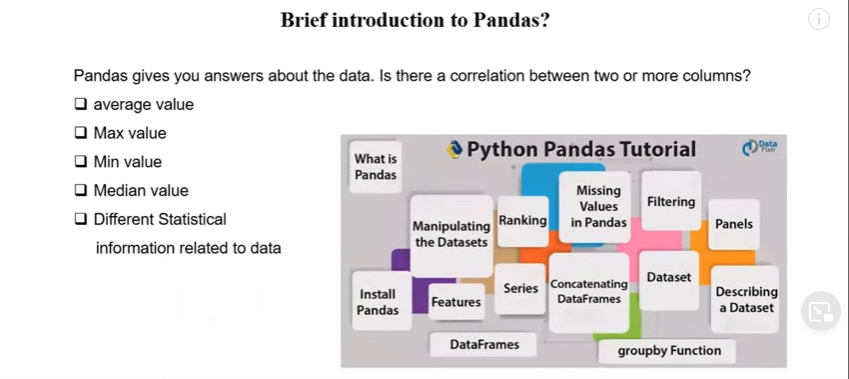
**Introduction to Pandas**

**Pandas:** *Pandas* is basically used for manipulating data and doing data analysis. It inherits some data structures & operations that are able to manipulate numerical tables and time series.

Pandas.

Tasks like Data cleaning, Data transformation, Future election, Encoding categorical data, Correlation, Imbalance data handling, Outlier detection and all sorts of other tasks can be done using *Pandas*.

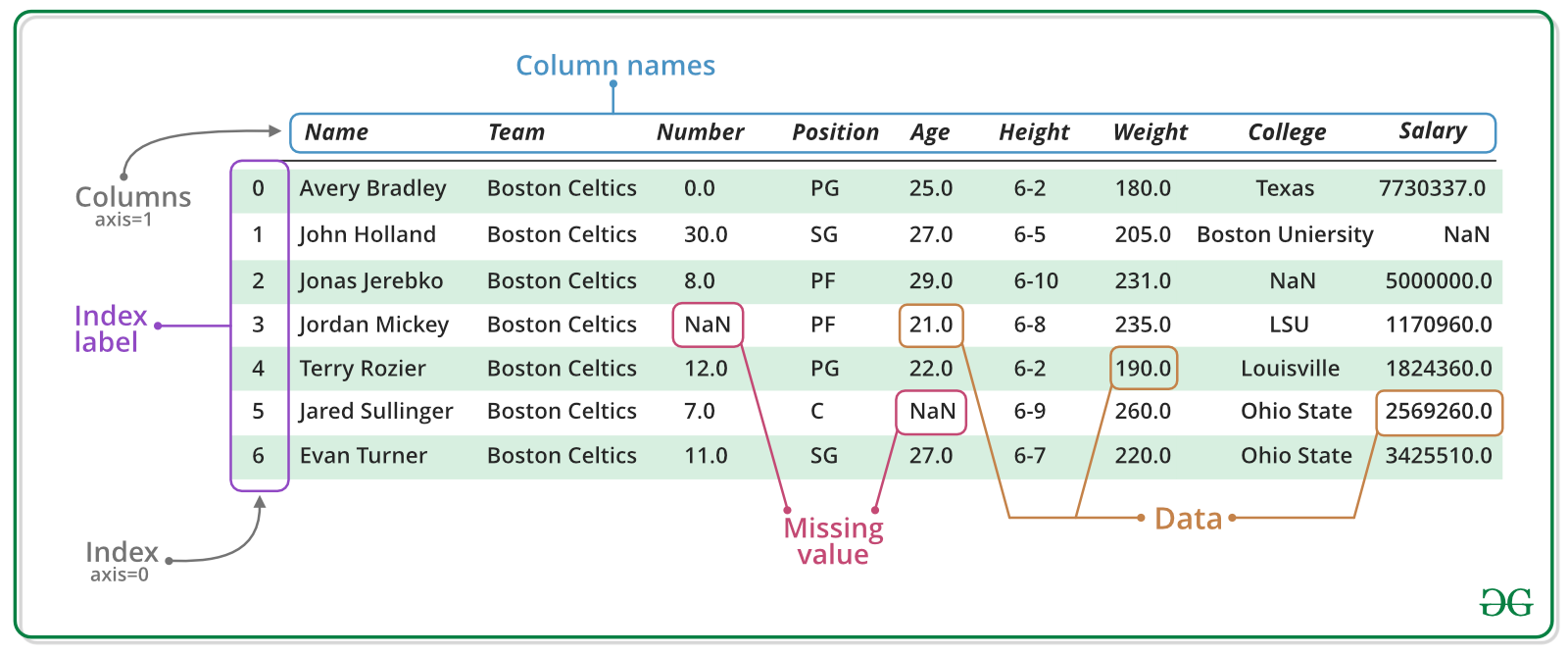
Pandas library’s objects are heavily dependent on *Numpy* library’s objects. This enables us to convert data to dataframe and vice versa.



