Relational Algebra

Basic Operations

Based on slides from J. Ullman
Taught by Hande Alemdar

What is an "Algebra"

- Mathematical system consisting of:
  - *Operands* --- variables or values from which new values can be constructed.
  - *Operators* --- symbols denoting procedures that construct new values from given values.

What is Relational Algebra?

- An algebra whose operands are relations or variables that represent relations.
- Operators are designed to do the most common things that we need to do with relations in a database.
  - The result is an algebra that can be used as a *query language* for relations.

Core Relational Algebra

- Union, intersection, and difference.
  - Usual set operations, but *both operands must have the same relation schema*.
- Selection: picking certain rows.
- Projection: picking certain columns.
- Products and joins: compositions of relations.
- Renaming of relations and attributes.

Selection

- R1 := σ_C(R2)
  - *C* is a condition (as in "if" statements) that refers to attributes of R2.
  - R1 is all those tuples of R2 that satisfy *C*.

Example: Selection

Relation Sells:

bar	beer	price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Miller	3.00

JoeMenu := $\sigma_{bar=\text{"Joe's"}}$(Sells):

bar	beer	price
Joe's	Bud	2.50
Joe's	Miller	2.75

Projection

- R1 := π_L (R2)
  - 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 for R1.
  - Eliminate duplicate tuples, if any.

Example: Projection

Relation Sells:

bar	beer	price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Miller	3.00

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

beer	price
Bud	2.50
Miller	2.75
Miller	3.00

Product

- R3 := R1 X R2
  - Pair each tuple t1 of R1 with each tuple t2 of R2.
  - Concatenation t1t2 is a tuple of R3.
  - Schema of R3 is the attributes of R1 and then R2, in order.
  - But beware attribute *A* of the same name in R1 and R2: use R1.*A* and R2.*A*.

Example: R3 := R1 X R2

R1(

A,	B)
1	2
3	4

R2(

B,	C)
5	6
7	8
9	10

R3(

A,	R1.B,	R2.B,	C)
1	2	5	6
1	2	7	8
1	2	9	10
3	4	5	6
3	4	7	8
3	4	9	10

Theta-Join

- R3 := R1 $\bowtie_C$ R2
  - Take the product R1 X R2.
  - Then apply σ_C to the result.
- C can be any boolean-valued condition comparing attributes of R1 to attributes of R2.

Example: Theta Join

Sells(

bar,	beer,	price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Coors	3.00

)

Bars(

name,	addr
Joe's	Maple St.
Sue's	River Rd.

)

BarInfo := Sells $\bowtie_{\text{Sells.bar = Bars.name}}$ Bars

BarInfo(

bar,	beer,	price,	name,	addr
Joe's	Bud	2.50	Joe's	Maple St.
Joe's	Miller	2.75	Joe's	Maple St.
Sue's	Bud	2.50	Sue's	River Rd.
Sue's	Coors	3.00	Sue's	River Rd.

)

Natural Join

- A useful join variant (*natural* join) connects two relations by:
  - Equating attributes of the same name, and
  - Projecting out one copy of each pair of equated attributes.
- Denoted R3 := R1 $\bowtie$ R2.

Example: Natural Join

Sells(

bar,	beer,	price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Coors	3.00

)

Bars(

bar,	addr
Joe's	Maple St.
Sue's	River Rd.

)

BarInfo := Sells $\bowtie$ Bars

Note: Bars.name has become Bars.bar to make the natural join "work."

BarInfo(

bar,	beer,	price,	addr
Joe's	Bud	2.50	Maple St.
Joe's	Millller	2.75	Maple St.
Sue's	Bud	2.50	River Rd.
Sue's	Coors	3.00	River Rd.

)

Renaming

- The ρ operator gives a new schema to a relation.
- R1 := $\rho_{R1(A1,\ldots,An)}$(R2) makes R1 be a relation with attributes A1,…,A*n* and the same tuples as R2.
- Simplified notation: R1(A1,…,A*n*) := R2.

Example: Renaming

Bars(

name,	addr
Joe's	Maple St.
Sue's	River Rd.

)

R(bar, addr) := Bars

R(

bar,	addr
Joe's	Maple St.
Sue's	River Rd.

