Part 1: Introduction to Data Types and Storage Methods

1. Theory:

   - Discuss different types of data (e.g., numerical, categorical, text).
     - ➢ Numerical Data: Numbers we can measure or count.

       Example: Age, Fare, Number of siblings.

     - ➢ Categorical Data: Labels or names.

       Example: Gender (male/female), Embarked (S/C/Q), Class (1st, 2nd, 3rd).

     - ➢ Text Data: Free-form words or sentences.

       Example: Passenger names, cabin info.

   - Explain storage methods (e.g., databases, files, cloud storage).
     - ➢ Flat Files: Like CSV, Excel, or TXT. Easy to use but not scalable.
     - ➢ Databases: Structured storage using tables (SQL, SQLite). Better for large data.
     - ➢ Cloud Storage: Online storage like AWS S3 or Google Drive. Good for access and backup.

2. Hands-On:

   - Load the Titanic dataset (available at the bottom of this page) and explore different types of data.

```
pclass      float64
survived    float64
name        object
sex         object
age         float64
sibsp       float64
parch       float64
ticket      object
fare        float64
cabin       object
embarked    object
boat        object
body        float64
home.dest   object
```

Part 2: Compare and Contrast Types of Data Sources

1. Theory:

   - Discuss various data sources (e.g., APIs, databases, CSV files, web scraping).

     1. CSV Files

        - Flat file with comma-separated values.

        - Easy to work with using ‘pandas’.

     2. APIs (Application Programming Interfaces)

        - Provide real-time data from websites or services.

        - Accessed using Python’s ‘requests’ library.

     3. Databases (like SQLite)

        - Structured data stored in tables.

        - Useful for large-scale applications.

     4. Web Scraping

        - Extracts data from websites by reading HTML pages.

2. Hands-On:

   - Write Python code to read data from different sources.

     (Please check Jupyter Notebook)

Part 3: Structured vs. Unstructured Data

1. Theory:

   - Define structured data (e.g., databases, spreadsheets).

     - Stored in a tabular format — rows and columns.

     - Easy to search, filter, and analyze using SQL or pandas.

     - Example: Excel sheets, relational databases, CSV files.

   - Define unstructured data (e.g., text, images).

     - No predefined format or schema.

- Harder to store and analyze directly.

- Example: Free text, images, videos, social media posts, emails.

2. Hands-On:

   - Create examples of structured and unstructured data using the Titanic dataset.

     (Please check Jupyter Notebook)

Part 4: Storage Considerations

1. Theory:

   - Discuss storage considerations (e.g., scalability, cost, speed, security).

     - When storing data, we need to think about more than just saving a file. Here are important factors:

     - Scalability: Can the storage handle more data in the future?

     - Cost: Is it affordable (especially for cloud storage)?

     - Speed: How fast can we read or write the data?

     - Security: Is the data protected from unauthorized access?

2. Hands-On:

   - Implement storage solutions in Python.

     I'll simulate saving and reading Titanic data using two common formats:

     - CSV (flat file)

     - SQLite (database)

     (Please check Jupyter Notebook)

Part 5: Integrate and Use an API

1. Theory:

   - Explain what an API is and how it can be used to fetch data.

     - API = Application Programming Interface

     - It lets programs talk to each other and share data.

     - For example, OpenWeatherMap gives weather info through its API.

   - Introduce the OpenWeatherMap API and how to use it.

     - Gives real-time weather data.

- Needs an API key to access it.

- Data is returned in JSON format (like a dictionary in Python).

2. Hands-On:

   - Fetch weather data for the departure (Southampton) and arrival (New York) locations of the Titanic.

     (Please check Jupyter Notebook)

Part 6: Data Quality Dimensions

1. Theory:

   - Discuss data quality dimensions (e.g., accuracy, completeness, consistency, timeliness).

     - Before analyzing or modeling data, we need to assess its quality.

     Key Data Quality Dimensions:

     - Accuracy: Is the data correct and reliable?
     - Completeness: Are all required values filled in?
     - Consistency: Are similar values stored in the same format?
     - Timeliness: Is the data up to date and relevant?

2. Hands-On:

   - Assess the quality of the Titanic dataset.

     (Please check Jupyter Notebook)

Part 7: Data Modeling

1. Theory:

   - Explain the concept of data modeling and its importance.

     - Data Modeling is a way to organize data using entities, attributes, and relationships.

     - Helps us design structured databases for real-world systems.

     Key Components:

     - Entity: Real-world object (e.g., Guest, Room, Booking)

     - Attribute: Details about that object (e.g., Guest Name, Room Number)

     - Primary Key: Uniquely identifies a record (e.g., Guest ID)

- Foreign Key: Links two tables (e.g., Guest ID in Booking table)

2. Hands-On:

   - Create a simple data model for a Hotel Management System.

     Example: **Hotel Management System**

     I'll create three entities:

     -Guests

     -Rooms

     -Bookings

     I'll simulate them using pandas DataFrames.

     (Please check Jupyter Notebook)

Part 8: Data Visualization

1. Theory:

   - Discuss the importance of data visualization.

     - Helps us see patterns and trends easily.

     - Makes complex data easier to understand.

