Description

Group Assignment 3 is due on Wednesday, May 12, 2021 at 12:00. The deadline is strict. The first part will result in a .twbx file and the second part in a .ipynb file. Please submit these two files via Canvas. For this assignment, you will receive a *grade* between 1 and 10.

In this assignment, you will set up two databases, named `Yelp` and `CoronaImpactEU`, based on data made available by Yelp, Eurostat, and the ECDC. Yelp publishes crowd-sourced reviews about businesses such as restaurants and other venues. Eurostat provides statistical information to EU institutions. **Warning!** The Yelp comprises over a million reviews and nearly four million checkins. Therefore, downloading and importing the data may take a long time. Please take this into account when scheduling your work on this assignment!

In Part I, you will use `CoronaImpactEU`, containing data on vaccinations and youth unemployment in the EU over the course of the covid pandamic, for visualisations in Tableau. In Part II, you will use `Yelp`, comprising much data on reviews of businesses, for several statistical analyses in Python.

Database installation

Installation

Please download the following files from Canvas, found under E_EOR2_DBFA > Files > Assignments:

1. `E_EOR2_DBFA.GA3.DATA.YELP.zip`
2. `E_EOR2_DBFA.GA3.DATA.CORONA.zip`

Once you have downloaded these two files, go over the following steps:

1. Unzip these files in two separate folders of your choosing, and let's refer to those folders as 'YELP' and 'CORONA' for brevity.
2. Open MySQL Workbench and open a connection to your server (Local Instance).
3. In the Server menu, click on 'Data Import' as seen in Figure 1.
4. In the interface that now opens (in line with Figure 2):
   (a) select the 'YELP' folder,
   (b) then make sure the checkbox for 'Yelp' is ticked, and
   (c) make sure 'Dump Stucture and Data' is selected at the bottom of the interface.
5. Next, go to the 'Import Progress' tab, and press the 'Start Import' button. **Warning!** For the Yelp data, this step can easily take up to half an hour.
6. When the import has completed, close the 'Data Import' interface.
7. If you now refresh your 'SCHEMAS', you should be able to see the 'Yelp' database, with tables 'Businesses', 'CheckIns', 'Reviews', and 'Users', as seen in Figure 3.

Repeat the same procedure, but now for the data in the ‘CORONA’ folder (which should be imported in at most a few seconds, as it is much smaller). If you now refresh your ‘SCHEMAS’, you should be able to see the ‘CoronaImpactEU’ database, with tables ‘geos’, ‘vaccinationrollout’, ‘vaccines’, and ‘youthunemployment’, as seen in Figure 4.

Installation check

Use the `COUNT` operator in MySQL Workbench (e.g. using statements like `SELECT COUNT(*) FROM R;`), to make sure that the installation of the databases was successful. Use Table 1 to check whether you have the same number of rows in each database table. If your counts are the same, you can assume that the data have been imported correctly.

