Lecture 5: Normalization II; Database design case studies

September 26, 2005

Lecturer: Rasmus Pagh
Today’s lecture

Normalization II:

- 3rd normal form.
- Multivalued dependencies.
- 4th normal form.
- Some observations on normalization.

Case studies in database design:

- Internet bookstore.
- TV series database.
Next: 3rd normal form
Consider the relation with schema Bookings(title, theater, city)

Under certain assumptions, it has the FD \( theater \rightarrow city \), but \( theater \) is not a superkey. The BCNF decomposition yields relation schemas Bookings1(theater, city) and Bookings2(theater, title).

These schemas and their FDs allow, e.g., the relation instances:

<table>
<thead>
<tr>
<th>theater</th>
<th>city</th>
<th>theater</th>
<th>title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guild</td>
<td>Menlo Park</td>
<td>Guild</td>
<td>The net</td>
</tr>
<tr>
<td>Park</td>
<td>Menlo Park</td>
<td>Park</td>
<td>The net</td>
</tr>
</tbody>
</table>

which violate the presumed FD \( title \) \( city \) \( → \) \( theater \).

Thus, there are implicit dependencies between values in different relations. *We cannot check FDs separately in each relation to see such a dependency.*
As we just saw, decomposition can result in a relational database schema where a functional dependency “disappeared”.

The problem in the previous example arose because we decomposed according to the FD \( \text{theater} \rightarrow \text{city} \), where \( \text{city} \) is part of a key for the Bookings relation. Thus we ended up splitting the key \( \{\text{city}, \text{theater}\} \).

This problem of FDs that are not preserved never arises if we do not decompose in this case.
We have motivated the following normal form which never splits a key of the original relation. An attribute of a key is called *prime*.

A relation is in **3rd normal form** (3NF) if any functional dependency among its attributes is either unavoidable, or has a prime (i.e., member of some key) on the right hand side.

In words: A relation is in 3NF if there are no unavoidable functional dependencies among non-key attributes.
Whether it is a good idea to stop decomposition when third normal form is reached depends on the specific scenario.

- Mostly, 3NF and BCNF coincide, so there is nothing to consider.
- If not, the redundancy in tuples in 3NF should be weighed against the fact that some FD is difficult to check/maintain in BCNF.

**Example:**
In the Bookings example, we might want to make the DBMS check that to every title and city, there is at most one theater. For the BCNF decomposed relations, this would involve a query on Bookings1 for every change of Bookings2, and vice versa.
Next: Multivalued dependencies.
Boyce-Codd normal form eliminates redundancy in each tuple, but may leave redundancy among tuples in a relation.

This happens, for example, if two many-many relationships are represented in a relation.

[Figure 3.29 shown on slide]

**Example:** In the relation StarsIn(name, street, city, title, year) we could represent two many-many relationships: between actors and addresses, and between actors and movies.
Curing it with NULL values?

Then what about something like one of these:

<table>
<thead>
<tr>
<th>name</th>
<th>street</th>
<th>city</th>
<th>title</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>C. Fisher</td>
<td>123 Maple St.</td>
<td>Hollywood</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>C. Fisher</td>
<td>5 Locust Ln.</td>
<td>Malibu</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>C. Fisher</td>
<td>NULL</td>
<td>NULL</td>
<td>Star Wars</td>
<td>1977</td>
</tr>
<tr>
<td>C. Fisher</td>
<td>NULL</td>
<td>NULL</td>
<td>Empire Strikes Back</td>
<td>1980</td>
</tr>
<tr>
<td>C. Fisher</td>
<td>NULL</td>
<td>NULL</td>
<td>Return of the Jedi</td>
<td>1983</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>name</th>
<th>street</th>
<th>city</th>
<th>title</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>C. Fisher</td>
<td>123 Maple St.</td>
<td>Hollywood</td>
<td>Star Wars</td>
<td>1977</td>
</tr>
<tr>
<td>C. Fisher</td>
<td>5 Locust Ln.</td>
<td>Malibu</td>
<td>Empire Strikes Back</td>
<td>1980</td>
</tr>
<tr>
<td>C. Fisher</td>
<td>NULL</td>
<td>NULL</td>
<td>Return of the Jedi</td>
<td>1983</td>
</tr>
</tbody>
</table>

Problem session (5 minutes): Criticize the above solutions.
A better idea is to eliminate redundancy by decomposing StarsIn as follows:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>name</strong></td>
<td><strong>street</strong></td>
<td><strong>city</strong></td>
</tr>
<tr>
<td>C. Fisher</td>
<td>123 Maple St.</td>
<td>Hollywood</td>
</tr>
<tr>
<td>C. Fisher</td>
<td>5 Locust Ln.</td>
<td>Malibu</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>name</strong></td>
<td><strong>title</strong></td>
<td><strong>year</strong></td>
</tr>
<tr>
<td>C. Fisher</td>
<td>Star Wars</td>
<td>1977</td>
</tr>
<tr>
<td>C. Fisher</td>
<td>Empire Strikes Back</td>
<td>1980</td>
</tr>
<tr>
<td>C. Fisher</td>
<td>Return of the Jedi</td>
<td>1983</td>
</tr>
</tbody>
</table>
When can we decompose?

