THE GEORGE WASHINGTON UNIVERSITY

WASHINGTON, DC

3. Relational Algebra

CSCI 2541 Database Systems & Team Projects

Wood & Chaufournier

Slides adapted from Prof. Bhagi Narahari; Silberschatz, Korth, and Sudarshan; and Ramakrishnan, Gerhke, and Lawrence

Last time…

Relational Model Definitions

Constraints and Relationships

Relational Algebra

this time…

Relational Algebra

A "formal query language"

- Theoretical foundation for SQL

Data is stored as a set of relations

- Relations implemented as tables
- Tuple in a relation is a row in the table
- Attribute (from domain) in relation is column in table

RA = A set of mathematical operators that compose, modify, and combine tuples within different relations

Relations are sets!

Why do we need RA?

Relational Algebra != SQL, which is the query language developers use…

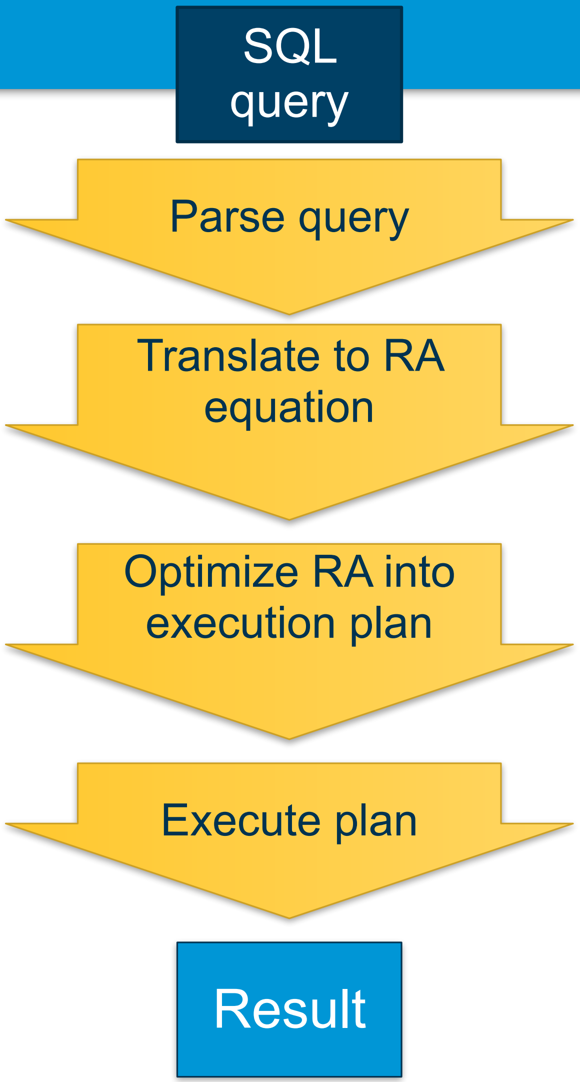
- SQL is designed for ease of use by programmers
- RA is for ease of use by the DBMS

SQL queries will be converted into RA for execution

- Understanding RA can help you write better queries
- Critical to understand if you want to build DBMS or optimize its execution

RA and SQL

Query execution in Relational DBMS



Relational Algebra is…

A **procedural language** consisting of a set of **operations** that **take one or two relations as input** and **produce a new relation** as their result.

Basic operators

- project: $\prod$
- select: σ
- union: $\cup$
- set difference: –
- Cartesian product: x
- Join: $\bowtie$

Equations operating on Tables

Tables in…
Tables out!

Since each operation returns a relation, operations can be composed!

Relational Algebra

Filtering Operators

$\prod$ σ

Joining Operators

x $\bowtie$

More Operators

$\cup$ - ρ $\leftarrow$

Project Operation

A unary operation that returns its argument relation, with certain attributes left out.

Notation:

$$\prod_{A_1, A_2, A_3 \ldots . A_k} (r)$$

where A_1, A_2, …, A_k are attribute names and r is a relation name.

The result is defined as the relation of k columns obtained by erasing the columns that are not listed

Duplicate rows removed from result, since relations are sets

Projection

$$\Pi_{ID,office,name}(\ instructor\ )$$

instructor Relation

ID	name	department	office
E1	Sam	EE	SEH 111
E2	Sam	CS	SEH 231
E3	Lily	ME	SEH 321
E4	Lily	CE	SEH 451
E5	Nick	BIO	SEH 341
E6	Sam	ECE	TOMP 231
E7	Sarah	LIT	Gelman 213
E8	Sarah	CS	SEH 125

ID	office	name
E1	SEH 111	Sam
E2	SEH 231	Sam
E3	SEH 321	Lily
E4	SEH 451	Lily
E5	SEH 341	Nick
E6	TOMP 231	Sam
E7	Gelman 213	Sarah
E8	SEH 125	Sarah

Projection

How many tuples will be projected?

$$T = \prod_{name} (\mathit{instructor})$$

instructor Relation

ID	name	department	office
E1	Sam	EE	SEH 111
E2	Sam	CS	SEH 231
E3	Lily	ME	SEH 321
E4	Lily	CE	SEH 451
E5	Nick	BIO	SEH 341
E6	Sam	ECE	TOMP 231
E7	Sarah	LIT	Gelman 213
E8	Sarah	CS	SEH 125

T

name
???

