Multivalued Dependencies

Fourth Normal Form

Reasoning About FD's + MVD's

Definition of MVD

- A *multivalued dependency* (MVD) on *R*, *X* ->-> *Y*, says that if two tuples of *R* agree on all the attributes of *X*, then their components in *Y* may be swapped, and the result will be two tuples that are also in the relation.
- i.e., for each value of *X*, the values of *Y* are independent of the values of *R*-*X*-*Y*.

Example: MVD

Drinkers(name, addr, phones, lemonadesLiked)

- A drinker's phones are independent of the lemonades they like.
  - name->->phones and name ->->lemonadesLiked.
- Thus, each of a drinker's phones appears with each of the lemonades they like in all combinations.
- This repetition is unlike FD redundancy.
  - name->addr is the only FD.

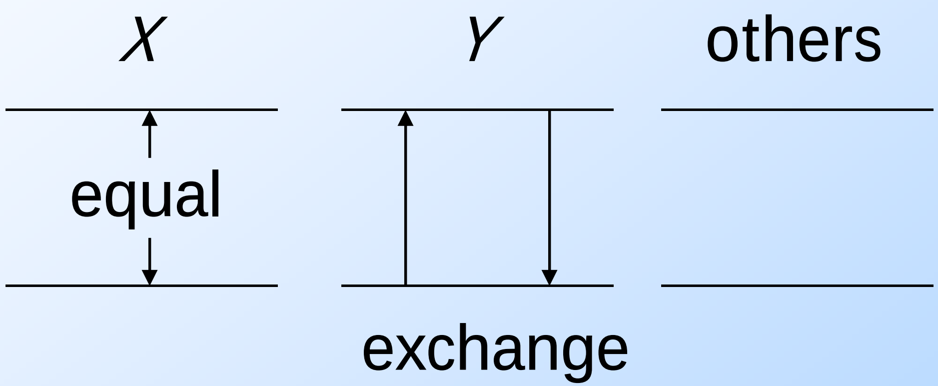
Tuples Implied by name->->phones

If we have tuples:

name	addr	phones	lemonadesLiked
sue	a	p1	l1
sue	a	p2	l2
sue	a	p2	l1
sue	a	p1	l2

Then these tuples must also be in the relation.

Picture of MVD *X* ->-> *Y*



MVD Rules

- Every FD is an MVD (*promotion*).
  - If *X* -> *Y*, then swapping *Y* 's between two tuples that agree on *X* doesn't change the tuples.
  - Therefore, the "new" tuples are surely in the relation, and we know *X* ->-> *Y*.
- *Complementation* : If *X* ->-> *Y*, and *Z* is all the other attributes, then *X* ->-> *Z*.

Splitting Doesn't Hold

- Like FD's, we cannot generally split the left side of an MVD.
- But unlike FD's, we cannot split the right side either --- sometimes you have to leave several attributes on the right side.

Example: Multiattribute Right Sides

Drinkers(name, areaCode, phone, lemonadesLiked, manf)

- A drinker can have several phones, with the number divided between areaCode and phone (last 7 digits).
- A drinker can like several lemonades, each with its own manufacturer.

Example Continued

- Since the areaCode-phone combinations for a drinker are independent of the lemonadesLiked-manf combinations, we expect that the following MVD's hold:

  name ->-> areaCode phone

  name ->-> lemonadesLiked manf

Example Data

Here is possible data satisfying these MVD's:

name	areaCode	phone	lemonadesLiked	manf
Sue	650	555-1111	Bud	A.B.
Sue	650	555-1111	WickedAle	Pete's
Sue	415	555-9999	Bud	A.B.
Sue	415	555-9999	WickedAle	Pete's

But we cannot swap area codes or phones by themselves. That is, neither name->->areaCode nor name->->phone holds for this relation.

Fourth Normal Form

- The redundancy that comes from MVD's is not removable by putting the database schema in BCNF.
- There is a stronger normal form, called 4NF, that (intuitively) treats MVD's as FD's when it comes to decomposition, but not when determining keys of the relation.

4NF Definition

- A relation *R* is in *4NF* if: whenever *X* ->-> *Y* is a nontrivial MVD, then *X* is a superkey.
  - *Nontrivial MVD* means that:
    1. *Y* is not a subset of *X*, and
    2. *X* and *Y* are not, together, all the attributes.
  - Note that the definition of "superkey" still depends on FD's only.

BCNF Versus 4NF

- Remember that every FD *X* -> *Y* is also an MVD, *X* ->-> *Y*.
- Thus, if *R* is in 4NF, it is certainly in BCNF.
  - Because any BCNF violation is a 4NF violation (after conversion to an MVD).
- But *R* could be in BCNF and not 4NF, because MVD's are "invisible" to BCNF.

Decomposition and 4NF

- If X ->-> Y is a 4NF violation for relation R, we can decompose R using the same technique as for BCNF.
  1. XY is one of the decomposed relations.
  2. All but $Y - X$ is the other.

Example: 4NF Decomposition

Drinkers(name, addr, phones, lemonadesLiked)

FD: name -> addr

MVD's: name ->-> phones

name ->-> lemonadesLiked

- Key is {name, phones, lemonadesLiked}.
- All dependencies violate 4NF.

