Politecnico di Torino

DBMG

Relational algebra

Relational model and relational algebra

Relational Algebra

Introduction

Relation Algebra

Relational Algebra

- Extends the algebra of sets for the relational model
- Defines a set of operators that operate on relations and whose output is another relation
- It satisfies the closure property
  - The result of any algebraic operation on relations is also a relation

Relational algebra operators

- Unary operator
  - selection (σ)
  - projection (π)
- Binary operator
  - cartesian product ($\times$)
  - join ($\bowtie$)
  - union ($\cup$)
  - intersection ($\cap$)
  - difference (-)
  - division (/)

- Set operators
  - union ($\cup$)
  - intersection ($\cap$)
  - difference (-)
  - cartesian product ($\times$)
- Relational operators
  - selection (σ)
  - projection (π)
  - join (   )
  - division (/)

Example of relations

Courses

CCode	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

Teachers

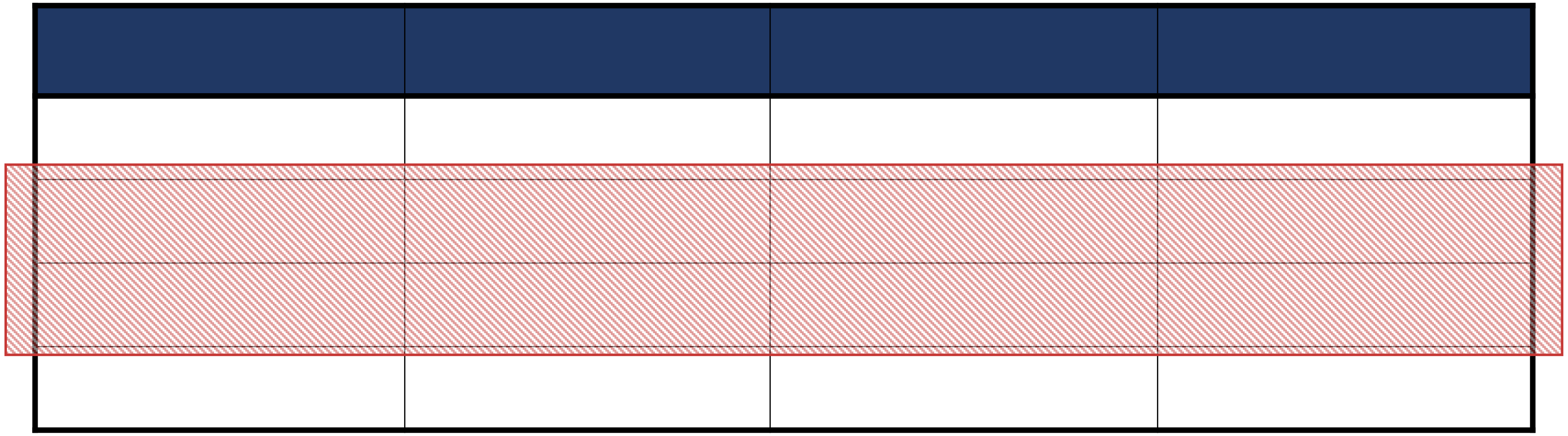
TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

Selection and projection

Relation Algebra

Selection

- The selection extracts a “horizontal” subset from the relation
  - It operates a horizontal partition of the relation



Selection: example

- *Find the courses held in the second semester*

Courses

CCode	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

R

CCode	CName	Semester	TeacherID
M4880	Digital systems	2	D104
F0410	Databases	2	D102

Selection: definition

$$R = \sigma_p A$$

- The selection generates a relation R
  - with the same schema as A
  - containing all the tuples of relation A for which predicate *p* is true
- Predicate *p* is a boolean expression (operators $\wedge, \vee, \neg$) combining expressions that compare attributes, or attributes and constants
  - p: City= 'Turin' $\wedge$ Age>18
  - p: ReturnDate>DeliveryDate+10

Selection: example

- *Find the courses held in the second semester*

R
||
$\sigma_{Semester=2}$
|
Courses

$R = \sigma_{Semester=2} Courses$

Courses

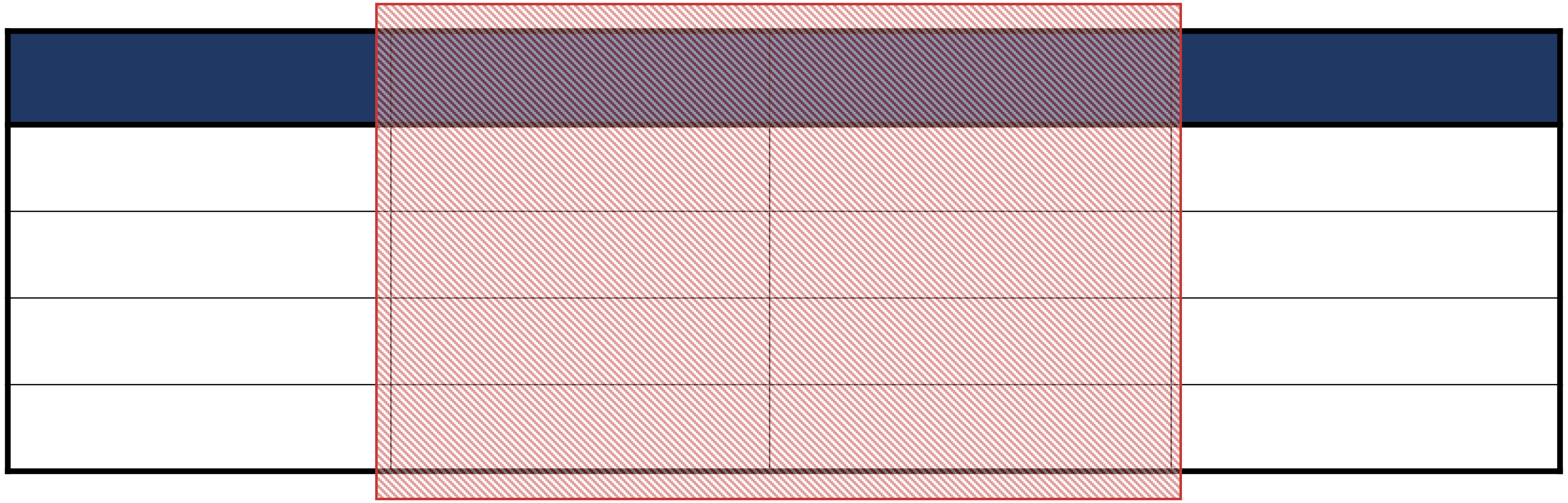
<u>CCode</u>	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

⇒

CCode	CName	Semester	TeacherID
M4880	Digital systems	2	D104
F0410	Databases	2	D102

