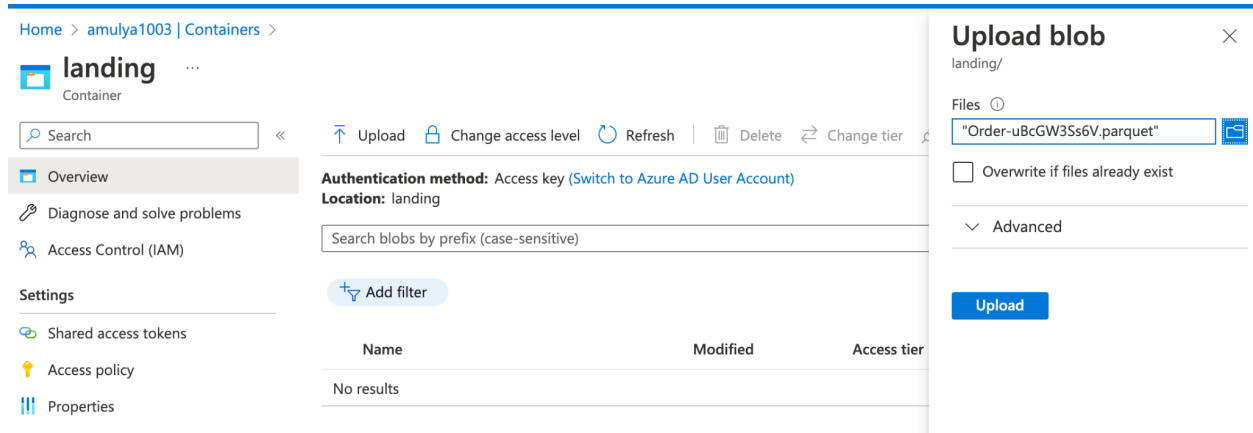


Copy Parquet file to JSON using Copy Activity in Azure data factory.

Create a storage Account if not exists and upload parquet file in a container.



Please refer to this link :

How to create storage account:

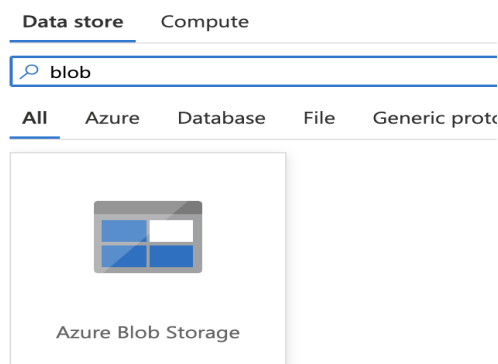
https://www.linkedin.com/posts/amulya1003_create-an-azure-storage-account-activity-6998852280389165057-BppX?utm_source=share&utm_medium=member_desktop

How to create linked service to connect to storage account.

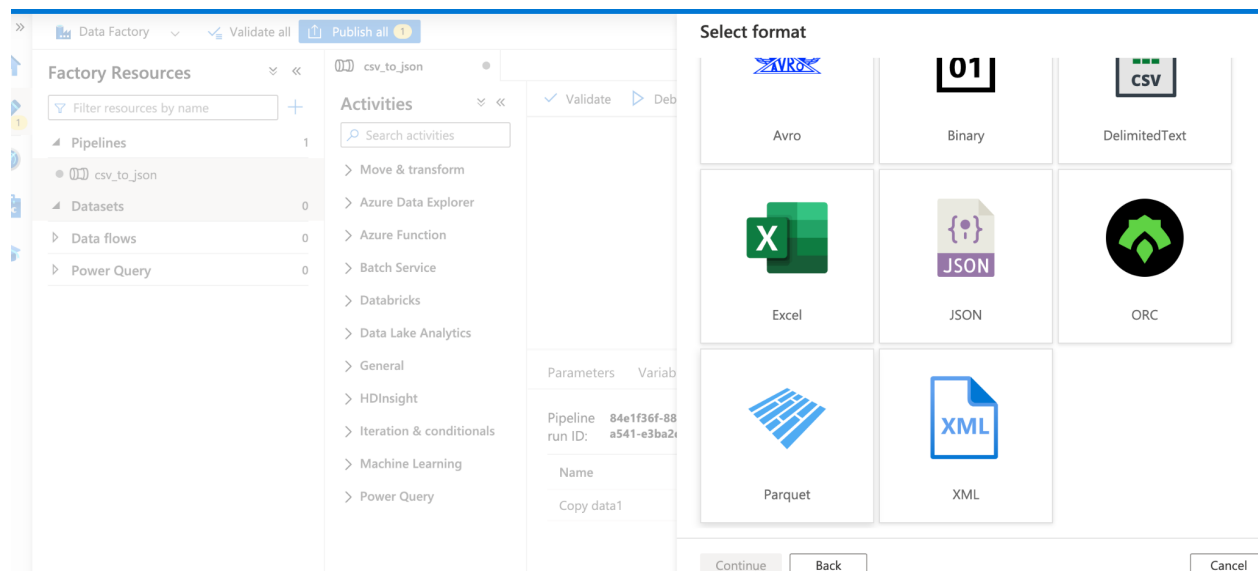
https://www.linkedin.com/posts/amulya1003_copy-activity-in-azure-data-factory-activity-6999064498095476736-HivU?utm_source=share&utm_medium=member_desktop

