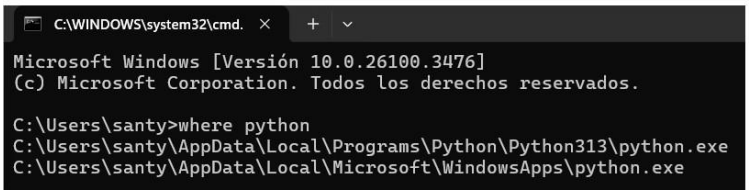
Tasca S8.02. Power BI amb Python

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.

Primero voy a descubrir donde tengo el ejecutable de Python en mi ordenador

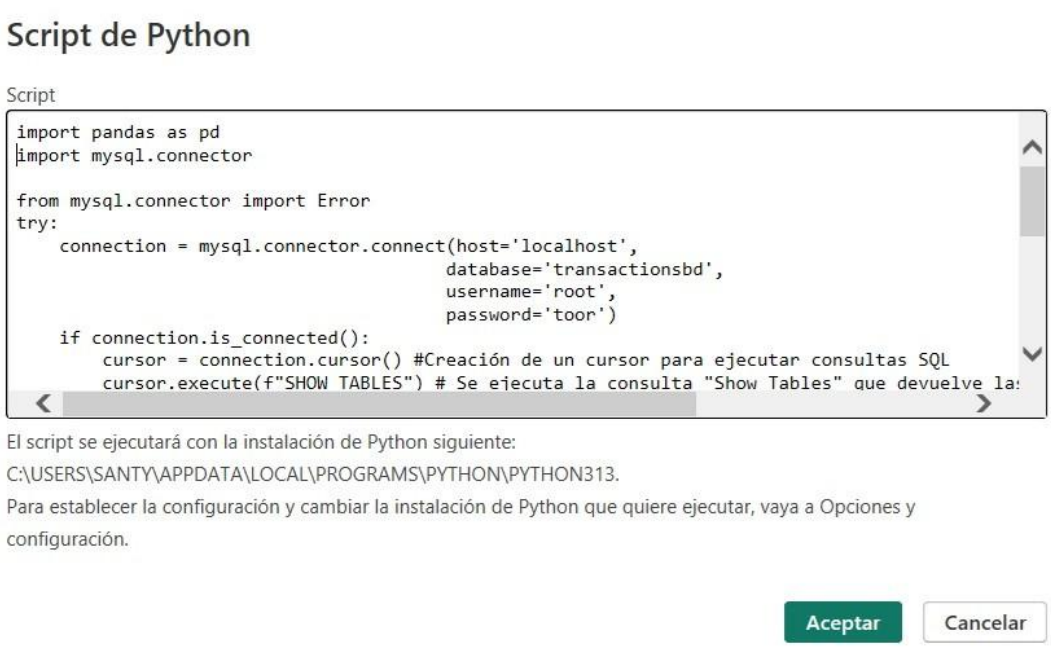


Esta información nos será muy útil para crear la conexión entre Power BI y Python en nuestro ordenador.

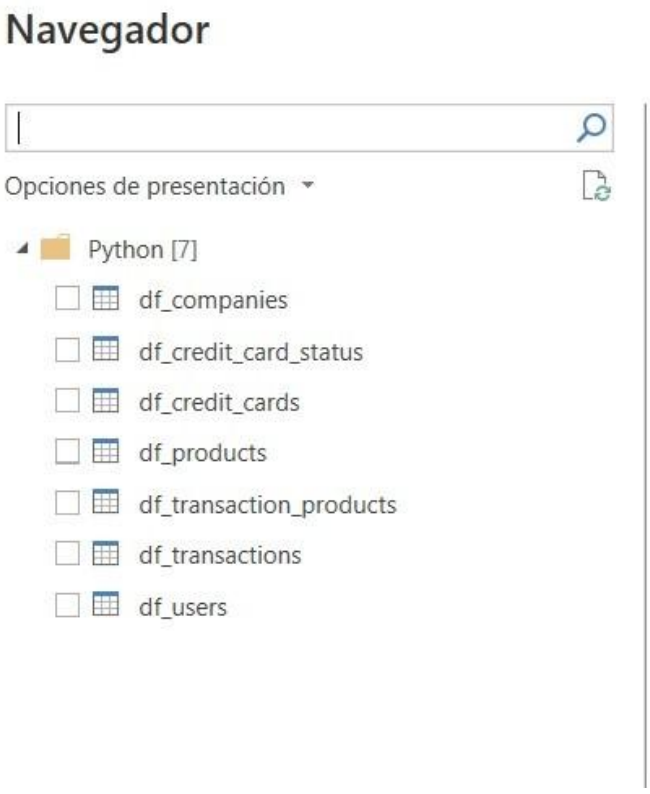
Vamos a configurar Power BI para poder usar Python:

Vamos a usar Python para poder instalar nuestra base de datos

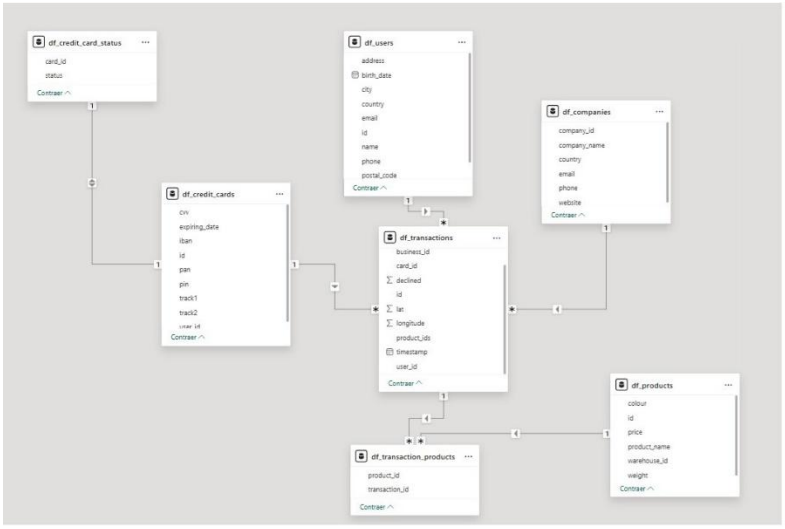
Nos saldrá el cuadro de dialogo donde podremos poner es script de Python que sea necesario para cargar la base de datos deseada. En este caso voy a poner el mismo que utilice en el Ejercicio 8.1



Y me carga las tablas de la base de datos transactionbd, transformándolas en dataframes.



Creamos las relaciones entre las tablas.



He realizado algunas transformaciones con Power Query como convertir algunas columnas a números decimales ya que Power BI los reconocía como texto al tener un punto en lugar de una coma para separar los decimales. También transforme el timestamp a Fecha/Hora ya que lo reconocía como texto.

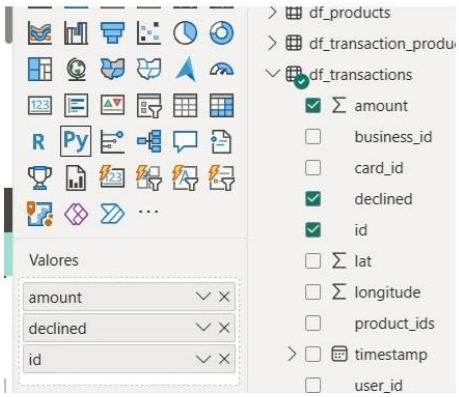
Ahora ya lo tenemos todo a punto para hacer los gráficos con Python y Power BI. Cogeremos como base lo que hicimos en el sprint 8_1.

Nivell 1

Els 7 exercicis del nivell 1 de la tasca 01

Ejercicio 1

Escogeremos la visualización de Python (Py) y marcaremos las siguientes columnas:

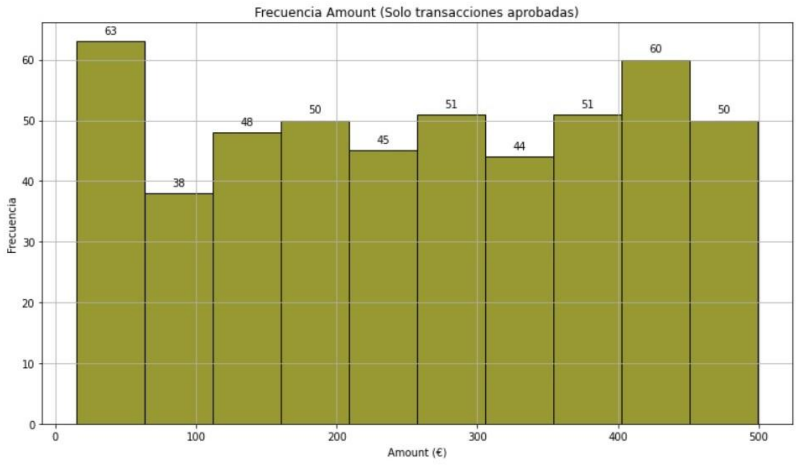


Basándonos en el código realizado en el 8.01 E1.1, realizamos las modificaciones necesarias para que funcione. El código resultante es el siguiente:

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

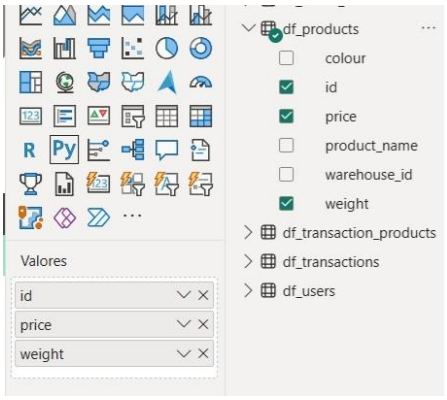
```
# dataset = pandas.DataFrame(amount, declined, id)
# dataset = dataset.drop_duplicates()

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

# Crear el gráfico de histograma
ax = sns.histplot(data=dataset[dataset['declined'] == 0], x="amount", bins=10, color="olive", edgecolor="black", alpha=0.8)

# Agregar etiquetas con los valores en cada barra
for patch in ax.patches:
   height = patch.get_height()  # Obtiene la altura de la barra (frecuencia)
   ax.annotate(f'{int(height)}',  # Convierte la frecuencia a entero
            xy=(patch.get_x() + patch.get_width() / 2, height),  # Posición del texto
            xytext=(0, 5),  # Desplazamiento en Y
            textcoords="offset points",
            ha='center', va='bottom', fontsize=10, color='black')

# Configuración del gráfico
plt.grid()
plt.title('Frecuencia Amount (Solo transacciones aprobadas)')
plt.ylabel('Frecuencia')
plt.xlabel('Amount (€)')

# Mostrar el gráfico
plt.show()
```

No da el siguiente resultado:



Ejercicio 2

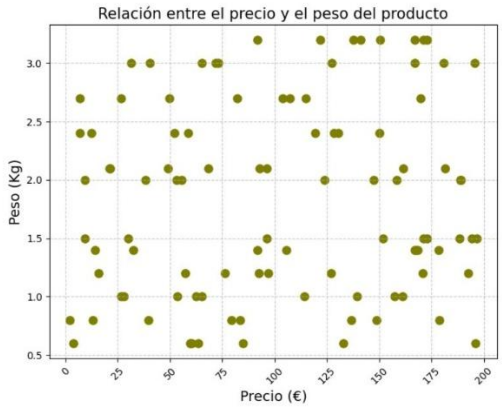
Escogeremos la visualización de Python (Py) y marcaremos las siguientes columnas:



Basándonos en el código realizado en el 8.01 E1.2, realizamos las modificaciones necesarias para que funcione. El código resultante es el siguiente:

```
# 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, price, weight)
# dataset = dataset.drop_duplicates()

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

price = dataset['price']
weight = dataset['weight']

# Crear gráfico de dispersión
plt.figure(figsize=(8, 6))
plt.scatter(x=price, y=weight, color='olive', s=65)

# Etiquetas de los ejes y título
plt.xlabel('Precio (€)', fontsize=14)
plt.ylabel('Peso (Kg)', fontsize=14)
plt.title("Relación entre el precio y el peso del producto", fontsize=15)

# Mostrar cuadrícula
plt.grid(True, linestyle='--', alpha=0.7)
plt.xticks(rotation=45, ha='center')

# Mostrar gráfico
plt.show()
plt.show()
```

