Relational model and relational algebra

Relational algebra

DBMG

Relational Algebra

- Introduction
- Selection and projection
- Cartesian product and join
- Natural join, theta-join and semi-join
- Outer join
- Union and intersection
- Difference and anti join
- Division and other operators

DBMG 2

Relational algebra

Introduction

DBMG

Relational algebra

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

DBMG 4

Relational algebra operators

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

DBMG 5

Relational algebra operators

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

DBMG 6

Example of relations

Courses

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

Professors

<u>ProfID</u>	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

DBMG 7

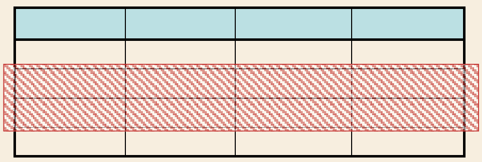
Relational algebra

Selection and projection

DBMG

Selection

- The selection extracts a "horizontal" subset from the relation
  - It operates a horizontal factorisation of the relation



DBMG 9

Selection: example

- *Find the courses held in the second semester*

DBMG 10

Selection: example

Courses

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

R

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

DBMG 11

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 because of which predicate *p* is true
- Predicate *p* is a boolean expression (operators $\wedge, \vee, \neg$) of expressions of comparison between attributes or between attributes and constants
  - p: City= 'Turin' $\wedge$ Age>18
  - p: ReturnDate>DeliveryDate+10

DBMG 12

Selection: example

- *Find the courses held in the second semester*

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

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

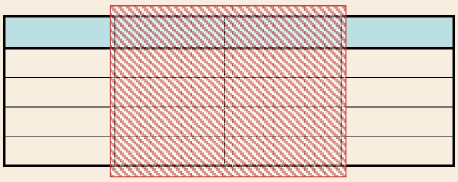
Courses

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

DBMG 13

Projection

- The projection extracts a "vertical" subset from the relation
  - It operates a vertical factorisation of the relation



DBMG 14

Projection: example (n. 1)

- *Find the names of professors*

DBMG 15

Projection: example (n. 1)

Professors

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

DBMG 16

Projection: example (n. 1)

Professors

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

R

PName
Green
Black
White

DBMG 17

Proiection: definition

$R = \pi_L A$

- The projection 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

DBMG 18

Projection: example (n. 1)

- *Find the names of professors*

R
||
π_{PName}
|
Professors

$R = \pi_{PName} Professors$

Professors

<u>ProfID</u>	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

DBMG 19

Projection: example (n. 2)

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

$R = \pi_{Department} Professors$

R
||
$\pi_{Department}$
|
Professors

DBMG 20

Projection: example (n. 2)

Professors

<u>ProfID</u>	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

R

Department
Computer engineering
Department of electronics

DBMG 21

Proiection: definition

$R = \pi_L A$

- The projection 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 caused by the exclusion of the attributes not contained in L are deleted
  - If L includes a candidate key, there are no duplicates

DBMG 22

Selection+projection: example

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

DBMG 23

Selection+projection: example

Courses

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

Selection

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

DBMG

Selection+projection: example

CCode	*CName*	Semester	ProfID
M4880	*Digital systems*	2	D104
F0410	*Databases*	2	D102

↓ Projection

R

CName
Digital systems
Databses

DBMG 25

Selection+projection: example

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

R = $\pi_{CName}(\sigma_{Semester=2}Courses)$

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

Courses

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

DBMG 26

Selection+projection: example (is it correct?)