As csv file is in blob storage account, we need to select blob storage.

New linked service

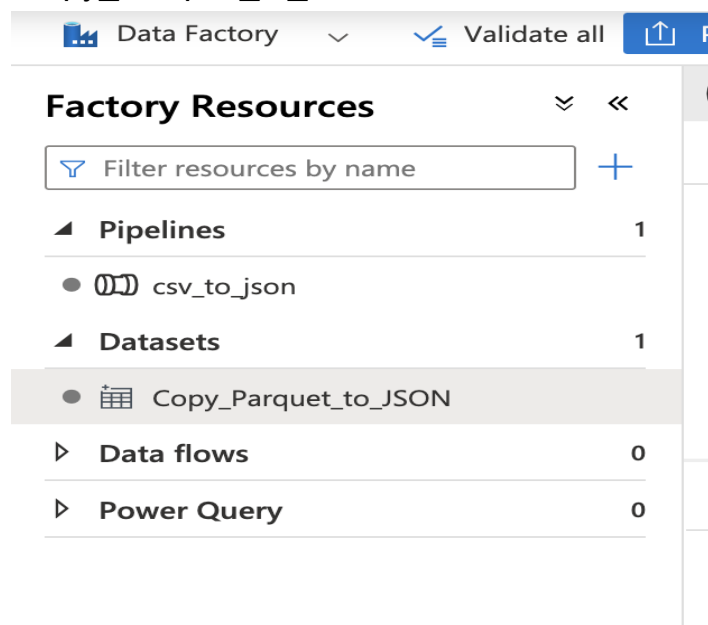


Select Parquet Datastore:



Connect to linked Service and select the root folder and click “OK” to create a dataset.

“Copy_Parquet_to_JSON” dataset created as shown below.



Create an other dataset with datasource as JSON and select destination path to land the file.

