

Learn / Courses / Python Toolbox
Course Outline
Daily XP 1750

### Exercise

#### Writing an iterator to load data in chunks (4)

In the previous exercises, you've only processed the data from the first DataFrame chunk. This time, you will aggregate the results over all the DataFrame chunks in the dataset. This basically means you will be processing the entire dataset now. This is neat because you're going to be able to process the entire large dataset by just working on smaller pieces of it!

You're going to use the data from 'ind\_pop\_data.csv', available in your current directory. The packages `pandas` and `matplotlib.pyplot` have been imported as `pd` and `plt` respectively for your use.

Instructions
100 XP

- Initialize an empty DataFrame `data` using `pd.DataFrame()`.
- In the `for` loop, iterate over `urb_pop_reader` to be able to process all the DataFrame chunks in the dataset.
- Concatenate `data` and `df_pop_ceb` by passing a list of the DataFrames to `pd.concat()`.

Take Hint (-30 XP)

```

1 # Initialize reader object: urb_pop_reader
2 urb_pop_reader = pd.read_csv('ind_pop_data.csv', chunksize=1000)
3
4 # Initialize empty DataFrame: data
5 data = pd.DataFrame()
6
7 # Iterate over each DataFrame chunk
8 for df_urb_pop in urb_pop_reader:
9
10     # Check out specific country: df_pop_ceb
11     df_pop_ceb = df_urb_pop[df_urb_pop['CountryCode'] == 'CEB']
12
13     # Zip DataFrame columns of interest: pops
14     pops = zip(df_pop_ceb['Total Population'],
15               df_pop_ceb['Urban population (% of total)'])
16
17     # Turn zip object into list: pops_list
18     pops_list = list(pops)
19
20     # Use list comprehension to create new DataFrame column 'Total Urban Population'
21     df_pop_ceb['Total Urban Population'] = [int(tup[0] * tup[1] * 0.01) for tup in pops_list]
22
23     # Concatenate DataFrame chunk to the end of data: data
24     data = pd.concat([data, df_pop_ceb])
25
26 # Plot urban population data
27 data.plot(kind='scatter', x='Year', y='Total Urban Population')
28 plt.show()

```

Run Code
Submit Answer

iPython Shell
Slides

In [1]:

Question: Writing an iterator to load data in chunks (4)

Correct Answer and Explanation:

Code Implementation:

```
# Initialize reader object: urb_pop_reader
urb_pop_reader = pd.read_csv('ind_pop_data.csv', chunksize=1000)
```

```
# Initialize empty DataFrame: data
data = pd.DataFrame()
```

```
# Iterate over each DataFrame chunk
for df_urb_pop in urb_pop_reader:
```

```
    # Check out specific country: df_pop_ceb
    df_pop_ceb = df_urb_pop[df_urb_pop['CountryCode'] == 'CEB']
```

```
    # Zip DataFrame columns of interest: pops
    pops = zip(df_pop_ceb['Total Population'],
               df_pop_ceb['Urban population (% of total)'])
```

```

# Turn zip object into list: pops_list
pops_list = list(pops)

# Use list comprehension to create new DataFrame column 'Total Urban
Population'
df_pop_ceb['Total Urban Population'] = [int(tup[0] * tup[1] * 0.01) for tup
in pops_list]

# Concatenate DataFrame chunk to the end of data: data
data = pd.concat([data, df_pop_ceb])

# Plot urban population data
data.plot(kind='scatter', x='Year', y='Total Urban Population')
plt.show()

```

Explanation:

1. ``urb_pop_reader = pd.read_csv('ind_pop_data.csv', chunksize=1000)``:  
- Reads the file ``ind_pop_data.csv`` in chunks of 1000 rows and assigns the iterator to ``urb_pop_reader``.
2. ``data = pd.DataFrame()``:  
- Initializes an empty DataFrame ``data`` to store processed chunks.
3. ``for df_urb_pop in urb_pop_reader:``:  
- Iterates over each chunk of data provided by ``urb_pop_reader``.
4. ``df_pop_ceb = df_urb_pop[df_urb_pop['CountryCode'] == 'CEB']``:  
- Filters rows where the ``CountryCode`` column equals 'CEB'.
5. ``pops = zip(df_pop_ceb['Total Population'], df_pop_ceb['Urban population (% of total)'])``:  
- Zips the ``Total Population`` and ``Urban population (% of total)`` columns together to create an iterable of tuples.
6. ``pops_list = list(pops)``:  
- Converts the zipped object into a list of tuples.
7. ``df_pop_ceb['Total Urban Population'] = [int(tup[0] * tup[1] * 0.01) for tup in pops_list]``:  
- Uses a list comprehension to calculate the 'Total Urban Population' for each tuple in ``pops_list``. The calculation multiplies the ``Total Population`` by the percentage (converted to a fraction) and converts the result to an integer.

8. ``data = pd.concat([data, df_pop_ceb])``:
  - Appends the processed chunk ``df_pop_ceb`` to the ``data`` DataFrame.
9. ``data.plot(kind='scatter', x='Year', y='Total Urban Population')``:
  - Plots the urban population data as a scatter plot with 'Year' on the x-axis and 'Total Urban Population' on the y-axis.
10. ``plt.show()``:
  - Displays the scatter plot.