CS 387 Project Guidelines and Expectations

Umesh Bellur

Lab Grade and Project!

- Project carries 60% of grade for CS 387
  - Assignments carry the other 40%
  - Notice the change!
- 4 person team (with 59 registered, one 3 person team is acceptable)
  - Grading will be uniform for all team members

Project Goals

- *To construct a real system that uses relational DBs.*
  - *Schema + data*
  - *Front end for people to use*

Software Engineering Exercise

- Conceptualization & Feasibility
  - Teams – roles within a team
  - Choose project domain
- Specification
  - Detail the functionality
- Design
  - ER, Relational, Schema, Front end
  - 3 Tier systems are acceptable.
- Implementation
- Testing
  - Test plans, testing
- Documentation and Demo

7 Stages

Stage #	Desc	Due on
1	Functional Spec	Aug 14th
2	E/R Model + Test plan	Aug 28th
3	Relational Model + Screen Design	Sept 18th
4	Normalization	Oct 1st
5	Schemas	Oct 10th
6	SQL + JSPs	Oct 30th
7	Tests + Final Demo	Nov 1st + 2nd week.

Specification

- What are the application specifications (i.e., what functionality will your completed system provide)? Number this for trace ability – you will have to account for each one of them in your demo.
  - What aspects of the application will your system model? What will your system not model?
- Team formation
  - What is the role of each project member in the project? Be specific.

Specification (2)

- Extensibility
  - What other "value-added" facilities could your system support (but that you will not build explicitly)?

Some ideas

- **Bibliography database**
  - www.connotea.org
  - www.citeulike.org
  - DBLP etc.
- **Nobel Awards Database**
  - www.nobel.se/index.html
- **Movies Database**
  - regal.hollywood.com
  - www.imdb.com

Stage 2 - E/R Modeling

- Starting from the FS that you turned in for Step 1 of the project, design an E/R diagram for your application. Your model should provide

1. 4-6 entity sets,
2. a similar number of relationships,
3. one example of a multi-way relationship, and
4. one example of inheritance, and
5. (preferably) one example of weak entity sets.

Your design must satisfy the first two criteria. Satisfying the last three criteria is *optional*. If your design does not satisfy any of the last three criteria, you must state in your response, why you think it is "unnatural" for your application to involve multi-way relationships, inheritance, and/or weak entity sets.

Stage 2 - Test plan

- For every numbered use case, create a set of tests that will be used to verify whether the feature works correctly.
  - You need to test *correctness* **AND** *exception* handling.
  - Test *parameter verification* **AND** *backend functionality*
- Output is a document of test cases
  - Spreadsheets are a easy way of organizing this.

Stage 3 – RM and Screen Design

- Convert the ER to a relational model. This will NOT be normalized.
  - Deviations from the standard conversion procedure
- Screen designs
  - You can do them in HTML, Powerpoint, Visio or any other tool. This is to get an idea of the **layouts** and the **flow** of screens.
  - Throwaways may be easier to get an idea of.

Stage 4 - Normalization

- Are all the relations in your chosen schema in 3NF? Are they are in BCNF? Explain both answers.
  1. If any of your relations are not in BCNF, normalize them to BCNF. If you choose to normalize your relations only till they are in 3NF, explain your reasons (e.g., the amount of redundancy introduced is limited or some other valid reason).
  2. Notice that if you decide to go ahead with normalization, you will have to list the violating FDs for each of your relations and explain why you think they violate 3NF and/or BCNF.
- At this stage, if your relations are not in 4NF, normalize them into 4NF.

Stage 5 - Schemas

- SQL Scripts to create the DB schema you need for your project.
  - Constraints
- Data population scripts to populate the DB with test data.
  - Test data creation
  - We are looking at data in the order of 40-50 tuples for each relation in your application. Get it from some web source (making sure that it is public first, of course) or write a program in any scripting/programming language (like Perl, TCL or C) that creates large files of records in a format acceptable to the bulk loader. Either way, realize that you may need to transform data from one form to another to be acceptable for use in PostgreSQL.

Stage 6 – SQL + Front end

- Write the SQL that will allow you to build the functionality of the application.
- Create the front end (JSPs, PHP, Web2.0) and hook it up to the SQL using JDBC.
- This is the "implementation" stage of your project.

More on complexity

- Your queries should preferably illustrate several different aspects of database querying, such as:
  - Queries over more than one relation (by listing more than one relation in the FROM clause)
  - Queries involving aggregate functions, such as SUM, COUNT, and AVG
  - Queries involving complicated selects and joins, and/or sub-queries
  - Queries involving GROUP BY, HAVING or other similar functions.
  - Queries that require the use of the DISTINCT or ALL keyword.

A word of caution

Do not cook up query problems to cover each and everyone of the above aspects! You do not have to illustrate all the above aspects, just the ones that occur *naturally* in your application. Try to infuse some reality into your project and think of some reasonable queries that people would want to make with your application. For example, in a movie application, writing a query such as "Find all actors whose ages are three times more than their street number" sounds ridiculous!

Stage 7

- Run through your test plan and fix issues that you can.
- Submit a final copy of the test status against your plan and summarize against requirements.
- Create a presentation about your project (20 slides at most).
- Demo it in the class.

Grading

Stage #	Desc	Due on	Weight (%)
1	Functional Spec	Aug 14th	5
2	E/R Model + Test plan	Aug 28th	15
3	Relational Model + Screen Design	Sept 18th	10
4	Normalization	Oct 1st	10
5	Schemas	Oct 10th	15
6	SQL + JSPs	Oct 30th	35
7	Tests + Final Demo	Nov 1st + 2nd week.	10