6. Normalization

Stéphane Bressan

January 28, 2015

This lecture is based on material by Professor Ling Tok Wang.

https://www.comp.nus.edu.sg/

~lingtw/cs4221/rm.pdf

Content

Readings

- Bernstein, Philip, "Synthesizing Third Normal Form relations from functional dependencies." ACM Trans. Database Syst. 1,4 (Dec. 1976) 277-298.

Three Methods

The three common methods for relational database schema design are the Decomposition Method, the Synthesis Method, and the Entity-Relationship Approach.

Decomposition

The decomposition method is based on the assumption that a database can be represented by a universal relation which contains all the attributes of the database (this is called the universal relation assumption) and this relation is then decomposed into smaller relations, fragments, in order to remove redundant data.

Synthesis

The synthesis method is based on the assumption that a database can be described by a given set of attributes and a given set of functional dependencies, and 3NF or BCNF relations, fragments, are then synthesized based on the given set of dependencies.

Note: Synthesis method assumes universal relation assumption also.

Entity-Relationship

We will discuss the Entity-Relationship Approach later.

Three Criteria

The three criteria for the decomposition and synthesis methods are losslessness (reconstructability), dependency preservation (covering) and freedom from globally redundant attributes.

Losslessness

The natural join (see the universal relation assumption) of all the fragments is equivalent to the original relation.

$$R = R_1 \bowtie \cdots \bowtie R_n$$

Dependency Preservation

We want to preserve the information captured by the functional dependencies. The union of the projected sets of functional dependencies, Σ_1, ..., Σ_n, must be equivalent to the original set of functional dependencies, Σ.

$$\Sigma^+ = (\Sigma_1 \cup \cdots \cup \Sigma_n)^+$$

Is it the following true?

$$\Sigma_1^+ \cup \Sigma_2^+ = (\Sigma_1 \cup \Sigma_2)^+$$

Globally Redundant Attributes

This is the rationale of the Ling, Tompa, Kameda Normal Form (LTKNF).
See `https://www.comp.nus.edu.sg/~lingtw/ltk.pdf` or read T.-W. Ling, F.W. Tompa, and T. Kameda, "An Improved Third Normal Form for Relational Databases", ACM Transactions on Database Systems, 6(2), June 1981, 329-346.

Iterative Decomposition

The decomposition into any of 2NF, 3NF, EKNF or BCNF follows similar algorithms (they only differ in the test of violation).

- Given a relation R and a set of fuctional dependencies Σ.
- If a functional dependency $\sigma \in \Sigma$ violates the normal form based on Zaniolo's definitions,
- Then apply the one step decomposition to R with Σ with σ and
- Decompose the two fragments obtained;
- Otherwise R is in the normal form.

Binary Decomposition According to one Functional Dependency

The one step decomposition into any of 2NF, 3NF, EKNF or BCNF follows the same algorithm.

- Given a relation R, a set of functional dependencies Σ and a functional depency $X \rightarrow \{A\} \in \Sigma^+$.
- Compute, X^+, the closure of X with respect to Σ.
- Create the first fragment, $R_1 := X^+$, from the closure.
- Find Σ_1, the set of projected functional dependencies onto R_1.
- Create the second fragment, $R_2 := (R - X^+) \cup X$, the complement of R with respect to X (this step ensures that R_1 and R_2 naturally join on X.
- Find Σ_2, the set of projected functional dependencies onto R_2.
- Return R_1, Σ_1, R_2, Σ_2.

Example: Projection of a Set of Functional Dependencies

$R = \{A, B, C\}$
$\Sigma = \{\{A\} \rightarrow \{B\}, \{B\} \rightarrow \{C\}\}$
$R_1 = \{A, C\}$

The projection of Σ onto $R_1 = \{A, C\}$ is:

$$\Sigma_1 = \{\{A\} \rightarrow \{C\}\}$$

It is not a subset of Σ but of Σ^+!

Projecting functional dependencies is not just about the attributes!

Theorem (Heath's Theorem)

A relation R that satisfies a functional dependency $X \to Y$ can always be losslessly decomposed into its projections $R_1 = \pi_{X \cup Y}(R)$ and $R_2 = \pi_{X \cup (R \setminus Y)}(R)$.

Theorem

Any relation can be losslessly decomposed into a collection of 2NF, 3NF, EKNF or BCNF relations.

Example 1

$R = \{A, B, C, D, E\}$
$\Sigma = \{\{A, B\} \rightarrow \{C, D, F\}, \{A\} \rightarrow \{C\}, \{D\} \rightarrow \{E\}\}$

$\{A\} \rightarrow \{C\}$ violates the 2NF definition.
R with Σ is not in 2NF.

