Basi di Dati e
Sistemi Informativi I

Relational algebra

(1/2)

Query languages for relational databases

- Operations on databases:
  - queries: "read" data from the database
  - updates: change the content of the database
- Both can be modeled as functions from databases to databases
- Foundations can be studied with reference to query languages:
  - relational algebra, a "procedural" language
  - relational calculus, a "declarative" language
  - (briefly) Datalog, a more powerful language
- Then, we will study SQL, a practical language (with declarative and procedural features) for queries and updates

Relational algebra

- A collection of operators that
  - are defined on relations
  - produce relations as results

  and therefore can be combined to form complex expressions
- Operators
  - union, intersection, difference
  - renaming
  - selection
  - projection
  - join (natural join, cartesian product, theta-join)

Union, intersection, difference

- Relations are sets, so we can apply set operators
- However, we want the results to be relations (that is, homogeneous sets of tuples)
- Therefore:
  - it is meaningful to apply union, intersection, difference only to pairs of relations defined over the same attributes

Union

Graduates

Number	Surname	Age
7274	Robinson	37
7432	O'Malley	39
9824	Darkes	38

Managers

Number	Surname	Age
9297	O'Malley	56
7432	O'Malley	39
9824	Darkes	38

Graduates $\cup$ Managers

Number	Surname	Age
7274	Robinson	37
7432	O'Malley	39
9824	Darkes	38
9297	O'Malley	56

Intersection

Graduates

Number	Surname	Age
7274	Robinson	37
7432	O'Malley	39
9824	Darkes	38

Managers

Number	Surname	Age
9297	O'Malley	56
7432	O'Malley	39
9824	Darkes	38

Graduates $\cap$ Managers

Number	Surname	Age
7432	O'Malley	39
9824	Darkes	38

Difference

Graduates

Number	Surname	Age
7274	Robinson	37
7432	O'Malley	39
9824	Darkes	38

Managers

Number	Surname	Age
9297	O'Malley	56
7432	O'Malley	39
9824	Darkes	38

Graduates - Managers

Number	Surname	Age
7274	Robinson	37

A meaningful but impossible union

Paternity

Father	Child
Adam	Cain
Adam	Abel
Abraham	Isaac
Abraham	Ishmael

Maternity

Mother	Child
Eve	Cain
Eve	Seth
Sarah	Isaac
Hagar	Ishmael

Paternity $\cup$ Maternity **???**

- the problem: Father and Mother are different names, but both represent a "Parent"
- the solution: rename attributes

Renaming

- unary operator
- "changes attribute names" without changing values
- removes the limitations associated with set operators
- notation:
  - $\rho_{Y \leftarrow X}(r)$
- example:
  - $\rho_{\text{Parent} \leftarrow \text{Father}}(\text{Paternity})$
- if there are two or more attributes involved then ordering is meaningful:
  - $\rho_{\text{Location,Pay} \leftarrow \text{Branch,Salary}}(\text{Employees})$

Renaming, example

Paternity

Father	Child
Adam	Cain
Adam	Abel
Abraham	Isaac
Abraham	Ishmael

$\rho_{\text{Parent} \leftarrow \text{Father}.}$(Paternity)

Parent	Child
Adam	Cain
Adam	Abel
Abraham	Isaac
Abraham	Ishmael

Renaming and union

Paternity

Father	Child
Adam	Cain
Adam	Abel
Abraham	Isaac
Abraham	Ishmael

Maternity

Mother	Child
Eve	Cain
Eve	Seth
Sarah	Isaac
Hagar	Ishmael

$\rho_{\text{Parent} \leftarrow \text{Father}}(\text{Paternity}) \cup \rho_{\text{Parent} \leftarrow \text{Mother}}(\text{Maternity})$

Parent	Child
Adam	Cain
Adam	Abel
Abraham	Isaac
Abraham	Ishmael
Eve	Cain
Eve	Seth
Sarah	Isaac
Hagar	Ishmael

Renaming and union, with more attributes

Employees

Surname	Branch	Salary
Patterson	Rome	45
Trumble	London	53

Staff

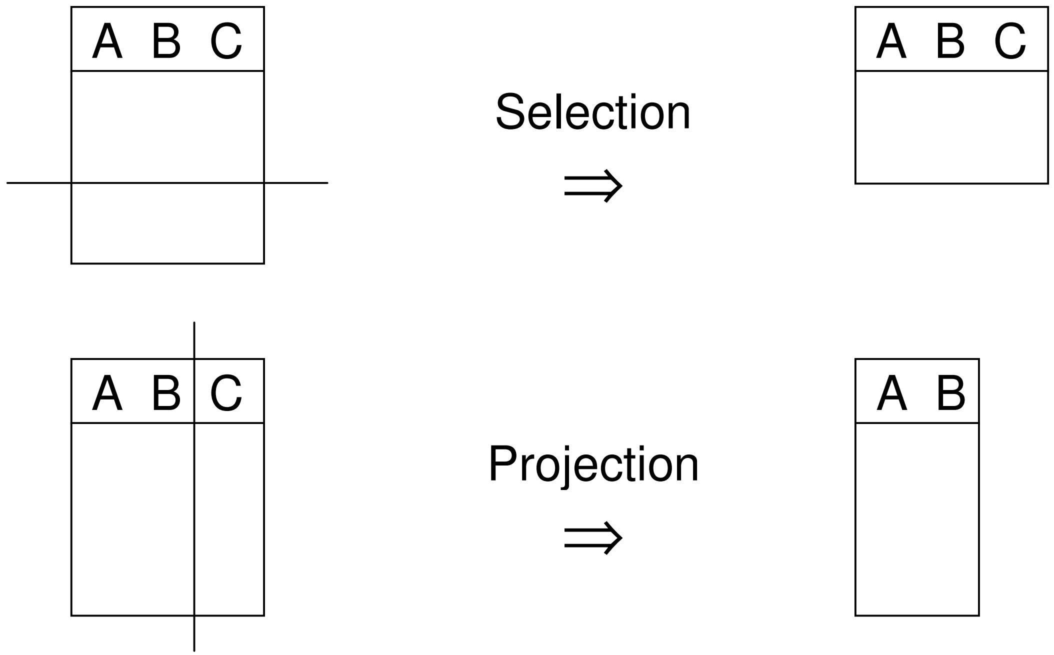
Surname	Factory	Wages
Patterson	Rome	45
Trumble	London	53

