CSC 540 - DATABASE MANAGEMENT CONCEPTS & SYSTEMS

FALL 2017 PROJECT 1 - DATABASE APPLICATION DESIGN & IMPLEMENTATION

PROJECT DESCRIPTION
COURSE ASSESSMENT SYSTEM

Introduction

The goal of the project is to design a relational database application for supporting course assessment at a university such as ours. The project should be carried **out in teams of four (4) a**nd team peer assessments will be part of the overall grade for each student. The project is expected to be executed in stages with interim deliverables submitted. The description has several details that you should pay attention to.

First, it describes the data and application requirements. It then gives information about deliverables and tentative deadlines (deadlines may change slightly) and "getting started" guidelines.
Note: You should necessarily assume that this is an imperfect description and will be subject to updates. Therefore, it is important that you read through the description in the coming days and ask questions to clarify any missing or ambiguous statements.

Project Specification

1. Background & Overview

The Assignment Management System Case Study:

Now that you have all used Gradiance you are ready to start thinking about your project. This case study describes an online automated system for managing homeworks/exercises similar to the Gradiance System. Listed in these sections following is a description of the data recorded, maintained, and accessed by the system to support the administration of its functions.

Courses

Every course has a course id or "token", a course name, start and end dates, a professor, one or more teaching assistants (TAs), the students enrolled in the course (can be added by TA or instructor), and a list of topics (e.g. The Relational Model, Disk and Storage, Security and Authorization, etc). Each topic will have a name and id. Each course is associated with a question bank that contain a list of possible questions relevant to the topics of the course. A course can be created by the professor, and the professor who creates the course is automatically

assigned as the professor teaching the course. A person enrolled as student for the course cannot be assigned as the TA for the same course, and vice versa.

Homework Exercises

A course may have several homework exercises. Exercises can be created by instructors and are made up of questions drawn from a question bank. An exercise has an id, a name, a deadline in terms of a time and date, Other attributes of an exercise include the total number of questions in the homework, the number of retries allowed (e.g. 1, 2, 3, unlimited), start date/time, end date/time, number of points per correct answer and penalty points per incorrect answer, and the scoring policy(explained in the section 'Scoring policy' below). Note that an exercise should not available to students before or after the start and end dates respectively (Although, they can be available to TAs and instructors). Only students who are enrolled in the course by TA or instructor can view homeworks for that course.

Each question in the question bank has an id, the actual question text, a topic, a difficulty level (ranging from 1,2,3,4,5,6) and optionally a hint. A question may have either a completely fixed or a parameterized structure: i.e. there is an overall "root" question whose structure contains both fixed and variable portions. Concrete questions can be created from parameterized questions by taking the root question and assigning values to the variables in the variable portions of its question structure. For e.g. for a root question such as "*if c1+c2=ans, where c1=x, and c2=y, find ans*", one possible concrete question can be obtained by assigning values to the variables *x* and *y* to produce "*if c1+c2=ans, where c1=2, and c2=3, find ans*."

Each question is also associated with a difficulty level, a set of correct answers, a set of incorrect answers and a single detailed solution/explanation. The detailed solution/explanation is only available to students after the deadline has passed. When creating a parameterized question, different concrete parameter value options must be provided for each variable. In this case, the set of correct and incorrect answers is separate for each parameter value combination that can be used to generate a concrete question e.g. for each *c1*, *c2* concrete combination.

Homework Exercise Creation and Generation

There are two possible exercise generation modes and the mode must be selected at the time of exercise creation: Standard and Adaptive. In the "Standard" mode, for each homework attempt by a student, the list of questions is a separate exercise instance and is generated by the system from the list of questions in the exercise created by the instructor.