Using $\{A\} \rightarrow \{C\}$, we decompose R with Σ into R_1 with Σ_1 and R_2 with Σ_2. Σ_1 and Σ_2 are the functional dependencies of Σ on R_1 and R_2, respectively (projected fiunctional dependencies).
$\{A\}^+ = \{A, C\}$.
$R_1 = \{A, C\}$
$\Sigma_1 = \{\{A\} \rightarrow \{C\}\}$
$R_2 = \{A, B, D, E\}$
$\Sigma_2 = \{\{A, B\} \rightarrow \{D, E\}, \{D\} \rightarrow \{E\}\}$

$R_1 = \{A, C\}$ with $\Sigma_1 = \{\{A\} \to \{C\}\}$ is in 2NF, 3NF and BCNF.
$R_2 = \{A, B, D, E\}$ with $\Sigma_2 = \{\{A, B\} \to \{D, E\}, \{D\} \to \{E\}\}$ is in 2NF but not in 3NF.
It is a lossless and dependency preserving decomposition.

Example

$R_2 = \{A, B, D, E\}$
$\Sigma_2 = \{\{A, B\} \to \{D, E\}, \{D\} \to \{E\}\}$

$\{D\} \to \{E\}$ violates the 3NF definition.
R_2 with Σ_2 is not in 3NF.

Using $\{D\} \to \{E\}$, we decompose R_2 with Σ_2 into R_{21} with Σ_{21} and R_{22} with Σ_{22}.
$\{D\}^+ = \{D, E\}$.
$R_{21} = \{D, E\}$
$\Sigma_{21} = \{\{D\} \to \{E\}\}$
$R_{22} = \{A, B, D\}$
$\Sigma_{22} = \{\{A, B\} \to \{D\}\}$

$R_1 = \{A, C\}$ with $\Sigma_1 = \{\{A\} \to \{C\}\}$,
$R_{21} = \{D, E\}$ with $\Sigma_{21} = \{\{D\} \to \{E\}\}$ and
$R_{22} = \{A, B, D\}$ with $\Sigma_{22} = \{\{A, B\} \to \{D\}\}$
are in BCNF. It is a lossless and dependency preserving decomposition.

Example 2

$R = \{A, B, C, D, E, F, G\}$

$\Sigma = \{\{A\} \to \{B\}, \{A\} \to \{C\}, \{B\} \to \{C\}, \{B\} \to \{D\}, \{D\} \to \{B\}, \{A, B, E\} \to \{F\}, \{A, E\} \to \{D\}\}$

Decompose in BCNF.

Keys

There is only one key $\{A, E, G\}$.

Result

$R_1(\underline{A}, B)$
$R_2(\underline{B}, C, D)$
$R_3(\underline{A, E}, F)$
$R_4(\underline{A, E, G})$
Verify that all the relations are in BCNF. We have not lost any functional dependency.

There may be several possible decompositions: order of funtional dependencies used for decomposition.

Decomposition into BCNF can be non dependency preserving.

Decomposition in BCNF may exists but not reachable by binary decomposition.

Simple Synthesis Algorithm

Let R be a relation schema with the set of functional dependencies Σ.

1. Find an extended minimal cover, Σ_{min}, of Σ.
2. For each functional dependency $X \to Y \in \Sigma_{min}$ create a fragment with schema $X \cup Y$ and designated key X.
3. If no fragment contains a candidate key of R with Σ, find a candidate key K and create a fragment with schema K and designated key K.

Theorem

Any relation can be losslessly decomposed with the Simple Synthesis Algorithm into a collection of dependency preserving EKNF relations.

Example 1

$R = \{A, B, C, D, E, F, G\}$
$\Sigma = \{\{A\} \rightarrow \{B\}, \{A\} \rightarrow \{C\}, \{B\} \rightarrow \{C\}, \{B\} \rightarrow \{D\}, \{D\} \rightarrow \{B\}, \{A, B, E\} \rightarrow \{F\}, \{A, E\} \rightarrow \{D\}\}$

Step 1: Extended Minimal Cover

$\Sigma = \{\{A\} \rightarrow \{B\}, \cancel{\{A\} \rightarrow \{C\},} \{B\} \rightarrow \{C\}, \{B\} \rightarrow \{D\}, \{A, \cancel{B,} E\} \rightarrow \{F\} \cancel{, \{A, E\} \rightarrow \{D\}}\}$

Step 2: Construct Relations

$R_1(\underline{A}, B)$
$R_2(\underline{B}, C, D)$
$R_3(\underline{A, E}, F)$
$R_4(\underline{A, E, G})$
What kind of foreign key constraints would you declare?