$$\rho_{\text{Location,Pay} \leftarrow \text{Branch,Salary}}(\text{Employees}) \cup \rho_{\text{Location,Pay} \leftarrow \text{Factory, Wages}}(\text{Staff})$$

Surname	Location	Pay
Patterson	Rome	45
Trumble	London	53
Cooke	Chicago	33
Bush	Monza	32

Selection and projection

- Two unary operators, in a sense orthogonal:
  - selection for "horizontal" decompositions
  - projection for "vertical" decompositions



Selection

- Produce results
  - with the same schema as the operand
  - with a subset of the tuples (those that satisfy a condition)
- Notation:
  - $\sigma_F(r)$
- Semantics:
  - $\sigma_F(r) = \{ t \mid t \in r$ and t satisfies $F\}$

Selection, example

Employees

Surname	FirstName	Age	Salary
Smith	Mary	25	2000
Black	Lucy	40	3000
Verdi	Nico	36	4500
Smith	Mark	40	3900

$\sigma_{Age<30 \vee Salary>4000}$ (Employees)

Surname	FirstName	Age	Salary
Smith	Mary	25	2000
Verdi	Nico	36	4500

Selection, another example

Citizens

Surname	FirstName	PlaceOfBirth	Residence
Smith	Mary	Rome	Milan
Black	Lucy	Rome	Rome
Verdi	Nico	Florence	Florence
Smith	Mark	Naples	Florence

$\sigma_{\text{PlaceOfBirth=Residence}}$ (Citizens)

Surname	FirstName	PlaceOfBirth	Residence
Black	Lucy	Rome	Rome
Verdi	Nico	Florence	Florence

Projection

- Produce results
  - over a subset of the attributes of the operand
  - with values from all its tuples
- Notation (given a relation r(X) and a subset Y of X):
  - $\pi_Y(r)$
- Semantics:
  - $\pi_Y(r) = \{ t[Y] \mid t \in r \}$

Projection, example

Employees

Surname	FirstName	Department	Head
Smith	Mary	Sales	De Rossi
Black	Lucy	Sales	De Rossi
Verdi	Mary	Personnel	Fox
Smith	Mark	Personnel	Fox

$\pi_{\text{Surname, FirstName}}$(Employees)

Surname	FirstName
Smith	Mary
Black	Lucy
Verdi	Mary
Smith	Mark

Projection, another example

Employees

Surname	FirstName	Department	Head
Smith	Mary	Sales	De Rossi
Black	Lucy	Sales	De Rossi
Verdi	Mary	Personnel	Fox
Smith	Mark	Personnel	Fox

$\pi_{\text{Department, Head}}$ (Employees)

Department	Head
Sales	De Rossi
Personnel	Fox

Cardinality of projections

- The result of a projections contains at most as many tuples as the operand
- It can contain fewer, if several tuples "collapse"
- $\pi_Y(r)$ contains as many tuples as r if and only if Y is a superkey for r;
  - this holds also if Y is "by chance" (not defined as a superkey in the schema, but superkey for the specific instance), see the example

Tuples that collapse

Students

RegNum	Surname	FirstName	BirthDate	DegreeProg
284328	Smith	Luigi	29/04/59	Computing
296328	Smith	John	29/04/59	Computing
587614	Smith	Lucy	01/05/61	Engineering
934856	Black	Lucy	01/05/61	Fine Art
965536	Black	Lucy	05/03/58	Fine Art

$\pi_{\text{Surname, DegreeProg}}$ (Students)

Surname	DegreeProg
Smith	Computing
Smith	Engineering
Black	Fine Art

Tuples that do not collapse, "by chance"

Students

RegNum	Surname	FirstName	BirthDate	DegreeProg
296328	Smith	John	29/04/59	Computing
587614	Smith	Lucy	01/05/61	Engineering
934856	Black	Lucy	01/05/61	Fine Art
965536	Black	Lucy	05/03/58	Engineering

$\pi_{\text{Surname, DegreeProg}}$ (Students)

Surname	DegreeProg
Smith	Computing
Smith	Engineering
Black	Fine Art
Black	Engineering