Proyecto de Github

El objetivo del presente informe es demostrar mi conocimiento en las librerias Numpy, Pandas, Matplotlib y Seaborn del lenguaje de programación Python. Para el analisis se usará una base de datos ficticia.

En primer lugar importamos las librerias a usar en el código.

In [1]:
```
import numpy as np
```

In [2]:
```
import pandas as pd
```

In [3]:
```
import matplotlib.pyplot as plt
```

In [4]:
```
import seaborn as sb
```

En segundo lugar, configuramos con Seaborn el aspecto de los gráficos del archivo actual.

In [5]:
```
sb.set()
tamano_letra_ejes=16
tamano_letra_titulo=18
tamano_valores_eje_x=18
tamano_valores_eje_y=18
tamano_horizontal_grafico=10
tamano_vertical_grafico=6
sb.set_context("notebook",
rc={"font.size":18,
    "axes.titlesize":tamano_letra_ejes,
    "axes.titlesize":tamano_letra_titulo,
    "xtick.labelsize":tamano_valores_eje_x,
    "ytick.labelsize":tamano_valores_eje_y,
 }
 )
plt.rcParams['figure.figsize'] = (tamano_horizontal_grafico, tamano_vertical_grafico)
```

Luego definimos al data frame "datos_falsos" que va a tener toda la información. Este información esta contenida en el archivo llamado Datos Falsos.csv. El archivo debe estar en la misma carpeta que este jupyter notebook para que funcione.

In [6]:
```
datos_falsos = pd.read_csv('Datos Falsos.csv')
```

In [7]:
```
datos_falsos
```

Out[7]:

	id	Nombre	Apellido	email	Genero	Salario	Raza
0	1	Fredi	Ochiltree	fochiltree0@digg.com	Female	973729.0	Latin American Indian
1	2	Anneliese	Southey	asouthey1@prweb.com	Female	827507.0	White
2	3	Mauricio	Marchant	mmarchant2@tinypol.com	Male	814941.0	Delaware
3	4	Meara	Caitlin	mcaitlin3@tinypic.com	Female	975119.0	Aleut
4	5	Jory	Lacoste	jlacoste4@comcast.net	Male	680650.0	Colville
...	...	...	...	...	...	...	...
995	996	Darsie	Levane	dlevanern@sohu.com	Female	836998.0	Hmong
996	997	Camala	Walhall	cwalhallo@pen.io	Female	480701.0	Fijian
997	998	Jessie	Bolf	jbolfp@adobe.com	Genderfluid	727565.0	Asian
998	999	Maureen	Grieger	mgriegerq@google.com.br	Female	851410.0	Korean
999	1000	Dinnie	Poat	dpoatr@hc360.com	Female	NaN	Cuban

1000 rows × 7 columns

Antes de partir cualquier analisis,quiero definir a la columna "id" como mi columna indice, para eso usamos la siguiente instrucción:

In [8]:
```
datos_falsos = datos_falsos.set_index('id')
```

In [9]:
```
datos_falsos
```

Out[9]:

id	Nombre	Apellido	email	Genero	Salario	Raza
1	Fredi	Ochiltree	fochiltree0@digg.com	Female	973729.0	Latin American Indian
2	Anneliese	Southey	asouthey1@prweb.com	Female	827507.0	White
3	Mauricio	Marchant	mmarchant2@tinypol.com	Male	814941.0	Delaware
4	Meara	Caitlin	mcaitlin3@tinypic.com	Female	975119.0	Aleut
5	Jory	Lacoste	jlacoste4@comcast.net	Male	680650.0	Colville
...	...	...	...	...	...	...
996	Darsie	Levane	dlevanern@sohu.com	Female	836998.0	Hmong
997	Camala	Walhall	cwalhallo@pen.io	Female	480701.0	Fijian
998	Jessie	Bolf	jbolfp@adobe.com	Genderfluid	727565.0	Asian
999	Maureen	Grieger	mgriegerq@google.com.br	Female	851410.0	Korean
1000	Dinnie	Poat	dpoatr@hc360.com	Female	NaN	Cuban

