

Jasmine Omeke

CSC 453 Database Technologies 701/710 Assignment 4 (9/28)

Due 11:59:00pm, Monday 10/9.

1. Find a minimal cover for the following set F of functional dependencies.

$$WY \rightarrow Z \quad X \rightarrow YZ \quad Y \rightarrow WX \quad Z \rightarrow W$$

(Note that when attribute names are single letters, I will use shorthand of the form $WX \rightarrow YZ$ for the functional dependency $W,X \rightarrow Y,Z$.)

Final Answer: $\{ X \rightarrow Y; X \rightarrow Z; Y \rightarrow X; Z \rightarrow W \}$

a) Simplify right sides of dependencies

~~simplify~~ $W,Y \rightarrow Z$ $W \rightarrow Z$ (pseudotransitivity)
 $X \rightarrow Y$ $X \rightarrow Z$ (transitivity)
 $X \rightarrow Z$ $Y \rightarrow W$ (transitivity)
 $Y \rightarrow W$ $Y \rightarrow X$
 $Y \rightarrow X$ $Z \rightarrow W$

b) Check each attrn but to see if you can remove or simplify the left side:
 $W \rightarrow Z?$ $W^+ = \{W\}$
 $Y \rightarrow Z?$ let's look at closures of both
yes $Y^+ = \{Y, X, W, Z\}$

c) re-write what you have w singleton attributes on both sides.
Check for redundant dependencies

~~$X \rightarrow Z$~~
 $X \rightarrow Y$ keep
 $X \rightarrow Z$ keep
 ~~$Y \rightarrow W$~~
 $Y \rightarrow X$ keep
 $Z \rightarrow W$

pretending functional dependences above don't exist for each one (over respective one)

$Y^+ = \{Y, W, X, Z\}$ can remove $Y \rightarrow Z$
 $X^+ = \{X, Z, W\}$ keep $X \rightarrow Y$
doesn't include Y
 $X^+ = \{X, Y, W\}$ no Z
 $Y^+ = \{Y, X, Z, W\}$ can remove $Y \rightarrow W$
 $Y^+ = \{Y\}$
 $Z^+ = \{Z\}$

d) Go through + ask, 'if I remove each functional dependency, can I derive it from what remains?'

* FINAL ANSWER *

$\{X \rightarrow Y; X \rightarrow Z; Y \rightarrow X; Z \rightarrow W\}$

e) final check

$Y \rightarrow X \quad X \rightarrow Z \quad X \rightarrow Y \quad Y \rightarrow X \quad Z \rightarrow W$
 $Z \rightarrow W \quad X \rightarrow Z \quad X \rightarrow Z \quad X \rightarrow W \quad Z \rightarrow W$
 $W, Y \rightarrow Z \quad X \rightarrow YZ \quad Y \rightarrow W, X \quad Z \rightarrow W$

2. For the universal relation $R(a,b,c,d)$, consider the decomposition D consisting of $R_1(a,b,c)$ and $R_2(b,d)$, and the set F of functional dependencies $\{ a \rightarrow b, d ; b \rightarrow d ; c \rightarrow d, b \}$.

a. Compute the projection of F on R_1 .

a) Projection of F onto R_1

$$\boxed{a \rightarrow b \\ \cancel{b \rightarrow \text{nothing}} \\ c \rightarrow b}$$

$a \rightarrow b$

$c \rightarrow b$

b. Compute the projection of F on R_2 .

b) Projection of F onto R_2

$$\boxed{b \rightarrow d \\ \cancel{d \rightarrow \text{nothing}}}$$

$b \rightarrow d$

c. Does the decomposition D preserve the set of dependencies F ? Give a detailed explanation why or why not. (That is, don't just repeat back the

definition of the dependency preservation property, but rather show why the decomposition D either has or does not have this property.)

Can everything from F be derived from the projections? Yes

$$\{A\}^+ = \{A, \underline{B}, \underline{D}\}$$

$$\{B\}^+ = \{B, \underline{D}\}$$

$$\{C\}^+ = \{C, \underline{B}, \underline{D}\}$$

We are still able to reconstruct the original dependencies in F . Our goal w/ projections are to maintain integrity of sets & have set equivalence.

We don't want to lose dependencies or add newly made values while trying to derive F from the relations

The dependencies in F are maintained because everything from one set can be derived from the other. For instance, one can get $\{A\}^+$ being $\{A, B, D\}$ because $a \rightarrow b$ in R^1 and $b \rightarrow d$ in R^2 . Since everything in R^1 and R^2 was derived from F , our main aim is to see that F can be derived from both relations without losing functional dependencies or gaining any new ones that were not in the original set.

