Relational Database Design Theory

CPS 116

Introduction to Database Systems

Announcements (Thu. Sep. 15)

2

Motivation

3

SID	name	CID
142	Bart	CPS116
142	Bart	CPS114
857	Lisa	CPS116
857	Lisa	CPS130
…	…	…

- How do we tell if a design is bad, e.g., *StudentEnroll* (*SID*, *name*, *CID*)?
  - This design has redundancy, because the name of a student is recorded multiple times, once for each course the student is taking
    - Update, insertion, deletion anomalies
- How about a systematic approach to detecting and removing redundancy in designs?
  - Dependencies, decompositions, and normal forms

Functional dependencies

4

- A functional dependency (FD) has the form $X \rightarrow Y$, where X and Y are sets of attributes in a relation R
- $X \rightarrow Y$ means that 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*	*?*
…	…	…

Must be *b*

Could be anything

FD examples

5

Address (*street_address*, *city*, *state*, *zip*)

- *street_address*, *city*, *state* $\rightarrow$ *zip*
- *zip* $\rightarrow$ *city*, *state*
- *zip*, *state* $\rightarrow$ *zip*?
  - This is a trivial FD
  - Trivial FD: LHS $\supseteq$ RHS
- *zip* $\rightarrow$ *state*, *zip*?
  - This is non-trivial, but not completely non-trivial
  - Completely non-trivial FD: LHS $\cap$ RHS $= \varnothing$

Keys redefined using FD's

6

A set of attributes K is a key for a relation R if

- $K \rightarrow$ all (other) attributes of R
  - That is, K is a "super key"
- No proper subset of K satisfies the above condition
  - That is, K is minimal

Reasoning with FD's

7

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

- Does another FD follow from $\mathcal{F}$?
  - Are some of the FD's in $\mathcal{F}$ redundant (i.e., they follow from the others)?
- Is K a key of R?
  - What are all the keys of R?

Attribute closure

8

- 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, $Z \rightarrow A_1 A_2 \ldots$)
- Algorithm for computing the closure
  - Start with closure $= Z$
  - If $X \rightarrow Y$ is in $\mathcal{F}$ and X is already in the closure, then also add Y to the closure
  - Repeat until no more attributes can be added

A more complex example

9

StudentGrade (*SID*, *name*, *email*, *CID*, *grade*)

(Not a good design, and we will see why later)

Example of computing closure

10

- $\mathcal{F}$ includes:
  - *SID* → *name*, *email*
  - *email* → *SID*
  - *SID*, *CID* → *grade*
- { *CID*, *email* }$^+$ = ?
- *email* → *SID*
  - Add *SID*; closure is now { *CID*, *email*, *SID* }
- *SID* → *name*, *email*
  - Add *name*, *email*; closure is now { *CID*, *email*, *SID*, *name* }
- *SID*, *CID* → *grade*
  - Add *grade*; closure is now all the attributes in *StudentGrade*

Using attribute closure

11

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

- Does another FD $X \rightarrow Y$ follow from $\mathcal{F}$?
  - Compute X^+ with respect to $\mathcal{F}$
  - If $Y \subseteq X^+$, then $X \rightarrow Y$ follow from $\mathcal{F}$
- Is K a key of R?
  - Compute K^+ with respect to $\mathcal{F}$
  - If K^+ contains all the attributes of R, K is a super key
  - Still need to

Rules of FD's

12

- Armstrong's axioms
  - Reflexivity: If $Y \subseteq X$, then $X \rightarrow Y$
  - Augmentation: If $X \rightarrow Y$, then $XZ \rightarrow YZ$ for any Z
  - Transitivity: If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$
- Rules derived from axioms
  - Splitting: If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$
  - Combining: If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
- ☞ Using these rules, you can prove or disprove an FD given a set of FDs

Non-key FD's

13

- Consider a non-trivial FD $X \to Y$ where X is not a super key
  - Since X is not a super key, there are some attributes (say Z) that are not functionally determined by X

X	Y	Z
a	b	c_1
a	b	c_2
…	…	…

That b is always associated with a is recorded by multiple rows: redundancy, update/insertion/deletion anomalies

Example of redundancy

14

- *StudentGrade* (*SID*, *name*, *email*, *CID*, *grade*)
- *SID* $\to$ *name*, *email*

SID	*name*	*email*	*CID*	*grade*
142	Bart	bart@fox.com	CPS116	B-
142	Bart	bart@fox.com	CPS114	B
123	Milhouse	milhouse@fox.com	CPS116	B+
857	Lisa	lisa@fox.com	CPS116	A+
857	Lisa	lisa@fox.com	CPS130	A+
456	Ralph	ralph@fox.com	CPS114	C
…	…	…	…	…

Decomposition

15

SID	*name*	*email*	*CID*	*grade*
…	…	…	…	…

SID	*name*	*email*
142	Bart	bart@fox.com
123	Milhouse	milhouse@fox.com
857	Lisa	lisa@fox.com
456	Ralph	ralph@fox.com
…	…	…

SID	*CID*	*grade*
142	CPS116	B-
142	CPS114	B
123	CPS116	B+
857	CPS116	A+
857	CPS130	A+
456	CPS114	C
…	…	…

- Eliminates redundancy
- To get back to the original relation:

Unnecessary decomposition

16

SID	name	email
142	Bart	bart@fox.com
123	Milhouse	milhouse@fox.com
857	Lisa	lisa@fox.com
456	Ralph	ralph@fox.com
…	…	…

SID	name
142	Bart
123	Milhouse
857	Lisa
456	Ralph
…	…

SID	email
142	bart@fox.com
123	milhouse@fox.com
857	lisa@fox.com
456	ralph@fox.com
…	…

- Fine: join returns the original relation
- Unnecessary: no redundancy is removed, and now *SID* is stored twice!

Bad decomposition

17

SID	CID	grade
142	CPS116	B-
142	CPS114	B
123	CPS116	B+
857	CPS116	A+
857	CPS130	A+
456	CPS114	C
…	…	…

SID	CID
142	CPS116
142	CPS114
123	CPS116
857	CPS116
857	CPS130
456	CPS114
…	…

SID	grade
142	B-
142	B
123	B+
857	A+
456	C
…	…

- Association between *CID* and *grade* is lost
- Join returns more rows than the original relation

Lossless join decomposition

18

- Decompose relation R into relations S and T
  - $attrs(R) = attrs(S) \cup attrs(T)$
  - $S = \pi_{attrs(S)}(R)$
  - $T = \pi_{attrs(T)}(R)$
- The decomposition is a lossless join decomposition if, given known constraints such as FD's, we can guarantee that $R = S \bowtie T$
- Any decomposition gives $R \subseteq S \bowtie T$ (why?)
  - A lossy decomposition is one with $R \subset S \bowtie T$

Loss? But I got more rows!

19

- "Loss" refers not to the loss of tuples, but to the loss of information
  - Or, the ability to distinguish different original relations

SID	*CID*	*grade*
142	CPS116	B
142	CPS114	B-
123	CPS116	B+
857	CPS116	A+
857	CPS130	A+
456	CPS114	C
…	…	…

No way to tell
which is the original relation

SID	*CID*
142	CPS116
142	CPS114
123	CPS116
857	CPS116
857	CPS130
456	CPS114
…	…

SID	*grade*
142	B-
142	B
123	B+
857	A+
456	C
…	…

Questions about decomposition

20

- When to decompose
- How to come up with a correct decomposition (i.e., lossless join decomposition)

An answer: BCNF

21

- A relation R is in Boyce-Codd Normal Form if
  - For every non-trivial FD $X \rightarrow Y$ in R, X is a super key
  - That is, all FDs follow from "key → other attributes"
- When to decompose
  - As long as some relation is not in BCNF
- How to come up with a correct decomposition
  - Always decompose on a BCNF violation (details next)
  - ☞Then it is guaranteed to be a lossless join decomposition!

BCNF decomposition algorithm

22

- Find a BCNF violation
  - That is, a non-trivial FD $X \rightarrow Y$ in R where X is not a super key of R
- 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 all attributes of R that are in neither X nor Y
- Repeat until all relations are in BCNF

BCNF decomposition example

23

SID → *name*, *email*
email → *SID*
SID, *CID* → *grade*

StudentGrade (*SID*, *name*, *email*, *CID*, *grade*)

BCNF violation: *SID* → *name*, *email*

Student (*SID*, *name*, *email*) *Grade* (*SID*, *CID*, *grade*)

Another example

24

SID → *name*, *email*
email → *SID*
SID, *CID* → *grade*

StudentGrade (*SID*, *name*, *email*, *CID*, *grade*)

Why is BCNF decomposition lossless

25

Given non-trivial $X \rightarrow Y$ in R where X is not a super key of R, need to prove:

- ❖ Anything we project always comes back in the join: $R \subseteq \pi_{XY}(R) \bowtie \pi_{XZ}(R)$
  - Sure; and it doesn't depend on the FD
- ❖ Anything that comes back in the join must be in the original relation: $R \supseteq \pi_{XY}(R) \bowtie \pi_{XZ}(R)$
  - Proof makes use of the fact that $X \rightarrow Y$

Recap

26

- ❖ Functional dependencies: a generalization of the key concept
- ❖ Non-key functional dependencies: a source of redundancy
- ❖ BCNF decomposition: a method for removing redundancies
  - BNCF decomposition is a lossless join decomposition
- ❖ BCNF: schema in this normal form has no redundancy due to FD's

BCNF = no redundancy?

27

- ❖ *Student* (*SID*, *CID*, *club*)
  - Suppose your classes have nothing to do with the clubs you join
  - FD's?
  - BCNF?
  - Redundancies?

SID	*CID*	*club*
142	CPS116	ballet
142	CPS116	sumo
142	CPS114	ballet
142	CPS114	sumo
123	CPS116	chess
123	CPS116	golf
…	…	…

Multivalued dependencies

28

- A multivalued dependency (MVD) has the form $X \twoheadrightarrow Y$, where X and Y are sets of attributes in a relation R
- $X \twoheadrightarrow Y$ means that whenever two rows in R agree on all the attributes of X, then we can swap their Y components and get two new rows that are also in R

X	Y	Z
a	$b1$	$c1$
a	$b2$	$c2$
a	$b1$	$c2$
a	$b2$	$c1$
…	…	…

Must be in R too

MVD examples

29

Student (*SID*, *CID*, *club*)

- *SID*, *CID* $\twoheadrightarrow$ *club*
  - Trivial: LHS $\cup$ RHS = all attributes of R
- *SID*, *CID* $\twoheadrightarrow$ *SID*
  - Trivial: LHS $\supseteq$ RHS

Complete MVD + FD rules

30

- 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 $X \rightarrow Z$

Try proving things using these!

31

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 hypothesis 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

32

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	$b1$	$c1$	$d1$
a	$b2$	$c2$	$d2$

Need

A	B	C	D	
a	$b1$	$c2$	$d1$	✌
a	$b2$	$c1$	$d2$	✌

$A \twoheadrightarrow B$

a	$b2$	$c1$	$d1$
a	$b1$	$c2$	$d2$

$B \twoheadrightarrow C$

a	$b2$	$c1$	$d2$
a	$b2$	$c2$	$d1$

$B \twoheadrightarrow C$

a	$b1$	$c2$	$d1$
a	$b2$	$c1$	$d2$

33

Another proof by chase

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

Have

A	B	C	D
a	$b1$	$c1$	$d1$
a	$b2$	$c2$	$d2$

Need

$c1 = c2$ ✌

$A \rightarrow B$ $b1 = b2$

$B \rightarrow C$ $c1 = c2$

In general, both new tuples and new equalities may be generated

Counterexample by chase

34

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

Have

A	B	C	D
a	$b1$	$c1$	$d1$
a	$b2$	$c2$	$d2$

$A \twoheadrightarrow BC$

a	$b2$	$c2$	$d1$
a	$b1$	$c1$	$d2$

Counterexample!

Need

$b1 = b2$

4NF

35

- 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 $\rightarrow$ 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

4NF decomposition algorithm

36

- 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 $X \cup Z$ (Z contains 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

37

SID	CID	club
142	CPS116	ballet
142	CPS116	sumo
142	CPS114	ballet
142	CPS114	sumo
123	CPS116	chess
123	CPS116	golf
…	…	…

Student (*SID*, *CID*, *club*)

4NF violation: $SID \twoheadrightarrow CID$

Enroll (*SID*, *CID*)

4NF

SID	CID
142	CPS116
142	CPS114
123	CPS116
…	…

Join (*SID*, *club*)

4NF

SID	club
142	ballet
142	sumo
123	chess
123	golf
…	…

Summary

38

- Philosophy behind BCNF, 4NF:
  Data should depend on the key, the whole key, and nothing but the key!
- 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