CHAPTER 7

# Relational Database Design

## Practice Exercises

**7.1** Suppose that we decompose the schema $R = (A, B, C, D, E)$ into

$$(A, B, C)$$
$$(A, D, E).$$

Show that this decomposition is a lossless decomposition if the following set $F$ of functional dependencies holds:

$$A \rightarrow BC$$
$$CD \rightarrow E$$
$$B \rightarrow D$$
$$E \rightarrow A$$

**7.2** List all nontrivial functional dependencies satisfied by the relation of Figure 7.18.

**7.3** Explain how functional dependencies can be used to indicate the following:

- A one-to-one relationship set exists between entity sets *student* and *instructor*.

| $A$ | $B$ | $C$ |
|---|---|---|
| $a_1$ | $b_1$ | $c_1$ |
| $a_1$ | $b_1$ | $c_2$ |
| $a_2$ | $b_1$ | $c_1$ |
| $a_2$ | $b_1$ | $c_3$ |

**Figure 7.17** Relation of Exercise 7.2.

- A many-to-one relationship set exists between entity sets *student* and *instructor*.

**7.4** Use Armstrong's axioms to prove the soundness of the union rule. (*Hint*: Use the augmentation rule to show that, if $\alpha \rightarrow \beta$, then $\alpha \rightarrow \alpha\beta$. Apply the augmentation rule again, using $\alpha \rightarrow \gamma$, and then apply the transitivity rule.)

**7.5** Use Armstrong's axioms to prove the soundness of the pseudotransitivity rule.

**7.6** Compute the closure of the following set $F$ of functional dependencies for relation schema $R = (A, B, C, D, E)$.

$$A \rightarrow BC$$
$$CD \rightarrow E$$
$$B \rightarrow D$$
$$E \rightarrow A$$

List the candidate keys for $R$.

**7.7** Using the functional dependencies of Exercise 7.6, compute the canonical cover $F_c$.

**7.8** Consider the algorithm in Figure 7.19 to compute $\alpha^+$. Show that this algorithm is more efficient than the one presented in Figure 7.8 (Section 7.4.2) and that it computes $\alpha^+$ correctly.

**7.9** Given the database schema $R(A, B, C)$, and a relation $r$ on the schema $R$, write an SQL query to test whether the functional dependency $B \rightarrow C$ holds on relation $r$. Also write an SQL assertion that enforces the functional dependency. Assume that no null values are present. (Although part of the SQL standard, such assertions are not supported by any database implementation currently.)

**7.10** Our discussion of lossless decomposition implicitly assumed that attributes on the left-hand side of a functional dependency cannot take on null values. What could go wrong on decomposition, if this property is violated?

**7.11** In the BCNF decomposition algorithm, suppose you use a functional dependency $\alpha \rightarrow \beta$ to decompose a relation schema $r(\alpha, \beta, \gamma)$ into $r_1(\alpha, \beta)$ and $r_2(\alpha, \gamma)$.

a. What primary and foreign-key constraint do you expect to hold on the decomposed relations?

b. Give an example of an inconsistency that can arise due to an erroneous update, if the foreign-key constraint were not enforced on the decomposed relations above.

c. When a relation schema is decomposed into 3NF using the algorithm in Section 7.5.2, what primary and foreign-key dependencies would you expect to hold on the decomposed schema?

```
result := ∅;
/* fdcount is an array whose ith element contains the number
   of attributes on the left side of the ith FD that are
   not yet known to be in α+ */
for i := 1 to |F| do
   begin
     let β → γ denote the ith FD;
     fdcount [i] := |β|;
   end
/* appears is an array with one entry for each attribute. The
   entry for attribute A is a list of integers. Each integer
   i on the list indicates that A appears on the left side
   of the ith FD */
for each attribute A do
   begin
     appears [A] := NIL;
     for i := 1 to |F| do
       begin
         let β → γ denote the ith FD;
         if A ∈ β then add i to appears [A];
       end
   end
addin (α);
return (result);

procedure addin (α);
for each attribute A in α do
   begin
     if A ∉ result then
       begin
         result := result ∪ {A};
         for each element i of appears[A] do
           begin
             fdcount [i] := fdcount [i] − 1;
             if fdcount [i] := 0 then
               begin
                 let β → γ denote the ith FD;
                 addin (γ);
               end
           end
       end
   end
```

**Figure 7.18** An algorithm to compute $\alpha^+$.

**7.12** Let $R_1, R_2, \ldots, R_n$ be a decomposition of schema $U$. Let $u(U)$ be a relation, and let $r_i = \Pi_{R_I}(u)$. Show that

$$u \subseteq r_1 \bowtie r_2 \bowtie \cdots \bowtie r_n$$

**7.13** Show that the decomposition in Exercise 7.1 is not a dependency-preserving decomposition.

**7.14** Show that there can be more than one canonical cover for a given set of functional dependencies, using the following set of dependencies:

$$X \rightarrow YZ,\ Y \rightarrow XZ, \text{ and } Z \rightarrow XY.$$

**7.15** The algorithm to generate a canonical cover only removes one extraneous attribute at a time. Use the functional dependencies from Exercise 7.14 to show what can go wrong if two attributes inferred to be extraneous are deleted at once.

**7.16** Show that it is possible to ensure that a dependency-preserving decomposition into 3NF is a lossless decomposition by guaranteeing that at least one schema contains a candidate key for the schema being decomposed. (*Hint*: Show that the join of all the projections onto the schemas of the decomposition cannot have more tuples than the original relation.)

**7.17** Give an example of a relation schema $R'$ and set $F'$ of functional dependencies such that there are at least three distinct lossless decompositions of $R'$ into BCNF.

**7.18** Let a **prime** attribute be one that appears in at least one candidate key. Let $\alpha$ and $\beta$ be sets of attributes such that $\alpha \rightarrow \beta$ holds, but $\beta \rightarrow \alpha$ does not hold. Let $A$ be an attribute that is not in $\alpha$, is not in $\beta$, and for which $\beta \rightarrow A$ holds. We say that $A$ is **transitively dependent** on $\alpha$. We can restate the definition of 3NF as follows: A relation schema $R$ is in 3NF with respect to a set $F$ of functional dependencies if there are no nonprime attributes $A$ in $R$ for which $A$ is transitively dependent on a key for $R$. Show that this new definition is equivalent to the original one.

**7.19** A functional dependency $\alpha \rightarrow \beta$ is called a **partial dependency** if there is a proper subset $\gamma$ of $\alpha$ such that $\gamma \rightarrow \beta$; we say that $\beta$ is *partially dependent* on $\alpha$. A relation schema $R$ is in **second normal form** (**2NF**) if each attribute $A$ in $R$ meets one of the following criteria:

- It appears in a candidate key.
- It is not partially dependent on a candidate key.

Show that every 3NF schema is in 2NF. (*Hint*: Show that every partial dependency is a transitive dependency.)

**7.20** Give an example of a relation schema $R$ and a set of dependencies such that $R$ is in BCNF but is not in 4NF.