

Azure e-commerce pipeline:

-Aravind Swamy

Project Introduction

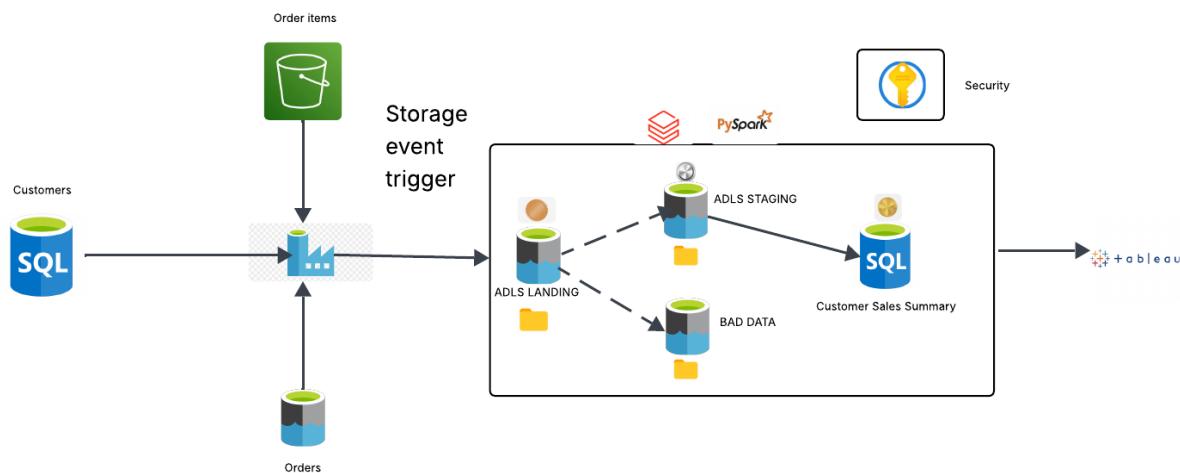
This project focuses on building a real-world, end-to-end data engineering pipeline using Microsoft Azure to simulate an e-commerce data processing system. The objective was to design an automated solution where incoming order data from third-party sources is ingested, validated, transformed, and stored for analytical consumption.

The pipeline is triggered automatically when new files arrive in Azure Data Lake Storage Gen2 and executes validation logic using Azure Databricks and PySpark. Data quality checks include duplicate detection and dynamic validation of order status using a lookup table maintained in Azure SQL Database. Based on validation results, data is either promoted to the staging layer or moved to a discarded layer.

The solution follows the Medallion Architecture (Bronze–Silver–Gold) and includes secure secret management using Azure Key Vault, parameterized pipelines for dynamic file handling, and cross-cloud data ingestion from Amazon S3. The final enriched dataset is stored in Azure SQL Database and used to generate key business insights such as total orders and customer spend.

The project follows a modern **event-driven architecture** and leverages **Azure Data Factory**, **Azure Databricks**, **Azure Data Lake Storage Gen2**, **Azure SQL Database**, **Amazon S3**, and **Azure Key Vault**.

Architecture:



Pipeline

The screenshot shows the Azure portal's Resource Group Overview page for 'Azure-project'. The top navigation bar includes links for 'How to manage changes with deployment tools?', 'Generate Bicep code to duplicate this resource group.', 'Export resource groups using Bicep or Terraform.', and various management actions like 'Create', 'Manage view', 'Delete resource group', 'Refresh', 'Export to CSV', 'Open query', 'Assign tags', 'Move', 'Delete', 'Export template', 'Open in mobile', and 'Add to service group'. The left sidebar lists navigation items such as 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Resource visualizer', 'Events', 'Settings', 'Cost Management', 'Monitoring', 'Automation', and 'Help'. The main content area shows the 'Essentials' section with details like 'Subscription (move)' to 'Azure subscription 1', 'Subscription ID' as '07c3dd0e-4cdc-4b0a-8812-8897e0742a2a', and 'Tags (edit)' with 'Add tags'. Below this are 'Deployments' (5 succeeded) and 'Location' (East US). The 'Resources' tab is selected, showing a list of resources with filters for 'Type', 'Name', and 'Location'. The listed resources include:

Type	Name	Location
Azure Databricks Service	project-sales-databricks-ws	East US
Data factory (V2)	project-sales-datafactory	East US
Key vault	project-sales-kvs	East US
Storage account	projectsaaravind	East US
Event Grid System Topic	projectsaaravind-6795bc93-c78a-44de-8a43-024cd04d0a70	East US
SQL database	projectsqljdb (projectsqlierveravaravind/projectsqljdb)	West US 2
SQL server	projectsqljerveravaravind	West US 2

In the initial pipeline as soon as a third-party service drops a file named orders.csv into the landing folder of Azure Data Lake Storage Gen2.

Business Requirements

As soon as the file arrives:

1. There should be no duplicate order_id in the file.
2. The order_status should be valid. (which will be cross checked against the valid status stored in Azure SQL DB)

If both conditions are satisfied:

- Move the file to the **staging** folder.

If any condition fails:

- Move the file to the **discarded** folder.

Thus we first create the 3 folders first

The screenshot shows the Azure portal's Container page for 'sales' in 'projectsaaravind'. The top navigation bar includes links for 'Search', 'Add Directory', 'Upload', 'Refresh', 'Delete', 'Copy', 'Paste', 'Rename', 'Acquire lease', 'Break lease', and 'Edit columns'. The left sidebar lists navigation items such as 'Overview', 'Diagnose and solve problems', 'Access Control (IAM)', and 'Settings'. The main content area shows a table of blobs with columns for 'Name', 'Last modified', 'Access tier', and 'Blob type'. The blobs listed are:

Name	Last modified	Access tier	Blob type
SalesmyFolder5	11/27/2023, 2:48:29 PM		
discarded	11/27/2023, 9:47:08 AM		
landing	11/27/2023, 6:15:58 PM		
staging	11/27/2023, 6:15:57 PM		

The screenshot shows the Azure Storage Explorer interface. On the left, there's a sidebar with options like 'Overview', 'Diagnosis and solve problems', 'Access Control (IAM)', and 'Settings'. The main area displays a blob container named 'sales' with a single item: 'orders.csv'. The 'Notifications' pane on the right lists several recent events:

- Successfully uploaded blob(s) - 9 minutes ago
- Subscription Registering Resource Provider - 13 minutes ago
- Successfully deleted blobs and directories - 13 minutes ago
- Successfully uploaded blob(s) - 20 minutes ago
- Successfully deleted blobs and directories - 29 minutes ago
- Successfully uploaded blob(s) - 31 minutes ago
- Successfully uploaded blob(s) - 31 minutes ago
- US\$200.00 credit remaining - 31 minutes ago

The taskbar at the bottom shows the date as 11/27/2025 and the time as 4:43 PM.

