

CSC 174 Fall 2020
Homework Assignment 3
(Total 200 points)

Please do NOT wait until the last minute to do this assignment. It may be much more time consuming than you thought.

This homework allows students to practice the development of a database using MySQL, as well as the design of views, store procedures, and stored functions. You can install MySQL in your local machine if you prefer to do so and know how to do so. Otherwise, please use the Athena account I gave to you. Before your submission, please test your code in your Athena account. You may want to pay attention to case sensitivity. I will run your commands in an Athena account for grading.

I do NOT debug for students. Solving assignment problems independently is one assessment criteria.

Section 1

Using SQL, create tables according to the given schema in Figure 1. ***You must create your database using exactly the same names for tables and attributes.*** The EER is shown in Figure 2.

Views related to the specialization are defined as follows. (please re-type the view definition instead of copy/paste to avoid run time problems)

```
CREATE VIEW TAView As  
Select S.SSN, S.StudentName, S.Address, S.Email, T.Salary  
From Student as S, TA as T  
Where S.SSN=T.SSN;
```

```
CREATE VIEW OnlineCourseView As  
Select C.CourseNo, C.CourseName, C.InstructorID, C.NoOfStudents, C.TASSN, W.URL  
From Course as C, OnlineCourse as W  
Where C.CourseNo = W.CourseNo;
```

```
CREATE VIEW TraditionalCourseView As  
Select C.CourseNo, C.CourseName, C.InstructorID, C.NoOfStudents, C.TASSN, T.ClassTime,  
T.RoomNo, T.Building  
From Course as C, TraditionalCourse as T  
Where C.CourseNo = T.CourseNo;
```

