# Tasca S8.02. Power BI amb Python

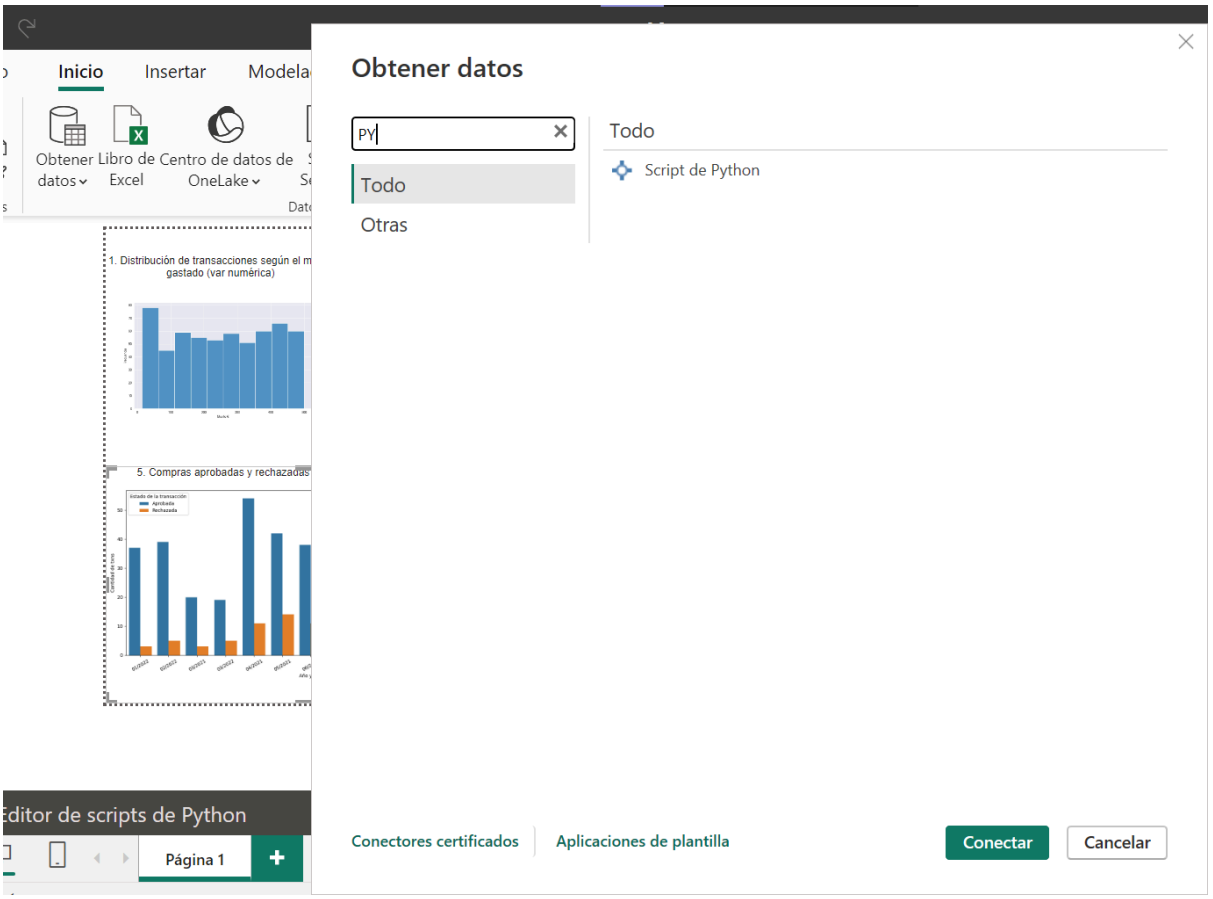
## Hecho por: Yatmelis Freites

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.

