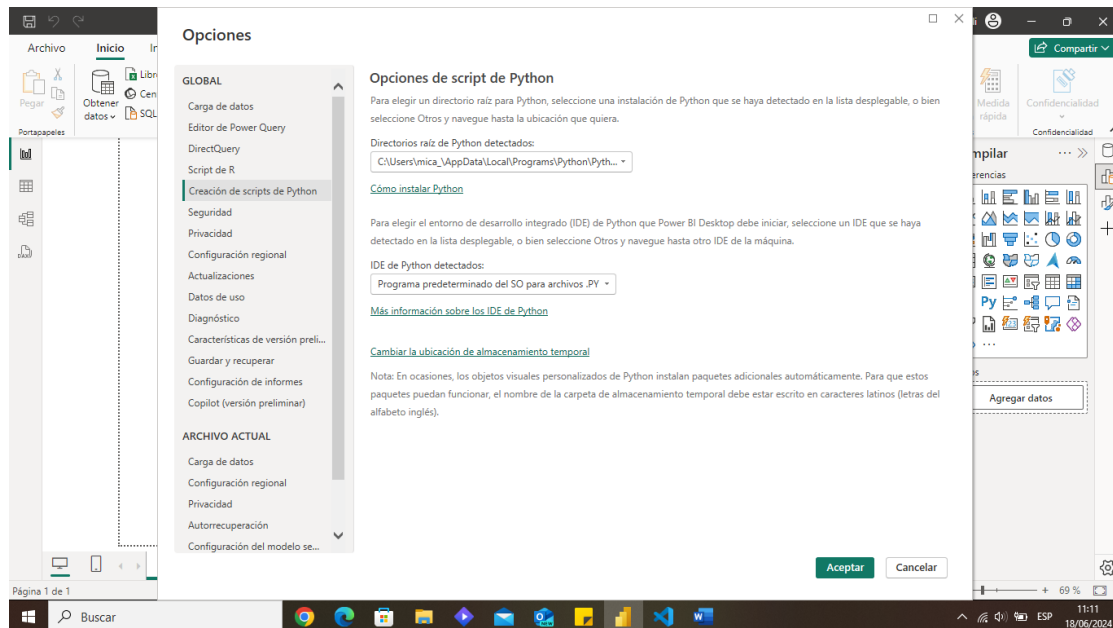


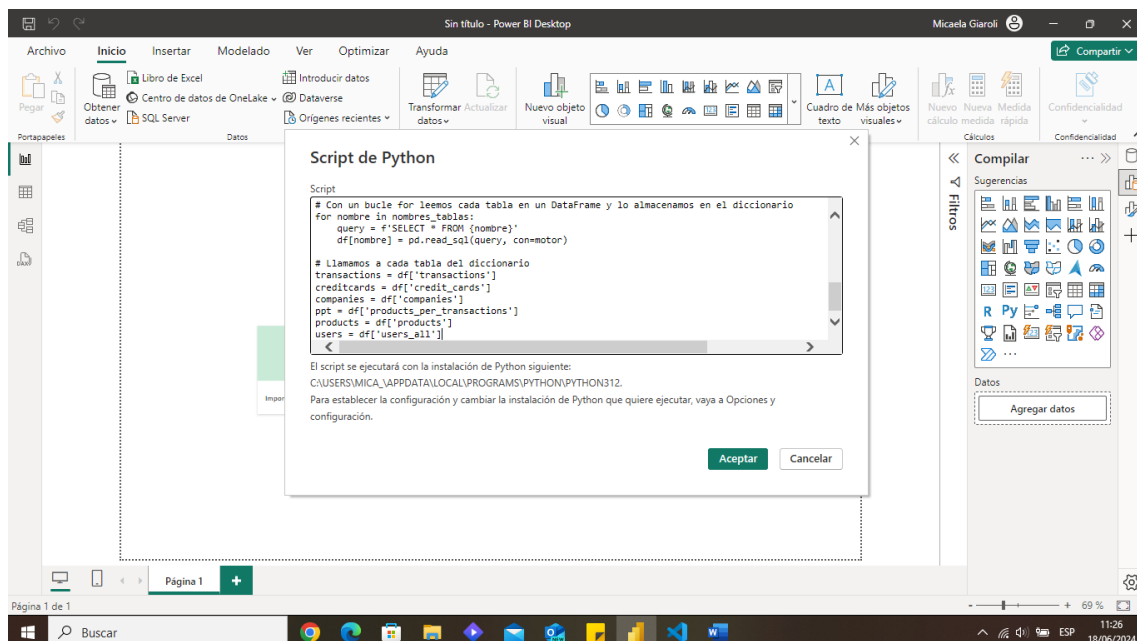
Nivel 1

Para ejecutar los scripts de Python en Power BI realizamos los siguientes pasos:

1. Nos aseguramos de tener instaladas las librerías de Python: Pandas y Matplotlib.
2. Configuramos Power BI:

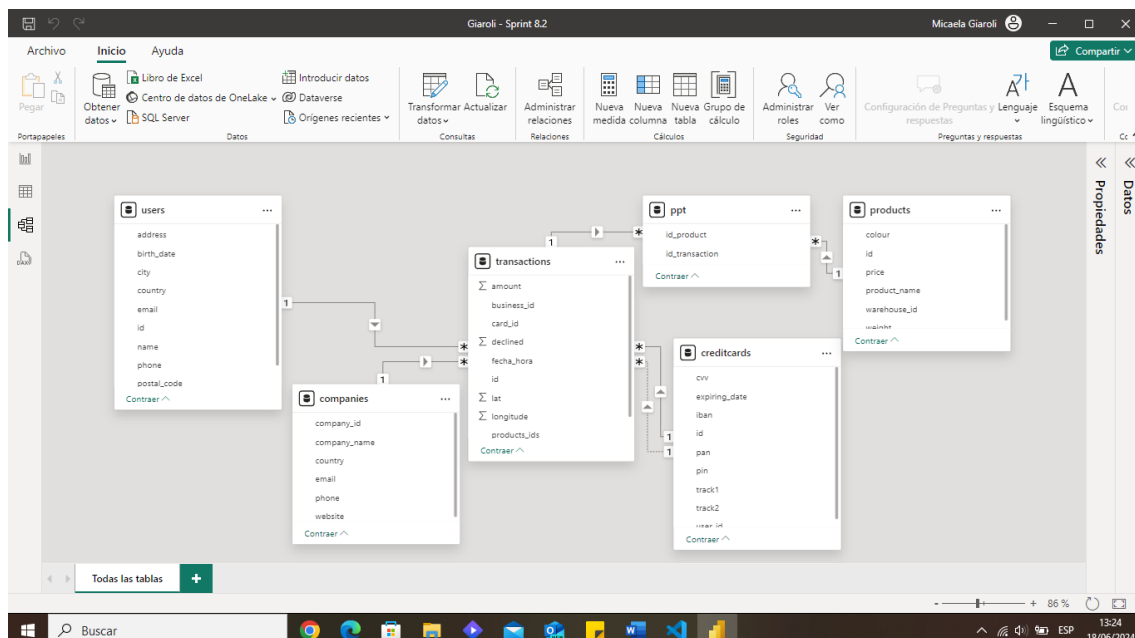


3. Obtenemos los datos copiando el script de Python



4. Antes de cargar los datos, transformamos los campos Price y Amount para reemplazar los puntos por comas, quitar el signo \$ y dar formato de decimal.

5. Una vez importados los datos de cada DF, establecemos las relaciones del modelo.



6. Habilitamos los objetos visuales de script de Python para comenzar a graficar:

Para cada gráfico debemos arrastrar a Valores los campos que utilizaremos y luego adaptamos el código del sprint anterior invocando **'dataset'** para indicar que queremos que el código corra usando los valores seleccionados. Por ej.:



Ejercicio 1 - UNA VARIABLE NUMÉRICA: Recuento de ID Productos

CÓDIGO:

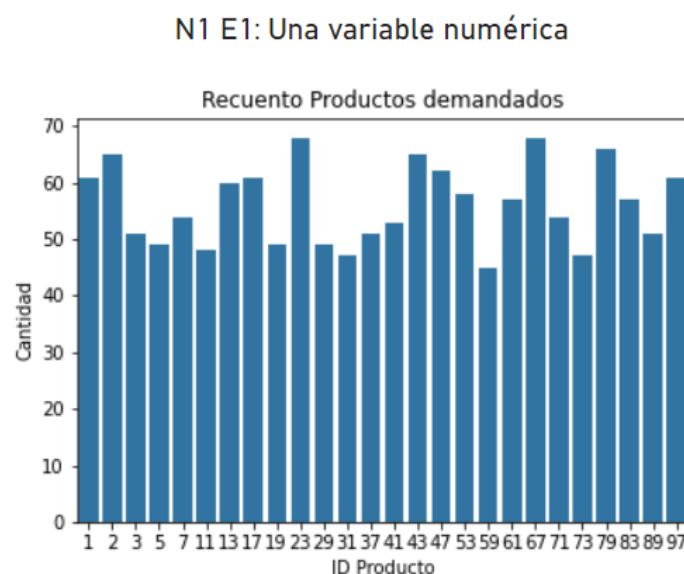
```
# Importamos librerías
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns

sns.countplot(dataset, x='id_product') # Graficamos

plt.xlabel('ID Producto') # Etiqueta para el eje x
plt.ylabel('Cantidad') # Etiqueta para el eje y
plt.title('Recuento Productos demandados') # Título

plt.show() # Imprimimos el gráfico
```

GRÁFICO:



Ejercicio 2 - DOS VARIABLES NUMÉRICAS: Precio y Peso

CÓDIGO:

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

# Ordenamos los datos por 'peso' y luego por 'precio'
data = dataset.sort_values(by=['weight', 'price'])

# Creamos el gráfico de dispersión
scatter = plt.scatter(data['weight'], data['price'], c=data['id'],
                      cmap='viridis', s=100)

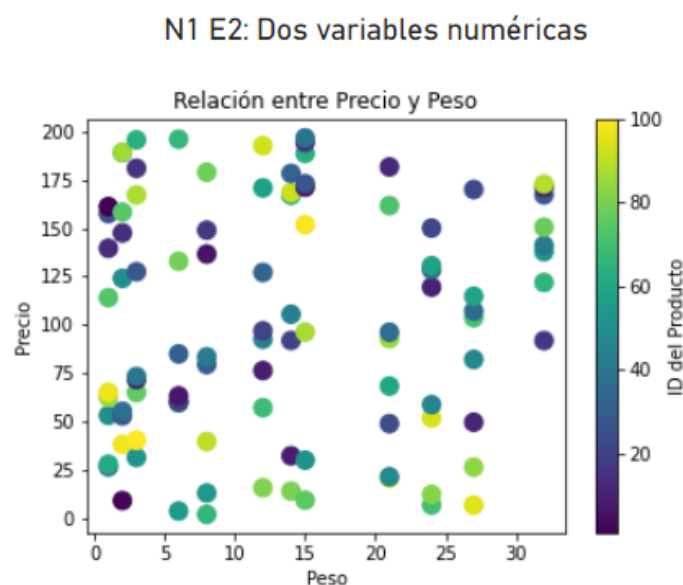
# Añadimos etiquetas y título
plt.xlabel('Peso')
plt.ylabel('Precio')
plt.title('Relación entre Precio y Peso')

# Añadimos una barra de colores
plt.colorbar(scatter, label='ID del Producto')

# Controlamos la densidad de las etiquetas del eje y para mejorar la legibilidad
from matplotlib.ticker import MaxNLocator
plt.gca().yaxis.set_major_locator(MaxNLocator(nbins=10))