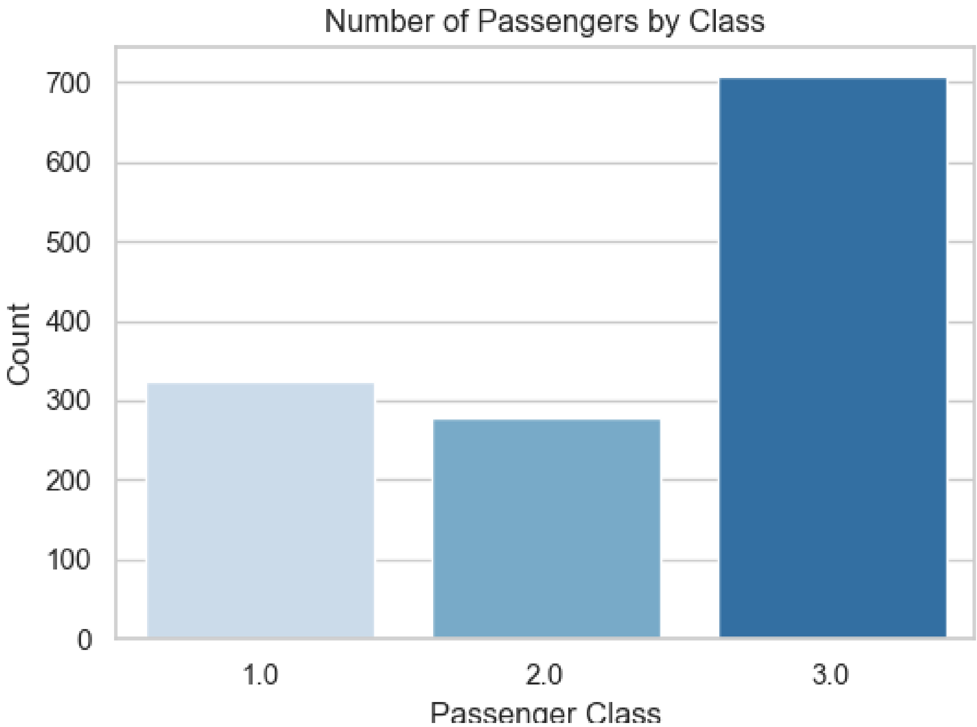
     - Useful for reports, dashboards, and presentations.

2. Hands-On:

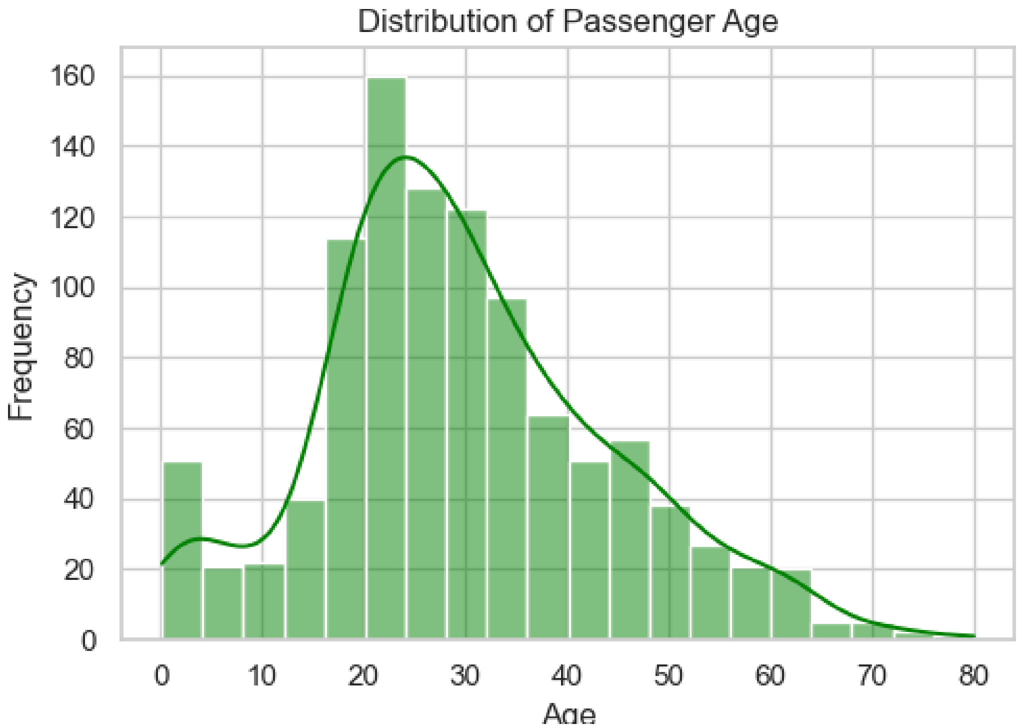
   - Create visualizations using matplotlib and seaborn.

     I'll create two simple visualizations:

     1. Bar chart of passengers by class



2. Histogram of passenger ages



Part 9: Web Scraping

1. Theory:

   ○ Explain what web scraping is and its applications.

   - Web scraping means **collecting data from websites** automatically.

   - We use tools like **BeautifulSoup** to read and extract data from HTML.

2. Hands-On:

   ○ Scrape additional data related to the Titanic from a website.

   I'll scrape a simple page related to the Titanic from Wikipedia.

   Target URL: https://en.wikipedia.org/wiki/RMS_Titanic

   I’ll extract:

   - The first paragraph from the page

   (Please check Jupyter Notebook)