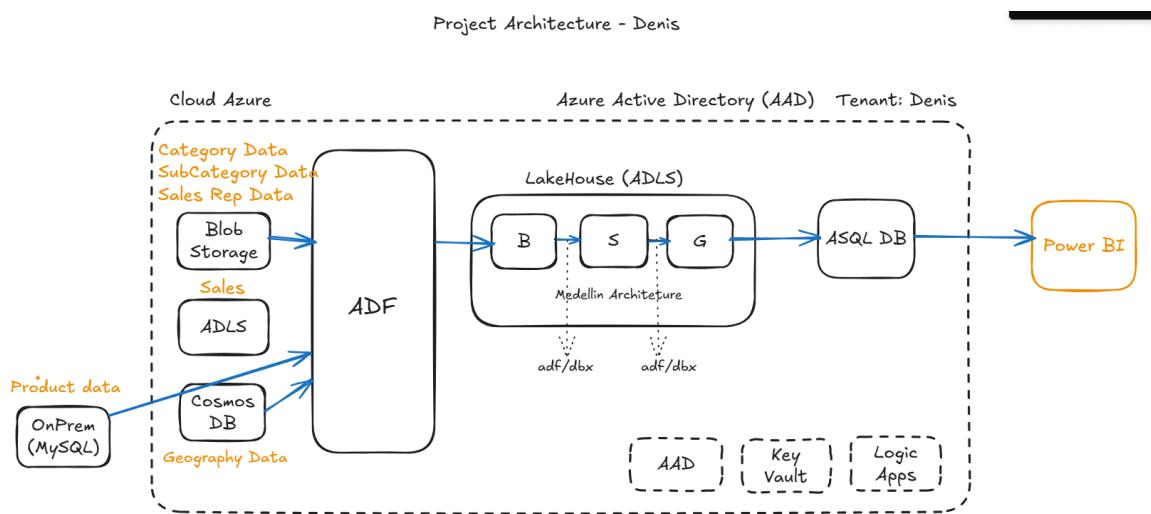


Project Objective: To gather data from various suppliers (On-Prem MySQL, Blob/ADLS, Cosmos DB), process it through the factory manager (Azure Data Factory) and assembly line (Lakehouse – Bronze, Silver, Gold), and store the finished products in the warehouse (Azure SQL DB) for easy access and further use.

Ingest Data from Multiple Sources → Transform with ADF Data Flows → Store in Gold Layer & Azure SQL Database

Project Architecture:



Required Steps :-

Step 1 :- Sign in to Azure Portal

1.1-Go to <https://portal.azure.com>

1.2-Sign in with your Microsoft/Azure account

Step 2: Create a Resource Group

A resource group is like a folder to keep related Azure resources (e.g., storage, databases) organized together.

- **2.1 How to Create:**
 1. In the left-hand menu, click “Resource groups”
 2. Click “+ Create”
- **2.2 Fill in Basic:**
 1. **Subscription:** Choose your Azure
 2. **Resource group name:** Denise-RG
 3. **Region:** Select your nearest

- Click Review + Create, then click Create

Microsoft Azure Upgrade

Home > Create a resource > Marketplace > Resource group >

Create a resource group

Basics Tags Review + create

[Automation Link](#)

Basics

Subscription	Azure subscription 1
Resource group name	Denise-rg
Region	East US

Tags

None

Previous Next Create

- You've now created a resource group!

Microsoft Azure Upgrade Search resources, services, and docs (G+/-) Copilot

Home > **Denise-rg** Resource group ... What are the best practices for managing this resource group? Are there any alerts fired for this resource group? How do I troubleshoot issues with this resource group?

Overview

Essentials

Subscription (move) : [Azure subscription 1](#) Deployments : [No deployments](#)
Subscription ID : d75f7ccf-48d9-4b76-8ee1-42d2edcb220e Location : East US
Tags (edit) : [Add tags](#)

Resources **Recommendations**

Filter for any field... Type equals all × Location equals all × Add filter

Showing 0 to 0 of 0 records. Show hidden types ⓘ

Name	Type	Location
------	------	----------

No grouping No grouping

No resources match your filters
Try changing or clearing your filters.

+ Create resources Clear filters Learn more

Add or remove favorites by pressing **Ctrl+Shift+F**

3. Source and Target Setup:-

3.1 Azure Blob Storage:-

- Blob storage lives inside something called a storage account — like a big folder in the cloud(without hierarchical)
 1. **Container Name:** e.g., raw-data
 2. **Data Format:** CSV / JSON / Parquet
 3. **Storage Account Name:** denisstroageblob
 4. **Click "Create a resource"**
 5. Search for "**Storage account**" and click Create

Click Review + Create, then Create Reference:

Create a storage account

...

[Basics](#) [Advanced](#) [Networking](#) [Data protection](#) [Encryption](#) [Tags](#) [Review + create](#)

[View automation template](#)

Basics

Subscription	Azure subscription 1
Resource group	Denis-rg
Location	East US
Storage account name	denisstorageblob
Primary service	
Performance	Standard
Replication	Locally-redundant storage (LRS)

Advanced

Enable hierarchical namespace	Disabled
Enable SFTP	Disabled
Enable network file system v3	Disabled
Allow cross-tenant replication	Disabled
Access tier	Hot
Enable large file shares	Enabled

Security

Secure transfer	Enabled
-----------------	---------

[Previous](#) [Next](#) [Create](#)

- Your storage account is now created!

Home >  denisstroageblob_1754803048377 | Overview ✖️ ⋮

Deployment

 Search ✖️ ⋮ Delete  Cancel  Redeploy  Download  Refresh 

 Overview ✖️ ⋮

 Inputs

 Outputs

 Template

 Your deployment is complete

 Deployment name: denisstroageblob_1754803048377
Subscription: Azure subscription 1
Resource group: Denis-rg

Start time: 8/10/2025, 10:51:57 AM
Correlation ID: 1d6cc4e1-6b1a-4ffc-8226-917f3f85003b 

 Deployment details

 Next steps

[Go to resource](#)

[Give feedback](#)

 Tell us about your experience with deployment

3.2 Azure Blob Storage:-

- Blob storage lives inside something called a storage account — like a big folder in the cloud (with hierarchical)
 1. **Container Name:** e.g., raw-data
 2. **Data Format:** CSV / JSON / Parquet
 3. **Storage Account Name:** denisstroageadls
 4. **Click "Create a resource"**
 5. Search for "**Storage account**" and click Create

Click Review + Create, then Create Reference:

[View automation template](#)

Basics

Subscription	Azure subscription 1
Resource group	Denis-rg
Location	East US
Storage account name	denisstroageadls
Primary service	Azure Blob Storage or Azure Data Lake Storage Gen 2
Performance	Standard
Replication	Locally-redundant storage (LRS)

Advanced

Enable hierarchical namespace	Enabled
Enable SFTP	Disabled
Enable network file system v3	Disabled
Allow cross-tenant replication	Disabled
Access tier	Hot
Enable large file shares	Enabled

Security

Secure transfer	Enabled
-----------------	---------

[Previous](#) [Next](#) **Create**

- Your storage account is now created!

[Home](#) / [denisstroageadls_1754803890236](#) | Overview [...](#)

 [denisstroageadls_1754803890236](#) | Overview Deployment

[Search](#) [Delete](#) [Cancel](#) [Redeploy](#) [Download](#) [Refresh](#)

Overview

 Your deployment is complete

Deployment name: denisstroageadls_1754803890236
Subscription: Azure subscription 1
Resource group: Denis-rg

Start time: 8/10/2025, 11:03:29 AM
Correlation ID: 2bf7101a-b819-4e03-94a8-c769c31c72e1 [Copy](#)

 Deployment details
 Next steps

Go to resource

[Give feedback](#)
[Tell us about your experience with deployment](#)

4.Create a Blob Container:-

4.1 Denisstroageblob:-

- Containers are like folders inside the storage account to organize files.

The screenshot shows the Azure Storage browser interface for the 'denisstroageblob' storage account. On the left, the navigation menu includes 'Overview', 'Activity log', 'Tags', 'Diagnose and solve problems', 'Access Control (IAM)', 'Data migration', 'Events', 'Storage browser' (which is selected), 'Storage Mover', 'Partner solutions', 'Resource visualizer', 'Data storage', and 'Containers'. The main content area displays 'Storage account metrics' with a note about data being updated every 2-4 hours. It shows four sections: 'Blob containers' (1 container, 3 blobs, 18.24 kB total), 'File shares' (0 shares, 0 files, 0 total data stored), 'Tables' (5 tables, 0 entities, 374 B total), and 'Queues' (0 queues, 0 messages, 0 total data stored). There are also 'Favorites' and 'Recently viewed' sections.

- Click "+ Stroage browser" and to the Blob container
- Give it a name : Denisraw

The screenshot shows the 'Blob containers' section of the Azure Storage browser. The left sidebar shows 'denisstroageblob' with 'Containers' selected. Under 'Containers', there are entries for '\$logs' and 'denisraw'. A search bar at the top right is empty. Below the search bar, a table lists items with columns for 'Name' and 'Type'. The items listed are '\$logs' and 'denisraw'.

- Click Create
- Now upload data into the denisraw (Categories.csv, SalesRep.xlsx, SubCategories.csv)

The screenshot shows the contents of the 'denisraw' blob container. The left sidebar shows 'denisstroageblob' with 'Containers' selected. Under 'Containers', there is an entry for 'denisraw'. The main content area shows a table with three items: 'Categories.csv', 'SalesRep.xlsx', and 'SubCategories.csv'. Each item has a 'Last modified' timestamp of 8/10/2025, 11:23:52 AM and an 'Access tier' of 'Hot (Inferred)'.

4.2 denisstroageadls:-

- Containers are like folders inside the storage account to organize files.

The screenshot shows the Azure Storage browser interface for the 'denisstroageadls' storage account. The left sidebar is identical to the previous screenshot, showing 'denisstroageblob' with 'Containers' selected. The main content area displays 'Storage account metrics' with a note about data being updated every 2-4 hours. It shows four sections: 'Blob containers' (0 containers, 0 blobs, 0 total data stored), 'File shares' (0 shares, 0 files, 0 total data stored), 'Tables' (0 tables, 0 entities, 0 total data stored), and 'Queues' (0 queues, 0 messages, 0 total data stored). There are also 'Favorites' and 'Recently viewed' sections.

- Click "+ Storage browser" and to the Blob container
- Give it a name : Denis-medallian

The screenshot shows the Azure Storage Explorer interface. On the left, a sidebar lists 'denisstorageadls' under 'Favorites', 'Recently viewed' (empty), and 'Blob containers' (selected). Below 'Blob containers' are 'File shares', 'Queues', and 'Tables'. On the right, the main pane displays 'Blob containers' with a search bar 'Search containers by prefix'. It shows two items: '\$logs' and 'denis-medallian'.

- Create the following directories:
 - bronze
 - silver
 - gold

The screenshot shows the contents of the 'denis-medallian' blob container. The left sidebar shows 'denisstorageadls' and 'File shares'. The main pane shows three items: 'Bronze', 'Gold', and 'Silver'. A search bar 'Search blobs by prefix (case-sensitive)' is at the top right. The table below lists the items:

Name	Last modified
Bronze	8/10/2025, 11:39:34 AM
Gold	8/10/2025, 11:39:50 AM
Silver	8/10/2025, 11:38:59 AM

5 Azure Data Factory:-

- Azure Data Factory (ADF) is a cloud-based ETL (Extract, Transform, Load) service that helps you move and transform data between different sources and destination
 - **Data Factory Name:** denisfactoryadf
 - **Version:** V2 (latest)
 - **Region:** Choose a location close to your data and users
 - **Click "Create a resource"**
 - **Search for "Data Factory" and click Create**

HOME > Create a resource > marketplace > Data factory >

Create Data Factory ...

