Delta Tables: Understanding the Two Patterns in Fabric Spark for creating Delta Tables

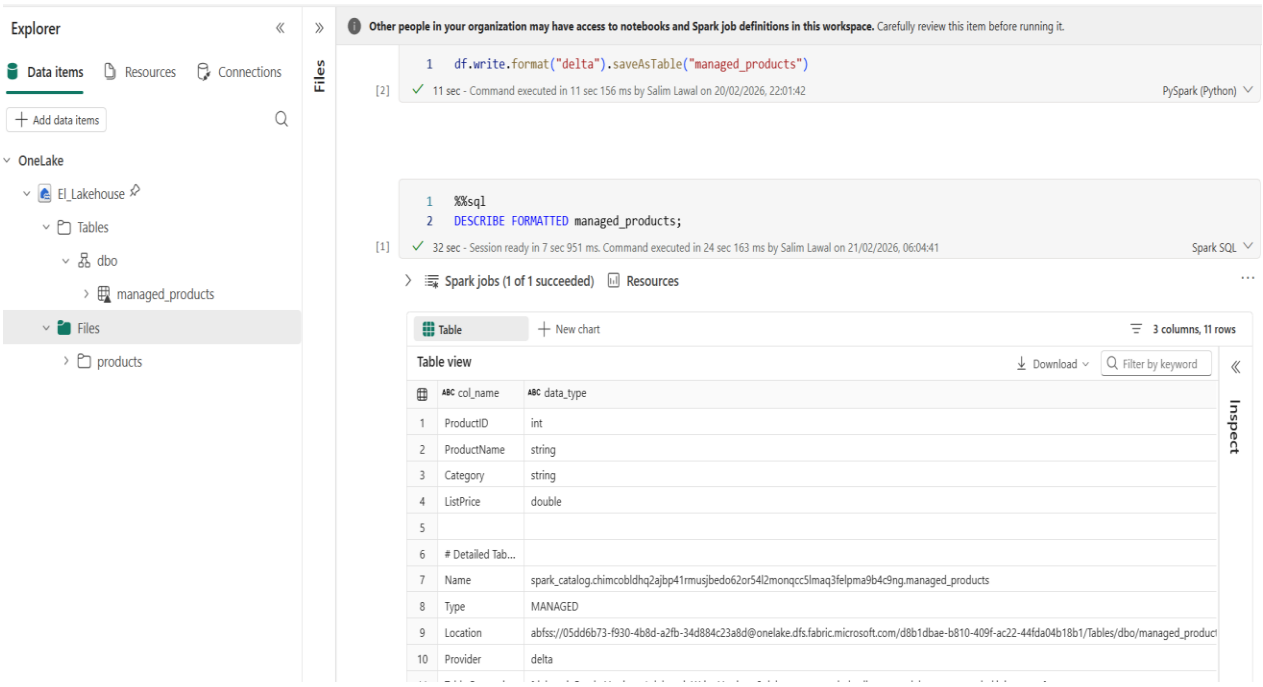
1. Save as a table (MANAGED table only)

Example

df.write.format("delta").saveAsTable("managed_products")

Result

- A managed table is created in the Lakehouse metastore
- Delta files are stored under the default managed table location



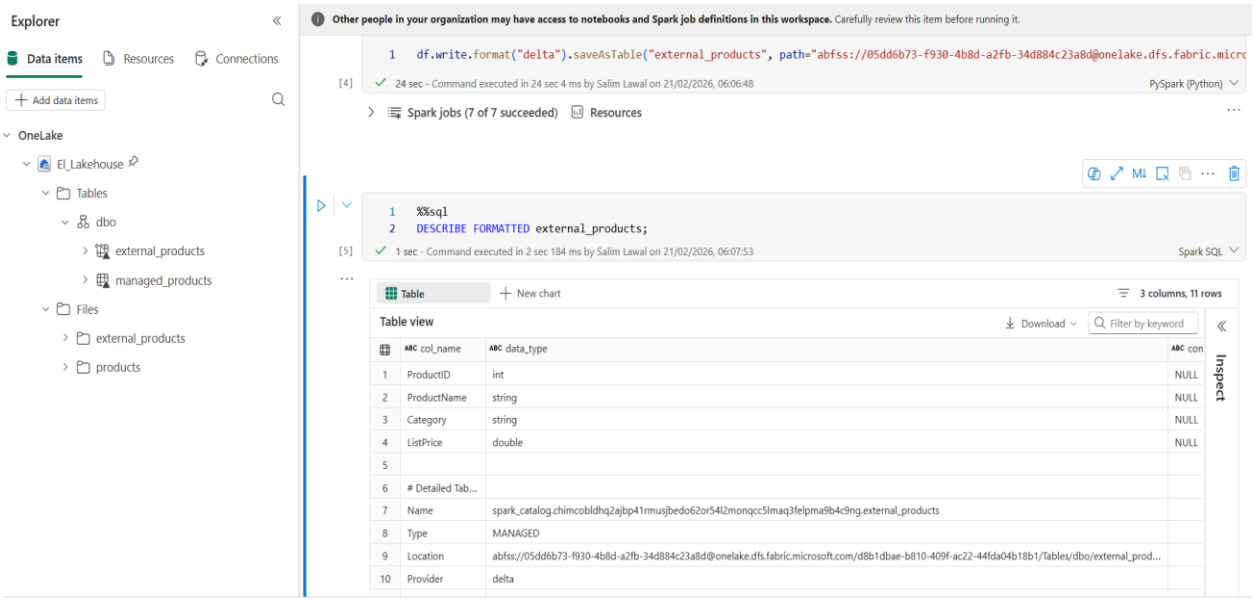
2. Save as a table and write Delta files to specified path (MANAGED table and Delta Files)

Example

df.write.format("delta").saveAsTable("external_products",
path="abfs_path/external_products")

Result

- A managed table is created in the Lakehouse metastore
- Delta files are stored under the default managed table location
- Executes the write operation and writes Delta files to the path provided



Fabric does NOT allow external tables created via Spark. It ignores the path= argument in saveAsTable() and does not link the table to the folder created in the specified path. Whereas, in **Databricks** it works. Databricks interprets this as:

- Create a metastore table
- Store its data on the custom path
- → Result, TYPE: external table

Eventhough the Microsoft Learn says external tables should be created this way.

Create an external table

You can also create external tables, which may be stored somewhere other than the lakehouse, with the schema metadata stored in the lakehouse.

1. In the Explorer pane, in the ... menu for the **Files** folder, select **Copy ABFS path**. The ABFS path is the fully qualified path to the lakehouse Files folder.
2. In a new code cell, paste the ABFS path. Add the following code, using cut and paste to insert the abfs_path into the correct place in the code:

Code Copy

```
df.write.format("delta").saveAsTable("external_products", path="abfs_path/external_products")
```

3. The full path should look similar to this:

Code Copy

```
%%sql
DESCRIBE FORMATTED external_products;
```

4. Run the cell and in the results, view the Location property for the table. Widen the Data type column to see the full path and notice that the OneLake storage locations ends with /Files/external_products.

Why Fabric behaves differently?

Fabric Spark enforces:

- **Managed tables only** inside a Lakehouse
- No external tables is created via Spark
- No custom locations for Spark tables

This is because Fabric uses a **unified metastore** and tightly controls table locations for governance, lineage, and OneLake consistency.