Just IT

B2W group

Apprenticeships | Training | Recruitment

Data Technician

Name:
Course Date:

Table of contents

Day 2: Task 1

It is a common software development interview question to create the below with a certain programming language. Create the below using Python syntax, test it and past the completed syntax and output below.

FizzBuzz:

Go through the integers from 1 to 100.
If a number is divisible by 3, print "fizz."
If a number is divisible by 5, print "buzz."
If a number is both divisible by 3 and by 5, print "fizzbuzz."
Otherwise, print just the number.

Paste your completed work to the right

```python
# Go through integers from 1 to 100
for x in range(1, 101):
    # if number divisible by both 3 and 5, print "fizzbuzz"
    if x % 3 == 0 and x % 5 == 0:
        print("fizzbuzz")
    # if number is divisible by 3, print "fizz"
    elif x % 3 == 0:
        print("fizz")
    # if number is divisible by 5, print "buzz"
    elif x % 5 == 0:
        print("buzz")
    # Otherwise, print just the number
    else:
        print(x , end= " ")
```

```
1 2 fizz
4 buzz
fizz
7 8 fizz
buzz
11 fizz
13 14 fizzbuzz
16 17 fizz
19 buzz
fizz
22 23 fizz
buzz
26 fizz
28 29 fizzbuzz
31 32 fizz
34 buzz
fizz
37 38 fizz
buzz
41 fizz
43 44 fizzbuzz
46 47 fizz
49 buzz
fizz
52 53 fizz
buzz
56 fizz
58 59 fizzbuzz
61 62 fizz
64 buzz
fizz
67 68 fizz
buzz
71 fizz
73 74 fizzbuzz
76 77 fizz
79 buzz
fizz
82 83 fizz
buzz
86 fizz
88 89 fizzbuzz
91 92 fizz
94 buzz
fizz
97 98 fizz
buzz
```

Day 3: Task 1

Using the 'student.csv' which can be downloaded here, complete the below exercises and paste your input and output.

Exercise 1: Loading and Exploring the Data

1. Question: "Write the code to read a CSV file into a Pandas DataFrame."
2. Question: "Write the code to display the first 5 rows of the DataFrame."
3. Question: "Write the code to get the information about the DataFrame."
4. Question: "Write the code to get summary statistics for the DataFrame."

```
1. df = pd.read_csv('student.csv')
2. df.head(5)
3. df.info()
4. df.describe()
```

Exercise 2: Indexing and Slicing

1. Question: "Write the code to select the 'name' column."
2. Question: "Write the code to select the 'name' and 'mark' columns."
3. Question: "Write the code to select the first 3 rows."
4. Question: "Write the code to select all rows where the 'class' is 'Four'."

```
1. df["name"]
2. df[["name", "mark"]]
3. df.head(3)
4. df[df['class'] == 'Four']
```

Exercise 3: Data Manipulation

1. Question: "Write the code to add a new column 'passed' that indicates whether the student passed (mark >= 60)."
2. Question: "Write the code to rename the 'mark' column to 'score'."
3. Question: "Write the code to drop the 'passed' column."

```
1. df['passed'] = df['mark'] >= 60
2. df.rename(columns={'mark': 'score'}, inplace=True)
3. df_new = df.drop(columns=['passed'])
```

Day 4: Task 1

Using the 'GDP (nominal) per Capita.csv' which can be downloaded here, complete the below exercises and paste your input and output. Work individually, but we will work and support each other in the room.

- Read and save the 'GDP (nominal) per Capita' data to a data frame called "df" in Jupyter notebook
- Print the first 10 rows
- Print the last 5 rows
- Print 'Country/Territory' and 'UN_Region' columns

```
[1] import pandas as pd
```

```
[6] from google.colab import files

    uploaded = files.upload()
```

```
Choose files GDP (nomi…r Capita.xlsx
```

