# Assignment: 4

**AIM:** Perform the different operations using Python on the Facebook metrics data set.

**PROBLEM STATEMENT /DEFINITION**

Perform the following operations using Python on the facebook metrics datasets

a. Create data subsets
b. Merge Data
c. Sort Data
d. Transposing Data
e. Shape and Reshape Data

**OBJECTIVE:**

To learn Python programming
To learn different data preprocessing techniques.

**THEORY:**

## a. Create data subsets

1. Create a subset of a Python dataframe using the loc() function

Python loc() function enables us to form a subset of a data frame according to a specific row or column or a combination of both.

The loc() function works on the basis of labels i.e. we need to provide it with the label of the row/column to choose and create the customized subset.

Syntax: pandas.dataframe.loc[]

2. Using Python iloc() function to create a subset of a dataframe

**Python iloc() function enables us to create subset choosing specific values from rows and columns based on indexes. That is, unlike loc() function which works on labels, iloc() function works on**

**index values. We can choose and create a subset of a Python dataframe from the data providing the index numbers of the** rows and columns.

Syntax: pandas.dataframe.iloc[]

3. Indexing operator to create a subset of a dataframe

**In a simple manner, we can make use of an indexing operator i.e. square brackets to create a subset of the data.**

**Syntax:dataframe[['col1','col2','colN']]**

## b. Merge Data

**Pandas has full-featured, high performance in-memory join operations idiomatically very similar to relational databases like SQL.**

**Pandas provides a single function, merge, as the entry point for all standard database join operations between DataFrame objects –**

```
pd.merge(left, right, how='inner', on=None, left_on=None, right_on=None,
left_index=False, right_index=False, sort=True)
```

**Here, we have used the following parameters –**

- **left – A DataFrame object.**
- **right – Another DataFrame object.**
- **on – Columns (names) to join on. Must be found in both the left and right DataFrame objects.**
- **left_on – Columns from the left DataFrame to use as keys. Can either be column names or arrays with length equal to the length ofthe DataFrame.**

- **right_on − Columns from the right DataFrame to use as keys. Can either be column names or arrays with length equal to the length of the DataFrame.**
- **left_index − If True, use the index (row labels) from the left DataFrame as its join key(s). In case of a DataFrame with a MultiIndex (hierarchical), the number of levels must match the number of join keys from the right DataFrame.**
- **right_index − Same usage as left_index for the right DataFrame.**
- **how − One of 'left', 'right', 'outer', 'inner'. Defaults to inner. Each method has been described below.**
- **sort − Sort the result DataFrame by the join keys in lexicographical order. Defaults to True, setting to False will improve the performance substantially in many cases.**

## c. Sort Data

**pandas.DataFrame.sort_values**

**DataFrame.sort_values(by, axis=0, ascending=True, inplace=False, kind='quicksort', na_position='last', ignore_index=False, key=None)[source]**

**Sort by the values along either axis.**

## d. Transposing Data

pandas.DataFrame.transpose

**DataFrame.transpose(**args*, *copy=False*)[source]**

**Transpose index and columns.**

**Reflect the DataFrame over its main diagonal by writing rows as columns and vice-versa. The property T is an accessor to the method transpose().**

## e. Shape and Reshape Data

### pandas.melt

**pandas.melt(*frame*, *id_vars=None*, *value_vars=None*, *var_name=None*, *value_name='value'*, *col_level=None*, *ignore_index=True*)**

**Unpivot a DataFrame from wide to long format, optionally leaving identifiers set.**

**This function is useful to massage a DataFrame into a format where one or more columns are identifier variables (id_vars), while all other columns, considered measured variables (value_vars), are "unpivoted" to the row axis, leaving just two non-identifier columns, 'variable' and 'value'.**

## pandas.DataFrame.pivot

**DataFrame.pivot(*index=None*, *columns=None*, *values=None*)**

**Return reshaped DataFrame organized by given index / column values.**

**Reshape data (produce a "pivot" table) based on column values. Uses unique values from specified index / columns to form axes of the resulting DataFrame. This function does not support data aggregation, multiple values will result in a MultiIndex in the column.**

# CONCLUSION:

**Understand different data preprocessing techniques**