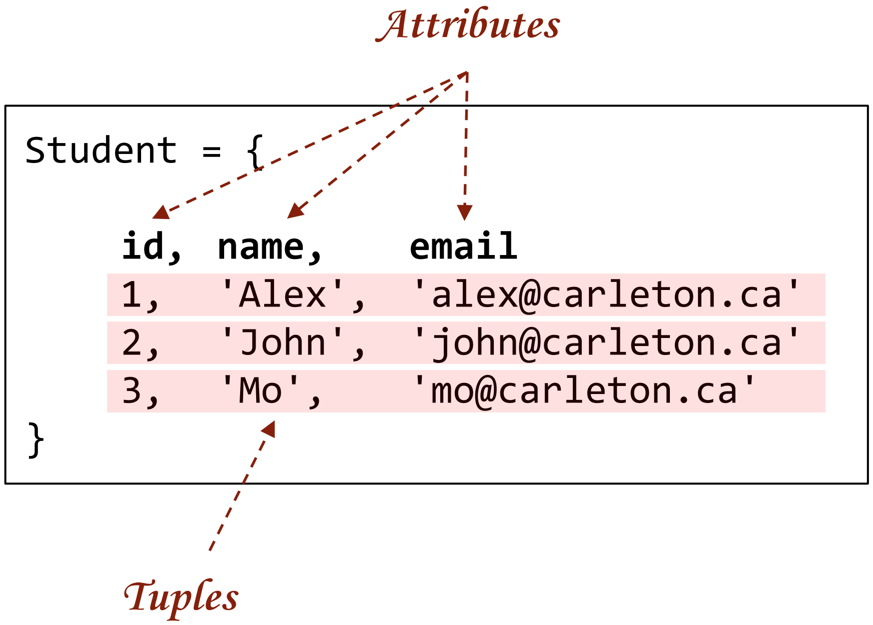
Relational Algebra

Recap Relational Database

Math Notation

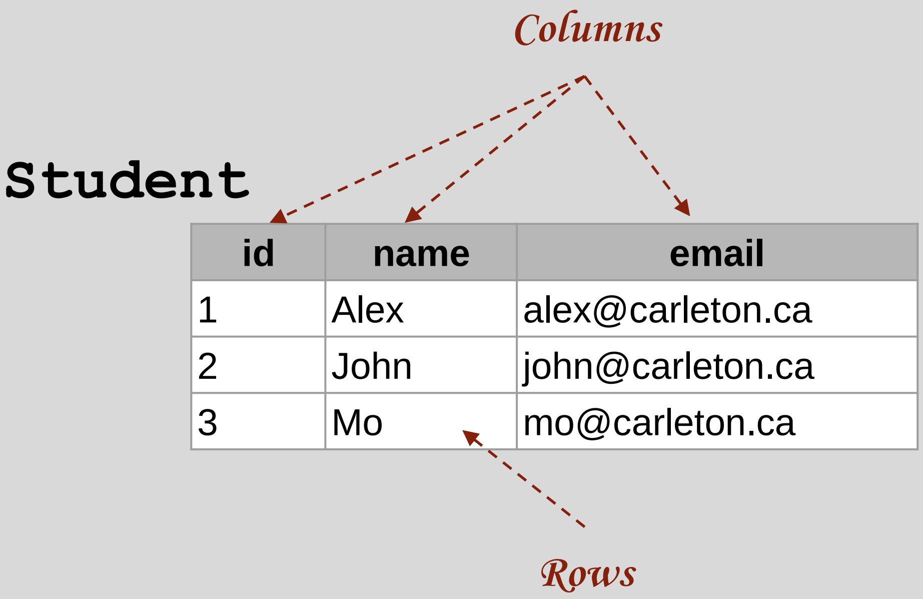
Student Relation



```
Student = {

     id, name,    email
     1,  'Alex',  'alex@carleton.ca'
     2,  'John',  'john@carleton.ca'
     3,  'Mo',    'mo@carleton.ca'
}
```

Visualization

Student Table



Student

id	name	email
1	Alex	alex@carleton.ca
2	John	john@carleton.ca
3	Mo	mo@carleton.ca

Recap Relational Database

Math Notation

Relation

```
Student_Course = {

    id, name,    email,                    cid,     title,   hours,   mark
    1,  'Alex', 'alex@carleton.ca',        1,       Math,    0.5      10
    1,  'Alex', 'alex@carleton.ca',        2,       Physics, 0.5      9
    1,  'Alex', 'alex@carleton.ca',        3,       DBMS,    0.5      10
    2,  'John', 'john@carleton.ca’         ...      ...
    3,  'Mo',   'mo@carleton.ca’           ...      ...
}
```

One Relation = Redundancy Problem

Recap Relational Database

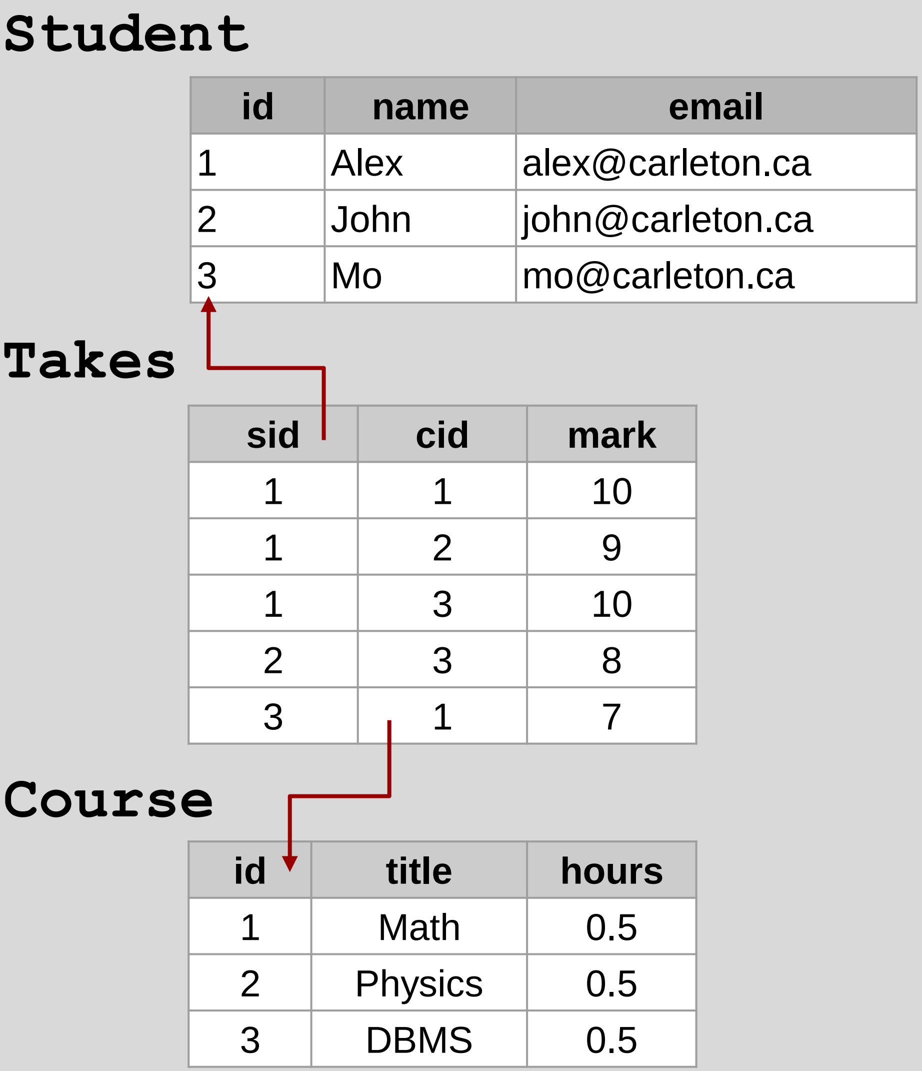
Math Notation

```
Student = {
     id,  name,    email
     1,   'Alex',  'alex@carleton.ca'
     2,   'John',  'john@carleton.ca'
     3,   'Mo',    'mo@carleton.ca'
}
```

```
takes = {
     sid, cid,     mark
     1,   1,       10
     1,   2,       9
     1,   3,       10
     2,   3,       8
     3,   1,       7
}
```

```
Course = {
     id,  title,             hours
     1,   'Math',            0.5
     2,   'Pythics',         0.5
     3,   'DBMS',            0.5
}
```

Visualization



Student

id	name	email
1	Alex	alex@carleton.ca
2	John	john@carleton.ca
3	Mo	mo@carleton.ca

Takes

sid	cid	mark
1	1	10
1	2	9
1	3	10
2	3	8
3	1	7

Course

id	title	hours
1	Math	0.5
2	Physics	0.5
3	DBMS	0.5

Recap Relational Database

Data Querying

Get the title of the courses of Alex.

title
Math
Physics
DBMS

1. From **Student**,
   Get Alex’s id
1. From **Takes**,
   Gets Courses’ ids associated with Alex’s id
1. From **Course**,
   Get the title of the courses’ ids you got in 2.

