COS 480/580: Database Management Systems

Sudarshan S. Chawathe

University of Maine

Fall 2007

This course covers database systems from the perspective of database designers and database programmers (not to be confused with database system implementors). The emphasis is on fundamental topics that should be familiar to every computer scientist and good programmer. In addition to traditional topics such as Entity-Relationship modeling, relational database design theory, relational algebra and calculus, SQL, and Datalog, the course also covers object-oriented and object-relational databases, with topics such as ODL, OQL, and SQL3.

News and Reminders:

- **Please read the newsgroup for timely announcements.**
- Class newsgroup: Local group `umaine.cos480` on NNTP server `news.cs.umaine.edu`. Web interface to get started: `http://cs.umaine.edu/~chaw/news/`.
- You may use last year's Web site to get a rough idea of the homeworks and exams: `http://cs.umaine.edu/~chaw/200609/cos480/`.
- You may access a local copy of the PostgreSQL documentation (with a slightly improved formatting) at `pgsql/doc/html/`. In particular, the section describing *psql* is at `pgsql/doc/html/app-psql.html`.
- Please use the PDF version of this document for printing and reference: `cos480.pdf`

Contact Information

Class meetings:

Time: Tuesdays & Thursdays, 12:30–1:45 p.m.
Location: Neville Hall, Room 210.

Instructor: Sudarshan S. Chawathe

Office: Neville Hall, Room 224.
Office hours: (Please check for changes.)
Tuesdays & Thursdays: 10:30–11:00 a.m., 1:45–2:00 p.m., 3:15–4:00 p.m.
Phone: +1-207-581-3930.
Email: `chaw@cs.umaine.edu`
Use email only for messages unsuitable for the newsgroup. (See below.) Please put the string *COS480* near the beginning of the Subject header of your messages to me.
Web: `http://cs.umaine.edu/~chaw/`.

Teaching Assistant: Mark Royer

Office: East Annex, Room 229.
Office hours: (Please check for changes.)
Mondays & Wednesdays: 1:00–4:00 p.m.
Phone: +1-207-581-2005.
Email: `mroyer@cs.umaine.edu`

Online Resources

Class Web site:
`http://cs.umaine.edu/~chaw/cos480/`
We will use the class Web site for posting announcements, homework assignments, hints, solutions, etc. Please monitor it.

Class Newsgroup: We will use the local USENET newsgroup `umaine.cos480` on the NNTP server `news.cs.umaine.edu` for electronic discussions. If you are unfamiliar with USENET, you may find the Web interface at `http://cs.umaine.edu/~chaw/news/` useful as a quick way to get started. You may find further information on USENET at `http://en.wikipedia.org/wiki/Usenet`.

Class mailing list: Please make sure you are on the class mailing list. A sign-up sheet will be circulated at the first class meeting. If you

missed it, you need to contact me to get on the list. We will use this mailing list only for urgent messages because all other messages will go on the class newsgroup. (I anticipate fewer than a dozen messages on this list over the semester.)

Grading Scheme

Grades: Grades will be based on class participation (5%) homeworks (15%), two mid-term exams (15% each), a final exam (20%), and a project (30%).

Class participation: Students are expected to contribute to learning by asking questions and making relevant comments in class. Quality is more important than quantity. Disruptive activity contributes negatively. Please make sure all disruptive devices are disabled while in class.

Homeworks: Homeworks include programming and non-programming ones. No collaboration is permitted. You are allowed to discuss the problems at a high level, but the final solution must be your individual work.

Exams: All exams are open book, open notes. You are free to bring with you any resources that you find useful. However, no communications are permitted other than between students and me.

Project: In addition to the programming and other homeworks, the course features a semester-long group project. Students will work in groups of three or four to design and implement a substantial database application. Projects will be graded based on a written project report, the submitted source code, a demonstration, and a question-and-answer session following the demo.

COS 580: There will be additional readings assigned to COS 580 students. The readings will be a mix of some classic papers of the database field and more recent publications. COS 580 students are expected to be comfortable reading such papers. There will also be additional and/or different questions on the exams and homeworks. Similarly, COS 580 students will be held to a higher standard during the question-and-answer session following the project demo.

Policies

Special needs: If you have special needs of any kind (including, but not limited to disabilities, absences due to participation in sports or other activities, etc.) please contact me *as soon as the need is known to you* and I will try to accommodate them as much as possible.

Attendance: Although I expect students to attend all class meetings, I will not be taking attendance. If you miss a class meeting, you are responsible for making up the lost material. If you have a valid reason for missing a class, let me know early and I will try to help you make up the class. (See above.)

Make-up classes: I may have to reschedule a few classes due to my other professional commitments. I will make every attempt to minimize the number of such occurrences and to reschedule for a time that works for most students. Further, I will make sure no student is penalized by such occurrences.