1000 rows × 6 columns

Ahora vamos a revisar si existen filas con datos incompletos:

In [10]:
```
filas_nan  = datos_falsos[datos_falsos.isna().any(axis=1)]
```

In [11]:
```
filas_nan
```

Out[11]:

id	Nombre	Apellido	email	Genero	Salario	Raza
7	Derrna	Shaw	dshaw6@ed.gov	Female	952131.0	NaN
24	Nikolai	Bingle	nbingler@mediafire.com	Male	718864.0	NaN
29	Christie	Luttger	cluttgers@studiopress.com	Male	NaN	Laotian
30	Lukas	Bonhome	lbonhomet@telegraph.co.uk	Male	NaN	Taiwanese
37	Hewe	Aynsley	haynsley10@networksolutions.com	Male	NaN	Salvadoran
...	...	...	...	...	...	...
971	Ebenezer	Lavington	elavingtonqy@elegantthemes.com	Male	629346.0	NaN
973	Everred	Minnis	eminnisr0@sina.com.cn	Male	922488.0	NaN
986	Nikita	Lemmertz	nlemmertzn@washingtonpost.com	Male	NaN	Shoshone
992	Kenon	Barnshaw	kbarnshawe@historyhorn.com	Male	NaN	Chickasaw
1000	Dinnie	Poat	dpoatr@hc360.com	Female	NaN	Cuban

110 rows × 6 columns

Se observa que hay 110 filas, con la siguiente instrucción eliminamos estas filas del dataframe:

In [12]:
```
datos_falsos.dropna(inplace=True)
```

In [13]:
```
datos_falsos
```

Out[13]:

id	Nombre	Apellido	email	Genero	Salario	Raza
1	Fredi	Ochiltree	fochiltree0@digg.com	Female	973729.0	Latin American Indian
2	Anneliese	Southey	asouthey1@prweb.com	Female	827507.0	White
3	Mauricio	Marchant	mmarchant2@tinypol.com	Male	814941.0	Delaware
4	Meara	Caitlin	mcaitlin3@tinypic.com	Female	975119.0	Aleut
5	Jory	Lacoste	jlacoste4@comcast.net	Male	680650.0	Colville
...	...	...	...	...	...	...
995	Pat	Langhor	planghorm9@jigsy.com	Female	913469.0	Dominican (Dominican Republic)
996	Darsie	Levane	dlevanern@sohu.com	Female	836998.0	Hmong
997	Camala	Walhall	cwalhallo@pen.io	Female	480701.0	Fijian
998	Jessie	Bolf	jbolfp@adobe.com	Genderfluid	727565.0	Asian
999	Maureen	Grieger	mgriegerq@google.com.br	Female	851410.0	Korean

890 rows × 6 columns

Ahora que no existen datos nulos, se puede analizar la base de datos:

Histograma

Vamos a hacer un histograma que nos muestre la distribución de los salarios.

Pero antes de hacer el histograma vamos a ver cual es el valor mínimo y el valor máximo

In [14]:
```
datos_falsos.Salario.min()
```

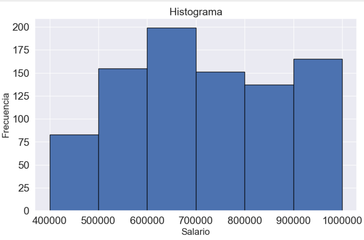
Out[14]:
```
451173.0
```

In [15]:
```
datos_falsos.Salario.max()
```

Out[15]:
```
999973.0
```

Se crea un histograma con valores entre 400.000 y 1.000.000.

In [16]:
```
bins=[400000,500000,600000, 700000, 800000,900000,1000000]
fig = plt.figure(figsize=(10, 6)) #Esta instrucción cambia el tamaño de la figura
plt.hist(datos_falsos.Salario,bins=bins, edgecolor='black') #Esta instrucción crea el histograma
plt.ticklabel_format(style='plain', axis='x')# Esta instrucción deja los valores del eje x como formato número.
plt.xlabel('Salario') #Esto indica que el titulo del eje x es Salario
plt.ylabel('Frecuencia')#Esto indica que el titulo del eje y es Frecuencia
plt.title('Histograma')#Esto indica que el titulo de la figura es Histograma
plt.show()
```



Se observa que la mayoría de los individuos tienen un salario entre 600.000 y 700.000.

