Fuentes de datos

Los datos pueden tener diversas fuentes a las que se puede acceder directa o indirectamente. La calidad, pertinencia y veracidad de un dato depende en gran medida de su origen y el modo en el que éste se obtiene. Una de las bases del método científico es la de poder identificar, aislar y de ser posible, reproducir la fuente de los datos que son objetos de un estudio

Datos continuos y Datos discretos.

Los datos cuantitativos se dividen en datos continuos que puede admitir cualquier valor intermedio dentro de un intervalo de los números reales. Por ejemplo el rendimiento académico, la estatura, el peso, salario, tiempo, volumen que pueden ser obtenidos por medición. Los datos discretos son aquella que admite interrupciones verdaderas en su medición y por lo tanto no admite valores intermedios. Por ejemplo, número de alumnos de la clase, el número de hijos de una familia, el número de clientes en espera en un almacén.

Observación estadística de los datos

Por ahora tenemos en forma natural los datos obtenidos del registro de pedidos e incluso la region constituyen una fuente de mucha información, los registros en forma de tabla tienen los siguientes metadatos:

- Fecha de orden
- Region
- Rep
- Item
- Units
- Unit Cost
- Total

La Distribución de frecuencias separa los datos en clases y muestra el número de ocurrencias en cada clase, o frecuencia de clase, este tipo de agrupación facilita la interpretación y la presentación en cuadros numéricos, que luego se representan en gráficos.

Histogramas

En Python, podemos graficar fácilmente un histograma con la ayuda de la función hist de matplotlib, simplemente debemos pasarle los datos y la cantidad de contenedores en los que queremos dividirlos. Aqui vamos a construir un Histograma de frecuencias:

- importamos pandas
- matplotlib
- numpy
- Creamos un DataFrame con Pandas
- Presentamos los datos

1. La biblioteca Matplotlib.

Matlplotib es la biblioteca de graficación basada en Python más popular y sobre la que una gran catidad de proyectos se basan para despliegue de gráficos y visualización de datos.

Ejercicio

Lo primero que se realiza es importar la libreria de pandas, matplotlib y numpy los cuales seran de mucha utilidad para hacer graficos detallados

Luego lo que se realiza es utilizar una variable para poder incluir el archivo HTML que es con el cual se ha estado trabajando, despues, se le añade un header para poder utilizar el encabezado de la tabla en ves de enumerarlas

In [2]:
```
# importamos la libreria Pandas, matplotlib y numpy que van a ser de mucha utilid
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# leemos los datos de la tabla del directorio Data de trabajo
url = 'file:///C:/Users/Villamar/Desktop/JOSS/SEXTOSEMESTRE/MODELAMIENTOSIMULACIO
dfs = pd.read_html(url, header=0)
df = dfs[0]
#Presentamos los datos en un DataFrame de Pandas
df
```

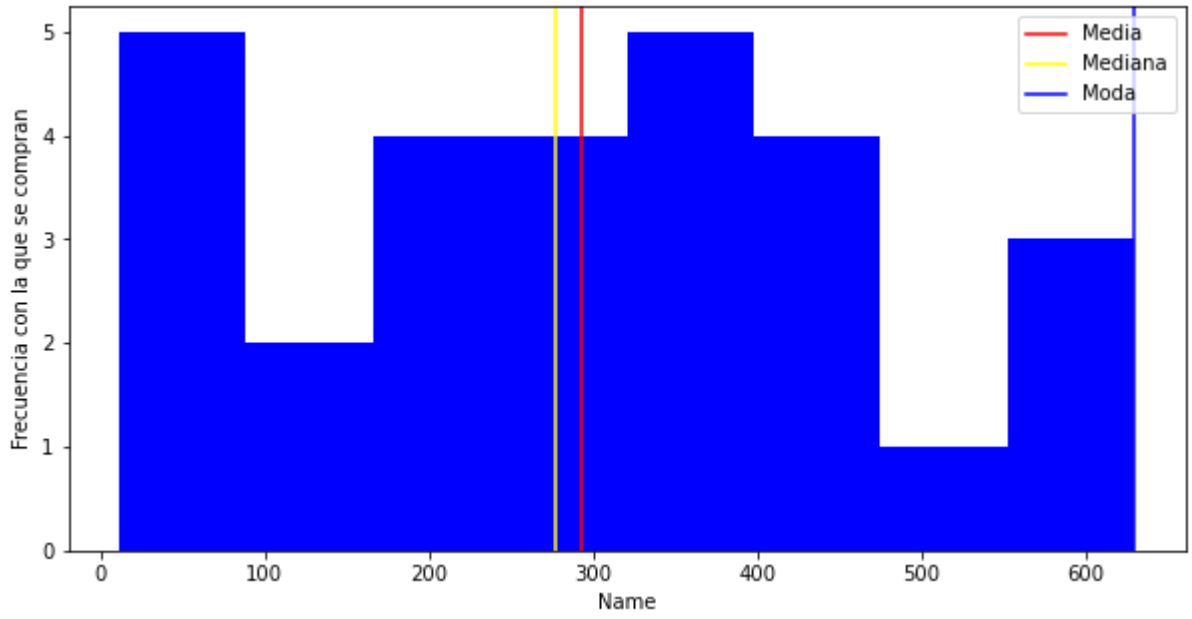
Out[2]:

	Name	Pos.	GP	GS	Min	G	A	Y	R	Caps	Goals
0	Campbell, Jane	GK	2.0	2.0	180.0	0	0.0	0.0	NaN	5.0	NaN
1	Cook, Alana	D	1.0	1.0	90.0	0	0.0	0.0	NaN	2.0	0.0
2	Dahlkemper, Abby	D	4.0	4.0	360.0	0	0.0	0.0	NaN	66.0	0.0
3	Davidson, Tierna	D	3.0	3.0	270.0	0	0.0	0.0	NaN	29.0	1.0
4	Dunn, Crystal	D	6.0	5.0	449.0	0	1.0	0.0	NaN	111.0	24.0
5	Ertz, Julie	M	7.0	7.0	602.0	0	0.0	0.0	NaN	110.0	20.0
6	Fox, Emily	D	1.0	0.0	11.0	0	0.0	0.0	NaN	4.0	0.0
7	Horan, Lindsey	M	7.0	4.0	424.0	1	4.0	1.0	NaN	93.0	20.0
8	Howell, Jaelin	M	1.0	0.0	28.0	0	0.0	0.0	NaN	2.0	0.0
9	Krieger, Ali	D	1.0	1.0	79.0	0	1.0	1.0	NaN	108.0	1.0
10	Krueger, Casey	D	2.0	1.0	54.0	0	1.0	0.0	NaN	34.0	0.0
11	Lavelle, Rose	M	7.0	5.0	398.0	1	1.0	0.0	NaN	53.0	14.0
12	Lloyd, Carli	F	7.0	4.0	325.0	1	4.0	1.0	NaN	301.0	124.0
13	Macario, Catarina	M	3.0	2.0	172.0	1	0.0	0.0	NaN	3.0	1.0
14	Mewis, Kristie	M	6.0	1.0	160.0	2	1.0	0.0	NaN	22.0	4.0
15	Mewis, Samantha	M	4.0	3.0	246.0	3	1.0	0.0	NaN	72.0	21.0
16	Morgan, Alex	F	5.0	2.0	242.0	2	1.0	0.0	NaN	175.0	109.0
17	Naeher, Alyssa	GK	5.0	5.0	450.0	0	0.0	0.0	NaN	69.0	NaN
18	O'Hara, Kelley	D	4.0	4.0	283.0	0	0.0	1.0	NaN	136.0	2.0
19	Press, Christen	F	5.0	4.0	347.0	2	1.0	0.0	NaN	144.0	60.0
20	Purce, Margaret	D	4.0	1.0	167.0	1	0.0	0.0	NaN	6.0	1.0
21	Rapinoe, Megan	F	7.0	5.0	349.0	7	1.0	0.0	NaN	175.0	59.0
22	Sauerbrunn, Becky	D	6.0	6.0	540.0	0	0.0	0.0	NaN	184.0	0.0
23	Smith, Sophia	F	3.0	0.0	68.0	0	0.0	0.0	NaN	4.0	0.0
24	Sonnett, Emily	D	5.0	2.0	266.0	0	1.0	0.0	NaN	51.0	0.0
25	Williams, Lynn	F	5.0	5.0	370.0	1	0.0	0.0	NaN	34.0	10.0
26	Own Goal	NaN	NaN	NaN	NaN	0	NaN	NaN	NaN	NaN	NaN

	Name	Pos.	GP	GS	Min	G	A	Y	R	Caps	Goals
27	TOTAL	NaN	7.0	7.0	630.0	22	18.0	4.0	0.0	NaN	NaN
28	Opponents	NaN	7.0	7.0	630.0	1	1.0	12.0	0.0	NaN	NaN