***DataFrame:*** DataFrame is a 2D labeled data structure with columns of potentially different types. In a dataframe, data is aligned in a tabular fashion in rows and columns.

You can think of it like a spreadsheet or SQL table, or a dict of Series objects. It is generally the most commonly used pandas object.

We can perform basic operations on rows/columns like selecting, deteting, adding and renaming.

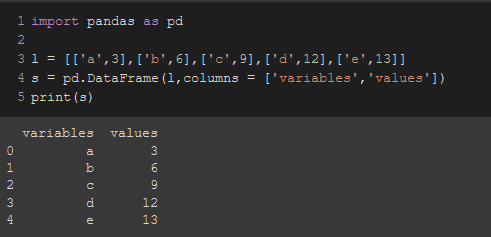


**Fig:** A pandas DataFrame

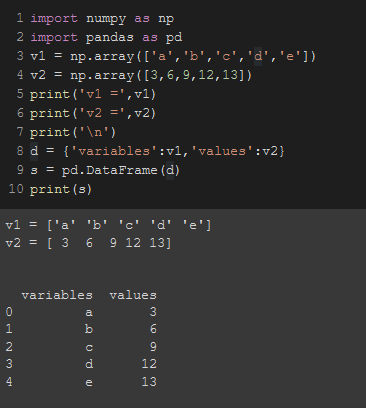
