Relational Database Design Theory

Introduction to Databases

CompSci 316 Spring 2017

DUKE
COMPUTER SCIENCE

Announcements (Wed. Feb. 1)

- Homework #1 due Monday 02/06 (11:59 pm)

Review: Motivation

uid	*uname*	*gid*
142	Bart	dps
123	Milhouse	gov
857	Lisa	abc
857	Lisa	gov
456	Ralph	abc
456	Ralph	gov
…	…	…

- redundancy is bad
  - user name is recorded multiple times
- Leads to update, insertion, deletion anomalies
- Have a systematic approach to detecting and removing redundancy in designs
- Dependencies, decompositions, and normal forms

Review: Functional dependencies

- A functional dependency (FD) $X \rightarrow Y$
  - X and Y are sets of attributes in a relation R
- whenever two tuples in R agree on all the attributes in X, they must also agree on all attributes in Y

X	Y	Z
a	b	c
a	b	$c1$
$a1$	b	$c1$

$X \rightarrow Y$

X	Y	Z	W
a	b	c	d1
a	b	c	d2
a	$b1$	c	d2

$XY \rightarrow Z$

NOTE: You can only say which FDs do not hold in an instance
Cannot say which ones hold
FDs are given by schema : must be true for all instances (like keys)

Review: Attribute closure

- Given
  - R
  - a set of FD's $\mathcal{F}$ that hold in R, and
  - a set of attributes Z in R
- The closure of Z (denoted Z^+) with respect to $\mathcal{F}$ is the set of all attributes $\{A_1, A_2, \ldots\}$ functionally determined by Z
  - that is, $Z \rightarrow A_1 A_2 \ldots$

uid → *uname, twitterid*
twitterid → *uid*
uid, gid → *fromDate*

- *{gid, twitterid}*$^+$ = ?
- *twitterid* → *uid* --------------- Closure grows to *{ gid, twitterid, uid }*
- *uid* → *uname, twitterid* ------------ Closure grows to *{ gid, twitterid, uid, uname }*
- *uid, gid* → *fromDate* ----------- Closure is now all attributes in *UserJoinsGroup*

Review: Superkeys and Keys

Given a relation R and set of FD's $\mathcal{F}$

- Compute K^+ with respect to $\mathcal{F}$
- If K^+ contains all the attributes of R, K is a super key
- If K is also minimal (no proper subset is a superkey), K is a key

Review: Motivation of BCNF decomposition

- Non-key FDs cause redundancy

X	Y	Z
a	b	c_1
a	b	c_2
$a1$	b	c_2

Here $X \rightarrow Y$

Detect such FDs where X is not a superkey, and decompose into two relations

1. One relation gets X, Y (X is a superkey there! this makes it lossless)
2. The other one gets X, Z (in general Z = everything else)

Note: you need to consider all FDs that can be inferred! not only the ones that are given

Review: BCNF decomposition example

uid → *uname*, *twitterid*
twitterid → *uid*
uid, *gid* → *fromDate*

UserJoinsGroup (*uid*, *uname*, *twitterid*, *gid*, *fromDate*)

BCNF violation: *twitterid* → *uid*

apply Armstrong's axioms and rules!

UserId (*twitterid*, *uid*)

BCNF

UserJoinsGroup' (*twitterid*, *uname*, *gid*, *fromDate*)

twitterid → *uname*
twitterid, *gid* → *fromDate*

BCNF violation: *twitterid* → *uname*

UserName (*twitterid*, *uname*)

BCNF

Member (*twitterid*, *gid*, *fromDate*)

BCNF

Lossy and Lossless Decomposition

X	Y	Z
a	b	c_1
a	b	c_2
$a1$	b	c_2

X	Y
a	b
$a1$	b

$\bowtie$

X	Z
a	c_1
a	c_2
$a1$	c_2

Lossless decomposition

X	Y
a	b
$a1$	b

$\bowtie$

Y	Z
b	c_1
b	c_2

Lossy decomposition

Check yourself!
if in one of the two new relations,
the common join attributes is a superkey,
then lossless

Review: Multi-valued Dependency motivation

- *User (uid, gid, place)*
- No FD like uid $\rightarrow$ gid or uid $\rightarrow$ place
- Still redundancy

uid	*gid*	*place*
142	dps	Springfield
142	dps	Australia
456	abc	Springfield
456	abc	Morocco
456	gov	Springfield
456	gov	Morocco
…	…	…

- Given a user, gid and place are independent

e.g. given uid = 456, all combinations exist for

(abc, gov) x (Springfield, Morocco)

Multivalued dependencies

- A multivalued dependency (MVD) has the form $X \twoheadrightarrow Y$, where X and Y are sets of attributes in a relation R

X	Y	Z
a	b_1	c_1
a	b_2	c_2
a	b_2	c_1
a	b_1	c_2
…	…	…

- $X \twoheadrightarrow Y$ means the following:
- whenever two rows in R agree on all the attributes of X
- then we can swap their Y components and get two rows that are also in R

Complete MVD + FD rules

check yourself!

