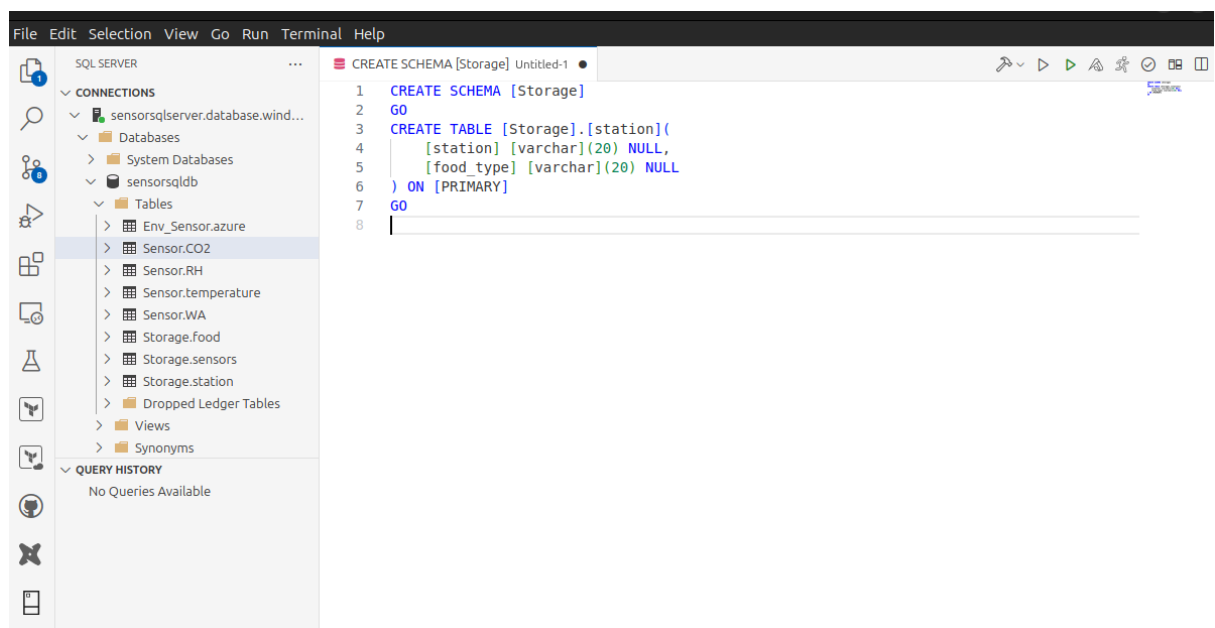


# Transfer data from Azure Blob Storage (Gen2) to SQL database

This tutorial is to show how to use Data Factory to transfer a CSV file to a SQL database table, including creating data sets for the file and table, configuring the copy activity, and verifying the successful transfer. I will also show how to debug the “mapping” schema to match the file to the table.

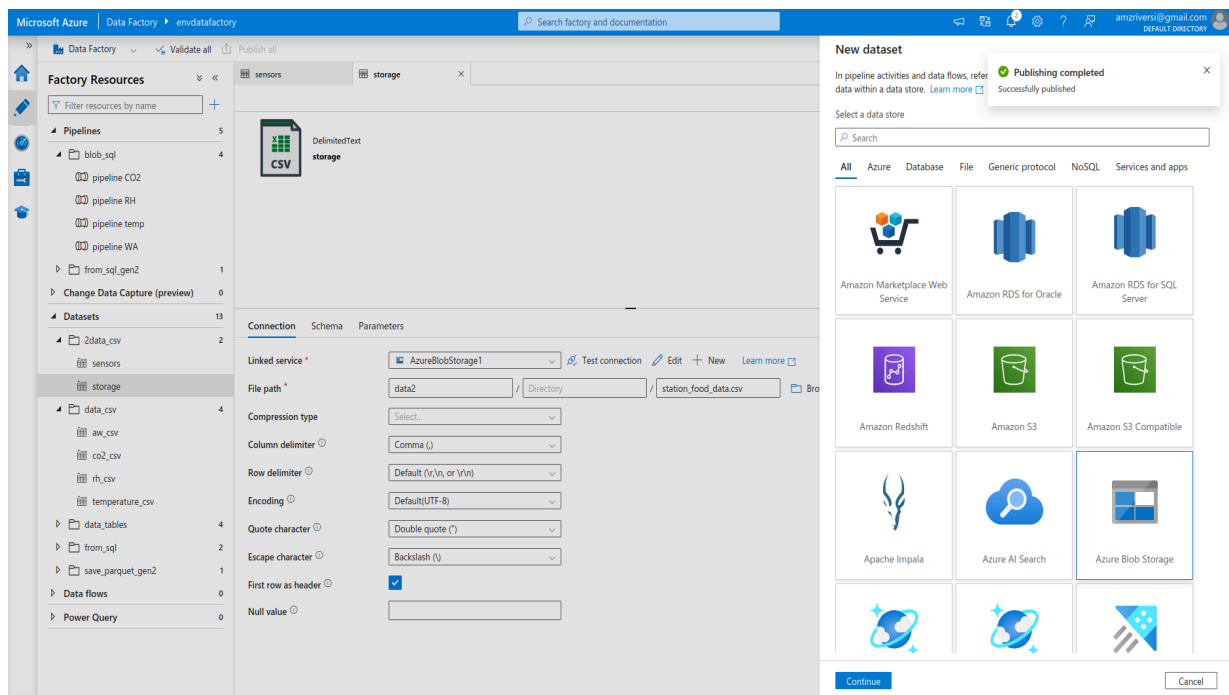
1. First, we create the schema and then the table. After running the code, we refresh the tables to see our database table. In Vscode you can connect with the Azure Server and remotely create a database.



