Part 7:

Multivalued Dependencies

Database System Concepts, 7th Ed.

Multivalued Dependencies (MVDs)

- Assume each instructor can have multiple children and multiple phone numbers
  - Children's names are not unique
  - A given phone number can be shared among several instructors
  - Two instructors can have the same child's name and same phone number, so that child's name and phone number cannot uniquely identify an instructor
- Suppose we record names of children, and phone numbers for instructors:
  - *inst_child*(*ID*, *child_name*)
  - *inst_phone*(*ID*, *phone_number*)
- If we were to combine these schemas to get
  - *inst_info*(*ID*, *child_name*, *phone_number*)
  - Example data:
(99999, David, 512-555-1234)
(99999, David, 512-555-4321)
(99999, William, 512-555-1234)
(99999, William, 512-555-4321)
- This relation is in BCNF but there is repetition of information
- Here, we note that a given instructor has a set of values for his/her children (rather than always a single value as in FD) independent of other attributes

Multivalued Dependencies (MVDs)

- Consider the relation

  Instructor (ID, dept_name, address)

- Assume each instructor can now be associated with multiple departments, and they also can have multiple residential addresses, and these are independent

- As *address* and *dept_name* are independent, the following instance is illegal

ID	dept_name	address
22222	Physics	North St., Rye
22222	Math	Main St., Manchester

Since it suggests *dept_name* and *address* are not independent

Multivalued Dependencies (MVDs)

- To make it legal, we need to repeat the department name once for each address that an instructor has, and repeat the address once for each department with which an instructor is associated

  ⇒ we need to add the two tuples

  （22222, Physics, Main St., Manchester）, and

  （22222, Math, North St., Rye）

ID	dept_name	address
22222	Physics	North St., Rye
22222	Physics	Main St., Manchester
22222	Math	North St., Rye
22222	Math	Main St., Manchester

Multivalued Dependencies

- Consider the EMP relation R (α, β, γ) below, where each employee can work in multiple projects and can have multiple dependents, and these are independent

EMP

Ename	Pname	Dname
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

α = Ename

β = Pname is repeated in tuples 1 & 3, and repeated in tuples 2 & 4 due to there being 2 Dnames associated with Smith – a given Pname is included once for each Dname

γ = Dname is repeated in tuples 2 & 3, and repeated in tuples 1 & 4 due to there being 2 Pnames associated with Smith – a given Dname is included once for each Pname

- Let R (α, β, γ) be a relation schema. The **multivalued dependency**

$$\alpha \rightarrow\rightarrow \beta$$

holds on R if in any legal relation $r(R)$, we have the constraint: if two tuples t_1 and t_2 exist in r such that $t_1[\alpha] = t_2[\alpha]$, then there exist tuples t_3 and t_4 in r such that:

$$t_1[\alpha] = t_2[\alpha] = t_3[\alpha] = t_4[\alpha]$$
$$t_3[\beta] = t_1[\beta] \text{ and } t_4[\beta] = t_2[\beta]$$
$$t_3[\gamma] = t_2[\gamma] \text{ and } t_4[\gamma] = t_1[\gamma]$$

The tuples t_1, t_2, t_3 and t_4 are not necessarily distinct

- $t_1[\alpha] = t_2[\alpha]$ implies α has multiple values (more than 1 value) for one of the other columns. Let this be the β column; then each of these 2 tuples have to be repeated due to multiple values (more than 1 value) occurring in the γ column

Multivalued Dependencies

EMP

Ename	Pname	Dname
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

- For $\alpha \rightarrow\rightarrow \beta$, we had:

$$t_1 [\alpha] = t_2 [\alpha] = t_3 [\alpha] = t_4 [\alpha]$$
$$t_3 [\beta] = t_1 [\beta] \text{ and } t_4 [\beta] = t_2 [\beta]$$
$$t_3 [\gamma] = t_2 [\gamma] \text{ and } t_4 [\gamma] = t_1 [\gamma]$$

- Since the ordering of 1 and 2 is arbitrary, swapping the subscripts 1 and 2, we get:

$$t_1 [\alpha] = t_2 [\alpha] = t_3 [\alpha] = t_4 [\alpha]$$
$$t_3 [\beta] = t_2 [\beta] \text{ and } t_4 [\beta] = t_1 [\beta]$$
$$t_3 [\gamma] = t_1 [\gamma] \text{ and } t_4 [\gamma] = t_2 [\gamma]$$

which is $\alpha \rightarrow\rightarrow \gamma$

- Thus, whenever $\alpha \rightarrow\rightarrow \beta$ holds in R (α, β, γ), so does $\alpha \rightarrow\rightarrow \gamma$

