Chapter 11

Relational Database Design Algorithms and Further Dependencies

Chapter Outline

- Designing a Set of Relations
- Properties of Relational Decompositions
- Algorithms for Relational Database Schema
- Multivalued Dependencies and Fourth Normal Form
- Join Dependencies and Fifth Normal Form
- Inclusion Dependencies
- Other Dependencies and Normal Forms

Database Design Approaches

- Top-Down design (Starting with conceptual design)
- Bottom-Up Design (relational synthesis)
Top-Down Design

- Design conceptual schema in a high-level data model
- Map the conceptual schema into a set of relations
- Normalize the relations

Relational Synthesis

- Assumes that all possible functional dependencies are known.
- First constructs a minimal set of FDs
- Then applies algorithms that construct a target set of 3NF or BCNF relations.
- Additional criteria may be needed to ensure the set of relations in a relational database are satisfactory

Designing a Set of Relations

Goals:
- Lossless join property (a must)
- Dependency preservation property – occasionally must be sacrificed to achieve 3NF or BCNF
- Additional normal forms
  - 4NF (based on multi-valued dependencies)
  - 5NF (based on join dependencies)
General Approach

- Construct a Universal relation containing ALL the attributes of the database
- Iteratively decompose the above relation into smaller relations based on the functional dependencies

Properties of Relational Decompositions

Relation Decomposition and Insufficiency of Normal Forms:

- Universal Relation Schema: a relation schema $R = \{A_1, A_2, ..., A_n\}$ that includes all the attributes of the database.
- Universal relation assumption: every attribute name is unique.
- Decomposition: The process of decomposing the universal relation schema $R$ into a set of relation schemas $D = \{R_1, R_2, ..., R_m\}$ that will become the relational database schema by using the functional dependencies.

Properties of Relational Decompositions

- Attribute preservation condition: Each attribute in $R$ will appear in at least one relation schema $R_i$ in the decomposition so that no attributes are "lost".
- Another goal of decomposition is to have each individual relation $R_i$ in the decomposition $D$ be in BCNF or 3NF.
- Additional properties of decomposition are needed to prevent the generation of spurious tuples
Properties of Relational Decompositions

Dependency Preservation Property of a Decomposition:
- Dependencies $X \rightarrow Y$ in the universal relation $R$ should either appear directly in some relation $R_i$ or can be inferred from the dependencies in $R_i$.
- It is not necessary that all dependencies from the relation $R$ appear in some relation $R_i$. It is sufficient that the union of the dependencies on all the relations $R_i$ be equivalent to the dependency on $R$.

Claim 1: It is always possible to find a dependency-preserving decomposition $D$ with respect to $F$ such that each relation $R_i$ in $D$ is in 3NF.

Lossless (Non-additive) Join Property of a Decomposition:
- A decomposition is lossless if the result of joining (natural join) all the decomposed relations together contains all the tuples from the original relation and does not contain any other tuples.
Properties of Relational Decompositions

Definition:
- **Lossless join property:** a decomposition \( D = \{ R_1, R_2, \ldots, R_m \} \) of \( R \) has the lossless (nonadditive) join property with respect to the set of dependencies \( F \) on \( R \) if, for every relation state \( r \) of \( R \) that satisfies \( F \), the following holds, where \( * \) is the natural join of all the relations in \( D \):
  \[
  * (\pi_{R_1}(r), \ldots, \pi_{R_m}(r)) = r
  \]

Note: The word loss in lossless refers to loss of information, not to loss of tuples. In fact, for loss of information a better term is “addition of spurious information.”

Successive Lossless Join Decomposition:

Claim 2 (Preservation of non-additivity in successive decompositions):
If a decomposition \( D = \{ R_1, R_2, \ldots, R_m \} \) of \( R \) has the lossless (non-additive) join property with respect to a set of functional dependencies \( F \) on \( R \), and if a decomposition \( D_i = \{ Q_1, Q_2, \ldots, Q_k \} \) of \( R_i \) has the lossless (non-additive) join property with respect to the projection of \( F \) on \( R_i \), then the decomposition \( D_2 = \{ R_1, R_2, \ldots, R_{i-1}, Q_1, Q_2, \ldots, Q_k, R_{i+1}, \ldots, R_m \} \) of \( R \) has the non-additive join property with respect to \( F \).

Algorithms for Relational Database Schema Design

Algorithm 11.2: Relational Synthesis into 3NF with Dependency Preservation (Relational Synthesis Algorithm)
Input: A universal relation \( R \) and a set of functional dependencies \( F \) on the attributes of \( R \).
1. Find a minimal cover \( G \) for \( F \);
2. For each left-hand-side \( X \) of a functional dependency that appears in \( G \), create a relation schema in \( D \) with attributes \( X \cup \{ A_1 \} \cup \{ A_2 \} \cup \ldots \cup \{ A_k \} \), where \( X \rightarrow A_1, X \rightarrow A_2, \ldots, X \rightarrow A_k \) are the only dependencies in \( G \) with \( X \) as left-hand-side (\( X \) is the key of this relation);
3. Place any remaining attributes (that have not been placed in any relation) in a single relation schema to ensure the attribute preservation property.