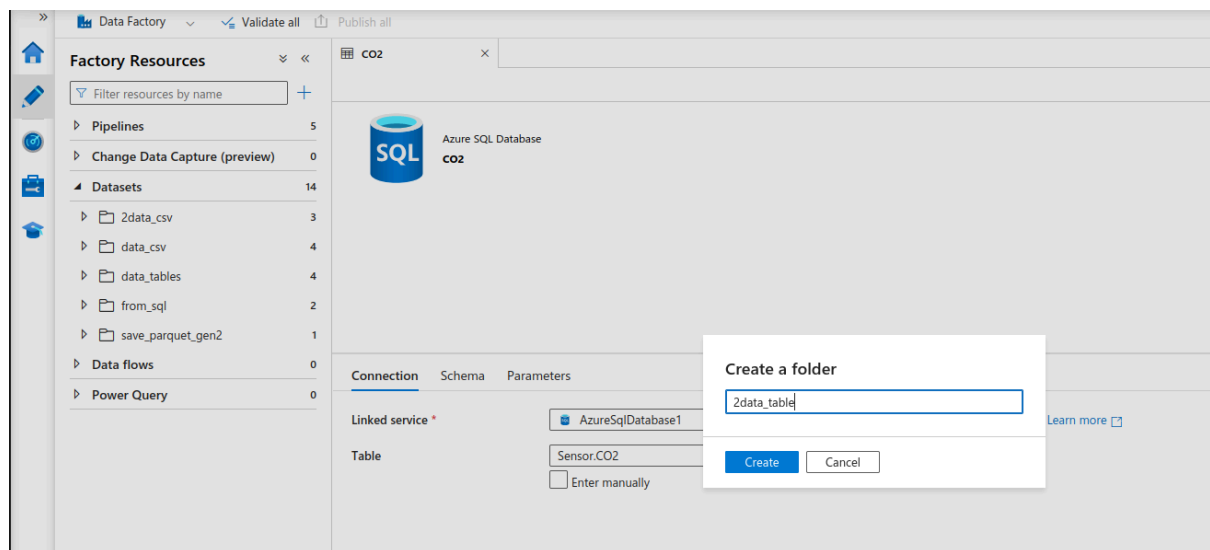
## 2. In Data Factory

- a. Create a new folder and then a pipeline within that folder. Next, create data sets for our source and sink.

