Asmita Porwal
Data Engineering
Batch-1
2/2/24

Coding Challenge -2 Question-1

Explain Pandas for Data Processing

Pandas is a powerful open-source data manipulation and analysis library for Python.

Pandas is a powerful Python library for data manipulation and analysis.

It provides easy-to-use data structures like Series and DataFrame, along with functions to efficiently manipulate, clean, and analyze structured data,making it an essential tool for data processing tasks.

Pandas is widely used in data science, machine learning, and data analysis tasks..

Here are some key aspects of Pandas for data processing:

Key Pandas Data Structures:

Series:

- A one-dimensional labeled array that can hold any data type.
- It is similar to a column in a spreadsheet or a single column in a DataFrame.
-

DataFrame:

- A two-dimensional labeled data structure with columns that can be of different data types.
- It is similar to a spreadsheet or a SQL table

Data Selection and Indexing:

- Pandas provides convenient ways to select, filter, and index data within a DataFrame.

- You can use labels, indices, or conditions to extract specific rows or columns.

Data Cleaning:

- Pandas offers various functions to handle missing data, including methods for filling, dropping, or interpolating missing values.
- It provides tools to deal with duplicate entries, outliers, and other data cleaning tasks.

Data Transformation:

- Pandas allows for easy reshaping and transforming of data. You can pivot, melt, and reshape data frames as needed.
- It supports the creation of new columns based on existing data or applying custom functions to existing columns.

Merging and Joining:

- Pandas facilitates combining multiple DataFrames through merging and joining operations.
- You can merge DataFrames based on common columns or indices, similar to SQL joins.

Grouping and Aggregation:

- Pandas supports grouping data based on one or more columns and applying aggregate functions (like sum, mean, count) to the grouped data.
- This is useful for summarizing and aggregating information within a dataset.

Input/Output:

- Pandas supports reading and writing data in various formats, including CSV, Excel, SQL databases, and more.
- This flexibility makes it easy to integrate Pandas into different data processing pipelines.

Visualization:

- While Pandas itself doesn't handle visualization, it seamlessly integrates with popular plotting libraries like Matplotlib and Seaborn, allowing for easy data visualization.

Efficiency:

- Pandas is built on top of NumPy, which makes it efficient for numerical operations. It is optimized for performance and memory usage.

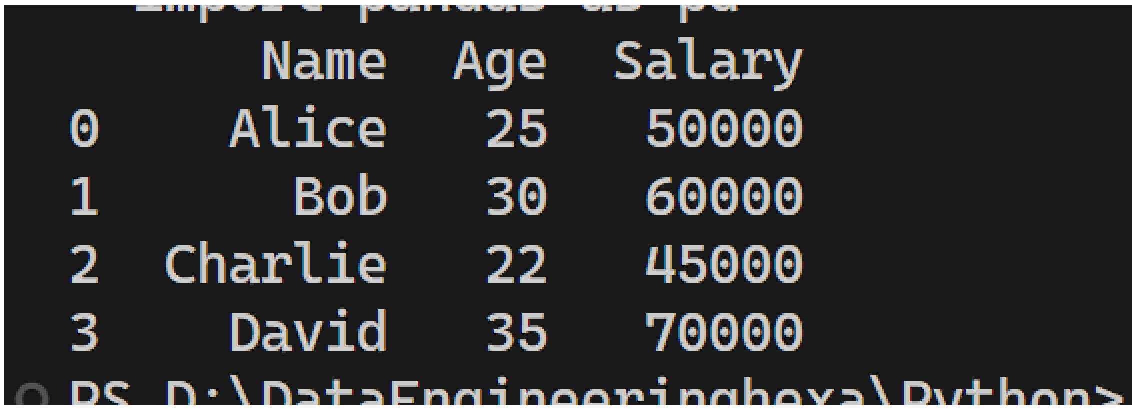
Overall, Pandas is a versatile and comprehensive library that simplifies many aspects of data processing.

It is widely used in data science, machine learning, and data analysis workflows due to its simplicity, flexibility, and extensive functionality.

Execute Reading CSV Data using Pandas

```python
# Importing the pandas library
import pandas as pd

# Reading CSV data into a DataFrame
df = pd.read_csv("D:\DataEngineeringhexa\Python\output.csv")

# Displaying the first few rows of the DataFrame
print(df.head())
```

