Answers to Review Questions

1. **What is normalization?**

   Normalization is the process for assigning attributes to entities. Properly executed, the normalization process eliminates uncontrolled data redundancies, thus eliminating the data anomalies and the data integrity problems that are produced by such redundancies.

   Normalization does not eliminate data redundancy; instead, it produces the carefully *controlled* redundancy that lets us properly link database tables.

2. **When is a table in 1NF?**

   A table is in 1NF when all the key attributes are defined (no repeating groups in the table) and when all remaining attributes are dependent on the primary key. However, a table in 1NF still may contain partial dependencies, i.e., dependencies based on only part of the primary key and/or transitive dependencies that are based on a non-key attribute.

3. **When is a table in 2NF?**

   A table is in 2NF when it is in 1NF and it includes no partial dependencies. However, a table in 2NF may still have transitive dependencies, i.e., dependencies based on attributes that are not part of the primary key.

4. **When is a table in 3NF?**

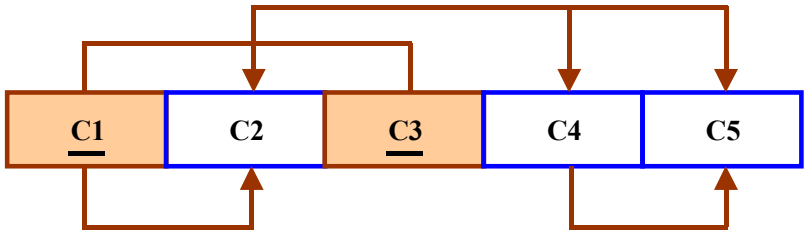
   A table is in 3NF when it is in 2NF and it contains no transitive dependencies.

5. **When is a table in BCNF?**

   A table is in Boyce-Codd Normal Form (BCNF) when it is in 3NF and every determinant in the table is a candidate key. For example, if the table is in 3NF *and it contains a nonprime attribute that determines a prime attribute*, the BCNF requirements are not met. (Reference the text's Figure 6.8 to support this discussion.)This description clearly yields the following conclusions:
   - If a table is in 3NF and it contains only one candidate key, 3NF and BCNF are equivalent.
   - BCNF can be violated only if the table contains more than one candidate key. Putting it another way, there is no way that the BCNF requirement can be violated if there is only one candidate key.

6. **Given the dependency diagram shown in Figure Q6.6, answer items 6a-6c:**

FIGURE Q5.6 Dependency Diagram for Question 6



a. Identify and discuss each of the indicated dependencies.

C1 → C2 represents a *partial dependency*, because C2 depends only on C1, rather than on the entire primary key composed of C1 and C3.

C4 → C5 represents a *transitive dependency*, because C5 depends on an attribute (C4) that is not part of a primary key.

C1, C3 → C2, C4, C5 represents a set of proper functional dependencies, because C2, C4, and C5 depend on the primary key composed of C1 and C3.

b. Create a database whose tables are at least in 2NF, showing the dependency diagrams for each table.

The normalization results are shown in Figure Q6.6b.

Figure Q6.6b The Dependency Diagram for Question 6b

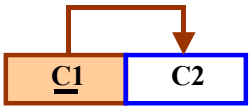


Table 1

Primary key: C1
Foreign key: None
Normal form: 3NF

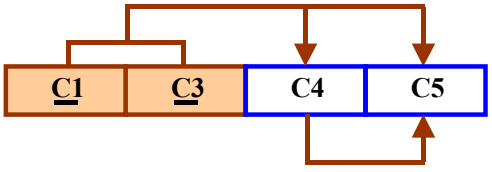


Table 2

Primary key: C1 + C3
Foreign key: C1 (to Table 1)
Normal form: 2NF, because the table exhibits the transitive dependencies C4 → C5

c. Create a database whose tables are at least in 3NF, showing the dependency diagrams for each table.

The normalization results are shown in Figure Q6.6c.

