CMSC 461, Database Management Systems
Spring 2018

Chapter 6 – Formal Relational Query Languages

These slides are based on "Database System Concepts" book and slides, 6th edition, and the 2009/2012 CMSC 461 slides by Dr. Kalpakis

Jennifer Sleeman

https://www.csee.umbc.edu/~jsleem1/courses/461/spr18

Logistics

- Homework 1 due Wednesday 2/7/2018
- Dr. Sleeman out on Wednesday
  - Class will still meet, guest lecturer
- Project is posted, we will review today
- Phase 1 of project is due 2/14/2018

Lecture Outline

- Intro to Relational Algebra
- Fundamental Operations
- Additional Operations
- Summary
- In Class Exercise

Lecture Outline

- ***Intro to Relational Algebra***
- Fundamental Operations
- Additional Operations
- Summary
- In Class Exercise

Relational Algebra

- A procedural query language based on the mathematical theory of sets that is the foundation of commercial DBMS query languages
- The operations typically take one or two relations as inputs and give a new relation as a result
- Can build expressions using multiple relational operations

Relational Algebra

- What is the difference between a procedural language and a non-procedural language?

Relational Algebra

- Procedural languages tell you how to process a query (a sequence of steps provide the how)
- Non-Procedural or declarative languages tell you what to process but not how to process

Relational Algebra

- Six basic operators
  - select: σ
  - project: $\prod$
  - union: $\cup$
  - set difference: –
  - Cartesian product: x
  - rename: ρ

Lecture Outline

- Intro to Relational Algebra
- **Fundamental Operations**
- Additional Operations
- Summary
- In Class Exercise

Select Operation

$\sigma_p(\boldsymbol{r}) = \{t \mid t \in r$ **and** $p(t)\}$

Where *p* is the **selection predicate**, a formula in propositional calculus consisting of **terms** connected by logical operators ∧ (**and**), ∨ (**or**), ¬ (**not**)

Each **term** is one of:

<attribute> *op* <attribute>

<attribute> *op* <constant>

where *op* is one of: $=\ \neq\ >\ \geq\ <\ \leq$

Select Operation

Instructor.dept_name = Department.dept_name (Simple pred)
Instructor.dept_name='Finance' (Simple pred)
Instructor.dept_name = Department.dept_name or Instructor.Name = 'Wu' (Boolean Combination pred)
Instructor.dept_name = Department.dept_name and Instructor.Name = 'Wu' (Boolean Combination pred)
Not Instructor.Name = 'Wu' (Boolean Combination pred)

Select Operation

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

r

$\sigma_{A=B \wedge D>5}(r)$

Select Operation

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

r

A	B	C	D
α	α	1	7
β	β	23	10

$\sigma_{A=B \wedge D>5}(r)$

Example Select Operation

$$\sigma_{\textit{dept\_name}=\text{"}\textit{Physics}\text{"}}(\textit{instructor})$$

Project Operation

$$\Pi_{A_1, A_2, \ldots, A_k}(r)$$

Where A_1, A_2 are attribute names and r is a relation name.
The result is defined as the relation of k columns obtained by dropping the columns that are not listed
Duplicate rows removed from result, since relations are sets

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

r

A	C
α	1
α	1
β	1
β	2

=

A	C
α	1
β	1
β	2

$\Pi_{A,C}(r)$

Example Project Operation

To eliminate the *dept_name* attribute of *instructor*

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

Union Operation

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

For $r \cup s$ to be valid, these relations have to be **union compatible**.

- *r* and *s* must have the *same **arity*** (same number of attributes)
- the domains of the corresponding attributes must be ***compatible*** *(example: 2nd column of r deals with the same type of values as does the 2nd column of s)*

A	B
α	1
α	2
β	1

r

$\cup$

A	B
α	2
β	3

s

$=$

A	B
α	1
α	2
β	1
β	3

$r \cup s$

Example Union Operation

To find all courses taught in the Fall 2009 semester, or in the Spring 2010 semester, or in both

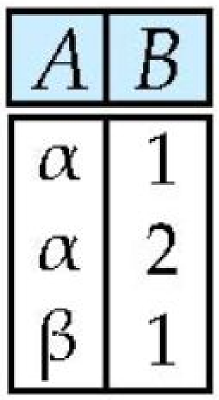
$$\Pi_{course\_id}\,(\sigma_{\ semester=\text{“Fall”}\ \land\ year=2009}\,(section))\ \cup$$
$$\Pi_{course\_id}\,(\sigma_{\ semester=\text{“Spring”}\ \land\ year=2010}\,(section))$$

Set Difference Operation

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

Set difference must be taken between compatible relations.

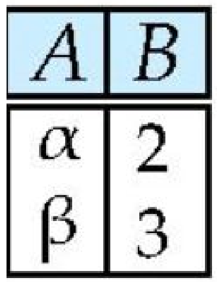
- *r* and *s* must have the same arity
- Attribute domains of r and s must be compatible



A	B
α	1
α	2
β	1

r

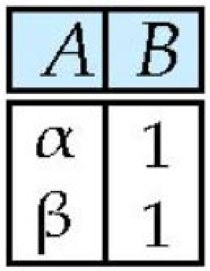
-



A	B
α	2
β	3

s

=



A	B
α	1
β	1

r – *s*

Example Set Difference Operation

To find all courses taught in the Fall 2009 semester, but not in the Spring 2010 semester

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

Cartesian-Product Operation

r x *s* = {*t q* | *t* ∈ *r* **and** *q* ∈ *s*}

Assume that attributes of r and s are disjoint. If attributes of *r* and *s* are not disjoint, then renaming must be used.

