Outline

- Overview of modeling
- Relational Model (Chapter 2)
  - Basics
  - Keys
  - Relational operations
  - Relational algebra basics
- SQL
  - Basic Data Definition (3.2)
  - Setting up the PostgreSQL database
  - Basic Queries (3.3-3.5)
  - Null values (3.6)
  - Aggregates (3.7)

Relational Query Languages

- Example schema: ***R(A, B)***
- Practical languages
  - SQL
    - select A from R where B = 5;
  - Datalog (sort of practical)
    - q(A) :- R(A, 5)
- Formal languages
  - Relational algebra

    $\pi_A (\sigma_{B=5} (R))$
  - Tuple relational calculus

    $\{ t : \{A\} \mid \exists s : \{A, B\} (R(A, B) \wedge s.B = 5) \}$
  - Domain relational calculus
    - Similar to tuple relational calculus

Modeling Languages

- Some of languages are "procedural" and provide a set of operations
  - Each operation takes one or two relations as input, and produces a single relation as output
  - Examples: Relational Algebra

- The "non-procedural" (also called "declarative") languages specify the output, but don't specify the operations
  - SQL, Relational calculus
  - Datalog (used as an intermediate layer in quite a few systems today)

Select Operation

Choose a subset of the tuples that satisfies some predicate
Denoted by σ in relational algebra

Relation r

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

$\sigma_{A=B \wedge D > 5}(r)$

A	B	C	D
α	α	1	7
β	β	23	10

Project

Choose a subset of the columns (for all rows)
Denoted by ∏ in relational algebra

Relation r

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

$\prod_{A,D}(r)$

A	D
α	7
α	7
β	3
β	10

A	D
α	7
β	3
β	10

Relational algebra following "set" semantics – so no duplicates
SQL allows for duplicates – we will cover the formal semantics later

Set Union, Difference

Relation r, s

r

A	B
α	1
α	2
β	1

s

A	B
α	2
β	3

$r \cup s$:

A	B
α	1
α	2
β	1
β	3

$r - s$:

A	B
α	1
β	1

Must be compatible schemas

What about intersection ?

Can be derived

$r \cap s = r - (r - s);$

$r \cap s$:

A	B
α	2

Cartesian Product

Combine tuples from two relations

If one relation contains N tuples and the other contains M tuples, the result would contain N*M tuples

The result is rarely useful – almost always you want pairs of tuples that satisfy some condition

Relation r, s

r

A	B
α	1
β	2

s

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

$r \times s$:

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

Joins

Combine tuples from two relations if the pair of tuples satisfies some constraint
(equivalent to Cartesian Product followed by a Select)

Relation r, s

r

A	B
α	1
β	2

s

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

$r \bowtie_{A=C} s$:

A	B	C	D	E
α	1	α	10	a
~~α~~	~~1~~	~~β~~	~~10~~	~~a~~
~~α~~	~~1~~	~~β~~	~~20~~	~~b~~
~~α~~	~~1~~	~~γ~~	~~10~~	~~b~~
~~β~~	~~2~~	~~α~~	~~10~~	~~a~~
β	2	β	10	a
β	2	β	20	b
~~β~~	~~2~~	~~γ~~	~~10~~	~~b~~

Natural Join

Combine tuples from two relations if the pair of tuples agree on the common columns (with the same name)

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

Figure 2.5 The *department* relation.

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

Figure 2.4 Unsorted display of the *instructor* relation.

department ⋈ instructor:

ID	*name*	*salary*	*dept_name*	*building*	*budget*
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
12121	Wu	90000	Finance	Painter	120000
15151	Mozart	40000	Music	Packard	80000
22222	Einstein	95000	Physics	Watson	70000
32343	El Said	60000	History	Painter	50000
33456	Gold	87000	Physics	Watson	70000
45565	Katz	75000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
76543	Singh	80000	Finance	Painter	120000
76766	Crick	72000	Biology	Watson	90000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000

Figure 2.12 Result of natural join of the *instructor* and *department* relations.

Rename Operation

- Allows us to name, and therefore to refer to, the results of relational-algebra expressions.
- Allows us to refer to a relation by more than one name.

  Example:

$$\rho_{x}(E)$$

  returns the expression *E* under the name *X*

  If a relational-algebra expression *E* has arity *n*, then

$$\rho_{x\,(A1,\,A2,\,\ldots,\,An)}(E)$$

  returns the result of expression *E* under the name *X*, and with the attributes renamed to $A_1, A2, \ldots, An$.