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

R = $\sigma_{Semester=2}(\pi_{CName}Courses)$

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

Courses

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

DBMG 27

Selection+projection: wrong solution

Courses

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

↓ Projection

CName
Computer science
Digital systems
Electronics
Databses

DBMG 28

Seection+projection: wrong solution

CName
Computer science
Digital systems
Electronics
Databses

- The Semester attribute does not exist any more
  - The information relative to the semester is no longer available
  - The selection operation cannot be carried out

DBMG 29

Selection+projection: wrong solution

- *Find the name of the courses in the second semester*

~~R = $\sigma_{Semester=2}(\pi_{CName}Courses)$~~

~~R || $\sigma_{Semester=2}$ | π_{CName}~~
|
Courses

Courses

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

DBMG 30

Relational algebra

Cartesian product and join

DBMG

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

DBMG 32

Cartesian product: example

- *Find the Cartesian product of courses and professors*

DBMG 33

Cartesian product: example

Courses

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

Professors

<u>ProfID</u>	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics

DBMG 34

Cartesian product: example

R

Courses CCode	Courses. CName	Courses. Semester	Courses. ProfID	Professors. ProfID	Professors .PName	Professors. Departmen
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*

DBMG 35

Cartesian product: example

R

Courses CCode	Courses. CName	Courses. Semester	Courses. ProfID	Professors. ProfID	Professors .PName	Professors. Departmen
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*
...	...	...	...	...	...	...

DBMG 36

Cartesian product: definition

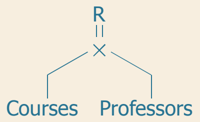
R = A × B

- The Cartesian product 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 formed by a tuple of A and a tuple of B
- The Cartesian product is
  - commutative
    - A × B = B × A
  - associative
    - (A × B) × C = A × (B × C)

DBMG 37

Cartesian product: example

- *Find the Cartesian product of courses and professors*



R = Courses × Professors

DBMG 38

Link between relations

R

Courses CCode	Courses. CName	Courses. Semester	Courses. ProfID	Professors. ProfID	Professors .PName	Professors. Departmen
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*
...	...	...	...	...	...	...

DBMG 39

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"*

DBMG 40

Join: example

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

DBMG 41

Join: example

Courses

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

Professors

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

DBMG 42

Join example

R

Courses CCode	Courses. CName	Courses. Semester	Courses. ProfID	Professors. ProfID	Professors .PName	Professors. Departmen
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*
...	...	...	...	...	...	...

DBMG 43

Join: example

R

Courses CCode	Courses. CName	Courses. Semester	Courses. ProfID	Professors. ProfID	Professors. PName	Professors. 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

DBMG 44

Join: example

R

Courses CCode	Courses. CName	Courses. Semester	Courses. ProfID	Professors. ProfID	Professors. PName	Professors. 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 hold any courses does not appear in the result of the join

DBMG 45

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 the interrogations
- There are different kinds of joins
  - natural join
  - theta-join (and its subcase equi-join)
  - semi-join

DBMG 46

Relational algebra

Natural join, theta-join and semi-join

DBMG

Natural join: definition

R = A⋈B

- The natural join of two relations A and B generates a relation R
  - whose schema is
    - 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

DBMG 48

Natural join: properties

R = A⋈B

- Natural join is commutative and associative

DBMG 49

Natural join: example

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

R
||
⋈
Courses Professors

DBMG 50

Natural join: example

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

R = Courses ⋈ Professors

R
||
⋈
Courses Professors

R

Courses. CCode	Courses. CName	Courses. Semester	ProfID	Professors. PName	Professors. 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

DBMG 51

Natural join: example

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

R = Courses ⋈ Professors

R
||
⋈
Courses Professors

R

Courses. CCode	Courses. CName	Courses. Semester	ProfID	Professors. PName	Professors. 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

DBMG 52

Natural join: example

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

R = Courses ⋈ Professors

R
||
⋈
Courses Professors

R

Courses. CCode	Courses. CName	Courses. Semester	ProfID	Professors. PName	Professors. 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

DBMG 53

Natural join: esempio

R

Courses. CCode	Courses. CName	Courses. Semester	ProfID	Professors. PName	Professors. 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

- *NB:* The common attribute ProfID (Professor Identifier) is present only once in the schema of the resulting relation R

DBMG 54

Theta-join

- 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"*

Theta-join: esxample

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

Theta-join: example

Courses C1

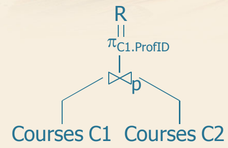
<u>CCode</u>	CName	Semester	ProfID
M2170	Computer science	1	D102
M4880	Digital systems	2	D104
F1401	Electronics	1	D104
F0410	Databases	2	D102