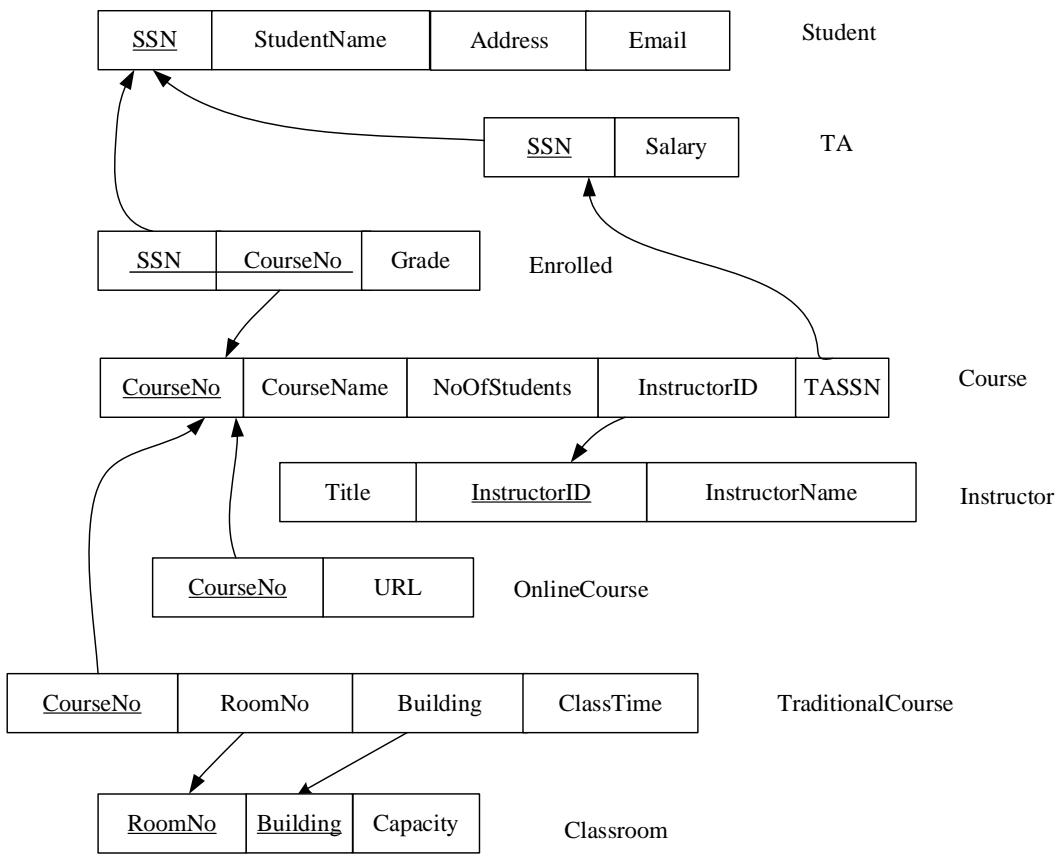


Figure 1. Relational Schema

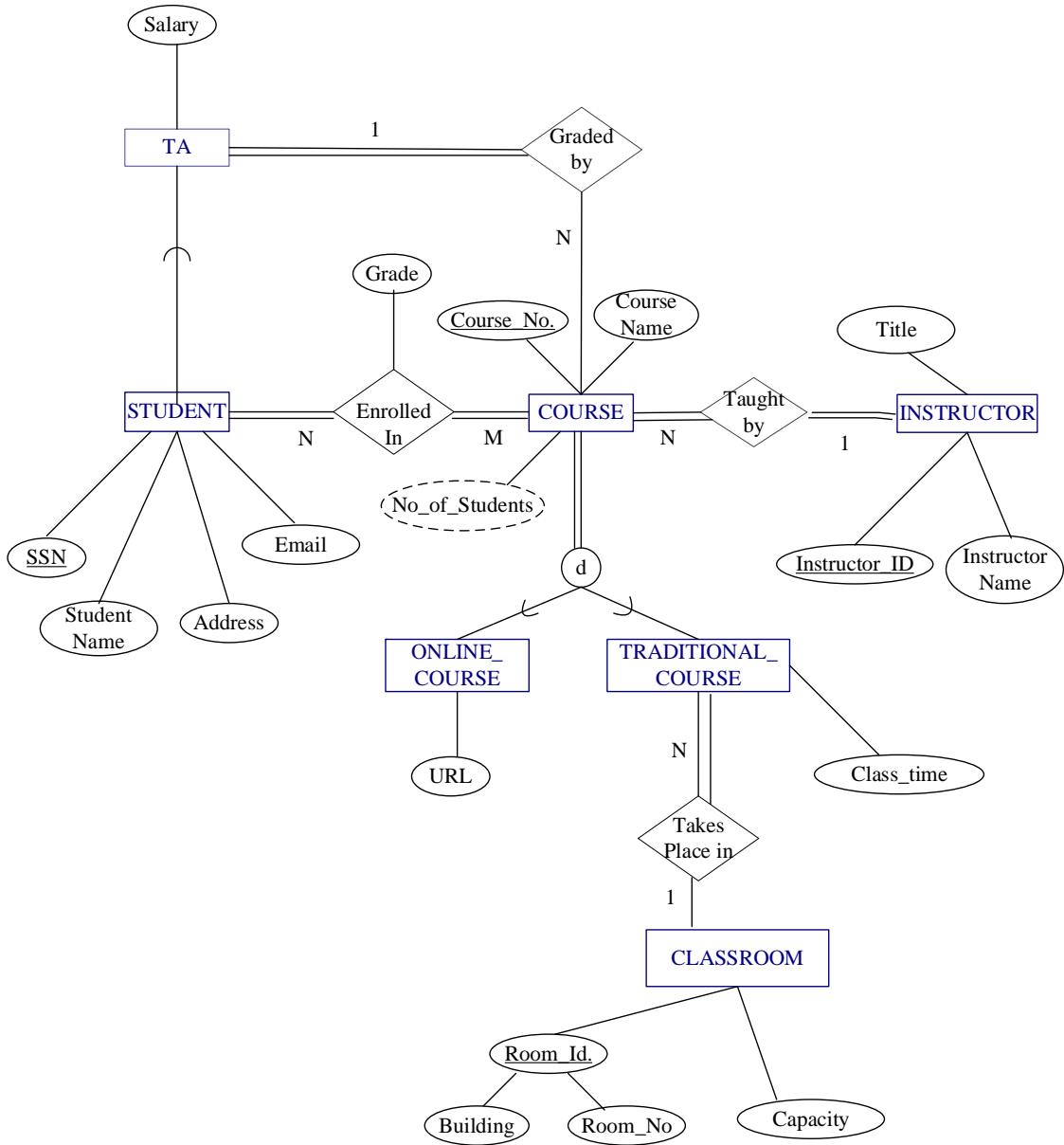


Figure 2. EER

Section 2

Populate the database. Please check sections 3, 4, and 5 for details.

