Tasca S8.02. Power BI amb Python

Aquesta tasca consisteix en l'elaboració d'un informe de Power BI, aprofitant les capacitats analítiques de Python. S'utilitzaran els scripts de Python creats prèviament en la Tasca 1 per a generar visualitzacions personalitzades amb les biblioteques Seaborn i Matplotlib. Aquestes visualitzacions seran integrades en l'informe de Power BI per a oferir una comprensió més profunda de la capacitat del llenguatge de programació en l'eina Power BI.

Configuració Power BI i connexió amb script de Python

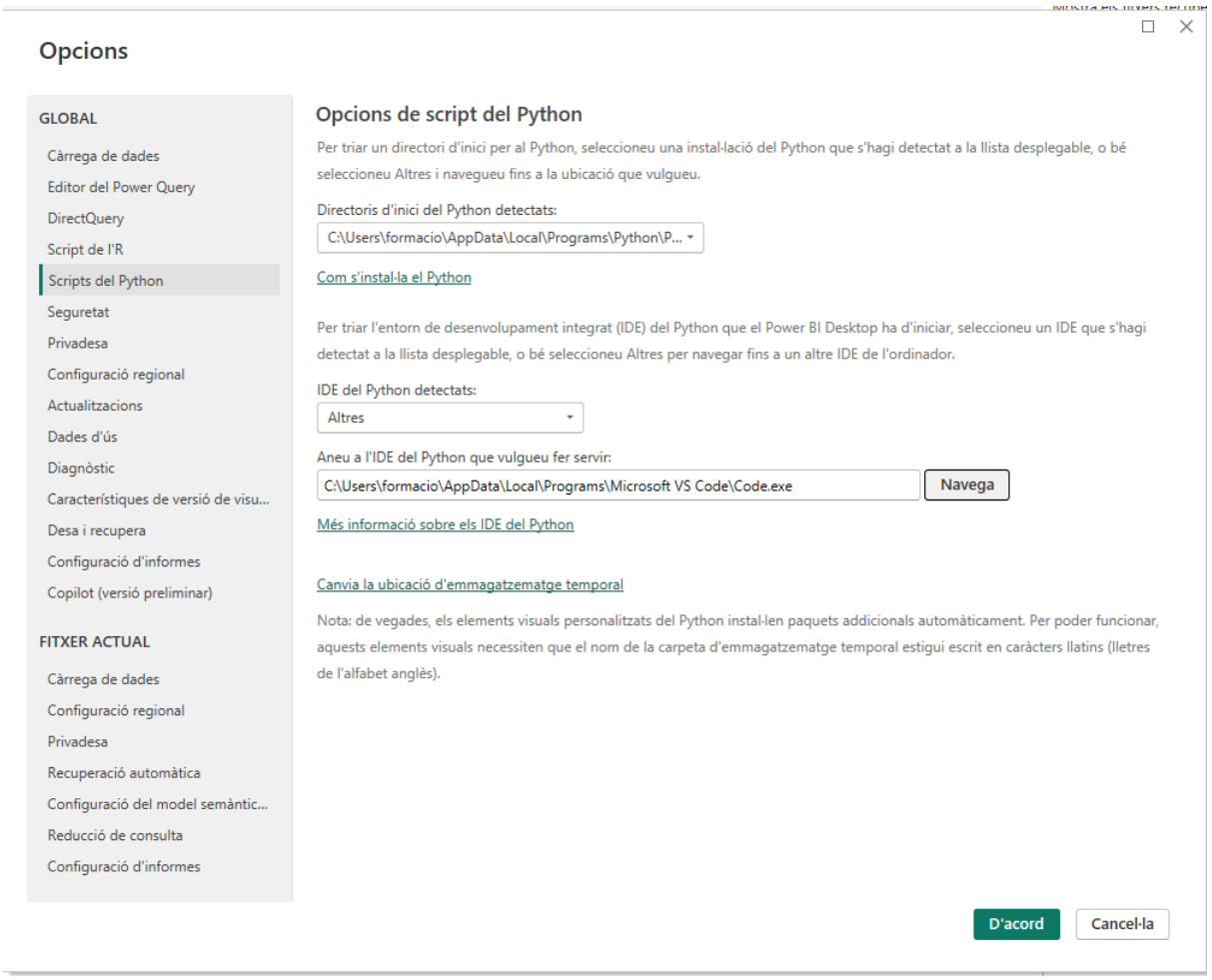
Nos aseguramos de tener las librerías pandas (para el manejo de dataframes), matplotlib y seaborn (para la creación de gráficos) en el equipo local.