A	B
α	1
β	2

r

x

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

s

=

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

r x *s*

Example Cartesian-Product Operation

To find the names of all instructors in the Physics department together with the course_id of all courses they taught:

$\Pi_{name,course\_id} (\sigma_{instructor.ID=teaches.ID}(\sigma_{depart\_name = \text{“Physics”}}(instructor\ x\ teaches)))$

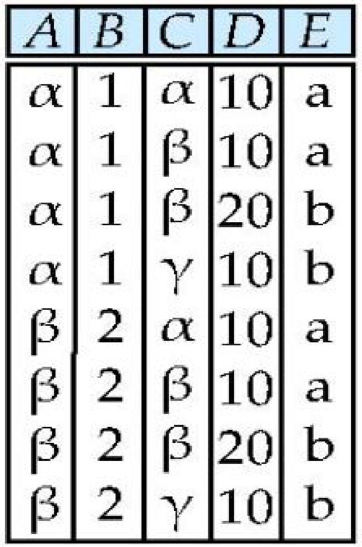
For r = instructor x teaches:
(instructor.ID, name, dept_name, salary
teaches.ID, course_id, sec_id, semester, year)

Composition of Operations

Can build expressions using multiple operations

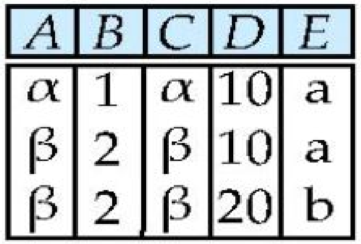
Relational-algebra expression – composition of relational-algebra operations

Example: $\sigma_{A=C}(r \times s)$



A	*B*	*C*	*D*	*E*
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

$r \times s$



A	*B*	*C*	*D*	*E*
α	1	α	10	a
β	2	β	10	a
β	2	β	20	b

$\sigma_{A=C}(r \times s)$

Rename Operation

$\rho_{X}(E)$

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 $A_1, A_2, \ldots, A_n$.

Rename Operation

- Allows us to name, and therefore to refer to, the results of relational-algebra expressions.
- Allows us to refer to a relation by more than one name.

Example Rename Operation

$\sigma_{\mathit{instructor.salary} < \mathit{d.salary}} (\text{instructor X } \rho_{d} (\mathit{instructor}))$

Using the rename operation to rename a reference to the instructor table so the relation can be referenced twice without ambiguity

Example 2 Rename Operation

$\rho_{d(InstructorID,InstructorName,InstructorDepartName,InstructorSalary)}(instructor)$

Using the rename operation to rename attributes

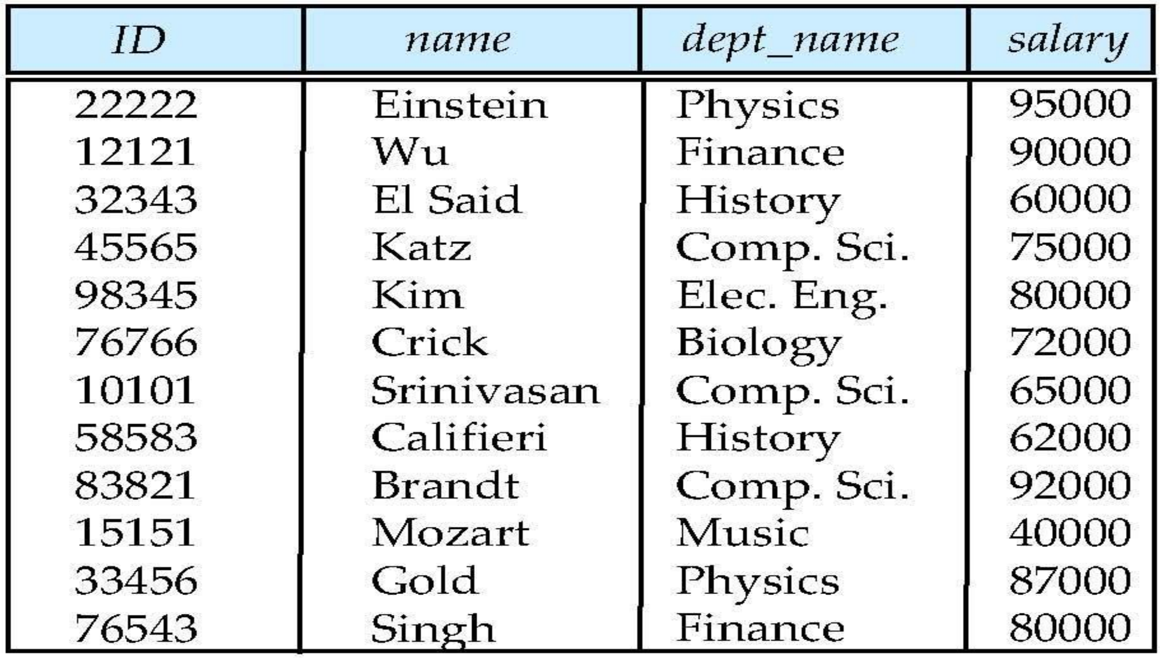
Alternative – Positional Notation

Name attributes of relation implicitly

- \$1 – first attribute, \$2 – second attribute …

Also applies to results of relational-algebra operations

Alternative – Positional Notation



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

What is the output? $\Pi_{\$4}(\sigma_{\$4 < \$8}(\textit{instructor X instructor}))$

Example Queries

Find the largest salary in the university

- Step 1: find instructor salaries that are less than some other instructor salary (i.e. not maximum)
    - using a copy of *instructor* under a new name *d*

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

- Step 2: Find the largest salary

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

Example Queries

Find the names of all instructors in the Physics department, along with the *course_id* of all courses they have taught

$$\Pi_{instructor.ID, course\_id}(\sigma_{dept\_name=\text{“Physics”}}(\sigma_{instructor.ID=teaches.ID}(instructor \times teaches)))$$