There are two major ways that a **standard** exercise can be created by an instructor. They can search the question bank either by question id or by selecting from the list of topics for that

course which generates questions relevant to that topic. Alternatively, they could find an existing homework on the same topic and reset the parameters, e.g. start date, end date, points, etc and save it under a new name. They also can add and remove questions from the old homework. For both ways, since each question has a level of difficulty associated with it, at the point of exercise creation, the system should create a level of difficulty for the exercise which will be the average level of difficulty of all the questions in the exercise. Standard homework exercises are automatically generated for each student for each of their attempts. This means that for every attempt, the questions are shuffled and for exercises with parameterized questions, concrete questions are generated for that student. For example, question 1 in attempt one can be question 6 in attempt two. Also, concrete question in attempt one will have different parameters in attempt two etc.

Some special kinds of exercises can be created in the "adaptive" mode by the instructor. They are different from the aforementioned standard exercises in that the questions are dynamically selected by the system based on the performance of the student. For an adaptive exercise, the instructor does not need to add questions manually to the exercise. The system will automatically and dynamically select the next question (for the student taking the test) based on the student's performance in the current question answered. The first question of the exercise will be of difficulty level 3. If the student answers the current question correctly, the next question will be of a higher difficulty level (the difficulty level increases by one, till 6), else a question with a lower difficulty level (the difficulty level decreases by one, till 1) is presented.

For example: A student starting an exercise is given question one (difficulty level 3), if he fails the question, then questions 2 should be one with difficulty level 2, else if he answers correctly, question 2 should should have difficulty level 4.
These questions are automatically selected from the question bank, however, the instructor can also specify a topic so that the only questions under that topic in the bank will be selected.
Note that, here parameterized questions can also be selected by the system and the system will have to generate concrete questions with different parameters for each student.

Exercise Submission
After a student submits an exercise, a report is displayed showing the total score of assessment, and for question whether it was answered correctly or incorrectly. For the questions answered incorrectly, it displays the hint corresponding to the question. If the assessment allows for multiple retries and the student has not exhausted number of tries, and the due date/time for exercise has not passed, the student may re-attempt the assessment exercise. The details of all submissions are stored for each student. The final score for each student is selected based on the scoring policy selected for the exercise. If a student logs in after the end of the submission period, they may additionally see the more detailed explanation of solution that is associated with each question.

For exercises with multiple retries allowed, we want to keep all information about each attempt for each student – time of submission, answer id selected for each question and total number of points for that attempt. All the attempts of all students for the course can be viewed by both the instructor and the TAs at all points of time.

Scoring policy

If multiple retries are allowed for an exercise, the final score for each student is selected based on the scoring policy selected for the exercise. A scoring policy can be one of "latest attempt, maximum score or average score" of all attempts. This means that total score could be the latest, average or maximum of all attempts.

Roles and Access Control Requirements

Students and professors have unique ids. Students also have levels associated with them (undergraduate or graduate) and only graduate students may be TAs.

There will be different access control restriction based on the roles:

1. Instructor – An Instructor can see all questions and answers related to any topic in any course (even if he/she is not the professor for that course) at all times. Instructors populate the question bank by adding questions (fixed or parameterized). Only instructors can create exercises and can do so only for their own course. These and the roles below can be done by the instructors.

2. Teaching Assistant (TA) – A TA can only see exercises created by the instructor, and their corresponding questions. The TA does not have access to the entire question bank. TAs can see the exercise as soon as it is created (even before the start date of the exercise). However, TAs cannot make changes to exercises at any time. TAs can enroll students in the class. TAs can see the class roll and homework attempts and grades of all students. A TA cannot take the course for which he is a TA.

3. Student - The access restrictions for students is as described in throughout the document.

The following shows examples of query classes and queries that should be supported by your application:

Retrieval SQL queries - used to find specific information

Find students who did not take exercise 1.

Find students who scored the maximum score on the first attempt for exercise 1.

Find students who scored the maximum score on the first attempt for any exercise.

Retrieve all attempts for exercise 1 for Student 5.

Reporting Queries. – used to find more general information
For each student, show total score for each exercise and average score across all exercises.
For each exercise and question, show the maximum and minimum score.
For all exercises to date, show the average number of attempts.

