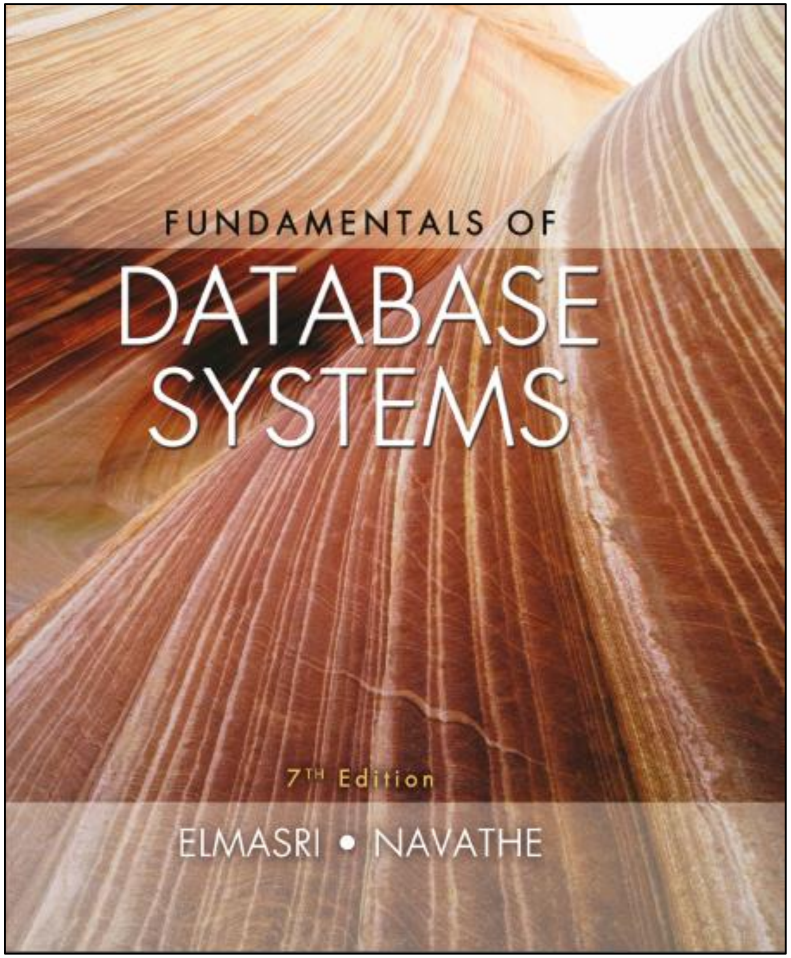
Fundamentals of Database Systems

Seventh Edition



Chapter 15

Relational Database Design Algorithms and Further Dependencies

Learning Objectives

15.1 Further topics in Functional Dependencies

Inference Rules for FDs

Minimal Sets of FDs

Functional Dependencies

A set of attributes X **functionally determines** a set of attributes Y if the value of X determines a unique value for Y.

If two rows agree on the X attributes, they must agree on the Y attributes.

SSN	LAST	FIRST
111	Smith	John
112	Brown	David
113	Smith	Joe
111	Smith	John
114	Jones	Susan
115	White	Ann

SSN → LAST
111 Smith
111 Smith

LAST → FIRST ??? No!!!
Smith John
Smith Joe

Functional Dependencies

A set of attributes X **functionally determines** a set of attributes Y if the value of X determines a unique value for Y.

If two rows agree on the X attributes, they must agree on the Y attributes.

R(SSN, STATE, COUNTRY)

F= {SSN $\rightarrow$ STATE,
STATE $\rightarrow$ COUNTRY}

R(A,B,C)

F={ A$\rightarrow$B, B$\rightarrow$ C}

Defining Functional Dependencies

- $X \rightarrow Y$ holds if whenever two tuples have the same value for X, they **must have** the same value for Y
  - For any two tuples t1 and t2 in any relation instance r(R); if $t1[X] = t2[X]$, **then** $t1[Y] = t2[Y]$
- $X \rightarrow Y$ in R specifies a **constraint** on all relation instances r(R)
- Written as $X \rightarrow Y$ can be displayed graphically on a relation schema as in Figures in Chapter 14. (denoted by the arrow:).
- FDs are derived from the real-world constraints on the attributes

15.1 Functional Dependencies : Inference Rules, Equivalence and Minimal Cover

- Our goal here is to determine the **properties** of functional dependencies and to find out the ways of manipulating them.
- Now we return to show how new dependencies can be inferred from a given set and discuss the concepts of

  closure,

  equivalence, and

  minimal cover

that we will need when we later consider a synthesis approach to design of relations given a set of FDs.

