



# ***Inove Coding School***

Proyecto Final

*“Python Analytics”*

**Alumno:**

**Valentín Marco Imperio**

**Profesores:**

**Johana Rangel**

**Julián Salinas**

**Fecha de Entrega:**

**28/01/2023**



## **Introducción**

El objetivo del presente proyecto final es aplicar los conocimientos aprendidos a lo largo del curso *Programador Python*. Particularmente se trabajará con los siguientes conceptos:

- Funciones y variables.
- Ciclos y bucles *for*.
- Listas, Tuplas y Diccionarios.
- Comprensión de Listas.
- Arrays y la librería Numpy.
- Bases de datos.
- SQL ORM con SQL Alchemy.
- Modularización.
- Gráficos con Matplotlib.
- Utilización de las Buenas Prácticas de la Programación.

Dado que el tema elegido es el análisis de un dataset de ventas de calzado internacionales, se explicará el proyecto y su aplicación en el próximo apartado.

## **Desarrollo**

Se explica a continuación en orden cronológico el objetivo de cada función utilizada.

### **Función `read_db()`**

La misma se utiliza para la limpieza, ordenamiento y lectura de los datos.

```
Producto ID 2152 del país United Kingdom, género
tamaño 11, precio $159.00 en la fecha 1/1/2014
Producto ID 2230 del país United States, género Male
tamaño 11.5, precio $159.20 en la fecha 1/1/2014
Producto ID 2160 del país Canada, género Male
tamaño 9.5, precio $119.20 en la fecha 1/1/2014
Producto ID 2234 del país United States, género Female
tamaño 9.5, precio $159.00 en la fecha 1/1/2014
Producto ID 2222 del país United Kingdom, género Female
tamaño 9, precio $159.00 en la fecha 1/1/2014
Producto ID 2173 del país United States, género Male
tamaño 10.5, precio $159.00 en la fecha 1/1/2014
Producto ID 2200 del país Germany, género Female
tamaño 9, precio $179.00 en la fecha 1/2/2014
Producto ID 2238 del país Canada, género Male
tamaño 10, precio $169.00 en la fecha 1/2/2014
Producto ID 2191 del país United States, género Male
tamaño 10.5, precio $139.00 en la fecha 1/2/2014
Producto ID 2237 del país United Kingdom, género Female
tamaño 9, precio $149.00 en la fecha 1/2/2014
```

**Figura 1** – Primeros 10 datos de la tabla Original – Screen Python

```

Producto ID 2230 del país United States, género Male
tamaño 11.5, precio $159.20 en la fecha 1/1/2014
Producto ID 2160 del país Canada, género Male
tamaño 9.5, precio $119.20 en la fecha 1/1/2014
Producto ID 2234 del país United States, género Female
tamaño 9.5, precio $159.00 en la fecha 1/1/2014
Producto ID 2222 del país United Kingdom, género Female
tamaño 9, precio $159.00 en la fecha 1/1/2014
Producto ID 2173 del país United States, género Male
tamaño 10.5, precio $159.00 en la fecha 1/1/2014
Producto ID 2200 del país Germany, género Female
tamaño 9, precio $179.00 en la fecha 1/2/2014
Producto ID 2238 del país Canada, género Male
tamaño 10, precio $169.00 en la fecha 1/2/2014
Producto ID 2191 del país United States, género Male
tamaño 10.5, precio $139.00 en la fecha 1/2/2014
Producto ID 2237 del país United Kingdom, género Female
tamaño 9, precio $149.00 en la fecha 1/2/2014
Producto ID 2197 del país United States, género Male
tamaño 10, precio $129.00 en la fecha 1/2/2014

```

**Figura 2** – Primeros 10 datos de la tabla Filtrada – Screen Python

Por ejemplo, al comparar la Figura 1 con la Figura 2, se observa que el producto 2152 fue removido por tener un dato faltante en “gender”.