Relational Algebra

- Those are the basic operations
- What about SQL Joins ?
  - Compose multiple operators together

    $\sigma_{A=C}(r \times s)$
- Additional Operations
  - Set intersection
  - Natural join
  - Division
  - Assignment

Additional Operators

- Set intersection ($\cap$)
  - $r \cap s = r - (r - s)$;
  - SQL Equivalent: intersect
- Assignment ($\leftarrow$)
  - A convenient way to right complex RA expressions
  - Essentially for creating "temporary" relations
    - $temp1 \leftarrow \prod_{R-S}(r)$
  - SQL Equivalent: "create table as…"

Additional Operators: Joins

- Natural join (⋈)
  - A Cartesian product with equality condition on common attributes
  - Example:
    - if *r* has schema ***R(A, B, C, D)***, and if ***s*** has schema ***S(E, B, D)***
    - Common attributes: ***B*** and ***D***
    - Then:

$$r \bowtie s = \prod_{r.A,\ r.B,\ r.C,\ r.D,\ s.E} (\sigma_{r.B = s.B \wedge r.D = s.D} (r \times s))$$

- SQL Equivalent:
  - select r.A, r.B, r.C, r.D, s.E from r, s where r.B = s.B and r.D = s.D, OR
  - select * from r natural join s

Additional Operators: Joins

- ***Equi-join***
  - A join that only has equality predicates
- ***Theta-join*** ($\bowtie_\theta$)
  - $r \bowtie_\theta s = \sigma_\theta(r \times s)$ (arbitrary predicates)
- ***Left outer join*** (⟕)
  - If have ***r(A, B), s(B, C)***, then:
    - r ⟕ s = (r ⋈ s) ∪ (*"all non-matching rows in r w/ nulls for s's attributes"*)
  - What is $(r - \pi_{r.A,\ r.B}(r \bowtie s))$? (rows of *r* that match rows of *s*)
  - $r ⟕ s = (r \bowtie s) \cup \rho_{temp\ (A,\ B,\ C)}((r - \pi_{r.A,\ r.B}(r \bowtie s)) \times \{(NULL)\})$

Additional Operators: Join Variations

- *Tables: r(A, B), s(B, C)*

name	Symbol	SQL Equivalent	RA expression
cross product	$\times$	select * from r, s;	$r \times s$
natural join	$\bowtie$	natural join	$\pi_{r.A,\ r.B,\ s.C}\sigma_{r.B\ =\ s.B}(r\ x\ s)$
equi-join	$\bowtie_\theta$ *(theta must be equality)*		
theta join	$\bowtie_\theta$	from .. where θ;	$\sigma_\theta(r\ x\ s)$
left outer join	$r ⟕ s$	left outer join (with "on")	(see previous slide)
full outer join	$r ⟗ s$	full outer join (with "on")	-
(left) semijoin	$r \ltimes s$	none	$\pi_{r.A,\ r.B}(r \bowtie s)$
(left) antijoin	$r \lhd s$	none	$r - \pi_{r.A,\ r.B}(r \bowtie s)$

Additional Operators: Division

- Assume ***r(R), s(S)***, for queries where ***S*** $\subseteq$ ***R:***
  - $r \div s$
- Think of it as "opposite of Cartesian product"
  - $r \div s = t$ *iff* $t \times s \subseteq r$

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
B	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

÷

A	B
α	1
β	2

=

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

Relational Algebra Examples

Find all loans of over $1200:

$\sigma_{\text{amount} > 1200}$ (loan)

Find the loan number for each loan of an amount greater than $1200:

$\Pi_{\text{loan-number}}$ ($\sigma_{\text{amount} > 1200}$ (loan))

Find names of all customers who have a loan, account, or both, from the bank:

$\Pi_{\text{customer-name}}$ (borrower) $\cup$ $\Pi_{\text{customer-name}}$ (depositor)

Relational Algebra Examples

Find names of customers who have a loan and an account at bank:

$\Pi_{\text{customer-name}}$ (borrower) $\cap$ $\Pi_{\text{customer-name}}$ (depositor)

Find names of customers who have a loan at the UMD branch:

$\Pi_{\text{customer-name}}$ ($\sigma_{\text{branch-name=``UMD''}}$ ($\sigma_{\text{borrower.loan-number = loan.loan-number}}$(borrower x loan)))

Relational Algebra Examples

Find *largest* account balance, assume relation is {(1), (2), (3)}:

Rename the account relation to d:

$$\prod_{balance}(account) - \prod_{account.balance} (\sigma_{account.balance < d.balance} (account \times \rho_d (account)))$$