)

Building Complex Expressions

- Combine operators with parentheses and precedence rules.
- Three notations, just as in arithmetic:
  1. Sequences of assignment statements.
  2. Expressions with several operators.
  3. Expression trees.

Sequences of Assignments

- Create temporary relation names.
- Renaming can be implied by giving relations a list of attributes.
- Example: R3 := R1 $\bowtie_C$ R2 can be written:

  R4 := R1 X R2

  R3 := σ_C(R4)

Expressions in a Single Assignment

- Example:

  R3 := π (R1 $\bowtie_C$ R2)

- Precedence of relational operators:
  1. [σ, π, ρ] (highest).
  2. [$\times$, $\bowtie$].
  3. $\cap$.
  4. [$\cup$, —]

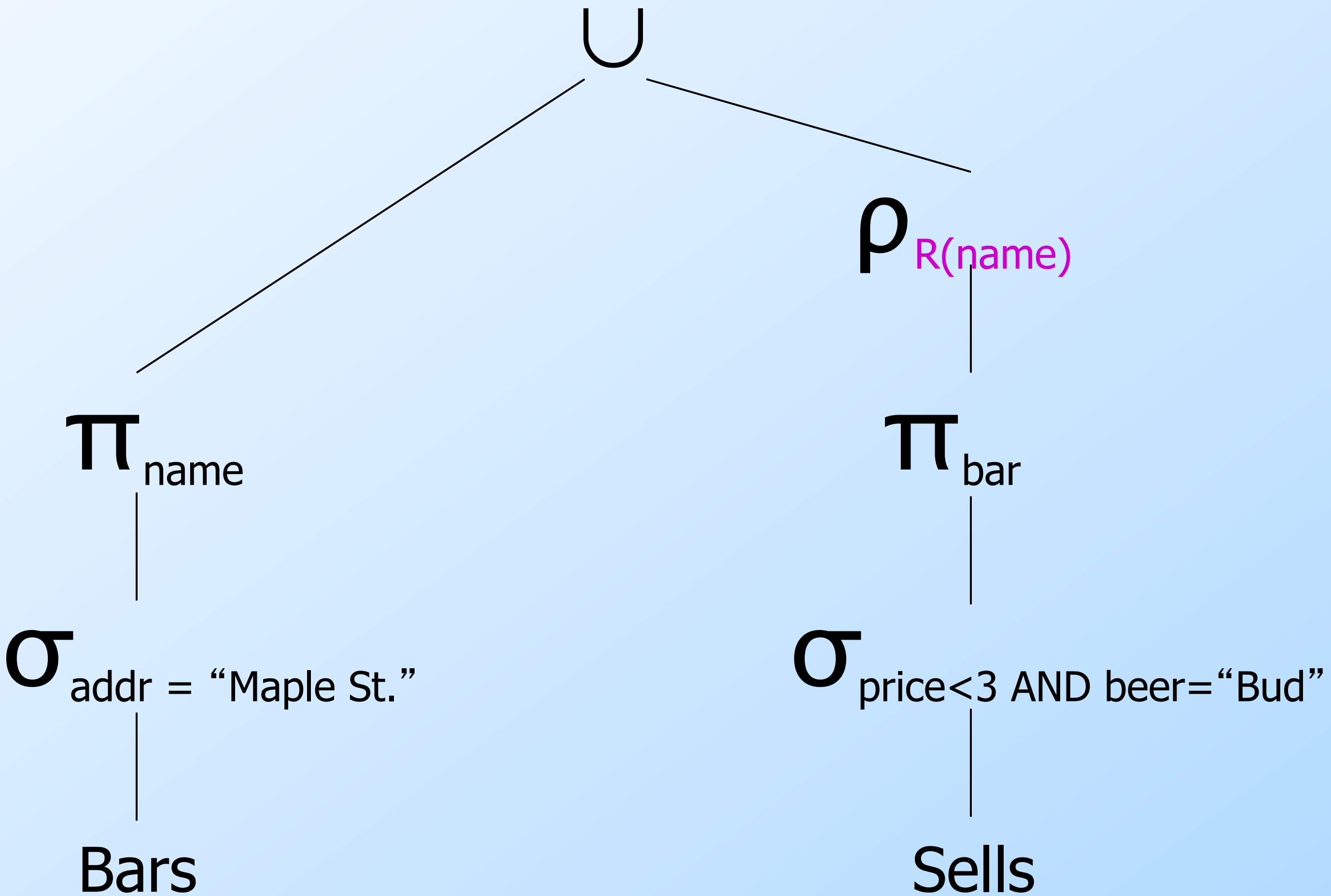
Expression Trees

- Leaves are operands --- either variables standing for relations or particular, constant relations.
- Interior nodes are operators, applied to their child or children.

