Project #1 (Milestone 3)

CSC 261/461 (Intro to Databases Fall 2020

Due Date: 10/31/2020 (11:59 pm)

Introduction

This milestone has both theoretical and application components. In the first part, you will provide us the functional dependencies present in all the relations (or tables) that you are using. Then you will convert these relations into BCNF (Boyce–Codd normal form) if the relations are not in BCNF already. For the second part, You will work on mostly **HTML** and **PHP + SQL** . This is the first time you will share your database with the whole world (or at least with everyone in UofR). In this penultimate milestone, you will transform the logical design of your database into a physical design. You will populate your tables and provide us 'real' interfaces for interacting with the database.

Task A: BCNF Normalization

A relational schema R is in Boyce–Codd normal form if and only if for every one of its dependencies $X \rightarrow Y$, at least one of the following conditions hold:

- $X \rightarrow Y$ is a trivial functional dependency ($Y \subseteq X$)
- X is a superkey for schema/relation/table R

For this milestone, you need to make sure all of your relations are in Boyce-Codd normal form. Provide us a list of dependencies for each relation. Decompose them if the tables are not in BCNF. After the decomposition, all the resultant relations should be in BCNF.

If you decide to keep a particular relation in 3NF instead of BCNF, justify the decision. (Hint: Lossless and/or Dependency preserving decomposition). Submit **TaskA.pdf** which contains the details of the transformation from the initial schema to the final schema where all the relations are in BCNF. This file should also contain all the functional dependencies you have started with. Note: This is quite possible that your initial schema is already in BCNF and in that case you just need to provide us the functional dependencies and convince us that the relations are already in BCNF.)

Task B: Creating and loading relations

In Milestone 2, you have designed the relations (table) those are required for your project. For this milestone, you will create the actual relations. Create a file **create.sql** which will create all the tables in your database. load these relations from data files (tab or comma separated files). The tab or comma separated files can be created by you (dummy values) or you can provide the sources. Create a **load.sql** file for bulk loading.

- Create a **readme.txt** file which states the source of your data.
- Put **create.sql , load.sql** , all (or an example of) the **.dat** files(or **.csv** files, or data files in any other format) and a readme.txt file into a directory taskB .
  (Note: **create.sql** and **load.sql** files should have the same structure as in Project 2 Part 1)

Task C: Accessing the relations from Web

For Task C, all we need is an address of a web page on Betaweb server. Before starting for Task C and Task D, please read the betaweb FAQ to get fundamental information on how to use betaweb as your web server: **http://www.cs.rochester.edu/courses/261/spring2018/others/betaweb-faq.html.** Note that you'll need VPN connection to visit your websites hosted on betaweb.

This webpage of yours should contain links to view the content of each relation/table you made in Task B. You may choose some (more than 3) relations to show if you have a large schema; you may also choose to show a couple rows/columns if some relations are huge.

Save the address of the web page as **taskC.txt** . This file should also contain a brief description of contributions made by each member. We would like to see an even distribution of workload.

Task D: Modify database with forms

For Task D, you should implement, in html forms, at least 2 of the interfaces mentioned in your Milestone 1 where the users queries or make change to the databases, and make them take effect via PHP on your database. The interface can be insert/delete data, upload bulk data file, query for results across multiple relations, etc. You should design your page such that the result of your query/modification can be clearly seen by the user. You may provide usage information on the page.

We provide an example website that assumes a database containing a table named Students, and you can copy it to a directory of your choice like, for example, your home directory:

```
cp -r ~csc261/proj3/p1m3_example ~/
```

The outcome of this Task is also a txt file **taskD.txt** containing an address to your page(s).

How to submit

Create a new directory **milestone3** . Copy file **taskA.pdf** , directory **taskB** , **taskC.txt** , and **taskD.txt** to **milestone3** . Archive (tar) the folder as **milestone3.tar** .

```
mkdir milestone3
cp taskA.pdf milestone3/
cp -r taskB milestone3/
cp taskC.txt milestone3/
cp taskD.txt milestone3/
tar -cvf milestone3.tar milestone3
```

Submit the tar file using the command:

```
~csc261/submit p1m3 milestone3.tar
```

You should get a feedback whether the submission is successful or not.

You may resubmit as many times as you like; however, only the latest submission (along with the timestamp) will be saved, and we will use your latest submission for grading your work. Submissions via email will not be accepted! No late submissions allowed.

You must adhere to the naming guideline to avoid losing points.