Basics Git configuration Networking Advanced Tags [Review + create](#)

[View automation template](#)

Basics

Subscription	Azure subscription 1
Resource group	Denis-rg
Name	denisfactoryadf
Region	East US
Version	V2

Networking

Connect via	Public endpoint
-------------	-----------------

[Previous](#) [Next](#) [Create](#)

<https://portal.azure.com/#>

Review + Create:

- Check all details and click **Create** to deploy the Data Factory.

The screenshot shows the Azure Data Factory Studio interface. At the top, there's a navigation bar with tabs like 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Diagnose and solve problems', 'Resource visualizer', 'Settings', 'Getting started', 'Monitoring', 'Automation', and 'Help'. Below the navigation bar, the 'Overview' tab is selected, displaying basic information about the Data Factory:

- Resource group: Denis-rg
- Status: Succeeded
- Location: East US
- Subscription: Azure subscription 1
- Subscription ID: d75f7ccf-48d9-4b76-8ee1-42d2edcb220e

On the right side, there's a large blue icon of a factory building. Below the icon, it says 'Azure Data Factory Studio' and has a 'Launch studio' button.

6. Azure SQL Database :

- Azure SQL Database is a fully managed cloud database service that stores structured data and supports T-SQL queries.

- **Database Name:** denis_db
- **Server Name:** denis-sql-server
- **Compute + Storage:** Basic/Standard/Premium (choose based on needs)
- **Click "Create a resource"**
- **Search for "SQL Database" and click Create**

The screenshot shows the Azure portal's 'denis_db' SQL database overview page. At the top, there's a navigation bar with tabs like 'Overview', 'Copy', 'Restore', 'Export', 'Set server firewall', 'Delete', 'Connect with...', and 'Feedback'. Below the navigation bar, the 'Overview' tab is selected, displaying basic information about the database:

- Resource group: Denis-rg
- Status: Online
- Location: West US 2
- Subscription: Azure subscription 1
- Subscription ID: d75f7ccf-48d9-4b76-8ee1-42d2edcb220e
- Tags: Add tags

On the right side, there's a table with columns for 'Server name', 'Elastic pool', 'Connection strings', 'Pricing tier', and 'Earliest restore point'. The values listed are: Server name: denis-sql-server.database.windows.net, Elastic pool: No elastic pool, Connection strings: Show database connection strings, Pricing tier: Basic, and Earliest restore point: No restore point available.

Below the table, there are links for 'Getting started', 'Monitoring', 'Properties', 'Features', 'Notifications (0)', 'Integrations', and 'Tutorials'.

7. Azure Cosmos DB :-

- Azure Cosmos DB is a globally distributed, multi-model NoSQL database service designed for high performance and scalability.
- **Account Name:** e.g., denis-cosmos
- **Workload Type:** Learning
- **API Type:** Core (SQL) – for document data in JSON format
- **Capacity Mode:** Provisioned throughput (or Serverless for testing)
- Click "Create a resource"
- Search for "Azure Cosmos DB" and click Create

The screenshot shows the Azure portal's Overview page for a Cosmos DB account. The account name is 'danisnosql'. Key details include:

- Status: Online
- Resource group: 'Denis-rg'
- Subscription ID: '075f7ccf-49d9-4b76-8ee1-42d2edcb220e'
- Total throughput limit: 1000 RU/s
- Read Locations: West US 2
- Write Locations: West US 2
- URI: https://danisnosql.mongo.cosmos.azure.com:443/
- Server Version: 7.0
- Free Tier Discount: Opted In

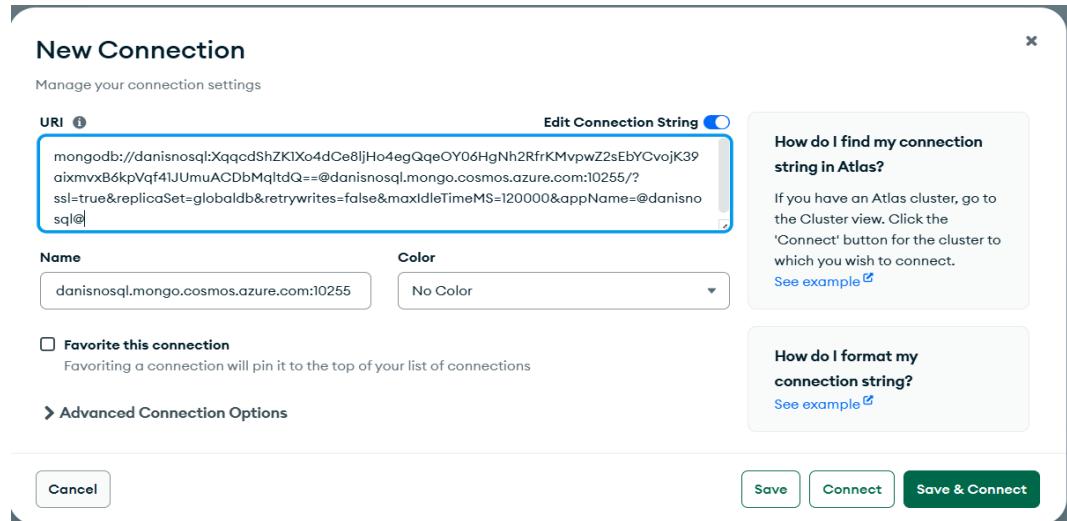
Monitoring section shows data for the last 24 hours. Connection strings section lists Primary and Secondary connection strings.

- **Connecting Cosmos DB (MongoDB API) to MongoDB Compass**

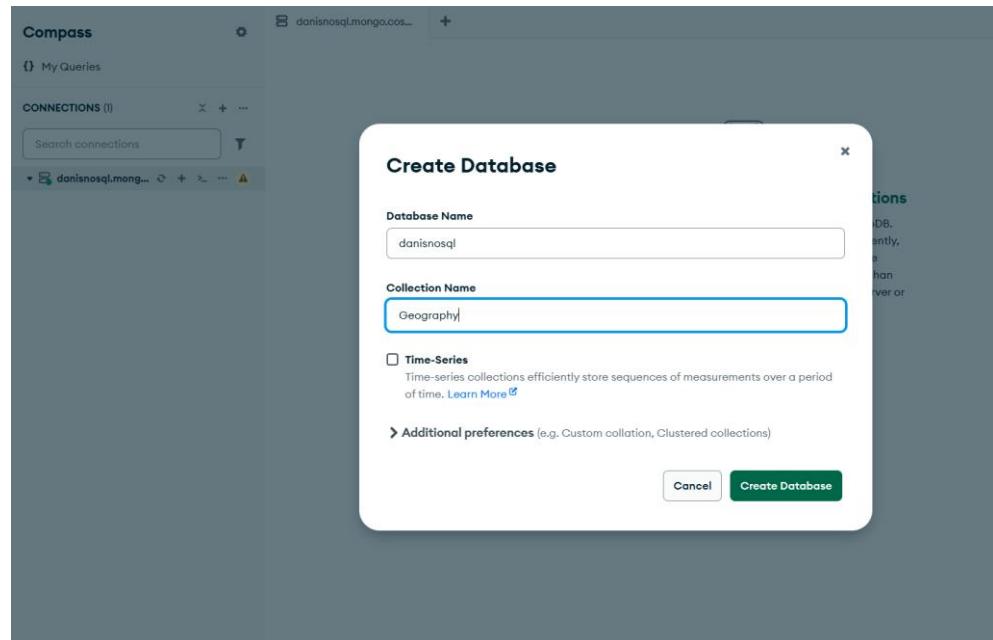
The screenshot shows the Connection strings page for the 'danisnosql' account. The primary connection string is displayed as follows:

```
mongodbs://(danisnosqlXqqcdShZK1Xo4dCe8jHo4egQqeOY06HgNhRfrKMvpwZ2sEDYCvojk39aximvxB6kpVqf41JlmuACDbMqtdQ==@danisnosql.mongo.cosmos.azure.com:10255/?ssl=true&replicaSet=globaldb&retrywr...
```

- In the Azure Portal, open your Cosmos DB account
- In the left menu, click settings in that press connecting strings.
- Under Connection String, copy the Primary Connection String (this string already contains the authentication details).
- Open MongoDB Compass.
- In the “Paste your connection string (SRV or Standard)” field, paste the copied connection string directly.



- Click **Connect**.
- Once connected, click **Create Database**.



- Enter:
 - **Database Name:** danisnosql
 - **Collection Name:** Geography
- Click **Create Database**.

✓ You now have a new database called **danisnosql** in Cosmos DB (MongoDB API), accessible via MongoDB Compass.

- Open the geography collection.
- Click **ADD DATA → Import File**.
- Select the **Geography.json** file from your system.
- Choose **JSON** as the file type and click **Import**.

The screenshot shows the Compass MongoDB interface. On the left, the 'Connections' sidebar lists a single connection named 'danisnosql'. Underneath it, a 'Geography' collection is selected. The main panel displays the 'Documents' tab with a message stating 'This collection has no data'. A green button at the bottom right labeled 'Import data' is visible.

The screenshot shows the same Compass MongoDB interface after data has been imported. The 'Documents' tab now lists four entries, each representing a city with its country, town, and Wikipedia link. A success message at the bottom left indicates 'Import completed. 7 documents imported.'

_id	Country	Town	Wikipedia
<code>ObjectId('68986a597c2d138f14344c9b')</code>	"Czech republic"	"Prague"	" https://en.wikipedia.org/wiki/Prague "
<code>ObjectId('68986a597c2d138f14344c9c')</code>	"Denmark"	"Copenhagen"	" https://en.wikipedia.org/wiki/Copenhagen "
<code>ObjectId('68986a597c2d138f14344c9d')</code>	"Germany"	"Frankfurt"	" https://en.wikipedia.org/wiki/Frankfurt "
<code>ObjectId('68986a597c2d138f14344c9e')</code>	"Germany"	"Berlin"	" https://en.wikipedia.org/wiki/Berlin "
<code>ObjectId('68986a597c2d138f14344c9f')</code>	"Germany"	"Dresden"	" https://en.wikipedia.org/wiki/Dresden "

- The danisnosql database now contains a geography collection populated from the Geography JSON file.

Creating Linked Services in Azure Data Factory :-

1. Create Linked Service for Azure Blob Storage :-

1. Open your Data Factory (denisfactoryadf) in the Azure Portal.
2. In the left menu, click Manage → Linked services.
3. Click + New.
4. Search for Azure Blob Storage
5. Select it and click Continue.
6. Fill in:
7. Name: blod2adf_ls
8. Authentication type: Account key

9. Storage account name: Select your storage account (denisstorageblob)
10. Click Test connection → Create.

The screenshot shows the 'Linked services' blade in the Azure Data Factory portal. On the left, there's a navigation menu with options like General, Connections, and Author. Under 'Connections', 'Linked services' is selected. A search bar at the top right says 'Filter by name' and 'Annotations: Any'. Below it, there's a 'Create linked service' button. The main area is titled 'New linked service' for 'Azure Blob Storage'. It has fields for 'Name' (set to 'blob2adf_ls'), 'Description', 'Connect via integration runtime' (set to 'AutoResolveIntegrationRuntime'), 'Authentication type' (set to 'Account key'), 'Account selection method' (radio button selected for 'From Azure subscription'), 'Azure subscription' (set to 'Azure subscription 1 (d75f7ccf-48d9-4b76-8ee1-42d2edcb220e)'), 'Storage account name' (set to 'denisstorageblob'), and 'Additional connection properties'. At the bottom, there are 'Create', 'Back', 'Test connection', and 'Cancel' buttons. A success message 'Connection successful' is visible next to the 'Test connection' button.

2. Create Linked Service for ADLS:-

1. Open your Data Factory (denisfactoryadls) in the Azure Portal.
2. In the left menu, click Manage → Linked services.
3. Click + New.
4. Search for Azure Data Lake Storage Gen2 (hierarchical)
5. Select it and click Continue.
6. Fill in:
7. Name: adls2adf
8. Authentication type: Account key
9. Storage account name: Select your storage account (denisstorageadls).