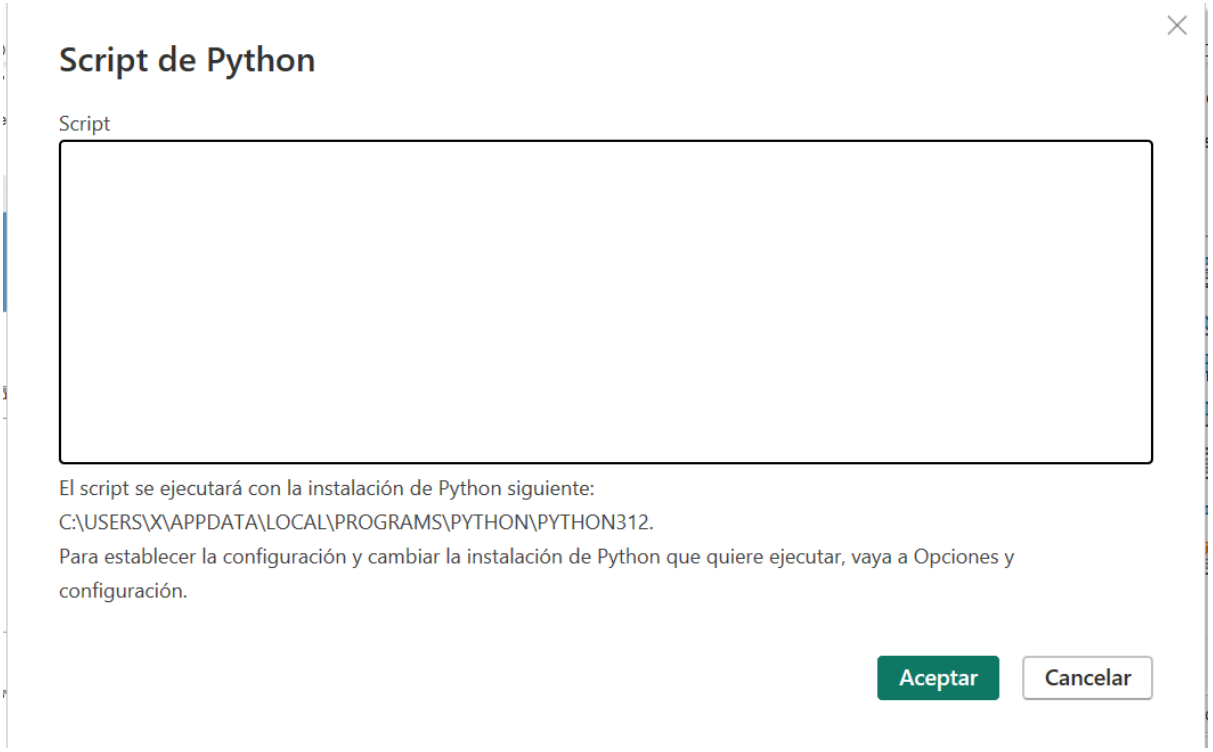
# Nivell 1

Els 7 exercicis del nivell 1 de la tasca 01

Para empezar, es necesario conectar con python, por lo que se selecciona “Obtener datos” y en la barra de búsqueda se escribe py y seleccionamos la opción “Script de Python”,  accedemos haciendo click en Conectar



Luego aparece un lugar donde se pega el código de python:




Y en mi caso he seleccionado todo el código referente a conexión y modificaciones del dataframe:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import mysql.connector
```

Python

```python
from sqlalchemy import create_engine
engine = create_engine('mysql+mysqlconnector://root:1234@localhost/geekstore')

df = {table: pd.read_sql(f'SELECT * FROM {table}', engine) for table in ['companies', 'products', 'transactions']}

engine.dispose()
```

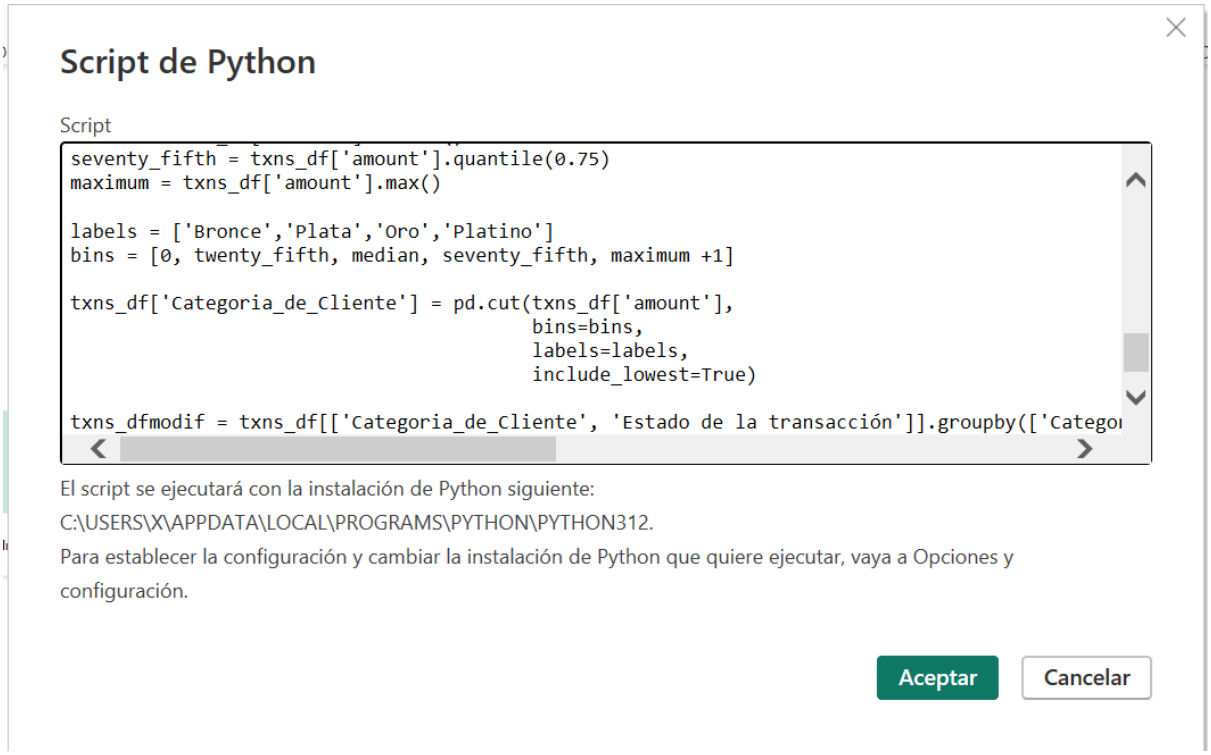
Python

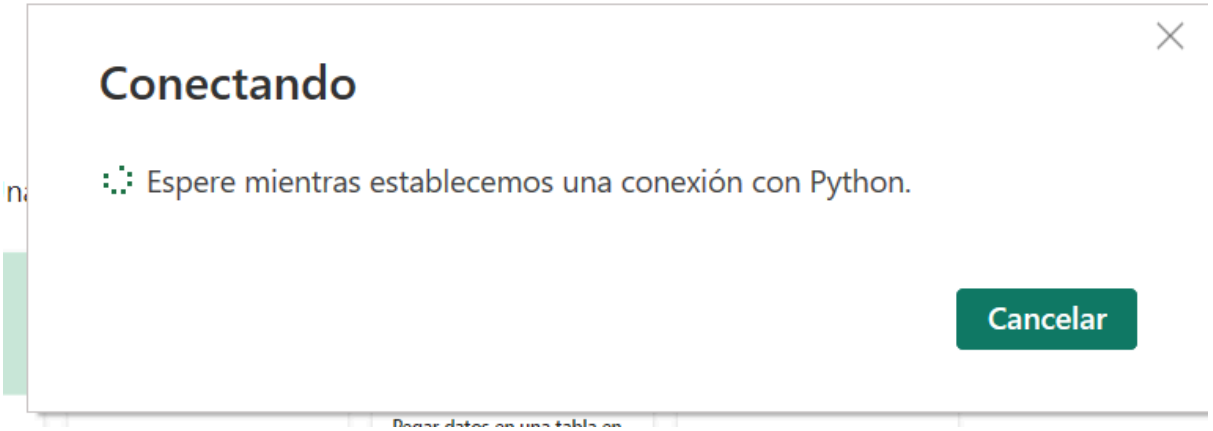
```python
companies_df = df['companies']
prod_df = df['products']
txns_df = df['transactions']
```

