Quantitative Data Analysis – Exercises

(Week 02)

In these exercises, we will learn how to work with different data types and data structures in Python. We will also learn how to read & write data from/to a relational data base and files. Therefore, we will use data from the use cases from week 01 of this course. In the data analysis process model, these exercises cover part of the step "Preparing & storing" data (see figure 1). Results of the exercises must be uploaded as separate files (no .zip files) by each student on Moodle. Details on how to submit the results can be found in the tasks below.

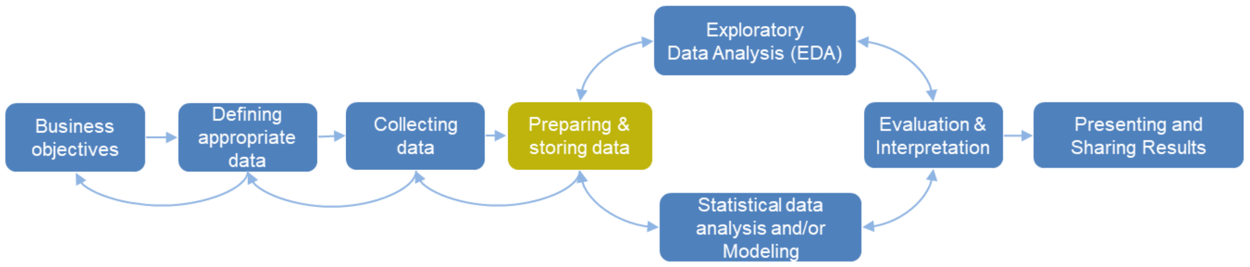
Business objectives
Defining appropriate data
Collecting data
Preparing & storing data
Exploratory Data Analysis (EDA)
Statistical data analysis and/or Modeling
Evaluation & Interpretation
Presenting and Sharing Results

Figure 1: Data analysis process model (see slides of week 01)

Task 1

In this exercise, you will learn to identify parts of strings using regular expressions (regex). A regular expression is a sequence of characters that specifies a search pattern in text. In data analytics, regex is useful because we often need numeric values (like the price, number of rooms or area of an apartment) which are often embedded in strings in the raw data. To test regular expressions, go to the webpage: https://regex101.com. Include the following string in the "TEST STRING" input-field of the website:

```
4,5 Zimmer, 108 m², CHF 3160.–
```

Use the regular expressions below and the "REGULAR EXPRESSION" input-field on https://regex101.com to search the string for the numerical value of the rental apartment price i.e. 3160 (without 'CHF' and '.-'). Note that two of the regular expressions below can do the job: one with a group (highlighted in green) and one without a group (highlighted in blue). Groups group multiple patterns as a whole, and capturing groups provide extra sub-match information when using a regular expression pattern to match against a string.

```
[0-9]
[0-9]+
\d+
[0-9]{4}
[^CHF, Zimmer m²]
[CHF, Zimmer m²]
(.*) Zimmer
Zimmer, (.*) m²
CHF (.*).{2}$
```

To be submitted on Moodle: Screenshot of one regular expression on https://regex101.com which matches the rental price as numerical value.

Task 2

In this exercise, you will learn how to read, prepare, and store the rental apartment data from week 01 using Python. On Moodle you can find rental apartment data covering the city of Winterthur and data covering the canton of Zurich. The tasks are:

a) In Visual Studio Code open the file 'apartments_data_winterthur.csv'.
b) Look which column separator is used in the file.
c) Check the column names for completeness.
d) Check the number of rows for completeness.
e) Open a new Terminal and use the command: **`file -i apartments_data_winterthur.csv`** to get the file format and encoding.
f) Run the Jupyter notebook 'apartments_data_preparation_winterthur.ipynb' step by step.
g) Make a copy of the Jupyter notebook and run it with the data from the canton of Zuerich available in the file 'apartments_data_zuerich.csv'.
h) Use the Jupyter notebook to check, whether there are missing or duplicated values after the number of rooms, area and price were extracted.
i) Save the copy as 'apartments_data_preparation_zuerich.ipynb'.
j) Save the prepared data as 'apartments_data_prepared.csv'.

To be submitted on Moodle:

a) Export of the Jupyter notebook containing the data from the canton of Zuerich as html-file ('apartments_data_preparation_zuerich.html').
b) Prepared data from the canton of Zuerich as 'apartments_data_prepared.csv'.

Task 3

In this exercise, you will learn how to read, prepare, and store the supermarket data from week 01. The tasks are:

a) Open the file ‘supermarkets.json’ with a json-formatter like https://jsonformatter.org.
b) A json-file has key:value pairs. See which key:value pairs are available.
c) Run the Jupyter notebook ‘supermarkets_data_preparation.ipynb’ step by step.
d) Extent the Jupyter Notebook with a section ‘Additional filters on supermarkets’.
e) The .loc method from the pandas library can be used to filter data frames, e.g.:

```
df_filtered = df.loc[(df['brand'] == 'Coop') & (df['city'] == 'Zürich')]
```

f) Filter only Migros supermarkets in the city of Zürich.
g) Filter and count all Coop supermarkets in the cities of Zürich, Basel & Bern.
h) Filter supermarkets with available brand, city, house number and postcode.
i) Include opening hours as additional variable in the data frame.
j) Filter supermarkets with available opening hours.
k) Save the Jupyter notebook with the filter as ‘supermarkets_data.ipynb’.

To be submitted on Moodle:

a) Export of the Jupyter notebook as html-file (‘supermarkets_data_preparation.html’) including the additional filters.

Task 4

In this exercise, you will learn how to store data in a database and how to query a table in a database using Python. The tasks are:

a) Run the Jupyter notebook ‘apartments_database_Python.ipynb’ step by step. Input data is the prepared apartment data from the canton of Zuerich from task 2.
b) In the Jupyter notebook, include a section ‘Additional SQL-queries’.
c) Write an SQL-query to filter all apartments with >= 4.0 rooms and where the area is >= 100m2. An example SQL query can be found in the Jupyter notebook.
d) Write an SQL-query to calculate the average price per room size, i.e. 1.0, 1.5, 2.5 …. This can be done using the AVG and GROUP BY statements:

```
'''SELECT rooms, AVG(price)
   FROM apartments_table
   GROUP BY rooms'''
```

```
➔ Note that the output table contains only two columns!
```

e) Write an SQL-query to calculate the average area per room size.
f) Use the Jupyter notebook for the apartments as template to create a data base for the supermarket data.
g) In the latter Jupyter notebook, write an SQL-query to filter all supermarkets in the city of Winterthur.
h) Save the Jupyter notebook as ‘supermarkets_database_Python.ipynb’.

To be submitted on Moodle:

a) Export of the Jupyter notebook 'apartments_database_Python.ipynb' as html-file (it must include the additional SQL-queries).
b) Export of the Jupyter notebook 'supermarkets_database_Python.ipynb' as html-file.