### **Función países\_unicos()**

Esta función sirve para obtener una lista de todos los países en los que se efectuó al menos una venta.

```

En el siguiente array se muestran los países en los que hubo ventas:
['Canada' 'Germany' 'United Kingdom' 'United States']

```

**Figura 3** – Lista de países

### **Función ventas\_pais()**

La misma sirve para obtener un diccionario de todos los países con la cantidad de ventas en dólares.

```

En el siguiente diccionario se muestra el total de ventas por país
{'Canada': 421860.1, 'Germany': 626249.4, 'United States': 839355.1, 'United Kingdom': 250397.3}

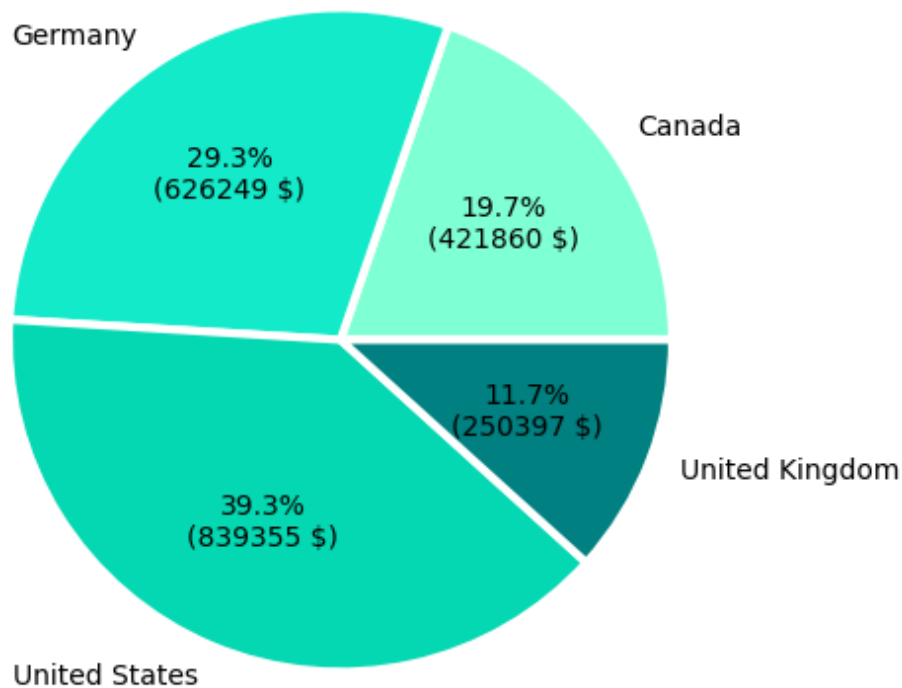
```

**Figura 4** – Diccionario con ventas en dólares por país

Se muestra el resultado obtenido en el siguiente gráfico de tortas.

## Ventas de calzado por país

En Dólares [\$]



### Función calzado\_pais()

Esta función sirve para obtener un diccionario del tamaño de calzado más vendido en cada país.

```
En el siguiente diccionario se muestra el tamaño de calzado más vendido en cada país
(array(['10', '10.5', '11', '11.5', '12', '13', '14', '15', '4.5', '5',
       '5.5', '6', '6.5', '7', '7.5', '8', '8.5', '9', '9.5'],
      dtype='<U32'), array([284, 278, 132, 86, 72, 12, 21, 27, 6, 6, 6, 36, 66,
       116, 196, 239, 357, 530, 455], dtype=int64))
```