Visualization

Student

id	name	email
1	Alex	alex@carleton.ca
2	John	john@carleton.ca
3	Mo	mo@carleton.ca

Takes

sid	cid	mark
1	1	10
1	2	9
1	3	10
2	3	8
3	1	7

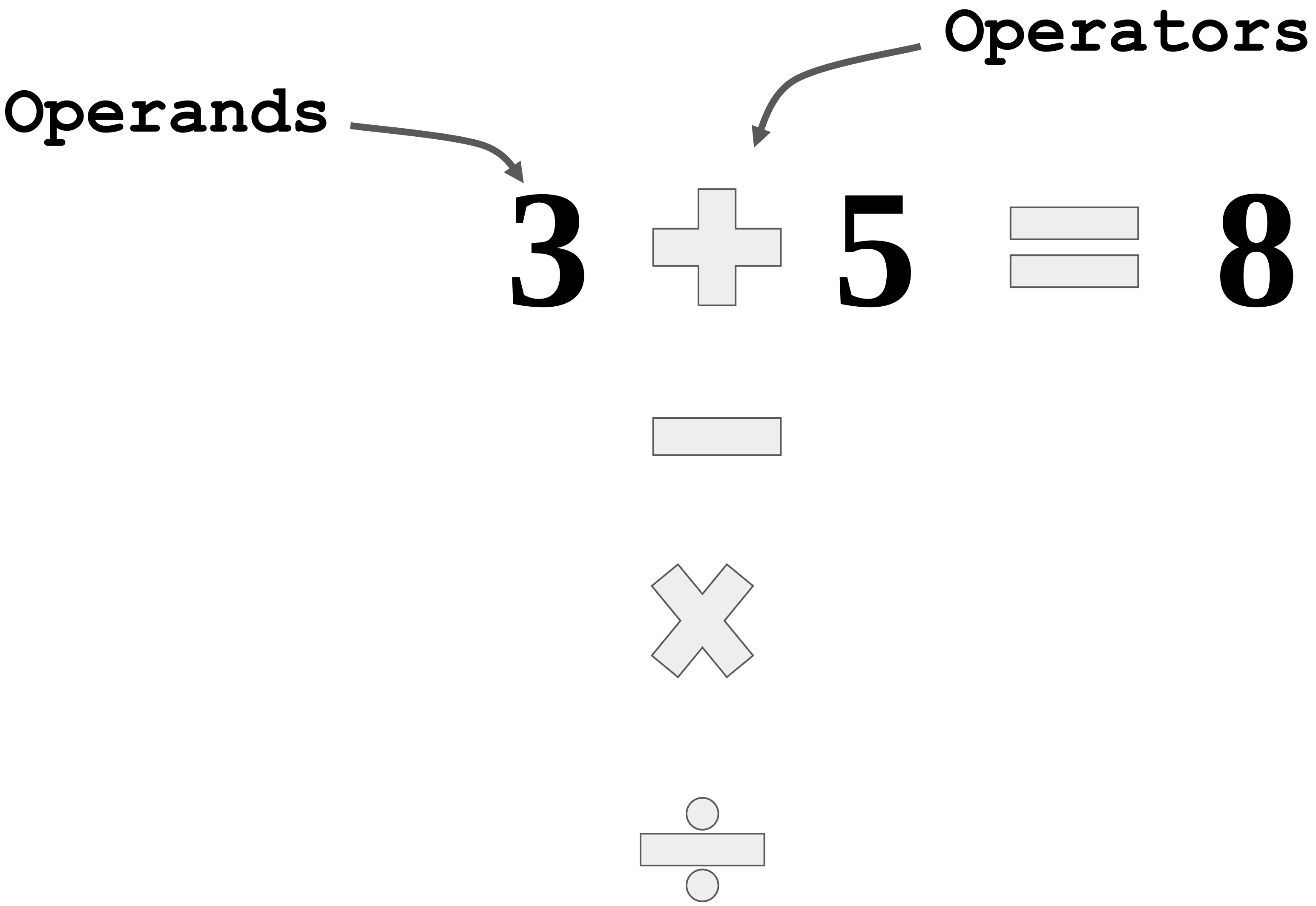
Course

id	title	hours
1	Math	0.5
2	Physics	0.5
3	DBMS	0.5

Relational Algebra

What is Algebra?

