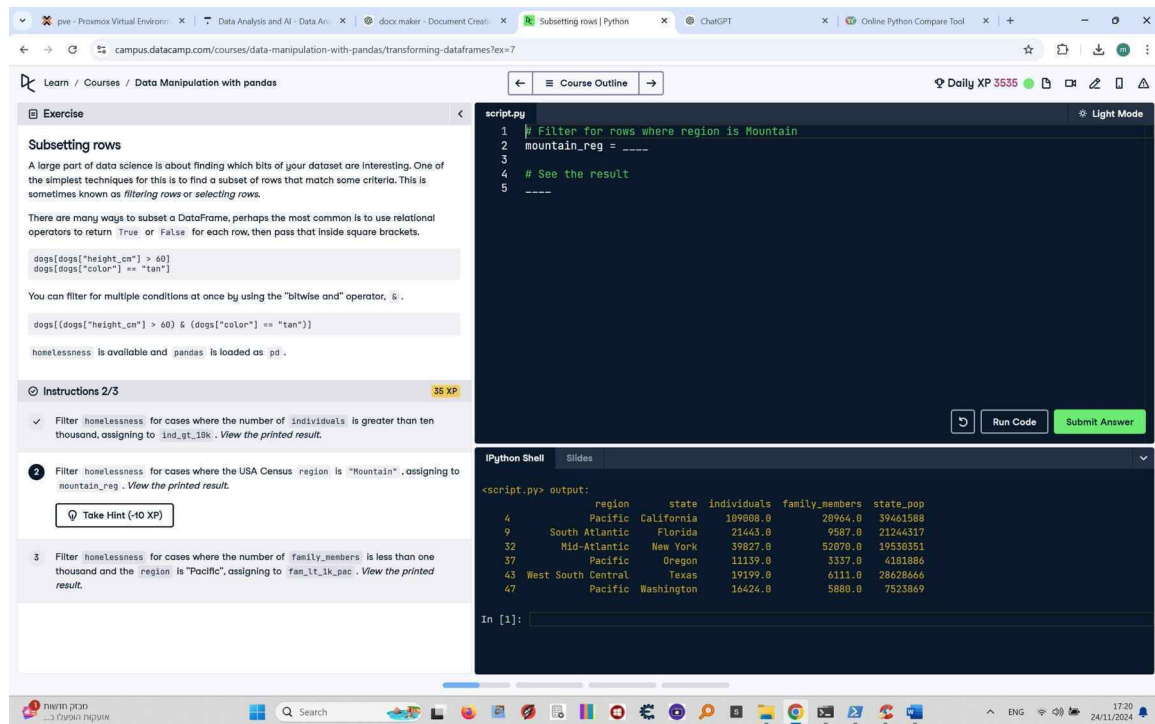


Subsetting Rows - Region is Mountain

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 titled "Subsetting rows". The exercise is part of a course on "Data Manipulation with pandas". The instructions are as follows:

Subsetting rows

A large part of data science is about finding which bits of your dataset are interesting. One of the simplest techniques for this is to find a subset of rows that match some criteria. This is sometimes known as *filtering rows* or *selecting rows*.

There are many ways to subset a DataFrame, perhaps the most common is to use relational operators to return `True` or `False` for each row, then pass that inside square brackets.

```
dogs[dogs["height_cm"] > 60]
dogs[dogs["color"] == "tan"]
```

You can filter for multiple conditions at once by using the "bitwise and" operator, `&`.

```
dogs[(dogs["height_cm"] > 60) & (dogs["color"] == "tan")]
```

`homelessness` is available and `pandas` is loaded as `pd`.

Instructions 2/3 (35 XP)

- Filter `homelessness` for cases where the number of `individuals` is greater than ten thousand, assigning to `ind_gt_10k`. View the printed result.
- Filter `homelessness` for cases where the USA Census region is "Mountain", assigning to `mountain_reg`. View the printed result.
- Filter `homelessness` for cases where the number of `family_members` is less than one thousand and the region is "Pacific", assigning to `fam_lt_1k_pac`. View the printed result.

The solution code is provided in a script editor:

```
script.py
1 # Filter for rows where region is Mountain
2 mountain_reg = ____
3
4 # See the result
5 ____
```

The output of the script is shown in the "iPython Shell" window:

```
<script.py> output:
   region      state  individuals  family_members  state_pop
4    Pacific  California    109088.0         20964.0    39461588
9  South Atlantic    Florida     21443.0         9587.0    21244317
32 Mid-Atlantic    New York     39827.0        52070.0    19530351
37    Pacific    Oregon     11139.0         3337.0     4181886
43 West South Central    Texas     19199.0        6111.0    28628666
47    Pacific    Washington    16424.0         5080.0     7523869
```

Question

Filter `homelessness` for cases where the region is "Mountain", assigning the result to `mountain_reg`. View the printed result.

Answer

```
# Filter for rows where region is Mountain
mountain_reg = homelessness[homelessness["region"] == "Mountain"]
```

```
# See the result
print(mountain_reg)
```

Code Explanation

Explanation of the code:

1. ``homelessness["region"] == "Mountain"``: This creates a boolean mask, where each row evaluates to ``True`` if the value in the ``region`` column matches "Mountain", and ``False`` otherwise.
2. ``homelessness[homelessness["region"] == "Mountain"]``: Filters the ``homelessness`` DataFrame by keeping only rows where the mask is ``True``.
3. ``mountain_reg``: Stores the filtered DataFrame for further use.
4. ``print(mountain_reg)``: Prints the resulting DataFrame to verify the filter.