Titanic Dataset Overview

The Titanic dataset includes the following columns:

- **PassengerId**: Unique identifier for each passenger.
- **Survived**: Whether the passenger survived (1) or not (0).
- **Pclass**: Passenger class (1 = 1st, 2 = 2nd, 3 = 3rd).
- **Name**: Name of the passenger.
- **Sex**: Gender of the passenger.
- **Age**: Age of the passenger.
- **SibSp**: Number of siblings or spouses aboard the Titanic.
- **Parch**: Number of parents or children aboard the Titanic.
- **Ticket**: Ticket number.
- **Fare**: Passenger fare.
- **Cabin**: Cabin number.
- **Embarked**: Port of embarkation (C = Cherbourg, Q = Queenstown, S = Southampton).

Exercises with the Titanic Dataset:

1. **Basic Data Exploration**:
    - Display the first 10 rows of the dataset.
    - Display the summary statistics of the dataset.
    - Check for missing values in each column.
2. **Data Selection**:
    - Select only the columns `Name`, `Sex`, and `Age` and display the first 5 rows.
    - Display rows where `Fare` is greater than 50.
3. **Data Filtering**:
    - Filter and display the rows where `Survived` is 1 and `Pclass` is 3.
    - Find all passengers who embarked from 'C' and paid more than 50 for their ticket.
4. **Aggregation and Grouping**:
    - Calculate the average age of passengers for each `Pclass`.
    - Find the total number of survivors grouped by `Pclass`.
    - Find the maximum `Fare` paid for each class (`Pclass`).
5. **Handling Missing Values**:
    - Fill missing values in the `Age` column with the median age.

    - Drop rows where the `Cabin` column has missing values.
6. **Creating New Columns**:
    - Create a new column called `FamilySize` which is the sum of `SibSp` and `Parch`.
    - Create a new column `Fare_per_person` which is the `Fare` divided by `FamilySize + 1`.
7. **Sorting Data**:
    - Sort the dataset by `Fare` in descending order.
    - Sort the dataset by `Age` and `Fare` in ascending order.
8. **Value Counts and Unique Values**:
    - Find the number of unique values in the `Embarked` column.
    - Get the count of each unique value in the `Pclass` column.
9. **Visualization** (if plotting libraries like matplotlib or seaborn are available):
    - Plot the distribution of passenger ages.
    - Plot a bar chart showing the number of passengers for each `Pclass`.
10. **String Operations**:
    - Extract the last name of each passenger into a new column called `LastName`.
    - Create a new column `Title` by extracting titles (like Mr, Mrs, Miss) from the `Name` column.
11. **Conditional Columns**:
    - Create a new column `AgeGroup` that categorizes passengers as 'Child', 'Adult', or 'Senior' based on their age.
    - Create a new column `IsAlone` which is `True` if `FamilySize` is 0, otherwise `False`.
12. **Merging DataFrames**:
    - Create a separate dataframe with the columns `PassengerId` and `Survived`. Merge this dataframe back with the original dataframe on `PassengerId`.
13. **Pivot Table**:
    - Create a pivot table showing the average `Fare` paid for each combination of `Pclass` and `Embarked`.
14. **Crosstab**:
    - Create a crosstab showing the count of passengers who survived versus those who did not for each `Sex`.
15. **Advanced Grouping**:
    - Find the most common cabin letter (first character of `Cabin`) for each `Pclass`.

Iris Dataset Overview

The Iris dataset consists of 150 samples from three species of iris flowers: Setosa, Versicolor, and Virginica. It has the following columns:

- **sepal length (cm)**: Length of the sepal in centimeters.
- **sepal width (cm)**: Width of the sepal in centimeters.
- **petal length (cm)**: Length of the petal in centimeters.
- **petal width (cm)**: Width of the petal in centimeters.
- **target**: Numerical representation of species (0 = Setosa, 1 = Versicolor, 2 = Virginica).

Exercises with the Iris Dataset:

1. **Basic Data Exploration:**
    - Display the first 5 rows of the dataset.
    - Display the summary statistics for each numerical column.
    - Check for missing values in the dataset.
2. **Data Selection:**
    - Select the columns `sepal length (cm)` and `sepal width (cm)` and display the first 5 rows.
    - Display the rows where `petal length (cm)` is greater than 5.
3. **Data Filtering:**
    - Filter and display the rows where `target` is 1 and `sepal width (cm)` is less than 3.
    - Find all samples where the `sepal length (cm)` is greater than 6 and the species is not 'Setosa'.
4. **Aggregation and Grouping:**
    - Calculate the average `petal length (cm)` for each species.
    - Find the maximum `sepal width (cm)` for each species.
    - Get the count of samples for each species.
5. **Creating New Columns:**
    - Create a new column called `sepal_area` which is the product of `sepal length (cm)` and `sepal width (cm)`.
    - Create a new column `petal_area` which is the product of `petal length (cm)` and `petal width (cm)`.
6. **Handling Categorical Data:**

    - Replace the numerical values in the `target` column with the actual species names: 0 for 'Setosa', 1 for 'Versicolor', and 2 for 'Virginica'.
    - Create a new column `species_type` that categorizes the species as 'Small', 'Medium', or 'Large' based on the average `petal length (cm)`.
7. **Sorting Data:**
    - Sort the dataset by `sepal length (cm)` in ascending order.
    - Sort the dataset by `target` and then by `petal width (cm)` in descending order.
8. **Value Counts and Unique Values:**
    - Find the number of unique values in the `target` column.
    - Get the count of each unique value in the `sepal length (cm)` column.
9. **Visualization:**
    - Plot the distribution of sepal lengths.
    - Plot a scatter plot comparing `sepal length (cm)` and `sepal width (cm)` with different colors for each species.
10. **String Operations:**
    - Create a new column `species_code` which takes the first three letters of the species name.
11. **Conditional Columns:**
    - Create a new column `flower_size` that categorizes flowers as 'Small', 'Medium', or 'Large' based on the `petal length (cm)`.
12. **Statistical Analysis:**
    - Calculate the correlation matrix for the numerical columns.
    - Find the median `sepal width (cm)` for each species.
13. **Pivot Table:**
    - Create a pivot table showing the average `sepal length (cm)` for each combination of `target` and `sepal width (cm)`.
14. **Crosstab:**
    - Create a crosstab showing the count of each `target` for different `sepal length (cm)` ranges.
15. **Advanced Grouping:**
    - For each species, find the most common value for `sepal length (cm)`.
16. **Subsetting Data:**
    - Create a subset of the data containing only `sepal length (cm)`, `petal length (cm)`, and `target`.
    - Create a subset containing only rows where `sepal width (cm)` is greater than 3.

17. **Merging DataFrames:**
    - Split the dataset into two parts: one with only sepal-related columns and the other with petal-related columns. Merge them back together on the index.
18. **Conditional Indexing:**
    - Create a new dataframe containing only rows where the sum of `sepal length (cm)` and `sepal width (cm)` is greater than 10.
19. **Removing Duplicates:**
    - Remove duplicate rows from the dataset (if any).
20. **Data Imputation:**
    - Simulate missing values by setting 10 random values in the `sepal width (cm)` column to `NaN`. Then, fill these missing values with the column's mean.