4003-486/4005-772 **Database System Implementation** Fall 2010-1/Raj

P3 Handout

September 4, 2010

Objectives

- To select and analyze an existing open source database engine.
- To identify and enhance features in the selected database engine.
- To produce a research report documenting your work (including analysis, design, and related work).
- To present and demo your work to the instructor during class.

Each team, typically of size 3, will select and then analyze an open source database engine its design and implementation. The team will then propose a few extensions (e.g., aggregation functions, query optimization methods, or additional access methods) to its selected database engine, and then design and implement these extensions. The team will also get a chance to re-assess and revise its selected extensions (to ensure implementation during the quarter), and then present their project to the class and submit the final report.

Open-source Database Implementations

Here are some open source database engines that students have explored in the past:

- http://www.sqlite.org
- http://www.h2database.com
- http://www.oracle.com/technology/software/products/berkeley-db/index.html (C or Java)

There are additional alternatives. For example, if your team would like to work on another open-source database engine, a good place to look for alternatives is:

- http://java-source.net/open-source/database-engines

Yet another alternative is for you to

- Find and select a modern open-source "NO SQL" database system.

If you decide to do either of these alternatives, you must get instructor approval before proceeding. Email me a short description of the engine and schedule a meeting with me to discuss your selection.

Initial Schedule and Deliverables *(Dates may be revised so monitor myCourses calendar)*

Date	Event	High-level Description
Sep 15	Team formation	In class. Each team must be only graduate or only undergraduate.
Sep 22	Phase 0	Install source code, compile, set up a simple database, and submit interaction log with your own installation.
Oct 13	Phase 1	Before submission, schedule a meeting to discuss your ideas with the instructor. Detailed engine analysis and initial detailed proposal for database extensions.
Oct 27	Phase 2	Initial implementation of extensions and revised project proposal.
Nov 08, 10	Demo	Demo/presentation in class.
Nov 10	Phase 3	Final submission.
Weekly	Peer feedback	Each team member will confidentially report on the contributions of other team members; see Surveys area.
Nov 12	Overall peer evaluation	Each team member will confidentially report on the overall contributions of all team members; see Surveys area.

Note: Ongoing discussions with instructor are critical to P3 success!

Phase 0: Installation of Source Code

Download the source (preferably using CVS or similar code configuration management tool), compile it for your chosen platform, and install it.

Write a simple database application (use a subset of any previous application or tutorial from the web and make it work with your installation); at minimum, you should show how you created a table, do the CRUD (insert-select-update-delete) operations, and dropped a table. Submit an SQL log that shows your interaction with your installation to convince me that it works.

Phase 1: Analysis of Engine and Initial Proposal for Extensions

- Present a summary of your understanding of the design and functionality of each component of the existing database engine. Include a discussion of the limitations of each component and overall software.
    - First, identify which open source product you decided to use and why. Second, focus on areas such as (a) relations and indexes, (b) record insertion into files, (c) key values and addresses insertion into the indexes, (d) page fetching, (e) query processing (and any optimization) steps. Also, focus on limitations of the existing software.
- Describe two or three advanced database features you plan to add to this open source software package; the features must be selected from different categories shown:
    1. Storage management (including data layout, indexing, buffer management, etc.)
    2. Query management (query analysis, compilation, planning, etc.)
    3. Transaction management (including recovery from failures)
    - While you are free to develop your own extensions, some likely extensions are listed here to give you a flavor of what might be considered appropriate.
        - Additional query processing capabilities such as joins or nested queries.
        - Support for multimedia (images, audio, or video) data files.
        - Support for a richer web interface for querying and data aggregation.
    - Graduate students (consult with instructor) must include some research ideas (see papers in the recent issues of SIGMOD or proceedings from conferences such as SIGMOD, ICDE, or VLDB).
- Discuss why you decided to implement these proposed features and outline the approach you will use. Your knowledge of database applications (from 485 or 771!) should help you guide your choices for features here. For each feature, identify (a) potential benefits, (b) basic components (e.g. commands or modules) you will need to incorporate into the open source package, and (c) any difficulties anticipated in adding these features.
- For full credit, identify files and the lines of code that you plan to change code for this project. I expect you will do different things as you start code modification, but I need to see that you are working seriously on what actual changes will be needed.
- <u>After your team has made some initial progress toward this deliverable, schedule a meeting with the instructor to discuss your extensions.</u>