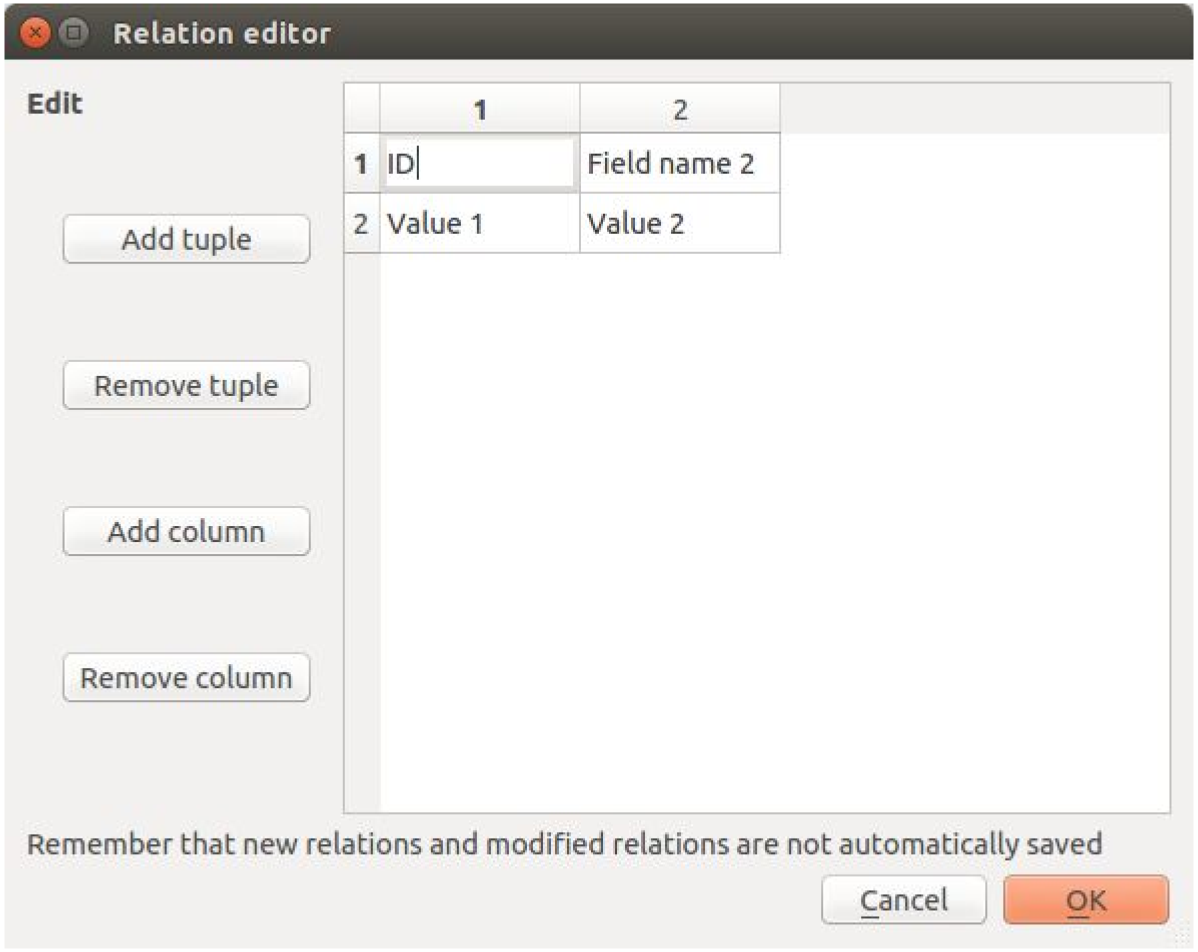
$$\Pi_{instructor.ID, course\_id}(\sigma_{instructor.ID=teaches.ID}(\sigma_{dept\_name=\text{“Physics”}}(instructor) \times teaches))$$

Experimenting with Relational Algebra - Relational

http://ltworf.github.io/relational/
On Github https://github.com/ltworf/relational/

Query := Query BinaryOp Query
Query := (Query)
Query := σ PYExprWithoutParenthesis (Query) | σ (PYExpr) (Query)
Query := π FieldList (Query)
Query := ρ RenameList (Query)
FieldList := Ident | Ident , FieldList
RenameList := Ident ➡ Ident | Ident ➡ Ident , RenameList
BinaryOp := * | - | □ | □ | ÷ | □□ | □LEFT□ | □RIGHT□ | □FULL□

