# INDEX

# Unit I: Introduction to Data Wrangling

## Introduction:-

→ Introduction to Data Analysis, preparing data for analysis is often the large portion of the job time-wise for those working with data.

→ It should be noted that data wrangling isn't something we do merely once in our analysis

→ It is highly likely that we will do some data wrangling and move on to another analysis task. Such as data Visualization.

→ We need our data to be in for the visualization that best convey what we are looking to show, and the data points we should collect for our analysis.

→ This comes with experience, so we must practice the skills that are covered in this topic.

## Data Wrangling:

→ Data Wrangling is the process of removing errors and combining complex data sets to make them more accessible & easier to analyze.

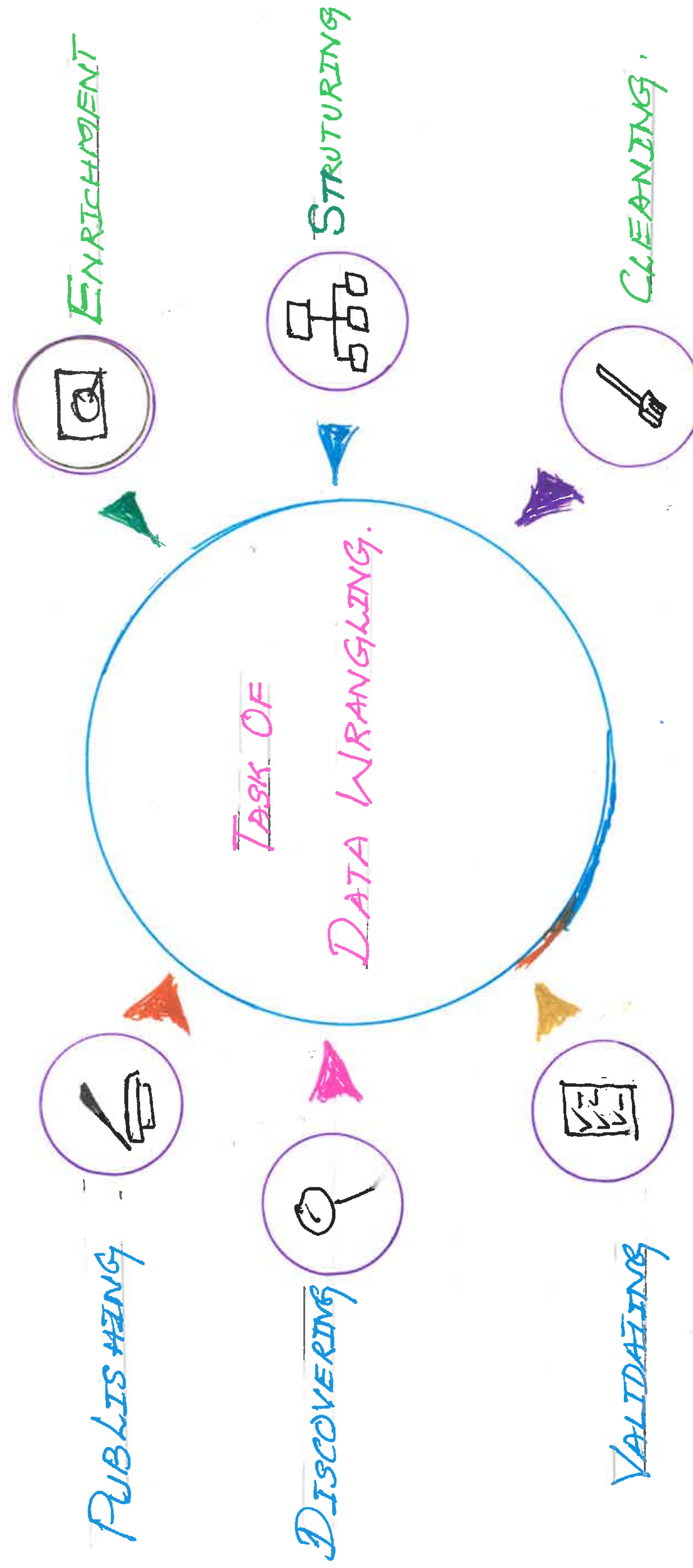


Fig:1: Tasks Of Data Wrangling.

→ Due to the rapid expansion of the amount of data & data sources available today, storing & organizing large quantities of data for analysis is becoming increasingly necessary.

→ A data wrangling process also known as a data munging process, consists of reorganizing, transforming and mapping data from one "raw" form into another in order to make it more usable & valuable for a variety of downstream uses including analytics.

## Importance of Data Wrangling.

→ Data professionals spend almost 80% of their time wrangling the data, leaving a mere 20% for exploration & modeling.

→ A simple analogy will help you understand. The foundation of a skyscraper is expensive & time consuming before the above ground structure starts.

→ Similarly, for data handling, once the code and infrastructure foundation are gathered, it will deliver immediate results [sometimes almost instantly] for as long as process is relevant.

→ However skipping necessary data wrangling steps will lead to significant downfalls, missed opportunities and erroneous models that damage the reputation of analysis within organization.

### Primary importance of data wrangling:

- Making raw data usable
- Getting all the data from various sources
- Piecing together raw data according to the required format.
- Automated data integration tools are used as data wrangling techniques.
- Cleaning the data from the noise or flawed, missing elements
- Data wrangling acts as a preparation stage of the data mining process.
- Helping business users make concrete, timely decisions.

