Lab 4

Marzo 2024

Power Platform
Bootcamp

Microsoft Fabric

Trabajando con Datos de 0 a 100

Power Platform Madrid

Contenido

Estructura del Documento

El laboratorio incluye pasos que el usuario debe seguir, junto con capturas de pantalla asociadas que proporcionan ayuda visual. En cada captura de pantalla, las secciones se resaltan con cuadros naranjas para indicar las áreas en las que el usuario debe centrarse.

Introducción

En este laboratorio se trabajará sobre varias características clave de Microsoft Fabric. Se trata de un taller introductorio destinado a presentarte las distintas experiencias de productos y artefactos disponibles en Fabric. Al final de este taller, aprenderá a usar **Lakehouse**, **Datawarehouse**, **Dataflow Gen2, Data Pipeline** y la función **DirectLake**.

Al final de este laboratorio, se habrá aprendido:

1. Crear y Describir los diferentes tipos de Tablas Delta
2. Utilizar la Característica de Time Travel

Requisitos previos

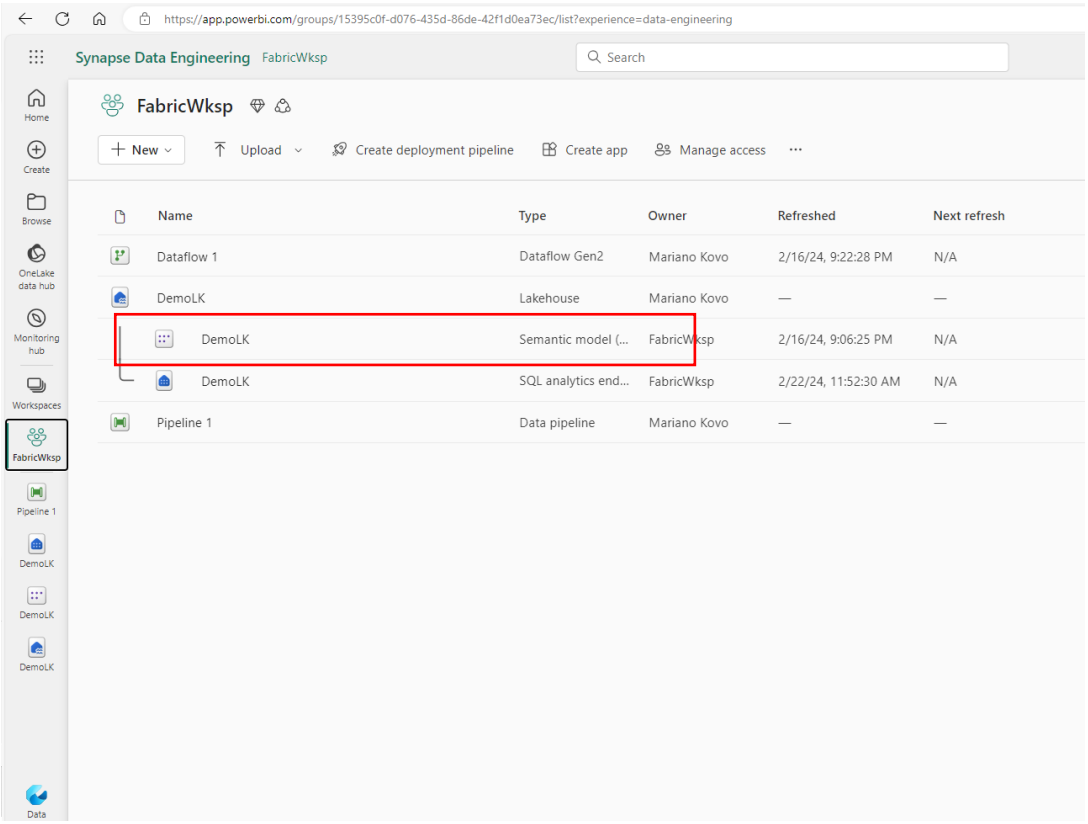
- Haber completado los pasos de los Lab1 y Lab2, ya que este tutorial requiere tener un Workspace con Lakehouse ya creado previamente.