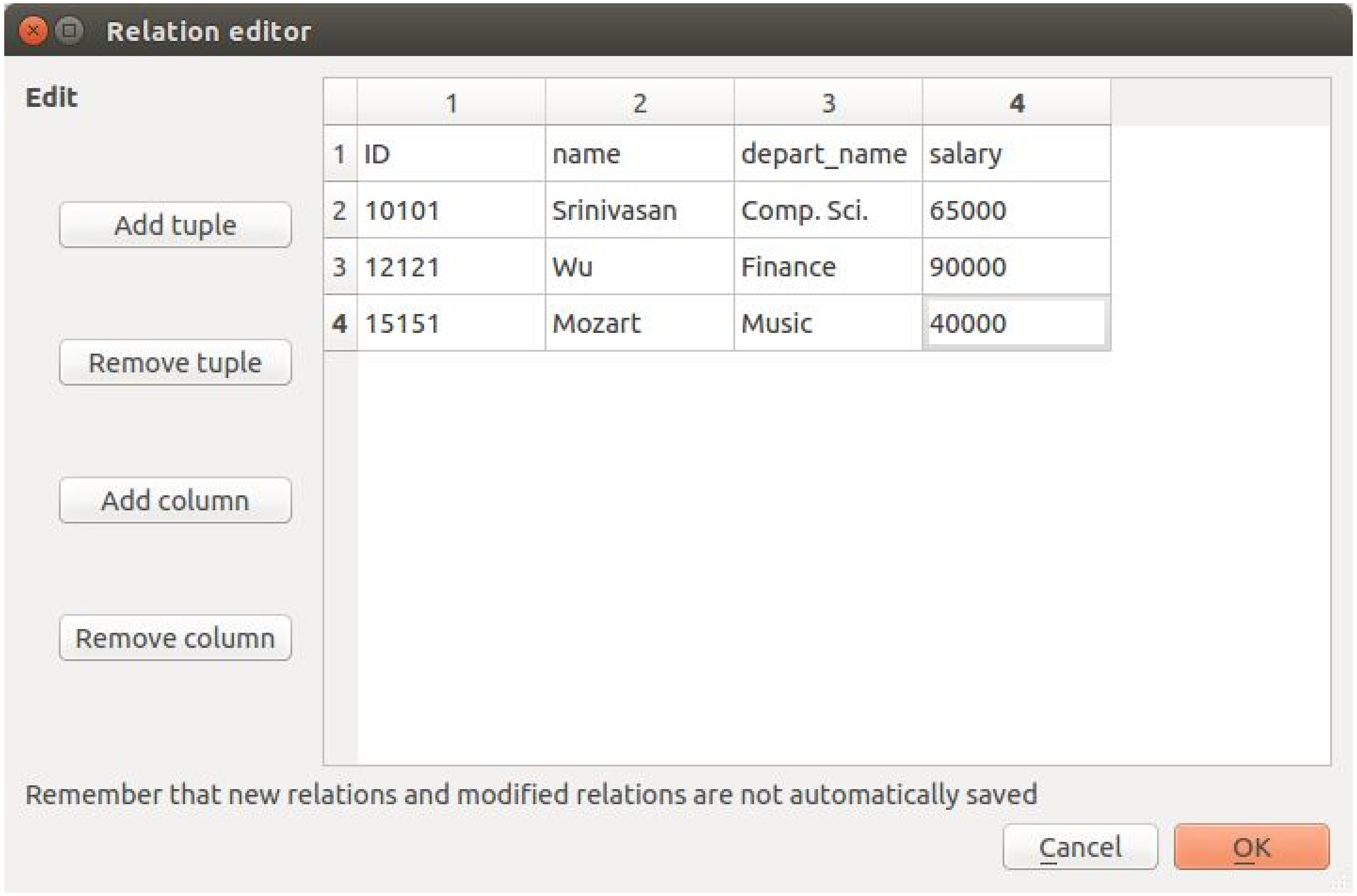
Relational – Creating a relation



	1	2
1	ID	Field name 2
2	Value 1	Value 2

Remember that new relations and modified relations are not automatically saved

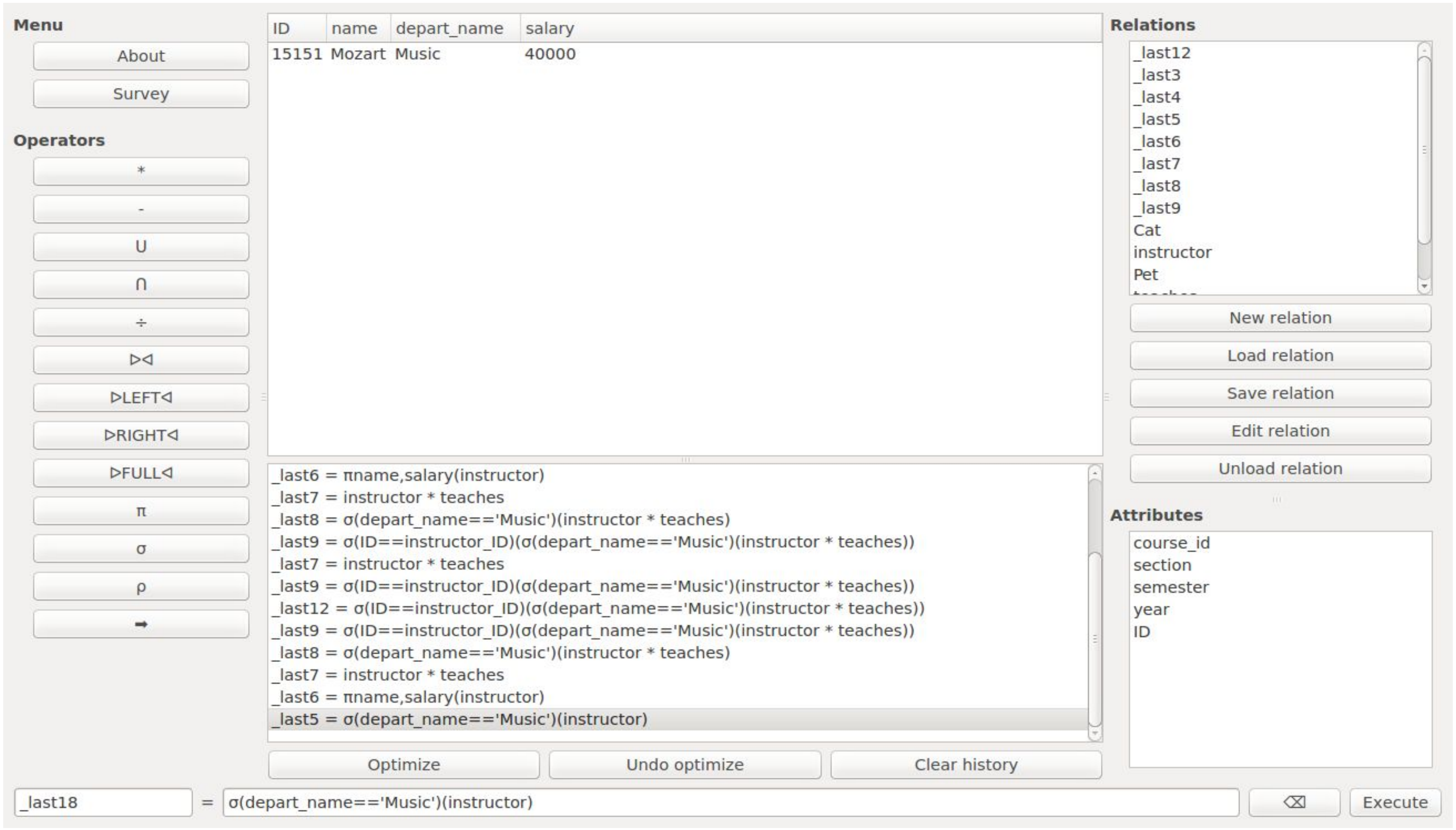
Adding tuples - Relational



	1	2	3	4
1	ID	name	depart_name	salary
2	10101	Srinivasan	Comp. Sci.	65000
3	12121	Wu	Finance	90000
4	15151	Mozart	Music	40000