Projection

- The projection extracts a “vertical” subset from the relation
  - it operates a vertical partition of the relation



Projection: example (n. 1)

- *Find the names of teachers*

Teachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

R

PName
Green
Black
White

Projection: definition

$$R = \pi_L A$$

- The projection π_L generates a relation R
  - whose schema is the list of attributes L (subset of A's schema)
  - containing all of the tuples present in A
- The duplicates that may be caused by excluding the attributes not contained in L are deleted
  - if L includes a candidate key, there are no duplicates

Projection: example (n. 1)

- *Find the names of teachers*

R
||
π_{PName}
|
Teachers

$R = \pi_{PName} Teachers$

Teachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

Projection: example (n. 2)

- *Find the names of the departments in which at least one professor is present*

R

||

$\pi_{Department}$

|

Teachers

$R = \pi_{Department} Teachers$

Teachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

R

Department
Computer engineering
Department of electronics

Selection+projection: example

- *Select the names of courses in the second semester*

Courses

CCode	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

Selection

CCode	CName	Semester	TeacherID
M4880	Digital systems	2	D104
F0410	Databases	2	D102

Projection

R

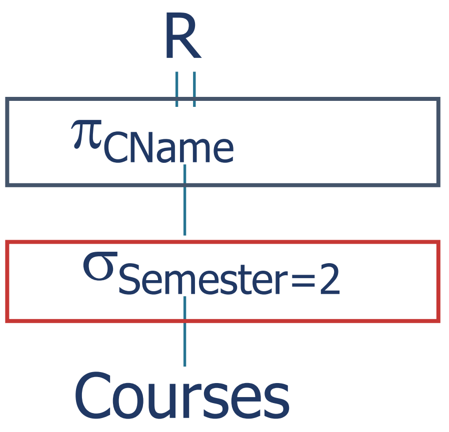
CName
Digital systems
Databases

Selection+projection: example

- *Select the names of courses in the second semester*

Courses

<u>CCode</u>	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102



Selection

CCode	CName	Semester	TeacherID
M4880	Digital systems	2	D104
F0410	Databases	2	D102

Projection

R

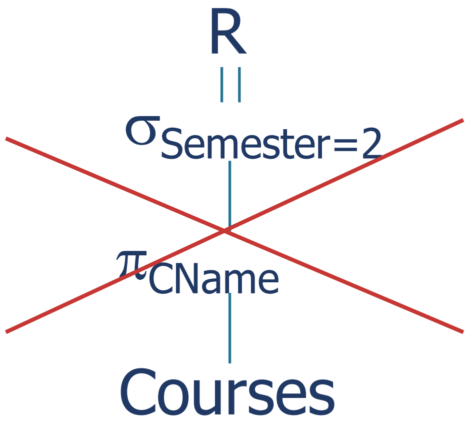
CName
Digital systems
Databases

Selection+projection: wrong solution

- *Select the names of courses in the second semester*

CCode	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

Courses



Projection

CName
Computer science
Digital systems
Electronics
Databases

<u>The Semester attribute is not available in the output relation</u>: the selection operation cannot be carried out

Cartesian product and join

Relation Algebra

Cartesian product

- The Cartesian product of two relations A and B generates all the pairs formed by a tuple of A and a tuple of B

Cartesian product: example

- *Find the Cartesian product of courses and teachers*

Cartesian product: example

Courses

CCode	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

Teachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

Cartesian product: example

R

Courses CCode	Courses. CName	Courses. Semester	Courses. TeacherID	Teachers. TeacherID	Teachers. Pname	Teachers. Department
M2170	*Computer science*	*1*	*D102*	*D102*	*Green*	*Computer engineering*
M2170	*Computer science*	*1*	*D102*	*D105*	*Black*	*Computer engineering*
M2170	*Computer science*	*1*	*D102*	*D104*	*White*	*Department of electronics*

Cartesian product: example

R

Courses CCode	Courses. CName	Courses. Semester	Courses. TeacherID	Teachers. TeacherID	Teachers. Pname	Teachers. Department
M2170	*Computer science*	*1*	*D102*	*D102*	*Green*	*Computer engineering*
M2170	*Computer science*	*1*	*D102*	*D105*	*Black*	*Icomputer engineering*
M2170	*Computer science*	*1*	*D102*	*D104*	*White*	*Department of electronics*
M4880	*Digital systems*	*2*	*D104*	*D102*	*Green*	*Computer engineering*
M4880	*Digital systems*	*2*	*D104*	*D105*	*Black*	*Icomputer engineering*
M4880	*Digital systems*	*2*	*D104*	*D104*	*White*	*Department of electronics*
…	…	…	…	…	…	…

Cartesian product: example

R

Courses CCode	Courses. CName	Courses. Semester	Courses. TeacherID	Teachers. TeacherID	Teachers. Pname	Teachers. Department
M2170	Computer science	1	D102	D102	Green	Computer engineering
M2170	Computer science	1	D102	D105	Black	Computer engineering
M2170	Computer science	1	D102	D104	White	Department of electronics
M4880	Digital systems	2	D104	D102	Green	Computer engineering
M4880	Digital systems	2	D104	D105	Black	Computer engineering
M4880	Digital systems	2	D104	D104	White	Department of electronics
F1401	Electronics	1	D104	D102	Green	Computer engineering
F1401	Electronics	1	D104	D105	Black	Computer engineering
F1401	Electronics	1	D104	D104	White	Department of electronics
F0410	Databases	2	D102	D102	Green	Computer engineering
F0410	Databases	2	D102	D105	Black	Computer engineering
F0410	Databases	2	D102	D104	White	Department of electronics

Cartesian product: definition

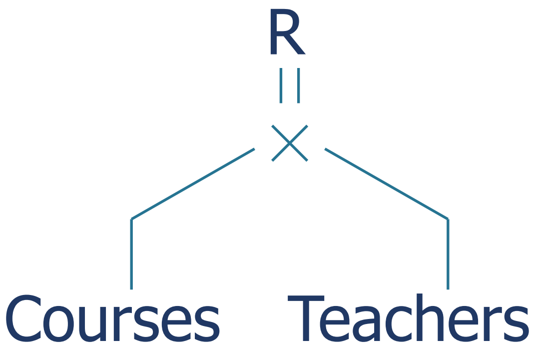
$$R = A \times B$$

- The Cartesian product of two relations A and B yields a relation R
  - whose schema is the union of the schemas of A and B
  - containing all the pairs formed by a tuple of A and a tuple of B
- The Cartesian product is
  - commutative
    - $A \times B = B \times A$
  - associative
    - $(A \times B) \times C = A \times (B \times C)$