Trabajando con Tablas Delta

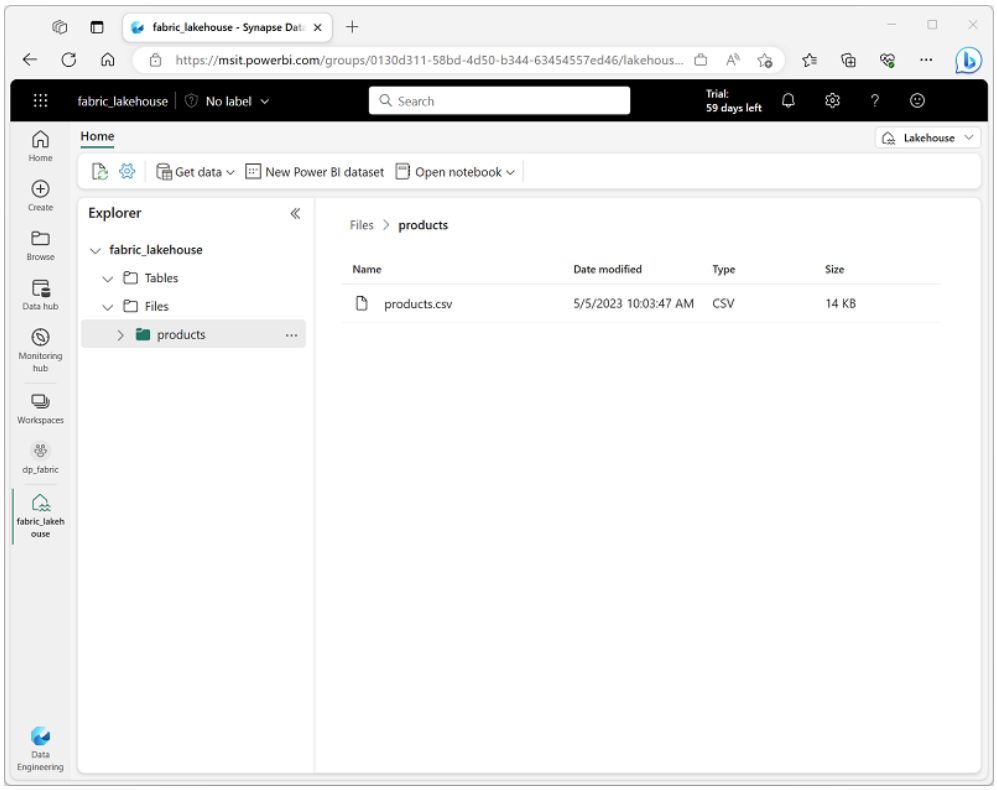
Tarea 1: Copiar datos a una Tabla desde un Archivo CSV

Las tablas de un lakehouse de Microsoft Fabric se basan en el formato *Delta Lake de código abierto* para Apache Spark. Delta Lake agrega compatibilidad con la semántica relacional para las operaciones de datos por lotes y de streaming, y permite la creación de una arquitectura de Lakehouse en la que se puede usar Apache Spark para procesar y consultar datos en tablas basadas en archivos subyacentes de un Lakehouse.

1. En la página principal de Microsoft Fabric en https://app.fabric.microsoft.com, seleccione **Ingeniería de datos de Synapse**.
2. En la barra de menú de la izquierda, seleccione **Workspaces** y seleccione el Workspace de trabajo creado previamente.
3. Seleccione luego el **Lakehouse** creado en este Workspace.



4. Descargue el archivo de datos de este ejercicio desde https://github.com/MicrosoftLearning/dp-data/raw/main/products.csv y guárdelo como **products.csv en** el equipo local (o en la máquina virtual del laboratorio, si procede).
5. Regrese a la pestaña del navegador web que contiene su lakehouse y, en el **...** en la carpeta **Archivos** del panel **Explorador**, seleccione **Nueva subcarpeta** y cree una carpeta denominada **productos**.
6. En el **...**en la carpeta de productos, seleccione **Cargar y Cargar archivos y,** a continuación, cargue el **archivo products.csv desde el equipo local (o la máquina virtual de laboratorio, si procede) en el lakehouse.**
7. Una vez cargado el archivo, seleccione la carpeta **de productos** y compruebe que se ha cargado el **archivo products.csv**, como se muestra aquí:



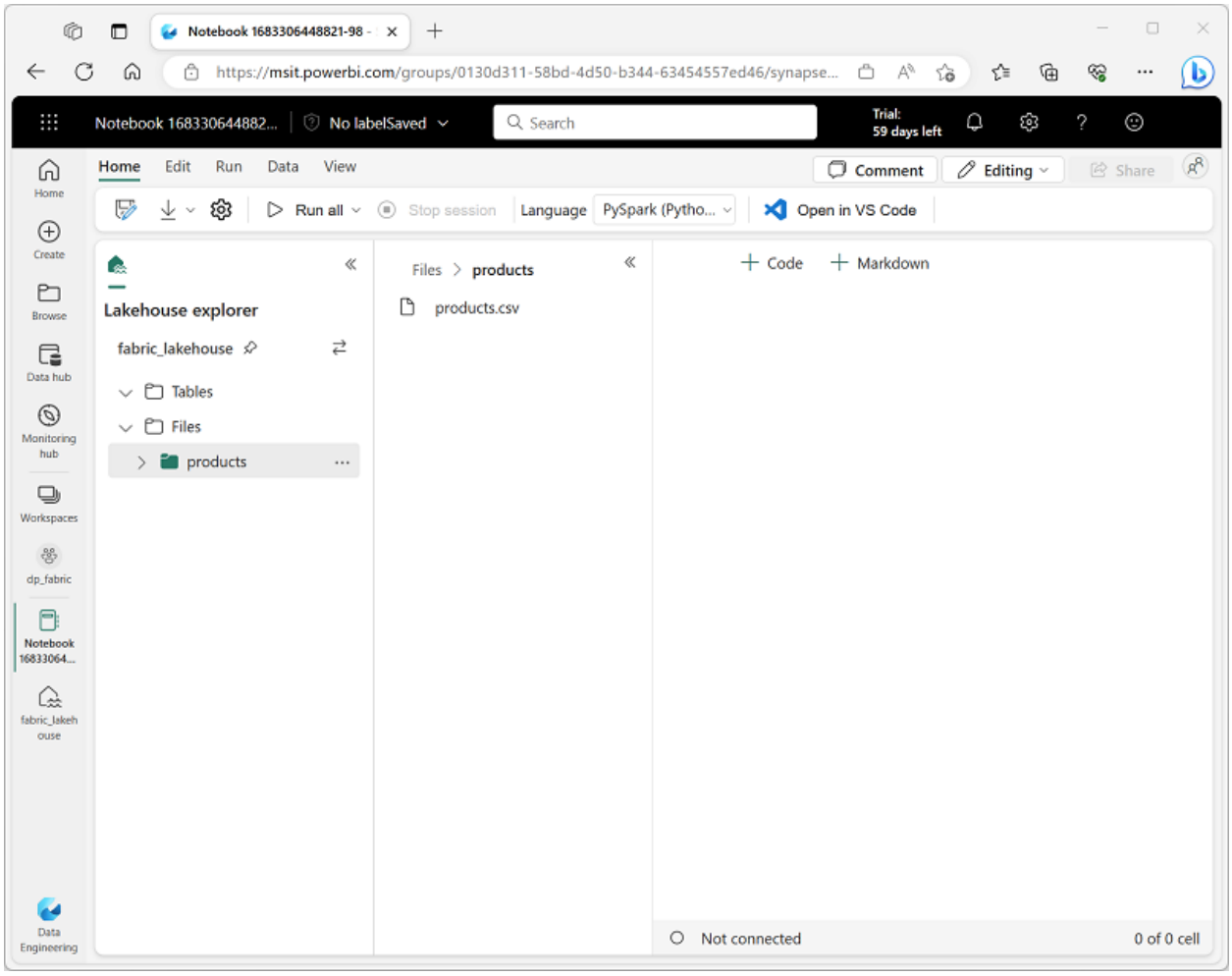
Tarea 2: Explorar los datos en un dataframe

1. En la página **principal**, mientras visualiza el contenido de la **carpeta products** en el lago de datos, en el menú Abrir bloc de notas **, seleccione** Nuevo Notebook**.**

   Después de unos segundos, se abrirá un nuevo cuaderno que contiene una sola *celda*. Los blocs de notas se componen de una o varias celdas que pueden contener *código* o *markdown* (texto con formato).