10. Click Test connection → Create.

The screenshot shows the 'Linked services' blade in the Azure Data Factory portal. The left navigation menu is identical to the previous screenshot. The main area shows a table of existing linked services, with one entry for 'blob2adf_ls' of type 'Azure Blob Storage'. Below the table, there's a 'Create linked service' button. The right side of the screen shows the 'New linked service' configuration for 'Azure Data Lake Storage Gen2'. It has fields for 'Name' (set to 'adls2adls'), 'Description', 'Connect via integration runtime' (set to 'AutoResolveIntegrationRuntime'), 'Authentication type' (set to 'Account key'), 'Account selection method' (radio button selected for 'From Azure subscription'), 'Azure subscription' (set to 'Azure subscription 1 (d75f7ccf-48d9-4b76-8ee1-42d2edcb220e)'), 'Storage account name' (set to 'denisstorageadls'), and 'Test connection' (radio button selected for 'To linked service'). At the bottom, there are 'Create', 'Back', 'Test connection', and 'Cancel' buttons. A success message 'Connection successful' is visible next to the 'Test connection' button.

3. Create Linked Service for Azure Cosmos DB (MongoDB API):-

1. Click + New.
2. In the search bar, type Cosmos.
3. Select Azure Cosmos DB (MongoDB API) and click Continue.
4. Fill in the details:
 - 5. Name: CosmosDb2adf_ls
 - 6. Database name: danisnosql
7. Click Test connection to ensure it works.
8. Click Create to save the linked service.

New linked service

Azure Cosmos DB for MongoDB Learn more

Name *
CosmosDb2adf

Description

Connect via integration runtime *
AutoResolveIntegrationRuntime

Connection string Azure Key Vault

Account selection method
From Azure subscription Enter manually

Azure subscription Select all

Azure Cosmos DB account name *
danisnosql

Database name *
danisnosql

Annotations

Create Back

Connection successful
Test connection Cancel

4. Create Linked Service for On-Prem MySQL:-

1. + New in Linked Services.
2. Search for MySQL → Continue.
3. Fill in:
4. Name: OnPrem2adf
5. Connect via integration runtime:-As Self-Hosted:-

Step – Install Self-Hosted Integration Runtime:-

1. Name it: SelfHosted and click Create.
2. Once created, click Download to get the Self-Hosted Integration Runtime installer.
3. Install it on a machine that can access your On-Prem MySQL server.
4. During installation, it will ask for a key — copy the Authentication Key from ADF and paste it into the installer.
5. Finish installation and ensure the IR status in ADF shows Running.

Integration runtime setup

Private network support is realized by installing integration runtime to machines in the same on-premises network/VNET as the resource the integration runtime is connecting to. Follow below steps to register and install integration runtime on your self-hosted machines.

Name * ⓘ
[IntegrationRuntime1]

Description
Enter description here...

Type
Self-Hosted

[Create](#) [Back](#) [Cancel](#)

Integration runtime setup

[Settings](#) [Nodes](#) [Auto update](#) [Sharing](#) [Links](#)

Install integration runtime on Windows machine or add further nodes using the Authentication Key.

Name ⓘ

integrationRuntime1

Self-contained interactive authoring ⓘ

Disable Enable

Option 1: Express setup

[Click here to launch the express setup for this computer](#)

Option 2: Manual setup

Step 1: [Download and install integration runtime](#)

Step 2: Use this key to register your integration runtime

Name Authentication key

Key1 IR@9c97256b-2b4a-403d-ae7b-f108e1fe5ccf@denisfactoryadf@Service

Key2 IR@9c97256b-2b4a-403d-ae7b-f108e1fe5ccf@denisfactoryadf@Service

[Close](#)

Microsoft Azure | Data Factory > denisfactoryadf

Would you like to see Data Factory inside of Microsoft Fabric, Microsoft's newest cloud-first data analytics SaaS platform? Click [here](#) to get started with Microsoft Integration Runtime Configuration Manager

Register Integration Runtime (Self-hosted)

Welcome to Microsoft Integration Runtime Configuration Manager. Before you start, register your Integration Runtime (Self-hosted) node using a valid Authentication Key.

IR@9c97256b-2b4a-403d-ae7b-f108e1fe5ccf@denisfactoryadf@ServiceEndpoint=denisfactoryadf.eastus.datafactory.azure.net@+tmSRb+zBFNShhD7N +zb/f7xhZ4eBKW3mhM2d1Hdylk=

Show Authentication Key [Learn how to find the Authentication Key](#)

HTTP Proxy

Current Proxy: No proxy [Change](#)

Diagnostic Tool

Troubleshoot problems (preview)

Data flow libraries

Security

Credentials

Customer managed key

[Register](#) [Cancel](#)

Integration runtime setup

[Settings](#) [Nodes](#) [Auto update](#) [Sharing](#) [Links](#)

Install integration runtime on Windows machine or add further nodes using the Authentication Key.

Name ⓘ

integrationRuntime1

Self-contained interactive authoring ⓘ

Disable Enable

Option 1: Express setup

[Click here to launch the express setup for this computer](#)

Option 2: Manual setup

Step 1: [Download and install integration runtime](#)

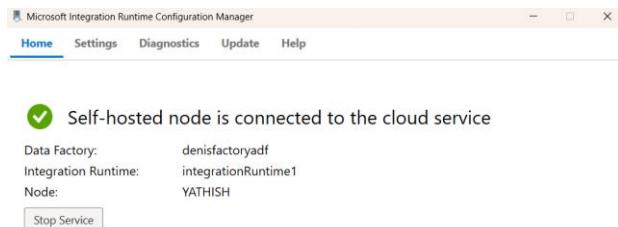
Step 2: Use this key to register your integration runtime

Name Authentication key

Key1 IR@9c97256b-2b4a-403d-ae7b-f108e1fe5ccf@denisfactoryadf@Service

Key2 IR@9c97256b-2b4a-403d-ae7b-f108e1fe5ccf@denisfactoryadf@Service

[Close](#)



Connected to the cloud service (Data Factory V2)

After completed the setup now fill the details

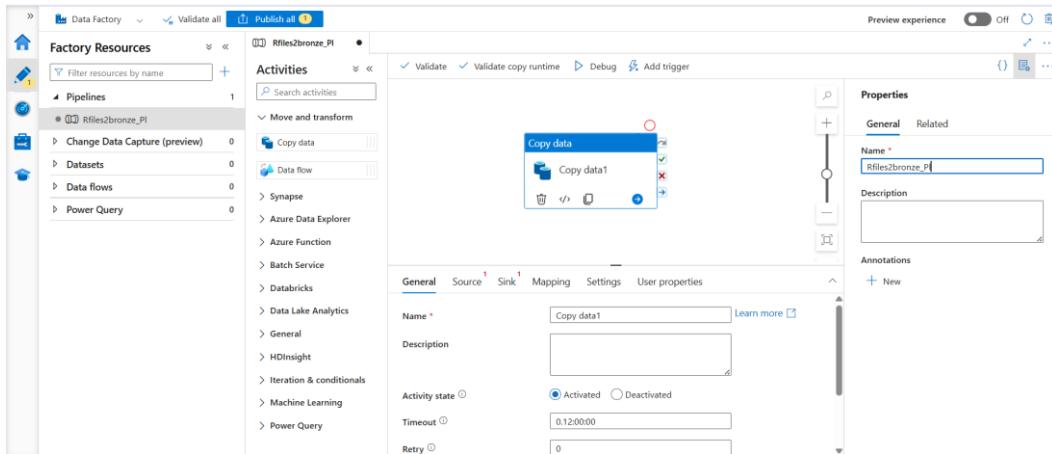
The screenshot shows the 'Linked services' configuration page in Microsoft Azure Data Factory. The left sidebar shows 'General', 'Connections', and 'Linked services' selected. The main area shows three existing linked services: 'adls2adaf_ls' (Azure Data Lake Storage), 'blob2adaf_ls' (Azure Blob Storage), and 'CosmosDb2adaf' (Azure Cosmos DB). On the right, a new linked service is being configured for MySQL. The 'Name' field is set to 'OnPrem2adf'. The 'Connect via integration runtime' dropdown is set to 'integrationRuntime1'. The 'Server name' is 'localhost', 'Port' is '3306', 'Database name' is 'denis_db', and 'User name' is 'root'. The 'Password' field has 'Azure Key Vault' selected. A success message at the bottom right says 'Connection successful'.

6. **Server name :-localhost**
7. **Port:-3306**
8. **Database name :- denise_db**
9. **Username :- root**
10. **Password :- ******
11. **Test connection → Create.**

The screenshot shows the 'Linked services' configuration page in Microsoft Azure Data Factory. The left sidebar shows 'General', 'Connections', and 'Linked services' selected. The main area shows four linked services: 'adls2adaf_ls', 'blob2adaf_ls', 'CosmosDb2adaf', and 'OnPrem2adf'. A success message at the top right says 'Successfully created' and 'Successfully created OnPrem2adf (Linked service)'.

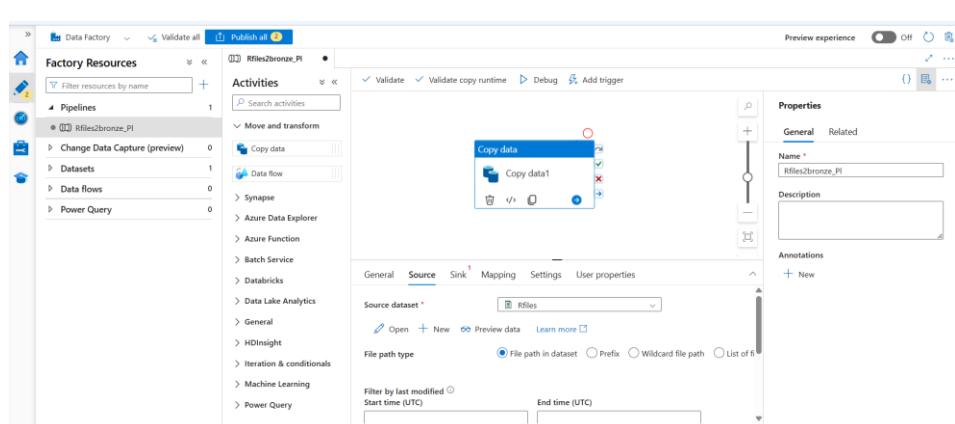
1. Creating the Rfiles2bronze_Pl Pipeline:-

1. Go to Azure Data Factory → Author tab.
2. Create new pipeline → Name it: **Rfiles2bronze_Pl**.
3. Drag a Copy Data activity into the canvas.



Source Settings

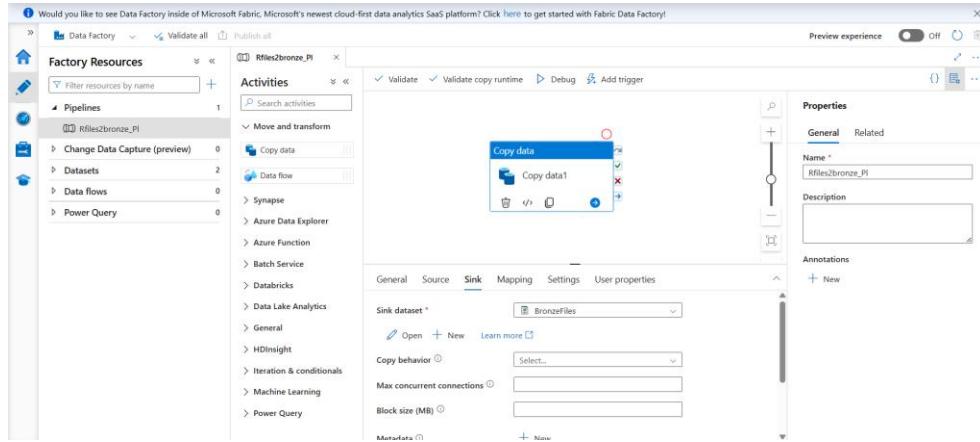
- Dataset type → Azure Blob Storage.
- Linked Service → your Blob linked service(**blob2adf_ls**)
- Container → Rfiles.
- File format → CSV.
- First row as header → checked.
- Wildcard path (optional) if you want to copy all CSVs in the folder: *.csv.