### TASK of DATA WRANGLING:

- To prepare your data for analysis as part of data munging. there are 6 basic steps one need to follow,

They are

**1. Data Discovery:** This is an all encompassing term that describes understanding what your data is all about. You get familiar with your data.

**2. Data Structuring:** When you collect raw data, it initially is in all shapes & sizes, and has no definite structure. Such data need to be restructured to suit that your enterprise plans to deploy.

### 3. Data Cleaning:

- Raw data comes with some errors that need to be fixed before data is passed on to the next stage.
- Cleaning involves the tackling of outliers, making corrections or deleting bad data completely.

### 4. Data Enriching:

By this stage, you have kind of become familiar with the data in hand.

Enriching the data is an optional step that you only need to take if your current data doesnot meet your requirements.

### 5. Data Validating:

Validating the data is an activity that services any issues in the quality of your data so they can be

- addressed with the appropriate transformations.

### Rules of Data Validation:

- Quality
- Consistency
- Accuracy
- Security
- Authenticity

### 6. Data Publishing:

→ By this time, all the steps are completed and the data is ready for analytics. All that's left is publish the newly wrangled data in a place where it can be easily accessed and used by you & other stakeholders.

→ The final output of your efforts will be high quality of data that you can use to gain insights.

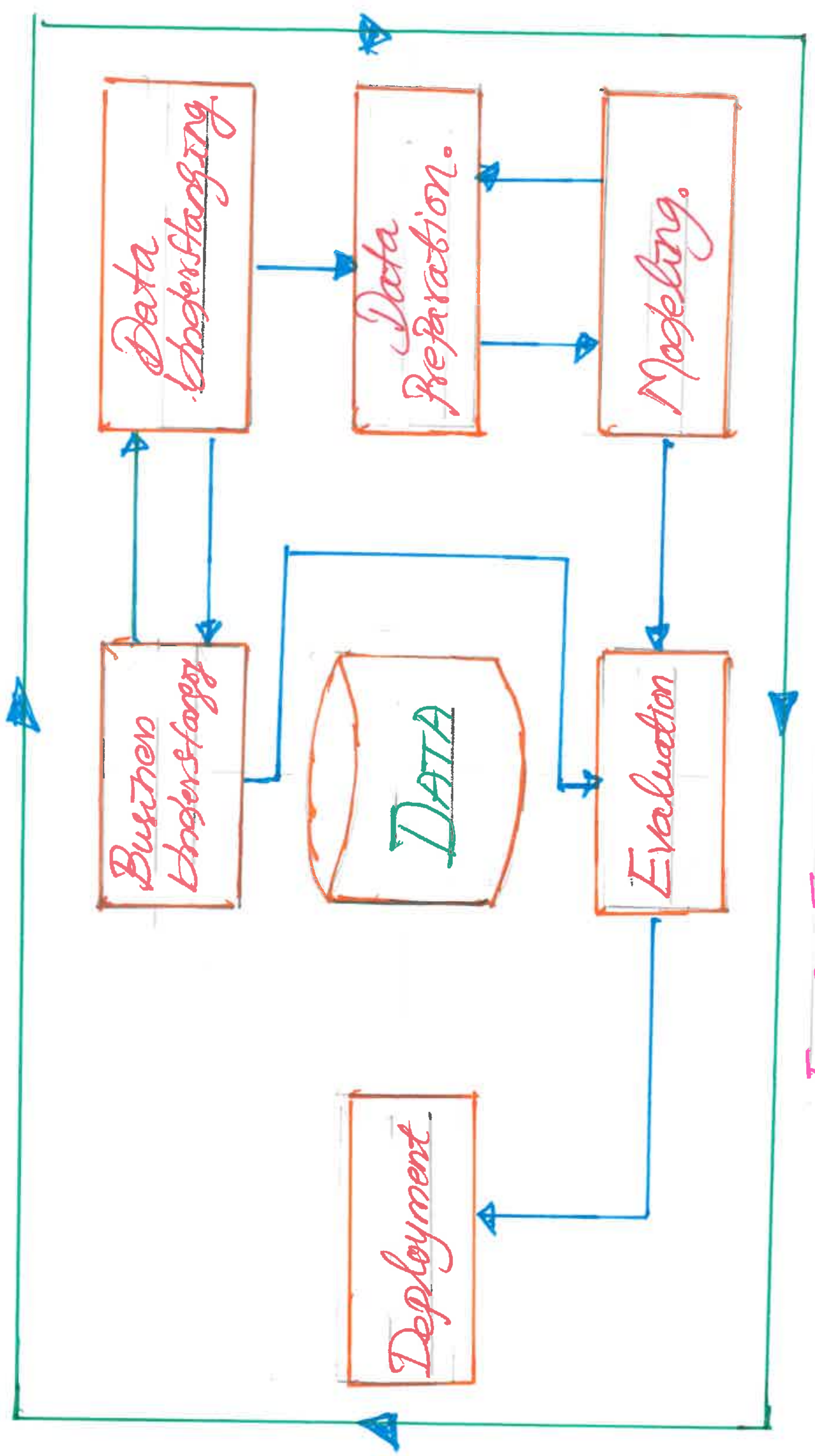


Fig-2: IMPORTANCE OF DATA WRANGLING

### Data Wrangling Tools:

1. Parsehub.
2. Scrapy.
3. Talend
4. Alteryx APA Platform.
5. Altair Monarch.
6. Microsoft Power Query.
7. Tableau Desktop.

