Multivalued Dependencies

• FD is a powerful formalism for decomposing schema to eliminate redundancy

• Idea behind FD:
  - the value of a particular attribute uniquely determine the value of some other attribute
  - what if change uniquely determine by restrict?
  - FD rules out existence of certain tuples in the relation:
    A→B means there’s no two tuples $t_1$ and $t_2$ such that $t_1[A]=t_2[A] \land t_1[B]\neq t_2[B]$

• Multivalued dependency (MVD)
  - MVD requires other tuples of a certain form be present
  - consequence of 1NF that disallows a set of values for an attribute in a tuple
  - if two or more multivalued independent attributes in the relation, every value of one attribute must be repeated with every value of other attribute to keep it consistent
MVD

Let R be a relation schema, and X and Y be disjoint subsets of R (i.e., \( X \subseteq R, Y \subseteq R, X \cap Y = \emptyset \)), and \( Z = R - XY \).

A relation \( r(R) \) satisfies \( X \rightarrow\rightarrow Y \) if for any two tuples \( t_1 \) and \( t_2 \), \( t_1(X) = t_2(X) \), then there exist \( t_3 \) in \( r \) such that \( t_3(X) = t_1(X), t_3(Y) = t_1(Y), t_3(Z) = t_2(Z) \).

By symmetry, there exist \( t_4 \) in \( r \) such that \( t_4(X) = t_1(X), t_4(Y) = t_2(Y), t_4(Z) = t_1(Z) \).

\[
\begin{array}{ccc}
X & Y & Z \\
--- & --- & --- \\
t_1 & x1 & y1 & z1 \\
t_2 & x1 & y2 & z2 \\
t_3 & x1 & y1 & z2 \\
t_4 & x1 & y2 & z1 \\
\end{array}
\]

• Intuition

The MVD \( X \rightarrow\rightarrow Y \) says that the relationship between \( X \) and \( Y \) is independent of the relationship between \( X \) and \( R - Y \).
MVD

<table>
<thead>
<tr>
<th>Employee</th>
<th>(E-name)</th>
<th>P-name</th>
<th>D-name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith X</td>
<td>X</td>
<td>John</td>
<td></td>
</tr>
<tr>
<td>Smith Y</td>
<td>Y</td>
<td>Ann</td>
<td></td>
</tr>
<tr>
<td>Smith X</td>
<td>X</td>
<td>Ann</td>
<td></td>
</tr>
<tr>
<td>Smith Y</td>
<td>Y</td>
<td>John</td>
<td></td>
</tr>
</tbody>
</table>

- MVDs E-name →→ P-name and E-name →→ D-name hold in the relation:
  The employee named Smith works on projects X and Y, and has two dependents John and Ann.

- If we store only the first two tuples in the relation, it would incorrectly show the associations among attributes.

If we have MVDs in a relation, we may have to repeat values redundantly in the tuples. In the Employee relation, values X and Y of P-name are repeated with each value of D-name.

--- clearly undesirable

Problem: Employee schema is in BCNF because no FDs hold for it
Notes on MVD

- Trivial MVD
  If MVD $X \rightarrow\rightarrow Y$ is satisfied by all relations whose schemas include $X$ and $Y$, it is called *trivial* MVD.
  - $X \rightarrow\rightarrow Y$ is trivial whenever $Y \subseteq X$ or $X \cup Y = R$

- If a relation $r$ fails to satisfy a given MVD, a relation $r'$ that satisfies the MVD can be constructed by adding tuples to $r$
  - MVD is called "tuple generating dependency"
  - compare it with FD: need to delete tuples to make the relation to satisfy a given FD

- MVD can be used in two ways
  - test relations to determine whether they are legal under a given set of FDs and MVDs
  - specify constraints on a set of relations
Inference Rules for Computing $D^+$

$D$: a set of FDs and MVDs

$D^+$: the closure of $D$, the set of all FDs and MVDs logically implied by $D$

Sound and complete rules

1. reflexivity: if $Y \subseteq X$ then $X \rightarrow Y$
2. augmentation: if $X \rightarrow Y$ then $WX \rightarrow Y$
3. transitivity: if $X \rightarrow Y$ and $Y \rightarrow Z$ then $X \rightarrow Z$
4. complementation: if $X \rightarrow Y$ then $X \rightarrow R - XY$
5. MV augmentation: if $X \rightarrow Y$ and $W \subseteq R$, $V \subseteq W$, then $WX \rightarrow VY$
6. MV transitivity: if $X \rightarrow Y$ and $Y \rightarrow Z$ then $X \rightarrow Z - Y$
7. replication: if $X \rightarrow Y$ then $X \rightarrow \rightarrow Y$
8. coalescence: if $X \rightarrow Y$ and $Z \subseteq Y$, $W \subseteq R$, $W \cap Y = \emptyset$, $W \rightarrow Z$, then $X \rightarrow Z$