Sink Settings

- Dataset type → Azure Data Lake Storage Gen2.
- Linked Service → your ADLS linked service.
- Container → bronzefiles.

1. Path → folder name we created earlier for Bronze stage (e.g., denis-medallian/bronze/).
2. Copy behavior → *Preserve hierarchy* (if you want same folder structure as source)
or *Flatten hierarchy* (if you just want all files in one place).



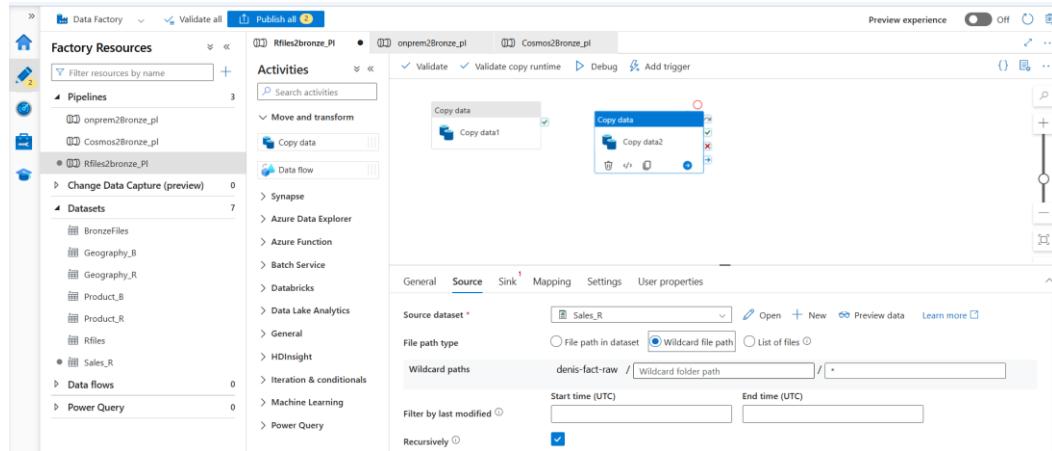
Final Steps:- Once successful → Publish All.

In same pipeline we doing another activate also

1.1ADLS2bronze_Pl

Source

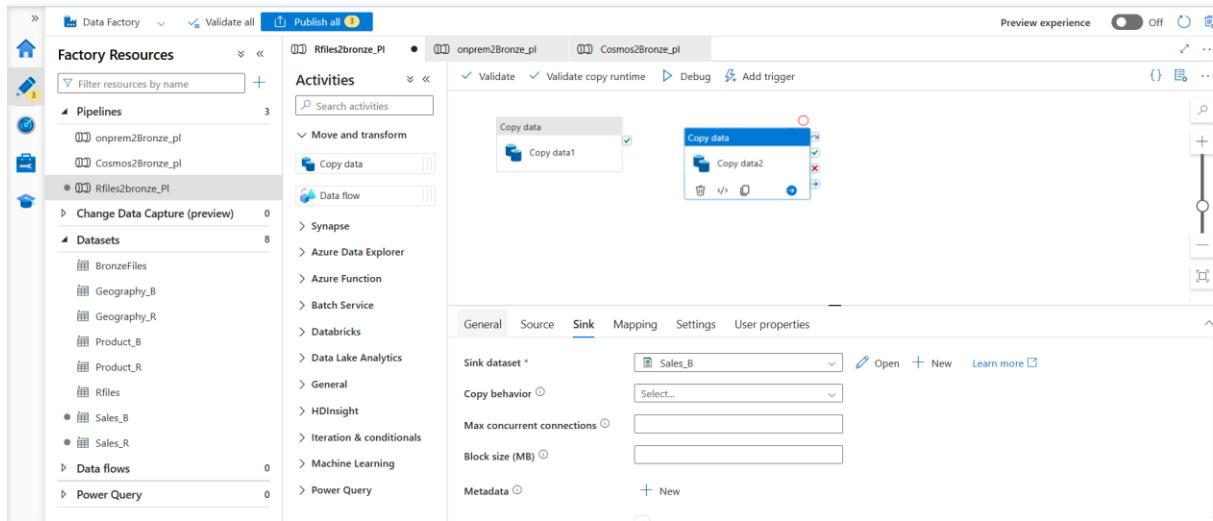
- Dataset: **Azure Data Lake Storage Gen2**.
- File format: Select CSV / JSON / Parquet depending on your file type.
- Linked Service: **adls2adf_ls**
- Path: *wherever your source files are*
- If multiple files: Enable "Wildcard file path" and use something like:



Sink

- Dataset: **Azure Data Lake Storage Gen2**.
- Linked Service: **Same ADLS linked service**.
- Path: */ denis-medallianbronze/Sales/(target folder)*.
- File format: Same as source (ADF will copy without changing type unless mapping says otherwise).

- If you want to preserve file names → enable "Preserve hierarchy" option in copy activity.

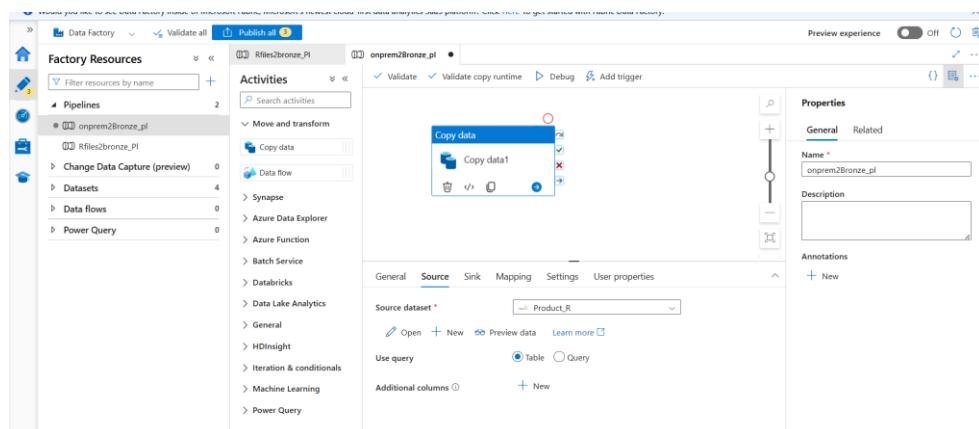


2.Pipeline: MySQL2bronze_Pl :-

- Go to Azure Data Factory → Author → New Pipeline.
Name → MySQL2bronze_Pl.
- Add a Copy Data activity

Source Settings :-

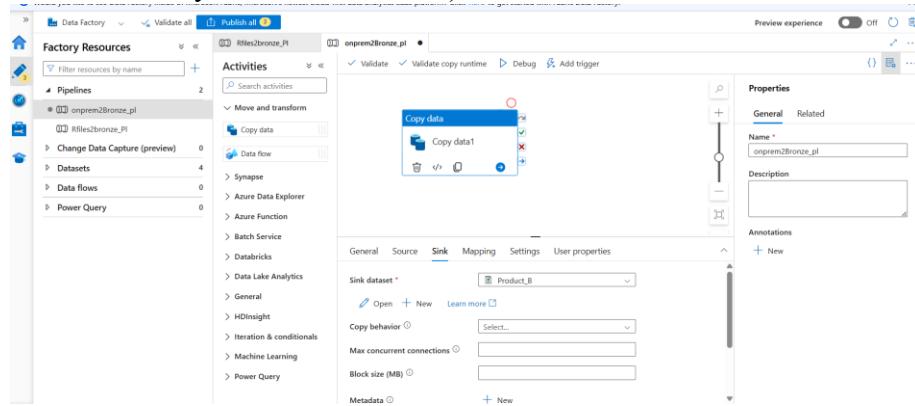
- Dataset type → MySQL(local machine).
- Linked Service → OnPrem2adf (We created)
- Table → Select the table you want to copy from MySQL.



Sink Settings

- Dataset type → Azure Data Lake Storage Gen2.
- File format → Parquet or CSV (Parquet is recommended for later transformations).
- Linked Service → adls2bronze
- Container → bronzefiles.

- Path → e.g., denis-medallian/bronze/ **Product.csv** (so you don't overwrite your blob CSV data).



Extra Note for On-Prem:- Since this is **on-prem MySQL**, you'll need:

- Self-hosted Integration Runtime (SHIR)** installed and running.
 - SHIR linked to your ()linked service.
- Publish All.**

3.Pipeline: Cosmos2bronze_Pl

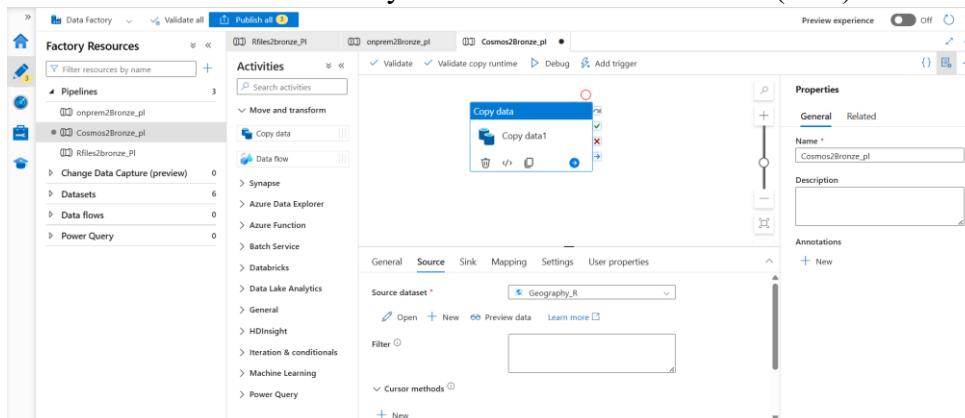
- Go to Azure Data Factory → Author → New Pipeline.

Name → Cosmos2bronze_Pl.

- Add a **Copy Data** activity.

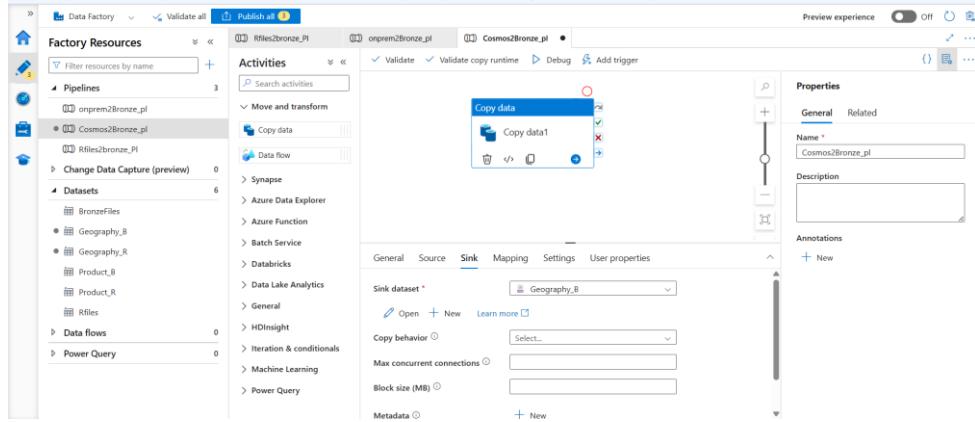
Source (Cosmos DB)

- Dataset → **Azure Cosmos DB**.
- Linked Service → **CosmosDb2adf**
- Choose your Database and Collection.(Json)



Sink (ADLS)

- Dataset → **Azure Data Lake Storage Gen2**.
- File format → JSON.
- Linked Service → **adls2adf_ls**.
- Path → **denis-medallian/bronze/ Geography.json** (to keep data separate).



Master Pipeline Setup(Raw2Bronze)

1. Go to Azure Data Factory → Author → New Pipeline

Name it → **Raw2Bronze_PL**.

