CSC 443 **Fall 2011** **Dr. R. M. Siegfried**

Answers to Assignment #1

1.14. *Consider the Database below:*

If the name of the 'CS' (Computer Science) Department changes to 'CSSE' (Computer Science and Software Engineering) Department and the corresponding prefix for the course number also changes, identify the columns in the database that would need to be updated.
Can you restructure the columns in the COURSE, SECTION, and PREREQUISITE tables so that only one column will need to be updated.
The Database

Student

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

Course

Course_name	Course_number	Credit_hours	Department
Introduction to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Structures	MATH2410	3	MATH
Database	CS3380	3	CS

Section

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

Grade_Report

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

Prerequisite

Course_number	Prerequisite_number
CS3380	CS3220
CS3380	MATH2410
CS3380	CS1310

a. The following columns will need to be updated.

Table	Column(s)
STUDENT	Major
COURSE	CourseNumber and Department
SECTION	CourseNumber
PREREQUISITE	CourseNumber and PrerequisiteNumber

b. You should split the following columns into two columns:

Table	Column	Split Columns
COURSE	CourseNumber	CourseDept and CourseNum
SECTION	CourseNumber	CourseDept and CourseNum
PREREQUISITE	CourseNumber	CourseDept and CourseNum
PREREQUISITE	PrerequisiteNumber	PreReqDept and PreReqNum

Note that in the COURSE table, the column CourseDept will not be needed after the above change, since it is redundant with the Department column.

CSC 443 **Fall 2011** **Dr. R. M. Siegfried**

Answers to Assignment #2

2.12 Think of different users for the database shown in Figure 1.2 (it was show in detail in Assignment #1. What types of applications would each user need? To which user category would each belong?

(a) Registration Office User: They can enter data that reflect the registration of students in sections of courses, and later enter the grades of the students. Applications can include:

- Register a student in a section of a course
- Check whether a student who is registered in a course has the appropriate prerequisite
- courses
- Drop a student from a section of a course
- Add a student to a section of a course
- Enter the student grades for a section

Application programmers can write a number of canned transactions for the registration office end-users, providing them with either forms and menus, or with a parametric interface.

(b) Admissions Office User: The main application is to enter newly accepted students into the database. Can use the same type of interfaces as (a).

(c) Transcripts Office User: The main application is to print student transcripts. Application programmers can write a canned transaction using a report generator utility to print the transcript of a student in a prescribed format. The particular student can be identified by name or social security number. Another application would be to generate grade slips at the end of each semester for all students who have completed courses during that semester. Again, this application could be programmed using a report generator utility.

These are the answers in the instructor's manual, but faculty and students would also need access (as in CLASS). For that reasons, the answers the class provided would also be correct.

CSC 443 **Fall 2011** **Dr. R. M. Siegfried**

Answers to Assignment #3

3.15. *Consider the following relations for a database that keeps track of business trips of salespersons in a sales office:*

SALESPERSON(Ssn, Name, Start_year, Dept_no)
TRIP(Ssn, From_city, To_city, Departure_date, Return_date, Trip_id)
EXPENSE(Trip_id, Account#, Amount)
A trip can be charged to one or more accounts. Specify the foreign keys for this schema, stating any assumption that you make.

The schema of this question has the following two foreign keys:

1. The attribute SSN of relation TRIP that references relation SALESPERSON, and
2. The attribute Trip_ID of relation EXPENSE that references relation TRIP.

In addition, the attributes Dept_No of relation SALESPERSON and Account# of relation EXPENSE are probably also foreign keys referencing other relations of the database not mentioned in the question.

3.16 *Consider the following relations for a database that keeps track of student enrollment in courses and the books adopted for each course:*

STUDENT(Ssn, Name, Major, Bdate)
COURSE(Course#, Cname, Dept)
ENROLL(Ssn, Course#, Quarter, Grade)
BOOK_ADOPTION(Course#, Quarter, Book_isbn)
Specify the foreign keys for this schema, stating any assumptions that you make.

The schema of this question has the following four foreign keys:

1. The attribute SSN of relation ENROLL that references relation STUDENT,
2. The attribute Course# in relation ENROLL that references relation COURSE,
3. The attribute Course# in relation BOOK_ADOPTION that references relation COURSE, and
4. The attribute Book_ISBN of relation BOOK_ADOPTION that references relation TEXT.