And we set up Azure Databricks cluster and write our validation and transformation code in pyspark

The screenshot shows the Microsoft Azure Databricks Compute page. The left sidebar includes sections for 'Compute', 'Data Engineering', 'Job Runs', 'AI/ML', 'Playground', 'Experiments', 'Features', 'Models', and 'Serving'. The main area is titled 'Compute' and shows an 'All-purpose compute' cluster. A warning message states: 'This account may not have enough CPU cores to start a cluster. Contact your administrator to increase the limits.' Below this, a table lists the cluster details:

State	Name	Runtime	Active memory	Active cores	Active DBU / h	Source	Creator	Notebooks	
Running	aravindswamy198@gmail.com's Cluster	2025-11-27 13:19:20	11.3	16 GB	4 cores	1	UI	aravindswamy198@gmail.com	1

At the bottom, there are navigation links for 'Previous', 'Next', and '20 / page'.

1

```
dbutils.fs.unmount('/mnt/sales')
dbutils.fs.mount(
  source = 'wasbs://sales@projectsaaravind.blob.core.windows.net',
  mount_point = '/mnt/sales',
  extra_configs={'fs.azure.account.key.projectsaaravind.blob.core.windows.net': '79j1x9z38RE61fFb5K9t807pozD07hu289oZ3#fCDbyqCIGYJDVqg61Bp+9WxQPeAD0Xhng+ASTU4ISqg--'}
```

/mnt/sales has been unmounted.
Out[48]: True

2

```
%fs  
ls /mnt/sales/landing
```

OK

3

```
spark
```

SparkSession - hive
SparkContext
SparkUI
Version
2.4.3.0
Master
local[*, 4]
AppName
Databricks Shell

And we set up the trigger

Edit trigger

Name: trigger

Description: as soon as file drops in landing folder, this trigger should work

Type: BlobEventTrigger

Account selection method: From Azure subscription

Azure subscription: Azure subscription 1 (07c3dde0-4cd4-4b0a-8612-8897e0742a2a)

Storage account name: projectsaaravind

Container name: sales

Blob path begins with: landing

Blob path ends with: orders.csv

Event: Blob created Blob deleted

Ignore empty blobs: Yes

Annotations

Continue Cancel

The screenshot shows the Microsoft Azure Storage Explorer interface. The left sidebar has a 'Sales' container selected. Under 'Containers', there is a single item named 'orders_new.csv'. The main pane shows a table with one row, indicating the file was modified on 11/27/2023 at 3:07:10 PM by 'Host (Inferred)' and is 2.68 MB in size.

We have also added the file `orders_new.csv` here to prove that our pipeline is *generic and dynamic*, not hardcoded to `orders.csv`

Now the pipeline is run

The screenshot shows the Microsoft Azure Data Factory pipeline runs page. The left sidebar has 'Pipeline runs' selected. The main area shows a table for 'Activity runs' with one row:

Activity name	Activity status	Activity type	Run start	Duration	Integration runtime	User properties	Activity run ID	Log
Notebook1	Succeeded	Notebook	11/27/2023, 6:15:02 PM	1m 7s	AutoResolveIntegrationRuntime (East US)	f8cb5bab-7341-4cf5-0b12-7424f7166c4b		

After ADF

The screenshot shows the Microsoft Azure Storage Explorer interface. The left sidebar has a 'Sales' container selected. Under 'Containers', there is a folder named 'landing'. The main pane shows a table with zero items, indicating the folder is empty.

We can see that the landing folder is empty

The screenshot shows the Azure Storage Explorer interface. The left sidebar has a 'Sales' container selected. The main area displays two files: 'orders.csv' and 'orders_new.csv'. A status bar at the bottom indicates 'Last modified' times and 'Access tier' (Hot) for both files.

And both the files have been transferred to the staging folder.

Pipeline trigger runs

The screenshot shows the 'Pipeline runs' page in Microsoft Azure Data Factory. The left sidebar shows 'Runs' under 'Pipeline runs'. The main table lists 7 runs for 'pipeline1' over the last 24 hours. The columns include Pipeline name, Run start, Run end, Duration, Triggered by, Status, Run ID, and Annotations. Some runs show a green checkmark (Succeeded), while others show a red X (Failed).

The screenshot shows the 'pipeline1' details page in Microsoft Azure Data Factory. The left sidebar shows 'Factory Resources' with 'Pipelines' selected. The main area shows the 'Activities' tab, which lists various activities like Move and transform, Synapse, Azure Data Explorer, etc. A 'Notebook' activity is selected, showing its properties. The 'Settings' tab is active, displaying the 'Notebook path' as '/Users/aravindswamy198@gmail.com/sal...'.

Linked services were set up in order to connect various elements of our pipeline

Edit linked service

 Azure Databricks [Learn more](#) 

AzureDatabricks1project_ls

Description

Connect via integration runtime *

AutoResolveIntegrationRuntime 



Account selection method *

From Azure subscription Enter manually

Databrick Workspace URL *

Authentication type *



[Access token](#) [Azure Key Vault](#)

AKV linked service *



Secret name *



Edit

Secret version



Edit

Select cluster



Existing cluster ID *



Edit linked service

 Azure Data Lake Storage Gen2 [Learn more](#) 

Name *

AzureDataLakeStorage1projects1

Description

To link DF to Storage

Connect via integration runtime *

AutoResolveIntegrationRuntime



Authentication type

Account key

Account selection method

From Azure subscription Enter manually

URL *

<https://projectsaaravind.dfs.core.windows.net/>

Storage account key *

.....

Test connection

To linked service To file path

Annotations

 New

> Parameters

> Advanced 

Save

Cancel

 Test connection

Edit linked service



Name *

AzureKeyVault1project_ls

Description

Azure key vault selection method

- From Azure subscription Enter manually



Base URL *

https://project-sales-kvs.vault.azure.net/

Authentication method

System-assigned managed identity



Managed identity name: **project-sales-datafactory**

Managed identity object ID: **2bd302fe-123e-47b2-9753-b88bc6662703**

Grant Data Factory service managed identity access to your Azure Key Vault. [Learn more](#)

Test connection

- To linked service To secret

Annotations

New

Parameters

Advanced

Save

Cancel

Test connection

The key vault is used to store various secret keys and passwords.

