CMSC424: Database Design Class Project

Spring 2011

Overall goal

The goal of this project is to build a bibliographic database that can be accessed either through a web portal or a stand-alone client. Your database must keep track of the following types of information/entities:

- **Publications.** Information you will need to track are title, journal, volume, issue, page, year, authors, PubMed ID, and abstract. If the publication is in a conference you need to keep track of the conference name. If it is a chapter in a book you need to keep track of that information as well.
- **Authors.** You will need to keep track of first name, last name, middle initial. Note that in some cases the authors may only be known by a first initial (rather than first name) and that they may have more than one middle initial. Also, an author's full name cannot be a key as multiple researchers may have the same name. You will need to suggest a way to resolve this ambiguity.
- **Journals.** You will need to keep track of the journal name.
- **Conferences.** For papers in conference proceedings you need to keep track of the conference name and year.
- **Books.** You need to keep track of the book name, publisher, and ISBN.
- **Tags.** Each article should receive one or more tags. Note that the entities above will be retrieved from an online database, however the tags are information that will be entered by the users of the system.

The users must be able to enter or modify information in the database in three ways:

- Through a form interface either on a web page, or through a GUI application;
- Through direct query to PubMed, using the E-Utils system: http://eutils.ncbi.nlm.nih.gov/
- By importing a BibTeX file (see www.bibtex.org for details)

Your database must support two modes of operation:

- **Administrator.** A user will have to log on as an administrator and should be able to add more information to the database, or modify existing records.
- **Client.** A user should be able, without logging on, to retrieve one or more references using queries against the database. More details will be provided below.

Part 0: Forming teams Due: April 5 @ midnight

Please submit the names of all students participating in each team by April 5. Send an email to both the instructor, and to the TAs.

Please do not wait until this deadline to form the teams and start working on the project.

Note: You can have teams of up to 4 students.

Note: The amount of work required for full credit on the project will not depend on the number of people in the team. I.e. it's in your interest to find team partners so you can complete the project in time.

Part 1: Database Design Due: April 14 @ midnight 25% of project grade

The first part of your project is to design the schema you plan to use for this project. Please provide both an E-R diagram and the relational schema that implements the requirements listed above.

Also provide a design document that outlines how the database component will link to a UI front-end, and highlight the functionality you will make available to the users.

Note: Do not restrict yourselves to just the attributes I've outlined above. Use the BibTeX documentation as well as the PubMed E-utils documentation to figure out whether other attributes are also needed.

Note: Make ample use of web resources to figure out how to organize your database. There are a number of bibliographic databases for which schemas are available and these can help you figure out how to organize your own database. Please don't copy these schemas blindly, rather build your own.

Part 2: Implementation Due: May 9-13, 2011 (all code due May 9 @ midnight, demos to be scheduled during week of May 9) 75% of project grade

Your implementation should include a database back-end (using any of the commonly-available database servers: Oracle, MySQL, Postgres, etc.) as well as a user interface component. The latter can be either a standalone client, or a web interface. You can use any programming language that you choose for the implementation.

Data input. The users must be allowed to enter information in the database:

- Through a form interface;
- Through direct query to PubMed, using the E-Utils system: http://eutils.ncbi.nlm.nih.gov/
- By importing a BibTeX file (see www.bibtex.org for details)

Basic queries. You must allow the ability to display references and information about them as follows:

- Query by name, title, or journal + volume + page number.
- Query by tag, and by tag + year. You must allow either exact year specification or year ranges: e.g. before 1995, 1997-2010, since 2008

The references received should have checkmarks next to them that allow the user to select the 'interesting' references among them. You should provide ways to check or uncheck individual, or all references retrieved by a query.

Reference sets. You should provide for a way for a user to create and delete references from a reference set (that is specific to the user). For example, users could run a query, check several references in the output, then click a button to add these to the current reference set. The users should also have the ability to remove references from the reference set.

Data export. You should provide mechanisms to export the current reference set into HTML or BibTeX formats.

Advanced queries. Your system should support the following advanced queries.

- For a given author, retrieve a list of this person's co-authors.
- Given a pair of authors, retrieve the papers they have co-authored.
- Given a pair of authors, retrieve the number and names of other authors that have published with both of the authors. Note that if you select authors A and B, an author C will be part of the output of this query if C has co-authored a paper with A, or with B. It is not necessary that C has co-authored a same paper with both A and B.
- For every pair of authors in the database, retrieve an "interaction strength" computed as the fraction of shared co-authors, i.e. the number of authors that co-authored papers with both A and B divided by the total number of authors that co-authored papers with A or with B.
- For a given pair of authors in the database, compute the distance between them in terms of co-authorship relationships. I.e. if A coauthored a paper with B and B co-authored a paper with C, the distance between A and C is 1 (A and B, and B and C are at distance 0).

Mystery queries. You should design your system with flexibility in mind. One week before the due date I will issue a new query that must be supported by the system. You will only have a short period of time to refine your interface and implement the new query.

Additional requirements

Your code must be well organized and documented, including in-code comments, and user manuals.

Submission

Submit your documents and code (including the relational schema and any other SQL code you write) through the submit server.

You will also have to demo the project to either one of the TAs or to the instructor. Details will be available closer to the due date.

Grading

If the code does not work, irrespective of reason, you cannot get more than 40% credit.

E-R diagram and relational design: 30%
Code is well documented: 10%
System works as advertised:
- Data input: 10%
- Simple queries: 10%
- Managing reference sets: 10%
- Data export: 5%
- Advanced queries: 15%

Mystery query: 10%

Additional Logistical Considerations

Teams: You can work in teams up to 4 students. If you decide to work in a team, when submitting the project you will each need to sign the following statement:

I certify that I have contributed XX % of effort to this project and I agree to be graded accordingly.
*Signed:*______________

The XX value should ideally be the same for all members of the project, however if one or more of you do not pull your weight during the project you should list an accordingly smaller percentage.

Note: I will not arbitrate disputes between the team members.
Note2: In case of cheating, all team members will receive disciplinary action irrespective of which one of you cheated.

Late policy: The usual late policy applies: 10% -off for 1 day late, 20%-off for 2 days late, no credit for 3 days or more.

Extension requests: As already mentioned you will not be able to receive individual extensions unless you have a university sanctioned excuse. I have, in the past, allowed students extensions for non-official excuses such as job interviews and conference participation. However, such extension will not be granted within the last week before the project is due.

Note: Being busy with other classes or project, being tired, not realizing how hard the project really was, etc., are not appropriate excuses. Please don't even try.

How to do well

Start the project early, and ask questions if you cannot figure out all the requirements. Do not expect much help from me or the TAs during the last week before the project is due. While we will try to accommodate your request for help, there is no guarantee we will be available outside the official office hour schedule. In any case, the last week before the project is due should be devoted to ironing out minor bugs in your code. That is not the time to start working on the project.