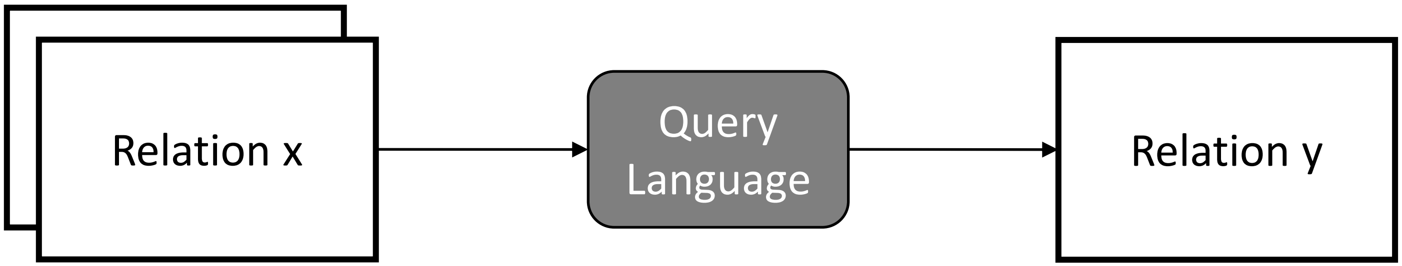
Relational Algebra and Query Languages

Venkatesh Vinayakarao
venkateshv@cmi.ac.in
http://vvtesh.co.in

Chennai Mathematical Institute

Relational Algebra and Query Languages

Query Languages

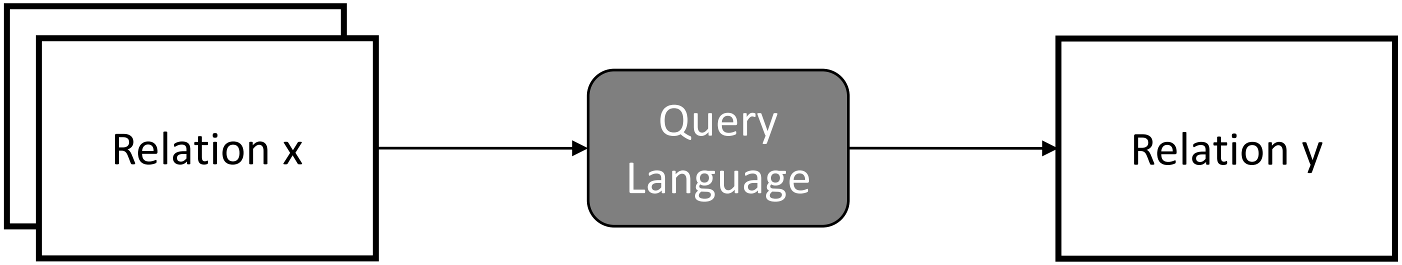


Procedural Language
Relational Algebra

Declarative Languages
Tuple Relational Calculus
Domain Relational Calculus

Popular Language
SQL

Relational Algebra



Fundamental Operations

select, project, rename

set difference, cartesian product, union

More Operations

set intersection, natural join, assignment

Select Operation

- Notation: $\sigma_p(r)$ where p is called the selection predicate

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

(a) The *instructor* table

- Example of selection:

$$\sigma_{\text{dept\_name}=\text{“Physics”}}(\text{instructor})$$

Select Operation

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

(a) The *instructor* table

- Example of selection:

$$\sigma_{\text{dept\_name}=\text{“Physics”}}(\text{instructor})$$

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
33456	Gold	Physics	87000

Project Operation

- Notation: $\prod_{A1, A2,..,Ak} (r)$ where A_i are attribute names

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

(a) The *instructor* table

- Example of projection:

$$\prod_{ID,\ name,\ salary} (instructor)$$

- Duplicate rows removed from result, since relations are sets

Projection

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

(a) The *instructor* table

$\Pi_{ID,\ name,\ salary}$ (*instructor*)

