15CSE302 Database Management Systems

Lecture 22 Canonical Cover

B.Tech /III Year CSE/V Semester **L T P C 2 0 2 3**

DBMS Team

Dr G Jeyakumar

Bindu K R

Dr Priyanka Kumar

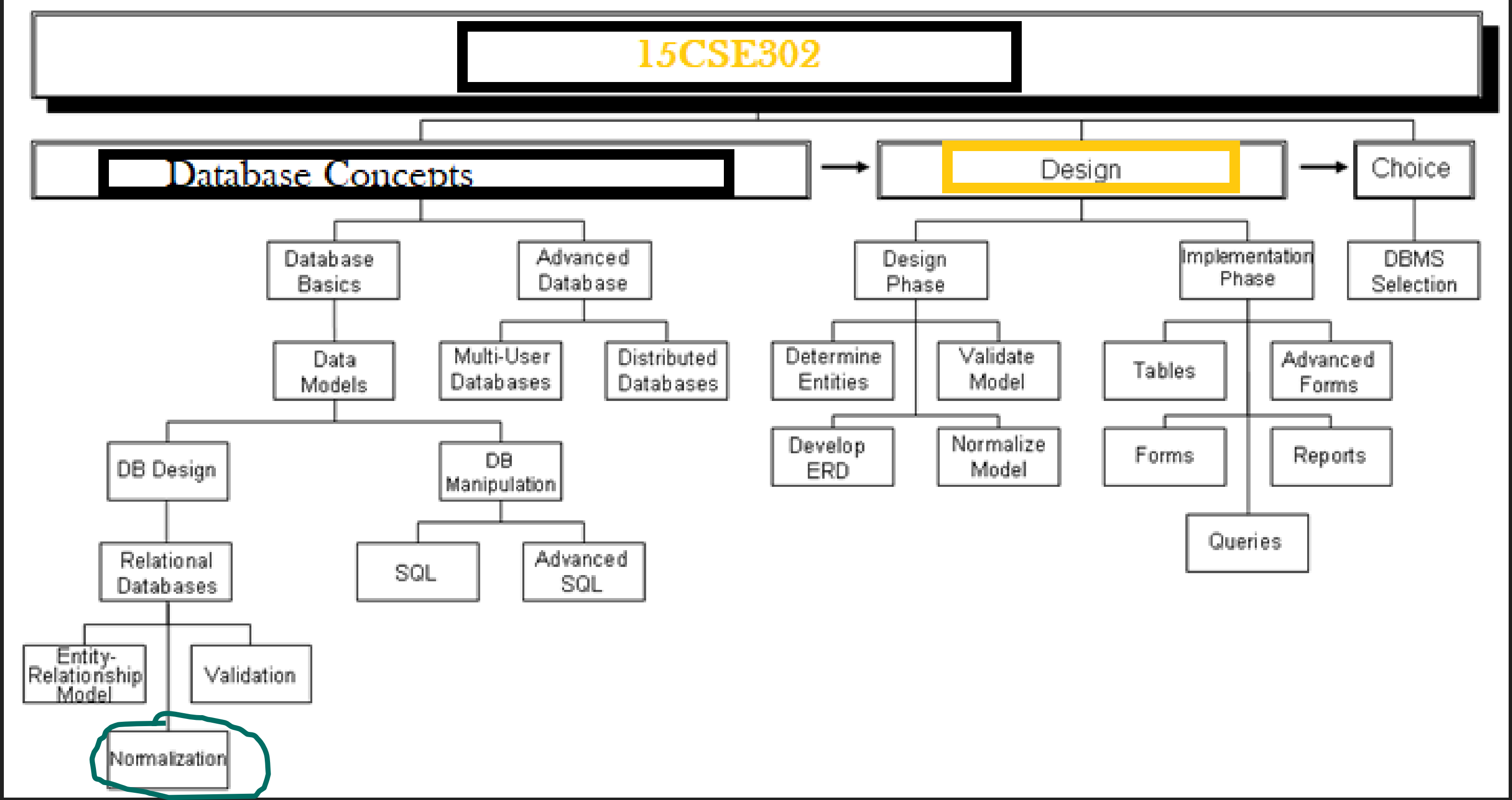
R. Manjusha

Department of CSE

Amrita School of Engineering

Slides Courtesy : Carlos Alvarado, San Jose State University

Syllabus



15CSE302
Database Concepts
Design
Choice
Database Basics
Advanced Database
Design Phase
Implementation Phase
DBMS Selection
Data Models
Multi-User Databases
Distributed Databases
Determine Entities
Validate Model
Tables
Advanced Forms
DB Design
DB Manipulation
Develop ERD
Normalize Model
Forms
Reports
Queries
Relational Databases
SQL
Advanced SQL
Entity-Relationship Model
Validation
Normalization

Brief Recap of Previous Lecture

- Closure of FD
- Closure of Attributes

Today we'll discuss

Canonical Cover

Canonical Cover

- Suppose that we have a set of functional dependencies *F* on a relation schema.
- Whenever a user performs an update on the relation, the database system must ensure that the update does not violate any functional dependencies; that is, all the functional dependencies in *F* are satisfied in the new database state.
- If an update violates any functional dependencies in the set *F*, the system must roll back the update.
- We can reduce the effort spent in checking for violations by testing a simplified set of functional dependencies that has the same closure as the given set.
- This simplified set is termed the **canonical cover**

Extraneous Attribute

- To define canonical cover we must first define **extraneous attributes**.