```python
companies_df = df['companies']
prod_df = df['products']
txns_df = df['transactions']

prod_df['price'] = prod_df['price'].str.replace('$', '')
print(prod_df['price'].head())
prod_df['price'] = prod_df['price'].astype(float)

txns_df = txns_df.rename(columns={'business_id':'company_id'})
print(txns_df.head())

txns_df['Año y Mes de Compra'] = txns_df['timestamp'].dt.to_period('M')

txns_df['Año y Mes de Compra'] = txns_df['timestamp'].dt.strftime('%B %Y')

txns_df['declined'] = txns_df['declined'].astype(str)

txns_df['declined'] = txns_df['declined'].str.replace('0', 'Aprobada')
txns_df['declined'] = txns_df['declined'].str.replace('1', 'Rechazada')

txns_df = txns_df.rename(columns={'declined':'Estado de la transacción'})

twenty_fifth = txns_df['amount'].quantile(0.25)
median = txns_df['amount'].median()
seventy_fifth = txns_df['amount'].quantile(0.75)
maximum = txns_df['amount'].max()

labels = ['Bronce','Plata','Oro','Platino']
bins = [0, twenty_fifth, median, seventy_fifth, maximum +1]

txns_df['Categoria_de_Cliente'] = pd.cut(txns_df['amount'],
                                         bins=bins,
                                         labels=labels,
                                         include_lowest=True)