plt.show() # Mostramos el gráfico
```

GRÁFICO:



Ejercicio 3 – UNA VARIABLE CATEGÓRICA: Países

CÓDIGO:

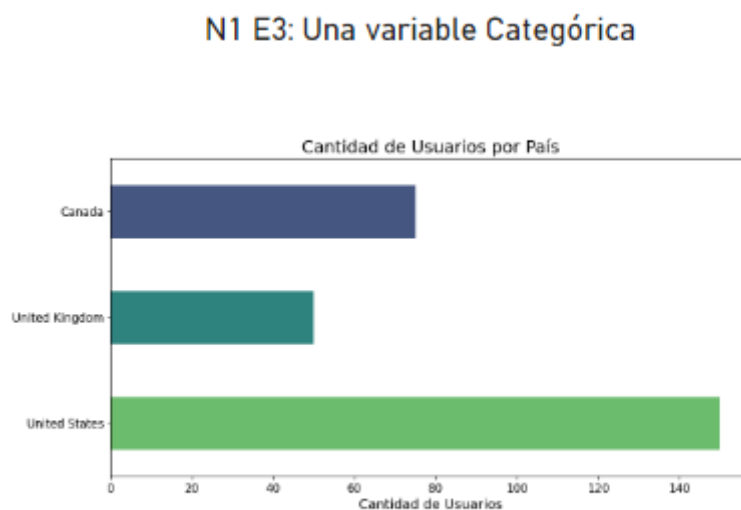
```
# Importamos las librerías
import seaborn as sns
import matplotlib.pyplot as plt

# Creamos el gráfico de conteo, ordenando las barras según frecuencia de
mayor a menor
plt.figure(figsize=(12, 6)) # Ajustamos el tamaño de la figura
sns.countplot(y=dataset['country'], data=dataset, palette='viridis',
width=0.5)

# Añadimos etiquetas y título
plt.xlabel('Cantidad de Usuarios', fontsize=14)
plt.ylabel('País', fontsize=12)
plt.title('Cantidad de Usuarios por País', fontsize=18)
plt.xticks(fontsize=12) # Ajusta el tamaño de fuente de las etiquetas
del eje X
plt.yticks(fontsize=12)

# Mostramos el gráfico
plt.show()
```

GRÁFICO:



Ejercicio 4 – UNA VARIABLE CATEGÓRICA Y UNA NUMÉRICA: Monto y Países

CÓDIGO:

```
import matplotlib.pyplot as plt
import pandas as pd

# Agrupamos los datos por país y calculamos la suma de las transacciones
para cada país
transacciones_por_pais = dataset.groupby('country')['amount'].sum()

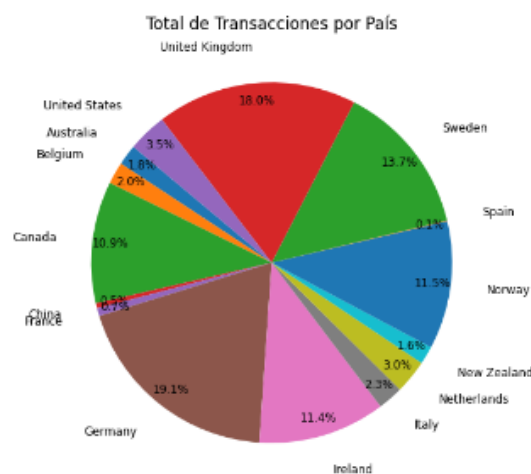
# Creamos el gráfico de pastel
plt.figure(figsize=(6, 6)) # Ajustamos el tamaño de la figura
plt.pie(transacciones_por_pais,
        labels=transacciones_por_pais.index, #etiquetas según el índice
        de la variable transacciones_por_pais (sería país)
        autopct='%1.1f%%', #formato del número, float, con un decimal y
        símbolo de porcentaje
        startangle=140, #ángulo donde comienza a dibujarse el pastel
        textprops={'fontsize': 8.5}, #tamaño del texto
        pctdistance=0.9, #distancia de los porcentajes
        labeldistance=1.2) #distancia de las etiquetas

# Añadimos título
plt.title('Total de Transacciones por País')

# Mostramos el gráfico
plt.show()
```

GRÁFICO:

N1 EJ4: Una Categórica y una Numérica



Ejercicio 5 – DOS VARIABLES CATEGÓRICAS: Países y Declined

CÓDIGO:

```
import pandas as pd
import matplotlib.pyplot as plt

# Separamos en dos tablas a partir del 'dataset'
companies = dataset[['company_id', 'country']]
transactions = dataset[['business_id', 'declined']]