**Figura 4** – Lista de arrays con el N° de ventas por tamaño de calzado en Canadá

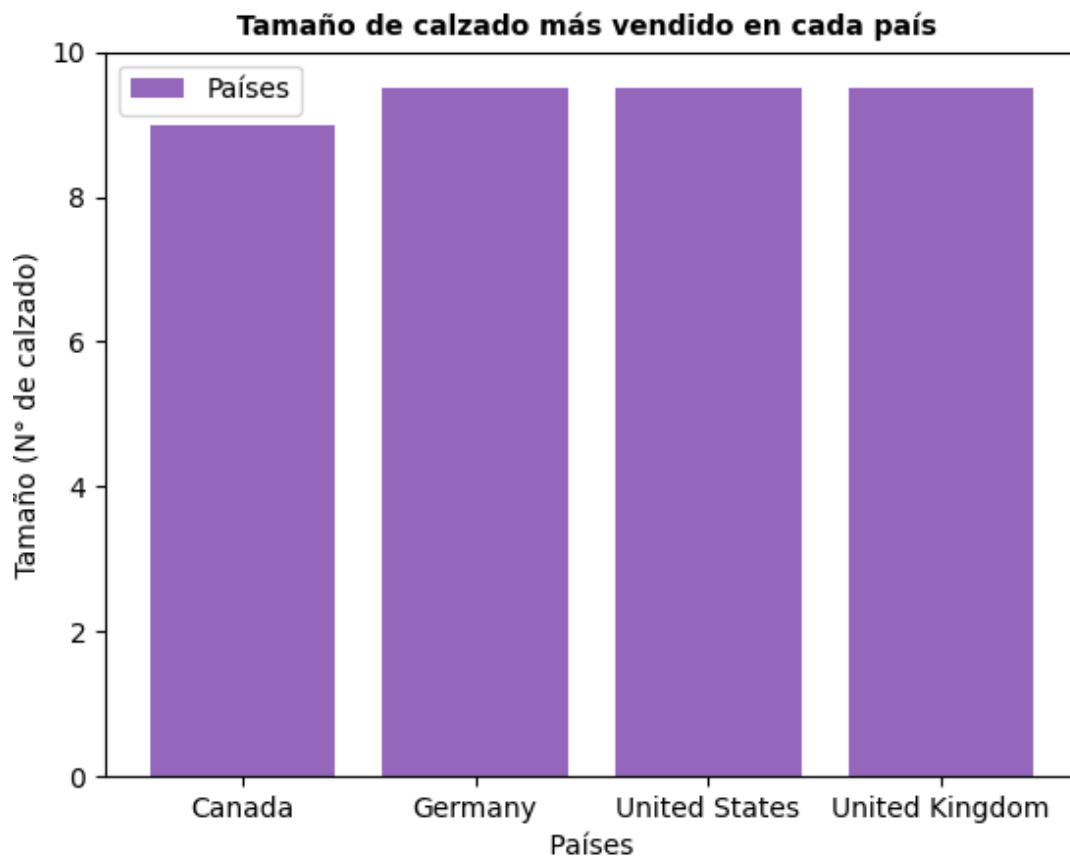
Se obtienen la cantidad de ventas según el tamaño de calzado para cada país, en este ejemplo se utilizó Canadá. Efectivamente se obtiene que tamaño más comercializado es el calzado N°9, con 530 ventas realizadas.

Luego en la figura 5 se muestran el resultado para cada país:

```
En el siguiente diccionario se muestra el tamaño de calzado más vendido en cada país
{'Canada': '9', 'Germany': '9.5', 'United States': '9.5', 'United Kingdom': '9.5'}
```

**Figura 5** – Diccionario con el N° de calzado más vendido por país

Se muestra el resultado obtenido en el siguiente gráfico de barras.



### **Función `ventas_genero_pais()`**

En esta función el usuario puede elegir un género target y de acuerdo al mismo se muestran las ventas de calzado por país. A continuación, se pueden ver dos ejemplos, para *Female* y para *Male*.

```
En el siguiente diccionario se muestran las ventas para el género Female en cada país  
{'Canada': 1108, 'Germany': 1700, 'United States': 2444, 'United Kingdom': 666}
```