Example: Tree for a Query

- Using the relations Bars(name, addr) and Sells(bar, beer, price), find the names of all the bars that are either on Maple St. or sell Bud for less than $3.

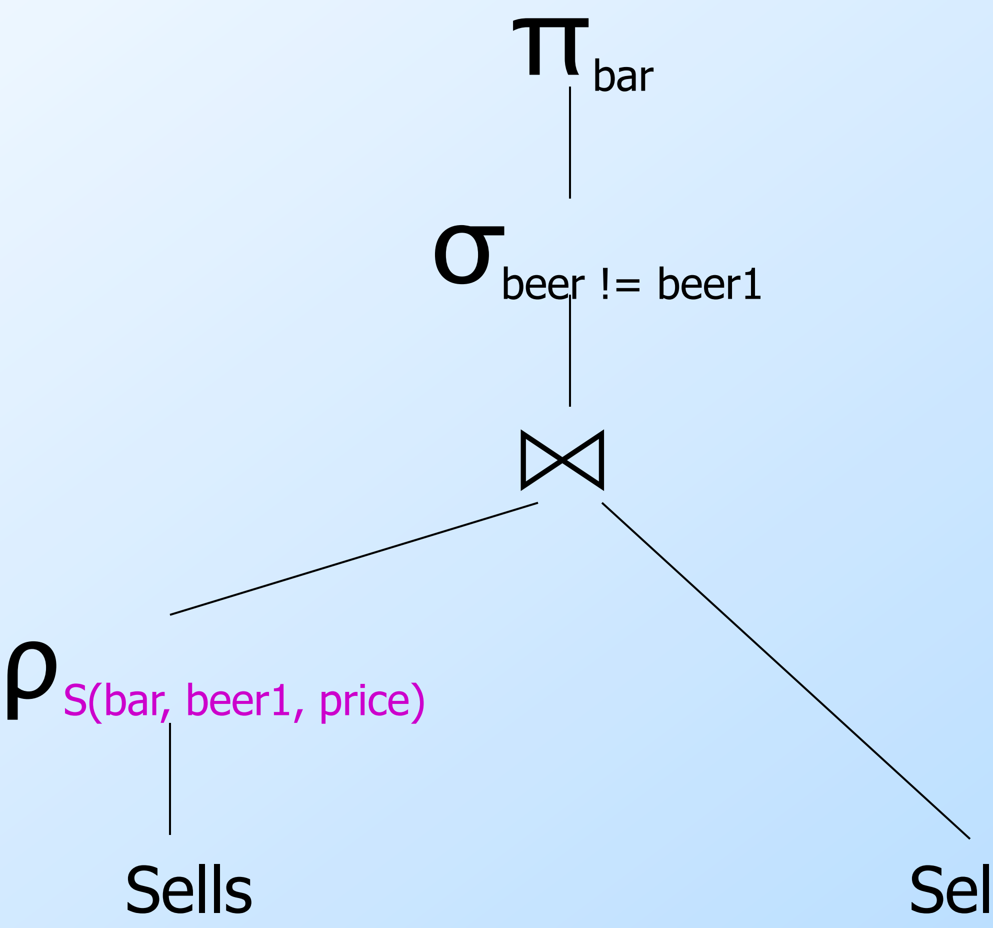
As a Tree:


∪
ρ R(name)
π name
π bar
σ addr = “Maple St.”
σ price<3 AND beer=“Bud”
Bars
Sells

Example: Self-Join

- Using Sells(bar, beer, price), find the bars that sell two different beers at the same price.
- Strategy: by renaming, define a copy of Sells, called S(bar, beer1, price). The natural join of Sells and S consists of quadruples (bar, beer, beer1, price) such that the bar sells both beers at this price.

