# Lab 4

## Lab Overview

## Rules

1. You must follow the rules of the honor code.
2. To present the P+ assignment, you must pass the mandatory tasks.
3. You are not allowed to remove or alter any insert statements given to you in this lab.
4. Code must be well documented with reasonable variable names and function-descriptions.

## Lab Presentation

Make sure you have the following ready to be presented to the TA:

- Code in an editor with text highlights
- Terminal ready to run the code
- Motivations for how the solution for each requested task is sensible. (Not all tasks have one answer, but they have to be good enough for e.g. a client to accept.)

## Preparation

- If you are doing the lab from home you must use the KTH VPN to connect to the servers. Go to the "postgreSQL installation guide" pdf and scroll down to "connecting VPN" and follow the guide.
- Download the python files for this lab.
- Install the following: pip, psycopg2-binary

## Tips

- If you are having connection issues, try running the python file in your student shell

# LAB 4

## Tasks

1. Use the provided python files, make sure that everything works and that you are able to connect to the server.
2. Add the following functionality in the code:
    a) Let users search for airports, both by their name and/or the IATA code. List the airport name, IATA code, and the country of the airport.
        - Make it so that searching for "rlan" includes Arlanda. ***Hint:*** *Wildcard*

    b) Show a list of countries that speak a given language and the number of speakers in said countries.

    c) Create a new desert in a given province, from a given name, area, province, country, and coordinates.
        - Check that the province and country code exist in the province table before creating the desert
        - The coordinates are given as a GeoCoord: `GeoCoord(1.2, 3.4)`
        - If the desert already exists in the `desert` table, only insert it into `geo_desert`. Otherwise, insert it into both `geo_desert` and `desert`.

You are free to make your program look and function however you want as long as it has the necessary functionality. It could be a text-based terminal app or a fully fledged GUI. The important aspect is that the user should be able to input data without having to rewrite the code.

One suggestion is to have a numbered menu with different functionality. E.g. Typing 1 could lead to a prompt "Search for airports: ".

# Lab 4

## P+

**Task 1: Implement more functionality**
Implement the following additional constraints. You can implement these as PSQL constraints, or perform additional checks in Python before inserting the data.

- A desert can only span a maximum of 9 provinces
- A country can only contain a maximum of 20 separate deserts
- The area of a desert can be at most 30 times larger than the area of any province it occupies

**Task 2: Protect against SQL injection**
According to *OWASP 2021:s top 10 web application security risks*, injection attacks are among the third most common vulnerabilities. SQL injections are caused by users being able to enter plaintext in, for example, a search field without proper data sanitation. More information about SQL injections can be found at:
https://www.w3schools.com/sql/sql_injection.asp

Naturally, there are several ways to prevent this. One of the easiest is to use prepared statements, which are sometimes called parameterized statements. This splits up the query creation in the code language into two parts: query creation and parameter insertion. This ensures no user input is interpreted as SQL code, and as such, no SQL injection is possible. The task for P+ is to implement this for all your queries.
***Note:*** *You need to understand and briefly explain what an SQL injection is during the presentation*

1. Give an example of a user input causing the table of fines to be dropped when the non-parametrized query "`SELECT * FROM mountain WHERE name LIKE '%" + usersearch + "%';`" is executed. Also include the resulting query after inserting the user input, that the server executes.
2. Replace all of your existing queries with prepared statements to protect against SQL injection.