Algebra



Operators

Operands

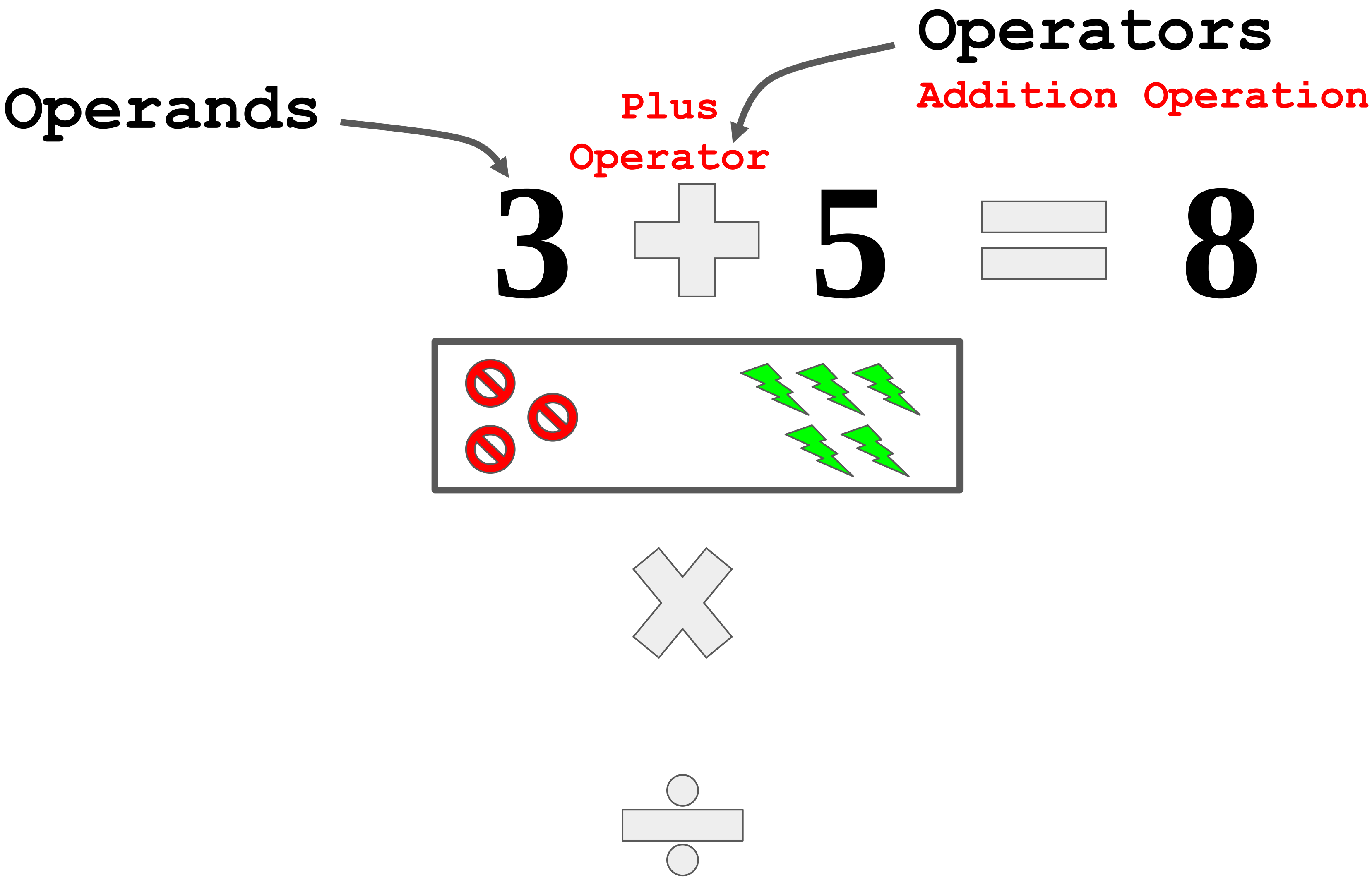
3 + 5 = 8

−

×

÷

Algebra



Operators

Addition Operation

Operands

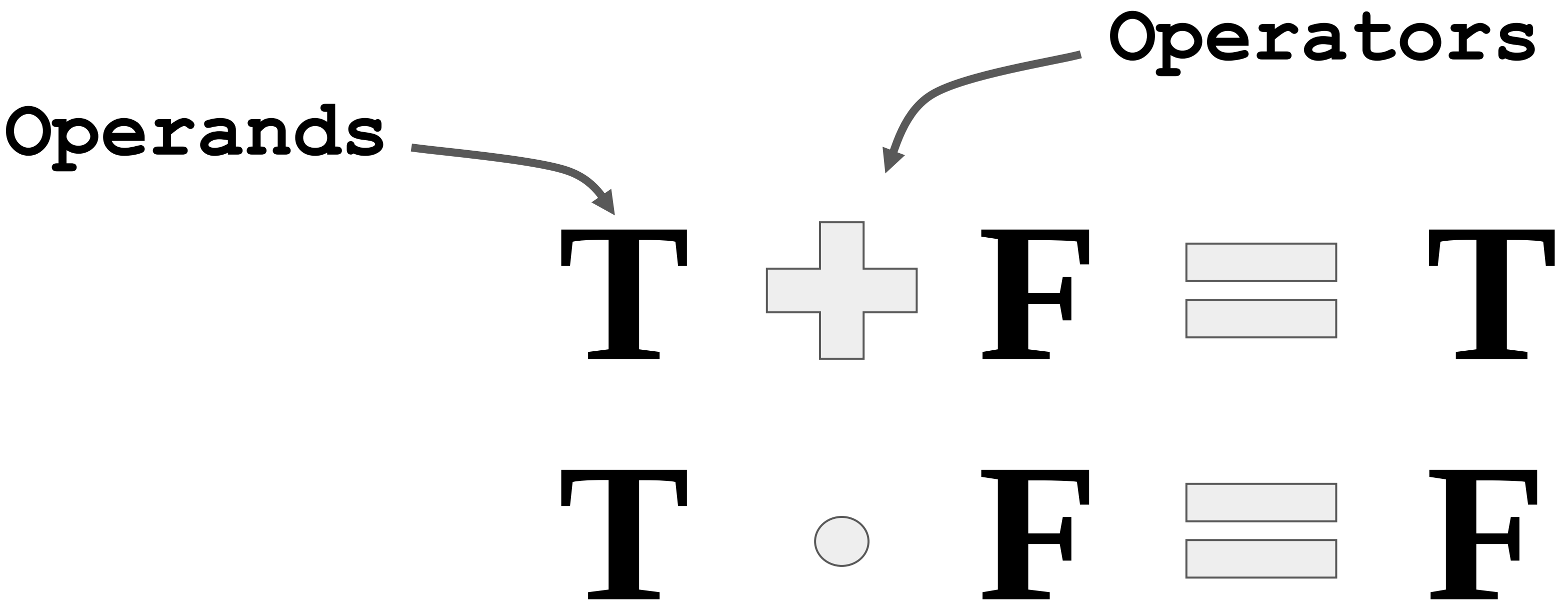
Plus Operator

3 + 5 = 8

×

÷

Boolean Algebra

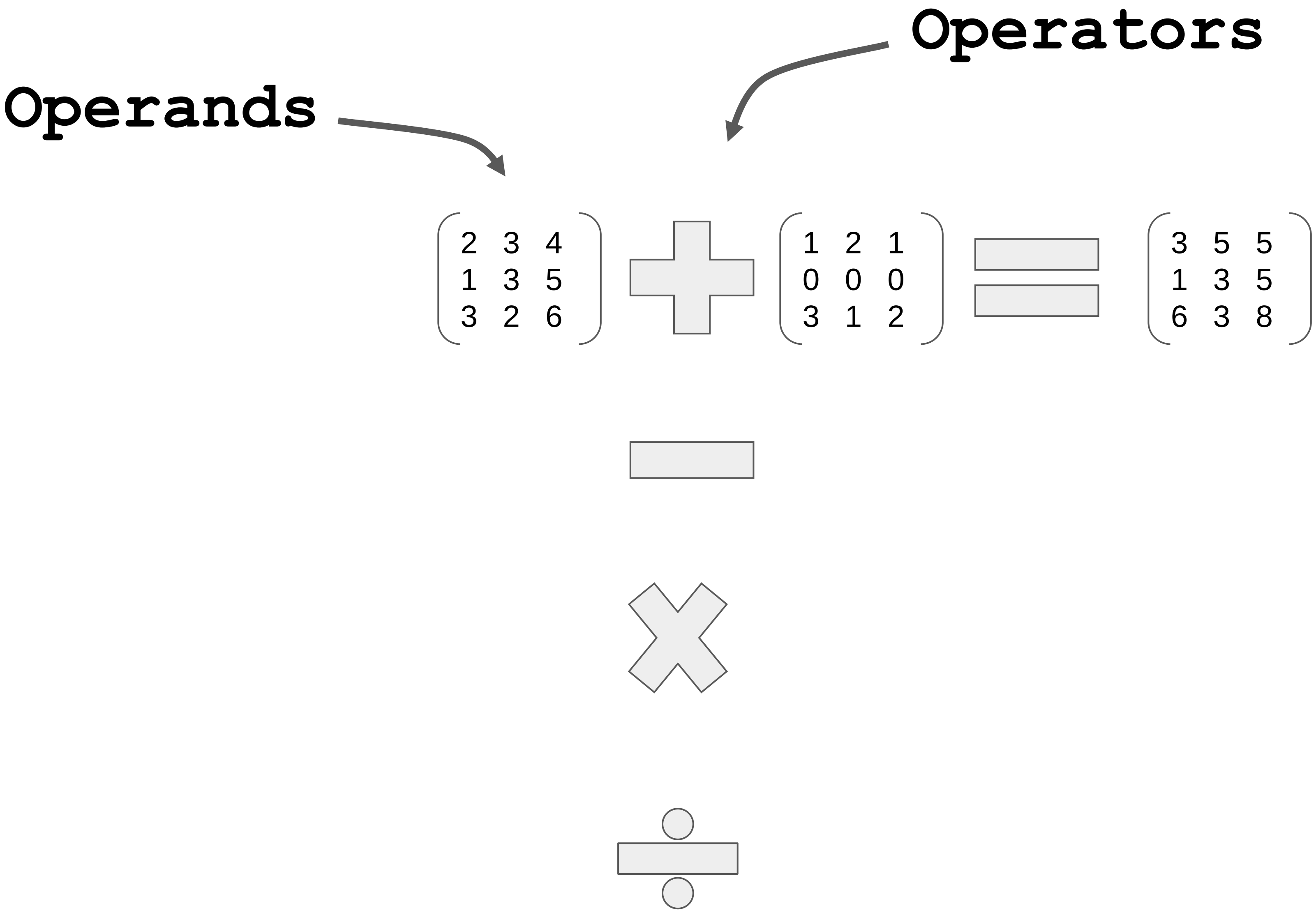


Operands

Operators

$$T + F = T$$

$$T \cdot F = F$$



Operands

Operators

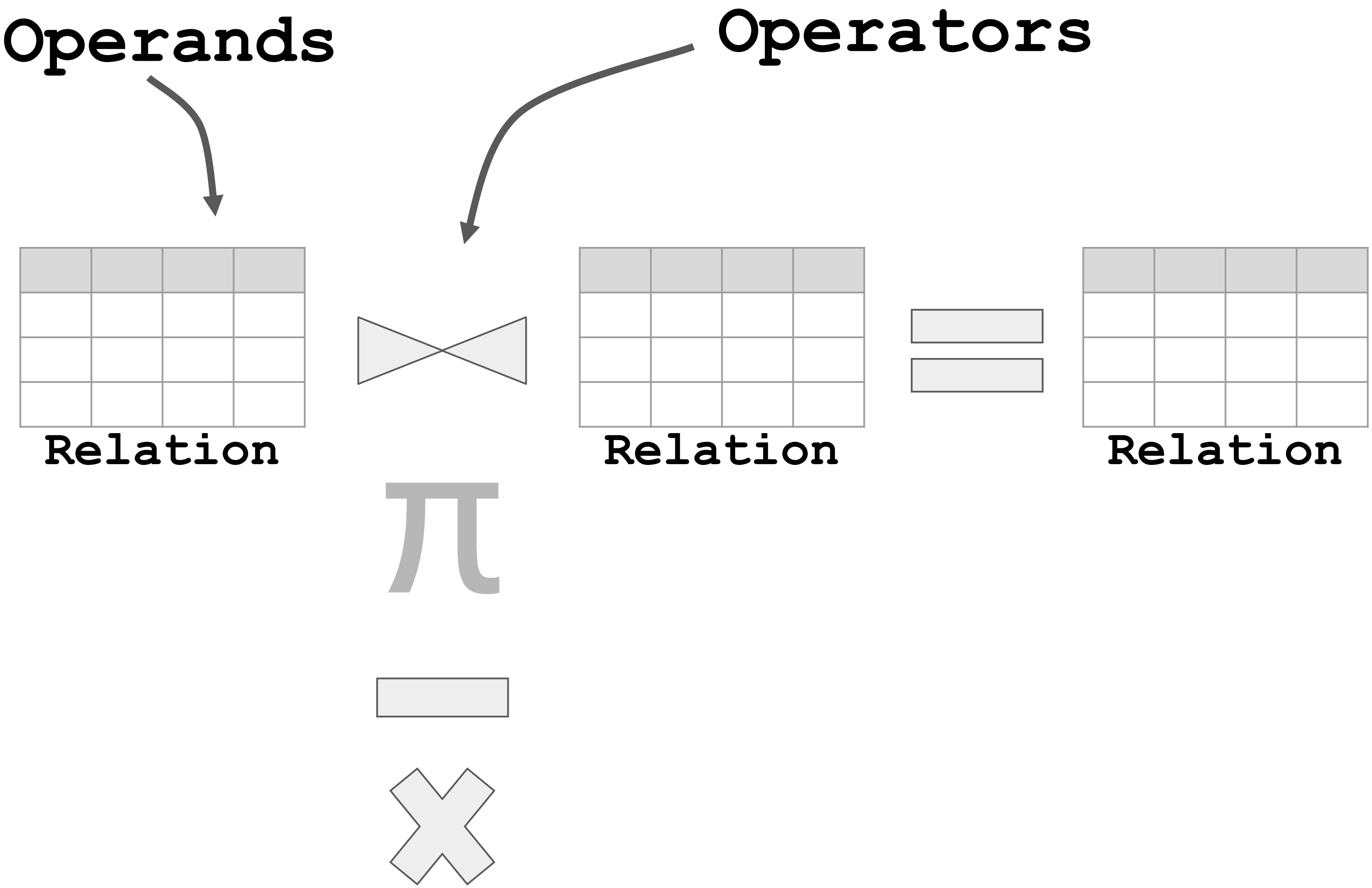
$$\begin{pmatrix} 2 & 3 & 4 \\ 1 & 3 & 5 \\ 3 & 2 & 6 \end{pmatrix} + \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ 3 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 3 & 5 & 5 \\ 1 & 3 & 5 \\ 6 & 3 & 8 \end{pmatrix}$$

