CSCC43 Tutorial #9

Functional Dependencies and Normal Forms

Andrew Leung

Part 1 – Properties of Decomposition

1. Suppose we have a relation with attributes cdf, name, grade. Here is an instance of that relation:

cdf	name	grade
g3tout	Amy	91
g4foobar	David	78
c0zhang	David	85

Part 1 – Properties of Decomposition

1. a) Suppose we were to decompose this into two new relations:

R1(cdf, name) and R2(name, grade). Project the data onto those two new relations.

R1

cdf	name
g3tout	Amy
g4foobar	David
c0zhang	David

R2

name	grade
Amy	91
David	78
David	85

Part 1 – Properties of Decomposition

1. b) Now compute R1⋈ R2 to rebuild the original table.

cdf	name	grade
g3tout	Amy	91
g4foobar	David	78
c0zhang	David	85
g4foobar	David	85
C0zhang	David	78

Part 1 – Properties of Decomposition

1. c) What was lost?

The rebuilt table has 5 rows. We have lost the information that the grade of 78 is for g4foobar and the grade of 85 is for c0zhang.

Aside

BCNF: LHS of every non-trivial FD for R must be a superkey

To convert R into BCNF:

1. Let a BCNF violation be X → Y, compute X+
2. Let R1 = X+, R2 = X U (R – X+)
3. Compute the FDs of R1 and R2 and recursively repeat until there are no more BCNF violations

This algorithm only ensures a lossless join decomposition but not dependency preserving. To satisfy both, we relax the BCNF condition to get 3NF.

Aside

3NF: For every non-trivial FD of R, either the LHS is a superkey OR the RHS contains only prime attributes (a prime attribute is one that is contained in some candidate key)

To convert R into 3NF:

1. Find a minimal basis for the FDs, G
2. For each FD X → A in G, construct XA as a schema to use in one of the relations in the decomposition
3. If none of these relations are a superkey of R, add a relation whose schema is a key for R

Part 1 – Properties of Decomposition

2. Suppose we have a relation with attributes (movie, theatre, city) and FDs {theatre → city; movie, city → theatre}. The FD theatre → city violates BCNF, and applying the BCNF decomposition algorithm, we get two new relations:

- R1(theatre, city) with one FD: theatre → city
- R2(theatre, movie) with no FDs

a) Create small instances of R1 and R2 that satisfy their own FDs, but when natural-joined together, violate one of the original FDs.

R1

theatre	city
Kingsway Theatre	Toronto
Varsity Cinema	Toronto

R2

theatre	movie
Kingsway Theatre	The Matrix
Varsity Cinema	The Matrix

R1⋈ R2

theatre	city	movie
Kingsway Theatre	Toronto	The Matrix
Varsity Cinema	Toronto	The Matrix

Part 1 – Properties of Decomposition

2. b) In the original relation, with attributes movie, theatre, city, does the functional dependency theatre → city violate 3NF?

No, city is part of the key (city, movie).

2. c) In the original relation, with attributes movie, theatre, city, does the functional dependency theatre → city violate BCNF?

Yes, because theatre is not a key. It does not functionally determine movie.

Part 2 - BCNF

1. a) Suppose we have a relation Students(SID, email, course, term, prof), and that these FDs hold:

{SID → email; course, term → prof; SID, course → grade}

Is this relation in BCNF?

SID+ = {SID; email} which is not all the attributes. No.

Part 2 - BCNF

1. b) Suppose we have a relation Customers(name, DOB, address, favouriteCar, manufacturer) and these FDs hold:

{name → DOB, favouriteCar; favouriteCar → manufacturer}

Is this relation in BCNF?

Calculate the closure of name to see that it is not all the attributes.

name+ = { name, DOB, favouriteCar, manufacturer}.

It does not include address.

No.

Part 2 - BCNF

1. c) Suppose we have a relation Parts(part, manufacturer, seller, price) and these FDs hold:

{part → manufacturer; part, seller → price}.

Is this relation in BCNF?

part+ = {part, manufacturer} which does not include seller or price. No.

Part 2 - BCNF

1. d) Suppose we have a relation R(A, B, C, D, E) and these FDs hold:

{B → AC; CB → E; A → D}.

Is this relation in BCNF?

A+ = {A, D} which is not the whole set of attributes of R. No.

Part 2 - BCNF

2. Consider again the relation:

Parts(part, manufacturer, seller, price) with these FDs:

{part → manufacturer; part, seller → price}.

a) Keeping in mind the FDs, make an instance of this relation that has redundant information.

Solution: Here is one of an infinite number of possibilities

part	manufacturer	Seller	price
p1	man1	seller1	45.99
p1	man1	seller2	30.49

Part 2 - BCNF

2. b) If we applied the decomposition step from BCNF decomposition, what attributes would each of the new relations have?

R1(part; manufacturer) and R2(part; seller; price)

2. c) Project the FDs onto each of the new relations

R projected onto R1: T = {part $\rightarrow$ manufacturer}

R projected onto R2: T = {part, seller $\rightarrow$ price}

Part 2 - BCNF

2. d) Put the same data as in part (a) into your new schema. Is there any redundancy?

part	manufacturer
p1	man1

part	Seller	price
p1	seller1	45.99
p1	seller2	30.49

2. e) Is it possible to create redundancy with this new schema?

No.

Part 3 – Decomposing into BCNF

1. Suppose you are given a relation R with four attributes ABCD. For each of the following sets of FDs, assuming those are the only dependencies that hold for R, do the following:

i. Identify the candidate key(s) for R.

ii. Identify the best normal form that R satisfies (1NF, 2NF, 3NF, or BCNF).

iii. If R is not in BCNF, decompose it into a set of BCNF relations that preserve the dependencies.

a) $C \rightarrow D$, $C \rightarrow A$, $B \rightarrow C$

i. Candidate keys: B

ii. R is in 2NF but not 3NF

iii. $C \rightarrow D$ and $C \rightarrow A$ both cause violations of BCNF. One way to obtain a lossless join preserving decomposition is to decompose R into ACD and BC

Part 3 – Decomposing into BCNF

1. Suppose you are given a relation R with four attributes ABCD. For each of the following sets of FDs, assuming those are the only dependencies that hold for R, do the following:

i. Identify the candidate key(s) for R.

ii. Identify the best normal form that R satisfies (1NF, 2NF, 3NF, or BCNF).

iii. If R is not in BCNF, decompose it into a set of BCNF relations that preserve the dependencies.

b) B→ C, D → A

i. Candidate keys: BD

ii. R is in 1NF but not 2NF

iii. Both B → C and D → A cause BCNF violations. One possible decomposition: AD, BC, BD is BCNF and lossless and join-preserving