# Unimos las tablas 'transactions' y 'companies' por los campos
# 'business_id' y 'company_id'
datos = pd.merge(transactions, companies, how='inner',
left_on='business_id', right_on='company_id')

# Agrupamos los datos por país y contamos las transacciones declinadas
# Con unstack() pivotamos el índice interno del dataframe resultante para
# generar columnas con los valores únicos de 'declined'
agrupados = datos.groupby('country')['declined'].value_counts().unstack()

# Renombramos las columnas para que en el gráfico, en lugar de 1 y 0,
# diga Aceptadas y Rechazadas con el atributo .columns
agrupados.columns = ['Aceptadas', 'Rechazadas']

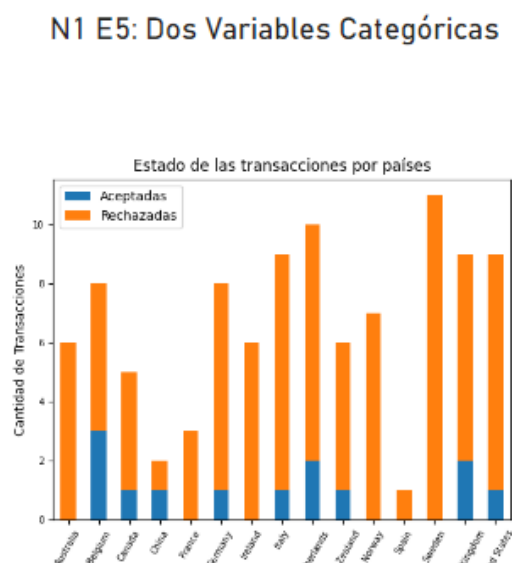
# Graficamos
agrupados.plot(kind='bar', stacked=True, fontsize=7) # Con stacked=True
hacemos que las columnas sean apiladas

# Colocamos título y etiquetas a los ejes
plt.title('Estado de las transacciones por países')
plt.xlabel('País')
plt.ylabel('Cantidad de
Transacciones')

# Rotamos las etiquetas del
eje x
plt.xticks(rotation=60)

plt.show()
```

GRÁFICO:



Ejercicio 6 – TRES VARIABLES: Monto, Fecha de Nacimiento y País de usuario

CÓDIGO:

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Convertimos birth_date a datetime y calculamos la edad
dataset['birth_date'] = pd.to_datetime(dataset['birth_date'])
dataset['age'] = (pd.Timestamp.now() - dataset['birth_date']).dt.days //
365  #// es una división entera que redondea al número entero hacia abajo

# Creamos grupos de edad
bins = [0, 19, 36, 51, np.inf] #np.inf es como si fuera un límite
infinito
labels = ['0-18', '19-35', '36-50', '51+']
dataset['age_group'] = pd.cut(dataset['age'], bins=bins, labels=labels,
right=False) #right=False indica intervalos semi-abiertos en el extremo
derecho ej: [19, 36)

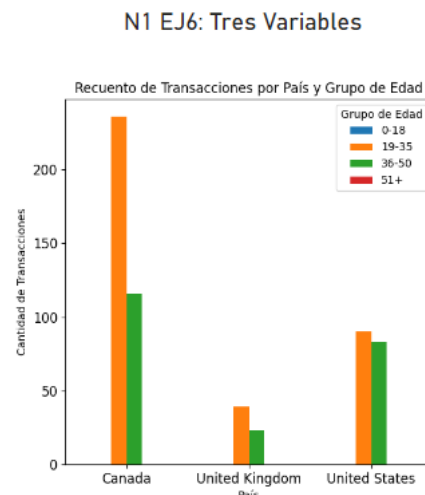
# Agrupamos los datos por país y por grupo de edad y contamos las
transacciones
grouped = dataset.groupby(['country', 'age_group'],
observed=False).size().unstack(fill_value=0) #size() cuenta la cantidad
de filas por grupo en el df

# Creamos el gráfico de barras agrupadas, damos formato a tamaño y texto
grouped.plot(kind='bar', stacked=False, figsize=(6, 6), rot=0,
fontsize=12)

# Colocamos título y etiquetas a los ejes
plt.title('Recuento de Transacciones por País y Grupo de Edad')
plt.xlabel('País')
plt.ylabel('Cantidad de
Transacciones')
plt.legend(title='Grupo de Edad')

# Mostramos el gráfico
plt.show()
```

