

CS 448 Spring 2024 Homework 2

Sean Rettig

February 6, 2024

1

Suppose that each of the following operations is applied directly to the database state shown in the Figure 5.6 and database schema in the Figure 5.7 in the Textbook (7th Edition). The figures are also shown in slides 5-31 and slides 5-37 respectively. Discuss all integrity constraints violated by each operation (including the tables affected), if any, and the different ways to enforce the constraints (without modifying the operation) with some relevant explanation(s). For example, the operation Delete the PROJECT tuple with Pname = 'ProductX' violates a referential integrity constraint with WORKS_ON. Possible ways to enforce the operation are: (i) reject the delete; (ii) delete cascade (delete all tuples in WORKS_ON that correspond to Pname = 'ProductX').

- (a) Delete the WORKS_ON tuples with Essn = '999887777'.
- (b) Modify the Super_ssn attribute of the EMPLOYEE tuple with Ssn = '999887777' to NULL.
- (c) Insert tuple <'555123456', Phil, 'M', 2000-12-01, 'Son'> into Dependent.
- (d) Delete the DEPT_LOCATIONS tuples where Dnumber = 3.

Answer:

- (a) This operation is acceptable. There are no integrity Constraints that are violated. Referencing Figure 5.6, it deletes two tuples in the WORKED_ON table.
- (b) This operation is acceptable. There are no integrity constraints that are violated. Referencing Figure 5.6, it modifies a single value of a tuple in the EMPLOYEE table to NULL.
- (c) This insertion violates the referential integrity constraint specified on Essn in DEPENDENT because no corresponding referenced tuple exists in EMPLOYEE with Ssn = 555123456. Possible ways to enforce the operation are: (i) reject the delete;
- (d) This deletion is acceptable. There are no integrity constraints that are violated. Referencing Figure 5.6, it deletes a single tuple in the DEPT_LOCATIONS table.

2

Database design often involves decisions about the storage of attributes. For example, the SSN can be subdivided in groups and stored in multiple attributes for security/validation purpose (validate the last four digits). Another example is the address, which can be stored as a single attribute or split in multiple attributes (City, State, ZIP, Street Address).

- (a) Discuss specific scenarios where it is useful to store the address as one attribute (i.e., a text field) or multiple attributes (i.e., separate fields for city, state, zip, number). Explain your answer.
- (b) Discuss at least 2 additional scenarios of information that can benefit of such decisions (split in multiple attributes)? You need to use other examples i.e. other than “Address” and “SSN”. Explain your answer.

Answer:

- (a) Storing an address as a single text field can make information insertion and storage much simpler, since all the user would need to insert is the string with their address, and all that would need to be stored is the string. However, storing multiple attributes like this could be helpful if one needs to gather information about the address. If you wanted to sort all users by city, or state, it would be hard if the address was just a text field instead of split into attributes by city, state, zip, number that you could easily query.
- (b) (1st scenario) A phone number could benefit from being split into multiple attributes. It could consist of the country code of origin, the area code, the telephone prefix, and line number. The first two could be useful information to use within queries.

(2nd scenario) A Date entry could benefit from being split into multiple attributes. This could allow for further granularity if it allowed an optional NULL valued seconds and nanoseconds, etc. Additionally, if, for example, you wanted to select all within the month of May, it would be easier to grab the month number instead of parsing a string of the date.