ID	*name*	*salary*
10101	Srinivasan	65000
12121	Wu	90000
15151	Mozart	40000
22222	Einstein	95000
32343	El Said	60000
33456	Gold	87000
45565	Katz	75000
58583	Califieri	62000
76543	Singh	80000
76766	Crick	72000
83821	Brandt	92000
98345	Kim	80000

Union Operation

- Notation: $r \cup s$. Defined as:

$$r \cup s = \{t \mid t \in r \text{ or } t \in s\}$$

- For $r \cup s$ to be valid:
  - *r, s* must have the *same* **arity** (same number of attributes)
  - The attribute domains must be **compatible** (example: 2nd column of *r* deals with the same type of values as does the 2nd column of *s*)

- Example: to find all courses taught in the Fall 2009 semester, or in the Spring 2010 semester, or in both

$$\Pi_{course\_id} (\sigma_{semester=\text{“Fall”} \land year=2009} (section)) \cup$$

$$\Pi_{course\_id} (\sigma_{semester=\text{“Spring”} \land year=2010} (section))$$

Union, Selection and Projection

- $\Pi_{course\_id}\,(\sigma_{semester=\text{“Fall”}\ \land\ year=2009}\,(section))\ \cup$
  $\Pi_{course\_id}\,(\sigma_{semester=\text{“Spring”}\ \land\ year=2010}\,(section))$

course_id	*sec_id*	*semester*	*year*	*building*	*room_number*	*time_slot_id*
BIO-101	1	Summer	2009	Painter	514	B
BIO-301	1	Summer	2010	Painter	514	A
CS-101	1	Fall	2009	Packard	101	H
CS-101	1	Spring	2010	Packard	101	F
CS-190	1	Spring	2009	Taylor	3128	E
CS-190	2	Spring	2009	Taylor	3128	A
CS-315	1	Spring	2010	Watson	120	D
CS-319	1	Spring	2010	Watson	100	B
CS-319	2	Spring	2010	Taylor	3128	C
CS-347	1	Fall	2009	Taylor	3128	A
EE-181	1	Spring	2009	Taylor	3128	C
FIN-201	1	Spring	2010	Packard	101	B
HIS-351	1	Spring	2010	Painter	514	C
MU-199	1	Spring	2010	Packard	101	D
PHY-101	1	Fall	2009	Watson	100	A

➡

course_id
CS-101
CS-315
CS-319
CS-347
FIN-201
HIS-351
MU-199
PHY-101

Set Difference Operation

- Notation: $r - s$. Defined as

$$r - s = \{t \mid t \in r \textbf{ and } t \notin s\}$$

- Example: to find all courses taught in the Fall 2009 semester, but not in the Spring 2010 semester

$$\Pi_{course\_id}\,(\sigma_{semester=\text{“Fall”}\ \land\ year=2009}\,(section))\ -$$
$$\Pi_{course\_id}\,(\sigma_{semester=\text{“Spring”}\ \land\ year=2010}\,(section))$$

Union, Selection and Projection

- $\Pi_{course\_id}(\sigma_{semester=\text{"Fall"} \wedge year=2009}(section))$ –
  $\Pi_{course\_id}(\sigma_{semester=\text{"Spring"} \wedge year=2010}(section))$

course_id	sec_id	semester	year	building	room_number	time_slot_id
BIO-101	1	Summer	2009	Painter	514	B
BIO-301	1	Summer	2010	Painter	514	A
CS-101	1	Fall	2009	Packard	101	H
CS-101	1	Spring	2010	Packard	101	F
CS-190	1	Spring	2009	Taylor	3128	E
CS-190	2	Spring	2009	Taylor	3128	A
CS-315	1	Spring	2010	Watson	120	D
CS-319	1	Spring	2010	Watson	100	B
CS-319	2	Spring	2010	Taylor	3128	C
CS-347	1	Fall	2009	Taylor	3128	A
EE-181	1	Spring	2009	Taylor	3128	C
FIN-201	1	Spring	2010	Packard	101	B
HIS-351	1	Spring	2010	Painter	514	C
MU-199	1	Spring	2010	Packard	101	D
PHY-101	1	Fall	2009	Watson	100	A

→

course_id
CS-347
PHY-101

Set-Intersection Operation

- Notation: $r \cap s$
- Defined as:
- $r \cap s = \{\, t \mid t \in r$ **and** $t \in s \,\}$

Quiz: True/False?