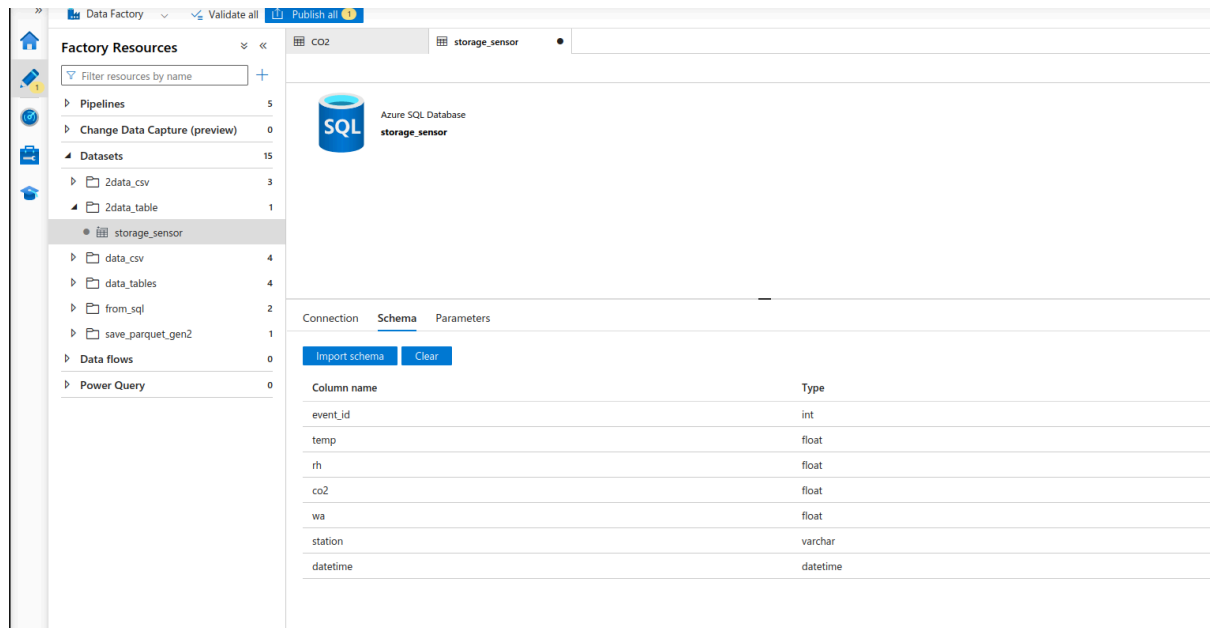
To begin, specify the linked service for the CSV file, which is the **blob storage**. Provide the file path and specify the delimiter as a comma. Since the CSV file has a header, the data factory can automatically read the columns from the file. This data set will serve as instructions for the data factory to read the data.



- b. Next, define the database table as a dataset in the Data Factory. Specify the linked service as the SQL database and provide the table name.

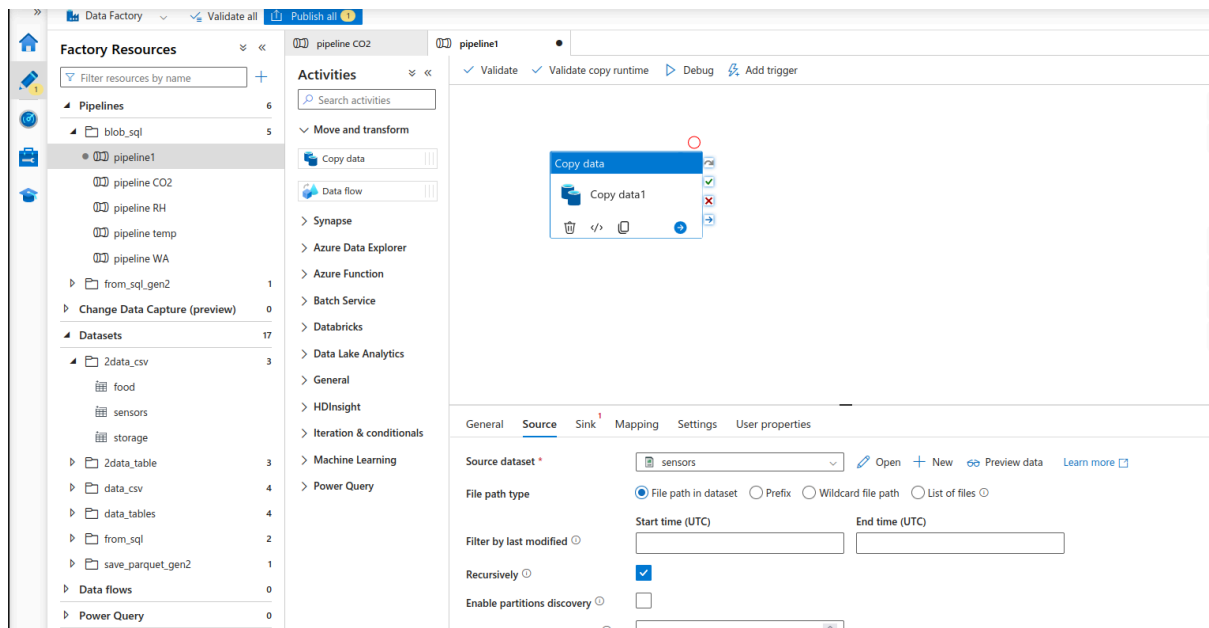


- c. Once specified the table name, Data Factory can read the columns from the actual table. These columns will be used as instructions for the data factory to write the data to the database.

