

## 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

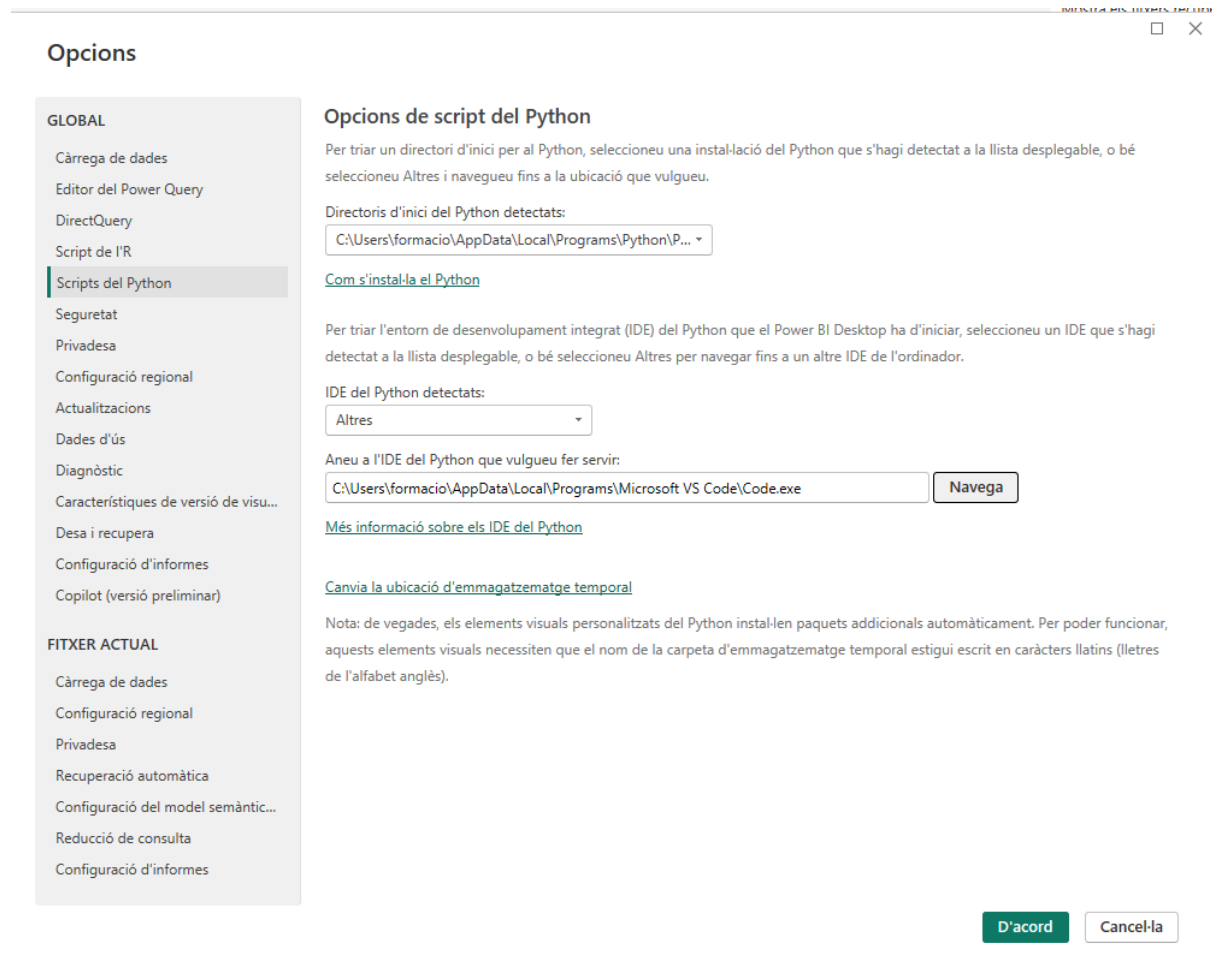
Nos aseguramos de tener las librerías pandas (para el manejo de dataframes), matplotlib y seaborn (para la creación de gráficos) en el equipo local.

