_EcoInformática

Data Análisis in Python
Pandas & Seaborn

Virginia Domínguez García

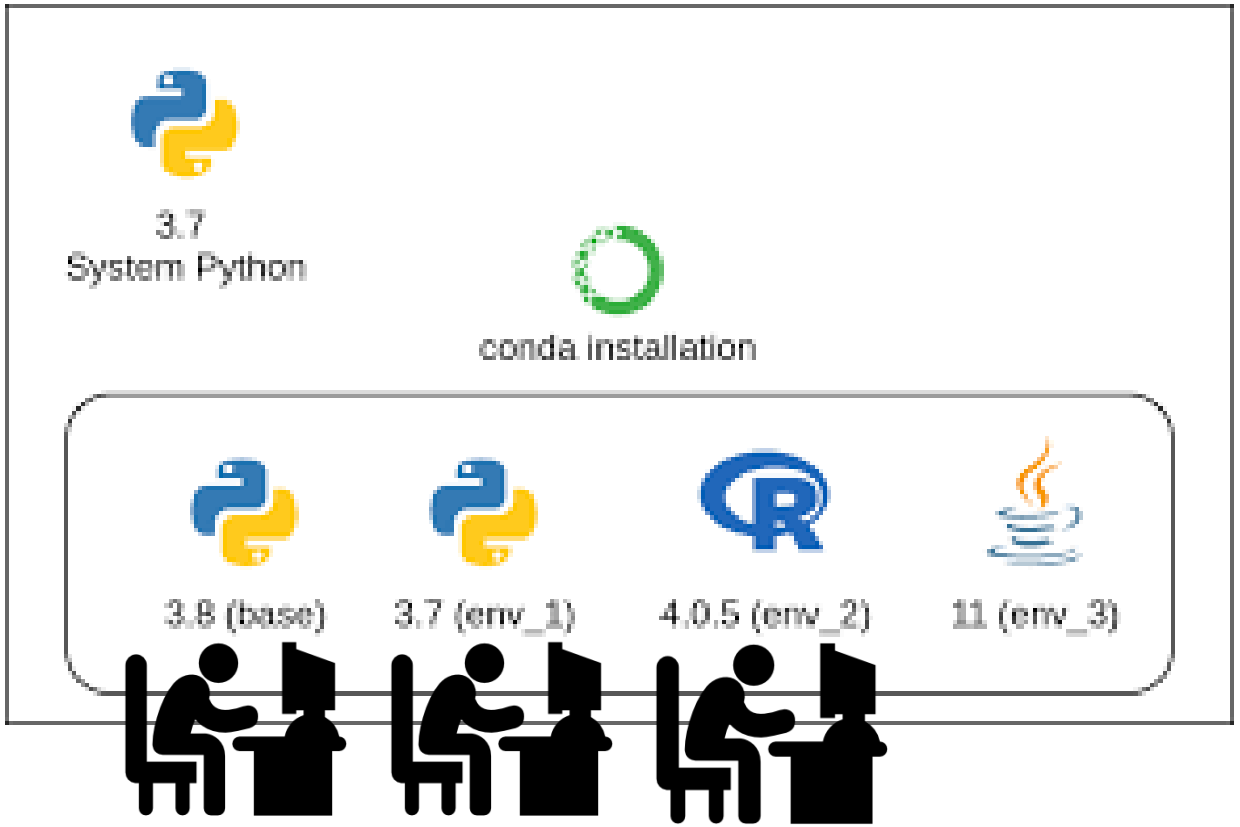
Seminarios Ecoinformática AEET

2022

Estación Biológica de Doñana CSIC

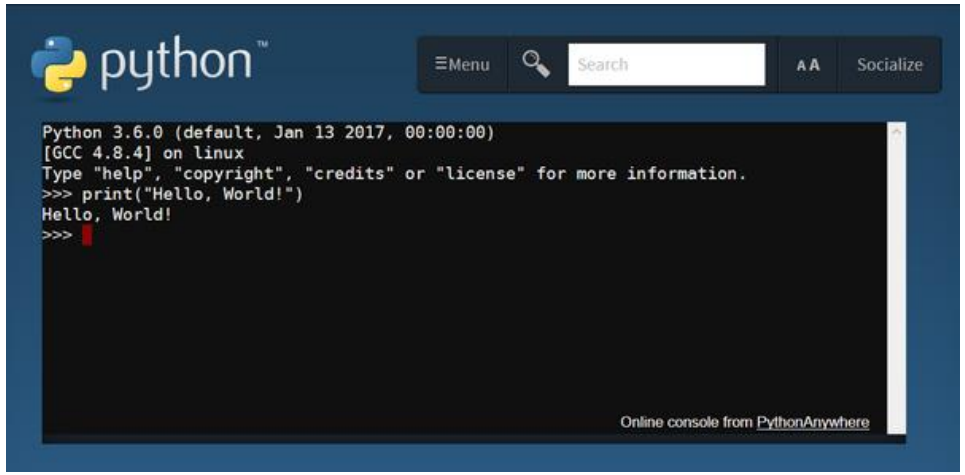
1 - ¿Por qué Python en Anaconda?

1 - ¿Por qué Python en Anaconda?

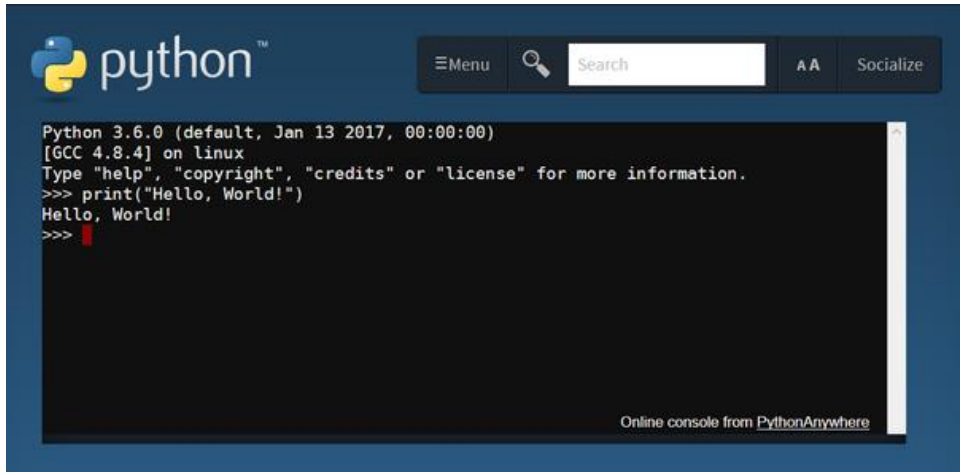


https://whiteboxml.com/blog/the-definitive-guide-to-python-virtual-environments-with-conda

2 - ¿Cómo trabajar en Python?

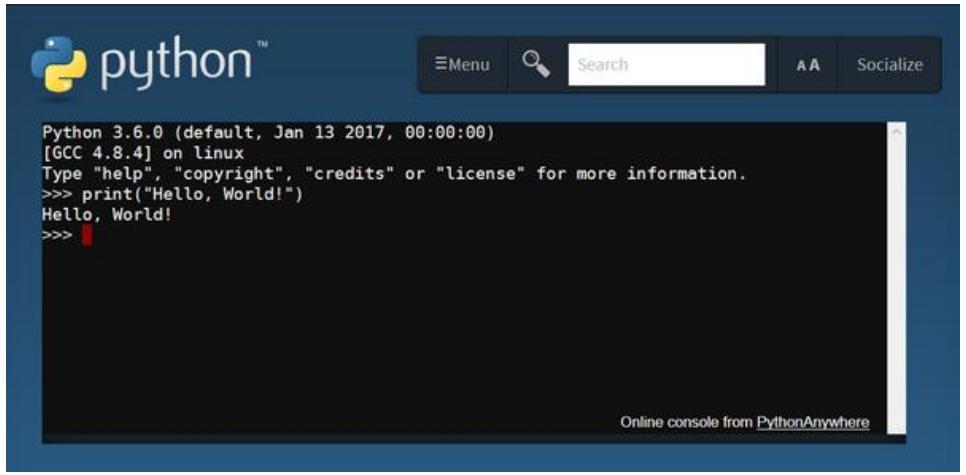


2 - ¿Cómo trabajar en Python?