Azure Key Vault interface showing the 'Secrets' blade. There are two secrets listed:

Name	Type	Status	Expiration date
databricks-access-token		Enabled	
sqlserverpassword		Enabled	

Azure SQL DB is created

Azure SQL Database interface showing the 'Getting started' tab selected. The page displays basic information about the database and provides links to start working with it.

Start working with your database

Connect to your database and start working with data with a few simple steps. [Learn more](#)

Configure access

Configure network access to your SQL server. [Learn more](#)

Configure

Connect to application

Use connection strings to connect to your SQL database from your applications and favorite tools.

See connection strings

Start developing

Work in your database by using tools to add, modify and query data. [Compare tools](#)

Open Azure Data Studio

Open in Visual Studio

Open in Visual Studio Code

Mirror database in Fabric

Replicate existing databases in Fabric, and help your team achieve streamlined ETL and operational analytics goals. [Learn more](#)

The screenshot shows the Azure portal interface for a SQL database named 'projectsqlserveraravind/projectsqldb'. The left sidebar includes options like Overview, Activity log, Tags, Diagnose and solve problems, and a 'Query editor (preview)' section which is currently selected. The main area displays a 'Query 1' editor with the query 'select * from valid_order_status;'. Below the editor, the results pane shows the following data:

status_name
ON_HOLD
PAYOUT_REVIEW
PROCESSING
CLOSED
SUSPECTED_FRAUD
COMPLETE
PENDING
CANCELLED

And a table is created this particular table will hold the valid order statuses that are recognized. This is to check the valid orders based on the status.

The screenshot shows the 'Features' blade for the 'projectsqlserveraravind/projectsqldb' database. It lists several features with their current configuration status:

- Microsoft Extra admin**: NOT CONFIGURED
- Microsoft Defender for SQL**: NOT CONFIGURED
- Automatic tuning**: CONFIGURED
- Auditing**: NOT CONFIGURED
- Fallover groups**: NOT CONFIGURED

Under 'Available resources', there is a table showing one database entry:

Name	Type	Status	Pricing tier
projectsqlldb	SQL database	Online	Basic

Now showing the parameterized Approach: (Any file) and not hardcoded

Edit trigger

Trigger Run Parameters

i Parameters that are not provided a value will not be included in the trigger.

Name	Type	Value
filename	string	@triggerBody().filename



Make sure to "Publish" for trigger to be activated after clicking "OK"

OK

Cancel

General Azure Databricks **Settings** User properties

Notebook path *

✓ Base parameters

<input type="checkbox"/> Name	Value
<input type="checkbox"/> filename	<input type="text" value="@pipeline().parameters.filename"/>

✓ Append libraries

Parameters Variables Settings Output

<input type="checkbox"/> Name	Type	Default value
<input type="checkbox"/> filename	String	<input type="text"/>

Now to show that it works,

The landing folder is currently empty

Search

Overview

Authentication method: Access key (Switch to Microsoft Entra user account)

sales > landing

Showing all 0 items

Name	Last modified	Access tier	Blob type

And the staging folder also doesn't have the data that we are going to add.

Now triggering the pipeline by uploading a new file in the landing folder,

Pipeline name	Run start	Run end	Duration	Triggered by	Status	Run	Parameters	Annotations	Run ID
pipeline1	11/27/2025, 8:07:58 PM	--	14s	trigger	In progress	Original			Sa0d01d5-a0c0-44ef-802b-9da5ced5d5
pipeline1	11/27/2025, 6:15:00 PM	11/27/2025, 6:16:10 PM	1m 10s	trigger	Succeeded	Original			24536a5d-d9bb-42f1-9cd9-a29e05e
pipeline1	11/27/2025, 6:11:38 PM	11/27/2025, 6:12:35 PM	57s	trigger	Failed	Original			666c0172-c6c1-48f2-972c-95ab4819-cc
> pipeline1	11/27/2025, 6:09:28 PM	11/27/2025, 6:10:36 PM	1m 9s	Manual trigger	Failed	Rerun (Latest)			ae071b74-cc4f-4d6b-864e-60298391f9k
pipeline1	11/27/2025, 5:33:38 PM	11/27/2025, 5:49:25 PM	15m 47s	trigger	Failed	Original			5205cd94-cb61-47fe-87eb-84ec31545
pipeline1	11/27/2025, 5:16:06 PM	11/27/2025, 5:18:48 PM	2m 43s	trigger	Failed	Original			f20099a5-5a8b-4153-9783-2253c5ce9
pipeline1	11/27/2025, 5:04:40 PM	11/27/2025, 5:11:33 PM	6m 54s	trigger	Failed	Original			75e196ce-8225-4ab9-9c58-3361eb415k
> pipeline1	11/27/2025, 4:42:17 PM	11/27/2025, 4:55:36 PM	13m 19s	Manual trigger	Failed	Rerun (Latest)			26b0f5cb-44ea-4c7e-aaf0-e82c418189

Succeeded

Activity name	Activity status	Activity type	Run start	Duration	Integration runtime	User properties	Activity run ID	Log
Notebook1	Succeeded	Notebook	11/27/2025, 8:08:01 PM	1m 21s	AutoResolveIntegrationRuntime (East US)		0cf51971-4cc3-483e-959e-ba6e5d9a2544	

It works, now we can see the new file added to our staging area.

Microsoft Azure | Overview | Container

Home > Azure-project > projectsaaravind | Containers > sales

Search | Add Directory | Upload | Refresh | Delete | Copy | Paste | Rename | Acquire lease | Break lease | Edit columns

Diagnose and solve problems | Access Control (IAM) | Settings

Search blobs by prefix (case-sensitive)

Showing all 3 items

Name	Last modified	Access tier	Block type	Size	Lease state
11	11/27/2021, 8:09:12 PM	Hot (Inferred)	Block blob	224 B	Available
neworders.csv	11/27/2021, 6:15:58 PM	Hot (Inferred)	Block blob	224 B	Available
orders.csv	11/27/2021, 3:07:10 PM	Hot (Inferred)	Block blob	2.86 MB	Available
orders_new.csv					

Making the pipeline more generic (generic mount code):

```

alreadyMounted = False
for x in dbutils.fs.mounts():
    if x.mountPoint == "/mnt/sales":
        alreadyMounted = True
        break
    else:
        alreadyMounted = False
print(alreadyMounted)

```

storageAccountKey = dbutils.secrets.get(scope = databricksScope,key='storage-account-key')

```

if not alreadyMounted:
    dbutils.fs.mount(
    source = 'wasbs://sales@projectsaaravind.blob.core.windows.net',
    mount_point = '/mnt/sales',
    extra_configs={'fs.azure.account.key.projectsaaravind.blob.core.windows.net':storageAccountKey}
)
alreadyMounted = True
print("Mounting done successfully")
else:
    print('Already mounted')

/mnt/sales has been unmounted.
Out[40]: True

```

We have also stored the storage account key in key vault rather than using it hardcoded.