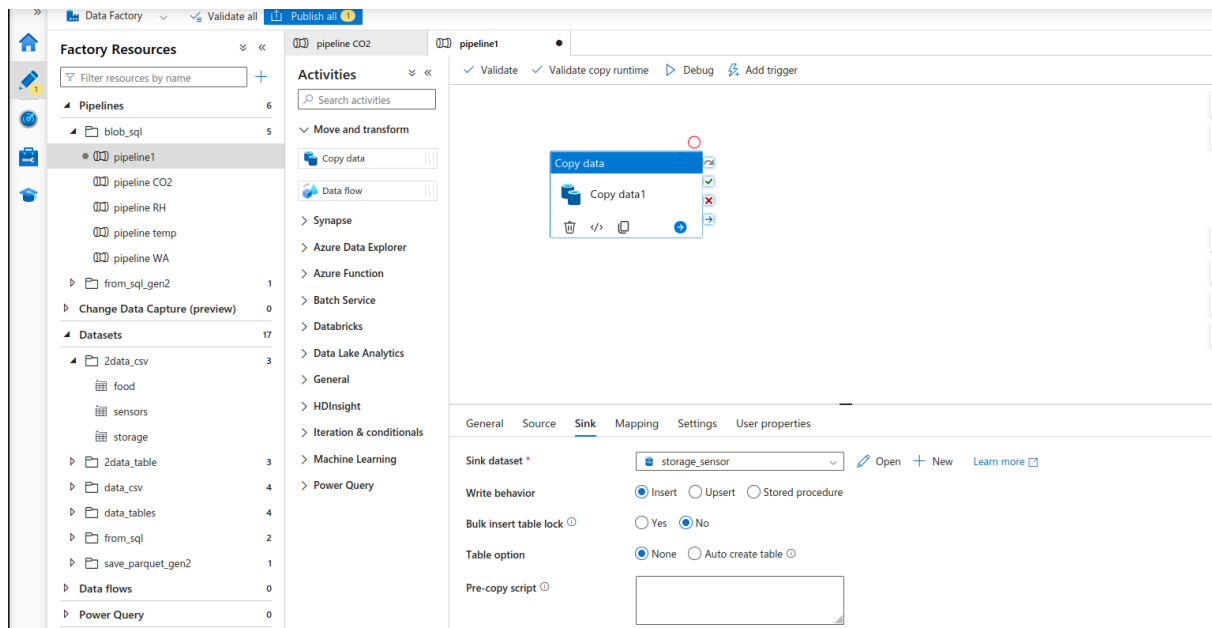


After completing these steps, configure the copy activity in the pipeline, add the **Copy file** that you can find in the pipeline **Activities** tab, and let the Data Factory handle the copying process.

- First select the **source** (csv files)

