Course Name:	**Data Analysis Laboratory (216H03L501)**	**Semester:**	**V**
Date of Performance:	**14 / 07 / 2025**	**DIV/ Batch No:**	**D6**
Student Name:	**Samiksha Sharma**	**Roll No:**	**16010123298**

Experiment No: 1

Title: Studies on Pandas library of python.

Objectives of the Experiment:

1. To understand and apply the fundamental functionalities of the pandas library for data analysis.
2. To manipulate and transform datasets using filtering, sorting, and column operations.
3. To analyze data using grouping and aggregation techniques to derive meaningful insights.

COs to be achieved:

CO1: Understand basic concepts of data analytics to solve real-world problems

Books/ Journals/ Websites referred:

1. Students should write

Theory:

Students should write about pandas

Problem statement/ Tasks

Task 1: Import Required Libraries and Dataset

- Import pandas and load a real-world CSV dataset (e.g., Titanic, Student Performance, COVID-19).
- Display the first and last 5 records using head() and tail().

Task 2: Basic Exploration of the Dataset

- Display the dataset shape using .shape, column names using .columns, and data types using .dtypes.
- Generate summary statistics using .describe() and data info using .info().

Task 3: Identify Missing and Duplicate Data

- Detect missing values using .isnull().sum().

- Remove or fill missing values using .dropna() or .fillna().
- Check for and remove duplicate rows using .duplicated() and .drop_duplicates().

Task 4: Filtering Records

- Extract rows based on specific conditions (e.g., students who scored more than 80%, passengers who survived).

Task 5: Sorting the Dataset

- Sort the dataset based on one or more columns using .sort_values().
  - Example: Sort by age or total score.

Task 6: Creating or Modifying Columns

- Create new columns from existing ones (e.g., Total Marks = Math + Science + English).
- Drop unnecessary columns using .drop().
- Rename columns using .rename().

Task 7: Grouping and Aggregation

- Use .groupby() to find average, count, or sum based on a categorical column.
- Example:
  - Average marks by gender: df.groupby('Gender')['Marks'].mean()
  - Survival rate by class: df.groupby('Pclass')['Survived'].mean()

Task 8: Pivot Tables or Multi-Level Grouping (Optional for advanced students)

- Create pivot tables using .pivot_table() to summarize complex data.
  - Example: Average score by gender and class.

Task 9: Insight Generation

- Write 3-5 key insights based on the group-by and aggregated data.
- Example:
  - "Female students have higher average marks in English."
  - "Survival rate is highest for first-class passengers."

Code :

Task 1:

```
[4] import pandas as pd
    df = pd.read_csv("/content/jadavpur,-kolkata-air-quality.csv")
    df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2032 entries, 0 to 2031
Data columns (total 7 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   date    2032 non-null   object
 1    pm25   2032 non-null   object
 2    pm10   2032 non-null   object
 3    o3     2032 non-null   object
 4    no2    2032 non-null   object
 5    so2    2032 non-null   object
 6    co     2032 non-null   object
dtypes: object(7)
memory usage: 111.3+ KB
```

```
[5] df.head()
```

	date	pm25	pm10	o3	no2	so2	co
0	2025/7/1	90	38	7	10		2
1	2025/7/2	81	39	7	6		2
2	2025/7/3	82	47	20	9		3
3	2025/7/4	93	49	6	14	1	3
4	2025/7/5	96	42	6	11		3

```
[6] df.tail()
```

	date	pm25	pm10	o3	no2	so2	co
2027	2021/6/10		40	12	7	2	3
2028	2020/10/26		81	41	27	7	10
2029	2020/9/27		67	9	11	2	6
2030	2019/12/31		142	5	25	2	16
2031	2019/12/2		110		32	1	16

Task 2:

```
[9] df.shape
```

```
(2032, 7)
```

```
[10] df.columns
```

```
Index(['date', ' pm25', ' pm10', ' o3', ' no2', ' so2', ' co'], dtype='object')
```

```
df.dtypes
```

	0
date	object
pm25	object
pm10	object
o3	object
no2	object
so2	object
co	object

dtype: object

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2032 entries, 0 to 2031
Data columns (total 7 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   date    2032 non-null   object
 1    pm25   2004 non-null   float64
 2    pm10   2005 non-null   float64
 3    o3     1999 non-null   float64
 4    no2    1965 non-null   float64
 5    so2    1950 non-null   float64
 6    co     1990 non-null   float64
dtypes: float64(6), object(1)
memory usage: 111.3+ KB
```