Remember that new relations and modified relations are not automatically saved

Select Operation - Relational



ID	name	depart_name	salary
15151	Mozart	Music	40000

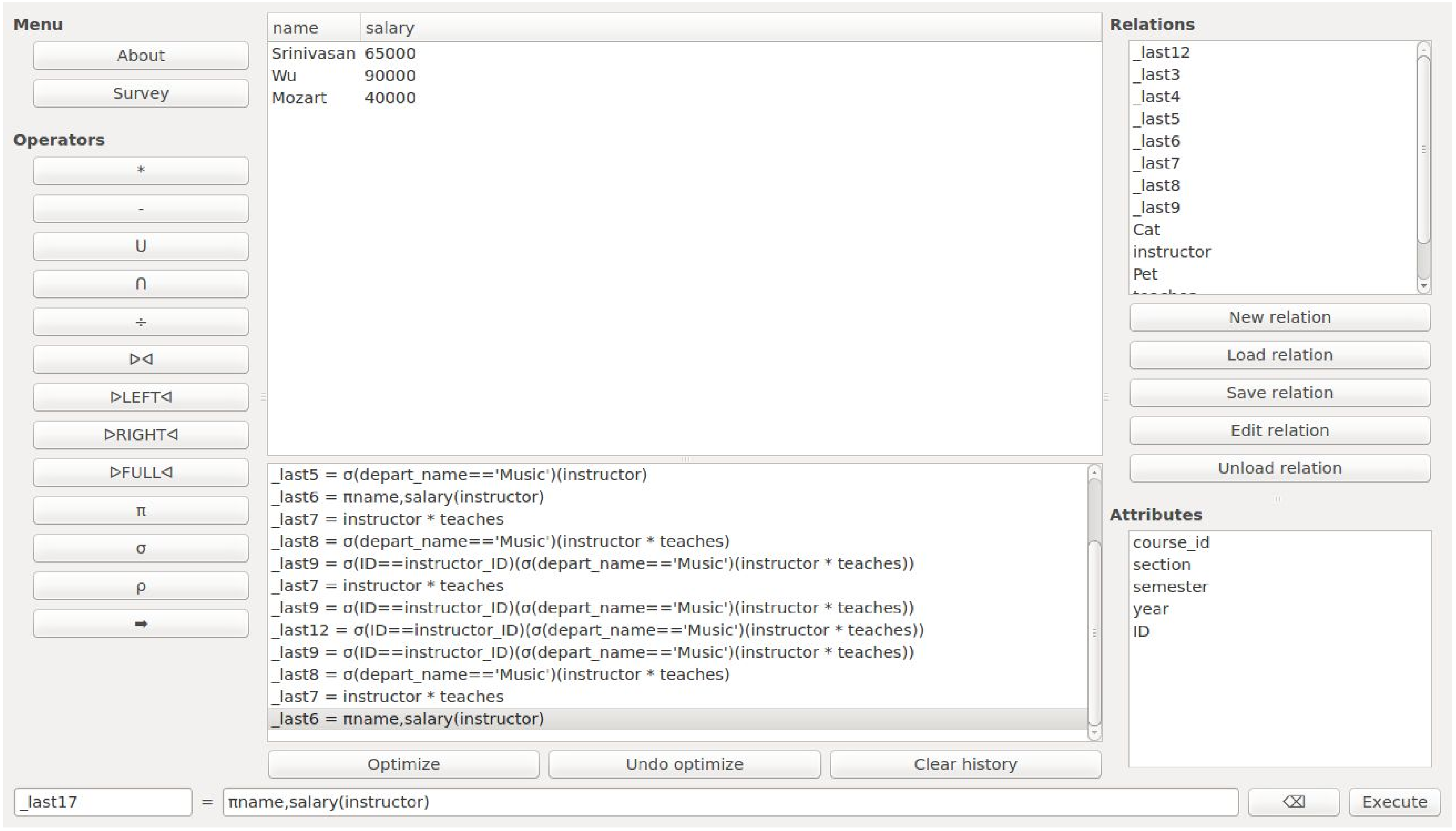
```
_last6 = πname,salary(instructor)
_last7 = instructor * teaches
_last8 = σ(depart_name=='Music')(instructor * teaches)
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last7 = instructor * teaches
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last12 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last8 = σ(depart_name=='Music')(instructor * teaches)
_last7 = instructor * teaches
_last6 = πname,salary(instructor)
_last5 = σ(depart_name=='Music')(instructor)
```

Relations:
_last12
_last3
_last4
_last5
_last6
_last7
_last8
_last9
Cat
instructor
Pet

Attributes:
course_id
section
semester
year
ID

```
_last18 = σ(depart_name=='Music')(instructor)
```

Project Operation - Relational



name	salary
Srinivasan	65000
Wu	90000
Mozart	40000

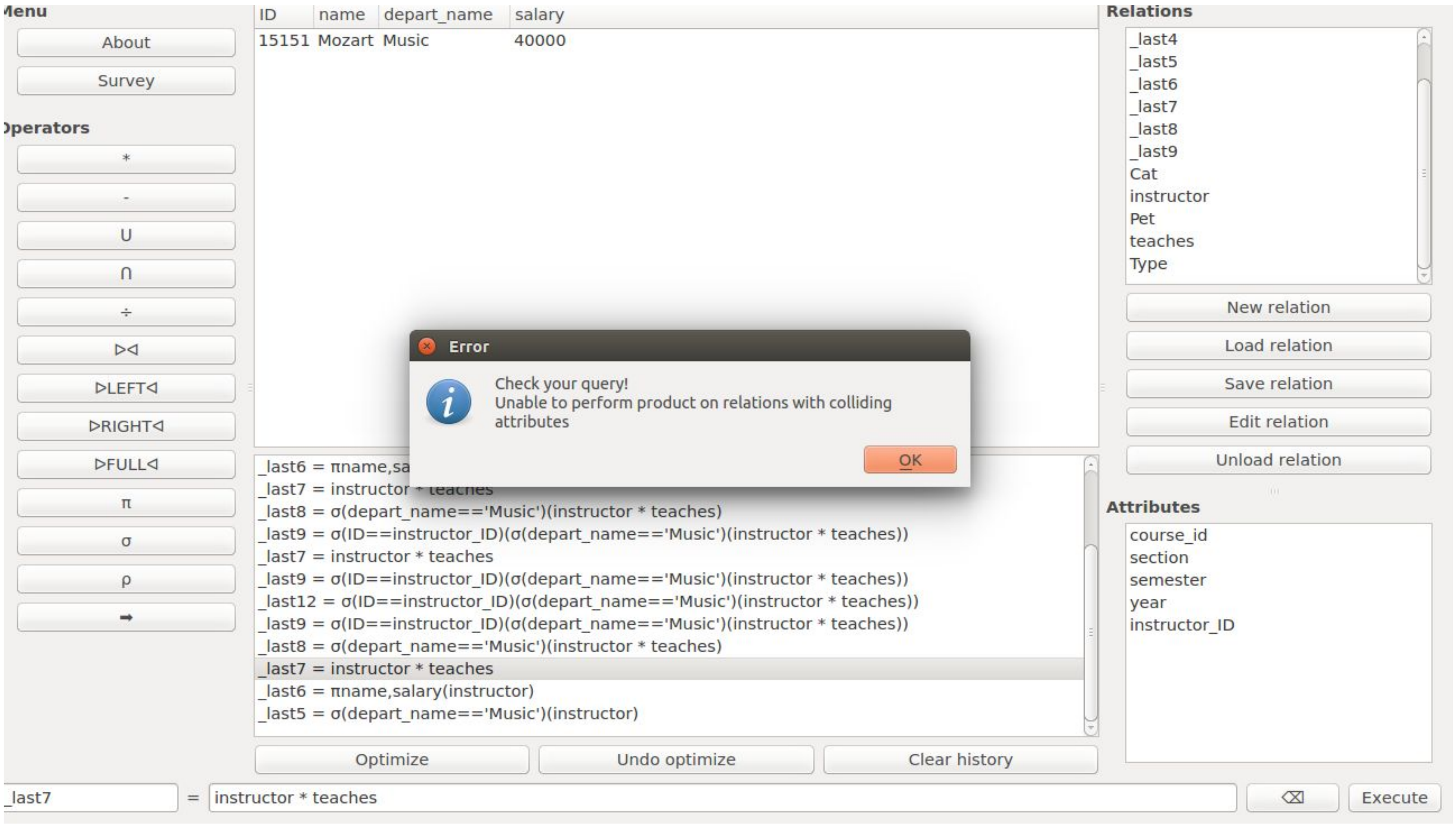
```
_last5 = σ(depart_name=='Music')(instructor)
_last6 = πname,salary(instructor)
_last7 = instructor * teaches
_last8 = σ(depart_name=='Music')(instructor * teaches)
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last7 = instructor * teaches
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last12 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last8 = σ(depart_name=='Music')(instructor * teaches)
_last7 = instructor * teaches
_last6 = πname,salary(instructor)
```