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

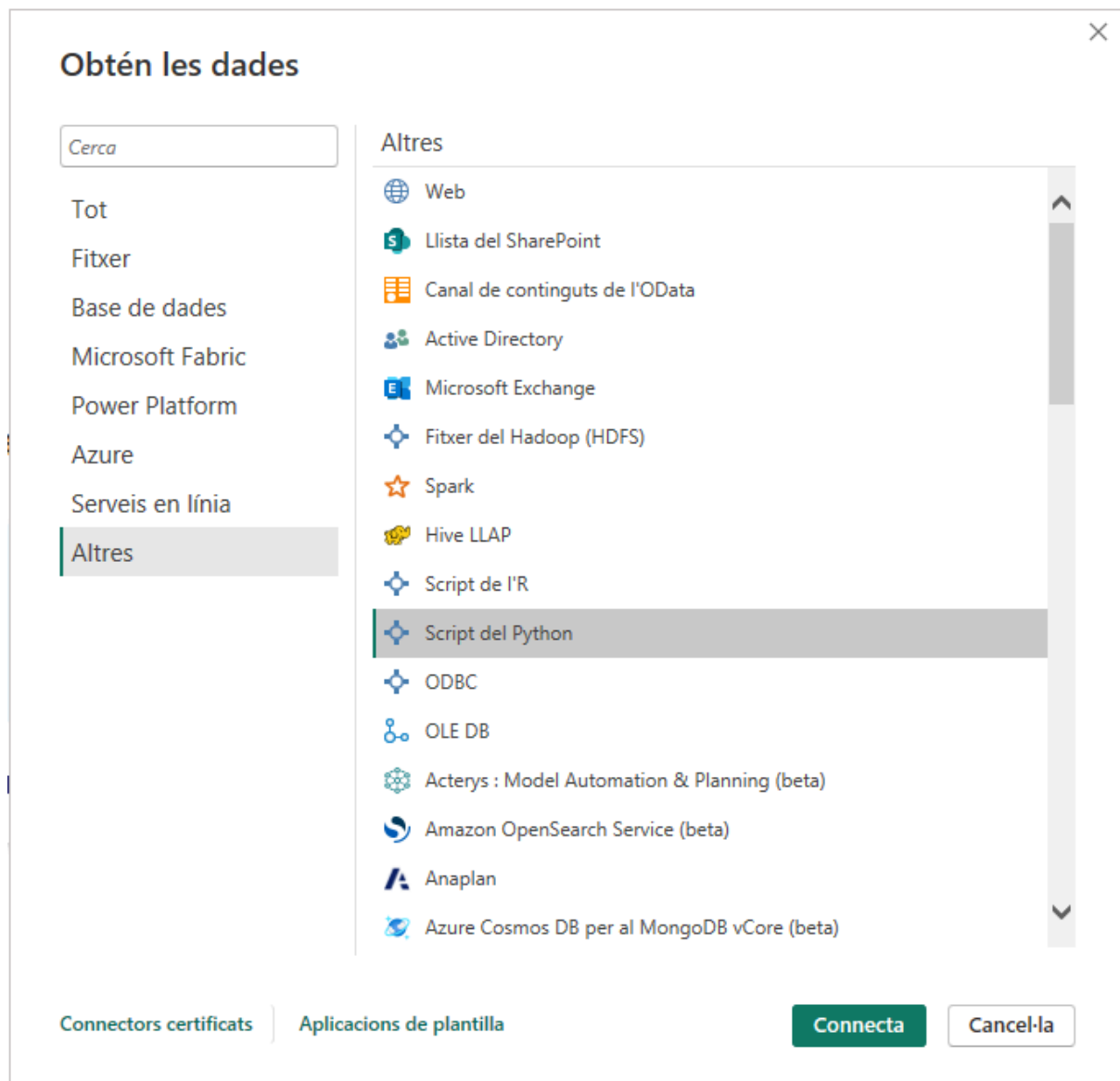
```
PS C:\Users\formacio\Documents\DataAnalyticsAlex> py --list
-V:3.12 *          Python 3.12 (64-bit)
PS C:\Users\formacio\Documents\DataAnalyticsAlex> py -m pip list
Package           Version
-----
asttokens         2.4.1
certifi           2024.12.14
charset-normalizer 3.4.1
colorama          0.4.6
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>
```