GRÁFICO:



Ejercicio 7 – PAIRPLOT: de los DF Transactions y Companies

CÓDIGO:

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

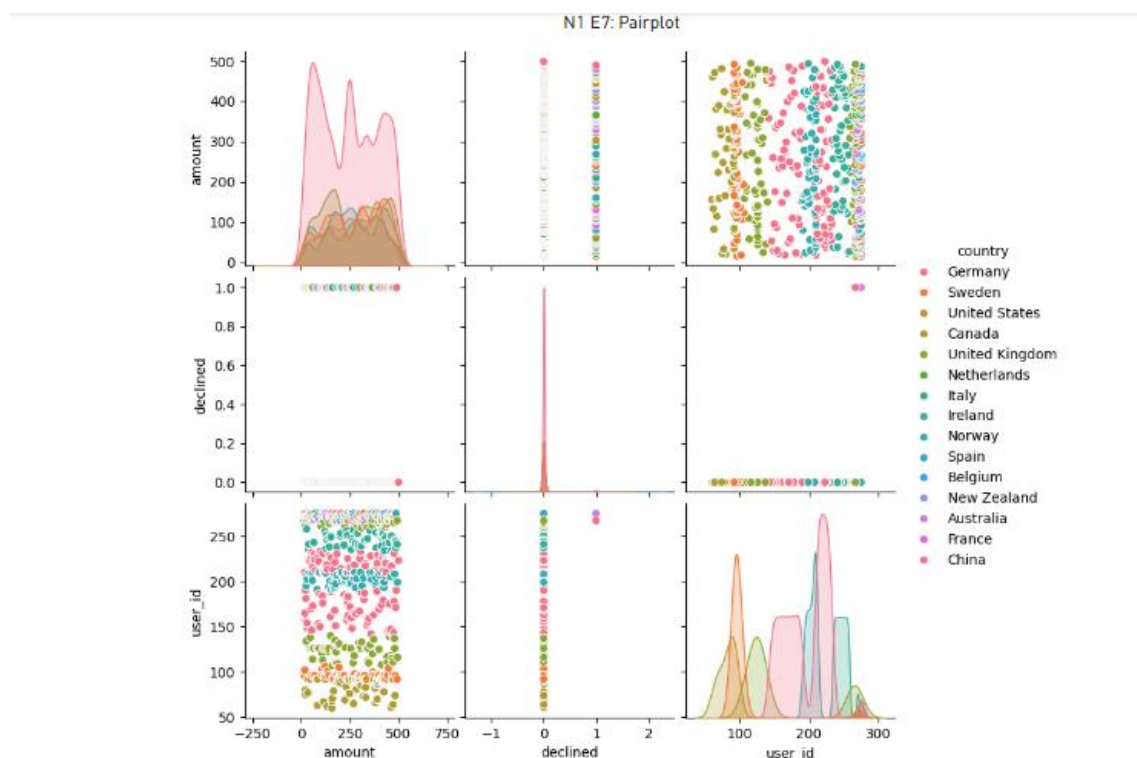
# Separamos en dos tablas a partir del dataset
companies = dataset[['company_id', 'country']]
transactions = dataset[['amount', 'business_id', 'declined', 'user_id']]

# Unimos los DataFrames
datosmerged = pd.merge(transactions, companies, left_on='business_id',
                        right_on='company_id')

