PRESENTATION

By Rishabh Dubey

AGENDA

1. Quick Revision Quiz (POP & OOP)
2. Introduction to Pandas (What & Why, 10 min)
3. Core Pandas Concepts (Series & DataFrame, 10 min)
4. Essential Pandas Functions (Head, Tail, Info, Describe, etc., 30 min)
5. Data Cleaning & Manipulation (Handling missing values, filtering, grouping, merging, 20 min)
6. Class Hands-on Exercise (Apply learned concepts, 10 min)
7. Q&A & Wrap-up (5 min)

Quiz Time

QUICK REVISION QUIZ (MCQS & TRUE/FALSE)

Q1: Which of the following is NOT a valid lambda function syntax?

A) lambda x: x + 2
B) lambda x, y: x * y
C) lambda: print("Hello")
D) lambda x: (x**2, x**3)

QUICK REVISION QUIZ (MCQS & TRUE/FALSE)

Q2: What keyword is used to define a class in Python?

A) class
B) def
C) object
D) self

QUICK REVISION QUIZ (MCQS & TRUE/FALSE)

Q3: How do you indicate a private variable in Python?

A) _var
B) __var
C) var
D) private var

QUICK REVISION QUIZ (MCQS & TRUE/FALSE)

Q4: Which type of inheritance allows a class to inherit from multiple parent classes?

A) Single
B) Multilevel
C) Multiple
D) Hybrid

QUICK REVISION QUIZ (MCQS & TRUE/FALSE)

Q5: What happens when a child class has a method with the same name as a method in the parent class?

A) Parent method is always called
B) Child method overrides the parent method
C) Python throws an error
D) Both methods execute

Introduction to Pandas

INTRODUCTION TO PANDAS

📌 What is Pandas?

- Python library for data manipulation & analysis
- Built on NumPy, designed for handling structured data
- Key components: Series (1D) & DataFrame (2D)

BASIC PANDAS OBJECTS

📌 Series – 1D labeled array

📌 DataFrame – 2D table-like data

PANDAS SERIES (1D DATA STRUCTURE)

📌 What is a Pandas Series?

- A one-dimensional labeled array (like a list with an index)
- Can store integers, floats, strings, or objects

📝 Creating a Series from a List

```
import pandas as pd
data = [10, 20, 30, 40]
s = pd.Series(data)
print(s)
```

🛠️ Key Features:

✅ Auto-generated index (0,1,2,...)
✅ Supports custom indexing
✅ Fast & optimized

PANDAS DATAFRAME (2D DATA STRUCTURE)

📌 What is a DataFrame?

- A two-dimensional table (like an Excel sheet)
- Rows & columns with labeled axes

📝 Creating a DataFrame from a Dictionary

```
data = {
'Name': ['Alice', 'Bob', 'Charlie'],
 'Age': [25, 30, 35]
}
df = pd.DataFrame(data)
print(df)
```

🛠️ Key Features:

✅ Handles structured data easily

✅ Supports filtering, sorting, and transformations

✅ Can import/export data (CSV, Excel, SQL)

WAYS TO CREATE PANDAS SERIES & DATAFRAMES

- 📌 Ways to Create a Pandas Series
- 1️⃣ From a List
- 2️⃣ From a NumPy Array
- 3️⃣ From a Dictionary (Key = Index, Value = Data)

- 📌 Ways to Create a Pandas DataFrame
- 1️⃣ From a Dictionary
- 2️⃣ From a List of Lists
- 3️⃣ From a NumPy Array

ESSENTIAL PANDAS FUNCTIONS

📌 Exploring Data in Pandas

1. head(n) & tail(n) – View the first & last n rows
2. info() – Summary of dataset (data types, memory usage)
3. describe() – Statistical summary (mean, min, max, etc.)
4. shape & columns – Get dimensions & column names
5. value_counts() – Count unique values in a column

PRACTICE QUESTION

1. Load the dataset (Create your own)
2. Display the first 7 rows using head()
3. Get dataset summary with info()
4. Find the mean and max values using describe()
5. Count unique values in a categorical column

DATA CLEANING & MANIPULATION

📌 Key Concepts

1️⃣ Handling Missing Values – **dropna(), fillna(), isnull().sum()**

2️⃣ Filtering Data – Conditional selection using **loc[]** & **query()**

3️⃣ Grouping & Aggregation – **groupby()**, **agg()** for summary statistics

4️⃣ Merging & Joining – **merge()**, **concat()** for combining datasets

HANDLING MISSING VALUES

Concept: Missing data is common in datasets; handling it properly is crucial.