An attribute of a functional dependency in F is **extraneous** if we can remove it without changing F^+

Extraneous Attribute

Removing an attribute from the left side of a functional dependency could make it a stronger constraint.

For example

- if we have $AB \rightarrow C$ and remove B, we get the possibly stronger result $A \rightarrow C$.
- It may be stronger because $A \rightarrow C$ logically implies $AB \rightarrow C$, but $AB \rightarrow C$ does not, on its own, logically imply $A \rightarrow C$

Extraneous Attribute

But, depending on what our set F of functional dependencies happens to be, we may be able to remove B from $AB \to C$ safely.

For example, suppose that

- $F = \{AB \to C, A \to D, D \to C\}$
- Then we can show that F logically implies $A \to C$, making extraneous in $AB \to C$.

Extraneous Attribute

- An attribute of a functional dependency in F is **extraneous** if we can remove it without changing F^+

Extraneous Attribute

Consider a set *F* of functional dependencies and the functional dependency $\alpha \rightarrow \beta$ in *F*.

- **Remove from the left side:**

Attribute A is **extraneous** in α if

- $A \in \alpha$ and
- F logically implies $(F - \{\alpha \rightarrow \beta\}) \cup \{(\alpha - A) \rightarrow \beta\}$.

Extraneous Attribute

Consider a set F of functional dependencies and the functional dependency $\alpha \rightarrow \beta$ in F.

Remove from the right side:

- Attribute A is extraneous in β if
  - $A \in \beta$ and
  - The set of functional dependencies

$$(F - \{\alpha \rightarrow \beta\}) \cup \{\alpha \rightarrow (\beta - A)\} \text{ logically implies } F.$$

- Note: implication in the opposite direction is trivial in each of the cases above, since a "stronger" functional dependency always implies a weaker one

Testing an Extraneous Attribute

Let R be a relation schema and let F be a set of functional dependencies that hold on R.

Consider an attribute in the functional dependency $\alpha \rightarrow \beta$.

- To test if attribute $A \in \beta$ is extraneous in β
  - Consider the set:

    $$F' = (F - \{\alpha \rightarrow \beta\}) \cup \{\alpha \rightarrow (\beta - A)\},$$

  - check that α^+ contains A;
    - if it does, ***A* is extraneous in β**

Testing an Extraneous Attribute

- To test if attribute $A \in \alpha$ is extraneous in α
  - Let $\gamma = \alpha - \{A\}$.
  - Check if $\gamma \rightarrow \beta$ can be inferred from *F*.
    - Compute γ^+ using the dependencies in *F*
    - If γ^+ includes all attributes in β then , *A* is extraneous in α

Testing an Extraneous Attribute

- Let F = {AB $\rightarrow$ CD, A $\rightarrow$ E, E $\rightarrow$ C }
- To check if C is extraneous in AB $\rightarrow$ CD, we:
  - Compute the attribute closure of AB under F' = {AB $\rightarrow$ D, A $\rightarrow$ E, E $\rightarrow$ C}
  - The closure is ABCDE, which includes CD
  - This implies that C is extraneous

Canonical Cover

A **canonical cover** for F is a set of dependencies F_c such that

- F logically implies all dependencies in F_c , and
- F_c logically implies all dependencies in F, and
- No functional dependency in F_c contains an extraneous attribute, and
- Each left side of functional dependency in F_c is unique.

That is, there are no two dependencies in F_c

  - $\alpha 1 \rightarrow \beta 1$ and $\alpha 2 \rightarrow \beta 2$ such that
  - $\alpha 1 = \alpha 2$

Canonical Cover

- To compute a canonical cover for *F*:

  repeat

  Use the union rule to replace any dependencies in F of the form

  $\alpha_1 \rightarrow \beta_1$ and $\alpha_1 \rightarrow \beta_2$ with $\alpha_1 \rightarrow \beta_1 \beta_2$

  Find a functional dependency $\alpha \rightarrow \beta$ in F_c with an extraneous attribute either in α or in β

  /* Note: test for extraneous attributes done using $F_{c,}$ not F */

  If an extraneous attribute is found, delete it from $\alpha \rightarrow \beta$

  until (F_c not change)

