Homework Assignment 2 for CSCI-GA-2433 Database System

Yue Zhou

June 17, 2021

Abstract

The exercises are from textbook Fundamental of database systems, seventh edition.

1 Question1, exercise 1.9

What is the difference between controlled and uncontrolled redundancy? Illustrate with examples.

1.1 Redundancy

Redundancy is defined as "duplicate some or all of the same data for multiple times.[1]"

1.2 Controlled Redundancy

Redundancy is defined as controlled redundancy when the duplicated data is consistent, such as no information mismatching.

St_Number	St_Name	St_ID	Course_Num	Grade
17	Smith	112	Math2410	B
17	Smith	119	CS1310	C
8	Brown	85	Math2410	A
8	Brown	92	CS1310	A

St_Number	St_Name	St_ID	Course_Num	Grade	Course_Num	Grade
17	Smith	112	Math2410	B	CS1310	C
8	Brown	85	Math2410	A	CS1310	A

In the left graph, there exists two controlled redundancies, where the student Number 17 and student name Smith have appeared twice, as well as the student number 8 and student name Brown.
The advantage of controlled redundancy is that it is easy to retrieve the data within the same specific format. In this case, it is the format of "student number, student name, student ID, course number, and grade".
The graph above on the right hand side is the database with no redundancy.

1.3 Uncontrolled Redundancy

Redundancy is defined as uncontrolled redundancy when the data is inconsistent, hence one or multiple copies of the data are mismatched or causing flaws.

St_Number	St_Name	St_ID	Course_Num	Grade
17	Brown	112	Math2410	B

In this particular example, the inconsistent part is student name, which should be Smith instead of Brown.

2 Question2, exercise 1.12

Cite some examples of integrity constraints that you think can apply to the database shown in Figure 1.2.
There are four types of integrity constraints: domain constraint, entity integrity constraint, referential integrity constraint, and key constraint.

2.1 Domain Constraint

The domain constraint is defined as the validity of the values within an attribute. In a database, one attribute has potential data types, such as string, character, float, integer. In one specific attribute, the values of data elements in the attribute should satisfy the domain constraint.

Name	St_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS
3	25	2	CS

For example, in the graph above, the data elements in name column(attribute) are expected to have string as their datatype; however, an integer appears on the third row. Hence, the domain constraint is not satisfied.
For another example, the Credit hours in the table COURSE should be an integer or float datatype, which is an element from $\{0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5....\}$.

2.2 Entity Integrity Constraint

The entity integrity constraint defines that the primary key value should be valid.

Name	St_number	Class	Major
Smith	17	1	CS
Brown		2	CS

For example, the student number is the primary key which identifies individual rows in relational database. The value of it cannot be null.

For another example, the Section identifier in SECTION table satisfies the entity integrity constraint, since each data element is not null.

2.3 Referential Integrity Constraint

Referential integrity constraint defines that the foreign keys in table 1 should be either all null values or all available values, whenever the foreign keys in table 1 work as primary keys in table 2.

COURSE

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

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

For example, the course number in course is the foreign key, which also works as the primary key in prerequisite. Hence, in this case, the referential integrity constraint is satisfied since all the data elements in course number attribute in course are available.

2.4 Key Constraint

Keys are the entity set that is used to identify an entity. Hence, the key constraint defines that the primary key value should either be null or unique.

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

In the graph above, student number attribute does not satisfy the key constraint since both 17 and 8 are repeated and not unique. The section identifier satisfies the key constraint since each data element in its column is unique.
Another example is that the Course number is unique in Course table, hence it satisfies the key constraint.

3 Question3, exercise 2.14

If you were designing a Web-based system to make airline reservations and sell airline tickets, which DBMS architecture would you choose from Section 2.5? Why? Why would the other architectures not be a good choice?

Basic client/server architectures would work. The idea is to define specialized servers with specific functionalities. For example it is possible to connect a number of PCs or workstations as clients to a file server. The resources provided by specialized server can be accessed by many client machines. Each client machine provides the interface to the user.When each client requires the access to the database, it connects to the server.[2] However, compared to Three-tier architecture, it lacks of credentials. Meanwhile, basic client/server architectures is going to cause burden either on client side or server side.

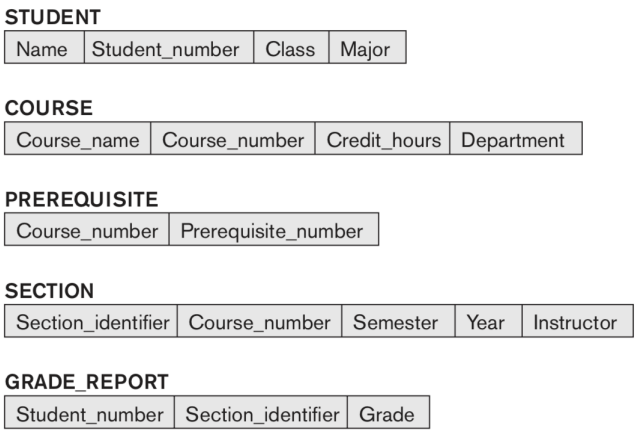
Three-tier and n-tier architecture improves the security based on basic client/server architecture. The intermediate layer in three-tier and n-tier architecture improves the security by checking client credentials.Meanwhile, the middle layer can also act as a Web server, and the database server stores the data. [2]

The centralized DBMSs architecture dose not work since it does not have a common shared database which is accessible to each client.
Similarly, the two-tier client/server architectures for DBMSs does not work since it only separates interface and application programs from the server to the client; moreover, it is not a distributive database system.

