
Announcements

Surveys

- Midterm survey is opened: 20 questions
- Deadline: Friday, 25 Oct 2019
- *Benefit:*
 - Allow multiple attempt for Assignment 04
 - Assignment 04 MRQ will show the mark immediately so it might be useful to redo the attempt

Assignment 04

- *Release:* Sunday, 27 Oct 2019
- *Last download:* Friday, 8 Nov 2019 at 11.50pm
- *Deadline:* Saturday, 9 Nov 2019 at 11.59pm
- cs2102-fd-nf.herokuapp.com will be closed from Sat, 26 Oct 2019

Announcements

Assignment 03 and Midterm Marking

- Should finish by end of week
- I will run **plagiarism checker** on both (since it's programming)
- If you admit to plagiarism before I run the checker (Thu, 24 Oct 2019)
 - You will only get minimal penalty (i.e., 0 mark for assignment3 and/or midterm)
- If you are found to have plagiarized but did not admit
 - Will be reported to Registrar Office

Plagiarism policy

*Any student found to have **committed** or **aided** and **abetted** the offence of plagiarism may be subject to disciplinary action. In addition, the student may receive a reduced grade (possibly even zero mark) for the relevant academic assignment, project, or thesis; and could receive a failed grade for the module. Any student caught plagiarizing will be required to retain the plagiarized module as graded, and will not be allowed to exercise the S/U option for that module.*

- This includes
 - Giving your answers to a friend
 - Using your friends' answers as your own

CS2102

Database Systems

Slides adapted from Prof. Chan Chee Yong

LECTURE 10

NORMAL FORMS

Decomposition

Definition

- The **decomposition of schema** R is a set of schemas $\{R_1, R_2, \dots, R_n\}$ (called **fragments**) such that
 - $R_i \subseteq R$ for each R_i
 - Each fragment is **simpler** than the original schema
 - $R = R_1 \cup R_2 \cup \dots \cup R_n$
 - No attributes are **missing**
- Consider a relation r of R , the decomposition of r into $\{r_1, r_2, \dots, r_n\}$ is
 - $r_i = \pi_{R_i}(r)$
 - **Projection operation!**

Lossless-join decomposition

Definition

- $r = \pi_{R_1}(r) \bowtie \pi_{R_2}(r) \bowtie \cdots \bowtie \pi_{R_n}(r)$
- **Lemma 1**
 - $r \subseteq \pi_{R_1}(r) \bowtie \pi_{R_2}(r) \bowtie \cdots \bowtie \pi_{R_n}(r)$
- **Theorem 1**
 - $F \models R_1 \cap R_2 \rightarrow R_1$ OR $F \models R_1 \cap R_2 \rightarrow R_2 \Rightarrow$ lossless
- **Corollary 1**
 - If $a \rightarrow b$ is a completely non-trivial FD that holds on R , then the decomposition of R into $\{R - b, ab\}$ is a lossless-join decomposition
- **Theorem 2**
 - If $\{R_1, R_2, \dots, R_n\}$ is a lossless-join decomposition of R , and $\{R_{1,1}, R_{1,2}\}$ is a lossless-join decomposition of R_1 then $\{R_{1,1}, R_{1,2}, R_2, \dots, R_n\}$ is a lossless join decomposition of R

Dependency-preserving decomposition

Projection

- $F_a = \{ b \rightarrow c \in F^+ \mid bc \subseteq a \}$

What?

- $(F_{R_1} \cup F_{R_2} \cup \dots \cup F_{R_n})$ is equivalent to F
- $(F_{R_1} \cup F_{R_2} \cup \dots \cup F_{R_n}) \equiv F$
- $(F_{R_1} \cup F_{R_2} \cup \dots \cup F_{R_n})^+ = F^+$
- $(F_{R_1} \cup F_{R_2} \cup \dots \cup F_{R_n}) \models F \wedge F \models (F_{R_1} \cup F_{R_2} \cup \dots \cup F_{R_n})$

Lemma 2

- For every decomposition $\{R_1, R_2, \dots, R_n\}$ of R
- $F \models (F_{R_1} \cup F_{R_2} \cup \dots \cup F_{R_n})$
- By definition, $F_{R_i} = \{ b \rightarrow c \in F^+ \mid bc \subseteq R_i \}$
- For all $b \rightarrow c$ in F_{R_i} , we also have $b \rightarrow c$ in F^+

Algorithms

Algorithm #1

- **Input** A set of attributes $a \subseteq R$ and a set of FDs F on R
- **Output** a^+ w.r.t. F

Algorithm #2

- **Input** A set of FDs F
- **Output** A minimal cover for F

Algorithm #3

- **Input** A set of attributes $a \subseteq R$ and a set of FDs F on R
- **Output** F_a

Algorithm #4

- **Input** A decomposition $\{R_1, \dots, R_n\}$ of R with FDs F
- **Output** *YES* (if dependency-preserving) or *NO* (otherwise)

- Preliminary
 - Boyce-Codd normal form
Algorithm
 - 3rd normal form
Algorithm
-

Overview

- Preliminary
- Boyce-Codd normal form
Algorithm
- 3rd normal form
Algorithm

Preliminary

Preliminary

Normal forms

- A **normal form** restricts the set of data dependencies that are allowed to hold on a schema to **avoid certain undesirable redundancy and update problems** in database
- Anomalies:
 - Insertion anomaly
 - Deletion anomaly
 - Update anomaly

Preliminary

Normal forms

- A **normal form** restricts the set of data dependencies that are allowed to hold on a schema to **avoid certain undesirable redundancy and update problems** in database
- Anomalies:
 - Insertion anomaly
 - Deletion anomaly
 - Update anomaly
- ❖ **Occurrences of anomalies is related to FD**
 - Consider $R(A, B, C)$ without any FD
 - Any triple $\langle a, b, c \rangle$ is valid
 - Consider $R(A, B, C)$ with FDs $F = \{A \rightarrow B\}$
 - $\langle a, b_1, c \rangle$ should not be with $\langle a, b_2, c \rangle$
 - But if A is a PK, $\langle a, b, c_1 \rangle$ cannot be with $\langle a, b, c_2 \rangle$

Preliminary

Normal forms

- A **normal form** restricts the set of data dependencies that are allowed to hold on a schema to **avoid certain undesirable redundancy and update problems** in database
- Anomalies:
 - Insertion anomaly
 - Deletion anomaly
 - Update anomaly
- ❖ **Occurrences of anomalies is related to FD**
 - Consider $R(A, B, C)$ with FDs $F = \{A \rightarrow B\}$
 - There will be no anomalies if we have $R_1(A, B)$ and $R_2(B, C)$
 - $\langle a, b_1 \rangle$ cannot be with $\langle a, b_2 \rangle$
 - Any pair $\langle b, c \rangle$ will be valid
 - R_1 is basically an entity/relationship

- Preliminary
- Boyce-Codd normal form
Algorithm
- 3rd normal form
Algorithm

Boyce-Codd normal form

Normal forms

Boyce-Codd normal form (BCNF)

- Definition
 - Consider a relation schema R with FDs F
 - R is in **Boyce-Codd normal form (BCNF)** if for every FD $a \rightarrow A$ in F either
 1. $a \rightarrow A$ is **trivial**, **OR**
 2. a is a **superkey** of R
 - An FD $a \rightarrow b$ that holds on R is said to **violate** BCNF if
 - $a \rightarrow b$ is **non-trivial**, **AND**
 - a is **not a superkey** of R
- Decomposition
 - A decomposition $\{R_1, R_2, \dots, R_n\}$ of R (with FDs F) is in BCNF if
 - Each $R_i \in \{R_1, R_2, \dots, R_n\}$ is in BCNF (w.r.t. F_{R_i})

Normal forms

Boyce-Codd normal form (BCNF)

- Definition
 - Consider a relation schema R with FDs F
 - R is in **Boyce-Codd normal form (BCNF)** if for every FD $a \rightarrow A$ in F either
 1. $a \rightarrow A$ is **trivial**, **OR**
 2. a is a **superkey** of R
- Check
 - To check if any R_i is in BCNF, check if there exists some non-trivial FD f which holds on R_i that **violates** BCNF
 1. If F is the set of FDs that **hold on R_i**
 \Rightarrow we check for any **violating non-trivial FD in F**
 2. If F is the set of FDs that **hold on R and R_i is a decomposed relation schema of R**
 \Rightarrow we check for any **violating non-trivial FD in F_{R_i}**

Normal forms

Boyce-Codd normal form (BCNF)

- Question
 - Consider a relation schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Is R in BCNF?

Normal forms

Boyce-Codd normal form (BCNF)

- Question
 - Consider a relation schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Is R in BCNF?
- Answer
 - Consider $A \rightarrow B$
 - $A \rightarrow B$ is non-trivial
 - $A \rightarrow B$ is not a superkey of R
 - $A^+ = AB \neq R$ w.r.t. F
 - ❖ R is not in BCNF

Normal forms

Boyce-Codd normal form (BCNF)

- Question
 - Consider a relation schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Let $\{R_1(A, B), R_2(A, C, D, E)\}$ be a schema decomposition of R
 - Is R_2 in BCNF?

Normal forms

Boyce-Codd normal form (BCNF)

- Question
 - Consider a relation schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Let $\{R_1(A, B), R_2(A, C, D, E)\}$ be a schema decomposition of R
 - Is R_2 in BCNF?
- Answer
 - Consider $A \rightarrow B$
 - $A \rightarrow B$ does not hold in R_2

Normal forms

Boyce-Codd normal form (BCNF)

- Question
 - Consider a relation schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Let $\{R_1(A, B), R_2(A, C, D, E)\}$ be a schema decomposition of R
 - Is R_2 in BCNF?
- Answer
 - Consider $A \rightarrow B$
 - $A \rightarrow B$ does not hold in R_2
 - Consider $BC \rightarrow D$
 - $BC \rightarrow D$ does not hold in R_2

Normal forms

Boyce-Codd normal form (BCNF)

- Question
 - Consider a relation schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Let $\{R_1(A, B), R_2(A, C, D, E)\}$ be a schema decomposition of R
 - Is R_2 in BCNF?
- Answer
 - Consider $A \rightarrow B$
 - $A \rightarrow B$ does not hold in R_2
 - Consider $BC \rightarrow D$
 - $BC \rightarrow D$ does not hold in R_2
 - ❖ R_2 is in BCNF

Normal forms

Boyce-Codd normal form (BCNF)

- Question
 - Consider a relation schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Let $\{R_1(A, B), R_2(A, C, D, E)\}$ be a schema decomposition of R
 - Is R_2 in BCNF?
- Answer
 - Consider $AC \rightarrow D$
 - $AC \rightarrow D \in F_{R_2}$ (since $AC^+ = ACD$ w.r.t. F_{R_2})
 - $AC \rightarrow D$ is non-trivial
 - $AC \rightarrow D$ is not a superkey of R_2
 - ❖ R_2 is not in BCNF

Normal forms

Boyce-Codd normal form (BCNF)

- Lemma 3
 - For any relation schema R with exactly two attributes, R is in BCNF
- Proof
 - Let A, B be the attributes
 - We consider 4 cases
 1. $F = \{ \}$
 - All FDs are trivial \Rightarrow no violation of BCNF can occur
 2. $F = \{A \rightarrow B\}$
 - $A \rightarrow B$ is the only non-trivial FD and A is superkey
 3. $F = \{B \rightarrow A\}$
 - $B \rightarrow A$ is the only non-trivial FD and B is superkey
 4. $F = \{A \rightarrow B, B \rightarrow A\}$

Normal forms

Boyce-Codd normal form (BCNF)

- Lemma 3
 - For any relation schema R with exactly two attributes, R is in BCNF
- Improved check
 - To check if any R_i is in BCNF, check if there exists some non-trivial FD f which holds on R_i that **violates** BCNF
 - If R_i has exactly two attributes then it is in BCNF
 - Otherwise, check for any violating FDs in F_{R_i} (if R_i not decomposed then $R_i = R$ and $F_{R_i} = F^+$)
 - For each $a \rightarrow b \in F_{R_i}$
 - **If $a \rightarrow b$ is trivial, then check next $a' \rightarrow b'$**
 - **If a is a superkey, then check next $a' \rightarrow b'$**
 - Otherwise, we have found a violation

Normal forms

Boyce-Codd normal form (BCNF)

- Algorithm #5
 - Input F is a set of FDs that hold on schema R
and R_i is either R or a decomposed schema of R
 - Output A completely non-trivial FD that violates BCNF
if R_i is not in BCNF; otherwise *null*
1. if (R_i has exactly 2 attributes)
 2. return null
 3. for each ($a \subseteq R$ such that $a \neq \emptyset$)
 4. let $X = a^+ \cap R_i$ w.r.t. F // compute $a \rightarrow X$
 5. if ($a \subset X \subset R_i$)
 6. return $a \rightarrow (X - a)$
 7. return null

$a \subset X$ implies
non-trivial
(since trivial FD
means $a = X$)

$X \subset R_i$ implies
non-superkey
(superkey
means $X = R_i$)

Normal forms

Boyce-Codd normal form (BCNF)

- Decomposition
 - Given a relation schema R that violates BCNF, can we construct a decomposed schema $\{R_1, \dots, R_n\}$ that is in BCNF?

Normal forms

Boyce-Codd normal form (BCNF)

- Decomposition
 - Given a relation schema R that violates BCNF, can we construct a decomposed schema $\{R_1, \dots, R_n\}$ that is in BCNF?
- Trivial!
 - Let each $R_i \in \{R_1, \dots, R_n\}$ consists of exactly 2 attributes
 - Not guaranteed to be lossless-join
 - Not guaranteed to be dependency-preserving

Normal forms

Boyce-Codd normal form (BCNF)

- Decomposition
 - Given a relation schema R that violates BCNF, can we construct a decomposed schema $\{R_1, \dots, R_n\}$ that is in BCNF?
 - Can we guarantee lossless-join & dependency-preserving?

Normal forms

Boyce-Codd normal form (BCNF)

- Decomposition
 - Given a relation schema R that violates BCNF, can we construct a decomposed schema $\{R_1, \dots, R_n\}$ that is in BCNF?
 - Can we guarantee lossless-join & dependency-preserving?
 - We can guarantee lossless-join
 - We cannot guarantee dependency-preserving

Normal forms

Boyce-Codd normal form (BCNF)

- Decomposition
 - Given a relation schema R that violates BCNF, can we construct a decomposed schema $\{R_1, \dots, R_n\}$ that is in BCNF?
 - Can we guarantee lossless-join & dependency-preserving?
- Idea
 - Consider R , we have 2 cases
 - R is in BCNF \Rightarrow we are done
 - R is not in BCNF
 - There's a completely non-trivial FD $a \rightarrow b$ that violates BCNF
 - Decompose R into $\{R_1(R - b), R_2(ab)\}$
 - Check if R_1 and R_2 violates BCNF and decompose as necessary

Normal forms

Boyce-Codd normal form (BCNF)

- Algorithm #6
 - Input Schema R with FDs F
 - Output A lossless BCNF decomposition of R
1. initialize $\delta = \emptyset$; $i = 1$; $\theta = \{R\}$
 2. while ($\theta \neq \emptyset$)
 3. remove some R' from θ
 4. let $f = \text{Algorithm\#5}(F, R')$
 5. if ($f = \text{null}$) then $\delta = \delta \cup \{R'\}$ // in BCNF
 6. else
 7. let f be $a \rightarrow b$ // completely non-trivial
 8. let $c = R' - b$
 9. $\theta = \theta \cup \{R_i(ab), R_{i+1}(c)\}$ // decompose
 10. $i = i + 2$
 11. return δ

Normal forms

Boyce-Codd normal form (BCNF)

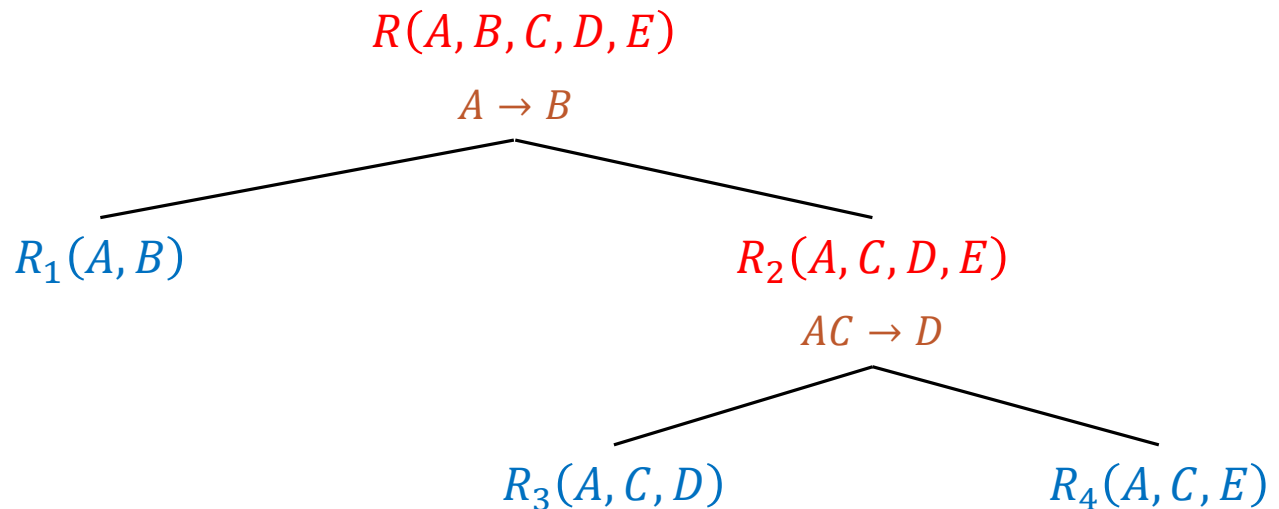
- Question
 - Consider a schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Find a BCNF decomposition of R
- Simplified steps
 - $\{R_1(A, B), R_3(A, C, D), R_4(A, C, E)\}$
 - Lossless-join?

$R(A, B, C, D, E)$

Normal forms

Boyce-Codd normal form (BCNF)

- Question
 - Consider a schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Find a BCNF decomposition of R
- Simplified steps
 - $\{R_1(A, B), R_3(A, C, D), R_4(A, C, E)\}$
 - Lossless-join?



Normal forms

Boyce-Codd normal form (BCNF)

- Question
 - Consider a schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Find a BCNF decomposition of R
- Simplified steps
 - $\{R_1(A, B), R_3(A, C, D), R_4(A, C, E)\}$
 - Dependency-preserving?

Normal forms

Boyce-Codd normal form (BCNF)

- Question
 - Consider a schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Find a BCNF decomposition of R
- Simplified steps
 - $\{R_1(A, B), R_3(A, C, D), R_4(A, C, E)\}$
 - Dependency-preserving?
 - $F_{R_1} = \{A \rightarrow B\}$
 - $F_{R_3} = \{AC \rightarrow D\}$
 - $F_{R_4} = \{ \}$
 - $BC^+ = BC$ w.r.t. $(F_{R_1} \cup F_{R_3} \cup F_{R_4})$
 - ❖ $(F_{R_1} \cup F_{R_3} \cup F_{R_4}) \not\models BC \rightarrow D$

Normal forms

Boyce-Codd normal form (BCNF)

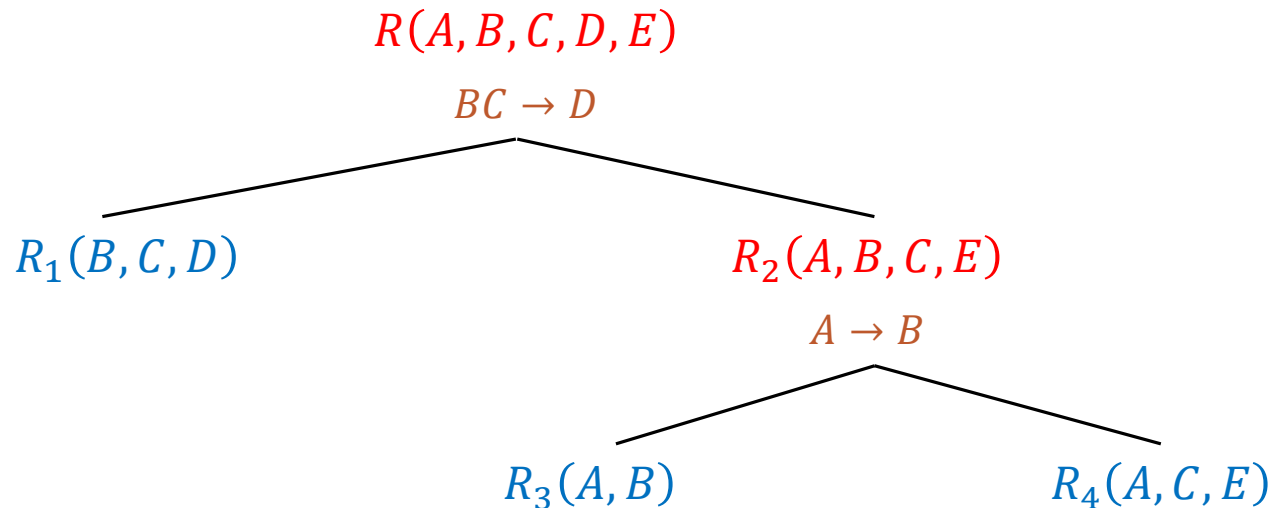
- Question
 - Consider a schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Find a BCNF decomposition of R
- Simplified steps
 - $\{R_1(B, C, D), R_3(A, B), R_4(A, C, E)\}$
 - Lossless-join?

$R(A, B, C, D, E)$

Normal forms

Boyce-Codd normal form (BCNF)

- Question
 - Consider a schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Find a BCNF decomposition of R
- Simplified steps
 - $\{R_1(B, C, D), R_3(A, B), R_4(A, C, E)\}$
 - Lossless-join?



Normal forms

Boyce-Codd normal form (BCNF)

- Question
 - Consider a schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Find a BCNF decomposition of R
- Simplified steps
 - $\{R_1(B, C, D), R_3(A, B), R_4(A, C, E)\}$
 - Dependency-preserving?

Normal forms

Boyce-Codd normal form (BCNF)

- Question
 - Consider a schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Find a BCNF decomposition of R
- Simplified steps
 - $\{R_1(B, C, D), R_3(A, B), R_4(A, C, E)\}$
 - Dependency-preserving?
 - $F_{R_1} = \{BC \rightarrow D\}$
 - $F_{R_3} = \{A \rightarrow B\}$
 - $F_{R_4} = \{ \}$
 - $(F_{R_1} \cup F_{R_3} \cup F_{R_4}) = \{A \rightarrow B, BC \rightarrow D\} = F$
 - ❖ $(F_{R_1} \cup F_{R_3} \cup F_{R_4}) \equiv F$

Normal forms

Boyce-Codd normal form (BCNF)

- Algorithm #6
 - Consider R , we have 2 cases
 - R is in BCNF \Rightarrow we are done
 - R is not in BCNF
 - There's a completely non-trivial FD $a \rightarrow b$ that violates BCNF
 - Decompose R into $\{R_1(R - b), R_2(ab)\}$
 - Check if R_1 and R_2 violates BCNF and decompose as necessary

Normal forms

Boyce-Codd normal form (BCNF)

- Algorithm #6
 - Consider R , we have 2 cases
 - R is in BCNF \Rightarrow we are done
 - R is not in BCNF
 - There's a completely non-trivial FD $a \rightarrow b$ that violates BCNF
 - Decompose R into $\{R_1(R - b), R_2(ab)\}$
 - Check if R_1 and R_2 violates BCNF and decompose as necessary
- Properties
 - The algorithm will terminate
 - Each step, for R violating BCNF, it will be decomposed
 - Each decomposition reduces the number of attributes
 - Once the number of attributes is 2, it will stop
 - At the worst-case, each fragment will have exactly 2 attributes

Normal forms

Boyce-Codd normal form (BCNF)

- Algorithm #6
 - Consider R , we have 2 cases
 - R is in BCNF \Rightarrow we are done
 - R is not in BCNF
 - There's a completely non-trivial FD $a \rightarrow b$ that violates BCNF
 - Decompose R into $\{R_1(R - b), R_2(ab)\}$
 - Check if R_1 and R_2 violates BCNF and decompose as necessary
- Properties
 - The decomposition is a lossless-join decomposition
 - Each decomposition is based on $a \rightarrow b$, note that
 - $a \rightarrow b$ is completely non trivial
 - The fragments are $\{R_1(R - b), R_2(ab)\}$
 - By corollary 1, it is a lossless-join decomposition

Normal forms

Boyce-Codd normal form (BCNF)

- Algorithm #6
 - Consider R , we have 2 cases
 - R is in BCNF \Rightarrow we are done
 - R is not in BCNF
 - There's a completely non-trivial FD $a \rightarrow b$ that violates BCNF
 - Decompose R into $\{R_1(R - b), R_2(ab)\}$
 - Check if R_1 and R_2 violates BCNF and decompose as necessary
- Properties
 - The decomposition may not be dependency-preserving
 - Consider a schema $R(A, B, C)$ with FDs $F = \{A \rightarrow B, BC \rightarrow A\}$
 - Keys are $\{AC\}$ and $\{BC\}$
 - R is not in BCNF because $A \rightarrow B$ and A is not a superkey
 - Decomposition into $\{R_1(A, B), R_2(B, C)\}$ does not preserve $BC \rightarrow A$

Normal forms

Boyce-Codd normal form (BCNF)

- Algorithm #6
 - Consider R , we have 2 cases
 - R is in BCNF \Rightarrow we are done
 - R is not in BCNF
 - There's a completely non-trivial FD $a \rightarrow b$ that violates BCNF
 - Decompose R into $\{R_1(R - b), R_2(ab)\}$
 - Check if R_1 and R_2 violates BCNF and decompose as necessary
- Properties
 - The algorithm will **terminate**
 - The decomposition is a **lossless-join** decomposition
 - The decomposition may **not be dependency-preserving**

Normal forms

Boyce-Codd normal form (BCNF)

- Properties
 - The algorithm will **terminate**
 - The decomposition is a **lossless-join** decomposition
 - The decomposition may **not be dependency-preserving**
- What went wrong?
 - We want non-redundancy
 - Might not be feasible in the case of overlapping key
 - Consider a schema $R(A, B, C)$ with FDs $F = \{A \rightarrow B, BC \rightarrow A\}$
 - Keys are $\{AC\}$ and $\{BC\} \Rightarrow$ overlap
 - The attributes $\{AB\}$ is an entity due to $A \rightarrow B$
 - The attributes $\{ABC\}$ is also an entity due to $BC \rightarrow A$
 - Both entities need to exist!
 - But if $R_i(A, B, C)$ exists, $A \rightarrow B$ violated BCNF

Normal forms

Boyce-Codd normal form (BCNF)

- Example of non dependency-preserving
 - Consider $R(\text{Course}, \text{Prof}, \text{Time})$ such that
 - Every Course is managed by exactly one Prof
 - $\text{Course} \rightarrow \text{Prof}$
 - A Prof cannot teach two Course at the same Time
 - $(\text{Prof}, \text{Time}) \rightarrow \text{Course}$
 - Keys: $\{\text{Course}, \text{Time}\}$ and $\{\text{Prof}, \text{Time}\}$
 - R is not in BCNF because $\text{Course} \rightarrow \text{Time}$ and Course is not a superkey of R
 - The decomposition $\{R_1(\text{Course}, \text{Prof}), R_2(\text{Course}, \text{Time})\}$ does not preserve $(\text{Prof}, \text{Time}) \rightarrow \text{Course}$

- Preliminary
 - Boyce-Codd normal form
Algorithm
 - 3rd normal form
Algorithm
-

3rd normal form

Normal forms

3rd normal form (3NF)

- Definition
 - Consider a relation schema R with FDs F
 - R is in 3rd normal form (BCNF) if for every FD $a \rightarrow A$ in F either
 1. $a \rightarrow A$ is trivial, OR
 2. a is a superkey of R , OR
 3. A is a prime attribute
 - An FD $a \rightarrow b$ that holds on R is said to violate 3NF if
 - $a \rightarrow b$ is non-trivial, AND
 - a is not a superkey of R , AND
 - A is a non-prime attribute
- Decomposition
 - A decomposition $\{R_1, R_2, \dots, R_n\}$ of R (with FDs F) is in 3NF if
 - Each $R_i \in \{R_1, R_2, \dots, R_n\}$ is in 3NF (w.r.t. F_{R_i})

Normal forms

3rd normal form (3NF)

- Definition
 - Consider a relation schema R with FDs F
 - R is in 3rd normal form (BCNF) if for every FD $a \rightarrow A$ in F either
 1. $a \rightarrow A$ is trivial, OR
 2. a is a superkey of R , OR
 3. A is a prime attribute
- Check
 - To check if any R_i is in 3NF, check if there exists some non-trivial FD f which holds on R_i that violates 3NF
 1. If F is the set of FDs that hold on R_i
 \Rightarrow we check for any violating non-trivial FD in F
 2. If F is the set of FDs that hold on R and R_i is a decomposed relation schema of R
 \Rightarrow we check for any violating non-trivial FD in F_{R_i}

Normal forms

3rd normal form (3NF)

- Question
 - Consider a relation schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Is R in 3NF?

Normal forms

3rd normal form (3NF)

- Question
 - Consider a relation schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Is R in 3NF?
- Answer
 - Consider $A \rightarrow B$
 - $A \rightarrow B$ is non-trivial
 - $A \rightarrow B$ is not a superkey of R
 - B is not a prime attribute (key = $\{A, C, E\}$)
 - ❖ R is not in 3NF

Normal forms

3rd normal form (3NF)

- Question
 - Consider a relation schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Let $\{R_1(A, B), R_2(A, C, D, E)\}$ be a schema decomposition of R
 - Is R_2 in 3NF?

Normal forms

3rd normal form (3NF)

- Question
 - Consider a relation schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Let $\{R_1(A, B), R_2(A, C, D, E)\}$ be a schema decomposition of R
 - Is R_2 in 3NF?
- Answer
 - Consider $A \rightarrow B$
 - $A \rightarrow B$ does not hold in R_2
 - Consider $BC \rightarrow D$
 - $BC \rightarrow D$ does not hold in R_2
 - ❖ R_2 is in 3NF

Normal forms

3rd normal form (3NF)

- Question
 - Consider a relation schema $R(A, B, C, D, E)$ with FDs $F = \{A \rightarrow B, BC \rightarrow D\}$
 - Let $\{R_1(A, B), R_2(A, C, D, E)\}$ be a schema decomposition of R
 - Is R_2 in 3NF?
- Answer
 - Consider $AC \rightarrow D \in F_{R_2}$ (since $AC^+ = ACD$ w.r.t. F_{R_2})
 - $AC \rightarrow D$ is non-trivial
 - $AC \rightarrow D$ is not a superkey of R_2
 - D is not a prime attribute (key = $\{A, C, E\}$)
 - ❖ R_2 is not in 3NF

Normal forms

3rd normal form (3NF)

- Question
 - Consider a relation schema $R(A, B, C)$ with FDs $F = \{A \rightarrow B, BC \rightarrow A\}$
 - We know R is not in BCNF
 - Is R in 3NF?

Normal forms

3rd normal form (3NF)

- Question
 - Consider a relation schema $R(A, B, C)$ with FDs $F = \{A \rightarrow B, BC \rightarrow A\}$
 - We know R is not in BCNF
 - Is R in 3NF?
- Answer
 - Key are $\{AC\}$ and $\{BC\}$
 - Consider $A \rightarrow B \quad \Rightarrow B$ is a prime attribute
 - Consider $BC \rightarrow A \quad \Rightarrow BC$ is a superkey
 - ❖ R is in 3NF

Normal forms

3rd normal form (3NF)

- Decomposition
 - Given a relation schema R that violates BCNF, can we construct a decomposed schema $\{R_1, \dots, R_n\}$ that is in 3NF?

Normal forms

3rd normal form (3NF)

- Decomposition
 - Given a relation schema R that violates BCNF, can we construct a decomposed schema $\{R_1, \dots, R_n\}$ that is in 3NF?
- Trivial again!
 - Make every fragment exactly 2 attributes
 - Not guaranteed to be lossless-join
 - Not guaranteed to be dependency-preserving

Normal forms

3rd normal form (3NF)

- Decomposition
 - Given a relation schema R that violates BCNF, can we construct a decomposed schema $\{R_1, \dots, R_n\}$ that is in 3NF?
 - Can we guarantee lossless-join & dependency-preserving?

Normal forms

3rd normal form (3NF)

- Decomposition
 - Given a relation schema R that violates BCNF, can we construct a decomposed schema $\{R_1, \dots, R_n\}$ that is in 3NF?
 - Can we guarantee lossless-join & dependency-preserving?
 - We can guarantee lossless-join
 - We can guarantee dependency-preserving

Normal forms

3rd normal form (3NF)

- Decomposition
 - Given a relation schema R that violates BCNF, can we construct a decomposed schema $\{R_1, \dots, R_n\}$ that is in 3NF?
 - Can we guarantee lossless-join & dependency-preserving?
- Idea
 - Consider R with FDs F
 - Consider $a \rightarrow b \in F$
 - What is the property of $R_i(ab)$?
 - Can we ensure that there is no $a' \subseteq a$ such that
 - $a' \rightarrow B$
 - a' is not superkey
 - B is not a prime attribute
 - Yes: minimal cover

Normal forms

3rd normal form (3NF)

- Algorithm #7
- Input Schema R with FDs F which is a minimal cover
- Output A lossless and dependency-preserving 3NF decomposition of R

1. initialize $\delta = \emptyset$
2. apply union rule to combine FDs in F
3. let $G = \{f_1, f_2, \dots, f_n\}$ be the resultant set of FDs
4. for each (FD f_i of the form $a_i \rightarrow b_i$ in G)
5. create a relation schema $R_i(a_i b_i)$ for FD f_i
6. insert $R_i(a_i b_i)$ into δ
7. choose a key K of R and insert $R_{n+1}(K)$ into δ
8. remove redundant relation schema from δ
9. \Rightarrow delete R_i from δ if $\exists R_j \in \delta \cdot i \neq j \wedge R_i \subseteq R_j$
10. return δ

Normal forms

3rd normal form (3NF)

- Algorithm #7
 - Consider R with FDs F which is a minimal cover
 - Union rule into G
 - For each $a_i \rightarrow b_i \in G$ create $R_i(a_i b_i)$
 - Create $R_{n+1}(K)$ for any key K
 - Remove redundant relation schema
- Properties
 - The algorithm will terminate
 - There are limited $a \rightarrow b \in G$
 - The decomposition is a valid decomposition
 - Consider an attribute a not appearing in any FD
 - Then a must be part of all key K
 - This attribute is covered by R_{n+1} (not removed by redundancy check)

Normal forms

3rd normal form (3NF)

- Algorithm #7
 - Consider R with FDs F which is a minimal cover
 - Union rule into G
 - For each $a_i \rightarrow b_i \in G$ create $R_i(a_i b_i)$
 - Create $R_{n+1}(K)$ for any key K
 - Remove redundant relation schema
- Properties
 - The decomposition is a lossless-join decomposition
 - proof omitted

Normal forms

3rd normal form (3NF)

- Algorithm #7
 - Consider R with FDs F which is a minimal cover
 - Union rule into G
 - For each $a_i \rightarrow b_i \in G$ create $R_i(a_i b_i)$
 - Create $R_{n+1}(K)$ for any key K
 - Remove redundant relation schema
- Properties
 - The decomposition is dependency-preserving
 - Since each $a_i \rightarrow b_i$ is made into $R_i(a_i b_i)$
 - The FD is preserved by R_i

Normal forms

3rd normal form (3NF)

- Algorithm #7
 - Consider R with FDs F which is a minimal cover
 - Union rule into G
 - For each $a_i \rightarrow b_i \in G$ create $R_i(a_i b_i)$
 - Create $R_{n+1}(K)$ for any key K
 - Remove redundant relation schema
- Properties
 - The algorithm will **terminate**
 - The decomposition is a **valid decomposition**
 - The decomposition is a **lossless-join** decomposition
 - The decomposition is **dependency-preserving**

Summary

□ Normal forms

- Consider a relation schema R with FDs F

□ BCNF

- For every FD $a \rightarrow A$ in F_R , either

- $a \rightarrow A$ is trivial [OR] a is a superkey of R

- **Lemma 3:** Any relation schema R with exactly 2 attributes is in BCNF

- **Algorithm #5:** Checks BCNF violation; find a completely non-trivial FD violation

- **Algorithm #6:** Perform BCNF decomposition

- Check (algo5) and decompose on violation

- Properties: (a) lossless-join but (b) may not be dependency-preserving

□ 3NF (relaxation on BCNF)

- For every FD $a \rightarrow A$ in F_R , either

- $a \rightarrow A$ is trivial [OR] a is a superkey of R [OR] A is a prime attribute

- **Lemma 3*:** Any relation schema R in BCNF is also in 3NF

- **Algorithm #7:** Construct 3NF decomposition (constructive method)

- Properties: (a) lossless-join and (b) dependency-preserving