Courses C2

<u>CCode</u>	CName	Semester	ProfID
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 professors that hold at least two courses*



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

R = $\pi_{C1.ProfID}$((Courses C1) $\bowtie_p$ (Courses C2))

Theta-join: example

Courses C1. CCode	Courses C1. CName	Courses C1. Semester	Courses C1. ProfID	Courses C2. CCode	Courses C2. CName	Courses C2. Semester	Courses C2. ProfID
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	Electronins	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
…	…	…	…	…	…	…	…

Theta-join: example

Courses C1. CCode	Courses C1. CName	Courses C1. Semester	Cources C1. ProfID	Courses C2. CCode	Courses C2. CName	Courses C2. Semester	Courses C2. ProfID
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

R

Courses C1.ProfID
D102
D104

Theta-join: definition

$R = A \bowtie_p B$

- The theta-join of two relations A and B generates a relation R
  - whose schema is the union of the schemes 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
- The theta-join is commutative and associative

DBMG 61

Equi-join: definition

$R = A \bowtie_p B$

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

DBMG 62

Semi-join

- The semi-join of two relations A and B selects all the tuples of A that are *"semantically linked"* to at least a tuple of B
  - the information of B does not appear in the result

DBMG 63

Semi-join: example

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

DBMG 64

Semi-join: example

Courses

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

Professors

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

DBMG 65

Semi-join: example

Professors. ProfID	Professors. PName	Professors. Department	Courses. CCode	Courses. CName	Courses. Semester	Courses. ProfID
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 66

Semi-join: example

Professors. ProfID	*Professors. PName*	*Professors. Department*	Courses. CCode	Courses. CName	Courses. Semester	Courses. ProfID
D102	*Green*	*Computer engineering*	M2170	Computer science	1	D102
D102	*Green*	*Icomputer engineering*	F0410	Databases	2	D102
D104	*White*	*Department of electronics*	M4880	Digital systems	2	D104
D104	*White*	*Elettronica*	F1401	Electronics	3	D104

R

Professors. ProfID	Professors. PName	Professors. Department
D102	Green	Computer engineering
D104	White	Department of electronics

67

Semi-join: definition

$R = A \ltimes_p B$

- 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 of B)

68

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

69

Semi-join: example

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

$R = Professors \ltimes_p Courses$

R = ⋉p (Professors, Courses)

p: Professors.ProfID=Courses.ProfID

R

Professors. ProfID	Professors. PName	Professors. Department
D102	Green	Computer engineering
D104	White	Department of electronics

70

Relational algebra

Outer join

Outer-join

- Version of join that allows us to conserve 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

72

Left outer-join

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

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

73

Left outer-join: example

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

74

Left outer-join: example

Courses

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

Professors

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

75

Left outer-join: example

R

Professors. ProfID	Professors. PName	Professors. Department	Courses. CCode	Courses. CName	Courses. Semester	Courses. ProfID
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*	*Department of electronics*	*F1401*	*Electronics*	*1*	*D104*

76

Left outer-join: example

R

Professors. ProfID	Professors. PName	Professors. Department	Courses. CCode	Courses. CName	Courses. Semester	Courses. ProfID
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	Department of electronics	F1401	Electronics	1	D104
D105	*Black*	*Computer engineering*	*null*	*null*	*null*	*null*

77

Left outer-join: definition

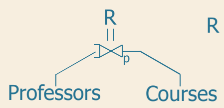
$$R = A \bowtie_p B$$

- 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 tuples of B completed with null values for all of the attributes of B
- The left outer-join *is not* commutative

78

Left outer-join: example

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



R = Professors ⟕$_p$ Courses

p: Professors.ProfID=Courses.ProfID

DBMG

Left outer-join: example

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

R

Professors. ProfID	Professors. PName	Professors. Department	Courses. CCode	Courses. CName	Courses. Semester	Courses. ProfID
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	Department of electronics	F1401	Electronics	1	D104
D105	Black	Computer engineering	null	null	null	null