No da el siguiente resultado:



Ejercicio 3

Escogeremos la visualización de Python (Py) y marcaremos las siguientes columnas:

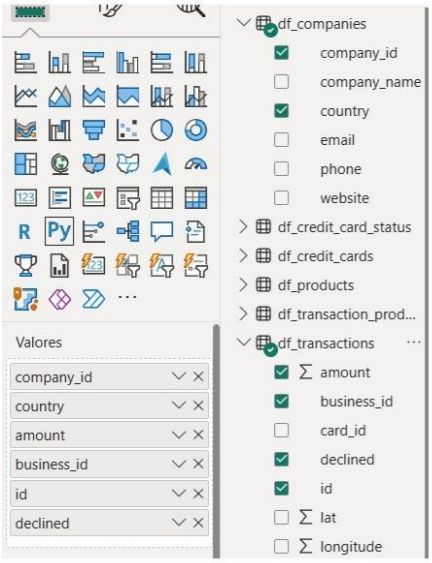
Basándonos en el código realizado en el 8.01 E1.3, realizamos las modificaciones necesarias para que funcione. El código resultante es el siguiente:

```
# 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, country)
# dataset = dataset.drop_duplicates()

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

# Guardar la lista de países en orden descendente según la cantidad de empresas
orden_country = dataset['country'].value_counts().index

# Crear la figura
plt.figure(figsize=(10, 7))

# Generar el gráfico de barras con un solo color
ax = sns.countplot(y='country', data=dataset, order=orden_country, color='olive', width=0.6)

# Etiquetas de los ejes y título
plt.xlabel('Cantidad de empresas', fontsize=20)
plt.ylabel('País', fontsize=20)
plt.title('Cantidad de empresas por país', fontsize=20)

# Ajustar tamaño de las etiquetas en los ejes
plt.xticks(fontsize=14)
plt.yticks(fontsize=14)

# Agregar etiquetas con valores en las barras
for container in ax.containers:
    ax.bar_label(container, fmt='%d', fontsize=12, padding=5)

# Mostrar el gráfico
plt.show()
```

No da el siguiente resultado:

Ejercicio 4

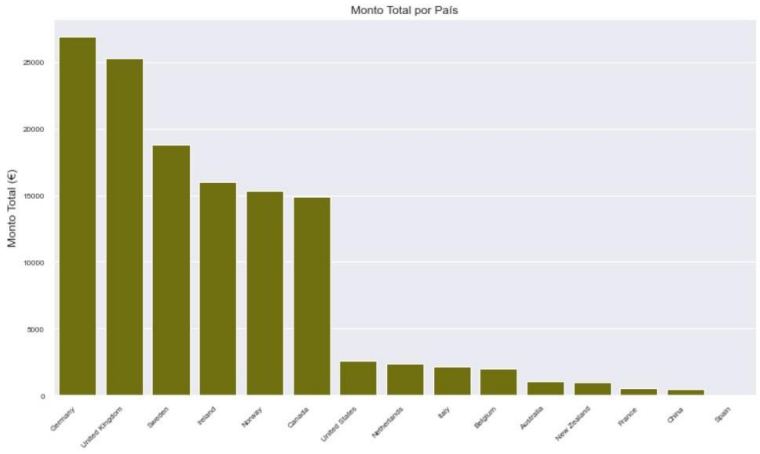
Escogeremos la visualización de Python (Py) y marcaremos las siguientes columnas:



Basándonos en el código realizado en el 8.01 E1.4, realizamos las modificaciones necesarias para que funcione. El código resultante es el siguiente:

```
# 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, country, amount, business_id, id, declined)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:

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

# Filtrar solo las transacciones aprobadas (declined = 0)
df_transactions_approved = dataset[dataset["declined"] == 0]

# Agrupar por país y sumar los montos
df_transactions_companies_group =
df_transactions_approved.groupby("country")["amount"].sum().sort_values(ascending=False).reset_index()

# Configurar el tema del gráfico
sns.set_theme(style='darkgrid')

# Crear el gráfico de barras
sns.barplot(
   data=df_transactions_companies_group, x="country", y="amount", color="olive")

# Configuración del título y etiquetas
plt.title("Monto Total por País")
plt.xticks(rotation=45, ha="right")
plt.xlabel("Pais", fontsize=15)
plt.ylabel("Monto Total (€)", fontsize=12)
plt.tick_params(axis='both', labelsize=8)

# Mostrar el gráfico
plt.show()
```