When can we decompose a relation $R$? Suppose we decompose into two relations (for simplicity we assume that there is just one common attribute):

$$R_1(A, B_1, B_2, \ldots, B_m)$$
$$R_2(A, C_1, C_2, \ldots, C_k)$$

Now consider a specific value $a$ for attribute $A$, occurring in the set of tuples $T_1$ from $R_1$ and in the set of tuples $T_2$ from $R_2$.

When we join $R_1$ and $R_2$, every pair of tuples from $T_1$ and $T_2$ are combined.
--- When can we decompose (2)? ---

Example:

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>a</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>N</td>
<td>1</td>
<td>E</td>
</tr>
<tr>
<td>1</td>
<td>S</td>
<td>1</td>
<td>NE</td>
</tr>
<tr>
<td>2</td>
<td>U</td>
<td>1</td>
<td>NW</td>
</tr>
<tr>
<td>2</td>
<td>D</td>
<td>1</td>
<td>SE</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>1</td>
<td>SW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>90</td>
</tr>
</tbody>
</table>
When we can decompose $R$ into relations

$$R_1(A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m)$$

$$R_2(A_1, A_2, \ldots, A_n, C_1, C_2, \ldots, C_k)$$

(with no Bs among the Cs) then we say that there is a **multivalued dependency** (MVD) from the As to the Bs, written

$$A_1 A_2 \ldots A_n \rightarrow B_1 B_2 \ldots B_m$$

**Example:** Since StarsIn can be decomposed into

StarsIn1(name, street, city) and StarsIn2(name, title, year)

it has the MVD name $\rightarrow$ street city.
Multi-valued dependencies, book’s definition

A1 A2...An →→ B1 B2...Bm

holds exactly if:

For every pair of tuples $t$ and $u$ from $R$ that agree on all As, we can find some tuple $v$ in $R$ that agrees:

- With both $t$ and $u$ on the As
- With $t$ on the Bs
- With $u$ on the Cs

[Figure 3.30 shown on slide]

Homework: Convince yourselves that this definition, used in the coursebook, is equivalent to the one given previously in this lecture.
Unavoidable and trivial MVDs

If \(\{A_1, A_2, \ldots, A_n\}\) form a superkey, then for any \(B_1, B_2, \ldots, B_m\) we unavoidably have:

\[
A_1 \ A_2 \ldots A_n \rightarrow B_1 \ B_2 \ldots B_m
\]

An MVD is said to be **trivial** if either

- One of the Bs is among the As, or
- All the attributes of R are among the As and Bs.
Next: 4th normal form.
Roughly speaking, a relation is in 4th normal form if it cannot be meaningfully decomposed into two relations. More precisely:

A relation is in **fourth normal form** (4NF) if any multivalued dependency among its attributes is either unavoidable or trivial.

**Example:** StarsIn has the MVD name $\rightarrow \rightarrow$ street city which is nontrivial. Since name is not a superkey the relation is not in 4NF.
Suppose we have a relation $R$ which is not in 4NF. Then there is a nontrivial MVD

$$A_1 A_2 \ldots A_n \rightarrow B_1 B_2 \ldots B_m$$

which is not unavoidable.

To eliminate the MVD we split $R$ into two relations:

- One with all attributes of $R$ except $B_1, B_2, \ldots, B_m$.
- One with attributes $A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m$.

If any of the resulting relations is not in 4NF, the process is repeated.
Recall the relation StarsIn with schema

StarsIn(name, street, city, title, year)

It has the following nontrivial MVD, which is not unavoidable:

\[ \text{name} \rightarrow \rightarrow \text{street city} \]

Thus the decomposition yields the following relations (both in 4NF):

StarsIn1(name, street, city)
StarsIn2(name, title, year)
Next: Some observations on normalization
### Relationship among normal forms

**Inclusion among normal forms:**

Any relation in 4NF is also in BCNF.

Any relation in BCNF is also in 3NF.

[Figure 3.31 shown on slide]

**Properties of normal forms:**

A “higher” normal form has less redundancy, but may not preserve functional and multivalued dependencies.

[Figure 3.32 shown on slide]
How should normal forms be used?

The various normal forms may be seen as *guidelines* for designing a good relation schema. Some complexities that arise are:

- Should we split keys, introducing dependencies between relations (in 3NF we do not)?
- What is the effect of decomposition on performance?
- How does decomposition affect query programming?
Consider the following E/R diagram shown on the slide, which is supposed to model data on sales to customers.

- Convert the E/R diagram to relations.
- Consider whether there are any 4NF violations, and if so perform the decomposition into 4NF.
After this week you should:

- Be able to determine whether a relation is in 3rd or 4th normal form.
- Be able to split a relation in several relations to achieve 3rd or 4th normal form.