- FD reflexivity, augmentation, and transitivity
- MVD complementation:
  If $X \twoheadrightarrow Y$, 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 (FD is MVD):
  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 $X \rightarrow Z$

An elegant solution: chase

- Given a set of FD's and MVD's $\mathcal{D}$, does another dependency d (FD or MVD) follow from $\mathcal{D}$?
- Procedure
  - Start with the premise of d, and treat them as "seed" tuples in a relation
  - Apply the given dependencies in $\mathcal{D}$ repeatedly
    - If we apply an FD, we infer equality of two symbols
    - If we apply an MVD, we infer more tuples
  - If we infer the conclusion of d, we have a proof
  - Otherwise, if nothing more can be inferred, we have a counterexample

Proof by chase

- In $R(A, B, C, D)$, does $A \twoheadrightarrow B$ and $B \twoheadrightarrow C$ imply that $A \twoheadrightarrow C$?

Have:

	A	B	C	D
	a	b_1	c_1	d_1
	a	b_2	c_2	d_2
$A \twoheadrightarrow B$	a	b_2	c_1	d_1
	a	b_1	c_2	d_2
$B \twoheadrightarrow C$	a	b_2	c_1	d_2
	a	b_2	c_2	d_1
$B \twoheadrightarrow C$	a	b_1	c_2	d_1
	a	b_1	c_1	d_2

Need:

A	B	C	D	
a	b_1	c_2	d_1	✌
a	b_2	c_1	d_2	✌

Another proof by chase

- In $R(A, B, C, D)$, does $A \to B$ and $B \to C$ imply that $A \to C$?

Have:

A	B	C	D
a	b_1	c_1	d_1
a	b_2	c_2	d_2

Need:

$c_1 = c_2$ ✌

$A \to B$ $\qquad b_1 = b_2$

$B \to C$ $\qquad c_1 = c_2$

In general, with both MVD's and FD's,
chase can generate both new tuples and new equalities

Counterexample by chase

- In $R(A, B, C, D)$, does $A \twoheadrightarrow BC$ and $CD \to B$ imply that $A \to B$?

Have:

A	B	C	D
a	b_1	c_1	d_1
a	b_2	c_2	d_2

$A \twoheadrightarrow BC$

a	b_2	c_2	d_1
a	b_1	c_1	d_2

Counterexample!

Need:

$b_1 = b_2$ 👎

Note: the FD must hold on all instances, so showing one instance as a counterexample suffices!

4NF

- A relation R is in Fourth Normal Form (4NF) if
  - For every non-trivial MVD $X \twoheadrightarrow Y$ in R, X is a superkey
  - That is, all FD's and MVD's follow from "key → other attributes" (i.e., no MVD's and no FD's besides key functional dependencies)

- 4NF is stronger than BCNF
  - Because every FD is also a MVD
  - why? because trivially if two tuples have same X value, they also have the same Y value, no question in swapping the Y values!

4NF decomposition algorithm

- Find a 4NF violation
  - A non-trivial MVD $X \twoheadrightarrow Y$ in R where X is not a superkey
- Decompose R into R_1 and R_2, where
  - R_1 has attributes $X \cup Y$
  - R_2 has attributes $X \cup Z$ (where Z contains R attributes not in X or Y)
- Repeat until all relations are in 4NF

- Almost identical to BCNF decomposition algorithm
- Any decomposition on a 4NF violation is lossless

4NF decomposition example

uid	*gid*	*place*
142	dps	Springfield
142	dps	Australia
456	abc	Springfield
456	abc	Morocco
456	gov	Springfield
456	gov	Morocco
…	…	…

User (uid, gid, place)

4NF violation: *uid* ↠ *gid*

Member (uid, gid)

4NF

uid	*gid*
142	dps
456	abc
456	gov
…	…

Visited (uid, place)

4NF

uid	*place*
142	Springfield
142	Australia
456	Springfield
456	Morocco
…	…

Summary

- Philosophy behind BCNF, 4NF:
  Data should depend on the key,
  the whole key,
  and nothing but the key!
  - You could have multiple keys though

- Other normal forms
  - 3NF: More relaxed than BCNF; will not remove redundancy if doing so makes FDs harder to enforce
  - 2NF: Slightly more relaxed than 3NF
  - 1NF: All column values must be atomic

Next: Project Mixer!