COP 4710: Database Systems
Spring 2004

-Day 13 – February 18, 2004 –
Introduction to Normalization – Part 4

Instructor : Mark Llewellyn
markl@cs.ucf.edu
CC1 211, 823-2790
http://www.cs.ucf.edu/courses/cop4710/spr2004

School of Electrical Engineering and Computer Science
University of Central Florida

Multi-valued Dependencies

- Basically, a multi-valued dependency is an assertion that two attributes or sets of attributes are independent of one another.
- This is a generalization of the notion of a functional dependency, in the sense that every fd implies a corresponding multi-valued dependency.
- However, there are certain situations involving independence of attributes that cannot be explained as functional dependencies.
- There are situations in which a relational schema may be in BCNF, yet the relation exhibits a kind of redundancy that is not related to functional dependencies.

Multi-valued Dependencies (cont.)

- The most common source of redundancy in BCNF schemas is an attempt to put two or more M:M relationships in a single relation.

name	city	classes	vehicles
Mark	Orlando	COP 4710	Mercedes E320
Mark	Orlando	COP 4710	Ford F350
Mark	Orlando	COP 3502	Mercedes E320
Mark	Orlando	COP 3502	Ford F350
Kristy	Milan	COP 3502	Mercedes E500
Kristy	Milan	CDA 3103	Mercedes E500
Kristy	Milan	COT 4810	Mercedes E500
Kristy	Milan	COP 3502	Ford F350
Kristy	Milan	CDA 3103	Ford F350
Kristy	Milan	COT 4810	Ford F350

Multi-valued Dependencies (cont.)

- Focusing on the relation on the previous page, notice that there is no reason to associate a given class with a given vehicle and not another vehicle.
- To express the fact that classes and vehicles are independent properties of a person, we have each class appear with each class.
- Clearly, there is redundancy in this relation, but this relation does not violate BCNF. In fact there are no non-trivial functional dependencies at all in this schema.
- We know from our earlier discussions of normal forms based on functional dependencies that redundancies were removed, yet here is a schema in BCNF that clearly contains redundant information.

Multi-valued Dependencies (cont.)

- For example, in this relation, attribute city is not functionally determined by any of the other three attributes.
- Thus the fd: name class vehicle $\rightarrow$ city does not hold for this schema because we could have two persons with the same name, enrolled in the same class, and drive the same type of vehicle.
- You should verify that none of the four attributes in functionally determined by the other three. Which means that there are no non-trivial functional dependencies that hold on this relation schema.
- Thus, all four attributes form the only key and this means that the relation is in BCNF, yet clearly is redundant.

Multi-valued Dependencies (cont.)

- A multi-valued dependency (mvd) is a statement about some relation R that when you fix the values for one set of attributes, then the values in certain other attributes are independent of the values of all the other attributes in the relation.
- More precisely, we have the mvd

$$A_1A_2...A_n \;?\; B_1B_2...B_m$$

holds for a relation R if when we restrict ourselves to the tuples of R that have particular values for each of the attributes among the A's, then the set of values we find among the B's is independent of the set of values we find among the attributes of R that are **not** among the A's or B's.

Multi-valued Dependencies (cont.)

