Relational Algebra

Carla Teixeira Lopes

Bases de Dados
Mestrado Integrado em Engenharia Informática e Computação, FEUP

Based on Jennifer Widom slides

Agenda

Introduction to Relational Algebra

Operators

Alternate notations

Extensions to Relational Algebra

What is 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?

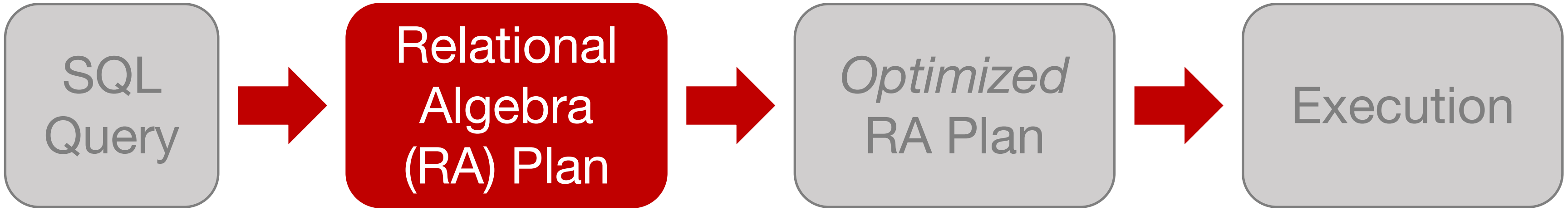
An algebra whose operands are relations or variables that represent relations.

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.

RDBMS Architecture

How does a SQL engine work?



Declarative query (from user)

Translate to relational algebra expression

Find logically equivalent- but more efficient- RA expression

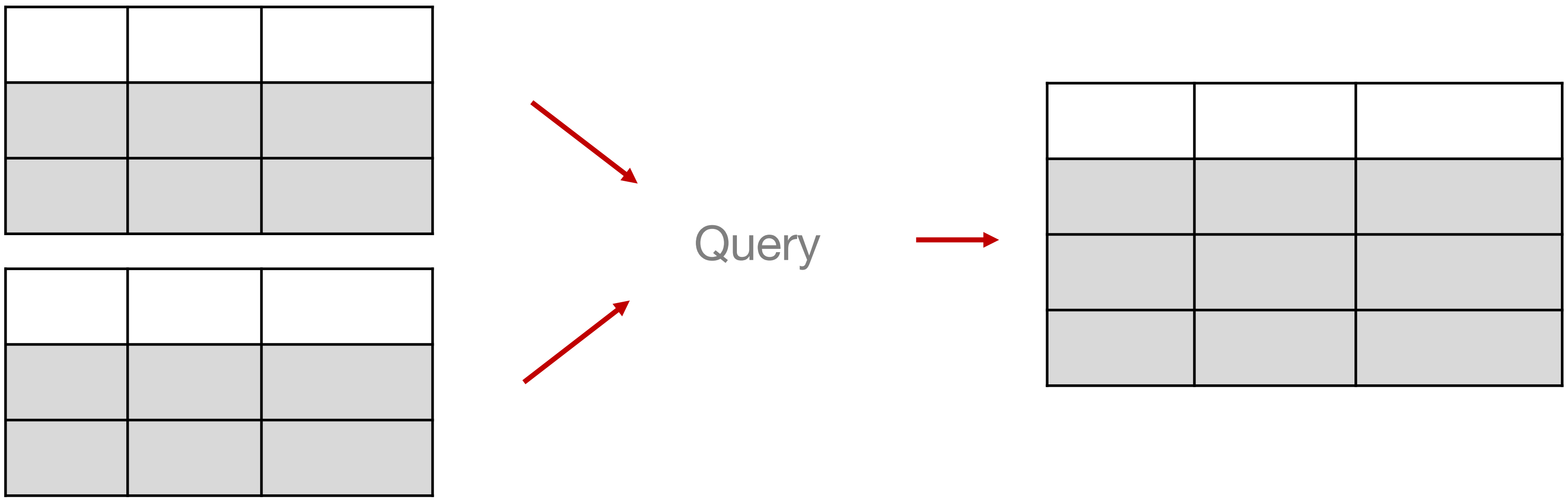
Execute each operator of the optimized plan

