Data Modeling and Databases
(Project Description [7 pages])

Publication Management System

Dainfos Laboratory

September, 2015
Innopolis University

Overview

In this project you will develop a complete system to manage the publication records. The overall project is divided into four phases; 1) Design and Implement Relational Model using an existing DBMS; 2) Develop web based user interface to interact with the database created in phase 1; 3) Develop your own DBMS and replace with the DBMS used in phase 1; 4) A realtime dashboard for database tuning; 5) Novelty/Creativity.

- Phases 1 and 2 are compulsory for all the students.
- Phase 3 is compulsory for BS3 and Master students.
- Phases 4 and 5 are compulsory for Master students in Sadegh's group.

All the students are welcomed to work on all the phases. If a group manages a not compulsory phase, the weak points of their compulsory phases can be compensated. Each phase is explain in detail in the subsequent sections.

Phase 1: Design and Implement Relational Model

The objective of this phase is to utilize the database knowledge for real life problems.

1. Design the relational model and transform them into relations (create ER diagram and translate it into Relations).
2. Design the relations in an existing existing database management system (DBMS) by creating physical tables and their relationships.

There are many repositories containing research articles with their information, e.g., DBLP, Google Scholar, ACM digital library. The research articles are usually categorized based on their venues, authors, types and years. A publication record reference in research articles can have several attributes along with the article name. For example:

- Sadegh Nobari, Farhan Tauheed, Thomas Heinis, Panagiotis Karras, Stéphane Bressan, and Anastasia Ailamaki.
- ”TOUCH: in-memory spatial join by hierarchical data-oriented partitioning.”
- In Proceedings of SIGMOD ’13.
- ACM, New York, USA, 701-712.
- http://doi.acm.org/10.1145/2463676.2463700

Functionality

- Select, Insert, delete, and update publication records
- Import at least 1 million publications that are crawled from an existing publication repository, e.g. DBLP or Google Scholar, ACM digital library or etc. Realtime crawling is a +.
- Search the publication based on research area, author name, publication year, venue(conference/journal name), title, keyword, type of paper, institution or *related articles* and obtained ordered results.
- Ability to query related articles, based on your own defined methods, at least two methods.
- Ability to sort papers based on your own defined ranking methods, at least two methods.

Phase 2: Web-based User Interface

After creating the physical database, you need to develop a web-based graphical user interface (GUI) on top of it to interact with the database. It helps the users to carry out all the specified functionalities. It is more like Content Management System CMS. Only authorized/registered users are allowed to use this interface.
The web interface must neatly report the results with ranking. A nice visualization of the results, like showing graphs for related articles, is a +.

Phase 3: Develop DBMS

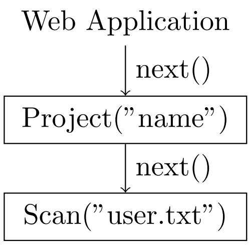
The goal of this phase is to replace the relational database with your own implementation of a database. You are asked to implement query operators (scan, select, project, join, group by, insert) using a standard programming language. You will replace all SQL code with calls to the query operators. Note that from the user's point of view, nothing will change on the front end side of your web application. These are the requirements of the third phase of the project:

- Your database must provide methods to query data, as well as insert new record and update existing records.

- Your database must at least implement the following query operators: scan, project, select, sort, join, group by, and insert.

- In particular, your database should at least answer queries that join two tables.

- Use the iterator model to implement query operators. That is, operators should pull tuples from underlying operators using next() calls. For example, the query "SELECT name FROM users" can be answered using the two operators project and scan:

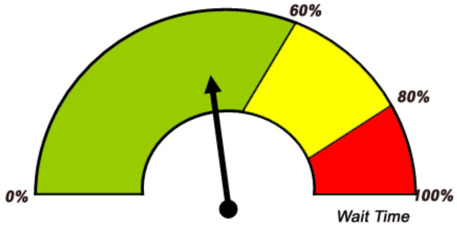


- Indexing on Primary keys with ability to add index on other attributes.