### Introduction to Python:

- Python is a programming language that let you work quickly and integrate system more efficiently.
- There are two major Python versions Python 2 and Python 3. Both are quite different.
- It was designed with an emphasis on code readability. This programming language were quickly & Integrate System more efficiently

Windows: There are many interpreters alavible freely to run Python Scripts like IDLE that Comes with Python Software downloaded. from http://Python.org/

Linux: Python Comes Preinstalled with popular Linux Such as Ubuntu and Fedora.

Mac OS: Python 2.7 Comes bundled with Mac OS & Manually install Python 3 from http://Python.org/

Note: Python doesnot require a Semicolon at the end of each Statement. Comment begin with a '#' and extend to the end of the line.

Python Source Code:

Source file use the ".py" extensions and are Called "Modules" with module hello.py earliest way to run the Shell Command.

## Data Meant to be Read: By Machines.

Data Can be Stored in Many ways of format & types.. Some formats Store data in a way easily handled by Machines. While others Store data in a way easily readable by a human.

Microsoft Word document are an example of the latter. While CSV, JSON 4 XML are examples of the former.

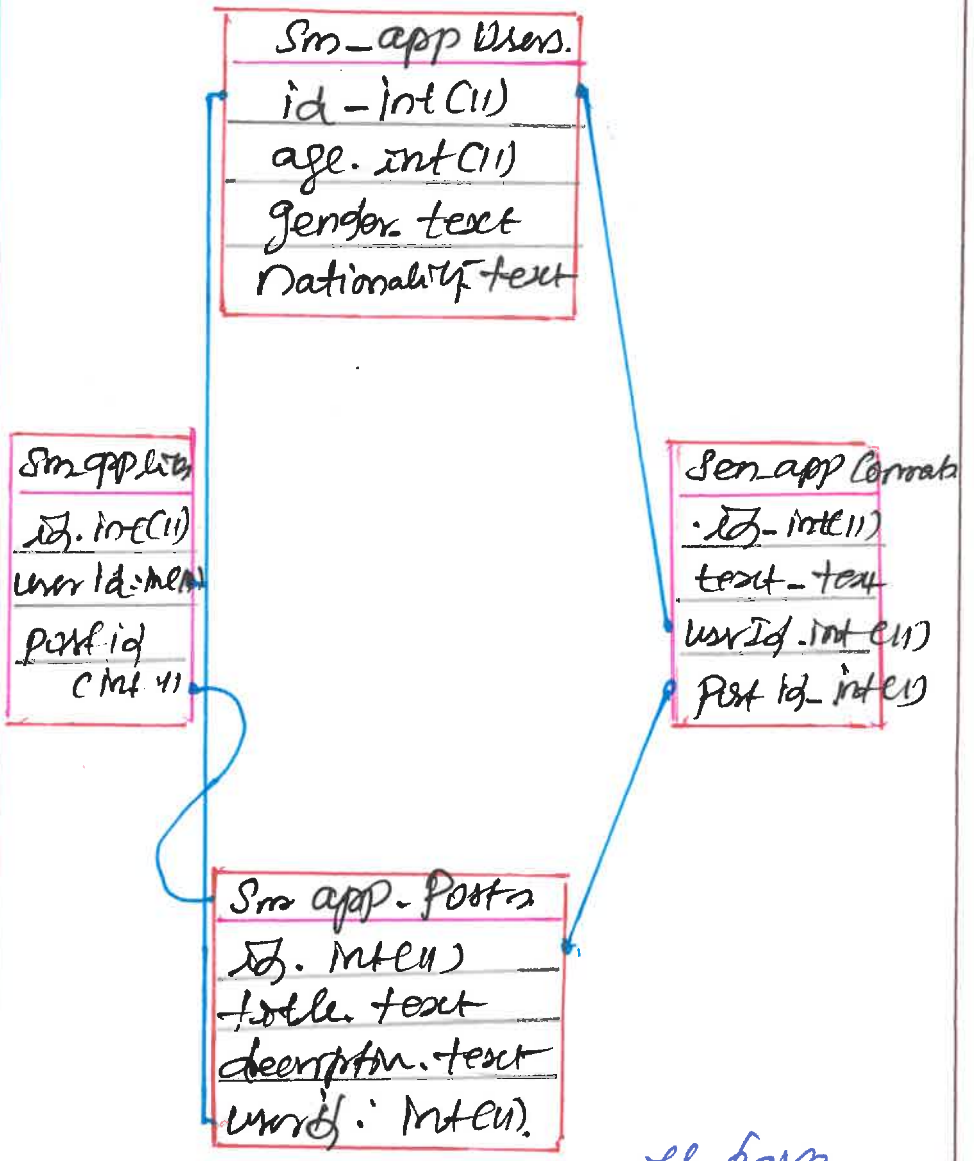
- Comma-Separate Values [CSV]
- Java Script Object Notation [JSON]
- Extensible Markup Language. (XML)

## Understanding the Database Schema

→ In this topic a Very Small database for a Social Media application. The data Consist of four table:

1. Users 2. Posts 3. Command 4. likes.

A high level diagram of the database Schema is shown below



→ Both Users & Posts will have One-to many relationship. IIly One Users Can post many Comments.

→ One post Can post also have multiple Comments.

# UNIT II

## INTRODUCTION TO PYTHON SQL LIBRARIES

→ All Software application interact with data, most Common through a database Management System [DBMS]

→ In this unit you will explore the different Python SQL Libraries that You Can use & Develop a application to interact with SQLite, MySQL & PostgreSQL database.

- Connect to different DBMS with Python SQL Libraries.
- Interact with SQLite, MySQL & PostgreSQL
- Perform common database queries using Python application
- Develop application across different database using a Python Script.