Cartesian product: example

- *Find the Cartesian product of courses and teachers*

$$R = \text{Courses} \times \text{Teachers}$$



Link between attributes

R

Courses CCode	Courses. CName	Courses. Semester	Courses. TeacherID	Teachers. TeacherID	Teachers.P name	Teachers. Department
M2170	*Computer science*	*1*	*D102*	*D102*	*Green*	*Computer engineering*
M2170	*Computer science*	*1*	*D102*	*D105*	*Black*	*Icomputer engineering*
M2170	*Computer science*	*1*	*D102*	*D104*	*White*	*Department of electronics*
M4880	*Digital systems*	*2*	*D104*	*D102*	*Green*	*Computer engineering*
M4880	*Digital systems*	*2*	*D104*	*D105*	*Black*	*Icomputer engineering*
M4880	*Digital systems*	*2*	*D104*	*D104*	*White*	*Department of electronics*
…	…	…	…	…	…	…

Join

- The join of two relations A and B generates all the pairs formed by a tuple of A and a tuple of B that are *"semantically linked"*

Join: example

- *Find information about courses and the teachers that hold them*

Join: example

- *Find information about courses and the teachers that hold them*

Courses

CCode	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

Teachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

Join: example

R

Courses CCode	Courses. CName	Courses. Semester	Courses. TeacherID	Teachers. TeacherID	Teachers. Pname	Teachers. Department
M2170	*Computer science*	*1*	*D102*	*D102*	*Green*	*Computer engineering*
M2170	*Computer science*	*1*	*D102*	*D105*	*Black*	*Icomputer engineering*
M2170	*Computer science*	*1*	*D102*	*D104*	*White*	*Department of electronics*
M4880	*Digital systems*	*2*	*D104*	*D102*	*Green*	*Computer engineering*
M4880	*Digital systems*	*2*	*D104*	*D105*	*Black*	*Icomputer engineering*
M4880	*Digital systems*	*2*	*D104*	*D104*	*White*	*Department of electronics*
...	...	...	...	...	...	...

Join: example

R

Courses CCode	Courses. CName	Courses. Semester	Courses. TeacherID	Teachers. TeacherID	Teachers. Pname	Teachers. Department
M2170	Computer science	1	D102	D102	Green	Computer engineering
M4880	Digital systems	2	D104	D104	White	Department of electronics
F1401	Electronics	1	D104	D104	White	Department of electronics
F0410	Databases	2	D102	D102	Green	Computer engineering

Join: example

R

Courses CCode	Courses. CName	Courses. Semester	Courses. TeacherID	Teachers. TeacherID	Teachers. Pname	Teachers. Department
M2170	Computer science	1	D102	D102	Green	Computer engineering
M4880	Digital systems	2	D104	D104	White	Department of electronics
F1401	Electronics	1	D104	D104	White	Department of electronics
F0410	Databases	2	D102	D102	Green	Computer engineering

- *NB:* Professor (D105,Black,Computer engineering), who does not teach any courses does not appear in the result of the join

Join: definition

- The join is a derived operator
  - it can be expressed using operators x, σ_p, π_L
- The join is defined separately as it expresses synthetically many recurrent operations in database queries
- There are different kinds of joins
  - natural join
  - theta-join (and its special case equi-join)
  - semi-join

Natural join, theta-join and semi-join

Relation Algebra

Natural join: definition and properties

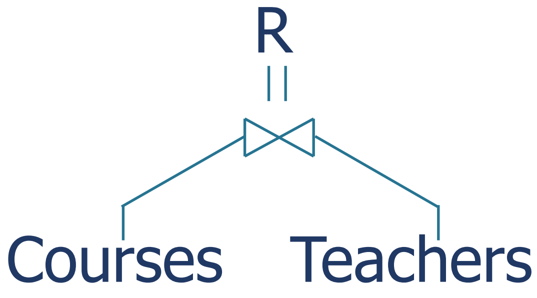
$$R = A \bowtie B$$

- The natural join of two relations A and B generates a relation R
  - whose schema is composed of
    - the attributes which are present in A's schema and not in B's
    - the attributes present in B's schema and not in A's
    - a single copy of common attributes (with the same name in the schema of A and B)
  - containing all of the pairs made up of a tuple of A and a tuple of B for which the value of common attributes is the same
- Natural join is commutative and associative

Natural join: example

- *Find information about the courses and the teachers that hold them*

R = Courses ⋈ Teachers



R R

Courses CCode	Courses. CName	Courses. Semester	Courses. TeacherID	Teachers. Pname	Teachers. Department
M2170	Computer science	1	D102	Green	Computer engineering
M4880	Digital systems	2	D104	White	Department of electronics
F1401	Electronics	1	D104	White	Department of electronics
F0410	Databases	2	D102	Green	Computer engineering

Note: The common attribute TeacherID is present only once in the schema of the resulting relation R

Theta-join: definition

$$R = A \bowtie_p B$$

- The theta-join of two relations A and B generates all the pairs formed by a tuple of A and B that satisfy a generic *"join/link condition"*
- The theta-join of two relations A and B generates a relation R
  - whose schema is the union of the schemas of A and B
  - containing all the pairs made up of a tuple of A and a tuple of B for which the predicate *p* is true
- The predicate *p* is in the form X θ Y
  - X is an attribute of A, Y is an attribute of B
  - θ is a comparison operator compatible with the domains of X and of Y
- Theta-join is commutative and associative

Equi-join: definition

$$R = A \bowtie_p B$$

- Equi-join
  - Particular case of theta-join in which θ is the equivalence operator (=)

Theta-join: example

- *Find the identifiers of the teachers that hold at least two courses*

Courses C1

CCode	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

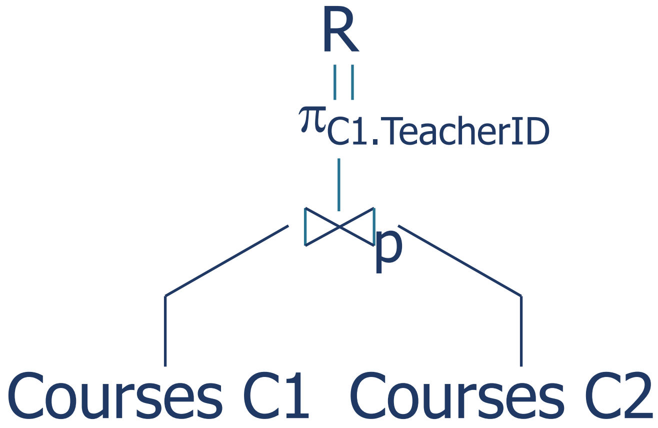
Courses C2

CCode	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

Theta-join: example

- *Find the identifiers of the teachers that hold at least two courses*

$$R = \pi_{C1.TeacherID}((\text{Courses } C1) \bowtie_{p} (\text{Courses } C2))$$



