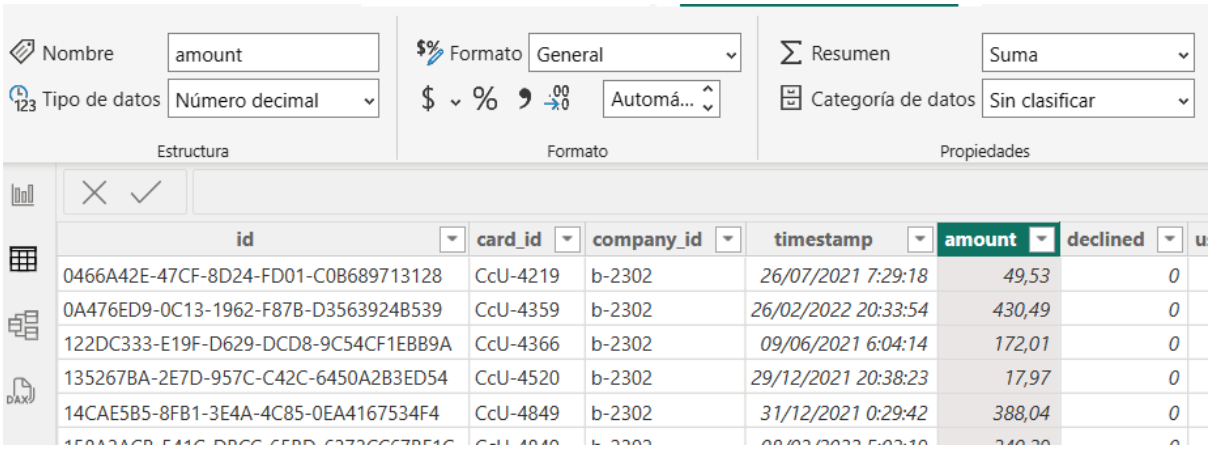
SPRINT 8.02

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.

Antes de empezar a trabajar con los datos, tuve que hacer un ajuste en las columnas numéricas porque no las leía bien Power BI. Tuve que cambiar los parámetros a texto, para después volverles a dar un valor decimal para que el programa lo leyera bien. También hemos tenido que utilizar el valor id en todos los dataset de las visualizaciones para que cogiera los valores correctamente aunque luego esa columna no se use para los gráficos.

Para poder gestionar los datos, hemos tenido que reemplazar los dataframes del Sprint 8.01 por el dataset que genera Power BI al seleccionar las columnas con las que queremos trabajar.

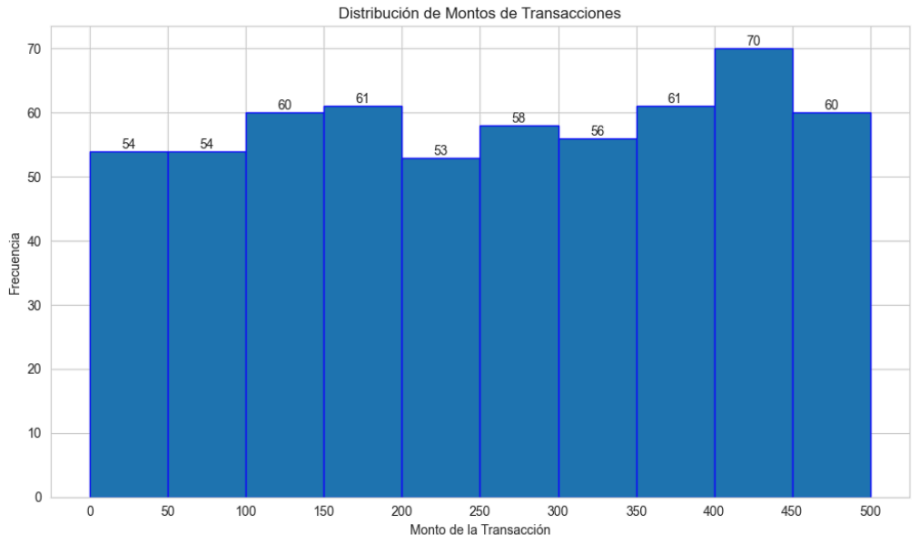


id	card_id	company_id	timestamp	amount	declined
0466A42E-47CF-8D24-FD01-C0B689713128	CcU-4219	b-2302	26/07/2021 7:29:18	49,53	0
0A476ED9-0C13-1962-F87B-D3563924B539	CcU-4359	b-2302	26/02/2022 20:33:54	430,49	0
122DC333-E19F-D629-DCD8-9C54CF1EBB9A	CcU-4366	b-2302	09/06/2021 6:04:14	172,01	0
135267BA-2E7D-957C-C42C-6450A2B3ED54	CcU-4520	b-2302	29/12/2021 20:38:23	17,97	0
14CAE5B5-8FB1-3E4A-4C85-0EA4167534F4	CcU-4849	b-2302	31/12/2021 0:29:42	388,04	0

NIVELL 1

EXERCICI 1

Una variable numèrica.



