Algorithm

6.2 Membership Algorithm

Input: A set of functional dependencies **F** and the functional dependency $X \rightarrow Y$.

Output: Is $X \rightarrow Y \in F^+$ or not?

Compute X^+ using Algorithm 6.1.
if $Y \subseteq X^+$ *then* $X \rightarrow Y \in F^+ := true;$
else $X \rightarrow Y \in F^+ := false.$

If **G** covers **F** and if no proper subset $G'(G' \subseteq G)$ covers **F**, **G** is called a **nonredundant cover.**

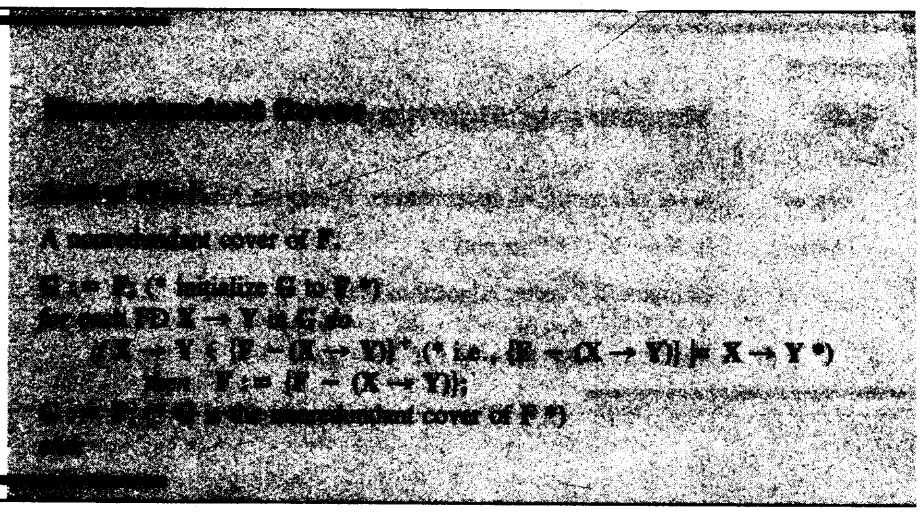
Definition: Given a set of FDs **F**, we say that it is nonredundant if no proper subset **F'** of **F** is equivalent to **F**, i.e., no **F'** exists such that $F'^+ = F^+$.

Given a functional dependency $X \rightarrow Y$, where $Y = A_1A_2A_3 \ldots A_n$, the functional dependency $X \rightarrow Y$ can be replaced by an equivalent set of FDs $\{X \rightarrow A_1, X \rightarrow A_2, X \rightarrow A_3, \ldots, X \rightarrow A_n\}$ by using the inference axioms **F4** and **F5** (additivity and projectivity). A nontrivial FD of the form $X \rightarrow A_i$ where the right-hand side has only one attribute is called a **simple** FD. Thus every set of FDs **F** can be replaced by an equivalent set of FDs G where G contains only simple FDs.

6.4.6 Nonredundant and Minimum Covers

Given **F**, a set of FDs, if a proper subset **F'** of **F** covers **F** (i.e., $F' \subset F$ and $F'^+ = F^+$), then **F** is redundant and we can remove some FD, say $X \rightarrow Y$, from **F** to find a nonredundant cover of **F.** Algorithm 6.3 finds a nonredundant cover of **F.** It does so by removing one FD $X \rightarrow Y$ from **F** and then checking if this FD is implied by the FD set $\{F - (X \rightarrow Y)\}$ by using Algorithms 6.1 and 6.2—finding the cover X^+ under the set of FDs $\{F - (X \rightarrow Y)\}$. If $\{F - (X \rightarrow Y)\} \models X \rightarrow Y$, then $X \rightarrow Y$ can be removed from **F.** Algorithm 6.3 repeats this procedure for each FD that remains in **F.** Note that the nonredundant cover so obtained depends on the order in which the functional dependencies are considered. Thus, starting with a set **F** of functional dependencies we can derive more than one nonredundant cover. (See Exercise 6.7.)