Due dates: All due dates are strict, as announced in class. If you believe your work was delayed by truly exceptional circumstances, let me know *as soon as those circumstances are known to you* and I will try to make a fair allowance. However, the default is that you get a zero if you don't turn in the work on time.

Academic honesty: I expect you to hold yourselves to the highest standards of academic honesty. Please take this point very seriously. If you are not sure if something is permitted, check with me. All help you receive, even if permitted, must be prominently noted in all work you submit. Plagiarism and other forms of cheating will result in very stiff penalties (including, but not limited to, an F grade in the course and further disciplinary action from the university).

Programming

Programming: We will use PostgreSQL as the database system for programming assignments. You are free to program in any programming language you choose. However, if you are likely to need assistance, you should check with me before making your decision.

Class accounts: Class accounts for Unix and PostgreSQL will be generated based on the forms distributed at the first class meeting. If you missed them, please get in touch with me. You should be able to access your accounts from anywhere on the Internet (including the labs in Neville Hall and elsewhere on campus) by using *ssh* to connect to `cs.umaine.edu`. On most Unix hosts, the command `ssh -l` *`username`* `cs.umaine.edu` should suffice. For Windows hosts, the freely available *Putty* program works well: `http://www.chiark.greenend.org.uk/~sgtatham/putty/`. *Do not use unencrypted telnet sessions to connect to your account!*

Schedule

At the beginning and end of each class, I will announce sections of the textbook covered in each class and those due at the next class. An approximate schedule appears in Figure 1. Please use it only as a rough guide to plan your studies. Do *not* use it to schedule travel or other events. If you need a definite answer on when something will or will not occur, you should check with me.

Textbook and Readings

Textbook: *A first course in database systems.* Jeffrey D. Ullman and Jennifer Widom. Prentice-Hall, Upper Saddle River, New Jersey, second edition, 2001.

The textbook's Web site has many useful resources: `http://www-db.stanford.edu/~ullman/fcdb.html`. In particular, for a more detailed listing of course topics, please refer to the textbook's table of contents: `http://www-db.stanford.edu/~ullman/pub/fcdb-toc.txt`

Readings: Items marked with ⋆ are required for COS 580 students. COS 480 students may wish to read them if they plan to attempt the extra-credit questions on tests. Readings marked with ⋆⋆ are extra credit for COS 580 students and double-extra credit for COS 480 students. Students who wish to receive credit for ⋆⋆ items must discuss the specifics with me first. Everyone is encouraged to at least browse all the readings.

#	Date	Material
1	*09-04*	3.0, 3.1, 5.0, 5.1, 5.2.
2	-06	6.1, 6.2.
3	*-11*	HW1 assigned; 6.3, 6.4.
4	-13	6.5, 6.6.
5	*-18*	6.7, 5.3, 5.4.
6	-20	1.*.
7	*-25*	PR1 & HW1 due; catch-up; review.
8	-27	Midterm Exam 1; HW2 assigned.
9	*10-02*	8.1, 8.3, 8.4, 8.5
10	-04	8.6, 8.7
	-09	No class (Fall break Oct. 6th–9th).
11	-11	2.1, 2.2; Readings 1 & 2.
12	*-16*	2.3, 2.4.
13	-18	HW2 due; HW3 assigned; 3.2, 3.3.
14	*-23*	3.4, 3.5.
15	-25	3.6, 3.7.
16	*-30*	HW3 due; catch-up; review.
17	11-01	Midterm Exam 2.
18	*-06*	7.1, 7.2.
19	-08	HW03 due; 7.3, 7.4.
20	*-13*	4.1, 4.2, 4.3.
21	-15	PR02 due; 4.4, 4.5, 4.6, 4.7.
22	*-20*	9.1, 9.2, 9.3.
	-22	No class (Thanksgiving break Nov. 23rd–26th).
23	*-27*	9.4, 9.5.
24	-29	10.1, 10.2; Reading 3.
25	*12-04*	10.3, 10.4.
26	-06	8.2, extra
27	*-11*	demos; catch-up; review
28	-13	demos; catch-up; review
29	-18	Final exam, **9:30 a.m.–11:30 a.m.**; location TBA.

Figure 1: ***Approximate*** *Schedule. The numbers refer to sections in the textbook. PR denotes project report and HW denotes homework. Tuesday dates are in italics.*

1. Edgar F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, June 1970.

2. **Notes** on Codd's paper: `notes/rmodel.pdf`; `notes/rmodel/rmodel.html`.

3. [A recent paper for 480 and 580 will be added here.]

4. ⋆ Goetz Graefe. Query evaluation techniques for large databases. *ACM Computing Surveys*, 25(2):73–170, June 1993.