**Creating DataFrames**

We can create pandas dataframe from

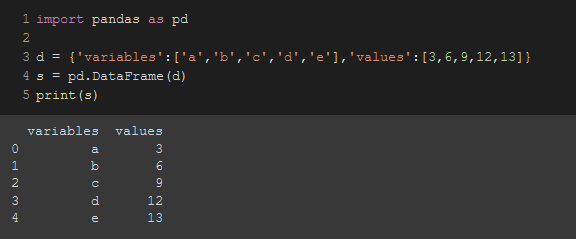
* a list of lists



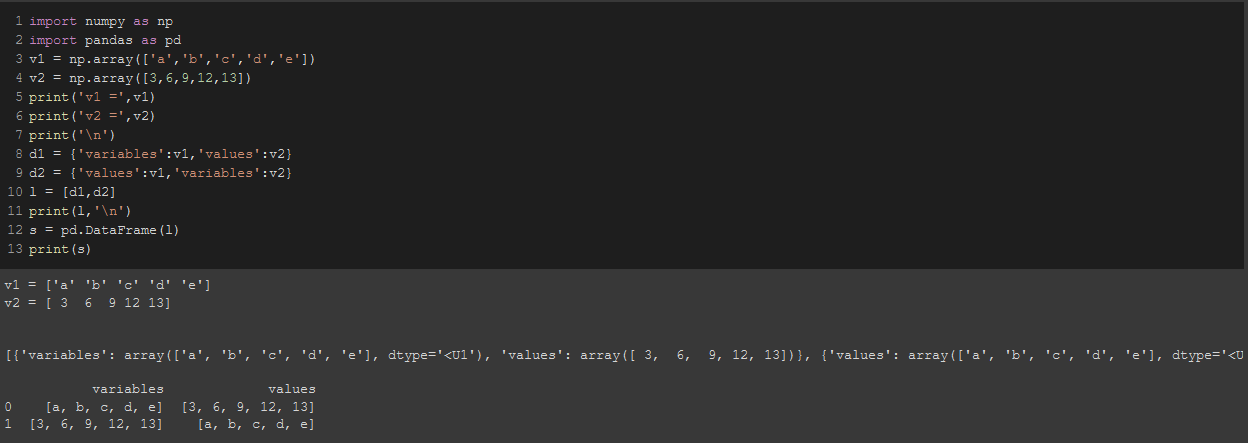
* dictionary of numpy arrays



* dictionary of lists

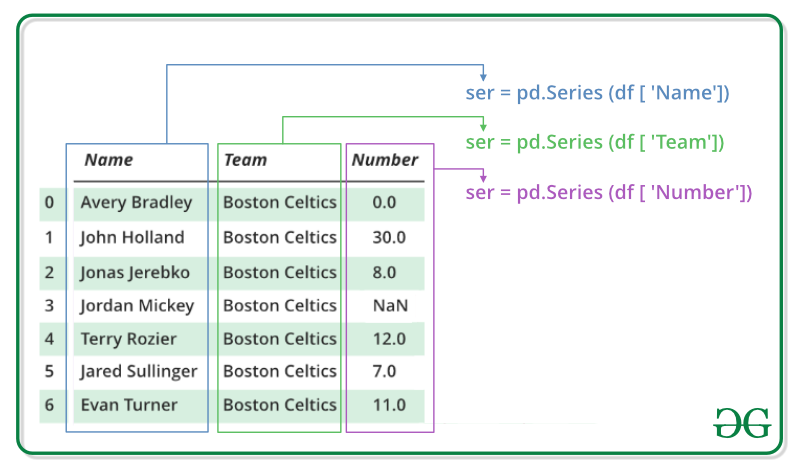


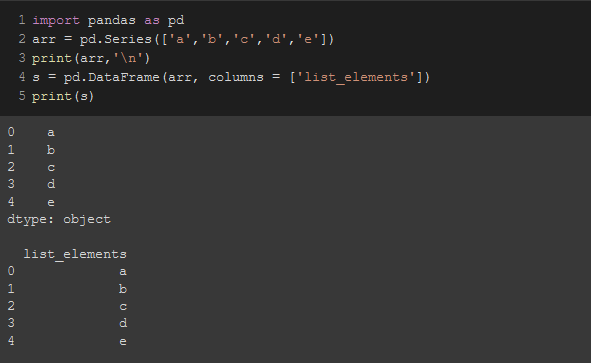
* list of dictionaries



* dictionary of pandas series

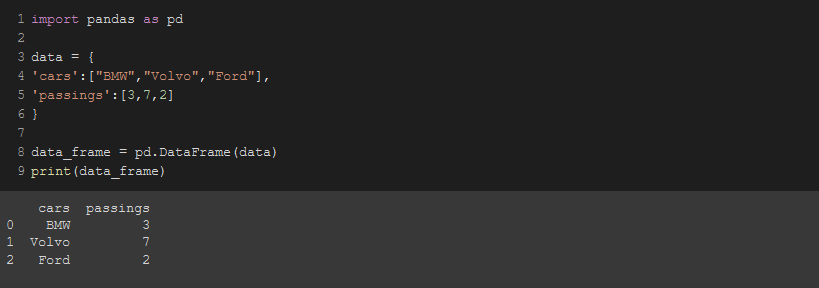
**Pandas Series:** Pandas Series is a one-dimensional labeled array capable of holding data of any type (integer, string, float, python objects, etc.)