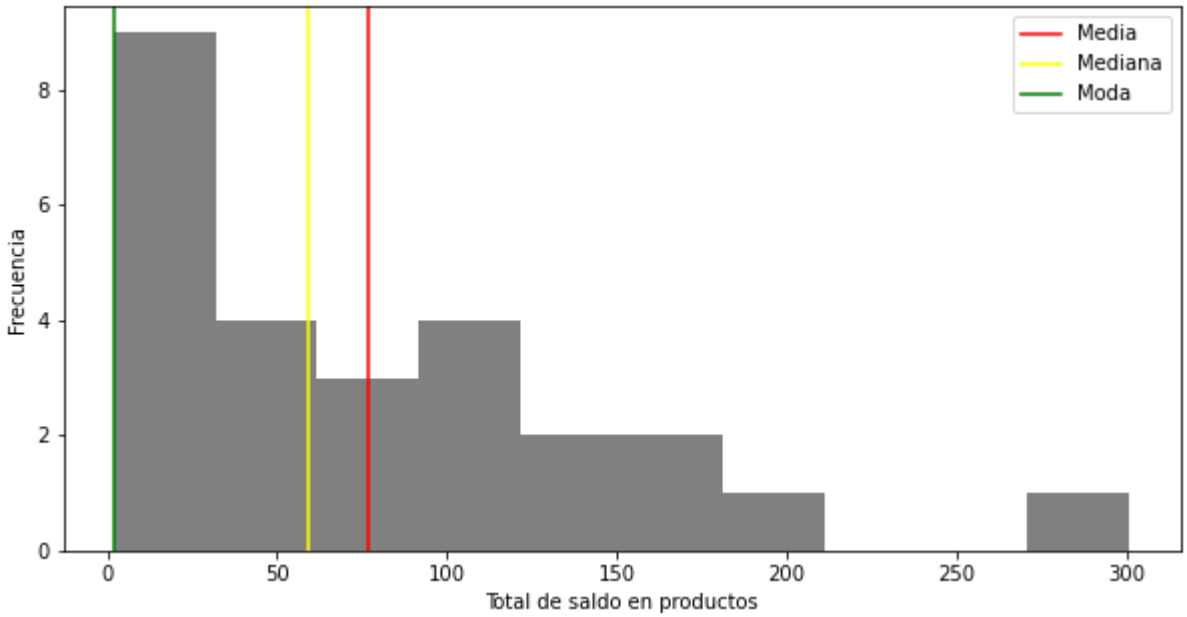
Luego se va a realizar un grafico de las unidades para eso se realiza la siguiente codificacion utilizando los colores, la media, la moda y la mediana

In [3]:
```
x=df["Min"]
plt.figure(figsize=(10,5))
plt.hist(x,bins=8,color='blue')
plt.axvline(x.mean(),color='red',label='Media')
plt.axvline(x.median(),color='yellow',label='Mediana')
plt.axvline(x.mode()[0],color='blue',label='Moda')
plt.xlabel('Name')
plt.ylabel('Frecuencia con la que se compran')
plt.legend()
plt.show()
```



El siguiente grafico va a mostrar el total de los datos parecido al grafico anterior pero para este se utilizo otra de las columnas

In [4]:
```
x=df["Caps"]
plt.figure(figsize=(10,5))
plt.hist(x,bins=None,color='grey')
plt.axvline(x.mean(),color='red',label='Media')
plt.axvline(x.median(),color='yellow',label='Mediana')
plt.axvline(x.mode()[0],color='green',label='Moda')
plt.xlabel('Total de saldo en productos')
plt.ylabel('Frecuencia')
plt.legend()
plt.show()
```



En esta parte se envia la mediana de cada uno de los datos de las figuras anteriores para eso se especfica tres variables de las tres columnas de las cuales se relizaron los graficos

In [5]:
```
# enviando las medias a t1, t2, t3 para su utilización
print("Mediana: " )
t1 = df.median()
t2 = df.median()
t3 = df.median()
print( "la Mediana de GP: ", t1)
print( "la Mediana en Gs: ", t2)
print( "la Mediana entre el total de Caps: ", t3)
print("DIRECTAMENTE DEL DATAFRAME ")
df.median()
```

```
Mediana:
la Mediana de GP:  GP         5.0
GS         3.5
Min      276.5
G          0.0
A          0.5
Y          0.0
R          0.0
Caps      59.5
Goals      1.5
dtype: float64
la Mediana en Gs:  GP         5.0
GS         3.5
Min      276.5
G          0.0
A          0.5
Y          0.0
R          0.0
Caps      59.5
Goals      1.5
dtype: float64
la Mediana entre el total de Caps:  GP         5.0
GS         3.5
Min      276.5
G          0.0
A          0.5
Y          0.0
R          0.0
Caps      59.5
Goals      1.5
dtype: float64
DIRECTAMENTE DEL DATAFRAME
```

Out[5]:
```
GP         5.0
GS         3.5
Min      276.5
G          0.0
A          0.5
Y          0.0
R          0.0
Caps      59.5
Goals      1.5
dtype: float64
```

En esta parte se envia la media de cada uno de los datos de las figuras anteriores para eso se

especifica tres variables de las tres columnas de las cuales se relizaron los graficos

