# CS145 Lecture Notes #14
# Lossless Decomposition, 3NF, 4NF

## Lossless Decomposition

Recall that we learned how to "normalize" relations (i.e., put them in BCNF) by decomposing their schemas into two or more sets of attributes
Example: `Enroll(student, class, TA)`
- In any given class, each student is assigned to exactly one TA

- One TA can assist only one class

Recall that a relation $R$ is in *BCNF* if for every nontrivial FD $X \to Y$ in $R$, $X$ is a superkey
- $X \to Y$ is a *BCNF violation* if it is nontrivial and $X$ does not contain any key of $R$
- Based on a BCNF violation $X \to Y$, decompose $R$ into two relations:
  - One with $X \cup Y$ as its attributes
    (i.e., everything in the FD)
  - One with $X \cup (attrs(R) - X - Y)$ as its attributes
    (i.e., left side of FD plus everything not in the FD)

Example: turn `Enroll` into BCNF
- BCNF violation:
- Decomposed relations:

What does this decomposition "work"? Why can't we just tear sets of attributes apart as we like?
- ⇝ The decomposed relations need to represent the same information as the original
- ⇝ We must be able to reconstruct the original from the decomposed relations

Formally: suppose $R$ is decomposed into $S$ and $T$
- $attrs(R) = attrs(S) \cup attrs(T)$
- $S = \pi_{attrs(S)}(R), T = \pi_{attrs(T)}(R)$
- The decomposition is *lossless* if we can guarantee $R = S \bowtie T$

Example of lossless decomposition: BCNF decomposition for `Enroll`

Example of lossy decomposition: what if a TA may assist multiple classes?

- $\leadsto$ The join returns more tuples than the original relation
- $\leadsto$ "Lossy" refers not to the loss of tuples, but to the loss of information (the ability to distinguish different states of the original relation)
- $\leadsto$ FD is what makes a decomposition lossless!

*Theorem:* Suppose we decompose a relation with schema $XYZ$ into $XY$ and $XZ$ and project the relation for $XYZ$ onto $XY$ and $XZ$; then, $XY \bowtie XZ$ is *guaranteed* to reconstruct $XYZ$ if either $X \to Y$ or $X \to Z$ holds

*Proof:*

- Anything we project always comes back in the join
  - Sure; and it does not depend on FD's

- Anything that comes back in the join was in the original relation

On the other hand, if neither $X \to Y$ nor $X \to Z$ holds, then we could find a counterexample where the join returns too much:

## 3NF

One FD structure causes problems:

- If we decompose, we cannot check all FD's in decomposed relations
- If we don't decompose, we violate BCNF

Example: `Enroll(student, class, TA)`

- FD's: `student class` $\rightarrow$ `TA` and `TA` $\rightarrow$ `class`
- BCNF decomposition:
  `Assist(TA, class)` and `Assign(TA, student)`
- Cannot check `student class` $\rightarrow$ `TA` without joining decomposed relations back together

"Elegant" solution: define the problem away!

$R$ is in *Third Normal Form (3NF)* if for every nontrivial FD $X \rightarrow A$, either

- $X$ is superkey of $R$, or
- $A$ is a member of at least one key of $R$

Tradeoff:

- We can check all FD's in the decomposed relation
- But now we might have redundancy due to FD's

Example: `Enroll(student, class, TA)` is in 3NF, but not in BCNF

### Lossless & Dependency-Preserving Decomposition into 3NF

The "obvious" approach of doing a BCNF decomposition, but stopping when a relation schema is in 3NF, does not always work—it might still allow some FD's to get lost

$\leadsto$ 3NF decomposition algorithm:

- Given: a relation $R$ and a basis $\mathcal{F}$ for the FD's that hold in $R$

1. Find $\mathcal{F}_c$, a *canonical cover* for $\mathcal{F}$
2. For each FD $X \rightarrow Y$ in $\mathcal{F}_c$, create a relation with schema $XY$
3. Eliminate a relation if its schema is a subset of another
4. If none of the schemas created so far contains a key of $R$, add a relation schema containing a key of $R$

A *canonical cover* $\mathcal{F}_c$ for $\mathcal{F}$ is a set of FD's with the following 4 properties:

(a) $\mathcal{F}_c$ is equivalent to $\mathcal{F}$
  - $\mathcal{F}$ logically implies all FD's in $\mathcal{F}_c$ and vice versa

(b) No FD in $\mathcal{F}_c$ is redundant, i.e., $\mathcal{F}_c$ is a minimal basis
  - If we remove any FD from $\mathcal{F}_c$, the set of remaining FD's will no longer be equivalent to $\mathcal{F}_c$

(c) No FD in $\mathcal{F}_c$ contains redundant attributes

– For any FD $X \to Y$ in $\mathcal{F}_c$, if we remove an attribute from either $X$ or $Y$, the result FD together with the other FD's in $\mathcal{F}_c$ will no longer be equivalent to $\mathcal{F}_c$

(d) No two FD's in $\mathcal{F}_c$ have same left sides

– $\mathcal{F}_c$ cannot contain $X \to Y$ and $X \to Z$ as separate FD's; they should have been combined into $X \to YZ$

Example: $R(A, B, C, D, E)$

$\mathcal{F} = \{A \to BC, B \to C, A \to B, AB \to C, BD \to A\}$

1. Find a canonical cover $\mathcal{F}_c$

   Repeat until no change:

   · Remove redundant FD's

   · Remove redundant attributes from FD's

   · Combine FD's with common left sides

2. Create a relation for each FD in $\mathcal{F}_c$

3. Eliminate a relation if its schema is a subset of another

4. If no schema contains a key of $R$, add one containing a key of $R$

   First, what are the keys of $R$?

⇝ Final answer:

# 4NF

BCNF does not eliminate all redundancies

Example: `Student(SID, class, club)`

– No nontrivial FD's; `Student` is in BCNF

– Suppose your classes have nothing to do with the clubs you join

⇝ Still contains tons of redundancies!

⇝ Often comes up when converting from an ODL design

**Multivalued Dependencies**

The *multivalued dependency (MVD)* $X \twoheadrightarrow Y$ holds in a relation $R$ if whenever we have two tuples of $R$ that agree on all attributes of $X$, then we can swap their $Y$ components and get two new tuples that are also in $R$
Example: in `Student`, `SID` $\twoheadrightarrow$ `class`

$\leadsto$ This property must hold for *all* pairs of tuples that agree on `SID`, not just one pair

$\leadsto$ Intuitively, given `SID`, `class` and `club` are "independent"

Trivial and nontrivial MVD's:

- *Trivial:* $X \twoheadrightarrow Y$ where $Y$ is a subset of $X$ or $X \cup Y$ contains all attributes of the relation
- *Nontrivial:* $X \twoheadrightarrow Y$ where $Y$ is not a subset of $X$ and $X \cup Y$ does not contain all attributes of the relation

MVD rules:

- *FD is MVD:* If $X \rightarrow Y$ holds in $R$, then $X \twoheadrightarrow Y$ also holds in $R$
  - Because if $X \rightarrow Y$, then swapping $Y$'s between tuples that agree on $X$ will not create any new tuples
- *Complementation:* If $X \twoheadrightarrow Y$ in $R$, then $X \twoheadrightarrow attrs(R) - X - Y$ also holds in $R$
  - Intuitively, if $X$ is given, $Y$ and the rest of the attributes in $R$ are "independent"

Sound and complete set of axioms for inferring FD's and MVD's (for your reference only):

- FD reflexivity: if $Y \subseteq X$, then $X \rightarrow Y$
- FD augmentation: if $X \rightarrow Y$, then $XZ \rightarrow YZ$
- FD transitivity: if $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$
- MVD complementation: if $X \twoheadrightarrow Y$ in $R$, then $X \twoheadrightarrow attrs(R)-X-Y$
- MVD augmentation: if $X \twoheadrightarrow Y$ and $V \subseteq W$, then $XW \twoheadrightarrow YV$
- MVD transitivity: if $X \twoheadrightarrow Y$ and $Y \twoheadrightarrow Z$, then $X \twoheadrightarrow (Z - Y)$
- Replication: if $X \rightarrow Y$, then $X \twoheadrightarrow Y$
- Coalescence: if $X \twoheadrightarrow Y$ and $Z \subseteq Y$ and there is some $W$ disjoint from $Y$ such that $W \rightarrow Z$, then $X \rightarrow Z$

**Lossless Decomposition into 4NF**

A relation $R$ is in *Forth Normal Form (4NF)* if for every nontrivial MVD $X \twoheadrightarrow Y$, $X$ is a superkey
$\leadsto$ Since every FD is also an MVD, 4NF implies BCNF

4NF decomposition algorithm is almost identical to BCNF decomposition algorithm: repeatedly decompose using any 4NF violation you can find

*Theorem:* Suppose we decompose relation with schema $XYZ$ into $XY$ and $XZ$ and project the relation for $XYZ$ onto $XY$ and $XZ$; then, $XY \bowtie XZ$ is *guaranteed* to reconstruct $XYZ$ if either $X \twoheadrightarrow Y$ or $X \twoheadrightarrow Z$ holds

Example: turn `Student` into 4NF
- FD's and MVD's:
- Keys:
- 4NF violations:
- Decomposed relations:

# Summary

4NF is more stringent than BCNF, which is more stringent than 3NF

| | Preserve FD's? | Guarantee no redundancy due to FD's? | Guarantee no redundancy due to MVD's? |
|---|---|---|---|
| 3NF | | | |
| BCNF | | | |
| 4NF | | | |

Of course, all decompositions should be lossless!