**Figura 6** – Diccionario con las ventas de calzado para mujeres en cada país

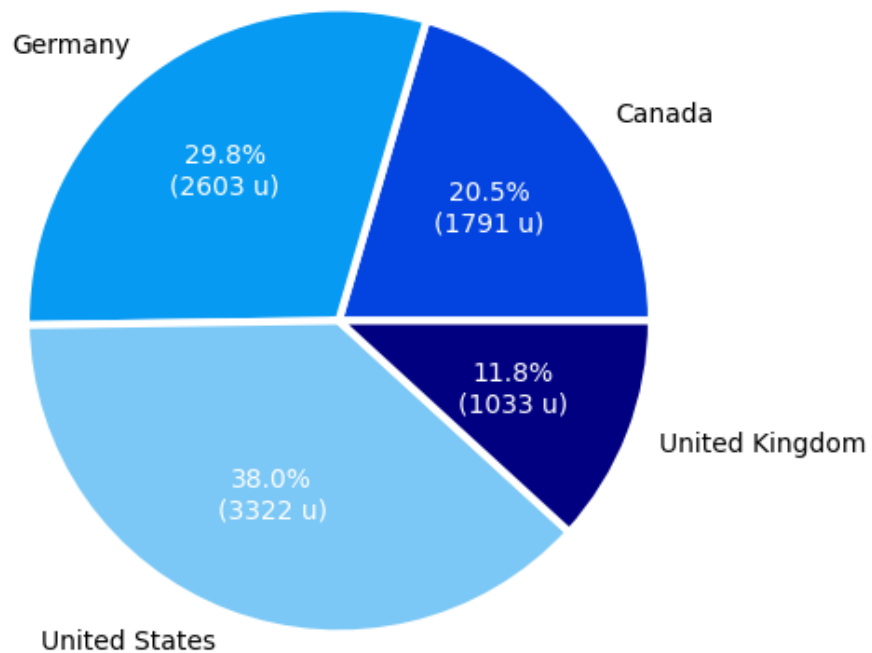
```
En el siguiente diccionario se muestran las ventas para el género Male en cada país:  
{'Canada': 1791, 'Germany': 2603, 'United States': 3322, 'United Kingdom': 1033}
```

**Figura 7** – Diccionario con las ventas de calzado para hombres en cada país

Se muestra el resultado obtenido en el siguiente gráfico de tortas.

## Ventas de calzado para el genero Male por país

En pares [u]



### **Bonus Track: Gráficos**

Se elaboraron distintos gráficos para cada función utilizando la librería *Matplotlib*. Los mismos se encuentran con su función correspondiente dentro de cada apartado.

## Anexo: Código

Se adjunta el código utilizado para el llevado a cabo del proyecto correspondiente al archivo main.py y graficos.py

```
#####

# Proyecto Final - Inove Escuela de Código
# Tema Elegido: Python Analytics

# Alumno: Valentín Imperio.
# Versión: 1.0
# Profesores: Johana Rangel y Julián Salinas.
# Curso: Programador Python.
# Fecha de Entrega: 28/01/2023

# ARCHIVO PRINCIPAL: MAIN.PY

#####

# DESARROLLO DEL PROYECTO #

#### Librerías utilizadas ####

import numpy as np
import re # Para convertir a float y remover $
import sqlalchemy
from sqlalchemy import Column, Integer, String # Herramientas para crear
la tabla.
from sqlalchemy.ext.declarative import declarative_base # Molde para
tablas de bases de datos.
from sqlalchemy.orm import sessionmaker # Sirve para poder iniciar la
sesión

#### Creación del motor (engine) de la base de datos

engine = sqlalchemy.create_engine("sqlite:///ventas_calzados.db")
base = declarative_base()

##### Creación de la Clase Ventas #####
# Para armar la estructura de la tabla dada (visualizador de bases de
datos en
# https://extendsclass.com/sqlite-browser.html) con el ORM de SQLAlchemy
se
# genera la siguiente clase.

class Ventas(base):
```

```

__tablename__ = "venta"
id = Column(Integer , primary_key=True)
date = Column(String,nullable=True)
product_id = Column(Integer,nullable=True)
country = Column(String,nullable=True)
gender = Column(String,nullable=True)
size = Column(String,nullable=True)
price = Column(String,nullable=True)

def __repr__(self): # Método informativo
    return f"Producto ID {self.product_id} del país {self.country},
género {self.gender} \
    \n tamaño {self.size}, precio {self.price} en la fecha
{self.date}"

##### Funciones #####

# Función READ_DB: Para leer la base de datos ventas_calzado.db y
organizar los datos.
def read_db(limit=10):

    # Crear la sesión
    Sesion = sessionmaker(bind=engine)
    sesion = Sesion()

    # Buscar todas las ventas
    query = sesion.query(Ventas)

    # Aplica el limite si está definido
    if limit > 0:
        query1 = query.limit(limit)

    # Leer una venta a la vez e imprime en pantalla
    print("Estos son los primeros 10 datos de la tabla ORIGINAL")
    for venta in query1:
        print(venta)

    nro_filas=query.count()
    print("La tabla ORIGINAL tiene",nro_filas,"filas.") #14967

    # Se eliminan las filas que tengan un valor nulo
    print("\n")
    print("Removiendo filas con valores vacíos...")
    print("\n")

    # Estas son las filas con valores nulos.
    query2 = sesion.query(Ventas).filter(
        (Ventas.date == "") |
        (Ventas.product_id == "") |