2. Add "Execute Pipeline" activities

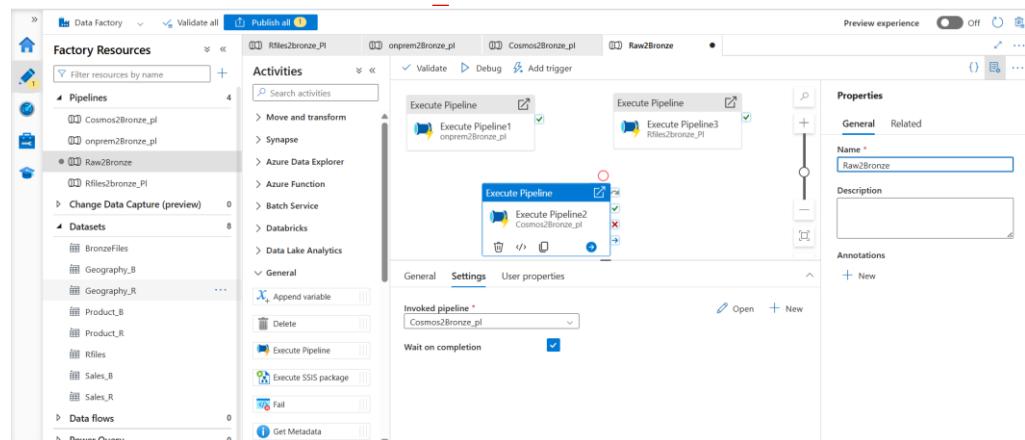
- o From Activities pane → Search for **Execute Pipeline**.
- o Drag one into the canvas for each pipeline you want to call.

3. Configure Each Execute Pipeline

- o **Settings** → Choose the pipeline you want to run.

Example:

- First → **Rfiles2bronze_Pl** same inside **ADLS2bronze_Pl**
- Second → **MySQL2bronze_Pl**
- Third → **Cosmos2bronze_Pl**



4. Link them in sequence or parallel

- o If you want all at once (parallel) → Don't connect activities with arrows, just place them side-by-side.
- o If you want one after another (sequence) → Connect with success (green) arrows.

5. Publish & Trigger

- o Click **Publish All**
- o Trigger the **Master_AllBronze_PL** → This will execute all child pipelines as per your arrangement.

The image contains two side-by-side screenshots of the Azure Data Factory 'Pipeline runs' interface. Both screens show the 'Raw2Bronze - Activity runs' section for a specific pipeline run ID.

Screenshot 1 (Top):

Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime	User prop...	Activity run ID
Execute Pipeline3	In progress	Execute Pipeli...	8/11/2025, 11:49:22 AM	6s		d74f6565-bd20-4...	
Execute Pipeline1	In progress	Execute Pipeli...	8/11/2025, 11:49:22 AM	6s		8c54fb21-cb5f-49	
Execute Pipeline2	In progress	Execute Pipeli...	8/11/2025, 11:49:22 AM	6s		eea9df1c-6695-45	

Screenshot 2 (Bottom):

Activity name	Activity st...	Activit...	Run start	Duration	Integration runtime	User prop...	Activity run ID
Execute Pipeline3	Succeeded	Execute Pipeli...	8/11/2025, 11:49:22 AM	21s		d74f6565-bd20-4...	
Execute Pipeline1	Succeeded	Execute Pipeli...	8/11/2025, 11:49:22 AM	24s		8c54fb21-cb5f-49	
Execute Pipeline2	Succeeded	Execute Pipeli...	8/11/2025, 11:49:22 AM	18s		eea9df1c-6695-45	

Perfect !

So your **Raw → Bronze** data movement is fully functional now — from Blob, MySQL, and Cosmos DB all landing neatly in ADLS Bronze folders.

The screenshot shows the Azure Storage browser interface for the 'denisstroageadls' storage account. The left sidebar navigation includes 'Storage browser' under 'Data storage'.

The main pane displays the 'denis-medallian' blob container, which contains the following items:

- File shares
- Queues
- Tables
- Logs
- denis-fact/raw
- denis-medallian
- ..
- sales
- Categories.csv
- Geography.json
- Product.csv
- SalesRep.xlsx
- SubCategories.csv

The logical next step in the **medallion architecture** would be:

1. Bronze → Silver:

- Clean, validate, and standardize the data
- Convert formats if needed (e.g., CSV → Parquet for analytics)

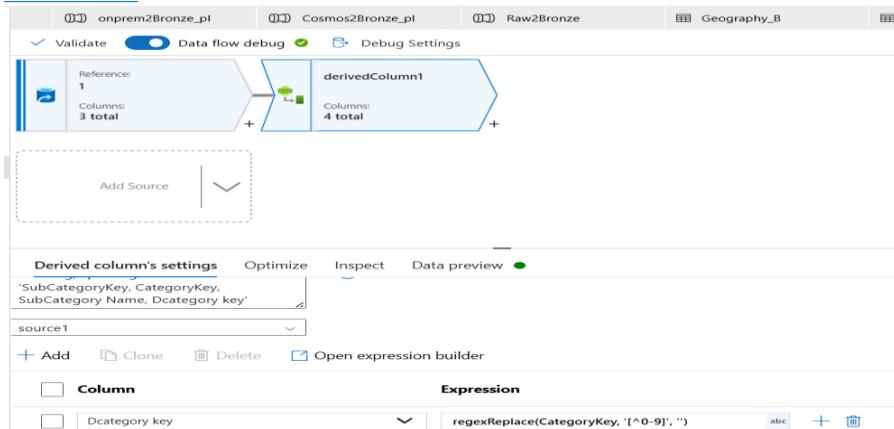
1.Data Flow for SubCategories.csv:-

1.1 Source:-

- **File:** SubCategories.csv (inside sales folder in ADLS container denis-medallian in bronze)
- **Dataset:** Delimited text, schema from file

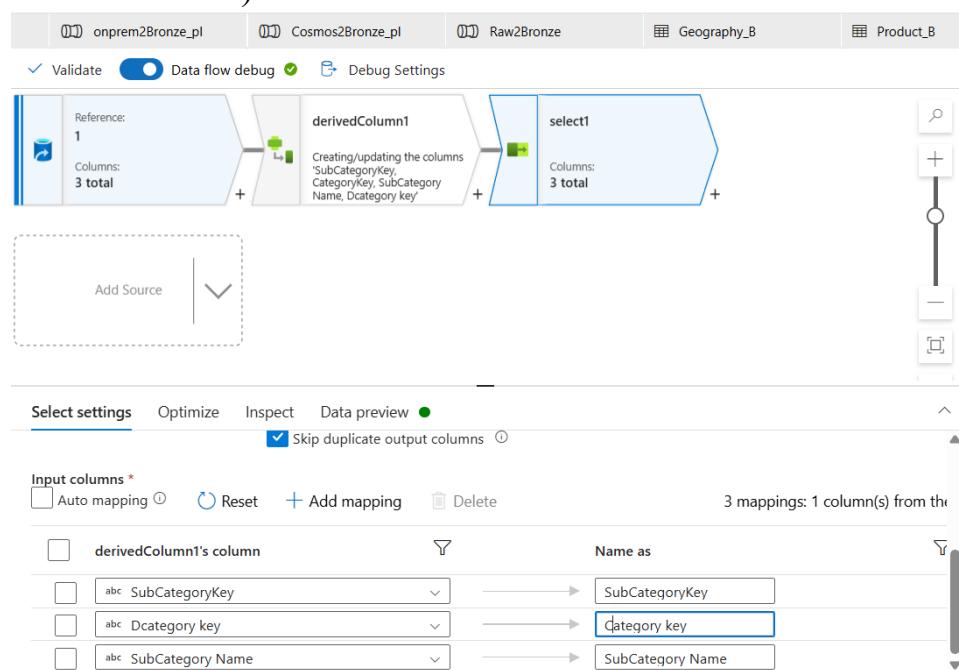
1.2 Derived Column (DerivedCategoriesKey):-

- Created a **new column:** DCategorieKey
- **Expression:** `regexReplace(CategoryKey, '[^0-9]', '')`



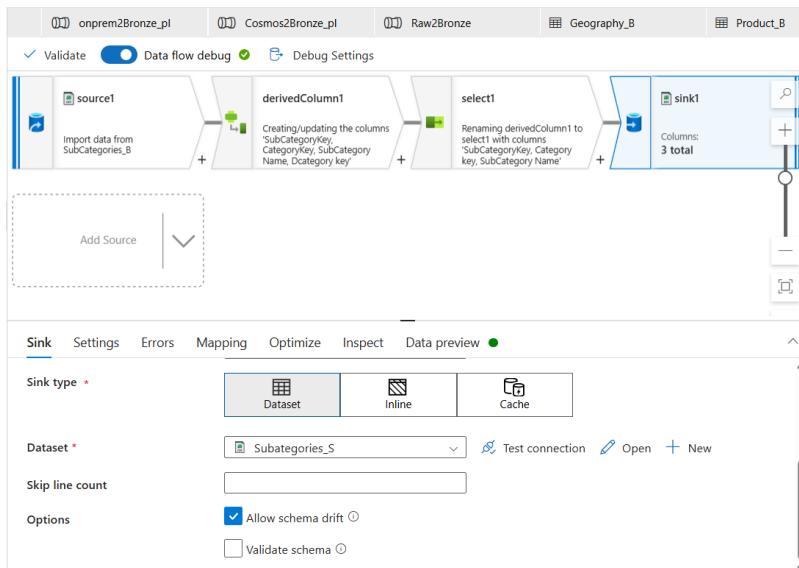
1.3 Select Transformation :-

- Removed the original CategoryKey column (since you have the derived one now)



1.4 Sink :-

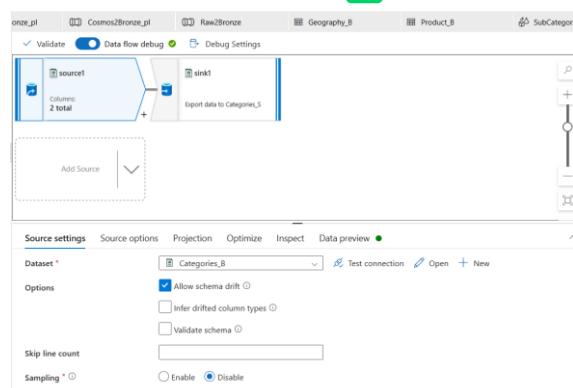
- Path: Possibly Bronze → Silver folder in ADLS
- Format: Parquet or CSV (depending on your target design)
- Note :- go to setting Select File name option in the select
- Pattern :- give name as(Subcategories.csv) and select the Optimize option in that select Single partition
- Now Click Publish All



2. Steps for Categories_DF:-

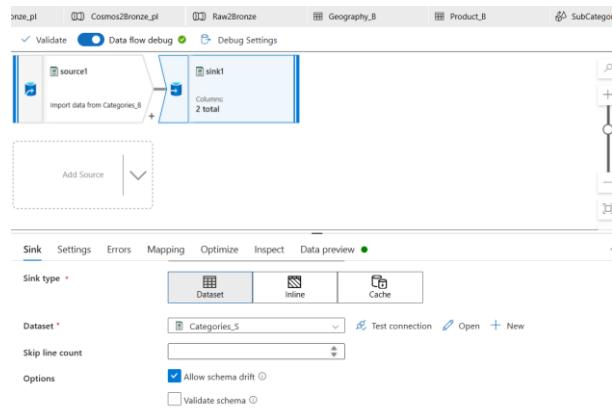
2.1 Source:-

- File: Categories.csv from Bronze (in ADLS denis-medallian sales folder).
- Format: Delimited text
- First row as header



2.2 Sink:-

- Silver Layer path: denis-medallian /silver/Categories



- Now Click Publish All

3. Steps for Geography_df :-

3.1 Source:-

- File: Geography.json from Bronze (denis-medallian/sales folder).
- Format: JSON.
- Document form: If it's a standard array of objects, set "Document per line" to false. If each object is in a separate line, set it to true.
- Projection: Enable schema drift if structure may change.