En una instancia del terminal ejecuto el Python Launcher y le pido una lista de paquetes instalados mediante pip:

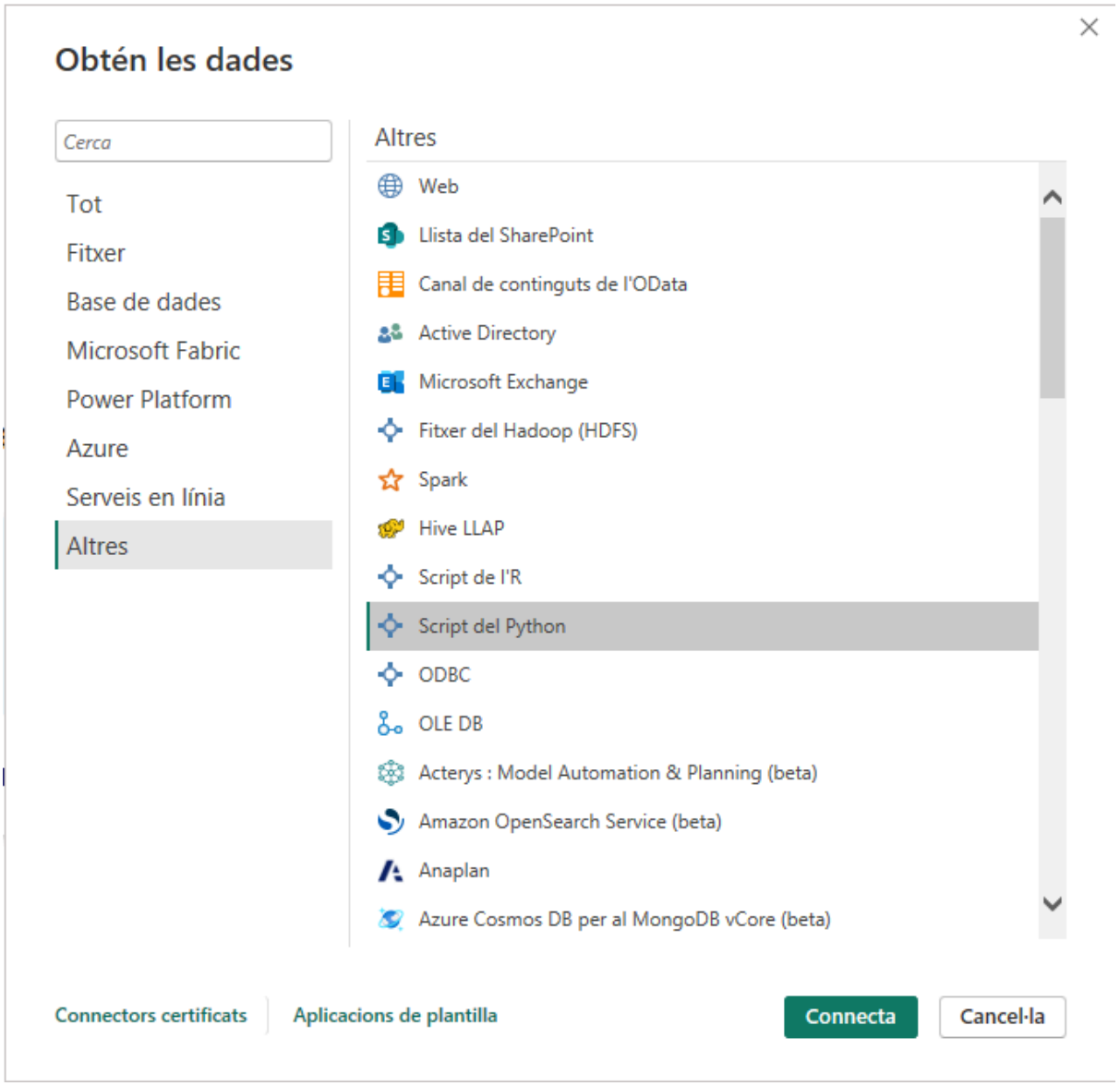
```
PS C:\Users\formacio\Documents\DataAnalyticsAlex> py --list
 -V:3.12 *        Python 3.12 (64-bit)
PS C:\Users\formacio\Documents\DataAnalyticsAlex> py -m pip list
Package                Version
---------------------- -----------
asttokens              2.4.1
certifi                2024.12.14
charset-normalizer     3.4.1
colorama               0.4.6
comm                   0.2.2
contourpy              1.3.0
cycler                 0.12.1
debugpy                1.8.5
decorator              5.1.1
et-xmlfile             1.1.0
executing              2.1.0
fontawesomefree        6.6.0
fonttools              4.53.1
greenlet               3.1.1
highlight-text         0.2
idna                   3.10
ipykernel              6.29.5
ipython                8.27.0
jedi                   0.19.1
jupyter_client         8.6.2
jupyter_core           5.7.2
kiwisolver             1.4.7
matplotlib             3.10.0
matplotlib-inline      0.1.7
mysql-connector-python 9.1.0
nest-asyncio           1.6.0
numpy                  2.1.1
openpyxl               3.1.5
packaging              24.1
panda                  0.3.1
pandas                 2.2.2
parso                  0.8.4
pillow                 10.4.0
```

```
pillow                  10.4.0
pip                     25.0
platformdirs            4.2.2
prompt_toolkit          3.0.47
psutil                  6.0.0
pure_eval               0.2.3
Pygments                2.18.0
pypalettes              0.1.5
pyparsing               3.1.4
python-dateutil         2.9.0.post0
pytz                    2024.1
pywaffle                1.1.1
pywin32                 306
pyzmq                   26.2.0
requests                2.32.3
seaborn                 0.13.2
setuptools              75.8.0
six                     1.16.0
SQLAlchemy              2.0.37
squarify                0.4.4
stack-data              0.6.3
tornado                 6.4.1
traitlets               5.14.3
typing_extensions       4.12.2
tzdata                  2024.1
urllib3                 2.3.0
wcwidth                 0.2.13
PS C:\Users\formacio\Documents\DataAnalyticsAlex>
```