−

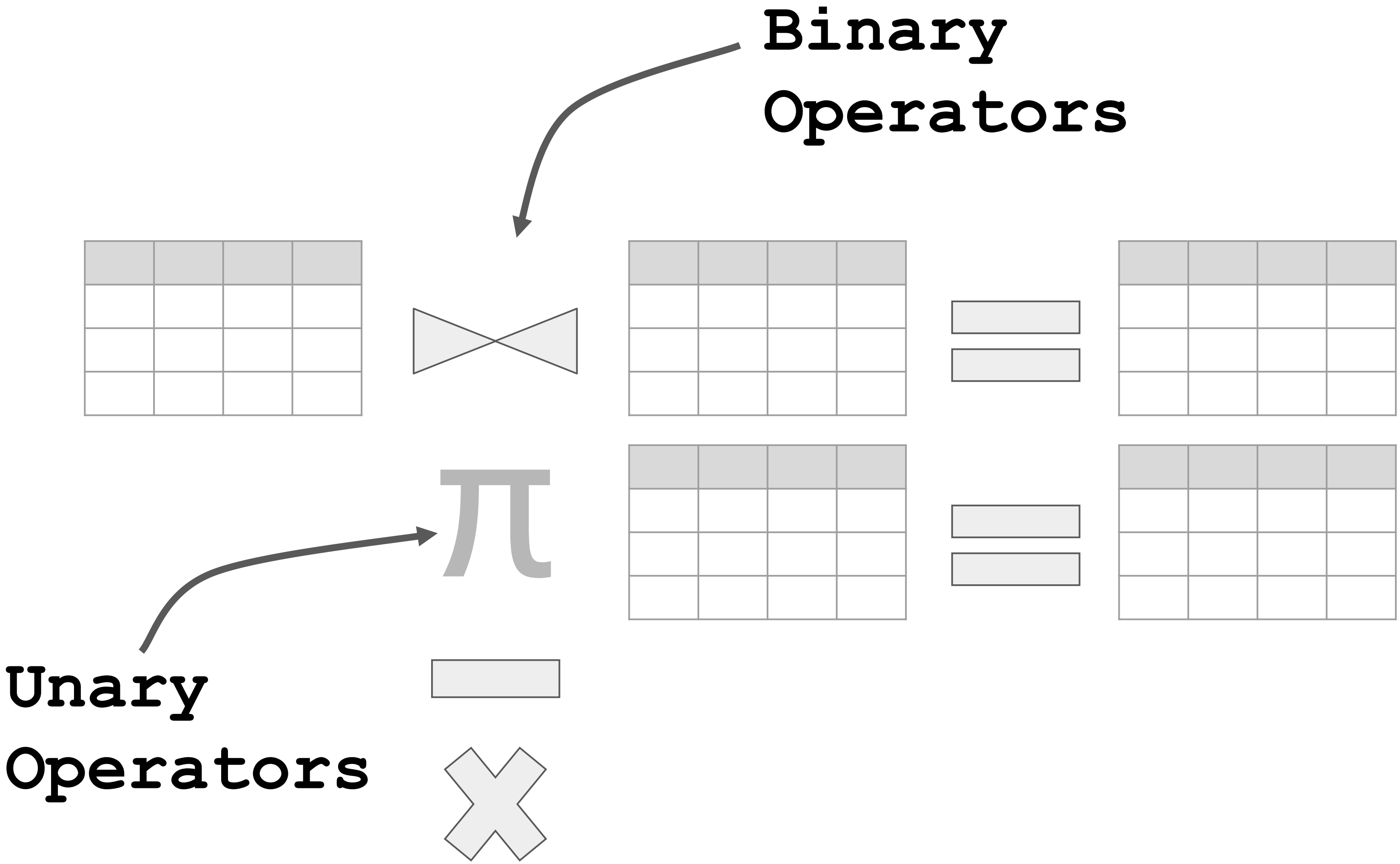
×

÷

Relational Algebra



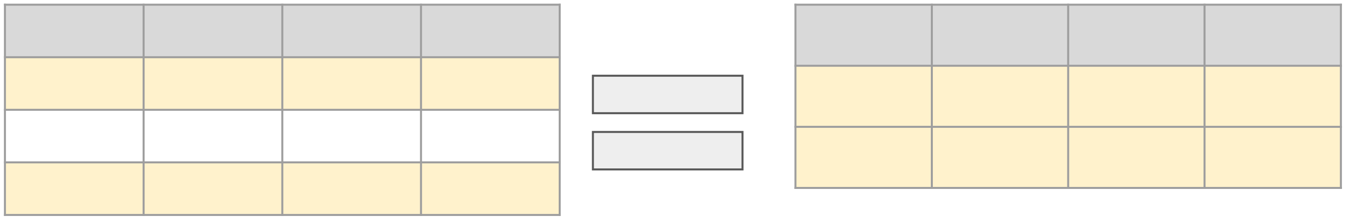
Relational Algebra



Operator σ

Unary Operators - Sigma (σ) - Selection

$\sigma_{\text{Condition}}$



Unary Operators - Sigma (σ) - Selection

$\sigma_{\text{salary>=1000}}$

Employee

id	name	email	salary
1	Alex	a@c	1000
2	John	j@c	600
3	Mo	m@c	1200

=

Employee

id	name	email	salary
1	Alex	a@c	1000
3	Mo	m@c	1200

Get employees whose salary is greater than or equal to $1000?

Unary Operators - Sigma (σ) - Selection

$\sigma_{salary>=1000}$

Employee

id	name	email	salary
1	Alex	a@c	1000
2	John	j@c	600
3	Mo	m@c	1200

=

Employee

id	name	email	salary
1	Alex	a@c	1000
3	Mo	m@c	1200

Get employees whose salary is greater than or equal to $1000?

$\sigma_{Salary>=1000}$ (Employee)

RA

Relational Algebra - Unary Operators: σ π ρ

Unary Operators - Sigma (σ) - Expressions

σ σ

Employee

id	name	email	salary
1	Alex	a@c	1000
2	John	j@c	600
3	Mo	m@c	1200

=

Employee

id	name	email	salary
1	Alex	a@c	1000
3	Mo	m@c	1200

⇩

Employee

id	name	email	salary
1	Alex	a@c	1000

$$\sigma_{name=Alex}\ (\sigma_{Salary>=1000}\ (Employee))$$

Unary Operators - Sigma (σ) - Expressions

σ σ

Employee

id	name	email	salary
1	Alex	a@c	1000
2	John	j@c	600
3	Mo	m@c	1200

=

Employee

id	name	email	salary
1	Alex	a@c	1000

Commutative

$$\sigma_{\text{Salary>=1000}}\ (\sigma_{\text{name=Alex}}\ \text{Employee}))$$

$$\sigma_{\text{name=Alex}}\ (\sigma_{\text{Salary>=1000}}\ \text{Employee}))$$

The Same

Unary Operators - Sigma (σ) - Expressions

σ σ

Employee

id	name	email	salary
1	Alex	a@c	1000
2	John	j@c	600
3	Mo	m@c	1200

Employee

id	name	email	salary
1	Alex	a@c	1000

and(∧), or(V), not(¬)

$\sigma_{Salary>=1000}\ (\sigma_{name=Alex}\ Employee))$

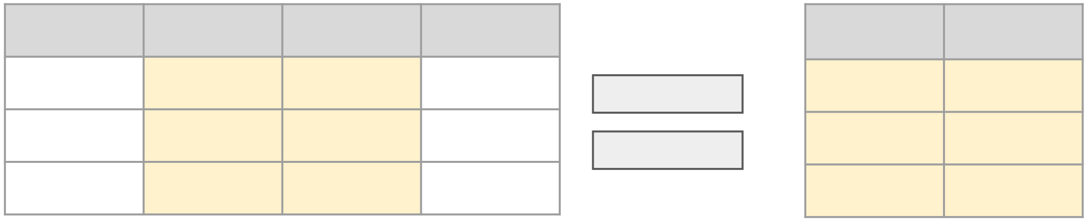
$\sigma_{name=Alex\ \wedge\ Salary>=1000}\ (Employee)$

The Same

Operator π

Unary Operators - Pi (π)- Projection

π_{Columns}



Unary Operators - Pi (π)- Projection

$\pi_{\text{name, email}}$

Employee

id	name	email	salary
1	Alex	a@c	1000
2	John	j@c	600
3	Mo	m@c	1200

=

Employee

name	email
Alex	a@c
John	j@c
Mo	m@c

Get the name and email of all employees?