Now for our project, we need two more datasets in addition to our orders.csv data

Order items will come from third party (say amazon s3)

Customers (will be from azure sql db)

Now we have created a bucket in amazon s3

The screenshot shows the AWS S3 Buckets page. At the top, a green banner displays the message "Successfully created bucket 'project-sales-s3'". Below the banner, there are two tabs: "General purpose buckets" (selected) and "Directory buckets". Under the "General purpose buckets" tab, a table lists one bucket: "project-sales-s3" (Info), located in "US East (Ohio) us-east-2", created on "November 27, 2025, 21:20:01 (UTC-05:00)". To the right of the table are three cards: "Account snapshot" (Info), "External access summary - new" (Info), and "Storage Lens provides visibility into storage usage and activity trends".

File uploaded

The screenshot shows the AWS S3 Upload status page. A green banner at the top indicates "Upload succeeded". Below the banner, the "Upload: status" section shows a summary: "Succeeded" (1 file, 34.1 MB (100.00%)) and "Failed" (0 files, 0 B (0%)). The "Files and folders" tab is selected, displaying a table with one item: "order_items.json" (Info), which is an application/json file of size 34.1 MB and status "Succeeded".

Now I create IAM role to access this file from outside and after that storing those keys in azure key vault

The screenshot shows the Azure Key Vault Secrets page. The left sidebar includes options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Access policies, Resource visualizer, Events, Objects, Keys, and Secrets (selected). The main area shows a table of secrets:

Name	Type	Status
awsSecretKey		Enabled
aws-access-key-ID		Enabled
databricks-access-token		Enabled
sqlserverpassword		Enabled
storage-account-key		Enabled

A message at the top right states: "The secret 'awsSecretKey' has been successfully created."

Now going to ADF and creating a linked service to connect to aws s3

New linked service

 Amazon S3 [Learn more](#) 

Connect via integration runtime * 

AutoResolveIntegrationRuntime 

Authentication type



Access key ID

Azure Key Vault

AKV linked service * 

AzureKeyVault1project_ls  

Secret name * 

aws-access-key-ID 

Edit

Secret version 

Latest version 

Edit

Secret access key

Azure Key Vault

AKV linked service * 

AzureKeyVault1project_ls 

Secret name * 

awsSecretKey 

Edit

Secret version 

Latest version 

Edit

[Add dynamic content \[Alt+Shift+D\]](#)

Service URL 



 Connection successful

[Create](#)

[Back](#)

 [Test connection](#)

[Cancel](#)

Now as soon as orders file arrive in the landing folder, of ADLS Gen 2, we need to also bring the order_items file from s3 to adls gen2

The screenshot shows the Microsoft Data Factory pipeline editor interface. On the left, the 'Factory Resources' sidebar lists 'Pipelines' (1 item), 'Datasets' (1 item), and 'Power Query' (0 items). The 'Datasets' section contains a single item named 'Json1'. The main workspace displays a pipeline named 'pipeline1' with a single step labeled 'Json1'. The 'Properties' pane on the right shows the dataset 'Json1' with its name and a description field. The 'Annotations' pane below it has a 'New' button. The bottom section contains tabs for 'Connection', 'Schema', and 'Parameters', with the 'Connection' tab active. The 'Connection' tab shows settings for a linked service named 'AmazonS3_ls', a file path of 'project-sales:s3/order-items/order_items.json', and encoding options.

Data Factory Validate all Publish all

Factory Resources < Filter resources by name +

Pipelines 1

pipeline1

Change Data Capture (preview) 0

Datasets 1

Json1

Data flows 0

Power Query 0

Properties

General Related (1)

Name * Json1

Description

Annotations + New

Connection Schema Parameters

Linked service * AmazonS3_ls Test connection Edit + New Learn more

File path * project-sales:s3 / order-items / order_items.json Browse | Preview data

Compression type No compression

Encoding Default(UTF-8)

And its working

The screenshot shows the Azure Data Factory interface with a pipeline named 'pipeline1' and a dataset named 'Json1'. The 'Preview data' tab is selected, displaying the contents of the 'order_items.json' file. The data is shown as a JSON array of objects, each representing an order item with fields like id, order_id, product_id, quantity, subtotal, and price.

order_item_id	order_id	product_id	quantity	subtotal	product_price
1	1	957	1	299.98	299.98
2	2	1073	1	199.99	199.99
3	2	502	5	250.00	50.00
4	-	-	-	-	-

Setting up sink destination

The screenshot shows the Azure Storage Explorer interface. The left sidebar has a 'Container' section with 'sales' selected. Below it are 'Overview', 'Diagnose and solve problems', 'Access Control (IAM)', and 'Settings'. The main area shows a table of items in the 'sales' container. The table includes columns for Name, Last modified, Access tier, Blob type, Size, and Lease state. The items listed are:

Name	Last modified	Access tier	Blob type	Size	Lease state
\$_\$azuretmpfolder\$	11/27/2025, 2:48:29 PM				...
discarded	11/27/2025, 9:47:08 AM				...
landing	11/27/2025, 8:59:00 PM				...
order_items	11/27/2025, 9:54:12 PM				...
staging	11/27/2025, 8:59:00 PM				...

