Lecture 10

Relational Algebra

Week 5

Overview

- Relational algebra as a query language
- Selection, projection, cross product
- Union / intersection / set difference
- Attribute renaming
- Join
  - – join
  - – equijoin and natural join
- Division
- Simplifying relational algebra expressions

Relational algebra as a query language

- *Objects*: relations (sets of tuples)
- *Operations* on relations result relations
- Relational algebra expressions can be evaluated, and the result is always a relation
- We can assign the result to a temporary relation

Selection

$\sigma_C R$

select tuples from relation R
where each tuple satisfies condition C

Example $\sigma_{\text{salary} > 70000}$ Employee

Evaluates to a relation which is a subset of Employee, those employees who earn >70,000

Employee

78000
58000
72000
36000
41000
92000
70500
41300
38950

$\sigma_{\text{salary} > 70000}$ Employee

78000
72000
72000
70500

Properties of selection

- number of tuples:
  $|\sigma_C R| \leq |R|$
- number of attributes:
  degree $(\sigma_C R)$ = degree (R)
- commutative
  $\sigma_C\ \sigma_D R = \sigma_D\ \sigma_C R\ (= \sigma_{C \text{ and } D} R)$

Projection

$\Pi_L\, R$

take a subset of the attributes of a relation R, where $L \subseteq R$; and then eliminate duplicates

Example: $\Pi_{\text{name, salary}}$ Employee

Evaluates to a relation with Employee names and salaries in it

Employee

ssn	name	salary
a	John	20000
b	Mary	20000
c	John	20000
d	Mary	20000

$\Pi_{\text{name, salary}}$ Employee

name	salary
John	20000
Mary	20000
~~John~~	~~20000~~
~~Mary~~	~~20000~~

Employee

ssn	name	salary
a	John	20000
b	Mary	20000
c	John	20000
d	Mary	20000

$\Pi_{\text{name, salary}}$ Employee

name	salary
John	20000
Mary	20000
~~John~~	~~20000~~
~~Mary~~	~~20000~~

If SSN was a primary key… Are these real duplicates?

So…was dropping the primary key a good idea?

Properties of projection

- number of tuples:
  $|\Pi_L\, R| \leq |\, R\,|$
- number of attributes:
  degree $(\Pi_L\, R) \leq$ degree (R)
- *not* commutative
  $\Pi_L\, \Pi_S\, R \neq \Pi_S\, \Pi_L\, R$ (Note: $L \subseteq S$, but $S \not\subseteq L$)
- $\Pi_L\, \Pi_S\, R = \Pi_S\, \Pi_L\, R$ if $L \subseteq S$

Note on constraint notation

- Remember the notation we used to describe foreign key constraints, e.g.
  E(<u>k</u>,c)
  A(<u>k</u>,a) **fk: k is k in E**
  meaning that every k-value in A must be an actual k-value in E.
- We could have used relational algebra and logic to represent this constraint, i.e.
  k is k in E
  could have been written as
  $\Pi_k\, A \subseteq \Pi_k\, E$

Set operations

$A \cup B$ Union

$A \cap B$ Intersection

$A \setminus B$ Difference

$A \times B$ Cartesian product

For union, intersection, and difference relations A and B must be *union compatible*:

- degree (A) = degree (B)
  operands have the same degree
- domain (A_i) = domain (B_i)
  (i.e. corresponding attributes must have the same value domain)

Example

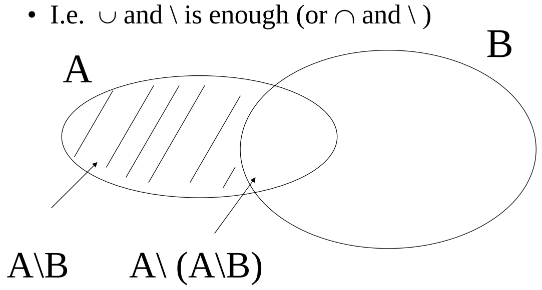
"Employees who work for department 5 *and* have salary >70000"

$\sigma_{DNO=5}$ Employee $\cap$ $\sigma_{SALARY>70000}$ Employee

Properties of union, intersection, difference

- number of tuples in intersection:
  $|A \cap B| \leq$ Min ($|A|$, $|B|$)
- number of tuples in union:
  $|A \cup B| \leq |A| + |B|$
- number of tuples in difference:
  $|A \setminus B| \leq |A|$
- degree (A op B) = degree (A) = degree (B)

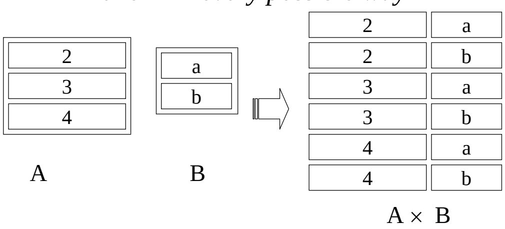
- Note that not all operations are necessary:
  $A \cap B = A \setminus (A \setminus B)$
- I.e. $\cup$ and \ is enough (or $\cap$ and \ )



Cartesian product

$A \times B$

concatenate the attributes of all tuples in A and B in *every possible way*



Properties of cartesian product

- number of tuples in product:

  $| A \times B | = |A| * |B|$

  I.e. every possible combination is produced

- degree ($A \times B$) = degree (A) + degree (B)

Unary and binary operations

- Unary operations bind more strongly than binary ones, hence paratheses can be omitted

$(\sigma_{dno=5}\, Employee) \cap (\sigma_{salary>70000}\, R) =$
$\sigma_{dno=5}\, Employee \cap \sigma_{salary>70000}\, R$

Attribute renaming

- Sometimes we need to store the result of an expression and give the attributes a different name

$Result(sssn, sname) \leftarrow \Pi_{ssn,\, name}\, Employee$

The end