Unary Operators - Pi (π)- Projection

$\pi_{\text{name, email}}$

Employee

id	name	email	salary
1	Alex	a@c	1000
2	John	j@c	600
3	Mo	m@c	1200

=

Employee

name	email
Alex	a@c
John	j@c
Mo	m@c

Get the name and email of all employees?

$\pi_{\text{name, email}}$ (Employee)

Unary Operators - Pi (π) - Expressions

π π

Employee

id	name	email	salary
1	Alex	a@c	1000
2	John	j@c	600
3	Mo	m@c	1200

=

Employee

name
Alex
John
Mo

π_{name} $\pi_{name,\ email}$ (Employee)

Unary Operators - Pi (π) - Expressions

ππ

Employee

id	name	email	salary
1	Alex	a@c	1000
2	John	j@c	600
3	Mo	m@c	1200

=

Employee

name
Alex
John
Mo

$\pi_{name}\ \pi_{name,\ email}$ (Employee)

π_{name} (Employee)

The Same

Unary Operators - Pi (π) - Expressions

ππ

Employee

id	name	email	salary
1	Alex	a@c	1000
2	John	j@c	600
3	Mo	m@c	1200

=

Employee

name
Alex
John
Mo

NOT Commutative

$$\Pi_{name}\ (\Pi_{name,\ email}\ (Employee))$$

$$\Pi_{name,\ email}\ (\Pi_{name}\ (Employee))$$

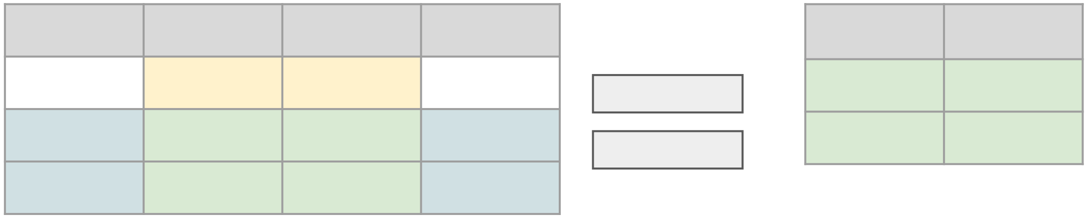
NOT
The
Same

Operator

σ and π

Unary Operators - Sigma (σ) and Pi (π)

πσ



Unary Operators - Sigma (σ) and Pi (π)

πσ

Employee

id	name	email	salary
1	Alex	a@c	1000
2	John	j@c	600
3	Mo	m@c	1200

=

Employee

name	email
John	j@c
Mo	m@c

Get the name and email of employees whose id greater than or equal to 2?

Unary Operators - Sigma (σ) and Pi (π)

πσ

Employee

id	name	email	salary
1	Alex	a@c	1000
2	John	j@c	600
3	Mo	m@c	1200

=

Employee

name	email
John	j@c
Mo	m@c

Get the name and email of employees whose id greater than or equal to 2?

$$\pi_{name,\ email}\ (\sigma_{id>=2}\ (Employee))$$

Unary Operators - Sigma (σ) and Pi (π)

πσ

Employee

id	name	email	salary
1	Alex	a@c	1000
2	John	j@c	600
3	Mo	m@c	1200

=

Employee

name	email
John	j@c
Mo	m@c

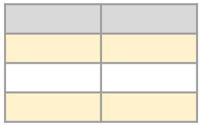
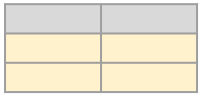
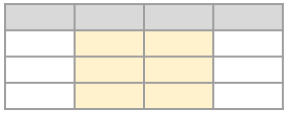
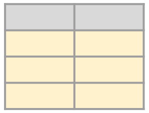
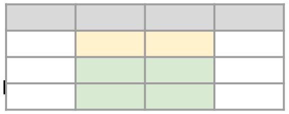
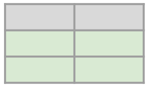
Get the name and email of employees whose id greater than or equal to 2?

$$\Pi_{name,\ mail}\ (\sigma_{id>=2}\ (Employee))$$

$$\sigma_{id>=2}\ (\Pi_{name,\ email}\ (Employee))$$

Can we change the order???

Relational Algebra

Unary Operators		Binary Operators	
$\sigma_{\text{Condition}}$ 			
π_{Columns} 			
$\pi_{\text{Columns}}\sigma_{\text{Conditio}}$ 			

Operator

× and ⋈

RA

Relational Algebra - Binary Operators: × ⋈ ∪ ∩ -

Binary Operators - Cartesian Product (×)

Employee

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

×

Department

name	budget
Finance	20,000
Sales	30,000
HR	25,000

=

id	name	email	Dept	name	budget
1	Alex	a@c	Sales	Finance	20,000
1	Alex	a@c	Sales	Sales	30,000
1	Alex	a@c	Sales	HR	25,000
2	John	j@c	Finance	Finance	20,000
2	John	j@c	Finance	Sales	30,000
2	John	j@c	Finance	HR	25,000
3	Mo	m@c	HR	Finance	20,000
3	Mo	m@c	HR	Sales	30,000
3	Mo	m@c	HR	HR	25,000

Binary Operators - Cartesian Product (×) with Sigma (σ)

Employee

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

×

Department

name	budget
Finance	20,000
Sales	30,000
HR	25,000

=

id	name	email	Dept	name	budget
1	Alex	a@c	Sales	Finance	20,000
1	**Alex**	**a@c**	**Sales**	**Sales**	**30,000**
1	Alex	a@c	Sales	HR	25,000
2	**John**	**j@c**	**Finance**	**Finance**	**20,000**
2	John	j@c	Finance	Sales	30,000
2	John	j@c	Finance	HR	25,000
3	Mo	m@c	HR	Finance	20,000
3	Mo	m@c	HR	Sales	30,000
3	**Mo**	**m@c**	**HR**	**HR**	**25,000**

Get the employees with the budget of their departments?

$$\sigma_{\text{Employee.Dept=Department.name}}\left(\text{Employee} \times \text{Department}\right)$$

Binary Operators - Inner Join (⋈) = (× with σ)

Employee

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

⋈

Department

name	budget
Finance	20,000
Sales	30,000
HR	25,000

=

id	name	email	Dept	name	budget
1	Alex	a@c	Sales	Finance	20,000
1	**Alex**	**a@c**	**Sales**	**Sales**	**30,000**
1	Alex	a@c	Sales	HR	25,000
2	**John**	**j@c**	**Finance**	**Finance**	**20,000**
2	John	j@c	Finance	Sales	30,000
2	John	j@c	Finance	HR	25,000
3	Mo	m@c	HR	Finance	20,000
3	Mo	m@c	HR	Sales	30,000
3	**Mo**	**m@c**	**HR**	**HR**	**25,000**

Get the employees with the budget of their departments?

$$\sigma_{\text{Employee.Dept=Department.name}}\left(\text{Employee} \times \text{Department}\right)$$

$$\text{Employee} \bowtie_{\text{Employee.Dept=Department.name}} \text{Department}$$

The Same

Binary Operators - Natural Join (⋈) = (× with σ)

Employee

id	name	email	dname
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

⋈

Department

dname	budget
Finance	20,000
Sales	30,000
HR	25,000

=

id	name	email	Dept	name	budget
1	Alex	a@c	Sales	Finance	20,000
1	**Alex**	**a@c**	**Sales**	**Sales**	**30,000**
1	Alex	a@c	Sales	HR	25,000
2	**John**	**j@c**	**Finance**	**Finance**	**20,000**
2	John	j@c	Finance	Sales	30,000
2	John	j@c	Finance	HR	25,000
3	Mo	m@c	HR	Finance	20,000
3	Mo	m@c	HR	Sales	30,000
3	**Mo**	**m@c**	**HR**	**HR**	**25,000**

Get the employees with the budget of their departments?

$$\sigma_{Employee.dname=Department.dname}\left(Employee \times Department\right)$$

$$Employee \bowtie Department$$

The

Same

Binary Operators - Inner Join (⋈)

No Match for some tuples

Employee

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

⋈

Department

name	budget
Finance	20,000
Sales	30,000
IT	40,000

=

id	name	email	Dept	name	budget
1	Alex	a@c	Sales	Sales	30,000
2	John	j@c	Finance	Finance	20,000

Get the employees with the budget of their departments?

$$\text{Employee} \bowtie_{\text{Employee.Dept=Department.name}} \text{Department}$$