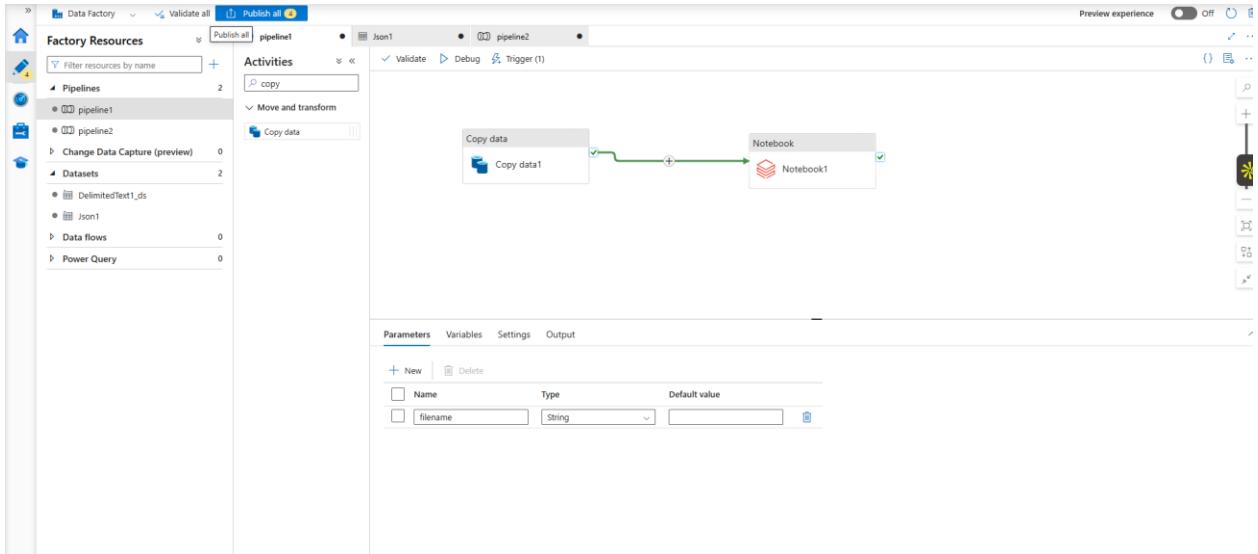
And we want the dataset in the sink (adls gen2) in csv format

Setting up sink

The screenshot shows the Azure Data Factory pipeline editor. On the left, the 'Factory Resources' sidebar lists 'Pipelines', 'Datasets', and 'Data flows'. The main workspace shows a pipeline named 'pipeline1' with a single 'Json1' dataset. Under 'Activities', there is a 'copy' job with a 'Copy data' activity. The properties pane on the right is set to 'Set properties' and shows the following configuration for the sink:

- Name:** DelimitedText1_ds
- Linked service:** AzureDataLakeStorage1/projects
- File path:** sales / order_items / [File name]
- Import schema:**
 - From connection/store
 - From sample file
 - None

Now as soon as a file arrives in the landing folder,



A copy data activity takes place which converts the json from s3 and puts it as csv in the adls gen 2 and after that the notebook is invoked

Now we create customer table in Azure SQL DB (Background activity) which will be our third data source in addition to our orders and order_items data

Creating a new Linked service for it

New linked service

 Azure SQL Database [Learn more](#)

From Azure subscription Enter manually

Azure subscription

Azure subscription 1 (07c3dde0-4c6c-4b0a-8612-8897e0742a2a)

Server name *

projectsqlserveraravind



Database name *

projectsqldb



Authentication type *

SQL authentication

User name *

aravindswamy

Password

Azure Key Vault

AKV linked service * ⓘ

AzureKeyVault1project_ls



Secret name *

sqlserverpassword



Edit

Secret version ⓘ

Latest version



Edit

Always encrypted ⓘ



Encrypt ⓘ

Mandatory

Always encrypted

 Connection successful

 Test connection

Cancel

Create

Back

Sourced customer data:

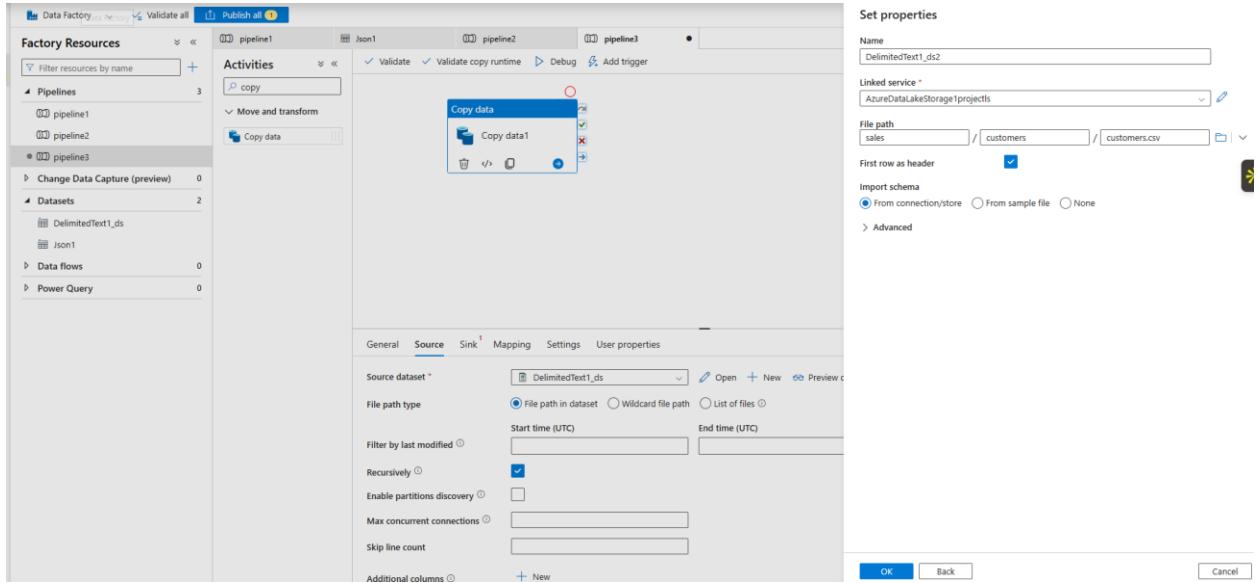
Created customer table structure

```

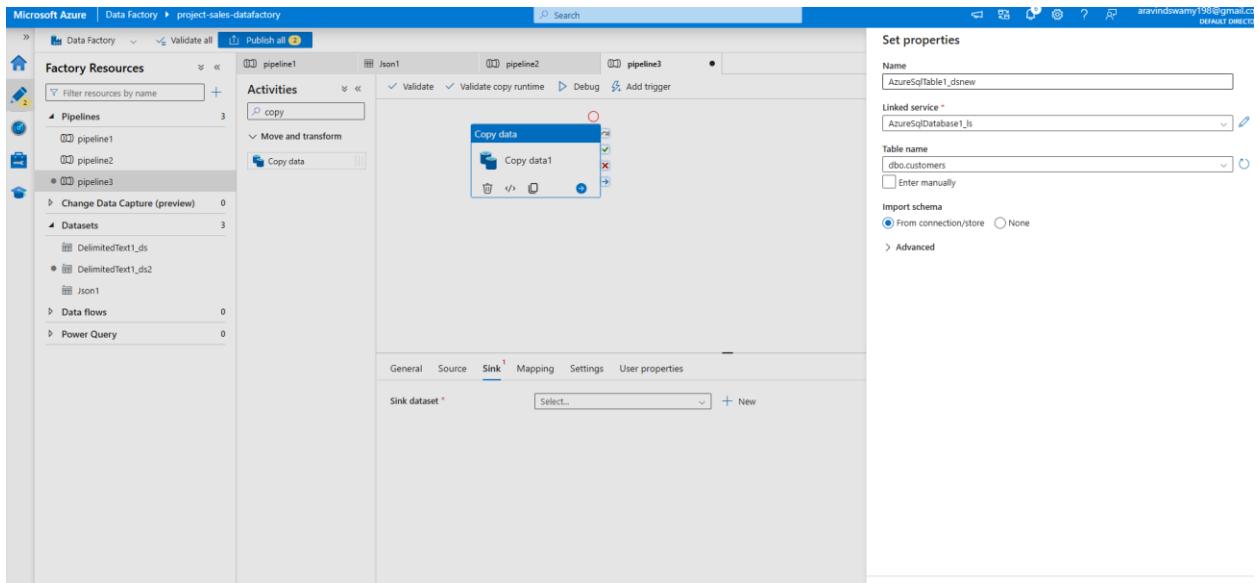
1 CREATE TABLE customers (
2     customer_id INT NOT NULL,
3     customer_fname VARCHAR(45) NOT NULL,
4     customer_lname VARCHAR(45) NOT NULL,
5     customer_email VARCHAR(45) NOT NULL,
6     customer_password VARCHAR(45) NOT NULL,
7     customer_street VARCHAR(255) NOT NULL,
8     customer_city VARCHAR(45) NOT NULL,
9     customer_state VARCHAR(45) NOT NULL,
10    customer_zipcode VARCHAR(45) NOT NULL,
11    PRIMARY KEY (customer_id)
12 );

