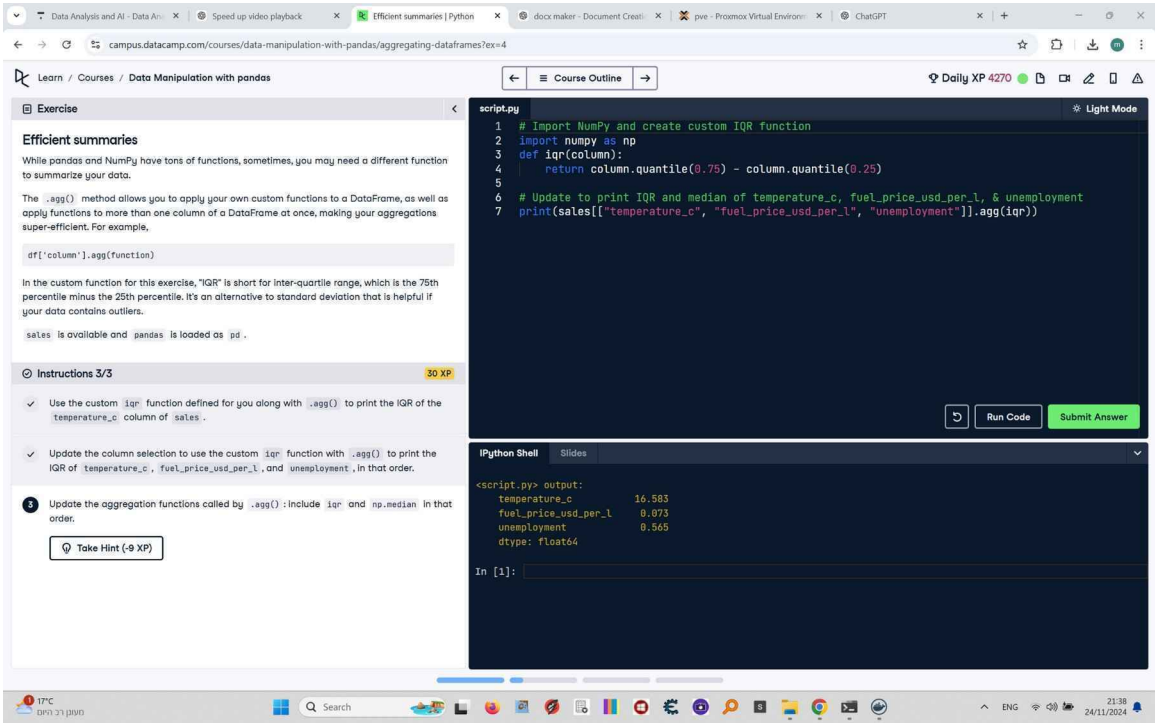
Efficient Summaries with Custom Functions (Final Version)

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:



Question

1. Use the custom `iqr` function defined for you along with `.agg()` to print the IQR of the `temperature_c` column of `sales`.
2. Update the column selection to use the custom `iqr` function with `.agg()` to print the IQR of `temperature_c`, `fuel_price_usd_per_l`, and `unemployment`, in that order.
3. Update the aggregation functions called by `.agg()`: include `iqr` and `np.median` in that order.

Answer

```
# Import NumPy
import numpy as np

# A custom IQR function
def iqr(column):
```

```
    return column.quantile(0.75) - column.quantile(0.25)

# Print IQR and median of temperature_c, fuel_price_usd_per_l, and unemployment
print(sales[['temperature_c', 'fuel_price_usd_per_l', 'unemployment']].agg([iqr, np.median]))
```

Code Explanation

Explanation of the code:

1. `import numpy as np`: Imports the NumPy library, which is necessary to use the `np.median` function.

2. `def iqr(column):`: Defines a custom function `iqr` to calculate the interquartile range (IQR) of a column by subtracting the 25th percentile from the 75th percentile.

3. `sales[['temperature_c', 'fuel_price_usd_per_l', 'unemployment']].agg([iqr, np.median])`: Applies both the `iqr` and `np.median` aggregation functions to the selected columns (`temperature_c`, `fuel_price_usd_per_l`, and `unemployment`) in the `sales` DataFrame and prints the results in the specified order.