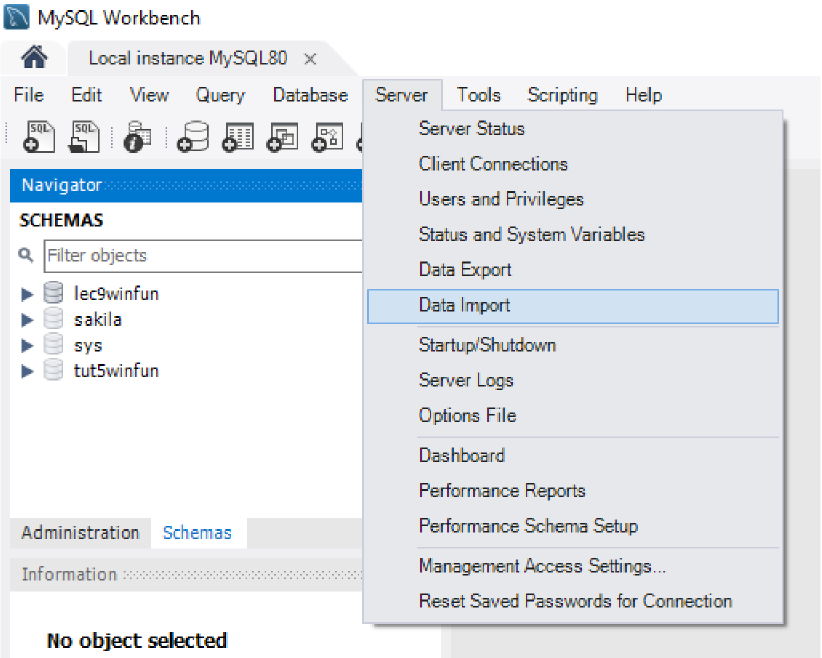
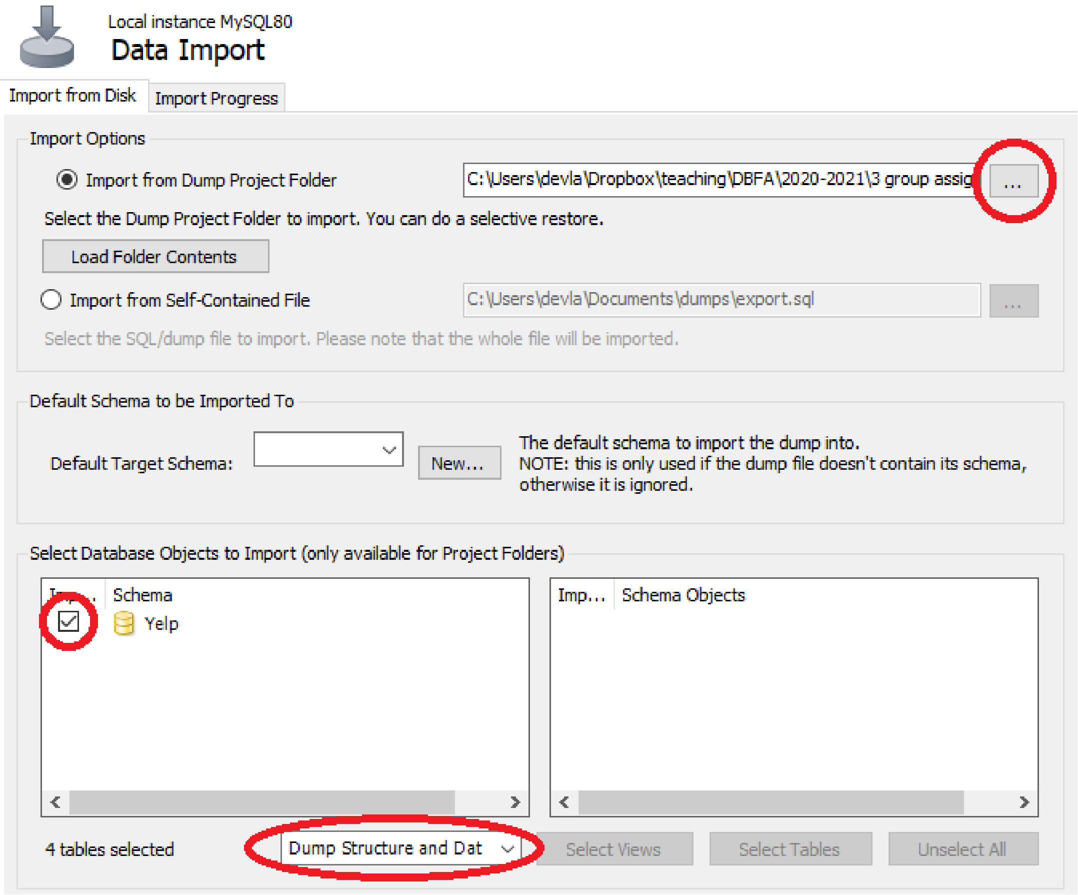


Figure 1. Opening the Data Import interface in MySQL Workbench.

Table	Row count
CoronaImpactEU.geos	34
CoronaImpactEU.vaccinationrollout	966
CoronaImpactEU.vaccines	6
CoronaImpactEU.youthunemployment	807
Yelp.Businesses	209,393
Yelp.CheckIns	3,750,000
Yelp.Reviews	1,490,000
Yelp.Users	1,968,703

Table 1. Row counts of tables in `CoronaImpact` and `Yelp`.



Local instance MySQL80

Data Import

Import from Disk | Import Progress

Import Options

● Import from Dump Project Folder C:\Users\devla\Dropbox\teaching\DBFA\2020-2021\3 group assig ...

Select the Dump Project Folder to import. You can do a selective restore.

Load Folder Contents

○ Import from Self-Contained File C:\Users\devla\Documents\dumps\export.sql ...

Select the SQL/dump file to import. Please note that the whole file will be imported.

Default Schema to be Imported To

Default Target Schema: New...

The default schema to import the dump into.
NOTE: this is only used if the dump file doesn't contain its schema,
otherwise it is ignored.

