CMPSC 380
Principles of Database Systems
Fall 2014

Laboratory Assignment Four
Visualizing Relational Database Schemas

Introduction

When either designing a new database or attempting to understand an existing one, it is often useful to produce a visualization of the relational schema. A technical diagram of a schema helps a database designer to see the relationships that already exist in a complex database, thus aiding understanding and the identification of defects. When designing a new database schema, a technical diagram helps a person to see the proposed relationships between tables and identify potential sources of performance bottlenecks or correctness concerns. In this laboratory assignment, we will explore how to use a well-known database visualization tool, SQLFairy, to show database schemas.

Learning About Schema Visualization

Since SQLFairy claims to produce something known as a "pseudo-ER" diagram, it is a good idea to review Section 7.5 of your textbook to learn more about how entity relationship (ER) diagrams aid in the design and understanding of database schemas. Instead of attempting to understand every detail about ER diagrams, you should make sure that you understand the basics (e.g., the meaning of a box and an arrow). Now, please visit the SQLFairy Web site, available at `http://sqlfairy.sourceforge.net/`, and learn more about the "pseudo-ER" diagrams that this tool produces. How are the textbook's diagrams similar to and different from those created by SQLFairy?

Using SQLFairy for Schema Visualization

The SQLFairy suite of tools is available on your workstation, through programs like `sqlt-graph`. In your terminal window, type the commands "`man sqlt-graph`" and "`sqlt-graph --help`" to learn more about how this program works. Once you and your partner understand the features that this tool provides, change into the course repository and type the command "`git pull`". If you change to the `cs380F2014-share/labs/lab4/` directory you should now see that there are a total of six SQL files that define the schemas of six databases. As you first learn to use the `sqlt-graph` tool, try to visualize the schema of the `University.sql` file that was downloaded from the Web site for the course's textbook. How does this visualization compare to the one in Figure 7.15?

As you are creating schema visualizations for this assignment, you should save the output of `sqlt-graph` in the PNG format. When you want to view a PNG file created by the visualizer, you can use the `xdg-open` command in your terminal or load the file through the graphical file browser.

Exploring Schema Visualization Options

As you may have noticed, SQLFairy supports different dialects of the structured query language using the "`-d`" option. In particular, the tool supports the PostgreSQL, MySQL, and SQLite formats. How does the specification of these three options change the resulting pseudo-ER diagram? Additionally, the SQLFairy tool supports the "`--show-datatypes`", "`--show-sizes`", and

"`--show-constraints`" command-line options. In order to successfully complete this laboratory assignment, you should run the `sqlt-graph` tool with all possible combinations of these three command-line options for all six of the relational schemas available in the Git repository. For each relational schema, you should also write a short commentary explaining at least one way in which the visualization helps you to understand an aspect of the database's structure. Your writing should clearly explain the meaning of all the key graphical elements, such as the boxes and the arrows, contained in the diagrams. Before you turn in each of the visualizations, you should make sure that you annotate them with labels that state the name of the SQL file and the chosen command-line options. Please see the instructor if you are not able to run `sqlt-graph` in the terminal.

Finally, you should make your own SQL file that contains an original schema. This schema should contain, in total, at least two tables, two constraints, and five attributes—and also be executable with the `sqlite3` command-line tool. As you are creating this schema, please regularly produce its visualization and confirm that it conforms to your mental model of the database. After you have finished exploring the different options for schema visualization with SQLFairy, you should use the best mix of parameters to produce a final visualization with `sqlt-graph`. In addition to turning in its SQL source code, please make sure that you submit a labelled version of this schema visualization that gives its name and the parameters used to create the diagram.

Summary of the Required Deliverables

You and your partner should always use a Git repository, hosted by Bitbucket, to store the schema visualizations and all of the other deliverables required by this assignment. The repository must be shared with the instructor and the version control log should accurately reflect each student's contribution to this assignment. In addition, this assignment invites your partnership to submit one printed version of the following deliverables; each member should write and submit their own version of the first deliverable. Please see the instructor if you have questions about these matters.

1. A two paragraph commentary on the work that each team member completed.
2. An overview of how the `sqlt-graph` command-line options influence the resulting diagrams.
3. A comparison of the University visualizations from the textbook and the `sqlt-graph` tool.
4. A representative sample of the visualizations that you produced for the provided schemas.
5. A brief commentary on the visualizations produced for each of the provided schemas.
6. The complete and commented SQL source code for the student-designed relational schema.
7. The final visualization of the student-designed relational schema, as produced by `sqlt-graph`.
8. A reflection on the challenges that you faced when completing this laboratory assignment.

In adherence to the Honor Code, students should complete this assignment while exclusively collaborating with the other member of their team. While it is appropriate for students in this class—who are not in the same team—to have high-level conversations about the assignment, it is necessary to distinguish carefully between the team that discusses the principles underlying a problem with another team and the team that produces an assignment that is identical to, or merely a variation on, the work of another team. Deliverables from one team that are nearly identical to the work of another team will be taken as evidence of violating Allegheny College's Honor Code.