CSC173: Project 4
The Relational Data Model

Project Goals

1. **Part 1**

   (a) Implement a database containing the relations (tables) shown in FOCS Figure 8.1 and 8.2 (also seen in class). Use the method described in Section 8.2 in the section "Representing Relations" and elaborated in Section 8.4 "Primary Storage Structures for Relations." Describe your implementation briefly but clearly in your writeup.

   (b) Implement the basic single-relation *insert*, *delete*, and *lookup* operations as functions. Describe your implementation briefly but clearly in your writeup.

   (c) Use your *insert* method to populate the tables with (at least) the data given in the figures. Demonstrate the operations with some examples using this data. Be sure that your program explains itself when run (using informative printed messages).

   (d) Implement functions for saving your database to one or more files, and loading from the same. Demonstrate this functionality also.

2. **Part 2**

   (a) Write a function to answer the query "What grade did *StudentName* get in *CourseName*?" as described in Section 8.6 "Navigation Among Relations." Demonstrate this functionality

   (b) Write a function to answer the query "Where is *StudentName* at *Time* on *Day*?" (assuming they are in some course). Demonstrate this functionality also.

3. **Part 3**

   (a) Implement the Relational Algebra operations as described in Section 8.8. Describe your implementation briefly but clearly in your writeup.

   (b) Use your implementation to do the operations on the "registrar" database described in Examples 8.12 (Selection), 8.13 (Projection), 8.14 (Join), and 8.15 (all three). You may also throw in any additional examples you feel illustrate relevant aspects of your implementation.

Extra Credit (20% max total)

1. The code you wrote for the "registrar" database is obviously specific to it. A true database system, like SQLite or MySQL, allows you to represent any database schema. Generalize your code to represent arbitrary databases consisting of arbitrary relations. Note that you do not need to understand SQL for this. What you need to do is create "generic" representations of tuples and tables and then use them to write code for some specific example. You can use the "registrar" example or the DMV database (Exercise 8.3.2) or some other application. Explain what you've done and how it will be demonstrated by the program(s) in your writeup.
2. Identify a subset of SQL that you can support with your implementation. Write a simple parser for that language, and use it to create and query one or more example databases. Explain what you've done and how it will be demonstrated by the program(s) in your writeup.

Project Requirements

Your project submission MUST include the following:

1. A more substantial writeup than just a README file. This PDF document should describe:
   (a) How to build and run your project
   (b) A brief but complete description of your approach to the problem(s). This should not be a description of your code, which should be well-commented also, but rather an explanation for why you did what you did and how it worked out.
   (c) Any collaborators (see below)
   (d) Acknowledgements for anything you did not code yourself or get from the textbook.
2. All source code for your project, including a `Makefile` or shell script that will build the program per your instructions (see below).

If you are a CSC major, you should use your CSUG account to ensure that your program can be built on the CSUG machines. If you don't know how to use your CSUG account, ask for help from the TAs. If you are not a CSC major, you can use your own or an IT department Mac with the Terminal program. My "C for Java Programmers" document from the start of the course has an overview of C development that may be helpful.

You must provide one or both of the following:

- A Makefile (see the "C for Java Programmers" document)
- A shell script (sequence of terminal commands) named "build.sh" with the exact sequence of commands needed to compile your code. For example:

```
cc -o mydb mydb.c
cc -c Table.c
cc -o myotherdb myotherdb.c Table.o
```

Your writeup should make it clear to the TAs (1) how to build the project from the terminal, and (2) how to run it from the terminal for each portion of the project. Saying that it builds in Eclipse and that you run it by pressing the green button in Eclipse is not sufficient.

The TAs must be able to cut-and-paste from your documentation in order to build and run your code. The easier you make this for them, the better your grade will be.

Late Policy

Don't be late. But if you are, 5% penalty for the first hour or part thereof, 10% penalty per hour or part thereof after the first.

Course Collaboration Policy

Collaboration on projects is permitted, subject to the following requirements:

- Groups of no more than 3 students, all currently taking CSC173.
- You must be able to explain anything you or your group submit, IN PERSON AT ANY TIME, at the instructor's or TA's discretion.
- One member of the group should submit on the group's behalf and the grade will be shared with other members of the group. Other group members should submit a short comment naming the other collaborators.
- All members of a collaborative group will get the same grade on the project.