3. Perform the test for the nonadditive join property for the universal relation $R(A_1, A_2, A_3, A_4)$, and the decomposition D and set of functional dependencies F given below:

$$D = \{R_1(A_1, A_4), R_2(A_2, A_3), R_3(A_2, A_4)\}$$

$$F = \{ A_4 \rightarrow A_1, A_2 \rightarrow A_4, A_1 \rightarrow A_2 \}$$

a. Show the initial state of the matrix S , and also its state at the end of each iteration of the loop (i.e., after each pass through the set of functional dependencies).

Initial state			
	A_1	A_2	A_3
R_1	a		
R_2		a	a
R_3		a	a

Loop 1
adding distinguished variables } Rules

① 2+ rows w/ distinguished variables on the LHS of functional dependency

② 1+ row w/ distinguished variable on RHS of functional dependency

③ 1+ row w/ non-distinguished variable for RHS of FD

$\wedge \rightarrow A$

Loop 1

	A_1	A_2	A_3	A_4
R_1	a			a
R_2		a	a	
R_3	a	a		a

Loop 1

$A_1 \ A_2 \ A_3 \ A_4$

R_1

a

a

R_2

a a

a

R_3

a

a

a

Loop 1
adding dis
variables

$A_4 \rightarrow A_1$,
 $A_2 \rightarrow A_4$

Loop 1

$A_1 \ A_2 \ A_3 \ A_4$

R_1

a

a

a

R_2

a a

a

R_3

a

a

a

Loop 1
adding dis
variables

$A_4 \rightarrow A_1$,
 $A_2 \rightarrow A_4$
 $A_1 \rightarrow A_2$

Loop 2

$A_1 \ A_2 \ A_3 \ A_4$

	A_1	A_2	A_3	A_4
R_1	a	a		a
R_2	a	a	a	a
R_3	a	a		a

$A_4 \rightarrow A_1$

$A_2 \rightarrow A_4$

$A_1 \rightarrow A_2$

1 row of
distinguished
variables
means this is
lossless/non-additive

- b. Does the decomposition D have the nonadditive join property? Explain why or why not in terms of the final state of the matrix S .

It does. after going through the matrix according to the rules for adding distinguished variables to this matrix (written above), I was able to continually adding distinguished variables until I had an entire row with all distinguish variables. If I had kept going through the loop without getting this result, then the decomposition wouldn't have had the nonadditive property.

4. Consider the universal relation

EMPLOYEE(ID, First, Last, Team, Dept, Salary) with the following set F of functional dependencies:

ID \rightarrow First

ID \rightarrow Last

First, Last \rightarrow ID

Last \rightarrow Team

ID \rightarrow Dept

ID \rightarrow Salary

Salary \rightarrow Dept

- Identify all candidate keys of EMPLOYEE.

ID

First, Last

Homework #4

Employee(ID, First, Last, Team, Dept, Salary)

① Identify candidate keys (superkey, not reducible/no redundancy)

$ID \rightarrow First$

$ID \rightarrow Last$

$First, Last \rightarrow ID$

$Last \rightarrow Team$

$ID \rightarrow Dept$

$ID \rightarrow Salary$

$Salary \rightarrow Dept$

$(ID)^+ = \{First, Last, Dept, Salary, Team, Dept\}$

$(First, Last)^+ = \{ID, Dept, Salary, Team\}$

Final answer

- ID
- First, Last

b. Construct a decomposition of EMPLOYEE into relations in 3NF that preserves dependencies. Be sure that you show that dependencies are preserved by either 1) showing it directly or 2) using an algorithm that is known to guarantee the dependency preservation property. (Hint: If you remove $ID \rightarrow Dept$ from F, the resulting set is a minimal cover of F.)