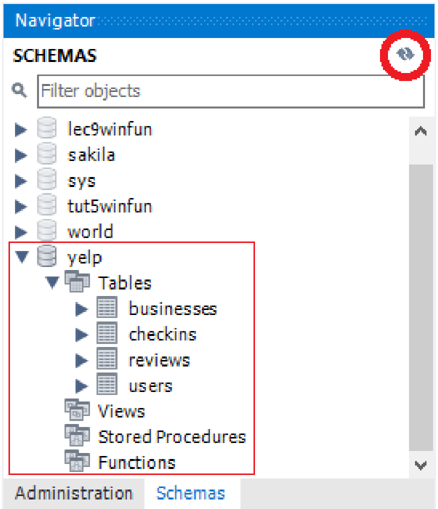
Select Database Objects to Import (only available for Project Folders)

Imp...	Schema
☑	Yelp

Imp...	Schema Objects

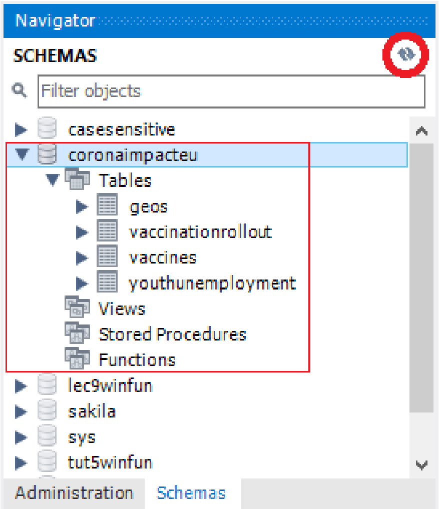
4 tables selected Dump Structure and Dat Select Views Select Tables Unselect All

Figure 2. Data Import interface after navigating to Yelp folder and ticking the checkbox.



```
Navigator
SCHEMAS
Filter objects
▶ lec9winfun
▶ sakila
▶ sys
▶ tut5winfun
▶ world
▼ yelp
  ▼ Tables
    ▶ businesses
    ▶ checkins
    ▶ reviews
    ▶ users
    Views
    Stored Procedures
    Functions
Administration  Schemas
```

Figure 3. Schemas after having imported the Yelp data.



```
Navigator
SCHEMAS
Filter objects
▶ casesensitive
▼ coronaimpacteu
  ▼ Tables
    ▶ geos
    ▶ vaccinationrollout
    ▶ vaccines
    ▶ youthunemployment
    Views
    Stored Procedures
    Functions
▶ lec9winfun
▶ sakila
▶ sys
▶ tut5winfun
Administration  Schemas
```

Figure 4. Schemas after having imported the Eurostat data.

Part I. Tableau (5 points)

Introduction

In this part of the assignment, you will create a number of visualisations in Tableau, to show the impact COVID-19 has had on youth unemployment in the EU, and how vaccinations are progressing in the various member states. `CoronaImpactEU` has four tables that contain information related to employment[1] and vaccination roll-out[2]. You can find the documentation of the database at the end of this document.

As `CoronaImpactEU` is relatively small, Tableau is a great tool for extracting insight from data while writing little to no SQL statements. Bigger datasets (such as the Yelp data[3]) require more sophisticated data manipulation, which can be cumbersome in Tableau.

In this part of the assignment, you will gain further experience with Tableau. The challenge is to figure out how to create desired visualisations. You are encouraged to use any educational resources (e.g. from Tableau), to help you create the desired visualisations. To find the Tableau resources, in the *Help* menu, click on *Watch Training Videos*.

Connecting to `CoronaImpactEU` in Tableau

First, let's start Tableau.

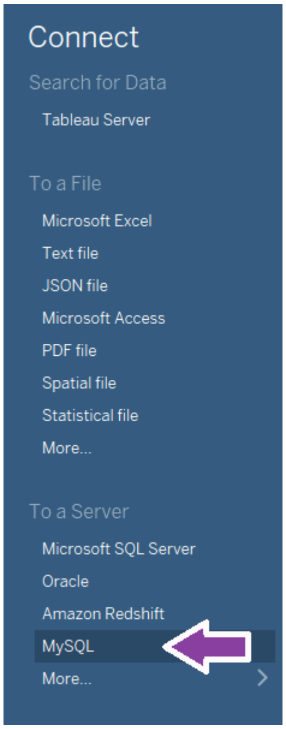


Figure 5. *Connect* panel in Tableau

- When you have started the application, you will be greeted by the main page. In the *Connect* panel on the left side of this main page, you can connect to various data sources such as Excel spreadsheets, PDF files and SQL databases (see Figure 5).
- Select MySQL under the *To a Server* section. Note: if you do not have the MySQL ODBC connector installed on your machine, you will be asked to install it—in that case, click on *Download Driver* and follow the instructions.
- Use the details below to connect to the database that you will use in this part of the tutorial:
  - Server: localhost
  - Port: 3306
  - Database: CoronaImpactEU
  - Username: root
  - Password: *enter your root password*

You should now be redirected to the *Data Sources* tab, in which you see `CoronaImpactEU` and its constituent tables on the left.

