**Data Wrangling in Python**

Data wrangling is a process of cleaning, structuring and enriching a raw data into a desired format for better decision making in less time.

Data wrangling is the process of removing errors and combining complex data sets to make them more accessible and easier to analyse.

It is a format for better understanding, decision-making, accessing, and analysis in less time. Data Wrangling is also known as Data Munging.

Data wrangling in Python deals with the below functionalities:

1. **Data exploration:**In this process, the data is studied, analysed, and understood by visualizing representations of data.
2. **Dealing with missing values:**Most of the datasets having a vast amount of data contain missing values of *NaN, they are needed to be taken*careof by replacing them with mean, mode, the most frequent value of the column, or simply by dropping the row having a *NaN*value.
3. **Reshaping data:**In this process, data is manipulated according to the requirements, where new data can be added or pre-existing data can be modified.
4. **Filtering data:**Sometimes datasets are comprised of unwanted rows or columns which are required to be removed or filtered
   1. **Data exploration in Python**

Here in Data exploration, we load the data into a dataframe, and then we visualize the data in a tabular format.

import pandas as pd

# Assign data

data = {]}

# Convert into DataFrame

df = pd.DataFrame(data)

# Display data

df

**Working with Missing Data in Pandas**

Missing Data can occur when no information is provided for one or more items or for a whole unit. Missing Data is a very big problem in a real-life scenario. Missing Data can also refer to as NA(Not Available) values in pandas. In Data Frame sometimes many datasets simply arrive with missing data, either because it exists and was not collected or it never existed.

In Pandas missing data is represented by two value:

* **None:** None is a Python singleton object that is often used for missing data in Python code.
* **NaN** : NaN (an acronym for Not a Number), is a special floating-point value recognized by all systems that use the standard IEEE floating-point representation

Pandas treat None and NaN as essentially interchangeable for indicating missing or null values. To facilitate this convention, there are several useful functions for detecting, removing, and replacing null values in Pandas DataFrame :

* isnull()
* notnull()
* dropna()
* fillna()

**Dropping missing values using dropna()**

To drop a null values from a dataframe, we used dropna() function this function drop Rows/Columns of datasets with Null values in different ways.

**df.dropna(axis = 1)** drop a columns which have missing values

**Case #1:** Dropping columns with at least 1 null value.

df.dropna(axis = 1)

Case #2: Dropping Rows with at least 1 null value in CSV file

# making new data frame with dropped NA values

new\_data = data.dropna(axis = 0, how ='any')

# importing pandas package

import pandas as pd

# making data frame from csv file

data = pd.read\_csv("employees.csv")

# filling a null values using fillna()

data["Gender"].fillna("No Gender", inplace = True)

data

**Replace all the null values**

import pandas as pd

# making data frame from csv file

data = pd.read\_csv("employees.csv")

# will replace Nan value in dataframe with value -99

data.replace(to\_replace = np.nan, value = -99)

**Removing Duplicate data from the Dataset**

import pandas as pd

# initializing Data

student\_data = {}

# creating dataframe

df = pd.DataFrame(student\_data)

# Here df.duplicated() list duplicate Entries in ROllno.

# So that ~(NOT) is placed in order to get non duplicate values.

non\_duplicate = df[df.duplicated('Roll\_no')]

# printing non-duplicate values

print(non\_duplicate)