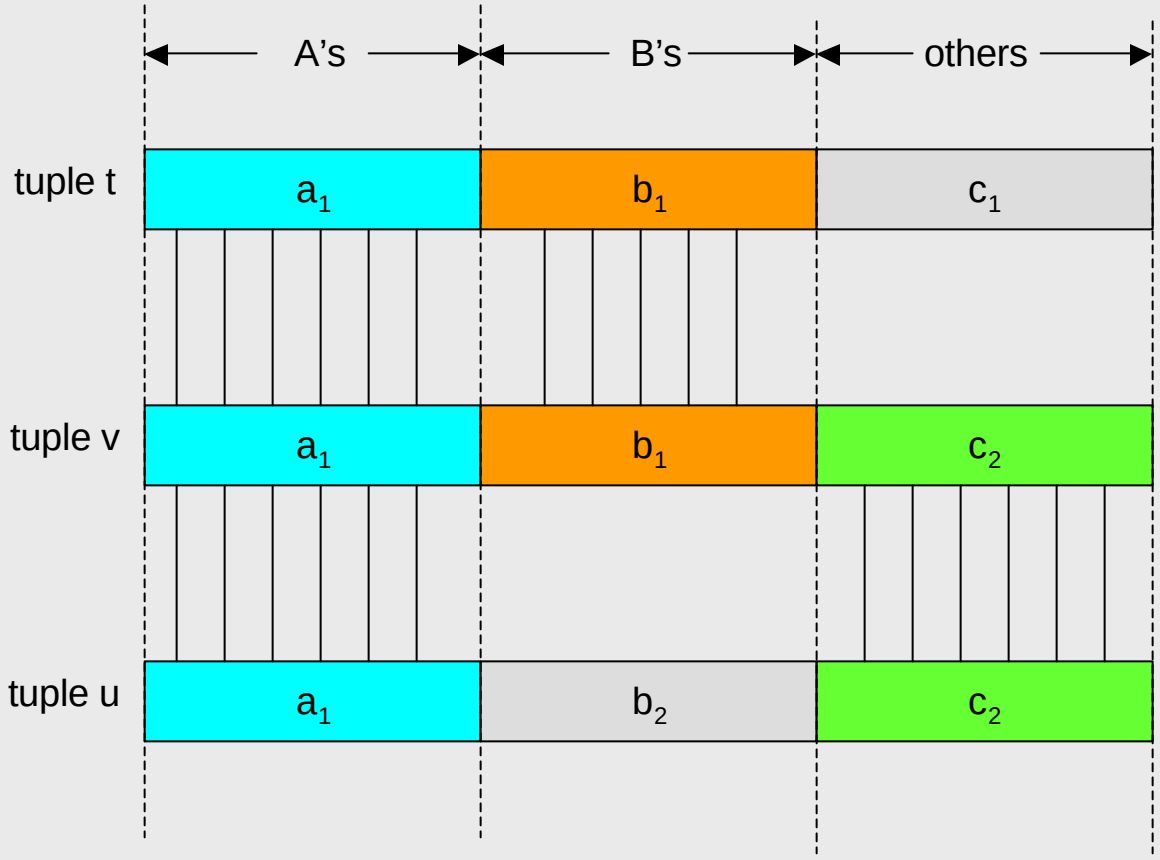
- Even more precisely, a mvd holds if:

  For each pair of tuples t and u of relation R that agree on all the A's, we can find in R some tuple v that agrees:

  1. With both t and u on the A's
  2. With t on the B's
  3. With u on all attributes of R that are not among the A's or B's.

  – Note that we can use this rule with t and u interchanged, to infer the existence of a fourth tuple w that agrees with u on the B's and with t on the other attributes. As a consequence, for any fixed values of the A's, the associated values of the B's and the other attributes appear in all possible combinations in different tuples.

Relationship of Tuple *v* to Tuple *t* When mvd Exists



A multi-valued dependency guarantees that tuple *v* exists

Multi-valued Dependencies (cont.)

- In general, we can assume that the A's and B's (left side and right side) of a mvd are disjoint.
- As with functional dependencies, it is permissible to add some of the A's to the right side.
- Unlike, functional dependencies where a set of attributes on the right side was a short-hand notation for a set of fds with single attribute right sides, with mvds, we must deal only with sets of attributes on the right side as it is not always possible to break the right side of mvds into single attributes.

Example: Multi-valued Dependencies

- Consider the following relation instance.

name	street	city	title	year
C. Fisher	123 Maple Street	Hollywood	Star Wars	1977
C. Fisher	5 Locust Lane	Malibu	Star Wars	1977
C. Fisher	123 Maple Street	Hollywood	Empire Strikes Back	1980
C. Fisher	5 Locust Lane	Malibu	Empire Strikes Back	1980
C. Fisher	123 Maple Street	Hollywood	Return of the Jedi	1983
C. Fisher	5 Locust Lane	Malibu	Return of the Jedi	1983

- The mvd name ? street city holds on this relation.
  - That is, for each star's name, the set of addresses appears in conjunction with each of the star's movies.

Example: Multi-valued Dependencies (cont.)