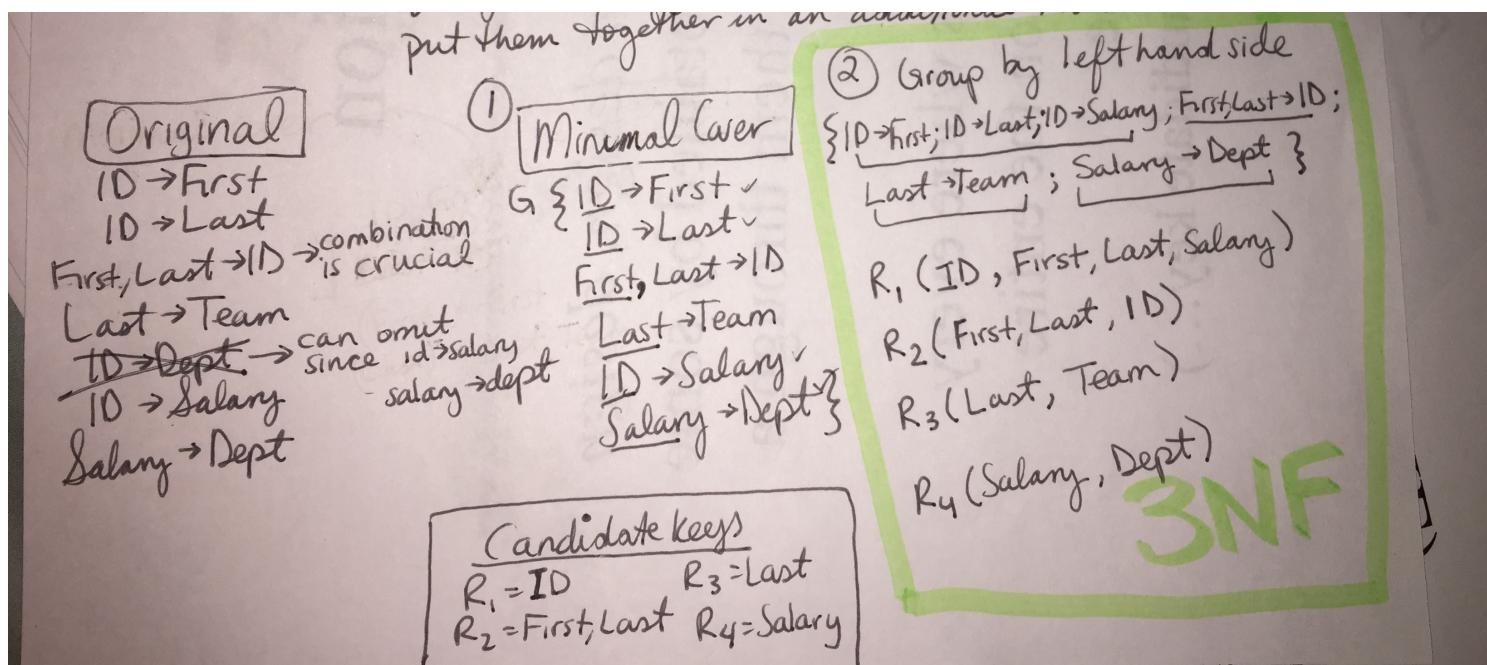
3NF =

R1 (ID, First, Last, Salary)

R2 (First, Last, ID)

R3 (Last, Team)

R4 (Salary, Dept)



The dependencies are preserved when looking at the projection of each relation (below)

projection onto R₁

$ID \rightarrow First$
 $ID \rightarrow Last$
 $ID \rightarrow Salary$
 $First, Last \rightarrow ID$

Going back through relations and checking dependencies

projection onto R₂

$First, Last \rightarrow ID$
 $ID \rightarrow First$
 $ID \rightarrow Last$

$ID \rightarrow First? Y$
 $ID \rightarrow First, Last Y$
 $ID \rightarrow Last Y$
 $ID \rightarrow Salary Y$
 $ID \rightarrow Team? (yes, by virtue of last)$
 $ID \rightarrow Dept? (yes, by virtue of salary)$

projection onto R₃

$Last \rightarrow Team$

projection onto R₄

$Salary \rightarrow Dept$

Testing for nonadditive/lossless property:

The union of all relations brings back an empty set. The nonadditive/lossless property is not upheld.

non-additive join test

$$R_1 \cap R_2 \cap R_3 \cap R_4 = \emptyset$$

don't all overlap ^{no}
non-additive join property

- c. Are all of the relations in your decomposition in BCNF? Either explain why they are, or identify one that is not and explain why it is not. (Note that for a relation to be in BCNF, the determinants of all functional dependencies in the relation must be superkeys *of that relation* – not superkeys of the original universal relation.)

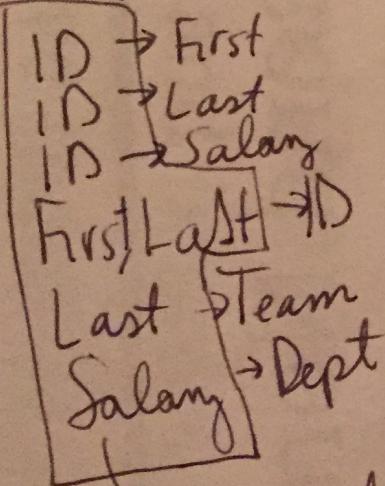
YES. Each determinant of the functional dependencies is a super key of a relation (not the universal relation).

BCNF

*in the functional
dependencies*

all determinants[^] are superkeys
of that relation

Dependences
in
Minimal Cover



all the determinants of
the functional dependences
are superkeys of their
respective relations

- R₁(**[ID]**, First, Last, Salary)
- R₂(**[First, Last]**, ID)
- R₃(**[Last]**, Team)
- R₄(**[Salary]**, Dept)