Principles of Relational Design

Chapter 12

Things can go wrong!

- All the relational designs we've seen so far have been pretty good
  - fairly small, intuitive examples
  - E/R model not far away
- A carelessly designed schema can lead to big problems
- How do we evaluate a schema?
- How do we design a good one?

Plan of Attack

- Study some informal principles (12.1)
- Functional dependencies: an important type of semantic constraint
  - define and illustrate (12.2)
  - use to define "normal forms" 2NF, 1NF, and BCNF (12.3-12.5)
- Decomposition algorithms (13.1)
- Multivalued dependencies and 4NF (13.2)
- Join dependencies and 5NF (13.3)

Informal Guidelines by Elmasri and Navathe

- Design a relation so that it is easy to explain its meaning.
- Design so that no insertion, deletion, or modification anomalies can occur.
- Avoid attributes whose values can be null.
- Design so that reasonable joins do not produce spurious tuples.

The "Universal Relation" Approach

- Assume we have identified all the individual pieces of data (attributes) of the problem.

The database design problem: group the attributes into relations.

- The informal guidelines are one way of evaluating the result.
- The theory of functional dependencies and normal forms gives a more precise way.

Functional dependency defined

- Let X and Y be attributes
- X → Y means that Y is a function of Y.
  I.e., if you know the value of X, there's only one possible value of Y. We say that "Y is functionally dependent on X" or "X determines Y."
- Note: X → Y does not imply Y → X!
Examples

- If you know the SSN, there's only one possible name (is the reverse true?)
  \[ \text{SSN} \rightarrow \text{NAME} \]
- If you know the department number, you know the department name
  \[ \text{DNO} \rightarrow \text{DNAME} \]

Dependencies between sets of attributes

- Given today's date and the date of birth, the age and the years until 65 are determined.
  \[ \{ \text{TD, DB} \} \rightarrow \{ \text{AGE, YTO65} \} \]
- If you know the file pathname, you can determine its size, owner, and date of last modification.
  \[ \text{FP} \rightarrow \{ \text{SZ, O, MDT} \} \]
- Normally, when writing \( X \rightarrow Y \), we assume that \( X \) and \( Y \) are sets of attributes.

Facts about FD

- FDs are purely semantic in nature
- FDs are facts about the abstract relation, not just about a particular relation instance
  - They must hold for all possible legal instances of a relation
- All attributes of a relation are functionally dependent upon its key!
  - In fact, we can formally define keys in terms of FDs.

Inference Rules for FDs

- Given a set of FDs, it may be possible to deduce others by purely syntactic means.
- Example: Given that \( \{ A \} \rightarrow \{ B, C \} \), it follows that \( \{ A \} \rightarrow \{ B \} \) also (and that \( \{ A \} \rightarrow \{ C \} \))

Armstrong’s and other rules

- Armstrong's rules (provable from 1st principles):
  - IR1. Reflexive rule
  - IR2. Augmentation rule
  - IR3. Transitive rule
- Some other rules (provable from IR1-2):
  - IR4. Decomposition
  - IR5. Union (additive) rule
  - IR6. Pseudotransitivity

Closures

- \( X^+ \) is the "closure" of \( X \): the set of all attributes functionally determined by \( X \)
  (given a set of FDs)