Homework: Database design and web applications

1. Creation of the database

Create the database for the management of a cycling championship considering the logical scheme below.

```
CYCLIST (CID, Name, Surname, Nationality, TID, BirthYear)
TEAM (TID, NameT, YearFoundation, OfficeLocation*)
STAGE (Edition, SID, DepartureCity, ArrivalCity, Length, DeltaHeight,
Difficulty)
INDIVIDUAL_RANKING (CID, SID, Edition, Location)
```

Note: Underlined fields identify the primary key of each relationship while asterisks indicate optional fields. Some details about the fields of the various tables are given below:

- SID is an incremental positive integer (1, 2, etc.).
- Edition is an incremental positive integer (1, 2, etc.).
- Length is expressed in meters.
- Location contains the arrival position of the cyclist in that stage (1, 2, etc.).
- Difficulty is a positive integer between 1 and 10.
- Difference in height is expressed in meters.
- YearFoundation and YearBirth are whole values between 1900 and 2000.
- All text fields have a maximum length of 50 characters.

The following activities are required:

- Create a SQL script (**creaDB.sql**) with:
  - Instructions for creating the database corresponding to the logical scheme indicated and specifying the appropriate integrity constraints.
  - Instructions for populating the database created in the previous point (at least one record for each table). The data can be entered at will by the student.

- Write SQL statements to do the following:
  - *Query of the produced database:*
    Fixed the cyclist Id and the stage Id (at your choice), view the list of positions obtained by the cyclist in the various editions. For the chosen cyclist, display the name, surname, name of the team to which they belong and the identifying attributes of the chosen stage. Sort the results by ascending order of edition.
  - *Population of the database:*
    Enter in the database the data relating to a new cyclist and a position in the personal ranking. Properly manage entries using transactions.

2. Creation of the database

Design a web application in Python for online consultation of the previously created database. The application must allow you to perform two tasks corresponding to the SQL statements written for Section 1.

Note: All parameters of any GET requests must match the attribute names of the logical schema, as in the following example (pay attention to capitalization):

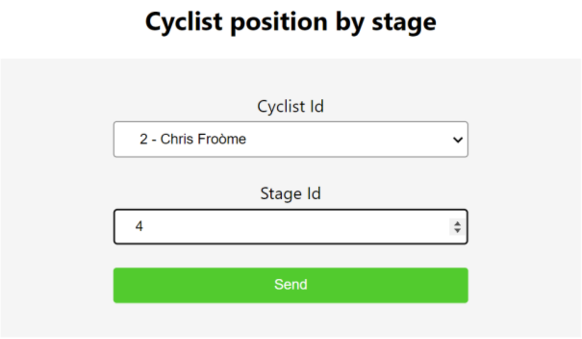
```
insertion?CId=1&Name=Marco&Surname=Pantani&BirthYear=1970&Nationality=Italian&TID=5
```

2.1 Query of the database

After selecting a cyclist Id (from a drop-down menu populated with the values retrieved from the database) and the Id of a stage (free text field), view the list of positions obtained by the cyclist in the various editions. For each cyclist, display the name, surname, and name of the team to which they belong.

A specific error message must be generated in case of missing data (at least one of the parameters empty) or not typed correctly (verify that the cyclist Id and the stage Id are whole values).

Below is an example of the form to be filled in for the query:



Cyclist position by stage

Cyclist Id

2 - Chris Froòme

Stage Id

4

Figure 1. Example form for the database query

2.2 Insertion in the database made

Two forms must be created: one for the insertion of a new cyclist and one for the insertion of a positioning of a cyclist in a stage (already existing):

- The first form must request all the data necessary for the insertion of a new cyclist. Before insertion it is necessary:
  - Verify that the data types entered are correct (verify that the cyclist Id, team Id and year of birth are positive integer values).
  - Verify that the team for which you are requesting entry already exists in the database.
  - Make sure there are no cyclists with the same Id already.