txns_dfmodif = txns_df[['Categoria_de_Cliente', 'Estado de la transacción']].groupby(['Categoria_de_Cliente', 'Estado de la transacción'])['Estado de la transacción'].count().reset_index(name='cuenta')
```

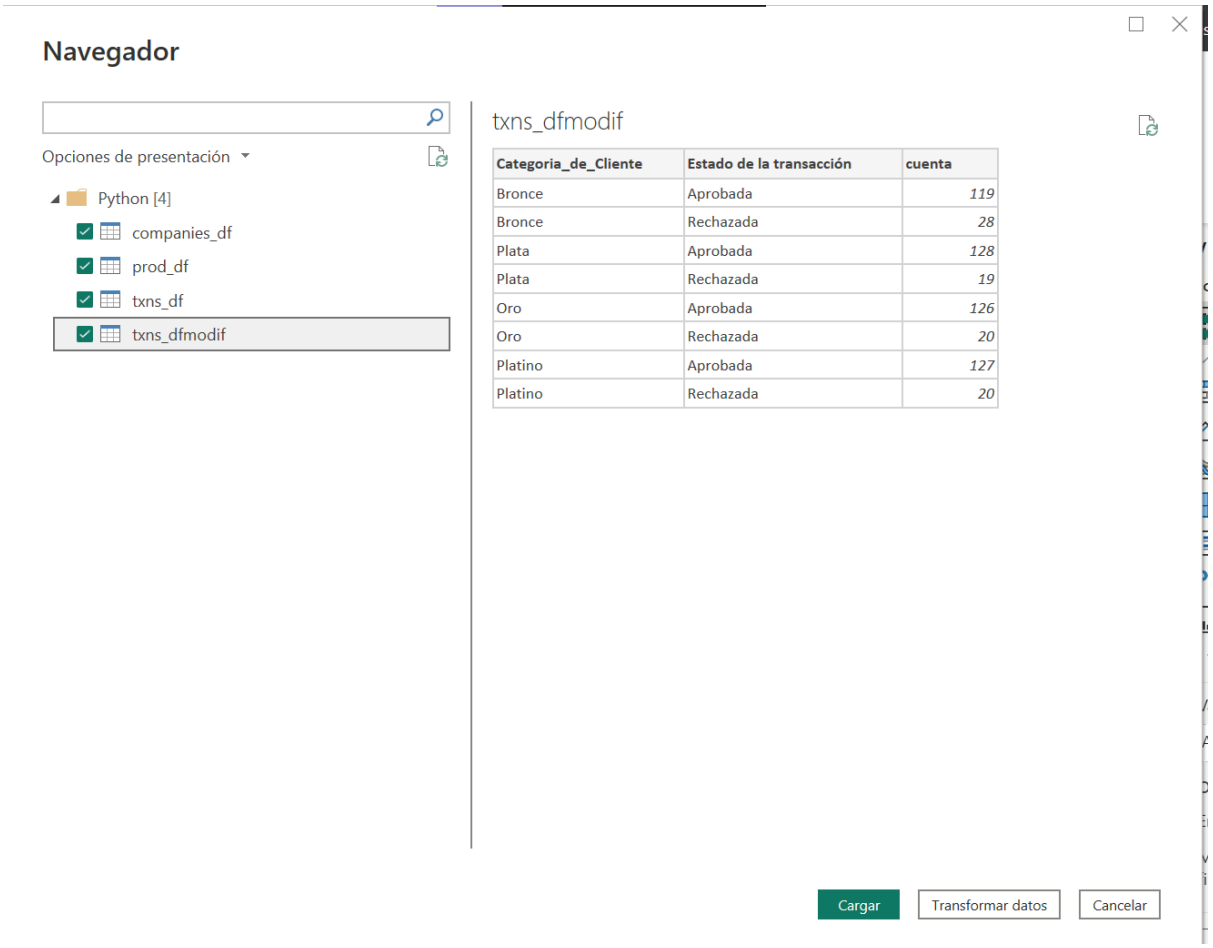
Python

Pego el código y aparece un menú con las tablas que se están importando a powerbi






Seleccioné todas las tablas ya que las uso todas

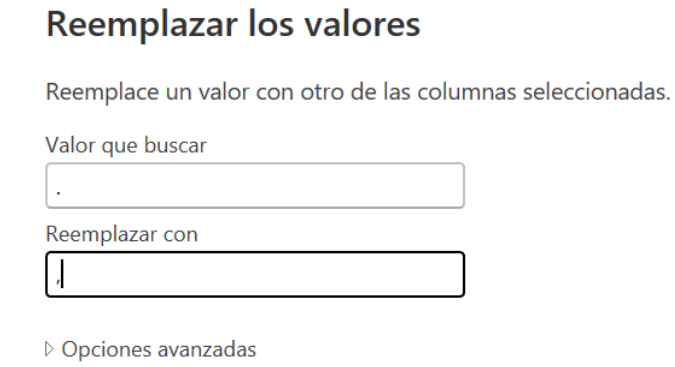


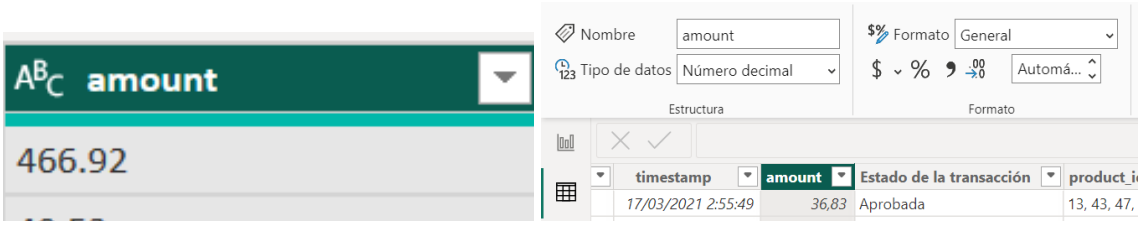

| Categoria_de_Cliente | Estado de la transacción | cuenta |
|---|---|---|
| Bronce | Aprobada | 119 |
| Bronce | Rechazada | 28 |
| Plata | Aprobada | 128 |
| Plata | Rechazada | 19 |
| Oro | Aprobada | 126 |
| Oro | Rechazada | 20 |
| Platino | Aprobada | 127 |
| Platino | Rechazada | 20 |

# - Exercici 1

Una variable numèrica.

Al momento de realizar este ejercicio, presentaba problemas con el gráfico, y era necesario realizar cambios en el tipo de dato.



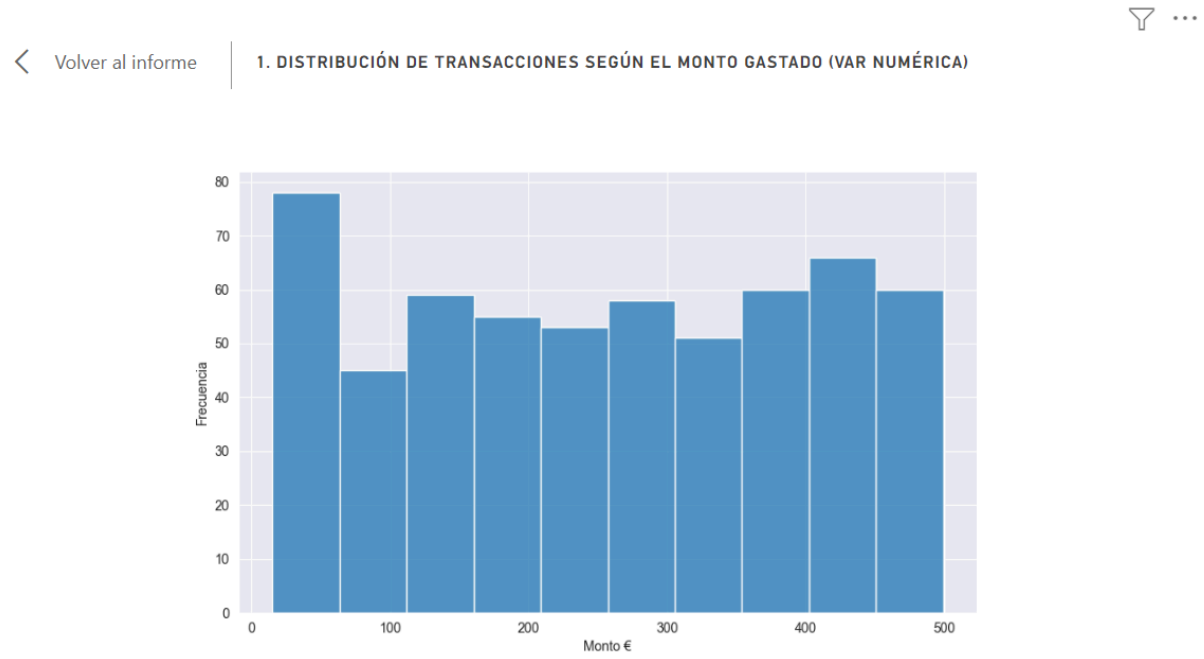

Se encontraba en formato de texto, y con un punto en lugar de coma, por lo tanto hice un reemplazo de puntos por comas en esta columna y luego, lo cambié a número decimal a sugerencia de mi tutora para evitar problemas con la aplicación, queda en formato general por esto mismo en lugar de formato de moneda
Primero debe importarse en cada una de las visualizaciones matplotlib.pyplot como plt y seaborn como sns.
PowerBI hace cambios en el nombre del dataframe y hace necesario denominarlo "dataset" en cada una de las visualizaciones
Usé el método set_style para cambiar el estilo de la visualizacion con seaborn solo por curiosidad de ver si powerbi aceptaba algún cambio en la estética, y con plt.figure cambié el tamaño del gráfico con matplotlib.
Con sns hago el plotting del histograma poniendo como fuente de dato dataset e indicando que me refiero a la columna amount, cosa que solo hice como paso extra en este ejercicio ya que no es necesario si se selecciona previamente la columna a trabajar con powerbi:


Por cuestiones de estética de la visualización, le asigno un nombre a la variable, h en este caso, para luego cambiarle el nombre a la visualización y a las etiquetas con matplotlib.
Al final solo se escribe plt.show() para poder ver nuestro gráfico al ejecutar.

- Exercici 1 Una variable numèrica.

