

RELATIONAL ALGEBRA

CS 564- Fall 2015

RELATIONAL QUERY LANGUAGES

- manipulation and retrieval of data from a database
- 2 types:
 - **Declarative**: Tuple Relational Calculus (**TRC**)
Domain Relational Calculus (**DRC**)
 - describe what a user wants, rather than how to compute it
 - **Procedural**: Relational Algebra (**RA**)
 - operational, useful for representing execution plans

QUERY VS PROGRAMMING LANGUAGES

Query Languages:

- are not “Turing complete”
- are not intended to be used for complex calculations
- support easy, efficient access to large data sets

WHAT IS RELATIONAL ALGEBRA?

- **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
- **relational algebra**: an algebra whose operands are relations or variables that represent relations
 - operators do the most common things that we need to do with relations in a database
 - can be used as a **query language** for relations

RELATIONAL ALGEBRA PRELIM

- **Query:**
 - Input: relational instances
 - Output: relational instances
 - specified using the schemas
 - may produce different results for different instances
 - the schema of the result is fixed
- **Positional vs named-field notation:**
 - C.name or
 - 2

RELATIONAL ALGEBRA PRELIM

- Basic operations:
 - *Selection* $\{\sigma\}$: selects a subset of rows
 - *Projection* $\{\pi\}$: deletes columns
 - *Cross-product* $\{\times\}$: combines two relations
 - *Set-difference* $\{-\}$, *Union* $\{U\}$
- When the relations have named fields:
 - *Renaming* $\{\rho\}$
- Additional operations:
 - *Intersection, join, division*

BASIC OPERATIONS

SELECTION

Notation: $\sigma_C(R)$

- C is a condition that refers to the attributes of R
- outputs the **rows** of R that satisfy C
- output schema: same as input schema

Example

- $\sigma_{age>24}(Person)$
- $\sigma_{age>24 \text{ and } age\leq 28}(Person)$
- $\sigma_{age>24 \text{ and } name="Paris"}(Person)$

SELECTION EXAMPLE

Person

SSN	name	age	phoneNumber
934729837	Paris	24	608-374-8422
934729837	Paris	24	603-534-8399
123123645	John	30	608-321-1163
384475687	Arun	25	206-473-8221

$\sigma_{age>24}(Person)$

SSN	name	age	phoneNumber
123123645	John	30	608-321-1163
384475687	Arun	25	206-473-8221

PROJECTION

Notation: $\pi_{A_1, A_2, \dots, A_n}(R)$

- outputs only the **columns** A_1, A_2, \dots, A_n
- removes any duplicate tuples (why?)
- output schema: $R(A_1, A_2, \dots, A_n)$

Example

- $\pi_{SSN, age}(Person)$
- $\pi_{SSN, phoneNumber, age}(Person)$

PROJECTION EXAMPLE

Person

SSN	name	age	phoneNumber
934729837	Paris	24	608-374-8422
934729837	Paris	24	603-534-8399
123123645	John	30	608-321-1163
384475687	Arun	20	206-473-8221

$\pi_{SSN, name}(Person)$

SSN	name
934729837	Paris
123123645	John
384475687	Arun

UNION

Notation: $R_1 \cup R_2$

- outputs all tuples in R_1 **or** R_2
- both relations must have the same schema!
- output schema: same as input

A	B
a ₁	b ₁
a ₂	b ₁
a ₂	b ₂

U

A	B
a ₁	b ₁
a ₃	b ₁
a ₄	b ₄

=

A	B
a ₁	b ₁
a ₂	b ₁
a ₂	b ₂
a ₃	b ₁
a ₄	b ₄

DIFFERENCE

Notation: $R_1 - R_2$

- outputs all tuples in R_1 **and not** in R_2
- both relations must have the same schema!
- output schema: same as input

