Database Design: Schema Refinement

Davood Rafiei

Introduction

Problems with SuperRelation shown in class:
- More redundancy, space is wasted
- Insertion anomalies:
  insert a new vendor
  No transactions available yet.
  Value of unspecified columns? nulls/defaults ...

- Deletion anomalies:
  delete a vendor
  delete the row? all transactions of customers will be deleted.
  What if a customer only has transactions with the vendor being deleted?
  - should we delete the customer?
  - should we set some values to null?

- Update anomalies
  update a vendor
  - how many tuples need to be updated?

Normal forms
All, 1NF, 2NF, 3NF, BCNF, 4NF, 5NF

Basic Concepts

Def. functional dependency X → Y holds over relation R if whenever two tuples of R have the same X-value, they must also have the same Y-value.
- reads "X determines Y" or "Y is functionally dependent on X".
- must hold over all instances.
- X can be a set of attributes.
- X → YZ is equivalent to X → Y and X → Z.

Example. Students(sid, name, address)
FDs: {sid → name, sid → address}
Meaning of FDs
Implications for keys?

sid	name	address
100	Joe	100 St.
~~100~~	~~Joe~~	~~100 St.~~

Example. R(A,B,C,D,E)
FDs: {AB → CDE}

AB →

- AB is a super key because AB → ABCDE
- AB is a key because it is minimal

Example. Students(sid, name, address)
Suppose sid is a key. What can we say about FDs?

FD: {sid →name, sid → address}

Question. Given the following FDs
A → B,
B → C
does it imply
A → C?

A real example:
{sid → phone, phone → address} implies sid → address

Given some FDs, we can infer additional FDs.
How? using Armstrong axioms: Reflexivity, Augmentation, Transitivity
(check them in the textbook)

Completeness
Soundness

Transitivity
$$\frac{A \to B \quad B \to C}{A \to C}$$

reflexivity
A → A
AB → A
AB → B

augmentation
A → B
AC → BC

Def. Given a set F of FDs, the closure of F, denoted by F^+, is the set of all FDs logically implied by the FDs in F.

Example. F={empno → sin, sin → empno, empno → deptno, deptno → address}

F^+= ... F ∪ {empno → address, sin → deptno, sin → sin, sin → address, ... }

F^+ could be large (how large?), and we want to avoid computing it.

Exercise. Give a relation R and a set of FDs F such that $|F^+|$ is exponential in the size of F.

Question. Is (X → Y) in F^+ ??
Answer. compute X^+, the closure of X under F
(the set of attributes determined by X)
(X→Y) is in F^+ if Y in X^+.

Algorithm 1. compute the closure of X under F
$X^+ = X$
while there exists (U → V) in F
such that $U \subseteq X^+$ and not $V \subseteq X^+$ do
$X^+ = X^+ \cup V$

Example. R(A,B,C,D), F={A→B, BC→D}
$A^+ = AB$ $A^+ = AB$
$(AC)^+ = ABCD$ AC is a key

$(AC)^+ = ACBD$ ✓ $(AC)^+ = ACBD$

prove that AC is a key.

AC is unique

AC is minimal

$A^+ = AB \not\supseteq ABCD$

$C^+ = C \not\supseteq ABCD$

Boyce-Codd Normal Form (BCNF)

Def. A relation R is in BCNF if for every non-trivial FD X→Y on R ($X, Y \subseteq R$), X is a super key of R.

$A \to A$
$AB \to A$

X → Y is a trivial FD iff $Y \subseteq X$.

A Schema is in BCNF if all its relations are in BCNF.

Example. R(A,B,C,D,E,F), FDs={A→BC, D→EF}
Is the relation in BCNF? NO

$A \to BC \quad A^+ = ABC \not\supseteq ABCDEF$ X
$D \to EF \quad D^+ = DEF$ "

Question. Why does a BCNF violation produce a "bad" relation?
- Let's consider a real example:
  Loans(sid, name, address, isbn, title, author)
  FDs = {sid → name, address,
  isbn → author, title}

sid	name	address	isbn	title	author
100	John	Edm.	i20		
100	John	Edm	i30		
100	John	Edm	i4		

Example. Given the FDs {A→BC, D→AEF},
are the relations R1(A,D,E,F), R2(A,B,C) in BCNF? yes
$D \to AEF$ $A \to BC$

$D^+ = DAEF$
$A^+ = ABC$

What about relations R3(B,C,D) and R4(A,B,D)?
$D \to BC$ ✓ $A \to B$ — $A^+ = ABC \not\supseteq ABD$
$D \to A$

$D^+ = ABCDEF \supseteq BCD$
$D^+ = DAEFBC$

R4 is not in BCNF
R3 is in BCNF

Third Normal Form (3NF)

Def. A relation R is in 3NF if for every non-trivial FD X→Y on R
(X, Y ⊆ R)
- X is a super key of R or
- Y is part of a key (prime).