Este es el código usado para el gráfico:

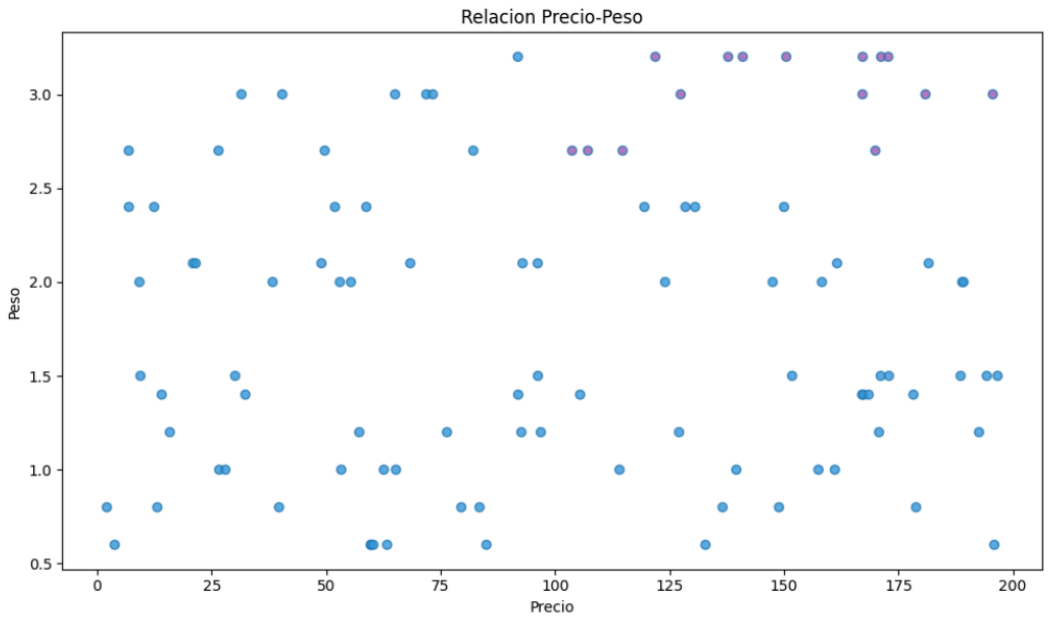
-df_transactions: amount

```python
# El código siguiente, que crea un dataframe y quita las filas duplicadas, siempre se ejecuta y actúa como un preámbulo del script:

# dataset = pandas.DataFrame(id, amount)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style('whitegrid') # Poner el fondo blanco y con rejilla
plt.figure(figsize=(10, 6))
values, bins, bars = plt.hist(dataset['amount'], bins=np.arange(0, 525, 50), edgecolor="blue")
plt.xticks(bins)
plt.title('Distribución de Montos de Transacciones')
plt.xlabel('Monto de la Transacción')
plt.ylabel('Frecuencia')
plt.bar_label(container=bars)
plt.tight_layout()

plt.show()
```

Python

EXERCICI 2

Dos variables numèriques.



Este es el código usado para el gráfico:

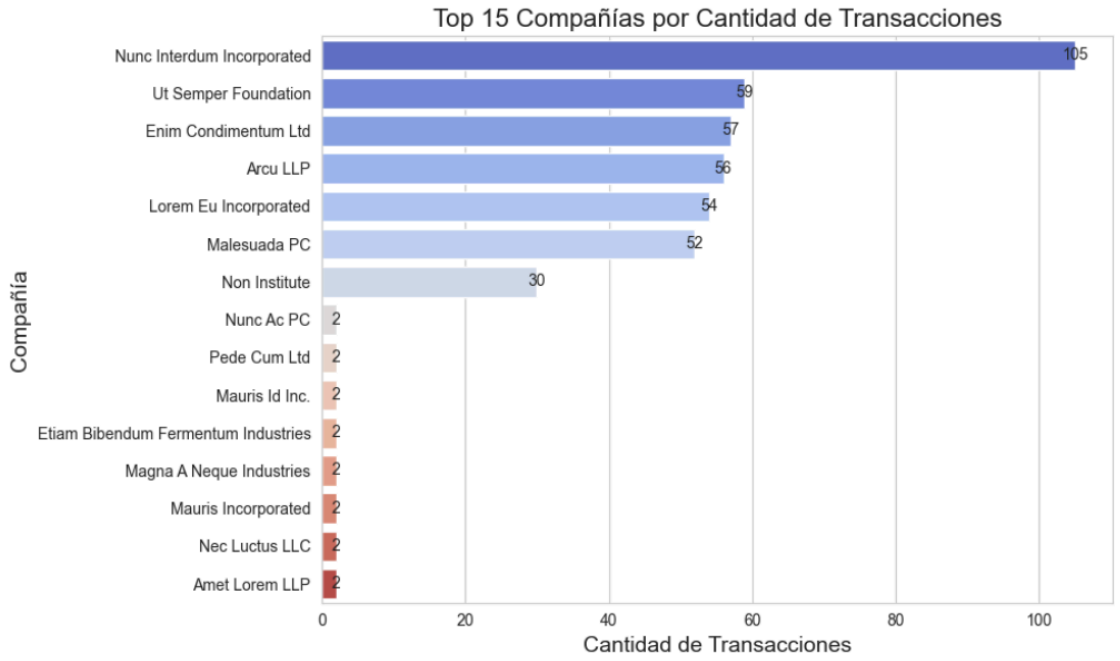
Variables:

-df_products: price, weight

```
# El código siguiente, que crea un dataframe y quita las filas duplicadas, siempre se ejecuta y actúa como un preámbulo del script:

# dataset = pandas.DataFrame(price, weight, id)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(10, 6))
value=(dataset['price']>100) & (dataset['weight']>2.5)
dataset['color']= np.where( value==True , "#9b59b6", "#3498db")
sns.regplot(data=dataset, x="price", y="weight", fit_reg=False, scatter_kws={'facecolors':dataset['color']})
plt.title('Relación Precio-Peso')
plt.xlabel('Precio')
plt.ylabel('Peso')
plt.show()
```

EXERCICI 3

Una variable categòrica.



Este es el código usado para el gráfico:

Variables:

df_transactions: company_id

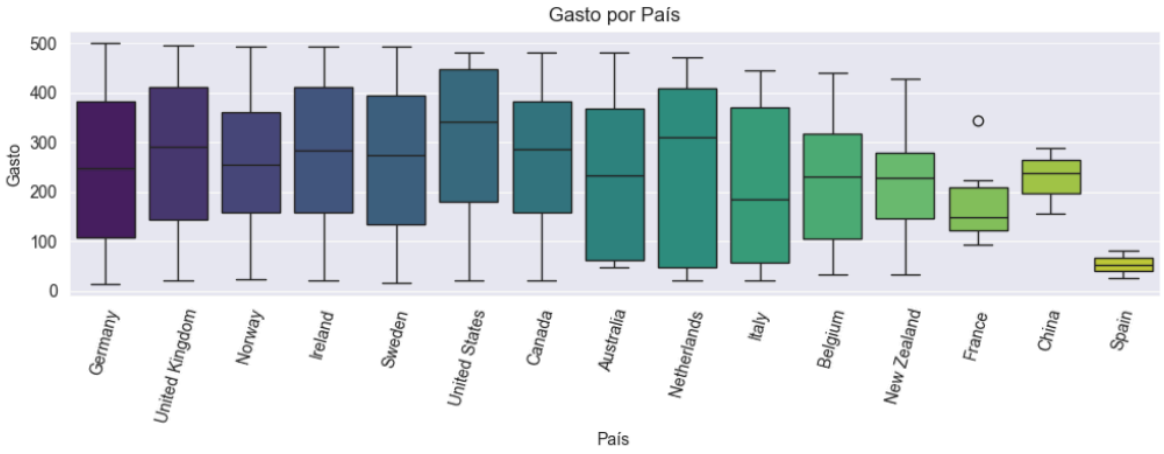
df_companies: company_name

```
# El código siguiente, que crea un dataframe y quita las filas duplicadas, siempre se ejecuta y actúa como un preámbulo del script:

# dataset = pandas.DataFrame(company_id, company_name, id)
# dataset = dataset.drop_duplicates()
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

transacciones_por_compania = dataset['company_id'].value_counts().reset_index()
transacciones_por_compania.columns = ['company_id', 'cantidad_transacciones']
df_company_count = pd.merge(dataset, transacciones_por_compania, left_on='id', right_on='company_id')
df_company_count = df_company_count[['company_name', 'cantidad_transacciones']].drop_duplicates()
df_company_count = df_company_count.sort_values(by='cantidad_transacciones', ascending=False).head(15)

sns.set_style('whitegrid')
fig, ax = plt.subplots(figsize=(10, 6))
sns.barplot(data=df_company_count, y='company_name', x='cantidad_transacciones', palette="coolwarm", dodge=False)
plt.title('Top 15 Compañías por Cantidad de Transacciones', fontsize=16)
plt.xlabel('Cantidad de Transacciones', fontsize=14)
plt.ylabel('Compañía', fontsize=14)
plt.yticks(fontsize=7)

for index, value in enumerate(df_company_count['cantidad_transacciones']):
    plt.text(value, index, str(value), fontsize=10, ha='center', va='center')
plt.show()
```

En el siguiente gráfico hemos tenido que utilizar el .drop_duplicates() para eliminar datos duplicados, ya que al usar el mismo código que en el sprint 8.01 nos salían los datos erróneos y con drop.duplicates se solventó el problema.