How many tuples in T?

a) 0
b) 4
c) 5
d) 8

Projection

How many tuples will be projected?

$$T = \prod_{name} (\ instructor\ )$$

instructor Relation

ID	name	department	office
E1	Sam	EE	SEH 111
E2	Sam	CS	SEH 231
E3	Lily	ME	SEH 321
E4	Lily	CE	SEH 451
E5	Nick	BIO	SEH 341
E6	Sam	ECE	TOMP 231
E7	Sarah	LIT	Gelman 213
E8	Sarah	CS	SEH 125

How many tuples in T?

a) 0
b) 4
c) 5
d) 8

T

name
Sam
Nick
Sarah
Lily

A relation is a **set**! No duplicates! Unordered!

(may not be true in practice with a SQL DBMS)

Select Operator

Fetches tuples that satisfy a given predicate.

Notation: $\sigma_{p}(r)$

p is called the selection predicate

- Compare against other attributes or constants
- =, ≠ , >, <, >=, <=,
- Combine predicates: ∧ (**and**), ∨ (**or**), ¬ (**not**)

Example: *select tuples in the instructor relation where the instructor is in the "CS" department*

$$\sigma_{\text{department} = \text{"CS"}}(\text{instructor})$$

Selection

$$\sigma_{\text{department} = \text{“CS”}}(\text{instructor})$$

instructor Relation

ID	name	department	office
E1	Sam	EE	SEH 111
E2	Sam	CS	SEH 231
E3	Lily	ME	SEH 321
E4	Lily	CE	SEH 451
E5	Nick	BIO	SEH 341
E6	Sam	ECE	TOMP 231
E7	Sarah	LIT	Gelman 213
E8	Sarah	CS	SEH 125

ID	name	department	office
E2	Sam	CS	SEH 231
E8	Sarah	CS	SEH 125

Selection Example

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

$$\sigma_{title\,=\,'EE'}(\mathrm{Emp})$$

Selection Example

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

$$\sigma_{title\ =\ 'EE'}(\text{Emp})$$

eno	ename	title	salary
E1	J. Doe	EE	30000
E6	L. Chu	EE	30000

$$\sigma_{salary\ >\ 35000\ \vee\ title\ =\ 'PR'}(\text{Emp})$$

Logic operators: $\wedge$ AND, $\vee$ OR, $\neg$ NOT

Selection Example

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

$$\sigma_{title\ =\ 'EE'}(\text{Emp})$$

eno	ename	title	salary
E1	J. Doe	EE	30000
E6	L. Chu	EE	30000

$$\sigma_{salary\ >\ 35000\ \vee\ title\ =\ 'PR'}(\text{Emp})$$

eno	ename	title	salary
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

Logic operators: $\wedge$ AND, $\vee$ OR, $\neg$ NOT

Question: How many rows are returned by this query

$$T = \sigma_{\texttt{salary >= 30000} \overset{\text{And}}{\wedge} \underbrace{\texttt{(title='SA} \vee \texttt{title='PR')}}} \textbf{(Emp)}$$

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

How many tuples in T?

a) 0
b) 3
c) 4
d) other

Logic operators: ∧ AND, ∨ OR, ¬ NOT

Question: How many rows are returned by this query

$T = \sigma_{salary >= 30000 \wedge (title='SA' \vee title='PR')}(Emp)$

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

How many tuples in T?

a) 0
b) 3
c) 4
d) other

Logic operators: $\wedge$ AND, $\vee$ OR, $\neg$ NOT

Combining Operators

$$\Pi_{name} (\sigma_{department = \text{"CS"} \vee department = \text{"EE"}} (instructor))$$

We can do both!

Use parenthesis to clarify order of operations

instructor Relation

ID	name	department	office
E1	Sam	EE	SEH 111
E2	Sam	CS	SEH 231
E3	Lily	ME	SEH 321
E4	Lily	CE	SEH 451
E5	Nick	BIO	SEH 341
E6	Susan	EE	TOMP 231
E7	Sarah	LIT	Gelman 213
E8	Sarah	CS	SEH 125

name
Sam
Sarah
Susan

Logic operators:
$\wedge$ AND, $\vee$ OR, $\neg$ NOT

Relational Algebra

Basic Operators

$\prod$ σ

Joining Operators

x $\bowtie$

More Operators

$\cup$ - ρ $\leftarrow$

Operators that combine relations

How to connect two relations ?