The Tree



Schemas for Results

- Union, intersection, and difference: the schemas of the two operands must be the same, so use that schema for the result.
- Selection: schema of the result is the same as the schema of the operand.
- Projection: list of attributes tells us the schema.

Schemas for Results --- (2)

- Product: schema is the attributes of both relations.
  - Use R.*A*, etc., to distinguish two attributes named *A*.
- Theta-join: same as product.
- Natural join: union of the attributes of the two relations.
- Renaming: the operator tells the schema.

One more operator:Division

$R3 := R1 \div R2$

$R1(a_1, a_{2\ldots}\ a_{k,}\ a_{k+1\ldots}\ a_m)$ $R2(a_1, a_{2\ldots}\ a_k)$

$R3(a_{k+1\ldots}\ a_m)$

- All attributes in R2 (denominator) must also be present in R1 (numerator)
- R1 has some extra attributes
- The schema of the result relation R3 contains the attributes defined in R1 and not defined in R2

Division - following

- Tuples in R3 are those in $\pi_{ak+1 \ldots am}$ (R1) which concatenated to each tuple in R2 leads to a tuple in R1.

Example…

- Likes(<u>drinker</u>, <u>beer</u>)
- BeerNames(beer)

Products and Joins

- Product: R1 X R2
  = {(t1, t2) : t1 in R1 and t2 in R2}
- Theta Join: R1 $\bowtie_{C}$ R2 = σ_{C}(R1 X R2)
- Natural Join: R1 $\bowtie$ R2

  = $\pi_{\mathit{schema(R1)\ SETUNION\ schema(R2)}}$

  (R1 $\bowtie_{\mathit{R1.A=R2.A\ and\ R1.B=R2.B\ and\ldots}}$ R2)

Example: Product

R1(

A,	B)
1	2
3	4

R2(

B,	C)
5	6
4	8
2	10

R3 := R1 X R2

R3(

A,	R1.B,	R2.B,	C)
1	2	5	6
1	2	4	8
1	2	2	10
3	4	5	6
3	4	4	8
3	4	2	10

Example: Theta Join

R1(

A,	B)
1	2
3	4

R2(

B,	C)
5	6
4	8
2	10

$$R3 := R1 \bowtie_{R1.B=R2.B} R2$$

R3(

A,	R1.B,	R2.B,	C)
~~1~~	~~2~~	~~5~~	~~6~~
~~1~~	~~2~~	~~4~~	~~8~~
1	2	2	10
~~3~~	~~4~~	~~5~~	~~6~~
3	4	4	8
~~3~~	~~4~~	~~2~~	~~10~~

Example: Natural Join

R1(

A,	B)
1	2
3	4

R2(

B,	C)
5	6
4	8
2	10

$R3 := R1 \bowtie R2$

R3(

A,	~~R1~~.B,	~~R2.B,~~	C)
~~1~~	~~2~~	~~5~~	~~6~~
~~1~~	~~2~~	~~4~~	~~8~~
1	2	~~2~~	10
~~3~~	~~4~~	~~5~~	~~6~~
3	4	~~4~~	8
~~3~~	~~4~~	~~2~~	~~10~~

Sample Problem #1

Drinkers(<u>name</u>, addr, phone)
Likes(<u>drinker</u>, <u>beer</u>)

Find names and addresses of all drinkers who like Bud.