- For an example of how the formal definition of this mvd applies, consider the first and fourth tuples from the previous relation instance.

name	street	city	title	year
C. Fisher	123 Maple Street	Hollywood	Star Wars	1977
C. Fisher	5 Locust Lane	Malibu	Empire Strikes Back	1980

- If we let the first tuple be t and the second tuple be u, then the mvd asserts that we must also find in R the tuple that has name C. Fisher, a street and city that agree with the first tuple, and other attributes (title and year) that agree with the second tuple. There is indeed such a tuple (the third tuple in the original instance).

name	street	city	title	year
C. Fisher	123 Maple Street	Hollywood	Empire Strikes Back	1980

Example: Multi-valued Dependencies (cont.)

- Similarly, we could let t be the second tuple below and u be the first tuple below (reversed from the previous page). Then the mvd tells us that there is a tuple of R that agrees with the second tuple in attributes name, street, and city with the first tuple in attributes name, title, and year.

name	street	city	title	year
C. Fisher	123 Maple Street	Hollywood	Star Wars	1977
C. Fisher	5 Locust Lane	Malibu	Empire Strikes Back	1980

- There is indeed such a tuple (the second tuple in the original instance).

name	street	city	title	year
C. Fisher	5 Locust Lane	Malibu	Star Wars	1977

Reasoning about Multi-valued Dependencies

- There are a number of inference rules that deal with mvds that are similar to the inference rules for functional dependencies.

1. Trivial multi-valued dependencies:

   If $A_1A_2...A_n \twoheadrightarrow B_1B_2...B_m$ holds for some relation, then so does $A_1A_2...A_n \twoheadrightarrow C_1C_2...C_k$ where the C's are the B's plus one or more of the A's.

   Conversely, we can also remove attributes from the B's if they are among the A's and infer the mvd $A_1A_2...A_n \twoheadrightarrow D_1D_2...D_r$ if the D's are those B's that are not among the A's.

Reasoning about Multi-valued Dependencies

2. Transitive rule for multi-valued dependencies:

   If $A_1A_2...A_n \twoheadrightarrow B_1B_2...B_m$ and $B_1B_2...B_m \twoheadrightarrow C_1C_2...C_k$ both hold for some relation, then so does $A_1A_2...A_n \twoheadrightarrow C_1C_2...C_k$. However, any C's that are also B's must be deleted from the right side.

- mvds do not obey the additivity/projectivity rules as do functional dependencies.

Reasoning about Multi-valued Dependencies

- Consider the same relation schema as before, where the mvd name ? street city held. If the projectivity (splitting) rule held we would expect that

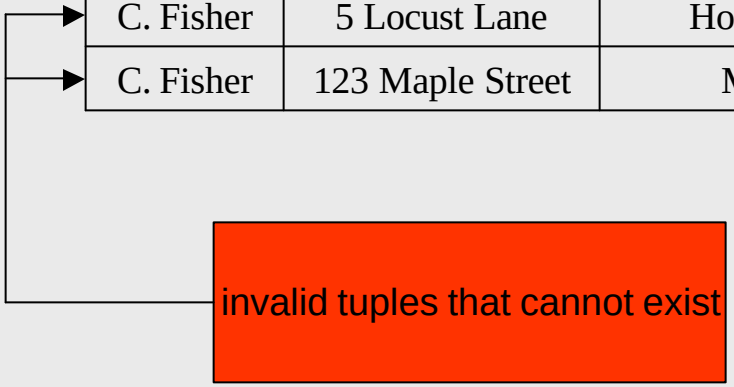
  name ? street would also be true. This mvd states that each star's street addresses are independent of the other attributes (including city). However, that statement is false. The first two tuples in the relation instance indicate that this is not true.

name	street	city	title	year
C. Fisher	123 Maple Street	Hollywood	Star Wars	1977
C. Fisher	5 Locust Lane	Malibu	Star Wars	1977

Reasoning about Multi-valued Dependencies

- This hypothetical mvd name ? street, if it held would allow us to infer that the tuples with the streets interchanged would be in the relation instance. However, these tuples are not there because the home at 5 Locust Lane is in Malibu and not Hollywood.

name	street	city	title	year
C. Fisher	5 Locust Lane	Hollywood	Star Wars	1977
C. Fisher	123 Maple Street	Malibu	Star Wars	1977