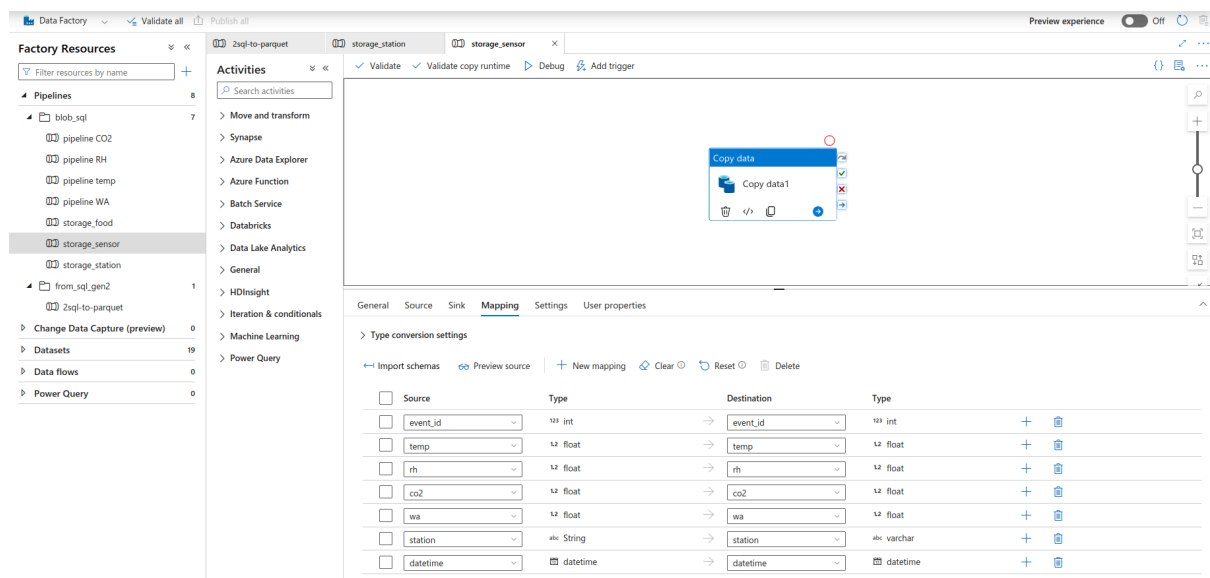


- Select destination (**sink**) for the file, in this example I have added all tables that will be the destination for the csv files in a folder 2data\_table. Select the table and do **Mapping**.



- e. The mapping tab allows us to easily see how our source data will be transformed and loaded into our destination database. By importing the schemas, Data Factory can automatically match up the columns from our CSV file to the corresponding columns in our database table. This seamless mapping process ensures that our data is accurately transferred without any manual intervention required.

However, if the schema does not match you can manually edit the json file (you can find the json file by clicking on the icon `{ }`) →



```
Pipeline name: pipeline1

Copy to clipboard

22      "recursive": true,
23      "enablePartitionDiscovery": false
24    },
25    "formatSettings": {
26      "type": "DelimitedTextReadSettings"
27    }
28  },
29  "sink": {
30    "type": "AzureSqlSink",
31    "writeBehavior": "insert",
32    "sqlWriterUseTableLock": false
33  },
34  "enableStaging": false,
35  "translator": {
36    "type": "TabularTranslator",
37    "mappings": [
38      {
39        "source": {
40          "name": "event_id",
41          "type": "String",
42          "physicalType": "String"
43        },
44        "sink": {
45          "name": "event_id"
```

## 4. Copy to SQL

After you can copy the csv file to the database:

**Validate** the pipeline, if everything is ok you can **Debug** and the data will be copied to SQL database.

The screenshot shows the Azure Data Factory interface. On the left, the 'Activities' pane lists various activity types. The main workspace displays a pipeline named 'pipeline1' with a 'Copy data' activity. Below the workspace, the 'Output' tab of the pipeline run is visible, showing the status of the 'Copy data' activity.

**Pipeline run ID:** e93bb4ab-9c53-46bb-b7f2-11752a9581f6

**Pipeline status:** Succeeded

**Activity name:** Copy data1

**Activity status:** Succeeded

**Activity type:** Copy data

**Run start:** 4/21/2024, 2:18:17 PM

**Duration:** 25s

**Integration runtime:** AutoResolveIntegration


If succeeded you can check the amount of data transferred (rows read - rows written )

Details


Refresh

Learn more on copy performance details from here.

Activity run id: 25725798-e8d2-40fe-824c-59d365b4f076

 Azure Blob Storage  
Region: UK South

Succeeded

 Azure SQL Database  
Region: UK South

Data read: ⓘ

8.957 MB

Files read: ⓘ

1

Rows read:

87,370

Peak connections: ⓘ

3

Data written: ⓘ

4.019 MB

Rows written: ⓘ

87,370

Peak connections: ⓘ

2

Copy duration

00:00:23

Throughput ⓘ

507.153 MB/s

Now you can use data from SQL databases!

Login

New Query

Open query

Feedback

Getting started

Showing limited object explorer here. For full capability please click here to open Azure Data Studio.

Tables

Sensor.CO2

Sensor.RH

Sensor.temperature

Sensor.WA

Storage.food

Storage.sensors

Storage.station

Views

Stored Procedures

Query 1 ×

Query 2 ×

Run

Cancel query

Save query

Export data as

Show only Editor

1

SELECT TOP (1000) \* FROM [Storage].[food]

Results

Messages

Search to filter items...

food_id	food_type	station	quantity	price_unity	price_total
1	Dried food	A	521	2.94789885067527	1535.8553012018158
2	Dried food	B	200	6.171390642246001	1234.2781284492003
3	Dairy	C	728	8.769832610226885	6384.438140245173
4	Dairy	D	566	7.46663569850433	4226.115805353451
5	Drinks	E	909	2.8980589688483382	2634.3356026831393
6	Drinks	F	776	8.700747450405473	6751.780021514647
7	Fresh veg	G	463	8.77889481336016	4064.6282985857542
8	Fresh veg	H	192	9.140818824400549	1755.0372142849055

Query succeeded | 0s