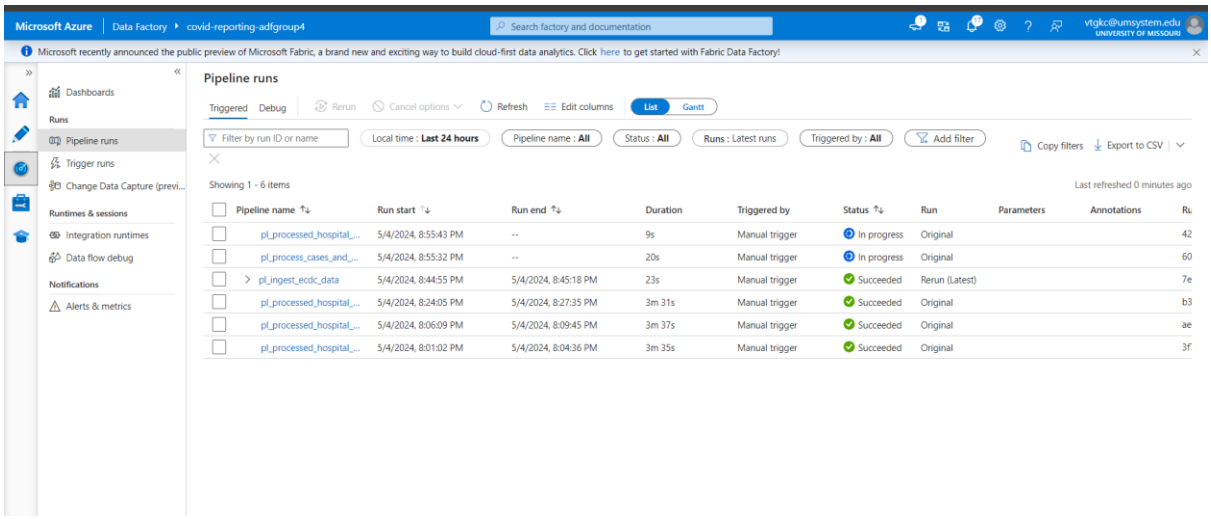
Then we perform the location change in the dataset,



We do the same with all the datasets,

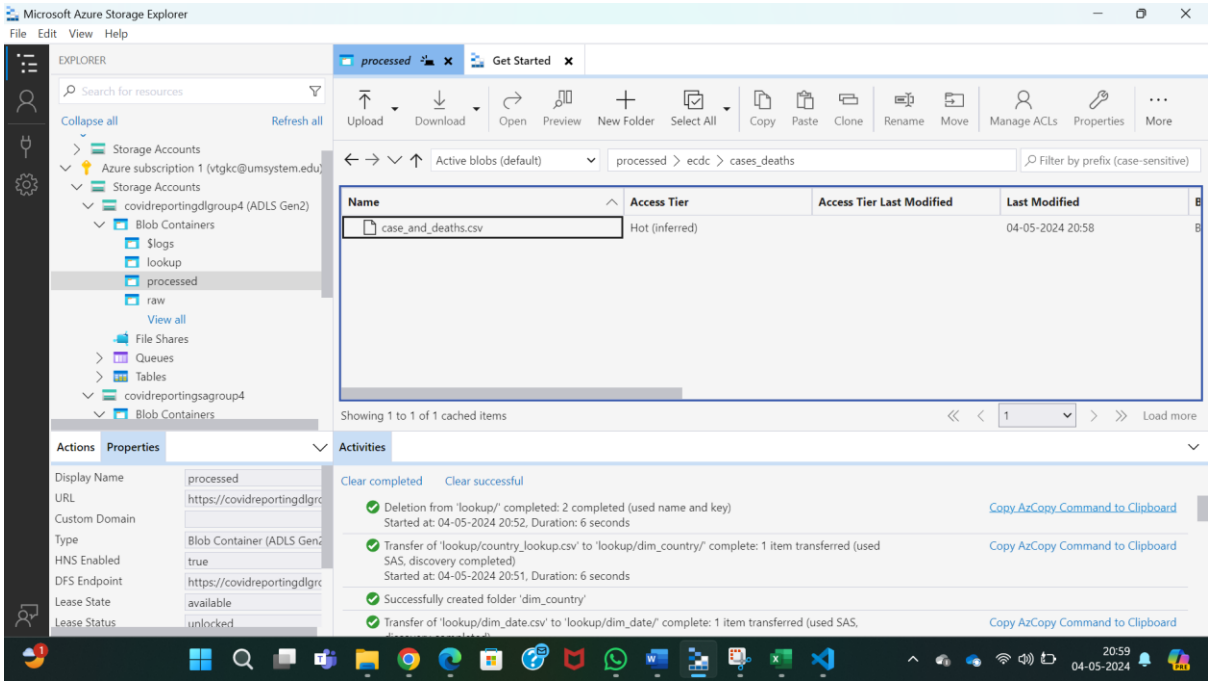
Then we publish all the changes,

Now trigger the two pipelines and monitor

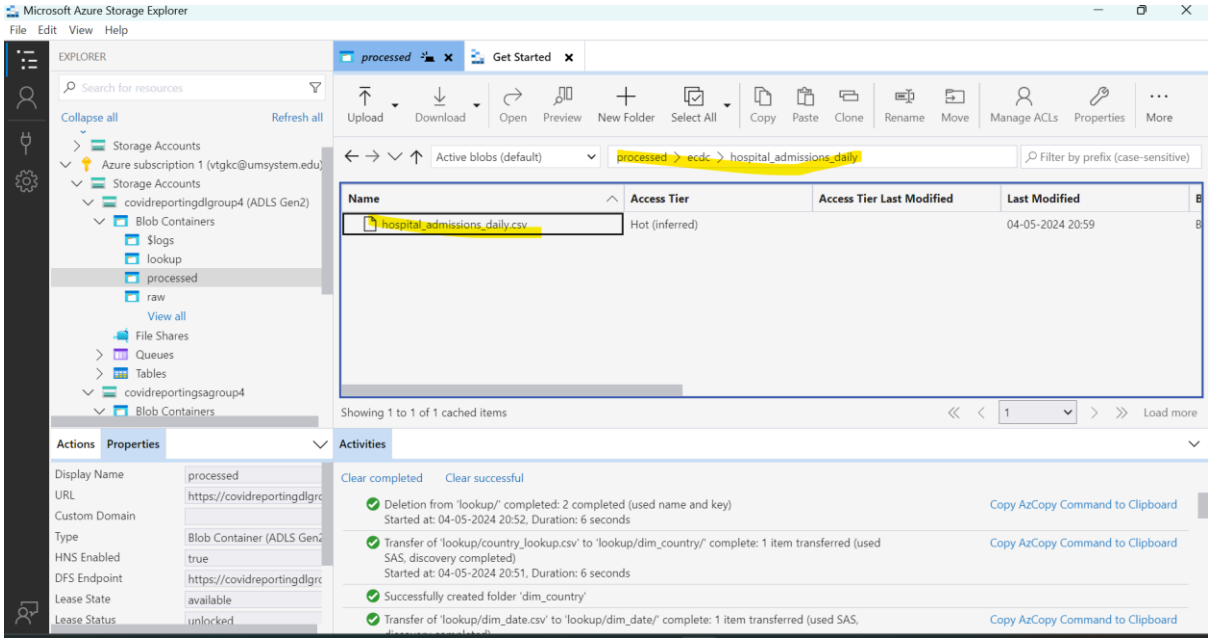


Once done, make sure that the changes are seen in the explorer.

navigate to the path mentioned below and now you can see the updated file over there.



we can see both the daily and weekly data is also updated



Copy the data to Azure SQL:

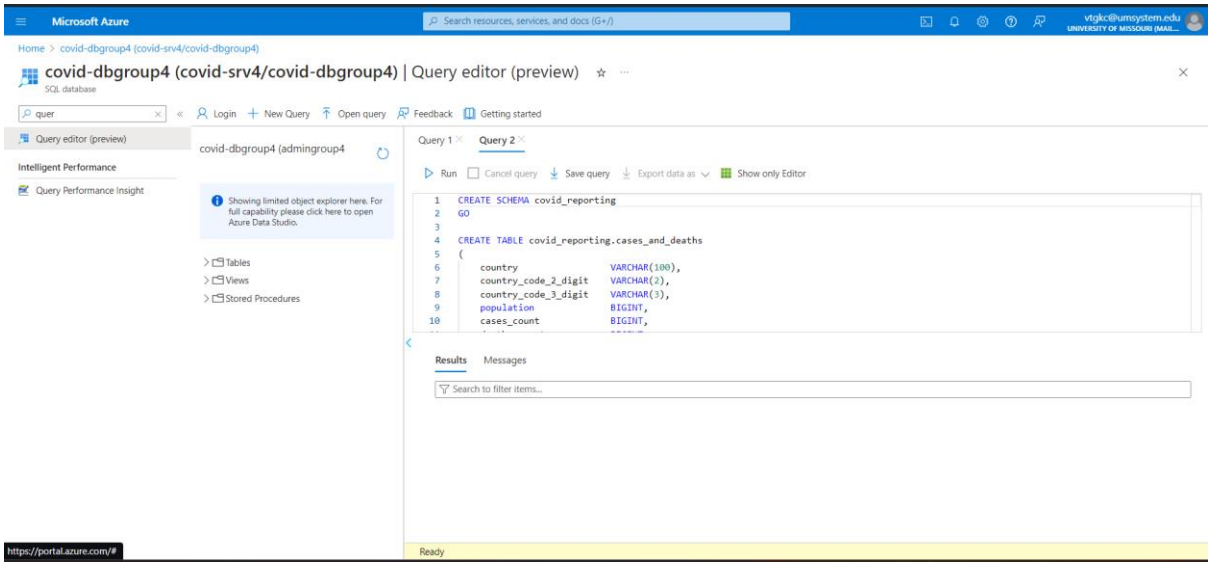
Till now we have transformed the data using dataflow, HDInsight and databricks. The transformed data is written back to the dtaalake. No we need to copy this data back to the SQL data. Below 3 data are to be copied to azure sql.

