CSC 540 Project 1

Dustin Lambright, Darshan Bhandari, Guanxu Yu, Leonard Kerr, Yuchen Sun

ER Diagram

Before describing the details of each table, I would like to go over the notation choices and explanations for our diagram. The diagram can be found on the last page (Figure 1**Error! Reference source not found.**).

We decided to use crows foot notation and include our attributes in line for each table. We believe this increases readability and allows for a clearer high level view. However, due to notation changes from class, I've assembled a table to describe the meaning of each symbol.

Symbol	Meaning
++	This table has at most one of this relationship
⊢<	This table has many of this relationship
Solid Line	There is at least one of this relationship (identifying)
Dotted Line	There can be 0 of this relationship (non-identifying)
🔑🔑	Key attribute (Red are foreign)
◇◆◇	Non-key attributes (Blue are required, red are foreign, white are not required)

User

Description: The base user table. Contains login information, first name, and last name.
Keys: **id**
Constraints: All fields must be non-null

Graduate

Description: Users that are graduate students.
Keys: **grad_id** (FK on User.id)
Constraints: None

Instructor

Description: Users that are instructors.
Keys: **inst_id** (FK on User.id)
Constraints: None

Student

Description: Users that are students (graduate and undergraduate).
Keys: **student_id** (FK on User.id)
Constraints: None

TAFor

Description:	A **relationship** table, relates graduate students to courses that they TA for.
Keys:	**ta_id** (FK on Graduate.grad_id), **course_id** (FK on Course.course_id)
Constraints:	None

EnrolledIn

Description:	A **relationship** table, relates students to courses that they are enrolled in.
Keys:	**student_id** (FK on Student.student_id), **course_id** (FK on Course.course_id)
Constraints:	On courses where graduate is true, student_id must exist in graduate table.

Course

Description:	Table containing basic information for a course.
Keys:	**course_id**
Constraints:	Start date must be before end date
	Number enrolled must be less than max enrolled
	Course id must match regex '[A-Z]{3}[0-9]{3}"
	All fields must be non-null except num-enrolled and graduate

Exercise

Description:	Table containing basic information for an exercise, **weak entity**.
Keys:	**course_id** (FK on Course.course_id), **ex_id**
Constraints:	Minimum difficulty must be below maximum difficulty
	Start date must be before end date
	Number of attempts must be greater than 1
	Scoring policy must be one of ('last', 'average', 'highest')
	All fields must be non-null
	Both difficulties must be between 1-5

ExQuestions

Description:	A **relationship** table, relates exercises to the questions in them.
Keys:	**ex_id** (FK on Exercise.ex_id), **ques_id** (FK on Question.ques_id)
Constraints:	None

Question

Description:	A table to store questions, contains the basic information.
Keys:	**ques_id**
Constraints:	All fields except hint must be non-null.
	Difficulty must be between 1-5

Answer

Description:	An answer table for a question, can be right or wrong and optionally contain parameters, **weak entity.**
Keys:	**ques_id** (FK on Question.ques_id), **ans_id**
Constraints:	All fields must be non-null except param_id (if there are no parameters)

Parameters

Description:	A table for storing parameters for both questions and answers, **weak entity**.
Keys:	**ques_id** (FK on Question.ques_id), **param_id**

Constraints: id must be unique, all fields must be non-null

AttAnswers

Description: A **relationship** table which stores the answers for each question on an exercise attempt.
Keys: **att_id** (FK on Attempt.att_id), **ans_id, ques_id** (Both FK on Answer.ans_id and Answer.ques_id)
Constraints: None

Attempt

Description: A table to store basic information about an attempt, **weak entity**. Cascades on deletion of exercise, course, or student.
Keys: **att_id**
Constraints: All fields must be non-null

Topic

Description: Table to store topics.
Keys: **topic_id**
Constraints: All fields must be non-null

CourseTopic

Description: A **relationship** table which stores the topics for each course.
Keys: id
Constraints: None

Constraints

- **Number of students enrolled in a course should not surpass max enrolled in Course table.**
  Trigger(s): EnrollStudent, CheckEnrolled, DropStudent
  This constraint was met by creating 3 triggers, two which update Course.num_enrolled when a student is added/dropped from a course, and then a last trigger which ensures num_enrolled never exceeds max_enrolled.

- **Students should only be able to attempt exercises for the courses they're in.**
  Trigger(s): AttExercise
  This constraint we implemented as a trigger which queries the enrolledin table before making an insertion on the attempts table.

- **Only graduate students should be able to enroll in graduate courses.**
  Trigger(s): GradEnroll
  This constraint was implemented as a trigger on insert to enrolledin which checks if the course is graduate, then queries the graduate table if it is.

- **TAs should not be able to enroll in classes that they are assisting.**
  Trigger(s): GradEnroll
  This constraint was implemented as a trigger on insert into the enrolledin which checks if there is an entry in the TAfor table with the same user id and course id.

- **Number of attempts should be limited by num_attempts in exercise table.**
  Trigger(s): NewAttempt
  This was achieved through a trigger which keeps track of current attempt number, checking against num_attempts in the exercise table.

Constraints Implemented in the Application

- **Students should only be able to attempt exercises that are currently open.**
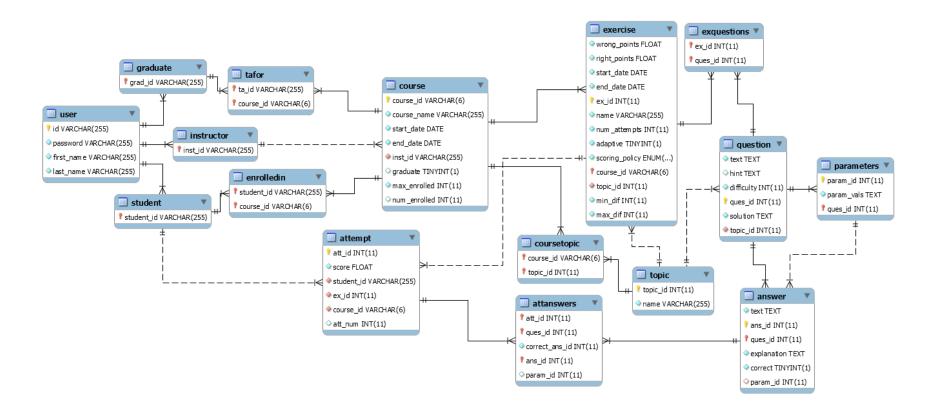 This constraint was implemented on the application side, because if we used CURDATE() to check attempts as we made them, then we couldn't load the sample data.

Figure 1. Our EER Diagram