Algorithm 6.3



Example 6.10 If $\mathbf{F} = \{A \rightarrow BC, CD \rightarrow E, E \rightarrow C, D \rightarrow AEH, ABH \rightarrow BD, DH \rightarrow BC\}$ then the FDs $CD \rightarrow E$ and $DH \rightarrow BC$ are redundant. We find that $(CD)^+$ under $\{\mathbf{F} - (CD \rightarrow E)\}$ is equal to $ABCDEH$, and since the right-hand side of the FD $(CD \rightarrow E) \subseteq (CD)^+$ under $\{\mathbf{F} - (CD \rightarrow E)\}$, $\{\mathbf{F} - (CD \rightarrow E)\} \models CD \rightarrow E$. We now remove this redundant FD from $\mathbf{F}$ and then find that for the FD $(DH \rightarrow BC)$, $(DH)^+$ under $\{\mathbf{F} - (DH \rightarrow BC)\}$ is $ABCDEH$. Since the right-hand side of the FD $(DH \rightarrow BC) \subseteq (DH)^+$, the FD $(DH \rightarrow BC)$ is also redundant. No remaining FDs can be removed from the modified $\mathbf{F}$. Thus a nonredundant cover for $\mathbf{F}$ is $\{A \rightarrow BC, E \rightarrow C, D \rightarrow AEH, ABH \rightarrow BD\}$. ■

If $\mathbf{F}$ is a set of FDs and if $\mathbf{G}$ is a nonredundant cover of $\mathbf{F}$, then it is not true that $\mathbf{G}$ has the minimum number of FDs. In fact, there may exist a cover $\mathbf{G}'$ of $\mathbf{F}$ that has fewer FDs than $\mathbf{G}$. Thus, a minimum cover $\mathbf{G}'$ of $\mathbf{F}$ has as small a number of FDs as any other cover of $\mathbf{F}$. It is needless to add that a minimum cover $\mathbf{G}'$ of $\mathbf{F}$ has no redundant FDs; however, a nonredundant cover of $\mathbf{F}$ need not be minimal, as we see in Example 6.11. We will not discuss an algorithm to derive a minimum cover in this text. The interested reader is referred to the bibliographic notes at the end of the chapter.

6.4.7 Canonical Cover

A set of functional dependencies $\mathbf{F}_c$ is a **canonical cover** if every FD in $\mathbf{F}_c$ satisfies the following:

1. Each FD in $\mathbf{F}_c$ *is simple*. Recall that in a simple FD the right-hand side has a single attribute, i.e., each FD is of the form $\mathbf{X} \rightarrow A$.

2. For no FD $X \rightarrow A$ with $Z \subset X$ is $\{(F_c - (X \rightarrow A)) \cup (Z \rightarrow A)\} \vdash F_c$. In other words, the left-hand side of each FD does not have any extraneous attributes, or the *FDs* in F_c are left reduced.
3. No FD $X \rightarrow A$ is redundant, i.e., $\{F_c - (X \rightarrow A)\}$ does not logically imply F_c.

A canonical cover is sometimes called **minimal.**

Given a set **F** of functional dependencies we can find a canonical set F_c. Obviously F_c covers **F.**

SRINIVAS COLLEGE OF P.G. MANAGEMENT STUDIES
MANGALORE
575 001.

Example 6.11 If $F = \{A \rightarrow BC, CD \rightarrow E, E \rightarrow C, D \rightarrow AEH, ABH \rightarrow BD, DH \rightarrow BC\}$, then a nonredundant cover for **F** is $\{A \rightarrow BC, E \rightarrow C, D \rightarrow AEH, ABH \rightarrow BD\}$. The FD $ABH \rightarrow BD$ can be decomposed into the FDs $ABH \rightarrow B$ and $ABH \rightarrow D$. Now, since the FD $A \rightarrow B$ is in **F,** we can left reduce these decomposed FDs into $AH \rightarrow B$ and $AH \rightarrow D$. We also notice that $AH \rightarrow B$ is redundant since the FD $A \rightarrow B$ is already in **F.** This gives us the canonical cover as $\{A \rightarrow B, A \rightarrow C, E \rightarrow C, D \rightarrow A, D \rightarrow E, D \rightarrow H, AH \rightarrow D\}$. ■

Note that if F_c is a canonical cover and if we form **G** using the additivity axiom (such that the FDs with the same left-hand sides are merged into a single FD with the right-hand sides combined), then F_c and **G** are equivalent. However, G will contain nonsimple FDs.