## SQLite:

SQlite database are Serverless and Self-Contained, Since they Read & write data file. This means that, unlike with mySQL & Postgre' SQL, You don't even need to install.

Here how SQLite3 to Connect to an SQLite database. in Python.

Python

```
1   import Sqlite3
2   from Sqlite3 import Error
3
4   def Create_Connection(Path):
5       Connection = None
6       try:
7           Connection = sqlite3.Connect(Path)
8           Print("Connection to SQLite DB
                                Successful")
9       except Error as e:
10          print(f "The error "{e}' occured)
11
12      return Connection
```

- Line 1 & 2 import sqlite3 & The Module error Class
- Line 4 define function. Create_Connection()
- Line 7 uses. Connect() from the sqlite3 module & take the SQLite database path a parameter
- Line 8 Print the Status of the Successful Connection.
- Line 9 Catches any exception that might be thrown if .Connect() fails to establish a Connection
- Line 10 displays the error message in the Console.
- The following Script Create a Connection to the SQLite DB.

```
Connection = Create_Connection("E:\\Sm_
                         app. sqlite")
```

## MySQL:

→ Unlike SQLite, there's no default Python SQL module that You Can use to Connect to a MySQL DB.

→ Instead, need to install a Python SQL driver for MySQL in Order to interact with a MySQL

→ Download this Python SQL module with PiP

```
$ PiP install mysql-Connector-Python
```

MySQL database has a two Step Proces.

1. Make a Connection to a MySQL Server
2. Execute a Seperate query to Create DB.

a function that Connects to MySQL database Server. & Return the Connection object.

Python

```
import mysql.Connector
from mysql. Connector import Error

def Create_Connection (host_name, user name,
                              user_Password):
    Connection = None
    try:
        Connection = mysqL. Connector. Connect(
            host = host_name,
            user = user_name,
            Passwd = user_Password
        )
        Print("Connection to MySQL DB
                          Successful")
    except Error as e:
        Print(f" The error '{e}' occured)

    return Connection

Connection = Create_Connection ("Localhost"
                          " root", " ")
```

From the above Script, function Create_Connection() that accept three parameter

1. host_name
2. User_name
3. User_Password

## PostgreSQL:

Like MySQL, there is nodefault Python SQL library that you Can use to interact with a postgreSQL DB. Instead need to install a 3rd Party Python SQL driver to interact with PostgreSQL.

One such Python SQL driver for PostgreSQL is Psycopg2.

Execute the following Command on your terminal to install the Psycopg2 Python SQL module

```
$. Pip install Psycopg2
```

Make a Connection with your PostgreSQL database

Python

```
import Psycopg2
from Psycopg2 import OperationalError

def Create_Connection (db_name, db_user,
        db_host, db_Port):
    Connection = None
    try:
        Connection = Psycopg2 .Connect(
            database = db_name,
            user = db_user
            Password = db_Password,
            host = db_host,
            Port = db_Port,
        )
        Print("Connection to PostgreSQL
                DB Successful)
    except OperationalError as e:
        Print (f "The error '{e}' occured")
    return Connection
```

## Creating Tables:

In this Section, how to create table inside there three databases. & Four table are

1. users
2. Posts
3. Comments
4. likes.

## SQLite:

- To execute query in SQLite, we Cursor. execute().
- To define a function execute_query()
- execute() Can execute any query passed to it

**Note:** This Script should be executed in the Same file where you Created the Connection for your SQLite DB.

Python

```
def execute_query(Connection, query):
    Cursor = Connection. Cursor()
    try:
        Cursor. execute(query)
        Connection. Commit()
        Print("Query execute Successfully")
    except Error as e:
        Print (f "The error "{e}" occured)
```

This Code execute the given query and Prints an error message

Python

```
Create_users_table = """
CREATE TABLE IF NOT EXITS users (
id INTEGER PRIMARY KEY
AUTO_INCREMENT, name TEXT NOT NULL,
age INTEGER, gender TEXT,
nationality TEXT
);
""",,
```

above Code Says to Create a table users with following five Columns:

1. id 2. name 3. age
4. gender 5. nationality.

## MySQL:

Create tables in MySQL.

Just like with SQLite, need to pass your query to Cursor. execute(), which is returned by Calling .cursor() on the Connection object.

Python

```
1 def execute_ query(Connection, query):
2       Curser = Connection. query()
3       try:
4           Cusser. execute(query)
5           Connection. Commit()
6           Print ("Query execute Successful")
7       except Error as e:
8           Print (f "The Error '{e}" occured)
```

In line 4. Pass query to Cursor. execute().

Python

```
Create_users_table = """
CREATE TABLE IF NOT EXITS users (
    id INT AUTO_INCREMENT,
    name TEXT NOT NULL,
    age INT, gender TEXT,
   nationality TEXT, PRIMARY KEY (id).
) ENGINE = InnoDB
"""
```

```
execute_query(Connection, Create_user_table)
```

MySQL uses the AUTO_INGREMET keyword to Create Column, the values are automatically incremented.

## PostgreSQL:

→ Like with SQLite & MySQL database, the Connection object returned by Psycopg2. Connect() cursor object.

→ To execute Python SQL queries on your PostgreSQL DB.