2. Seleccione la celda existente en el bloc de notas, que contiene un código simple, y luego use su 🗑 icono (*Eliminar*) en la parte superior derecha para eliminarla, no necesitará este código.

3. En el panel del **explorador de Lakehouse** de la izquierda, expanda **Archivos** y seleccione **productos** para mostrar un nuevo panel que muestre el **archivo products.csv** que cargó anteriormente:



4. En el **...** para **products.csv**, seleccione **Cargar datos** > **Spark**. Se debe agregar al Notebook una nueva celda de código que contenga el siguiente código:

```
df = spark.read.format("csv").option("header","true").load("Files/products/products.csv")
# df now is a Spark DataFrame containing CSV data from "Files/products/products.csv".
display(df)
```

Consejo: Puede ocultar el panel que contiene los archivos a la izquierda utilizando su **icono «**. Si lo haces, te ayudará a concentrarte en el Notebook.

5. Utilice el **botón ▷** (*Ejecutar celda*) a la izquierda de la celda para ejecutarla.

Nota: Dado que es la primera vez que ejecuta código de Spark en este Notebook, se debe iniciar una sesión de Spark. Esto significa que la primera ejecución puede tardar aproximadamente un minuto en completarse. Las ejecuciones posteriores serán más rápidas.

6. Cuando se haya completado el comando de la celda, revise el resultado debajo, que debería tener un aspecto similar al siguiente:

Index	ProductID	ProductName	Category	ListPrice
1	771	Mountain-100 Silver, 38	Mountain Bikes	3399.9900
2	772	Mountain-100 Silver, 42	Mountain Bikes	3399.9900
3	773	Mountain-100 Silver, 44	Mountain Bikes	3399.9900
…	…	…	…	…

Tarea 3: Crear Tablas Delta

Puede guardar el DataFrame como una tabla delta mediante el método saveAsTable. Delta Lake admite la creación de tablas *administradas* y *externas*.

Puede guardar el marco de datos como una tabla delta mediante el método saveAsTable. Delta Lake admite la creación de tablas *administradas* y *externas*.

Creación de una tabla administrada

Las tablas administradas son tablas para las que Fabric administra tanto los metadatos del esquema como los archivos de datos. Los archivos de datos de la tabla se crean en la **carpeta Tablas**.

1. En los resultados devueltos por la primera celda de código, use el icono **+ Código** para agregar una nueva celda de código si aún no existe una.
   Consejo: Para ver el icono **+ Código**, mueva el ratón justo debajo y a la izquierda de la salida de la celda actual. Como alternativa, en la barra de menús, en la pestaña **Editar**, seleccione **+ Agregar celda de código**.

2. Copie el siguiente código en la nueva celda y ejecútelo:

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

3. En el panel explorador **de Lakehouse**, en el **...** en la carpeta **Tablas**, seleccione **Actualizar**. A continuación, expanda el nodo Tablas y compruebe que se **ha creado la tabla** managed_products.

Crear una *tabla* externa

También puede crear tablas *externas* para las que los metadatos del esquema estén definidos en el metastore de lakehouse, pero los archivos de datos se almacenen en una ubicación externa.

4. Agregue otra celda de código nueva y agréguele el siguiente código:

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

5. En el panel Explorador de **Lakehouse**, en el ... en la carpeta **Archivos**, seleccione **Copiar ruta ABFS**.

   La ruta ABFS es la ruta completa a la carpeta **Archivos** en el almacenamiento de OneLake para su lakehouse, similar a esta:

   abfss://workspace@tenant-onelake.dfs.fabric.microsoft.com/lakehousename.Lakehouse/Files

6. En el código que ha introducido en la celda de código, reemplace **abfs_path** por la ruta de acceso que copió en el portapapeles para que el código guarde el marco de datos como una tabla externa con archivos de datos en una carpeta denominada **external_products** en la ubicación de la carpeta **Archivos**. La ruta completa debería tener un aspecto similar al siguiente:

   abfss://workspace@tenant-onelake.dfs.fabric.microsoft.com/lakehousename.Lakehouse/Files/external_products

7. En el panel Explorador de **Lakehouse**, en el ... en la carpeta **Tablas**, seleccione **Actualizar**. A continuación, expanda el nodo **Tablas** y compruebe que se **ha creado la tabla** external_products.
8. En el panel Explorador de **Lakehouse**, en el ... en la carpeta **Archivos**, seleccione **Actualizar**. A continuación, expanda el nodo **Archivos** y compruebe que se ha creado la carpeta external_products para los archivos de datos de la tabla.