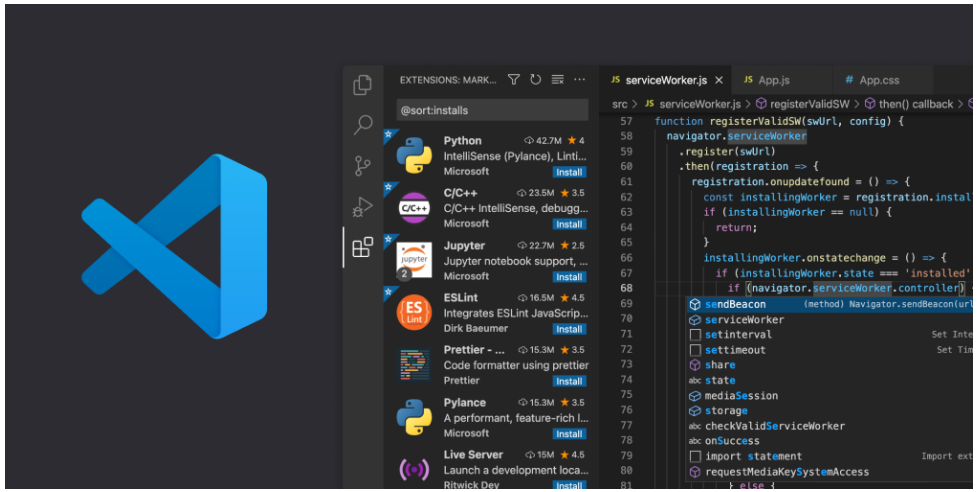



```python
def parse_arguments():
    """Read arguments from a command line."""
    parser = argparse.ArgumentParser(description='Arguments get parsed via --commands')
    parser.add_argument('-v', metavar='verbosity', type=int, default=2,
                        help='Verbosity of logging: 0 -critical, 1 -warning, 2 -info, 3 -debug')

    args = parser.parse_args()
    verbose = {0: logging.CRITICAL, 1: logging.WARNING, 2: logging.INFO, 3: logging.DEBUG}
    logging.basicConfig(format='%(message)s', level=verbose[args.v], stream=sys.stdout)

    return args


def main():
    pass


if __name__ == '__main__':
    args = parse_arguments()
    main()
```

2 - ¿Cómo trabajar en Python?

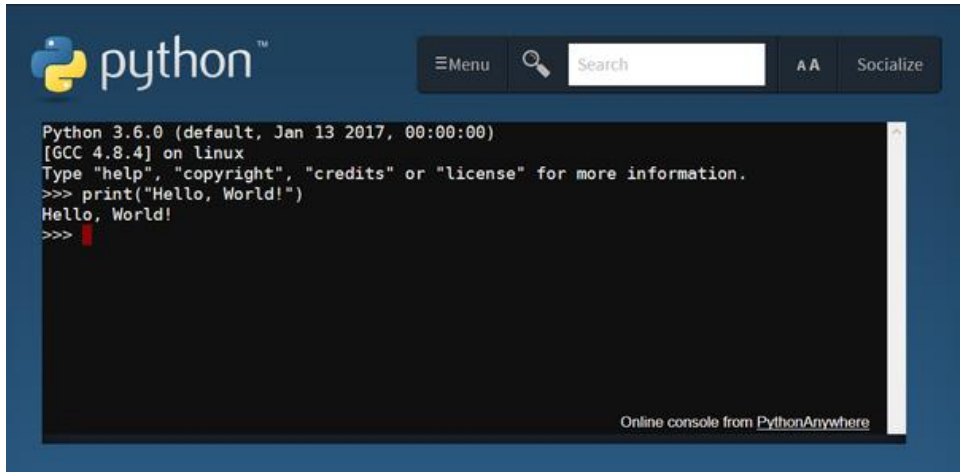



```python
def parse_arguments():
    """Read arguments from a command line."""
    parser = argparse.ArgumentParser(description='Arguments get parsed via --commands')
    parser.add_argument('-v', metavar='verbosity', type=int, default=2,
                        help='Verbosity of logging: 0 -critical, 1 -warning, 2 -info, 3 -debug')

    args = parser.parse_args()
    verbose = {0: logging.CRITICAL, 1: logging.WARNING, 2: logging.INFO, 3: logging.DEBUG}
    logging.basicConfig(format='%(message)s', level=verbose[args.v], stream=sys.stdout)

    return args


def main():
    pass


if __name__ == '__main__':
    args = parse_arguments()
    main()
```

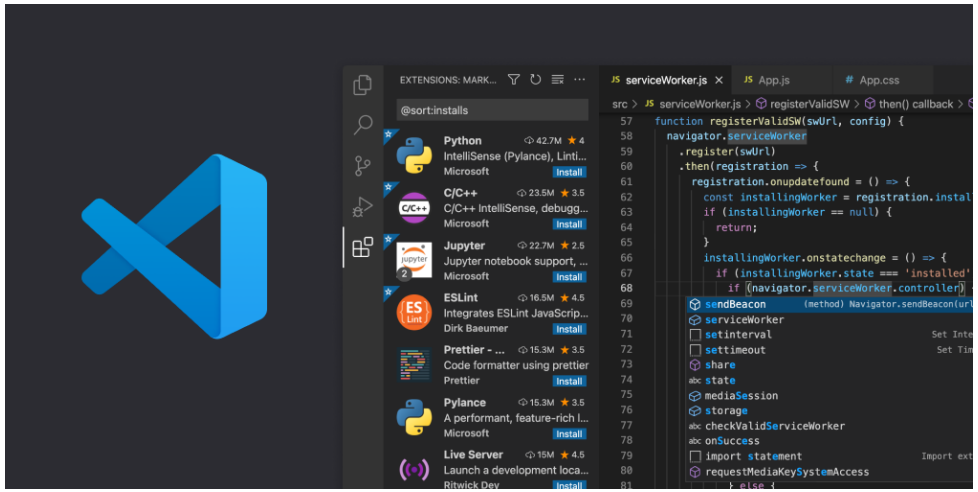


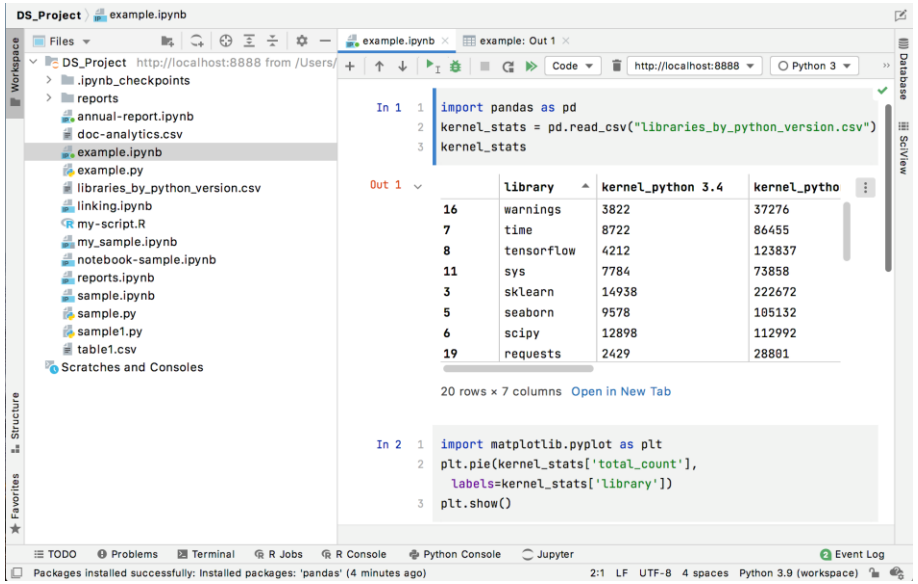
https://code.visualstudio.com/

2 - ¿Cómo trabajar en Python?



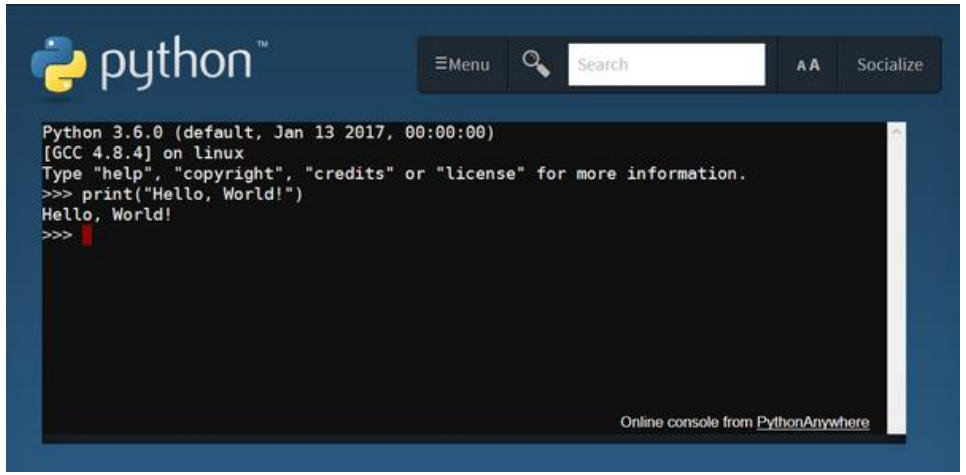

```python
def parse_arguments():
    """Read arguments from a command line."""
    parser = argparse.ArgumentParser(description='Arguments get parsed via --commands')
    parser.add_argument('-v', metavar='verbosity', type=int, default=2,
                        help='Verbosity of logging: 0 -critical, 1 -warning, 2 -info, 3 -debug')

    args = parser.parse_args()
    verbose = {0: logging.CRITICAL, 1: logging.WARNING, 2: logging.INFO, 3: logging.DEBUG}
    logging.basicConfig(format='%(message)s', level=verbose[args.v], stream=sys.stdout)

    return args


def main():
    pass


if __name__ == '__main__':
    args = parse_arguments()
    main()
```