- **GDP (nominal) per Capita.xlsx**(application/vnd.openxmlformats-officedocument.spreadsheetml.sl
- 100% done

```
Saving GDP (nominal) per Capita.xlsx to GDP (nominal) per Capita (1).xlsx
```

```
[10] df = pd.read_excel('GDP (nominal) per Capita (1).xlsx')
```

```
# First 10 rows
print(df.head(10))
```

```
   Unnamed: 0 Country/Territory UN_Region  IMF_Estimate  IMF_Year
0           1            Monaco    Europe             0         0
1           2     Liechtenstein    Europe             0         0
2           3        Luxembourg    Europe        132372      2023
3           4           Ireland    Europe        114581      2023
4           5           Bermuda  Americas             0         0
5           6            Norway    Europe        101103      2023
6           7       Switzerland    Europe         98767      2023
7           8         Singapore      Asia         91100      2023
8           9      Isle of Man    Europe             0         0
9          10    Cayman Islands  Americas             0         0

   WorldBank_Estimate  WorldBank_Year  UN_Estimate UN_Year
0              234316            2021       234317    2021
1              157755            2020       169260    2021
2              133590            2021       133745    2021
3              100172            2021       101109    2021
4              114090            2021       112653    2021
5               89154            2021        89242    2021
6               91992            2021        93525    2021
7               72794            2021        66822    2021
8               87158            2019            0       0
9               86569            2021        85250    2021
```

```
# Last 5 rows
print(df.tail(5))
```

```
     Unnamed: 0 Country/Territory UN_Region  IMF_Estimate  IMF_Year
218         219            Malawi    Africa           496      2023
219         220       South Sudan    Africa           467      2023
220         221      Sierra Leone    Africa           415      2023
221         222       Afghanistan      Asia           611      2020
222         223           Burundi    Africa           249      2023

     WorldBank_Estimate  WorldBank_Year  UN_Estimate UN_Year
218                 635            2021          613    2021
219                1072            2015          400    2021
220                 480            2021          505    2021
221                 369            2021          373    2021
222                 222            2021          311    2021
```

```
[16] # Specific columns
     print(df[['Country/Territory', 'UN_Region']])
```

```
     Country/Territory UN_Region
0              Monaco    Europe
1       Liechtenstein    Europe
2          Luxembourg    Europe
3             Ireland    Europe
4             Bermuda  Americas
..                ...       ...
218           Malawi    Africa
219       South Sudan    Africa
220      Sierra Leone    Africa
221       Afghanistan      Asia
222           Burundi    Africa

[223 rows x 2 columns]
```

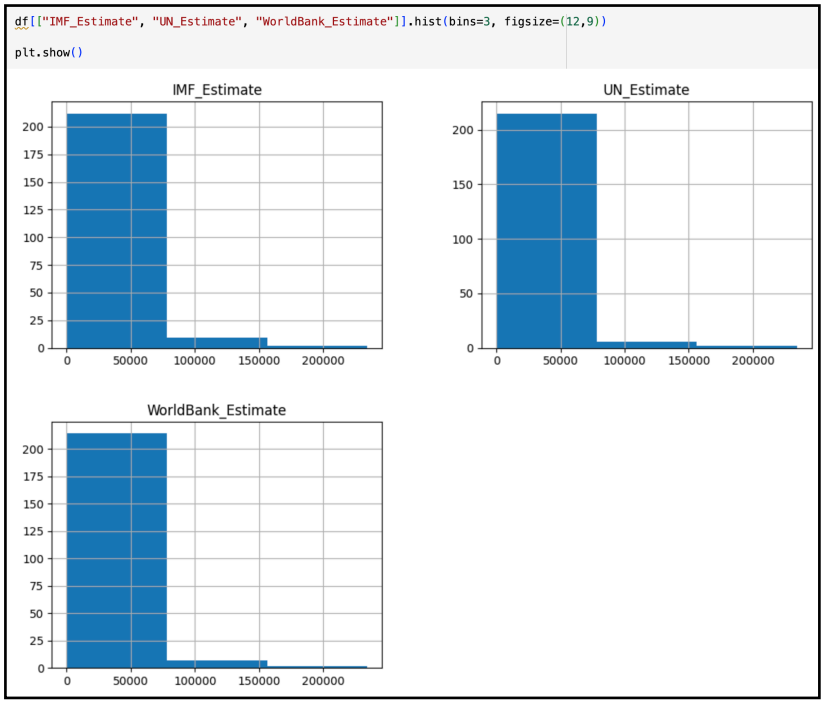
Day 4: Task 2

Back with ‘GDP (nominal) per Capita’. As a group, import and work your way through the Day_4_Python_Activity.ipynb notebook which can be found here. There are questions to answer, but also opportunities to have fun with the data – paste your input and output below.

Once complete, and again as a group, work with some more data and have some fun –there is no set agenda for this section, other than to embed the skills developed this week. Paste your input and output below and upon return we’ll discuss progress made.