Exercici 4

Una variable categòrica i una numèrica.



Este es el código usado para el gráfico:

Variables:

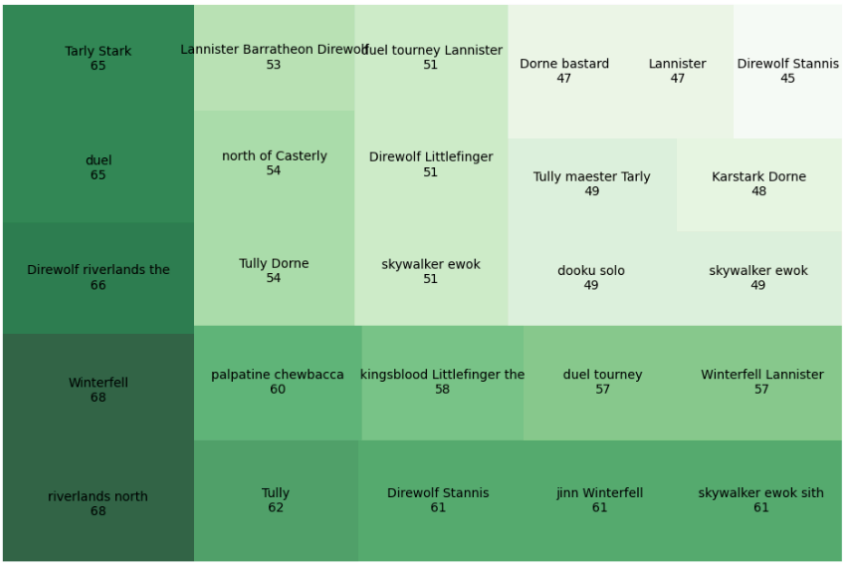
df_transactions: amount

df_companies: country

```
# El código siguiente, que crea un dataframe y quita las filas duplicadas, siempre se ejecuta y actúa como un preámbulo del script:

# dataset = pandas.DataFrame(country, amount, id)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:

import matplotlib.pyplot as plt
import seaborn as sns

dataset = dataset.sort_values(by='amount', ascending=False)

sns.set_style('darkgrid')
plt.figure(figsize=(10,4))
sns.boxplot(data=dataset, x='country', y='amount', hue= 'country', legend=False, palette='viridis')
plt.xlabel('País')
plt.ylabel('Gasto')
plt.title('Gasto por País')
plt.xticks(rotation=75)
plt.tight_layout()
plt.show()
```

Exercici 5

Dues variables categòriques.



Este es el código usado para el gráfico:

Variables:

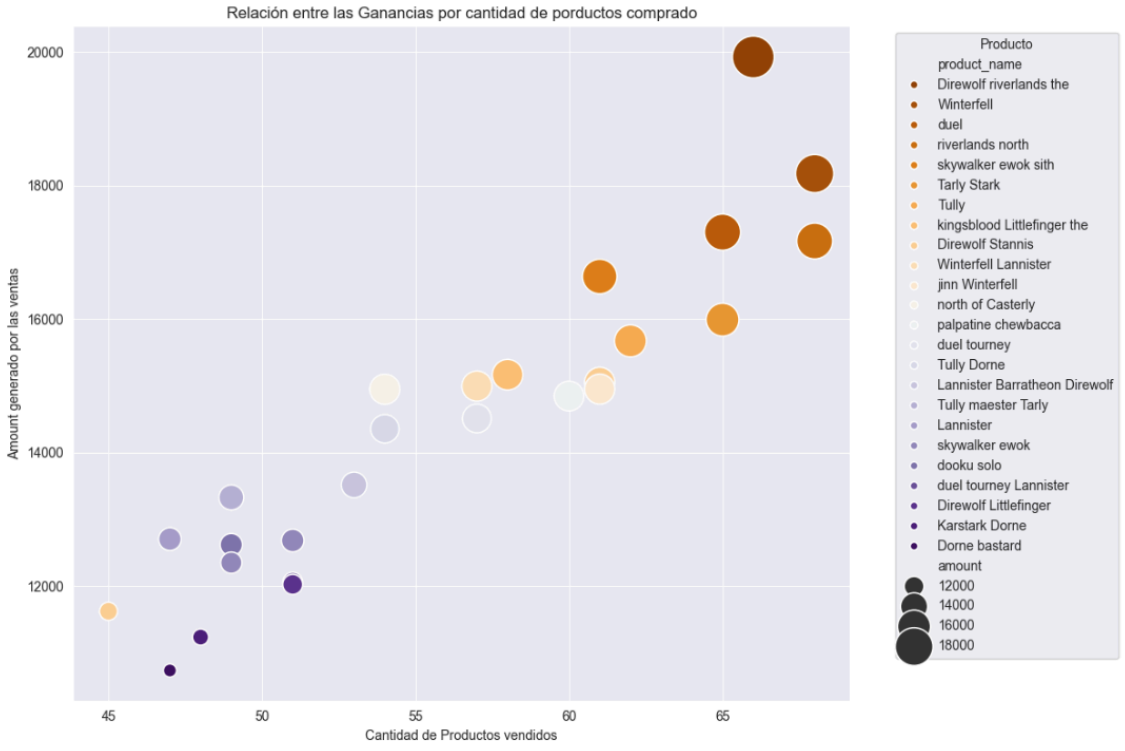
df_products: product_name, id

df_products_transactions: product_id

```
# El código siguiente, que crea un dataframe y quita las filas duplicadas, siempre se ejecuta y actúa como un preámbulo del script:

# dataset = pandas.DataFrame(product_id, product_name, id)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:
import pandas as pd
import matplotlib.pyplot as plt
import squarify

transacciones_por_producto = dataset['product_id'].value_counts().reset_index()
transacciones_por_producto.columns = ['product_id', 'cantidad_productos_comprados']
df_best_products = pd.merge(dataset, transacciones_por_producto, left_on='id', right_on='product_id')
df_best_products = df_best_products[['product_name', 'cantidad_productos_comprados']].drop_duplicates()
df_best_products = df_best_products.sort_values(by='cantidad_productos_comprados', ascending=False)

# Creación del Treemap
cmap = plt.cm.Greens
mini = min(df_best_products['cantidad_productos_comprados'])
maxi = max(df_best_products['cantidad_productos_comprados'])
norm = plt.Normalize(vmin=mini, vmax=maxi)
colors = [cmap(norm(value)) for value in df_best_products['cantidad_productos_comprados']]
plt.figure(figsize=(12, 8))
squarify.plot(sizes=df_best_products['cantidad_productos_comprados'], alpha=.8, color=colors,
              label=df_best_products['product_name']+ "\n" + df_best_products['cantidad_productos_comprados'].astype(str))
plt.axis('off')
plt.show()
```

Exercici 6

Tres variables.



Este es el código usado para el gráfico:

Variables:

df_transactions: amount

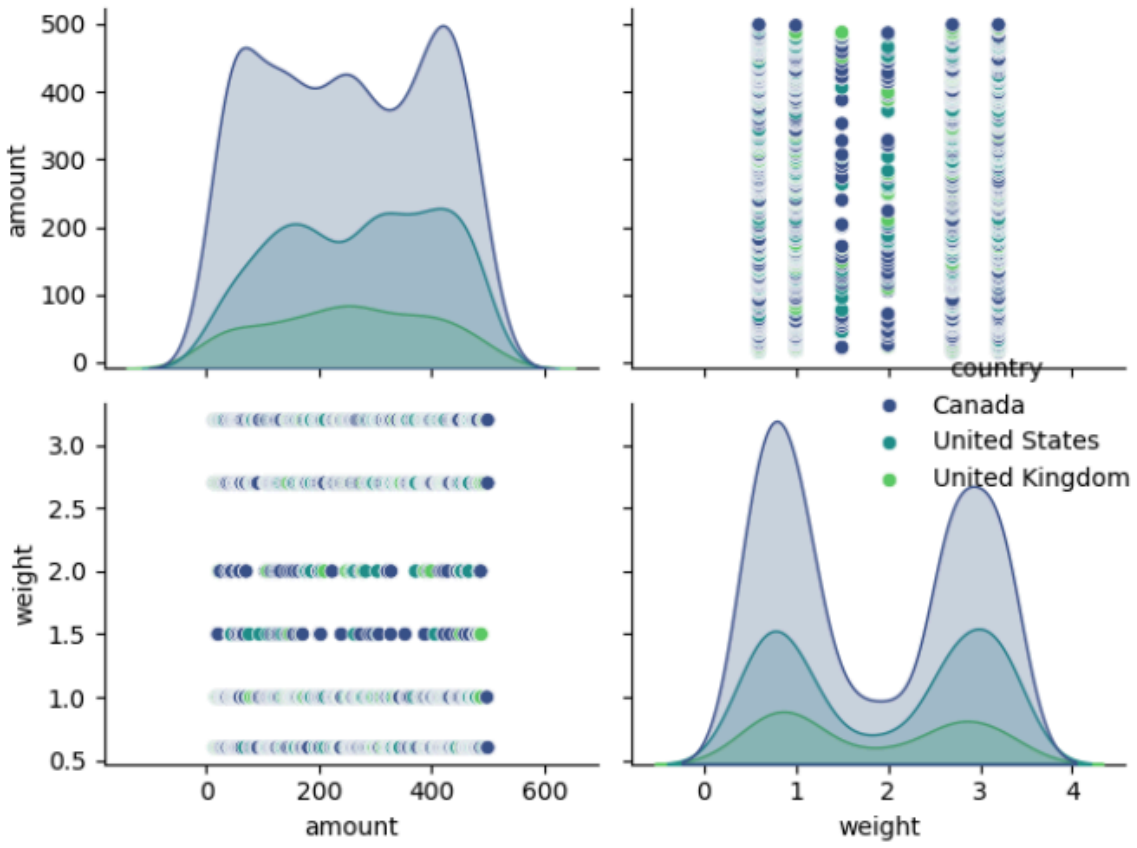
df_products_transactions: transaction_id,product_id