Example. R(A,B,C), FDs={AB→C, C→B)

- Is R in BCNF? no because C is not a super key. C→B $C^+ = CB \not\supseteq ABC$

- Is R in 3NF? yes

AB→C ✓

C→B
- Is C a superkey? NO
- Is B part of a key? yes

$(AC)^+ = ACB$

$A^+ = A$

$C^+ = CB$

Keys: AB, AC

A<u>B</u> A<u>C</u>

Finding keys of a relation
1 - start with one attribute and add more attributes until it is unique
2 - check for minimality
3 - repeat steps 1-2 until all keys are found (i.e. all options are exhausted)
use some heuristics to prune the search space

Example. R(A,B,C), FD={A→B, B→C}

- Is R in 3NF? NO

A→B ✓ $A^+ = ABC$ ✓

B→C
- Is B a superkey? $B^+ = BC \not\supseteq ABC$
- Is C part of a key? NO

Keys: A

A is the only key ; <u>prove it</u>

Example. Bookings(title, theater, city)

FD : {theater → city, — Is city part of a key? yes

title city → theater} — the left side is a superkey.

Keys: (theater, title), (title,city)

BCNF: no, why?

3NF: yes, why?

Exercise. Given superrelation(account, cname, prov, balance, crlimit, vno, vname, city, amount) with FDs

account → {cname, prov, balance, crlimit},

vno → {vname, city},

{account, vno} → amount}

Is the relation in BCNF? How about 3NF? Why?

A decomposition of superrelation into BCNF:

customers(account, cname, prov, balance, crlimit) account → cname, prov, ...

vendors(vno, vname, city) vno → vname, city

transactions(account, vno, amount) account, vno → amount

Two important properties of a decomposition:

- Lossless join

customers ⋈ vendors ⋈ transactions = superrelation

- dependencies are preserved.

Not all decompositions are lossless-join. See an example.

Check out the textbook for definitions and more details (Sec 7.1 in SKS and 6.6.1 in KBL).

R

name	city
John	Edmonton
Mary	Calgary

decomposed to

R1

name
John
Mary

R2

city
Edmonton
Calgary

a lossy decomp.

$R1 \bowtie R2$

name	city
John	Edmonton
John	Calgary
Mary	Edmonton
Mary	Calgary

Lossless-join decomposition into BCNF:

Algorithm 2.

1. For every FD X → Y that is defined on R(Z) and violates BCNF, decompose R(Z) into R1($X^+ \cap Z$) and R2(($Z-X^+$) U X).
2. Repeat Step 1 until there is no violation.

A discussion of the correctness...

- the algorithm produces a lossless-join decomposition
- the resulting relations are in BCNF

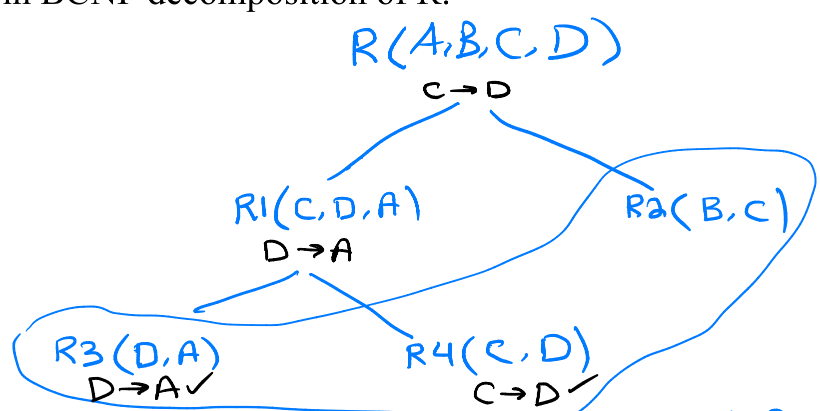
The algorithm is non-deterministic.

Example. Consider relation R(A,B,C,D) and FDs
AB → C, D → A, C → D

$C^+ = CDA$

$D^+ = DA$

BCNF violations:
C → D, D → A
Find a lossless-join BCNF decomposition of R.



A lossless join decomp into BCNF: R3, R4, R2

Exercise. Consider relation R(A,B,C,D) and FDs B → C, B → D

$B^+ = BCD$

BCNF violations?
A BCNF decomposition?
Projection of dependencies on each relation:

- non-trivial FDs X → B where X and B are attributes of the relation
- in principle, must compute the closure of every subset

Consider the projection of the following FDs on R1(D,A), R2(C,D), R3(B,C).
AB → CD, D → A, C → D

D → A C → D

lost AB → C

$R(A, B, C, D)$

$B \to C$

$R1(B, C, D)$ $\qquad$ $R2(A, B)$

$B \to C$

$B \to D$

$B \to C$

$B \to D$

$B^+ = BCD$

AB → C
D → A
C → D

We have lost a dependency!