Claim 3: Every relation schema created by Algorithm 11.2 is in 3NF.
Algorithm 11.3: Relational Decomposition into BCNF with Lossless (non-additive) Join property
Input: A universal relation $R$ and a set of functional dependencies $F$ on the attributes of $R$.
1. Set $D := \{ R \}$;
2. While there is a relation schema $Q$ in $D$ that is not in BCNF do {
   choose a relation schema $Q$ in $D$ that is not in BCNF;
   find a functional dependency $X \rightarrow Y$ in $Q$ that violates BCNF;
   replace $Q$ in $D$ by two relation schemas ($Q - Y$) and ($X \cup Y$);
};

Algorithm 11.4: Relational Synthesis into 3NF with Dependency Preservation and Lossless (Non-Additive) Join Property
Input: A universal relation $R$ and a set of functional dependencies $F$ on the attributes of $R$.
1. Find a minimal cover $G$ for $F$ (Use Algorithm 10.2).
2. For each left-hand-side $X$ of a functional dependency that appears in $G$, create a relation schema in $D$ with attributes $\{X \cup (A_1) \cup (A_2) \ldots \cup (A_n)\}$, where $X \rightarrow A_1, X \rightarrow A_2, \ldots, X \rightarrow A_n$ are the only dependencies in $G$ with $X$ as left-hand-side ($X$ is the key of this relation).
3. If none of the relation schemas in $D$ contains a key of $R$, then create one more relation schema in $D$ that contains attributes that form a key of $R$.

Algorithm 11.4a: Finding a Key $K$ for $R$ Given a set $F$ of Functional Dependencies
Input: A universal relation $R$ and a set of functional dependencies $F$ on the attributes of $R$.
1. Set $K := R$.
2. For each attribute $A$ in $K$ {
   compute $(K - A)^*$ with respect to $F$;
   if $(K - A)^*$ contains all the attributes in $R$,
   then set $K := K - \{ A \}$; }

Issues with null-value joins. (a) Some EMPLOYEE tuples have null for the 
join attribute DNUM.

<table>
<thead>
<tr>
<th>EMPLOYEE</th>
<th>ISSUE</th>
<th>DATE</th>
<th>DEPARTMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith, John B</td>
<td>1234567890</td>
<td>1965-01-01</td>
<td>731 Fokker, Houston, TX</td>
</tr>
<tr>
<td>Wang, Frank T</td>
<td>1234567890</td>
<td>1965-01-01</td>
<td>731 Fokker, Houston, TX</td>
</tr>
<tr>
<td>Zhang, Li</td>
<td>1234567890</td>
<td>1965-01-01</td>
<td>731 Fokker, Houston, TX</td>
</tr>
</tbody>
</table>

Issues with null-value joins. (b) Result of applying NATURAL JOIN to the EMPLOYEE 
and DEPARTMENT relations. (c) Result of applying LEFT OUTER JOIN to EMPLOYEE 
and DEPARTMENT.

The "dangling tuple" problem. (a) The relation EMPLOYEE_1 (includes all 
attributes of EMPLOYEE from figure 11.2a except DNUM).
Algorithms for Relational Database Schema Design

The “dangling tuple” problem. (b) The relation EMPLOYEE_2 (includes DNUM attribute with null values). (c) The relation EMPLOYEE_3 (includes DNUM attribute but does not include tuples for which DNUM has null values).

Problems with Normalization Algorithms

- The database designer must first specify all the relevant functional dependencies among the database attributes.
- These algorithms are not deterministic in general.
- It is not always possible to find a decomposition into relation schemas that preserves dependencies and allows each relation schema in the decomposition to be in BCNF (instead of 3NF as in Algorithm 11.4).

Fourth Normal Form

**Definition:**

A relation schema R is in **4NF** with respect to a set of dependencies \( F \) (that includes functional dependencies and multivalued dependencies) if, for every nontrivial multivalued dependency \( X \rightarrow\leftrightarrow Y \) in \( F' \), \( X \) is a superkey for R.

Note: \( F' \) is the (complete) set of all dependencies (functional or multivalued) that will hold in every relation state \( r \) of \( R \) that satisfies \( F \). It is also called the closure of \( F \).
Fourth Normal Form

(a) The EMP relation with two MVDs: ENAME —> PNAME and ENAME —> DNAME.
(b) Decomposing the EMP relation into two 4NF relations EMP_PROJECTS and EMP_DEPENDENTS.

<table>
<thead>
<tr>
<th>EMP</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ENAME</td>
<td>DNAME</td>
</tr>
<tr>
<td>Smith</td>
<td>X</td>
<td>John</td>
</tr>
<tr>
<td>Smith</td>
<td>Y</td>
<td>Anna</td>
</tr>
<tr>
<td>Smith</td>
<td>Y</td>
<td>Anna</td>
</tr>
<tr>
<td>Smith</td>
<td>Y</td>
<td>John</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMP_PROJECTS</th>
<th>EMP_DEPENDENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>ENAME</td>
<td>PNAME</td>
</tr>
<tr>
<td>Smith</td>
<td>X</td>
</tr>
<tr>
<td>Smith</td>
<td>Y</td>
</tr>
</tbody>
</table>