DBMG

Right outer-join: definition

R = A ⟖$_p$ 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 tuples of A completed with null values for all of the attributes of A
- Il right outer-join *is not* commutative

DBMG 81

Full outer-join: definition

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

DBMG 82

Full outer-join: definition

R = A ⟗$_p$ B

- The full outer-join of two relations A and B generates the relation R
  - 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 tuples of B completed with null values for all of the attributes of B
    - a tuple of B that is not correlated by means of the predicate *p* to tuples of A completed with null values for all of the attributes of A

DBMG 83

Full outer-join: properties

R = A ⟗$_p$ B

- The full outer-join is commutative

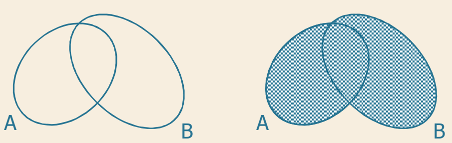
DBMG 84

Relational algebra

Union and intersection

Union

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



86

Union: example of relations

DegreeCourseProf

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

MasterCourseProf

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

87

Union: example

- *Find information relative to the professors of degree courses or master's degrees*

88

Union: example

DegreeCourseProf

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

MasterCourseProf

ProfID	PName	Department
D102	*Green*	*Computer engineering*
D101	*Red*	*Department of electrics*

R

ProfID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics
D101	Red	Department of electrics

- *NB:* Duplicated tuples are deleted

89

Union: definition

$R = A \cup B$

- 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 to both)
- *Compatibility*
  - the relations A and B have to have the same schema (number and kind of attributes)
- Duplicated tuples are deleted
- The union is commutative and associative

90

Union: example

- *Find information relative to the professors of degree courses or master's degrees*

R = DegreeCourseProf ∪ MasterCourseProf

DegreeCoursesProf MasterCourseProf

R

ProfID	PName	Department
D102	Green	Computer engineering
D105	Black	Computer engineering
D104	White	Department of electronics
D101	Red	Department of electrics

91

Intersection: example of relations

DegreeCourseProf

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

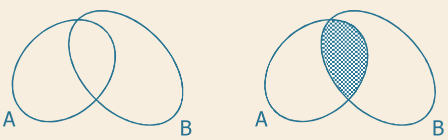
MasterCourseProf

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

94

Intersection

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



92

Intersection: example

DegreeCourseProf

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

MasterCourseProf

ProfID	PName	Department
D102	*Green*	*Computer engineering*
D101	Red	Department of electrics

R

ProfID	PName	Department
D102	Green	Computer engineering

95

Intersection: example

- *Find information relative to professors teaching both degree courses and master's*

Intersection: definition

R = A ∩ B

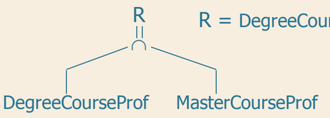
- The intersection of two relations A and B generates a relation R
  - with the same schema of A and B
  - containing all the tuples belonging to both A and B
- *Compatibility*
  - relations A and B must have the same schema (number and type of attributes)
- Intersection is commutative and associative

96

Intersection: example

- *Find information relative to professors teaching both degree courses and master's*

R = DegreeCourseProf ∩ MasterCourseProf



R

ProfID	PName	Department
D102	Green	Computer engineering

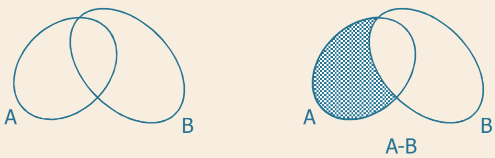
97

Relational algebra

Difference and anti-join

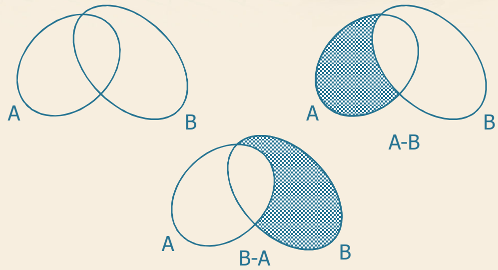
Difference

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



99

Difference



A-B≠B-A

100

Difference: example (n.1)