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

Cartesian-Product Operation

- Notation *r* x *s*

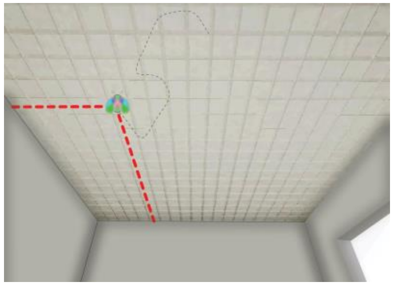
ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000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

instructor relation

ID	*course_id*	*sec_id*	*semester*	*year*
10101	CS-101	1	Fall	2009
10101	CS-315	1	Spring	2010
10101	CS-347	1	Fall	2009
12121	FIN-201	1	Spring	2010
15151	MU-199	1	Spring	2010
22222	PHY-101	1	Fall	2009
32343	HIS-351	1	Spring	2010
45565	CS-101	1	Spring	2010
45565	CS-319	1	Spring	2010
76766	BIO-101	1	Summer	2009
76766	BIO-301	1	Summer	2010
83821	CS-190	1	Spring	2009
83821	CS-190	2	Spring	2009
83821	CS-319	2	Spring	2010
98345	EE-181	1	Spring	2009

teaches relation

The Fly on the Ceiling - Descartes



French mathematician René Descartes (1596-1650)

Image Source:
http://sites.psu.edu/solvingproblems
https://wild.maths.org/ren%C3%A9-descartes-and-fly-ceiling
vectorstock.com

The *instructor* X *teaches table*

instructor.ID	*name*	*dept_name*	*salary*	*teaches.ID*	*course_id*	*sec_id*	*semester*	*year*
10101	Srinivasan	Comp. Sci.	65000	10101	CS-101	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	10101	CS-315	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	10101	CS-347	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	12121	FIN-201	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	15151	MU-199	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
12121	Wu	Finance	90000	10101	CS-101	1	Fall	2017
12121	Wu	Finance	90000	10101	CS-315	1	Spring	2018
12121	Wu	Finance	90000	10101	CS-347	1	Fall	2017
12121	Wu	Finance	90000	12121	FIN-201	1	Spring	2018
12121	Wu	Finance	90000	15151	MU-199	1	Spring	2018
12121	Wu	Finance	90000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
15151	Mozart	Music	40000	10101	CS-101	1	Fall	2017
15151	Mozart	Music	40000	10101	CS-315	1	Spring	2018
15151	Mozart	Music	40000	10101	CS-347	1	Fall	2017
15151	Mozart	Music	40000	12121	FIN-201	1	Spring	2018
15151	Mozart	Music	40000	15151	MU-199	1	Spring	2018
15151	Mozart	Music	40000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
22222	Einstein	Physics	95000	10101	CS-101	1	Fall	2017
22222	Einstein	Physics	95000	10101	CS-315	1	Spring	2018
22222	Einstein	Physics	95000	10101	CS-347	1	Fall	2017
22222	Einstein	Physics	95000	12121	FIN-201	1	Spring	2018
22222	Einstein	Physics	95000	15151	MU-199	1	Spring	2018
22222	Einstein	Physics	95000	22222	PHY-101	1	Fall	2017
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...

Join Operation

- The Cartesian-Product *instructor X teaches* associates every tuple of instructor with every tuple of teaches.
- Most of the resulting rows have information about instructors who did NOT teach a particular course.
- To get only those tuples of instructor X teaches that pertain to instructors and the courses that they taught, we write:

$$\sigma_{instructor.id = teaches.id}(instructor \times teaches))$$

Join Operation (Cont.)