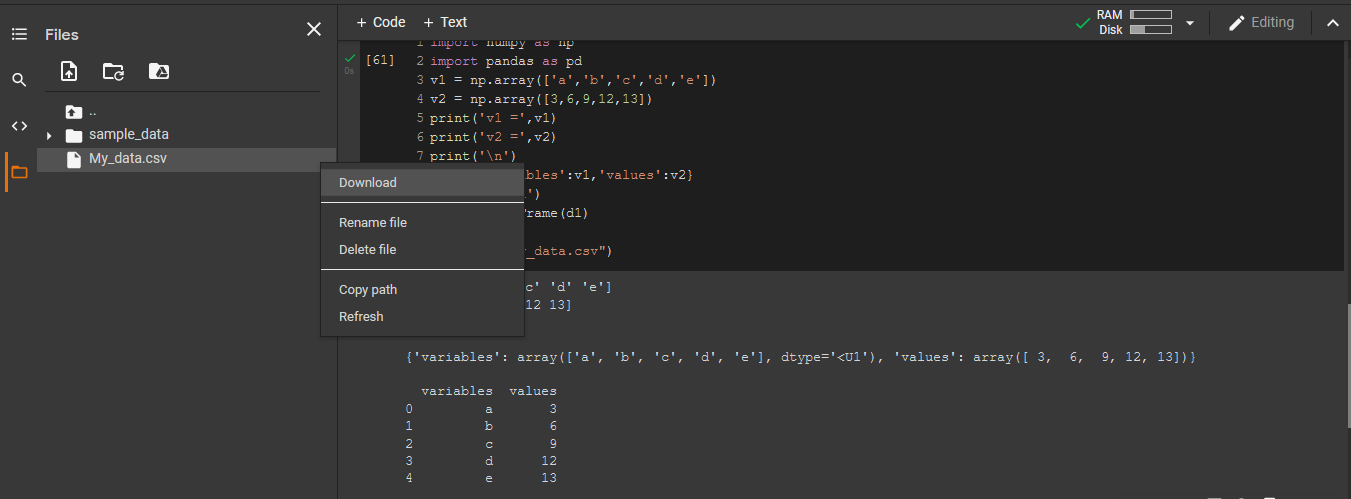
* We can also do this by importing external excel or csv files and assigning the data in a DataFrame.
* We can also import data from a remote cloud XML/JSON database.

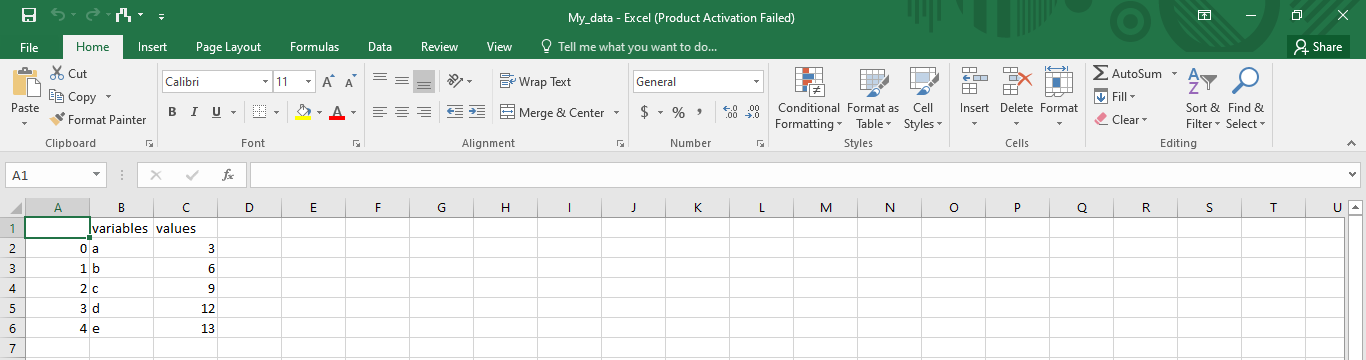
**Example:** Creating dataframe from a dictionary