Relations:
- _last12
- _last3
- _last4
- _last5
- _last6
- _last7
- _last8
- _last9
- Cat
- instructor
- Pet

Attributes:
- course_id
- section
- semester
- year
- ID

```
_last17 = πname,salary(instructor)
```

Cartesian Product - Relational



ID	name	depart_name	salary
15151	Mozart	Music	40000

Relations: _last4, _last5, _last6, _last7, _last8, _last9, Cat, instructor, Pet, teaches, Type

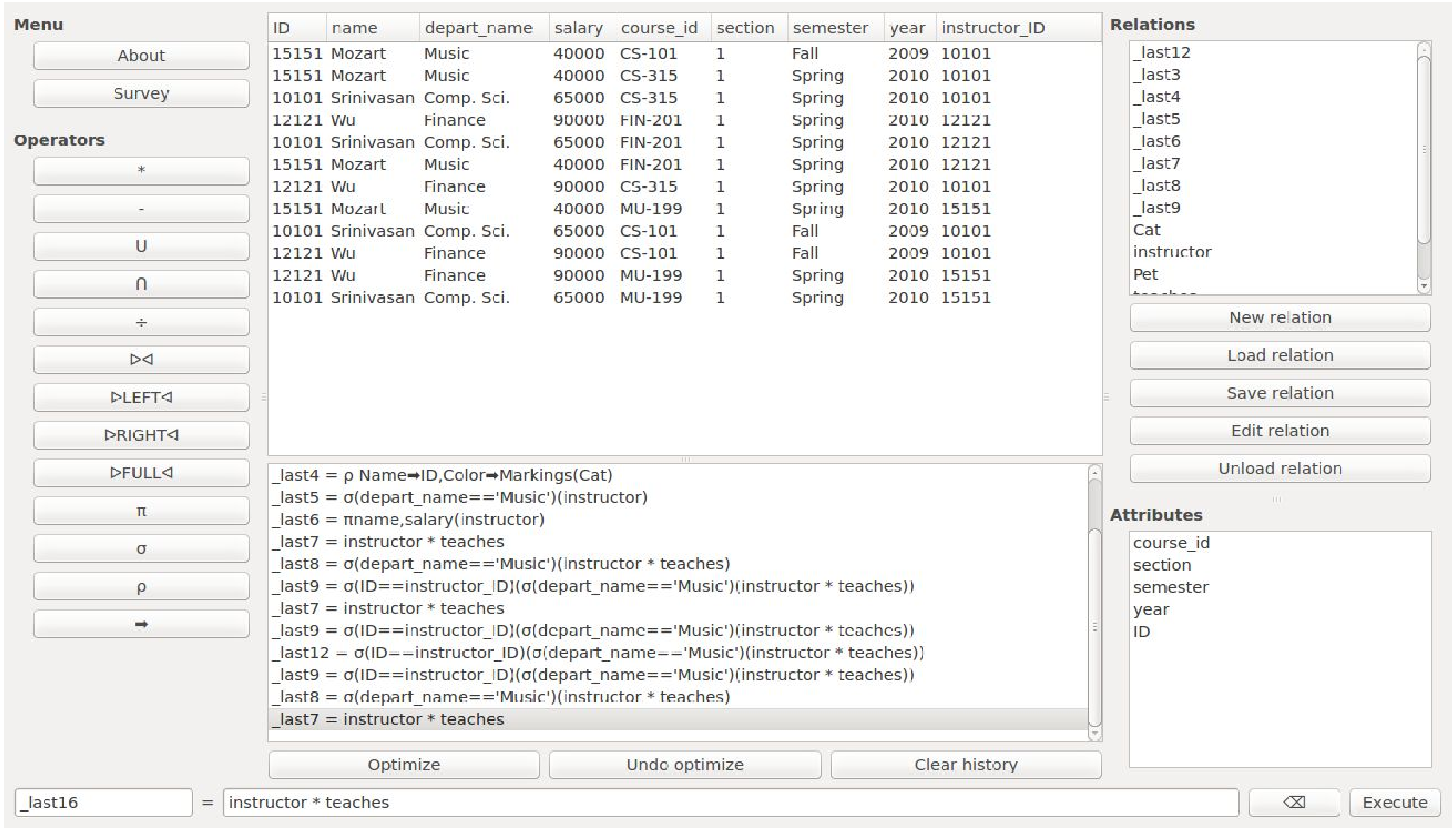
Check your query!
Unable to perform product on relations with colliding attributes

```
_last6 = πname,sa
_last7 = instructor * teaches
_last8 = σ(depart_name=='Music')(instructor * teaches)
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last7 = instructor * teaches
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last12 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last8 = σ(depart_name=='Music')(instructor * teaches)
_last7 = instructor * teaches
_last6 = πname,salary(instructor)
_last5 = σ(depart_name=='Music')(instructor)
```

