

# Notebooks, Pandas y más

ExactasPrograma - Datos

Facultad de Ciencias Exactas y Naturales, UBA

Invierno 2020

# Organización del curso

- El curso consiste en seis encuentros virtuales sincrónicos
- Durante tres semanas, martes y viernes de 14 a 17hs
- Trabajo en parejas, por Zoom

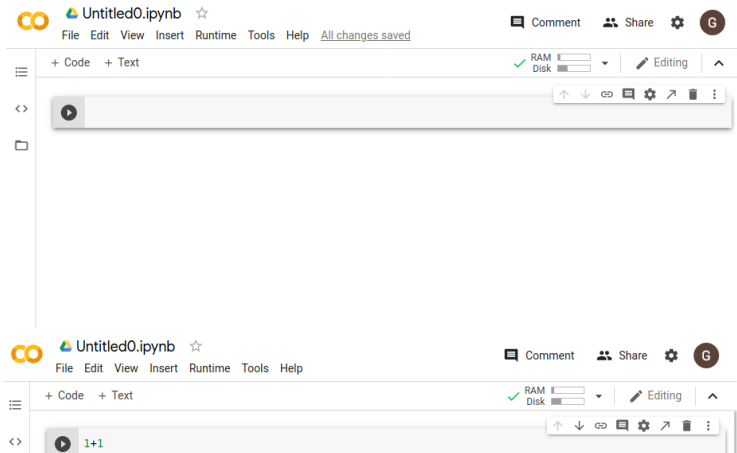
Sobre la interacción y las consultas:

- ¡Estamos para ayudarles!
- ¡**Internet** está para ayudarles!

# Tutor ya fue, Spyder ya fue

Vamos a usar un nuevo entorno de trabajo: *Notebooks*.

- Altamente usados por la comunidad
- Permiten ejecutar *porciones* de nuestro código
- Muy útiles para intercalar código con texto descriptivo



# DataFrame, ¿qué es? Excel te queremos


Vamos a trabajar con las respuestas que ustedes completaron en la inscripción.

Si las abrimos en algún programa de edición de planillas de cálculo, veremos algo como esto:

	A1	B	C	D	E	F
1	ID	ApellidoNombre	Genero	Edad	Carrera	Avance
2	1	Marley, Valentina	F	22.17442811	Ciencias Biológicas	5
3	2	Gonzalez, María	F	47.59274205	Paleontología	3
4	3	Sosa, Santiago	M	43.14851229	Ciencias de la Computación	3
5	4	Leloir, Tomás	M	48.27996017	Otra	5
6	5	Houssay, Santiago	M	19.53554825	Paleontología	4
7	6	San Martin, Lucía	F	23.47345245	Otra	3
8	7	Marley, Juan	M	24.52059587	Ciencias de la Computación	3
9	8	Romero, Catalina	F	19.3213189	Ciencias Químicas	3
10	9	Lopes, Thiago	M	23.16221722	Paleontología	5
11	10	Leloir,	M	19.23302366	Paleontología	3
12	11	Sadosky, Matías	M	57.23109404	Ciencias Geológicas	5
13	12	Marley, Santiago	M	40.15912771	Ciencias Matemáticas	3
14	13	Romero, Santiago	M	27.83287313	Ciencias Geológicas	1
15	14	Rodriguez, Elena	F	35.07996682	Ciencias Biológicas	5
16	15	Rodriguez, Elena	F	22.86492188	Ciencias de la Computación	3

# Para trabajar usaremos un archivo CSV

Una forma de trabajar con una planilla es transformarla en un archivo CSV (comma-separated values). Es decir, algo como esto:

C: > Users > Dario > Downloads >  datos\_1.csv

```

1 ID,"ApellidoNombre","Genero","Edad","Carrera","Avance","Promedio","NotaBaja","Inglés","Francés",
2 "1","Marley, Valentina","F","22.17442810580311","Ciencias Biológicas","5","6.941137506331256","1","
3 "2","Gonzalez, María","F","47.59274204857483","Paleontología","3","7.735495067564363","5","5","4",
4 "3","Sosa, Santiago","M","43.14851228859155","Ciencias de la Computación","3","5.184567753930846",
5 "4","Leloir, Tomás","M","48.27996016825624","Otra","5","7.802958819580786","5","4","3","Boca Junior
6 "5","Houssay, Santiago","M","19.535548249977673","Paleontología","4","9.751351847841455","3","2","1
7 "6","San Martin, Lucía","F","23.473452451023846","Otra","3","8.807721259195368","3","5","2","Racing
8 "7","Marley, Juan","M","24.52059587491095","Ciencias de la Computación","3","8.17873464998706","1",
9 "8","Romero, Catalina","F","19.321318895187936","Ciencias Químicas","3","5.522911573294247","4","4"
10 "9","Lopes, Thiago","M","23.16221721859209","Paleontología","5","8.886328618650573","3","3","2","Ir
11 "10","Leloir, ","M","19.233023660100077","Paleontología","3","9.327678884328638","5","4","2","Barce
12 "11","Sadovsky, Matías","M","57.23109403969551","Ciencias Geológicas","5","7.802522068427701","4","3
13 "12","Marley, Santiago","M","40.15912771295349","Ciencias Matemáticas","3","6.905161981746668","4",
14 "13","Romero, Santiago","M","27.832873127432887","Ciencias Geológicas","1","8.344334853044161","3",
15 "14","Rodriguez, Elena","F","35.07996682438139","Ciencias Biológicas","5","8.004444423464033","4",

```

## ¿Cómo lo leemos desde Python?

Usaremos el módulo `Pandas` para levantar nuestro “Excel”

```
import pandas as pd

df = pd.DataFrame("archivo.csv")
```

Normalmente, renombramos el módulo “pandas” a “pd”. Esto no es necesario, pero vamos a encontrar en Internet que lo hace una gran parte de la comunidad, es algo que se estila hacer y resulta cómodo.

La otra cosa que normalmente encontraremos es la variable “df” que es el diminutivo de `DataFrame`. Como cualquier variable dentro de `Python`, podemos poner cualquier nombre, pero en ejemplos chicos también encontraremos que se usa ese. ¡Podemos cambiarlo!

# ¿Qué tenemos dentro del DataFrame?

Si probamos hacer un `print` del DataFrame, ¡tenemos el Excel en Python!

```
print(df)
```

	ID	ApellidoNombre	Genero	...	Zodiaco	HoroscopoChino	Toro
0	1	Marley, Valentina	F	...	Sagitario	Conejo	1023.200756
1	2	Gonzalez, María	F	...	Virgo	NaN	1035.208593
2	3	Sosa, Santiago	M	...	Libra	Gallo	909.341347
3	4	Leloir, Tomás	M	...	Aries	Rata	1031.125285
4	5	Houssay, Santiago	M	...	Leo	Serpiente	993.740374
5	6	San Martin, Lucía	F	...	Tauro	Caballo	743.609141
6	7	Marley, Juan	M	...	Piscis	NaN	860.368046
7	8	Romero, Catalina	F	...	Capricornio	NaN	680.555630
8	9	Lopes, Thiago	M	...	Virgo	NaN	937.013405

## Ya cargamos el Excel en `Python`, ¿y Ahora?

Una vez que tenemos el `DataFrame` en `Python` podemos:

- Ver el contenido de una o más columnas.
- Contar la cantidad de valores distintos que haya en alguna de ellas
- Podemos filtrar información, por ejemplo, quedarnos solo con las filas que tienen altura mayor a *170cm*.
- Ordenar las filas según algún criterio que nosotros queramos.