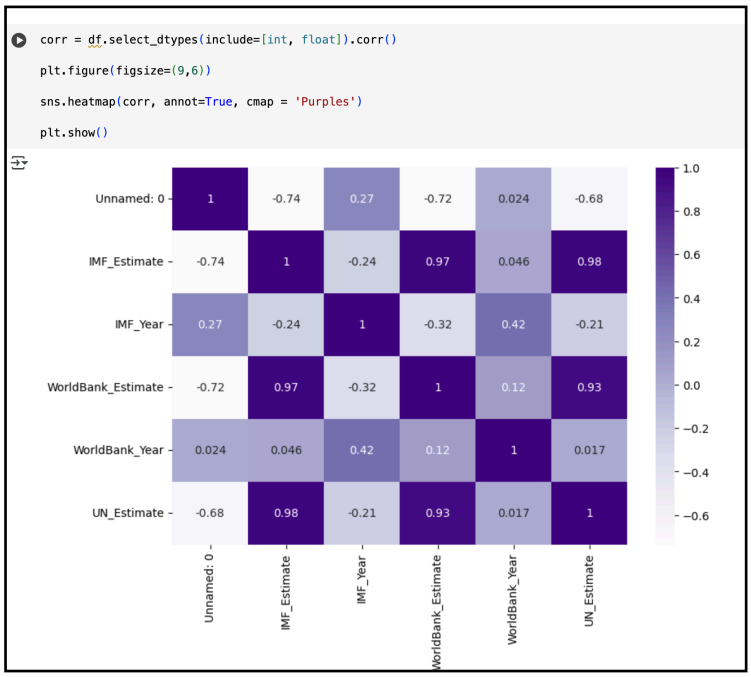
Additional data found here.

Histogram created in Google Colab, the code df[["IMF_Estimate", "UN_Estimate", "WorldBank_Estimate"]].hist(bins=3, fig-size=(12,9)) followed by plt.show() is a quick and effective way to visualise and compare the distribution of GDP per capita estimates from different institutions. Using just one line, it generates side-by-side histograms, making it easy to identify general patterns, trends, or anomalies across the three sources. Setting bins=3 provides a high-level view by grouping the data into broad categories, while the larger fig-size ensures clarity and readability. This approach is especially useful during exploratory data analysis to spot outliers or inconsistencies.



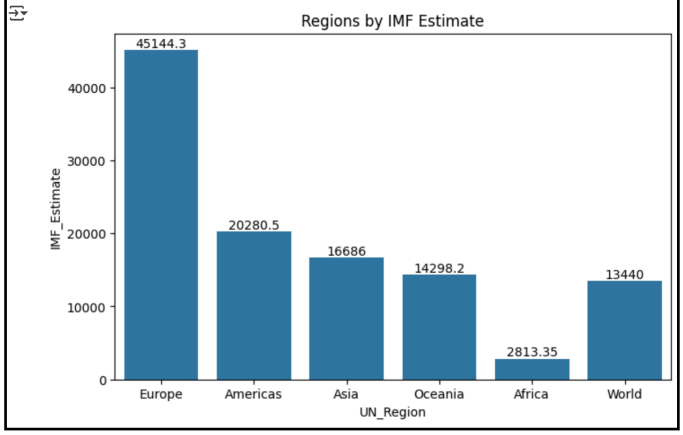
The code block calculates the correlation matrix for all numerical columns in the dataset using df.select_dtypes(include=[int, float]).corr(), and then visualises it with a heat map. This allows for a clear and concise overview of how different numerical variables relate to each other. The heat map, created with sns.heatmap(), uses the 'Purples' colour map and includes annotations to display exact correlation values, making it easier to interpret. The fig-size=(9,6) ensures the plot is well-sized for readability. This is especially useful in exploratory data analysis to quickly identify strong positive or negative relationships between variables.

```
corr = df.select_dtypes(include=[int, float]).corr()
plt.figure(figsize=(9,6))
sns.heatmap(corr, annot=True, cmap = 'Purples')
plt.show()
```