Relational Algebra

Formal language

Operates on relations and produce relations as a result

Operators are used to filter, slice and combine



Agenda

~~Introduction to Relational Algebra~~

Operators

Alternate notations

Extensions to Relational Algebra

College Admission Database

College (<u>cName</u>, state, enr)

Student (<u>sID</u>, sName, GPA, HS)

Apply (<u>sID</u>, <u>cName</u>, <u>major</u>, dec)

Demo in Relax: https://dbis-uibk.github.io/relax/

College

<u>cName</u>	**state**	**enr**

Student

<u>sID</u>	**sName**	**GPA**	**HS**

Apply

<u>sID</u>	**<u>cName</u>**	**<u>major</u>**	**dec**

Simplest query: relation name

Student

Student

sID	sName	GPA	HS

Select operator (σ)

Returns all tuples which satisfy a condition

Notation: $\sigma_{condition} Relation$

The condition can involve $=, <, \leq, >, \geq, <>$

Examples

Students with GPA>3.7

$\sigma_{GPA>3.7}\ Student$

Students with GPA>3.7 and HS<1000

$\sigma_{GPA>3.7 \wedge HS<1000}\ Student$

Applications to Stanford CS major

$\sigma_{cName='Stanford' \wedge major='CS'}\ Apply$

Student

sID	sName	GPA	HS
12	Mary	3.5	90
23	John	3.8	500
31	Jane	3.9	1000

Apply

sID	cName	major	dec
12	Stanford	CS	Y
23	MIT	CS	N
12	MIT	CS	N

Project operator (π)

Picks certain columns

Notation: $\pi_{A_1,\ldots,A_n}\ Relation$

sID and decision of all applications

$\pi_{sID,dec}\ Apply$

Apply

sID	**cName**	**major**	**dec**
12	Stanford	CS	Y
23	MIT	CS	N
12	MIT	CS	N

sID	**dec**
12	Y
23	N
12	N

Combining the Select and Project Operators

ID and name of students with GPA>3.7

$$\pi_{sID,sName}\ (\sigma_{GPA>3.7}\ Student)$$

Redefinition of operators

$$\sigma_{condition}(Expression)$$

$$\pi_{A_1,\ldots,A_n}\ (Expression)$$

Student

sID	sName	GPA	HS
12	Mary	3.5	90
23	John	3.8	50
31	Jane	3.9	1000

Sets, Bags and Lists

Sets

- Only one occurrence of each element
- Unordered elements

Bags (or multisets)

- More than one occurrence of an element
- Unordered elements and their occurrences

Lists

- More than one occurrence of an element
- Occurrences are ordered

Duplicates

Relational Algebra

Eliminates duplicates

Based on sets (although there is also a multiset relation algebra)