```



```

        (Ventas.country == "") |
        (Ventas.gender == "") |
        (Ventas.size == "") |
        (Ventas.price == "")
    )
    nro_filas2 = query2.count()
    print("Se eliminaran",nro_filas2,"filas de la tabla por datos
faltantes") #120
    print("\n")

    # Se procede a borrar las filas mencionadas.
    sesion.query(Ventas).filter(
        (Ventas.date == "") |
        (Ventas.product_id == "") |
        (Ventas.country == "") |
        (Ventas.gender == "") |
        (Ventas.size == "") |
        (Ventas.price == "")
    ).delete()

    sesion.commit() # Para guardar los cambios.

    # Se vuelve a hacer la consulta para ver la tabla filtrada
    query = sesion.query(Ventas)

    # Aplica el limite si está definido
    if limit > 0:
        query1 = query.limit(limit)

    print("Estos son los primeros 10 datos de la tabla FILTRADA")
    for venta in query1:
        print(venta)
    nro_filas3 = query.count()
    print("Quedan",nro_filas3,"filas en la tabla.") #14847

    query3 = sesion.query(Ventas)
    country = np.array([])
    gender = np.array([])
    size = np.array([])
    price = np.array([])

    for i in query3:
        country = np.append(country, i.country)
        gender = np.append(gender,i.gender)
        size = np.append(size,i.size)
        price_str = " ".join(i.price)
        price_i = float(re.sub(r'^\d\.-.', '', price_str)) # Sacar $ y
pasar a float
        price = np.append(price,price_i)

```

```

    return country, gender, size, price

# Función PAÍSES ÚNICOS: Para ver en qué países hubo ventas:
def paises_unicos(country):

    paises_unicos = np.unique(country)
    print(paises_unicos)

def ventas_pais(countries, country, price):

    # Crear unas máscaras para obtener los índices del array
    # que corresponden a los países
    mask_0 = country == countries[0] #"Canada"
    mask_1 = country == countries[1] #"Germany"
    mask_2 = country == countries[2] #"United States"
    mask_3 = country == countries[3] #"United Kingdom"

    # Utilizo las máscaras para acceder a los índices del array price
    # que corresponden a cada país
    ventas_canada = price[mask_0]
    ventas_germany = price[mask_1]
    ventas_united_states = price[mask_2]
    ventas_united_kingdom = price[mask_3]

    suma_canada = np.sum(ventas_canada)
    suma_germany = np.sum(ventas_germany)
    suma_united_states = np.sum(ventas_united_states)
    suma_united_kingdom = np.sum(ventas_united_kingdom)

    Countries_dict = {}
    Countries_dict.update({ countries[0]: suma_canada, countries[1]:
suma_germany, countries[2]: suma_united_states, countries[3]:
suma_united_kingdom})

    return(Countries_dict)

# Función CALZADO PAÍS: Para ver qué tamaño fue el más vendido en cada
país.
def calzado_pais(countries, country, size):

    # Crear unas máscaras para obtener los índices del array
    # que corresponden a los países
    mask_0 = country == countries[0] #"Canada"
    mask_1 = country == countries[1] #"Germany"
    mask_2 = country == countries[2] #"United States"
    mask_3 = country == countries[3] #"United Kingdom"

    # Utilizo las máscaras para acceder a los índices del array size