- Cases and deaths
- Hospital admissions
- Testing

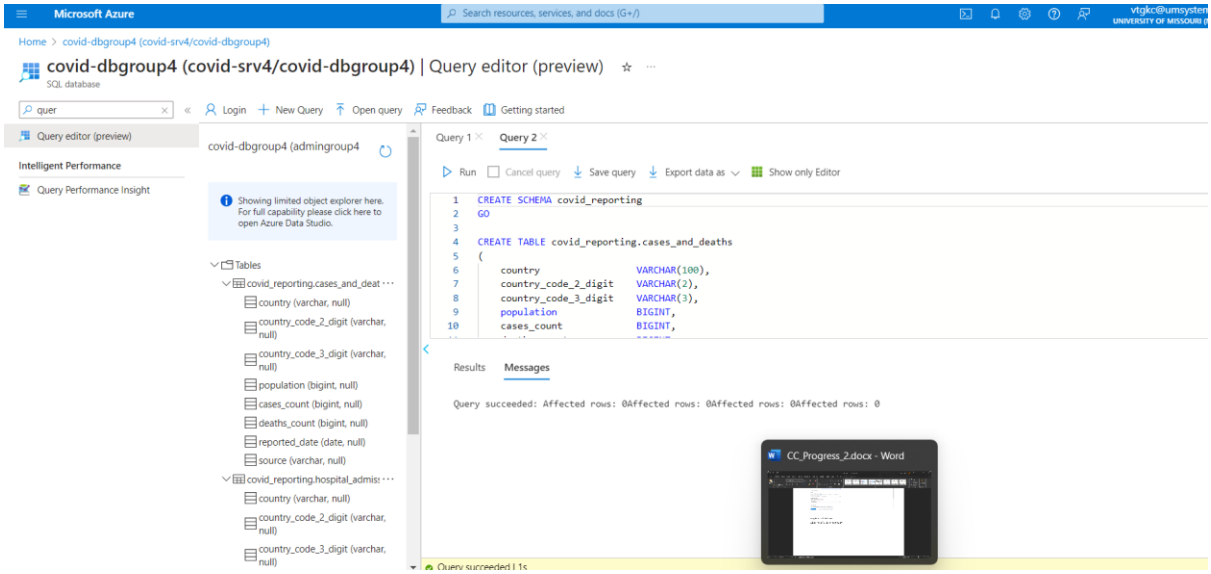
For that we need to create SQL

Navigate to covid database

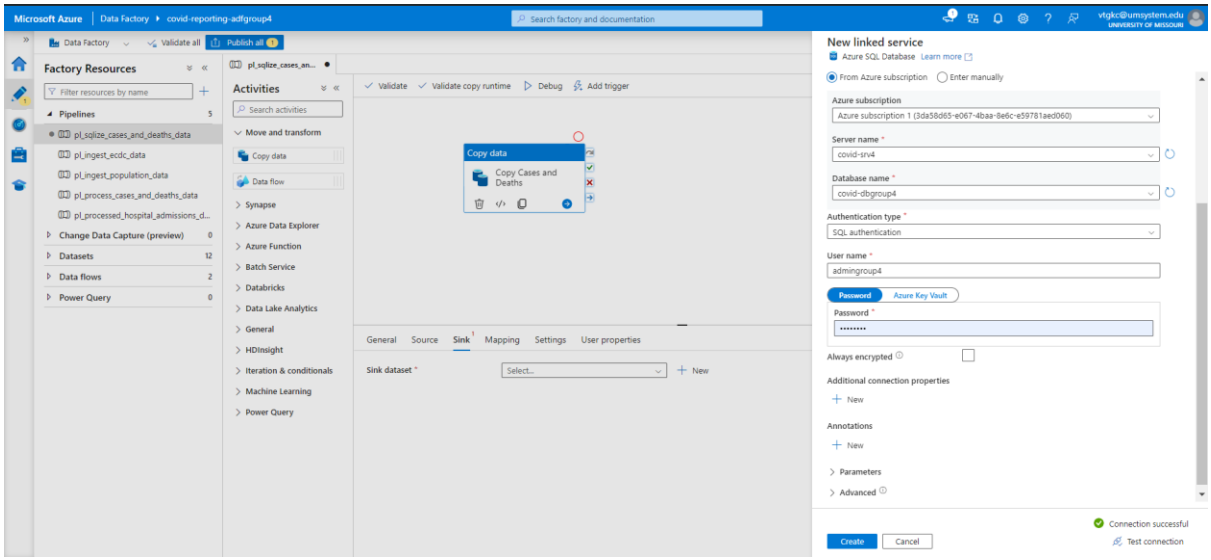
Upload the sql file. (Click on open query)



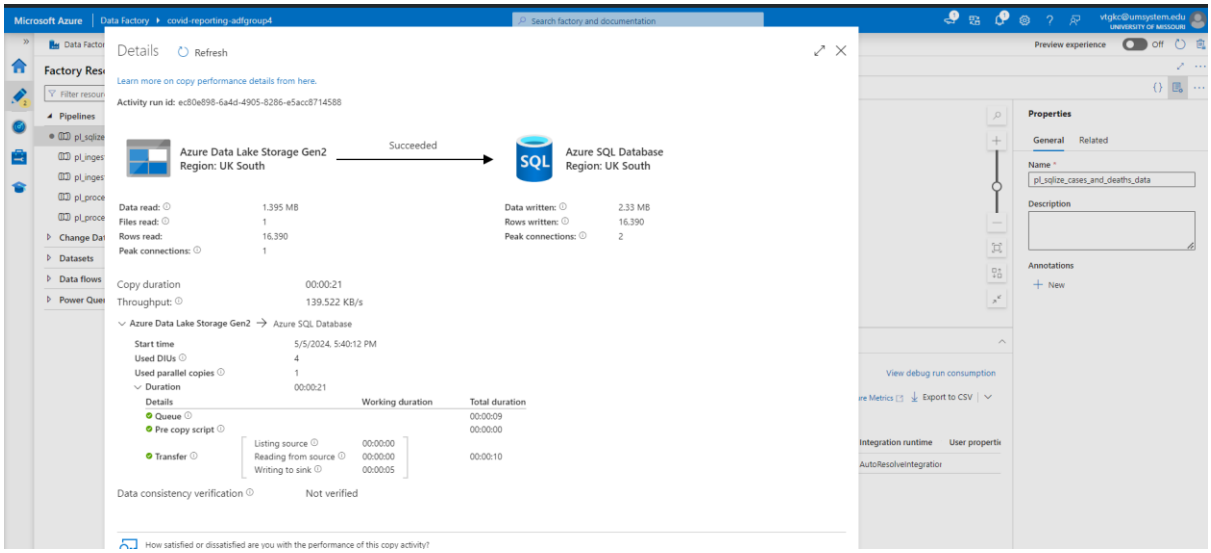
Query succeeded, created tables



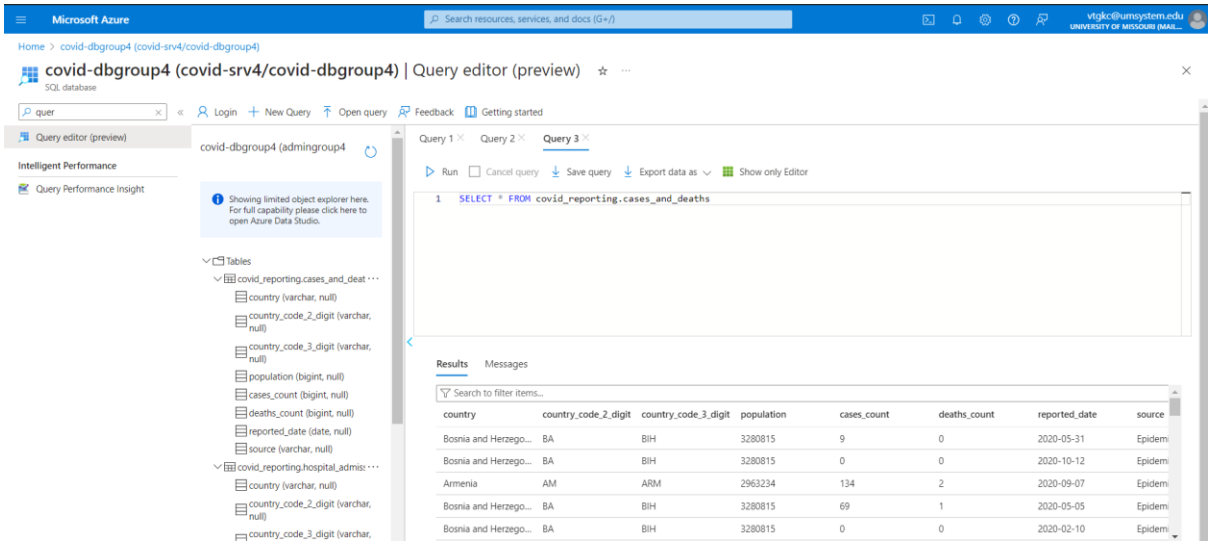
Create a new pipeline pl_sqlize_cases_and_deaths_data alongwith new linked service