4 Question4, exercise 2.15

Consider Figure 2.1. In addition to constraints relating the values of columns in one table to columns in another table, there are also constraints that impose restrictions on values in a column or a combination of columns within a table. One such constraint dictates that a column or a group of columns must be unique across all rows in the table. For example, in the STUDENT table, the Student number column must be unique (to prevent two different students from having the same Student number). Identify the column or the group of columns in the other tables that must be unique across all rows in the table.

Figure 2.1
Schema diagram for the database in Figure 1.2.



The Student number in STUDENT table must be unique.
The Course number in COURSE table must be unique. The Course number is the primary key for course table.
*The combination of Course number and Prerequisite number in the PREREQUISITE table must be unique. Since a course may have multiple prerequisite courses; hence a Course number may appear multiple times in the Course number column. Similarly, a prerequisite number can appear multiple times for being the prerequisite for different courses. Thus, we need both Course number and Prerequisite number.
The Section identifier in SECTION table must be unique. No two sections can have the same section identifier.
The combination of Student number and Section identifier in GRADE REPORT table must be unique. Since student number can be repeated, and the section identifier can be repeated; hence we need both of them to fix the uniqueness.

5 Question4, exercise 5.16

5.16. 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)

TEXT(Book_isbn, Book_title, Publisher, Author)

Specify the foreign keys for this schema, stating any assumptions you make.

Assumption: the combination of Ssn,Course♯, and Quarter is the primary key of the table ENROLL.

The foreign key, the combination of Course♯ and Quarter in the relational table ENROLL, is also the primary key in BOOK ADOPTION table.
The foreign key, course♯ in the relation ENROLL, is also the primary key in COURSE table.
The foreign key, Ssn in relation ENROLL, is also the primary key in STUDENT table.
The foreign key, Course♯ in relation BOOK ADOPTION, is also the primary key in COURSE table.
The foreign key, Book isbn in relation BOOK ADOPTION, is also the primary key in TEXT table.

6 Question4, exercise 5.18

Database design often involves decisions about the storage of attributes. For example, a Social Security number can be stored as one attribute or split into three attributes (one for each of the three hyphen-delineated groups of numbers in a Social Security number—XXX-XX-XXXX). However, Social Security numbers are usually represented as just one attribute. The decision is based on how the database will be used. This exercise asks you to think about specific situations where dividing the SSN is useful.

There is one situation that storing the SSN as first five digits as an attribute and the last four digits as another attribute can be useful. The situation is that the bank system checks the credentials with their customers. The system usually asks the client to input the last four digits of their social security number when they log into the system. In this case, storing the SSN separately is designed for this functionality, which is searching for the last four digits of social security number in the database system. Hence, we can avoid the problem of partial string matching.

7 Question4, exercise 5.20

5.20. Recent changes in privacy laws have disallowed organizations from using Social Security numbers to identify individuals unless certain restrictions are satisfied. As a result, most U.S. universities cannot use SSNs as primary keys (except for financial data). In practice, Student_id, a unique identifier assigned to every student, is likely to be used as the primary key rather than SSN since Student_id can be used throughout the system.

a. Some database designers are reluctant to use generated keys (also known as *surrogate keys*) for primary keys (such as Student_id) because they are artificial. Can you propose any natural choices of keys that can be used to identify the student record in a UNIVERSITY database?

b. Suppose that you are able to guarantee uniqueness of a natural key that includes last name. Are you guaranteed that the last name will not change during the lifetime of the database? If last name can change, what solutions can you propose for creating a primary key that still includes last name but remains unique?

c. What are the advantages and disadvantages of using generated (surrogate) keys?

7.1 question a

Yes, in this case, the primary key(unique identifier) can be the combination of multiple attributes, such as the combination of first name, last name, and date of birth. In this case, we still have to store the student's first name, last name, and date of birth separately in different columns.

7.2 question b

This problem can be solved by making an additional column to keep the copy of the student's initial last name when the person's data is added to the database system at the beginning. In addition, we use this copied last name as the primary key. Even though the person's last name is changed later on, the copied last name remains unchanged and still works as the primary key.

7.3 question c

The advantage of using surrogate key is that it ensures the uniqueness of each entity in the database. Meanwhile, it keeps everyone's private information unrevealed.
The disadvantage is that the generated(surrogate) key takes extra storage space on disk and may cause extra searching steps(computations) when I want to find the student's information across multiple tables.

References

[1] Ramez Elmasri and Shamkant Navathe. “Fundamentals of Database Systems”. In: Pearson, 2015. Chap. 1.6.

[2] Ramez Elmasri and Shamkant Navathe. “Fundamentals of Database Systems”. In: Pearson, 2015. Chap. 2.