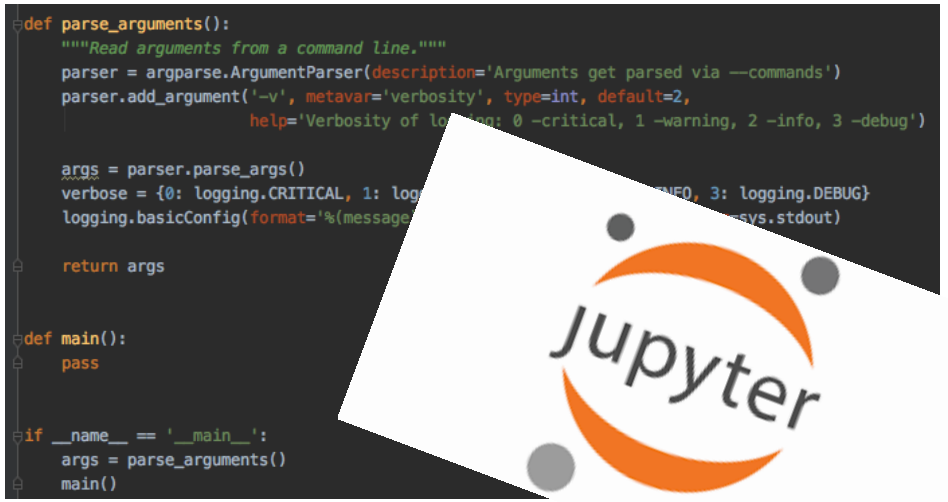


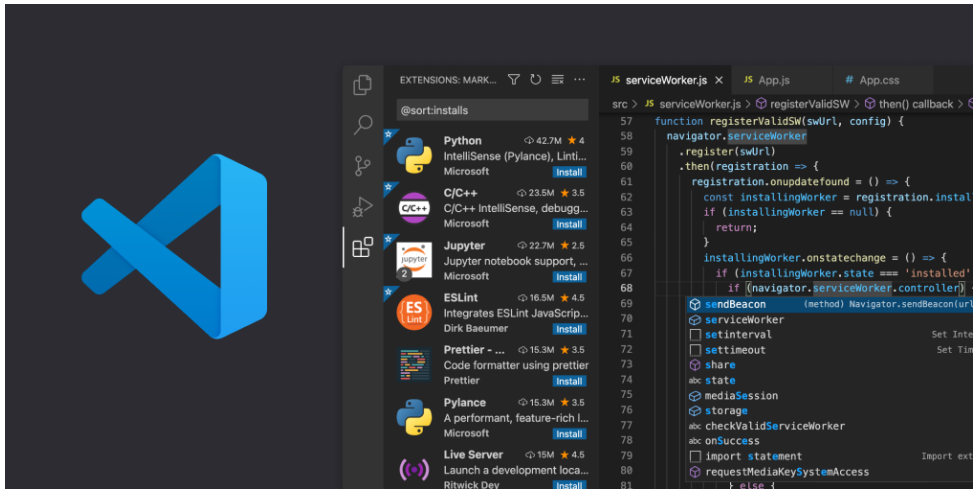


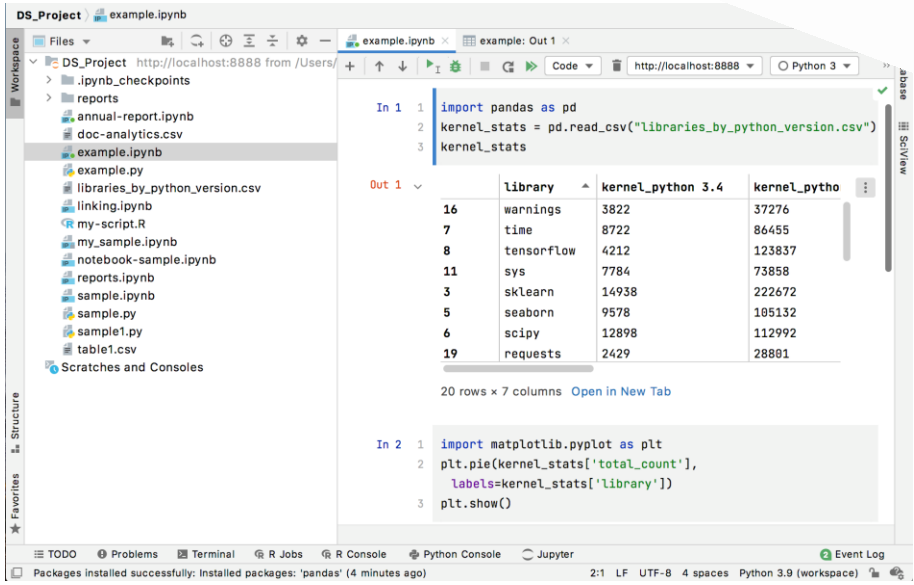
https://code.visualstudio.com/

2 - ¿Cómo trabajar en Python?









https://code.visualstudio.com/

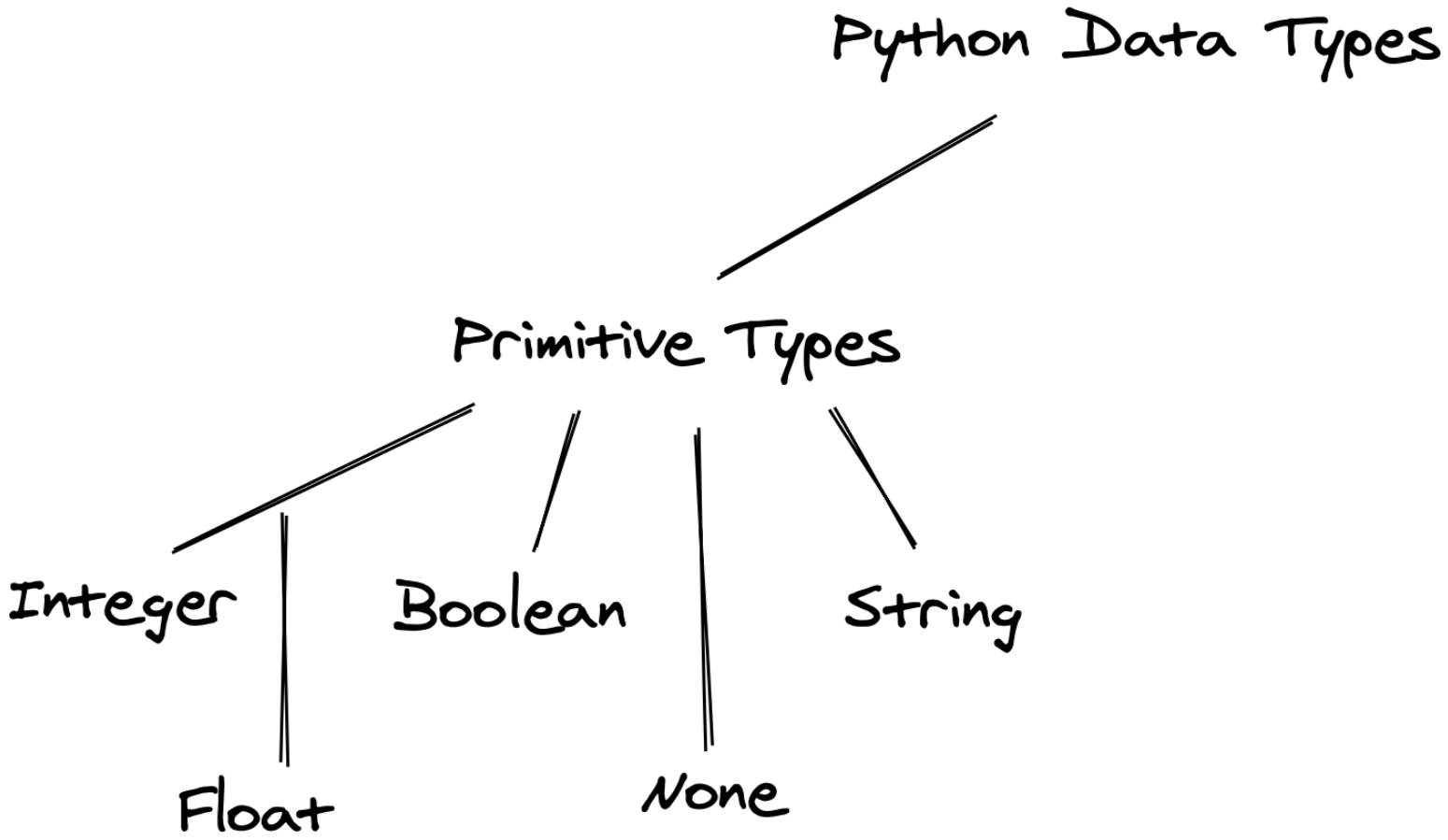
2 – Instalar jupyter notebook

https://towardsdatascience.com/how-to-set-up-anaconda-and-jupyter-notebook-the-right-way-de3b7623ea4a

