Projection Specification

Posted: Sept. 23
Due: 11:59pm, Oct. 9

1. Description

The goal of this project is to implement a simple web application consisting of database queries. Web applications employ client-server architectures. Clients are typically browsers, such as IE, Firefox, etc. Currently, the most popular web servers are Apaches. When a web application involves databases applications, as it normally does, database servers are also employed. Clients, web servers and database servers are inter-connected through the Internet. Users enter inputs at the client side, via web pages. These inputs are then delivered to the web server for processing. The web server then invokes the application logic to carry out corresponding actions, including connecting to the database server for query processing. Web pages and application logic are all stored at the server side. Web pages are dynamically fetched by clients, and application logic are invoked by the web servers at runtime.

Your task is to design and implement web pages and application logic. The web pages allow users to invoke several database queries, and the application logic formulates these queries and submits them to the database server. The following is the detail.

- Database: It contains three relations with the following schema:

  Suppliers(*sid*: integer, *sname*: string, *address*: string)
  Parts(*pid*: string, *pname*: string, *color*: string)
  Catalog(*sid*: integer, *pid*: string, *cost*: real)

  You first create a database with the name 'SupplyDB'. Then you create the above three schemas. The names, including the spellings and cases, for the database, relations and attributes must exactly match those in the above. Then you insert data into each relation. The data for the three relations are all contained in the file data.doc. For your convenience, I have also placed each into a separate .txt file from which you can load in batch mode. The DBMS that you use is MySql.

- Web page: it contains the following generic queries:

a. Given the name of a part, retrieve the information about each supplier who supplied it. This information can be any *subset* of the supplier's id, supplier's name, address, and the cost the supplier charged for that part.
b. Given the cost, retrieve the names of the suppliers who have ever supplied a part with that cost or higher.

c. Given the pid, retrieve the names and addresses for the suppliers who charge the most for that part.
d. Given the color and address, retrieve the names of the parts with that color which were supplied by all the suppliers in the given address.
e. Given the address, retrieve the sids and names of the suppliers in that address who do not supply any part.

For each query, the user is prompted to enter an input. (For example, for query a, the user is prompter to enter the name of a part, and some pieces of information he/she wants to know about each supplier who supplied that part.) Then the browser submits the query and gets the response. The response must be displayed as a table placed at the top center of the screen. The queries are answered separately. So you must include each query in a separate form.

To make your web page user friendly, for each query, you should first give an English statement to describe it, so that the user will have a clear idea of what kinds of inputs are expected and what kinds of outputs will be returned. (For example, the statement may be something like "*this query retrieves the information about suppliers who supply a given part*.") This statement is then followed by the input fields that prompt the user for inputs.

- Application logic: It is a script which implements five types of queries listed under the title 'web page' above. You must write it in Python. It accepts, indirectly via the web server, the inputs submitted by the browser, formulates the SQL statement, which it submits to the DBMS, and then gets the answer. It then returns the answer, again via the web server, to the browser for display. To make all these work, you must incorporate your script into the webserver and also connect it to the database. Refer to the next session for the detail.

2. How to work on your PC/laptop

This project has been designed to give you the convenience of working on your own PC/laptop. To make this possible, however, it requires you to install some packages and work out some simple configurations. Follow the steps below.

- Download Apache, MySql and Python onto your PC, accepting the default settings. Pay special attention to MySql. The default user name is 'root', host is 'localhost', port number is 3306. You can choose a password at your discretion.
- Install a MySql driver from Python in one of the following two ways:
    - When you install MySql, you will have an option for installing a connector for Python. Select the option.
    - In case you don't install the connector in the above way, navigate to the folder where your Python interpreter is installed. By default it should be (version # may be different):

C:\Users\yourname\AppData\Local\Programs\Python\Python36-32

Under this folder, type the following command:

python -m pip install mysql-connector-python

If you are not sure if you already have MySQL connector installed, or you have installed it in one of the above two ways, but want to test if the installation was successful, create a Python file with the following content:

```
import mysql.connector
```

Run this file. If you don't get an error message, the installation is successful.

- Incorporate you Python script to web server: Within the Apache directory tree, there are two sub-directories, 'htdocs' and 'cgi-bin'. The former serves as the document root, so you should place the html files in it. The second is where the web server will look for application logic and so you should place you Python scripts. For any Python script, write the following as the first line

  #!path for the Python interpreter on your machine

  If you accept the default path when you install Python, it should be similar to:

  C:/Users/yourname/AppData/Local/Programs/Python/Python36-32/python

- Name for your html file: use <your MUN user name>.html as the name for your html file. DO NOT USE 'index.html'.

- For the purpose of this project, both the browser and the web server run on the same machine: your PC. When you test your project, enter "localhost/<your MUN user name>.html into the address bar.

Submission: create three subfolders on your PC, <report>, <html files>, <python files>. The files contained in each of them are the following:

- <report>: This subfolder contains a project report. The report should include 1. a cover page where your name, MUN user id, student number and an optional project name are written, 2. an explanation of how to run your application, 3. Screenshots of sample runs of your application. You must type the entire report with a text editor. Do not hand write. It's best to place the entire report in a single pdf file.
- <html files>: This subfolder contains html files to be display by the browser.
- <python files>: This subfolder contains Python files containing Python scripts that answer the five types of queries under the title 'web page' above. (Note: do

not include the scripts that you used to connect to MySql, create database, create tables and insert tuples.)

Zip the above three subfolders into a single file and drop it to the dropbox.

A Remainer:

1. You must use the data set exactly as was posted in D2L, and use the database names, table and attribute names exactly as mentioned above. Never change any word, including the spellings and cases. Then, make sure your project works with them. Otherwise, when I run your project on my copy of the database, I may get incorrect answers. This will affect your final marks.

2. Start early. No late submission is acceptable.