Problems

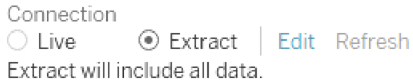
Problem 1: Data preparation (1 point)

- Drag and drop tables into the data model of Tableau, and establish the necessary links between the tables, using the documentation at the end of this assignment.

- **Warning!** The moment you have added one or more tables to the data model, please switch the connection type from *Live* to *Extract*, as shown in Figure 6. Only this connection type ensures I will be able to open and grade the **.twbx files** that you hand in.

- Go over all tables in the data model, and consider all their columns. Make sure that the data types have been parsed correctly (i.e. interpreted correctly) and fix inconsistencies if you find any. Pay particular attention to the *geos* tables.

  – Please note that Tableau has trouble parsing the *Period* column in *YouthUnemployment*. Leave that issue for now; this will be addressed in Problem 2.



Connection

○ Live ● Extract | Edit Refresh

Extract will include all data.

Figure 6. Switching connection type to *Extract* mode.

Problem 2: Youth-unemployment dashboard (2 points)

The main table you will use for creating this dashboard is called *YouthUnemployment*. Please read all bullet points before starting.

- Fixing dates: create an additional column in *YouthUnemployment* named *MonthStart* by concatenating "-01" to the column *Period*. Make sure to set the newly created column's data type to date.

- In the following steps: make sure to put each separate visualisation in a different *Worksheet*. Once all visualisations are ready, they can be combined in a single *Dashboard*.

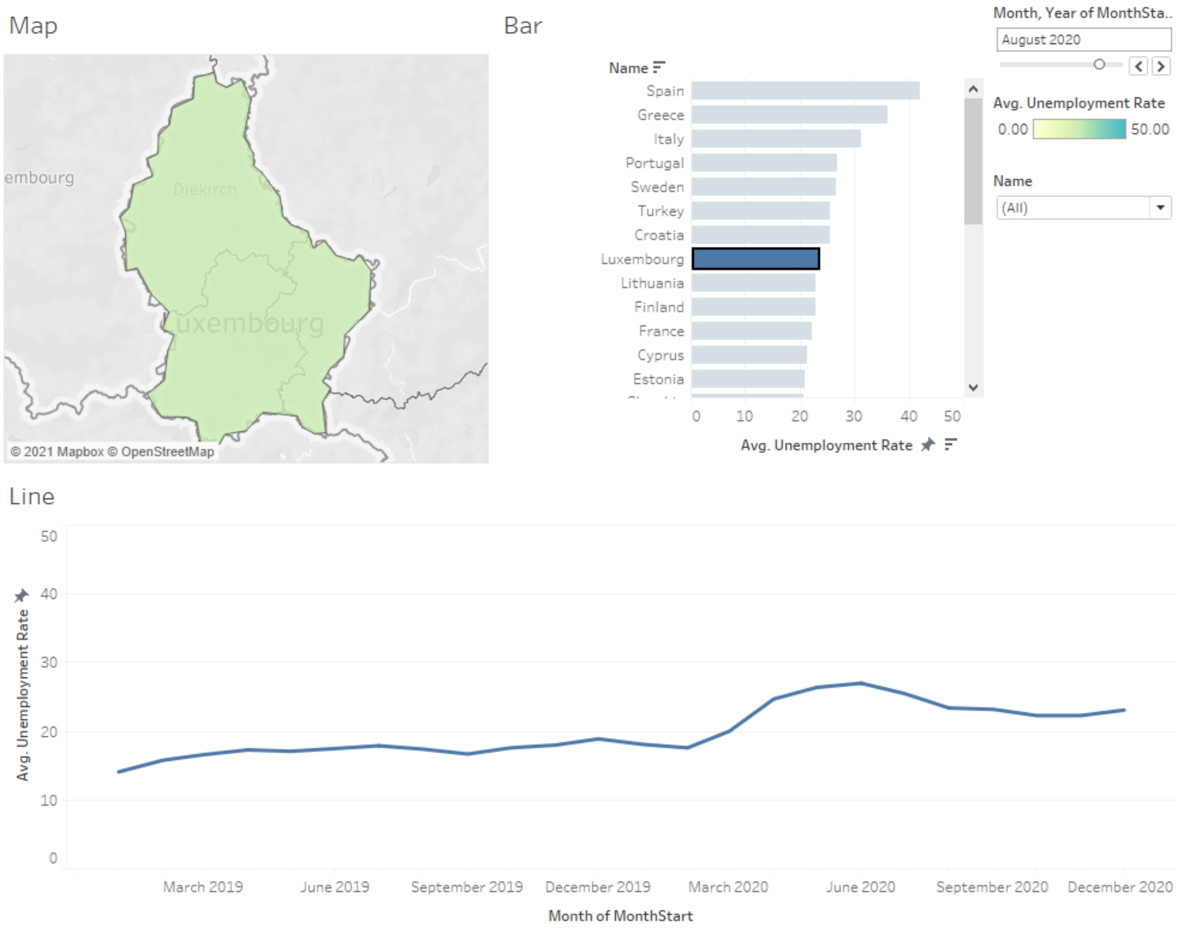
- Map visualisation: create a map visualisation that colours countries based on their average youth-unemployment rate. Choose Blue-Green sequential palette. Remove any values that were not recognised as countries from the visualisation. Set 0% and 50% as start and end of the ranges for these colours.