p: $C1.TeacherID=C2.TeacherID \wedge C1.CCode<>C2.CCode$

Theta-join: example

Condition: <> ... =

Courses C1. CCode	Courses C1. CName	Courses C1 Semester	Courses C1. TeacherID	Courses C2. CCode	Courses C2. CName	Courses C2. Semester	Courses C2. TeacherID
M2170	Computer science	1	D102	M2170	Computer science	1	D102
M2170	Computer science	1	D102	M4880	Digital systems	2	D104
M2170	Computer science	1	D102	F1401	Electronics	1	D104
M2170	**Computer science**	**1**	**D102**	**F0410**	**Databases**	**2**	**D102**
M4880	Digital systems	2	D104	M2170	Computer science	1	D102
M4880	Digital systems	2	D104	M4880	Digital systems	2	D104
M4880	**Digital systems**	**2**	**D104**	**F1401**	**Electronics**	**1**	**D104**
M4880	Digital systems	2	D104	F0410	Databases	2	D102
F1401	Electronics	1	D104	M2170	Computer science	1	D102
F1401	**Electronics**	**1**	**D104**	**M4880**	**Digital systems**	**2**	**D104**
F1401	Electronics	1	D104	F1401	Electronics	1	D104
F1401	Electronics	1	D104	F0410	Databases	2	D102
F0410	**Databases**	**2**	**D102**	**M2170**	**Computer science**	**1**	**D102**
F0410	Databases	2	D102	M4880	Digital systems	2	D104
F0410	Databases	2	D102	F1401	Electronics	1	D104
F0410	Databases	2	D102	F0410	Databases	2	D102

Theta-join: example

Courses C1. CCode	Courses C1. CName	Courses C1. Semester	Courses C1. TeacherID	Courses C2. CCode	Courses C2. CName	Courses C2. Semester	Courses C2. TeacherID
M2170	Computer science	1	*D102*	F0410	Databases	2	D102
M4880	Digital systems	2	*D104*	F1401	Electronics	1	D104
F1401	Electronics	1	*D104*	M4880	Digital systems	2	D104
F0410	Databases	2	*D102*	M2170	Computer science	1	D102

Projection

R

Courses C1. TeacherID
D102
D104

Semi-join: definition and properties

$$R = A \ltimes_p B$$

- The semi-join of two relations A and B selects all the tuples of A that are *"semantically linked"* to at least one tuple of B
  - the information from B does not appear in the result
- The semi-join of two relations A and B generates a relation R
  - which has the same schema as A
  - containing all the tuples of A for which the predicate specified by *p* is true
- The predicate *p* is expressed in the same form as the theta-join (comparison between the attributes of A and B)

Semi-join: properties

- The semi-join can be expressed as a function of the theta-join
  - $A \ltimes_p B = \pi_{schema(A)}(A \bowtie_p B)$
- The semi-join *does not satisfy* the commutative property

Semi-join: example

- *Find information relative to teachers that hold at least one course*

Courses

CCode	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

Teachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

Semi-join: example

Teachers. TeacherID	Teachers. Pname	Teachers. Department	Courses. CCode	Courses.CName	Courses. Semester	Courses. TeacherID
D102	*Green*	*Computer engineering*	*M2170*	*Computer science*	*1*	*D102*
D102	Green	Computer engineering	M4880	Digital systems	2	D104
D102	Green	Computer engineering	F1401	Electronics	1	D104
D102	*Green*	*Computer engineering*	*F0410*	*Databases*	*2*	*D102*
D105	Black	Computer engineering	M2170	Computer science	1	D102
D105	Black	Computer engineering	M4880	Digital systems	2	D104
D105	Black	Computer engineering	F1401	Electronics	1	D104
D104	*White*	*Department of electronics*	*F1401*	*Electronics*	*1*	*D104*
…	…	…	…	…	…	…

DBMG

Semi-join: example

Teachers. TeacherID	Teachers. Pname	Teachers. Department	Courses. CCode	Courses. CName	Courses. Semester	Courses. TeacherID
D102	*Green*	*Computer engineering*	M2170	Computer science	1	D102
D102	*Green*	*Computer engineering*	F0410	Databases	2	D102
D104	*White*	*Department of electronics*	M4880	Digital systems	2	D104
D104	*White*	*Electronics*	F1401	Electronics	3	D104

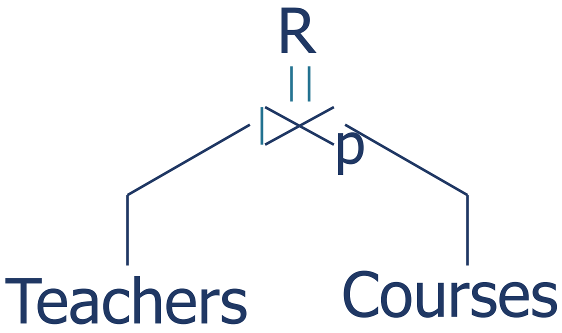
R

Teachers. TeacherID	Teachers. Pname	Teachers. Department
D102	Green	Computer engineering
D104	White	Department of electronics

Semi-join: example

- *Find information relative to teachers that hold at least one course*

$R = Teachers \ltimes_p Courses$



p: Teachers.TeacherID= Courses.TeacherID

R

Teachers. TeacherID	Teachers. Pname	Teachers. Department
D102	Green	Computer engineering
D104	White	Department of electronics

Outer join

Relation Algebra

Outer-join

- Version of join that allows us to preserve the information relative to tuples that are not semantically linked by the join predicate
  - complete the tuples that lack a counterpart with null values
- There are three kinds of outer-join
  - left: only the tuples of the first operand are completed
  - right: only the tuples of the second operand are completed
  - full: the tuples of both operands are completed

Left outer-join

- The left outer-join of two relations A and B generates all pairs of
  - a tuple of A and one of B that are *"semantically linked"*

    +
  - a tuple of A *"not semantically linked"* to any tuple of B, completed with null values for all the attributes of B

Semi-join: example

- *Find information about teachers and about the courses that they hold*

Courses

CCode	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

Teachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

Left outer-join: example

R

Teachers. TeacherID	Teachers. Pname	Teachers. Department	Courses. CCode	Courses. CName	Courses. Semester	Courses. TeacherID
D102	*Green*	*Computer engineering*	*M2170*	*Computer science*	*1*	*D102*
D102	*Green*	*Computer engineering*	*F0410*	*Databases*	*2*	*D102*
D104	*White*	*Department of electronics*	*M4880*	*Digital systems*	*2*	*D104*
D104	*White*	*Electronics*	*F1401*	*Electronics*	*3*	*D104*