Pipeline created pl_sqlize_cases_and_deaths

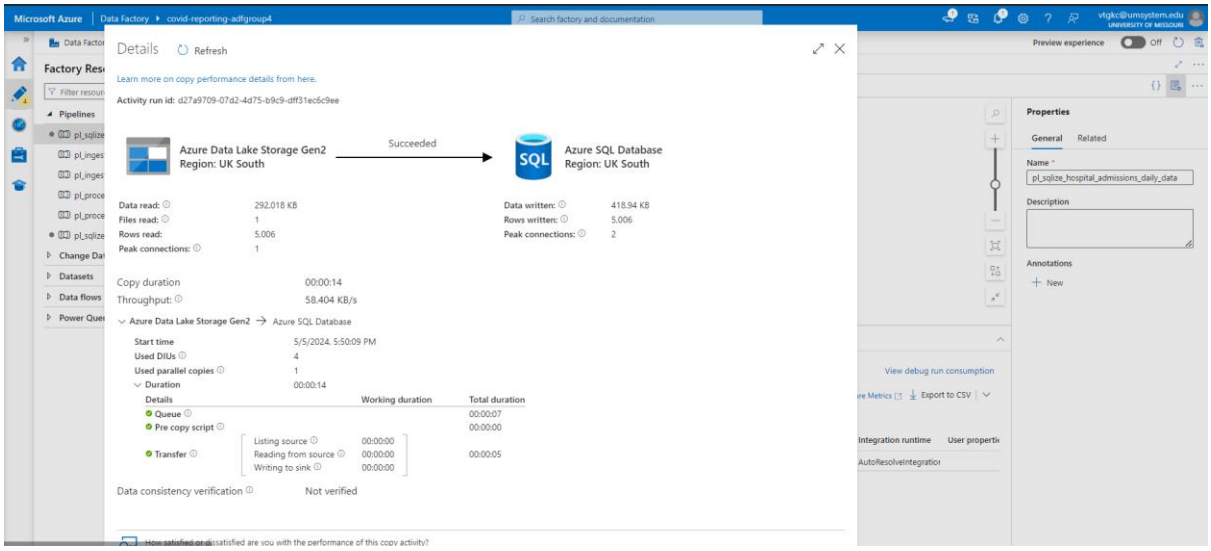


Succesfully running commands on the table covid_reporting.cases_and_deaths

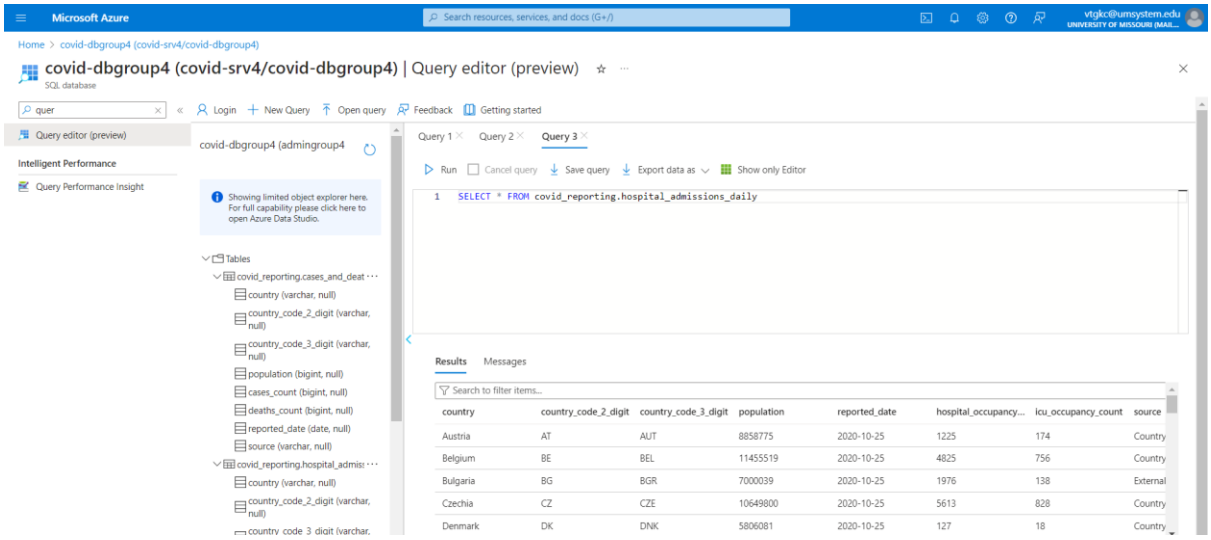


COPY activity Hospital admissions data

Pipeline created for hospital admissions daily data.



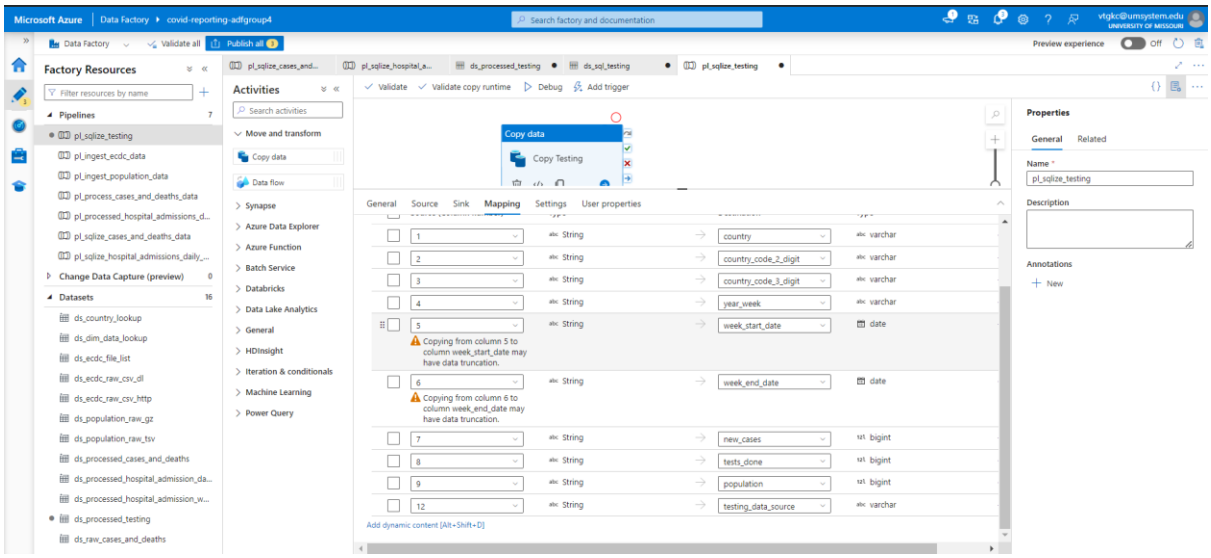
Successfully fetched data from hospital admissions data



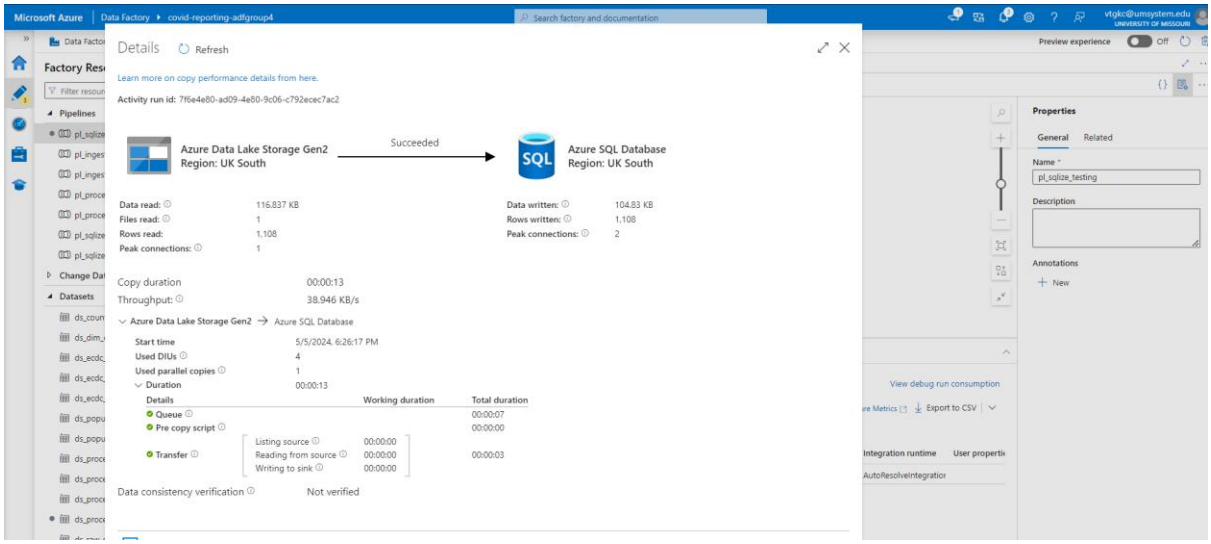
Copy activity Testing Data

Created gen 2 datatset and sql datatset for testing ds_processed_testing and ds_sql_testing

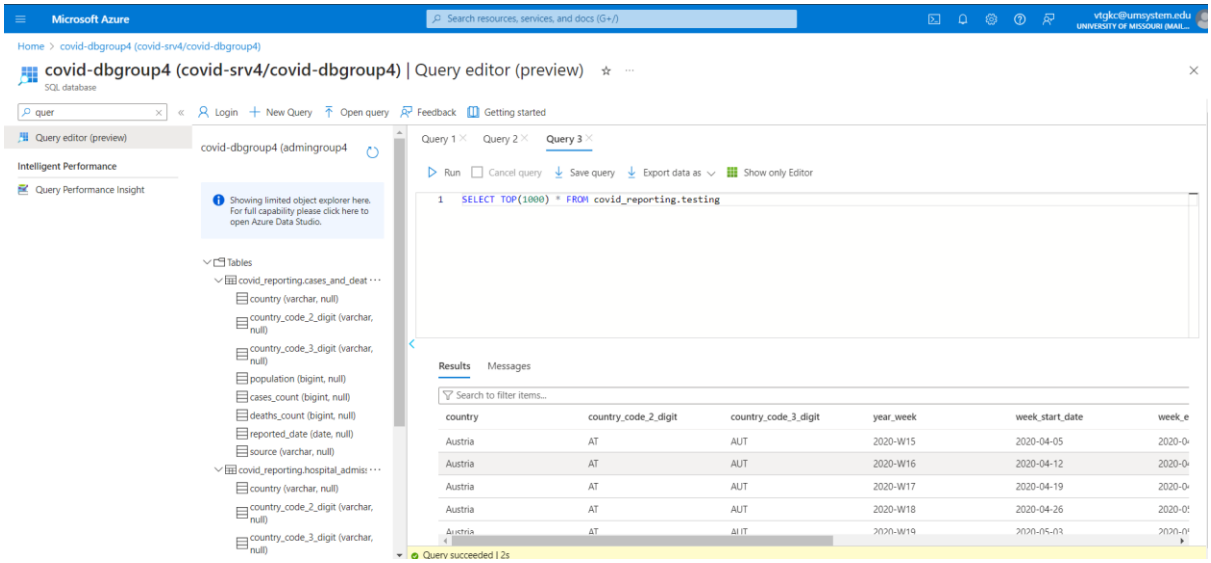
Removing last 9,10 columns in the file



Created pipeline for testing



SQL command successfully ran on testing data



Making Pipeline Production Ready

We automate all the pipelines instead of manual triggering.

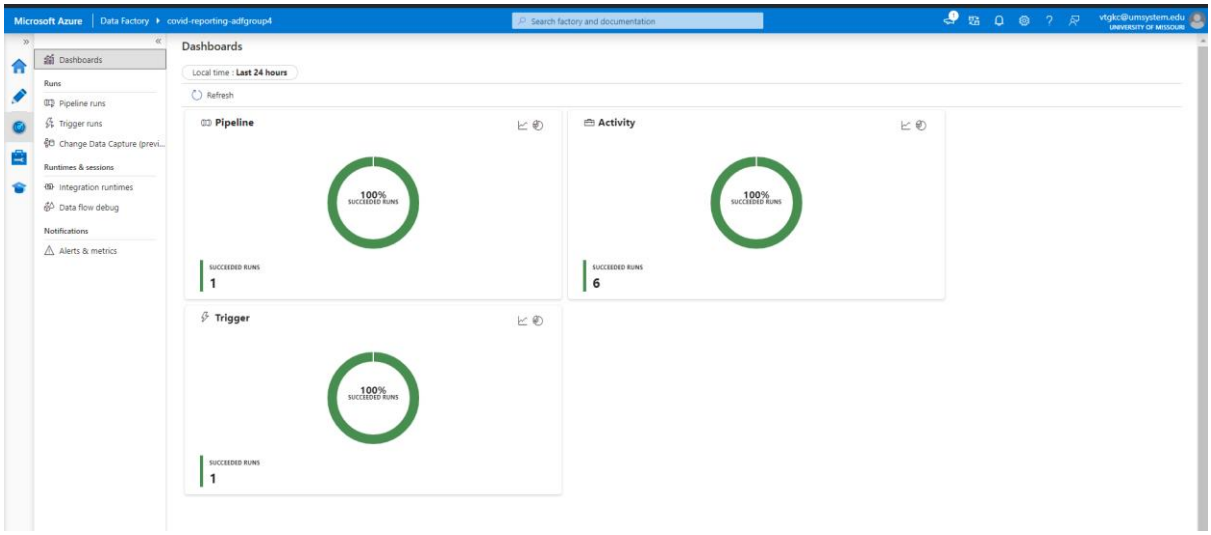
Monitoring

As part of monitoring, we want to ensure that the pipelines are up and running and in a healthy state.