- Horizontal bar chart: put country names in rows and their average unemployment rates in columns. Order countries in descending order, based on their average youth unemployment rate over the months. Set 0% to 50% as range on the *x*-axis (average unemployment rate).

  – Filter *MonthStart*: create a filter for *MonthStart*, filtering by *Month / Year*. Set it to a single-value slider, and make sure the filter also applies to the map visualisation.

- Line chart: create a line chart that shows youth unemployment rate (again, set the aggregation function to average). Let the y-axis range also range from 0% to 50%.
  - Filter Country: create a filter (that applies only to the line-chart visualisation) that allows the user to select which country is being displayed. It should be of the type *drop-down (single value)*. Also, there should NOT be an *All* option.
- First dashboard: combine all the above created items into one dashboard named *Youth unemployment*.
- Allow the Vertical Bar Chart and the Map visualisations to act as filters.

The end-result should look like the dashboard shown in Figure 7, which highlights, for instance, the data for Luxembourg in August 2020, after having clicked on Luxembourg in the bar plot, and after having used the slider to select that month.



Month, Year of MonthSta..: August 2020

Name: (All)

Figure 7. Example of dashboard for Problem 2, after clicking Luxembourg and selecting August 2020.

Problem 3: Vaccination roll-out dashboard (2 points)

Make sure to keep on working in the same workbook (i.e. do not remove any of the worksheets or the dashboard the you created under Problem 2). You can just add additional new worksheets and dashboards as needed. The main tables you will use for creating the second dashboard are called *vaccinationrollout* and *vaccines*. Please read all bullet points before starting.

- Fixing vaccination names: go back to the *Data Source* tab and provided aliases for the abbreviated names of the vaccines in the *Vaccine* column in the *vaccines* table, with full names of the vaccines as specified in the `CoronaImpactEU` documentation at the end of this assignment.
- Stacked bars: create a stacked-bar visualisation, to display how many first doses were administrated every week. Color the bars based on the vaccine name.
- Creating a parameter: create a parameter named 'doses'. The parameter should have three options: value 1 with 'First dose' as label, value 2 with 'Second dose' as label, and value 3 with 'No doses received' as label. You are going to use this new parameter to allow users to change what they see in the stacked-bar visualisation that you created in the previous step (i.e. do they see bars based on the first dose, the second dose, or bars based on the total number of dosis received?).
- Creating a parameter-based field: to achieve the result outlined in the previous step, once the parameter has been created, right click it, and in the *Create* submenu, click *Calculated field*, and use a statement along the following lines (make sure to replace . . . by appropriate bits of code), to create a field named 'Doses' that you can use for the bar plot:

  ```
  IF ([Parameters].[Doses]=1) THEN SUM([First Dose])
  ELSEIF (...=2) THEN ...)
  ELSE ... END
  ```

- Finishing the parameter: drag and drop the new 'Doses' field into your worksheet. In the Analysis menu, make sure the 'Doses' filter is switched on.
- Filtering countries: add a country filter, make it visible, and set it to the type *Single value (dropdown)*. Again, there should NOT be an *All* option.
- Custom SQL query: go back to the *Data Source* tab and create and execute a custom SQL query. This query should return the running percentage of vaccine-targeted population that has received first and second dose every week (Hint: you need SQL Window functions for this—see slides from Lecture 9).

  This query should return a result per week, per country (do not differentiate based on the vaccine). Also, the statement should return separate percentages for the first and second dose. There should be 4 columns resulting from your query: *geo_id*, *week_start*, *percent_vaccinated_first_dose*, *percent_vaccinated_second_dose*. So basically, this query shows us for every country and for every week, the cumulative percentage of the *Vaccine population* that has received the first vaccination and the cumulative percentage that has received the second vaccination.

- Dual-lines chart: create a new worksheet, and use the data you extracted using the custom SQL query, in the step above, to create a dual-lines chart that displays the percentage of the target population that has been vaccinated with the first and second dose over time. Set the aggregation to average.

- Country filter: create a filter that only applies to the dual-lines visualisation, which allows the user to select which country is displayed. This filter should be of the drop down (single value) type. Again, there should NOT be an *All* option.