MVD

EMP

Ename	Pname	Dname
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

- Let *R* (α, β, γ) be a relation schema
- We say that $\alpha \rightarrow\rightarrow \beta$ (α **multi-determines** β) if and only if for all possible relations *r* (*R*)

  $$< \alpha_1, \beta_1, \gamma_1 > \in r \text{ and } < \alpha_1, \beta_2, \gamma_2 > \in r$$

  then

  $$< \alpha_1, \beta_1, \gamma_2 > \in r \text{ and } < \alpha_1, \beta_2, \gamma_1 > \in r$$

- Note that since the behavior of β and γ are identical it follows that $\alpha \rightarrow\rightarrow \gamma$ if $\alpha \rightarrow\rightarrow \beta$, and this is sometimes written as $\alpha \rightarrow\rightarrow \beta|\gamma$

Example

- In our first example:

$$\mathit{ID} \rightarrow\rightarrow \mathit{child\_name}$$
$$\mathit{ID} \rightarrow\rightarrow \mathit{phone\_number}$$

- The above formal definition is to formalize the notion that given a particular value of α (*ID*) it has associated with it a set of values of β *(child_name)* and a set of values of γ *(phone_number)*, and these two sets are independent of each other

Use of Multivalued Dependencies

- We use multivalued dependencies in two ways:
  1. To test relations to **determine** whether they are legal under a given set of functional and multivalued dependencies
  2. To specify **constraints** on the set of legal relations
     - we shall concern ourselves *only* with relations that satisfy a given set of functional and multivalued dependencies
- If a relation *r* fails to satisfy a given multivalued dependency, we can construct a relation *r′* that does satisfy the multivalued dependency by adding tuples to *r*

Theory of MVDs

- From the definition of multivalued dependency, we have
  - If $\alpha \rightarrow \beta$, then $\alpha \rightarrow\rightarrow \beta$

  That is, every functional dependency is also a multivalued dependency
- Intuitively, there is a set of β values associated with a given α value, and the cardinality of that set can be one
- Functional dependencies rule out certain tuples from being in a relation
  - If $\alpha \rightarrow \beta$, then we cannot have two tuples with the same α value but different β values
  - Therefore, FDs are sometimes called equality-generating dependencies
- Multivalued dependencies do not rule out the existence of certain tuples
  - Instead they require that other tuples of a certain form be present in the relation
  - Therefore, MVDs are sometimes called tuple-generating dependencies
- The **closure** D^+ of D is the set of all functional and multivalued dependencies logically implied by D

Restriction of Multivalued Dependencies

- The **restriction** of D to R_i is the set D_i consisting of
  - All functional dependencies in D^+ that include only attributes of R_i
  - All multivalued dependencies of the form

    $$\alpha \rightarrow\rightarrow (\beta \cap R_i)$$

  where $\alpha \subseteq R_i$ and $\alpha \rightarrow\rightarrow \beta$ is in D^+

Trivial MVD

- An MVD $\alpha \rightarrow\rightarrow \beta$ is called a trivial MVD if
  - (a) $\beta \subseteq \alpha$, or
  - (b) $\alpha \cup \beta = R$
- An MVD that satisfies neither (a) nor (b) is called a non-trivial MVD
  - The above indicates that any non-trivial MVD requires at least 3 columns
- A trivial MVD will hold in any relation state *r* of *R*, and does not specify any significant or meaningful constraint on *R*
  - Condition (a) is similar to that for trivial FD, and condition (b) reduces *R* to the (two) columns α and β
- For the following relations, we have the trivial MVDs due to condition (b)
  - Ename $\rightarrow\rightarrow$ Pname in EMP_PROJECTS
  - Ename $\rightarrow\rightarrow$ Dname EMP_DEPENDENTS

EMP_PROJECTS

Ename	Pname
Smith	X
Smith	Y

EMP_DEPENDENTS

Ename	Dname
Smith	John
Smith	Anna

Fourth Normal Form

- A relation schema R is in **4NF** with respect to a set D of functional and multivalued dependencies if for all multivalued dependencies in D^+ of the form $\alpha \rightarrow\rightarrow \beta$, where $\alpha \subseteq R$ and $\beta \subseteq R$, at least one of the following holds:
  - $\alpha \rightarrow\rightarrow \beta$ is a trivial MVD
  - α is a superkey for schema R
- We can say: any (non-trivial) multi-determinant in R must be a superkey of R
- If a relation is in 4NF it is in BCNF