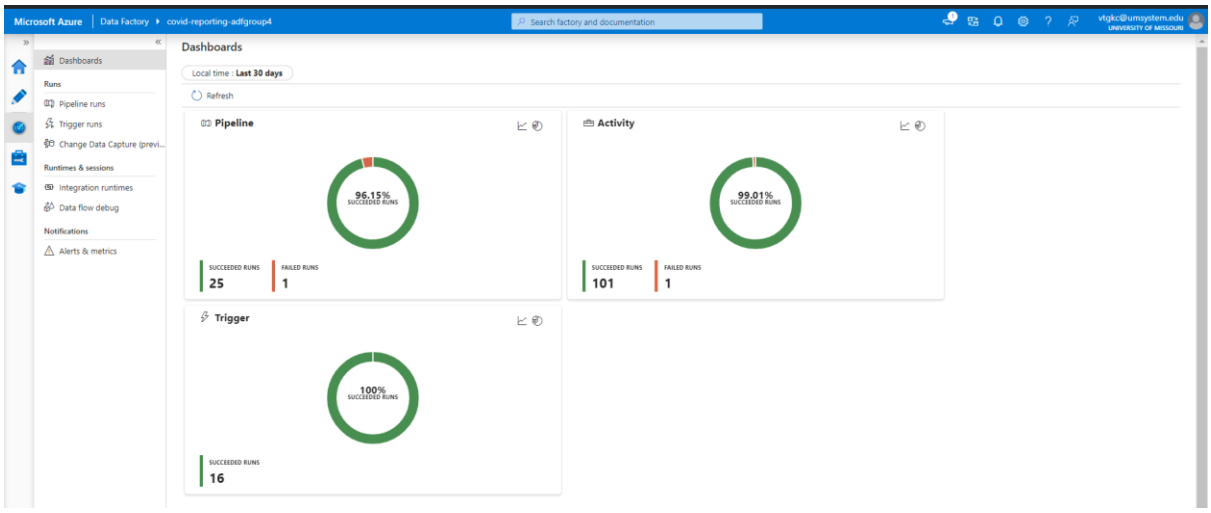
Data Factory Monitor

Here we look at the status of triggers and pipelines.

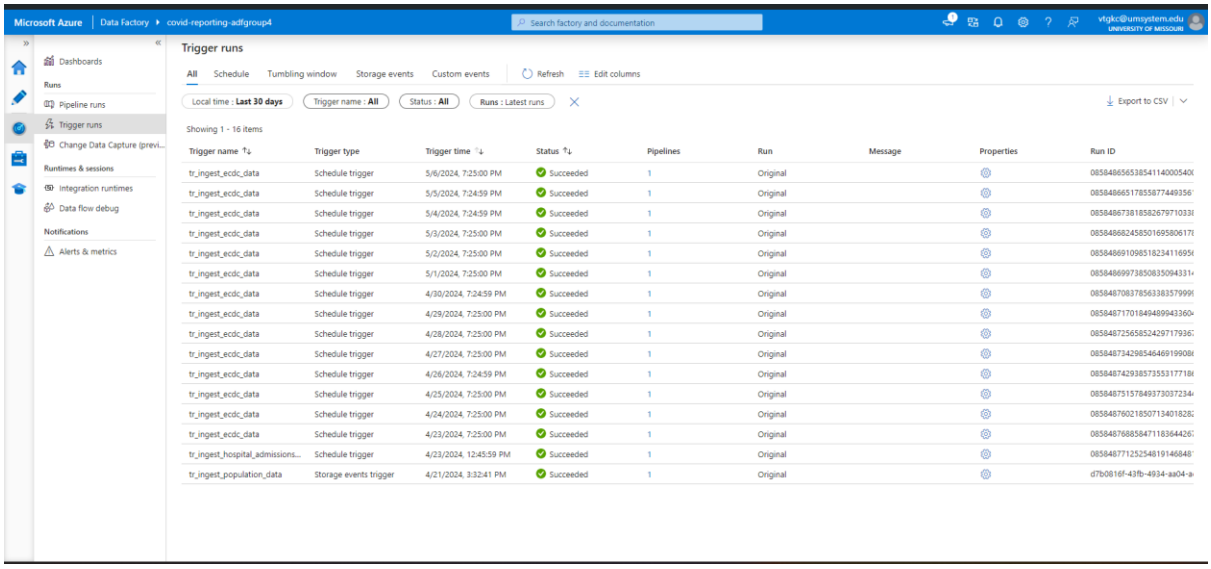
Monitoring results in last 24 hours



Monitoring results in last 30 days



We can see all the trigger runs in the figure below. We have set the option to view the trigger runs from last 30 days. If you run a failed one or if you're rerunning a trigger you will see them here as well.

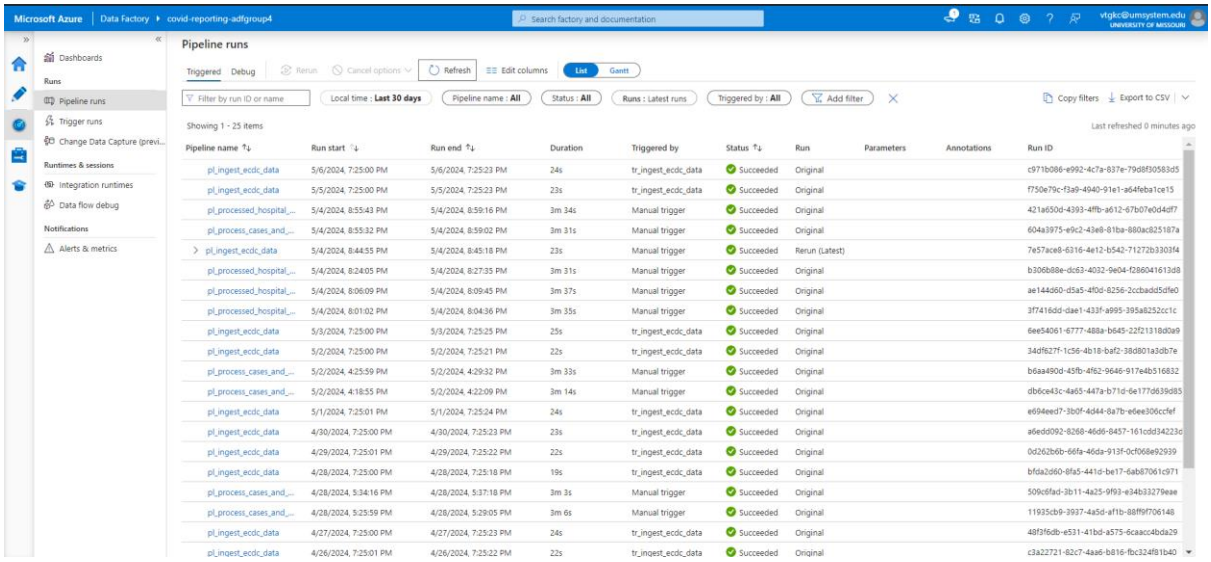


Here we will see the start and end, and then the duration, and then whatever trigger it's triggered. So we'll see any parameters, we don't have any parameters on those ones.

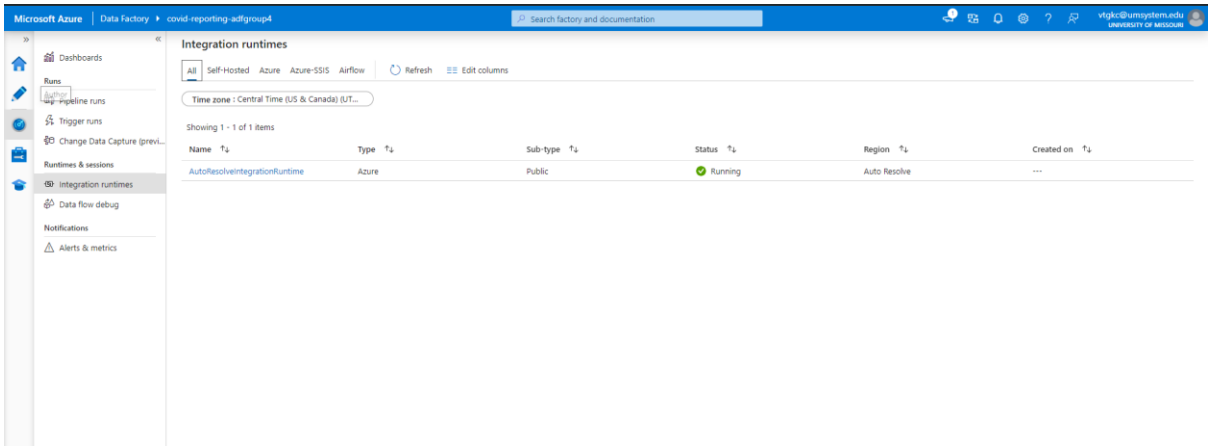
And if we put any annotations, we can see them as well, If they're errors, we'll see them and we can add further filters as well.

So we've got the filters on pipeline names, status and runs, and we can add annotations.

filter on the annotation for example. So that is what we do there.

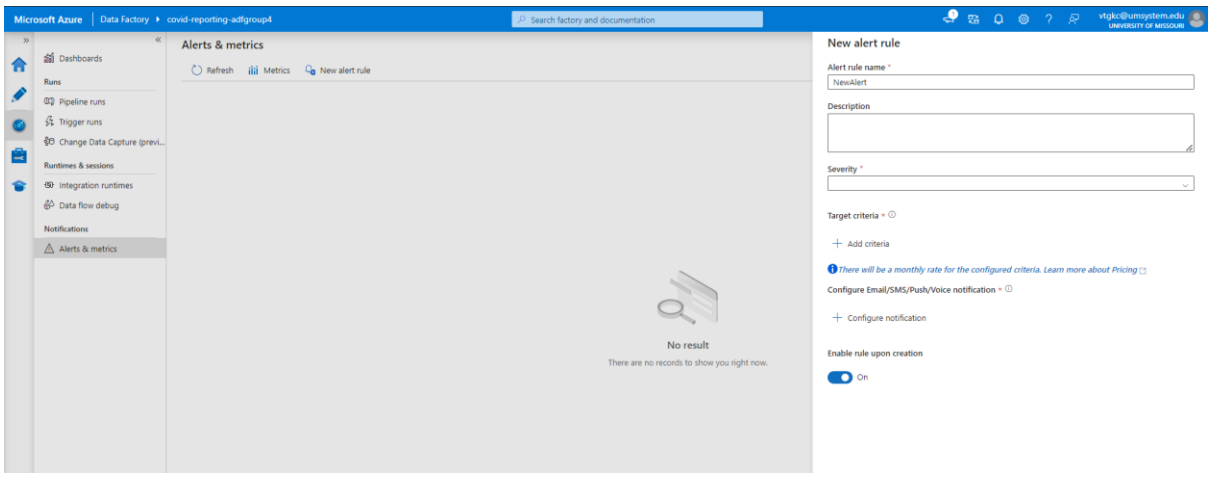


Next we've got the integration runtimes, so we can report on the status of the integration runtimes from here. At the moment, we only have the default order to resolve integration runtime.