Figure Q6.6c The Dependency Diagram for Question 6c

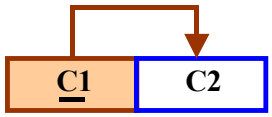


Table 1
Primary key: C1
Foreign key: None
Normal form: 3NF

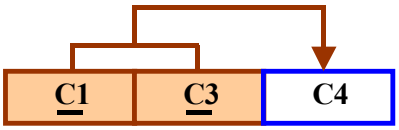


Table 2
Primary key: C1 + C3
Foreign key: C1 (to Table 1)
C4 (to Table 3)
Normal form: 3NF

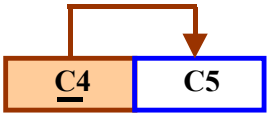
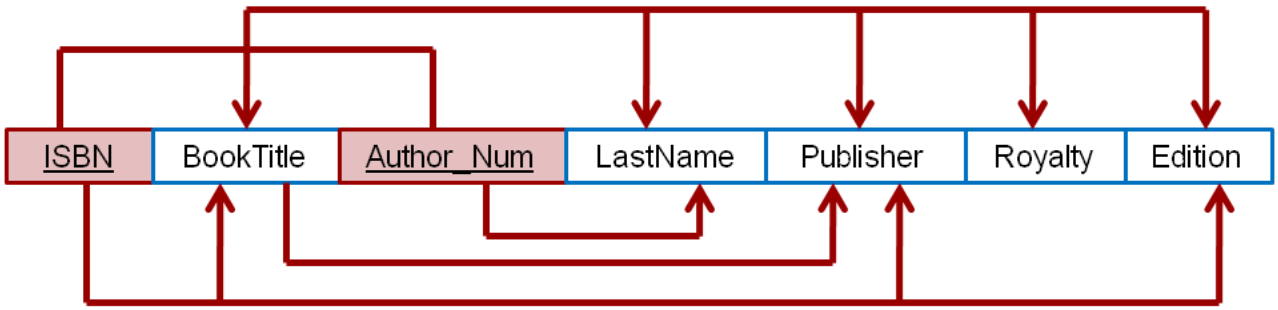


Table 3
Primary key: C4
Foreign key: None
Normal form: 3NF

7. The dependency diagram in Figure Q6.7 indicates that authors are paid royalties for each book that they write for a publisher. The amount of the royalty can vary by author, by book, and by edition of the book.

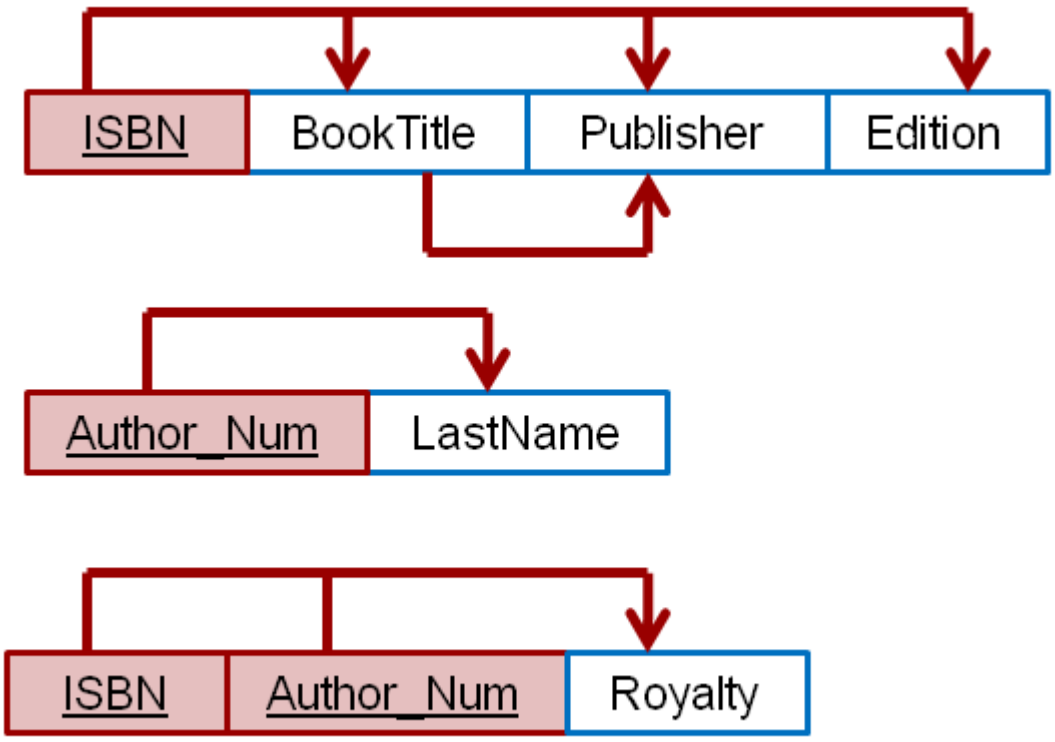
Figure Q6.7 Book royalty dependency diagram



a. Based on the dependency diagram, create a database whose tables are at least in 2NF, showing the dependency diagram for each table.

The normalization results are shown in Figure Q6.7a.