4NF Decomposition Algorithm

result: = {*R*};
done := false;
compute D^+;
Let D_i denote the restriction of D^+ to R_i

while (**not** *done*)
    if (there is a schema $\mathbf{R}_i$ in *result* that is not in 4NF) **then**
        begin
            let $\alpha \rightarrow\rightarrow \beta$ be a nontrivial multivalued dependency
              that holds on R_i such that α is not a superkey of R_i, and $\alpha \cap \beta = \phi$;
            result := (*result* - R_i) $\cup$ (R_i - β) $\cup$ (α, β);
        end
    else *done*:= true;

4NF Lossless Decomposition

- Let R be a relation schema, and let D be a set of FDs and MVDs on R.
- Let R_1 and R_2 form a decomposition of R. Then this decomposition is lossless if and only if at least one of the following MVDs is in D^+

$$R_1 \cap R_2 \rightarrow\rightarrow R_1$$

$$R_1 \cap R_2 \rightarrow\rightarrow R_2$$

- In the 4NF decomposition algorithm, we had $\alpha \rightarrow\rightarrow \beta$ leading to the decomposition $(R_i - \beta)$, (α, β)
  - Here the intersection of these two tables is α which multi-determines the second table (since $\alpha \rightarrow\rightarrow \alpha$, $\alpha \rightarrow\rightarrow \beta$, therefore $\alpha \rightarrow\rightarrow \alpha\beta$)
  - Thus, this is a lossless decomposition

Example

- $R = (A, B, C, G, H, I)$

  $F = \{ A \rightarrow\rightarrow B$

  $B \rightarrow\rightarrow HI$

  $CG \rightarrow\rightarrow H \}$
- R is not in 4NF since $A \rightarrow\rightarrow B$ and A is not a superkey for R
- Decomposition

  a) $R_1 = (A, B)$ (R_1 is in 4NF since $A \cup B = R_1$, therefore $A \rightarrow\rightarrow B$ is trivial)

  b) $R_2 = (A, C, G, H, I)$ (R_2 is not in 4NF, since $CG \rightarrow\rightarrow H$ and CG not a superkey of R_2 ; decompose R_2 into R_3 and R_4)

  c) $R_3 = (C, G, H)$ (R_3 is in 4NF, trivial MVD)

  d) $R_4 = (A, C, G, I)$ (R_4 is not in 4NF, decompose into R_5 and R_6)

  - $A \rightarrow\rightarrow B$ and $B \rightarrow\rightarrow HI$ implies $A \rightarrow\rightarrow HI$, (MVD transitivity), and
  - and hence $A \rightarrow\rightarrow I$ (MVD restriction to R_4) and A is not a superkey

  e) $R_5 = (A, I)$ (R_5 is in 4NF, trivial MVD)

  f) $R_6 = (A, C, G)$ (R_6 is in 4NF, no non-trivial MVD)

Further Normal Forms

- **Join dependencies** generalize multivalued dependencies
  - lead to **project-join normal form (PJNF)** (also called **fifth normal form**)
- A class of even more general constraints, leads to a normal form called **domain-key normal form**.
- Problem with these generalized constraints: are hard to reason with, and no set of sound and complete set of inference rules exists.
  - Hence rarely used

Overall Database Design Process

We have assumed schema *R* is given

- *R* could have been generated when converting E-R diagram to a set of tables
- *R* could have been a single relation containing *all* attributes that are of interest (called **universal relation**)
- Normalization breaks *R* into smaller relations
- *R* could have been the result of some ad hoc design of relations, which we then test/convert to normal form

ER Model and Normalization

- When an E-R diagram is carefully designed, identifying all entities correctly, the tables generated from the E-R diagram should not need much further normalization
- However, in a real (imperfect) design, there can be functional dependencies from non-key attributes of an entity to other attributes of the entity
  - Example: an *employee* entity with
    - attributes
      department_name and *building*,
    - functional dependency
      $department\_name \rightarrow building$
    - Good design would have made department an entity
- Functional dependencies from non-key attributes of a relationship set possible, but rare --- and most relationships are binary

Denormalization for Performance

- May want to use non-normalized schema for performance
- For example, displaying *prereq* along with *course_id,* and *title* requires join of *course* with *prereq* (assuming *prereq* only holds *course_id*)
- Use denormalized relation containing attributes of *course* as well as *prereq* with all above attributes
  - faster lookup
  - extra space and extra execution time for updates