Methods:

- **dropna():** Removes missing values (entire row/column if NaN is present)
- **fillna():** Fills missing values with a specified value (mean, median, mode, forward fill, backward fill)
- **isnull().sum():** Counts missing values in each column
- **interpolate():** Estimates missing values using interpolation techniques
- **replace():** Replaces missing or specific values with defined values

HANDLING MISSING VALUES

Concept: Missing data is common in datasets; handling it properly is crucial.

Link to github gist :
https://gist.github.com/Rishabh7406/c462ab3b56f0219de4d4ad3447e0ea05

FILTERING DATA

Concept: Selecting specific data using conditions.

Methods:

- **loc[]:** Selects rows based on conditions (label-based indexing)
- **iloc[]:** Selects rows based on index position (integer-based indexing)
- **query():** Filters using SQL-like syntax
- **between():** Filters values within a range
- **mask():** Replaces values where conditions hold true

FILTERING DATA

Concept: Selecting specific data using conditions.

Link to github gist :
https://gist.github.com/Rishabh7406/c8473ee85595bb21006a077f92979736

MISSING VALUES & FILTERING

📌 **Practice 1:**

✅ **Task 1: Drop all rows with missing values.**

✅ **Task 2: Fill missing values in Salary with the column mean.**

✅ **Task 3: Count total missing values in each column.**

✅ **Task 4: Select rows where Age > 30 using .loc[].**

✅ **Task 5: Use .query() to get rows where Score >= 80.**

MISSING VALUES & FILTERING

```
import pandas as pd
import numpy as np

# Sample DataFrame
df = pd.DataFrame({
    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve'],
    'Age': [25, np.nan, 35, 40, np.nan],
    'Salary': [50000, 60000, np.nan, 80000, 75000],
    'Score': [85, 90, 78, np.nan, 88],
    'Department': ['HR', 'IT', 'Finance', 'IT', 'HR']
})

display(df)
```

GROUPING & AGGREGATION

Concept: Summarizing data for insights.

Methods:

- **groupby():** Groups data based on a column (aggregation performed per group)
- **agg():** Performs summary statistics like mean, sum, count, min, max
- **transform():** Applies operations while maintaining original structure
- **pivot_table():** Creates multi-level aggregation summaries

GROUPING & AGGREGATION

Concept: Summarizing data for insights.

Link to github Gist:
https://gist.github.com/Rishabh7406/db9e01bf7623c5769c9f23499a573555

MERGING & JOINING DATA

Concept:Combining datasets to enhance analysis.

Methods:

- **merge():** Joins two datasets based on a common column (on parameter)
- **concat():** Stacks datasets vertically or horizontally
- **join():** Similar to merge(), but works on indices
- **combine_first():** Fills missing values from another dataset

MERGING & JOINING DATA

Concept:Combining datasets to enhance analysis.

Link to github gist:
https://gist.github.com/Rishabh7406/398c98d4365e439c98c22af70fd2f97d

GROUPING, AGGREGATION & MERGING

📌**Practice 2:**

✅ **Task 6:** Find the average Salary per Department using groupby().
✅ **Task 7:** Get the max Age for each Job Role using agg().
✅ **Task 8:** Merge Employee & Salary DataFrames using an inner join.
✅ **Task 9:** Concatenate two DataFrames vertically using concat().
✅ **Task 10:** Perform a left join on Employee_ID between two DataFrames.

GROUPING, AGGREGATION & MERGING

```
# Employee Data
employees = pd.DataFrame({
    'Employee_ID': [1, 2, 3, 4],
    'Name': ['Alice', 'Bob', 'Charlie', 'David'],
    'Job Role': ['Analyst', 'Engineer', 'Manager', 'Engineer'],
    'Department': ['Finance', 'IT', 'HR', 'IT']
})

# Salary Data
salaries = pd.DataFrame({
    'Employee_ID': [2, 3, 4, 5],
    'Salary': [60000, 75000, 80000, 50000]
})

display(employees, salaries)
```

THANK YOU