

Combo-Attack! Combining Multiple Data Manipulations

This document includes the question, the solution, and a breakdown of the code provided in the screenshot.

Uploaded Screenshot

Below is the screenshot of the task:

The screenshot shows a web browser window displaying a DataCamp exercise page. The page title is "Combo-attack! Combining Multiple Data Manipulations". The exercise instructions are as follows:

- You've seen the four most common types of data manipulation: sorting rows, subsetting columns, subsetting rows, and adding new columns. In a real-life data analysis, you can mix and match these four manipulations to answer a multitude of questions.
- In this exercise, you'll answer the question, "Which state has the highest number of homeless individuals per 10,000 people in the state?" Combine your new pandas skills to find out.

The instructions are followed by a list of steps:

- Add a column to `homelessness`, `indiv_per_10k`, containing the number of homeless individuals per ten thousand people in each state, using `state_pop` for state population.
- Subset rows where `indiv_per_10k` is higher than 20, assigning to `high_homelessness`.
- Sort `high_homelessness` by descending `indiv_per_10k`, assigning to `high_homelessness_srt`.
- Select only the `state` and `indiv_per_10k` columns of `high_homelessness_srt` and save as `result`. Look at the `result`.

Below the instructions is a "Take Hint (-30 XP)" button. To the right of the instructions is a code editor with the following Python code:

```
1 # Create indiv_per_10k col as homeless individuals per 10k state pop
2 homelessness["indiv_per_10k"] = 10000 * homelessness["state_pop"]
3
4 # Subset rows for indiv_per_10k greater than 20
5 high_homelessness = homelessness[indiv_per_10k > 20]
6
7 # Sort high_homelessness by descending indiv_per_10k
8 high_homelessness_srt = high_homelessness.sort_values("indiv_per_10k", ascending=False)
9
10 # From high_homelessness_srt, select the state and indiv_per_10k cols
11 result = high_homelessness_srt[["state", "indiv_per_10k"]]
12
13 # See the result
14 print(result)
```

Below the code editor is a "Run Code" button and a "Submit Answer" button. At the bottom of the page is a "Python Shell" section with a "In [1]:" prompt.

Question

1. Add a column to `homelessness`, `indiv_per_10k`, containing the number of homeless individuals per 10,000 people in each state, using `state_pop` for state population.
2. Subset rows where `indiv_per_10k` is higher than 20, assigning to `high_homelessness`.
3. Sort `high_homelessness` by descending `indiv_per_10k`, assigning to `high_homelessness_srt`.
4. Select only the `state` and `indiv_per_10k` columns of `high_homelessness_srt` and save as `result`. View the `result`.

Answer

```
# Create indiv_per_10k col as homeless individuals per 10k state pop
homelessness['indiv_per_10k'] = 10000 * homelessness['individuals'] /
homelessness['state_pop']
```

```
# Subset rows for indiv_per_10k greater than 20
high_homelessness = homelessness[homelessness['indiv_per_10k'] > 20]

# Sort high_homelessness by descending indiv_per_10k
high_homelessness_srt = high_homelessness.sort_values('indiv_per_10k',
ascending=False)

# From high_homelessness_srt, select the state and indiv_per_10k cols
result = high_homelessness_srt[['state', 'indiv_per_10k']]

# See the result
print(result)
```

Code Explanation

Explanation of the code:

1. `homelessness['indiv_per_10k'] = 10000 * homelessness['individuals'] / homelessness['state_pop']`: Adds a new column `indiv_per_10k` to the `homelessness` DataFrame, calculating the number of homeless individuals per 10,000 people in each state.
2. `homelessness[homelessness['indiv_per_10k'] > 20]`: Subsets the rows where `indiv_per_10k` is greater than 20 and assigns the result to `high_homelessness`.
3. `high_homelessness.sort_values('indiv_per_10k', ascending=False)`: Sorts the `high_homelessness` DataFrame in descending order based on the `indiv_per_10k` column and assigns it to `high_homelessness_srt`.
4. `high_homelessness_srt[['state', 'indiv_per_10k']]`: Selects the `state` and `indiv_per_10k` columns from the `high_homelessness_srt` DataFrame and assigns the result to `result`.
5. `print(result)`: Prints the resulting DataFrame to verify the calculations and filtering.