Functional Dependencies

Functional Dependencies

- Functional dependencies (FDs) are used to specify *formal measures* of the "goodness" of relational designs
- FDs and keys are used to define **normal forms** for relations
- FDs are **constraints** that are derived from the *meaning* and *interrelationships* of the data attributes
- A set of attributes X *functionally determines* a set of attributes Y if the value of X determines a unique value for Y

Functional Dependencies (2)

- X → 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 → Y in R specifies a *constraint* on all relation instances r(R)
- Written as X → Y; can be displayed graphically on a relation schema as in Figures. (denoted by the arrow:).
- FDs are derived from the real-world constraints on the attributes

Examples of FD constraints (1)

- social security number determines employee name
  SSN → ENAME
- project number determines project name and location
  PNUMBER → {PNAME, PLOCATION}
- employee ssn and project number determines the hours per week that the employee works on the project
  {SSN, PNUMBER} → HOURS

Examples of FD constraints (2)

- An FD is a property of the attributes in the schema R
- The constraint must hold on *every relation instance* r(R)
- If K is a key of R, then K functionally determines all attributes in R (since we never have two distinct tuples with t1[K]=t2[K])

Inference Rules for FDs

- Given a set of FDs F, we can *infer* additional FDs that hold whenever the FDs in F hold

Armstrong's inference rules:

IR1. (**Reflexive**) If $X \supseteq Y$, then $X \rightarrow Y$

IR2. (**Augmentation**) If $X \rightarrow Y$, then $XZ \rightarrow YZ$

$\{X \rightarrow Y\} \models \{XZ \rightarrow YZ\}$

(XZ stands for X U Z)

IR3. (**Transitive**) If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$

$\{X \rightarrow Y, Y \rightarrow Z\} \models X \rightarrow Z$

Proofs of Armstrong's Axioms

IR1: Suppose $X \supseteq Y$ and that two tuples t1 and t2 exists in some relation instance r of R s. t. t1[X]=t2[X].

Then t1[Y]=t2[Y] because $X \supseteq Y$; hence $X \rightarrow Y$ must hold in R.

IR2: Assume that $X \rightarrow Y$ holds in a relation instance r of R, but $XZ \rightarrow YZ$ does not hold.

Then there must exist two tuples t1 and t2 in r s.t.

(1) t1[X] = t2[X] (2) t1[Y] = t2[Y]

(3) t1[XZ] = t2[XZ] (4) t1[YZ] $\neq$ t2[YZ]

This is not possible because from (1) and (3) we deduce

(5) t1[Z] = t2[Z] and from (2) and (5) we deduce

(6)t1[YZ] = t2[YZ], contradicting (4).

Proofs of Armstrong's Axioms

IR3: Assume that **(1)** $X \rightarrow Y$ and **(2)** $Y \rightarrow Z$ both hold in a relation r. Then for any two tuples t1 and t2 in r such that t1[X] = t2[X], we must have

(3) t1[Y] = t2[Y] (from assumption 1). Hence we must also have

(4) t1[Z] = t2[Z] (from 3 and assumption (2))

Hence $X \rightarrow Z$ must hold in r.

Additional Inference Rules

<u>Some **additional inference rules** that are useful:</u>

(**Decomposition**) If X -> YZ, then X -> Y and X -> Z

(**Union**) If X -> Y and X -> Z, then X -> YZ

(**Psuedotransitivity**) If X -> Y and WY -> Z, then WX -> Z

- The last three inference rules, as well as any other inference rules, can be deduced from IR1, IR2, and IR3

Proofs of Additional Inference Rules

IR4: {X→ YZ} $\models$ X → Y

Proof:

1. X → YZ (given)
2. YZ → Y (using IR1 and knowing that YZ ⊇ Y)
3. X → Y (using IR3 on 1 and 2)

IR5: {X → Y, X → Z} $\models$ X → YZ

Proof:

- X → Y (given)
- X → Z (given)
- X → XY (using IR2 on 1 by augmenting with X)
- XY → YZ (using IR2 on 2 by augmenting with Y)
- X → YZ (using IR3 on 3 and 4)

Proofs of Additional Inference Rules

IR6: {X→ Y, WY → Z} ⊨ WX → Z

Proof:

1. X → Y (given)
2. WY → Z (given)
3. WX → WY (using IR2 on 1 and augmenting with W)
4. WX → Z (using IR3 on 3 and 2)

Armstrong's axioms are SOUND & COMPLETE

Soundness:

Given a set of FDs F specified on a relational schema R, any dependency that we can infer from F by using IR1 through IR3 holds in every relation state r of R that satisfies the dependencies in F.

Completeness:

Using IR1 through IR3 repeatedly to infer dependencies until no more dependencies can be inferred results in the complete set of all possible dependencies that can be inferred from F

Closure of a set of FDs *F*

- **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 of a set of FDs *F*

Algorithm: Determining X^+, the closure of X under F

```
X⁺ ← X;
repeat
   oldX⁺ ← X⁺;
   for each FD Y → Z in F do
        if X⁺ ⊇ Y then X⁺ ← X⁺ ∪ Z;
until (X⁺ = oldX⁺);
```

Closure of a set of FDs *F*

Example: Given the relation
EMP_PROJ (<u>SSN, PNUMBER</u>, HOURS, ENAME, PNAME, PLOCATION)
and a set of FDs F on it, as follows:

F = {SSN → ENAME,
PNUMBER → {PNAME, PLOCATION},
{SSN, PNUMBER → HOURS}

Find F^+ the closure of F.