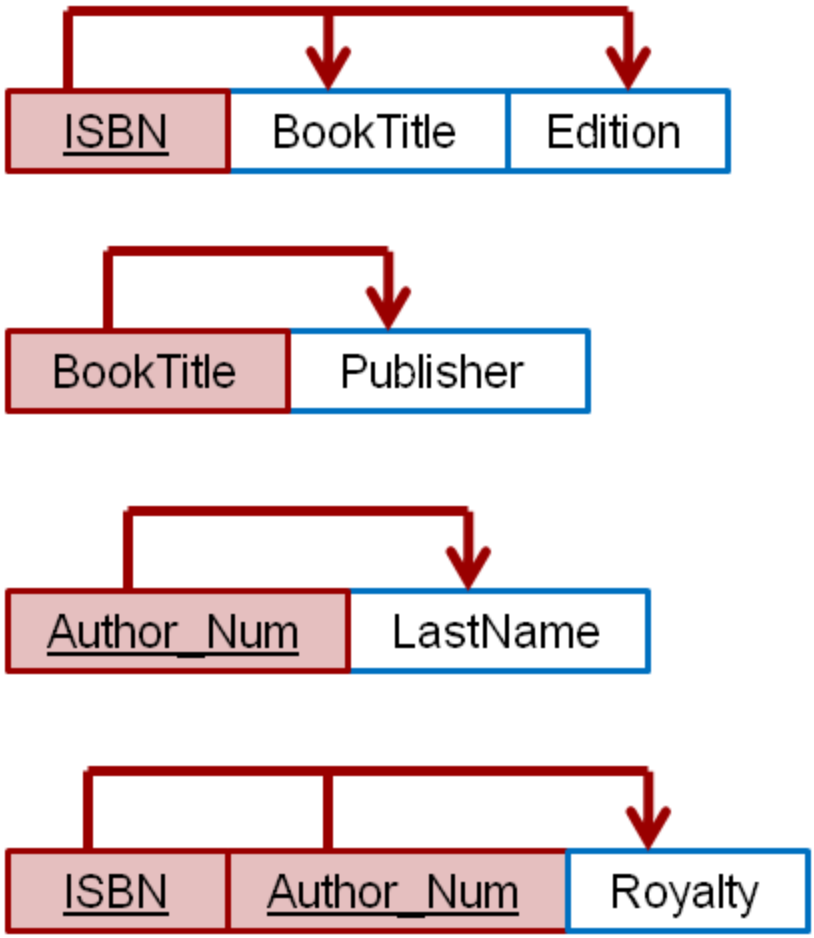
Figure Q6.7a The 2NF normalization results for Question 7a.



b. Create a database whose tables are at least in 3NF, showing the dependency diagram for each table.

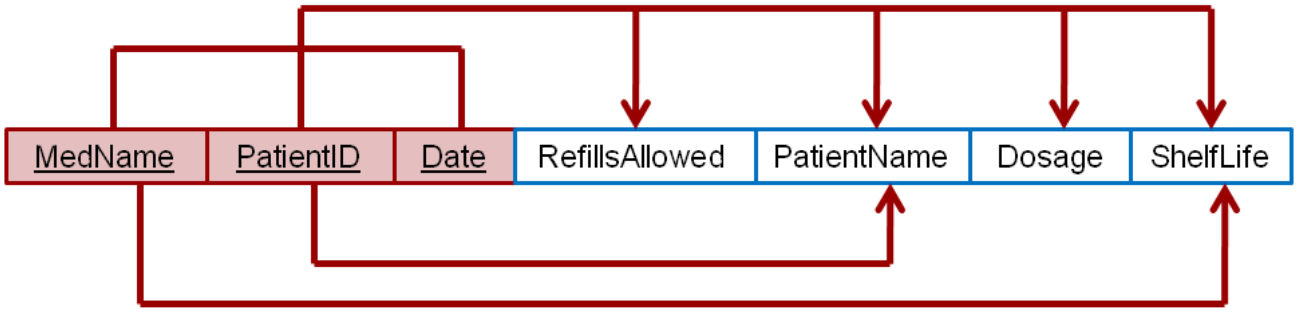
The normalization results are shown in Figure Q6.7a.

Figure Q6.7b The 3NF normalization results for Question 7b.