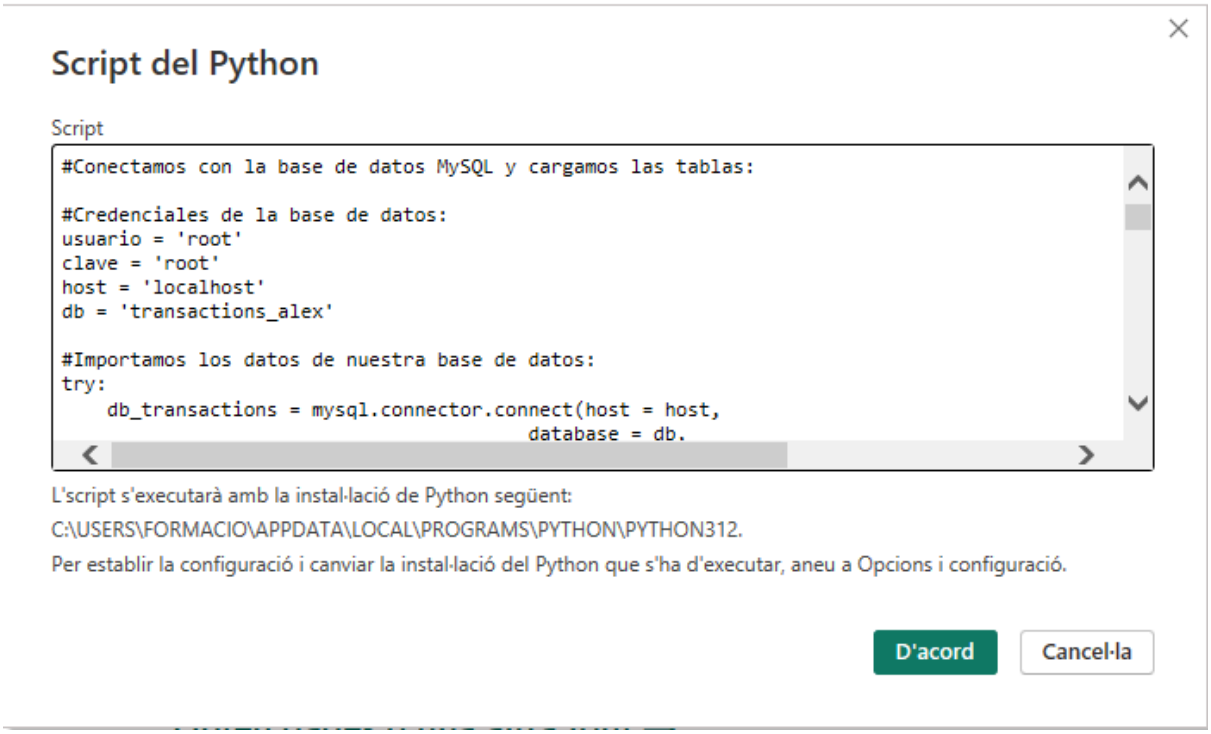
En Power BI, habilitamos la ejecución de scripts de Python:



Escogeremos la opción **Obtener datos / Script de Python**:

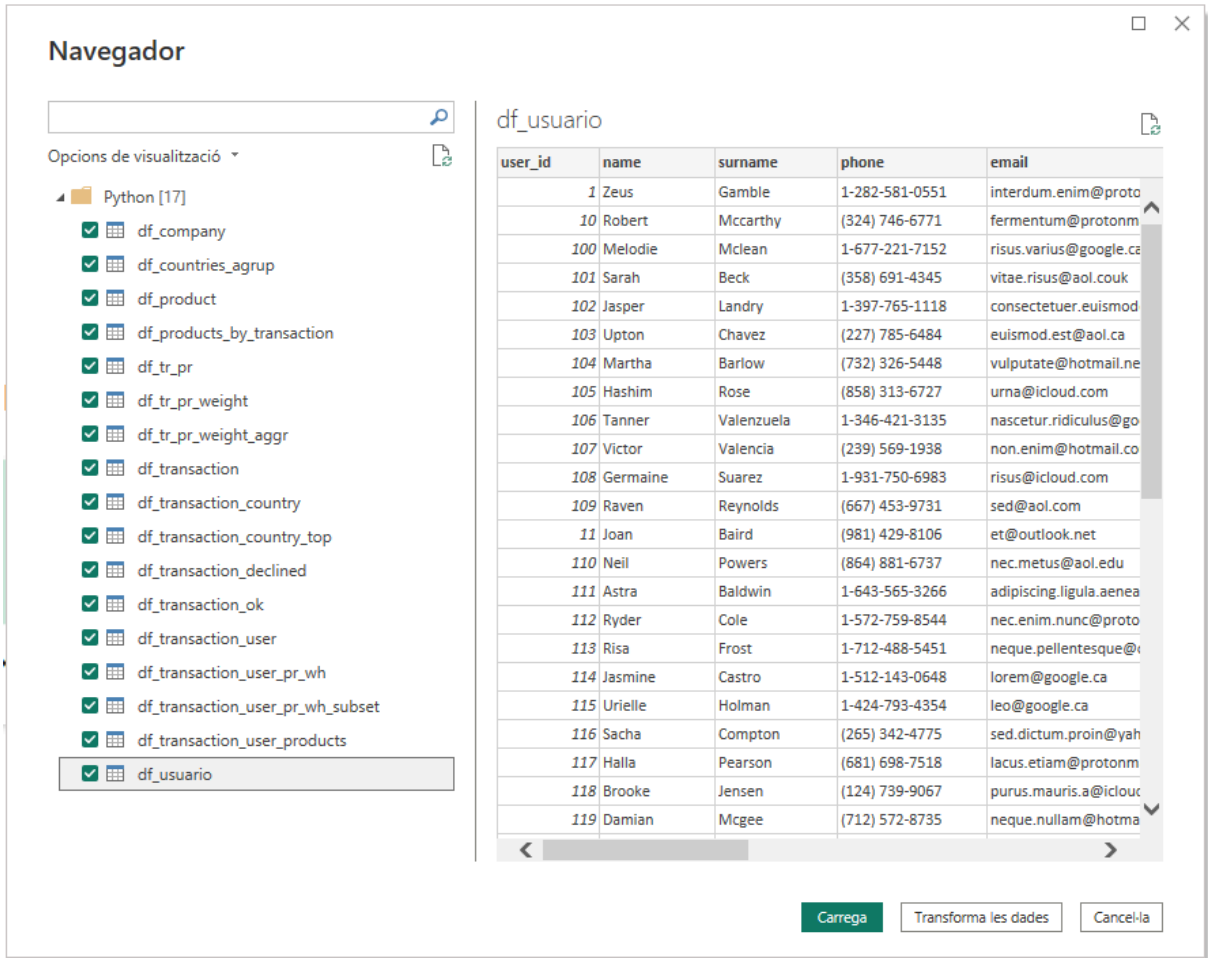


Y en el cuadro habilitado para el script cargaremos los dataframes que hemos creado en el script de la TascaS801, tanto los creados a partir de la base de datos **transaction_alex** que importamos