df_products: product_name

```
# El código siguiente, que crea un dataframe y quita las filas duplicadas, siempre se ejecuta y actúa como un preámbulo del script:
# dataset = pandas.DataFrame(product_id, product_name, amount, id,transaction_id)
# dataset = dataset.drop_duplicates()
# Pegue o escriba aquí el código de script:

import matplotlib.pyplot as plt
import seaborn as sns

df_agg = dataset.groupby(['product_id', 'product_name']).agg(
    cantidad_productos=('product_id', 'count'),
    total_amount=('amount', 'sum')
).reset_index()
df_agg = df_agg.sort_values(by='total_amount', ascending=False)

sns.set_style('darkgrid')
plt.figure(figsize=(12, 8))
sns.scatterplot(data=df_agg, x='cantidad_productos', y='total_amount', size='total_amount', sizes=(100, 1000), hue='product_name', palette='PuOr')
plt.title('Relación entre las Ganancias por cantidad de porductos comprado')
plt.xlabel('Cantidad de Productos vendidos')
plt.ylabel('Amount generado por las ventas')
plt.legend(title='Producto', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```

Exercici 7

Graficar un Pairplot.



Este es el código usado para el gráfico:

```
Variables:

df_users: country

df_transactions: id, amount

df_products: weight
```

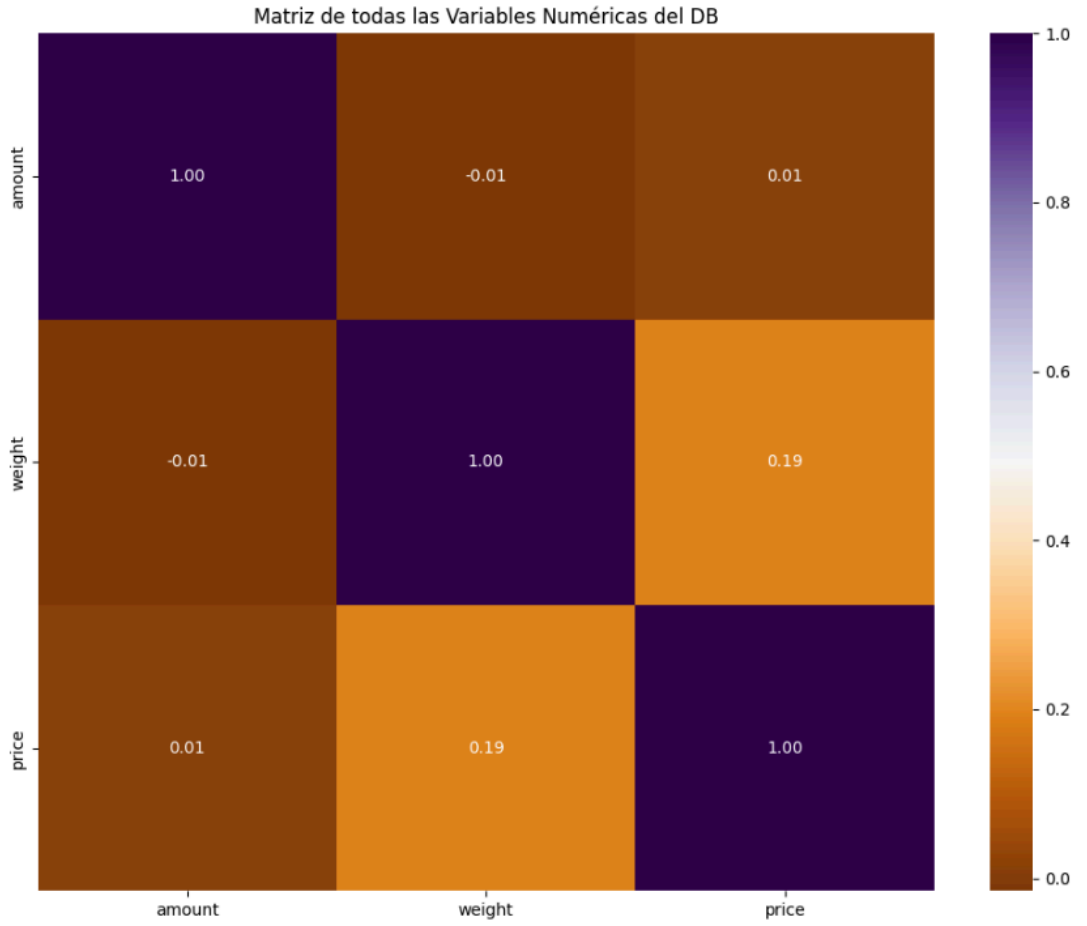
```python
# El código siguiente, que crea un dataframe y quita las filas duplicadas, siempre se ejecuta y actúa como un preámbulo del script:

# dataset = pandas.DataFrame(country, weight, id, amount)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:
import seaborn as sns
import matplotlib.pyplot as plt

df_filter_merge = dataset[['country', 'amount','weight']]

sns.pairplot(df_filter_merge, hue= 'country', palette='viridis')
plt.show()
```

En este caso en concreto no he sabido cómo mover la leyenda que aparece a partir del hue 'country', como se puede observar aparece dentro del gráfico. En el Sprint 8.01 al realizar el gráfico de la misma manera, la 'leyenda' aparece colocada de manera correcta.

NIVELL 2

Exercici 1

Correlació de totes les variables numèriques.



Este es el código usado para el gráfico:

Variables:

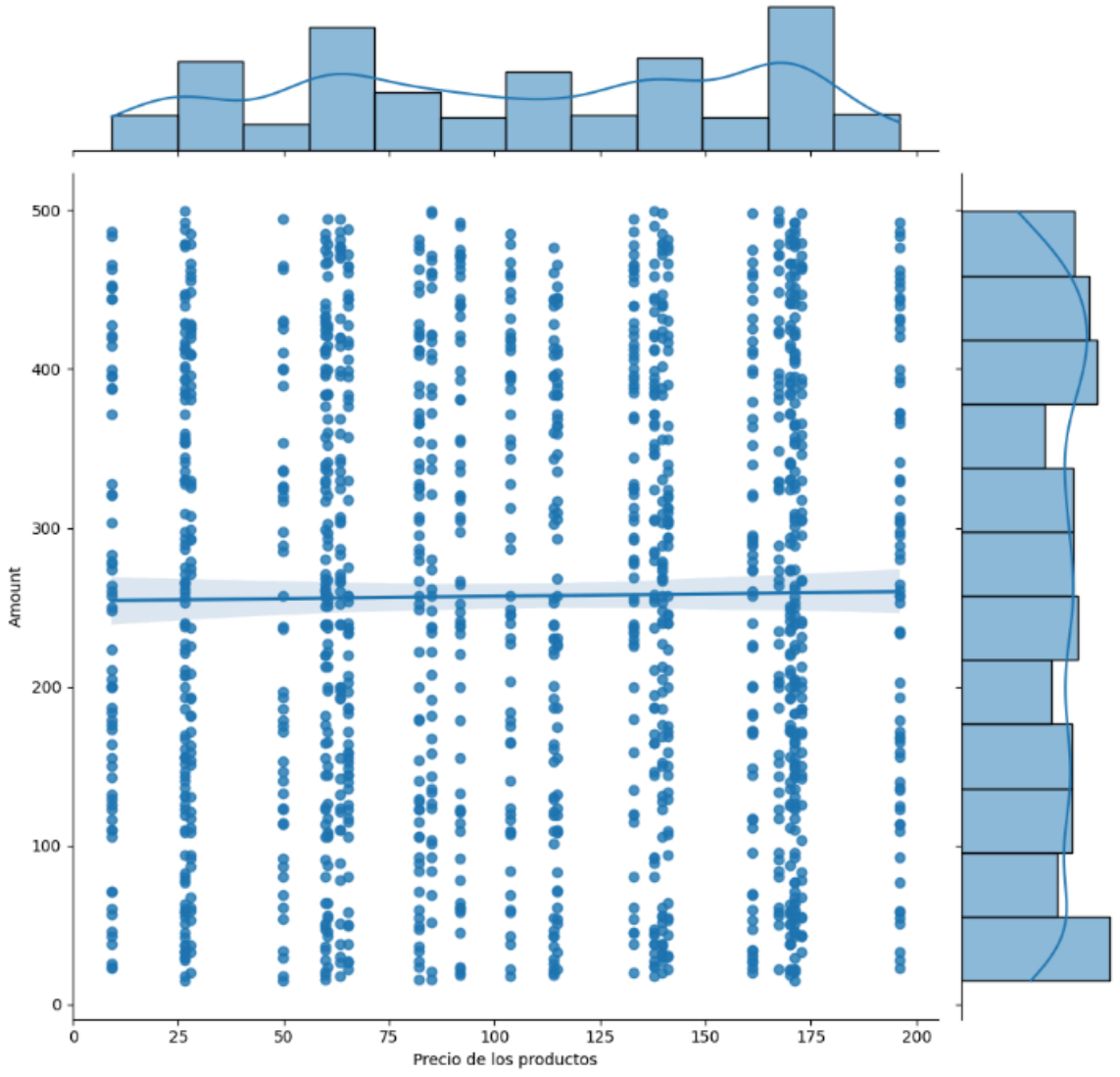
df_products: weight, price

df_transactions:amount