A	B
a ₁	b ₁
a ₂	b ₁
a ₂	b ₂

—

A	B
a ₁	b ₁
a ₃	b ₁
a ₄	b ₄

=

A	B
a ₂	b ₁
a ₂	b ₂

CROSS-PRODUCT

Notation: $R_1 \times R_2$

- matches each tuples in R_1 with each tuple in R_2
- input schema: $R_1(A_1, A_2, \dots, A_n), R_2(B_1, B_2, \dots, B_m)$
- output schema: $R(A_1, \dots, A_n, B_1, \dots, B_m)$

CROSS-PRODUCT EXAMPLE

Person

SSN	name
934729837	Paris
123123645	John

Dependent

depSSN	depname
934729837	Helen
934729837	Bob



Person \times *Dependent*

SSN	name	depSSN	depname
934729837	Paris	934729837	Helen
123123645	John	934729837	Bob
934729837	Paris	934729837	Helen
123123645	John	934729837	Bob

RENAMING

Notation: $\rho_{A_1, A_2, \dots, A_n}(R)$

- does not change the instance, only the schema!
- input schema: $R(B_1, B_2, \dots, B_n)$
- output schema: $R(A_1, \dots, A_n)$

Why is it necessary?

RENAMING EXAMPLE

Person

SSN	name
934729837	Paris
123123645	John

Dependent

SSN	name
934729837	Helen
934729837	Bob

↓ $Person \times \rho_{depSSN, depname} (Dependent)$

SSN	name	depSSN	depname
934729837	Paris	934729837	Helen
123123645	John	934729837	Bob
934729837	Paris	934729837	Helen
123123645	John	934729837	Bob

DERIVED OPERATIONS

INTERSECTION

Notation: $R_1 \cap R_2$

- outputs all tuples in R_1 **and** R_2
- output schema: same as input
- can be expressed as: $R_1 - (R_1 - R_2)$

A	B
a ₁	b ₁
a ₂	b ₁
a ₂	b ₂

 \cap

A	B
a ₁	b ₁
a ₃	b ₁
a ₄	b ₄

 $=$

A	B
a ₁	b ₁

JOIN (THETA JOIN)

Notation: $R_1 \bowtie_C R_2 = \sigma_C(R_1 \times R_2)$

- cross-product followed by a selection
- C can be any boolean-valued condition
- might have less tuples than the cross-product!

THETA JOIN EXAMPLE

Person

SSN	name	age
934729837	Paris	26
123123645	John	22

Dependent

dSSN	dname	dage
934729837	Helen	23
934729837	Bob	28



$Person \bowtie_{Person.age > Dependent.dage} Dependent$

SSN	name	age	dSSN	dname	dage
934729837	Paris	26	934729837	Helen	23

EQUI-JOIN

Notation: $R_1 \bowtie_C R_2$

- special case of join where the condition C contains only equalities!
- output schema: similar to cross-product, but only one copy for the fields in the equality

Example for $R(A, B), S(C, D)$

- $R \bowtie_{B=C} S$
- output schema: $T(A, B, D)$

NATURAL JOIN

Notation: $R_1 \bowtie R_2$

- equi-join on all the common fields

Person

SSN	name	age
934729837	Paris	26
123123645	John	22

Dependent

SSN	dname
934729837	Helen
934729837	Bob



Person \bowtie Dependent

SSN	name	age	dname
934729837	Paris	26	Helen
934729837	Paris	26	Bob

NATURAL JOIN

Natural Join $R \bowtie S$

- Input schema: $R(A, B, C, D), S(A, C, E)$
 - Output schema?
- Input schema: $R(A, B, C), S(D, E)$
 - Output schema?
- Input schema: $R(A, B, C), S(A, B, C)$
 - Output schema?

SEMI-JOIN

Notation: $R_1 \bowtie R_2$

- natural join followed by projection on the attributes of R_1

Example:

- $R(A, B, C), S(B, D)$
- $R \bowtie S = \pi_{A,B,C}(R \bowtie S)$
- output schema?

