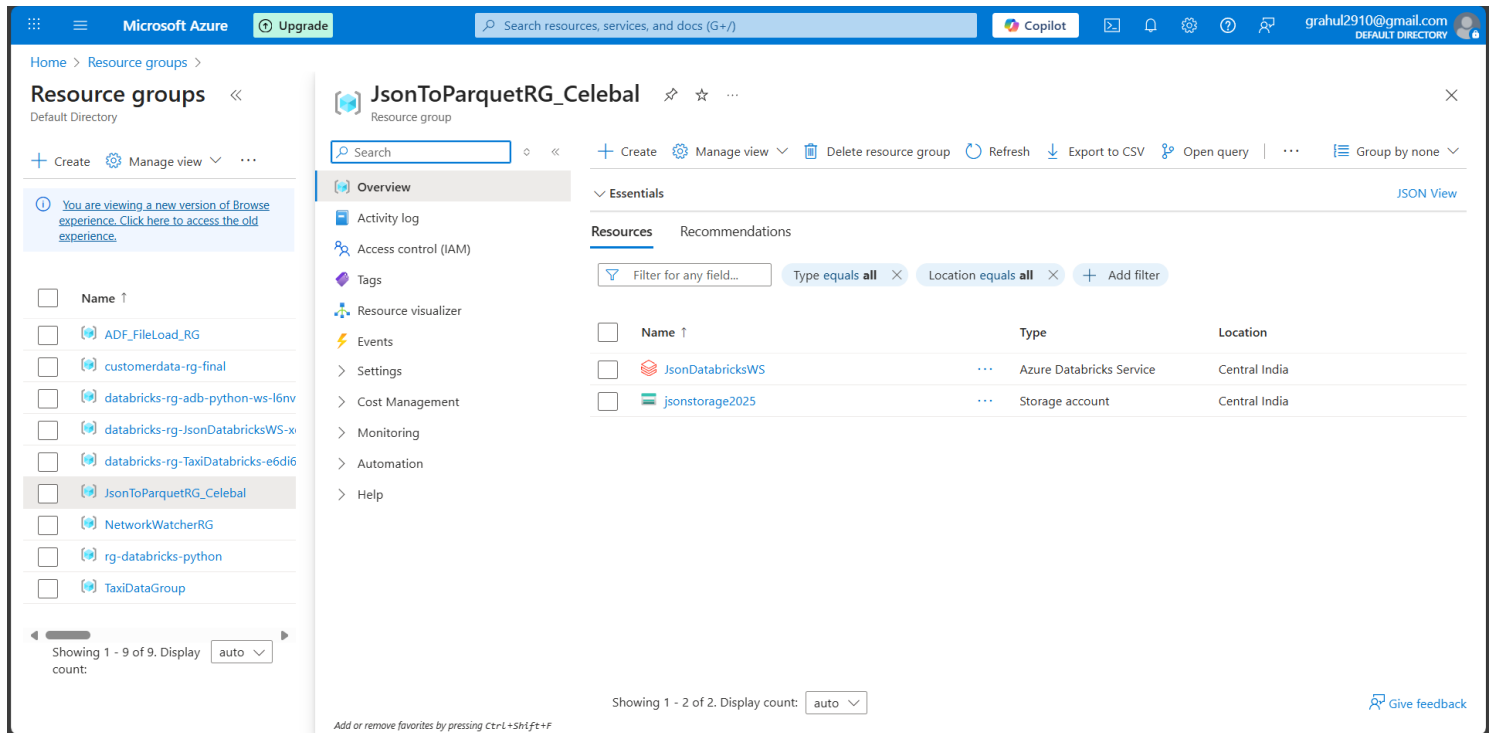


Flatten Nested JSON and Write as External Parquet Table

1. Azure Environment Setup

1. Created a Resource Group

- Name: JsonResourceGroup

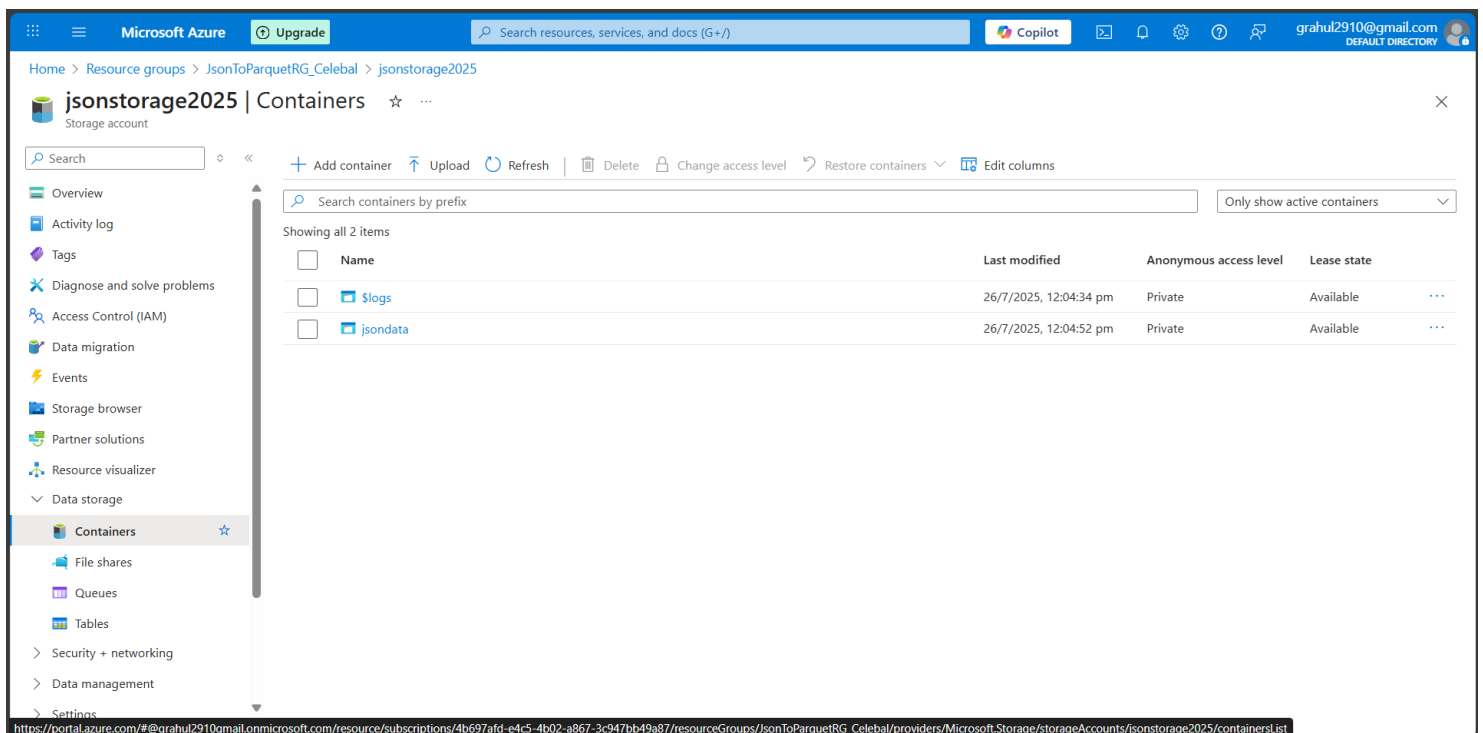


The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes the 'Microsoft Azure' logo, an 'Upgrade' button, a search bar, and a user profile. The left sidebar shows the 'Resource groups' section with a list of resource groups. The main pane displays the 'JsonToParquetRG_Celebal' resource group. The 'Overview' tab is selected, showing a table of resources within the group.