Define a function execute_query():

Python

```
def execute_query(Connection, query):
    Connection. autoCommit = True
    Cursor = Connection. Cursor()
    try:
        Cursor. execute(query)
        Print ("Query Executed Successfully")
    except Operational Error as e;
        Print(f" The error "{e}" occured)
```

Python

```
Create_user_table = """
CREATE TABLE IF NOT EXITS user (
id SERIAL PRIMARY KEY
name TEXT NOT NULL, age INTEGER,
gender TEXT, nationality TEXT
) """
execute_query(Connection, Create_user_table)
```

users Table PostgreSQL is slightly different than SQLite & MySQL. The keyword SERIAL is used to Create Column that increment automatically.

## Updating Table Records:

In this Section Covers the Process for updating records using Python SQL libraries ~~for~~ SQLite, PostgreSQL, & MySQL

### SQLite:

As an example, You can update the description of the post with an id of 2

Python

```
Select_Post_description = "SELECT description
                           From post WHERE id=2"
Post_description = execute_read_query(Connection,
                   Select_post_description)
for description in post_description:
    Print(description)
```

Python

```
update_post_description = """
  Update
    posts
  SET
    description = "The Weather has become
                   Pleasant now"
  WHERE
    id = 2  """
```

```
execute_query(Connection, update_post_description)
```

execute the query again, & see the following Result

Shell

```
("The Weather has become Pleasant now",)
```

### MySQL:

The Process of updating records in MySQL with mysql-Connector-Python is also a Carbon of the sqlite3 Python SQL Module.

For example, the following Script updates the description of the Post with an id 2

Python

```
update_post_description = """
 UPDATE
   posts
 SET
   description = "The Weather has Become
                  Pleasant now"
 WHERE
   id = 2
"""
execute_query(Connection, update_post_description)
```

### PostgreSQL:

The update query for PostgreSQL is Similar to what you have seen with SQLite and MySQL. Use above Scripts to update records in your PostgreSQL table.

## Deleting Table Records:

To delete the table records using Python SQL modules for SQLite, MySQL & PostgreSQL. The process of deleting records is uniform for all three database

### SQLite:

To delete records from your SQLite database, execute_query() will Create a Cursor object using the Connection & pass the String to Cursor.execute(), which will delete the records.

Python

```
delete_Comment = "DELETE FROM Comments
                  WHERE id=5"
execute_query(Connection, delete_Comment)
```

### MySQL:

The Process of deletion in MySQL is also Similar to SQLite.

Ex:

Python

```
delete_Comment = "DELETE FROM Comments
                  WHERE id=2"
execute_query(Connection, delete_Comment)
```

### PostgreSQL:

The delete query for PostgreSQL is also Similar to SQLite & MySQL. Write a delete query String by using the DELETE Keyword and then passing the query & the Connection object to execute_query(). This will delete the specified records from your PostgreSQL DB.

## Conclusion:

Three Common Python SQL libraries, Sqlite3, mysql-Connector-Python & Psycopg2 allow you to Connect a Python application to SQLite, MySQL & PostgreSQL

- Interact
- Use
- Execute

## SQLite

### Supported Data Types

- null
- integer
- real
- text
- blob

## MySQL

### Supported Data Types

- tinyint
- smallint
- mediumint
- int (or) integer
- bigint
- float
- dec, decimal, fixed (or) numeric
- bool or boolean
- bit

### Date and Time Types

- date
- datetime
- timestamp
- time
- year

### String Types

- char
- varchar
- binary
- varbinary
- blob
- tiny blob
- mediumblob
- long blob
- text
- tiny text
- medium text
- long text
- enum
- set

## PostgreSQL

### Numeric Types

- bigint
- bigserial
- double precision
- integer
- numeric (or) decimal
- real
- smallint
- small serial
- serial

### Character Types

- character
- character varying (or) varchar
- text

### Date and Time Types

- date
- interval
- time (or) time without time zone
- time with time zone
- timestamp (or) timestamp without time zone
- timestamp with time zone

### Geometric Types

- box
- circle
- line
- lseg
- path
- point
- polygon

### Network address Types :-

- cidr
- inet
- macaddr

### Bit String Types:-

- bit
- bit varying

### Text Search Types:-

- tsquery
- tsvector

### JSON Types :-

- json
- jsonb

### Other Data Types:-

- boolean
- bytea
- money
- pg_lsn
- uuid
- xml
- txid_snapshot

↳ A user level transaction ID Snapshot

# Unit III Data Cleanup

## Why Clean Data?

↳ It is the process of fixing or removing incorrect, Corrupted, incorrectly formatted duplicate or incomplete data within a dataset.

## Basics

1. Remove duplicate or irrelevant observations
2. Fix Structural errors
3. Filter unwanted outliers
4. Handle missing data
5. Validate and QA

- Substitute missing data, such as N/A for null categories or 0 for numerical values.
- Substitute average numbers for the missing numerical values.

## Identifying Values for Data Clean up.

- Remove duplicates
- Remove irrelevant data
- Standardize capitalization
- Convert data type
- Clear formatting
- Fix errors
- Language Translation
- Handle missing values

## Data Formatting

↳ Converting data to the required format

↳ When merging data from several sources, check all the variables inside a particular attribute are written Consistently.

↳ Formats for dates, money (4.03, $4.03, or even $4.03), addresses, and so on.

↳ Across the dataset, the input format should be Constant.

## Finding outliers and Bad data

1. Sort your data
2. Graph your data
3. Calculate the z-score

$$Z = (x - \mu) \div \sigma$$

where

$x$ = raw measurement

$\mu$ = the mean

$\sigma$ = the standard deviation

4. Calculate the interquartile range

$$IQR = Q3 - Q1$$

$Q_3$ = the third quartile = the median of the upper half of the dataset.

$Q_1$ = the first quartile = the median of the lower half of the dataset.