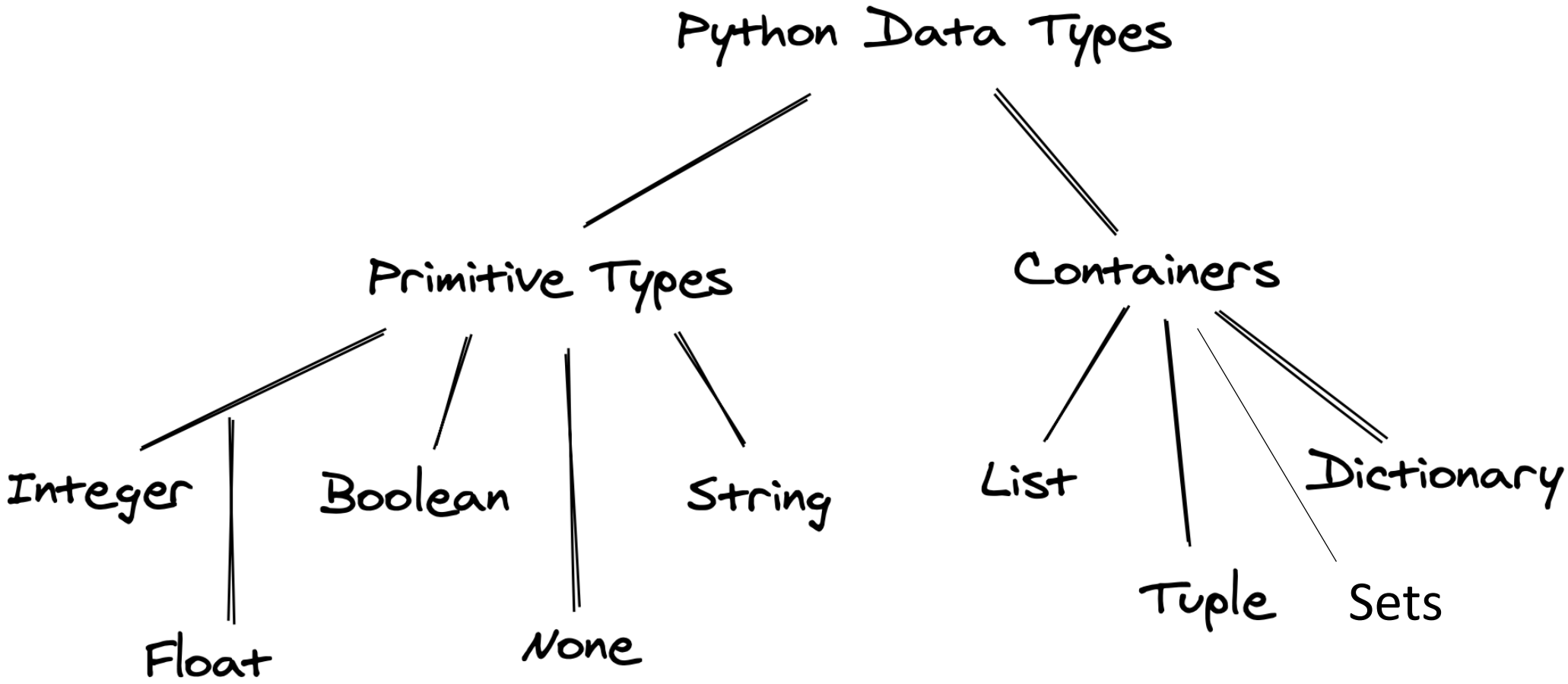
Install the notebook
conda install -c conda-forge notebook
conda install -c conda-forge nb_conda_kernels

Add environment kernel to notebook
conda activate cenv
(cenv)$ conda install ipykernel
(cenv)$ ipython kernel install --user --name=<any_name_for_kernel>