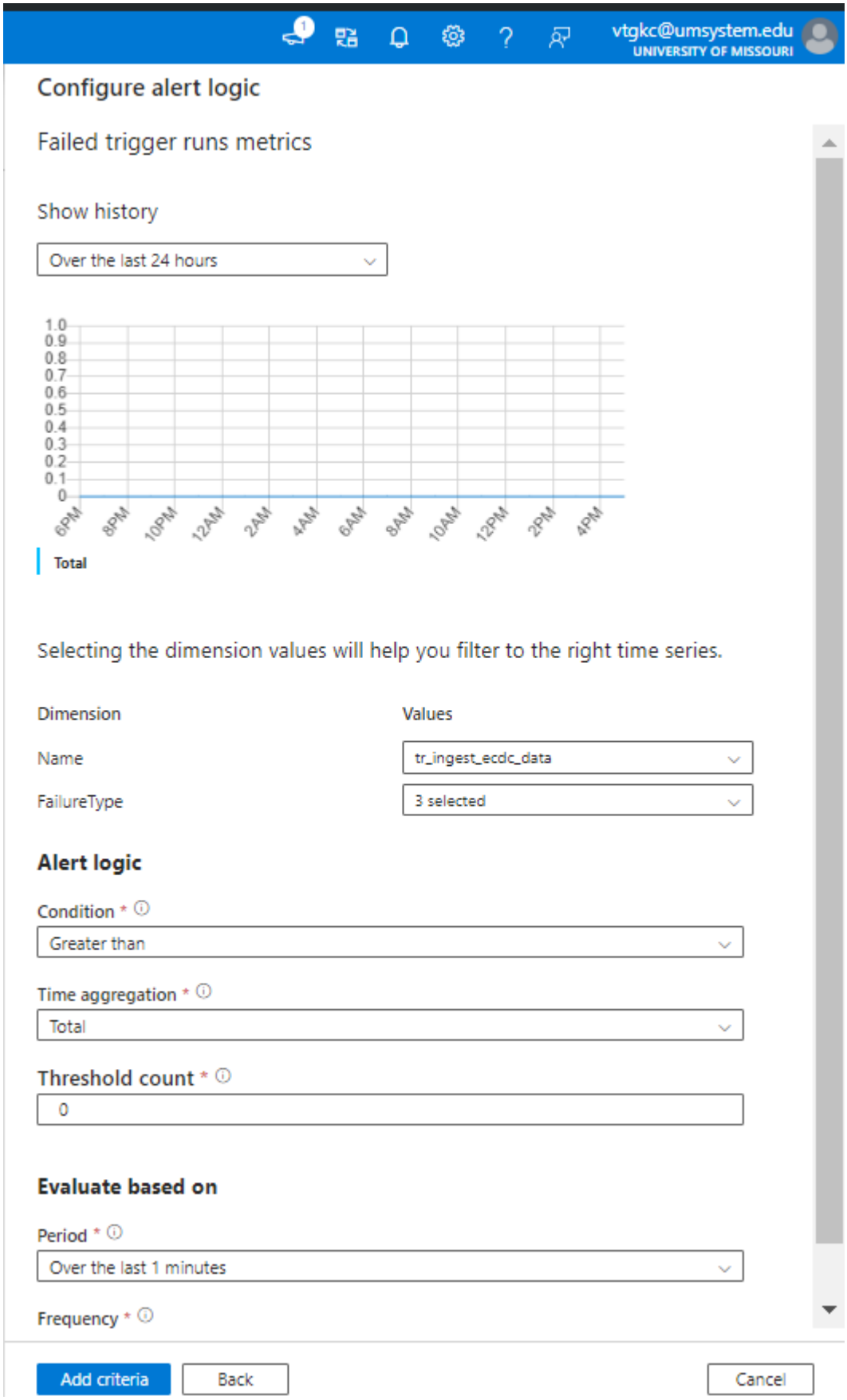


If we want a report on any failures, or successes of triggers, or activities, or pipelines, we can create a new alert.

So we set your target criteria and tell what to do, whether to send an email, or send an SMS, or a voice notification for example.



Give the details to add the criteria for which the alert shall be created.



Now, create a notification for the alert.

Configure notification

Notify your team via email and text messages or automate actions using webhooks, runbooks, functions logic apps or integrating with external ITSM solutions.