SQL

Does not eliminate duplicates

Based on multisets or bags

List of application majors and decisions

$\pi_{major,dec}\ Apply$

No duplicates

Apply

sID	cName	major	dec
12	Stanford	CS	Y
23	MIT	CS	N
12	MIT	CS	N

$\xrightarrow{\pi_{major,dec}\ Apply}$

major	dec
CS	Y
CS	N

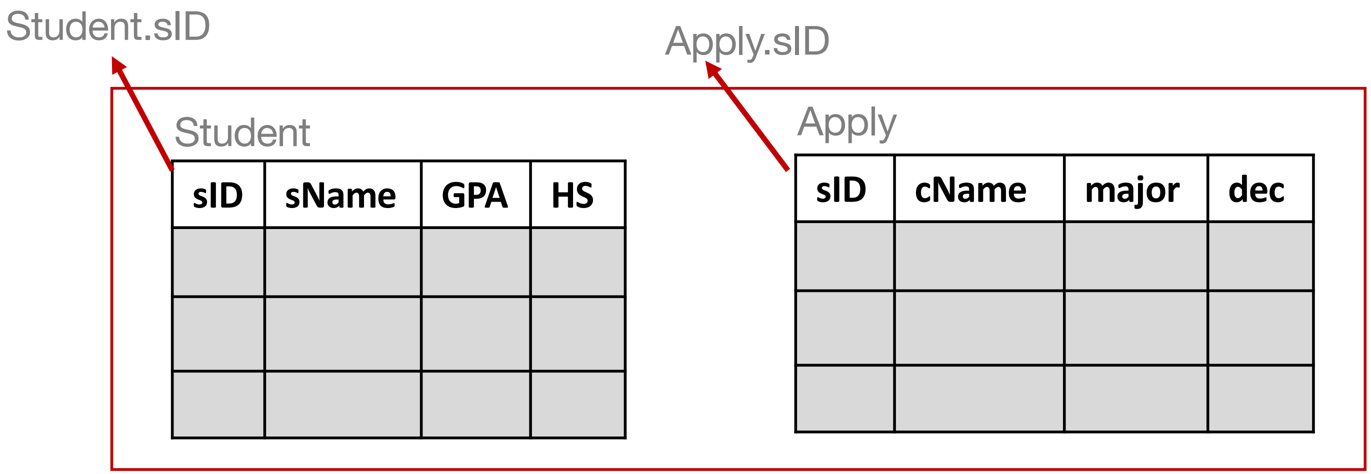
Cross-product

Also known as Cartesian product

Notation: Rel1 x Rel2

Student x Apply

Attributes with the same name are prefaced with the name of the relation



Cross-product

One tuple for every combination of tuples from the student and apply relations

Student

sID	sName	GPA	HS

S tuples

Student x Apply

S x A tuples

Apply

sID	cName	major	dec

A tuples

Example 1

Student

sID	sName	GPA	HS
12	Mary	3.5	90
23	John	3.8	50

Apply

sID	cName	major	dec
12	Stanford	CS	Y
23	MIT	CS	N

Student x Apply

Student.sID	sName	GPA	HS	Apply.sID	cName	major	dec
12	Mary	3.5	90	12	Stanford	CS	Y
12	Mary	3.5	90	23	MIT	CS	N
23	John	3.8	50	12	Stanford	CS	Y
23	John	3.8	50	23	MIT	CS	N

Example 2

Names and GPAs of students with HS>100 who applied to CS and were rejected

Student x Apply — All combinations

$\sigma_{Student.sID=Apply.sID}(Student\ x\ Apply)$ — Combinations that make sense

$\sigma_{Student.sID=Apply.sID \wedge HS>100 \wedge major='CS' \wedge dec='N'}(Student\ x\ Apply)$ — Additional filtering

$\pi_{sName,GPA}(\sigma_{Student.sID=Apply.sID \wedge HS>100 \wedge major='CS' \wedge dec='N'}(Student\ x\ Apply))$

Student

sID	sName	GPA	HS
12	Mary	3.5	90
23	John	3.8	5000

Apply

sID	cName	major	dec
12	Stanford	CS	Y
23	MIT	CS	N

Natural Join

Operator: ⋈

Cross product enforcing equality on all attributes with same name

Eliminate one copy of duplicate attributes

College

cName	**state**	**enr**

Student

sID	**sName**	**GPA**	**HS**

Apply

sID	**cName**	**major**	**dec**

Example 1

Student

sID	sName	GPA	HS
12	Mary	3.5	90
23	John	3.8	50

Apply

sID	cName	major	dec
12	Stanford	CS	Y
23	MIT	CS	N

$Student \bowtie Apply$

sID	sName	GPA	HS	cName	major	dec
12	Mary	3.5	90	Stanford	CS	Y
23	John	3.8	50	MIT	CS	N

Example 2

Names and GPAs of students with HS>100 who applied to CS and were rejected

$$Student \bowtie Apply$$

$$\sigma_{HS>100 \wedge major='CS' \wedge dec='N'}(Student \bowtie Apply)$$

$$\pi_{sName,GPA}(\sigma_{HS>100 \wedge major='CS' \wedge dec='N'}(Student \bowtie Apply))$$

