CS422 Principles of Database Systems
Normalization

Chengyu Sun
California State University, Los Angeles

Schema Design

Problem Description
-> ER Design

ER Diagram
-> Relational Schema

Transform Bad into Good
-> Good Relational Schema

Bad Schema

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>address</th>
<th>assignment</th>
<th>due</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>123 Main St.</td>
<td>HW1</td>
<td>2009-06-22</td>
<td>A-</td>
</tr>
<tr>
<td>1</td>
<td>John</td>
<td>123 Main St.</td>
<td>HW2</td>
<td>2009-07-10</td>
<td>B</td>
</tr>
<tr>
<td>2</td>
<td>Jane</td>
<td>456 State St.</td>
<td>HW1</td>
<td>2009-06-22</td>
<td>A</td>
</tr>
</tbody>
</table>

class_records

- Update anomaly
- Delete anomaly

Normalization

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>address</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>123 Main St.</td>
</tr>
<tr>
<td>2</td>
<td>Jane</td>
<td>456 State St.</td>
</tr>
</tbody>
</table>

students

<table>
<thead>
<tr>
<th>name</th>
<th>due</th>
</tr>
</thead>
<tbody>
<tr>
<td>HW1</td>
<td>2009-06-22</td>
</tr>
<tr>
<td>HW2</td>
<td>2009-07-10</td>
</tr>
</tbody>
</table>

assignments

<table>
<thead>
<tr>
<th>student</th>
<th>assignment</th>
<th>grades</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>HW1</td>
<td>A-</td>
</tr>
<tr>
<td>2</td>
<td>HW1</td>
<td>A</td>
</tr>
</tbody>
</table>

Functional Dependency (FD)

- A functional dependency on table R is the assertion that two records having the same values for attributes \{A_1,...,A_n\} must also have the same value for attribute B

- \{A_1,...,A_n\} \rightarrow B, or \{A_1,...,A_n\} functionally determine B

Questions To Be Answered

- How do we decide whether a schema is bad?
- How do we decompose a table to turn a bad schema into a good one?
About FD

A FD is an assertion based on assumptions about all possible data, not just the existing data.

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
</tr>
<tr>
<td>2</td>
<td>Jane</td>
</tr>
</tbody>
</table>

{id} → name ✓
{name} → id ×

FD with Multiple Attributes

{A₁, A₂, A₃, ..., Aₙ} → B₁
{A₁, A₂, A₃, ..., Aₙ} → B₂
...
{A₁, A₂, A₃, ..., Aₙ} → Bₘ

A → B

Trivial Functional Dependency

FD: {A₁, A₂, A₃, ..., Aₙ} → {B₁, B₂, B₃, ..., Bₘ}

- FD is trivial if all B's are in A
- FD is nontrivial if at least one B is not in A
- FD is completely nontrivial if no B is in A

From now on, when we talk about FD, we mean completely nontrivial FD unless otherwise noted.

FD Example 1

- Musicians (id, name, address)
- Bands (id, name)
- Band_Members (band_id, musician_id)

FD Example 2

- Books (id, title)
- Authors (id, name)
- Book_Authors (book_id, author_id, author_order)

FD Example 3

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>address</th>
<th>assignment</th>
<th>due</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>123 Main St.</td>
<td>HW1</td>
<td>2009-06-22</td>
<td>A-</td>
</tr>
<tr>
<td>1</td>
<td>John</td>
<td>123 Main St.</td>
<td>HW2</td>
<td>2009-07-10</td>
<td>B</td>
</tr>
<tr>
<td>2</td>
<td>Jane</td>
<td>456 State St.</td>
<td>HW1</td>
<td>2009-06-22</td>
<td>A</td>
</tr>
</tbody>
</table>

class_records

- Functional dependencies??
A key of table R is if:
- A functionally determines all attributes of R
- No proper subset of A functionally determines all attributes of R

A Few Things about Keys:
- A table may have multiple keys
- A key may consist of multiple attributes
- Superset of a key is called a super key
- The definition doesn't say anything about uniqueness
- A key has to be minimal, but not necessarily minimum

Key Examples:
- Musicians and bands
- Books and authors
- Class_records

Boyce-Codd Normal Form (BCNF):
- A table R is in BCNF if for every nontrivial FD A → B in R, A is a super key of R.

