CS 1555

Lecture 21

**Database System Design (continued)**

- Functional dependency is a semantic notion

- Good database schema

- Relations should have simple meaning

- Bad schema: relations have multiple meanings

- Types of functional dependencies

- Trivial FD: X 🡪 Y is trivial if it is true for any X and Y of any relation, regardless of semantics

- Partial FD: X 🡪 Y is partial if there is an attribute A in X that can be removed from X and the dependency can still hold

- Full FD: can’t remove any part of X and be able to determine Y

- Transitive FD: X 🡪 Y is transitive if there is a set of attributes Z that is not a subset of any key of R and both X 🡪 Z and Z 🡪 Y hold

- Multivalued FD: If X is a key of R, Z is in R and Z 🡪 Y

First normal form (1NF)

- Every attribute has a single atomic value

- Not in second normal form

Second normal forms (2NF)

- In 1NF and does not have partial dependencies

Third normal forms (3NF)

- In 2NF and does not have transitive dependencies to attributes that are not part of a key

- If X 🡪 A is an FD then a) it is trivial, b) X is a superkey, or c) A is a subset of candidate key

Normal forms – BCNF

- Boyce-Codd normal form: A relation is in 3NF and has no transitive dependencies

- Theorem: given a schema R and a set of FD ‘F’, we can always decompose it to schemas R1, …, Rn, such that

- R1, …, Rn are in BCNF and

- the decompositions are lossless

- But some decompositions might lose dependencies 🡪 use 3NF

Normal forms – 4NF

- A relation is in BCNF and has no multivalue dependencies

Universal relational approach

- One single, large table 🡪 normalize it

The normalization process

- Finding good (stable) set of relations that is a faithful model of the enterprise

- Decomposition (top-down process)

- Start with universal relation

- Identify FDs

- Identify keys

- Use decomposition to split into set of relations