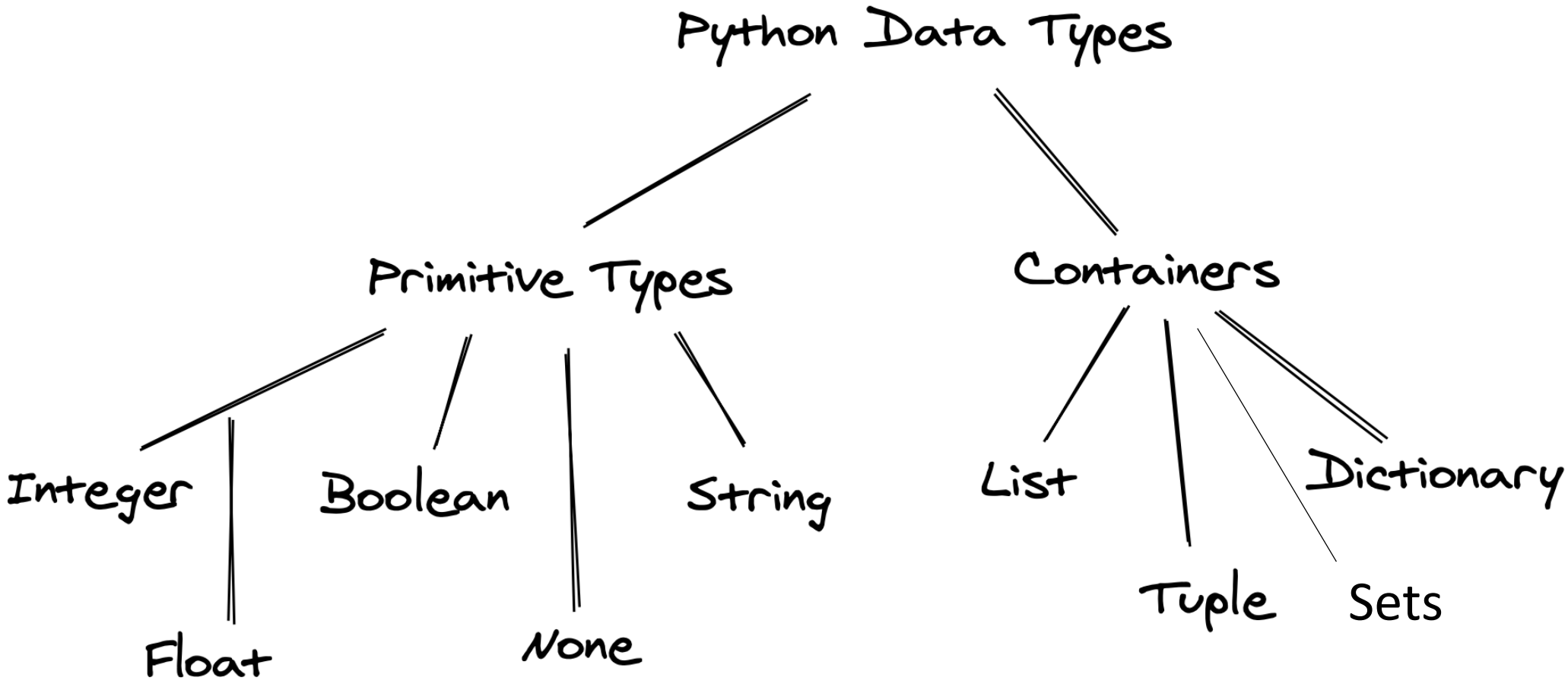
3 – Tipos de datos en Python



3 – Tipos de datos en Python

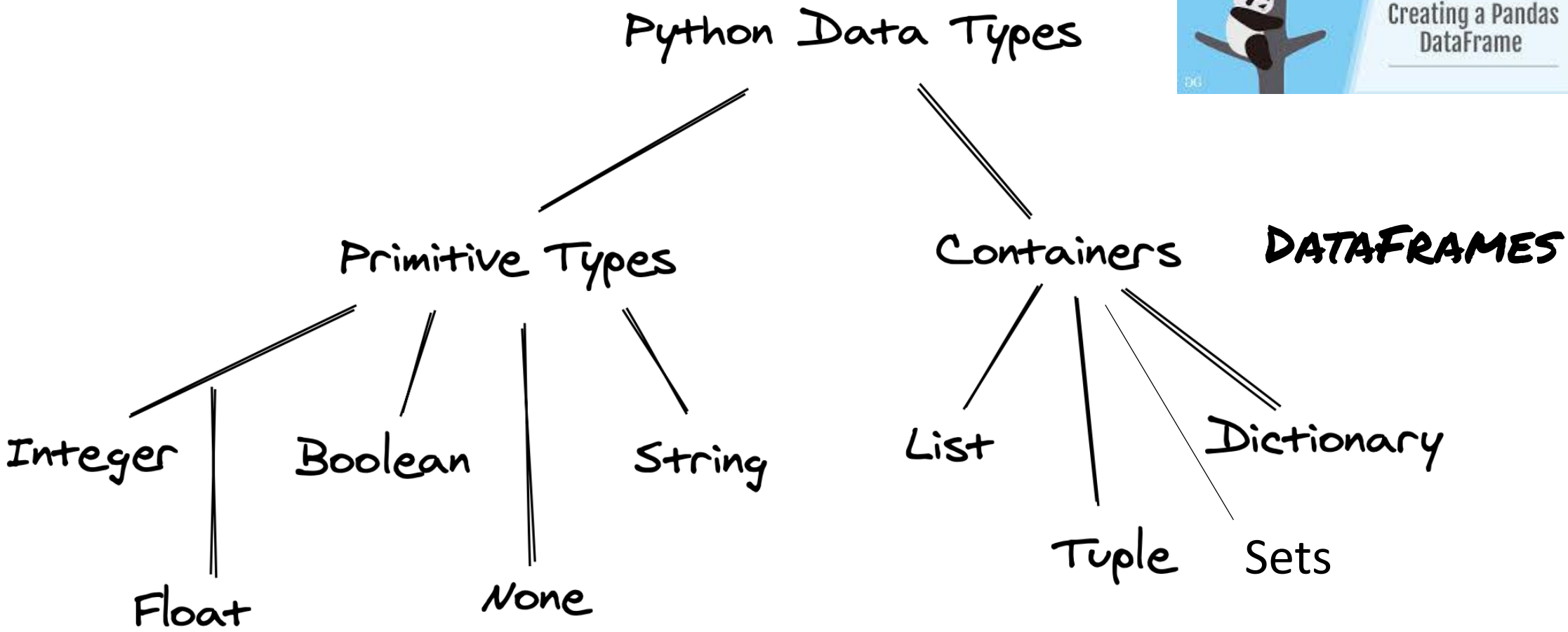


3 – Tipos de datos en Python



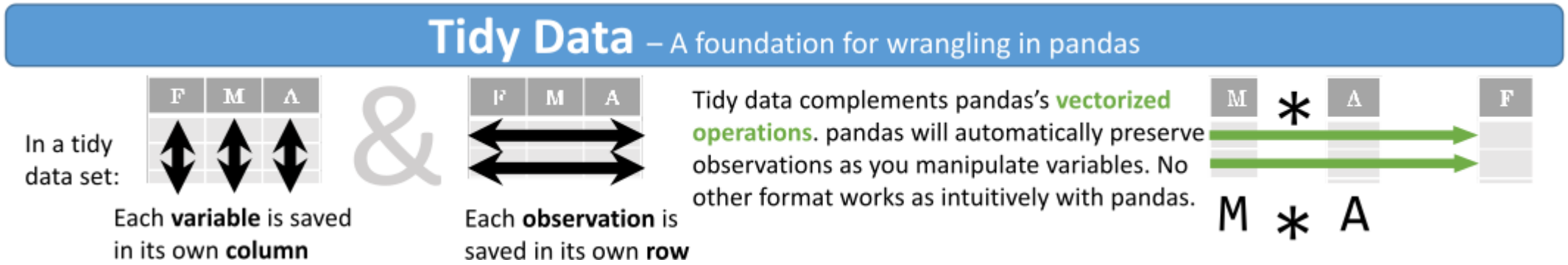
Accesso a elemento: container.element
Método: container.method.()

3 – Tipos de datos en Python



Accesso a elemento: container.element
Método: container.method.()

3 – Tipos de datos en Python



Accesso a elemento: container.element
Método: container.method.()

3 – Creando un DataFrame

df.info()
pokemon.df.hist()

4 – Data Overview

Consultar Inicio y Final

df.head(5)

df.tail(5)

Recuperar el número de filas y columnas (shape

df.shape

Información general

df.info() -> tipos de datos

df.describe() -> estadística básica

df.hist() -> histograma

4 – Data Overview

Consultar Inicio y Final

df.head(5)

df.tail(5)

Recuperar el número de filas y columnas (shape

df.shape

Información general

df.info() -> tipos de datos

df.describe() -> estadística básica

Información de distribución de datos y correlaciones

df.hist() -> histograma

df.corr() -> correlación

pandas provides a large set of summary functions that operate on different kinds of pandas objects (DataFrame columns, Series, GroupBy, Expanding and Rolling (see below)) and produce single values for each of the groups. When applied to a DataFrame, the result is returned as a pandas Series for each column. Examples:

`sum()`
Sum values of each object.

`count()`
Count non-NA/null values of each object.

`median()`
Median value of each object.

`quantile([0.25,0.75])`
Quantiles of each object.

`apply(function)`
Apply function to each object.

`min()`
Minimum value in each object.

`max()`
Maximum value in each object.

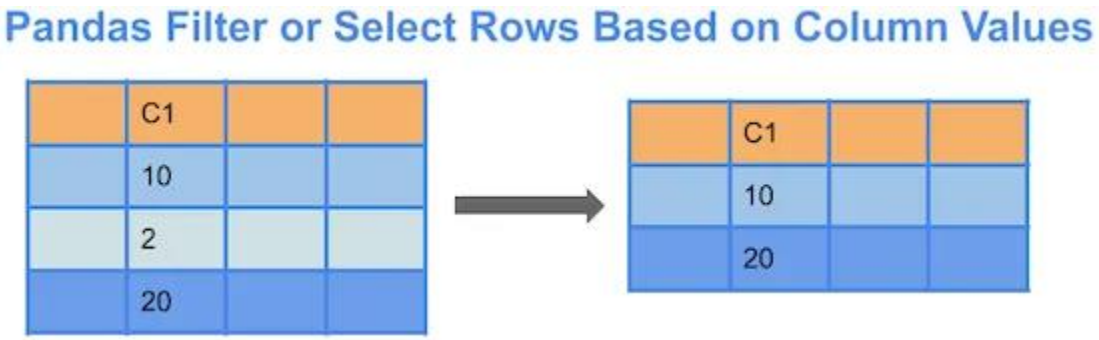
`mean()`
Mean value of each object.

`var()`
Variance of each object.

`std()`
Standard deviation of each object.

5 – Selección de datos

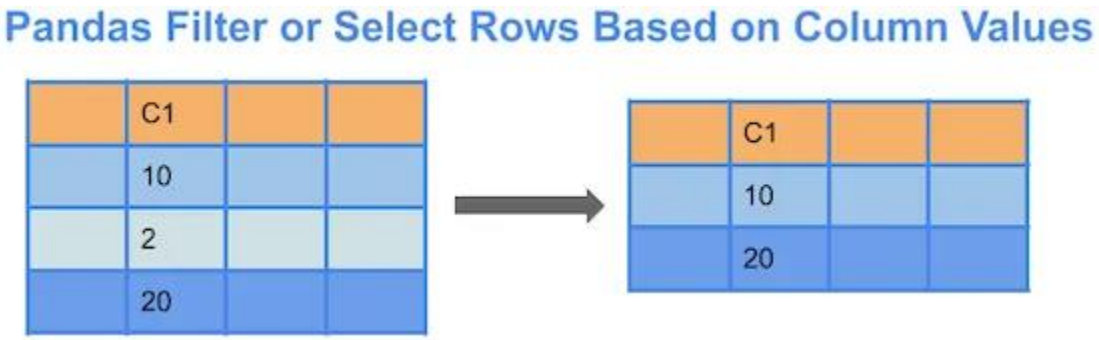
Selección **Booleana**: Atendiendo a los valores de las celdas



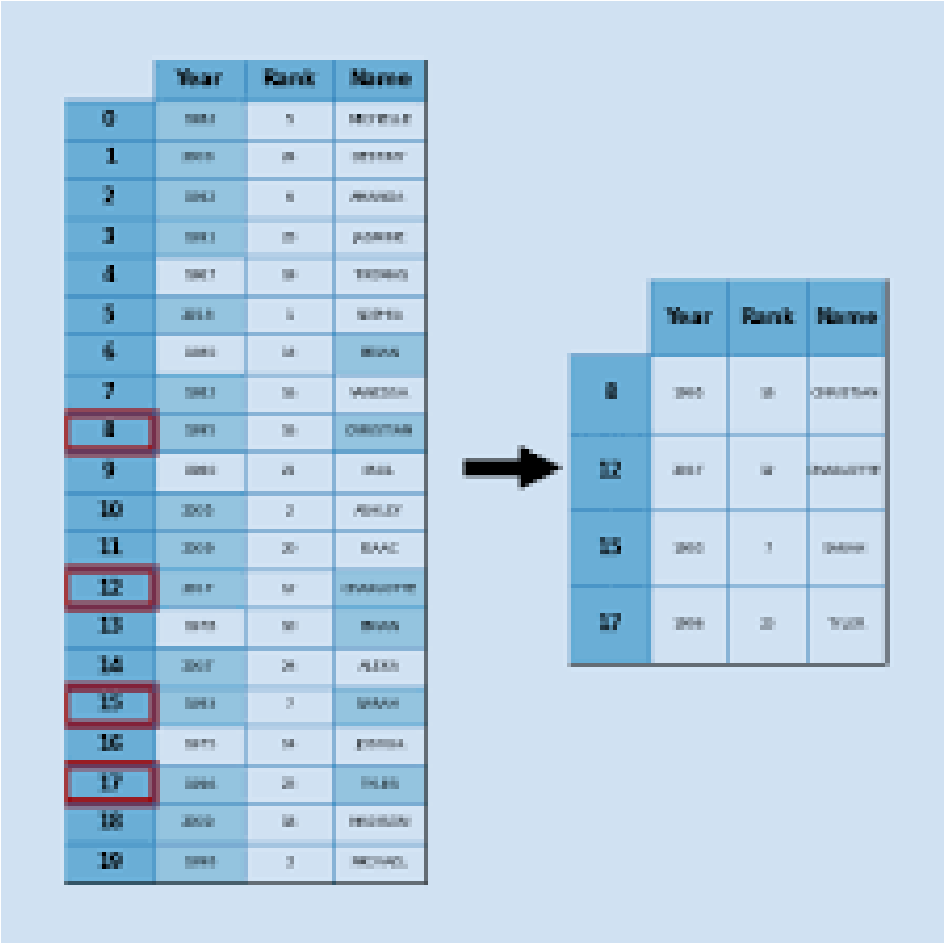
Selección **Indexada**: Atendiendo a las etiqueta
s que hayamos usado para indexar