High outlier $\geq Q_3 + (1.5 \times IQR)$

Low outlier $\leq Q_1 - (1.5 \times IQR)$

5. Use a hypothesis test.

## Finding Duplicates :-

↳ The process of duplicate detection is preceded by a data preparation stage, during which data entries are stored in an uniform manner in the database.

## Fuzzy Matching

↳ identifies similar, but not identical elements in data table sets.

## Levenshtein distance :-

1. Power → Powder (insertion of "w") – Distance :- 1
2. Lovin → Loving (insertion of "g") – Distance :- 1
3. Porpoise → Purpose (insertion of "i", substitution "o" for "u") – Distance :- 2

## Hamming Distance :-

Eg:- "Corn" and "Cork"

The binary ASCII character table assigns the Code (01101110) for n and (01101011) for k, the difference between each letter's code occurs in two locations, thus an HD of 2.

## Damerau – Levenshtein

String 1 : Micheal

String 2 : Michaela

Operation 1 :- Transposition : swap characters "a" and "e"

Operation 2 :- insert "a" (end of string 2)

Distance = 2

Each operation has a Count of "1", so each insertion, deletion, transposition, etc is weighted equally.

## Regular Expressions (Regex)

↳ is extremely and amazingly powerful in searching and manipulating text strings, particularly in processing text files.

- * character
- * Special Regex characters
- * Escape Sequences
- * A sequence of characters
- * OR operator (|)
- * character class
- * Occurrence Indicators (or Repetition operators)
- * Metacharacters
- * Position Anchors
- * Parenthesized Back References
- * Laziness (Curb Greediness for Repetition operators) :

*?, +?, ??, {m,n}?, {m,}?

## Normalization and Standardization

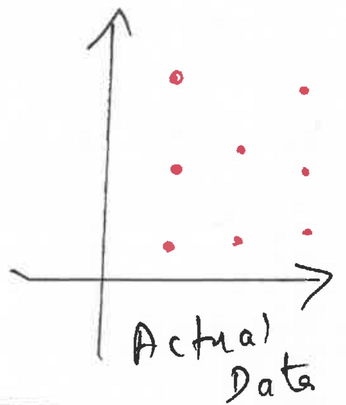
↳ is a part of data processing and cleansing techniques.

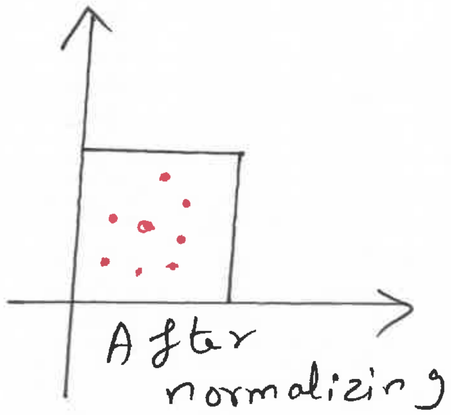
↳ is to make the data homogenous overall records and fields.

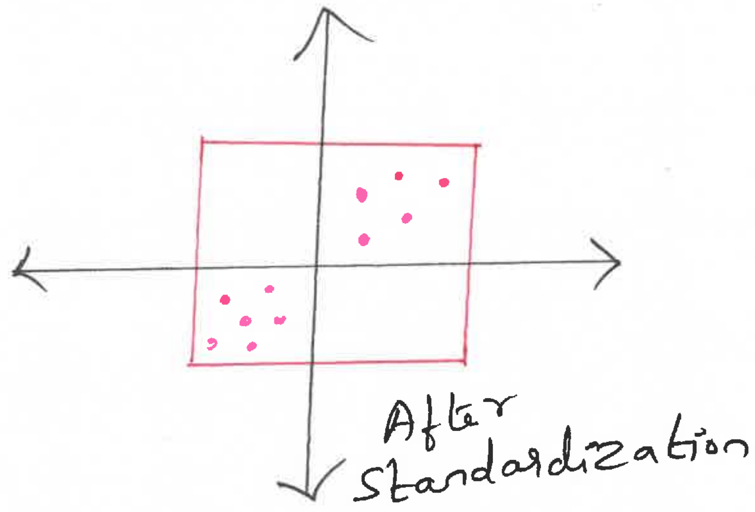
↳ helps in creating a linkage between the entry data which in turn helps in cleaning and improving data quality.

Standardization is the process of placing dissimilar features on the same scale.

↳ Rescaling the attributes in such a way that their mean is 0 and standard deviation becomes 1.







## Saving Data :-

Pickle module, which is a part of the standard library to save data in Python.

```
import pickle
Class Myclass():
    def __init__(self, param):
        self.param = param
def save_object(obj):
    try:
        with open("data.pickle", "wb") as f:
            pickle.dump(obj, f, protocol = pickle.HIGHEST_PROTOCOL)
    except Exception as ex:
        print(" Error during pickling object (Possibly unsupported):", ex)
obj = MyClass(10)
save_object(obj)
```

If you run this script, a file called data.pickle, which contains the saved data.

## Determining Suitable Data cleanup

1. Look into your data
2. Look at the proportion of missing data.
3. Check the datatype of each column

4. If Columns of strings are available, check for trailing whitespaces.

5. Dealing with Missing values (NaN values)

6. Extracting more information from your dataset to get more variables.

7. check the unique values of Columns.