Output



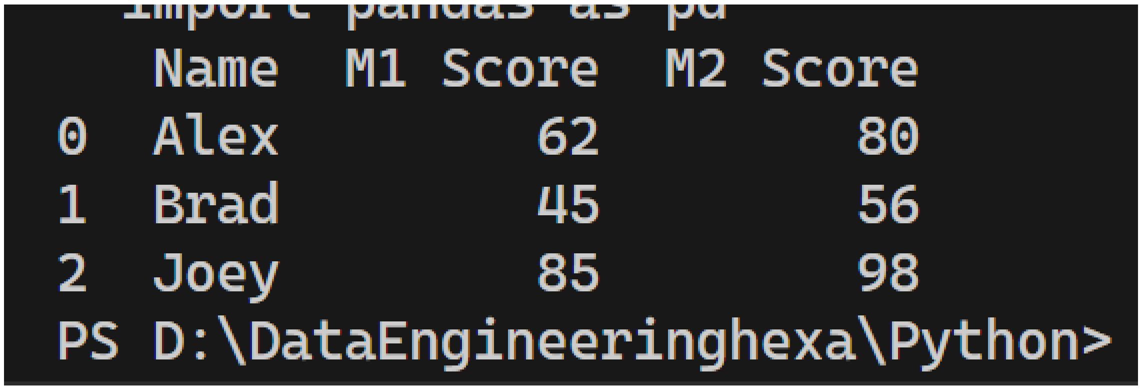
```
       Name  Age  Salary
0     Alice   25   50000
1       Bob   30   60000
2   Charlie   22   45000
3     David   35   70000
PS D:\DataEngineeringhexa\Python>
```

Read Data from CSV Files to Pandas Dataframes

```python
# Importing the pandas library
import pandas as pd

# Reading CSV data into a DataFrame
df = pd.read_csv("D:\DataEngineeringhexa\Python\Stu_data.csv")
```

```
# Displaying DataFrame
print(df)
```

Output



```
   Name  M1 Score  M2 Score
0  Alex        62        80
1  Brad        45        56
2  Joey        85        98
PS D:\DataEngineeringhexa\Python>
```

Import Pandas: The import pandas as pd statement imports the Pandas library and aliases it as pd for convenience.
Reading CSV Data: The pd.read_csv() function is used to read data from a CSV file. You need to provide the path to your CSV file as an argument (file_path in this example).
Displaying Data: The df.head() function is used to display the first few rows of the DataFrame. This helps you quickly inspect the loaded data.

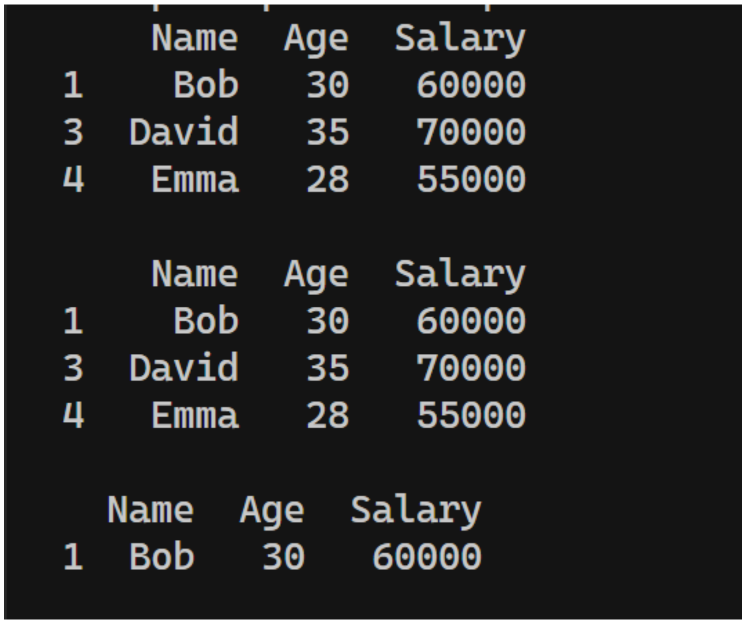
Filter Data in Pandas Dataframe using query.

The **query method** is used to filter the DataFrame df.
The query expression is a string that specifies the conditions for filtering

```
# Importing the Pandas library
import pandas as pd

# Creating a sample DataFrame
data = {
```

```python
    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Emma'],
    'Age': [25, 30, 22, 35, 28],
    'Salary': [50000, 60000, 45000, 70000, 55000]
}

df = pd.DataFrame(data)

# Using the query method to filter data
filtered_df = df.query('Age > 25 and Salary > 50000')

# Displaying the filtered DataFrame
print(filtered_df,"\n")

# Filter data where Age is between 25 and 35 and Salary is greater than
50000
filtered_df = df.query('25 <= Age <= 35 and Salary > 50000')

# Displaying the filtered DataFrame
print(filtered_df,"\n")

# Define a variable and use it in the query
target_age = 30
filtered_df = df.query(f'Age == {target_age}')

# Displaying the filtered DataFrame
print(filtered_df,"\n")
```

Output



```
    Name  Age  Salary
1    Bob   30   60000
3  David   35   70000
4   Emma   28   55000

    Name  Age  Salary
1    Bob   30   60000
3  David   35   70000
4   Emma   28   55000

  Name  Age  Salary
1  Bob   30   60000
```