```

Creating another copy data activity for customers data (background activity)



Now sink,



Running,

The screenshot shows the Microsoft Azure Data Factory interface. On the left, the 'Factory Resources' sidebar lists 'Pipelines' (pipeline1, pipeline2, pipeline3), 'Datasets' (AzureSqlTable1_dsvnew, DelimitedText1_dsv, DelimitedText1_dsv2, Json1), and 'Data flows'. Pipeline3 is selected. In the main workspace, 'Activities' are listed under 'Move and transform' with one item: 'Copy data'. Below the activities, the 'Output' tab is selected for Pipeline run ID 3bf89970-04db-41cd-a4f4-609c3b0d4af6. A table shows the run status: 'Copy data1' succeeded at 11/27/2025, 10:34:49 PM, duration 20s, using AutoResolveIntegrationRuntime (West US 2). The 'Properties' pane on the right shows the pipeline's name and description.

The screenshot shows the Azure SQL | SQL databases page. Under 'Azure SQL Database', 'SQL databases' is selected, showing 'projectsqldb'. The 'Query editor (preview)' tab is active, displaying a query window with the following content:

```
1 select * from customers;
```

The results pane shows a table of customer data:

t	customer_city	customer_state	customer_zipcode
laza	Brownsville	TX	78521
bers Ridge	Littleton	CO	80126
eer Bend	Caguas	PR	00725
imon	San Marcos	CA	92069
Mall	Caguas	PR	00725
ial Promenade	Passaic	NJ	07055
ession	Caguas	PR	00725
4			

Data is populated in the azure sql db

So now we have 3 datasets in picture one is populated in azure db through background activity and Now as soon as orders file arrive in the landing folder, of ADLS Gen 2, we need to also bring the order_items file from s3 to adls gen2, the 3 files are orders, order_items and customers.

Now we are arriving at the final goal,

We have customers data in azure SQL DB (pipeline3 already run)

The screenshot shows the Azure Data Factory pipeline editor interface. On the left, the 'Factory Resources' sidebar lists 'Pipelines' (pipeline1, pipeline2, pipeline3), 'Datasets' (AzureSqlTable1_dnew, DelimitedText1_d1, DelimitedText1_d2, json1), and 'Data flows'. The main workspace displays a pipeline named 'pipeline1' with a single activity: 'Copy data' (Copy data1). The 'Output' tab is selected, showing a table of the run's details:

Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime	User prop...	Activity run ID
Copy data1	Succeeded	Copy data	11/27/2025, 10:34:49 PM	20s	AutoResolveIntegrationRuntime (West US 2)		39838201-6704-46e4-8492-28aa232

At the bottom right of the table, there are links for 'Monitor in Azure Metrics' and 'Export to CSV'.

When we put the orders data in the landing folder, the other main pipeline should bring the order items from s3 to adls gen 2 and then run our notebook to generate our final desired table to which the reporting team can connect

Our main goal, how many orders are placed by each customer and how much amount is spent by each

This final data will be stored in our Azure SQL DB (Our Gold Layer)

So we are going to finally see the entire pipeline in action, both staging and order items are empty so is landing

The screenshot shows the Azure Storage Explorer interface. On the left, the 'sales' container is selected under 'Container'. The main area shows the 'Overview' tab with the following details:

- Search bar: sales
- Action buttons: Add Directory, Upload, Refresh, Delete, Copy, Paste, Rename, Acquire lease, Break lease, Edit columns
- Path: sales > landing
- Authentication method: Access key (Switch to Microsoft Entra user account)
- Search blobs by prefix (case-sensitive):
- Table headers: Name, Last modified, Access tier, Blob type, Size, Lease state
- Data: Showing all 0 items

The screenshot shows the Azure Storage Explorer interface for the 'sales' container. The 'Overview' tab is selected. The 'staging' folder is expanded, showing a single blob named '[]'. The blob type is listed as 'Block blob' and its size is '2.86 MiB'. Other columns include 'Last modified', 'Access tier', and 'Size'.

The screenshot shows the Azure Storage Explorer interface for the 'sales' container. The 'Overview' tab is selected. The 'order_items' folder is expanded, showing a single blob named '[]'. The blob type is listed as 'Block blob' and its size is '2.86 MiB'. Other columns include 'Last modified', 'Access tier', and 'Size'.

Now uploading the data orders_new in the landing

The screenshot shows the Azure Storage Explorer interface for the 'sales' container. The 'Overview' tab is selected. The 'landing' folder is expanded, showing a single blob named 'orders_new.csv'. The blob type is listed as 'Block blob' and its size is '2.86 MiB'. Other columns include 'Last modified', 'Access tier', and 'Lease state'. A filter option 'Only show active objects' is visible.