## Scripting the cleanup and Testing with New Data.

The following script is both a testing and cleaning script to test for all lowercase letters, and to lower case them if they are not already lower case.

```
aimReturnStatus = True;
aimReturn Value = aimValue;
if (aimValue ! = null)
{
lower = aimValue.toLowerCase();
if (! aimValue. equals (lower))
   {
aimReturnStatus = false;
aimReturnValue = lower;
   }
}
```

When applied as a test script, the following script flags longer than 16 characters as failed. When applied as a cleaning script, it truncates the string and replaces the last 3 characters with "...".

```
limit = 16;
aimReturnStatus = true;
aimReturnValue = aimValue;
if (aimValue! =null)
  {
len = aimValue.length;
if (len > limit)
  {
aimReturnStatus=false;
aimReturnValue = aimValue.substring (0, limit -3)
    + "...";
  }
}
```

Data cleaning is the process by which you correct errors and anomalies in the case data.

↳ Save your time from having to do a code review and make corrections and also from having to enter a commit message each time

# UNIT IV

## Exploring Data:-

↳ is the first step of data analysis.

↳ used to explore and visualize data.

## Importing Data

- Delimited text files (.txt)
- Comma Separated values text files (.csv)

## Exploring Table Functions

- Head
- Info
- Describe
- Shape
- Columns & Index
- astype

## Data Exploration and Analysis.

- Sort Values
- Value_ Counts
- drop_ duplicates
- Subsetting
- Adding a new Column
- Set Index and Reset index
- loc and iloc
- groupBy
- Pivot Table

## Joining numerous datasets:-

The three best and most time saving ways to Combine multiple datasets using python pandas methods

- merge() - To Combine the datasets on Common Column or index or both.
- Concat() - To Combine the datasets across rows or columns
- join() - To Combine the datasets on key Column or index

## Identifying Correlations

The statistics. Correlation() method in python is used to return Pearson's Correlation Coefficient between two inputs.

$$r_{xy} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

where

$r_{xy}$ = the Correlation coefficient of the linear relationship between the variables $x$ and $y$.

$x_i$ = the values of the $x$-variable in a sample

$\bar{x}$ = the mean of the values of the $x$-variable.

$y_i$ = the values of the $y$-variable in a sample.

$\bar{y}$ = the mean of the values of the $y$-variable

## Methods to detect outliers in python.

- * Z-Score
- * Interquartile Range (IQR)
- * Tukey's fences.

## Creating Groupings

The groupby() function in pandas is used to group DataFrame rows based on one or more Columns.

- * Import the required libraries
- * Create a sample dataframe
- * Group the dataframe based on a Condition
- * Group the dataframe based on multiple Conditions

## Data Analysis

↳ is the technique of Collecting, transforming, and organizing data to make future predictions and informed data-driven decisions.

- · Specify Data Requirement
- · Prepare or Collect Data
- * clean and Process
- * Analyze
- * Share
- * Report.
- *

## Dataset Splitting:

sklearn is the most useful and robust library for machine learning in python.

Parameters:-

- * arrays
- * test_size
- * train_size
- * random_state
- * shuffle
- * stratify

Arrays:- inputs such as list, arrays, data frames or matrices

test_size:- is a float value whose value ranges between 0.0 and 1.0

train_size:- is a float value whose value ranges between 0.0 and 1.0

random_state:- Controls the shuffling applied to the data before applying the split

Shuffle:- is used to shuffle the data before splitting.

Stratify:- split the data is a stratified fashion

## Data Visualization in Python

- Line plots
- Box plots
- Heatmaps
- Lag Plots
- AutoCorrelation plots.

Analytics Software is software with a source code that anyone can inspect, modify or enhance.

## Python libraries for Data Visualisation:-

- Matplotlib
- Plotly
- Seaborn
- GGplot
- Altair
- Bokeh
- Pygal
- Geoplotlib
- Folium
- Gleam

## Publishing the Data-open Source Platform.

The steps to publish a python package are follows:-

- Write your python module and save it under a directory
- Create the setup.py file with necessary information
- Choose a LICENSE and provide a README file for your project
- Generate the distribution archives on local machine
- Try installing the package on your local machine.
- Publish the package to the TestPyPi repository to check if all works well.
- Finally, publish the package to the PyPi repository.

## Open-Source Platform

- Seal Report
- KNIME
- Knowage
- Metabase
- Microsoft Power BI Desktop
- Rapid Miner
- Report Server
- Tableau
- Web Data Rocks
- Zoho Analytics
- Dataiku DSS Free Edition

# UNIT V VISUALIZING DATA WITH PANDAS AND MATPLOTLIB

## An Introduction to Matplotlib

↳ is a low level graph plotting library in python that serves as a visualization utility.

↳ Created by John D. Hunter

↳ is an open source and we can use it freely.

↳ mostly written in python, few segments are in C, and Javascript for platform compatibility

### Basics :-

↳ used to create 2D graphs and plots by using python scripts.