Student

sID	sName	GPA	HS
12	Mary	3.5	90
23	John	3.8	5000

Apply

sID	cName	major	dec
12	Stanford	CS	Y
23	MIT	CS	N

Example 3

Names and GPAs of students with HS>100 who applied to CS at college with enr>10,000 and were rejected

$$Student \bowtie (Apply \bowtie College)$$

$$\sigma_{HS>100 \wedge major='CS' \wedge dec='N' \wedge enr>10,000}(Student \bowtie (Apply \bowtie College))$$

$$\pi_{sName,GPA}(\sigma_{HS>100 \wedge major='CS' \wedge dec='N' \wedge enr>10,000}(Student \bowtie (Apply \bowtie College)))$$

College

cName	state	enr
MIT	NULL	30000
Stanford	NULL	20000

Student

sID	sName	GPA	HS
12	Mary	3.5	90
23	John	3.8	5000

Apply

sID	cName	major	dec
12	Stanford	CS	Y
23	MIT	CS	N

Natural Join

Given R(A, B, C, D), S(A, C, E), what is the schema of $R \bowtie S$?

Given R(A, B, C), S(D, E), what is $R \bowtie S$?

Given R(A, B), S(A, B), what is $R \bowtie S$?

Natural Join does not add expressive power

Can be rewritten using the cross-product

$$Exp1 \bowtie Exp2 \equiv \pi_{schema(E1) \cup schema(E2)}(\sigma_{E1.A1=E2.A1 \wedge E1.A2=E2.A2 \wedge \ldots}(Exp1\ x\ Exp2))$$

It is convenient in terms of notation

Theta Join

A join that involves a predicate

Notation: $\bowtie_\theta$

$Exp_1 \bowtie_\theta Exp_2 \equiv \sigma_\theta(Exp_1 \; x \; Exp_2)$

Basic operation implemented in DBMS

Term “join” often means theta join

θ can be any condition

- If θ is an equality, the join is called an equi-join

Example

Student

ID	sName	GPA	HS
12	Mary	3.5	90
23	John	3.8	50

Apply

sID	cName	major	dec
12	Stanford	CS	Y
23	MIT	CS	N

$Student \bowtie_{ID=sID} Apply$

ID	sName	GPA	HS	sID	cName	major	dec
12	Mary	3.5	90	12	Stanford	CS	Y
23	John	3.8	50	23	MIT	CS	N

Semijoin

Notation: $\ltimes$

$$Exp_1 \ltimes Exp_2 \equiv \pi_{A1,\dots,An}\,(Exp_1 \bowtie Exp_2)$$

Where $A_1, \dots, A_n$ are atributes in Exp_1

Returns the tuples of Exp_1 with a pair in Exp_2

Example

Student

sID	sName	GPA	HS
12	Mary	3.5	90
23	John	3.8	50
35	Jane	3.9	60

Apply

sID	cName	major	dec
12	Stanford	CS	Y
23	MIT	CS	N

$Student \ltimes Apply$

sID	sName	GPA	HS
12	Mary	3.5	90
23	John	3.8	50

Union operator

Operator: ∪

List of college and student names

Can we do it using previous operators?

$\pi_{cName} College \cup \pi_{sName} Student$

Combines information vertically

Technically, the two operands have to have the same schema

Not the case in the example above, but we'll correct it later

College

cName	state	enr
MIT	NULL	NULL
Washington	NULL	NULL

Student

sID	sName	GPA	HS
12	Mary	3.5	90
23	Washington	3.8	50

Example

Student

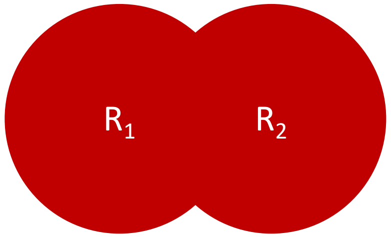
sID	sName	GPA	HS
12	Mary	3.5	90
23	Washington	3.8	50

College

cName	state	enr
MIT	NULL	NULL
Washington	NULL	NULL

$\pi_{cName}College \cup \pi_{sName}Student$

cName
Mary
Washington
MIT



Difference operator

Operator: –

IDs of students who didn't apply anywhere

Student

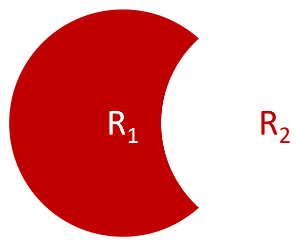
sID	sName	GPA	HS
12	Mary	3.5	90
23	Washington	3.8	50