- *Find the professors teaching degree courses but not master's*

101

Difference: example (n.1)

DegreeCourseProf

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

MasterCourseProf

ProfID	PName	Department
D102	Green	Computer engineering
D101	Red	Department of electrics

R

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

102

Difference: definition

R = A - B

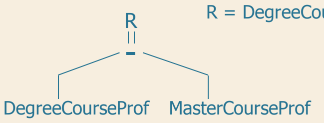
- The difference of two relations A and B generates a relation R
  - with the same schema of A and B
  - containing all tuples belonging to A that do not belong to B
- *Compatibility*
  - relations A and B must have the same schema (number and type of attributes)
- The difference *does not satisfy* the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

DBMG 103

Difference: example (n.1)

- *Find the professors teaching degree courses but not master's*

R = DegreeCourseProf - MasterCourseProf



R

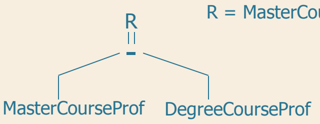
ProfID	PName	Department
D105	Black	Computer engineering
D104	White	Department of electronics

DBMG 104

Difference: example (n. 2)

- *Find the professors teaching master courses but not degree's*

R = MasterCourseProf - DegreeCourseProf



DBMG 105

Difference: example (n. 2)

MasterCourseProf

ProfID	PName	Department
D102	Green	Computer engineering
D101	*Red*	*Department of electrics*

DegreeCourseProf

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

R

ProfID	PName	Department
D101	Rossi	Department of electrics

DBMG 106

Difference: example (n. 3)

Courses

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

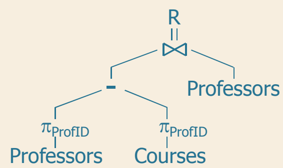
Professors

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

DBMG 107

Difference: example (n. 3)

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



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

DBMG 108

Difference: example (n. 3)

Professors

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

Professor Identifiers

Courses

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

Identifiers of professors that holds at least a course

109

Difference: example (n. 3)

ProfID
D102
D105
D104

ProfID
D102
D104

Difference

ProfID
D105

110

Difference: example (n. 3)

ProfID
D105

Professors

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

Natural Join

R

ProfID	PName	Department
D105	Black	Computer engineering

111

Anti-join

- 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

112

Anti-join: example

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

113

Anti-join: example

Professors

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

Courses

CCode	CName	Semester	ProfID
M2170	Informatica 1	1	D102
M4880	Sistemi digitali	2	D104
F1401	Elettronica	1	D104
F0410	Basi di dati	2	D102

R

ProfID	PName	Department
D105	Black	Computer engineering

114

Anti-join: definition

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

- 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

DBMG 115

Anti-join: example

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

R = Professors $\overline{\ltimes}_p$ Courses

Professors Courses

p: Professors.ProfID=Courses.ProfID

R

ProfID	PName	Department
D105	Black	Computer engineering

DBMG 116

Relational algebra

Division and other operators

DBMG

Division: example

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

PassedExams

StudentID	CCourse
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

CCourse
...
...
...
...

DBMG 118

Division: example

PassedExams

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

FirstYearsCourses

CCourse
C1

R

StudentID
S1
S2

DBMG 119

Division: example (n. 2)

PassedExams

StudentID	CCourse
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

CCourse
C2
C4

R

StudentID
S1
S4

DBMG 120

Division: example (n. 3)

PassedExams

StudentID	CCourse
S1	*C1*
S1	*C2*
S1	*C3*
S1	*C4*
S1	*C5*
S1	*C6*
S2	C1
S2	C2
S3	C2
S4	C2
S4	C4
S4	C5

FirstYearCourse

CCourse
C1
C2
C3
C4
C5
C6

R

StudentID
S1

121

Division: definition

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

122

Division: example

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

R
=
/
PassedExams FirstYearCourses

R = PassedExams / FirstYearCourses

123

Other operators

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

124