# Graficamos
sns.pairplot(data=datosmerged, hue="country") #diferenciamos los países
por color para poder tener otro nivel de análisis

plt.show()
```

GRÁFICO:



Nivel 2

Ejercicio 1 - Heatmap

CÓDIGO:

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

# Seleccionamos los campos que queremos analizar
seleccion = dataset[['amount', 'user_id', 'declined', 'id_product',
                    'price', 'weight']]

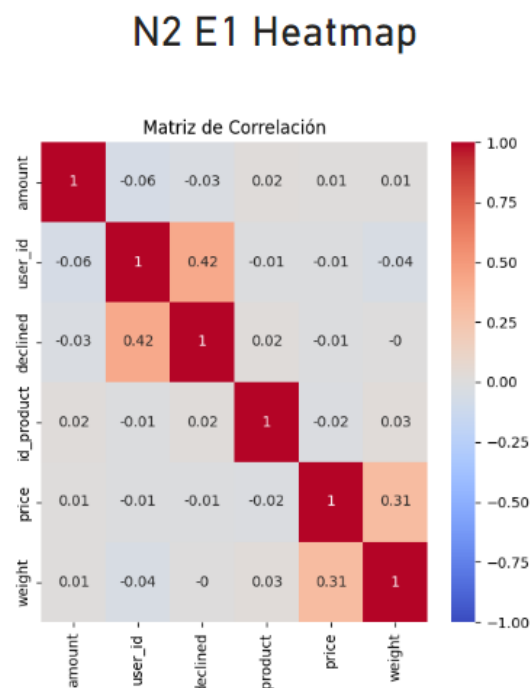
# Primero calculamos la matriz de correlación, redondeamos a 2 decimales
matriz_correlacion = seleccion.corr().round(2)

# Configuramos el tamaño del gráfico
plt.figure(figsize=(6, 6))

# Creamos el heatmap
sns.heatmap(matriz_correlacion,
            annot=True,          #annot=true añade los valores a los
recuadros
            cmap='coolwarm', # con esto le damos el estilo de colores
            vmin=-1, vmax=1) # establecemos valores máximos y mínimos
                             para que coincida con el coeficiente de correlación

# Mostramos el gráfico
plt.title('Matriz de
Correlación')
plt.show()
```

GRÁFICO:



Ejercicio 2 - Joinplot

CÓDIGO:

Queremos analizar la relación entre id_usuario y amount, pero también colorear según el país de usuario

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

```
# Definimos el tamaño del lienzo, la paleta de colores y el estilo
plt.figure(figsize=(12, 10))
```

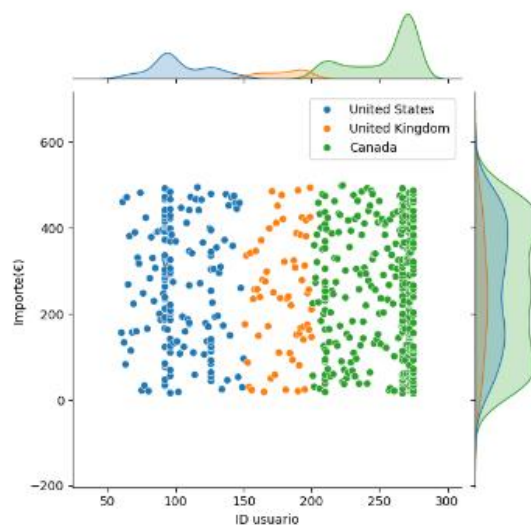
```
# Creamos el gráfico y colocamos etiquetas a los ejes
grafico = sns.jointplot(x='user_id', y='amount', data=dataset,
hue='country')
plt.xlabel('ID usuario')
plt.ylabel('Importe(€)')
```

```
# Reubicamos la leyenda
grafico.ax_joint.legend(loc='upper right', bbox_to_anchor=(1, 1))
```

```
plt.show()
```

GRÁFICO:

N2 E 2: Joinplot Monto vs User_id



Nivel 3

Ejercicio 1: Violinplot y gráfico de dispersión

CÓDIGO:

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

# Creamos el lienzo para colocar los dos gráficos con una fila y dos
columnas
fig, axs = plt.subplots(1, 2, figsize=(12, 6))

# Nombramos ax1 y ax2 a cada uno de los gráficos para que sea más fácil
trabajar con sus títulos y ejes
# Primer gráfico: violinplot
ax1 = sns.violinplot(x='amount',
                    y='country',
                    hue='declined',
                    data= dataset,
                    ax=axs[0])