5 – Selección de datos

Selección **Booleana**: Atendiendo a los valores de las celdas

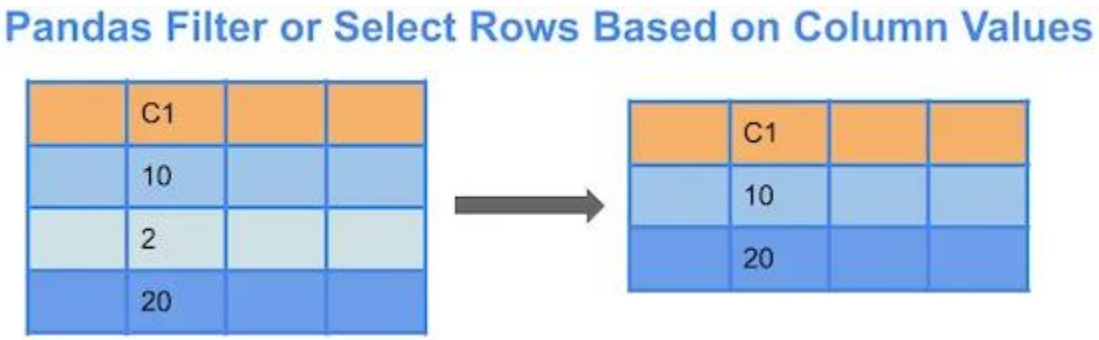


Selección **Indexada**: Atendiendo a las etiqueta
s que hayamos usado <u>para</u> indexar

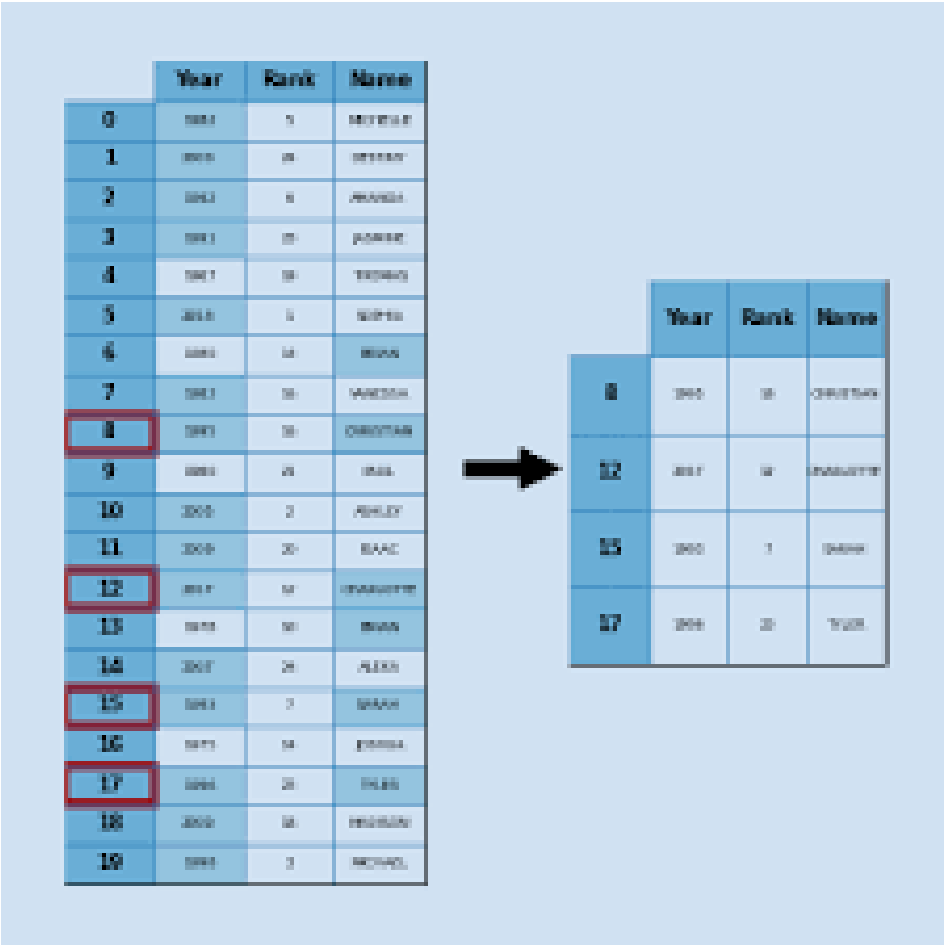


5 – Selección de datos

Selección **Booleana**: Atendiendo a los valores de las celdas



Selección **Indexada**: Atendiendo a las etiqueta
s que hayamos usado para indexar



Selección **por tipo de dato**

6 – Limpieza de datos

Información que falta: NaN

* Los localizamos con **.isna()**
* Eliminamos las filas con NaN con **.dropna()**
* Rellenamos los valors que faltan con **.fillna()**

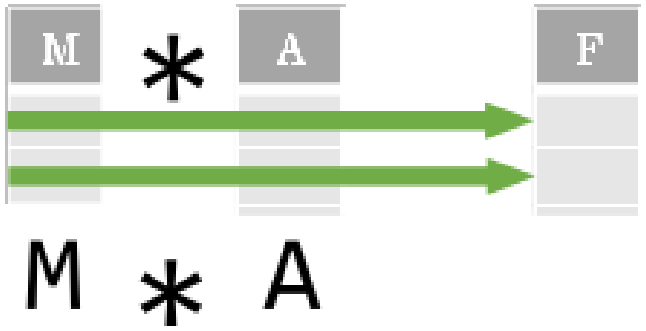
Renombrando columnas y mapeando valores:

Cambiar nombres **de** columnas**: df.rename(**columns=diccionario**)**

Cambiar valores **en** la columns: df[‘column']**.map(**diccionario**)**

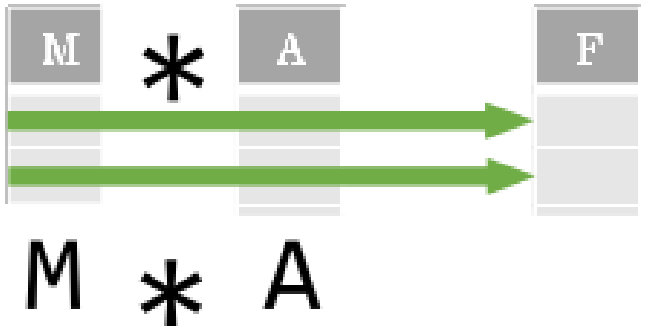
7 – Operando en la DataFrame

Las Operaciones que hagamos van a ser vectoriales por defecto (afectan a toda la df)

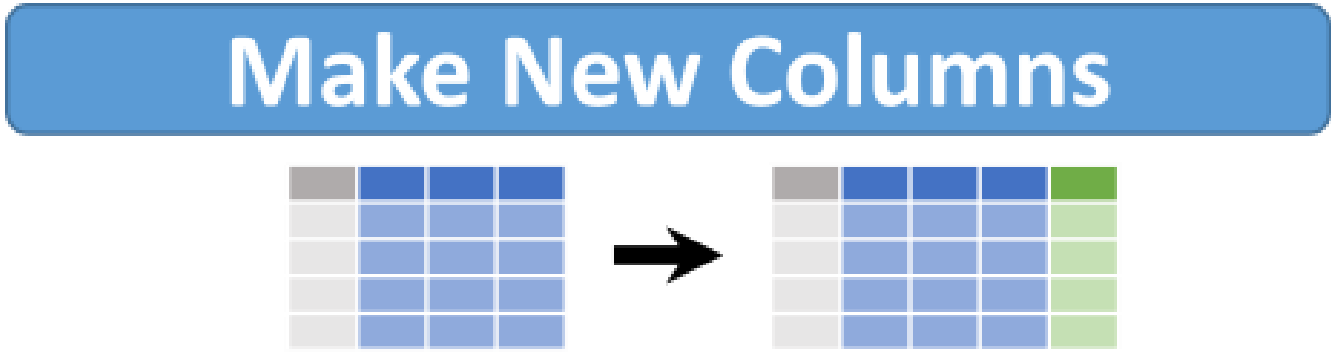


7 – Operando en la DataFrame

Las Operaciones que hagamos van a ser vectoriales por defecto (afectan a toda la df)



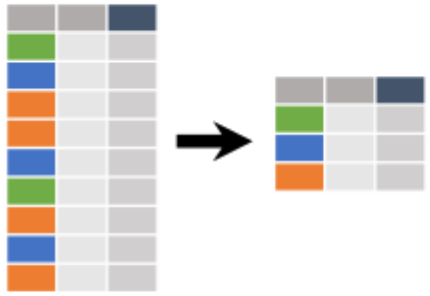
Para guardar la información **creamos neuevas columnas**:
df**.loc[:,"**Name"**]**= operation on rows



8 – Agrupando información y resumiendo

Agrupamos por categorias y aplicamos agregación con las summary functions

Group Data



`df.groupby(by="col")`
Return a GroupBy object, grouped by values in column named "col".