15.1.1 Inference Rules for FDs

- **Definition:** An $X \rightarrow Y$ is **inferred from** or **implied by** a set of dependencies *F* specified on *R* if $X \rightarrow Y$ holds in **every** legal relation state *r* of *R*; that is, whenever *r* satisfies all the dependencies in *F*, $X \rightarrow Y$ also holds in *r*.
- Given a set of FDs F, we can **infer** additional FDs that hold whenever the FDs in F hold

Inference Rules for FDs

- Armstrong's inference rules:
  - IR1. (**Reflexive**) If Y **subset-of** X, then $X \to Y$
  - IR2. (**Augmentation**) If $X \to Y$ then $XZ \to YZ$
    - (Notation: XZ stands for X U Z)
  - IR3. (**Transitive**) If $X \to Y$ and $Y \to Z$ then $X \to Z$

- IR1, IR2, IR3 form a **sound** and **complete** set of inference rules

These are rules hold and all other rules that hold can be deduced from these

sound: Produce only functional dependencies belonging to the closure

complete: Produce all the functional dependencies in the closure

Inference Rules for FDs

- Armstrong's inference rules:
  - IR1. (**Reflexive**) If Y **subset-of** X, then $X \to y$

Exp: X = {a, b, c, d, e}
Y = {a, b, c}

Exp: {fname, lname} $\to$ {fname}

- The reflexive rule **can also be** stated as $X \to X$; that is, any set of attributes functionally determines itself.

- The reflexive rule (IR1) states that a set of attributes always determines itself or any of its subsets, which is obvious. Because IR1 generates dependencies that are always true, such dependencies are called *trivial*.

- A functional dependency $X \to Y$ is **trivial** if $X \supseteq Y$; otherwise, it is **nontrivial**.

Inference Rules for FDs

- Armstrong's inference rules:
  - IR2. (**Augmentation**) If $x \rightarrow y$ then $xz \rightarrow yz$
    - (Notation: XZ stands for X U Z)

A → B

AC → BC

The augmentation rule (IR2) says that adding the same set of attributes to both the left- and right-hand sides of a dependency results in another valid dependency.

Inference Rules for FDs

- Armstrong’s inference rules:
  - IR3. (**Transitive**) If $x \rightarrow y$ and $y \rightarrow z$ then $x \rightarrow z$

According to IR3, functional dependencies are transitive.

Inference Rules for FDs

- Some additional inference rules that are useful:
  - **Decomposition:** If X → YZ, then X → Y and X → Z

SSN → Fname, Lname ➔ SSN → Fname

SSN → Lname

  - **Union:** If X → Y and X → Z, then X → YZ

SSN → Lname

SSN → Fname ➔ SSN → Fname, Lname

  - **Psuedotransitivity:** If X → Y and WY → Z, then WX → Z

SSN → Lname

Lic#, Lname → Fname ➔ Lic#, SSN → Fname

Inference Rules for FDs

Decomposition, Union, Psuedotransitivity inference rules, as well as any other inference rules, can be deduced from IR1, IR2, and IR3 (completeness property).

Closure

- Given a set of functional dependencies, what can we **logically imply**?

R(A,B,C)

F={ A→B, B→ C}

A→B

B→C

A→C (transitivity)

A→A (reflexivity)

B→B

C→C

….

Closure

- Given a set of functional dependencies, what can we logically imply?

