CS 1555

Lecture 20

**Database System Design**

Design

- Database design: activity of specifying the schema of a database in a given data model

- Functional design

- High level specification of transactions

- DBMS independent

- Event diagrams, UML

- Application program design

- DBMS specific

- Language and environment specific

Relational database design

- One single, large table: simple but cumbersome

Goal

- Want tables where the attributes depend on the primary key, on the whole key, and nothing but the key

Pitfalls

- Redundance: waste of space (repeated values, NULL), update anomalies (inconsistencies), insertion/deletion anomalies

Solution: decomposition

- Split offending table in two (or more)

Functional dependencies

- How do we split/recognize a bad table?

- Set of rules: normal forms

- Functional dependencies (FD)

- X 🡪 Y if the value of X uniquely determines Y (X is the domain for the function, Y is the range)

- Not a property of a particular relation state

- Property of R, not r(R)

- Cannot be automatically inferred from r(R)

Decompositions

- There are careless, bad decompositions

- Correct decompositions are lossless, dependency-preserving

- Non-lossy decomposition is called lossless or non-additive decompositions

- Lossy decompositions cannot recover the original table with a join

Theorem

- A decomposition is lossless if the joining attribute is a part of a superkey in at least one of the new tables

Dependency preservation & canonical cover

- Don’t want the original FDs to span two tables

- More specifically, the FDs of the canonical cover

- Canonical cover is the minimum set of FDs without any trivial, extraneous, and implied FDs

- Ex: A 🡪 B, B 🡪 C, AB 🡪 C

- Canonical cover is A🡪 B & B 🡪 C

Rules of inference: Armstrong’s axioms

- Reflexivity: if B is a subset of A, then A 🡪 B

- Augmentation: if A 🡪 B, then AC 🡪 BC

- Transitive: if A 🡪 B and B 🡪 C, then A 🡪 C

- Decomposition: if A 🡪 BC, then A 🡪 B and A 🡪 C

- Union: if A 🡪 B and A 🡪 C, then A 🡪 BC

- Pseudotransitivity, composition: if A 🡪 B and CB 🡪 D, then CA 🡪 D

- Self-determination: A 🡪 A

Dependency-preserving decomposition

- The original (canonical) FDs should not span across tables

Summary

- Decompositions should always be lossless

- Should be dependency preserving whenever possible