(•) Create new ( ) Use existing

Action group name *

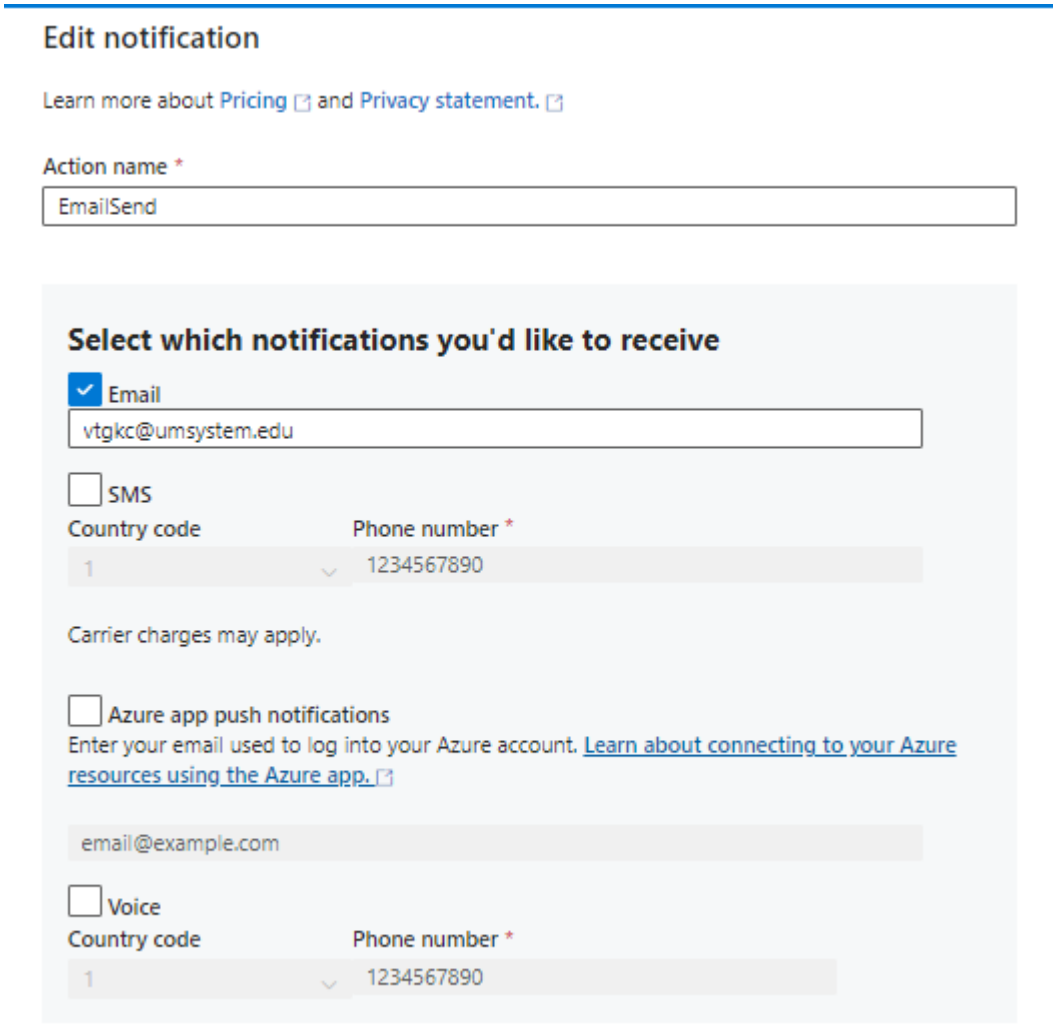
covid-support-project

Short name *

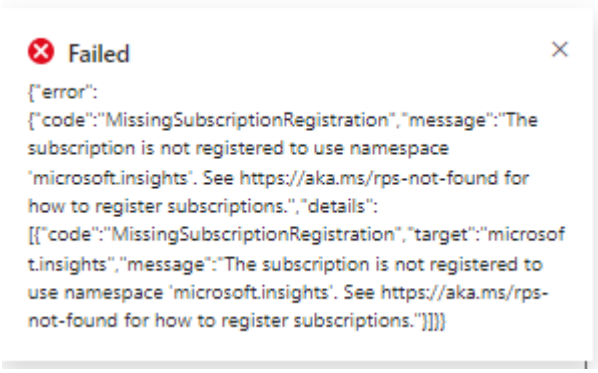
covidsupport

Notifications	Action type	Actions
EmailSend	Email/SMS/Push/Voice	

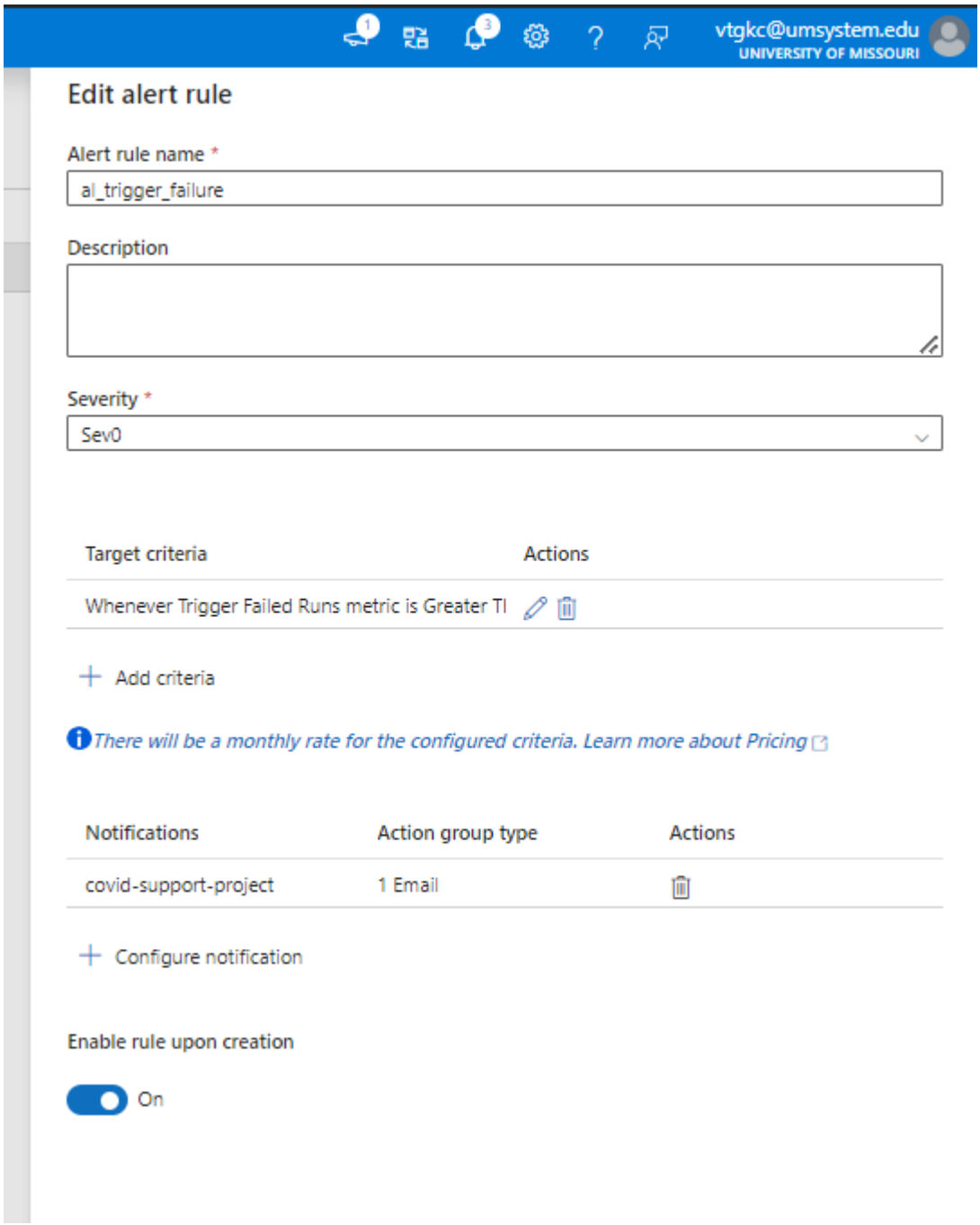
+ Add notification



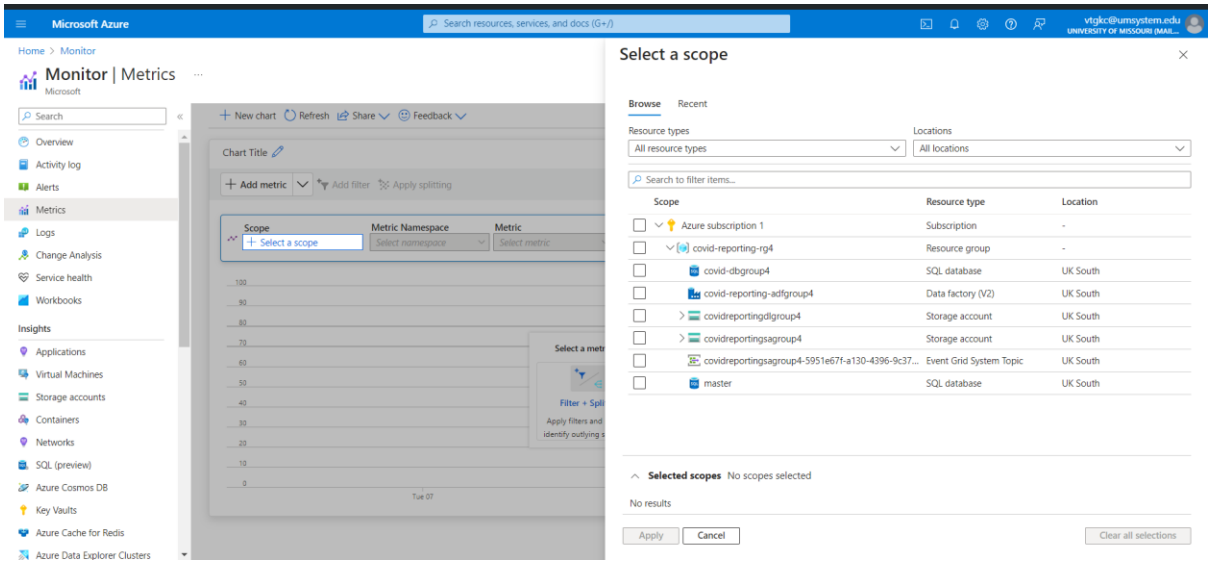
Error faced due to no subscription to Microsoft.insights.



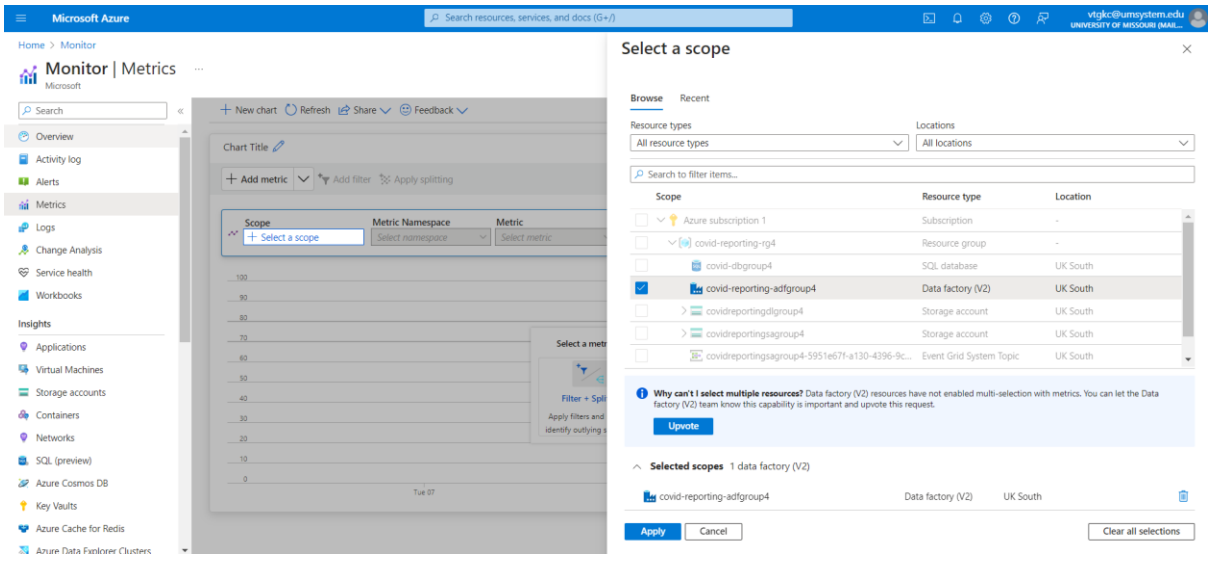
Now, we can see the alert is created and we can edit the alert as shown below.



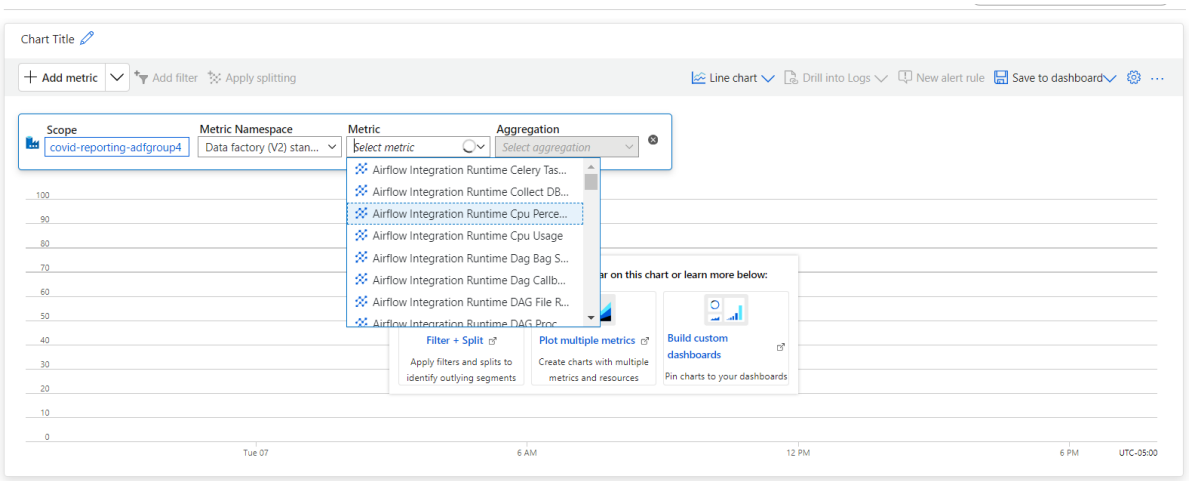
Now to monitor the metrics, navigate to alerts and metrics and click on metrics. Now we can see that a new tab is opened for monitoring. Now navigate to metrics to seeall the resources available in our susbscription.



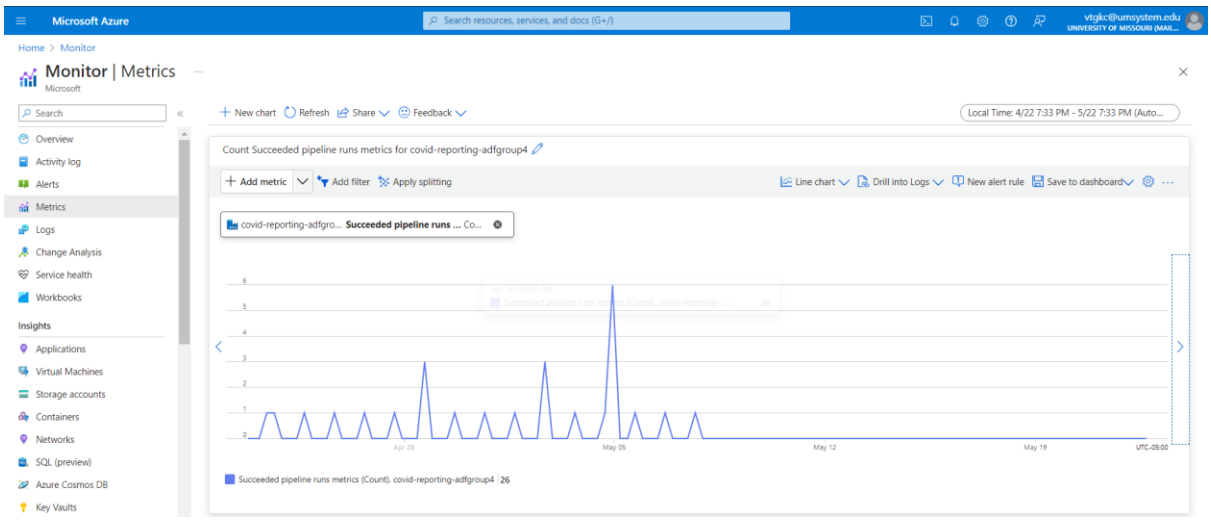
Now select data factory resource.