- Note: Union rule may become applicable after some extraneous attributes have been deleted, so it has to be re-applied

Compute Canonical Cover

- $R = (A, B, C)$
  $F = \{A \rightarrow BC \quad B \rightarrow C \quad A \rightarrow B \quad AB \rightarrow C\}$

- Combine $A \rightarrow BC$ and $A \rightarrow B$ into $A \rightarrow BC$
  - Set is now $\{A \rightarrow BC, B \rightarrow C, AB \rightarrow C\}$

Compute Canonical Cover

- $R = (A, B, C)$
  $F = \{A \rightarrow BC \quad B \rightarrow C \quad A \rightarrow B \quad AB \rightarrow C\}$
- A is extraneous in $AB \rightarrow C$
  - Check if the result of deleting A from $AB \rightarrow C$ is implied by the other dependencies
    - Yes: in fact, $B \rightarrow C$ is already present!
  - Set is now $\{A \rightarrow BC, B \rightarrow C\}$

Compute Canonical Cover

- $R = (A, B, C)$
  $F = \{A \rightarrow BC \quad B \rightarrow C \quad A \rightarrow B \quad AB \rightarrow C\}$
- C is extraneous in $A \rightarrow BC$
  - Check if $A \rightarrow C$ is logically implied by $A \rightarrow B$ and the other dependencies
    - Yes: using transitivity on $A \rightarrow B$ and $B \rightarrow C$.
      - Can use attribute closure of A in more complex cases
- The canonical cover is: $A \rightarrow B$
  $B \rightarrow C$

Example 2 : Compute Canonical Cover

Consider another set F of functional dependencies:

F={A → BC , CD → E , B → D, E → A }

Example 2 : Compute Canonical Cover

Consider another set F of functional dependencies:

$F=\{A \rightarrow BC, \quad CD \rightarrow E, \quad B \rightarrow D, \quad E \rightarrow A\}$

- The left side of each functional dependency in F is unique.
- None of the attributes in the left or right side of any functional dependency is extraneous (Checked by applying definition of extraneous attributes on every functional dependency).
- **Hence, the canonical cover F_c is equal to F.**

Example 3 : Compute Canonical Cover

Compute the minimal cover

R = (A, B, C,D,E,F,G)

FD = {$ABC \rightarrow DE$ $BD \rightarrow DE$ $E \rightarrow CF$ $EG \rightarrow F$ }

Example 3 : Compute Canonical Cover

- $R = (A, B, C, D, E, F, G)$

  $FD = \{ABC \rightarrow DE \quad BD \rightarrow DE \quad E \rightarrow CF \quad EG \rightarrow F\}$

The minimal cover is

$= \{ABC \rightarrow D, \quad BD \rightarrow E, E \rightarrow C, E \rightarrow F\}$

$ABC \rightarrow D$

$ABC \rightarrow E$

$BD \rightarrow D$ //reflexive

$BD \rightarrow E$

$E \rightarrow C$

$E \rightarrow F$

$EG \rightarrow F$ //Augmentation

Example 3 : Compute Canonical Cover

- R = (A, B, C,E)

  F = {$A \rightarrow BC$ $B \rightarrow CE$ $A \rightarrow E$ }

Iteration 1

- F = {$A \rightarrow BCE$ $B \rightarrow CE$}

Check for extraneous attributes

- B extraneous A→BCE No
- C extraneous A→BCE Yes
- E extraneous A→BE No

  A→B A->CE A->E

- E extraneous B→CE No
- C extraneous B→CE No

Iteration 2

- F = {$A \rightarrow B$ $B \rightarrow CE$}

Check for Extraneous Attributes

- C extraneous B→CE No
- E extraneous B→CE No

Boyce and Codd Normal Form (BCNF)

For a table to satisfy the **Boyce-Codd Normal Form,** it should satisfy the following **two conditions:**

- It should be in the **Third Normal Form**.
- for any dependency $A \rightarrow B$, A should be a **super key**.

In simple words, it means, that for a dependency $A \rightarrow B$
A cannot be a **non-prime attribute**, if B is a **prime attribute**.

Boyce and Codd Normal Form (BCNF)

- **Boyce and Codd Normal Form** is a higher version of the Third Normal form.
- This form deals with certain type of anomaly that is not handled by 3NF.
- **A 3NF table which does not have multiple overlapping candidate keys is said to be in BCNF.**

Goals of Normalisation

- Let R be a relation scheme with a set F of functional dependencies.
- Decide whether a relation scheme R is in "good" form.
- In the case that a relation scheme R is not in "good" form, need to decompose it into a set of relation scheme $\{R_1, R_2, ..., R_n\}$ such that:
  - Each relation scheme is in good form
  - The decomposition is a lossless decomposition
  - Preferably, the decomposition should be dependency preserving.

