CSC 540 - DATABASE MANAGEMENT CONCEPTS & SYSTEMS

SPRING 2017 PROJECT 1 - DATABASE APPLICATION DESIGN & IMPLEMENTATION

PROJECT DESCRIPTION
COURSE REGISTRATION SYSTEM

Introduction

The goal of the project is to design a relational database application for supporting course registration at a university such as ours. The project should be carried out in teams of four (4) and 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

A university needs a database application to support the registration of courses by students. The application will manage different kinds of data about students and courses subject to a variety of application requirements. You are required to design a database application that will support the given use cases. In some parts of this document, there might be statements about requirements that seem ambiguous or unclear or even gaps in the description which you are encouraged to request for clarification about, earlier rather than later. You should read the description several times in order to pick up all the subtle details. Your application development process will begin with developing an E-R diagram, translating that to a relational schema, data loading and then development of the application. You will also be given a set of queries to implement on your design. Your applications will be demonstrated in person on our scheduled demo days (TBA). Some additional guidelines about user interface will be given in a few weeks. Sample data to be used for the demo will be given closer to the demo period (in the meantime, while developing you should create your own data for development and testing).

2. Data Requirements

Courses - The university offers various undergraduate and graduate courses that are uniquely identified by a course id. Each course is associated with a title, department (identified by codes e.g. CSC, ECE),

number of credits and a set of prerequisite courses and conditions and a classification level e.g. graduate or undergraduate (identified by integers 1 or 2 respectively). For some courses, the number of credits is fixed and specified when the course is first added in the system (i.e. a new course addition). On the other hand, for some courses e.g. an independent study course, only a range of credits is specified at the time the course is added. However, for each individual registration in such a course, the number of credits should be specified.

Some courses have preconditions that a student must satisfy in order to enroll. Preconditions will include (i) a list of prerequisite courses and grade requirement - may also be at least or all semantics here (ii) overall minimum GPA requirement or (iii) special permission of department offering the course. These conditions may be captured as special codes (e.g. PREREQ or SPPERM - for special permission) in the database. By default, there is a precondition that applies to all course registrations - students cannot register for courses outside their level e.g. undergraduate students cannot enroll in graduate courses and vice versa with graduate students.

Courses may be offered one or more times in an academic year, and each offering in a semester has an associated schedule e.g. M, W, F : 10 am-11 am, location and list of faculty members that teach the course (a course could have one instructor but for some courses it is possible to have several faculty co-instruct). Each offering will have a maximum number of students that can be enrolled. Each offering has an instructor. Note that instructor or faculty information is needed, faculty is only an attribute of course offering. For each semester, there is a deadline for adding and dropping and students can enroll in any course on or before this deadline, provided they meet the preconditions for registering for that course. If a class is full, (i.e. enrollment has reached the class capacity, then the students are given the option to be added to the waitlist. Should they want to be placed on the waitlist, they will need to indicate which course they would like dropped from their schedule IF adding the waitlisted course causes the student a course overload (i.e. the total number of credits greater than the maximum number of credits). A student is given a waitlist number generated in the order in which the waitlist requests are made. However, a student is only eligible to be placed on waiting list, if they meet all other preconditions for the course. Further, the waiting list has a fixed number slots and therefore must still have available slots for a student to be placed on it. Therefore, if a student requests to be placed on a waitlist and other requirements are not satisfied e.g. prerequisite course conditions haven't been met or waitlist is full, the application should return an informative response stating why this isn't possible. When a student drops a course, space is created, and the first person on the waitlist is automatically enrolled in the course and the rest of the students on the waitlist move forward.

Students - All students are uniquely identified with a student id. They also have a first name, last name, department (identified by codes), GPA, email, account password, a level classification i.e. undergraduate or graduate student (identified by integers 1 or 2 resp.) and a residency classification which includes in-state or out-of-state and international (also identified by integers). All students depending on their level and residency classification can have either or both of maximum and minimum credit limit requirements. All undergraduate and graduate students have a maximum number of credits (but the number is different for undergraduate and graduate students). International students additionally have a minimum number of credits requirement due to visa restrictions.