Gráfico circular

Ahora vamos a hacer un analisis del salario según el genero del individuo. Quiero crear un gráfico circular que me muestre el porcentaje de personas con genero masculino, femenino y de otros generos.

Con la siguiente instrucción, se clasifican todos los individuos de otros generos como 'Other Gender'

In [17]:
```
serie_generos = datos_falsos['Genero'].apply(lambda x: 'Other Gender' if x not in ['Male', 'Female'] else x)
```

Luego vemos la cantidad de individuos hay para cada genero:

In [18]:
```
contador_masculino=0
contador_femenino=0
contador_otros=0
for genero in serie_generos:
        if genero=='Male':
            contador_masculino=contador_masculino+1
        if genero=='Female':
            contador_femenino=contador_femenino+1
        if genero=='Other Gender':

            contador_otros=contador_otros+1

print(f"Hay {contador_masculino} varones")
print(f"Hay {contador_femenino} damas")
print(f"Hay {contador_otros} personas que se identifican con otro genero")
```

```
Hay 383 varones
Hay 415 damas
Hay 92 personas que se identifican con otro genero
```

Se define una lista de generos y una lista de contadores, las cuales se usaran como argumentos para crear el gráfico circular.

In [19]:
```
lista_generos=['Masculino','Femenino','Otro género']
lista_contadores=[contador_masculino,contador_femenino,contador_otros]
```

In [20]:
```
plt.pie(lista_contadores, labels=lista_generos, autopct='%1.2f%%', startangle=0)#Se usa como valores lista_contadore
s,
# como etiqueta lista_generos, el resto de los argumentos sirve para configurar el aspecto del gráfico
plt.title('Distribución de género')# Esto indica el titulo.
plt.show()
```

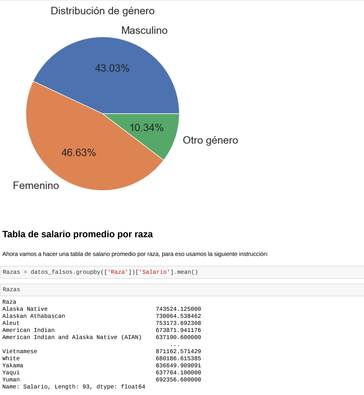


Tabla de salario promedio por raza

Ahora vamos a hacer una tabla de salario promedio por raza, para eso usamos la siguiente instrucción:

In [21]:
```
Razas = datos_falsos.groupby(['Raza'])['Salario'].mean()
```

In [22]:
```
Razas
```

Out[22]:
```
Raza
Alaska Native                          743524.125000
Alaskan Athabascan                     730064.538462
Aleut                                  753173.692308
American Indian                        673871.941176
American Indian and Alaska Native (AIAN)  637190.600000
                                           ...
Vietnamese                             871162.571429
White                                  680186.615385
Yakama                                 836649.909091
Yaqui                                  637764.100000
Yuman                                  692356.600000
Name: Salario, Length: 93, dtype: float64
```

Se observa que hay muchas razas en el análisis, vamos a ordenarlas de mayor a menor y vamos a ver las 10 razas con mayor ingreso promedio

In [23]:
```
Razas.sort_values(ascending=False).head(10)
```

Out[23]:
```
Raza
Vietnamese                      871162.571429
Potawatomi                      854158.222222
Osage                           843981.900000
Venezuelan                      837659.750000
Yakama                          836649.909091
Creek                           833273.800000
Asian                           830230.166667
Bangladeshi                     808420.750000
Houma                           807762.666667
Black or African American       806661.750000
Name: Salario, dtype: float64
```

Se observa que la raza de mayor ingreso es la Vietnamita seguida por la Potawomi.

In [ ]:
```

```