Explanation

Develop a simple web scraper that will scrape Wikipedia pages for football players and can store them into a .csv file. URLs for players that have to be scraped can be found in the playersURLs.csv attachment. The scraper has to be written in Python. In case if some URL entries in the playersURLs.csv file don't point to a football player profile page or if the page doesn't contain any of the required data, you can ignore those entries. Additional requirements are the following:

1. Each player must have the following values (some of them can be NULL in the database if they don't exist on the webpage): Player ID, URL, name, full name, date of birth, age, place of birth, country of birth, positions, current club, national team, number of appearances in the current club, goals in the current club, scraping timestamp.
2. All data should be stored in a SQL database of your choice. Example databases are SQLite, PostgreSQL, SQL Server Express, and MySQL.
3. After you have the database ready, you have to import the initially delivered data (playersData.csv) into the SQL database (either using SQL query or Python script)
   a. If some columns are missing from the .csv file, simply leave them blank (NULL) in the database
   b. If the .csv file contains additional columns, not existing in the database, ignore them (don't include into the database)
   c. If the .csv file contains rows with invalid data (not football players), you can either ignore them or import them into the database and leave empty fields as blanks or NULLs
4. Then, import the scraped data into the SQL database (either using SQL query or Python)
   a. In case any player values were updated (for the same URL), your logic should update the existing player entry in the database
5. The scraper script has to be easily runnable from the command line using the URLs file (e.g. python playersScraper.py playersURLs.csv)

Once you have developed the scraper, it is time to implement SQL queries on the data in the database.

1. Write a query (or queries) that will enrich all players' data with the following columns:
   a. AgeCategory - string
      i. depending on their age player can be 'Young' [<=23], 'MidAge' [24-32], 'Old' [>=33]
   b. GoalsPerClubGame - numeric (float)
2. Write a query that will calculate the average age, the average number of appearances and the total number of players by club
3. Write a query that will do the following: for every player from one chosen club, extract the number of players who are younger, play in the same position and have a higher number of current club appearances than that player (players can be from any club).