In [6]:
```
# enviando las medias a t1, t2, t3 para su utilización
print("Media:", )
t1 = df.mean()
t2 = df.mean()
t3 = df.mean()

print( "la Media de GP: ", t1)
print( "la Media en GS: ", t2)
print( "la Media entre el Caps: ", t3)
print("DIRECTAMENTE DEL DATAFRAME ")
df.mean()
```

```
Media:
la Media de GP:  GP         4.464286
GS          3.250000
Min       292.500000
G           1.551724
A           1.321429
Y           0.714286
R           0.000000
Caps       76.653846
Goals      19.625000
dtype: float64
la Media en GS:  GP         4.464286
GS          3.250000
Min       292.500000
G           1.551724
A           1.321429
Y           0.714286
R           0.000000
Caps       76.653846
Goals      19.625000
dtype: float64
la Media entre el Caps:  GP         4.464286
GS          3.250000
Min       292.500000
G           1.551724
A           1.321429
Y           0.714286
R           0.000000
Caps       76.653846
Goals      19.625000
dtype: float64
DIRECTAMENTE DEL DATAFRAME
```

Out[6]:
```
GP         4.464286
GS         3.250000
Min      292.500000
G          1.551724
A          1.321429
Y          0.714286
R          0.000000
Caps      76.653846
Goals     19.625000
dtype: float64
```

Despues se realiza la moda de cada uno de los datos de las figuras anteriores para eso se

Despues se realiza la moda de cada uno de los datos de las figuras anteriores para ello se especfica tres variables de las tres columnas de las cuales se relizaron los graficos lueg mostramos ambas modas con su respectivo reusultado

In [7]:
```
# enviando las modas a mo1, mo2, mo3 para su utilización
print("Moda:")
mo1 = df["GP"].mode()
mo2 = df["GS"].mode()
mo3 = df["Caps"].mode()
print( "la Moda del total de GP: ", mo1)
print( "la Moda de las GS: ", mo2)
print( "la Moda del Caps: ", mo3)
pd.DataFrame(mo1)
pd.DataFrame(mo2)
pd.DataFrame(mo3)
```

```
Moda:
la Moda del total de GP:  0    7.0
dtype: float64
la Moda de las GS:  0    1.0
1    4.0
2    5.0
dtype: float64
la Moda del Caps:  0      2.0
1      4.0
2     34.0
3    175.0
dtype: float64
```

Out[7]:

	0
0	2.0
1	4.0
2	34.0
3	175.0

A continuacion, se tomaran los dataos de las columnas, se ha usado Units, Total y UniCost por que son datos numericos esto lo usamos seguido de un describe, lo cual ns decribe directamente la media, desviacion estandae, valor minimo, maximo, cada uno de los cuartiles

In [8]:
```
# Tomamos los datos de las columnas
df[['GP','GS','Caps']].describe()
# describe(), nos presenta directamente la media, desviación standar, el valor mi
#o, valor máximo, el 1er cuartil, 2do Cuartil, 3er Cuartil
```

Out[8]:

	GP	GS	Caps
count	28.000000	28.000000	26.000000
mean	4.464286	3.250000	76.653846
std	2.116639	2.204793	74.929269
min	1.000000	0.000000	2.000000
25%	3.000000	1.000000	10.000000
50%	5.000000	3.500000	59.500000
75%	6.250000	5.000000	110.750000
max	7.000000	7.000000	301.000000

Seguido vamos a seleccionar los datos estadisticos los primeros 30 registros

In [9]:
```
df_1 = df[:30]
print ("Estadisticos de 30 REGISTROS")
print ("-----------------------------")
df_1[['GP','GS','Caps']].describe()
```

```
Estadisticos de 30 REGISTROS
-----------------------------
```

Out[9]:

	GP	GS	Caps
count	28.000000	28.000000	26.000000
mean	4.464286	3.250000	76.653846
std	2.116639	2.204793	74.929269
min	1.000000	0.000000	2.000000
25%	3.000000	1.000000	10.000000
50%	5.000000	3.500000	59.500000
75%	6.250000	5.000000	110.750000
max	7.000000	7.000000	301.000000

Con estas dos lineas de codigo vamos a mostrar un intervalo de 15 numeros los cuales comenzaran del 15 al 30

In [10]:
```
df_2 = df[15:30]
df_2
```

Out[10]:

	Name	Pos.	GP	GS	Min	G	A	Y	R	Caps	Goals
15	Mewis, Samantha	M	4.0	3.0	246.0	3	1.0	0.0	NaN	72.0	21.0
16	Morgan, Alex	F	5.0	2.0	242.0	2	1.0	0.0	NaN	175.0	109.0
17	Naeher, Alyssa	GK	5.0	5.0	450.0	0	0.0	0.0	NaN	69.0	NaN
18	O'Hara, Kelley	D	4.0	4.0	283.0	0	0.0	1.0	NaN	136.0	2.0
19	Press, Christen	F	5.0	4.0	347.0	2	1.0	0.0	NaN	144.0	60.0
20	Purce, Margaret	D	4.0	1.0	167.0	1	0.0	0.0	NaN	6.0	1.0
21	Rapinoe, Megan	F	7.0	5.0	349.0	7	1.0	0.0	NaN	175.0	59.0
22	Sauerbrunn, Becky	D	6.0	6.0	540.0	0	0.0	0.0	NaN	184.0	0.0
23	Smith, Sophia	F	3.0	0.0	68.0	0	0.0	0.0	NaN	4.0	0.0
24	Sonnett, Emily	D	5.0	2.0	266.0	0	1.0	0.0	NaN	51.0	0.0
25	Williams, Lynn	F	5.0	5.0	370.0	1	0.0	0.0	NaN	34.0	10.0
26	Own Goal	NaN	NaN	NaN	NaN	0	NaN	NaN	NaN	NaN	NaN
27	TOTAL	NaN	7.0	7.0	630.0	22	18.0	4.0	0.0	NaN	NaN
28	Opponents	NaN	7.0	7.0	630.0	1	1.0	12.0	0.0	NaN	NaN

Despues vamos a seleccionar los datos estadisticos de Unidad, Total y costo de unidad

In [11]:
```
print ("Estadisticos")
print ("----------------------------")
df_2[['GP','GS','Caps']].describe()
```

```
Estadisticos
----------------------------
```

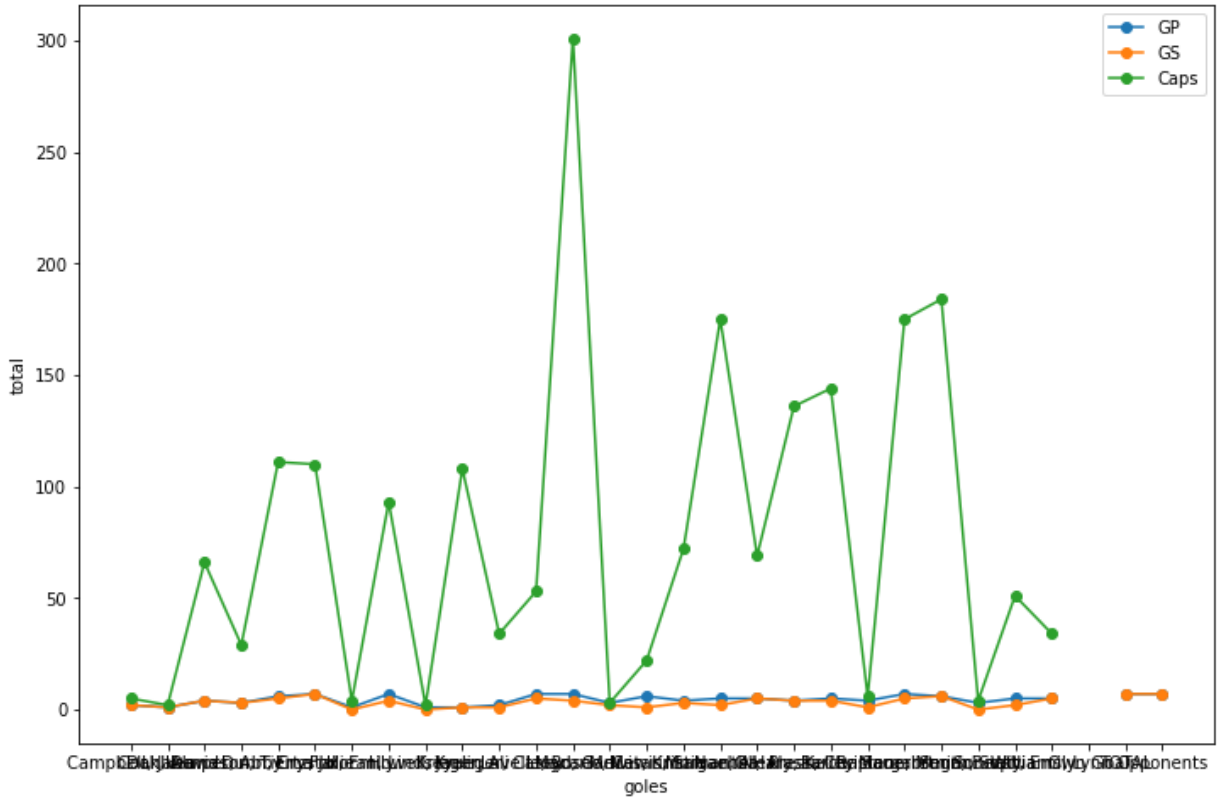
Out[11]:

	GP	GS	Caps
count	13.000000	13.000000	11.000000
mean	5.153846	3.923077	95.454545
std	1.281025	2.215910	69.184339
min	3.000000	0.000000	4.000000
25%	4.000000	2.000000	42.500000
50%	5.000000	4.000000	72.000000
75%	6.000000	5.000000	159.500000
max	7.000000	7.000000	184.000000

Ahora vamos realizar una nueva grafica con algunos puntos de acuerdo a los datos pedidos para la ejecucion

In [12]:
```
#x = np.arange(61)
x = df["Name"]
t1 = df["GP"]
t2 = df["GS"]
t3 = df["Caps"]
plt.figure(figsize=(12,8))
plt.plot(x,t1,x,t2,x,t3,marker='o')
plt.xlabel('goles')
plt.ylabel('total')
plt.legend(('GP','GS','Caps'), prop = {'size':10},loc='upper right')
```

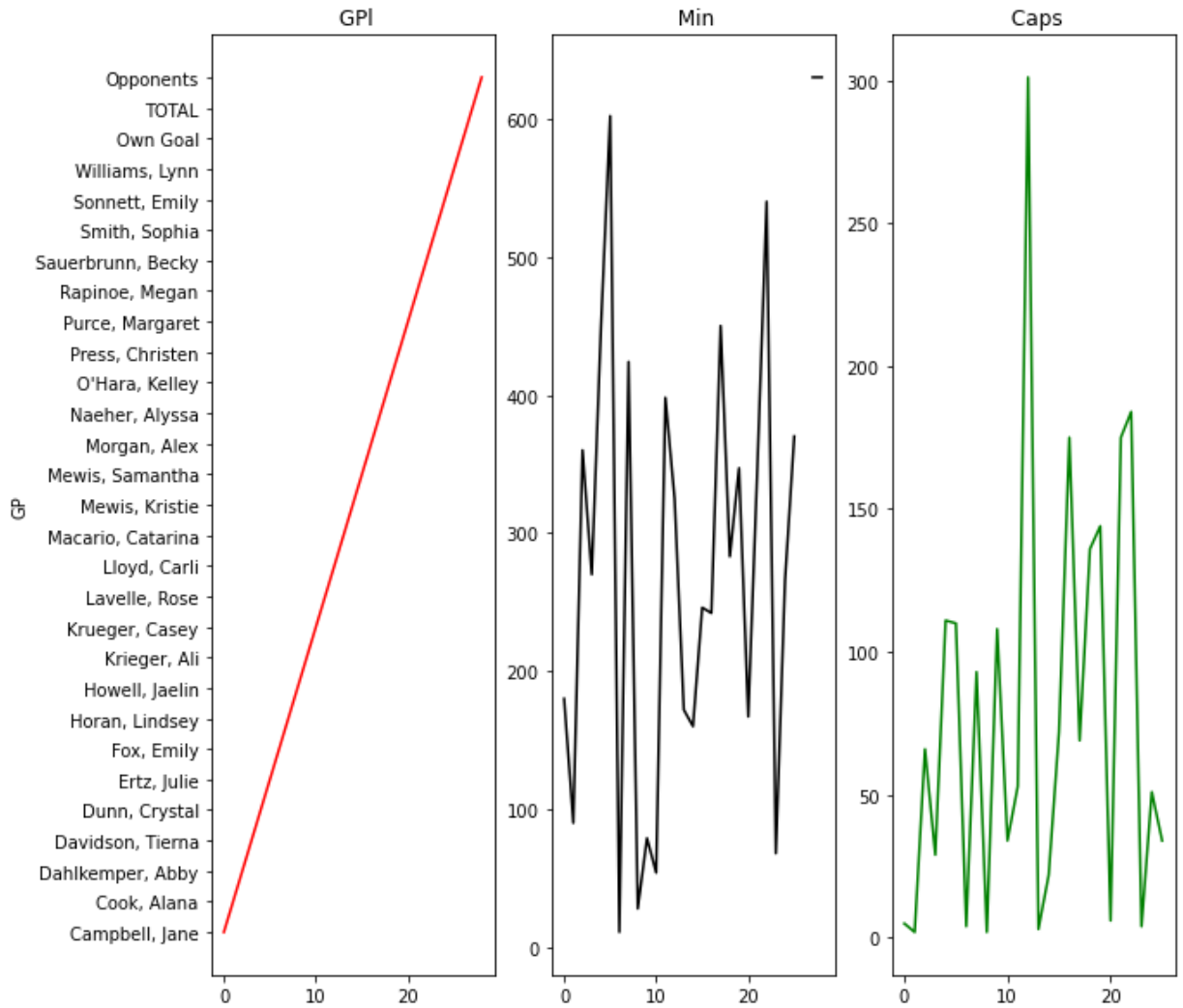
Out[12]:
```
<matplotlib.legend.Legend at 0x1f221029850>
```