```
df.describe()
```

	pm25	pm10	o3	no2	so2	co
count	2004.000000	2005.000000	1999.000000	1965.000000	1950.000000	1990.000000
mean	111.702595	69.085786	17.479240	10.541476	2.555385	5.515578
std	49.925950	38.019779	16.255081	8.654428	1.792968	4.339132
min	18.000000	1.000000	1.000000	1.000000	1.000000	1.000000
25%	67.750000	38.000000	8.000000	5.000000	1.000000	3.000000
50%	105.000000	59.000000	13.000000	8.000000	2.000000	4.000000
75%	157.000000	98.000000	23.000000	14.000000	3.000000	8.000000
max	317.000000	249.000000	402.000000	96.000000	16.000000	37.000000

Task 3:

```
if df.isna().sum().sum() < 50:
    df.dropna(inplace=True)
else:
    df.fillna(df.median(numeric_only=True), inplace=True)

print(df)
```

```
            date   pm25   pm10    o3   no2   so2    co
0       2025/7/1   90.0   38.0   7.0  10.0   2.0   2.0
1       2025/7/2   81.0   39.0   7.0   6.0   2.0   2.0
2       2025/7/3   82.0   47.0  20.0   9.0   2.0   3.0
3       2025/7/4   93.0   49.0   6.0  14.0   1.0   3.0
4       2025/7/5   96.0   42.0   6.0  11.0   2.0   3.0
...          ...    ...    ...   ...   ...   ...   ...
2027   2021/6/10  105.0   40.0  12.0   7.0   2.0   3.0
2028  2020/10/26  105.0   81.0  41.0  27.0   7.0  10.0
2029   2020/9/27  105.0   67.0   9.0  11.0   2.0   6.0
2030  2019/12/31  105.0  142.0   5.0  25.0   2.0  16.0
2031   2019/12/2  105.0  110.0  13.0  32.0   1.0  16.0

[2032 rows x 7 columns]
```

```
[32] dup = df.duplicated()
     print(dup)
```

```
0       False
1       False
2       False
3       False
4       False
        ...
2027    False
2028    False
2029    False
2030    False
2031    False
Length: 2032, dtype: bool
```

Task 4:

```
high_co = df[df[' co'] > 5]
print(high_co)
```

```
            date   pm25   pm10    o3   no2  so2    co
49      2025/5/6  124.0   69.0  40.0  14.0  9.0   6.0
77      2025/6/3  143.0   51.0   9.0  40.0  6.0  20.0
78      2025/6/4  101.0   55.0  12.0  42.0  NaN  37.0
79      2025/6/5  109.0   62.0  11.0  35.0  NaN  32.0
93     2025/6/19   89.0   52.0   6.0  14.0  4.0   8.0
...          ...    ...    ...   ...   ...  ...   ...
2025  2022/11/16    NaN  163.0   5.0  22.0  5.0  21.0
2028  2020/10/26    NaN   81.0  41.0  27.0  7.0  10.0
2029   2020/9/27    NaN   67.0   9.0  11.0  2.0   6.0
2030  2019/12/31    NaN  142.0   5.0  25.0  2.0  16.0
2031   2019/12/2    NaN  110.0   NaN  32.0  1.0  16.0

[710 rows x 7 columns]
```

Task 5:

```
df.sort_values(' pm25')
```

	date	pm25	pm10	o3	no2	so2	co
1843	2020/5/19	18.0	14.0	12.0	2.0	2.0	3.0
1868	2020/6/13	18.0	20.0	9.0	4.0	1.0	3.0
1845	2020/5/21	21.0	34.0	21.0	2.0	2.0	3.0
1733	2020/7/30	23.0	16.0	6.0	3.0	2.0	3.0
1866	2020/6/11	24.0	17.0	8.0	6.0	2.0	3.0
...	...	...	...	...	...	...	...
1698	2020/12/29	259.0	158.0	8.0	25.0	3.0	14.0
1909	2020/1/24	289.0	61.0	6.0	24.0	4.0	9.0
1162	2022/1/6	291.0	148.0	10.0	38.0	2.0	10.0
1161	2022/1/5	293.0	157.0	29.0	39.0	6.0	13.0
1886	2020/1/1	317.0	118.0	11.0	22.0	2.0	11.0

2032 rows × 7 columns

Task 6:

```python
df.columns = df.columns.str.strip()

pollutant_cols = ['pm25', 'pm10', 'o3', 'no2', 'so2', 'co']
df[pollutant_cols] = df[pollutant_cols].apply(pd.to_numeric, errors='coerce')

df['total_pollutants'] = (df['pm25'] + df['pm10'] + df['o3'] + df['no2'] + df['so2'] + df['co'])
df = df.drop(columns=['date'])
df = df.rename(columns={
    'pm25': 'PM2_5',
    'pm10': 'PM10',
    'o3': 'O3',
    'no2': 'NO2',
    'so2': 'SO2',
    'co': 'CO'
})
```

[3] ✓ 0.0s Python

