Relational Database Design

Jan Chomicki
University at Buffalo

Outline

- Functional dependencies
- Normal forms
- Multivalued dependencies
“Good” and “bad” database schemas

**“Bad” schema**
- Repetition of information. Leads to redundancies, potential inconsistencies, and update anomalies.
- Inability to represent information. Leads to anomalies in insertion and deletion.

**“Good” schema**
- relation schemas in normal form (redundancy- and anomaly-free): BCNF, 3NF.

Schema decomposition
- improving a bad schema
- desirable properties:
  - lossless join
  - dependency preservation

Integrity constraints

Functional dependencies
- key constraints cannot express uniqueness properties holding in a proper subset of all attributes
- key constraints need to be generalized to functional dependencies

Other constraints
- not relevant for decomposition
- need to be accounted for
Functional dependencies (FDs)

Notation
- Relation schema $R(A_1, \ldots, A_n)$
- $r$ is an instance of $R$
- Sets of attributes of $R$: $X, Y, Z, \ldots \subseteq \{A_1, \ldots, A_n\}$
- $A_1 \cdots A_n$ instead of $\{A_1, \ldots, A_n\}$
- $XY$ instead of $X \cup Y$.

Functional dependency
- syntax: $X \rightarrow Y$
- semantics: $r$ satisfies $X \rightarrow Y$ if for all tuples $t_1, t_2 \in r$:
  \[
  \text{if } t_1[X] = t_2[X], \text{ then also } t_1[Y] = t_2[Y].
  \]

Dependency implication

Implication
A set of FDs $F$ implies an FD $X \rightarrow Y$, if every relation instance that satisfies all the dependencies in $F$, also satisfies $X \rightarrow Y$.

Notation
$F \models X \rightarrow Y$ ($F$ implies $X \rightarrow Y$).

Closure of a dependency set $F$
The set of dependencies implied by $F$.

Notation
$F^+ = \{X \rightarrow Y : F \models X \rightarrow Y\}$. 
Keys

Key

A set $X \subseteq \{A_1, \ldots, A_n\}$ is a key of $R$ if:
1. The dependency $X \rightarrow A_1 \cdots A_n$ is in $F^+$.
2. For all proper subsets $Y$ of $X$, the dependency $Y \rightarrow A_1 \cdots A_n$ is not in $F^+$.

Related notions

- **superkey**: superset of a key.
- **primary key**: one designated key.
- **candidate key**: one of the keys.

Inference of functional dependencies

Dependency inference

How to tell whether $X \rightarrow Y \in F^+$?

Inference rules (Armstrong axioms)

- **reflexivity**: infer $X \rightarrow Y$ if $Y \subseteq X \subseteq \text{attrs}(R)$ (trivial dependency)
- **augmentation**: From $X \rightarrow Y$ infer $XZ \rightarrow YZ$ if $Z \subseteq \text{attrs}(R)$
- **transitivity**: From $X \rightarrow Y$ and $Y \rightarrow Z$, infer $X \rightarrow Z$. 
Properties of axioms

Armstrong axioms are:
- **sound**: if \( X \rightarrow Y \) is derived from \( F \), then \( X \rightarrow Y \in F^+ \).
- **complete**: if \( X \rightarrow Y \in F^+ \), then \( X \rightarrow Y \) is derived from \( F \).

Additional (implied) inference rules
4. **union**: from \( X \rightarrow Y \) and \( X \rightarrow Z \), infer \( X \rightarrow YZ \)
5. **decomposition**: from \( X \rightarrow Y \) infer \( X \rightarrow Z \), if \( Z \subseteq Y \)

Boyce-Codd Normal Form (BCNF) and 3NF

**BCNF**
A schema \( R \) is in BCNF if for every nontrivial FD \( X \rightarrow A \in F \), \( X \) contains a key of \( R \).

Each instance of a relation schema which is in BCNF does not contain a redundancy (that can be detected using FDs alone).

**3NF**
\( R \) is in 3NF if for every nontrivial FD \( X \rightarrow A \in F \):
- \( X \) contains a key of \( R \), or
- \( A \) is part of some key of \( R \).