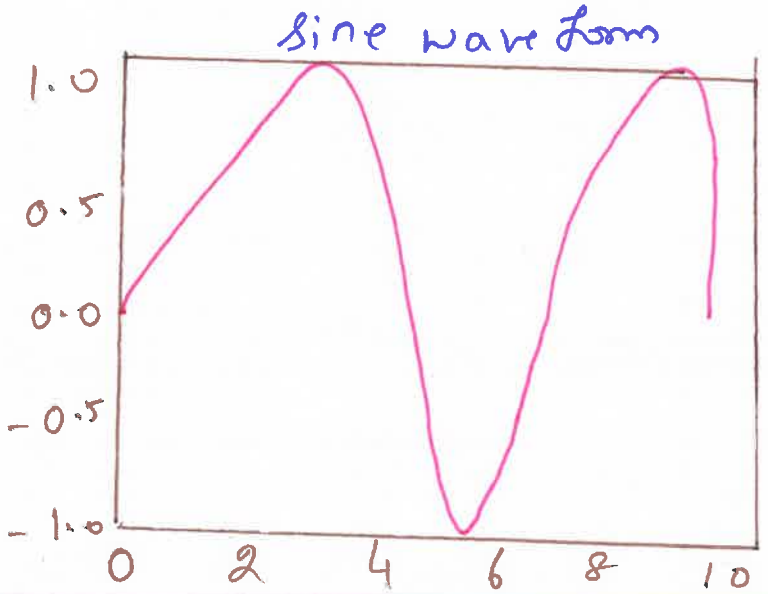
↳ uses pyplot which provides feature to control line styles, font properties, formatting axes, etc.

↳ supports a wide variety of graphs and plots namely - histogram, bar charts, power spectra, error charts etc.

Example :-

```
import numpy as np
import matplotlib.pyplot as plt
x = np.arrange(0, 3 * np.pi, 0.1)

y = np.sin(x)
plt.title("sine wave form")
plt.plot(x, y)
plt.show()
```



## Plot Components

axis - Axes of the graph that are typically x, y and z (for 3D plots)

title - Title of the plot.

label - Name for the axis (e.g xlabel or ylabel).

legend - Legend for the plot.

ticklabel - Text or Values represented on the axis.

Symbol - Symbol for data points that can be presented with different symbol, shapes, colors.

Size - Size of, for example, a point on a scatter plot.

linestyle - The style how the line should be drawn. Can be solid or dashed, for example

linewidth - The width of a line on a plot.

alpha - Transparency level of a filled element in a plot.

tick(s) - Refers to the tick marks on a plot.

annotation - Refers to the text added to a plot.

padding - The distance between an (axis/tick) label and the axis.

## Plotting with Pandas

| Parameter | Type |
|---|---|
| Kind | String |
| x/y | String (or) list |
| ax | Axes |
| subplots | Boolean |
| layout | Tuple of (rows, columns) |
| figsize | Tuple of (width, height) |
| title | string for the plot title or a list of strings for subplot titles |
| legend | Boolean |
| label | string if a single column is being plotted. |
| style | string if a single column is being plotted. |
| Color | string if a single column is being plotted. |
| Colormap | String or matplotlib colormap object |
| logx/logy/log log | Boolean |
| xticks/yticks | List of values |
| rot | Integer |
| xlim/ylim | Tuple of the form (min, max) |
| sharex/sharey | Boolean |
| fontsize | Integer |
| grid | Boolean |

### Plotting

↳ Pandas uses the plot() method to create diagrams.

### Scatter Plot

↳ Specify that you want a scatter plot with the kind argument.

kind = 'scatter'

A scatter plot needs an x and y axis.

## Lag Plot

↳ checks whether a dataset or time series is random or not.

↳ Random data should not exhibit any identifiable structure in the lag plot.

↳ Non random structure in the lag plot indicates that the underlying data are not random.

## Auto Correlation Plots :-

↳ is a Commonly used tool for checking randomness in a data set.

### Syntax :-

```
matplotlib. pyplot. accor(x, *, data = None, ** kwargs)
```

### Parameters :-

x, detrend, normed, userlines, maxlags, linestyle and marker.

## Bootstrap Plot :-

↳ is used to eliminate the uncertainty of a statistic by relying on random sampling with replacement.

↳ This function will generate bootstrapping plots for Mean, Median and mid-range statistics for the given number of samples of the given size.

## Counts and Frequencies

↳ counting the frequency of occurence of a word in a body of text is often needed during text processing.

```
import nltk
from nltk. Corpus import brown
cfd = nltk. ConditionalFreqDist( (genre, word)
        for genre in brown. categories()
        for word in brown. words( categories = genre))
categories = [ 'hobbies', 'romance', 'humor']
searchwords = [ 'may', 'might', 'must', 'will']
cfd. tabulate (conditions = categories, samples = searchwords)
```

### Output:

```
          may  might  must  will
hobbies   131    22    83   264
romance    11    51    45    43
humor       8     8     9    13
```

UNIT VI

# CAPSTONE PROJECT (Use SQL and Python)

## ① Analyzing Road Safety in the UK

* Evaluate the median severity value of accidents Caused by various Motorcycles.

* Evaluate Accident Severity and Total Accidents per vehicle type.

* Calculate the average severity by vehicle type

* Calculate the average severity and Total Accidents by Motor cycle.

## ② International Debt Statistical Analysis

* What is the total amount of money that Countries owe to the World Bank?

* Which Country has the highest debt, and how much is that?

* What is the mean debt owed by Countries for different debt indicators?

## ③ Analyzing the World Population

* Which Country has the highest population?

* Which Country has the least number of people?

* Which Country is witnessing the highest population growth?

* Which Country has an extraordinary number for the population?

* Which is the most densely Populated Country in the world?

## ④ Analyzing Sales Data

* What was the total revenue generated by the retailer during the time period Covered by the dataset?

* Which product had the highest total revenue?

* Which customers generated the most revenue for the retailer?

* How many unique product were sold?

## ⑤ Inventory Control Management

* Which year had the highest sales?

* How was the weather during the year of highest sales?

* Conclude whether the weather has an essential impact on sales.

* Do the sales always rise nearthe holiday season for all the years?

* Analyze the relationship between sales and the different macroeconomic variables in the dataset.