Left outer-join: example

R

Teachers. TeacherID	Teachers. Pname	Teachers. Department	Courses. CCode	Courses. CName	Courses. Semester	Courses. TeacherID
D102	*Green*	*Computer engineering*	M2170	Computer science	1	D102
D102	*Green*	*Computer engineering*	F0410	Databases	2	D102
D104	*White*	*Department of electronics*	M4880	Digital systems	2	D104
D104	*White*	*Electronics*	F1401	Electronics	1	D104
D105	*Black*	*Computer engineering*	*null*	*null*	*null*	*null*

Left outer-join: definition

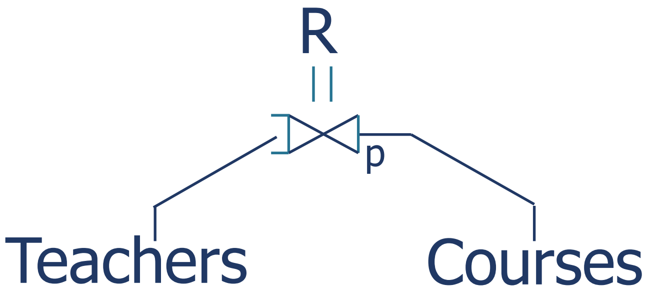
$$R = A \;⟕_p\; B$$

- The left outer-join of two relations A and B generates a relation R
  - whose schema is the union of the schemas of A and B
  - containing the pairs made up of:
    - a tuple of A and a tuple of B for which the predicate *p* is true
    - a tuple of A that is not correlated by means of the predicate *p* to any tuple of B completed with null values for all the attributes of B

➢ The left outer-join *is not* commutative

Left outer-join: example

- *Find information about teachers and about the courses that they hold*



$$R = \text{Teachers} ⟕_{p} \text{Courses}$$

p: Teachers.TeacherID=Courses.TeacherID

Teachers. TeacherID	Teachers. Pname	Teachers. Department	Courses. CCode	Courses. CName	Courses. Semester	Courses. TeacherID
D102	Green	Computer engineering	M2170	Computer science	1	D102
D102	Green	Computer engineering	F0410	Databases	2	D102
D104	White	Department of electronics	M4880	Digital systems	2	D104
D104	White	Electronics	F1401	Electronics	1	D104
D105	Black	Computer engineering	null	null	null	null

Right outer-join: definition

$$R = A \rtimes_p B$$

- The right outer-join of two relations A and B generates a relation R
  - whose schema is the union of the schemas of A and B
  - containing the pairs made up of
    - a tuple of A and a tuple of B for which the predicate *p* is true
    - a tuple of B that is not correlated by means of the predicate *p* to any tuple of A completed with null values for all the attributes of A

➢ The right outer-join *is not* commutative

Full outer-join: definition and properties

$$R = A \,⟗_p\, B$$

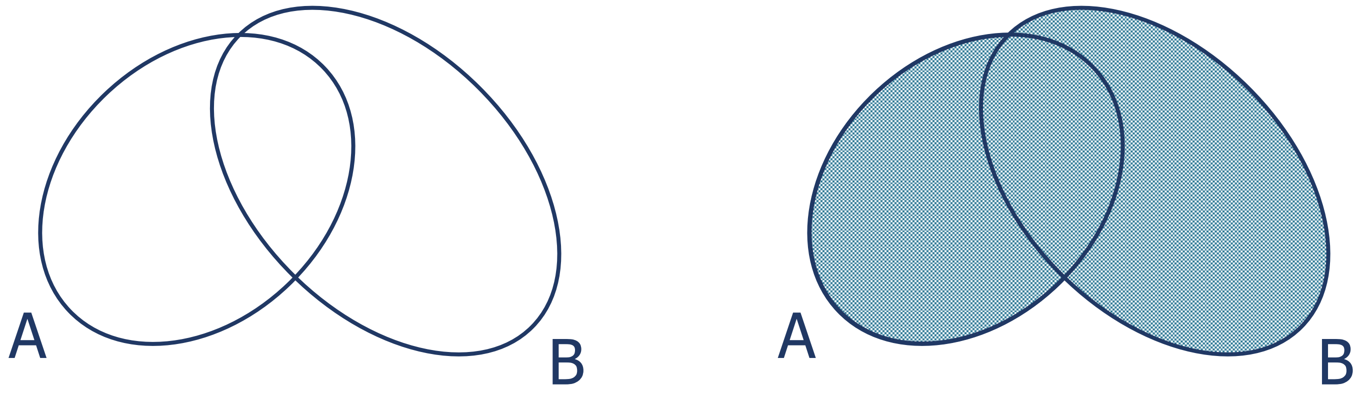
- The full outer-join of two relations A and B generates the relation R
  - whose schema is the union of the schemas of A and B
- containing the pairs formed by:
  - a tuple of A and a tuple of B for which predicate *p* is true
  - a tuple of A that is not correlated by means of the predicate *p* to any tuple of B completed with null values for all the attributes of B
  - a tuple of B that is not correlated by means of the predicate *p* to any tuple of A completed with null values for all the attributes of A
- The full outer-join is commutative

Union and intersection

Relation Algebra

Union

- The union of two relations A and B selects all the tuples present in at least one of the two relations



Union: example

- *Find information relative to the teachers of bachelor's degree or master's degree courses*

BachelorTeachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

MasterTeachers

TeacherID	PName	Department
D102	Green	Computer engineering
D101	Rossi	Department of electrics

Union: example

- *Find information relative to the teachers of bachelor's degree or master's degree courses*

BachelorTeachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

MasterTeachers

TeacherID	PName	Department
D102	Green	Computer engineering
D101	Rossi	Department of electrics

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics
D101	Rossi	Department of electrics

Note: Duplicate tuples are deleted

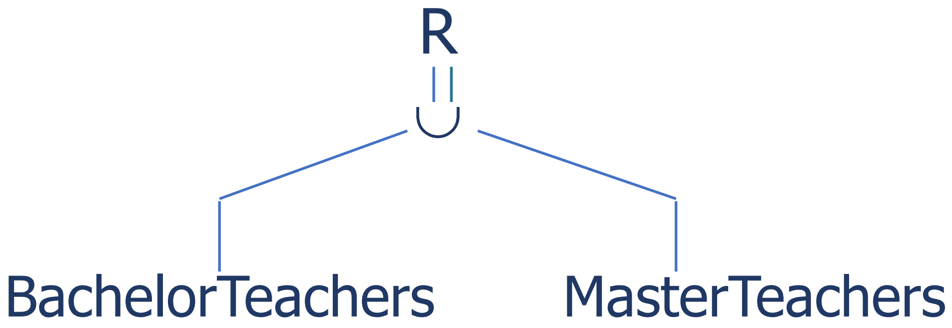
Union: definition and properties

$$R = A \cup B$$

- The union of two relations A and B generates the relation R
  - which has the same schema of A and B
  - containing all the tuples belonging to A and all the tuples belonging to B (or both)
- *Compatibility*
  - the relations A and B must have the same schema (number and type of attributes)
- Duplicate tuples are eliminated
- The union is commutative and associative