```python
df.head()
```

[4] ✓ 0.0s Python

	PM2_5	PM10	O3	NO2	SO2	CO	total_pollutants
0	65.0	24.0	12.0	7.0	4.0	6.0	118.0
1	58.0	33.0	10.0	8.0	2.0	2.0	113.0
2	72.0	34.0	9.0	11.0	1.0	3.0	130.0
3	74.0	32.0	10.0	10.0	2.0	5.0	133.0
4	72.0	27.0	11.0	8.0	2.0	5.0	125.0

Task 7:

```python
df['main_pollutant'] = df[['PM2_5', 'PM10', 'O3', 'NO2', 'SO2', 'CO']].idxmax(axis=1)

avg_by_main_pollutant = df.groupby('main_pollutant')[['PM2_5', 'PM10', 'O3', 'NO2', 'SO2', 'CO', 'total_pollutants']].mean()

count_by_main_pollutant = df['main_pollutant'].value_counts()

print("Average levels by main pollutant:")
print(avg_by_main_pollutant)
print("\nCounts of main pollutants:")
print(count_by_main_pollutant)
```

[5] ✓ 0.0s

```
Average levels by main pollutant:
                     PM2_5     PM10         O3        NO2       SO2      CO  \
main_pollutant
NO2                    NaN      NaN        NaN   6.000000       NaN     NaN
O3               49.250000 28.80000  59.100000   6.600000  8.300000  4.3000
PM10             52.989011 89.79562  21.139706   8.345588  4.820312  4.5000
PM2_5           114.587265 70.79063  25.443735  12.802491  4.719615  5.1907

                total_pollutants
main_pollutant
NO2                          NaN
O3                    169.875000
PM10                  194.387500
PM2_5                 237.427945

Counts of main pollutants:
main_pollutant
PM2_5    1759
PM10      137
O3         10
NO2         1
```

Task 8:

```python
pivot_pollutants = df.pivot_table(values=['PM2_5', 'PM10', 'O3', 'NO2', 'SO2', 'CO', 'total_pollutants'],
                                  index='main_pollutant',
                                  aggfunc='mean')

print(pivot_pollutants)
```

[6] ✓ 0.0s

```
                    CO        NO2         O3      PM10       PM2_5       SO2  \
main_pollutant
NO2                NaN   6.000000        NaN       NaN         NaN       NaN
O3              4.3000   6.600000  59.100000  28.80000   49.250000  8.300000
PM10            4.5000   8.345588  21.139706  89.79562   52.989011  4.820312
PM2_5           5.1907  12.802491  25.443735  70.79063  114.587265  4.719615

                total_pollutants
main_pollutant
NO2                          NaN
O3                    169.875000
PM10                  194.387500
PM2_5                 237.427945
```

Task 9:

```python
insights = [
    "PM2.5 is the dominant pollutant in most readings.",
    "Rows where PM2.5 is dominant have the highest average total pollutants.",
    "O3 is rarely the dominant pollutant compared to PM2.5 or PM10.",
    "SO2-dominant rows tend to have lower overall total pollutants.",
    "PM10 dominance is linked with higher NO2 readings than average."
]

for i, insight in enumerate(insights, start=1):
    print(f"{i}. {insight}")
```

[7] ✓ 0.0s

```
1. PM2.5 is the dominant pollutant in most readings.
2. Rows where PM2.5 is dominant have the highest average total pollutants.
3. O3 is rarely the dominant pollutant compared to PM2.5 or PM10.
4. SO2-dominant rows tend to have lower overall total pollutants.
5. PM10 dominance is linked with higher NO2 readings than average.
```

Post Lab Subjective/Objective type Questions:

1. **What is the difference between .info() and .describe() in pandas?**
   .info() gives you the structure of the dataset while .describe() gives you the statistical summary of the dataset.
2. **How does pandas handle missing data? Mention at least two functions used for this purpose.**
   isnull() functions helps detect the null/empty values. dropna() and fillna() are major functions used to handle missing data. dropna() directly deletes the row, this can be done when the number of missing data is less. While, fillna() enters dummy data in place of empty data, ex: with median or mean or mode, etc.
3. **What is a pivot table in pandas, and how is it useful in summarizing data?**
   A pivot table is a way to summarize data in a df, allowing you to calculate aggregations like sum, means etc based on one or more keys.
4. **What were the key insights you discovered from the dataset during your analysis?**
   PM2.5 was the dominant pollutant in most readings.
   Rows where PM2.5 was dominant had the highest average total pollutants.
   O3 was rarely the dominant pollutant.
   SO2-dominant readings had much lower overall pollutant totals.
   PM10 dominance was often paired with higher NO2 levels than normal.

Conclusion:

From this experiment, we now know how to handle data nad how to conduct a thorough analysis and prep it for further use.