`df.groupby(level="ind")`
Return a GroupBy object, grouped by values in index level named "ind".

All of the summary functions listed above can be applied to a group.
Additional GroupBy functions:

`size()`
Size of each group.

`agg(function)`
Aggregate group using function.

pandas provides a large set of summary functions that operate on different kinds of pandas objects (DataFrame columns, Series, GroupBy, Expanding and Rolling (see below)) and produce single values for each of the groups. When applied to a DataFrame, the result is returned as a pandas Series for each column. Examples:

`sum()`
Sum values of each object.

`count()`
Count non-NA/null values of each object.

`median()`
Median value of each object.

`quantile([0.25,0.75])`
Quantiles of each object.

`apply(function)`
Apply function to each object.

`min()`
Minimum value in each object.

`max()`
Maximum value in each object.

`mean()`
Mean value of each object.

`var()`
Variance of each object.

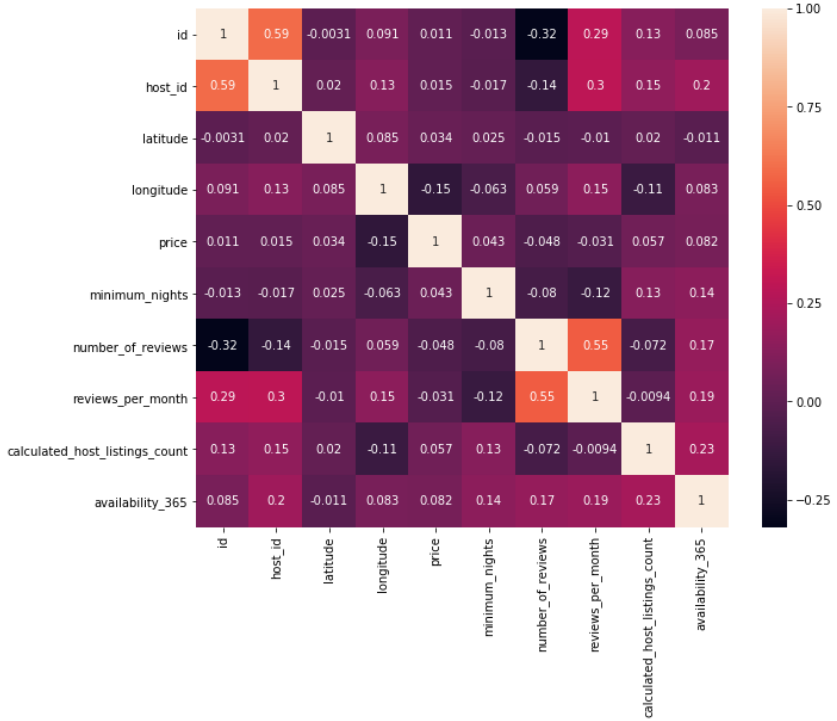
`std()`
Standard deviation of each object.

9 – Correlaciones

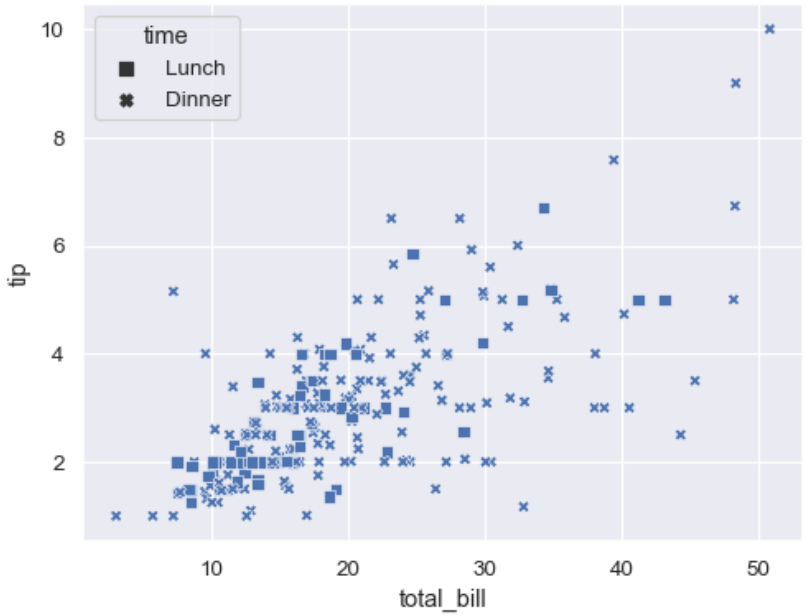
Podemos usar métodos solamente numéricos : **df.corr()**

Podemos usar métodos más gráficos

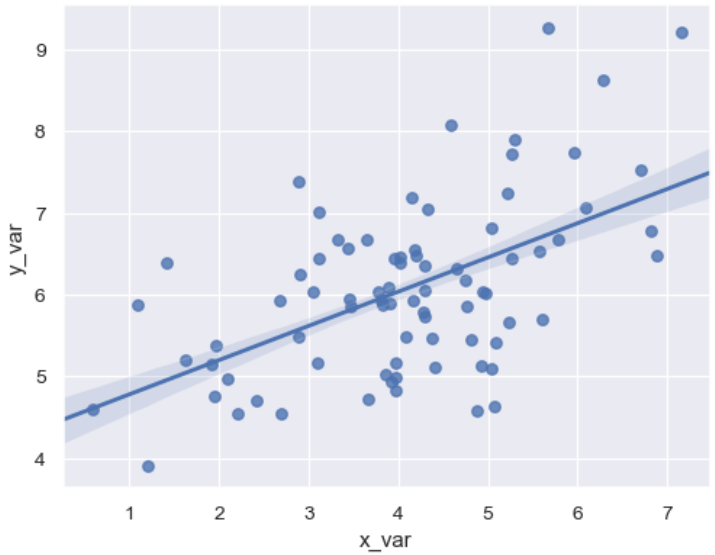
sns.heatmap(df.corr())



sns.scatterplot()



sns.regplot()



10 – Sorting and Ranking

Podemos usar métodos solamente numéricos : **df.corr()**

Ranking

The examples below can also be applied to groups. In this case, the function is applied on a per-group basis, and the returned vectors are of the length of the original DataFrame.

shift(1)
Copy with values shifted by 1.

rank(method='dense')
Ranks with no gaps.

rank(method='min')
Ranks. Ties get min rank.

rank(pct=True)
Ranks rescaled to interval [0, 1].

rank(method='first')
Ranks. Ties go to first value.

shift(-1)
Copy with values lagged by 1.

cumsum()
Cumulative sum.

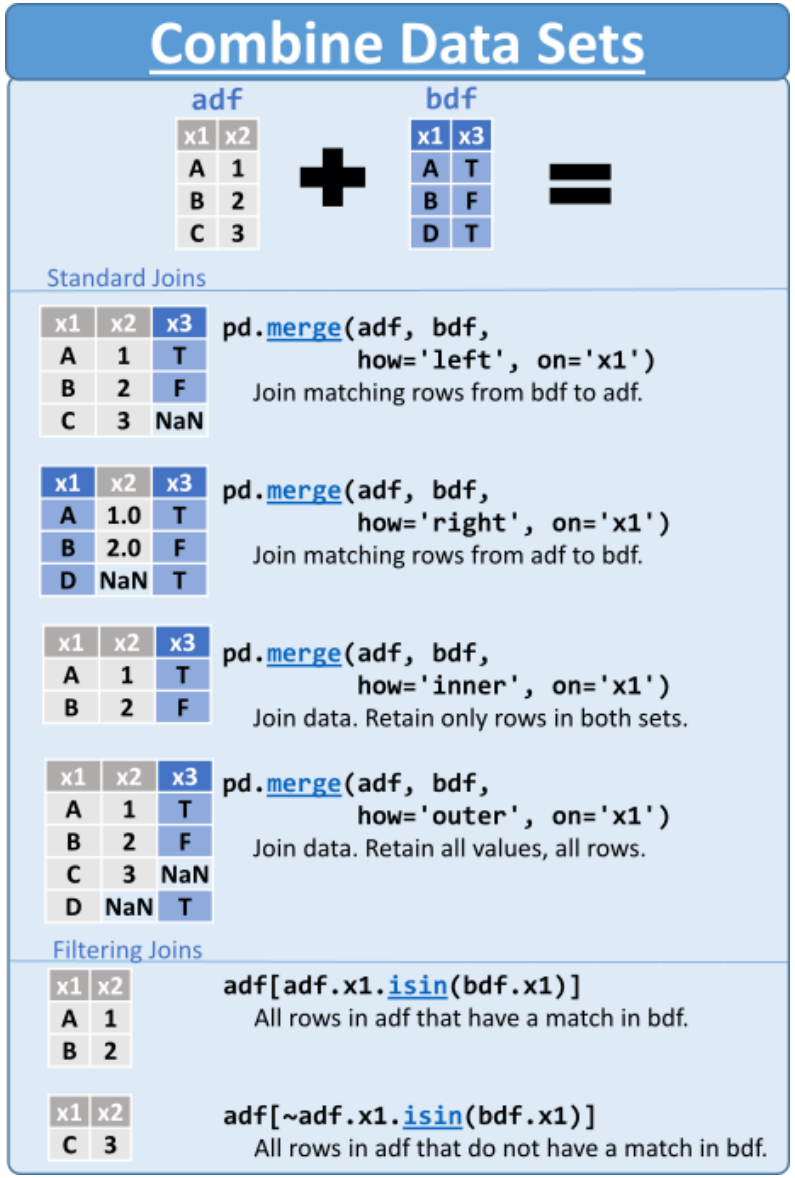
cummax()
Cumulative max.