**Saving datasets as external files**

***.to\_csv(“File\_name.extension”)*** method can be used to save a dataset as an external file.





**Importing Datasets**



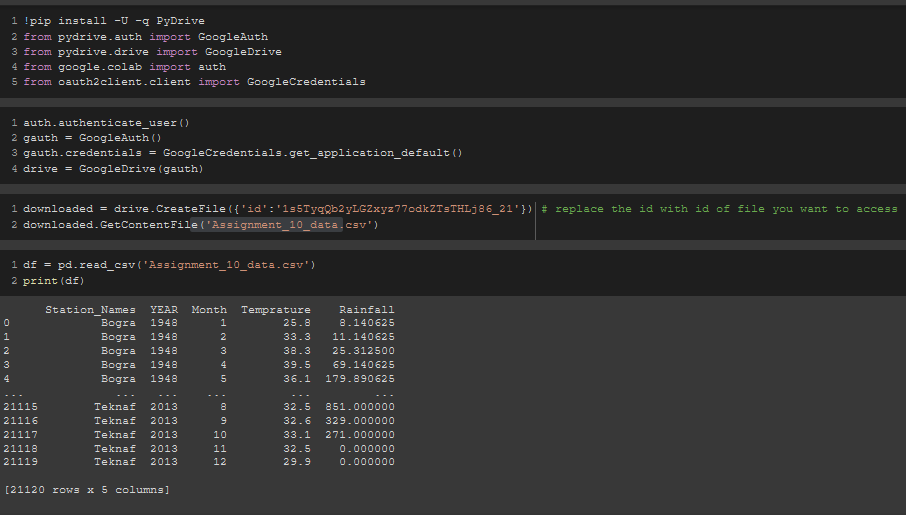
In the image above,

1. When we don’t have the excel/csv file in the jupyter notebook folder. In that case, We will have to specify the directory.
2. When the excel/csv file is in the same folder as jupyter notebook. In that case, We just have to mention the file name.

After importing, We need to read the file first before using it.

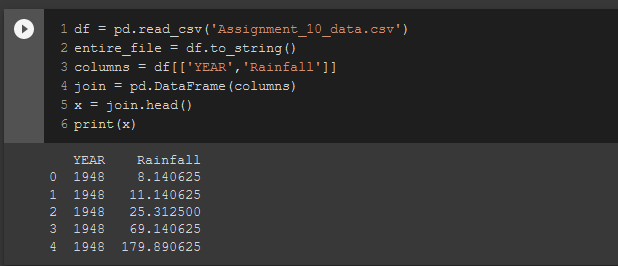
To read a file, we use ***pd.read\_extension(‘file name.extension’)***

I am using colab so I used this technique to solve it.



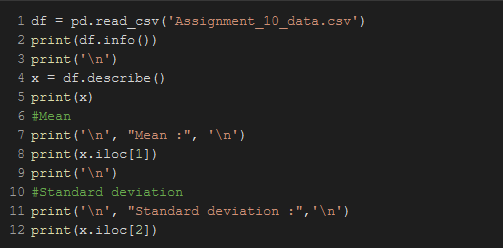
**Reading file & extracting data**

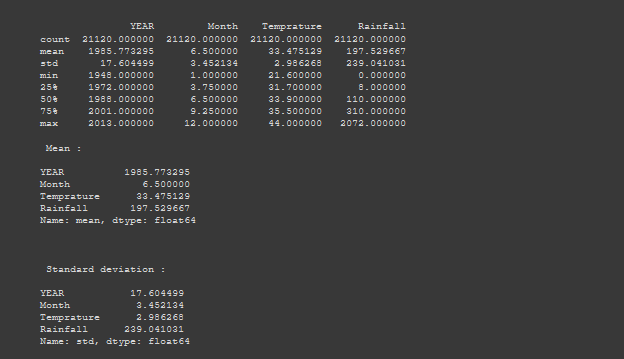
If we want to see selective columns, We just mention the column names as list elements in the ***read variable.***

****

If we want to see a single selective row, We use the row number in the ***.iloc(row\_number)*** method.

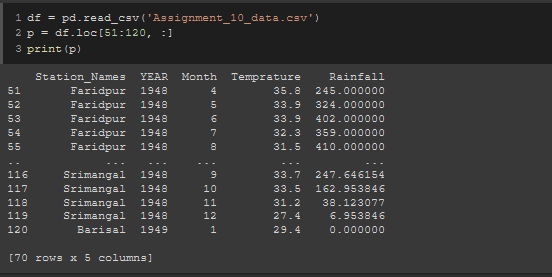
**Note:** dataframe index starts from 0





If we want to multiple selected rows, We use the row numbers in the ***.loc[ first\_row\_number:final\_row\_number , first\_column\_number:final\_column\_number]*** method.