- Calculated fields: now click 'Create calculated field' in the Analysis menu. Call the calculated field *Vaccines used %*, and let it calculate the percentage of delivered vaccines that have already been administrated (either as first or second dose). Create another calculation called *First dose %* that calculates the percentage of the vaccine-targeted population that has already received their first jab. Create *Second dose %* that calculates the percentage of the vaccine-targeted population that has already received their second jab.

- Summary table: in a new worksheet, create a summary table that displays (1) the amount of vaccines supplied, (2) the number of first shots given, (3) the number of second shots given, and (4) percentage of vaccines used (use *Vaccines used %*). Split data into separate columns based on the vaccine name. Order the table based on the amount of vaccine doses supplied. Display totals across the various vaccines.

- Map visualisation: create a map visualisation that colors the countries based on *First dose %*. The tool-tip should also display the percentage of people that have received the second jab (*Second dose %*). Exclude countries that do not have vaccination data (United Kingdom, Turkey, and Switzerland).

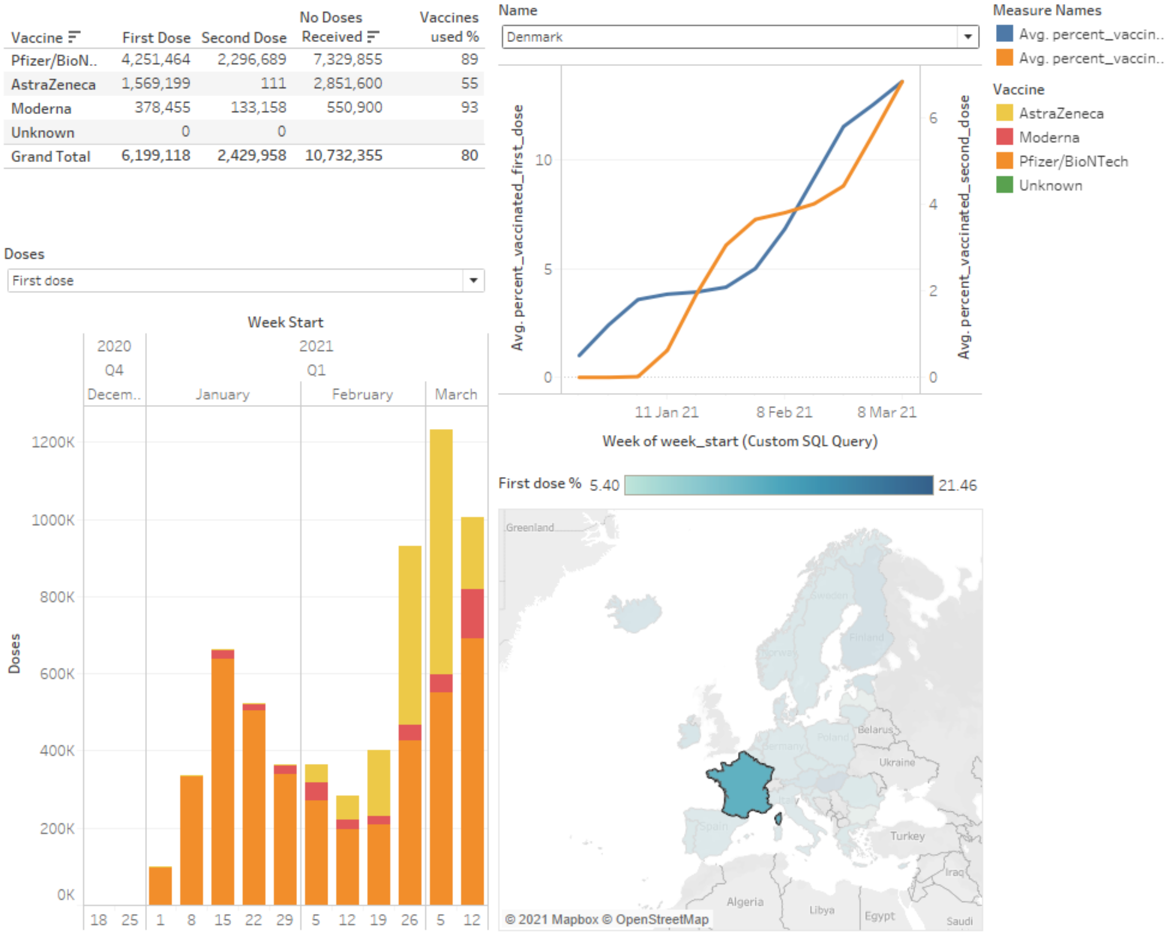
- Second dashboard: combine all the above created items into one dashboard named *Vaccination rollout*.

- Allow the map visualisation to act as a filter. This filer should not apply to dual-lines chart.