Union: example

- *Find information relative to the teachers of bachelor's degree or master's degree courses*



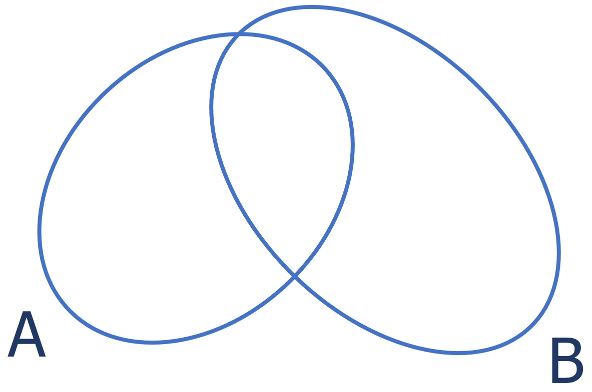
R = BachelorTeachers ∪ MasterTeachers

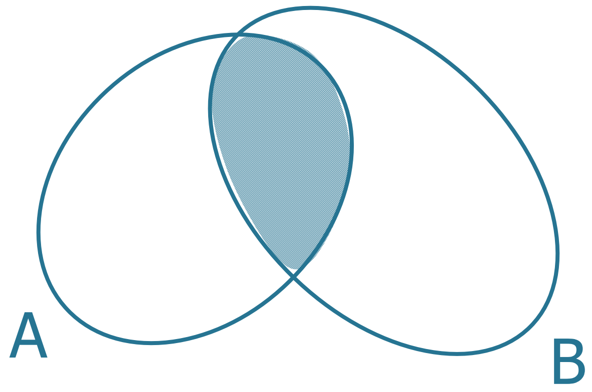
R

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics
D101	Rossi	Department of electrics

Intersection

- The intersection of two relations A and B selects all the tuples present in both relations





Intersection: example

- *Find information relative to the teachers of both bachelor's degree and master's degree courses*

BachelorTeachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

MasterTeachers

TeacherID	PName	Department
D102	Green	Computer engineering
D101	Rossi	Department of electrics

Intersection: example

- *Find information relative to the teachers of both bachelor's degree and master's degree courses*

BachelorTeachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

MasterTeachers

TeacherID	PName	Department
D102	Green	Computer engineering
D101	Rossi	Department of electrics

TeacherID	PName	Department
D102	Green	Computer engineering

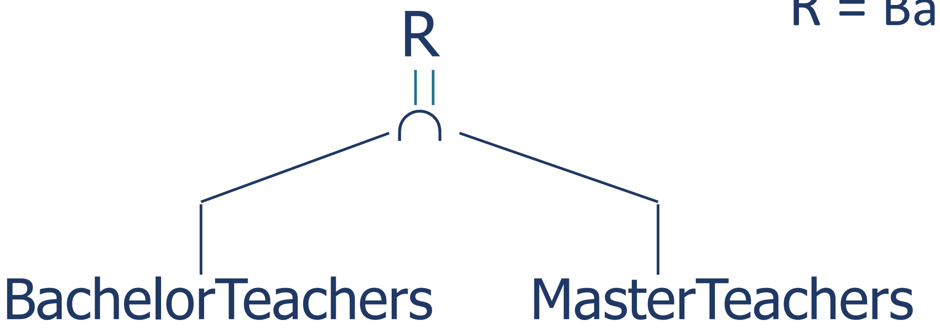
Intersection: definition and properties

$$R = A \cap B$$

- The intersection of two relations A and B generates a relation R
    - with the same schema of A and B
    - containing all the tuples belonging to both A and B
- *Compatibility*
    - relations A and B must have the same schema (number and type of attributes)
- Intersection is commutative and associative

Intersection: example

- *Find information relative to the teachers of both bachelor's degree and master's degree courses*



R = BachelorTeachers ∩ MasterTeachers

R

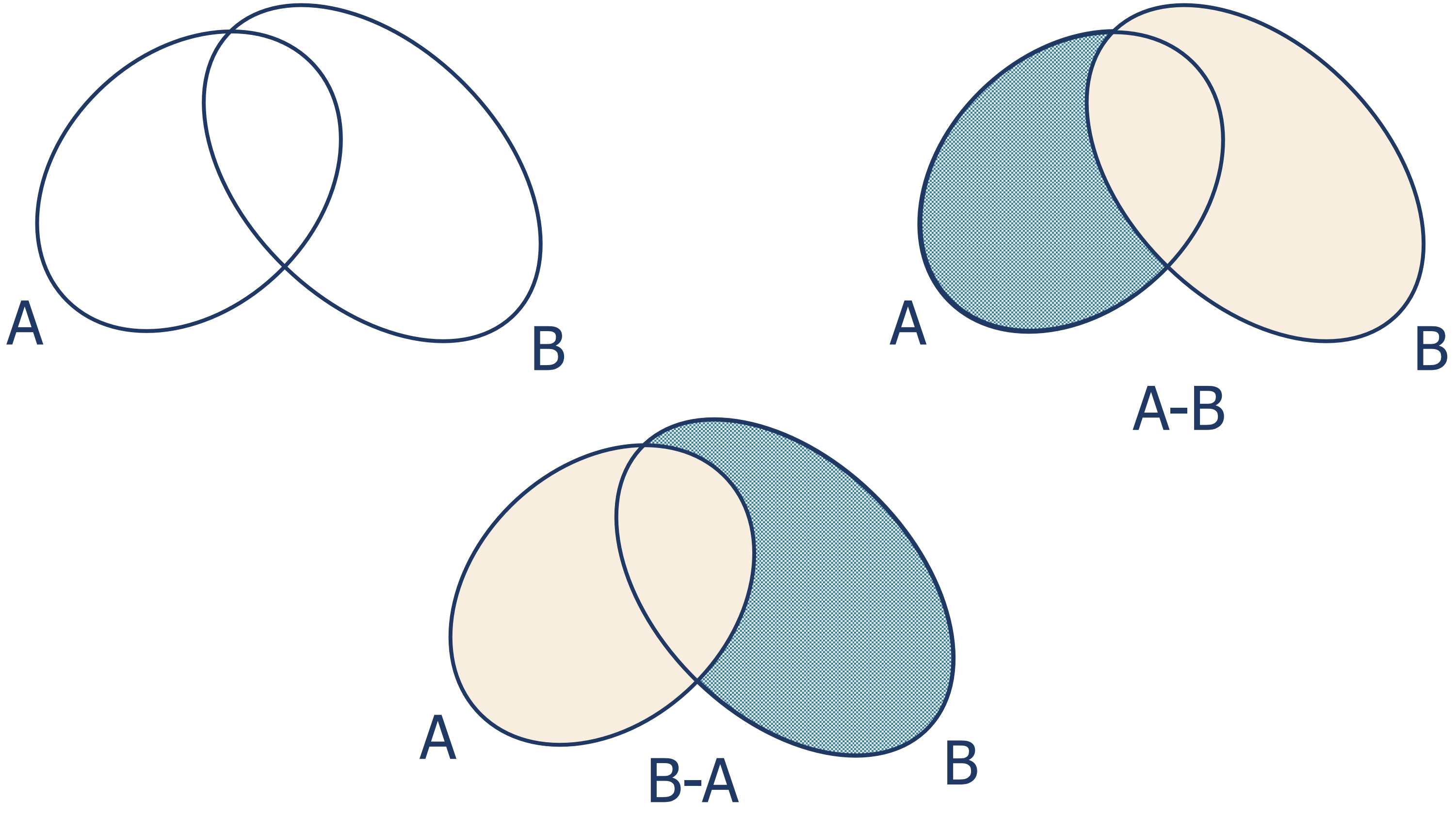
TeacherID	PName	Department
D102	Green	Computer engineering