Select format

Choose the format type of your data



Avro



Binary



DelimitedText



Excel



JSON



ORC

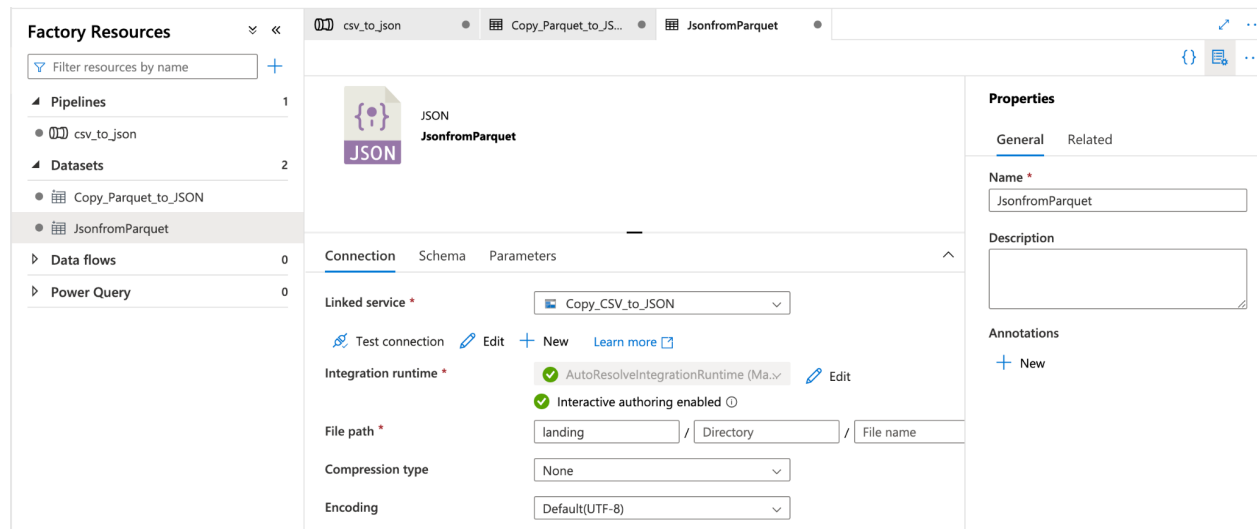


Continue

Back

Cancel

“JsonFromParquet” dataset created as shown below



Two datasets Successfully created .Now we need to create a pipeline to pick parquet file and change parquet to json and move it to Azure storage account.

Data Factory uses the Copy activity to move source data from **source** data location to a **sink** data store.

Select **source** and **sink** in toolbox pane and select source and destination dataset



parquet_to_json



Parquet_to_JSON



JsonfromParquet



>>

✓ Validate

✓ Validate copy runtime



Debug



Add trigger

Copy data



Copy data1



Expand toolbox pane

General

Source

Sink

Mapping

Settings

User properties

Source dataset *



Parquet_to_JSON



Open



New



Preview data

[Learn more](#)



File path type



File path in dataset



Prefix



Wildcard file path

Filter by last modified ⓘ

Start time (UTC)

End time (UTC)

Click on Debug to run the pipeline.

The screenshot shows the Azure Data Factory console. At the top, there are buttons for 'Validate all' and 'Publish all' (with a '3' indicating 3 items to publish). Below this, the pipeline 'parquet_to_json' is selected, showing its steps: 'Parquet_to_JSON' and 'JsonfromParquet'. The 'Debug' button is highlighted. On the right, the 'Properties' pane shows the 'General' tab with the name 'parquet_to_json' and a description field. Below the pipeline diagram, the 'Output' tab shows the pipeline run ID: 'd7d2134a-b40a-4387-a2b7-502de5317c29'. A table below shows the run details:

Name	Type	Run start	Duration	Status
ParquetToJson	Copy data	2022-11-17T05:35:28.388Z	00:00:58	Succeeded

At the end. Publish all to save all changes

This screenshot shows the top of the Azure Data Factory interface. The 'Publish all' button is highlighted with a yellow circle and the number '3', indicating that there are three items to be published. The pipeline 'parquet_to_json' is still selected.

Now we can see JSON file created :

The screenshot shows the Azure Storage Explorer interface. The 'landing' container is selected, and its contents are displayed in a table. The table has columns for 'Name', 'Modified', 'Access tier', 'Archive status', and 'Blob type'. Two files are listed:

Name	Modified	Access tier	Archive status	Blob type
Order-uBcGW3Ss6V.json	11/16/2022, 11:36:24...	Hot (Inferred)		Block blob
Order-uBcGW3Ss6V.parquet	11/16/2022, 11:28:55...	Hot (Inferred)		Block blob

Parquet is an open source, column-oriented data file format designed for efficient data storage and retrieval. Parquet is optimized to work with complex data in bulk and features different ways for efficient data compression and encoding types. This approach is best especially for those queries that need to read certain columns from a large table. Parquet can only read the needed columns therefore greatly minimizing the IO.

Parquet is a binary-based (rather than text-based) file format optimized for computers, so **Parquet files aren't directly readable by humans**. You can't open a Parquet file in a text editor the way you might with a CSV file and see what it contains.

Home > amulya1003 | Containers > landing >

Order-uBcGW3Ss6V.parquet ...

Blob

Save Discard Download Refresh Delete

Overview Versions Snapshots Edit Generate SAS

The file 'Order-uBcGW3Ss6V.parquet' may not render correctly as it contains an unrecognized extension.