Note: The first three rules are Armstrong’s axioms.
Fourth Normal Form

A relation scheme $R$ is in 4NF w.r.t. $D$, if for every non-trivial MVD $X \rightarrow \rightarrow Y$ in $D^+$, $X$ is a superkey for $R$

- 4NF and BCNF
  - 4NF is different from BCNF only in the use of $D$ (FD + MVD) instead of $F$ (FDs)
  - every 4NF schemas are also in BCNF. Why?
    By replication rule, $X \rightarrow Y$ implies $X \rightarrow \rightarrow Y$.
    If $R$ is not in BCNF, there exists a non-trivial FD $X \rightarrow Y$
    where $X$ is not a superkey --- $R$ cannot be in 4NF

<ex> Employee (E-name, P-name, D-name) is not in 4NF, since E-name $\rightarrow \rightarrow$ P-name but E-name is not a key.
    Decompose into Emp-proj (E-n, P-n) and Emp-dep (E-n, D-n)

<ex> Borrow (Loan#, C-name, Street, C-city) is in BCNF, but not in 4NF, because C-name $\rightarrow \rightarrow$ Loan# is a non-trivial MVD,
    where C-name is not a key in this schema.
    $R_1$=$(C$-name, Loan#), $R_2=$(C-name, Street, C-city)
# Benefits of Fourth Normal Form

- Reduced number of tuples
- No anomalies for insert/delete/update

<table>
<thead>
<tr>
<th>&lt;ex&gt; Employee</th>
<th>(E-name)</th>
<th>P-name</th>
<th>D-name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>X</td>
<td></td>
<td>John</td>
</tr>
<tr>
<td>Smith</td>
<td>Y</td>
<td></td>
<td>Ann</td>
</tr>
<tr>
<td>Smith</td>
<td>X</td>
<td></td>
<td>Ann</td>
</tr>
<tr>
<td>Smith</td>
<td>Y</td>
<td></td>
<td>John</td>
</tr>
<tr>
<td>Brown</td>
<td>W</td>
<td></td>
<td>Jim</td>
</tr>
<tr>
<td>Brown</td>
<td>X</td>
<td></td>
<td>Jim</td>
</tr>
<tr>
<td>Brown</td>
<td>Y</td>
<td></td>
<td>Jim</td>
</tr>
<tr>
<td>Brown</td>
<td>W</td>
<td></td>
<td>Bob</td>
</tr>
<tr>
<td>Brown</td>
<td>X</td>
<td></td>
<td>Bob</td>
</tr>
<tr>
<td>Brown</td>
<td>Y</td>
<td></td>
<td>Bob</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Emp-proj (E-name P-name)</th>
<th>Emp-dep (E-name D-name)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Smith X</td>
<td>Smith John</td>
</tr>
<tr>
<td>Smith Y</td>
<td>Smith Ann</td>
</tr>
<tr>
<td>Brown W</td>
<td>Brown Jim</td>
</tr>
<tr>
<td>Brown X</td>
<td>Brown Bob</td>
</tr>
<tr>
<td>Brown Y</td>
<td></td>
</tr>
</tbody>
</table>
Lossless Join Decomposition

- The decomposition of $R$ into $R_1$ and $R_2$ is a lossless join decomposition iff one of the following MVDs hold in $D^+$
  
  $R_1 \cap R_2 \rightarrow R_1$ (or $R_1 - R_2$)
  $R_1 \cap R_2 \rightarrow R_2$ (or $R_2 - R_1$)
  
  --- whenever $R$ is decomposed into $R_1 = (X \cup Y)$ and $R_2 = (R - Y)$ based on an MVD $X \rightarrow Y$ that holds in $R$, it is a lossless join decomposition

- Algorithm

  set $D := \{R\}$
  while there is a schema $Q$ in $D$ that is not in 4NF do begin
    choose $Q$ in $D$ not in 4NF
    find a non-trivial MVD $X \rightarrow Y$ in $Q$ that violates 4NF
    replace $Q$ in $D$ by $(Q - Y)$ and $(X \cup Y)$
  end.

- Dependency preservation is not guaranteed
4NF

- Goal of database design
  - 4NF (BCNF if there is no MVD)
  - dependency preservation
  - lossless join decomposition
- If cannot satisfy all these, which one to compromise?
  The first one: 4NF > BCNF > 3NF to ensure other two
- BCNF and 4NF
  - although they are well known, they are not widely accepted as 1NF, 2NF, and 3NF, since dependency preservation is not guaranteed
- Comparing FD and MVD
  - if we have \((a_1 b_1 c_1 d_1) \in r\) and \((a_1 b_2 c_2 d_2) \in r\)
    \(A \rightarrow B\) implies \(b_1 = b_2\)
    \(A \rightarrow \rightarrow B\) implies \((a_1 b_1 c_2 d_2) \in r\)