No da el siguiente resultado:



Ejercicio 5

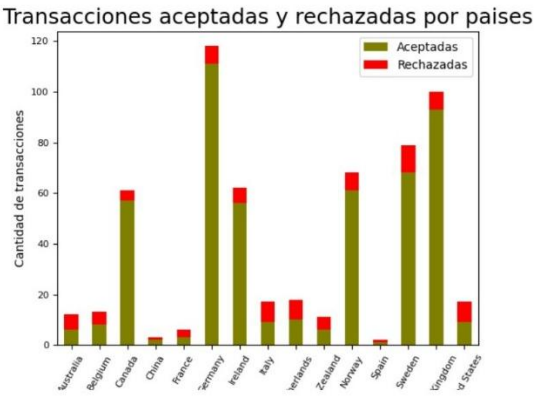
Escogeremos la visualización de Python (Py) y marcaremos las siguientes columnas:

Basándonos en el código realizado en el 8.01 E1.5, realizamos las modificaciones necesarias para que funcione. El código resultante es el siguiente:

```
# 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, declined, id)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:

import pandas as pd
import matplotlib.pyplot as plt

# Contar transacciones aceptadas (declined = 0) y rechazadas (declined = 1) por país
data_agrupada = dataset.groupby('country')['declined'].value_counts().unstack(fill_value=0)

# Renombrar columnas (suponiendo que False = 0 = Aceptadas y True = 1 = Rechazadas)
data_agrupada.columns = ['Aceptadas', 'Rechazadas']

# Colores personalizados para cada barra
colors = ['olive', 'red']  # Aceptadas = olive, Rechazadas = red

# Graficar
data_agrupada.plot(kind='bar', stacked=True, fontsize=8, color=colors)

plt.title('Transacciones aceptadas y rechazadas por paises', fontsize=18)
```

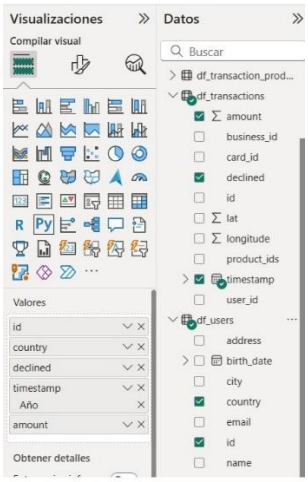
```
plt.xlabel('Pais')
plt.ylabel('Cantidad de transacciones')
plt.xticks(rotation=60)

plt.show()
```

No da el siguiente resultado:



Ejercicio 6

Escogeremos la visualización de Python (Py) y marcaremos las siguientes columnas:

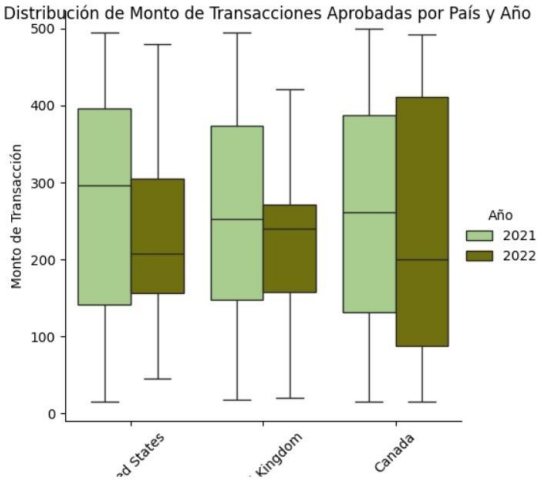


Basándonos en el código realizado en el 8.01 E1.6, realizamos las modificaciones necesarias para que funcione. El código resultante es el siguiente:

```
# 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, country, declined, Año, amount)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:

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

# Filtrar solo las transacciones aprobadas
df_transactions_filtered = dataset[dataset['declined'] == 0]

# Definir la paleta de colores personalizada
palette_colors = {2021: "#A6D785", 2022: "#808000"}  # Verde claro y oliva
```