Comparación de *tablas administradas* y *externas*

Exploremos las diferencias entre las tablas administradas y las externas.

9. Agregue otra celda de código y ejecute el siguiente código:

```
%%sql

DESCRIBE FORMATTED managed_products;
```

En los resultados, vea la propiedad **Location de la** tabla, que debe ser una ruta de acceso al almacenamiento de OneLake para la casa de lago que termina en **/Tables/managed_products** (es posible que tenga que ampliar **la columna Tipo de datos** para ver la ruta de acceso completa).

10. Modifique el comando DESCRIBE para mostrar los detalles de la tabla **external_products** como se muestra aquí:

```
%%sql

DESCRIBE FORMATTED external_products;
```

En los resultados, vea la propiedad **Location de la** tabla, que debe ser una ruta de acceso al almacenamiento de OneLake para la casa del lago que termina en **/Files/external_products** (es posible que tenga que ampliar **la columna Tipo de datos** para ver la ruta de acceso completa).

Los archivos de la tabla administrada se almacenan en la **carpeta Tablas** del almacenamiento de OneLake para el lakehouse. En este caso, se ha creado una carpeta denominada **managed_products** para almacenar los archivos Parquet y **delta_log** carpeta de la tabla que ha creado.

11. Agregue otra celda de código y ejecute el siguiente código:

```
%%sql

DROP TABLE managed_products;
DROP TABLE external_products;
```

12. En el panel Explorador de **Lakehouse**, en el ... en la carpeta **Tablas**, seleccione **Actualizar**. A continuación, expanda el nodo **Tablas** y compruebe que no hay tablas en la lista.

13. En el panel del **explorador de Lakehouse**, expanda la carpeta **Archivos** y compruebe que el **external_products** no se ha eliminado. Seleccione esta carpeta para ver los archivos de datos de Parquet y **_delta_log** carpeta de los datos que se encontraban anteriormente en la **tabla external_products**. Se eliminaron los metadatos de la tabla externa, pero los archivos no se vieron afectados.

Usar SQL para crear una tabla

14. Agregue otra celda de código y ejecute el siguiente código:

```
%%sql

CREATE TABLE products
USING DELTA
LOCATION 'Files/external_products';
```

15. En el panel Explorador de **Lakehouse**, en el ... en la carpeta **Tablas**, seleccione **Actualizar**. A continuación, expanda el nodo **Tablas** y compruebe que aparece una nueva tabla denominada **products**. A continuación, expanda la tabla para comprobar que su esquema coincide con el marco de datos original que se guardó en la **carpeta external_products**.

16. Agregue otra celda de código y ejecute el siguiente código:

```
%%sql

SELECT * FROM products;
```

Explorar el control de versiones de tablas

El historial de transacciones de las tablas delta se almacena en archivos JSON en la **carpeta delta_log**. Puede utilizar este registro de transacciones para administrar el control de versiones de datos.

17. Agregue una nueva celda de código al cuaderno y ejecute el código siguiente:

```
%%sql

UPDATE products
SET ListPrice = ListPrice * 0.9
WHERE Category = 'Mountain Bikes';
```

    Este código implementa una reducción del 10% en el precio de las bicicletas de montaña.

18. Agregue otra celda de código y ejecute el siguiente código:

```
%%sql

DESCRIBE HISTORY products;
```

    Los resultados muestran el historial de transacciones registradas para la tabla.

19. Agregue otra celda de código y ejecute el siguiente código:

```
delta_table_path = 'Files/external_products'

# Get the current data
current_data = spark.read.format("delta").load(delta_table_path)
display(current_data)

# Get the version 0 data
original_data = spark.read.format("delta").option("versionAsOf", 0).load(delta_table_path)
display(original_data)
```

Los resultados muestran dos dataframes: uno que contiene los datos después de la reducción de precios y el otro que muestra la versión original de los datos.