# Cambiamos el nombre de los ejes y añadimos título al primer gráfico
ax1.set_xlabel("Importe(€)", fontsize=14)
ax1.set_ylabel("País", fontsize=14)
ax1.set_title('Distribución del Importe', fontsize=16)

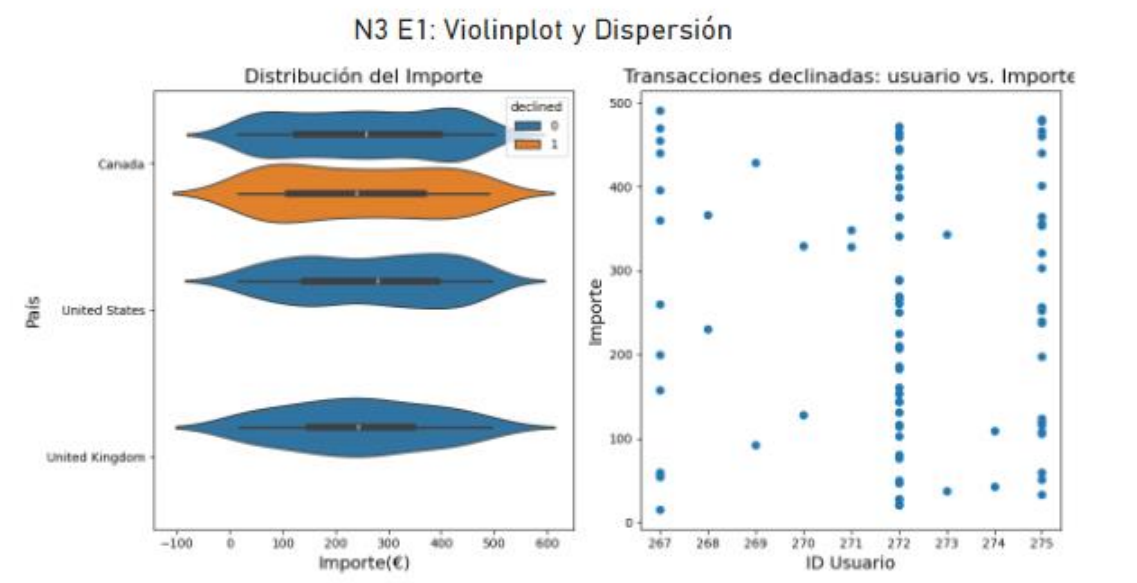
# Para el segundo gráfico filtraremos el df sólo por las transacciones
declinadas para ver la relación entre user_id y amount en un gráfico de
dispersión
transacciones_declinadas = dataset[dataset['declined'] == 1]

# Segundo gráfico: gráfico de dispersión
ax2 = axs[1].scatter(x=transacciones_declinadas['user_id'], #el eje al
                    usar matplotlib se pasa antes! no como parámetro como en seaborn
                    y=transacciones_declinadas['amount'])

# Cambiamos el nombre de los ejes y añadimos el título para el segundo
gráfico
axs[1].set_xlabel('ID Usuario', fontsize=14)
axs[1].set_ylabel('Importe', fontsize=14)
axs[1].set_title('Transacciones declinadas: usuario vs. Importe',
                fontsize=16)

fig.tight_layout() #Ajusta automáticamente los parámetros para que se
ajusten dentro de la figura

plt.show()
```

GRÁFICO:**Ejercicio 2: FacetGrid****CÓDIGO:**

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

# Seleccionamos los campos que queremos analizar
datos_seleccionados = dataset[['amount', 'declined', 'country']]

# Graficamos
plt.figure(figsize=(12, 10))
g = sns.FacetGrid(datos_seleccionados, col='country', col_wrap = 5, hue=
'declined')
g.map(plt.hist, 'amount')

# Añadimos la leyenda
g.add_legend()
g.set_ylabels('Cantidad')

#Colocamos título
plt.subplots_adjust(top=0.9) # Ajusta el espacio superior para el título
g.fig.suptitle('Importe, Cantidad y Estado de las Transacciones por
País') #suptitle coloca el título en la parte de arriba

plt.show()
```

GRÁFICO:

