COMP1002 – Advanced Python

Lab8

Assignment-1

(a) Create a DataFrame with the following data:

Product	Price	Quantity	Category
apple	25	18	Fruit
banana	30	25	Fruit
potato	10	30	Vegetable
orange	35	7	Fruit
broccoli	40	5	Vegetable
spinach	15	12	Vegetable
cherry	25	21	Fruit

(b) Indexing and selecting data:
- Select and print the '`Category`' column.
- Select and print the rows where the '`Quantity`' is greater than `20`.
- Select and print the row where the '`Product`' is '`orange`'.

(c) Add a new column named '`Total Price`'. This column should represent the total price for each product, calculated by multiplying the '`Price`' and '`Quantity`' columns.

(d) Calculate and display the average price of all products.

(e) Filter and display products that belong to the '`Fruit`' category and have a '`Total Price`' greater than `250`.

Filter Pandas Dataframe with multiple conditions:

```
import pandas as pd
# Sample DataFrame
data = {'A': [1, 2, 3, 4, 5],
        'B': [6, 7, 8, 9, 10],
        'C': [11, 12, 13, 14, 15]}
df = pd.DataFrame(data)

# Filtering with multiple conditions
filtered_df = df[(df['A'] > 2) & (df['B'] < 9)]
print(filtered_df)
```

In this example, we're filtering the DataFrame df based on two conditions:
- Values in column 'A' greater than 2.
- Values in column 'B' less than 9.

The conditions are combined using the & operator to perform element-wise logical AND operation. The resulting DataFrame filtered_df will contain only the rows that satisfy both conditions.

Assignment-2

(a) Read Data from CSV (CSV stands for comma-separated value) file:
- Read the population data from the CSV file named '*population_data.csv*' into a DataFrame.

(b) Filtering Rows:
- Filter and print the DataFrame to include only rows where the '`Country`' name starts with the letter 'T'.
- Filter and print the DataFrame to include only rows where the '`Continent`' is '`Asia`'.

(c) Applying Custom Functions:
- Define a custom function to calculate population density (Population/Area_km2) and apply this function to create a new column '`Population_Density`'.
  - The formula is:

$$Population\ Density\ =\ Population\ /\ Area\ (in\ square\ kilometers)$$

(d) Sorting:
- Sort and display the DataFrame by the '`Population`' column in descending order.
- Sort and display the DataFrame by the '`Population_Density`' column in ascending order.

(e) Grouping:
- Group the DataFrame by the '`Continent`' column and then calculate the total population for each continent. Display the DataFrame.

(f) Display the first 10 rows of the DataFrame.

(g) Display the last 10 rows of the DataFrame.

`head()` and `tail()` methods

`head()`: This method is used to view the first few rows of a DataFrame. By default, it displays the first 5 rows, but you can specify the number of rows you want to see by passing an argument. For example, df.head(10) will display the first 10 rows of the DataFrame df.

`tail()`: Conversely, the tail() method shows the last few rows of a DataFrame. By default, it displays the last 5 rows, but like head(), you can specify the number of rows you want to see by passing an argument. For instance, df.tail(7) will display the last 7 rows of the DataFrame df.