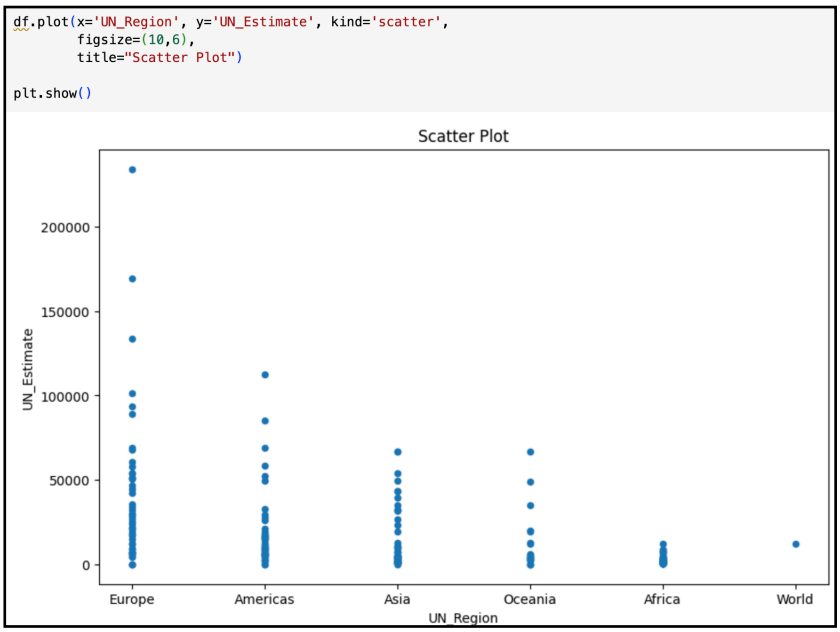
Used a bar plot to visualise average IMF GDP per capita estimates by UN region. Using sns.barplot(), it groups the data by "UN_Region" and plots the corresponding "IMF_Estimate" values without error bars, providing a clean comparison across regions. The figure size (8,5) ensures the plot is easy to read, while ax.bar_label(ax.containers[0]) adds value labels on top of each bar for clarity. The chart is titled “Regions by IMF Estimate,” making it an effective and visually appealing way to compare regional economic performance during exploratory data analysis.

```
fig = plt.figure(figsize = (8,5))
ax = sns.barplot(x = "UN_Region",  y = "IMF_Estimate",
                 data = df, errorbar = None)

ax.bar_label(ax.containers[0])


ax.set_title("Regions by IMF Estimate")
plt.show()
```



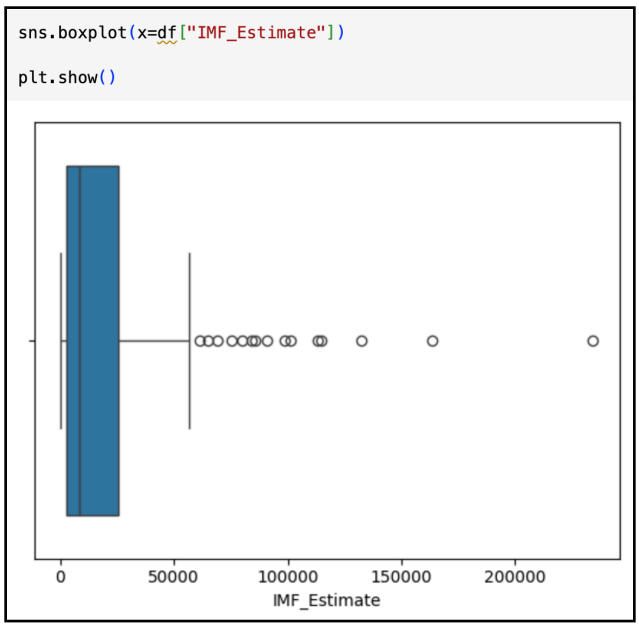
Made a scatter plot using df.plot() to visualise the relationship between UN regions and their corresponding UN GDP per capita estimates. By setting kind='scatter', it highlights the spread and variation of estimates across regions, which can help identify patterns or outliers. The fig-size=(10,6) ensures the plot is large enough for clear interpretation, and the title="Scatter Plot" adds context. This is a simple yet effective tool for visually examining regional disparities in UN GDP estimates during exploratory data analysis.

```
df.plot(x='UN_Region', y='UN_Estimate', kind='scatter',
        figsize=(10,6),
        title="Scatter Plot")

plt.show()
```



Created a box plot of the "IMF_Estimate" column using sns.boxplot(), which is useful for visualising the distribution, central tendency, and spread of the data. It highlights key statistics such as the median, quartiles, and potential outliers, making it easier to detect skewness or anomalies in GDP per capita estimates. This type of plot is especially helpful during exploratory data analysis for quickly assessing the variability and detecting extreme values in a single numerical variable.

```
sns.boxplot(x=df["IMF_Estimate"])

plt.show()
```



Course Notes

It is recommended to take notes from the course, use the space below to do so, or use the revision guide shared with the class:

We have included a range of additional links to further resources and information that you may find useful, these can be found within your revision guide.

END OF WORKBOOK

Please check through your work thoroughly before submitting and update the table of contents if required.

Please send your completed work booklet to your trainer.