5. **Notes** on Graefe's paper: `notes/qeval.pdf`; `notes/qeval/qeval.html`.

6. [A recent paper for 580 will be added here.]

7. ⋆⋆ François Bancilhon and Raghu Ramakrishnan. An amateur's introduction to recursive query processing strategies. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)*, pages 16–52, Washington, D.C., May 1986.

Further Reading: These books are *not* required reading and nothing in the course will depend directly on reading them. However, they are good sources for different explanations of some concepts, additional information on various topics, examples, and exercises.

1. Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.
   This book is a textbook for COS 598, Advanced Topics in Databases, and focuses on Database Theory. The book is not light reading but it is much easier than reading the equivalent set of papers.

2. Hector Garcia-Molina, Jeffrey D. Ullman, and Jennifer Widom. *Database Systems: The Complete Book*. Prentice-Hall, 2002.
   The first half of this book is essentially identical to the main textbook. The second half covers topics in database system implementation, and is a good resource for learning more about how database systems are implemented. Since the terminology and style is consistent with the main textbook, it should be easy reading.

3. Christopher J. Date. *An Introduction to Database Systems*. Addison-Wesley, Reading, Massachusetts, 2000.
   A classic database-systems textbook.

4. Raghu Ramakrishnan and Johannes Gehrke. *Database Management Systems*. McGraw-Hill, third edition, 2002.
   Another standard textbook with detailed coverage of some topics that we will cover briefly.

5. Roderic Geoffrey Galton Cattell. *Object Data Management: Object-Oriented and Extended Relational Database Systems*. Addison-Wesley, Reading, Massachusetts, 1994.
   A good introduction to object and object-relational databases.

6. François Bancilhon, Claude Delobel, and Paris Kanellakis. *Building an Object-Oriented Database System: The Story of O2*. Morgan Kaufmann, 1992.
   Another good book on object databases.

7. Michael Stonebraker and Joseph M. Hellerstein, editors. *Readings in Database Systems*. Morgan Kaufmaann, San Francisco, California, third edition, 1998.
   This collection of papers, including some classics, provides a sampling of topics in database system implementation.

Homeworks and Tests

Homework assignments, exams, and solutions will appear here as we move along the semester. It may be useful to refer to the homeworks and tests from the previous session: `http://cs.umaine.edu/~chaw/200609/cos480/`.

- Homework 1: `hwq/hw01.pdf`. sample solutions: `p/hw01s.pdf`.
- Midterm 1: `hwq/mt01.pdf`; sample solutions: `p/mt01s.pdf`.
- Homework 2: `hwq/hw02.pdf`.
- Midterm 2: `hwq/mt02.pdf`; sample solutions: `p/mt02s.pdf`.
- Homework 3: `hwq/hw03.pdf`.

Project Reports

The requirements for both intermediate and final group-project reports are described below. The intermediate reports do not affect your grade directly

(they have 0 weight) and are thus optional. However, their submission is strongly encouraged to enable early feedback on projects.

Instructions for all reports: The quality of your report is an important component of your final grade (roughly a third of your project score, or 10% overall), so please make sure you pay attention to organization, grammar, punctuation, typography, layout, and overall clarity. Word and page limits are strict. Figures contribute to the word-count an amount equal to the number of words in the figure. The first page of each report should include appropriate title and identifying information (group name and group members). Do *not* use a separate title page or cover page. Do not include sensitive information (SSNs, passwords, etc.) anywhere in your report. *In each report submission, include all previous report submissions in an appendix.* You should also include a section in each report detailing how you have addressed (or plan to address) any comments I may have made on your earlier reports. You are also free to include additional appendices containing information you believe to be useful to the reader. However, like all appendices, the material in such appendices should not be required to understand the report. That is, you should not assume that the appendices will be read. The appendices do not contribute to the page or word limit.

Submission: You should submit your project reports electronically in PDF format. Your file should be named using the scheme *gname*`-`*X*`-`*N*`.pdf` where *gname* is the name of your project group (e.g., `widgetmasters`), *X* is one of `pr1`, `pr2`, and `pr3` for the first, second, and third project reports, respectively, and *N* is an arbitrary 4-digit number. For example, the *widgetmasters* group may submit a file `widgetmasters-PR2-4242.pdf`. Please use only lowercase letters in *gname* and use *gname* consistently for all submissions. For the source-code submission, use the file-naming scheme *gname*`-src.tgz` for the tarred, gzipped package containing your source code. You should upload this file by anonymous FTP (`anonymous` as the user name and your email address as the password) to the FTP server `cs.umaine.edu` in directory the `/incoming/cs/cos480/`. If you need to upload an updated version of your submission for any reason, you can follow this procedure again using a different four-digit integer in the file name. If you try using the same file name as your earlier submission, the upload will likely fail. I will grade the most recent submission before each deadline. You will not be able to list the FTP upload directory (standard security setup), so pay attention to the diagnostic messages from your FTP program.