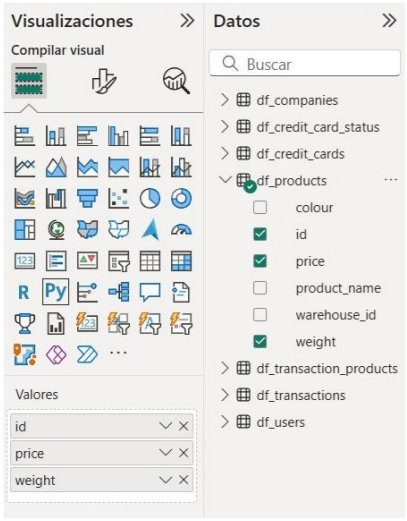
```
# Crear el gráfico de cajas con la paleta personalizada
plot = sns.catplot(
    data=dataset,
    kind='box',
    x='country',
    y='amount',
    hue='Año',
    palette=palette_colors
)

# Añadir etiquetas y título
plot.set_axis_labels("País", "Monto de Transacción")  # Etiquetas de los ejes
plot.fig.suptitle("Distribución de Monto de Transacciones Aprobadas por País y Año")  # Título del gráfico

# Rotar las etiquetas del eje X si hay muchos países
plt.xticks(rotation=45)

plt.show()
```

No da el siguiente resultado:



Ejercicio 7

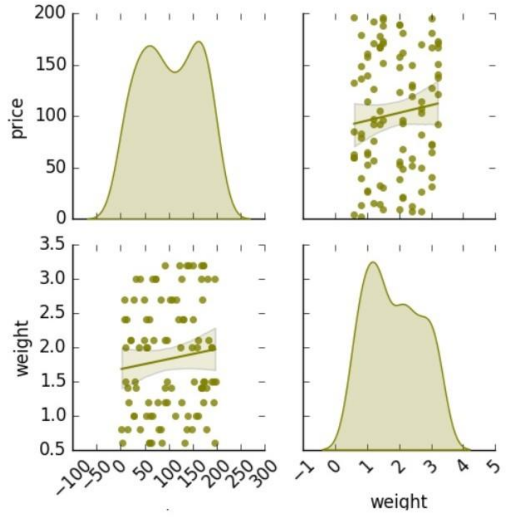
Escogeremos la visualización de Python (Py) y marcaremos las siguientes columnas:



Basándonos en el código realizado en el 8.01 E1.7, realizamos las modificaciones necesarias para que funcione. El código resultante es el siguiente:

```
# 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, price, weight)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:

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

# Estilo clásico de Matplotlib
plt.style.use('classic')

# Crear pairplot con densidad en la diagonal y regresión en los gráficos de dispersión
pairplot = sns.pairplot(dataset, vars=['price', 'weight'], diag_kind='kde', kind='reg',
                        plot_kws={'color': 'olive'}, diag_kws={'color': 'olive'})

# Título de la figura
pairplot.figure.suptitle('Pairplot del precio y el peso de los productos', fontsize=16, y=1.1)

# Rotar etiquetas del eje X
for ax in pairplot.axes.flatten():
    plt.setp(ax.get_xticklabels(), rotation=45)

# Mostrar gráfico
plt.show()
```

No da el siguiente resultado:

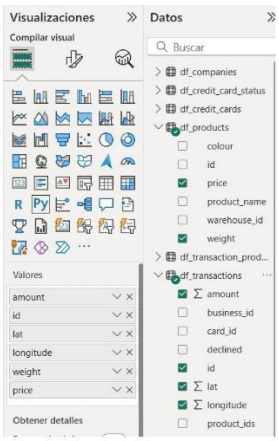


Nivell 2

Els 2 exercicis del nivell 2 de la tasca 01

Ejercicio 1

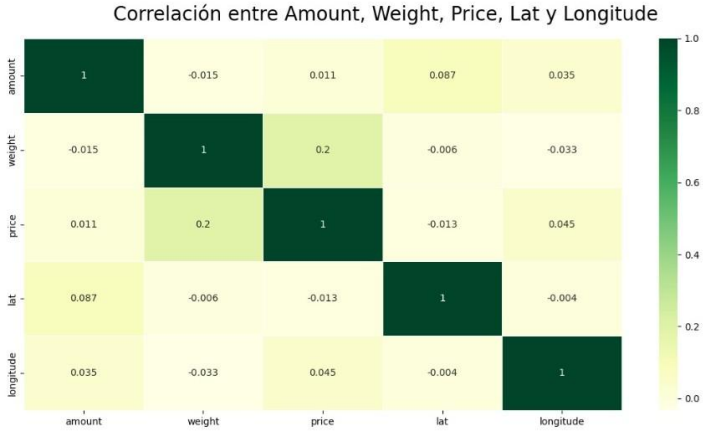
Escogeremos la visualización de Python (Py) y marcaremos las siguientes columnas:



Basándonos en el código realizado en el 8.01 E2.1, realizamos las modificaciones necesarias para que funcione. El código resultante es el siguiente:

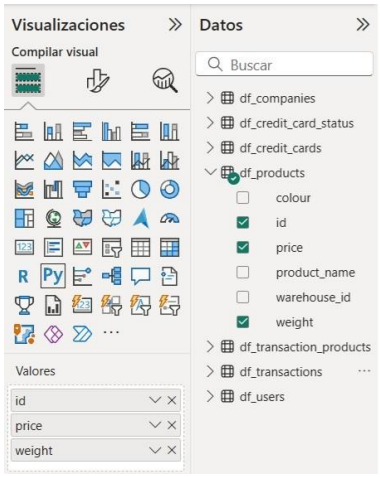
```
# El código siguiente, que crea un dataframe y quita las filas duplicadas, siempre se ejecuta y actúa como un preámbulo del script:

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

# Pegue o escriba aquí el código de script:

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

# Seleccionar las columnas numéricas
df_numeric = dataset[["amount", "weight", "price", "lat", "longitude"]]

# Calcular la correlación
Correlation = round(df_numeric.corr(), 3)

# Configurar la figura
plt.figure(figsize=(14, 7))

# Graficar el heatmap con una paleta de colores oliva
sns.heatmap(data=Correlation, annot=True, cmap="YlGn", linewidths=0.5)

# Agregar título
plt.suptitle("Correlación entre Amount, Weight, Price, Lat y Longitude", fontsize=20, y=0.95)

# Mostrar la gráfica
plt.show()
```

No da el siguiente resultado:



Ejercicio 2

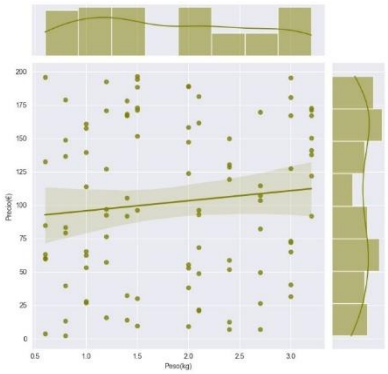
Escogeremos la visualización de Python (Py) y marcaremos las siguientes columnas:



Basándonos en el código realizado en el 8.01 E2.2, realizamos las modificaciones necesarias para que funcione. El código resultante es el siguiente:

```
# 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, price, weight)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:

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

# Establece el estilo de la gráfica con una cuadrícula oscura
sns.set_style('darkgrid')

# Crea un gráfico conjunto (jointplot)
ax2 = sns.jointplot(x="weight", y="price", data=dataset, kind="reg", color="olive", height=8)

# Agrega un título a la figura principal
ax2.fig.suptitle('"Relacion entre peso y precio"', y=1.02)

# Ajusta automáticamente el diseño para evitar solapamientos
ax2.fig.tight_layout()

# Ajusta el espacio superior del título para que no se corte
ax2.fig.subplots_adjust(top=0.95)

# Etiquetas de los ejes
ax2.set_axis_labels("Peso(kg)", "Precio(€)")

# Muestra la gráfica
plt.show()
```