R(A,B,C)

F={ A→B, B→ C}

A→B

B→C

A→C

A→A

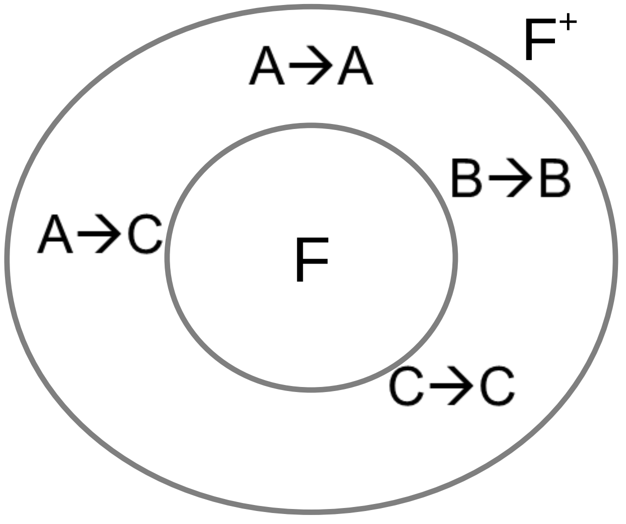
B→B

C→C

…

F^+ (F closure) is the super set of FDs.

F^+ is the set of FDs logically implied from F.



Closure

We need a formal mechanism in order to determine whether a FD is in F closure or not in F closure

- **Closure** of a set F of FDs is the set F^+ of all FDs that can be inferred from F
- **Closure** of a set of attributes X with respect to F is the set X^+ of all attributes that are functionally determined by X.

X^+ can be calculated by repeatedly applying IR1, IR2, IR3 using the FDs in F

Closure

Applications of closure set of attributes:

(1) To identify the additional FD's.

(2) To identify the keys.

(3) To identify the equivalences of the FD's

(4) To identify irreducible set of FD's or
canonical forms of FD's or
standard form of FD's.

Algorithm to determine Closure

- **Algorithm 15.1.** Determining X^+, the Closure of *X* under *F*
- **Input:** A set *F* of FDs on a relation schema R, and a set of attributes *X*, which is a subset of R.

X: one or more attributes from R.

X^+ is a set of attributes that may be derived by applying Armstrong's axioms.

Algorithm to determine Closure

- **Algorithm 15.1.** Determining X^+, the Closure of *X* under *F*
- **Input:** A set *F* of FDs on a relation schema R, and a set of attributes *X*, which is a subset of R.

$X^+ := X;$

repeat

old $X^+ := X^+;$

for each functional dependency $Y \to Z$ in F do

if $X^+ \supseteq Y$ then $X^+ := X^+ \cup Z;$

until ($X^+ =$ old X^+);

Example of Closure

Example 1: R(A, B, C)

Let *F*, the set of functional dependencies for the above relation include the following f.d.s:

FD1: $A \rightarrow B$

FD2: $B \rightarrow C$

- The closures of attributes or sets of attributes for some example sets:

$A^+ = \{ A, B, C\}$

$B^+ = \{ B, C\}$

$C^+ = \{C\}$

Closure

Example1: R(A, B, C, D, E)

F: {A→D, D→B, B→C, E→B}

A+

D+

E+

ACE+

AE+

Closure

Example2: R(A, B, C, D, E, F, G)

F: {A$\rightarrow$B, BC$\rightarrow$DE, AEG$\rightarrow$G}

AC+

Closure

Example3: R(A, B, C, D, E)

F: {A$\rightarrow$BC, CD$\rightarrow$E, B$\rightarrow$D, E$\rightarrow$A}

B+

Closure

Example4: R(A, B, C, D, E, F)

F: {AB→C, BC→AD, D→E, CF→B}

AB+

Closure

Example1: R(A, B, C, D, E, F, G, H)

F: {A→BC, CD→E, E→C, D→AEH, ABH→BD, DH→BC}

Is BCD → H ???

Is ABC → H ???

BCD+

ABC+

Example of Closure (1 of 2)

Example 2: Consider the following relation schema about classes held at a university in a given academic year.

CLASS (Classid, Course#, Instr_name, Credit_hrs, Text, Publisher, Classroom, Capacity).

Let *F*, the set of functional dependencies for the above relation include the following FD's:

FD1: Classid $\rightarrow$ Course#, Instr_name, Credit_hrs, Text, Publisher, Classroom, Capacity;

FD2: Course# $\rightarrow$ Credit_hrs;

FD3: {Course#, Instr_name} $\rightarrow$ Text, Classroom;

FD4: Text $\rightarrow$ Publisher

FD5: Classroom $\rightarrow$ Capacity

These FD's above represent the meaning of the individual attributes and the relationship among them and defines certain rules about the classes.