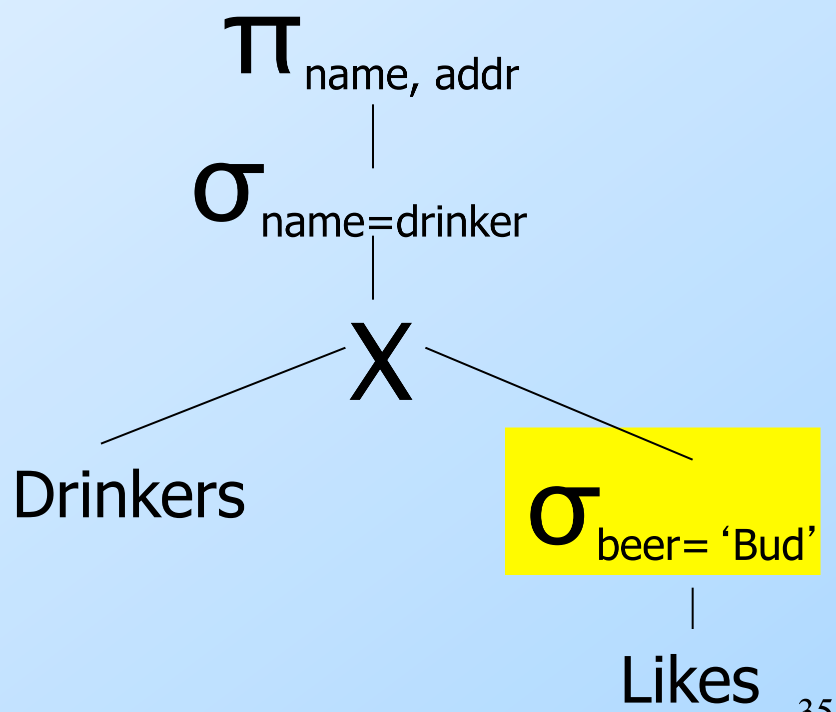
Sample Problem #1

Drinkers(name, addr, phone)

Likes(drinker, beer)

Find names and addresses of all drinkers who like Bud.

- Method 1: filter, then concatenate



Sample Problem #1

Drinkers(name, addr, phone)

Likes(drinker, beer)

Find names and addresses of all drinkers who like Bud.

- Method 2: concatenate, then filter

$\pi_{\text{name, addr}}$

$\sigma_{\text{name=drinker AND beer= 'Bud'}}$

X

Drinkers Likes

Sample Problem #2

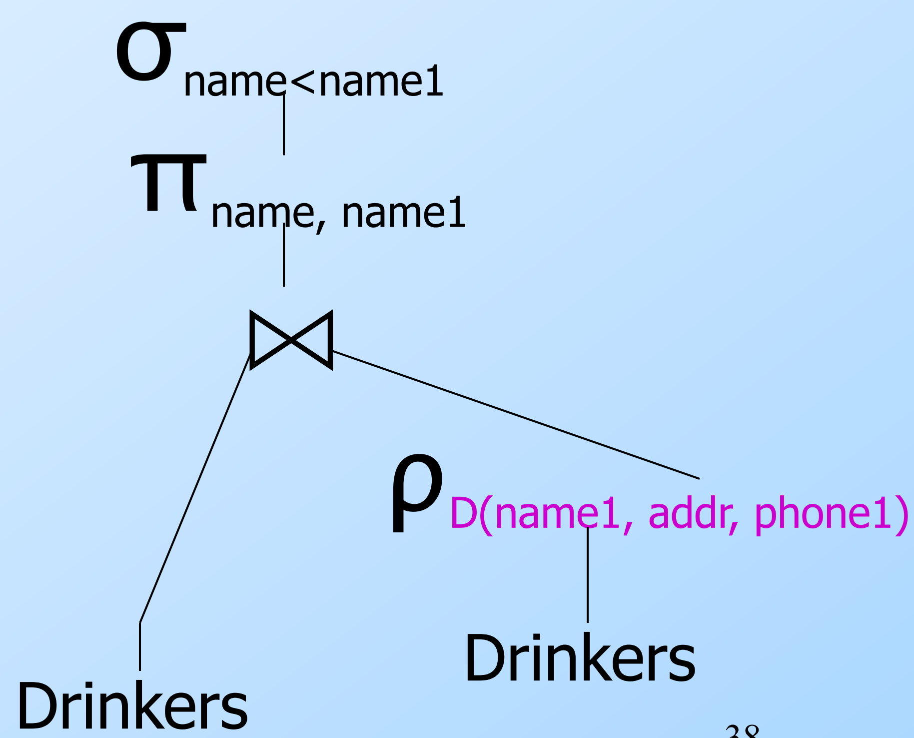
Drinkers(name, addr, phone)

Find names of all pairs of drinkers who live at the same address.

Sample Problem #2

Drinkers(<u>name</u>, addr, phone)

Find names of all pairs of drinkers who live at the same address.



- Comparing drinkers with other drinkers, so reuse and rename
- Natural join contains tuples (name, name1, addr, phone, phone1) such that both drinkers live at this address
- Select condition ensures no duplicates