Schema Refinement & Normalization Theory

Normal Forms

Boyce-Codd Normal Form (BCNF)

- Reln R with FDs F is in BCNF if **for each non-trivial FD** $X \rightarrow A$ in F**, X is a super key for R** (i.e., $X \rightarrow R$ in F^+).
  - An FD $X \rightarrow A$ is said to be "trivial" if $A \subseteq X$.
- In other words, R is in BCNF if the only non-trivial FDs that hold over R are *key constraints*.
- If BCNF:
  - No "data" in R can be predicted using FDs alone. Why:
  - Because X is a (super)key, we can't have two different tuples that agree on the X value

X	Y	A
x	y1	a
x	y2	?

Suppose we know that this instance satisfies $X \rightarrow A$. This situation cannot arise if the relation is in BCNF.

BCNF

- Consider relation R with FDs F. If $X \rightarrow A$ in F over R ($X \subseteq R$, $A \subseteq R$) violates BCNF, it means
  - A is not in X, <u>and</u> → non-trivial FD
  - $X \rightarrow R$ is not in F^+ → X is not a superkey

- In other words, for $X \rightarrow A$ in F over R to satisfy BCNF requirement, <u>at least one</u> of the followings must be true:
  - $X \rightarrow A$ is trivial, i.e. A is in X, <u>or</u>
  - X is a superkey, i.e. $X \rightarrow R$ is in F^+

Decomposition of a Relation Schema

- When a relation schema is not in BCNF: **decompose**.
- Suppose that relation R contains attributes *A1 … An.* A *decomposition* of R consists of replacing R by two or more relations such that:
  - Each new relation scheme contains a subset of the attributes of R (and no attributes that do not appear in R), and
  - Every attribute of R appears as an attribute of at least one of the new relations.
- Intuitively, decomposing R means we will store instances of the relation schemes produced by the decomposition, instead of instances of R.

Decomposition example

Original relation (not stored in DB!)

S	N	L	R	W	H
123-22-3666	Attishoo	48	8	10	40
231-31-5368	Smiley	22	8	10	30
131-24-3650	Smethurst	35	5	7	30
434-26-3751	Guldu	35	5	7	32
612-67-4134	Madayan	35	8	10	40

Decomposition (in the DB)

=

S	N	L	R	H
123-22-3666	Attishoo	48	8	40
231-31-5368	Smiley	22	8	30
131-24-3650	Smethurst	35	5	30
434-26-3751	Guldu	35	5	32
612-67-4134	Madayan	35	8	40

$\bowtie$

R	W
8	10
5	7

Problems with Decompositions

- There are three potential problems to consider:
  - ❶ Some queries become more expensive.
    - e.g., How much did sailor Attishoo earn? (earn = W*H)
  - ❷ Given instances of the decomposed relations, we may not be able to reconstruct the corresponding instance of the original relation!
    - Fortunately, not in the SNLRWH example.
  - ❸ Checking some dependencies may require joining the instances of the decomposed relations.
    - Fortunately, not in the SNLRWH example.
- *Tradeoff*: Must consider these issues vs. redundancy.

Example of problem 2

Student_ID	Name	Dcode	Cno	Grade
123-22-3666	Attishoo	INFS	501	A
231-31-5368	Guldu	CS	102	B
131-24-3650	Smethurst	INFS	614	B
434-26-3751	Guldu	INFS	614	A
434-26-3751	Guldu	INFS	612	C

$\neq$

Name	Dcode	Cno	Grade
Attishoo	INFS	501	A
Guldu	CS	102	B
Smethurst	INFS	614	B
Guldu	INFS	614	A
Guldu	INFS	612	C

$\bowtie$

Student_ID	Name
123-22-3666	Attishoo
231-31-5368	Guldu
131-24-3650	Smethurst
434-26-3751	Guldu

Lossless Join Decompositions

- Decomposition of R into R_1 and R_2 is *lossless-join* w.r.t. a set of FDs F if, for every instance *r* that satisfies F, we have:

$$\pi_{R_1}(r) \bowtie \pi_{R_2}(r) = r$$

- It is always true that

$$r \subseteq \pi_{R_1}(r) \bowtie \pi_{R_2}(r)$$

- In general, the other direction does not hold! If it does, the decomposition is *lossless-join*.

Example (lossy decomposition)

r

A	B	C
1	2	3
4	5	6
7	2	8

$\pi_{AB}(r)$

A	B
1	2
4	5
7	2

$\pi_{BC}(r)$

B	C
2	3
5	6
2	8

$\pi_{AB}(r) \bowtie \pi_{BC}(r)$

A	B	C
1	2	3
4	5	6
7	2	8
1	2	8
7	2	3

Example (lossless join decomposition)

r

A	B	C
1	2	3
4	5	6
7	2	3

$\pi_{AB}(r)$

A	B
1	2
4	5
7	2

$\pi_{BC}(r)$

B	C
2	3
5	6

$\pi_{AB}(r) \bowtie \pi_{BC}(r)$

A	B	C
1	2	3
4	5	6
7	2	3

$Suppose\ (AB \cap BC) \rightarrow BC$

Lossless Join Decomposition

- The decomposition of R into R_1 and R_2 is lossless-join wrt F if and only if F^+ contains:
  - $R_1 \cap R_2 \rightarrow R_1$, or
  - $R_1 \cap R_2 \rightarrow R_2$
- In particular, the decomposition of R into (UV) and (R-V) is lossless-join if $U \rightarrow V$ holds on R
  - assume U and V do not share attributes.
  - WHY?

Decomposition

- Definition extended to decomposition into 3 or more relations in a straightforward way.
- *It is essential that all decompositions used to deal with redundancy be lossless! (Avoids Problem (2))*

Decomposition into BCNF

- Recall: Consider relation R with FDs F. If $X \rightarrow A$ in F over R ($X \subseteq R$, $A \subseteq R$) violates BCNF, it means
  - A is not in X, <u>and</u> $\rightarrow$ non-trivial FD
  - $X \rightarrow R$ is not in F^+ $\rightarrow$ X is not a superkey

- Recall that for $X \rightarrow A$ in F over R to satisfy BCNF requirement, <u>at least one</u> of the followings must be true:
  - $X \rightarrow A$ is trivial, i.e. A is in X, <u>or</u>
  - X is a superkey, i.e. $X \rightarrow R$ is in F^+

Decomposition into BCNF

- Consider relation R with FDs F. If $X \rightarrow A$ in F over R ($X \subseteq R$, $A \subseteq R$) violates BCNF, i.e.,
  - A is not in X, <u>and</u> → non-trivial FD
  - $X \rightarrow R$ is not in F^+ → X is not a (super)key
- Then: decompose R into R - A and XA.
- Repeated application of this idea will give us a collection of relations that are in BCNF; lossless join decomposition, and guaranteed to terminate.

BCNF Decomposition Example

- $R = (A, B, C)$
  $F = \{A \rightarrow B; B \rightarrow C\}$
  Key = $\{A\}$
- R is not in BCNF ($B \rightarrow C$ but B is not a superkey)
- Decomposition
  - $R_1 = (B, C)$
  - $R_2 = (A, B)$

How do we know R is in BCNF?

- If R has only two attributes, then it is in BCNF
- If F only uses attributes in R, then:
  - R is in BCNF *if and only if* for each $X \rightarrow Y$ in F (***not F^+!***), X is a superkey of R, i.e., $X \rightarrow R$ is in F^+ (not F!).
- What if F uses attributes not in R?
  - Next

Checking for BCNF Violations

- List all non-trivial FDs
- Ensure that left hand side of each FD is a superkey
- Does not work on decomposed tables
  - Consider $R = (A, B, C, D, E)$, with $F = \{A \rightarrow B, BC \rightarrow D\}$
    - Decompose R into $R_1 = (A,B)$ and $R_2 = (A,C, D, E)$
    - Neither of the dependencies in F contain only attributes from (A,C,D,E) so we might be mislead into thinking R_2 satisfies BCNF.
    - In fact, dependency $AC \rightarrow D$ in F^+ shows R_2 is not in BCNF.

Testing Decomposition for BCNF

- To check if a relation R_i in a decomposition of R is in BCNF,
  - Either test R_i for BCNF with respect to the **restriction** of F to R_i (that is, all FDs in F^+ that contain only attributes from R_i)
  - or use the original set of dependencies F that hold on R, but with the following test:
    - for every set of attributes $X \subseteq R_i$, check that X^+ either includes no attribute of R_i- X, or includes all attributes of R_i.
  - If the condition is violated by some $X \rightarrow Y$ in F, the dependency $X \rightarrow (X^+ - X) \cap R_i$ can be shown to hold on R_i, and R_i violates BCNF.
  - We use above dependency to decompose R_i

Example of BCNF Decomposition

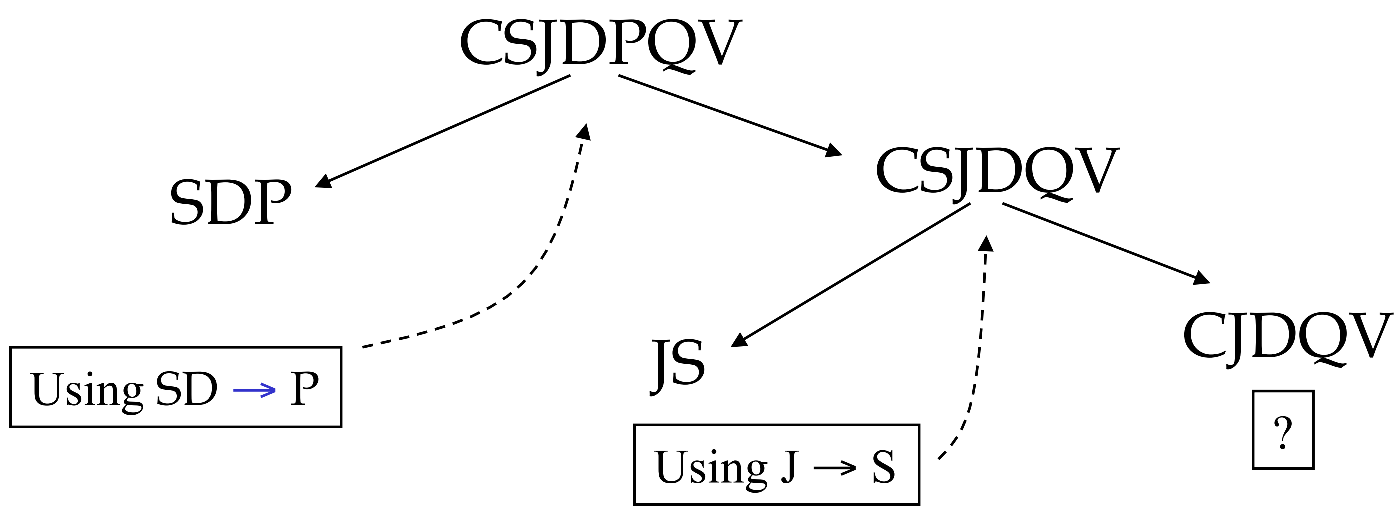
- *class* (*course_id*, *title*, *dept_name*, *credits*, *sec_id*, *semester*, *year*, *building*, *room_number*, *capacity*, *time_slot_id*)
- Functional dependencies:
  - *course_id*→ *title*, *dept_name*, *credits*
  - *building*, *room_number*→*capacity*
  - *course_id*, *sec_id*, *semester*, *year*→*building*, *room_number*, *time_slot_id*
- A candidate key {*course_id*, *sec_id*, *semester*, *year*}.
- BCNF Decomposition:
  - *course_id*→ *title*, *dept_name*, *credits* holds
    - but *course_id* is not a superkey.
  - We replace *class* by:
    - *course*(*course_id*, *title*, *dept_name*, *credits*)
    - *class-1* (*course_id*, *sec_id*, *semester*, *year*, *building*, *room_number*, *capacity*, *time_slot_id*)

BCNF Decomposition (Cont.)

- *course* is in BCNF
  - How do we know this?
- *building*, *room_number* → *capacity* holds on *class-1*
  - but {*building*, *room_number*} is not a superkey for *class-1*.
  - We replace *class-1* by:
    - *classroom* (*building*, *room_number*, *capacity*)
    - *section* (*course_id*, *sec_id*, *semester*, *year*, *building*, *room_number*, *time_slot_id*)
- *classroom* and *section* are in BCNF.

BCNF Decomposition Example 2

- Assume relation schema CSJDPQV:
  Contracts(*contract_id, supplier, project, dept, part, qty, value*)
  - key C, JP → C, SD → P,   J → S
- To deal with SD → P, decompose into SDP, CSJDQV.
- To deal with J → S, decompose CSJDQV into JS and CJDQV
- A tree representation of the decomposition:



BCNF Decomposition

- In general, several dependencies may cause violation of BCNF. The order in which we "deal with" them could lead to very different sets of relations!