6.4.8 Functional Dependencies and Keys

Earlier we discussed the concept of uniquely identifying an entity within an entity set by a key, the key being a set of attributes of the entity. A relation scheme **R** has a similar concept, which can be explained using functional dependencies.

Definition: Given a relation scheme $R\ \{A_1A_2A_3 \ldots A_n\}$ and a set of functional dependencies **F,** a key of **R** is a subset of **R** such that $K \rightarrow A_1A_2A_3 \ldots A_n$ is in F^+ and for any $Y \subset K$, $Y \rightarrow A_1A_2A_3 \ldots A_n$ is not in F^+.

The first requirement indicates that the dependency of all attributes of **R** on **K** is given explicitly in **F** or can be logically implied from **F.** The second requirement indicates that no proper subset of **K** can determine all the attributes of **R.** Thus, the key used here is minimal with respect to this property and the FD $K \rightarrow R$ is left reduced. A superset of **K** can then be called a superkey.

If there are two or more subsets of **R** such that the above conditions are satisfied, such subsets are called candidate keys. In such a case one of the candidate keys is designated as the primary key or simply as the key.

We do not allow any attribute in the key of a relation to have a null value.

Example 6.12 If **R** *(ABCDEH)* and **F** = $\{A \rightarrow BC, CD \rightarrow E, E \rightarrow C, D \rightarrow AEH, ABH \rightarrow BD, DH \rightarrow BC\}$, then *CD* is a key of **R** because $CD \rightarrow ABCDEH$ is in $\mathbf{F}^+$ (since $(\mathbf{CD})^+$ under **F** is equal to *ABCDEH* and $ABCDEH \subseteq ABCDEH$). Other candidate keys of **R** are *AD* and *ED*. ■

Full Functional Dependency

The concept of left-reduced FDs and fully functionally dependency is defined below and illustrated in Example 6.13.

Definition: **Given a relational scheme R and an FD $X \rightarrow Y$, Y is fully functionally dependent on X if there is no Z, where Z is a proper subset of X such that $Z \rightarrow Y$. The dependency $X \rightarrow Y$ is left reduced, there being no extraneous attributes in the left-hand side of the dependency.**

Example 6.13 In the relation scheme **R** *(ABCDEH)* with the FDs **F** = $\{A \rightarrow BC, CD \rightarrow E, E \rightarrow C, CD \rightarrow AH, ABH \rightarrow BD, DH \rightarrow BC\}$, the dependency $A \rightarrow BC$ is left reduced and *BC* is fully functionally dependent on *A*. However, the functional dependency $ABH \rightarrow D$ is not left reduced, the attribute *B* being extraneous in this dependency. ■

Prime Attribute and Nonprime Attribute

We defined the key of a relation scheme earlier. We distinguish the attributes that participate in any such key as indicated by the following definition.

Definition: **An attribute *A* in a relation scheme R is a prime attribute or simply prime if *A* is part of any candidate key of the relation. If *A* is not a part of any candidate key of R, *A* is called a nonprime attribute or simply nonprime.**

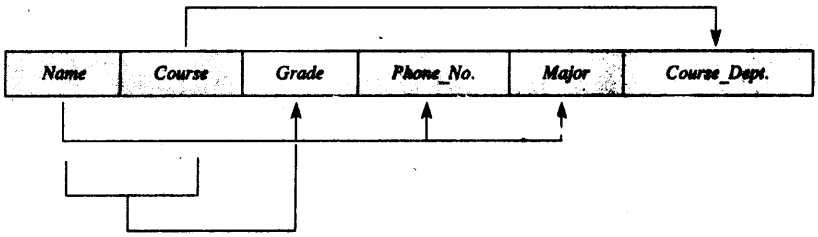
Example 6.14 If **R** *(ABCDEH)* and **F** = $\{A \rightarrow BC, CD \rightarrow E, E \rightarrow C, AH \rightarrow D\}$, then *AH* is the only candidate key of R. The attributes *A* and *H* are prime and the attributes *B*, *C*, *D*, and *E* are nonprime. ■

Partial Dependency

Let us introduce the concept of partial dependency below. We illustrate partial dependencies in Example 6.15.