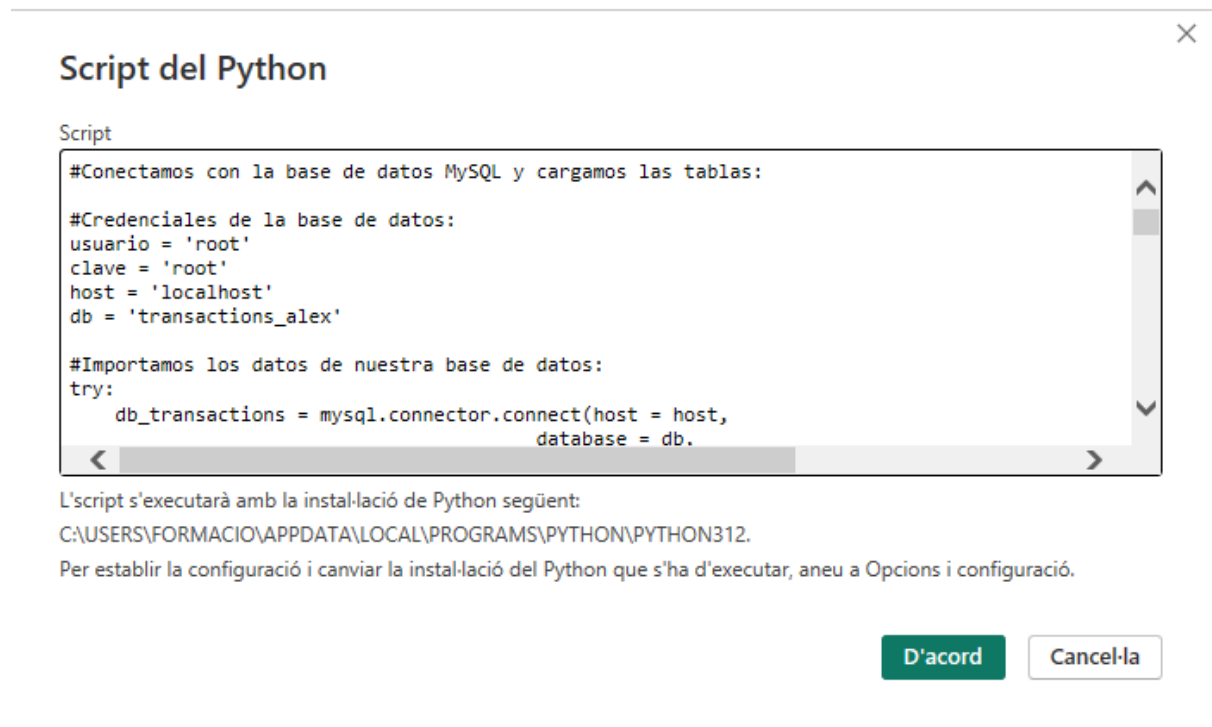
En Power BI, habilitamos la ejecución de scripts de Python:



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:

### Navegador

Opcions de visualització ▾

Python [17]