Name	Type	Location
JsonDatabricksWS	Azure Databricks Service	Central India
jsonstorage2025	Storage account	Central India

2. Created an Azure Storage Account

- Name: jsonstorage2025
- Enabled Hierarchical Namespace (for ADLS Gen2)

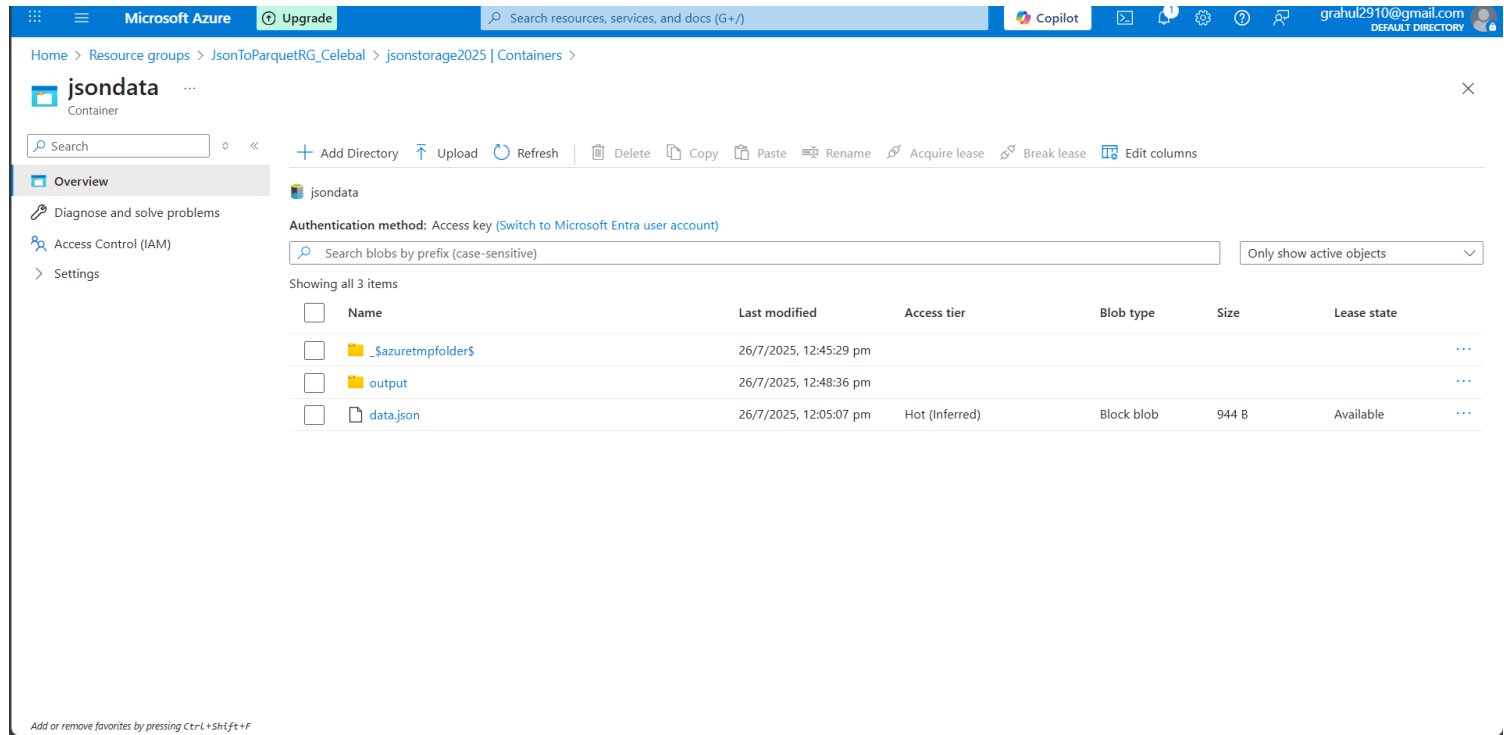


The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes the 'Microsoft Azure' logo, an 'Upgrade' button, a search bar, and a user profile. The left sidebar shows the 'Storage accounts' section with a list of storage accounts. The main pane displays the 'jsonstorage2025' storage account. The 'Containers' tab is selected, showing a table of containers within the storage account.