3.2 Select Transformation :-

- Remove these columns:
 - .wikipedia
 - _id

3.3 Surrogate Key Transformation :-

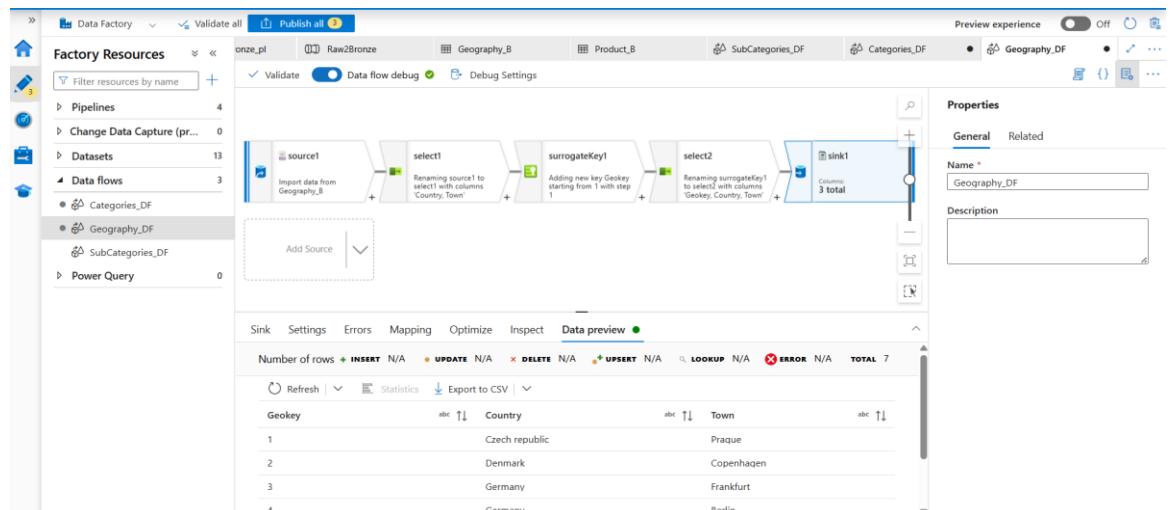
- Name: GeoKey
- This will generate unique keys for each row

3.4 Select Transformation (Reorder) :-

- Arrange the columns so GeoKey is first, followed by the rest of your business columns.

3.5 Sink:-

- Output Path: - denis-medallian/sales/silver/Geography/



4.Product_DF Pipeline Logic:-

4.1 Source:-

- File: Product.csv from Bronze folder in ADLS.
- Format: CSV.
- Enable schema drift.

4.2 Aggregate Transformation :-

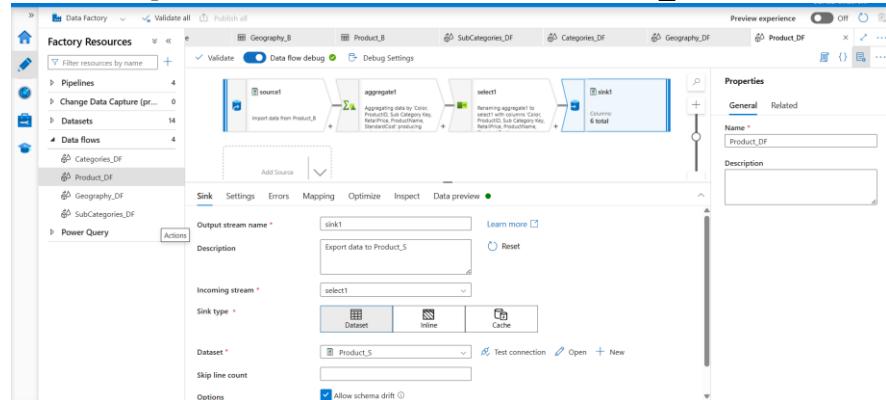
- **Group By** → all product columns (so that duplicates collapse).
- **Aggregates** →
 - Name: CountRow
 - Expression:count(ProductID)
 - This counts how many times each product appears.

4.3 Select Transformation:-

- Remove the CountRow column (since we only used it for duplicate checking).
- Keep only the grouped product columns.

4.4 Sink:-

Output Path:-denis-medallian/silver/Product_S/



5.SalesRep_DF Pipeline Logic :-

5.1Source

- File: SalesRep.xlsx from Bronze in ADLS.
- Format: Excel (point to correct sheet, enable header).
- Enable schema drift.

5.2 Derived Column Transformation:-

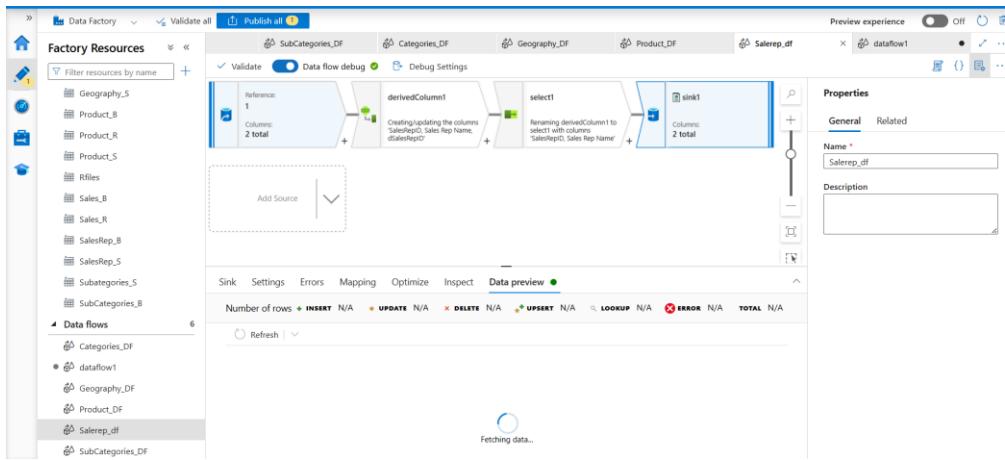
- Add column: dSalesRepID
- Expression: regexReplace(SalesRepID, '[^0-9]', '')

5.3 Select Transformation:-

- Remove original SalesRepID column.
- Rename dSalesRepID → SalesRepID

5.4 Sink:-

- Output Path: denis-medallian//silver/SalesRep_S/



6. Sales_DF Data Flow:-

6.1 Source

- File: sales from Bronze in ADLS
- Format: CSV (schema drift enabled)

6.2 Derived Column:-

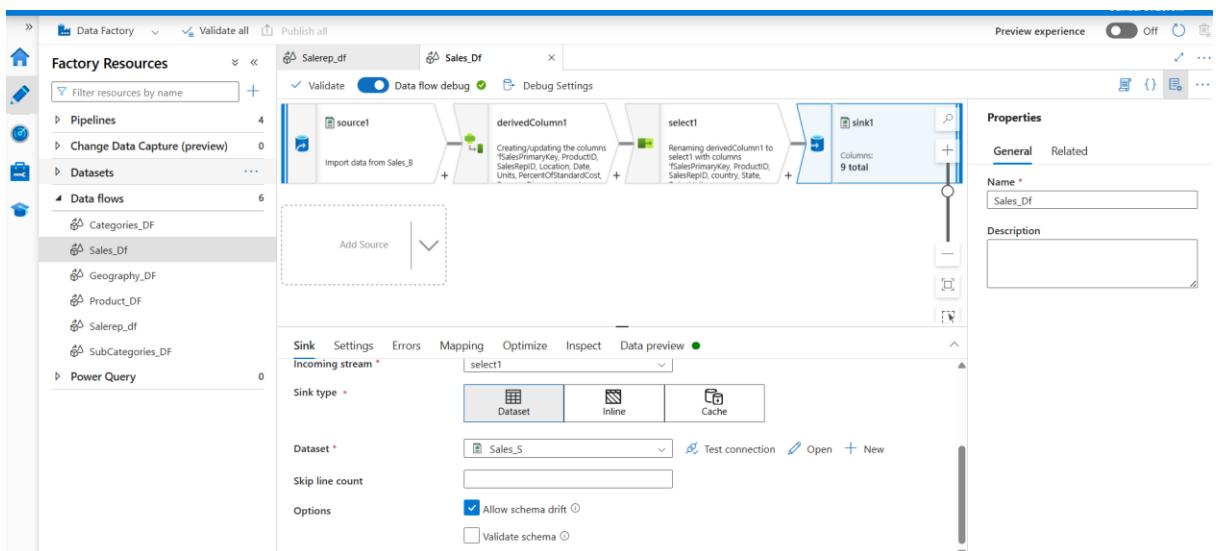
- Add column: **country** (1)
- Expression: `split(Location, ';')[1]`
- Add column: **state** (2)
- Expression: `split(Location, ';')[2]`
- Add column: **Date** (3)
- Expression: `toDate(Date, 'dd.MM.yyyy')`

6.3 Select Transformation:-

- Remove Location (original combined column)
- Keep the new country, state, and cleaned Date along with other necessary columns

6.4 Sink:-

Path:- denis-medallian/silver/Sales_S



Master Pipeline Setup(Bronze2Silver):-

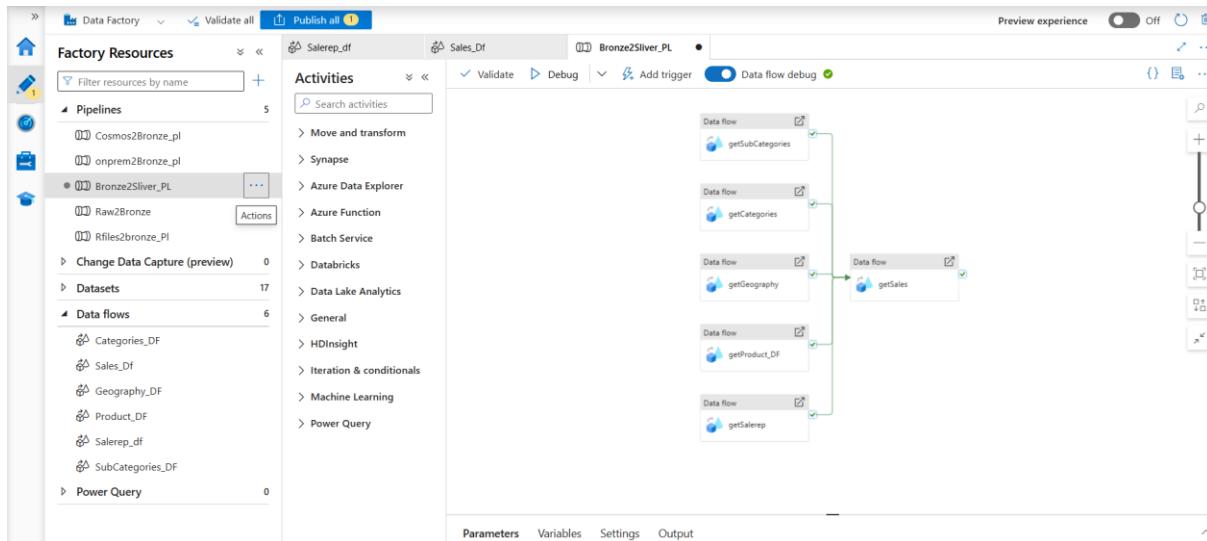
- Name it something like Bronze2Silver_PL

Add Data Flow Activities

- For each transformation you built earlier:

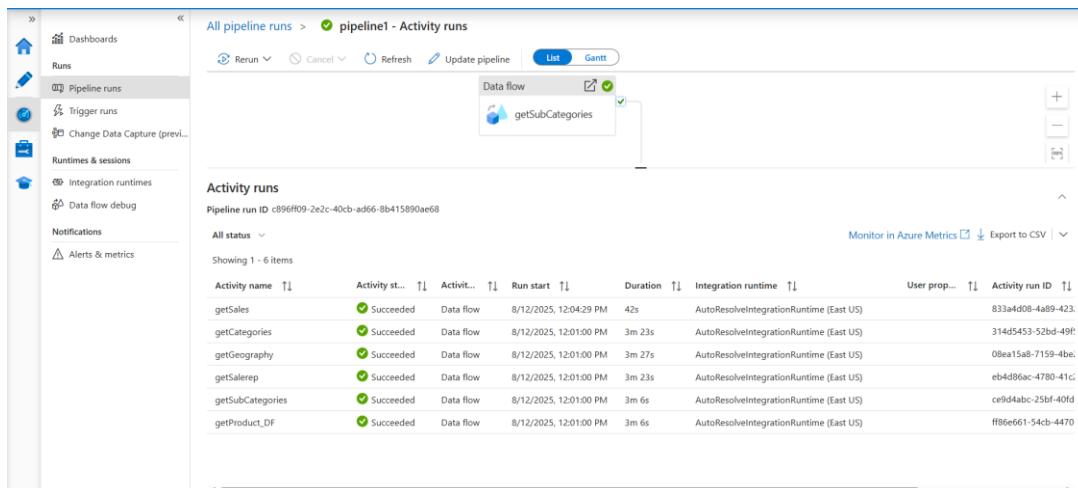
1. Drag a Data Flow activity into the pipeline canvas.
2. Select the corresponding data flow (e.g., SubCategories_DF, Category_DF, Geography_DF, Product_DF, SalesRep_DF, Sales_DF)

Execution Order:-



Testing

- Debug run the pipeline — check each Data Flow's output in Monitor tab.



