ICS 321 Fall 2009

Schema Refinement & Normal Forms

Asst. Prof. Lipyeow Lim

Information & Computer Science Department

University of Hawaii at Manoa

The Problem with Redundancy

Hourly_Emps

SSN	Name	Lot	Rating	Hourly_wages	Hours_worked
123-22-2366	Attishoo	48	8	10	40
231-31-5368	Smiley	22	8	10	30
131-24-3650	Smethurst	35	5	7	30
434-26-3751	Guldu	35	5	7	32
612-67-4134	Madayan	35	8	10	40

- Suppose hourly wages are determined by rating
- **Redundant storage** : (8,10) stored multiple times
- **Update anomaly** : change hourly wages in row 1
- **Insertion anomaly** : requires knowing hourly wages for the rating
- **Deletion anomaly** : deleting all (8,10) loses info

Using Two Smaller Tables

Hourly_Emps

<u>SSN</u>	Name	Lot	Rating	Hours_ worked
123-22-2366	Attishoo	48	8	40
231-31-5368	Smiley	22	8	30
131-24-3650	Smethurst	35	5	30
434-26-3751	Guldu	35	5	32
612-67-4134	Madayan	35	8	40

RatingWages

Rating	Hourly_ wages
5	7
8	10

- *<u>Notation</u>*: denote relation schema by listing the attributes SNLRWH
- **Update anomaly** : Can we change W for Attishoo?
- **Insertion anomaly** : What if we want to insert an employee and don't know the hourly wage for his rating?
- **Deletion anomaly** : If we delete all employees with rating 5, do we lose the information about the wage for rating 5?

Decomposition

Hourly_Emps

<u>SSN</u>	Name	Lot	Rating	Hours_ worked
123-22-2366	Attishoo	48	8	40
231-31-5368	Smiley	22	8	30
131-24-3650	Smethurst	35	5	30
434-26-3751	Guldu	35	5	32
612-67-4134	Madayan	35	8	40

RatingWages

Rating	Hourly_ wages
5	7
8	10

- Remove redundancy by decomposition
  - Since hourly wage is completely determined by rating, factor out hourly wage.
- Pros: less redundancy less anomalies
- Cons: retrieving the hourly wage of an employee requires a join

Functional Dependency

- A functional dependency X -> Y holds over relation R if, for every allowable instance *r* of R:
  - for all tuples *t1,t2* in *r,*

    $$\pi_X(t1) = \pi_X(t2) \ \text{ implies } \ \pi_Y(t1) = \pi_Y(t2)$$

  - i.e., given two tuples in *r*, if the X values agree, then the Y values must also agree. (X and Y are *sets* of attributes.)
- An FD is a statement about *all* allowable relations.
  - Must be identified based on semantics of application.
  - Given some allowable instance *r1* of R, we can check if it violates some FD *f*, but we cannot tell if *f* holds over R!
- K is a candidate key for R means that K -> R
  - However, K -> R does not require K to be *minimal*!

FD Example

Hourly_Emps

<u>SSN</u>	Name	Lot	Rating	Hourly_wages	Hours_worked
123-22-2366	Attishoo	48	8	10	40
231-31-5368	Smiley	22	8	10	30
131-24-3650	Smethurst	35	5	7	30
434-26-3751	Guldu	35	5	7	32
612-67-4134	Madayan	35	8	10	40

- Two FDs on Hourly_Emps:
  - *ssn* is the key: S -> SNLRWH
  - *rating* determines *hourly_wages*: R -> W

Reasoning about FDs

- Given some FDs, we can usually infer additional FDs:
  - *ssn -> did*, *did -> lot* implies *ssn -> lot*
- Armstrong's Axioms
  - Let X, Y, Z are sets of attributes:
  - *Reflexivity*: If X is a subset of Y, then Y -> X
  - *Augmentation*: If X -> Y, then XZ -> YZ for any Z
  - *Transitivity*: If X -> Y and Y -> Z, then X -> Z
- These are *sound* and *complete* inference rules for FDs!

Example: Armstrong's Axioms

Hourly_Emps

SSN	Name	Lot	Rating	Hourly_Wages	Hours_worked
123-22-2366	Attishoo	48	8	10	40
231-31-5368	Smiley	22	8	10	30
131-24-3650	Smethurst	35	5	7	30
434-26-3751	Guldu	35	5	7	32
612-67-4134	Madayan	35	8	10	40

- *Reflexivity*: If X is a subset of Y, then Y -> X
  - SNLR is a subset of SNLRWH, SNLRWH -> SNLR
- *Augmentation*: If X -> Y, then XZ -> YZ for any Z
  - S -> N, then SLR -> NLR
- *Transitivity*: If X -> Y and Y -> Z, then X -> Z
  - S -> R, R -> W, then S -> W