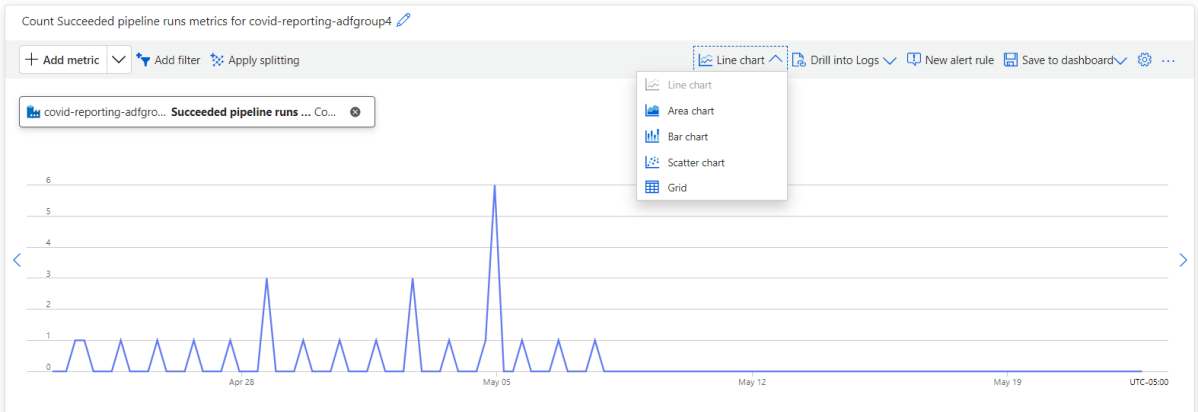
These are all the metrics available, we can select the metrics based upon the requirements.



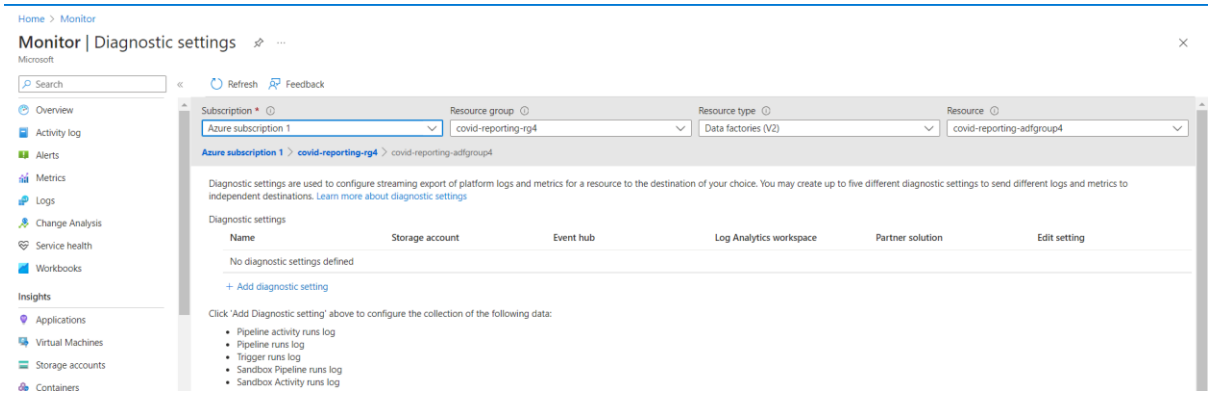
So now we have selected succeeded pipeline run metrics with aggregation as count, below we can see metrics for last 30 days.



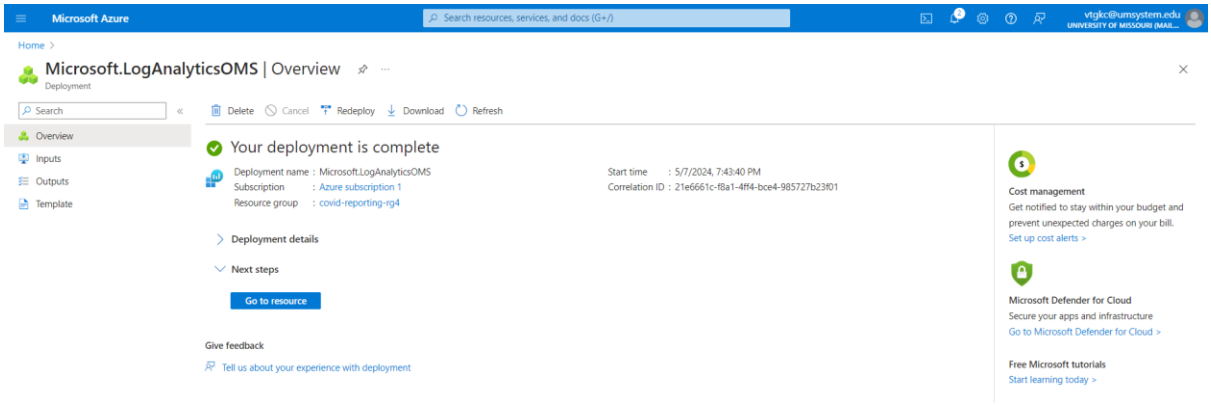
We can use different types of plots to visualize the metrics.



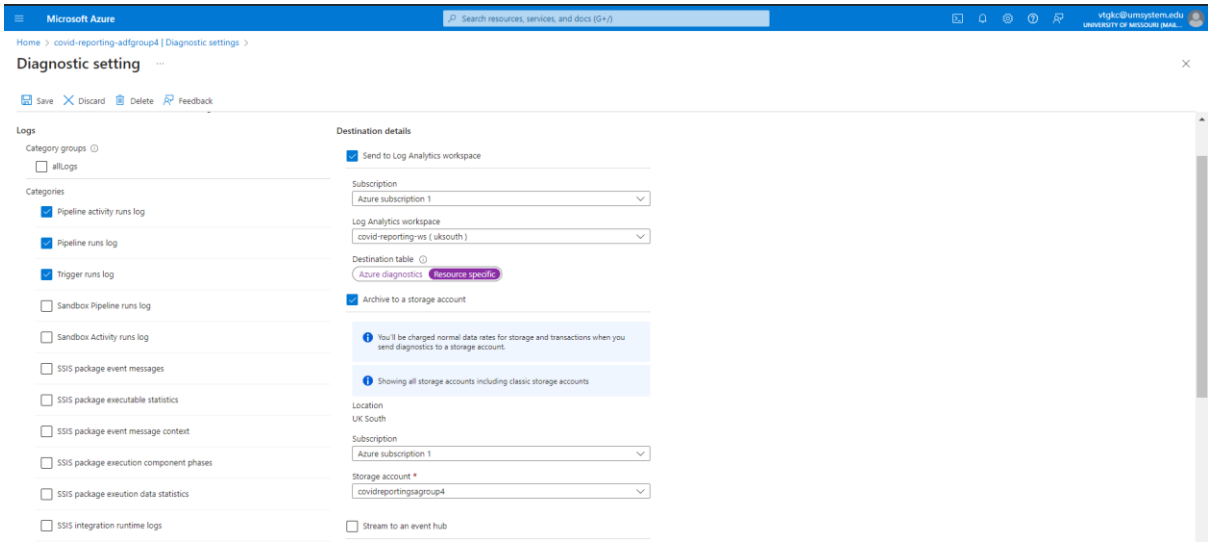
We can store the logs and metrics in the storage account by adding a diagnostic.



Deployment is completed for creating log analytics.



In the diagnostic settings, we are enabling option to store the metrics and logs in storage account. Selecting ‘Resource specific’ option to create separate tables for each of the metrics and logs.

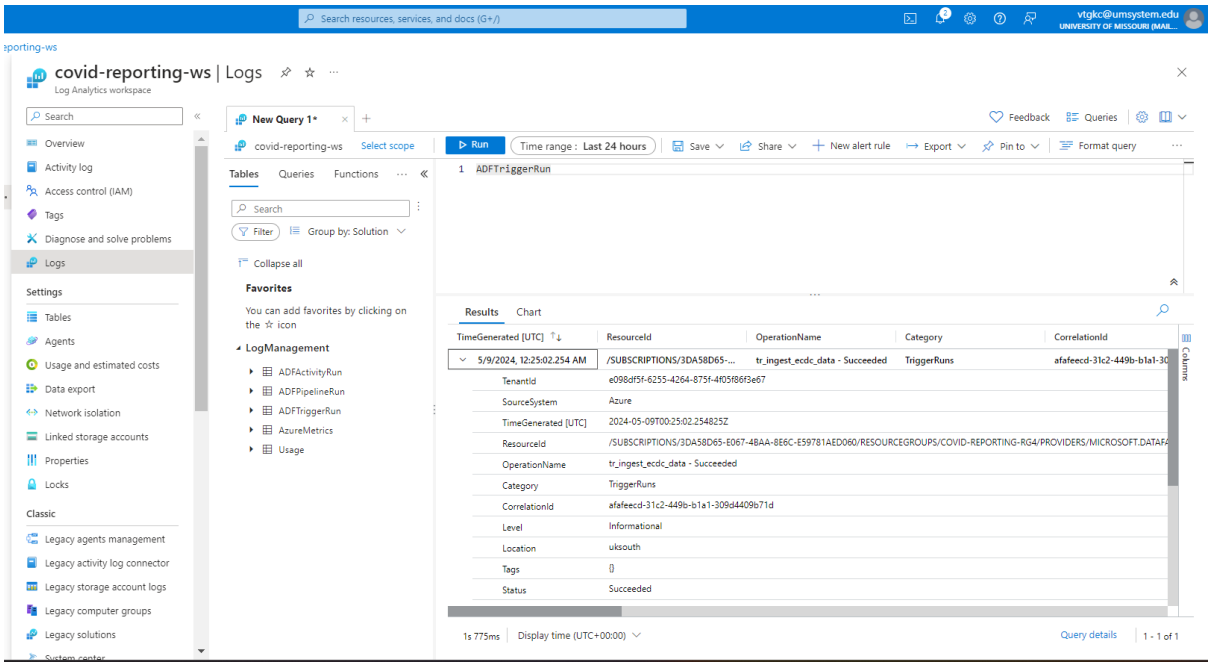


Navigate to ‘Log Analytics Workspace’ to see the logs. We can see the tables under the log management.

Using Kusto Query language to query on these tables.

Querying on the trigger run table to count the number of records.