Nice 🎉

That means your bronze → silver ETL layer is done, and your silver tables are now clean, structured, and ready for joining.

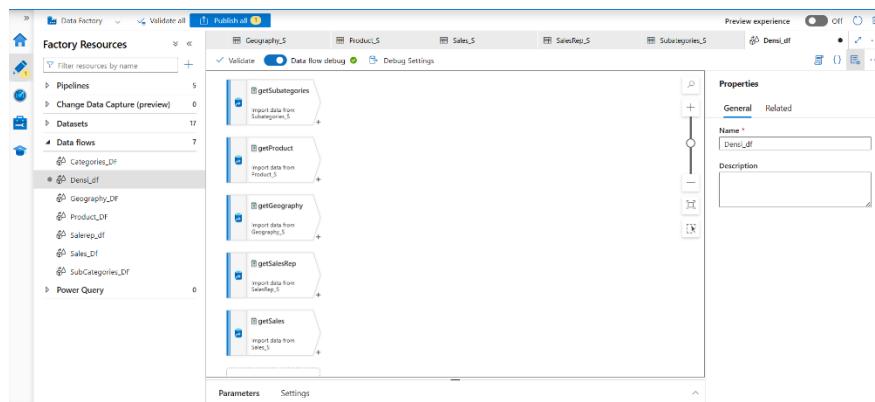
Now the fun part begins — Silver → Gold:

- Joining all your silver dimension tables with the Sales_DF fact staging table
- Creating the Fact_Sales table with surrogate keys and cleaned measures

Denis_DF data flow is doing this:

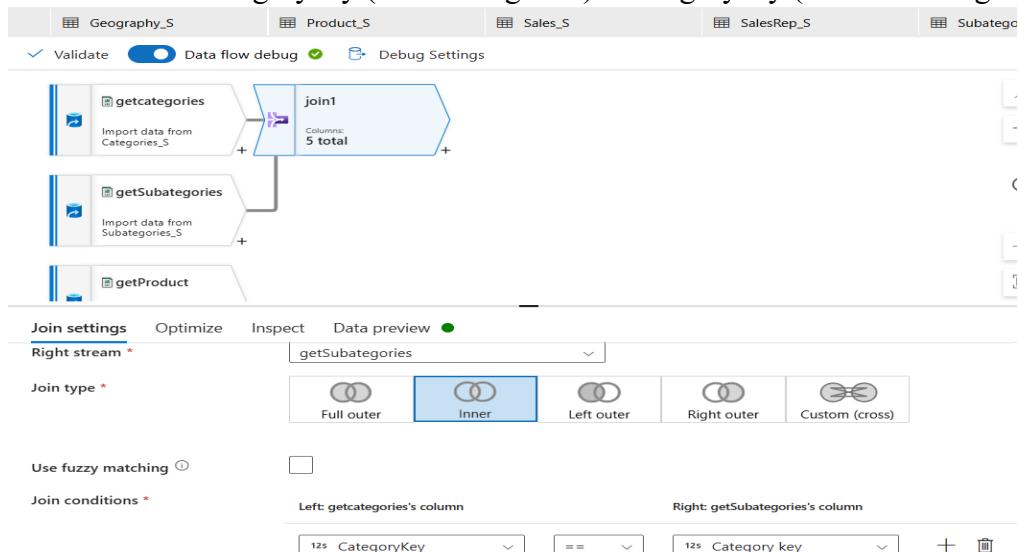
1. Sources:

- Added 6 silver-layer source datasets:
 - Categories_S → getCategories_S
 - Subcategories_S → getSubcategories_S
 - Product_S → getProduct_S
 - Geography_S → getGeography_S
 - SalesRep_S → getSalesRep_S
 - Sales_S → getSales_S



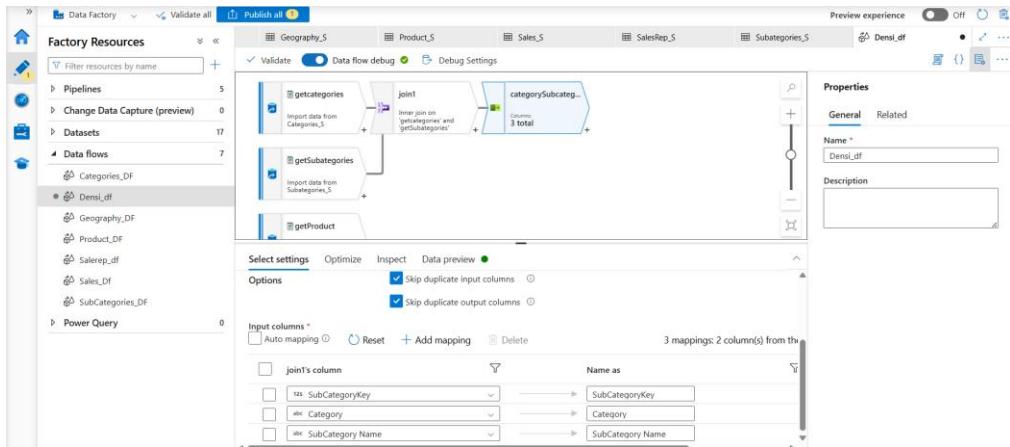
2. Join Step 1:

- Join: getCategories_S INNER JOIN getSubcategories_S
- Condition: categorykey (from Categories) = categorykey (from Subcategories)



3. Select Step 1:

- Removed categorykey columns from both joined streams to avoid duplication
- Output stream name: **categorySubcategoriesData**

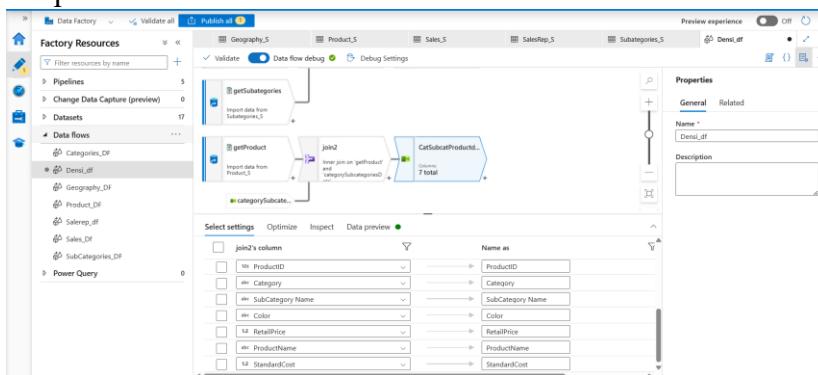


4. Join Step 2

- Join: categorySubcategoriesData INNER JOIN getProduct_S
- Condition: subcategorykey (from categorySubcategoriesData) = subcategorykey (from Product_S)

5. Select Step 2

- Removed the duplicate subcategorykey from both streams
- Output stream name: catSubcatProductData

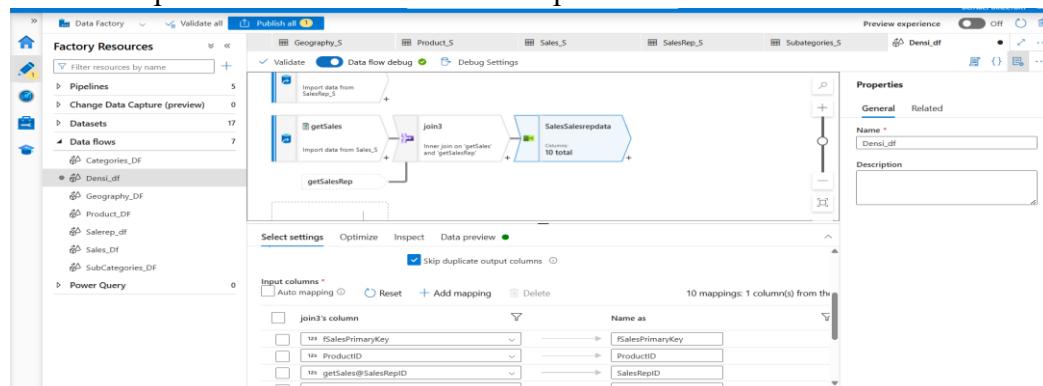


6. Join Step 3

- Join: getSales_S INNER JOIN getSalesRep_S
- Condition: SalesRepID (from Sales_S) = SalesRepID (from SalesRep_S)

7. Select Step 3

- Removed one duplicate SalesRepID column
- Output stream name: SalesSalesRepData

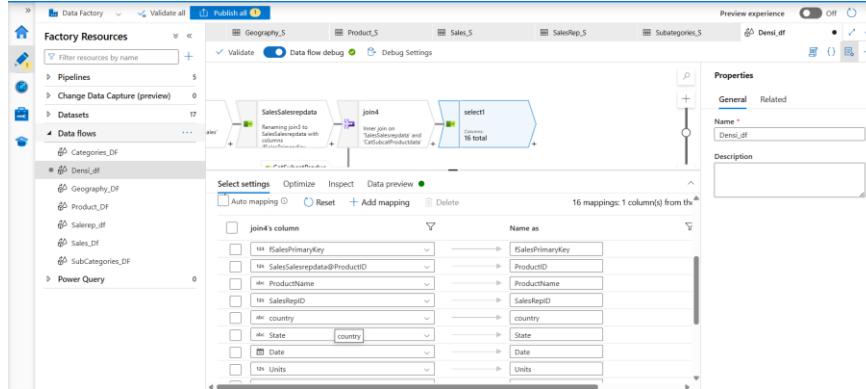


8. Join Step 4

- Join: SalesSalesRepData INNER JOIN catSubcatProductData
- Condition: productkey (from SalesSalesRepData) = productkey (from catSubcatProductData)