a través de MySQL Connector para Python como los dataframes creados a partir de la función merge() de la librería pandas con las transformaciones pertinentes.



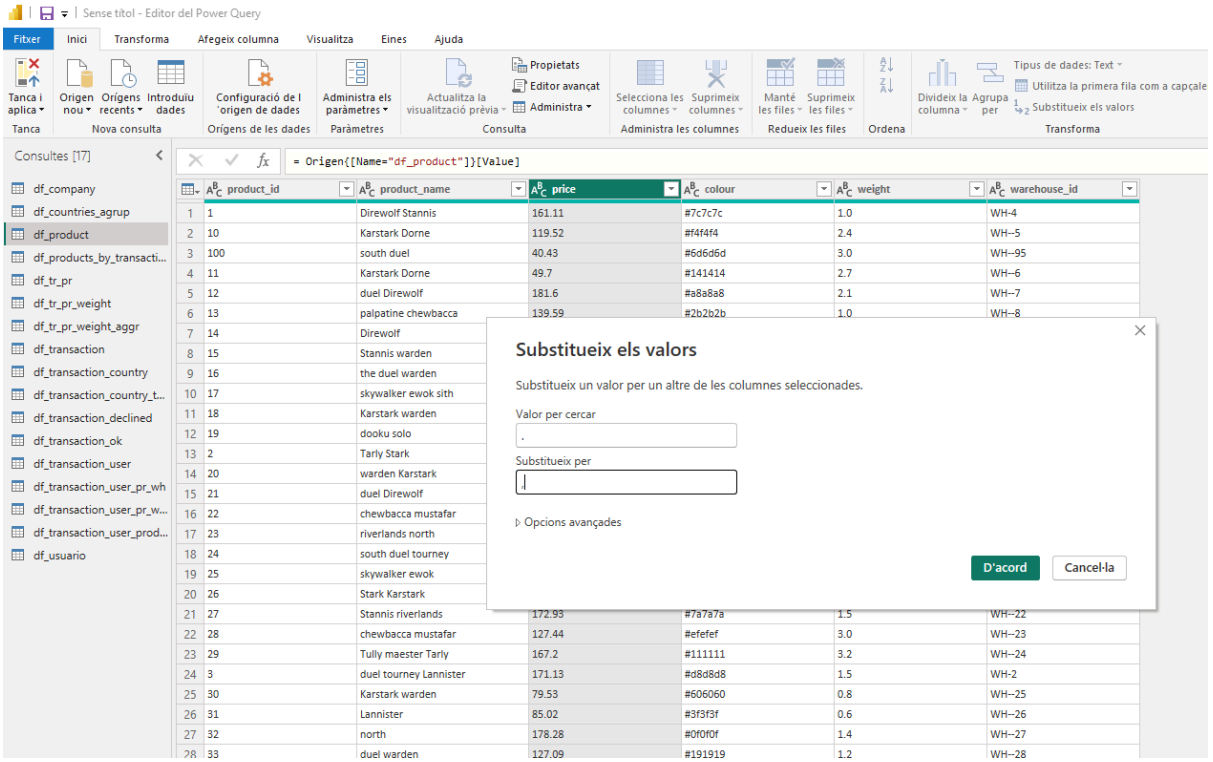
(Se puede ver el script completo en la primera celda del archivo **TascaS802.ipynb**.)