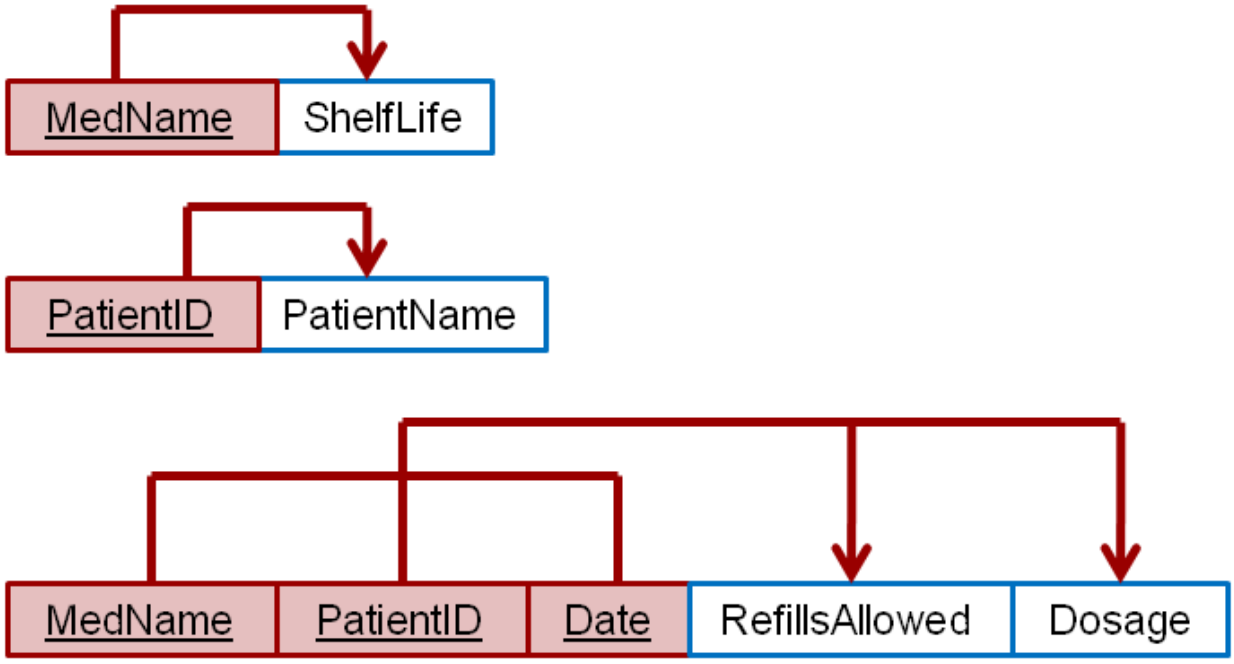
8. **The dependency diagram in Figure Q6.8 indicates that a patient can receive many prescriptions for one or more medicines over time. Based on the dependency diagram, create a database whose tables are in at least 2NF, showing the dependency diagram for each table.**

Figure Q6.8 Prescription dependency diagram



The normalization results are shown in Figure Q6.8a.

Figure Q6.8a The 2NF normalization results for Question 8.



9. **What is a partial dependency? With what normal form is it associated?**

   A partial dependency exists when an attribute is dependent on only a portion of the primary key. This type of dependency is associated with 1NF.

10. **What three data anomalies are likely to be the result of data redundancy? How can such anomalies be eliminated?**

The most common anomalies considered when data redundancy exists are: update anomalies, addition anomalies, and deletion anomalies. All these can easily be avoided through data normalization. Data redundancy produces data integrity problems, caused by the fact that data entry failed to conform to the rule that all copies of redundant data must be identical.