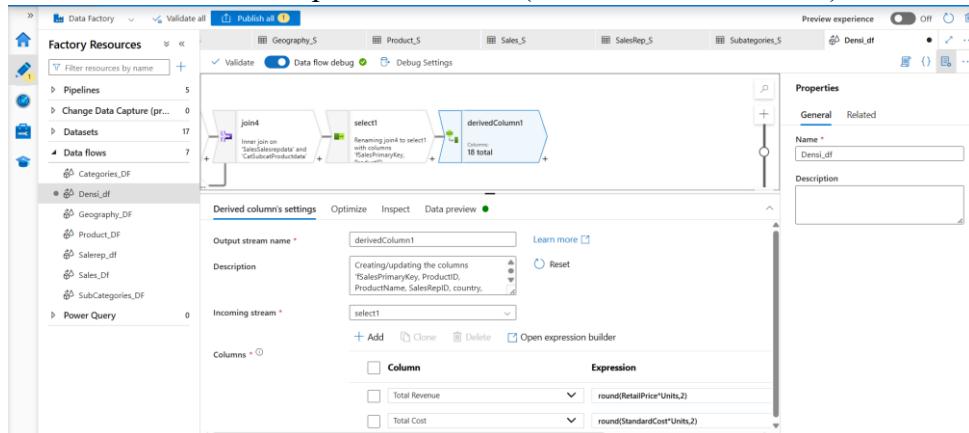
9. Select Step 4

- Removed one duplicate productkey column



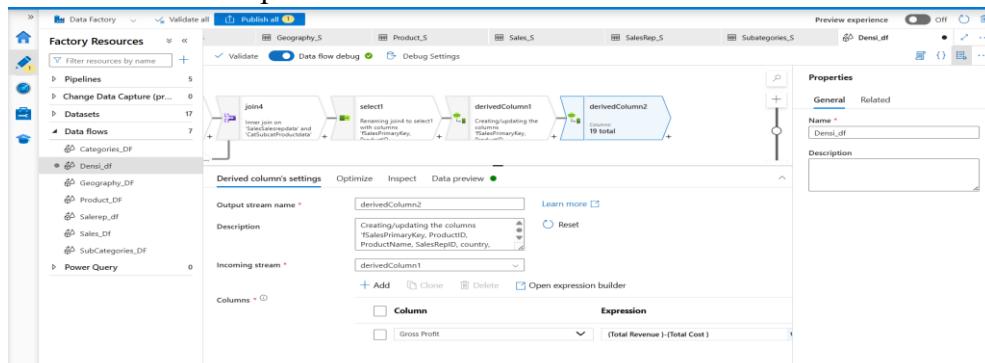
10. Derived Column Step

- New Columns:
 - TotalRevenue → Expression:Round(RetailPrice * Units,2)
 - TotalCost → Expression: Round(StandardCost * Units,2)



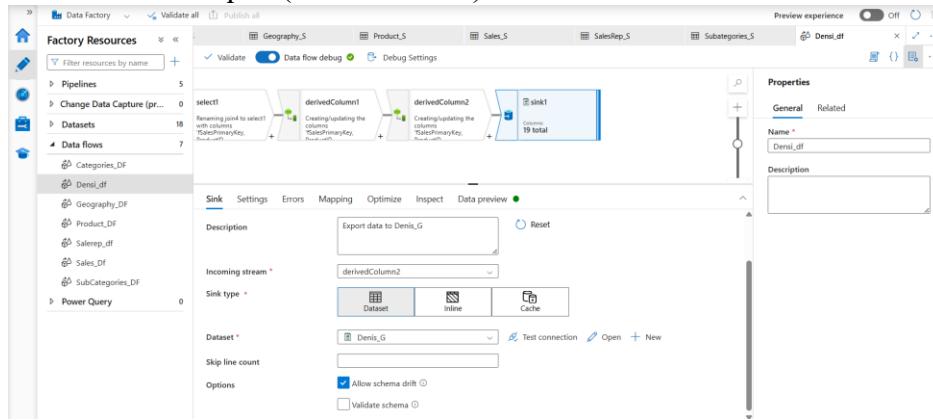
11. Derived Column (Profit Calculation)

- Input stream: from previous step (with TotalRevenue and TotalCost columns).
- New Column:
 - Profit → Expression: TotalRevenue - TotalCost



12. Sink (Gold Layer)

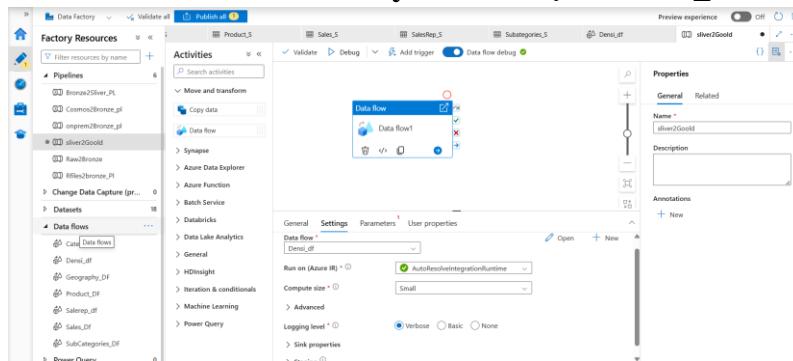
- Sink Type: ADLS Gen2
- Folder Path: denis-medallian/gold/Denis_G
- File Format: Parquet (recommended) or CSV



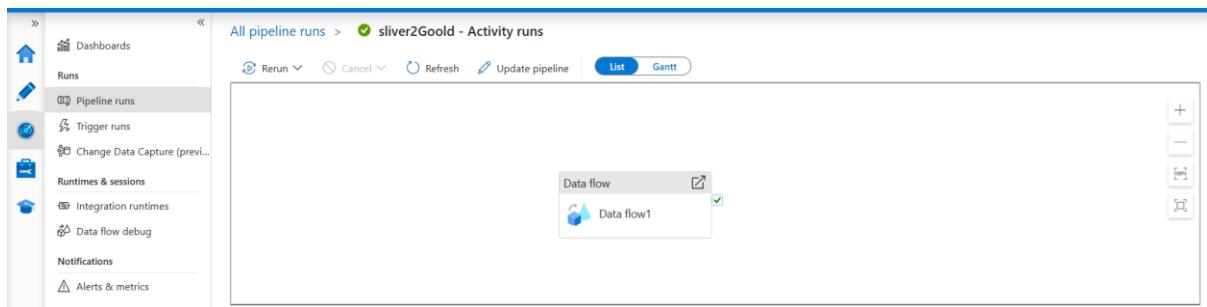
Now publish it

Silver2Gold pipeline:-

1. Add Data Flow activity → Select your Denis_DF data flow.



1. Set parameters (if you made any dynamic paths or filters in the Data Flow).
2. Publish All so it's saved in the Data Factory.

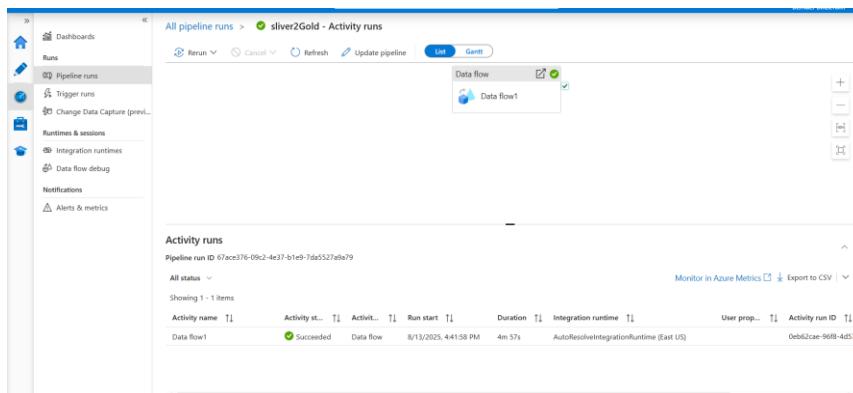
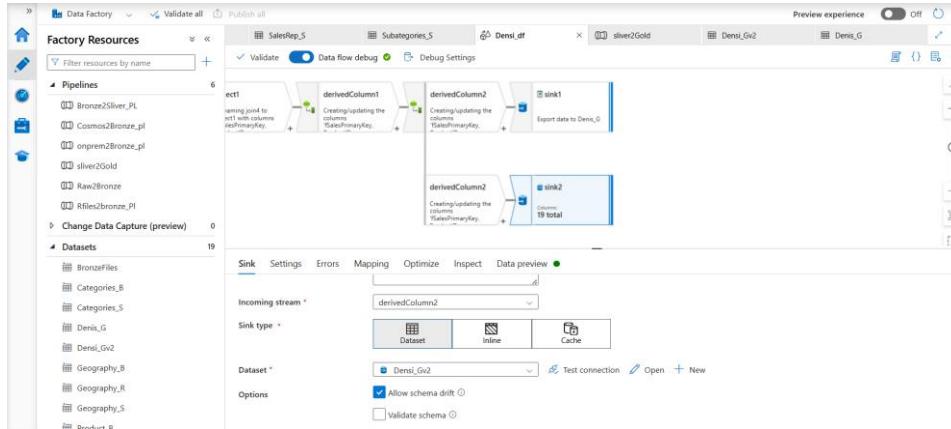


full Medallion architecture in ADF:

Raw → Bronze → Silver → Gold ✓

Add Sink in Data Flow

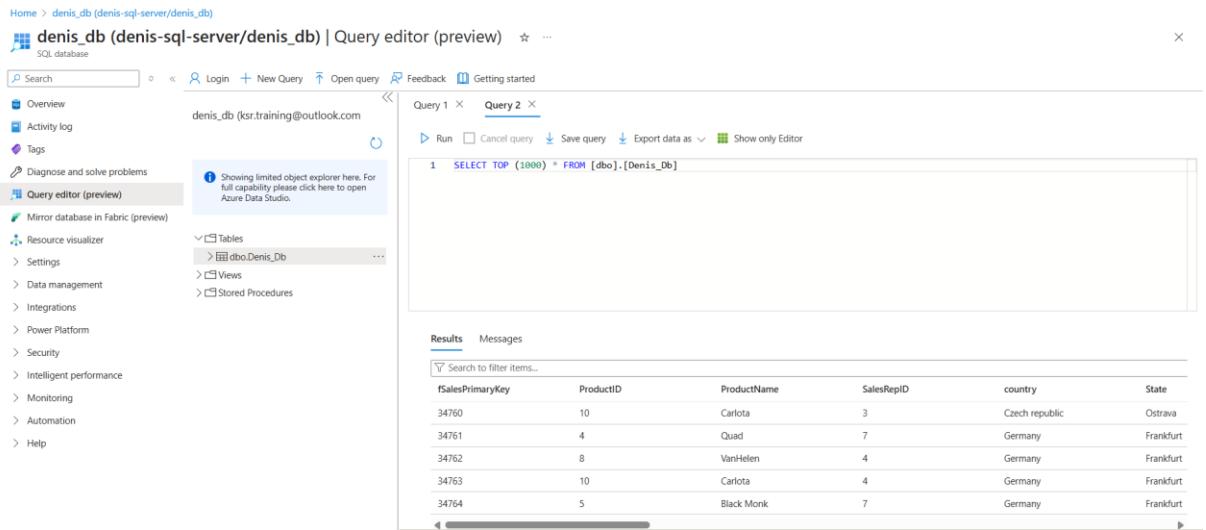
- Open your Denis_DF Data Flow
- After your last Derived Column (where you calculate profit), add a Sink transformation.
- Sink type → Azure SQL Database
- Linked service → Denis_G
- Table name → Create a new table in SQL



✓ Perfect — so after running your Silver2Gold pipeline, you've verified:

- Gold layer in ADLS → Denis-medallian/gold/Denis_G contains the final Fact_Sales file.

- Azure SQL Database → Table (e.g., Fact_Sales_Gold) contains the same transformed data.



The screenshot shows the Azure SQL Database Query editor (preview) interface. The left sidebar includes links for Overview, Activity log, Tags, Diagnose and solve problems, and a prominent Query editor (preview) link which is selected. Other links like Resource visualizer, Settings, Data management, Integrations, Power Platform, Security, Intelligent performance, Monitoring, Automation, and Help are also present. The main area has two tabs: Query 1 and Query 2. Query 1 is active and contains the following SQL code:

```
1 SELECT TOP (1000) * FROM [dbo].[Denis_Db]
```

Below the code, the Results tab is selected, displaying a table with six columns: fSalesPrimaryKey, ProductID, ProductName, SalesRepID, country, and State. The data in the table is as follows:

fSalesPrimaryKey	ProductID	ProductName	SalesRepID	country	State
34760	10	Carlota	3	Czech republic	Ostrava
34761	4	Quad	7	Germany	Frankfurt
34762	8	VanHelen	4	Germany	Frankfurt
34763	10	Carlota	4	Germany	Frankfurt
34764	5	Black Monk	7	Germany	Frankfurt