CS425 - Fall 2013 - Course Project
A Library Management System

Boris Glavic

September 12, 2013

1 Project Timeline

The project has three deliverables. Deadlines are announced on the course webpage `http://cs.iit.edu/~cs425/schedule.html`. This document can be downloaded from `http://cs.iit.edu/~cs425/files/projectdesc.pdf`. Each group will **demo** their application at the end of the semester. The deliverables are:

- **ER-model**: Each group should develop an ER-model for the application. This can be uploaded as any type of image file (please do not use esoteric formats).
- **Relational schema**: The second deliverable is a translation of the ER-model into a relational schema implemented as an SQL script. The script should use Oracle's SQL dialect and should execute without errors on the `fourier.cs.iit.edu` Oracle accounts. Besides from defining tables and constraints, this script should create indexes where appropriate.
- **Application**: The last deliverable is a library management application that uses the relational schema defined in the first two deliverables. This application can be either a web or desktop application.

Some of the requirements are marked as optional **bonus** requirements. You are free to not realize these requirements, but you can get extra credits by implementing them.

Every member of the group has to contribute in each phase of the project and you will be graded on your individual contribution and on the overall project result.

2 Overview

The goal is to build a library management application using a database backend to store information about member, documents, and so on. The application will enable users to search for documents, borrow documents, enter new documents, ... For the project we record information for a single library and its users.

3 Data Requirements

3.1 Users

We will consider two types of users:

- **Members**: Members can search for documents, borrow documents, and return documents. Documents can be borrowed for certain time periods. If a member does not return a document by the due date, he will be notified.

- **Librarians**: Librarians can check overdue documents, manage users, and add/delete/modify documents.

3.2 Documents

The main asset of the library are documents. For the project we will consider the following types of documents:

- **Books**: Books are written by one or more authors and published by a publisher. Books have a title and may exist in different editions - each published at a certain year. *For example, "Database System Concepts" is a book written by authors Silberschatz, Korth and Sudarshan and published by McGraw-Hill Science/Engineering/Math. The sixth edition was published in 2010.*

- **Journal Articles**: Journal articles are written by one or more authors, have a title, and are published in a certain journal. Journals have a title and are published by a publisher in issues. Issues have editors and a publication date. *For example, "Science" is scientific journal. The article "Antibody Therapeutics in Cancer" written by Mark X. Sliwkowski and Ira Mellman was published in the "13 September 2013" issue of this journal.*

- **Magazines**: Magazines have a name and come in issues that are published at a certain time. Each issue of a magazine has a title. Magazine issues have editors and contributors.

There is some information that needs to be recorded for each type of document. For each document we need to store a list of keywords (e.g., "databases", "course material") and a hierarchical classification (e.g., "Science-CS-Databases" or "Poetry-English-1800's-William Blake").

3.3 BONUS: Additional Document Types

Store information about additional document types such as thesis and technical reports.

3.4 BONUS: Citation and Crossreference Information

Most documents cite other documents. In this extension your schema should be extended to model the citation relationship between documents. For example, “Database System Concepts” cites Korth’s Journal of the ACM article “Locking Primitives in Database Systems”.

3.5 Copies

A library may have several copies of each document. Some documents may not exist as copies in the library. Copies are stored a certain place inside the library, e.g., a room, level, ... For each copy we need to know whether it is currently borrowed and if yes by which member. Members are only allowed to borrow documents for a certain time (e.g., 2 weeks). We may also place restrictions on the number of documents that can be borrowed by a user at a certain time.

4 Application Requirements

The application should support the following actions. We indicate for each action whether it can be executed by librarians or library members.

- Search for documents (`librarian/member`)
- Borrow/return copies of documents (`member`)
- Add/Delete/Modify document and copies (`librarian`)

4.1 Search

The central component of the application is the document search interface. The application should support different search criteria (e.g., by author, by title, by classification) and combinations of these search strategies. *For example, return all books (document type) that are written by Peter Jackson (author) for which there are copies available for borrowing.* Both complete matches and partial matches should be supported. The results of a search should be shown as a list of documents to the user. The user should then be able to access detailed information for each document or refine/modify his search. You should also support searching for terms in different criteria (e.g., return documents that have “database” in their title, keywords, or classification).

4.1.1 BONUS: Saved Searches

Enables the user to store searches and re-execute them later.

4.1.2 BONUS: Ranking Search Results

Extend the search so that partial results are ranked on how good they match the search criteria. For example, one way to rank documents is by the number of criteria that match the search. If the user searches for the term "databases", then documents that have the term in their title and keywords should be ranked higher than documents that only have the term in their title. Another way to rank is to allow more than one search term and rank the documents based on the number of contained search terms.

4.1.3 BONUS: Display search results with citation links

Show links for all documents cited by each search result. Clicking such a link should bring up the information about the referenced document.

4.2 Borrow/Return Documents

Library members can borrow copies of documents. The system needs to make sure that it is not possible to borrow a copy of a document if all copies of this document are currently borrowed by other users. Borrowing should be integrated with search. E.g., if the user has found a document, then he/she can borrow a copy (if available). The application should warn the user about borrowed documents that are past due dates.

4.3 Add/Delete/Modify Documents and Manage Copies

Librarians should be able to modify the document information and manage copies for documents. For example, a librarian can add a new copy for a book to the library and record where it is kept within the library.

4.3.1 BONUS: Import .bib files

The BibTex file format (`.bib`) is a format for storing bibliographic information. Write an importer that enables users of your application to import a bibtex file into the database.