Solution Homework 2

1. (70 points) Transform the instructor’s ER diagram for the Student Registration System (see the Homework 1 folder in MyCourses) to relations using the approach discussed in Chapter 4 of the Lecture Notes. When transforming the IS_A hierarchy, use Method 1 (i.e., sub entity set only explicitly inherits the key from the sub entity set). For composite attributes, use Method 1 (i.e., use the more specific attributes only) to perform the transformation and rename the component attributes to avoid ambiguity as needed. For each relation obtained, underscore the primary key, specify (using words) other candidate keys (if any) and foreign keys (if any), and specify the constraints associated with the relation (including all constraints that are described in the Requirements Document).

Answer: The ER Diagram is transformed into the following relations. The constraints are given below each relation.

Departments(deptname, office, phone#)
Other candidate keys: office, phone#

Students(B#, first_name, last_name, status, gpa, email, bdate, deptname)
Other candidate key: email
Constraints: valid values for status: {freshman, sophomore, junior, senior, MS, PhD}
0 <= gpa <= 4.0
deptname is a foreign key referencing deptname of Departments

TAs(B#, level, pay_rate, office, office_hour_start_time, office_hour_end_time)
Constraint: office_hour_start_time < office_hour_end_time
The pay rate of a TA depends on his/her level
B# is a foreign key referencing B# of Students

TA_Office_Hour_Days(TA_B#, day)
valid values for day: {Monday, Tuesday, Wednesday, Thursday, Friday}.
TA_B# is a foreign key referencing B# of TAs

Courses(dept_code, course#, title, credits, deptname)
Constraints: 100 <= course# <= 799
valid values for credits: {3, 4}
if course# < 500, then credits is 4 and if course# >= 500, then credits is 3
deptname is a foreign key referencing deptname of Departments

Prerequisites(dept_code, course#, pre_dept_code, pre_course#)
Both (dept_code, course#) and (pre_dept_code, pre_course#) are foreign keys referencing (dept_code, course#) of Courses.
Courses and prerequisite courses do not form a cycle.

Classes(dept_code, course#, sect#, year, semester, start_time, end_time, limit, size, room, TA_B#)

Constraints: size <= limit

Valid values for semester {Spring, Fall, Summer 1, Summer 2}

start_time < end_time

size >= 5 for graduate classes and size > 10 for undergraduate classes

Each TA assists one class in a semester

(dept_code, course#) is foreign key referencing (dept_code, course#) of Courses

TA_B# is a foreign key referencing B# of TAs

Classes_days(dept_code, course#, sect#, year, semester, day)

valid values for day: {Monday, Tuesday, Wednesday, Thursday, Friday}.

(dept_code, course#, sect#, year, semester) is a foreign key referencing (dept_code, course#, sect#, year semester) of Classes

Faculty(B#, first_name, last_name, rank, office, email, phone#, deptname)

Other candidate keys: email

Constraints: valid values for rank: {lecturer, assistant professor, associate professor, professor}

deptname is a foreign key referencing deptname of Departments

Teaches(Instructor_B#, dept_code, course#, sect#, year, semester)

Constraints: A class may be taught by up to two faculty members

No faculty can teach classes with overlapping class times

Each faculty can teach up to three classes in a semester.

Instructor_B# is a foreign key referencing the B# of Faculty

(dept_code, course#, sect#, year, semester) is a foreign key referencing (dept_code, course#, sect#, year, semester) of Classes

Enrollment(Student_B#, dept_code, course#, sect#, year, semester, lgrade, ngrade)

Constraints: valid values for lgrade: {A, A-, B+, B, B-, C+, C, C-, D, F, I, null}.

valid values for ngrade: {0, 1, 1.7, 2, 2.3, 2.7, 3, 3.3, 3.7, 4, null}

lgrade and ngrade must match: A ←→ 4, A- ←→ 3.7, B+ ←→3.3, B ←→ 3, B- ←→ 2.7, C+ ←→ 2.3, C ←→ 2, C- ←→ 1.7, D ←→ 1, F ←→ 0 and I ←→ null.

No student can take the same course more than once

A student must have completed all prerequisite courses with a grade of at least C in order to enroll into a class

Students cannot enroll into classes with overlapping class times

Student_B# is a foreign key referencing B# of Students

(dept_code, course#, sect#, year, semester) is a foreign key referencing (dept_code, course#, sect#, year, semester) of Classes

Each student can only enroll into between 1 and 5 classes in a given semester

Each class must have at least 5 students

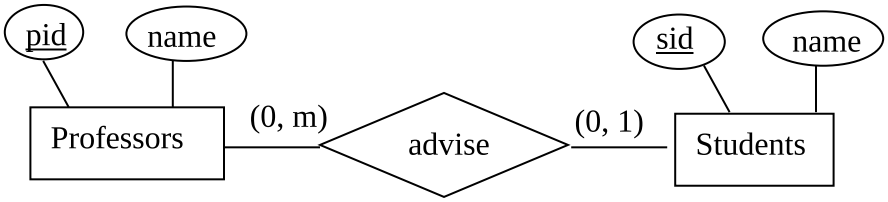
2. (10 points) Consider a relation R with 3 attributes A, B and C. What is the maximum number of possible keys R could have? What they may look like (list all possibilities)? Justify your answer.

Answer: The maximum number is 3. They could be either {A, B, C} or {AB, AC, BC}. In the former case, each of A, B and C is a key. In this case, there cannot be other keys (any other possible key would contain one of the three attributes, making it a superkey, but not a key). In the latter case, none of A, B and C is a key, but each combination of two attributes forms a key. In this case, the combination of ABC cannot be a key. If there is only one single-attribute key, say A, then we would have at most 2 keys (the other possible one would be BC).

3. (10 points) Let A be the primary key of relation R1 and B be the primary key of R2. If A is declared as a foreign key referencing B and B is also declared a foreign key referencing A. What problem this mutual foreign key references can cause when using R1 and R2? Explain.

Answer: With foreign keys declared in this way, no tuple can be inserted into R1 or R2, making them useless. To see this, let's try to insert a tuple t1 into R1. Suppose initially both R1 and R2 are empty. Based on the entity integrity rule, since A is the primary key of R1, t1[A] cannot be null. But based on the referential integrity rule, t1[A] must match a value under B of R2, which is not possible since R2 is empty at this time. Therefore, t1 cannot possibly be inserted into R1. Using similar argument, we can show that no tuple can be inserted into R2.

4. (10 points) Consider the following ER Diagram:



If the relationship is transformed into a foreign key attribute in relation Students, can null value be avoided in this attribute? Explain. Propose a different way to transform this relationship such that null values can be avoided.

Answer: No, null value cannot be avoided if the relationship is transformed into a foreign key in Students. Let pid be added to Students as a foreign key. Connectivity (0, 1) means that students are not required to have an advisor. Suppose s1 is such a student without an advisor. Then for this student, there is no corresponding pid. As a result, for this student, its pid value will be null. A simple way to avoid null value is to transform the relationship into a separate relation with two attributes (sid, pid). In this case, only when a student has an advisor, the sid of the student and the pid of his/her advisor will appear in this relation, leading to a relation with no null values.