- $\sigma_{\,instructor.id\ =\ teaches.id}\,(instructor\ \ x\ teaches))$

instructor.ID	*name*	*dept_name*	*salary*	*teaches.ID*	*course_id*	*sec_id*	*semester*	*year*
10101	Srinivasan	Comp. Sci.	65000	10101	CS-101	1	Fall	2017
10101	Srinivasan	Comp. Sci.	65000	10101	CS-315	1	Spring	2018
10101	Srinivasan	Comp. Sci.	65000	10101	CS-347	1	Fall	2017
12121	Wu	Finance	90000	12121	FIN-201	1	Spring	2018
15151	Mozart	Music	40000	15151	MU-199	1	Spring	2018
22222	Einstein	Physics	95000	22222	PHY-101	1	Fall	2017
32343	El Said	History	60000	32343	HIS-351	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-101	1	Spring	2018
45565	Katz	Comp. Sci.	75000	45565	CS-319	1	Spring	2018
76766	Crick	Biology	72000	76766	BIO-101	1	Summer	2017
76766	Crick	Biology	72000	76766	BIO-301	1	Summer	2018
83821	Brandt	Comp. Sci.	92000	83821	CS-190	1	Spring	2017
83821	Brandt	Comp. Sci.	92000	83821	CS-190	2	Spring	2017
83821	Brandt	Comp. Sci.	92000	83821	CS-319	2	Spring	2018
98345	Kim	Elec. Eng.	80000	98345	EE-181	1	Spring	2017

Join Operation (Theta Join)

- The **join** operation allows us to combine a select operation and a Cartesian-Product operation into a single operation.
- Consider relations *r* (*R*) and *s* (*S*). Let "theta" be a predicate on attributes in the schema R "union" S. The join operation $r \bowtie_{\theta} s$ is defined as follows:

$$r \bowtie_{\theta} s = \sigma_{\theta}\,(r \times s)$$

- Thus

$$\sigma_{instructor.id = teaches.id}\,(instructor \times teaches\,))$$

- Can equivalently be written as

$$instructor \bowtie_{instructor.id = teaches.id} teaches.$$

Natural Join

- A special case where equality predicate holds on all attributes of same name.

Employee

Name	EmpId	DeptName
Harry	3415	Finance
Sally	2241	Sales
George	3401	Finance
Harriet	2202	Sales
Mary	1257	Human Resources

Dept

DeptName	Manager
Finance	George
Sales	Harriet
Production	Charles

Employee ⋈ Dept

Name	EmpId	DeptName	Manager
Harry	3415	Finance	George
Sally	2241	Sales	Harriet
George	3401	Finance	George
Harriet	2202	Sales	Harriet

Note that neither the employee named Mary nor the Production department appear in the result.

Source: https://en.wikipedia.org/wiki/Relational_algebra#Natural_join_(%E2%8B%88)

Rename Operation

- 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

$$\rho_{x(A_1, A_2, \ldots, A_n)}(E)$$

returns the result of expression *E* under the name *X*, and with the attributes renamed to $A_1, A_2, \ldots, A_n$.

Find the highest salary in the university

- Steps to reach the solution:

We find all salaries that are lesser than some existing salary.

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

➡

salary
65000
90000
40000
60000
87000
75000
62000
72000
80000
92000

➡ **95000**

The only salary left out in step2 is the max salary.

Find the highest salary in the university

- Compute instructor x instructor
- Compute

$$\Pi_{instructor.salary}\,(\sigma_{instructor.salary\ <\ d.salary}\,(instructor\ \times\ \rho_d\,(instructor)))$$

- Compute

$$\Pi_{salary}\,(instructor) -$$
$$\Pi_{instructor.salary}\,(\sigma_{instructor.salary\ <\ d.salary}\,(instructor\ \times\ \rho_d\,(instructor)))$$

We use rename operation to distinguish the two salary attributes.

Relational Algebra

- A basic expression in the relational algebra consists of either one of the following:
  - A relation in the database
  - A constant relation
- Let E_1 and E_2 be relational-algebra expressions; the following are all relational-algebra expressions:
  - $E_1 \cup E_2$
  - $E_1 - E_2$
  - $E_1 \times E_2$
  - $\sigma_p (E_1)$, P is a predicate on attributes in E_1
  - $\prod_s(E_1)$, S is a list consisting of some of the attributes in E_1
  - $\rho_x (E_1)$, x is the new name for the result of E_1

Problem

Consider the following relational schema.