Reasoning about Multi-valued Dependencies

- There are however, several new inference rules that apply only to multi-valued dependencies.
- First, every fd is a mvd. That is, if $A_1A_2...A_n \rightarrow B_1B_2...B_m$ holds for some relation, then so does $A_1A_2...A_n \twoheadrightarrow B_1B_2...B_m$ hold.
- Second, complementation has no fd counterpart. The complementation rule states: if $A_1A_2...A_n \twoheadrightarrow B_1B_2...B_m$ is a mvd that holds on some relation R, then R also satisfies $A_1A_2...A_n \twoheadrightarrow C_1C_2...C_k$, where the C's are all attributes of R that are not included in the A's or B's.
  - Thus, if name ? street city holds, the complementation rule states that name ? title year also holds, because street and city are not mentioned in the first mvd. The inferred mvd intuitively means that each star has a set of movies that they appeared in, which are independent of their address.

Fourth Normal Form

- The redundancy that we've seen in the relation instances in this section of the notes are caused by the existence of multi-valued dependencies.
- As we did with functional dependencies, we can use multi-valued dependencies and a different decomposition algorithm to produce a stronger normal form which is based not on functional dependencies but the multi-valued dependencies.
- Fourth Normal Form (4NF) eliminates all non-trivial multi-valued dependencies (as are all fds that violate BCNF). The resulting decomposition scheme has neither the redundancy from fds nor redundancy from mvds.

Fourth Normal Form (cont.)

- A mvd $A_1A_2...A_n \twoheadrightarrow B_1B_2...B_m$ for a relation scheme R is non-trivial if:

  1. None of the B's is among the A's.
  2. Not all of the attributes of R are among the A's and B's.

- 4NF is essentially the BCNF condition, but applied to mvds instead of fds.

- Formally, a relation scheme R is in 4NF if whenever $A_1A_2...A_n \twoheadrightarrow B_1B_2...B_m$ is a non-trivial mvd, $\{A_1A_2...A_n\}$ is a superkey of R.

Fourth Normal Form (cont.)

- The example relation scheme that we have been dealing with is not in 4NF because name ? street city is a non-trivial mvd, yet name by itself is not a superkey. In fact, for this relation the only key is all the attributes.
- 4NF is truly a generalization of BCNF. Since every fd is a mvd, every BCNF violation is also a 4NF violation. In other words, every relation scheme that is in 4NF is therefore in BCNF.
- However, there are some relation that are in BCNF but not in 4NF. The relation instance we have been using in this section of notes is a case in point. It is clearly in BCNF, yet as we just illustrated, it is not in 4NF.

Decomposition into Fourth Normal Form

- The 4NF decomposition algorithm is analogous to the 3NF and BCNF decomposition algorithm:

- Find a 4NF violation, say $A_1A_2...A_n \twoheadrightarrow B_1B_2...B_m$ where $\{A_1A_2...A_n\}$ is not a superkey. Note that this mvd could be a true mvd or it could be derived from the corresponding fd $A_1A_2...A_n \rightarrow B_1B_2...B_m$, since every fd is an mvd. Then break the schema for R into two schemas where: (1) the first schema contains all the A's and B's and the second schema contains the A's and all the attributes of R that are not among the A's or B's.

Decomposition into Fourth Normal Form (cont.)

- Using our previous example relation that we now know is not in 4NF, let's decompose into a relation schema that is in 4NF.

- We know that name ? street city is a 4NF violation. The original schema R (5 attributes) will be replaced by one schema that contains only the three attributes from the mvd above, and a second schema that consists of the left side of the above mvd plus the attributes that do not appear in this mvd, which are the attributes title, and year.

  R1 = {name, street, city}

  R2 = {name, title, year}

Decomposition into Fourth Normal Form (cont.)

R1 = {name, street, city} R2 = {name, title, year}

- In each of these schema there are no non-trivial mvds or fds, so they are both in 4NF. Notice that in the relation scheme R1, the mvd name ? street city is now trivial since it involves every attribute. Likewise, in R2, the mvd name ? title year is also trivial.

Summary of Normal Forms

Property	3NF	BCNF	4NF
Eliminates redundancy due to functional dependencies	most	yes	yes
Eliminates redundancy due to multi-valued dependencies	no	no	yes
Preserves functional dependencies	yes	maybe	maybe
Preserves multi-valued dependencies	maybe	maybe	maybe
Has the lossless join property	yes	yes	yes