```
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style('darkgrid')
plt.figure(figsize=(10, 6)) #cambio el tamaño del gráfico con matplotlib

h = sns.histplot(dataset['amount'], bins=10)
h.figure.suptitle('Distribución de transacciones según el monto gastado', y=1.03)
h.set(xlabel='Monto €', ylabel='Frecuencia')
plt.show()
```

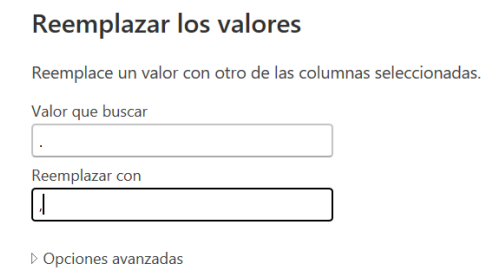
Ejecuto el script con el botón y se muestra este gráfico (ya editado con el formateo de powerbi)




# - Exercici 2

Dues variables numèriques.

Ya que la columna de amount había tenido errores en el tipo de dato hice una revisión en las columnas con las que trabajaría: price y weight de prod_df, por lo que hice el mismo reemplazo de puntos por comas en ambas columnas




| price | weight |
|---|---|
| 161,11 | 1 |
| 9,24 | 2 |
| 71,89 | 3 |
| 171,22 | 3,2 |
| 136,60 | 0,8 |
| 63,33 | 0,6 |
| 32,37 | 1,4 |
| | 1,2 |
| | 2,4 |

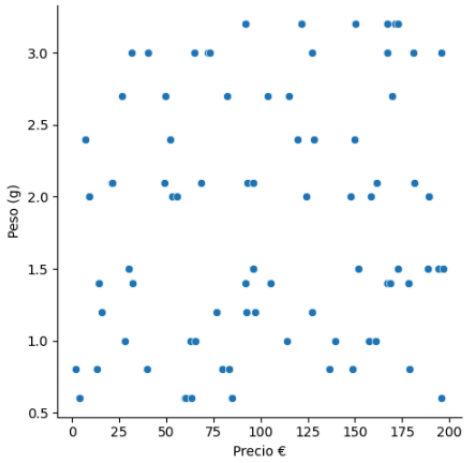
Primero importo matplotlib.pyplot como plt y seaborn como sns.
Con sns.relplot indico que estoy haciendo un plotting de un gráfico de relación entre datos, que como default usa los gráficos de dispersión (se puede especificar para escoger una línea en su lugar, aclaro que uso relplot porque me gusta la versatilidad del código en el sentido de que con menos cambios puedo ver algo distinto y por eso escribo de esta manera para tener en cuenta que a futuro puedo jugar con esta versatilidad). En powerbi arrastro las columnas con las que trabajo a “Valores”


Igual que con el gráfico anterior, le asigno un nombre a la variable: corr, para luego cambiarle el nombre a las etiquetas con matplotlib.
Al final solo se escribe plt.show() para poder ver nuestro gráfico al ejecutar.

- Exercici 2 Dues variables numèriques.