Students(rollno: integer, sname: string)
Courses(courseno: integer, cname: string)
Registration(rollno: integer, courseno: integer, percent: real)

Is the following relational algebra expression equivalent to the English sentence given below:

"Find the distinct names of all students who score more than 90% in the course numbered 107"

$$\prod_{sname}(\sigma_{courseno=107 \wedge percent>90}(Registration \bowtie Students))$$

[GATE 2013]

Problem

Consider the relational schema given below, where eId of the relation dependent is a foreign key referring to empId of the relation employee. Assume that every employee has at least one associated dependent in the dependent relation.

```
employee (empId, empName, empAge)
dependent(depId, eId, depName, depAge)
```

Consider the following relational algebra query:

$$\Pi_{empId}(employee) - \Pi_{empId}(employee \bowtie_{(empId\ =\ eID) \land (empAge\ \leq\ depAge)} dependent)$$

The above query evaluates to the set of *empIds* of employees whose age is greater than that of

(A) some dependent.
(B) all dependents.
(C) some of his/her dependents
(D) all of his/her dependents.

[GATE 2014]

Problem

Consider the relational schema given below, where eId of the relation dependent is a foreign key referring to empId of the relation employee. Assume that every employee has at least one associated dependent in the dependent relation.

```
employee (empId, empName, empAge)
dependent(depId, eId, depName, depAge)
```

Consider the following relational algebra query:

$$\Pi_{empId}(employee) - \Pi_{empId}(employee \bowtie_{(empId = eID) \land (empAge \leq depAge)} dependent)$$

The above query evaluates to the set of *empIds* of employees whose age is greater than that of

(A) some dependent.
(B) all dependents.
(C) some of his/her dependents
(D) all of his/her dependents.

Fundamental Operations

The select, project, rename, set difference, cartesian product, and union are sufficient to express queries!

Extended Operations

- Set Intersection
  - Find the set of all courses taught in both the Fall 2017 and the Spring 2018 semesters

$$\Pi_{course\_id} (\sigma_{semester=\text{"Fall"} \land year=2017} (section)) \cap \Pi_{course\_id} (\sigma_{semester=\text{"Spring"} \land year=2018} (section))$$

Result

course_id
CS-101

- Assignment
  - Find all instructors in the "Physics" and Music department.

$$Physics \leftarrow \sigma_{dept\_name=\text{"Physics"}}(instructor)$$
$$Music \leftarrow \sigma_{dept\_name=\text{"Music"}}(instructor)$$
$$Physics \cup Music$$

Aggregate Functions

- Improves ease of use
- Most common functions:

Functions
sum
max
min
avg
count

Calligraphic G Notation

$$\mathcal{G}_{\mathbf{sum(c)}}(r)$$

Compute sum of all values of attribute c on relation r.

Summary

- Relational Algebra is very much like normal algebra (x – y). We use relations instead of numbers as basic expressions.
- Basis for commercial query languages such as SQL.
- Three major components:
  - Fundamental Operations
  - Extended Operations
  - Aggregate Functions

Revision

Table A		
Id	Name	Age
12	Arun	60
15	Shreya	24
99	Rohit	11

Table B		
Id	Name	Age
15	Shreya	24
25	Hari	40
98	Rohit	20
99	Rohit	11

Table C		
Id	Phone	Area
10	2200	02
99	2100	01

How many rows will this query produce?

$$(A \cup B) \bowtie_{A.Id > 40 \ \vee \ C.Id < 15} C$$

[GATE 2012]

Revision

- 7 Rows

Id	Name	Age	Id	Phone	Area
12	Arun	60	10	2200	02
15	Shreya	24	10	2200	02
99	Rohit	11	10	2200	02
25	Hari	40	10	2200	02
98	Rohit	20	10	2200	02
99	Rohit	11	99	2100	01
98	Rohit	20	99	2100	01