The end-result should look like the dashboard shown in Figure 8, which highlights, for instance, what you ought to see when you (1) click on France in the map, (2) indicate you want to see data on the first dose in the bar plot, and (3) select Denmark in the dual-lines chart.

Submission format

Please submit your work on Part I as a single **.twbx (Tableau Packaged Workbook)** file. **Warnings!** (1) The default format of Tableau is not the right format! You need to hand in a file in **.twbx format** and NOT in .twb format. (2) Before storing the .twbx file, please double-check in the *Data Source* tab that the *Connection* is set to *Extract* (see Figure 6).



Vaccine	First Dose	Second Dose	No Doses Received	Vaccines used %
Pfizer/BioN..	4,251,464	2,296,689	7,329,855	89
AstraZeneca	1,569,199	111	2,851,600	55
Moderna	378,455	133,158	550,900	93
Unknown	0	0		
Grand Total	6,199,118	2,429,958	10,732,355	80

Doses: First dose

Name: Denmark

Figure 8. Example of dashboard for Problem 3, after clicking on France in the map, indicating first dose in the bar plot, and selecting Denmark in the dual-lines chart.

Part II. Python (4 points)

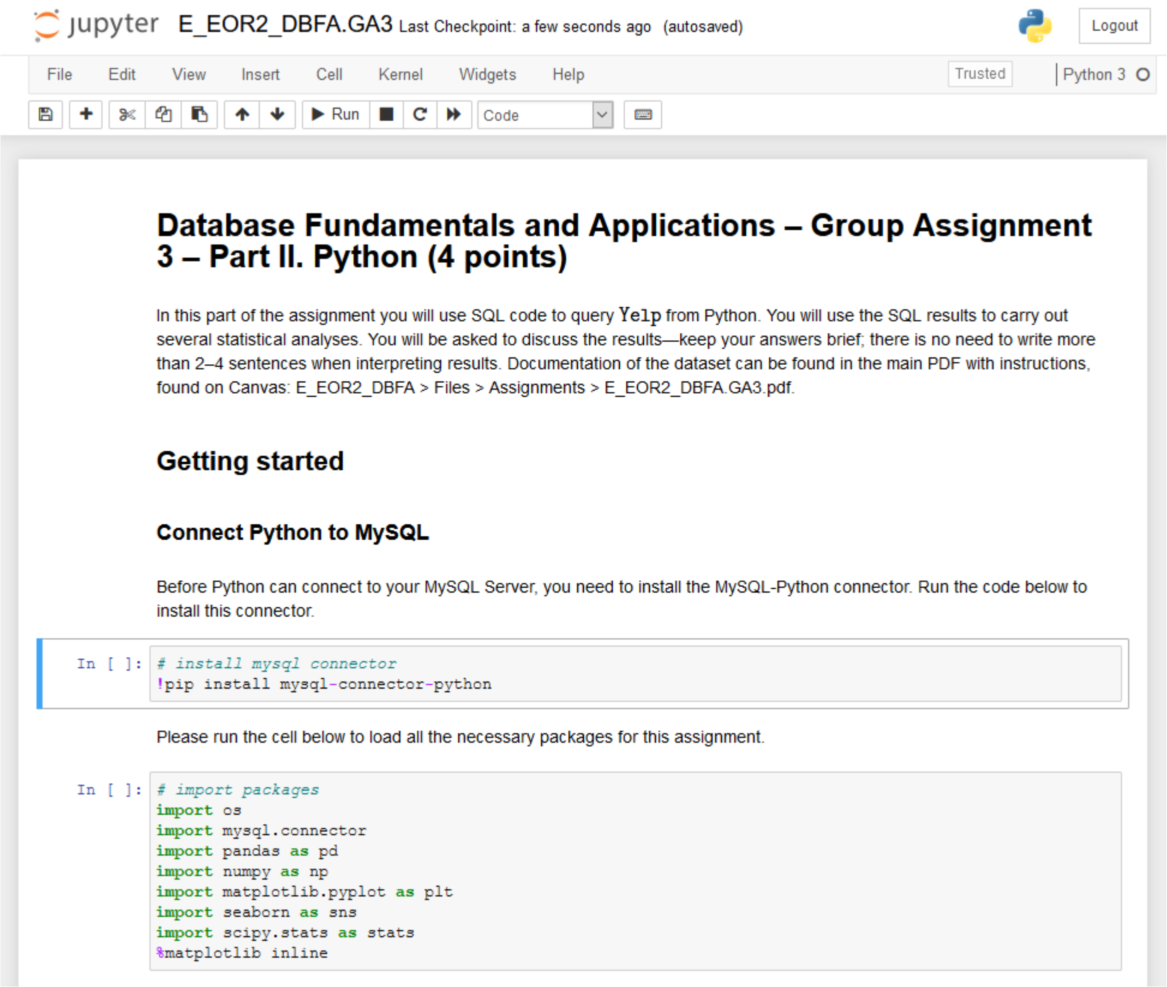
Introduction

In this part of the assignment, you will work with the large Yelp dataset and learn about this data using SQL queries. As indicated, working with very large datasets can be cumbersome in Tableau. Moreover, for application of statistical methods, MySQL Workbench is not the right environment either. Therefore, you will query `Yelp` using SQL statements that are issued from Python.