cummin()
Cumulative min.

cumprod()

11 – Combinando DataFrames: merge y concat

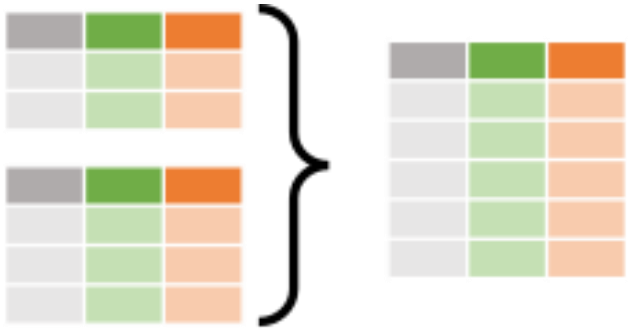
Merge: Une **dos** DataFrames con **al menos un índice en común**



11 – Combinando DataFrames: merge y concat

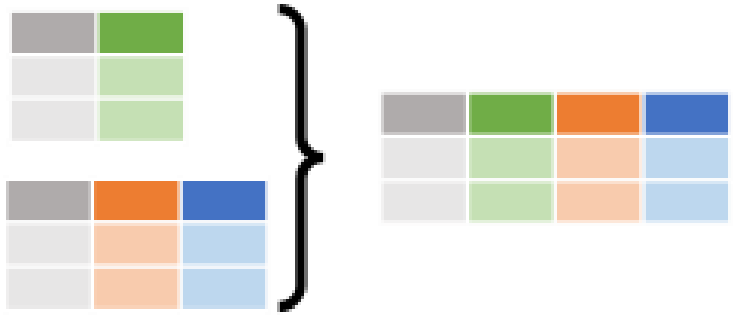
Merge: Une **dos** DataFrames con **al menos un índice en común**

Concat: Combina **varias** DataFrames en una nueva



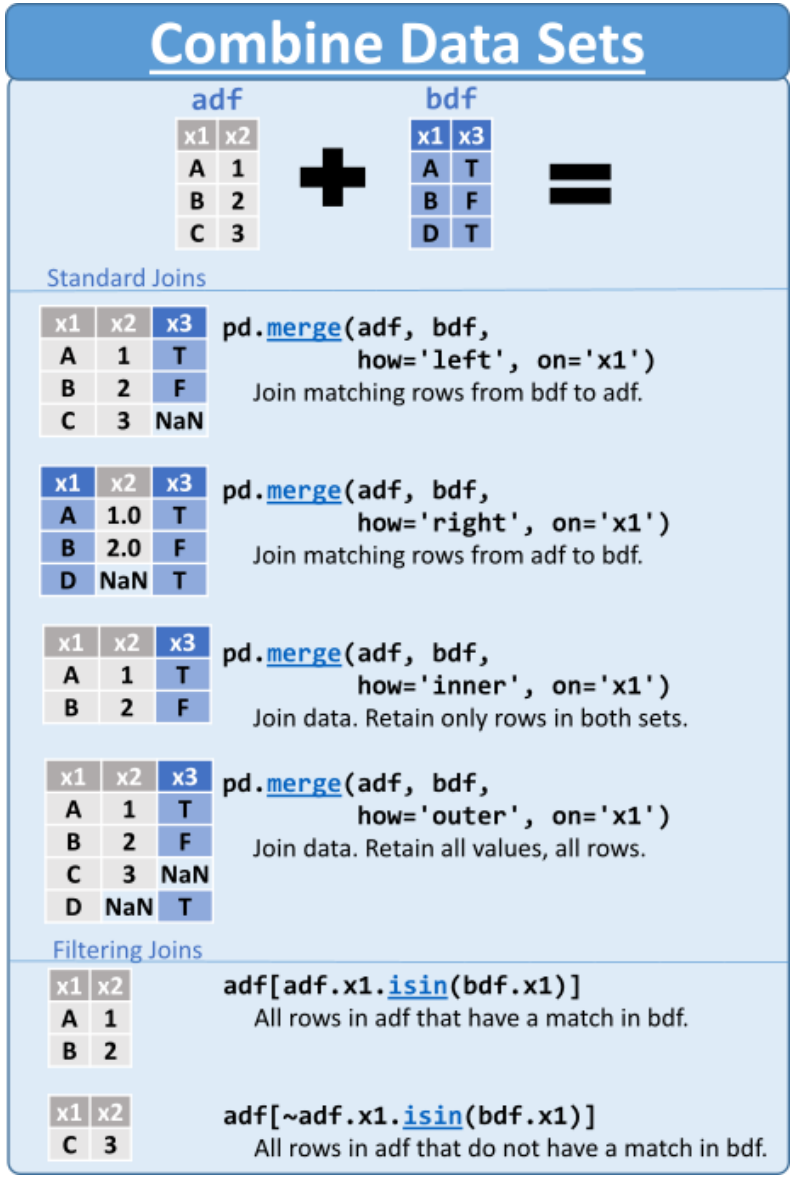
`pd.concat([df1,df2])`

Append rows of DataFrames



`pd.concat([df1,df2], axis=1)`

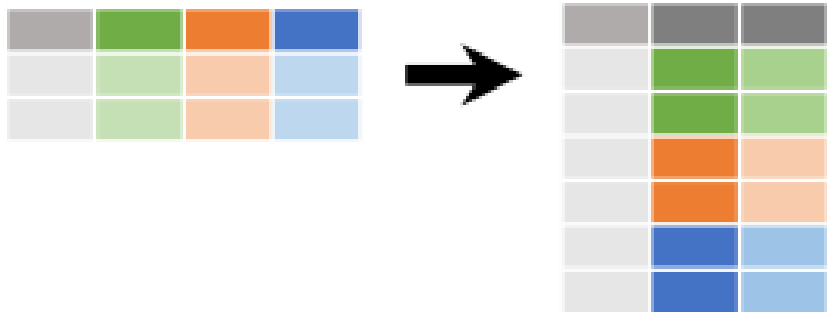
Append columns of DataFrames



12 – Reestucturar datos: .melt() y .pivot()

Hay casos en los que estaremos obligados a cambiar la estructura de la da base de datos, porque por ejemplo no sea tidy.

pd.melt() nos permite hacerlo



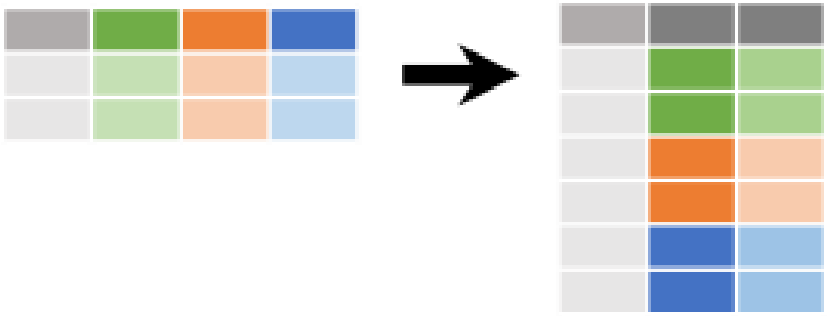
`pd.melt(df)`

Gather columns into rows.

12 – Reestucturar datos: .melt() y .pivot()

Hay casos en los que estaremos obligados a cambiar la estructura de la da base de datos, porque por ejemplo no sea tidy.

pd.melt() nos permite hacerlo ->



`pd.melt(df)`
Gather columns into rows.

`df.pivot(columns='var', values='val')`
Spread rows into columns.

En otros casos nos interesará pasar de una estructura tidy a una table cruzada, por ejemplo para convertir una lista de adyacencias de una red en una matriz.

¡Gracias por tu atención!

https://pandas.pydata.org/docs/user_guide/index.html

¡Muchos ejercicios!
https://www.w3resource.com/python-exercises/pandas/index.php