Triggered

Pipeline runs																		
Triggered		Debug		Rerun		Cancel options		Refresh		Edit columns								
										List								
Filter by run ID or name			Eastern Time (US & C... : Last 24 hours)		Pipeline name : All		Status : All		Runs : Latest runs									
Showing 1 - 14 items																		
Last refreshed 0 minutes ago																		
<input type="checkbox"/>	Pipeline name ↑	Run start ↑	Run end ↑	Duration	Triggered by	Status ↑	Run	Parameters	Annotations	Run ID								
<input type="checkbox"/>	pipeline1	11/27/2025, 11:50:32 PM	--	23s	trigger	In progress	Original			5ef74427-94fa-4d26-afe-c2bede45c51								
<input type="checkbox"/>	pipeline1	11/27/2025, 11:44:50 PM	11/27/2025, 11:46:03 PM	1m 14s	trigger	Failed	Original			167a4a10-f574-451d-997c-c52a0cb...								
<input type="checkbox"/>	pipeline1	11/27/2025, 11:34:44 PM	11/27/2025, 11:35:56 PM	1m 12s	trigger	Succeeded	Original			cd285440-cf8a-4ec2-9739-5aa5092b0...								
<input type="checkbox"/>	pipeline1	11/27/2025, 8:58:38 PM	11/27/2025, 8:59:14 PM	36s	trigger	Succeeded	Original			82e41005-2ab5-4ac6-a731-88d218110...								
<input type="checkbox"/>	pipeline1	11/27/2025, 8:54:32 PM	11/27/2025, 8:54:55 PM	24s	trigger	Failed	Original			d08bb338-919e-49eb-8e74-9eeff8e06ce								
<input type="checkbox"/>	pipeline1	11/27/2025, 8:50:49 PM	11/27/2025, 8:51:40 PM	51s	trigger	Failed	Original			fb511fff-5cdd-4799-84d9-9e1b1fc9449								
<input type="checkbox"/>	pipeline1	11/27/2025, 8:07:58 PM	11/27/2025, 8:09:23 PM	1m 26s	trigger	Succeeded	Original			5a0bd1d5-a0c0-44ef-802b-9da5ce5d5d								
<input type="checkbox"/>	pipeline1	11/27/2025, 6:15:00 PM	11/27/2025, 6:16:10 PM	1m 10s	trigger	Succeeded	Original			24536a5d-d9bb-421f-9cd9-a29e05e6ff								
<input type="checkbox"/>	pipeline1	11/27/2025, 6:11:38 PM	11/27/2025, 6:12:35 PM	57s	trigger	Failed	Original			666c0172-c6c1-48f2-972c-95aa48194c2								
<input checked="" type="checkbox"/>	> pipeline1	11/27/2025, 6:09:28 PM	11/27/2025, 6:10:36 PM	1m 9s	Manual trigger	Failed	Rerun (Latest)			ae071b74-cc4f-4c68-864e-60298391f95								
<input type="checkbox"/>	pipeline1	11/27/2025, 5:33:38 PM	11/27/2025, 5:49:25 PM	15m 47s	trigger	Failed	Original			5205cd94-cb61-476e-87eb-84ec31545								
<input type="checkbox"/>	pipeline1	11/27/2025, 5:16:06 PM	11/27/2025, 5:18:48 PM	2m 43s	trigger	Failed	Original			f20099a5-5a1b-4153-9783-2235c5ce6b9								
<input type="checkbox"/>	pipeline1	11/27/2025, 5:04:40 PM	11/27/2025, 5:11:33 PM	6m 54s	trigger	Failed	Original			75e196ce-8225-4ab9-9c50-3361eb4154								
<input type="checkbox"/>	> pipeline1	11/27/2025, 4:42:17 PM	11/27/2025, 4:55:36 PM	13m 19s	Manual trigger	Failed	Rerun (Latest)			26b0f5cb-44ea-4c7e-a40-e82c4181893								

Running,

All pipeline runs > ● pipeline1 - Activity runs

Activity runs										
All status		Activity name		Activity type		Run start		Duration		Log
Showing 1 - 2 items										Monitor in Azure Metrics Export to CSV
Pipeline run ID: 5ef74427-94fa-4d26-afe-c2bede45c51a										
Activity name ↑	Activity status ↑	Activity type	Run start ↑	Duration	Integration runtime	User properties	Activity run ID			
Notebook1	In progress	Notebook	11/27/2025, 11:50:51 PM	28s			3bc364bb-1251-4dc2-908a-4e0924bdd75f			
Copy data1	Succeeded	Copy data	11/27/2025, 11:50:34 PM	16s	AutoResolveIntegrationRuntime (East US)		b620a507-6504-42f5-bfce-fee77e5338d			

Succeeded

The screenshot shows a Databricks notebook titled "ADF_project-sales-datafactory_pipeline1_Notebook1_3cb364bb-1251-4dc2-908a-4e0924bdd75f run". The notebook contains a code cell with the following PySpark DataFrame output:

```

|customer_id|customer_name|customer_city|customer_state|customer_zipcode|num_orders_placed|total_amount|
|---|---|---|---|---|---|---|
|791|Mary|Smith|Canton|MI|48187|10324.17|
|9371|Mary|Patterson|Meredian|ID|83642|9299.03|
|8766|Mary|Duncan|Caguas|PR|00725|9295.14|
|1657|Betty|Phillips|Caguas|PR|00725|9223.71|
|2641|Betty|Spears|Carrollton|TX|75008|9130.92|
|1288|Evelyn|Thompson|Caguas|PR|00725|9019.11|
|3710|Ashley|Smith|Springfield|MO|65807|9018.1|
|4249|Mary|Butler|Caguas|PR|00725|8918.85|
|5654|Jerry|Smith|Caguas|PR|00725|8904.95|
|5624|Mary|Mata|Caguas|PR|00725|8761.98|
|5715|Kathy|Smith|Caguas|PR|00725|8595.13|
|1464|Amber|Dixon|Caguas|PR|00725|8409.22|
|10235|Joseph|Singh|Caguas|PR|00725|8404.22|
|664|Bobby|Jimenez|Holland|MI|49423|8394.26|
|10351|Teresa|Gray|Caguas|PR|00725|8392.26|

```

The notebook also displays a message: "Put this result again in Azure DB for reporting purpose". To the right, the "Details" panel shows the job ID, task run ID, run status, and duration.