```

```

# que corresponden a cada pais
tamaño_canada = size[mask_0]
tamaño_germany = size[mask_1]
tamaño_united_states = size[mask_2]
tamaño_united_kingdom = size[mask_3]

sizes_canada = np.unique(tamaño_canada, return_counts=True)
sizes_germany = np.unique(tamaño_germany, return_counts=True)
sizes_united_states =
np.unique(tamaño_united_states, return_counts=True)
sizes_united_kingdom =
np.unique(tamaño_united_kingdom, return_counts=True)

mask_mas_vendido_C = sizes_canada[1] == max(sizes_canada[1])
tamaño_mas_vendido_C = sizes_canada[0][mask_mas_vendido_C]
tamaño_mas_vendido_C = " ".join(tamaño_mas_vendido_C.tolist()) # Con
tolist() se remuev el dtype y con el join se pasa de lista a str
mask_mas_vendido_G = sizes_germany[1] == max(sizes_germany[1])
tamaño_mas_vendido_G = sizes_germany[0][mask_mas_vendido_G]
tamaño_mas_vendido_G = " ".join(tamaño_mas_vendido_G.tolist())
mask_mas_vendido_USA = sizes_united_states[1] ==
max(sizes_united_states[1])
tamaño_mas_vendido_USA = sizes_united_states[0][mask_mas_vendido_USA]
tamaño_mas_vendido_USA = " ".join(tamaño_mas_vendido_USA.tolist())
mask_mas_vendido_UK = sizes_united_kingdom[1] ==
max(sizes_united_kingdom[1])
tamaño_mas_vendido_UK = sizes_united_kingdom[0][mask_mas_vendido_UK]
tamaño_mas_vendido_UK = " ".join(tamaño_mas_vendido_UK.tolist())

Sizes_dict = {}
Sizes_dict.update({ countries[0]: tamaño_mas_vendido_C, countries[1]:
tamaño_mas_vendido_G, countries[2]: tamaño_mas_vendido_USA, countries[3]:
tamaño_mas_vendido_UK})

return(Sizes_dict)

# Función VENTAS GENERO PAÍS: Para ver qué genero compro la mayor
cantidad de calzado en cada país.
def ventas_genero_pais(countries, gender_target, country, gender):

    generos_unicos = np.unique(gender) # "Female", "Male" y "Unix"

    # Crear unas máscaras para obtener los índices del array
    # que corresponden a los paises
    mask_0 = country == countries[0] #"Canada"
    mask_1 = country == countries[1] #"Germany"
    mask_2 = country == countries[2] #"United States"
    mask_3 = country == countries[3] #"United Kingdom"

```

```

# Utilizo las máscaras para acceder a los índices del array size
# que corresponden a cada país
genero_canada = gender[mask_0]
genero_germany = gender[mask_1]
genero_united_states = gender[mask_2]
genero_united_kingdom = gender[mask_3]

genero_canada = np.unique(genero_canada, return_counts=True)
genero_germany = np.unique(genero_germany, return_counts=True)
genero_united_states =
np.unique(genero_united_states, return_counts=True)
genero_united_kingdom =
np.unique(genero_united_kingdom, return_counts=True)

mask_target_C = genero_canada[0] == gender_target
genero_target_C = genero_canada[1][mask_target_C]
genero_target_C = genero_target_C.tolist()[0] # Con tolist() se
remuev el dtype y con el [0] se obtiene el unico elemento de la lista.
mask_target_G = genero_germany[0] == gender_target
genero_target_G = genero_germany[1][mask_target_G]
genero_target_G = genero_target_G.tolist()[0]
mask_target_USA = genero_united_states[0] == gender_target
genero_target_USA = genero_united_states[1][mask_target_USA]
genero_target_USA = genero_target_USA.tolist()[0]
mask_target_UK = genero_united_kingdom[0] == gender_target
genero_target_UK = genero_united_kingdom[1][mask_target_UK]
genero_target_UK = genero_target_UK.tolist()[0]

Target_Genre_dict = {}
Target_Genre_dict.update({ countries[0]: genero_target_C,
countries[1]: genero_target_G, countries[2]: genero_target_USA,
countries[3]: genero_target_UK})

return(Target_Genre_dict)

##### Bloque principal del programa #####

if __name__ == "__main__":

    data = read_db()
    country = data[0]
    gender = data[1]
    size = data[2]
    price = data[3]

```

```

    countries = ["Canada","Germany","United States", "United Kingdom"]
    gender_target = "Male" # Las opciones a analizar son: "Female",
"Male" y "Unix"
    print("En el siguiente array se muestran los países en los que hubo
ventas:")
    paises_unicos(country)

    print("En el siguiente diccionario se muestra el total de ventas por
país:")
    print(ventas_pais(countries, country, price))

    print("En el siguiente diccionario se muestra el tamaño de calzado
más vendido en cada país:")
    print(calzado_pais(countries, country, size))

    print("En el siguiente diccionario se muestran las ventas para el
género "+ gender_target+ " en cada país:")
    print(ventas_genero_pais(countries, gender_target, country, gender))

    print("Termine! Que buen programa soy ^^")