Or

*The key, the whole key, and nothing but the key, so help me Codd.*

BCNF or Not?
- Musicians and bands
- Books and authors
- Class_records

Determine If a Table is BCNF:
- Step 1: identify all FDs
- Step 2: find all keys
- Step 3: check LHS of all non-trivial FDs and see if they are a superset of a key (i.e. a super key)
Decompose into BCNF

- Given table R with functional dependencies S
- Look among F for a BCNF violation A → B
- Compute A
- Decompose R into:
  - R₁ = A
  - R₂ = (R – A) U A
- Continue decomposition with R₁ and R₂ until all resulting tables are BCNF

BCNF Violation Example

- Class_records
  - Functional dependencies
  - Key(s)
  - BCNF violation(s)?

Closure of Attributes A⁺

- Given
  - a set of attributes A
  - a set of functional dependencies S
- Closure of A under S, A⁺, is the set of all possible attributes that are functionally determined by A based on the functional dependencies inferable from S

Simple Closure Example

- R: {A,B,C}
  - S: {A → B, B → C}
  - {A}⁺ ?
  - {B}⁺ ?
  - {C}⁺ ?

Armstrong’s Axioms

- Reflexivity
  - If B ⊆ A, then A → B
- Transitivity
  - If A → B and B → C, then A → C
- Augmentation
  - If A → B, then AC → BC for any C

Two More FD Rules

- Union
  - If A → B and A → C, then A → BC
- Decomposition
  - If A → BC, then A → B and A → C
Computing $A^+$

- Initialize $A^+ = A$
- Search in $S$ for $B \rightarrow C$ where
  - $B \subseteq A^+$
  - $C \in A^+$
- Add $C$ to $A^+$
- Repeat until nothing can be added to $A^+$

Computing $A^+$ Example

- $R( A, B, C, D, E, F)$
- $S$: $AB \rightarrow C$, $BC \rightarrow AD$, $D \rightarrow E$, $CF \rightarrow B$
- $\{A,B\}^+$ ?
- Is $\{A,B\}$ a key ?
- How do we find out the key(s) from $R$?

Example: BCNF Decomposition

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>address</th>
<th>assignment</th>
<th>due</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>John</td>
<td>123 Main St.</td>
<td>HW1</td>
<td>2009-06-22</td>
<td>A-</td>
</tr>
<tr>
<td>1</td>
<td>John</td>
<td>123 Main St.</td>
<td>HW2</td>
<td>2009-07-10</td>
<td>B</td>
</tr>
<tr>
<td>2</td>
<td>Jane</td>
<td>456 State St.</td>
<td>HW1</td>
<td>2009-06-22</td>
<td>A</td>
</tr>
</tbody>
</table>

Motivation for 3NF

- We lose the FD $(street\_address, city) \rightarrow zip$
  - $(street\_address, city) \rightarrow zip$
  - $(city, zip)$
- After decomposition, or in other words, it becomes unenforceable.

An Unenforceable FD

Before decomposition:

<table>
<thead>
<tr>
<th>street</th>
<th>city</th>
<th>zip</th>
</tr>
</thead>
<tbody>
<tr>
<td>445 Tech Sq.</td>
<td>Cambridge</td>
<td>02138</td>
</tr>
<tr>
<td>445 Tech Sq.</td>
<td>Cambridge</td>
<td>02139</td>
</tr>
</tbody>
</table>

After decomposition:

<table>
<thead>
<tr>
<th>street</th>
<th>city</th>
<th>zip</th>
</tr>
</thead>
<tbody>
<tr>
<td>445 Tech Sq.</td>
<td>Cambridge</td>
<td>02138</td>
</tr>
<tr>
<td>445 Tech Sq.</td>
<td>Cambridge</td>
<td>02139</td>
</tr>
</tbody>
</table>

The same data error can no longer be detected.

Third Normal Form (3NF)

- A table $R$ is in 3NF if for every nontrivial $FD A \rightarrow B$ in $R$,
  - $A$ is a super key of $R$
  - or $B$ is part of a key of $R$