DIVISION

Notation: R_1 / R_2

- suppose $R_1(A, B)$ and $R_2(B)$
- the output contains all A-tuples such that for every B-tuple in R_2 , there exists an (A,B) tuple in R_1
- output schema: $R(A)$

DIVISION EXAMPLE

A

A	B
a ₁	b ₁
a ₁	b ₂
a ₁	b ₃
a ₂	b ₁

B₁

B
b ₂
b ₃
b ₁

B₂

B
b ₁

A / B₁

A
a ₁

A / B₂

A
a ₁
a ₂

COMBINING RA OPERATORS

- We can build more complex queries by combining RA operators together

e.g. standard algebra: $(x + 1) * y - z^2$

- There are 3 different notations:
 - sequence of assignment statements
 - expressions with operators
 - expression trees

COMBINING RA OPERATORS

Input schema: $R(B, C), S(A, B)$

- expressions with operators

$$\pi_A(\sigma_{C=1}(R) \bowtie S)$$

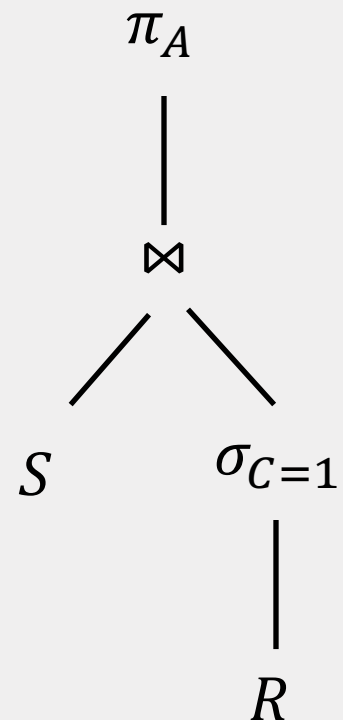
- sequence of assignment statements

$$R' = \sigma_{C=1}(R)$$

$$R'' = R' \bowtie S$$

$$R''' = \pi_A(R'')$$

- expression trees



EXPRESSIVE POWER

- RA cannot express **transitive closure**!

Edges

From	To
a	b
b	c
a	d
c	d

Transitive closure computes all pairs of nodes connected by a directed path

EXAMPLES

Sailors (sid, sname, rating, age)

Reserves (sid, bid, day)

Boats (bid, bname, color)

Q1: What are the names of the sailors who have reserved boat #100?

Q2: What are the names of the sailors who have reserved a red boat?

EXAMPLES

Sailors (sid, sname, rating, age)

Reserves (sid, bid, day)

Boats (bid, bname, color)

Q3: Find the names of the sailors who have reserved in a green **or** red boat

Q4: Find the names of the sailors who have reserved in a green **and** a red boat

EXAMPLES

Sailors (sid, sname, rating, age)

Reserves (sid, bid, day)

Boats (bid, bname, color)

Q5: Find the names of the sailors who have reserved all '470' boats

MORE EXAMPLES

Product (pid, name, price, category, maker-cid)

Purchase (buyer-ssn, seller-ssn, store, pid)

Company (cid, name, country)

Person (ssn, name, phone, city)

Q6: Find the phone numbers of people who bought iPads from Fred (the salesman)

Q7: Find the names of people who bought products from the USA

MORE EXAMPLES

Product (pid, name, price, category, maker-cid)

Purchase (buyer-ssn, seller-ssn, store, pid)

Company (cid, name, country)

Person (ssn, name, phone, city)

Q8: Find the names of people who bought products from the USA, but not from Greece

Q9: Find the names of people who bought products from the USA, and live in Madison

RECAP

- Relational Algebra
 - query language for relations
- Basic Operations
 - selection, projection
 - difference, union
 - cross-product, renaming
- Derived Operations
 - join, natural join, equi-join, division, etc