Definition: Given a relation scheme **R** with the functional dependencies **F** defined on the attributes of **R** and **K** as a candidate key, if **X** is a proper subset of **K** and if **F** $\models$ **X** $\rightarrow$ *A*, then *A* is said to be **partially dependent** on **K**.

Example 6.15



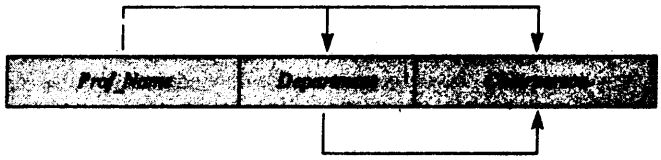
(a) In the relation scheme **STUDENT_COURSE_INFO**(*Name, Course, Grade, Phone_No, Major, Course_Dept)* with the FDs **F** = {*Name* $\rightarrow$ *Phone_NoMajor, Course* $\rightarrow$ *Course_Dept, NameCourse* $\rightarrow$ *Grade*}, *NameCourse* is a candidate key, *Name* and *Course* are prime attributes. *Grade* is fully functionally dependent on the candidate key. *Phone_No, Course_Dept,* and *Major* are partially dependent on the candidate key.
(b) Given **R** *(A, B, C, D)* and **F** = {*AB* $\rightarrow$ *C, B* $\rightarrow$ *D*}, the key of this relation is *AB* and *D* is partially dependent on the key. ■

Transitive Dependency

Another type of dependency which we have to recognize in database design is introduced below and illustrated in Example 6.16.

Definition: Given a relation scheme **R** with the functional dependencies **F** defined on the attributes of **R**, let **X** and **Y** be subsets of **R** and let *A* be an attribute of **R** such that **X** $\not\subset$ **Y**, *A* $\not\subset$ **XY**. If the set of functional dependencies {**X** $\rightarrow$ **Y**, **Y** $\rightarrow$ *A*} is implied by **F** (i.e., **F** $\models$ **X** $\rightarrow$ **Y** $\rightarrow$ *A* and **F** $\not\models$ **Y** $\rightarrow$ **X**), then *A* is transitively dependent on **X**.

Example 6.16



(a) In the relation scheme **PROF_INFO**(*Prof_Name, Department, Chairperson)* and the function dependencies **F** = {*Prof_Name* $\rightarrow$ *Department,*

Unnormalized Relation

Consider the table of Figure 6.9, which shows the preferences that faculty members have for teaching courses. As before, we allow the possibility of cross-departmental teaching. For instance, a faculty member in the Computer Science department may have a preference for a course in the Mathematics department, and so on. The table of Figure 6.9 is said to be **unnormalized.** Each row may contain multiple set of values for some of the columns; these multiple values in a single row are also called **nonatomic values.** In Figure 6.9 the row corresponding to the preferences of faculty in the Computer Science department has two professors. Professor Smith of the Computer Science department prefers to teach three different courses, and Professor Clark prefers four.

Definition: An unnormalized relation contains nonatomic *values*.

First Normal Form

The data of Figure 6.9 can be **normalized** into a relation, say CRS_PREF(*Prof, Course, Fac_Dept, Crs_Dept*), as shown in Figure 6.10. Note that we have shown

Figure 6.9 Course preferences.

Fac_Dept	*Prof*	*Course Preferences*	
		Course	*Course_Dept*
Comp Sci	Smith	353	Comp Sci
		379	Comp Sci
		221	Decision Sci
	Clark	353	Comp Sci
		351	Comp Sci
		379	Comp Sci
		456	Mathematics
Chemistry	Turner	353	Comp Sci
		456	Mathematics
		272	Chemistry
Mathematics	Jamieson	353	Comp Sci
		379	Comp Sci
		221	Decision Sci
		456	Mathematics
		469	Mathematics