3.17 *Consider the following relations for a database that keeps track of automobile sales in a car dealership (OPTION refers to some optional equipment installed on an automobile):*

CAR(Serial_no, Model, Manufacturer, Price)
OPTION(Serial_no, Option_name, Price)
SALE(Salesperson_id, Serial_no, Date, Sales_price)
SALESPERSON(Salesperson_id, Name, Phone)
*First, specify the foreign keys for this schema, stating any assumptions that you make. Next, populate the relations with a few sample tuples, and then given an example of an insertion in the **SALE** and **SALEPERSON** relations that **violate** the referential integrity constraints and of another insertion that does not.*

The schema of this question has the following three foreign keys:

1. The attribute Serial_no of relation OPTION that references relation CAR
2. The attribute Salesperson_id of relation SALE that references relation SALESPERSON
3. The attribute Serial_no of relation SALE that references relation CAR

CSC 443 **Fall 2011** **Dr. R. M. Siegfried**

Answers to Assignment #5

4.5 *Consider the database shown in Figure (1.2, whose schema in shown in Figure 2.1 What are the referential integrity constraint that should on the schema? Write the appropriate SQL DDL statement to define the database.*

STUDENT

Name	Student_number	Class	Major
Smith	15	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Structures	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH 2410	Fall	07	King
92	CS 1310	Fall	07	Anderson
102	CS 3320	Spring	08	Knuth
112	MATH 2410	Fall	08	Chang
119	CS 1310	Fall	08	Anderson
135	CS 3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Schemas

STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

The following referential integrity constraints should hold (we use the notation:
R.(A1, ..., An) --> S.(B1, ..., Bn)
to represent a foreign key from the attributes A1, ..., An of R (the referencing relation) to S (the referenced relation)):

PREREQUISITE.(CourseNumber) → COURSE.(CourseNumber)

PREREQUISITE.(PrerequisiteNumber) → COURSE.(CourseNumber)

SECTION.(CourseNumber) → COURSE.(CourseNumber)

GRADE_REPORT.(StudentNumber) → STUDENT.(StudentNumber)

GRADE_REPORT.(SectionIdentifier) → SECTION.(SectionIdentifier)

```
CREATE TABLE STUDENT
(Name               VARCHAR(30)         NOT NULL,
StudentNumber       INTEGER             NOT NULL,
Class               CHAR                NOT NULL,
Major               CHAR(4),
PRIMARY KEY (StudentNumber));


CREATE TABLE COURSE
(CourseName         VARCHAR(30)         NOT NULL,
CourseNumber        CHAR(8)             NOT NULL,
CreditHours         INTEGER,
Department          CHAR(4),
PRIMARY KEY (CourseNumber),
UNIQUE (CourseName));

CREATE TABLE PREREQUISITE
(CourseNumber       CHAR(8)             NOT NULL,
PrerequisiteNumber CHAR(8)              NOT NULL,
PRIMARY KEY (CourseNumber, PrerequisiteNumber),
FOREIGN KEY (CourseNumber) REFERENCES COURSE (CourseNumber),
FOREIGN KEY (PrerequisiteNumber) REFERENCES COURSE (CourseNumber));

CREATE TABLE SECTION
(SectionIdentifier        INTEGER       NOT NULL,
CourseNumber              CHAR(8)       NOT NULL,
Semester                  VARCHAR(6)    NOT NULL,
Year                      CHAR(4)       NOT NULL,
Instructor                VARCHAR(15),
PRIMARY KEY (SectionIdentifier),
FOREIGN KEY (CourseNumber) REFERENCES COURSE (CourseNumber));

CREATE TABLE GRADE_REPORT
(StudentNumber      INTEGER             NOT NULL,
SectionIdentifier   INTEGER             NOT NULL,
```

```
Grade              CHAR,
PRIMARY KEY (StudentNumber, SectionIdentifier),
FOREIGN KEY (StudentNumber) REFERENCES STUDENT (StudentNumber),
FOREIGN KEY (SectionIdentifier) REFERENCES SECTION (SectionIdentifier));
```