```
import matplotlib.pyplot as plt
import seaborn as sns

corr= sns.relplot(
            dataset,
            x = 'price',
            y = 'weight'
)

corr.set(xlabel='Precio €', ylabel='Peso (g)')
plt.show()
```

Volver al informe | 2.RELACIÓN ENTRE PRECIO Y PESO DE PRODUCTOS(2 VAR NUMÉRICAS)




## - Exercici 3

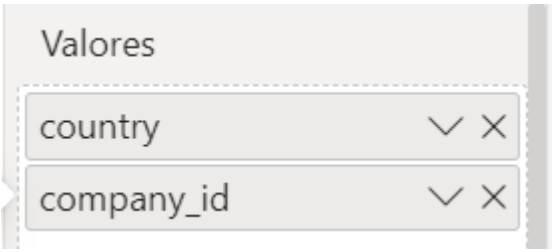
Una variable categòrica.

Primero importo matplotlib.pyplot como plt y seaborn como sns.
Con catplot se da por entendido que cualquier gráfico que se hace es de datos categóricos, al igual que con relplot lo uso para que en una futura revisión de código me quede claro que puedo ver los datos de una manera distinta con otro estilo de gráfico. En lugar de pasarle la variable categórica en el eje X como se acostumbra, lo hice en el eje Y para que se vea un

poco distinto. Ya que le he asignado como nombre de variable a este plotting “count”, puedo ponerle distintos nombres a los ejes.
Finalmente con plt.show puedo hacer que se vea el gráfico.

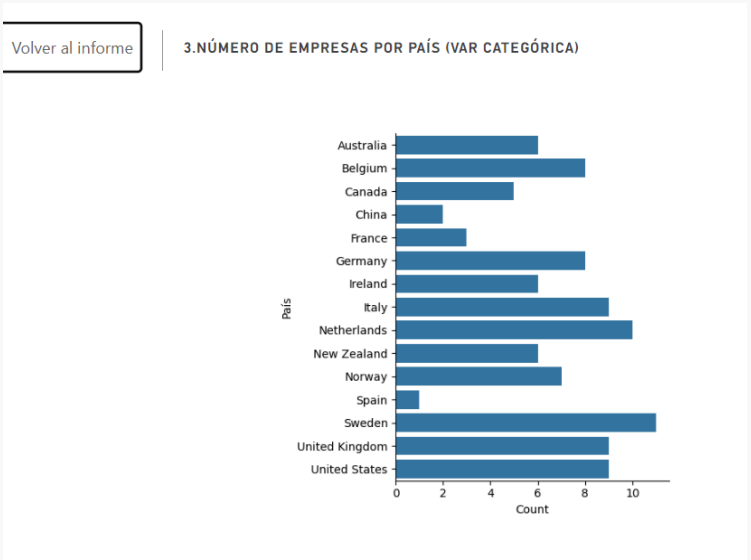
Recalco que para graficar tenía errores hasta recordar que con powerbi si realizas un “count” debes aclara “qué” y “por qué variable estás contando”.

En este caso solo agregué company_id a mis valores, ya que estoy contando los company ids por país




- Exercici 3 Una variable categòrica.

```
import matplotlib.pyplot as plt
import seaborn as sns

count = sns.catplot(dataset, y='country', kind='count')
count.set(xlabel='Count', ylabel='País')
plt.show()
```




## - Exercici 4

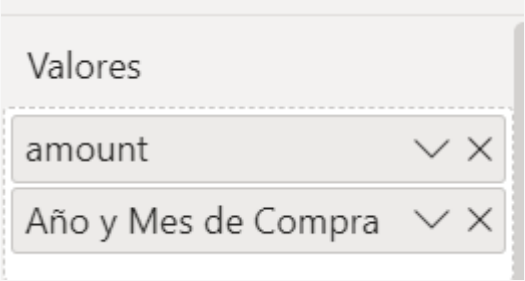
Una variable categòrica i una numèrica.

Para el ejercicio 4 y el 5 tuve que modificar los datos en la columna Año y Mes de Compra de la tabla txns_df, cambiando el tipo de timestamp a texto para evitar confusiones y problemas. ya que cuento las barras, necesito mantener 7 caracteres, así que modifico la columna para que solo muestre el mes y año (mas la barra)


| | | | | |
|---|---|---|---|---|
| 92 | -256402 | -74192 | 03/2021 | Platino |
| 248 | -125373 | -131892 | 03/2021 | Plata |
| 210 | -39921 | -149125 | 03/2021 | Plata |
| 137 | 690549 | 965033 | 03/2021 | Platino |

Este gráfico de cajas y bigotes lo hago con catplot, y además como tipo especifico “box”, y para observar que son categóricas las cajas, le agrego un hue de mi variable categórica.

En la versión de python pude ordenar las cajas usando una lista donde van los valores, pero en powerbi se me hizo muy complicado realizar este orden usando el mismo código que en el archivo del sprint 8.1, por lo cual tras comentarle a mi tutora decidimos que así quedaba y solo escribo esto por aclarar cualquier confusión que pueda surgir tras observar el gráfico en desorden

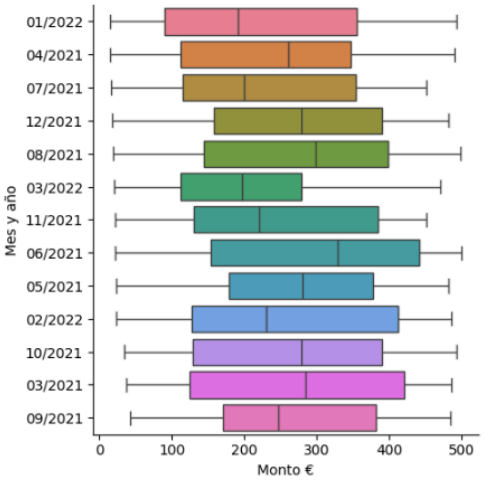


- Exercici 4 Una variable categòrica i una numèrica.