Section 3

Create the following **views**. Display the results of selecting all tuples from the views.

When you populate the database, insert data such that at least one tuple will be displayed as the result of querying the views (select all tuples from the views).

- 1) A view constructed by a list of students' names and email addresses who are teaching assistants. Also get the course names for which they are teaching assistants.

Name of the view: TA_Course

Attributes of the view: TA name, TA email, Course name

- 2) A view constructed by a list of students who got at least 2 "A" in their course.

Name of view: Student_Grade_A

Attributes of the view: student ssn, number of A's gotten

Section 4

Create a **function** to implement the following requirements:

Retrieve the instructor's name who teaches a given course.

Function name: Course_Instructor

Input parameter: course name

Return: Instructor name

When you populate the database, insert data such that the function can return an instructor name.

Section 5

Design store procedures to implement the following requirements:

- 1) A stored procedure to output the names of the TAs for a given instructor. The instructor can teach multiple courses, so retrieve the TAs for all his/her courses.

Name of the procedure: Get_TA

Input parameter: instructor ID.

Output: print TAs' names

- 3) A stored procedure to output all the students enrolled in a given course. (use select statement, as shown in the examples of MySQL procedure slides)

Name of the procedure: GetStudentCourse

Input parameter: Course Number

Output: Print SSN, Student_Name, Address and Email_ID (use select statement, as shown in the examples of MySQL procedure slides)

When you populate the database, insert data such that at least one result will be display as the result of calling your store procedure.

Section 6

Create triggers to maintain the consistency of the derived attribute NoOfStudents in the course table.

- 1) When add a new record in the enrollment table, update the corresponding value of NoOfStudents enrolled in the course.

Trigger Name: Inc_enrollment_number

- 2) When delete a record in the enrollment table, update the corresponding value of NoOfStudents enrolled in the course.

Trigger Name: Del_enrollment_number

Section 7

Specify the statements to drop all the tables, views, functions, triggers, and procedures.

Pay attention to the order of the drop statements in order to drop everything successfully.

Submission

1) What you need to do:

- Section 1: Create table statements as well as use the given views (TAView, OnlineCourseView, TraditionalCourseView).
- Section 2: Insert statements to populate database
- Section 3: View definitions. Print the results of selecting all tuples from the views.
- Section 4: definition of the function. Print how to call the function to generate a result.
- Section 5: procedure definition. Print how to call the procedures.
- Section 6: trigger definition. Show a scenario that each trigger can be triggered and the changes of the “NoOfStudents” value because of the trigger.
- Section 7: Statements to drop tables, views, functions, and procedures.

2) What you need to submit:

Submit the following files to Canvas. Please do NOT zip them.

You must execute the statements, procedures, functions, and triggers before your submission. 0 point will be given to each non-executable.

(a) Create table statements, as well as all the views defined by the given requirements (TAView, OnlineCourseView, TraditionalCourseView)

(1_create_table.sql)

I will copy everything from this file and execute it. Make sure the tables are listed in the correct order such that I can execute it without any error. For any table that cannot be created, you will lose points, even if the error caused by the incorrect order of creating tables. All the subsequent tasks, such as procedures, cannot be executed without tables. As a result, you will get 0 for any other task because I cannot run it.

(b) Insert statements to populate database (*2_populate_db.txt or .sql*)

(c) Create view statements (TA_course, Student_Grade_A)

(3_create_view.txt or .sql)

(d) Definition of the function. How to call the function to generate results

(4_func.txt or .sql)

(e) Definition of the procedures. How to call the procedures to generate results.

(5_proc.txt or .sql)

(f) Definition of the triggers. (*6_a_trigger.sql or .sql*)

For each trigger, show a scenario that the trigger can be triggered and the changes of the “NoOfStudents” value because of the trigger.

(6_b_trigger.pdf)

(g) Statements to drop all tables, views, functions, triggers and procedures

(7_drop_all.txt or .sql)