CS 589 Principles of Databases, Lecture 2-4

Normal Form Review

Key

FDs F on schema R, K ⊆ R

1. $K^+ = R$
2. No proper subset of K has this property

Can be multiple keys

An attribute A is <u>prime</u> if it is in some key K

Superkey: Just Part 1. above

Example

quarter dept course name number room prof time

Q D C N R P T

{DC → N, DN → C, QDCT → RP, P → D, QRT → DC}

QRT^+ = QRTDCNP

QR^+ = QR
QT^+ = QT
RT^+ = RT
not keys

Q^+ =
R^+ =
T^+ =
not keys

QRT is a key

Is DN a key? DN^+ = DNC - no!

David Maier & Lois Delcambre

More Examples

{DC → N, DN → C, QDCT → RP, P → D,
QRT → DC}

$QCPT^+$ = QCPTDNR

CPT^+ = CPTDN

QPT^+ = QPTD

QCT^+ = QCT

QCP^+ = QCPDN

QCPT is key

What about QRPT?

$QRPT^+$ = QRPTDNC

superkey not a key

but QRT is a key

Different Example

F**L**ight **D**ate **P**ilot **G**ate

{LD → PG, L → G}

LD is a key, so L, D are prime

P, G are not prime -

not part of any key

Third Normal Form (3NF)

$A \notin X$

Schema R with FDs F is in third normal form if for any nontrivial FD X → A,

then either X is a superkey for R or A is prime

Last example was not in 3NF

Consider L → G

not a superkey

not prime

$L^+ = LG$

Another Example

FLight Date PilotID PilotName

{LD → IN, I → N, N → I}

not a problem

$LD^+ = LDIN$

not a superkey

not prime

$N^+ = NI$

Aside: Boyce-Codd Normal Form (BCNF)

Schema R with FDs F is in <u>Boyce-Codd normal form</u> if for any nontrivial FD X → A, then X is a superkey of R

BCNF Example

Airport **C**ompany **B**illTo

{A C → B, B → C}

This schema is in 3NF

AC is a key   B isn't key, but C is prime

This schema is not in BCNF

B → C is a problem

B is not a key

Normalization through Decomposition

If R not in 3NF for FDs F, find an FD Y → Z in F⁺ where Y is not a key; split into

FLight Date PilotID PilotName

{LD → IN, I → N, N → I}

R(L D I)

LD → I

R1(I N)

I → N

N → I

Problems with Decomposition

1. Figuring out non-prime attributes can be hard
2. Might end up with more relations than you need
3. Might not be able to enforce FDs

Decomposition Example

R = A B C D E

{AB → CDE, AC → BDE, B → C,
C → B, C → D, B →E}

AB, AC are keys, so C → D is a problem

Decompose on C → D

R(A B C E)   R1 (C D)

B → E is a problem, decompose again

R2(A B C)   R3(B E)   R1(C D)

But ... could have R4(A B C), R5(B D E)

Enforcing FDs Example

R = A B C D E

{A → BCDE, CD → E, CE → B}

A is the only key, so CD → E is a problem

Decompose on CD → E

R1 (A B C D)   R2(C D E)

How do you enforce CE → B?

Normalization through Synthesis

Start with schema R and FDs F

Generate database schema R1, R2, …, Rn such that

- Every relation schema is in 3NF
- Can enforce F
- Avoid extra relation schemas
- Have a lossless decomposition from R onto R1, R2, …, Rn.

Depends on a cover for F of a certain form

What's a Cover?

A set of FDs G is a cover for a set of FDs F if $F^+ = G^+$

G is a non-redundant cover if you cannot eliminate FDs and still be a cover

{A → B, B → C, ~~A → C~~} redundant

Can test FDs one by one to see if any can be left out

$\{A \to B,\ B \to C\}$ non-redundant

Minimum Cover

A <u>minimum cover</u> is one with fewest possible FDs among all covers

Non-redundant, but not minimum:
{A → BC, B → A, AD → E, BD → I}

{A → BC, B → A, AD → E I}

Finding a Minimum Cover

Replace every FD X → Y in F with X → X^+, then find a non-redundant cover

{A → BC, B → A, AD → E, BD → I}

{A → ABC, B → ABC, AD → ABCDEI,
BD → ABCDEI}

⇓ non-redundant

{A → ABC, B → ABC, AD → ABCDEI}

CS 589 Principles of Databases, Lecture 2-4

Reduced Covers

Are there extraneous attributes?

Can appear on the right side or the left

{AB → C, A → B}

{A → BC, B → C}

How to remove?

For XA → Y, see if

See if $F \models X \to Y$

For X → YB, see if

See if $(F - \{X \to YB\} \cup \{X \to Y\}) \models X \to YB$

Reduce Left Sides First

G = {A → C, AB → DE,
AB → CDI, AC → J}

Reduce left sides

G′ = {A → C, AB → DE,
AB → CDI, A → J}

Reduce right sides

G″ = {A → C, AB → E,
AB → DI, A → J}

Synthesis Algorithm

Start with FDs F

Find G = minimum cover for F

Find H = reduced cover for G

For each X → Y in H, add a schema XY

Synthesis Example

F = {A → BC, B → A,
AD → E, BD → I}

G = {A → ABC, B → ABC,
AD → ABCDEI, ~~BD → ABCDEI~~}

H = {A → BC, B → A, AD → EI}

R1(ABC), R2(BA), R3(ADEI)

R12(ABC) R3(ADEI)

Lossless Decomposition

If none of the FDs represents a key for all the attributes, need to add one more relation to get a lossless decomposition

R = ABCDE

F = {A → B, C → D}

$\square \to ABCDE$

H

R1(A B), R2(C D)

$\square \to ACE$

R3($\cancel{D}$ A C E)