RA

Relational Algebra - Binary Operators: × ⋈ ∪ ∩ -

Binary Operators - Outer Join

- Left Outer Join (⟕)
- Right Outer Join (⟖)
- Full Outer Join (⟗)

No Match for some tuples

Employee

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

⋈

Department

name	budget
Finance	20,000
Sales	30,000
IT	40,000

=

id	name	email	Dept	name	budget
1	Alex	a@c	Sales	Sales	30,000
2	John	j@c	Finance	Finance	20,000

Get the employees with the budget of their departments?

$$\text{Employee} \bowtie_{\text{Employee.Dept=Department.name}} \text{Department}$$

RA Relational Algebra - Binary Operators: × ⋈ ∪ ∩ −

Binary Operators - Outer Join

- Left Outer Join (⟕)
- Right Outer Join (⟖)
- Full Outer Join (⟗)

No Match for some tuples

Employee

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

⟕

Department

name	budget
Finance	20,000
Sales	30,000
IT	40,000

=

id	name	email	Dept	name	budget
1	Alex	a@c	Sales	Sales	30,000
2	John	j@c	Finance	Finance	20,000
3	Mo	m@c	HR	NULL	NULL

Get the employees with the budget of their departments?

$$\text{Employee} \;⟕_{\text{Employee.Dept=Department.name}}\; \text{Department}$$

Binary Operators - Outer Join

- Left Outer Join (⟕)
- **Right Outer Join (⟖)**
- Full Outer Join (⟗)

No Match for some tuples

Employee

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

⟖

Department

name	budget
Finance	20,000
Sales	30,000
IT	40,000

=

id	name	email	Dept	name	budget
1	Alex	a@c	Sales	Sales	30,000
2	John	j@c	Finance	Finance	20,000
NULL	NULL	NULL	NULL	IT	40,000

Get the employees with the budget of their departments?

Employee ⟖$_{\text{Employee.Dept=Department.name}}$ Department

Binary Operators - Outer Join

- Left Outer Join (⟕)
- Right Outer Join (⟖)
- Full Outer Join (⟗)

No Match for some tuples

Employee

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

⟗

Department

name	budget
Finance	20,000
Sales	30,000
IT	40,000

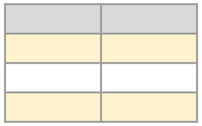
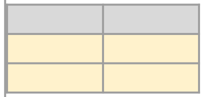
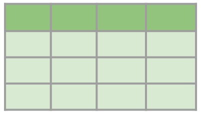
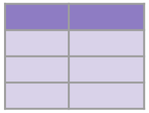
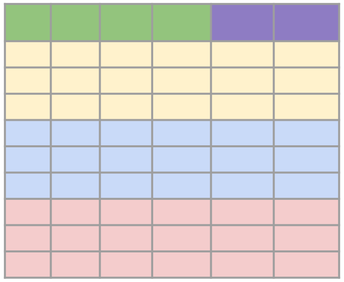
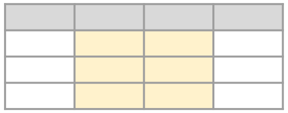
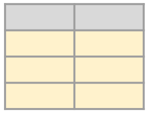
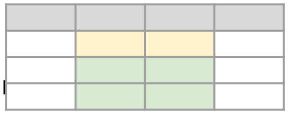
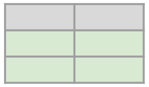
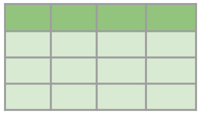
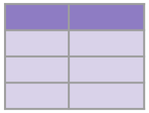
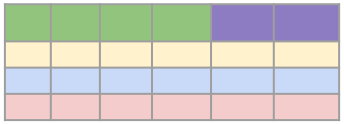
=

id	name	email	Dept	name	budget
1	Alex	a@c	Sales	Sales	30,000
2	John	j@c	Finance	Finance	20,000
3	Mo	m@c	HR	NULL	NULL
NULL	NULL	NULL	NULL	IT	40,000

Get the employees with the budget of their departments?

$$\text{Employee} ⟗_{\text{Employee.Dept=Department.name}} \text{Department}$$

Relational Algebra

Unary Operators		Binary Operators	
$\sigma_{Condition}$ 		 × 	
$\pi_{Columns}$ 			
$\pi_{Columns}\sigma_{Condition}$ 		 ⋈ 	
		⟕ ⟖ ⟗	

Set Operators

∪ ∩ -

RA Relational Algebra - Binary Operators: × ⋈ ∪ ∩ -

Binary Operators - Set Operators

- Intersection (∩)
- Union (∪)
- Minus (-)

Must be compatible

Employee

name	email	Dept
Alex	a@c	Sales
John	j@c	Finance
Mo	m@c	HR

3

∩

GradeStudent

name	email	Depar
Alex	a@c	Sales
Max	x@c	Finance
Go	g@c	HR

3

=

Same data type

Same data type

Same data type

Binary Operators - Set Operators

- Intersection (∩)
- Union (∪)
- Minus (-)

Employee

name	email	Dept
Alex	a@c	Sales
John	j@c	Finance
Mo	m@c	HR

∩

GradeStudent

name	email	Depar
Alex	a@c	Sales
Max	x@c	Finance
Go	g@c	HR

=

name	email	Dept
Alex	a@c	Sales

Get the employees which are graduate student at the same time?

Employee ∩ GradeStudent

Binary Operators - Set Operators

- Intersection (∩)
- Union (∪)
- Minus (-)

Compatibility problem

Employee

name	email	Dept	Salary
Alex	a@c	Sales	1000
John	j@c	Finance	1200
Mo	m@c	HR	1500

∩

GradeStudent

name	email	Depar
Alex	a@c	Sales
Max	x@c	Finance
Go	g@c	HR

=

name	email	Dept
Alex	a@c	Sales

Get the employees which are graduate student at the same time?

RA Relational Algebra - Binary Operators: × ⋈ ∪ ∩ -

Binary Operators - Set Operators

- Intersection (∩)
- Union (∪)
- Minus (-)

Compatibility problem

Employee

name	email	Dept	Salary
Alex	a@c	Sales	1000
John	j@c	Finance	1200
Mo	m@c	HR	1500

∩

GradeStudent

name	email	Depar
Alex	a@c	Sales
Max	x@c	Finance
Go	g@c	HR

=

name	email	Dept
Alex	a@c	Sales

Get the employees which are graduate student at the same time?

$$(\Pi_{name,email,Dept}\ Employee) \cap (GradeStudent)$$

Relational Algebra - Binary Operators: × ⋈ ∪ ∩ -

Binary Operators - Set Operators

- Intersection (∩)
- **Union (∪)**
- Minus (-)

Employee

name	email	Dept
Alex	a@c	Sales
John	j@c	Finance
Mo	m@c	HR

∪

GradeStudent

name	email	Depar
Alex	a@c	Sales
Max	x@c	Finance
Go	g@c	HR

=

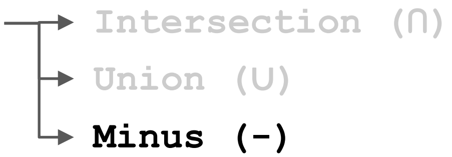
name	email	Dept
Alex	a@c	Sales
John	j@c	Finance
Mo	m@c	HR
Max	x@c	Finance
Go	g@c	HR

Get all employees and graduate students?

Employee ∪ GradeStudent

RA Relational Algebra - Binary Operators: × ⋈ ∪ ∩ -

Binary Operators - Set Operators



Intersection (∩)

Union (∪)

Minus (-)

Employee

name	email	Dept
Alex	a@c	Sales
John	j@c	Finance
Mo	m@c	HR

-

GradeStudent

name	email	Depar
Alex	a@c	Sales
Max	x@c	Finance
Go	g@c	HR

=

name	email	Dept
John	j@c	Finance
Mo	m@c	HR

Get all employees which are NOT graduate students?

Employee - GradeStudent

Binary Operators - Set Operators

- Intersection (∩)
- Union (∪)
- Minus (-)

Employee

name	email	Dept
Alex	a@c	Sales
John	j@c	Finance
Mo	m@c	HR

−

GradeStudent

name	email	Depar
Alex	a@c	Sales
Max	x@c	Finance
Go	g@c	HR

=

name	email	Dept
John	j@c	Finance
Mo	m@c	HR

Get all employees which are **NOT** graduate students?