```
# El código siguiente, que crea un dataframe y quita las filas duplicadas, siempre se ejecuta y actúa como un preámbulo del script:

# dataset = pandas.DataFrame(amount, price, weight)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:
import seaborn as sns
import matplotlib.pyplot as plt

df_numeros = dataset.select_dtypes(include=['float64', 'int64'])
matriz = df_numeros.corr()


plt.figure(figsize=(10, 8))
sns.heatmap(matriz, annot=True, cmap='PuOr', fmt='.2f')
plt.title('Matriz de todas las Variables Numéricas del DB')
plt.tight_layout()
plt.show()
```

Exercici 2

Implementa un jointplot.



Este es el código usado para el gráfico:

Variables:

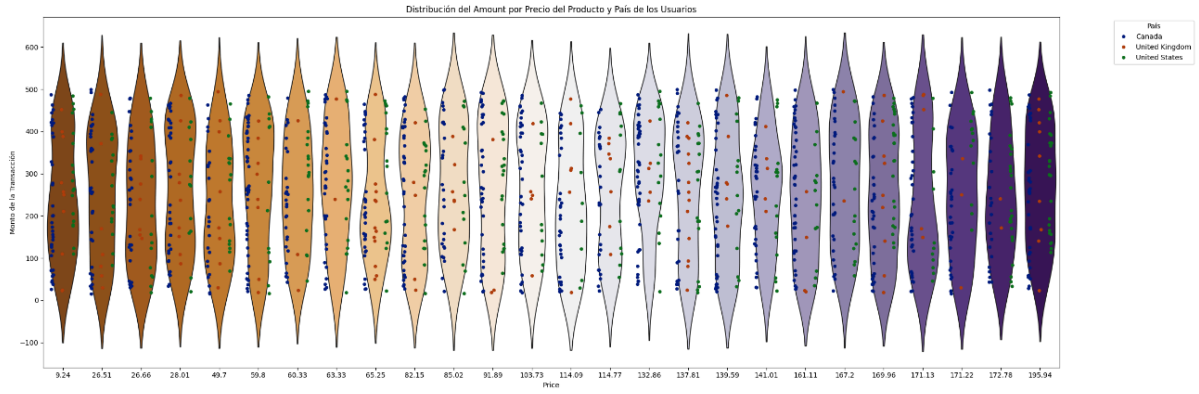
df_products: price df_transactions: id, amount

```
# El código siguiente, que crea un dataframe y quita las filas duplicadas, siempre se ejecuta y actúa como un preámbulo del script:

# dataset = pandas.DataFrame(price, amount)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:

import matplotlib.pyplot as plt
import seaborn as sns

dataset_order = dataset.sort_values(by='amount', ascending=False)

sns.jointplot(x='price', y='amount', data=dataset_order, kind='reg', height=10)
plt.suptitle('Relación entre Precio y Cantidad de productos comprados', y=1.02)
plt.xlabel('Precio de los productos')
plt.ylabel('Amount')
plt.tight_layout()
plt.show()
```

NIVELL 3

Exercici 1

Implementa un violinplot combinat amb un altre tipus de gràfic.



Este es el código usado para el gráfico:

Variables:

df_products: price

df_transactions: id, amount

df_users: country

```
# El código siguiente, que crea un dataframe y quita las filas duplicadas, siempre se ejecuta y actúa como un preámbulo del script:

# dataset = pandas.DataFrame(country, price, amount,id)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:

import seaborn as sns
import matplotlib.pyplot as plt
fig, ax = plt.subplots(figsize=(24, 8))

sns.violinplot(x='price', y='amount', data=dataset, inner=None, ax=ax, palette='PuOr')
sns.stripplot(x='price', y='amount', hue='country', data=dataset, dodge=True, ax=ax, palette='dark')

ax.set_title('Distribución del Amount por Precio del Producto y País de los Usuarios')
ax.set_xlabel('Price')
ax.set_ylabel('Monto de la Transacción')
plt.legend(title='País', bbox_to_anchor=(1.05, 1))
plt.tight_layout()
plt.show()
```