Example 2

$R = \{X_1, X_2, A, B, C, D\}$
$\Sigma = \{\{X_1, X_2\} \rightarrow \{A, D\}, \{C, D\} \rightarrow \{X_1, X_2\}, \{A, X_1\} \rightarrow \{B\}, \{B, X_2\} \rightarrow \{C\}, \{C\} \rightarrow \{A\}\}$

Σ is already a minimal cover!
$R_1(\underline{X_1, X_2}, A, D)$
$R_2(\underline{C, D}, X_1, X_2)$
$R_3(\underline{A, X_1}, B)$
$R_4(\underline{B, X_2}, C)$
$R_5(\underline{C}, A)$
Verify that all relations are in 3NF.
What is wrong?

There may be several possible decompositions: there may be several minimal covers.

Superfluous attributes, Missing keys, Too many Relations.

Bernstein Algorithm

Let R be a relation schema with the set of functional dependencies F.

1. (Eliminate extraneous attributes.) Let F be the given set of FDs. Eliminate extraneous attributes from the left side of each FD in F, producing the set G. An attribute is extraneous if its elimination does not alter the closure of the set of FDs.
2. (Covering.) Find a nonredundant covering H of G.
3. (Partition.) Partition G into groups such that all of the FDs in each group have identical left sides.
4. (Merge equivalent keys.) Let $J = \emptyset$. For each pair of groups, say H_1 and H_j , with left sides X and Y, respectively, merge H_1 and $H_j$together if there is a bijection $X \leftrightarrow Y$ in H^+. For each such bijection, add $X \rightarrow Y$ and $Y \rightarrow X$ to J. For each $A \in Y$, if $X \rightarrow A$ is in H, then delete it from H. Do the same for each $X \rightarrow B$ in H with $B \in X$.
5. (Eliminate transitive dependencies.) Find an $H' \subseteq H$ such that $(H + J)^+ = (H + J)^+$ and no proper subset of H' has this property. Add each FD of J into its corresponding group of H'.
6. (Construct relations.) For each group, construct a relation consisting of all the attributes appearing in that group. Each set of attributes that appears on the left side of any FD in the group is a key of the relation. (Step 1 guarantees that no such set contains any extra attributes.) All keys found by this algorithm will be called synthesized. The set of constructed relations constitutes a schema for the given set of FDs.

Theorem

Any relation can be (not losslessly) decomposed with Bernstein Algorithm into a collection of dependency preserving EKNF relations.

Theorem

Let S be a schema synthesized from a set of FDs F by using Bernstein Algorithm. Let S' be any schema embodying a set of FDs G that covers F (equivalent to F). Then $\mid S' \mid \geq \mid S \mid$.

Example 1

$\Sigma = \{\{A\} \to \{B\}, \{A\} \to \{C\}, \{B\} \to \{C\}, \{B\} \to \{D\}, \{D\} \to \{B\}, \{A, B, E\} \to \{F\}\}$

Step 1: Extraneous Attributes

$\Sigma = \{\{A\} \rightarrow \{B\}, \{A\} \rightarrow \{C\}, \{B\} \rightarrow \{C\}, \{B\} \rightarrow \{D\}, \{D\} \rightarrow \{B\}, \{A, \cancel{B,} E\} \rightarrow \{F\}\}$

Step 2: Find Covering

$\Sigma = \{\{A\} \rightarrow \{B\}, \cancel{\{A\} \rightarrow \{C\},} \{B\} \rightarrow \{C\}, \{B\} \rightarrow \{D\}, \{D\} \rightarrow \{B\}, \{A, E\} \rightarrow \{F\}\}$

Step 3: Partition

$H_1 = \{\{A\} \rightarrow \{B\}\}$
$H_2 = \{\{B\} \rightarrow \{C\}, \{B\} \rightarrow \{D\}\}$
$H_3 = \{\{D\} \rightarrow \{B\}\}$
$H_4 = \{\{A, E\} \rightarrow \{F\}\}$

Step 4: Merge Groups

$J = \{\{B\} \rightarrow \{D\}, \{D\} \rightarrow \{B\}\}$
$H_1 = \{\{A\} \rightarrow \{B\}\}$
$H_2' = H_2 \cup H_3 - \{\{B\} \rightarrow \{D\}, \{D\} \rightarrow \{B\}\} = \{\{B\} \rightarrow \{C\}\}$
$H_4 = \{\{A, E\} \rightarrow \{F\}\}$