- You are only allowed to use a single file for your entire DBMS.

Phase 4: Database Tuning

A Database Performance Dashboard is a useful tool for database administrators (DBAs) to monitor the health of the database and to support database performance investigation. The performance dashboard provides a one-stop overview of various key performance indicators (KPIs) on database performance. Users can then make use of the high-level information to further zoom in to identify the performance issues, and tune the system to improve performance.



Develop a realtime web-based application to help provide DBAs with a performance dashboard with key database statistics. The application should monitor the following database parameters:

- disk I/O
- memory space
- cpu time

You are encouraged to define additional parameters that need to be monitored. You should provide a discussion in the report to explain the rationale for including these additional parameters.

The web-based application should provide the following features:

1. 3 levels of breakdown for each database parameter being monitored.
   a. Top-level breakdown: Shows the health of the specific database parameter being monitored (green: healthy, yellow: not so healthy, red: need DBA attention)
   b. Second-level breakdown: Shows the aggregated values (e.g. total-cpu-time) for the database parameter per x unit time block as specified

by the user.
c. Low-level breakdown: Shows the aggregated values for the database parameter per y unit time block (y ¡ x) within each of the x unit time block. For example, a user can specify x to be 1 hour, and y 15 minutes. Thus after collecting data for 24 hours, besides seeing an average value for the 24 hours, the user can view the average for 0-1 hour block, 1-2 hour block, and so forth. User can also zoom in to look at a particular block, say 3-4 hour block, and (s)he gets to see the average values of the 4 15-minute blocks within that hour.
d. For each parameter provide full details upon user request through the dashboard GUI (e.g. Select spends $X\%$ of the cpu-time).

2. Provide a configuration page(s) to define the various thresholds used to determine whether a specific database parameter is healthy, not so healthy or need DBA attention.

3. For each of the performance issue identified that is in the "red" area, the application must record the incident in a log file (or a table) and optionally provide hints.

4. On-demand reports – where the users can specify the date ranges for the database parameters being monitored.

5. A Debug interface - Allows the user to issue SQL commands to the database. Both the results and the performance metrics should be displayed neatly on the webpage.

Phase 5: Novel/creative idea

At least one novel or creative idea to be implemented, explained in the presentation and written in the report. For example finding paper/authors/groups of interests/trendy. For example doing distributed/parallel processing. For example performing smart indexing.

Deliverables

1. Program source codes

2. README.txt: instructions on how to setup the web application to be evaluated.

3. Database Setup Scripts

   - setup.sql: scripts to create the tables that are used to support the application.
   - data.sql: pre-populates the tables with the data used by the application. The data must contain at least one million publications.

4. Project Report in pdf using ACM SIG Proceedings Templates [`https://www.acm.org/sigs/publications/proceedings-templates`]. The report should not exceed 15 pages.

   - Tasks: Breakdown of the key areas done by each member
   - System Design and Architecture: The overall architecture and design of the application. Discussion on the database design for the tables used to support.
   - The application: Details on how the various components developed.
     - the SQL commands used to obtain the statistics
     - Screenshots of the application
     - Highlight specific novel ideas that your team has used in the application
     - Insights of your learning during the course of doing the project.

5. 10 minutes presentation and demo.

Final notes

- Failing the project results to 'F' (Fail) grade for the course. This project will count 15% of the final grade.
- Student's are allowed to form groups up to 3 people. All students will be graded individually, and each student's understanding of the entire project they've been working on will be tested.
- Each group performs 10-minute presentation of the project.
- PostgresSQL is suggested for the database management system and any high-level programming language is allowed (Java, C++, C#, Python, PHP).
- It is not allowed to use object-relational mapping techniques such as Hibernate, Ruby on Rails and etc. You should implement all SQL queries yourself and be ready to present them.
- It is not allowed to use javascript to sort tables or results.