

Today's Lecture

- Designing relational schemas
 - More about 3NF
 - Properties of Decompositions
 - Lossless-Join Decomposition
 - Dependency-Preserving Decomposition
 - Algorithms for finding BCNF and 3NF decompositions

Winter 2003

1

Recommended Readings

- Textbook
 - Refer to Chapter 19.4 – 19.6 for additional information

Winter 2003

2

More on 3NF

- Let R be a relation schema, \mathcal{F} be a set of FDs given to hold over R , A is an attribute in R , and X is a subset of attributes in R
- R is in 3NF if
 - For every FD $X \rightarrow A$ in \mathcal{F} , one of the following is true
 - $X \rightarrow A$ is a trivial dependency (i.e., $A \subseteq X$)
 - X is a superkey
 - A is part of some key for R

Checking if a relation schema is in 3NF is NP-complete

- How can a relation schema violate 3NF?

Winter 2003

3

3NF violations

- One of the FD $X \rightarrow A$ in \mathcal{F} is such that
 - $X \rightarrow A$ is NOT a trivial dependency AND
 - X is NOT a superkey AND
 - A is NOT part of every key
- Suppose a dependency $X \rightarrow A$ causes a violation of 3NF. A violation can be classified into two cases:
 - Either X is a proper subset of some key K or X is not a proper subset of any key
- A set S is a proper subset of a set T if $S \subseteq T$ and $|S| < |T|$

Winter 2003

4

3NF violations

- $R(\underline{A, B}, C, D, E), \mathcal{F} = \{ AB \rightarrow DE, D \rightarrow A, B \rightarrow C \}$
 - Redundancy in storing B, C values
 - Partial dependency: A FD $X \rightarrow Y$ is a partial dependency if X is a proper subset of some key in relation schema R
- $R(\underline{A, B}, C, D, E), \mathcal{F} = \{ AB \rightarrow C, C \rightarrow D, C \rightarrow E \}$
 - Redundancy in storing C, E values
 - Transitive dependency: A FD $X \rightarrow Y$ is a transitive dependency if X is not a proper subset of every key in R
 - $\text{Key} \rightarrow X \rightarrow Y$

Winter 2003

5

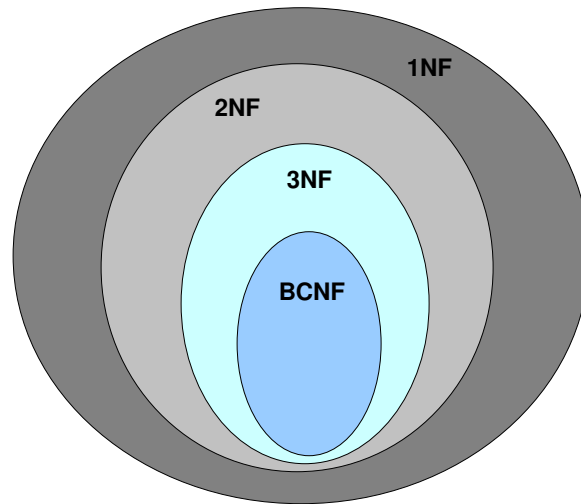
Second Normal Form

- Exists more for historical reasons
- Second normal form (2NF): No partial dependencies
- A relation schema R is in 2NF if for every non-key attribute A, A is functionally determined by the entire key
- Obviously if $K \rightarrow A$, where K is a key, it may happen that A is functionally determined by a smaller subset of attributes in K and this is not allowed in 2NF
- obsolete now
- 3NF has more “practical” value

Winter 2003

6

The big picture



Winter 2003

7

Main steps in relational database design

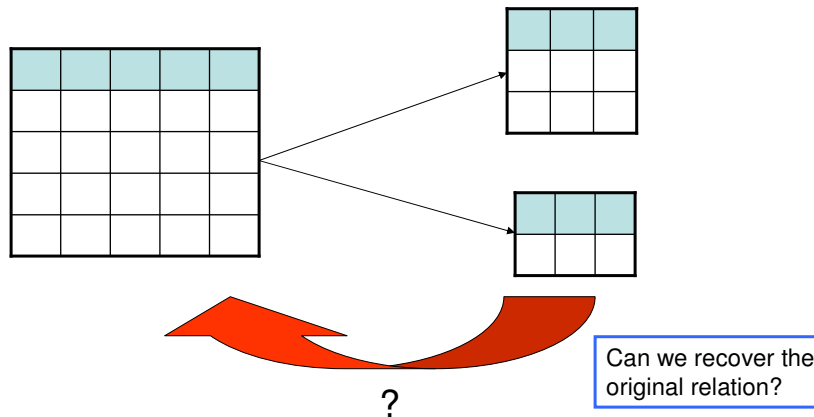
- ER diagrams to arrive at initial table design
- Determine the functional dependencies that exists in domain
- Use FDs to decompose relations in order to arrive normal forms
- Analyze query workload, adjust table design

Winter 2003

8

Decomposing relations

- Decomposing a relation into two or more relations allows us to remove redundancies



Winter 2003

9

Lossless-Join Decomposition

- Let R be a relation schema and \mathcal{F} be a set of FDs over R .
- A decomposition of R into n schemas with attribute sets X_1, \dots, X_n is a lossless-join decomposition with respect to \mathcal{F} if
 - For every instance r of R that satisfies \mathcal{F} ,

$$\pi_{X_1}(r) \bowtie \dots \bowtie \pi_{X_n}(r) = r$$

Natural join

- It is possible to recover R from the individual relations $\pi_{X_1}(r), \dots, \pi_{X_n}(r)$

Winter 2003

10

Example

- Let $R(A, B, C)$ be a relation schema with no functional dependencies
- Is the decomposition of R into schemas with attribute sets A, B and B, C a lossless decomposition?

Instance r_1

A	B	C
a1	b1	c1
a1	b1	c2
a1	b2	c3

$\pi_{A,B}(r_1)$

A	B
a1	b1
a1	b2

$\pi_{A,B}(r_1) \bowtie \pi_{B,C}(r_1)$

A	B	C
a1	b1	c1
a1	b1	c2
a1	b2	c3

$\pi_{B,C}(r_1)$

B	C
b1	c1
b1	c2
b2	c3

Winter 2003

11

Example

Instance r_2

A	B	C
a1	b1	c1
a2	b1	c2

$\pi_{A,B}(r_2)$

A	B
a1	b1
a2	b1

$\pi_{A,B}(r_2) \bowtie \pi_{B,C}(r_2)$

A	B	C
a1	b1	c1
a1	b1	c2
a2	b1	c1
a2	b1	c2

Lossy!

$\pi_{B,C}(r_2)$

B	C
b1	c1
b1	c2

- By projecting on $\{A, B\}$ and $\{B, C\}$, some information may be lost in general
- E.g., we no longer know that $(a1, b1, c2)$ does not exist
- Hence $\{A, B\}$ and $\{B, C\}$ is not a lossless-join decomposition of R

Winter 2003

12

Example

Instance r2

A	B	C
a1	b1	c1
a2	b1	c2

$\pi_{A,B}(r2)$

A	B
a1	b1
a2	b1

$\pi_{B,C}(r2)$

B	C
b1	c1
b1	c2

$\pi_{A,B}(r2) \bowtie \pi_{B,C}(r2)$

A	B	C
a1	b1	c1
a1	b1	c2
a2	b1	c1
a2	b1	c2

- What if we know $B \rightarrow A$? Then, r2 is not a legal instance since it does not satisfy the dependency
- Instance r1, however, is a legal instance

Winter 2003

13

A necessary and sufficient condition

- Obviously, we would like our decompositions to be lossless and be able to decide when a decomposition is lossless
- Let R be a relation and \mathcal{F} is be set of FDs that hold over R
- The decomposition of R into relations R_1 and R_2 is lossless if and only if \mathcal{F}^+ contains either
 - $R_1 \cap R_2 \rightarrow R_1$, or
 - $R_1 \cap R_2 \rightarrow R_2$

Winter 2003

14

Proof

- In class

Winter 2003

15

Example

- Recall the example in the last lecture
Employee(eid, name, addr, rank, salary-scale)
and FD rank \rightarrow salary-scale
- Is the decomposition of Employee into Emp(eid, name, addr, rank) and Rank(rank, salary-scale) a lossless decomposition?
- $R1 = \{eid, name, addr, rank\}$
- $R2 = \{rank, salary-scale\}$
- $R1 \cap R2 = \{rank\}$
- Obviously, $R1 \cap R2 \rightarrow R2$
- Therefore, the decomposition is lossless

Winter 2003

16

Observations

- Given a relation R , an FD $X \rightarrow Y$ that holds over R , and $X \cap Y$ is empty, then the decomposition of R into $R-Y$ and XY is lossless
 - $R_1 = R-Y$, $R_2 = XY$
 - $R_1 \cap R_2 = X$, $X \rightarrow Y$
 - Therefore, $R_1 \cap R_2 \rightarrow R_2$
- Repeated lossless decompositions:
 - Given a set of FDs \mathcal{F} , if R can be losslessly decomposed into R_1 and R_2 and, R_2 can be losslessly decomposed into R_3 and R_4 , then the decomposition of R into relations R_1 , R_3 , and R_4 is a lossless decomposition

Winter 2003

17

Dependency-Preserving Decomposition

- Consider the relation schema $R(A, B, C, D, E)$ with FDs $A \rightarrow BCDE$, $BD \rightarrow A$, and $CE \rightarrow B$
- Is this relation in BCNF?
 - No, because in the dependency $CE \rightarrow B$, CE is not a superkey
- Decompose the relation into $R_1(A, C, D, E)$, $R_2(C, E, B)$
 - Note that this is a lossless-join decomposition
 - But how can we enforce the dependency $BD \rightarrow A$ with these two relations?

Winter 2003

18

Dependency-Preserving Decomposition

- The solution is to join R1 and R2 and check that the dependency $BD \rightarrow A$ is not violated whenever a tuple is inserted or modified.
 - Only problem: expensive!
- It would be convenient if we could test if every dependency still holds, whenever a tuple is modified or inserted, just by testing whether dependencies hold locally at R1 and R2 respectively
- If a decomposition is dependency-preserving, it is possible to enforce all dependencies through such “local tests”

Winter 2003

19

Dependency-Preserving Decomposition

- Let R be a relation schema decomposed into two relations with attribute sets X and Y respectively and let \mathcal{F} denote the set of dependencies that should hold
- Let \mathcal{F}_X denote the set of FDs in \mathcal{F}^+ that involve only attributes in X. Similarly for \mathcal{F}_Y
- The decomposition of R with FDs \mathcal{F} into two schemas with attribute sets X and Y is dependency preserving if $(\mathcal{F}_X \cup \mathcal{F}_Y)^+ = \mathcal{F}^+$
- Intuitively, we can check that \mathcal{F}_X holds in the first decomposition and \mathcal{F}_Y holds in the second decomposition and infer that \mathcal{F} holds overall

Winter 2003

20

Example

- Consider another example $R(A,B,C)$ with dependencies $A \rightarrow B$, $B \rightarrow C$, and $C \rightarrow A$ and decomposed into $R_1(A,B)$ and $R_2(B,C)$
- Is this a lossless-decomposition? YES
- Is this decomposition dependency-preserving? YES
 - $A \rightarrow B, B \rightarrow A \in \mathcal{F}_{R_1}$
 - $B \rightarrow C, C \rightarrow B \in \mathcal{F}_{R_2}$
 - And $(\mathcal{F}_{R_1} \cup \mathcal{F}_{R_2})^+$ contains $C \rightarrow A$ and obviously contains $A \rightarrow B$ and $B \rightarrow C$

Note that \mathcal{F}_{R_i} is computed from \mathcal{F}^+

Winter 2003

21

Example

- Consider another example $R(A, B, C)$ with dependency $A \rightarrow B$ and decomposed into $R_1(A,B)$ and $R_2(B,C)$
- Is this decomposition dependency-preserving? YES
 - $A \rightarrow B \in \mathcal{F}_{R_1}$
 - $(\mathcal{F}_{R_1} \cup \mathcal{F}_{R_2})^+$ contains $A \rightarrow B$
- Is this a lossless-decomposition? NO
 - Can easily come up with an example to show that the example is not a lossless decomposition
 - It should be decomposed into $R_1(A,C)$ and $R_2(A,B)$ to be lossless

Winter 2003

22

Normalization

- Given a relation schema and functional dependencies, it is obviously desirable to obtain a set of BCNF relations from it
 - recall that BCNF relations are desirable because they have, in general, less redundancies than 3NF relations (see next slide)
- Given a relation schema and functional dependencies, it is always possible to decompose the schema into a set of 3NF relations that is lossless and dependency-preserving
- We can also decompose the relation schema into a set of BCNF relations that is lossless. However, it may not always be dependency-preserving

Winter 2003

23

3NF example - there may still be redundancies

- Consider $R(A, B, C, D)$, $A \rightarrow D$, and $D \rightarrow A$.
- BCD is also a key for R
- Therefore R is in 3NF
- However, A and D values may still occur redundantly

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d1
a1	b2	c3	d1
a2	b2	c3	d2

- An example where the relation is in 3NF but not in BCNF

Winter 2003

24

Example

- This example illustrates that it is not always possible to obtain a decomposition that is lossless and dependency preserving
- $R(\text{Student, Teacher, Subject})$
 - $\text{Teacher} \rightarrow \text{Subject}$
 - $\text{Student, Subject} \rightarrow \text{Teacher}$
 - $R_1(\text{Teacher, Subject}), R_2(\text{Student, Teacher})$
 - Lossless but the dependency $\text{Student, Subject} \rightarrow \text{Teacher}$ is not in $(\mathcal{F}_{R_1} \cup \mathcal{F}_{R_2})^+$

Winter 2003

25

BCNF Decomposition Algorithm

```
result:= {R}
done:= false
compute F+
while (not done) do
  if (there is a scheme S in result that is not in BCNF)
    then begin
      let  $X \rightarrow Y$  be a nontrivial functional dependency
        that holds on S such that  $X \rightarrow S$  is not in F+
        and X and Y are disjoint
      result:= (result-S)  $\cup$  (S-Y)  $\cup$  {XY}
    end
  else done:=true
end
```

The result is, in general, sensitive to the order in which the dependencies are considered

Winter 2003

26

Example

- Consider the relation schema $R(A, B, C, D, E)$ with FDs $A \rightarrow BCDE$, $C \rightarrow D$, and $CE \rightarrow B$
- result = $\{ABCDE\}$
- $ABCDE$ is not in BCNF because $CE \rightarrow B$ and CE is not a superkey
- result = $\{CEB, ACDE\}$
- CEB is in BCNF but $ACDE$ is not because $C \rightarrow D$ and C is not a superkey
- result = $\{CEB, CD, ACE\}$

Winter 2003

27

Another example

- Consider the relation schema $R(A, B, C, D, E)$ with FDs $A \rightarrow BCDE$, $BD \rightarrow A$, and $CE \rightarrow B$
- Result = $\{ABCDE\}$
- $ABCDE$ is not in BCNF because $CE \rightarrow B$ and CE is not a superkey
- Result = $\{CEB, ACDE\}$
- CEB is in BCNF and $ACDE$ is in BCNF
- The result is not dependency-preserving because of the dependency $BD \rightarrow A$

Winter 2003

28

Another example

- One possible get-around:
 - Materialize the relation BDA as well. Therefore we have three relations: CEB, ACDE, and BDA.
 - Each of these three relations is in BCNF and $\pi_{BDA} (CEB \bowtie ACDE) = BDA$
 - Although each relation is in BCNF (no redundancy in each relation), there is redundancy as a whole since the relation BDA can be obtained from $\pi_{BDA} (CEB \bowtie ACDE)$
- For practical reasons, 3NF is the normal form that people desire
- Every schema can be decomposed into a set of 3NF relations such that the decomposition is lossless **and** dependency preserving

Winter 2003

29

3NF Synthesis Algorithm

Let \mathcal{F} be a minimal cover

$S = \{ \}$

for each $X \rightarrow Y$ in \mathcal{F} do

if none of the schemes in S contains XY
then add XY to S

if $R - \text{attr}(S)$ is non-empty

then add $(R - \text{attr}(S))$ to S

if none of the schemes in S contains a candidate key for R
then add Z to S where Z is a candidate key

The result is, in general, not unique because \mathcal{F} is not unique

Winter 2003

30

Example

- $R(A, B, C)$ with FDs $A \rightarrow B$ and $C \rightarrow B$
- Minimal cover = $\{A \rightarrow B \text{ and } C \rightarrow B\}$
- $S = \{\}$
- $S = \{AB\}$
- $S = \{AB, CB\}$
- Since a key for R is AC , none of the schemes in S contains a candidate key. Therefore,
 - $S = \{AB, CB, AC\}$
- Lossless
- Dependency-preserving
- Is it in BCNF?

Winter 2003

31

Other dependencies

- Multi-valued dependencies
- Join dependencies
- Inclusion dependencies

Winter 2003

32

Summary

- For every relation schema and a set of FDs, there is always a lossless join, dependency-preserving decomposition into 3NF
- There is always a lossless-join decomposition into BCNF. Some dependencies, however, may not be preserved
- $BCNF \subseteq 3NF \subseteq 2NF \subseteq 1NF$
- Both the BCNF decomposition algorithms and 3NF synthesis algorithm produce non-unique results