11. Define and discuss the concept of transitive dependency.

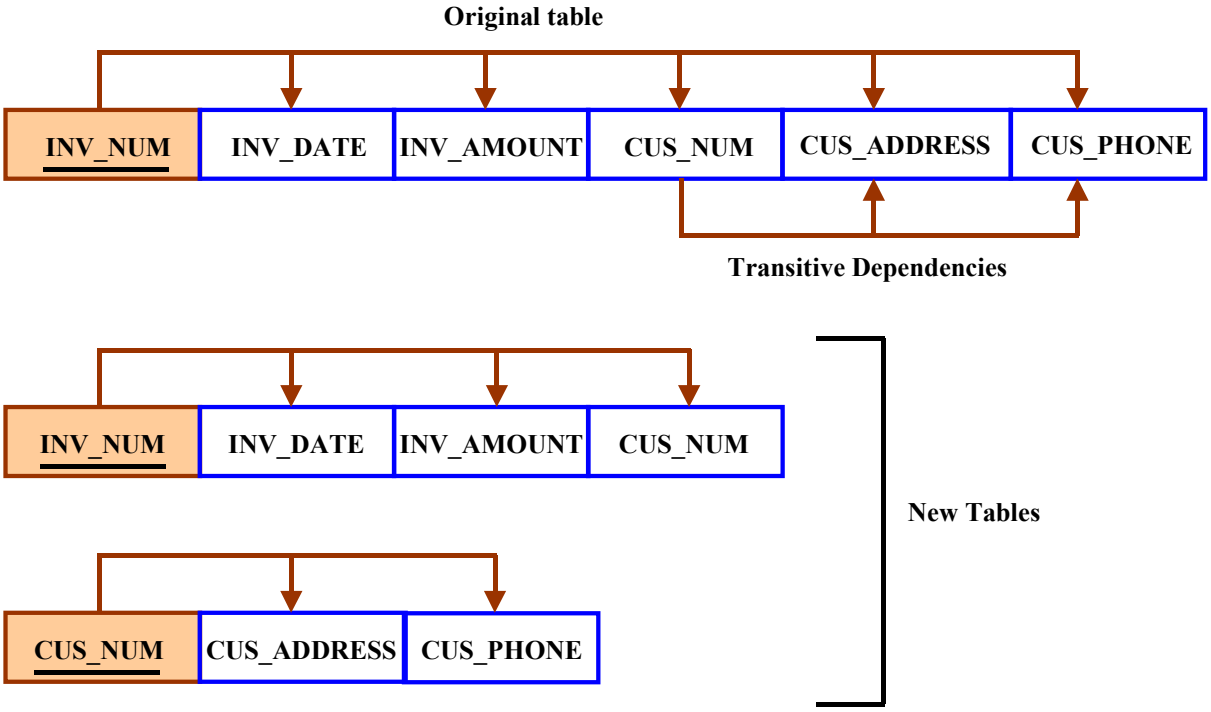
Transitive dependency is a condition in which an attribute is dependent on another attribute that is not part of the primary key. This kind of dependency usually requires the decomposition of the table containing the transitive dependency.

To remove a transitive dependency, the designer must perform the following actions:

- Place the attributes that create the transitive dependency in a separate table.
- Make sure that the new table's primary key attribute is the foreign key in the original table.

Figure Q6.9 shows an example of a transitive dependency removal.

Figure Q6.9 Transitive Dependency Removal



12. What is a surrogate key, and when should you use one?

A **surrogate key** is an artificial PK introduced by the designer with the purpose of simplifying the assignment of primary keys to tables. Surrogate keys are usually numeric, they are often automatically generated by the DBMS, they are free of semantic content (they have no special meaning), and they are usually hidden from the end users.

13. Why is a table whose primary key consists of a single attribute automatically in 2NF when it is in 1NF?

A dependency based on only a *part* of a composite primary key is called a **partial dependency**. Therefore, if the PK is a *single* attribute, there can be no partial dependencies.

14. How would you describe a condition in which one attribute is dependent on another attribute when neither attribute is part of the primary key?

This condition is known as a transitive dependency. A **transitive dependency** is a dependency of one nonprime attribute on another nonprime attribute. (The problem with transitive dependencies is that they still yield data anomalies.)

15. Suppose that someone tells you that an attribute that is part of a composite primary key is also a candidate key. How would you respond to that statement?

This argument is incorrect if the composite PK contains no redundant attributes. If the composite primary key is properly defined, *all* of the attributes that compose it are required to identify the remaining attribute values. By definition, a candidate key is one that can be used to identify all of the remaining attributes, but it was not chosen to be a PK for some reason. In other words, a candidate key can serve as a primary key, but it was not chosen for that task for one reason or another. Clearly, a part of a proper ("minimal") composite PK cannot be used as a PK by itself.

More formally, you learned in Chapter 3, "The Relational Database Model," Section 3-2, that a **candidate key** can be described as a superkey without redundancies, that is, a minimal superkey. Using this distinction, note that a STUDENT table might contain the composite key
STU_NUM, STU_LNAME
This composite key is a superkey, *but it is not a candidate key* because STU_NUM by itself is a candidate key! The combination
STU_LNAME, STU_FNAME, STU_INIT, STU_PHONE
might also be a candidate key, as long as you discount the possibility that two students share the same last name, first name, initial, and phone number.

If the student's Social Security number had been included as one of the attributes in the STUDENT table—perhaps named STU_SOCSECNUM—both it and STU_NUM would have been candidate keys because either one would uniquely identify each student. In that case, the selection of STU_NUM as the primary key would be driven by the designer's choice or by end-user requirements. Note, incidentally, that a primary key is a superkey as well as a candidate key.

16. A table is in ___3rd___ normal form when it is in ___2nd normal form___ and there are no transitive dependencies.

(See the discussion in Section 6-3c, “Conversion to Third Normal Form.”