Difference and anti-join

Relation Algebra

Difference

- The difference of two relations A and B selects all the tuples present *exclusively* in A



A-B≠B-A

Difference: example (n. 1)

- *Find information relative to professors who teach bachelor's degree courses, but not master's degree courses*

BachelorTeachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

MasterTeachers

TeacherID	PName	Department
D102	Green	Computer engineering
D101	Rossi	Department of electrics

TeacherID	PName	Department
D105	Black	Computer engineering
D104	White	Department of electronics

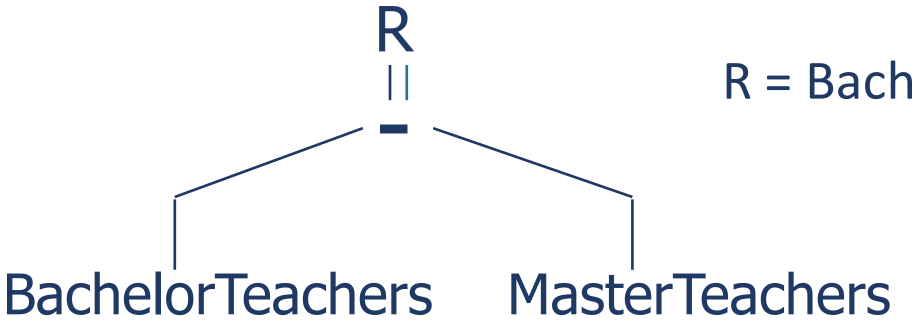
Difference: definition and properties

$$R = A - B$$

- The difference of two relations A and B generates a relation R
  - with the same schema of A and B
  - containing all tuples belonging to A that do not belong to B
- *Compatibility*
  - relations A and B must have the same schema (number and type of attributes)
- The difference *does not satisfy* the commutative property, nor the associative property

Difference: example (n. 1)

- *Find information relative to professors who teach bachelor's degree courses, but not master's degree courses*



R = BachelorTeachers - MasterTeachers

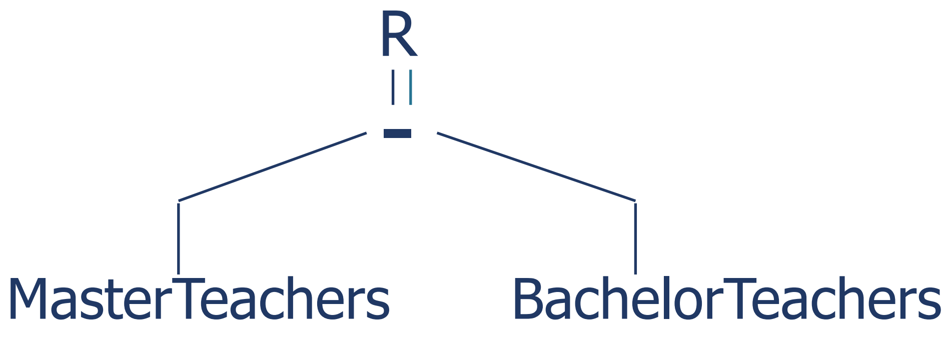
R

TeacherID	PName	Department
D105	Black	Computer engineering
D104	White	Department of electronics

Difference: example (n. 2)

- *Find information relative to professors who teach master's degree courses, but not bachelor's degree courses*

R = MasterTeachers - BachelorTeachers



MasterTeachers

TeacherID	PName	Department
D102	Green	Computer engineering
D101	Rossi	Department of electrics

BachelorTeachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

TeacherID	PName	Department
D101	Rossi	Department of electrics

Difference: example (n. 3)

- *Find identifier, name and department of teachers that are not holding any courses*

Courses

CCode	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

Teachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

Difference: example (n. 3)

- *Find identifier, name and department of teachers that are not holding any courses*

Projection

Teacher identifier

Teachers

TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

Courses

CCode	CName	Semester	TeacherID
M2170	Computer science	1	*D102*
M4880	Digital systems	2	*D104*
F1401	Electronics	1	*D104*
F0410	Databases	2	*D102*

Projection

Identifiers of teachers who hold at least one course

Difference: example (n. 3)

TeacherID
D102
D105
D104

TeacherID
D102
D104

Difference

TeacherID
D105

Natural Join

Teachers

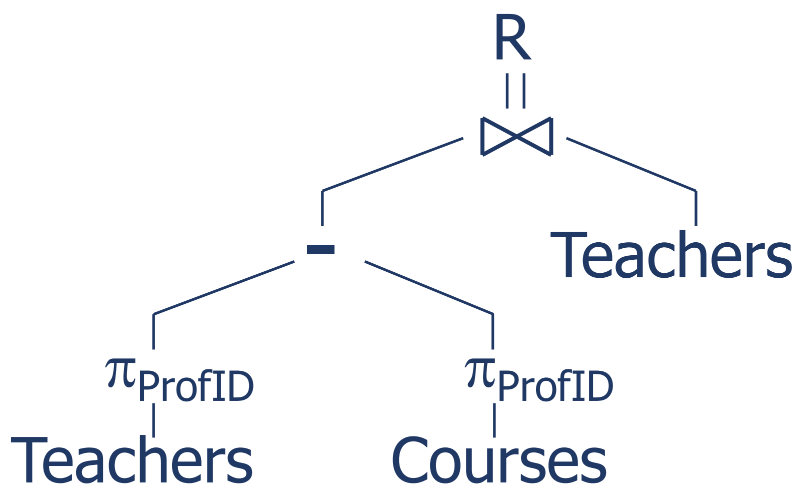
TeacherID	PName	Department
D102	Green	Computer engineering
D105	*Black*	*Computer engineering*
D104	White	Department of electronics