## ¿Cómo seleccionamos algunas columnas solamente?

Suponiendo que queremos seleccionar solo las columnas *ApellidoNombre*, *Genero* y *Edad* podemos correr el siguiente código:

```
import pandas as pd

df = pd.read_csv("archivo.csv")

columnas = ["ApellidoNombre", "Genero", "Edad"]

df_con_menos_columnas = df[columnas]

print(df_con_menos_columnas)
```

	ApellidoNombre	Genero	Edad
0	Marley, Valentina	F	22.174428
1	Gonzalez, María	F	47.592742
2	Sosa, Santiago	M	43.148512
3	Leloir, Tomás	M	48.279960
4	Houssay, Santiago	M	19.535548
5	San Martin, Lucía	F	23.473452
6	Marley, Juan	M	24.520596
7	Romero, Catalina	F	19.321319

## ¿Y como filtramos algunas filas?

En caso que quisiéramos filtrar filas y quedarnos solo con los de *Sagitario*, debemos crear un filtro y luego aplicarlo:

```
filtro_sagitario = df["Zodiaco"] == "Sagitario"

df_solo_sagitario = df[filtro_sagitario]

print(df_solo_sagitario)
```

	ID	ApellidoNombre	Genero	...	Zodiaco	HoroscopoChino	Toro
0	1	Marley, Valentina	F	...	Sagitario	Conejo	1023.200756
11	12	Marley, Santiago	M	...	Sagitario	Caballo	1271.538618
14	15	Rodriguez, Elena	F	...	Sagitario	Gallo	648.700208
18	19	Sosa,	M	...	Sagitario	NaN	856.920899
24	25	Rodriguez, Martina	F	...	Sagitario	Dragón	1030.055536
26	27	Leloir, Matías	M	...	Sagitario	NaN	1412.342605
30	31	San Martin, Santino	M	...	Sagitario	Serpiente	846.216396
39	40	Fernandez, Nicolás	M	...	Sagitario	NaN	1139.341166
40	41	Rodriguez,	M	...	Sagitario	Tigre	649.083031
45	46	Rodriguez, Matías	M	...	Sagitario	Perro	675.316971
48	49	Sosa, Juan	M	...	Sagitario	Caballo	1123.624584

¡Ojo! Los filtros para un DataFrame no sirven para otro DataFrame que tenga otros datos. **¡Pueden no tener las mismas filas!**

## ¡Podemos combinar distintos filtros!

En caso que queramos las filas que son *Sagitario* en la columna *Zodiaco* **Y/O** *Gallo* en la columna *HoroscopoChino* podemos combinar dos filtros:

```
filtro_sagitario = df["Zodiaco"] == "Sagitario"
filtro_gallo = df["HoroscopoChino"] == "Gallo"
```

```
df_filtrado_and = df[filtro_sagitario & filtro_gallo]
df_filtrado_or = df[filtro_sagitario | filtro_gallo]
```

```
print(df_filtrado_and)
print(df_filtrado_or)
```

```

      ID  ApellidoNombre Genero  ...  Zodiaco HoroscopoChino      Toro
14  15  Rodriguez, Elena      F  ...  Sagitario           Gallo  648.700208
```

```
[1 rows x 19 columns]
```

```
[1 rows x 19 columns]
```

```

      ID  ApellidoNombre Genero  ...  Zodiaco HoroscopoChino      Toro
0    1  Marley, Valentina      F  ...  Sagitario     Conejo  1023.200756
2    3    Sosa, Santiago      M  ...    Libra           Gallo   909.341347
11  12  Marley, Santiago      M  ...  Sagitario   Caballo  1271.538618
12  13  Romero, Santiago      M  ...    Leo           Gallo   537.436394
14  15  Rodriguez, Elena      F  ...  Sagitario           Gallo  648.700208
15  16    Lopes, Elena      F  ...    Tauro           Gallo   695.051581
16  17  Sadosky, Matías      M  ...    Leo           Gallo   795.085961
17  18  Marley, Santino      M  ... Capricornio       Gallo  1063.527643
18  19    Sosa,           M  ...  Sagitario           NaN   856.920899
```