**BCNF vs. 3NF**
- if \( R \) is in BCNF, it is also in 3NF
- there are relations that are in 3NF but not in BCNF.
Decompositions
We will identify a relation schema with its set of attributes.

Decomposition
Replacement of a relation schema \( R \) by two relation schema \( R_1 \) and \( R_2 \) such that \( R = R_1 \cup R_2 \).

Lossless-join decomposition
\((R_1, R_2)\) is a lossless-join decomposition of \( R \) with respect to a set of FDs \( F \) if for every instance \( r \) of \( R \) that satisfies \( F \):
\[
\pi_{R_1}(r) \times \pi_{R_2}(r) = r.
\]

A simple criterion for checking whether a decomposition \((R_1, R_2)\) is lossless-join:
- \( R_1 \cap R_2 \rightarrow R_1 \in F^+ \), or
- \( R_1 \cap R_2 \rightarrow R_2 \in F^+ \).

A sequence of decompositions of \( R \) into \( R_1 \) and \( R_2 \), \( R_1 \) into \( R_1' \) and \( R_1'' \) etc. may be viewed as a decomposition of \( R \) into more than two relation schemas.

Dependency preservation

Dependencies associated with relation schema \( R_1 \) and \( R_2 \) in a decomposition \((R_1, R_2)\):
\[
F_{R_1} = \{ X \rightarrow Y | X \rightarrow Y \in F^+ \land XY \subseteq R_1 \}
\]
\[
F_{R_2} = \{ X \rightarrow Y | X \rightarrow Y \in F^+ \land XY \subseteq R_2 \}.
\]

\((R_1, R_2)\) preserves a dependency \( f \) iff \( f \in (F_{R_1} \cup F_{R_2})^+ \).
Decomposition into BCNF

Algorithm: decomposition of schema $R$

1. For some nontrivial nonkey dependency $X \rightarrow A$ in $F^+$:
   - create a relation schema $R_1$ with the set of attributes $XA$ and FDs $F_{R_1}$.
   - create a relation schema $R_2$ with the set of attributes $R - \{A\}$ and FDs $F_{R_2}$.
2. Decompose further the resulting schemas which are not in BCNF.

This algorithm produces a lossless-join decomposition into BCNF which does not have to preserve dependencies.

Decomposition (synthesis) into 3NF

Minimal basis $F'$ for $F$

1. set of FDs equivalent to $F$ ($F^+ = (F')^+$),
2. all FDs in $F'$ are of the form $X \rightarrow A$ where $A$ is a single attribute,
3. further simplification by removing dependencies or attributes from dependencies in $F'$ yields a set of FDs inequivalent to $F$.

Algorithm: 3NF synthesis

1. Create a minimal basis $F'$.
2. Create a relation with attributes $XA$ for every dependency $X \rightarrow A \in F'$.
3. Create a relation $X$ for some key $X$ of $R$.
4. Remove redundancies.

This algorithm produces a lossless-join decomposition into 3NF which preserves dependencies.
Multivalued dependencies (MVDs)

**Notation**
- Relation schema \( R(A_1, \ldots, A_n) \).
- \( r \) is an instance of \( R \).
- Sets of attributes: \( X, Y, Z, \ldots \subseteq \{A_1, \ldots, A_n\} \).

**Multivalued dependency**
- Syntax: a pair \( X \rightarrow\rightarrow Y \).
- Semantics: \( r \) satisfies \( X \rightarrow\rightarrow Y \) if for all tuples \( t_1, t_2 \in r \):
  \[
  \text{if } t_1[X] = t_2[X], \text{ then there is a tuple } t_3 \in r \text{ such that } t_3[XY] = t_1[XY] \text{ and } t_3[Z] = t_2[Z],
  \]
  where \( Z = \{A_1, \ldots, A_n\} - XY \).

**Implication**
Defined in the same way as for FDs.

Fourth Normal Form (4NF)

\( F \) is the set of FDs and MVDs associated with a relation schema \( R = \{A_1, \ldots, A_n\} \).

4NF
\( R \) is in 4NF if for every multivalued dependency \( X \rightarrow\rightarrow Y \) entailed by \( F \):
- \( Y \subseteq X \) or \( XY = \{A_1, \ldots, A_n\} \) (trivial MVD), or
- \( X \) contains a key of \( R \).