Now checking azure DB (This will be our gold serving layer)

The screenshot shows the Azure SQL Database interface for the "projectssqldb" database. On the left, the "SQL databases" section lists "projectssqldb (projectssqlserveraravind/projectssqldb)". On the right, the "Query editor (preview)" window is open, displaying the following query and results:

```

1 select * From [dbo].[customer_sales_summary];

```

customer_id	customer_name	customer_name	customer_city	customer_state
791	Mary	Smith	Canton	MI
9371	Mary	Patterson	Meredian	ID
8766	Mary	Duncan	Caguas	PR
1657	Betty	Phillips	Caguas	PR
2641	Betty	Spears	Carrollton	TX
1288	Evelyn	Thompson	Caguas	PR
3710	Ashley	Smith	Springfield	MO

Which the reporting team can use to create their report.

Now we can also check the staging and order items folder to see if the data is transferred,

Now landing is empty,

Home > Resource Manager | Resource groups > Azure-project > projectsaaravind | Containers >

sales ...
Container

Search ...

+ Add Directory Upload Refresh | Delete Copy Paste Rename Acquire lease Break lease Edit columns

Overview

sales > landing

Authentication method: Access key ([Switch to Microsoft Entra user account](#))

Search blobs by prefix (case-sensitive) Only show active objects

Showng all 0 items

	Name	Last modified	Access tier	Blob type	Size	Lease state
<input type="checkbox"/>	└── landing					

Staging has orders data now

Home > Resource Manager | Resource groups > Azure-project > projectsaaravind | Containers >

sales Container

Search Add Directory Upload Refresh Delete Copy Paste Rename Acquire lease Break lease Edit columns

Overview

sales > staging

Authentication method: Access key (Switch to Microsoft Entra user account)

Search blobs by prefix (case-sensitive) Only show active objects

Showing all 1 items

Name	Last modified	Access tier	Blob type	Size	Lease state
1.j	11/27/2025, 11:51:14 PM	Hot (Inferred)	Block blob	2.86 MB	Available
orders_new.csv					

And finally

Order items is also transferred from s3 to adls

Thus our pipeline executed perfectly.

Now as a final check lets check what if our data is bad , I am gonna create a dummy order status which wont be in the lookup hence will fail because it's a bad data

Now creating a file with gibberish order status

Now similarly loading it in the landing page it should not go to the staging but should go to discarded

A screenshot of the Azure Storage Explorer interface. The left sidebar shows a tree structure with 'sales' as the selected container. Under 'sales', there is a 'landing' folder which contains a single file named 'l-1'. The main pane displays a table with the following data:

Name	Last modified	Access tier	Blob type	Size	Lease state
l-1	11/28/2025, 12:07:47 AM	Hot (inferred)	Block blob	177 B	Available

Pipeline triggered

A screenshot of the Azure Data Factory Pipeline runs history. The left sidebar shows 'Pipeline runs' as the selected item. The main pane lists 15 pipeline runs for 'pipeline1' over the last 24 hours. The runs are categorized by status: In progress (1), Succeeded (4), Failed (4), and Rerun (Latest) (1). The failed runs are explicitly noted as failing due to 'Bad data'.

Pipeline stops here since its detected bad data

A screenshot of the Microsoft Databricks notebook interface. The left sidebar shows 'Job Runs' as the selected item. The main pane displays the code and logs for the failed pipeline run 'ADF_project-sales-datafactory_pipeline1_Notebook1_7255b9ab-87db-424d-84ec-739a1399ee11 run'. The logs show the following error message:

```

17
if invalidRowDF.count() > 0:
    errorFlag = True
if errorFlag:
    dbutils.fs.rm('/mnt/sales/landing/1', recursive=True)
    dbutils.notebook.exit("error1", "errorMsg", "Invalid order status found")
else:
    dbutils.fs.mv('/mnt/sales/landing/1', '/mnt/sales/staging')

18
Notebook exited: ("error1", "true", "errorMsg": "Invalid order status found")

19

```

The right side of the screen shows the run details, parameters, compute, streaming metrics, and permissions for this specific run.

Data found in discarded now

The screenshot shows the Azure Storage Explorer interface. At the top, there's a navigation bar with 'Home > Resource Manager | Resource groups > Azure-project > projectsaaravind | Containers > sales'. Below the navigation is a toolbar with 'Search', 'Add Directory', 'Upload', 'Refresh', 'Delete', 'Copy', 'Paste', 'Rename', 'Acquire lease', 'Break lease', and 'Edit columns'. On the left, a sidebar shows 'Overview' selected, along with 'Diagnose and solve problems', 'Access Control (IAM)', and 'Settings'. The main area shows a blob container named 'sales' with a single item: 'discarded'. A search bar at the top says 'Search blobs by prefix (case-sensitive)' with 'sales' and 'discarded' entered. Below the search bar, there's a checkbox for 'Only show active objects'. A table lists the item: 'Name' (gibberishorder.csv), 'Last modified' (11/28/2025, 12:08:35 AM), 'Access tier' (Hot (Inferred)), 'Blob type' (Block blob), 'Size' (177 B), and 'Lease state' (Available). There are also three-dot ellipsis icons for more options.

Thus both pipelines are working perfectly

Conclusion

In this project, an end-to-end automated data engineering pipeline was successfully designed and implemented using Microsoft Azure services to simulate a real-world e-commerce data processing system. The solution demonstrated how cloud-native tools can be integrated to build scalable, reliable, and production-ready data pipelines.

The pipeline was built using an event-driven architecture where data ingestion is automatically triggered as soon as new files arrive in Azure Data Lake Storage Gen2. Azure Data Factory orchestrated the workflow, while Azure Databricks executed distributed validation and transformation logic using PySpark. Data quality was ensured through duplicate checks and dynamic business rule validation using a lookup table maintained in Azure SQL Database.

The project was further enhanced by introducing parameterized pipelines, generic mount logic, and secure secret management using Azure Key Vault. Cross-cloud integration was also implemented by ingesting order item data from Amazon S3, and customer data from Azure SQL Database, thereby simulating realistic multi-source enterprise data workflows.

Finally, all datasets were joined to generate analytics-ready outputs, which were stored in Azure SQL Database for reporting purposes. This project provided strong hands-on experience in building scalable, secure, and reusable data pipelines, and demonstrated key data engineering concepts such as automation, data validation, cloud integration, and Medallion Architecture. The overall solution closely reflects real-world industry data engineering practices.