- To find name of students taking a specific course with cid, we need to look at both Student and Takes (registration) tables

We need operators that produce a relation (set of tuples) after "**joining**" two different relations

Set theory provides us with the **cartesian product** operator (between two sets; but can be applied to product of any number of sets – to get a k-tuple)

Cartesian Product

R x S

- Concatenates every tuple in R with every tuple in S

STUDENT

sid	name	attends
1	Jill	2
2	Matt	1

SCHOOL

id	school
1	UPenn
2	GWU

STUDENT x SCHOOL

sid	name	attends	id	school
1	Jill	2	1	UPenn
2	Matt	1	1	UPenn
1	Jill	2	2	GWU
2	Matt	1	2	GWU

Cartesian Product

The least useful of all joins…

R x S

- Concatenates every tuple in R with every tuple in S

STUDENT

sid	name	attends
1	Jill	2
2	Matt	1

SCHOOL

id	school
1	UPenn
2	GWU

STUDENT x SCHOOL

sid	name	attends	id	school
1	Jill	2	1	UPenn
2	Matt	1	1	UPenn
1	Jill	2	2	GWU
2	Matt	1	2	GWU

- Not so useful by itself, but it is the basis for much more powerful operations!

Making x more useful

What operators could we use to make a more useful query that returns the students and only the school they attend?

Student

sid	name	attends
1	Jill	2
2	Matt	1

School

id	school
1	UPenn
2	GWU

Student x School

sid	name	attends	id	school
1	Jill	2	1	UPenn
2	Matt	1	1	UPenn
1	Jill	2	2	GWU
2	Matt	1	2	GWU

We need a way to restrict to certain columns… Π

We need a way to only select some rows… σ ✓

Making x more useful

What operators could we use to make a more useful query that returns the students and only the school they attend?

Student

sid	name	attends
1	Jill	2
2	Matt	1

School

id	school
1	UPenn
2	GWU

Student x School

sid	name	attends	id	school
1	Jill	2	1	UPenn
2	Matt	1	1	UPenn
1	Jill	2	2	GWU
2	Matt	1	2	GWU

$$\Pi_{\text{student.name, school.school}}(\sigma_{\text{student.attends = school.id}}(\text{student x school}))$$

Join Operator

$\sigma_{\text{student.attends = school.id}}$ **(student x school)** is messy!

Join operators simplify this notation

$R_1 \bowtie_p R_2$

$\sigma_{\text{student.attends = school.id}}$ (student x school))

is equivalent to

student $\bowtie_{\text{student.attends = school.id}}$ school

Naming for Natural Joins

If we name attributes appropriately, we can use **Natural Joins**

- Automatically uses **all** attributes with same name as tests for equality

Student

sid	name	id
1	Jill	2
2	Matt	1

School

id	school
1	UPenn
2	GWU

Student x School

sid	name	student.i	school.id	school
1	Jill	2	1	UPenn
2	Matt	1	1	UPenn
1	Jill	2	2	GWU
2	Matt	1	2	GWU

$\text{student} \bowtie \text{school}$

sid	name	id	school
2	Matt	1	UPenn
1	Jill	2	GWU

Join Example

What is the meaning of this query?

$$\Pi_{(Takes.sid,\ Teaches.fid)}\ (Takes \bowtie_{Takes.cid\ !=\ Teaches.cid} Teaches)$$

Sid	Fid
1	2
1	8
3	1
3	2

cid:1 cid:2

student course

Reg info

Takes

sid	exp-grade	cid
1	A	550-0103
3	A	501-0103
3	A	700-1003
3	C	501-0103
4	C	501-0103

faculty course

Teaches

fid	cid
1	550-0103
2	700-1003
8	501-0103

Relational Algebra

Basic Operators

$\prod$ σ

Joining Operators

x $\bowtie$

More Operators

$\cup$ - ρ $\leftarrow$

Time???

Rename Operator

General definition allows renaming specific attributes

- $\rho_{X(C,D)}\ (R\ (A,B))$
  - Relation R renamed to X
  - Fields A,B in R are now renamed to C,D in X

$$\rho_{Person(idnum,\ who)}\ (Student\ (sid,\ name))$$

Find pairs of student IDs who have the same name: ?

$$\Pi_{\ Student.sid,\ Person.idnum}$$

$$(Student \bowtie_{Student.name=Person.who} (\ \rho_{Person(idnum,who)}\ (Student(sid,name))\ ))$$

Note: not necessary to rename the attributes …below will also work:

$$\Pi\ Student.sid,\ Person.sid$$

$$(Student \bowtie_{Student.name=Person.name} (\ \rho_{Person}\ (Student)\ )$$

Sid	Name
1	Lucas
2	Tim
7	Lucas

X

1
7

Assignment Operator

Storing query results lets you get a complex result from a sequence of simpler queries