Una vez cargados los dataframes, pasamos a revisar los datos y realizar las transformaciones pertinentes:



Las columnas **price** y **weight** de los dataframes **df_product, df_tr_pr_weight** aparecen sin el separador de decimales, así como la columna **amount** en los dataframes **df_transaction, df_transaction_country, df_transaction_country_top, df_transaction_ok, df_transaction_user, df_transaction_user_pr_wh** y **df_transaction_user_pr_wh_subset**. En estos dos últimos dataframes, fruto de la unión con la tabla product, vemos que el resultado de la agregación multiplica los resultados en el script, a causa de la desaparición del separador de decimales.

Procedemos a transformar estos datos antes de cargar los dataframes. Por ejemplo, en el caso de la columna **price**, eliminamos la transformación automática de texto a número decimal, sustituimos el signo '.' por ',' y transformamos a decimal fijo. También eliminaremos la conversión de algunos id que Power BI ha transformado de string a número entero.



Nivell 1

- Exercici 1

Els 7 exercicis del nivell 1 de la tasca 01

En las representaciones de las gráficas de los tres niveles seguiremos el mismo mecanismo: seleccionamos **Elemento visual de Python**, seleccionamos en la columna **Datos** el dataframe que contiene la(s) columna(s) con los datos que queremos graficar, los añadimos al campo **Datos** de la pestaña **Compilación** y, a continuación, copiamos y adaptamos el script correspondiente del archivo **Tasca S801.ipynb** en el recuadro inferior para generar el gráfico en Power BI.

En los siguientes apartados incluiré solo las acciones adicionales al mecanismo descrito aquí arriba. Los scripts adaptados se pueden consultar en el archivo **Tasca S802.ipynb**

Nota: como aprovechaba la característica de Visual Studio Code de interpretar los objetos de matplotlib y seaborn, a los scripts les añado las instrucciones **plt.tight_layout()** para evitar que la gráfica aparezca cortada en el contenedor de Power BI, y **plt.show()** para poder representarlos en Power BI.

Ejercicio 1. Gráfica con una variable numérica

He sobreimpresionado la leyenda sobre el gráfico para mejorar la escala.

Ejercicio 5. Gráfica con dos variables categóricas

Al importar los dataframes a Power BI, este asigna el formato **Fecha** a la columna **period** y le asigna una jerarquía de fechas. Vuelvo a Power Query y elimino la conversión de formato; así mantengo la columna **period** con formato fecha. Sin embargo, al ejecutar el script original de la tarea S08.01, el orden del eje x es aleatorio: para solucionarlo, antes de mandar el dataset al método **lineplot**, fuerzo el orden (alfabético) de la columna **period**.