**Note:** We use a blank colon ( “ **:** ” ) to specify all rows or all columns. For the code from screenshot below, It specified all columns.

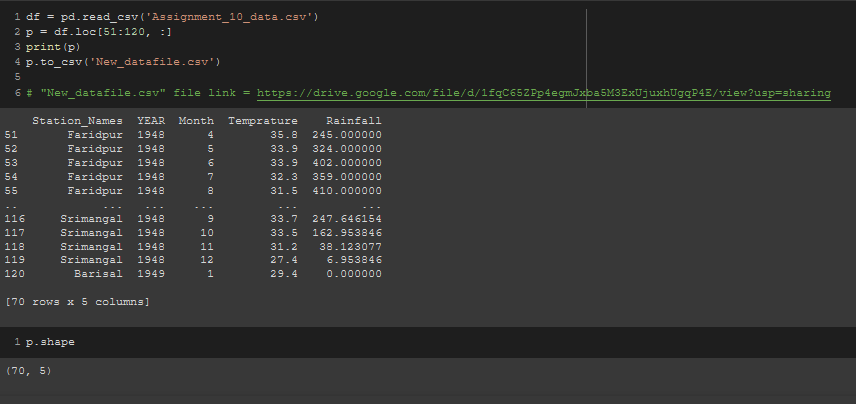


But the main difference between ***.loc()*** and ***.iloc()*** is,

.loc() is label-based, which means that you have to specify rows and columns based on their row and column labels.

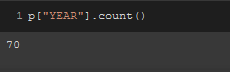
.iloc() is integer position-based, so you have to specify rows and columns by their integer position values (0-based integer position).

* ***.shape*** enables you to see how many columns and rows are there in a datarame



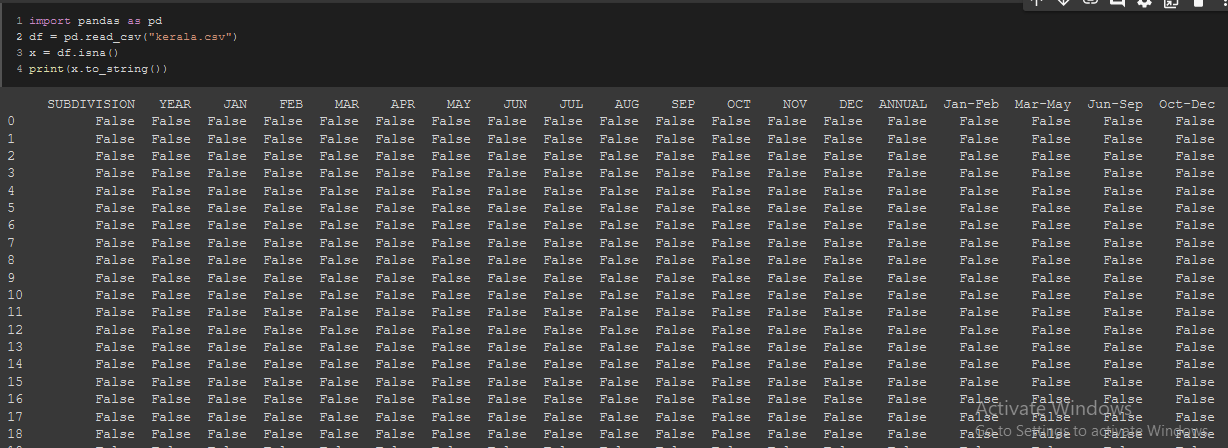
* ***df[“Column name”].count()*** gives the length of a column.

**Note:** It omits all the null values and gives the length except null values.



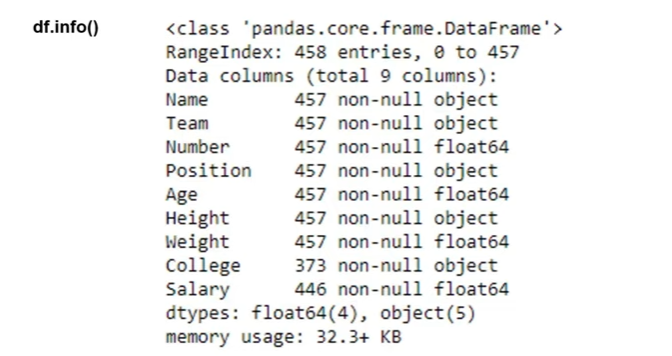
* ***df.isna()***

This method is **used to detect missing values**. It return a boolean same-sized object indicating if the values are NA. NA values, such as None or numpy. NaN, gets mapped to True values.



