**2025 Database Design Project**

**University Course Registration and Grade Management Database System**

**【Project purpose】**

The University Course Registration and Grade Management Database System is a core administrative platform that centralizes student records, course offerings, enrollments, and academic results. Implementing this system gives students practical experience across the full database lifecycle—requirements analysis, data modeling, schema design, SQL development, transactions and concurrency, indexing, and simple application integration—while reinforcing key concepts such as integrity constraints and query optimization. The project also requires implementing real-world rules (prerequisites, capacity limits, schedule-conflict checks), producing robust test data, ensuring basic security and concurrency correctness, and delivering documentation and artifacts suitable for a deployable university system.

**【Requirement】**

**Data assumption:**

1. The grade-management DBMS should assign each student a unique ID and store related information, including name, gender, date of birth, contact information, affiliated college, and specific major.
2. The system should include course and course-section (Section) information. Each course has a unique ID and contains the course name, credit value, affiliated department, and course type (e.g., general required course, major required course, major elective, university elective, practical course, etc.). Course sections are associated with a semester and the instructor, and each section has a maximum enrollment capacity.
3. The system should manage instructor information. Each instructor has a unique ID and may teach multiple courses (multiple sections), but may only teach one course at the same time slot (i.e., cannot teach multiple different courses simultaneously). Each instructor belongs to a specific department.
4. The system should manage students’ enrollment records (Enrollment), including enrollment status (enrolling, dropped, passed, failed, etc.), final grade (numeric / letter), and the grade points (GPA) earned for that course.
5. Course prerequisite requirements should be stored in a separate table (CoursePrereq), recording the course code, the required prerequisite course code(s), and the minimum passing grade required.
6. The system should support queries for all courses a student has taken (including course ID, name, credits, final grade, and grade points) to facilitate grade lookup and prerequisite verification, and must indicate which completed courses were failed and therefore require retaking.
7. Students may register for courses. The system must determine whether a student can enroll in a selected course based on the course’s time slot(s), the student’s current total planned credits, and the status of required prerequisite courses. Examples:

* If the time slot of the selected course conflicts with the time of an already selected course, enrollment fails.
* If the student’s planned course load for the term is less than 10 credits or more than 40 credits, timetable creation and export should fail.
* If a selected course has prerequisites that the student has not taken or for which the student’s prerequisite grade is insufficient, enrollment fails.
* Courses that a student has already taken and passed may still be submitted for a retake request, but retake requests require approval from the academic administrator.

1. Test data should be generated to exercise the system and cover various edge cases (e.g., student course time conflicts, instructor time conflicts, unmet prerequisite requirements, etc.). You may refer to the curriculum/degree plan of the School of Computer Science and Engineering as a basis for generating sample data.

**Functions:**

* 1. Students can register a unique account and query their personal course enrollments and completed course grades; they can update contact information and modify selected courses, but they cannot alter grade records.
  2. Instructors can register accounts subject to administrator approval. Instructors can update their personal information and enter or maintain final grades for the courses they teach, but they cannot change their affiliated department.
  3. Academic administrators (registrar/academic affairs staff) can maintain and update system base data (departments, courses, semesters, instructor affiliations, student enrollment status, etc.) and approve account registrations. They are responsible for setting course capacities and prerequisites; the system should perform automatic checks/enforcement based on those settings.
  4. The system should automatically compute and display students’ final grades and GPA, and provide grade-statistics features (e.g., course pass rates, GPA distribution).
  5. Other routine features: curriculum/degree-plan lookup (required credits and registration periods for general required courses, major required courses, major electives, university electives, etc.), transcript export, prerequisite validation, querying a student’s completed courses list, and so on.
  6. Students may extend the system with additional features to develop a more complete course registration and grade-management system. These extensions are optional but may earn extra credit.

**Related tools:**

* Database: SQL Server, Oracle, SQLite etc.
* Programming language: python, java, h5, php etc.

**Submitted materials:**

* Program manual and Source Code.
* Database design document (include E-R diagram and specifications), SQL commands for create table, query, update, etc.

**Submitted time & address:**

* Firm deadline: **Dec. 6th, 2025**
* submission:   
  1) International students: [](through Monitor);   
  2) Chinese students: submitted to teaching online system (教学在线).

Note: freely form a group with at most 4 people. But the submitted materials should include both the personal and group work (personal report specify the work that you have done in the group, better indicating the proportion).