Attributes: course_id, section, semester, year, instructor_ID

```
_last7 = instructor * teaches
```

Cartesian Product - Relational



ID	name	depart_name	salary	course_id	section	semester	year	instructor_ID
15151	Mozart	Music	40000	CS-101	1	Fall	2009	10101
15151	Mozart	Music	40000	CS-315	1	Spring	2010	10101
10101	Srinivasan	Comp. Sci.	65000	CS-315	1	Spring	2010	10101
12121	Wu	Finance	90000	FIN-201	1	Spring	2010	12121
10101	Srinivasan	Comp. Sci.	65000	FIN-201	1	Spring	2010	12121
15151	Mozart	Music	40000	FIN-201	1	Spring	2010	12121
12121	Wu	Finance	90000	CS-315	1	Spring	2010	10101
15151	Mozart	Music	40000	MU-199	1	Spring	2010	15151
10101	Srinivasan	Comp. Sci.	65000	CS-101	1	Fall	2009	10101
12121	Wu	Finance	90000	CS-101	1	Fall	2009	10101
12121	Wu	Finance	90000	MU-199	1	Spring	2010	15151
10101	Srinivasan	Comp. Sci.	65000	MU-199	1	Spring	2010	15151

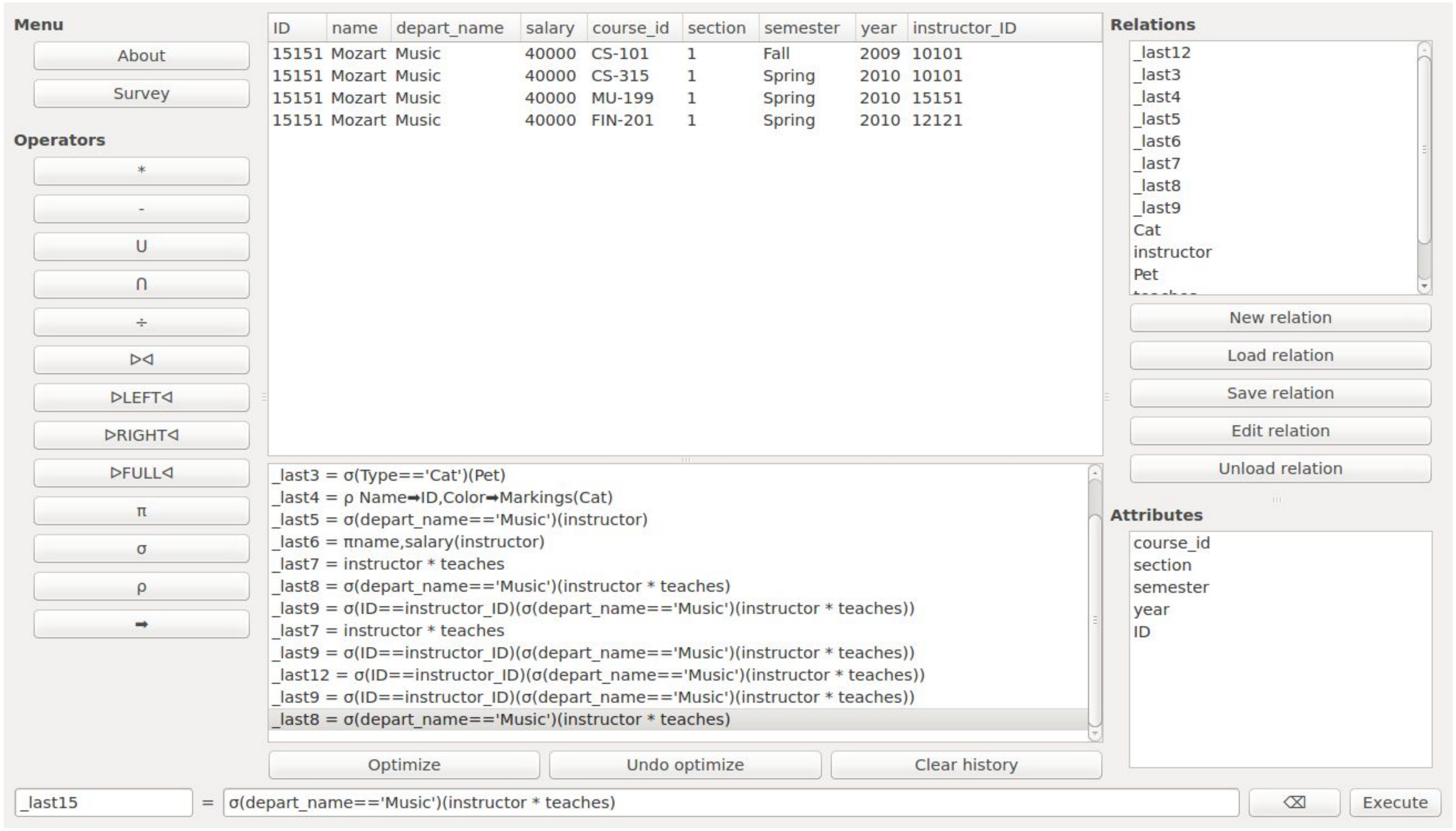
Relations:
```
_last12
_last3
_last4
_last5
_last6
_last7
_last8
_last9
Cat
instructor
Pet
```

Attributes:
```
course_id
section
semester
year
ID
```

```
_last4 = ρ Name➡ID,Color➡Markings(Cat)
_last5 = σ(depart_name=='Music')(instructor)
_last6 = πname,salary(instructor)
_last7 = instructor * teaches
_last8 = σ(depart_name=='Music')(instructor * teaches)
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last7 = instructor * teaches
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last12 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last8 = σ(depart_name=='Music')(instructor * teaches)
_last7 = instructor * teaches
```

```
_last16 = instructor * teaches
```