- Use the **assignment operator** ← to indicate that the result of an operation is assigned to a temporary relation

$$\text{empdoe} \leftarrow \sigma_{ename='J.\ Doe'}(\text{Emp})$$
$$\text{overtime} \leftarrow \sigma_{dur>40}(\text{WorkWeek})$$
$$\text{empwo} \leftarrow \text{empdoe} \bowtie \text{overtime}$$
$$\text{result} \leftarrow \Pi_{eno,pno,dur}(\text{empwo})$$

Union Operator

If two relations have the same structure ("union-compatible"), we can apply normal set operations

Union: $\mathbf{R1} \cup \mathbf{R2}$

- Combine all rows in R1 and R2

STUDENT

id	name
1	Billy
2	Matt
3	Dan
4	Maury

FACULTY

id	name
1	Billy
12	Youssef
18	Choi

STUDENT ∪ FACULTY

id	name
1	Billy
2	Matt
3	Dan
4	Marty
12	Youssef
18	Choi

Difference Operator

If two relations have the same structure ("union-compatible"), we can apply normal set operations

Union: **R1 - R2**

- Remove any tuples from R1 that exist in R2

STUDENT

id	name
1	Billy
2	Matt
3	Dan
4	Maury

FACULTY

id	name
1	Billy
12	Youssef
18	Choi

STUDENT - FACULTY

id	name
2	Matt
3	Dan
4	Marty

Set Difference Example

What is the meaning of this query?

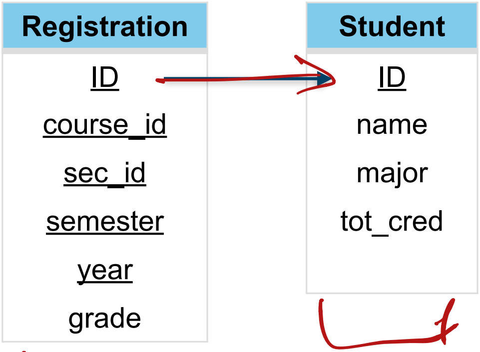
a) Students who are not registered for any courses

b) Students who are registered for all classes

c) Classes that don't have any registrations

d) Students with only one registration

$$\Pi_{ID}(\text{Student}) - \Pi_{ID}(\text{Registration})$$



Set Difference Example

What is the meaning of this query?

a) **Students who are not registered for any courses**

b) Students who are registered for all classes

c) Classes that don't have any registrations

d) Students with only one registration

$$\Pi_{ID}(\text{Student}) - \Pi_{ID}(\text{Registration})$$

Registration
ID
course_id
sec_id
semester
year
grade

Student
ID
name
major
tot_cred

Set Difference Example 2

What is the output of this query?

a) All tuples in Registration

b) All tuples in Student

c) Empty Set

d) Can't answer without knowing the data in the two tables

$$\Pi_{ID}(\text{Registration}) - \Pi_{ID}(\text{Student})$$

Registration
ID
course_id
sec_id
semester
year
grade

Student
ID
name
major
tot_cred

Set Difference Example 2

What is the output of this query?

a) All tuples in Registration

b) All tuples in Student

c) Empty Set

d) Can't answer without knowing the data in the two tables

$$\Pi_{ID}(\text{Registration}) - \Pi_{ID}(\text{Student})$$

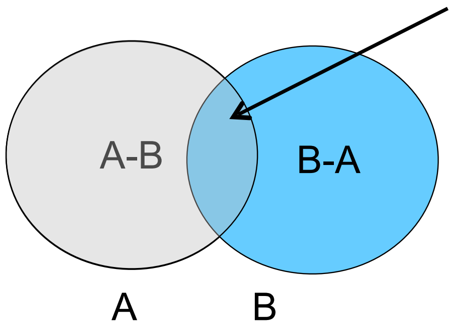
Registration
ID
course_id
sec_id
semester
year
grade

Student
ID
name
major
tot_cred

Set Intersection

How to find the "common" tuples between two relations?

Set intersection can be computed using Difference



$A \cap B$

$\equiv (A - (A\text{-}B)) = (B - (B\text{-}A))$

Tips

Filtering certain attributes π

Filtering certain tuples in one relation σ

Comparing tuples across two relations $\bowtie$

Comparing tuples within the same relation ρ

Combine/filter relations with the same structure $\cup\ \cap$

If query is getting long and messy, split up using assignment operator $\leftarrow$

RA and SQL

SELECT student.name, school.name FROM student, school WHERE student.attends = school.id

Parse query

Translate to RA equation

$$\Pi_{\text{student.name, school.name}} (\sigma_{\text{student.attends = school.id}} (\text{student x school}))$$

Optimize RA into execution plan

Execute plan

Result