No da el siguiente resultado:

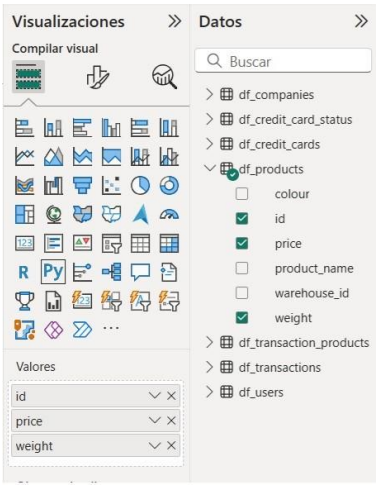


Nivell 3

Els 2 exercicis del nivell 3 de la tasca 01

Ejercicio 1

Escogeremos la visualización de Python (Py) y marcaremos las siguientes columnas:

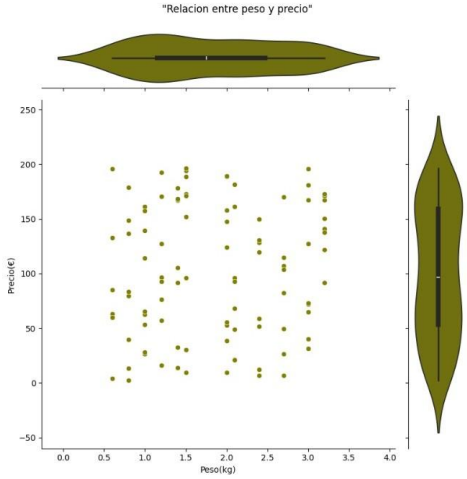


Basándonos en el código realizado en el 8.01 E3.1, realizamos las modificaciones necesarias para que funcione. El código resultante es el siguiente:

```
# 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, price, weight)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:

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

# Crear la figura
plt.figure(figsize=(15, 8))

# Crear el JointGrid
ax3 = sns.JointGrid(data=dataset, x="weight", y="price")
ax3.fig.set_size_inches(8, 8)

# Graficar los puntos (scatterplot) y las violín (violinplot)
```

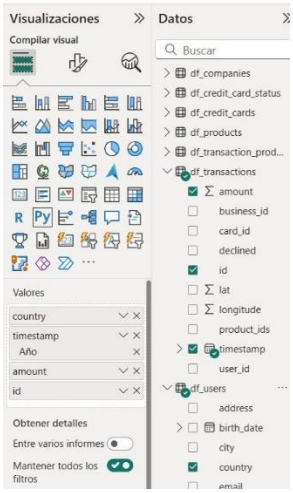
```
ax3.plot(sns.scatterplot, sns.violinplot, color="olive")

# Establecer el título del gráfico
ax3.fig.suptitle('"Relacion entre peso y precio"', y=1)

# Ajustar el diseño de la figura para evitar superposiciones
ax3.fig.tight_layout()

# Ajustar el espacio superior para que el título no se solape
ax3.fig.subplots_adjust(top=0.95)

# Establecer las etiquetas de los ejes X y Y
ax3.set_axis_labels("Peso(kg)", "Precio(€)")

# 9. Mostrar el gráfico final
plt.show()
```

No da el siguiente resultado:



Ejercicio 2

Escogeremos la visualización de Python (Py) y marcaremos las siguientes columnas:



Basándonos en el código realizado en el 8.01 E3.2, realizamos las modificaciones necesarias para que funcione. El código resultante es el siguiente:

```
# 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, Año, amount, id)
# dataset = dataset.drop_duplicates()

# Pegue o escriba aquí el código de script:

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

# Configura el estilo del gráfico
sns.set_style('white')

# Crea el FacetGrid usando 'country' para las columnas y 'Year' para las filas
g = sns.FacetGrid(dataset, col='country', row='Año')

# Mapea el gráfico de histograma a cada subgráfico con color 'olive'
g.map(sns.histplot, 'amount', color='olive', hue=None)

# Establece los títulos para las filas y columnas
g.set_titles(col_template='{col_name}', row_template='{row_name}')

# Muestra el gráfico
plt.show()
```

No da el siguiente resultado: