

COMP353 Databases

Schema Refinement: Minimal Bases (Canonical Covers)

1

Minimal Basis (Canonical Cover)

- Recall that the number of iterations to compute the closure of a set of attributes depends on the number of attributes
 - The complexity of some other algorithms which we will study (eg, decomposition algorithms) depend on the number of FD's
- To ease the situation, can we "minimize" F ?

2

Covers/bases

- Note that FD's on a relation may be represented in different but equivalent ways.
- Recall that, given two sets of FD's F and G on R , we say that:
 - " G follows from F ($F \models G$)", provided for every instance r of R , if r satisfies F , then r satisfies G . In this case, we may also say: " F implies G ", or " F covers G ", or " G is implied by F ".
- If $F \models G$ and $G \models F$ both hold, then we say that G and F are **equivalent** and denote this by $F \equiv G$. In this case, we may also say that F and G are covers for each other.
- Note that $F \equiv G$ iff $F^+ \equiv G^+$

3

Canonical Cover (minimal basis)

- Let F be a set of FD's. A **canonical cover** of F is a set G of FD's that satisfies the following conditions:
 - G is **equivalent** to F , that is, $G \equiv F$
 - Every FD in G has a single attribute on the right hand side.
 - G is **minimal**, that is, if we obtain a set H of FD's from G by deleting some FD's in G or by reducing the left hand side of some FD's, then H won't be equivalent to F (that is, $H \not\equiv F$)

4

Canonical Cover

- A canonical cover G is **minimal** in two respects:
- Every FD in G is "**required**" in order for G to be equivalent to F
 - Every FD in G is as "**small**" as possible, that is,
 - each attribute on the left hand side X is necessary.
 - Recall: the RHS of every FD in G is a single attribute

5

Computing Canonical Cover

Given a set F of FD's, how to compute a canonical cover G of F ?

- Step 1:** Put the FD's in the **simple** form (i.e., one attribute on the RHS)
 - Initialize $G := F$
 - Replace each FD $X \rightarrow A_1 A_2 \dots A_k$ in G with $X \rightarrow A_1, X \rightarrow A_2, \dots, X \rightarrow A_k$
- Step 2: Minimize** the left hand side X of every FD
 - E.g., if $AB \rightarrow C$ is in G , check if A or B on the LHS is redundant, i.e.,

$$(G - \{AB \rightarrow C\} \cup \{A \rightarrow C\})^+ \equiv F^+ \text{ or } (G - \{AB \rightarrow C\} \cup \{B \rightarrow C\})^+ \equiv F^+ ?$$
- Step 3: Delete** redundant FD's, if any
 - For each FD $X \rightarrow A$ in G , check if $(G - \{X \rightarrow A\})^+ \equiv F^+ ?$

6

Computing Canonical Cover

- $R = \{A, B, C, D, E, H\}$
- $F = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
- Step 1 – put FD's in the simple form
 - All present FD's are simple
 - $G = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$

7

Computing Canonical Cover

- $R = \{A, B, C, D, E, H\}$
- $F = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
- Step 2 – Check every FD to see if it is left reduced
 - For every FD $X \rightarrow A$ in G , check if the closure of a subset of X determines A . If so, remove the redundant attribute(s) from X

8

Computing Canonical Cover

- $R = \{A, B, C, D, E, H\}$
- $F = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
- $G = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
 - $A \rightarrow B$
→ obviously OK (no left redundancy)
 - $DE \rightarrow A$
 - $D^+ = D$
 - $E^+ = E$
 - OK (no left redundancy)

9

Computing Canonical Cover

- $R = \{A, B, C, D, E, H\}$
- $F = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
- $G = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
 - $BC \rightarrow E$
 - $B^+ = B$
 - $C^+ = C$
 - OK (no left redundancy)

10

Computing Canonical Cover

- $R = \{A, B, C, D, E, H\}$
- $F = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
- $G = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
 - $AC \rightarrow E$
 - $A^+ = AB$
 - $C^+ = C$
 - OK (no left redundancy)

11

Computing Canonical Cover

- $R = \{A, B, C, D, E, H\}$
- $F = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
- $G = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
 - $BCD \rightarrow A$
 - $B^+ = B$
 - $C^+ = C$
 - $D^+ = D$
 - $\{BC\}^+ = BCE$
 - $\{CD\}^+ = CD$
 - $\{BD\}^+ = BD$
 - OK (no left redundancy)

12

Computing Canonical Cover

- $R = \{A, B, C, D, E, H\}$
- $F = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
- $G = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
 - $AED \rightarrow B$
 - $A^* = AB$
 - E & D are redundant \rightarrow we can remove them from $AED \rightarrow B$
- $G = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, A \rightarrow B\}$
- $G = \{DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, A \rightarrow B\}$

13

Computing Canonical Cover

- $R = \{A, B, C, D, E, H\}$
- $F = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
- Step 3 – Find and remove redundant FDs
 - For every FD $X \rightarrow A$ in G
 - Remove $X \rightarrow A$ from G ; call the result G'
 - Compute X^* under G'
 - If $A \in X^*$, then $X \rightarrow A$ is redundant and hence we remove the FD $X \rightarrow A$ from G (that is, we rename G' to G)

14

Computing Canonical Cover

- $R = \{A, B, C, D, E, H\}$
- $F = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
- $G = \{DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, A \rightarrow B\}$
 - Remove $DE \rightarrow A$ from G
 - $G' = \{BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, A \rightarrow B\}$
 - Compute DE^* under G'
 - $\{DE\}^* = DE$ (computed under G')
 - Since $A \notin DE$, the FD $DE \rightarrow A$ is not redundant
 - $G = \{DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, A \rightarrow B\}$

15

Computing Canonical Cover

- $R = \{A, B, C, D, E, H\}$
- $F = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
- $G = \{DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, A \rightarrow B\}$
 - Remove $BC \rightarrow E$ from G
 - $G' = \{DE \rightarrow A, AC \rightarrow E, BCD \rightarrow A, A \rightarrow B\}$
 - Compute BC^* under G'
 - $\{BC\}^* = BC$
 - $\rightarrow BC \rightarrow E$ is not redundant
 - $G = \{DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, A \rightarrow B\}$

16

Computing Canonical Cover

- $R = \{A, B, C, D, E, H\}$
- $F = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
- $G = \{DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, A \rightarrow B\}$
 - Remove $AC \rightarrow E$ from G
 - $G' = \{DE \rightarrow A, BC \rightarrow E, BCD \rightarrow A, A \rightarrow B\}$
 - Compute $\{AC\}^*$ under G'
 - $\{AC\}^* = ACBE$
 - Since $E \in ACBE$, $AC \rightarrow E$ is redundant \rightarrow remove it from G
 - $G = \{DE \rightarrow A, BC \rightarrow E, BCD \rightarrow A, A \rightarrow B\}$

17

Computing Canonical Cover

- $R = \{A, B, C, D, E, H\}$
- $F = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
- $G = \{DE \rightarrow A, BC \rightarrow E, BCD \rightarrow A, A \rightarrow B\}$
 - Remove $BCD \rightarrow A$ from G
 - $G' = \{DE \rightarrow A, BC \rightarrow E, A \rightarrow B\}$
 - Compute BCD^* under G'
 - $\{BCD\}^* = BCDEA$
 - This FD is redundant \rightarrow remove it from G
 - $G = \{DE \rightarrow A, BC \rightarrow E, A \rightarrow B\}$

18

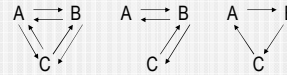
Computing Canonical Cover

- $R = \{A, B, C, D, E, F\}$
- $F = \{A \rightarrow B, DE \rightarrow A, BC \rightarrow E, AC \rightarrow E, BCD \rightarrow A, AED \rightarrow B\}$
- $G = \{DE \rightarrow A, BC \rightarrow E, A \rightarrow B\}$
 - Remove $A \rightarrow B$ from G
 - $G' = \{DE \rightarrow A, BC \rightarrow E\}$
 - Compute A^+ under G'
 - $A^+ = A$
 - This FD is not redundant (Another reason why need $A \rightarrow B$?)
 - $G = \{DE \rightarrow A, BC \rightarrow E, A \rightarrow B\}$
 - $\rightarrow G$ is a minimal cover for F

19

Several Canonical Covers Possible?

- Relation $R = \{A, B, C\}$ with $F = \{A \rightarrow B, A \rightarrow C, B \rightarrow A, B \rightarrow C, C \rightarrow B, C \rightarrow A\}$
- Several canonical covers exist
 - $G = \{A \rightarrow B, B \rightarrow A, B \rightarrow C, C \rightarrow B\}$
 - $G = \{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$



Can you find more ?

20

Computing a Canonical Cover

This example shows the order of steps 2 and 3 is important!

$R = \{A, B, C, D\}$ with $F = \{ABC \rightarrow D, AB \rightarrow C, D \rightarrow C\}$

1. (step 3; step 2): Doing step 3 first, no FD is redundant (Why?)
In step 2, $ABC \rightarrow D$ is left reduced to $AB \rightarrow D$. No more changes.
We thus obtain $G = \{AB \rightarrow D, AB \rightarrow C, D \rightarrow C\}$ which is equivalent to F but is **not** minimal! (The red FD is redundant!).
2. (step 2; step 3): Following our algorithm, in step 2 we get G above. In step 3, we remove the redundant FD in G .
This yields $\{AB \rightarrow D, D \rightarrow C\}$ which is equivalent to F and minimal.

21

How to Deal with Redundancy?

Relation Schema:

Star (name, address, representingFirm, spokesPerson)
 $F = \{ \text{name} \rightarrow \text{address}, \text{representingFirm}, \text{spokePerson}, \text{representingFirm} \rightarrow \text{spokesPerson} \}$

Relation Instance:

Name	Address	RepresentingFirm	SpokesPerson
Carrie Fisher	123 Maple	Star One	Joe Smith
Harrison Ford	789 Palm dr.	Star One	Joe Smith
Mark Hamill	456 Oak rd.	Movies & Co	Mary Johns

- We can **decompose** this relation into two smaller relations

22

How to Deal with Redundancy?

Given the relation schema below:

Star (name, address, representingFirm, spokesperson) with

$F = \{ \text{name} \rightarrow \text{address}, \text{representingFirm}, \text{spokePerson}, \text{representingFirm} \rightarrow \text{spokesPerson} \}$

Decompose Star into the following 2 relations:

Star (name, address, representingFirm)
 with $F_1 = \{ \text{name} \rightarrow \text{address}, \text{representingFirm} \}$

and

Firm (representingFirm, spokesPerson)
 with $F_2 = \{ \text{representingFirm} \rightarrow \text{spokesPerson} \}$

23

How to Deal with Redundancy?

Instance of Star before decomposition:

Name	Address	RepresentingFirm	Spokesperson
Carrie Fisher	123 Maple	Star One	Joe Smith
Harrison Ford	789 Palm dr.	Star One	Joe Smith
Mark Hamill	456 Oak rd.	Movies & Co	Mary Johns

The instance after the decomposition:

Name	Address	RepresentingFirm	RepresentingFirm	Spokesperson
Carrie Fisher	123 Maple	Star One	Star One	Joe Smith
Harrison Ford	789 Palm dr.	Star One	Movies & Co	Joe Smith
Mark Hamill	456 Oak rd.	Movies & Co	Movies & Co	Mary Johns

24

Decomposition

- A **decomposition** of a relation schema **R** consists of replacing **R** by two or more non-empty relation schemas such that each one is a subset of **R** and together they include all attributes of **R**. Formally, $\mathbf{R} = \{R_1, \dots, R_m\}$ is a decomposition if all conditions below hold:
 - $R_i \neq \emptyset$, for all i in $\{1, \dots, m\}$
 - $R_1 \cup \dots \cup R_m = R$
 - $R_i \neq R_j$, when $i \neq j$
 - When $m = 2$, the decomposition $\mathbf{R} = \{R_1, R_2\}$ is called **binary**
 - Not every decomposition of **R** is "desirable". Why?
 - Properties of a decomposition?
 - (1) Lossless-join – this is a **must**
 - (2) Dependency-preserving – this is **desirable**
- Explanation follows ...

25

Example

Relation Instance:

A	B	C
1	2	3
4	2	5

Decomposed into:

A	B
1	2
4	2

B	C
2	3
2	5

To "recover" information, we join the relations:

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

Why do we get new tuples?

26

Lossless-Join Decomposition

- R** is a relation schema and **F** is a set of FD's over **R**.
A binary decomposition of **R** into relation schemas **R₁** and **R₂** with attribute sets **X** and **Y** is said to be a **lossless-join decomposition with respect to F**, if for every instance **r** of **R** that satisfies **F**, we have $\pi_X(r) \bowtie \pi_Y(r) = r$
- Thm:** Let **R** be a relation schema and **F** a set of FD's on **R**. A binary decomposition of **R** into **R₁** and **R₂** with attribute sets **X** and **Y** is lossless if $X \cap Y \rightarrow X$ or $X \cap Y \rightarrow Y$, i.e., this binary decomposition is lossless if the common attributes of **X** and **Y** form a key of **R₁** or **R₂**

27

Example: Lossless-join

Relation Instance:

A	B	C
1	2	3
4	2	3

Decomposed into:

A	B
1	2
4	2

B	C
2	3

 $F = \{B \rightarrow C\}$ To recover the original relation **r**, we join the two relations:

A	B	C
1	2	3
4	2	3

No new tuples !

28

Example: Dependency Preservation

Relation Instance:

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

 $F = \{B \rightarrow C, B \rightarrow D, A \rightarrow D\}$

Decomposed into:

A	B
1	2
4	3

B	C	D
2	5	7
3	6	8

Can we enforce $A \rightarrow D$?
How ?

29

Dependency-Preserving Decomposition

- A **dependency-preserving** decomposition allows us to enforce every FD (on each insertion of a tuple or when modifying a tuple) by examining just **one single relation instance**
- Let **R** be a relation schema that is decomposed into two schemas with attribute sets **X** and **Y**, and let **F** be a set of FD's over **R**. The **projection of F on X** (denoted by F_X) is the set of FD's in F^+ that follow from **F** and involve only attributes in **X**
 - Recall that a FD $U \rightarrow V$ in F^+ is in F_X if all the attributes in **U** and **V** are in **X**; In this case, we say this FD is "**relevant**" to **X**
- The decomposition of $\langle R, F \rangle$ into two schemas with attribute sets **X** and **Y** is **dependency-preserving** if $(F_X \cup F_Y)^+ \equiv F^+$

30

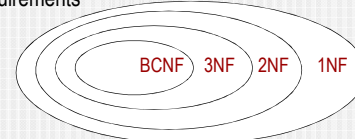
Normal Forms

- Given a relation schema R , we must be able to determine whether it is “good” or we need to decompose it into smaller relations, and if so, how?
- To address these issues, we need to study **normal forms**
- If a relation schema is in one of these normal forms, we know that it is in some “good” shape in the sense that *certain kinds of problems (related to redundancy) cannot arise*

31

Normal Forms

- The normal forms based on FD's are
 - First normal form (1NF)
 - Second normal form (2NF)
 - Third normal form (3NF)
 - Boyce-Codd normal form (BCNF)
- These normal forms have increasingly restrictive requirements



32

Third Normal Form (3NF)

Let R be a relation schema, F a set of FD's on R , $X \subseteq R$, and $A \in R$.

- We say R w.r.t. F is in 3NF (**third normal form**), if for every FD $X \rightarrow A$ in F , at least one of the following conditions holds:
 - $A \in X$, that is, $X \rightarrow A$ is a trivial FD, *or*
 - X is a superkey, *or*
 - If X is not a key, then A is part of some key of R
- To determine if a relation $\langle R, F \rangle$ is in 3NF:
 - We check whether the LHS of each nontrivial FD in F is a superkey
 - If not, we check whether its RHS is part of any key of R

33

Boyce-Codd Normal Form

Let R be a relation schema, F a set of FD's on R , $X \subseteq R$, and $A \in R$.

- We say R w.r.t. F is in **Boyce-Codd normal form**, if for every FD $X \rightarrow A$ in F , at least one of the following conditions holds:
 - $A \in X$, that is, $X \rightarrow A$ is a trivial FD, *or*
 - X is a superkey
- To determine whether R with a given set of FD's F is in BCNF
 - Check whether the LHS X of each nontrivial FD in F is a superkey
 - How? Simply compute X^+ (w.r.t. F) and check if $X^+ = R$

34