```
import matplotlib.pyplot as plt
import seaborn as sns

cajas= sns.catplot(dataset,
                   x='amount',
                   y='Año y Mes de Compra',
                   kind='box',
                   hue = 'Año y Mes de Compra')

cajas.set(xlabel='Monto €', ylabel='Mes y año')


plt.show()
```

Volver al informe

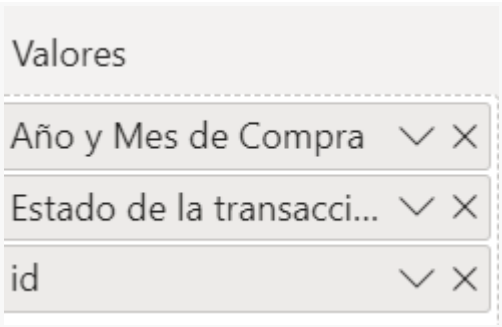
4.MONTO GASTADO EN COMPRAS POR MES Y AÑO (1 NUMÉRICA Y 1 CATEGÓRICA)



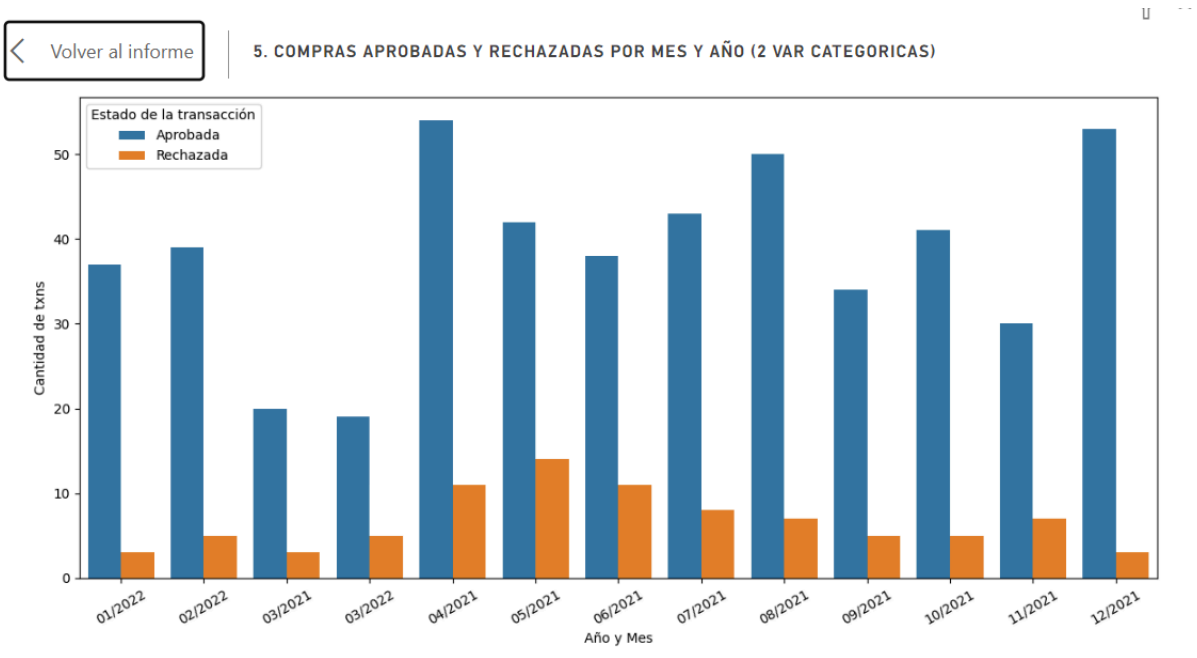

## - Exercici 5

Dues variables categòriques.

Para este gráfico haré un countplot colocando en el eje x la primera variable categórica y en hue la segunda variable categórica, asignándole el nombre de variable como “declinadas”. Como tenía problemas con la visualización en el dashboard, hice cambios con matplotlib del tamaño, la rotación de las etiquetas del eje x y con tight_layout evito que se superpongan las etiquetas. También pongo nombre a los ejes.




También tenía problemas para graficar, por lo tanto hice lo mismo que con el gráfico nº 3, y le agregué el id de transacciones a los valores ya que estoy contando los ids de transacciones rechazadas y aprobadas en cada año y mes de compra
Al igual que con el gráfico anterior, se me hacía un poco complicado ordenar por la variable Año y mes de compra


## - Exercici 6

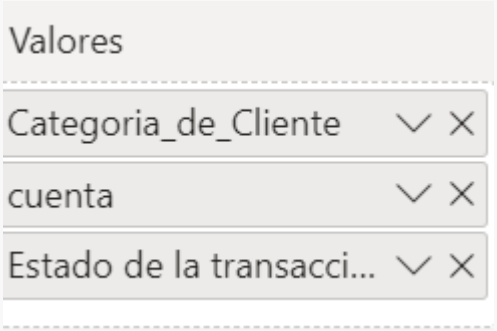
Tres variables.

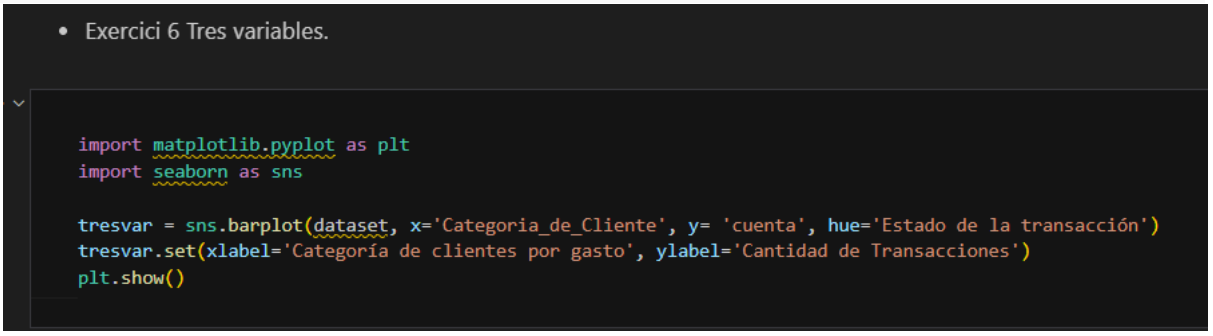
Selecciono las tres columnas de la tabla txns_modif, y realizo un gráfico de columnas agrupadas, poniendo a categoría de cliente en el eje x y a “cuenta” en el eje Y. Este ejercicio fue el que más me costó hacer ya que no quise adentrarme en el proceso de merge para no “sobrecomplicarme”, lo que terminó complicándome igual o más 😂

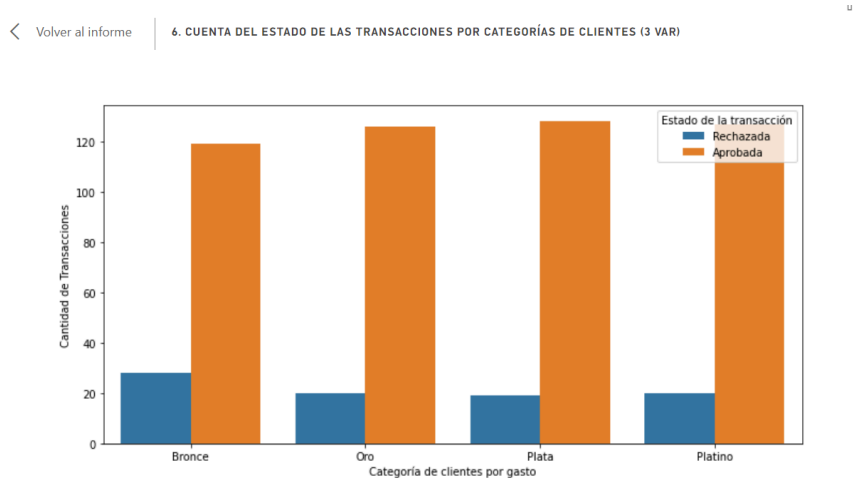
Así que tras pedir orientación a Alana, me ayudó a comprender que realizar un count en el dataframe sería una manera correcta de abordar el ejercicio con los datos que tenía, a diferencia de realizar un countplot como en ejercicios anteriores.

De esta manera, el hue va determinado por el estado de la transacción y sería la tercera variable.

Al igual que los ejercicios anteriores le he asignado un nombre a la variable que contiene el gráfico y así puede modificarse con matplotlib



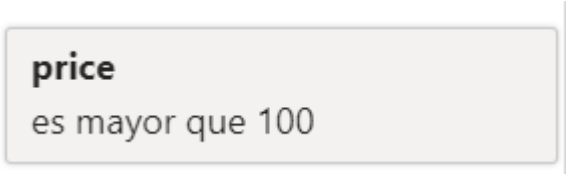



# - Exercici 7

Graficar un Pairplot.

En el sprint 8.1 hago un filtrado de datos a graficar con una query para poder entender un poco más sobre los productos que tienen precios superiores a 100€, por lo que para adaptarlo a powerbi simplemente pongo un filtro de “precio es mayor que 100” luego de seleccionar los campos que uso para graficar (un pairplot por default tomaría todos los campos numéricos, pero por simplicidad solo me he quedado con 2)




Al igual que en todos los demás ejercicios importo matplotlib.pyplot como plt y seaborn como sns.
Hago el plotting, y selecciono el tipo de diagrama que quiero en mi pairplot

```
import matplotlib.pyplot as plt
import seaborn as sns

pair= sns.pairplot(dataset,
                   diag_kind='hist')

pair.figure.suptitle('Distribución y Correlaciones de peso y precio de productos con un precio superior a 100 €', y=1.05)

plt.show()
```