Employee - GradeStudent ≠ GradeStudent - Employee

Operator

/ or ÷

Binary Operators - Divide by (/)

Employee

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

×

Department

name	budget
Finance	20,000
Sales	30,000
HR	25,000

=

Emp_Dep

id	name	email	Dept	name	budget
1	Alex	a@c	Sales	Finance	20,000
1	Alex	a@c	Sales	Sales	30,000
1	Alex	a@c	Sales	HR	25,000
2	John	j@c	Finance	Finance	20,000
2	John	j@c	Finance	Sales	30,000
2	John	j@c	Finance	HR	25,000
3	Mo	m@c	HR	Finance	20,000
3	Mo	m@c	HR	Sales	30,000
3	Mo	m@c	HR	HR	25,000

Binary Operators - Divide by (/)

Employee

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

×

Department

name	budget
Finance	20,000
Sales	30,000
HR	25,000

=

Emp_Dep

id	name	email	Dept	name	budget
1	Alex	a@c	Sales	Finance	20,000
1	Alex	a@c	Sales	Sales	30,000
1	Alex	a@c	Sales	HR	25,000
2	John	j@c	Finance	Finance	20,000
2	John	j@c	Finance	Sales	30,000
2	John	j@c	Finance	HR	25,000
3	Mo	m@c	HR	Finance	20,000
3	Mo	m@c	HR	Sales	30,000
3	Mo	m@c	HR	HR	25,000

Emp_Dep

id	name	email	Dept	name	budget
1	Alex	a@c	Sales	Finance	20,000
1	Alex	a@c	Sales	Sales	30,000
1	Alex	a@c	Sales	HR	25,000
2	John	j@c	Finance	Finance	20,000
2	John	j@c	Finance	Sales	30,000
2	John	j@c	Finance	HR	25,000
3	Mo	m@c	HR	Finance	20,000
3	Mo	m@c	HR	Sales	30,000
3	Mo	m@c	HR	HR	25,000

/

Department

name	budget
Finance	20,000
Sales	30,000
HR	25,000

=

Employee

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

Binary Operators - Divide by (/)

Stud_Course

sid	cname	mark
1	Math	3
1	Pythics	2
1	Network	3
2	Math	3
2	Pythics	2
2	Network	3
3	Network	3

Student

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

Course

cname	Hours
Math	3
Pythics	2
Network	3

=

Student

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance

Get students who studied **ALL** courses?

$$(\Pi_{sid,cname}\ \text{Stud\_Course})\ /\ (\Pi_{cname}\ \text{Course})$$

Binary Operators - Divide by (/)

Stud_Course

sid	cname	mark
1	Math	3
1	Pythics	2
1	Network	3
2	Math	3
2	Pythics	2
2	Network	3
3	Network	3

/

Course

cname	Hours
Math	3
Pythics	2
Network	3

=

Student

id
1
2

Student

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

Get students who studied **ALL** courses?

$$(\Pi_{sid,cname}\ Stud\_Course)\ /\ (\Pi_{cname}\ Course)$$

Binary Operators - Divide by (/)

Stud_Course

sid	cname	mark
1	Math	3
1	Pythics	2
1	Network	3
2	Math	3
2	Pythics	2
2	Network	3
3	Network	3

/

Course

cname	Hours
Math	3
Pythics	2
Network	3

=

Student

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance

Student

id	name	email	Dept
1	Alex	a@c	Sales
2	John	j@c	Finance
3	Mo	m@c	HR

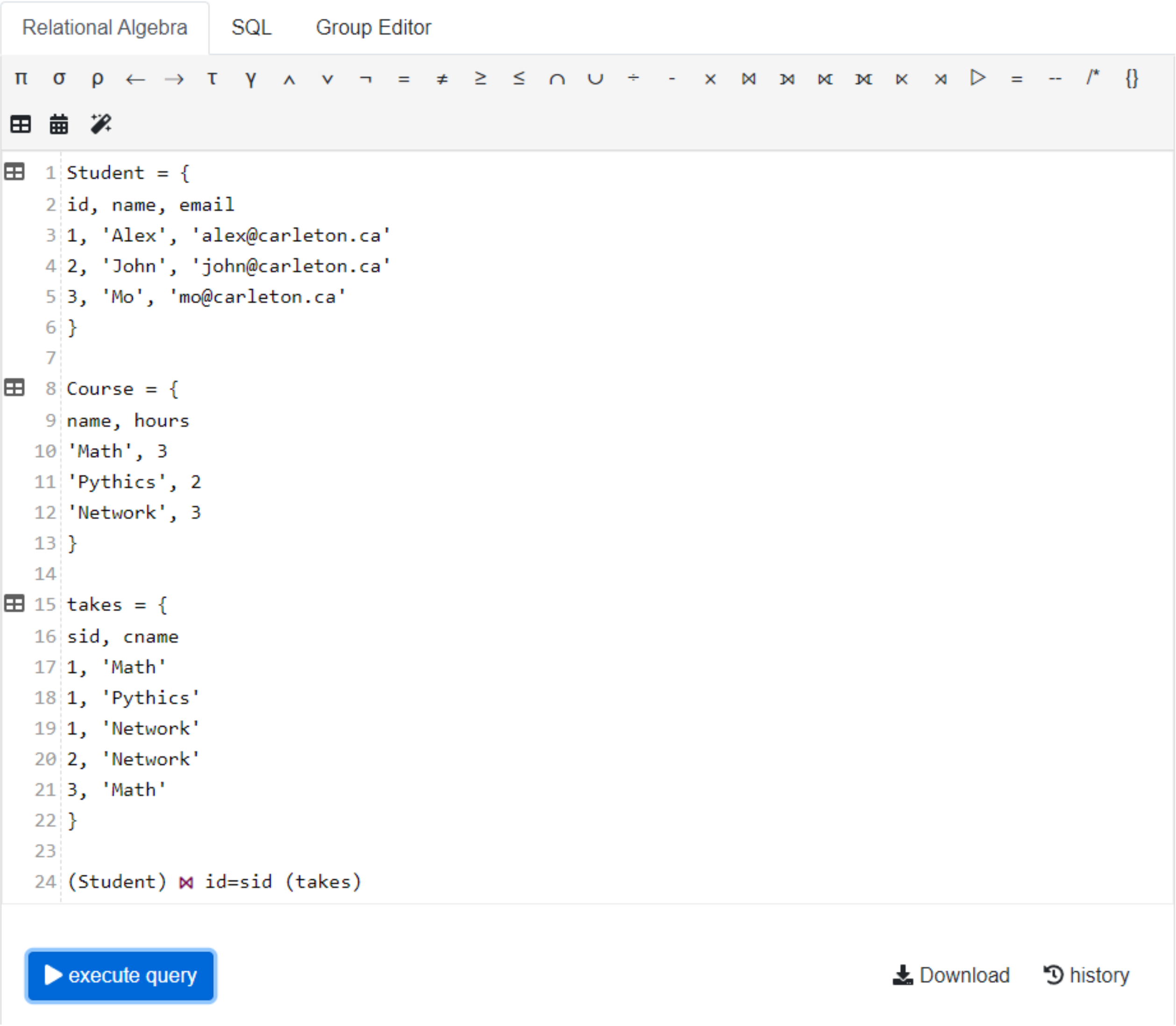
Get students who studied **ALL** courses?

$$\left(\left(\Pi_{sid,cname}\ Stud\_Course\right)\ /\ \left(\Pi_{cname}\ Course\right)\right)$$

$$\bowtie (Student)$$

RA RelaX

Try it https://dbis-uibk.github.io/relax/calc/local/uibk/local/0


Relational Algebra
SQL
Group Editor
π σ ρ ← → τ γ ∧ ∨ ¬ = ≠ ≥ ≤ ∩ ∪ ÷ - × ⋈ ⟕ ⟖ ⟗ ⋉ ⋊ ▷ = -- /* {}
execute query
Download
history