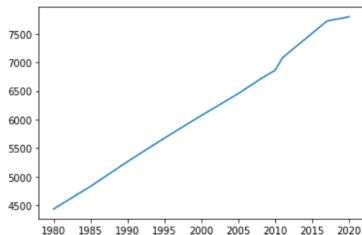
## Graficando como ya sabemos

En Exactas Programa ya aprendimos a graficar utilizando matplotlib. Supongamos que tenemos la población mundial de los últimos 40 años, lo podríamos graficar así. El primer argumento debe ser una lista de los valores del eje x, y el segundo los valores del eje y.

```
import matplotlib.pyplot as plt

years = [1980, 1985, 1990, 1995, 2000, 2005, 2008, 2010, 2011, 2017, 2020]
# Poblacion esta en millones
population = [4434, 4830, 5263, 5674, 6070, 6453, 6709, 6863, 7082, 7722, 7800]

plt.plot(years, population)
plt.show()
```



# Usando Seaborn

En esta edición usaremos el modulo `Seaborn`, que nos da algunas ventajas:

- Se integra con los *DataFrame* por lo que no debemos pasarles las listas una a una
- Con un *DataFrame* podemos graficar muchas cosas con poco esfuerzo
- Tiene muchos tipos diferentes de gráficos
- Los gráficos son más lindos :)

# DataFrame para graficar con Seaborn

Tenemos el siguiente *DataFrame* con la misma información que antes:

```
print(df)
```

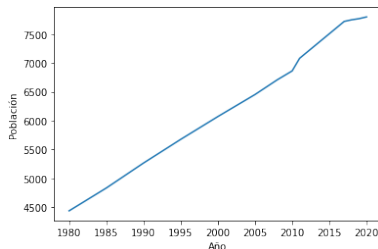
	Año	Población
0	1980	4434
1	1985	4830
2	1990	5263
3	1995	5674
4	2000	6070
5	2005	6453
6	2008	6709
7	2010	6863
8	2011	7082
9	2017	7722
10	2018	7750
11	2019	7770
12	2020	7800

Y ahora queremos graficar la población mundial al correr de los años :)

# DataFrame para graficar con Seaborn

Lo primero que debemos hacer antes de usar un modulo es importarlo y luego usaremos *Seaborn* para graficar la población mundial según el año.

```
import seaborn as sns  
  
sns.lineplot(data=df, x="Año", y="Población")
```

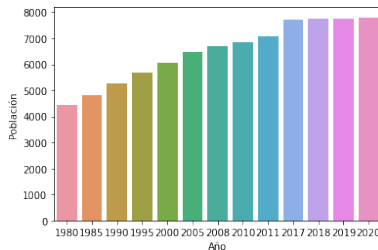


Veamos que a *Seaborn* le debemos decir cual es el *DataFrame* a usar bajo el argumento *data*, y como parámetros *x* e *y* le decimos el nombre de las columnas dentro del *DataFrame* que queremos graficar.

# DataFrame para graficar con Seaborn

¿Y si quisieramos otro tipo de gráfico? Bueno, podemos cambiar la función "lineplot" por alguna otra como puede ser "barplot"

```
sns.barplot(data=df, x="Año", y="Población")
```



¿Y qué otros gráficos existen? <https://seaborn.pydata.org/>



## ¿Cómo pido ayuda?

Palabras clave para buscar en internet:

- Pandas
- Seaborn + tipo de gráfico
- Probar, si es posible, términos en inglés: Filter, plot, figure

Ejemplo: ¡Mi gráfico es muy chiquito y no se ve! → “seaborn change plot size”

Si nuestro código no funciona, siempre es una buena opción probar incluir en la búsqueda *una parte* del mensaje de error

En Zoom, → “Ask For Help” y un docente se acercará a ayudarles :)

¡A trabajar!