**Name: Abhishek Bayas**

**Roll No: 281020**

**Prn: 22310349**

**Batch: A-1**

**Assignment 1**

**Problem Statement**

Perform the following operations using R/Python on suitable data sets:  
a) read data from different formats (like csv, xls)   
b) Find Shape of Data   
c) Find Missing Values   
d) Find data type of each column   
e) Finding out Zero's   
f) Indexing and selecting data, sort data,   
g) Describe attributes of data, checking data types of each column,   
h) counting unique values of data, format of each column, converting variable data type (e.g. from long to short, vice versa)

**Introduction**

Data handling is an essential step in data analysis, ensuring that raw data is structured, cleaned, and formatted for further processing. Python provides various libraries for handling datasets, with **Pandas** being one of the most widely used. Pandas simplifies data manipulation tasks such as reading datasets from different formats, checking missing values, transforming data types, and sorting data.

This document provides an overview of how to perform essential data handling operations using Pandas in Python.

**Objectives**

The primary objectives of this study are:

* To understand how to read data from different file formats.
* To analyse the shape and structure of a dataset.
* To detect and handle missing values.
* To examine data types and format conversion.
* To perform indexing, selecting, and sorting operations on data.
* To describe the attributes of a dataset.
* To count unique values and identify zero values in the dataset.

**Pandas**

Pandas is a Python library built for data manipulation and analysis. It provides two main data structures:

* **Series:** A one-dimensional labeled array.
* **DataFrame:** A two-dimensional table with labeled rows and columns.

Pandas is widely used for:

* Importing and exporting datasets.
* Cleaning and handling missing data.
* Performing mathematical and statistical operations.
* Reshaping, indexing, and filtering data.

**Methodology:**

**a) Reading Data from Different Formats (CSV, XLS, etc.)**

**Concept**

Data is available in multiple formats like CSV (Comma-Separated Values), XLS/XLSX (Excel files), JSON (JavaScript Object Notation), and databases. Efficient data handling starts with loading data into a structured format, such as a dataframe, which allows for easy manipulation and analysis.

* **CSV files:** Commonly used due to their simplicity and compatibility.
* **Excel files:** Preferred for structured data storage with multiple sheets.
* **JSON files:** Used for semi-structured data, especially in web applications.

**Methods**

* **Python:** Uses the pandas library with functions like read\_csv(), read\_excel(), and read\_json().

**b) Finding the Shape of Data**

**Concept**

The shape of a dataset refers to the number of rows (observations) and columns (features). Knowing the shape helps in understanding dataset size and structure.

**Methods**

* **Python:** df.shape returns a tuple (rows, columns).

**c) Finding Missing Values**

**Concept**

Missing values occur due to incomplete data entry, sensor failures, or other issues. Identifying and handling missing values is essential to prevent biases in analysis.

**Methods**

* **Python:** df.isnull().sum() counts missing values per column.

**d) Finding Data Type of Each Column**

**Concept**

Each column in a dataset has a specific data type (integer, float, string, etc.). Knowing these types ensures correct data processing.

**Methods**

* **Python:** df.dtypes returns column data types.

**e) Finding Out Zero’s**

**Concept**

Zero values in a dataset may represent valid values or missing/incomplete information. Identifying them helps in data cleaning.

**Methods**

* **Python:** (df == 0).sum() counts zeros in each column.

**f) Indexing and Selecting Data, Sorting Data**

**Concept**

* **Indexing and Selecting Data:** Selecting specific rows/columns for focused analysis.
* **Sorting Data:** Arranging values in ascending/descending order to identify patterns or outliers.

**Methods**

* **Python:**
  + Indexing: df.loc[] (label-based) and df.iloc[] (position-based).
  + Sorting: df.sort\_values(by='column\_name').

**g) Describing Attributes of Data and Checking Data Types**

**Concept**

Descriptive statistics provide insights into data distribution, including mean, median, standard deviation, and count. Checking data types ensures compatibility with operations.

**Methods**

* **Python:**
  + df.describe() provides summary statistics.
  + df.info() displays column types and missing values.

**h) Counting Unique Values, Formatting, and Converting Variable Data Type**

**Concept**

* **Counting Unique Values:** Identifies distinct categories or values in a column.
* **Formatting Data:** Adjusts how values are displayed (e.g., rounding decimals).
* **Converting Data Type:** Changes column types for compatibility (e.g., string to numeric).

**Methods**

* **Python:**
  + df['column\_name'].nunique() for unique values.
  + df.astype({'column\_name': 'int'}) for type conversion.

**Conclusion**

Pandas provides a powerful set of tools for handling and analysing data efficiently. By understanding and applying these concepts, data analysts can preprocess and transform datasets effectively, ensuring high-quality data for analysis and decision-making.