```

```

#####

# Proyecto Final - Inove Escuela de Código
# Tema Elegido: Python Analytics

# Alumno: Valentín Imperio.
# Versión: 1.0
# Profesores: Johana Rangel y Julián Salinas.
# Curso: Programador Python.
# Fecha de Entrega: 28/01/2023

# ARCHIVO BONUS: GRAFICOS.PY

#####

#### Librerías utilizadas ####

import matplotlib.pyplot as plt
import numpy as np
from main import read_db, ventas_pais, calzado_pais, ventas_genero_pais

# GRAFICOS DEL PROYECTO #

# Gráfico para Ventas por país

```

```

def grafico_ventas(countries, country, price):
    dict = ventas_pais(countries, country, price)

    def func(pct, allvals):
        absolute = int(np.round(pct/100.*np.sum(allvals)))
        return "{:.1f}%\n({:d} $)".format(pct, absolute)

    X = list(dict.values())
    Y = dict.keys()

    # Creo mi paleta de colores:
    colors = ["#7FFFD4", "#13EAC9", "#04D8B2", "#008080"]
    fig = plt.figure()
    fig.suptitle("Ventas de calzado por país", fontsize = 16, weight =
"bold")
    ax = fig.add_subplot()
    ax.pie(X, labels=Y, colors = colors, autopct=lambda pct: func(pct,
X), wedgeprops={'linewidth': 3.0, 'edgecolor': 'white'})
    ax.set_title("En Dólares [$]", fontsize = 12)
    ax.axis("equal")
    plt.show()

# Gráfico para N° de calzado por país

def grafico_calzado(countries, country, size):
    dict = calzado_pais(countries, country, size)

    X = list(dict.keys())
    Y = [float(x) for x in list(dict.values())]

    fig = plt.figure()
    ax = fig.add_subplot()
    ax.bar(X,Y, color="tab:purple", label="Países")
    ax.set_title("Tamaño de calzado más vendido en cada país ", fontsize
= 10, weight = "bold")
    ax.set_ylabel(" Tamaño (N° de calzado) ")
    ax.set_xlabel(" Países ")
    ax.legend()
    plt.xticks(X)
    plt.ylim(top=10)
    plt.show()

# Gráfico para cantidad de ventas por genero elegido para cada país

def grafico_genero_pais(countries, gender_target, country, gender):

    dict = ventas_genero_pais(countries, gender_target, country, gender)

    def func(pct, allvals):

```

```

        absolute = int(np.round(pct/100.*np.sum(allvals)))
        return "{:.1f}%\n({:d} u)".format(pct, absolute)

X = list(dict.values())
Y = dict.keys()

# Creo mi paleta de colores:
colors = ["#0343DF", "#069AF3", "#7BC8F6", "#000080"]
fig = plt.figure()
fig.suptitle("Ventas de calzado para el genero "+gender_target+" por país", fontsize = 16, weight = "bold")
ax = fig.add_subplot()
_, _, autotexts = ax.pie(X, labels=Y, colors = colors, autopct=lambda pct: func(pct, X), wedgeprops={'linewidth': 3.0, 'edgecolor': 'white'})
for ins in autotexts:
    ins.set_color('white')
ax.set_title("En pares [u]", fontsize = 12)
ax.axis("equal")
plt.show()

##### Bloque principal del programa #####

if __name__ == "__main__":

    data = read_db()
    country = data[0]
    gender = data[1]
    size = data[2]
    price = data[3]
    countries = ["Canada", "Germany", "United States", "United Kingdom"]
    gender_target = "Male" # Las opciones a analizar son: "Female", "Male" y "Unix"

    grafico_ventas(countries, country, price)

    grafico_calzado(countries, country, size)

    grafico_genero_pais(countries, gender_target, country, gender)

    print("Termine! Que buen programa soy ^^")

```