Phase 2: Initial Extensions and Revised Proposal for Extensions

- Submit working code that demonstrates your attempt to implement the selected Phase 1 database features.
- If your approach looks unlikely, or your code is not working, or you plan to make changes to your Phase 1 proposal, submit a revised proposal, stating your reasons to do so in detail.

Phase 3: Final Submission and Demo

- Submit working code that demonstrates your attempt to implement the advanced database features selected in Phase 1 or revised in Phase 2. Document your project using a report that provides the conceptual and implementation design for each of the three advanced features, along with any limitations of your implementation. Your final implementation should have a simple interface (GUI if possible) to demonstrate your engine's new functionality, as you are required to demo this in class.
- A report describing design and implementation considerations, system architecture, and design choices for the improved database engine. This document must be written up as an ACM style research paper following ACM guidelines: http://www.acm.org/sigs/publications/pubform.doc
    - Graduate students must relate this work to current research in data management, and describe this relationship clearly.
- Outline the accomplishments (or failures) of your project.

Note: Ongoing discussions with instructor are critical to P3 success!

- Demo/present your project to the instructor outside of class hours; a sign-up schedule will be posted.
- The final submission must include the following:
    - Source code of your system. Include relevant files and ensure your changes are well documented.
    - A README instruction file that describes the submission. The description must clearly identify which team member made what changes to the original package.
    - The report.
    - Soft copy of any presentation and supporting material.

Additions and Omissions

As we progress through the quarter, more details about each phase will be provided as needed. Do not assume that all required information is available in this handout. Au contraire … it is not! The project gives your team full control to make decisions within these broad guidelines. Make appropriate decisions, but clearly document each decision and its rationale, and include these in the final project report.

Submission Information

For each submission, place the files to be submitted (e.g., `*.c/java, *.sql, *.txt, *.doc, READMEs, and Makefiles`) in directories named `phase1`, `phase2`, and `phase3` respectively in your CS group account on a CS department SUN machine (even if your setup is on a different platform). Each submission should include all source files that you modified (even those submitted in earlier project submissions). Include all required files in each submission to make it self-contained. Include a README file that describes where to copy your modified files to a standard installation of your open source engine's code.

Submit each deliverable by 11:59pm on the due date electronically using:

```
submit -v rkr-grd dbsi-phase0 phase0
submit -v rkr-grd dbsi-phase1 phase1
submit -v rkr-grd dbsi-phase2 phase2
submit -v rkr-grd dbsi-phase3 phase3
```

- Name your directories as indicated and use your group CS UNIX accounts for each submission.

P3 Evaluation

Step 1: Team Score

Phase 0	5%	
Phase 1	15%	
Phase 2	20%	
Phase 3	20%	
Poster/demo	20%	
Paper (including documentation)	20%	
Overall team score	**100%**	
Scaled team score	**70%**	(70% of overall team score)

Step 2: Individual Score

Team score to individual adjustment	± **a**	Adjustment up or down based on (a) peer evaluation feedback, and (b) individual P3-related performance during demo and quarter (as observed by instructor).
Individualized team score	70%	Scaled team score ± **a**
Weekly status updates	10%	Individual submissions and scoring
Team peer evaluation	*-20%*	Only if you fail to submit a well-detailed overall peer evaluation
P3 questions during final exam	20%	If a student's performance on P3-related questions is significantly better or worse than her or his team's, individual grade will be further adjusted up or down.
Final P3 score	**100%**	See P3 component weight in the Course Syllabus handout.

Note: Ongoing discussions with instructor are critical to P3 success!