Step 5: Eliminate Transitive Dependencies

None!
$H_1 = \{\{A\} \rightarrow \{B\}\}$
$H_2' = \{\{B\} \rightarrow \{C\}, \{B\} \rightarrow \{D\}, \{D\} \rightarrow \{B\}\}$
$H_4 = \{\{A, E\} \rightarrow \{F\}\}$

Step 6: Construct Relations

$R_1(\underline{A}, B)$
$R_2(\underline{B}, C, \underline{D})$
$R_3(\underline{A, E}, F)$

Example 2

$\Sigma = \{\{X_1, X_2\} \rightarrow \{A, D\}, \{C, D\} \rightarrow \{X_1, X_2\}, \{A, X_1\} \rightarrow \{B\}, \{B, X_2\} \rightarrow \{C\}, \{C\} \rightarrow \{A\}\}$

Step 1: Extraneous Attributes

None!

$\Sigma = \{\{X_1, X_2\} \rightarrow \{A, D\}, \{C, D\} \rightarrow \{X_1, X_2\}, \{A, X_1\} \rightarrow \{B\}, \{B, X_2\} \rightarrow \{C\}, \{C\} \rightarrow \{A\}\}$

Step 2: Find Covering

Already!

$\Sigma = \{\{X_1, X_2\} \rightarrow \{A, D\}, \{C, D\} \rightarrow \{X_1, X_2\}, \{A, X_1\} \rightarrow \{B\}, \{B, X_2\} \rightarrow \{C\}, \{C\} \rightarrow \{A\}\}$

Step 3: Partition

$H_1 = \{\{X_1, X_2\} \rightarrow \{A, D\}\}$
$H_2 = \{\{C, D\} \rightarrow \{X_1, X_2\}\}$
$H_3 = \{\{A, X_1\} \rightarrow \{B\}\}$
$H_4 = \{\{B, X_2\} \rightarrow \{C\}\}$
$H_5 = \{\{C\} \rightarrow \{A\}\}$

Step 4: Merge Groups

$J = \{\{X_1, X_2\} \rightarrow \{C, D\}, \{C, D\} \rightarrow \{X_1, X_2\}\}$
$H_1' = H_1 \cup H_2 - J = \{\{X_1, X_2\} \rightarrow \{A\}\}$
$H_3 = \{\{A, X_1\} \rightarrow \{B\}\}$
$H_4 = \{\{B, X_2\} \rightarrow \{C\}\}$
$H_5 = \{\{C\} \rightarrow \{A\}\}$

Step 5: Eliminate Transitive Dependencies

$\{X_1, X_2\} \rightarrow \{A\}$ is a transitive dependency. Why?
$\{X_1, X_2\} \rightarrow \{C, D\}$ and $\{C\} \rightarrow \{A\}$
$J = \{\{X_1, X_2\} \rightarrow \{C, D\}, \{C, D\} \rightarrow \{X_1, X_2\}\}$
$H_1' \cup H_2 - J = \emptyset$
$H_3 = \{\{A, X_1\} \rightarrow \{B\}\}$
$H_4 = \{\{B, X_2\} \rightarrow \{C\}\}$
$H_5 = \{\{C\} \rightarrow \{A\}\}$

Step 6: Construct Relations

$R_1(\underline{X_1, X_2}, \underline{C, D})$
$R_2(\underline{A, X_1}, B)$
$R_3(\underline{B, X_2}, C)$
$R_4(\underline{C}, A)$

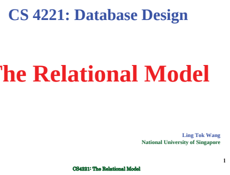
Step 6 without Steph 5: Construct Relations

$R_1(\underline{X_1, X_2}, \underline{C, D}, A)$
$R_2(\underline{A, X_1}, B)$
$R_3(\underline{B, X_2}, C)$
$R_4(\underline{C}, A)$
What is wrong?

Example

$\Sigma = \{\{X_1, X_2\} \rightarrow \{A, D\}, \{C, D\} \rightarrow \{X_1, X_2\}, \{A, X_1\} \rightarrow \{B\}, \{B, X_2\} \rightarrow \{C\}, \{C\} \rightarrow \{A\}\}$

We use slides 34-40.



https://www.comp.nus.edu.sg/~lingtw/cs4221/rm.pdf