Relational Database Design Theory
Part II

CPS 116
Introduction to Database Systems

Announcements (October 12)

- Midterm graded; sample solution available
- Homework #2 graded; sample solution already distributed
- Please verify all your grades on Blackboard
- Project milestone #1 due today

Review

- Functional dependencies
  - $X \rightarrow Y$: If two rows agree on $X$, they must agree on $Y$
    - A generalization of the key concept
  - Non-key functional dependencies: a source of redundancy
    - Non-trivial $X \rightarrow Y$ where $X$ is not a superkey
      - Called a BCNF violation
  - BCNF decomposition: a method for removing redundancies
    - Given $R(X, Y, Z)$ and a BCNF violation $X \rightarrow Y$, decompose $R$ into $R_1(X, Y)$ and $R_2(X, Z)$
      - A lossless join decomposition
  - Schema in BCNF has no redundancy due to FD's
Next

- 3NF (BCNF is too much)
- Multivalued dependencies: another source of redundancy
- 4NF (BCNF is not enough)

Motivation for 3NF

- Address (street_address, city, state, zip)
  - street_address, city, state → zip
  - zip → city, state
- Keys

- BCNF?

To decompose or not to decompose

Address1 (zip, city, state)
Address2 (street_address, zip)
- FD's in Address1
- FD's in Address2

- Hey, where is street_address, city, state → zip?
  - Cannot check without joining Address1 and Address2 back together
- Problem: Some lossless join decomposition is not dependency-preserving
- Dilemma: Should we get rid of redundancy at the expense of making constraints harder to enforce?
3NF

- $R$ is in Third Normal Form (3NF) if for every non-trivial FD $X \rightarrow A$ (where $A$ is a single attribute), either
  - $X$ is a superkey of $R$, or
  - $A$ is a member of at least one key of $R$
- Intuitively, BCNF decomposition on $X \rightarrow A$ would "break" the key containing $A$
- So Address is already in 3NF
- Tradeoff:
  - Can enforce all original FD’s on individual decomposed relations
  - Might have some redundancy due to FD’s

BCNF = no redundancy?

- Student ($SID, CID, club$)
  - Suppose your classes have nothing to do with the clubs you join
  - FD’s?
  - $BCNF$?
  - Redundancies?

Multivalued dependencies

- A multivalued dependency (MVD) has the form $X \rightarrow Y$, where $X$ and $Y$ are sets of attributes in a relation $R$
- $X \rightarrow Y$ means that whenever two rows in $R$ agree on all the attributes of $X$, then we can swap their $Y$ components and get two new rows that are also in $R$
MVD examples

Student (SID, CID, club)

Complete MVD + FD rules

- FD reflexivity, augmentation, and transitivity
- MVD complementation:
  If $X \rightarrow Y$, then $X \rightarrow attr(R) - X - Y$
- MVD augmentation:
  If $X \rightarrow Y$ and $V \subseteq W$, then $XW \rightarrow YV$
- MVD transitivity:
  If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z - Y$
- Replication (FD is MVD):
  If $X \rightarrow Y$, then $X \rightarrow Y$
- Coalescence:
  If $X \rightarrow Y$ and $Z \subseteq Y$ and there is some $W$ disjoint from $Y$
  such that $W \rightarrow Z$, then $X \rightarrow Z$

Try proving things using these!

An elegant solution: chase

- Given a set of FD’s and MVD’s $\mathcal{D}$, does another dependency $d$ (FD or MVD) follow from $\mathcal{D}$?

- Procedure
  - Start with the hypothesis of $d$, and treat them as “seed” tuples in a relation
  - Apply the given dependencies in $\mathcal{D}$ repeatedly
    - If we apply an FD, we infer equality of two symbols
    - If we apply an MVD, we infer more tuples
  - If we infer the conclusion of $d$, we have a proof
  - Otherwise, if nothing more can be inferred, we have a counterexample
Proof by chase

\( \diamond \) In \( R(A, B, C, D) \), does \( A \rightarrow B \) and \( B \rightarrow C \) imply that \( A \rightarrow C \)?

<table>
<thead>
<tr>
<th>Have</th>
<th>Need</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A \rightarrow B )</td>
<td>( b_2 = b_1 )</td>
</tr>
<tr>
<td>( B \rightarrow C )</td>
<td>( c_2 = c_1 )</td>
</tr>
</tbody>
</table>

Another proof by chase

\( \diamond \) In \( R(A, B, C, D) \), does \( A \rightarrow B \) and \( B \rightarrow C \) imply that \( A \rightarrow C \)?

<table>
<thead>
<tr>
<th>Have</th>
<th>Need</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A \rightarrow B )</td>
<td>( b_1 = b_2 )</td>
</tr>
<tr>
<td>( B \rightarrow C )</td>
<td>( c_1 = c_2 )</td>
</tr>
</tbody>
</table>

In general, both new tuples and new equalities may be generated.

Counterexample by chase

\( \diamond \) In \( R(A, B, C, D) \), does \( A \rightarrow BC \) and \( CD \rightarrow B \) imply that \( A \rightarrow B \)?

<table>
<thead>
<tr>
<th>Have</th>
<th>Need</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A \rightarrow BC )</td>
<td>( b_1 = b_2 )</td>
</tr>
</tbody>
</table>

Counterexample!
4NF

- A relation $R$ is in Fourth Normal Form (4NF) if
  - For every non-trivial MVD $X \rightarrow Y$ in $R$, $X$ is a superkey
  - That is, all FD's and MVD's follow from "key $\rightarrow$ other attributes" (i.e., no MVD's and no FD's besides key functional dependencies)

- 4NF is stronger than BCNF
  - Because every FD is also a MVD

4NF decomposition algorithm

- Find a 4NF violation
  - A non-trivial MVD $X \rightarrow Y$ in $R$ where $X$ is not a superkey
- Decompose $R$ into $R_1$ and $R_2$, where
  - $R_1$ has attributes $X \cup Y$
  - $R_2$ has attributes $X \cup Z$ ($Z$ contains attributes not in $X$ or $Y$)
- Repeat until all relations are in 4NF

- Almost identical to BCNF decomposition algorithm
- Any decomposition on a 4NF violation is lossless

4NF decomposition example

- Student ($SID, CID, club$)
  - 4NF violation: $SID \rightarrow CID$
- Enroll ($SID, CID$)
- Join ($SID, club$)
### 3NF, BCNF, 4NF, and beyond

<table>
<thead>
<tr>
<th></th>
<th>1NF</th>
<th>BCNF</th>
<th>2NF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Has FD's?</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Redundancy due to FD's</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Redundancy due to MVD's</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

- **Of historical interests**
  - 1NF: All column values must be atomic
  - 2NF: Slightly more relaxed than 3NF

### Summary

- **Philosophy behind BCNF, 4NF:**
  Data should depend on the key, the whole key, and nothing but the key!

- **Philosophy behind 3NF:**
  … But not at the expense of more expensive constraint enforcement!