R

TeacherID	PName	Department
D105	*Black*	*Computer engineering*

Difference: example (n. 3)

- *Find identifier, name and department of teachers that are not holding any courses*



$$R = \text{Teachers} \bowtie ((\pi_{\text{ProfID}}\text{Teachers}) - (\pi_{\text{ProfID}}\text{Courses}))$$

Anti-join: definition and properties

$$R = A \overline{\ltimes}_p B$$

- The anti-join of two relations A and B selects all the tuples of A that are *"not semantically linked"* to tuples of B
  - the information of B does not appear in the result
- The anti-join of two relations A and B generates a relation R
  - with the same schema of A
  - containing all the tuples of A for which there is no tuple of B for which the predicate *p* is true
- The predicate *p* is expressed in the same way as for the theta-join and the semi-join
- The anti-join *does not satisfy* the commutative property, nor the associative property

Anti-join: example

- *Find identifier, name and department of teachers that are not holding any courses*

Courses

CCode	CName	Semester	TeacherID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

Teachers

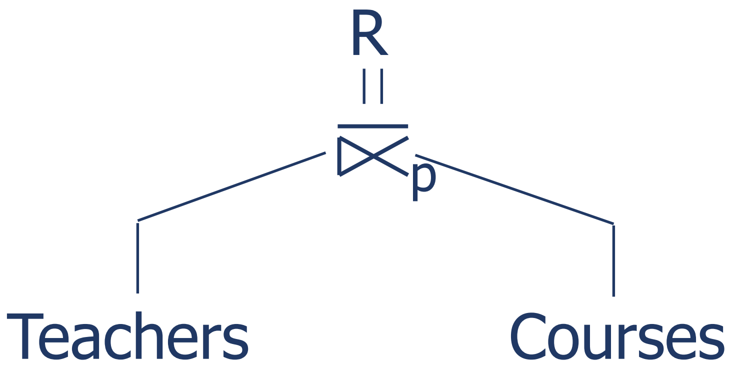
TeacherID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

→

TeacherID	PName	Department
D105	Black	Computer engineering

Anti-join: example

- *Find identifier, name and department of teachers that are not holding any courses*



$R = \text{Teachers} \; \overline{\triangleright}_p \text{Courses}$

p: Teachers.TeacherID=Courses.TeacherID

R

<u>TeacherID</u>	PName	Department
D105	Black	Computer engineering

Division and other operators

Relation Algebra

Division: example

- *Find the students that have passed the exams of all the courses in the first year*

PassedExams

StudentID	CCode
S1	C1
S1	C2
S1	C3
S1	C4
S1	C5
S1	C6
S2	C1
S2	C2
S3	C2
S4	C2
S4	C4
S4	C5

FirstYearCourses

CCode
...
...
...
...

Division: example

- *Find the students that have passed the exams of all the courses in the first year*

PassedExams

StudentID	CCode
S1	C1
S1	C2
S1	C3
S1	C4
S1	C5
S1	C6
S2	C1
S2	C2
S3	C2
S4	C2
S4	C4
S4	C5

FirstYearCourses

CCode
C1

R

StudentID
S1
S2

Division: example

- *Find the students that have passed the exams of all the courses in the first year*

PassedExams

StudentID	CCode
S1	C1
S1	C2
S1	C3
S1	C4
S1	C5
S1	C6
S2	C1
S2	C2
S3	C2
S4	C2
S4	C4
S4	C5

FirstYearCourses

CCode
C2
C4

R

StudentID
S1
S4

Division: example

- *Find the students that have passed the exams of all the courses in the first year*

PassedExams

StudentID	CCode
S1	C1
S1	C2
S1	C3
S1	C4
S1	C5
S1	C6
S2	C1
S2	C2
S3	C2
S4	C2
S4	C4
S4	C5

FirstYearCourses

CCode
C1
C2
C3
C4
C5
C6

R

StudentID
S1

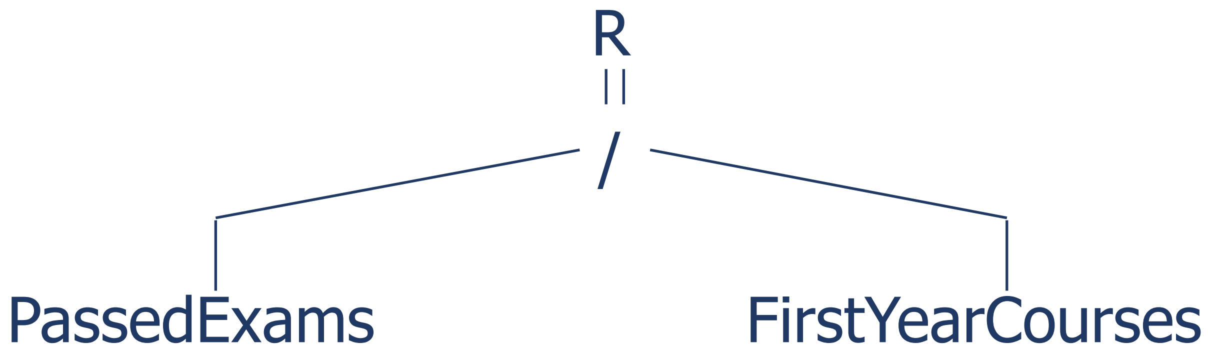
Division: definition and properties

$$R = A / B$$

- The division of relation A by relation B generates a relation R
  - whose schema is *schema(A) - schema(B)*
  - containing all the tuples of A such that for each tuple (Y:y) present in B there is a tuple (X:x, Y:y) in A
- Division *does not satisfy* the commutative property, nor the associative property

Division: example

- *Find the students that have passed the exams of all the courses in the first year*



R = PassedExams / FirstYearCourses

Other operators

- Various other operators have been proposed so as to extend the expressive power of relational algebra
  - extending relations with a new attribute, defined by a scalar expression
    - GROSS_WEIGHT=NET_WEIGHT+TARE
  - calculating aggregate function
    - max, min, avg, count, sum
    - possibly defining subsets in which to group the data (GROUP BY of SQL)