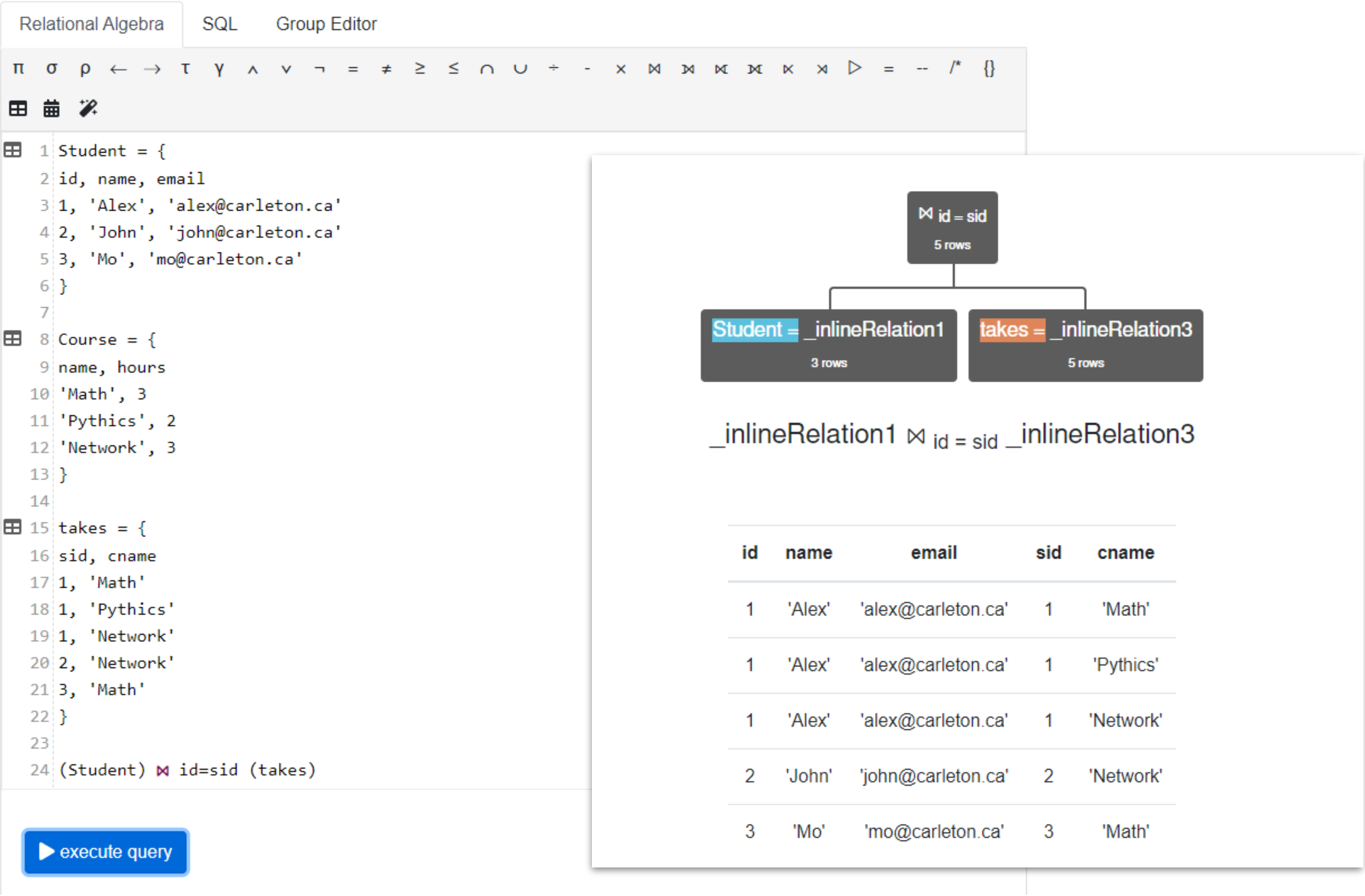
```
Student = {
id, name, email
1, 'Alex', 'alex@carleton.ca'
2, 'John', 'john@carleton.ca'
3, 'Mo', 'mo@carleton.ca'
}

Course = {
name, hours
'Math', 3
'Pythics', 2
'Network', 3
}

takes = {
sid, cname
1, 'Math'
1, 'Pythics'
1, 'Network'
2, 'Network'
3, 'Math'
}

(Student) ⋈ id=sid (takes)
```

Copy code from here

```
Student = {
id, name, email
1, 'Alex', 'alex@carleton.ca'
2, 'John', 'john@carleton.ca'
3, 'Mo', 'mo@carleton.ca'
}

Course = {
name, hours
'Math', 3
'Pythics', 2
'Network', 3
}

takes = {
sid, cname
1, 'Math'
1, 'Pythics'
1, 'Network'
2, 'Network'
3, 'Math'
}

(Student) ⋈ id=sid (takes)
```

RelaX

Try it https://dbis-uibk.github.io/relax/calc/local/uibk/local/0



```
Student = {
id, name, email
1, 'Alex', 'alex@carleton.ca'
2, 'John', 'john@carleton.ca'
3, 'Mo', 'mo@carleton.ca'
}

Course = {
name, hours
'Math', 3
'Pythics', 2
'Network', 3
}

takes = {
sid, cname
1, 'Math'
1, 'Pythics'
1, 'Network'
2, 'Network'
3, 'Math'
}

(Student) ⋈ id=sid (takes)
```

_inlineRelation1 ⋈ $_{\text{id = sid}}$ _inlineRelation3

id	name	email	sid	cname
1	'Alex'	'alex@carleton.ca'	1	'Math'
1	'Alex'	'alex@carleton.ca'	1	'Pythics'
1	'Alex'	'alex@carleton.ca'	1	'Network'
2	'John'	'john@carleton.ca'	2	'Network'
3	'Mo'	'mo@carleton.ca'	3	'Math'

Examples

Examples

Student

id	name	email	city
1	Alex	al@c.ca	Ottawa
2	John	jo@c.ca	Ottawa
3	Makela	ma@c.ca	Toronto

Takes

sid	cid	mark
1	1	9
1	2	10
1	3	8
2	1	8

Course

id	title	hours
1	Math	0.5
2	Physics	0.5
3	DBMS	0.5

Example: List all the students (name and email only) that live in Ottawa

Student

name	email
Alex	al@c.ca
John	jo@c.ca

EX **Examples**

Student

id	name	email	city
1	Alex	al@c.ca	Ottawa
2	John	jo@c.ca	Ottawa
3	Makela	ma@c.ca	Toronto

Takes

sid	cid	mark
1	1	9
1	2	10
1	3	8
2	1	8

Course

id	title	hours
1	Math	0.5
2	Physics	0.5
3	DBMS	0.5

Example: List all the students (name and email only) that live in Ottawa

Student

name	email
Alex	al@c.ca
John	jo@c.ca

$$\Pi_{\text{name, email}} \left(\sigma_{\text{city}='\text{Ottawa}'} \ (\text{Student}) \right)$$

Examples

Student

id	name	email	city
1	Alex	al@c.ca	Ottawa
2	John	jo@c.ca	Ottawa
3	Makela	ma@c.ca	Toronto

Takes

sid	cid	mark
1	1	9
1	2	10
1	3	8
2	1	8

Course

id	title	hours
1	Math	0.5
2	Physics	0.5
3	DBMS	0.5

Example: Return Makela’s email

Student

email
ma@c.ca

EX Examples

Student

id	name	email	city
1	Alex	al@c.ca	Ottawa
2	John	jo@c.ca	Ottawa
3	Makela	ma@c.ca	Toronto

Takes

sid	cid	mark
1	1	9
1	2	10
1	3	8
2	1	8

Course

id	title	hours
1	Math	0.5
2	Physics	0.5
3	DBMS	0.5

Example: Return Makela's email

Student

email
ma@c.ca

$$\pi_{\text{email}}\left(\sigma_{\text{name='Makela'}}\ (\text{Student})\right)$$

Examples

Student

id	name	email	city
1	Alex	al@c.ca	Ottawa
2	John	jo@c.ca	Ottawa
3	Makela	ma@c.ca	Toronto

Takes

sid	cid	mark
1	1	9
1	2	10
1	3	8
2	1	8

Course

id	title	hours
1	Math	0.5
2	Physics	0.5
3	DBMS	0.5

Example: Return the students' names, their courses' titles, and their marks in these courses.

Student

name	title	mark
Alex	Math	9
Alex	Physics	10
Alex	DBMS	8
John	Math	8

EX Examples

Student

id	name	email	city
1	Alex	al@c.ca	Ottawa
2	John	jo@c.ca	Ottawa
3	Makela	ma@c.ca	Toronto

Takes

sid	cid	mark
1	1	9
1	2	10
1	3	8
2	1	8

Course

id	title	hours
1	Math	0.5
2	Physics	0.5
3	DBMS	0.5

Example: Return the students' names, their courses' titles, and their marks in these courses.

Student

name	title	mark
Alex	Math	9
Alex	Physics	10
Alex	DBMS	8
John	Math	8

$$\Pi_{\texttt{name, title, mark}} \Big((\texttt{Student} \bowtie_{\texttt{Student.id=Takes.sid}} \texttt{Takes}) \bowtie_{\texttt{cid=Course.id}} \texttt{Course} \Big)$$

Examples

Student

id	name	email	city
1	Alex	al@c.ca	Ottawa
2	John	jo@c.ca	Ottawa
3	Makela	ma@c.ca	Toronto

Takes

sid	cid	mark
1	1	9
1	2	10
1	3	8
2	1	8

Course

id	title	hours
1	Math	0.5
2	Physics	0.5
3	DBMS	0.5

Example: Students' names that take courses.

Student

name
Alex
John

EX

Examples

Student

id	name	email	city
1	Alex	al@c.ca	Ottawa
2	John	jo@c.ca	Ottawa
3	Makela	ma@c.ca	Toronto

Takes

sid	cid	mark
1	1	9
1	2	10
1	3	8
2	1	8

Course

id	title	hours
1	Math	0.5
2	Physics	0.5
3	DBMS	0.5

Example: Students' names that take courses.

Student

name
Alex
John

$$\pi_{name}\left((Student \bowtie_{Student.id=Takes.sid} Takes) \right)$$

Note: In Relation Algebra, we use Sets which do not allow duplicates in the result. For example, if the output is {Alex, Alex, Alex, John}, this will be {Alex, John} only.

EX Examples

Student

id	name	email	city
1	Alex	al@c.ca	Ottawa
2	John	jo@c.ca	Ottawa
3	Makela	ma@c.ca	Toronto

Takes

sid	cid	mark
1	1	9
1	2	10
1	3	8
2	1	8

Course

id	title	hours
1	Math	0.5
2	Physics	0.5
3	DBMS	0.5

Example: Students' names that are **NOT** take courses.

Student

name
Alex
John
Makela

−

name
Alex
John

=

name
Makela

$$\pi_{name}\ (\text{Student})\ -$$

$$\boxed{\pi_{name}\ \left((\text{Student}\ \bowtie_{\text{Student.id=Takes.sid}}\ \text{Takes}) \right)}$$