CS 235: Introduction to Databases

Svetlozar Nestorov

Lecture Notes #8

Why Decomposition "Works"?

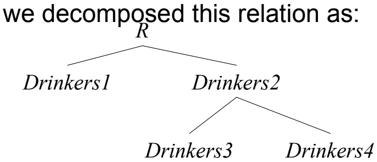
- What does it mean to "work"? Why can't we just tear sets of attributes apart as we like?
- Answer: the decomposed relations need to represent the same information as the original.
  - We must be able to reconstruct the original from the decomposed relations.
- Projection and join connect the original and decomposed relations

Example (1/3)

R =

name	*addr*	*beersLiked*	*manf*	*favoriteBeer*
Mike	111 E Ohio	Bud	A.B.	Blonde Ale
Mike	111 E Ohio	Blonde Ale	G.I.	Blonde Ale
Anna	123 W Grand	BudLite	A.B.	BudLite

- Recall we decomposed this relation as:



Example (2/3)

Project onto *Drinkers1(<u>name</u>, addr, favoriteBeer):*

name	*addr*	*favoriteBeer*
Mike	111 E Ohio	Blonde Ale
Anna	123 W Grand	BudLite

Project onto *Drinkers3(<u>beersLiked</u>, manf):*

beersLiked	*manf*
Bud	A.B.
Blonde Ale	G.I.
BudLite	A.B.

Example (3/3)

Project onto *Drinkers4(<u>name</u>, <u>beersLiked</u>):*

name	*beersLiked*
Mike	Bud
Mike	Blonde Ale
Anna	BudLite

Reconstruction

- Can we figure out the original relation from the decomposed relations?
- Sometimes, if we natural join the relations.

Example

Drinkers3 ⋈ *Drinkers4* =

name	*beersLiked*	*manf*
Mike	Bud	A.B.
Mike	Blonde Ale	G.I.
Anna	BudLite	A.B.

- Join of above with *Drinkers1* = original *R*.

Theorem

- Suppose we decompose a relation with schema *XYZ* into *XY* and *XZ* and project the relation for *XYZ* onto *XY* and *XZ*. Then *XY* ⋈ *XZ* is guaranteed to reconstruct *XYZ* if and only if $X \rightarrow\rightarrow Y$ (or equivalently, $X \rightarrow\rightarrow Z$).
- Usually, the MVD is really a FD, $X \rightarrow Y$ or $X \rightarrow Z$.

Implications

- BCNF: When we decompose *XYZ* into *XY* and *XZ*, it is because there is a FD $X \rightarrow Y$ or $X \rightarrow Z$ that violates BCNF.
  - Thus, we can always reconstruct *XYZ* from its projections onto *XY* and *XZ*.
- 4NF: when we decompose *XYZ* into *XY* and *XZ*, it is because there is an MVD $X \rightarrow\rightarrow Y$ or $X \rightarrow\rightarrow Z$ that violates 4NF.
  - Again, we can reconstruct *XYZ* from its projections onto *XY* and *XZ*.

Bag Semantics

- A relation (in SQL, at least) is really a *bag*.
- It may contain the same tuple more than once, although there is no specified order (unlike a list).
- Example: {1,2,1,3} is a bag and not a set.
- Select, project, and join work for bags as well as sets.
  - Just work on a tuple-by-tuple basis, and don't eliminate duplicates.

Bag Operations

- Union: sum the times an element appears in the two bags.
- Example: $\{1,2,1\} \cup \{1,2,3,3\} = \{1,1,1,2,2,3,3\}$.
- Intersection: take the minimum of the number of occurrences in each bag.
- Example: $\{1,2,1\} \cap \{1,2,3,3\} = \{1,2\}$
- Difference: subtract the number of occurrences in the two bags.
- Example: $\{1,2,1\} - \{1,2,3,3\} = \{1\}$.

Different Laws for Bags

- Some familiar laws continue to hold for bags.
  - Examples: union and intersection are still commutative and associative.
- But other laws that hold for sets do *not* hold for bags!

Example

- $R \cap (S \cup T) \equiv (R \cap S) \cup (R \cap T)$ holds for sets but not for bags!
- Let R, S, and T each be the bag {1}.
- Left side: $S \cup T = \{1,1\}$; $R \cap (S \cup T) = \{1\}$.
- Right side: $R \cap S = R \cap T = \{1\}$; $(R \cap S) \cup (R \cap T) = \{1\} \cup \{1\} = \{1,1\} \neq \{1\}$.

Extended Relational Algebra

- Adds features needed for SQL, bags.
- Duplicate-elimination operator δ.
- Extended projection.
- Sorting operator τ.

Duplicate Elimination

- $\delta(R)$ = relation with one copy of each tuple that appears one or more times in *R*.

Beers

beer	*price*
Amstel	4
Guinness	6
Guinness	7
Guinness	7
Bud	5

δ(*Beers*)

beer	*price*
Amstel	4
Guinness	6
Guinness	7
Bud	5

Sorting

- $\tau_L(R)$ = list of tuples of *R*, ordered according to attributes on list *L*.
- Note that result type is outside the normal types (set or bag) for relational algebra.
  - Consequence: τ cannot be followed by other relational operators.

Extended Projection

- Allow the columns in the projection to be functions of one or more columns in the argument relation.

Beers

beer	*cost*	*price*
Amstel	2	4
Guinness	4	6
Guinness	4	7
Guinness	4	8
Bud	1	5

$\pi_{price,price,price\text{-}cost}(Beers)$

price1	*price2*	*price-cost*
4	4	2
6	6	2
7	7	3
8	8	4
5	5	4

Sad Drinkers Example

- Find all drinkers that only frequent bars that do not sell their favorite beer.

Sells(bar, beer, price)
Bars(name, addr)
Frequents(drinker, bar)
Drinker(name, addr, favBeer)