1 PAR1 0 00 0\$, 0 0 Sub-Saharan Africa Asia 0000 0 0000 000Region 000Middle East and North Africa

2 000

3 me ~10\ 000Asia 000Sub-Saharan 0 y Europe~ 0,!000Central

4 { 04the Caribbean mus %0i Ocean0~0~00 0 (

5 B V00V 0b00 `0b 0" 0%0V00000~00009*0E0V000Q0 0

6 V00QF~

7 V>0V 0 T |

8 ~h0 <b0 Vs0 a

9 V*0 a 0 b{0 +00 V00V 0 eV 00 0 L 0 VH0

10 ~0 ~J0~@000 0V00V 000 ~009 N

11 0 00k0bq V00~00000Ve0V 0b[0B0 ~0095Vj0 ~J0~ 010Vh0~>0b00 W~A0Vw0V 0 T0u09 VI0~ 0~60 @ 0V2000 0%0 rN

12 0 0N 0 V00 <v 0 F~0 *VR0 z V200 0 L0W -wVy0 cV 0F0003 b0 0C0V000@0F00 0

13 Vf0~ 0V60~00B= VG000 bk 1 Vf0b70~00V00~60 0000B3 90B 0~00~ 0 [

14 V00000 0VC0 c (b0 VI00/0 x090V~0 0

15 bY0 #

16 Vw000 0 ~0 000h#0~\0 010 ~<0 2V 0v mVF00r02r0 m

Preview

The JSON stores key-value format. In the opposite side, **Parquet file format stores column data.**

[Home](#) > [amulya1003](#) | [Containers](#) > [landing](#) >

Order-uBcGW3Ss6V.json ...

Blob

Save Discard Download Refresh Delete

[Overview](#) [Versions](#) [Snapshots](#) [Edit](#) [Generate SAS](#)

```
1 [{"Prop_0": "Region", "Prop_1": "Country", "Prop_2": "ItemType", "Prop_3": "SalesChannel", "Prop_4": "OrderPriority", "Prop_5": "OrderID", "Prop_6": "Unit"},
2 {"Prop_0": "Middle East and North Africa", "Prop_1": "Libya", "Prop_2": "Cosmetics", "Prop_3": "Offline", "Prop_4": "M", "Prop_5": "686800706", "Prop_6": "Unit"},
3 {"Prop_0": "North America", "Prop_1": "Canada", "Prop_2": "Vegetables", "Prop_3": "Online", "Prop_4": "M", "Prop_5": "185941302", "Prop_6": "3018", "Prop_7": "1"},
4 {"Prop_0": "Middle East and North Africa", "Prop_1": "Libya", "Prop_2": "Baby Food", "Prop_3": "Offline", "Prop_4": "C", "Prop_5": "246222341", "Prop_6": "Unit"},
5 {"Prop_0": "Asia", "Prop_1": "Japan", "Prop_2": "Cereal", "Prop_3": "Offline", "Prop_4": "C", "Prop_5": "161442649", "Prop_6": "3322", "Prop_7": "205.7", "Prop_8": "1"},
6 {"Prop_0": "Sub-Saharan Africa", "Prop_1": "Chad", "Prop_2": "Fruits", "Prop_3": "Offline", "Prop_4": "H", "Prop_5": "645713555", "Prop_6": "9845", "Prop_7": "1"},
7 {"Prop_0": "Europe", "Prop_1": "Armenia", "Prop_2": "Cereal", "Prop_3": "Online", "Prop_4": "H", "Prop_5": "683458888", "Prop_6": "9528", "Prop_7": "205.7", "Prop_8": "1"},
8 {"Prop_0": "Sub-Saharan Africa", "Prop_1": "Eritrea", "Prop_2": "Cereal", "Prop_3": "Online", "Prop_4": "H", "Prop_5": "679414975", "Prop_6": "2844", "Prop_7": "1"},
9 {"Prop_0": "Europe", "Prop_1": "Montenegro", "Prop_2": "Clothes", "Prop_3": "Offline", "Prop_4": "M", "Prop_5": "208630645", "Prop_6": "7299", "Prop_7": "1"},
10 {"Prop_0": "Central America and the Caribbean", "Prop_1": "Jamaica", "Prop_2": "Vegetables", "Prop_3": "Online", "Prop_4": "H", "Prop_5": "266467225", "Prop_6": "1"},
11 {"Prop_0": "Australia and Oceania", "Prop_1": "Fiji", "Prop_2": "Vegetables", "Prop_3": "Offline", "Prop_4": "H", "Prop_5": "118598544", "Prop_6": "4800", "Prop_7": "1"},
12 {"Prop_0": "Sub-Saharan Africa", "Prop_1": "Togo", "Prop_2": "Clothes", "Prop_3": "Online", "Prop_4": "M", "Prop_5": "451010930", "Prop_6": "3012", "Prop_7": "1"},
13 {"Prop_0": "Europe", "Prop_1": "Montenegro", "Prop_2": "Snacks", "Prop_3": "Offline", "Prop_4": "M", "Prop_5": "220003211", "Prop_6": "2694", "Prop_7": "1"},
14 {"Prop_0": "Europe", "Prop_1": "Greece", "Prop_2": "Household", "Prop_3": "Online", "Prop_4": "C", "Prop_5": "702186715", "Prop_6": "1508", "Prop_7": "668"},
15 {"Prop_0": "Sub-Saharan Africa", "Prop_1": "Sudan", "Prop_2": "Cosmetics", "Prop_3": "Online", "Prop_4": "C", "Prop_5": "544485270", "Prop_6": "4146", "Prop_7": "1"},
16 {"Prop_0": "Asia", "Prop_1": "Maldives", "Prop_2": "Fruits", "Prop_3": "Offline", "Prop_4": "H", "Prop_5": "714135705", "Prop_6": "7332", "Prop_7": "9.33", "Prop_8": "1"}]
```

