**The Alumni Problem**

We need a database for storing alumni information. Right now the info is stored on paper as shown below.

|  |  |  |  |
| --- | --- | --- | --- |
| students | private | emails | majors |
| Gary Hudson | 123-45-6789 | [gary@hudsonlaw.com](mailto:gary@hudsonlaw.com) | Law |
|  | do not contact | [garyh@gmail.com](mailto:garyh@gmail.com) | CS |
|  |  |  |  |
|  |  |  |  |
| Barbara Johnson | 234-56-7891 | [barbara@johnsoneng.com](mailto:barbara@johnsoneng.com) | CS |
|  | interested in donating |  | Math |
|  |  |  |  |
|  |  |  |  |
| Mike Lopez | 345-67-8911  working on PhD | [mikeylikesit@gmail.com](mailto:mikeylikesit@gmail.com)  [mikethebike@mc.com](mailto:mikethebike@mc.com) | Math  Law |
|  |  |  |  |

You can assume that though we may need to store more info (more students, more emails for a particular student,…) we won't need to store any more fields, such as additional private info or info about a student's minor. Further, each student has only 1 private entry, as it is their social security number and some comments.

Assignment #1: **195\_createTables\_yourName**

1. Using the hosting site and phpAdmin
   1. write the sql to create whatever tables you need to store the above info efficiently and effectively
      1. note that the primary key for the student *must* be an auto increment integer
      2. note that the comment under private – will varchar work? are there other possibilities? spend a few minutes reading to decide
   2. use your sql to create the tables
   3. include a simple ER diagram for your database, followingthe example on pg. 11 of my course notes
   4. ***do not populate the tables***
   5. you must name the tables as shown
   6. you may need additional tables; if so name them meaningfully
2. Turn in:
   1. soft copy: 195 \_createTables\_yourName.sql
      1. comment must include

/\*

195

createTables.sql

your name/s

\*/

* 1. hard copy of this SQL file and your ER diagram, stapled together

1. Note: you have to get the design of the tables correct to be able to do the following tasks.

Assignment #2: **195\_populateTables\_yourName**

1. Using the database your created in Assignment #1, write a Java program that populates the **students**, **emails** and **majors** tables with the information shown.
   1. you may use console or gui interaction, whatever is easiest for you
   2. **you** **absolutely** **may not assume that** you know a student's id (the key field) or anything like that
   3. add a query at the end that displays all of the information in the students, emails and majors table that the work can be verified
2. It must be possible to rerun the **populateTables** program and add another student with all of their information, then rerun it again and do the same, ad infinitum, without knowing if there are already 1 or 1,000 entries.
3. *The person running this java program is presumed to be a low level clerk that knows nothing about the underlying database. They're just typing in data. For example:*
   1. *Angie Smith*
   2. [*asmith@thusspakezarathustra.com*](mailto:asmith@thusspakezarathustra.com)*,* [*angieistheboss@hotmail.com*](mailto:angieistheboss@hotmail.com)*,* [*asmith@gmail.com*](mailto:asmith@gmail.com)
   3. *Screw Driver Technology, Underwater Basket Weaving*
4. Note: you have to get the populating of the tables correct to be able to do the remaining tasks.
5. Turn in:
   1. soft copy: 195\_populateTables\_yourName.java, comments as described above.
   2. hard copy of the program

Assignment #3: **195\_populatePrivate\_yourName**

1. Using the database your created in Assignment #1, write a Java program that populates the **private** table with the information shown.
   1. **you** **absolutely** **may not assume that** you know a student's id (the key field) or anything like that
   2. add a query at the end that displays all of the information in private table that the work can be verified
2. The program that displays all the people in the database. The user chooses one of these people and inputs the appropriate **private** information.
3. *Again, the user is presumed to know nothing about the underlying db but in this case they've been cleared to see SSNs*.
4. And again it must be possible to rerun the program and add private information for *another* student.
5. Turn in:
   1. soft copy: 195\_populatePrivate\_yourName.java, comments as described above.
   2. hard copy of the program

Assignment #4: **195\_deleteFromTables**\_**yourName**

1. Again using the same database, write a Java program that displays all the people in the database.
2. The user may choose one of these people. *All* of their information is deleted from all of the tables.
3. It must be possible to rerun the deleteFromTables program again and delete another alum.
4. Again, the user knows nothing about the db.
5. Turn in:
   1. soft copy: 195\_deleteFromTables\_yourName.java, comments as described above
   2. hard copy of the program

Assignment #5: **195\_alumniQueries\_yourName**

1. After you've deleted some people and their info, rerun your populateTables program to repopulate the tables for the next task.
2. Write a Java program that connects to the above database and executes the following tasks and displays neatly labeled results (Query #A, Query #B, etc.).
   1. Show the student names and their matching SSNs.
   2. Input an alumni name from the keyboard and display that person's emails.
   3. Using the same alumni name, display their majors.
   4. Show who majored in CS.
   5. Did Barbara Johnson major in CS? Answer 'yes' or 'no' (nothing else).
   6. Show name of everyone that had only 1 major.
   7. Show names and emails of CS majors.
   8. Show names of everyone that did not major in CS. *Do not assume anything about their major such as 'it must be law or sociology or math'.*
3. Turn in:
   1. soft copy: 195\_alumniQueries\_yourName
   2. hard copy of the program