Example of Closure (2 of 2)

- The closures of attributes or sets of attributes for some example sets:

$\{$ Classid $\}^+ = \{$ Classid , Course#, Instr_name, Credit_hrs, Text, Publisher, Classroom, Capacity $\}$ = CLASS

$\{$ Course# $\}^+ = \{$ Course#, Credit_hrs $\}$

$\{$ Course#, Instr_name $\}^+ = \{$ Course#, Credit_hrs, Text, Publisher, Classroom, Capacity $\}$

Note that each closure above has an interpretation that is revealing about the attribute(s) on the left-hand-side. The closure of $\{\text{Classid}\}^+$ is the entire relation CLASS indicating that all attributes of the relation can be determined from Classid and hence it is a key.

New definition of a key

1) X is a **key** if it is functionally determines all the attributes in R (X → R)

2) X is minimal w.r.t. #1

(X is one or more attributes)

New definition of a key

1) X is a **key** if it is functionally determines all the attributes in R (X $\rightarrow$ R)

2) X is minimal w.r.t. #1

(X is one or more attributes)

minimal means the smallest possible amount, quantity, or degree,
whereas **minimum** means to the lowest degree.

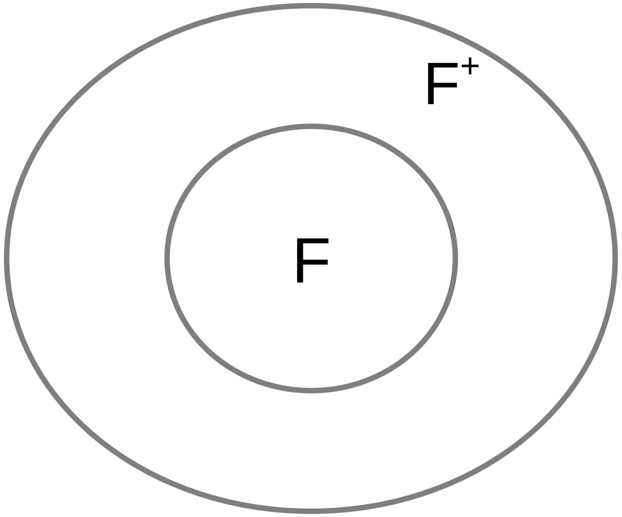
New definition of a key

1) X is a **key** if it is functionally determines all the attributes in R (X → R)

2) X is minimal w.r.t. #1

(X is one or more attributes)

minimal means the smallest possible amount, quantity, or degree, whereas **minimum** means to the lowest degree.



If we apply Armstrong's axioms to F,

we will not come up with any functional dependencies that go outside of F closure. We will find things inside of F^+

Algorithm to determine the key of a relation

- **Algorithm 15.2a Finding a Key K for R, given a set F of Functional Dependencies**
  - **Input: A universal relation R and a set of functional dependencies F on the attributes of R.**
    1. Set K := R;
    2. For each attribute A in K

$$\{\text{Compute } (K - A)^{+} \text{ with respect to } F;$$

$$\text{If } (K - A)^{+} \text{ contains all the attributes in } R,$$

$$\text{then set } K := K - \{A\} \};$$

Algorithm to determine the key of a relation

Example1: R(A, B, C, D, E)

F={A$\rightarrow$B, D$\rightarrow$E}

Superkey (SK): set of attributes whose closure contains all attributes of given relation

SK: ABCDE

$ABCDE^+$ ={A, B, C, D, E}

Find proper subset and check if it is a superkey

2^5-2 possibilities

Algorithm to determine the key of a relation

Example2: R(A, B, C, D)

F={A$\rightarrow$B, B$\rightarrow$C, C$\rightarrow$A}

SK: ABCD

$ABCD^+$ ={A, B, C, D}

Find proper subset and check if it is a superkey

Algorithm to determine the key of a relation

Example3: R(A, B, C, D, E, F)

F={AB$\rightarrow$C, C$\rightarrow$DE, E$\rightarrow$F, D$\rightarrow$A, C$\rightarrow$B}

SK: ABCDEF

$ABCDEF^+$ ={A, B, C, D, E, F}

Find proper subset and check if it is a superkey