Project Report 1: The most important parts of this report are the choice of group members and the choice of a suitable application. The maximum length is two pages. This report is optional. Points to cover:

- Group name: short and sweet.
- Group members: For each member, include full name (as registered), preferred name, major, and year.
- A brief description of your proposed application from the end-user point-of-view. (How would you describe your proposed work to a potential customer who is not familiar with databases and programming?)
- A brief description of your implementation plans. You may wish to include a rough system architecture. Mention the programming languages, database systems (e.g., Oracle, PostgreSQL), and major libraries or components (e.g., Apache, PHP, Jserv) that you plan to use. (How would you describe your work to a classmate or other database-savvy person?)
- Progress report: Outline what you have done so far and what you plan to do next. Try to set up some milestones for yourself.

Project Report 2: The maximum length is 10 pages. This report is optional. Points to cover:

- Any revisions to your project description or implementation plans.
- Conceptual (ER) model for your database application. Please make sure that you follow the standard conventions as described in the textbook and in class. Any additional features that you need should be included as annotations. Try to include as many constraints as you can (as annotations). Include explanations for any constraints that are not obvious. (For example, if some constraint is the result of your design decision to allow at most one shopping cart per registered user at any time, make sure you explain this reason.) Remember that the ER diagram is a design tool and your work will be evaluated for good design. Simply submitting a syntactically correct diagram will not get you very far if the

design is poor (or poorly explained). Feel free to include English explanations as needed in the main body of your report.

- The translation of your conceptual model to a logical (relational) model. *You must include details of all steps of this process, including the mapping of ER concepts to relation, the enumeration of functional dependencies, normalization, and any additional transformations.*

- A summary of the final logical model derived above.

- Partial physical model: Include create table statements that illustrate the attribute types. Explain non-obvious design choices (e.g., if you use an integer type instead of a date type for date-of-sale).

- Progress report, as in the earlier report.

- Appendix with earlier reports, if any.

Project Report 3: The final report consists of the following parts. Please note that, except for the first two parts, there is no limit (neither lower nor upper) on length. You should not feel the pressure to write a certain number of pages. For example, you don't need to write 10 pages of user documentation to get a good score. If you can say all that needs to be said in five pages, it's fine.

- Summary of work: This part should be no longer than 10 pages. It should include a clear description of your application and a high-level description of the functionality you implement. This part is your chance to make sure you get credit for the parts of your project work that may not be obvious. Be sure to highlight the novel, interesting, difficult, or otherwise noteworthy parts of your project.

- All the material required for the earlier project reports, subject to the corresponding length restrictions. This part is in addition to the verbatim inclusion of earlier reports in the appendix, as noted below. The description here may differ (and typically will differ) from what you submitted in earlier reports. For example, if there were problems with your ER diagram and normalization, you should include the fixed versions here.

- User documentation: This part is what you would include with your application if you were shipping it as a product. Note that by user, we mean the person setting up your application, not the end user. (For example, if you built an online bookstore application, the user here denotes the person working for the bookstore, charged with setting up the Web site, not the person buying books.) It should include a description of how your application works (major modules, processes, flow of control and data, etc.). There is no fixed page limit for this part; 10 pages is typical.

- Developer documentation: This part should contain a detailed description of your implementation that would be useful to someone interested in extending or modifying it (but that is not needed by someone interested in only using it as-is). For example, you should mention here how additional functionality could be added, or how something could be implemented in a more efficient or easier manner. Do not shy away from pointing out problems in the current implementation. You will not lose points for it. In fact, if you clearly describe why something you implemented is not great, and how it can be improved, you'll get more credit. There is no fixed page limit for this part; 10 pages is typical.

- Appendix with earlier reports, if any.

Source Code: As part of your final submission, you should upload all source code packaged as a tarred, gzipped file. (See submission instructions above.) This package should include a README file that describes the files in your submission and indicates how to compile them and set up your application. (As a simple test, a classmate who reads your project report and the README file should be able to set up your application.) You should include all code (and HTML pages and scripts) written by your group. *Do not include compiled code and libraries.* Instead, indicate how to obtain and set up the libraries. (For example, you can say "We use the Apache server version x.y.z, which is available at http://www.apache.org/"; do not include the Apache distribution!) You should include a small sample dataset (no larger than 10 MB) so that someone setting up your application can test it easily. You should also include any icons needed to get your application running (e.g., logo for a online store, image of a "for sale" button). Although, unlike

the programming homeworks, we do not require that your submission compile with a single make command, you are required to provide enough detail to enable someone else to set up your application, so please check to make sure you've included all the necessary files and instructions.