Relational Algebra Expressions - Relational



ID	name	depart_name	salary	course_id	section	semester	year	instructor_ID
15151	Mozart	Music	40000	CS-101	1	Fall	2009	10101
15151	Mozart	Music	40000	CS-315	1	Spring	2010	10101
15151	Mozart	Music	40000	MU-199	1	Spring	2010	15151
15151	Mozart	Music	40000	FIN-201	1	Spring	2010	12121

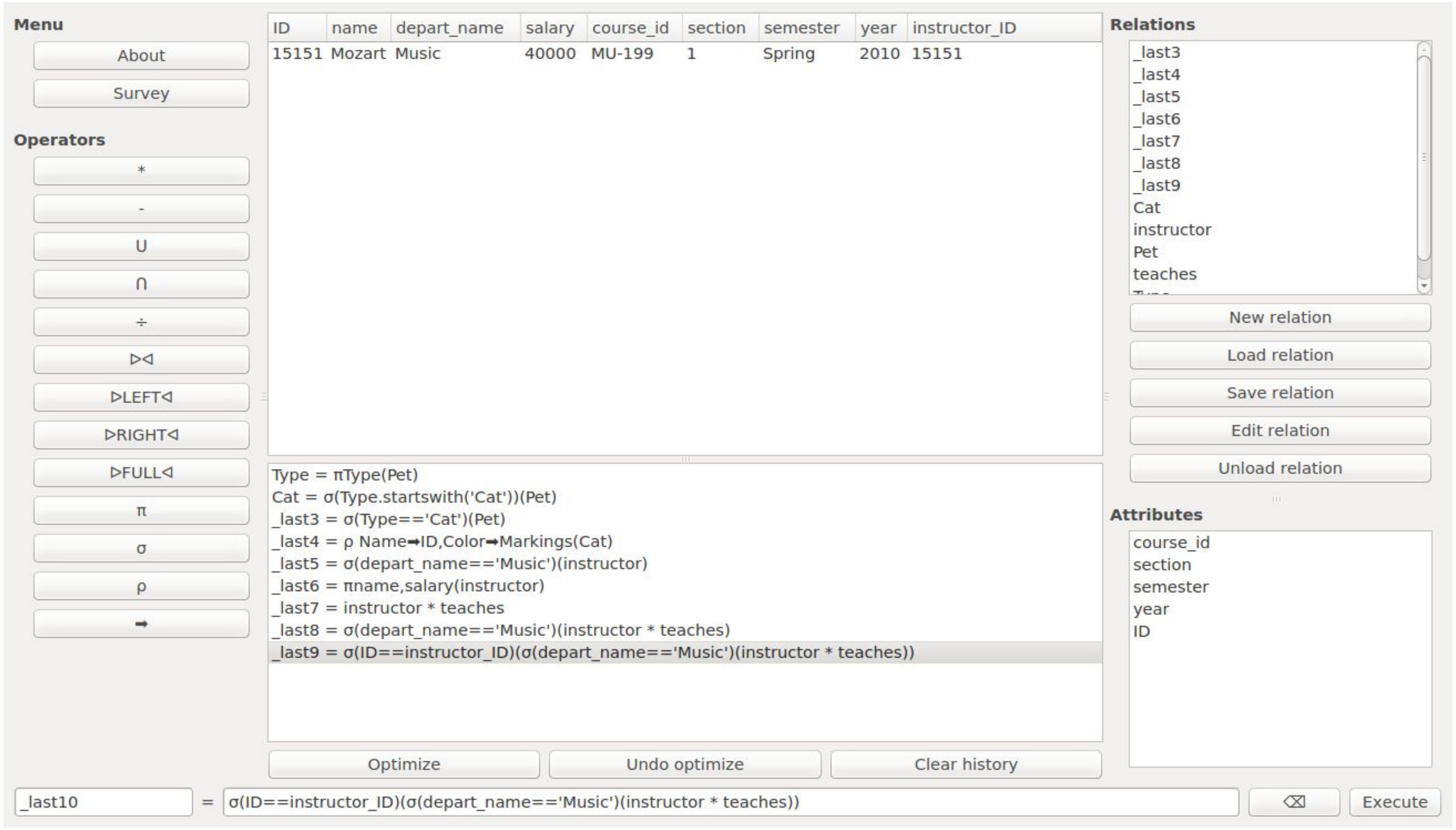
```
_last3 = σ(Type=='Cat')(Pet)
_last4 = ρ Name➡ID,Color➡Markings(Cat)
_last5 = σ(depart_name=='Music')(instructor)
_last6 = πname,salary(instructor)
_last7 = instructor * teaches
_last8 = σ(depart_name=='Music')(instructor * teaches)
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last7 = instructor * teaches
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last12 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
_last8 = σ(depart_name=='Music')(instructor * teaches)
```

Relations:
- _last12
- _last3
- _last4
- _last5
- _last6
- _last7
- _last8
- _last9
- Cat
- instructor
- Pet

Attributes:
- course_id
- section
- semester
- year
- ID

```
_last15 = σ(depart_name=='Music')(instructor * teaches)
```

Relational Algebra Expressions - Relational



ID	name	depart_name	salary	course_id	section	semester	year	instructor_ID
15151	Mozart	Music	40000	MU-199	1	Spring	2010	15151

```
Type = πType(Pet)
Cat = σ(Type.startswith('Cat'))(Pet)
_last3 = σ(Type=='Cat')(Pet)
_last4 = ρ Name➡ID,Color➡Markings(Cat)
_last5 = σ(depart_name=='Music')(instructor)
_last6 = πname,salary(instructor)
_last7 = instructor * teaches
_last8 = σ(depart_name=='Music')(instructor * teaches)
_last9 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
```

```
_last10 = σ(ID==instructor_ID)(σ(depart_name=='Music')(instructor * teaches))
```

Lecture Outline

- Intro to Relational Algebra
- Fundamental Operations
- **Additional Operations**
- Summary
- In Class Exercise

Additional Operations

- We define additional operations that do not add any expressive power to the relational algebra, but that simplify common queries.
  - Set intersection
  - Natural join
  - Division
  - Assignment