Example Continued

- Decompose using name -> addr:

1. Drinkers1(<u>name</u>, addr)
   - In 4NF; only dependency is name -> addr.
2. Drinkers2(<u>name</u>, <u>phones</u>, <u>lemonadesLiked</u>)
   - Not in 4NF. MVD's name ->-> phones and name ->-> lemonadesLiked apply. No FD's, so all three attributes form the key.

Example: Decompose Drinkers2

- Either MVD name ->-> phones or name ->-> lemonadesLiked tells us to decompose to:
  - Drinkers3(name, phones)
  - Drinkers4(name, lemonadesLiked)

Reasoning About MVD's + FD's

- Problem: given a set of MVD's and/or FD's that hold for a relation *R*, does a certain FD or MVD also hold in *R* ?
- Solution: Use a tableau to explore all inferences from the given set, to see if you can prove the target dependency.

Why Do We Care?

1. 4NF technically requires an MVD violation.
   - Need to infer MVD's from given FD's and MVD's that may not be violations themselves.
2. When we decompose, we need to project FD's + MVD's.

Example: Chasing a Tableau With MVD's and FD's

- To apply a FD, equate symbols, as before.
- To apply an MVD, generate one or both of the tuples we know must also be in the relation represented by the tableau.
- We'll prove: if A->->BC and D->C, then A->C.

The Tableau for A->C

Goal: prove that $c_1 = c_2$.

A	B	C	D
a	b_1	~~c_1~~ c_2	d_1
a	b_2	c_2	d_2
a	b_2	c_2	d_1

Use A->->BC (first row's D with second row's BC).

Use D->C (first and third row agree on D, therefore agree on C).

Example: Transitive Law for MVD's

- If A->->B and B->->C, then A->->C.
  - Obvious from the complementation rule if the Schema is *ABC*.
  - But it holds no matter what the schema; we'll assume *ABCD*.

The Tableau for A->->C

Goal: derive tuple (a, b_1, c_2, d_1).

A	B	C	D
a	b_1	c_1	d_1
a	b_2	c_2	d_2
a	b_1	c_2	d_2
a	b_1	c_2	d_1

Use A->->B to swap B from the first row into the second.

Use B->->C to swap C from the third row into the first.

Rules for Inferring MVD's + FD's

- Start with a tableau of two rows.
  - These rows agree on the attributes of the left side of the dependency to be inferred.
  - And they disagree on all other attributes.
  - Use unsubscripted variables where they agree, subscripts where they disagree.

Inference: Applying a FD

- Apply a FD X->Y by finding rows that agree on all attributes of *X*. Force the rows to agree on all attributes of *Y*.
  - Replace one variable by the other.
  - If the replaced variable is part of the goal tuple, replace it there too.

Inference: Applying a MVD

- Apply a MVD X->->Y by finding two rows that agree in *X*.
  - Add to the tableau one or both rows that are formed by swapping the *Y*-components of these two rows.

Inference: Goals

- To test whether U->V holds, we succeed by inferring that the two variables in each column of V are actually the same.
- If we are testing U->->V, we succeed if we infer in the tableau a row that is the original two rows with the components of V swapped.

Inference: Endgame

- Apply all the given FD's and MVD's until we cannot change the tableau.
- If we meet the goal, then the dependency is inferred.
- If not, then the final tableau is a counterexample relation.
  - Satisfies all given dependencies.
  - Original two rows violate target dependency.

A Complete Set of Inference Rules

1. *Reflexivity.* If $\{B_1, B_2, \ldots, B_m\} \subseteq \{A_1, A_2, \ldots, A_n\}$, then $A_1A_2\cdots A_n \rightarrow B_1B_2\cdots B_m$. These are what we have called trivial FD's.

2. *Augmentation.* If $A_1A_2\cdots A_n \rightarrow B_1B_2\cdots B_m$, then

$$A_1A_2\cdots A_nC_1C_2\cdots C_k \rightarrow B_1B_2\cdots B_mC_1C_2\cdots C_k$$

for any set of attributes $C_1, C_2, \ldots, C_k$.

3. *Transitivity.* If

$$A_1A_2\cdots A_n \rightarrow B_1B_2\cdots B_m \text{ and } B_1B_2\cdots B_m \rightarrow C_1C_2\cdots C_k$$

then $A_1A_2\cdots A_n \rightarrow C_1C_2\cdots C_k$.

Normal Forms

- Every component of every tuple is an atomic value (1NF)
- 2NF is permits transitive FD's in a relation, but forbids a nontrivial FD with a left side that is a proper subset of a key.
- If whenever $A_1A_2...A_n$->B is a nontrivial FD, either $\{A_1A_2...A_n\}$ is superkey, or B is a member of some key (3NF)
- If whenever there is a nontrivial FD $A_1A_2...A_n$->B, it is case that $\{A_1A_2...A_n\}$ is a superkey (BCNF)
- If whenever $A_1A_2...A_n$->->$B_1B_2...B_m$ is a nontrivial MVD $A_1A_2...A_n$->B , $\{A_1A_2...A_n\}$ is a superkey (4NF)