df\_company

df\_countries\_agrup

df\_product

df\_products\_by\_transaction

df\_tr\_pr

df\_tr\_pr\_weight

df\_tr\_pr\_weight\_aggr

df\_transaction

df\_transaction\_country

df\_transaction\_country\_top

df\_transaction\_declined

df\_transaction\_ok

df\_transaction\_user

df\_transaction\_user\_pr\_wh

df\_transaction\_user\_pr\_wh\_subset

df\_transaction\_user\_products

df\_usuario

df\_usuario

user_id	name	surname	phone	email
1	Zeus	Gamble	1-282-581-0551	interdum.enim@proto
10	Robert	Mccarthy	(324) 746-6771	fermentum@protonm
100	Melodie	Mclean	1-677-221-7152	risus.varius@google.ca
101	Sarah	Beck	(358) 691-4345	vitae.risus@aol.co.uk
102	Jasper	Landry	1-397-765-1118	consectetur.euismod
103	Upton	Chavez	(227) 785-6484	euismod.est@aol.ca
104	Martha	Barlow	(732) 326-5448	vulputate@hotmail.ne
105	Hashim	Rose	(858) 313-6727	urna@icloud.com
106	Tanner	Valenzuela	1-346-421-3135	nascetur.ridiculus@go
107	Victor	Valencia	(239) 569-1938	non.enim@hotmail.co
108	Germaine	Suarez	1-931-750-6983	risus@icloud.com
109	Raven	Reynolds	(667) 453-9731	sed@aol.com
11	Joan	Baird	(981) 429-8106	et@outlook.net
110	Neil	Powers	(864) 881-6737	nec.metus@aol.edu
111	Astra	Baldwin	1-643-565-3266	adipiscing.ligula.aenea
112	Ryder	Cole	1-572-759-8544	nec.enim.nunc@proto
113	Risa	Frost	1-712-488-5451	neque.pellentesque@t
114	Jasmine	Castro	1-512-143-0648	lorem@google.ca
115	Urielle	Holman	1-424-793-4354	leo@google.ca
116	Sacha	Compton	(265) 342-4775	sed.dictum.proin@yah
117	Halla	Pearson	(681) 698-7518	lacus.etiam@protonm
118	Brooke	Jensen	(124) 739-9067	purus.mauris.a@icloud
119	Damian	Mcgee	(712) 572-8735	neque.nullam@hotma

Carrega

Transforma les dades

Cancel·la

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.

Consultes [17]

df\_company  
df\_countries\_agrup  
df\_product  
df\_products\_by\_transacti...  
df\_tr\_pr  
df\_tr\_pr\_weight  
df\_tr\_pr\_weight\_aggr  
df\_transaction  
df\_transaction\_country  
df\_transaction\_country\_L...  
df\_transaction\_declined  
df\_transaction\_ok  
df\_transaction\_user  
df\_transaction\_user\_pr\_wh  
df\_transaction\_user\_pr\_w...  
df\_transaction\_user\_prod...  
df\_usuario

Substitueix els valors

Substitueix un valor per un altre de les columnes seleccionades.

Valor per cercar

Substitueix per

Opcions avançades

D'acord Cancel·la

A <sup>B</sup> product_id	A <sup>B</sup> product_name	A <sup>B</sup> price	A <sup>B</sup> colour	A <sup>B</sup> weight	A <sup>B</sup> warehouse_id
1	Direwolf Stannis	161.11	#7c7c7c	1.0	WH-4
2	Karstark Dorne	119.52	#4f4f4f	2.4	WH-5
3	100 south duel	40.43	#6d6d6d	3.0	WH-95
4	11 Karstark Dorne	49.7	#141414	2.7	WH-6
5	12 duel Direwolf	181.6	#8a8a8a	2.1	WH-7
6	13 palpatine chewbacca	139.59	#2b2b2b	1.0	WH-8
7	14 Direwolf				
8	15 Stannis warden				
9	16 the duel warden				
10	17 skywalker ewok sith				
11	18 Karstark warden				
12	19 dooku solo				
13	2 Tarly Stark				
14	20 warden Karstark				
15	21 duel Direwolf				
16	22 chewbacca mustafar				
17	23 riverlands north				
18	24 south duel tourney				
19	25 skywalker ewok				
20	26 Stark Karstark				
21	27 Stannis riverlands	172.93	#787878	1.5	WH-22
22	28 chewbacca mustafar	127.44	#efefef	3.0	WH-23
23	29 Tully maester Tarly	167.2	#111111	3.2	WH-24
24	3 duel tourney Lannister	171.13	#d8d8d8	1.5	WH-2
25	30 Karstark warden	79.53	#606060	0.8	WH-25
26	31 Lannister	85.02	#3f3f3f	0.6	WH-26
27	32 north	178.28	#0f0f0f	1.4	WH-27
28	33 duel warden	127.09	#191919	1.2	WH-28

## 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**.