- The second form requires all the data necessary to enter an entry in the table `INDIVIDUAL_RANKING`. In particular, it is required that:
    - The identification fields of the stage and the cyclist are populated using the data already present in the database and the positioning of the cyclist is entered through a free text field.
    - Verify that the data types entered are correct (cyclist Id, stage Id, edition and position must be positive integer values).
    - Check that the data are consistent (the cyclist with that Id already exists in the database as well as the stage Id and the edition).

Check that there is not already a position for the cyclist on the selected stage.

In the event of an error, it must be displayed with white text on a red background, otherwise the message of successful insertion must be displayed with white text on a green background. Below is an example of the two forms and error messages and correct insertion. CSS directives can be used to change the style to be printed directly from Python code (https://www.w3schools.com/css/css_colors.asp).

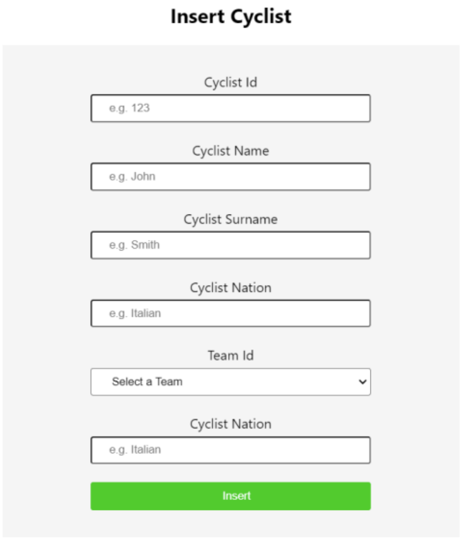
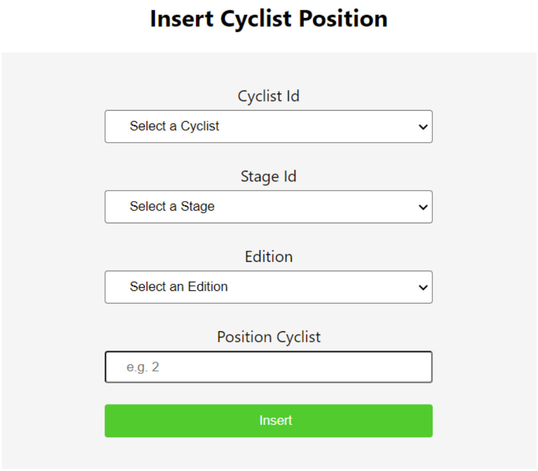


Figure 2. Example form for Cyclist insertion



Insert Cyclist Position

Cyclist Id

Select a Cyclist

Stage Id

Select a Stage

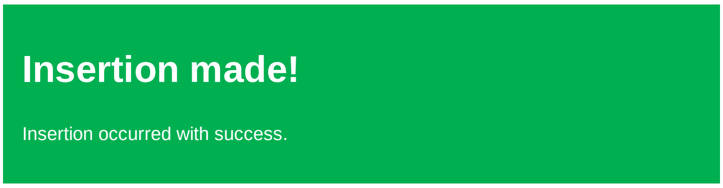
Edition

Select an Edition

Position Cyclist

e.g. 2

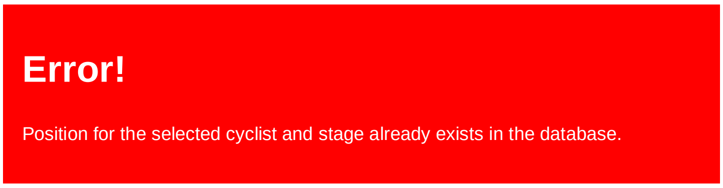
Figure 3. Example form for inserting a position in the personal ranking.



Insertion made!

Insertion occurred with success.

Figure 4. Positive message example



Error!

Position for the selected cyclist and stage already exists in the database.

Figure 5. Error message example