CSC 370

Database Systems: Relational Algebra

What is an **algebra**?

- An algebra is a mathematical system
- Consists of:
  - **operands**: variables or values from which new values may be constructed
  - **operators**: symbols which denote a procedure/ sequence of rules for constructing new values from given values
- Example: **algebra of arithmetic**
  - atomic operands such as variables and constants
  - operators: additional, subtraction, multiplication, division
  - there are rules for creating expressions (involving operands and operators)

Motivation for relational algebra

- This will provide a notation for a suitable query language
  - Less powerful language than, say, Java
  - However, easier to write expressions in this algebra
  - Easier to apply specific transformations that improve the query
- Goal for course: Gain ability move back-and-forth between relational algebra expressions and SQL query statements

Core of Relational algebra

- Set operations
  - Union: $\bigcup$
  - Intersection: $\bigcap$
  - Set difference: -
  - Assumption: both operands must have the same relation schema
- Special operators:
  - Selection (σ): picking rows
  - Project (π): picking columns
  - Products ($\times$) and Joins ($\bowtie$): compositions of relations
  - Renaming (ρ) of relations and attributes
- (Also make note of relationship between schemas on either side of assignment)

Selection (σ)

- Example:

  R1 = σ_C(R2)

- Note that:
  - C is a conditional that refers to attributes in R2 (i.e., C is an expression that evaluates to a boolean)
  - R1 corresponds to all those tuples of R2 that satisfy C
  - C can be simple or be complex (just like a conditional in a programming-language "if" statement)
- Schemas on either side of "=" must match in name, number and domain

Selection (σ): Example

Sells

pub	beer	price
Rob's	Amnesiac	7.50
Rob's	Blue Buck	3.25
Pat's	Amnesiac	7.50
Pat's	Blue Buck	2.95

PatMenu = $\sigma_{\text{pub="Pat's"}}$(Sells)

pub	beer	price
Pat's	Amnesiac	7.50
Pat's	Blue Buck	2.95

Projection (π)

- Example:

  R1 = π_L (R2)

- Note that:
  - L is a list of attributes from the schema of R2
  - R1 is constructed by looking at each tuple of R2...
  - ... extracting the attributes on list L, in the order specified...
  - ... and creating from those components a tuple in R1.
  - Duplicate tuples (if any) are eliminated from R1
- Schemas on either side of "=" need not match

Projection (π): Example

Sells

pub	beer	price
Rob's	Amnesiac	7.50
Rob's	Blue Buck	3.25
Pat's	Amnesiac	7.50
Pat's	Blue Buck	2.95

Prices = $\pi_{\text{beer, price}}$(Sells)

beer	price
Amnesiac	7.50
Blue Buck	3.25
Blue Buck	2.95

Extended Projection

- Using the same project operator:
  - We permit the list L to contain arbitrary expression involving attributes.
  - Some of the expressions may simply be the attributes themselves
- What this permits:
  - Arithmetic on attributes (e.g., A + B -> C)
  - Duplicate occurrences of the same attribute

Extended Projection: Example

R

A	B
1	2
3	4

$\pi_{A+B\text{->}C,\ A,\ A}(R)$

C	A1	A2
3	1	1
7	3	3

Product ($\times$)

- Example:

  $U = S \times T$

- Procedure for constructing result
  - Pair each tuple s of S with each tuple t of T
  - Concatenation of "st" is a tuple of U
  - Schema of U consists of the attributes of S and then T, in that order
  - In case attribute A has the same name in both S and T, we differentiate by writing S.A and T.A
- Schemas on either side of "=" will not match