The instructions for this part of this assignment can be found in the following Jupyter Notebook file (.ipynb) on Canvas: E_EOR2_DBFA > Files > Assignments >`E_EOR2_DBFA.GA3.ipynb`. Please download this file and open it using Jupyter Notebook. You should then see a notebook very similar to what is shown in Figure 9.

Submission format

Please submit your work on Part II as a single, completed **.ipynb (Jupyter Notebook)** file.



Database Fundamentals and Applications – Group Assignment 3 – Part II. Python (4 points)

In this part of the assignment you will use SQL code to query `Yelp` from Python. You will use the SQL results to carry out several statistical analyses. You will be asked to discuss the results—keep your answers brief; there is no need to write more than 2–4 sentences when interpreting results. Documentation of the dataset can be found in the main PDF with instructions, found on Canvas: E_EOR2_DBFA > Files > Assignments > E_EOR2_DBFA.GA3.pdf.

Getting started

Connect Python to MySQL

Before Python can connect to your MySQL Server, you need to install the MySQL-Python connector. Run the code below to install this connector.

```
In [ ]: # install mysql connector
        !pip install mysql-connector-python
```

Please run the cell below to load all the necessary packages for this assignment.

```
In [ ]: # import packages
        import os
        import mysql.connector
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
        import scipy.stats as stats
        %matplotlib inline
```

Figure 9. Example of the opened Jupyter Notebook for Part II.

Documentation `CoronaImpactEU`

Column	Description
geo_id	integer, unique geographic entity identifier
abbrv	string, geographic entity's name abbreviation
name	string, geographic entity's full name
vaccine_population	integer, geographic entity's vaccine-targeted population

Table 2. `geos` table

Column	Description
id	integer, unique vaccination rollout event identifier
geo_id	integer, unique geographic entity identifier - maps to geos table
vaccine_id	integer, unique vaccine identifier - maps to vaccines table
1st_done	integer, number of 1st doses of vaccine administrated in a week
second_done	integer, number of 2nd doses of vaccine administrated in a week
no_of_doses_received	integer, number of doses supplied to the geographic entity
week_start	date, first day of the week for which the data was reported for

Table 3. `vaccinationrollout` table

Column	Description
vaccine_id	integer, unique vaccine identifier

vaccine	string, abbreviated name of the vaccine
	COM = Pfizer/BioNTech
	MOD = Moderna
	CN = CNBG/Sinopharm
	SIN = Sinovac
	SPU = Sputnik V
	AZ = AstraZeneca
	UNK = Unknown

Table 4. `vaccines` table

Column	Description
id	integer, unique youthunemployment event identifier
geo_id	integer, unique geographic entity identifier - maps to geos table
period	string, starting month of the quarter
unemployment_rate	float, percentage youth unemployment

Table 5. `youthunemployment` table

Documentation Yelp

Column	Description
business_id	string, character unique string business id
name	string, the business's name
address	string, the full address of the business
city	string, the city
state	string, 2 character state code, if applicable
postal_code	string, the postal code
latitude	float, latitude
longitude	float, longitude
is_open	integer, 0 or 1 for closed or open, respectively
attributes	json, business attributes to values
categories	string, an array of strings of business categories

Table 6. Businesses table

Column	Description
checkin_id	integer, unique integer checkin id
business_id	string, character string business id - maps to businesses table
date	datetime, timestamp of checkin

Table 7. CheckIns table

Column	Description
review_id	string, character unique review id
business_id	string, character string business id - maps to businesses table
user_id	string, character string user id - maps to users table
stars	integer, star rating
cool	integer, number of cool votes received
funny	integer, number of funny votes received
useful	integer, number of useful votes received
date	datetime, timestamp of review

Table 8. Reviews table

Column	Description
user_id	string, character unique user id
name	string, the user's first name
fans	integer, number of fans the user has
joined	datetime, timestamp of user registration date

Table 9. Users table

References

[1] "Eurostat, datasets related to covid-19." https://ec.europa.eu/eurostat/web/covid-19/data.

[2] "Data on covid-19 vaccination in the eu/eea." https://www.ecdc.europa.eu/en/publications-data/data-covid-19-vaccination-eu-eea.

[3] "Yelp open dataset." https://www.yelp.com/dataset.