Application Requirements

In general, your application should support only role-authorized actions. This will largely be accomplished through the combination of menus that present only appropriate actions for the given role and context, and advanced features like Views, Procedures or Triggers and Authorization. Additional specification about menu format will be provided in a few weeks. This will allow a consistent menu format that will make grading the projects easier. It is expected that your application should handle errors elegantly and not reset on every simple error. For e.g. the application should prompt for another input in case the user input was invalid. Providing user-friendly messages to users when actions are invalid will also be expected.

Application Flow

This part of description gives a general idea of what the application should be like.
The application entry point should be an account creation screen or login (if already existing) for Instructors, TAs and Students. After instructors log in they should be given options like: ***Create courses, View course etc.*** A more complete application flow description will follow in the not too distant future**.**

Sample Queries

Queries on your database will be helpful for assessing the quality of database design. However, it isn't possible to leave that to only demo day. Consequently, you will need to implement some queries as part of your project. The list of sample queries will be given shortly.

Project Deliverables

Questions on the Forum - Post any questions for clarification of description on forum. **Due Sept. 17**

Project Milestone 1 - Report: Due Sept 30th

For the first milestone, you should:

1. Fill in the form for deciding team members as soon as possible.
2. An ER-Diagram along with a list of Entity and Relationship Types that you identify in the project description. For each relationship type, you should state the arity of the relationship e.g. if it is binary, ternary, etc. Relationships should include any hierarchical relationships (subtypes) that you identify**.** There is no need for verbose text, just a categorized listing of these is fine.
3. Relational Model: A list of tables, 2 - 3 sentences description of each including what constraints (including referential constraints) it encodes, a listing of **functional dependencies**, a discussion of normal form choices faced and justification for the decision made.
4. For this report, you should list any application constraints that you identify in the description. Also, include a list of functional dependencies that are present.

5. A statement acknowledging that you have asked all questions you need to clarify any ambiguities in the description.

Project Milestone 2 - Report And Application: Due Oct. 30th

For this milestone, your team will need to submit a complete report which includes:

1. ER Diagram along with the listing of entity and relationship types and a sentence description for each and list of key, participation constraints and other constraints represented in the ER model, along with a sentence description for each.
2. Constraints: A description of constraints that were not implemented as part of table definitions and why and how they were implemented in the final design. In particular, a separate subsection here should highlight constraints that couldn't be implemented in the database at all and had to be implemented in application code. Note that a key part of assessing your design is how well you used the DBMS to implement constraints V/S implementing in application code.
3. Two SQL files: First one should contain SQL for triggers, tables, constraints, procedure. The second file should contain queries for populating the tables with the sample data. **Sample data will be provided closer to demo date.**
4. Executable file (e.g. - Executable JAR file) and source Java Code.
5. README.txt - This should contain the names of the members of the team and any additional instructions that you might want to add that will be necessary to compile and execute your code.
6. A peer review. A link will be provided to submit the review when the milestone will be due.

Project Milestone 3 - Demo: {Dates TBA}

Each team will have to book a timeslot to demo the application to the Professor or Teaching Assistants. The details about the demo will be conveyed later.

Getting Started

Everyone here must have already have one or two homeworks on Gradiance so must be familiar with the concept. The aim is to create a database schema which can be used by any university to implement a similar system. You should already have an understanding of the basic functionalities provided by these applications and the aim would be to model them into a database.

I would recommend using an IDE(like Eclipse) for developing the project and some form of version control like GitHub for sharing project amongst team members.(You can create private repositories on NCSU Github).Using Oracle's SQL-Developer would also help you in writing long queries, triggers and procedures because of advanced features like debugging and static analysis of query.

Following links will help you to connect to the database using the JDBC Driver:

Creating a connection using JDBC

Students are encouraged to read more about proper handling of connection and JDBC Best Practices.

Points will be deducted for **improper handling** of connection in the java application.

Grading

Deliverable	Milestone	Percentage	Deadline
Report	Milestone 1	5	Sept. 30th
ER Diagram	Milestone 1	10	Sept. 30th
Relational Model + Constraint	Milestone 1	10	Sept. 30th
Reporting Queries + Final Report	Milestone 2	25	Oct. 30th
Demo	Milestone 3	50	TBA
Peer Review		% of Average of peer review grades	TBA