Consider the decomposition S(B,C,D) and T(A,D) of R(A,B,C,D). Find the projection of FDs on each relation.

S: ? C → D , BD → C

T: ? D → A

$(BD)^+ =$ BDAC

Example. Given relation R(A,B,C) and FDs {AB → C, C → A}, find a dependency-preserving BCNF decomposition.

$C^+ = CA$

R(A,B,C)

C → A

R1(C,A)

C → A

R2(B,C)

lost

AB → C

violation: C → A

decomposition: CA, CB

It is not always possible to find a dependency preserving BCNF decomposition!

Question. given a choice between dependencies that can be preserved, which ones do we really want to preserve?

A → B

B → C

A → C

Moral. use the minimal set of FDs; this is called minimal cover or canonical cover.

Algorithm 3. Finding the minimal cover of F

step 1: convert FDs so that they have only one attribute on the right side

step 2: remove all redundant attributes from the left sides

(remove an attribute if the closure of the leftside without the attribute includes the attribute)

step 3: remove all redundant FDs

~~step 3: remove all redundant FDs~~

See the textbook for more details and examples (Sec 7.4.3 in SKS and 6.8 in KBL).

e.g. find minimal FDs.

R(A,B,C,D,E)

FDs: DC → B, ~~E → AC~~, ~~DE → AB~~, A → B, B → C

E → A

~~E → C~~

DE → A

~~DE → B~~

DC → B

E → A

A → B

B → C

~~B → C~~

$(DE)^+ =$ DEACAB

$E^+ =$ EABC

$(DE)^+ =$ DEABC

Example 6.8.1 from the KBL book. Find a minimal cover for the following FDs:

ABH → C

A → D,

C → E,

~~BGH → F~~

F → AD

E → F

BH → E

$(BH)^+ = BHEFAD$

$(BGH)^+ = BGHEF$

A minimal cover is BH → C, A → D, C → E, F → A, E → F

Lossless-join and dependency-preserving decomposition into 3NF

Algorithm 4.

- Given a relation R with a minimal set of FDs F
- Find a lossless-join BCNF decomposition of R
- For every FD X → A in F which is not preserved after the decomposition, create a relation with schema XA
- Of the two relation schemes R1(X) and R2(Y) where X ⊂ Y, remove relation schema R1(X)

Discuss the correctness...

Can we violate 3NF in the third step?

Proof by contradiction. Discuss the possible cases of violations.

Example. Given relation R(A,B,C,D,E,F) and FDs
{A → B, CE → D, BC → D, AE → F, CD → A}

1) The FDs are minimal (check it)

2) Find a lossless-join BCNF decomposition of R.

3) Project the dependencies on each relation.

4) Any lost dependencies?

5) Find a lossless and dependency-preserving decomposition to 3NF.

R(A, B, C, D, E, F)

A → B ×

R1(A, B)

A → B ✓

R2(A, C, D, E, F)

AE → F ×

R3(A, E, F)

AE → F ✓

R4(A, C, D, E)

CD → A ×

R5(C, D, A)

CD → A ✓

R6(C, D, E)

CE → D

lossless-join decomp. to BCNF

A → B

CE → D ✓

BC → D

AE → F

CD → A

$(CE)^+ = CEDAF$

$(BC)^+ = BCDA$

$A^+ = AB$

$(AE)^+ = AEFB$

$(CD)^+ = CDAB$

last BC → D

create R7(B, C, D)

BC → D

a lossless-join dep.-preserving decomp. to 3NF is

R1, R3, R5, R6, R7

Question. why do we want to preserve the FDs?

Bookings(title, theater, city)

FDs : {theater → city,

city,title → theater}

What is wrong with the decomposition R1(theater, city) and R2(title,theater)?

Two notes about Algorithms 2 and 3:

- The BCNF decomposition algorithm in SKS, decomposes a relation R(Z) with violating FD X→Y to R1(X,Y) and R2(Z-Y). This will also give a lossless join decomposition to BCNF provided that Y has no redundancy (e.g. AB→C is ok but AB→AC is not). Also this can produce more tables than the algorithm discussed in class.
- The 3NF decomposition algorithm in the KBL textbook is missing the last step and that will generate redundant tables.

Desirable Properties of a decomposition:

1 - no redundancy

2 - minimal number of relations

3 - lossless-join

4 - dependency preserving