$\{SSN\}^+ = \{SSN, ENAME\}$

$\{PNUMBER\}^+ = \{PNUMBER, PNAME, PLOCATION\}$

$\{SSN, PNUMBER\}^+ = \{SSN, PNUMBER, ENAME, PNAME, PLOCATION, HOURS\}$

Equivalence of Sets of FDs

- Two sets of FDs F and G are **equivalent** if:
  - every FD in F can be inferred from G, *and*
  - every FD in G can be inferred from F
- Hence, F and G are equivalent if $F^+ = G^+$

<u>Definition:</u> F **covers** G if every FD in G can be inferred from F (i.e., if G^+ <u>*subset-of*</u> F^+)

- F and G are equivalent if F covers G and G covers F

Determining whether *F covers G*

- Calculate X+ with respect to F for each FD, X → Y in G
- Check whether this X+ includes the attributes in Y
- If this is the case for every FD in G, then ***F covers G***

- We can determine **whether F and G are equivalent** by checking whether ***F covers G and G covers F***

Minimal Sets of FDs

- A set of FDs is **minimal** if it satisfies the following conditions:

(1) Every dependency in F has a single attribute for its RHS.

(2) We cannot remove any dependency from F and still have a set of dependencies that is equivalent to F.

(3) We cannot replace any dependency X → A in F with a dependency Y → A, where Y is a proper subset of X and still have a set of dependencies that is equivalent to F.

Minimal Sets of FDs (2)

- A minimal set of dependencies is a set of dependencies in a standard canonical form and with no redundancies
- Condition 1 represents every dependency in a canonical form with a single attribute on the RHS
- Condition 2 and 3 ensure that there is no redundancy either by having a
  - Redundant dependency that can be inferred from the remaining FDs in F
  - Redundant attributes on the LHS of a dependency

Minimal Cover

- A Minimal cover of a set of FDs E is a minimal set of dependencies F that is equivalent to E
- There can be several minimal covers for a set of FDs
- Additional criteria for minimality
  - Minimal set with the smallest no. of dependencies
  - Minimal set with the smallest total length
    - Total Length is obtained by concatenating all the dependencies and treating them as one long character string

Algorithm: Finding a Minimal Cover F for a set of FDs E

1. Set F $\leftarrow$ E;
2. Replace each FD X $\rightarrow$ {A1, A2, ……,An} in F by the n FDs X $\rightarrow$ A1, X $\rightarrow$ A2,……., X$\rightarrow$ An.
3. For each FD X $\rightarrow$ A in F

   for each attribute B that is an element of X

   if {{F – {X $\rightarrow$ A}} U {(X - {B}) $\rightarrow$ A}} is equivalent to F, then replace X $\rightarrow$ A with (X – {B}) $\rightarrow$ A in F.
4. For each remaining FD X $\rightarrow$ A in F

   if { F - {X $\rightarrow$ A}} is equivalent to F,

   then remove X $\rightarrow$ A from F.

Example

Consider the relation schema

EMP_DEPT (ENAME, SSN, BDATE, ADDRESS, DNUMBER, DNAME, DMGRSSN) and the following set G of functional dependencies on EMP_DEPT:

G = {SSN ->{ENAME, BDATE, ADDRESS, DNUMBER} , DNUMBER ->{DNAME, DMGRSSN} }

Is the set of functional dependencies G minimal? If not, try to find a minimal set of functional dependencies that is equivalent to G. Prove that your set is equivalent to G.

ANSWER:

The set G of functional dependencies is not minimal, because it violates rule 1 of minimality (every FD has a single attribute for its right hand side). The set F is an equivalent minimal set:

F= {SSN ->{ENAME},SSN ->{BDATE},
SSN-> {ADDRESS}, SSN ->{DNUMBER} , DNUMBER ->{DNAME}, DNUMBER->{DMGRSSN}}

To show equivalence, we prove that G is covered by F and F is covered by G.

Proof that G is covered by F:

{SSN} + = {SSN, ENAME, BDATE, ADDRESS, DNUMBER, DNAME, DMGRSSN} (with respect to F), which covers SSN ->{ENAME, BDATE, ADDRESS, DNUMBER} in G

{DNUMBER} + ={DNUMBER, DNAME, DMGRSSN} (with respect to F), which covers DNUMBER ->{DNAME, DMGRSSN} in G

Proof that F is covered by G:

{SSN} + = {SSN, ENAME, BDATE, ADDRESS, DNUMBER, DNAME, DMGRSSN} (with respect to G), which covers SSN ->{ENAME}, SSN ->{BDATE}, SSN ->{ADDRESS}, and SSN ->{DNUMBER} in F

{DNUMBER} + = {DNUMBER, DNAME, DMGRSSN} (with respect to G), which covers DNUMBER ->{DNAME} and DNUMBER->{DMGRSSN} in F

Acknowledgement

Reference for this lecture is

- Ramez Elmasri and Shamkant B. Navathe, *Fundamentals of Database Systems,* Pearson Education.

The authors and the publishers are gratefully acknowledged.