Name	Last modified	Anonymous access level	Lease state
\$logs	26/7/2025, 12:04:34 pm	Private	Available
jsondata	26/7/2025, 12:04:52 pm	Private	Available

3. Created a Container in the Storage Account

- Container name: jsondata
- Uploaded a nested JSON file (e.g., sample_nested.json)

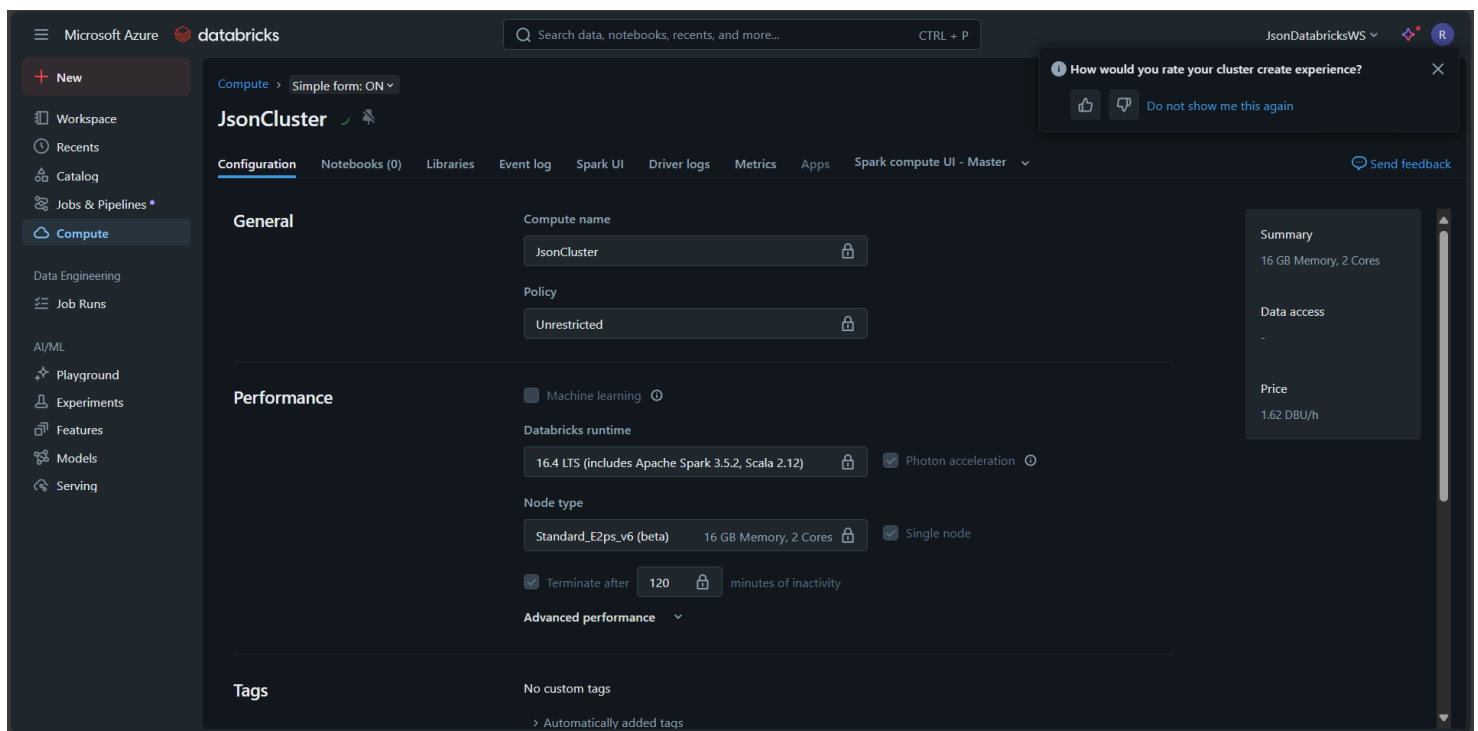


The screenshot shows the Microsoft Azure portal interface. The breadcrumb navigation indicates the path: Home > Resource groups > JsonToParquetRG_Celebal > jsonstorage2025 | Containers > jsondata. The 'jsondata' container overview is displayed, showing the authentication method as 'Access key' and a search bar for blobs. A table lists the contents of the container:

	Name	Last modified	Access tier	Blob type	Size	Lease state
<input type="checkbox"/>	._azuretmpfolder\$	26/7/2025, 12:45:29 pm				...
<input type="checkbox"/>	output	26/7/2025, 12:48:36 pm				...
<input type="checkbox"/>	data.json	26/7/2025, 12:05:07 pm	Hot (Inferred)	Block blob	944 B	Available

4. Created an Azure Databricks Workspace

- Name: JsonWorkspace2025



The screenshot shows the Azure Databricks workspace configuration page for a cluster named 'JsonCluster'. The 'General' tab is selected, showing the following configuration:

- Compute name:** JsonCluster
- Policy:** Unrestricted
- Performance:**
 - ☐ Machine learning
 - Databricks runtime:** 16.4 LTS (includes Apache Spark 3.5.2, Scala 2.12)
 - ☒ Photon acceleration
 - Node type:** Standard_E2ps_v6 (beta) 16 GB Memory, 2 Cores
 - ☒ Single node
 - ☒ Terminate after 120 minutes of inactivity
- Advanced performance:** (collapsed)
- Tags:** No custom tags

A summary sidebar on the right shows the cluster configuration: 16 GB Memory, 2 Cores, Data access, and Price (1.62 DBU/h).

5. Created a Databricks Cluster

- Name: JsonCluster

6. Created a Notebook

- Name: Flatten_JSON_Notebook

2. Access Storage from Databricks

1. Generated a SAS Token or Access Key for authentication
2. Mounted the ADLS Gen2 container

The screenshot shows the Databricks interface with a notebook titled 'MountJson'. The code in the notebook is as follows:

```
dbutils.fs.mount(  
    source = "wasbs://jsondata@jsonstorage2025.blob.core.windows.net",  
    mount_point = "/mnt/jsondata",  
    extra_configs = {  
        "fs.azure.account.key.jsonstorage2025.blob.core.windows.net":  
            "pmb17cQhVDFV6k4X04avjGZ5GDsAmQbqxdPFTIWSOn9ruin5jJhkgG1PsEdheoEz9rHtAV405e09M+AST37ck5w=="  
    })  
)
```

The output of the code is 'True'. Below the code cell, there are instructions: '[Shift+Enter] to run and move to next cell', '[Ctrl+Shift+P] to open the command palette', and '[Esc H] to see all keyboard shortcuts'.

3. Flatten the JSON Structure

Used pyspark.sql.functions to flatten the nested fields:

The screenshot shows the Databricks interface with a notebook titled 'FlattenJSON'. The code in the notebook is as follows:

```
"name",  
"email",  
"street",  
"city",  
"state",  
"zip",  
"order_id",  
"order_date",  
col("item.item_id").alias("item_id"),  
col("item.product").alias("product"),  
col("item.quantity").alias("quantity"),  
col("item.price").alias("price")  
)  
  
final_df.show(truncate=False)
```

The output of the code is a table with 11 columns: customer_id, name, email, street, city, state, zip, order_id, order_date, item_id, product, quantity, price. The table contains 3 rows of data.

customer_id	name	email	street	city	state	zip	order_id	order_date	item_id	product	quantity	price
101	Rahul Gupta	rahul.gupta@example.com	123 Sector 17	Chandigarh	Punjab	160017	ORD001	2025-06-15	A1	Laptop	1	75000
101	Rahul Gupta	rahul.gupta@example.com	123 Sector 17	Chandigarh	Punjab	160017	ORD001	2025-06-15	A2	Mouse	2	700
101	Rahul Gupta	rahul.gupta@example.com	123 Sector 17	Chandigarh	Punjab	160017	ORD002	2025-06-30	B1	Keyboard	1	1200