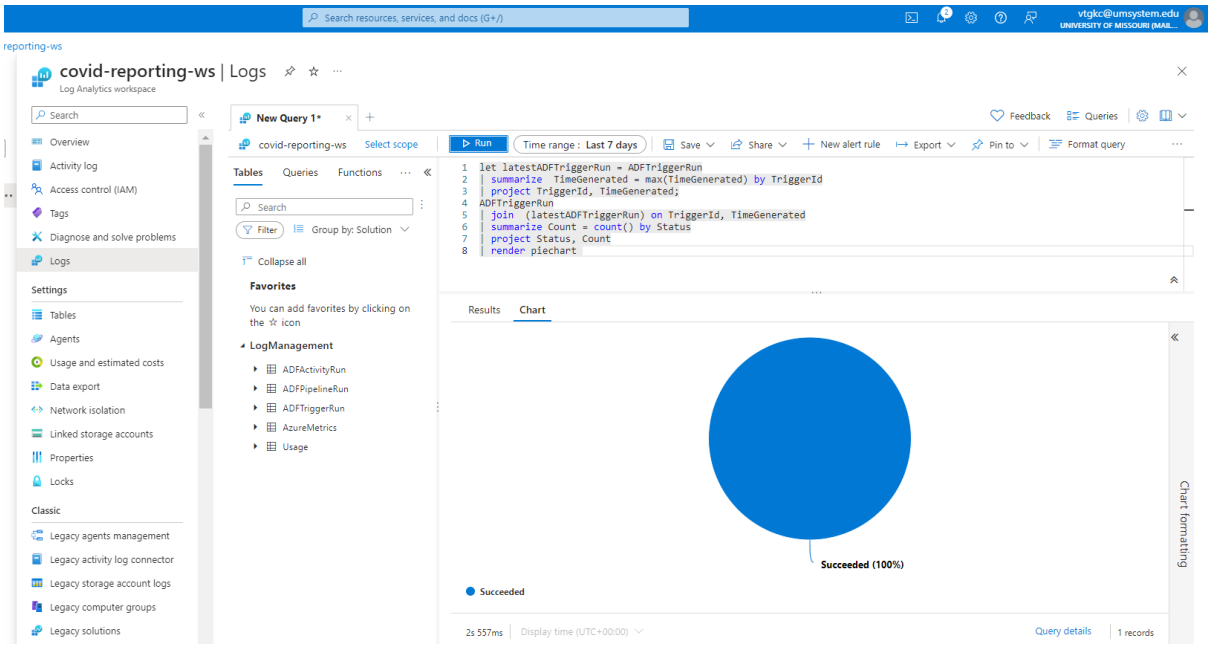
ADFTriggerRun – to see the details of the trigger run



To visualize the details of the trigger runs, the below code snippet in Kusto language is used.

We can find such details visually in the monitor section.

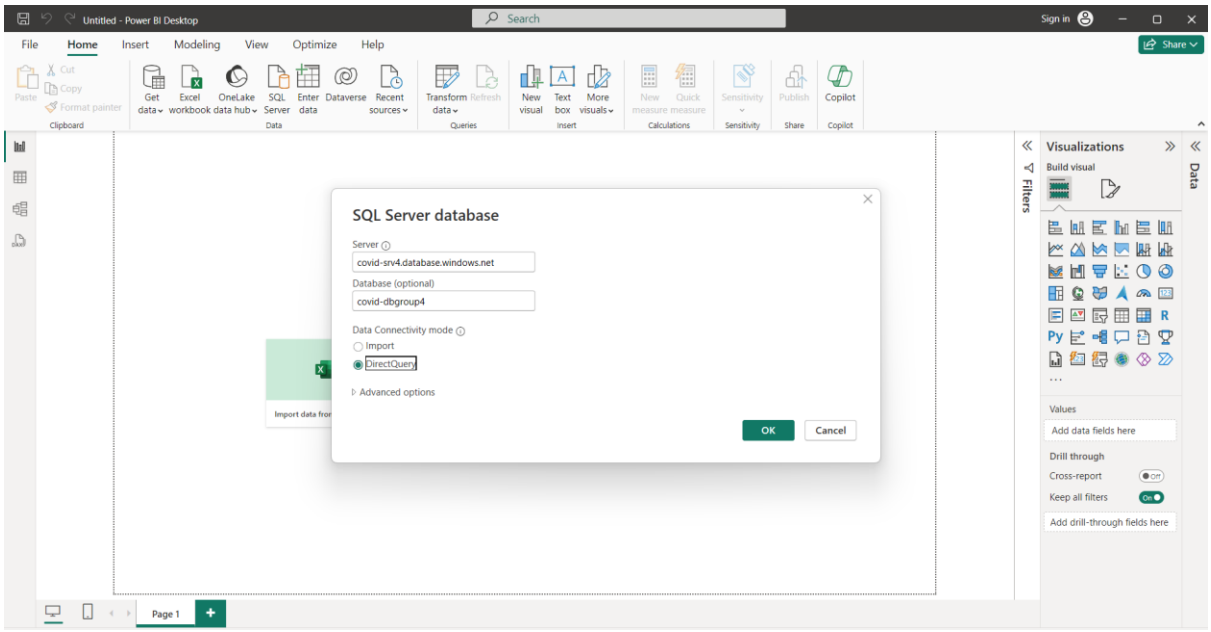
```
let latestADFTriggerRun = ADFTriggerRun
| summarize  TimeGenerated = max(TimeGenerated) by TriggerId
| project TriggerId, TimeGenerated;
ADFTriggerRun
| join  (latestADFTriggerRun) on TriggerId, TimeGenerated
| summarize Count = count() by Status
| project Status, Count
| render piechart
```



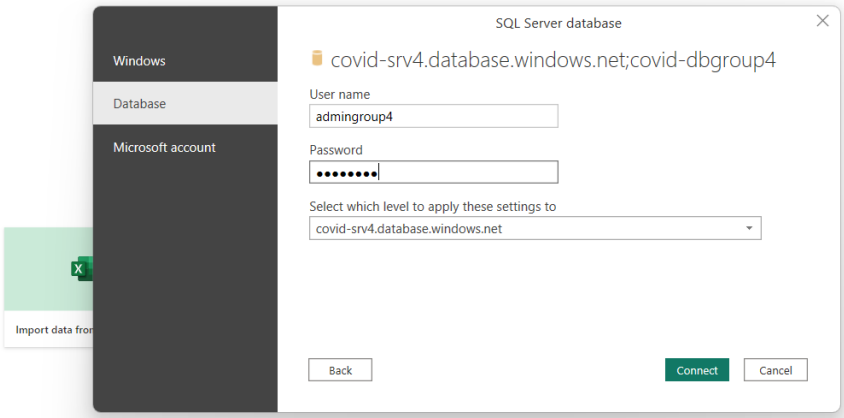
Power BI dashboard

Installed Microsoft Power BI Desktop from the link https://www.microsoft.com/en-us/download/details.aspx?id=58494

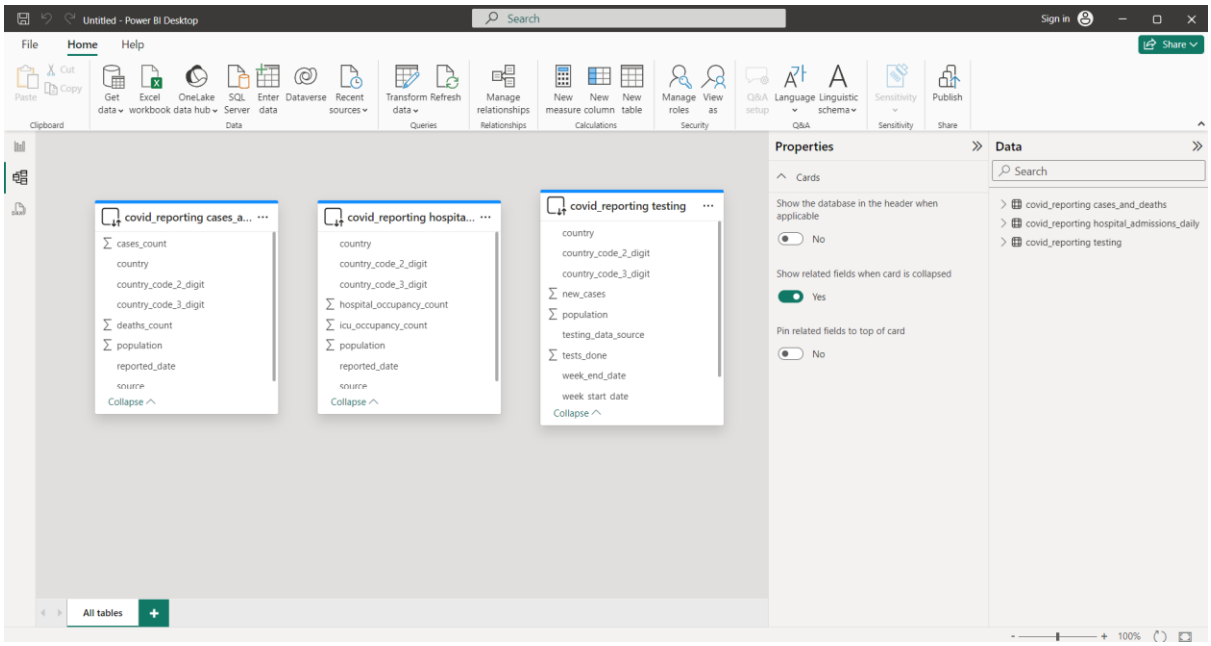
Connected the Azure SQL database to Power BI desktop



Giving credentials of our database



The tables that are linked from database.



Conclusion

To sum up, our project effectively constructed an Azure data processing and ingestion system. We successfully imported data from several sources, used pipelines to automate its flow, enhanced its quality through transformations, and utilized Azure SQL to enable perceptive analysis. The experience acquired taught me a lot about the value of monitoring and Azure's data management solutions. Future projects, including adding new data sources, executing sophisticated analytics, combining machine learning, or putting real-time processing in place, can make use of this foundation. By building on the accomplishments of this initiative, we may unleash even more possibilities for data-driven decision-making and additional innovation.

Challenges faced

The project faced a number of difficulties, including insufficient CPU cores for Databricks and HDInsights cluster creation, which required a special request to support and could have resulted in significant costs and delays; IP allocation issues that initially prevented login to the SQL server but were resolved by assigning an available IP; difficulties reproducing PowerBI visualizations because of IP allocation issues for the SQL server; and difficulties creating alerts because of no subscription to Microsoft.insights, which were fixed by adding it to the subscription.

Video Link:

https://drive.google.com/file/d/1jSFwxmFlDNdzbK_4VxyXTjeDZEz3cz5q/view?pli=1

Github repo link:

https://github.com/Viswanth13/CovidData_Azure4