!!!!Successfully moved PARQUET file to JSON file using COPY activity in pipeline!!!!

To avoid cost, you can delete all the resource groups if no more needed.

[Home](#) >

azurelib ☆ ☆ ...
Resource group

[+ Create](#) [Manage view](#) [Delete resource group](#)

[Overview](#)
[Activity log](#)
[Access control \(IAM\)](#)
[Tags](#)
[Resource visualizer](#)
[Events](#)

Settings
[Deployments](#)
[Security](#)
[Policies](#)
[Properties](#)
[Locks](#)

Cost Management

Notifications ✕
[More events in the activity log](#) → [Dismiss all](#) ▾

Deleting resource group azurelib

Running ✕

Deleting resource group azurelib

a few seconds ago

Successfully deleted storage account

✕

Successfully deleted storage account 'amulya1003'.

2 minutes ago

Upload Completed for Order-uBcGW3Ss6V.parquet

✕

62.02 KiB | amulya1003

13 minutes ago

Successfully deleted blob(s)

✕

Successfully deleted 1 blobs(s).

14 minutes ago

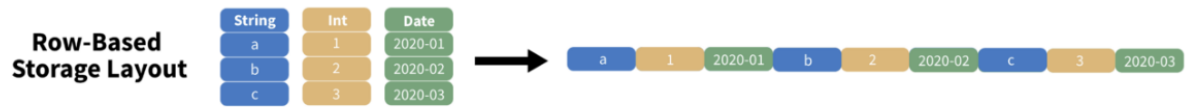
Successfully deleted blob(s)

✕

Successfully deleted 1 blobs(s).

15 minutes ago

In Parquet, each column is stored independently.



Row storage		Column storage	
Row 1	1	user_id	1
	US		2
	Free		3
Row 2	2	country	US
	UK		UK
	Paid		ES
Row 3	3	subscription_type	Free
	ES		Paid
	Paid		Paid

For more information on csv, json, parquet file formats:

	CSV	Parquet	JSON
Read Speed	✓	✓	
Small File Size		✓	
Splittable	✓	✓	✓
Included Data Types		✓	✓
Easy to Read	✓		✓
Nestable		✓	✓
Columnar		✓	
Complex Data Structures		✓	✓

JSON or **JavaScript Object Notation** is an open data format and is widely used by APIs (Application Programming Interface — basically how servers talk to each other) and several databases (like MongoDB).

JSON is an object notation and so it's in a sense **row-based**, not column-based. This makes it a little slower to work with.

For more updates . Follow me on LINKEDIN: [LinkedIn.com/in/amulya1003](https://www.linkedin.com/in/amulya1003)