Continuando con al ejecucion vamos a realizar tres graficas de las tres columnas que hemos utilizado el primer dato ingresado es el rango debe ser su umeracion exacta ya que sino no va a funcionar, luego sigue la informacion del primer dato incluyendo la r de red Mas adelante tenemos el siguiente dato con su respectiva infromacion y la b de blue Por ultimo tenemos el dato del total con g de green

In [13]:
```
#Entra del rango al Array con los datos que estan en la fuente de datos
x = range(29)
#Configuracion de las dimenciones que el grafico optendra
plt.figure(figsize=(10,10))
plt.subplot(131)
t1 = df["Name"]
p1 = plt.plot(x,t1, 'r')
plt.ylabel('GP')
plt.title(' GPl')
plt.subplot(132)
t2 = df["Min"]
p1 = plt.plot(x,t2,'k')
plt.title(' Min')
plt.subplot(133)
t3 = df["Caps"]
p1 = plt.plot(x,t3,'g')
plt.title(' Caps')
```

Out[13]: Text(0.5, 1.0, ' Caps')



Despues, vamos a obtener los datos unicos para eso usamos unique y guardamos los dtaos en una variable a la cual se le ha denominado dfclases

In [21]:
```
# OBTENER LOS DATOS UNICOS DE LA TABLA
lis = df["Name"].unique()
lis
dfclases=pd.DataFrame(lis,columns=["Name"])
dfclases
```

Out[21]:

	Name
0	Campbell, Jane
1	Cook, Alana
2	Dahlkemper, Abby
3	Davidson, Tierna
4	Dunn, Crystal
5	Ertz, Julie
6	Fox, Emily
7	Horan, Lindsey
8	Howell, Jaelin
9	Krieger, Ali
10	Krueger, Casey
11	Lavelle, Rose
12	Lloyd, Carli
13	Macario, Catarina
14	Mewis, Kristie
15	Mewis, Samantha
16	Morgan, Alex
17	Naeher, Alyssa
18	O'Hara, Kelley
19	Press, Christen
20	Purce, Margaret
21	Rapinoe, Megan
22	Sauerbrunn, Becky
23	Smith, Sophia
24	Sonnett, Emily
25	Williams, Lynn
26	Own Goal
27	TOTAL
28	Opponents

In [ ]:
```
####Tabla de frecuencias
```

In [23]:
```
#TABLA DE FRECUENCIAS ABSOLUTAS
# OBTENER FRECUENCIAS ABSOLUTAS DE CADA CLASE
import pandas as pd
datafi=pd.crosstab(index=df["Name"], columns = "M")
# Creamos una lista con los valores de las frecuencias
li = datafi.values
# agregamos una columna al dataframe
dfclases["M"] = li
#observamos dfclase
dfclases
```

Out[23]:

	Name	M
0	Campbell, Jane	1
1	Cook, Alana	1
2	Dahlkemper, Abby	1
3	Davidson, Tierna	1
4	Dunn, Crystal	1
5	Ertz, Julie	1
6	Fox, Emily	1
7	Horan, Lindsey	1
8	Howell, Jaelin	1
9	Krieger, Ali	1
10	Krueger, Casey	1
11	Lavelle, Rose	1
12	Lloyd, Carli	1
13	Macario, Catarina	1
14	Mewis, Kristie	1
15	Mewis, Samantha	1
16	Morgan, Alex	1
17	Naeher, Alyssa	1
18	O'Hara, Kelley	1
19	Press, Christen	1
20	Purce, Margaret	1
21	Rapinoe, Megan	1
22	Sauerbrunn, Becky	1
23	Smith, Sophia	1
24	Sonnett, Emily	1
25	Williams, Lynn	1
26	Own Goal	1
27	TOTAL	1

	Name	M
28	Opponents	1

In [24]:
```
# Columna de Frecuencia relativa
total = dfclases.sum(axis=0)
datahi = dfclases["M"]/total["M"] # aqui calculamos la frecuencia
datahi.values
# agregamos nueva columna de frecuencia relativa
dfclases["hi"] = datahi
dfclases
```

Out[24]:

	Name	M	hi
0	Campbell, Jane	1	0.034483
1	Cook, Alana	1	0.034483
2	Dahlkemper, Abby	1	0.034483
3	Davidson, Tierna	1	0.034483
4	Dunn, Crystal	1	0.034483
5	Ertz, Julie	1	0.034483
6	Fox, Emily	1	0.034483
7	Horan, Lindsey	1	0.034483
8	Howell, Jaelin	1	0.034483
9	Krieger, Ali	1	0.034483
10	Krueger, Casey	1	0.034483
11	Lavelle, Rose	1	0.034483
12	Lloyd, Carli	1	0.034483
13	Macario, Catarina	1	0.034483
14	Mewis, Kristie	1	0.034483
15	Mewis, Samantha	1	0.034483
16	Morgan, Alex	1	0.034483
17	Naeher, Alyssa	1	0.034483
18	O'Hara, Kelley	1	0.034483
19	Press, Christen	1	0.034483
20	Purce, Margaret	1	0.034483
21	Rapinoe, Megan	1	0.034483
22	Sauerbrunn, Becky	1	0.034483
23	Smith, Sophia	1	0.034483
24	Sonnett, Emily	1	0.034483
25	Williams, Lynn	1	0.034483
26	Own Goal	1	0.034483
27	TOTAL	1	0.034483
28	Opponents	1	0.034483

In [25]:
```
total1 = dfclases.sum(axis=0)
total1
```

Out[25]:
```
Name    Campbell, JaneCook, AlanaDahlkemper, AbbyDavid...
M                                                      29
hi                                                   1.0
dtype: object
```

In [27]:
```
# la suma de las frecuencias Relativas nos da 1
# aqui vamos a calcular la frecuencia absoluta
FA = dfclases["M"].values
# obtenemos FA
a=[]
b=0
for c in FA:
 b = c + b
 a.append(b)
dfclases["FA"] = a
HI = dfclases["hi"].values
# obtenemos HI
a=[]
b=0
for c in HI:
 b = c + b
 a.append(b)
dfclases["HI"] = a
dfclases
```

Out[27]:

	Name	M	hi	FA	HI
0	Campbell, Jane	1	0.034483	1	0.034483
1	Cook, Alana	1	0.034483	2	0.068966
2	Dahlkemper, Abby	1	0.034483	3	0.103448
3	Davidson, Tierna	1	0.034483	4	0.137931
4	Dunn, Crystal	1	0.034483	5	0.172414
5	Ertz, Julie	1	0.034483	6	0.206897
6	Fox, Emily	1	0.034483	7	0.241379
7	Horan, Lindsey	1	0.034483	8	0.275862
8	Howell, Jaelin	1	0.034483	9	0.310345
9	Krieger, Ali	1	0.034483	10	0.344828
10	Krueger, Casey	1	0.034483	11	0.379310
11	Lavelle, Rose	1	0.034483	12	0.413793
12	Lloyd, Carli	1	0.034483	13	0.448276
13	Macario, Catarina	1	0.034483	14	0.482759
14	Mewis, Kristie	1	0.034483	15	0.517241
15	Mewis, Samantha	1	0.034483	16	0.551724
16	Morgan, Alex	1	0.034483	17	0.586207
17	Naeher, Alyssa	1	0.034483	18	0.620690
18	O'Hara, Kelley	1	0.034483	19	0.655172
19	Press, Christen	1	0.034483	20	0.689655
20	Purce, Margaret	1	0.034483	21	0.724138
21	Rapinoe, Megan	1	0.034483	22	0.758621

	Name	M	hi	FA	HI
22	Sauerbrunn, Becky	1	0.034483	23	0.793103
23	Smith, Sophia	1	0.034483	24	0.827586
24	Sonnett, Emily	1	0.034483	25	0.862069
25	Williams, Lynn	1	0.034483	26	0.896552
26	Own Goal	1	0.034483	27	0.931034
27	TOTAL	1	0.034483	28	0.965517
28	Opponents	1	0.034483	29	1.000000

In [28]:
```
total1 = dfclases.sum(axis=0)
total1
```

Out[28]:
```
Name    Campbell, JaneCook, AlanaDahlkemper, AbbyDavid...
M                                                      29
hi                                                    1.0
FA                                                    435
HI                                                   15.0
dtype: object
```

In [29]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29 entries, 0 to 28
Data columns (total 11 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Name    29 non-null     object 
 1   Pos.    26 non-null     object 
 2   GP      28 non-null     float64
 3   GS      28 non-null     float64
 4   Min     28 non-null     float64
 5   G       29 non-null     int64  
 6   A       28 non-null     float64
 7   Y       28 non-null     float64
 8   R       2 non-null      float64
 9   Caps    26 non-null     float64
 10  Goals   24 non-null     float64
dtypes: float64(8), int64(1), object(2)
memory usage: 2.6+ KB
```

In [31]:
```
df["Min"]+100
```

Out[31]:
```
0     280.0
1     190.0
2     460.0
3     370.0
4     549.0
5     702.0
6     111.0
7     524.0
8     128.0
9     179.0
10    154.0
11    498.0
12    425.0
13    272.0
14    260.0
15    346.0
16    342.0
17    550.0
18    383.0
19    447.0
20    267.0
21    449.0
22    640.0
23    168.0
24    366.0
25    470.0
26      NaN
27    730.0
28    730.0
Name: Min, dtype: float64
```

In [ ]:
```

```