Consider the following relationship : **R (A,B,C,D)**

and following dependencies :

A -> BCD
BC -> AD
D -> B

Above relationship is already in 3rd NF. Keys are **A** and **BC.**

Hence, in the functional dependency, **A -> BCD**, A is the super key.
in second relation, **BC -> AD**, BC is also a key.
but in, **D -> B**, D is not a key.

Hence we can break our relationship R into two relationships **R1** and **R2**.

R(A, B, C, D)

R1 (A, D, C)　　　　R2(D, B)

Breaking, table into two tables, one with A, D and C while the other with D and B.

- R1(A,B,C,D,E) R2(B,F)
- CANDIDATE KEY IS B
- A→BCDE
- BC→ADE
- D→E

- 2NF

R1

R2(A,B,C,D) R3(D,E)

Normalisation

- – Steps
- 1NF
- – Removing repeating groups
- 2NF
- – Remove partial dependencies
- • 3NF
- – Remove transitive dependencies
- BCNF
- – Remove non-candidate key dependencies

Project Code	Project Title	Project Manager	Project Budget	Employee No.	Employee Name	Department No.	Department Name	Hourly Rate
PC010	Pensions System	M Phillips	24500	S10001	A Smith	L004	IT	22.00
PC010	Pensions System	M Phillips	24500	S10030	L Jones	L023	Pensions	18.50
PC010	Pensions System	M Phillips	24500	S21010	P Lewis	L004	IT	21.00
PC045	Salaries System	H Martin	17400	S10010	B Jones	L004	IT	21.75
PC045	Salaries System	H Martin	17400	S10001	A Smith	L004	IT	18.00
PC045	Salaries System	H Martin	17400	S31002	T Gilbert	L028	Database	25.50
PC045	Salaries System	H Martin	17400	S13210	W Richards	L008	Salary	17.00
PC064	HR System	K Lewis	12250	S31002	T Gilbert	L028	Database	23.25
PC064	HR System	K Lewis	12250	S21010	P Lewis	L004	IT	17.50
PC064	HR System	K Lewis	12250	S10034	B James	L009	HR	16.50

1NF Tables: Repeating Attributes Removed

Project Code	Employee No.	Employee Name	Department No.	Department Name	Hourly Rate
PC010	S10001	A Smith	L004	IT	22.00
PC010	S10030	L Jones	L023	Pensions	18.50
PC010	S21010	P Lewis	L004	IT	21.00
PC045	S10010	B Jones	L004	IT	21.75
PC045	S10001	A Smith	L004	IT	18.00
PC045	S31002	T Gilbert	L028	Database	25.50
PC045	S13210	W Richards	L008	Salary	17.00
PC064	S31002	T Gilbert	L028	Database	23.25
PC064	S21010	P Lewis	L004	IT	17.50
PC064	S10034	B James	L009	HR	16.50

Project Code	Project Title	Project Manager	Project Budget
PC010	Pensions System	M Phillips	24500
PC045	Salaries System	H Martin	17400
PC064	HR System	K Lewis	12250

2NF Tables: Partial Key Dependencies Removed

Project Code	Project Title	Project Manager	Project Budget
PC010	Pensions System	M Phillips	24500
PC045	Salaries System	H Martin	17400
PC064	HR System	K Lewis	12250

Project Code	Employee No.	Hourly Rate
PC010	S10001	22.00
PC010	S10030	18.50
PC010	S21010	21.00
PC045	S10010	21.75
PC045	S10001	18.00
PC045	S31002	25.50
PC045	S13210	17.00
PC064	S31002	23.25
PC064	S21010	17.50

Employee No.	Employee Name	Department No.	Department Name
S10001	A Smith	L004	IT
S10030	L Jones	L023	Pensions
S21010	P Lewis	L004	IT
S10010	B Jones	L004	IT
S31002	T Gilbert	L028	Database
S13210	W Richards	L008	Salary
S10034	B James	L009	HR

References

- Hillyer Mike, MySQL AB. An Introduction to Database Normalization, http://dev.mysql.com/tech-resources/articles/intro-to-normalization.html, accessed October 17, 2006.

- Microsoft. Description of the database normalization basics, http://support.microsoft.com/kb/283878 , accessed October 17, 2006.

- Wikipedia. Database Normalization. http://en.wikipedia.org/wiki/Database_normalization.html , accessed October 17, 2006.
  https://www.db-book.com/db6/index.html
- https://www.youtube.com/watch?v=mfVCesoMaGA&list=PLroEs25KGvwzmvIxYHRhoGTz9w8
  LeXek0&index=22

Summary

- Normalization basics
- Anomalies

Next Lecture

Functional dependency

Thank You

Happy to answer any questions ! ! !