Once a grade is entered for course, the GPA for a student is updated. GPAs are calculated in the usual manner. Students can view their current schedule, add and drop courses to their current schedule (subject to constraints such as no conflicts) and view the listing of courses already taken and grades assigned. They should also be able to see their GPA, modify personal information but not all of it (e.g. student id).

Each student's schedule is associated with a bill. There are three billing schemes (dollars per credit hour) based on level and residency classification. Students classified as undergraduates and in-state will have a different billing rate than that of undergraduates who are out-of-state or international students. As students add and drop courses (prior to deadline), their bills are adjusted to reflect the changes. Billing rate is determined by the level classification of the student not the level of the course. For example, if an undergraduate student is permitted to take a graduate course, that course is still billed at an undergraduate billing rate. You are not expected to maintain a billing history. Only the latest amount due is fine.

Administrators are responsible for adding new courses and new students to the system, entering grades and approve special permission requests to enroll in a class. Each administrator will have a unique employee Id (also the username), SSN, First and Last Name and a password which they will use to login into the application.
For the purposes of simplicity, we will assume that administrators work across all departments i.e. are not department-specific. For new courses they enter information such a unique course id, course name, department code, classification level i.e. grad or undergrad (identified by integers 1 or 2 respectively), number of credits and the required preconditions. For registering new students, they enter some required fields such as a unique student id, level(graduate or undergraduate), residency classification, a default password(same as student id), initial billing amount (if any), first name and last name. All the details related to the course enrollment and student's contact information like password, email address, personal address, phone number can be entered and updated by the student later. All the other fields like GPA, level and residency classification etc, can only be viewed by the student but a student cannot modify them whereas an administrator should be able to edit them. On the other hand, an administrator shouldn't be able to view or edit a student's password.
Administrators approve or reject the special permission enrollment requests. A student submits the request which includes their student id, the number of units and the course they want approval for. If approved, information about the approval (who and date) is recorded and the administrator updates the student's schedule to include the course.
After the registration phase has ended, administrators will initiate the process to clean up the rolls and the waiting list. For students that still have an outstanding bill, the Administrator will drop all classes they've enrolled in and remove them from waiting lists that they are on.

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. **The forum for project 1 has a sample**

used in previous projects so that you have an idea what to expect. 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. A more complete application flow description will follow in the not too distant future**.**

The application entry point should be an account creation screen or login (if already existing) for Administrator and Students. After students log in they should be given options like: ***Enroll for courses, View course (details: no of units, instructor and schedule), View GPA, View Bill, Pay Bill***. The administrator can ***Create a student, View all students and details, View all Courses and number of students enrolled, View pending courses that require his approval, Approve courses, Enter Grades.*** Another important menu option for Administrators is the option to clean up the rolls and waiting lists after registration is done (see the last paragraph of data requirements)

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

Project Milestone 1 - Report: Due Feb 15th

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

Project Milestone 2 - Report: Due Feb 28th

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.

Project Milestone 3 - Report And Application: Due April 1st

For the final 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. 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.
3. Constraints: A description of constraints that were not implemented as part of table definitions and why and how they were implemented. 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.
4. 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.**
5. Executable file (e.g. - Executable JAR file) and source Java Code.
6. 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.
7. A peer review. A link will be provided to submit the review when the milestone will be due.

Project Milestone 4 - 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 registered courses using the mypack course enrollment wizard 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	15th February
Report	Milestone 2	5	28th February
ER Diagram	Milestone 3	20	1st April
Relational Model + Constraint	Milestone 3	20	1st April
Report Queries	Milestone 3	20	1st April
Demo	Milestone 4	30	TBA
Peer Review		% of Average of peer review grades	TBA