Apply

sID	cName	major	dec
12	Stanford	CS	Y

$$\pi_{sID} Student - \pi_{sID} Apply$$

sID
23



Example

Names of students who didn't apply anywhere

$\pi_{sName}Student - \pi_{sID}Apply$?

$$\pi_{sName}(\underbrace{(\pi_{sID}Student - \pi_{sID}Apply) \bowtie Student}_{\text{Schema equal to the student relation}})$$

Student

sID	sName	GPA	HS
12	Mary	3.5	90
23	Washington	3.8	50

$\pi_{sID}Student - \pi_{sID}Apply$

sID
23

Apply

sID	cName	major	dec
12	Stanford	CS	Y

Intersection operator

Operator: ∩

Names that are both a college name and a student name

$$\pi_{cName} College \cap \pi_{sName} Student$$

Technically, the two operands have to have the same schema

Not the case in the example above, but we'll correct it later

College

cName	state	enr
MIT	NULL	NULL
Washington	NULL	NULL

Student

sID	sName	GPA	HS
12	Mary	3.5	90
23	Washington	3.8	50

Example

Student

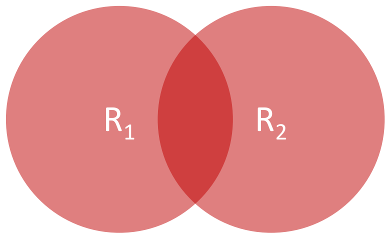
sID	sName	GPA	HS
12	Mary	3.5	90
23	Washington	3.8	50

College

cName	state	enr
MIT	NULL	NULL
Washington	NULL	NULL

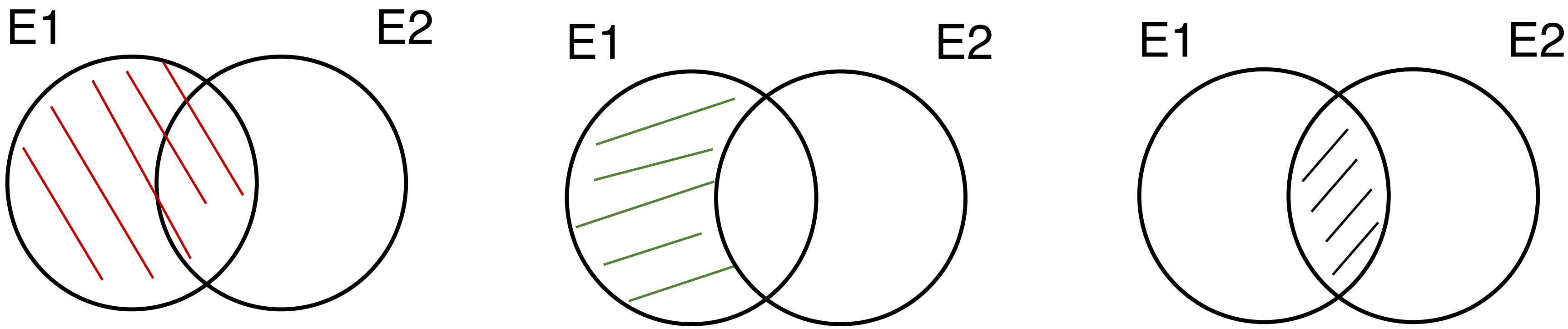
$$\pi_{cName} College \cap \pi_{sName} Student$$

cName
Washington



Intersection doesn't add expressive power

$$E_1 \cap E_2 \equiv E_1 - (E_1 - E_2)$$



Intersection doesn't add expressive power

$$E_1 \cap E_2 \equiv E_1 \bowtie E_2$$

Identical schema

Nevertheless, the intersection can be very useful in queries

Kahoot time!

Any doubts?

Readings

Jeffrey Ullman, Jennifer Widom, A first course in Database Systems 3rd Edition

- Section 2.4 – An Algebraic Query Language
- Section 5.2 – Extended Operators of Relational Algebra