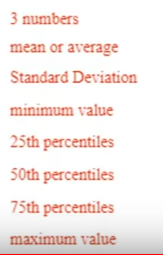
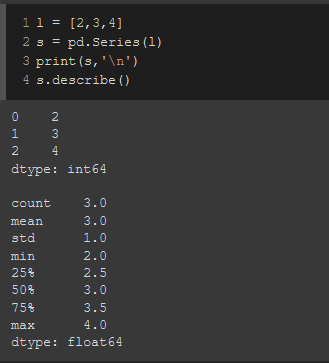
**Information of Datasets**

We can use the ***df.info()*** function to retrieve the information about a dataframe.



**Describing a DataSet**

To describe a dataset, We can use the ***.describe()*** method.

****

**Note:**

**Standard deviation:** a quantity expressed by how much the members of a group differ from the mean value for the group.

If we get a low standard deviation then it means that the values tend to be close to the mean whereas a high standard deviation tells us that the values are far from the mean value.

Here, The mean value is 3 and the standard deviation is 1 due to the other two members of the list differing only by 1 from the mean.

**Variance:** Variance measures variability from the average or mean. Variance is a measure of how spread out a data set is.

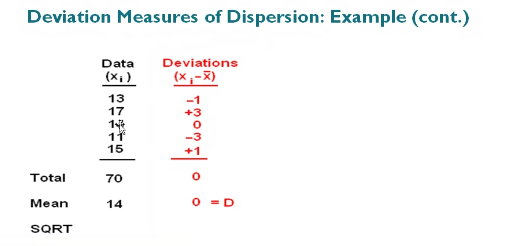
It is calculated by taking the differences between each number in the data set and the mean, then squaring the differences to make them positive, and finally dividing the sum of the squares by the number of values in the data set.

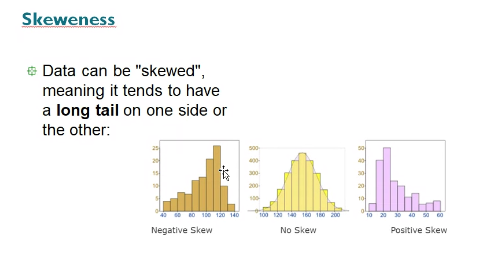
**Mean:** Mean is the synonym of average

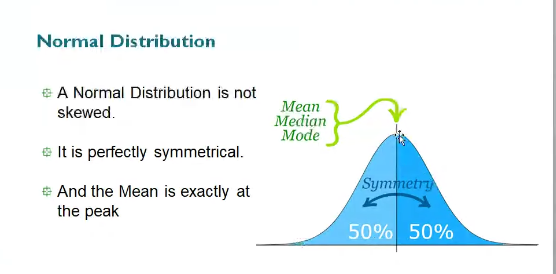
**Median:** The median is the middle number in a sorted, ascending or descending, list of numbers and can be more descriptive of that data set than the average. The median is sometimes used as opposed to the mean when there are outliers in the sequence that might skew the average of the values.

**Mode:** The mode is the value that appears most often in a set of data values.

**Deviation:** The difference of all data from the mean.







**Viewing a DataSet**

There are multiple methods to view the data row or column-wise from a dataframe.

Most commonly used 2 are,\

* .head()

.head() enables us to see the headers and top 5 rows from a dataset.

* .tail()

.tail() enables us to see the footers and bottom 5 rows from a dataset.

**Note:** If we specify a number in the ***.head()*** or ***.tail()*** method, It will return us that many rows. If no numbers are given, then it returns 5 rows from top or bottom according to the called method.

To print the entire pandas DataFrame, we can use these methods.

* .to\_string()
* .option\_context()
* .set\_options()
* .to\_markdown()

