ĐẠI HỌC BÁCH KHOA

25 YEARS ANNIVERSARY

SOICT

ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

IT3090E - Databases

Chapter 3: Relational Algebra

Muriel VISANI
murielv@soict.hust.edu.vn

Contents

Learning points

- 1. Simple relational algebra operators
- 2. Composite relational algebra operators

Learning objectives

- Upon completion of this lesson, students will be able to:
  - Master relational algebra.
  - Translate a query written in English into relational algebra.

What is relational algebra?

Set of operations for handling tables (*i.e.* a set of records).

A query is an algebrical expression which is applied on a set of tables and products a final table (result).

Relational algebra is composed of a set of both UNARY and BINARY operators

Selection, projection, cartesian product, union, difference.

σ π $\times$ $\cup$ $-$

Selection

- *SELECTION (unary operator)*

Example: I want to extract from the table PRODUCTS only the records which unit price is ≥ 155€.

reference	designation	unit_price	producer
139	Lamp rhapsody	30.00	1623
248	Female shoes camargue	75.00	1623
258	Male shoes camargue	87.00	1623
416	Backpack dolpo	100.00	1369
426	Backpack nepal	155.00	1369
765	tent	300.00	1502

Selection corresponds to: $R1 = \sigma_{unit\_price>=155}(PRODUCTS)$

reference	designation	unit_price	producer
426	Backpack nepal	155.00	1369
765	Tent	300.00	1502

Projection

•*PROJECTION (unary operator)*

Projection $\pi_{a1,a2,a3,a4,\ldots ak}(R)$ applies to a table R and extracts only the attributes a1,a2,a3,…ak from that table.

Therefore, contrary to the selection, the columns (and not the lines) are suppressed

PRODUCTS	reference	designation
	139	Lamp rhapsody
	248	Female shoes camargue
	258	Male shoes camargue
	416	Backpack dolpo
	426	Backpack nepal
	765	tent

$$R2 = \pi_{\text{reference, designation}}(\text{PRODUCTS})$$

Composition of selection and projection

- *SELECTION + PROJECTION*

$R1 = \sigma_{unit\_price>=155}(PRODUCTS)$

$R2 = \pi_{reference,\ designation}(R1)$

TABLE PRODUCTS

reference	designation	unit_price	producer
139	Lamp rhapsody	30.00	1623
248	Female shoes camargue	75.00	1623
258	Male shoes camargue	87.00	1623
416	Backpack dolpo	100.00	1369
426	Backpack nepal	155.00	1369
765	tent	300.00	1502

TABLE R2

reference	designation
426	Backpack népal
765	Tent

Cartesian product

•*CARTESIAN PRODUCT (binary operator)*

The first and most important binary operator is the cartesian product x. The cartesian product between two tables R and S is denoted by: R x S and creates a new table where each record from R is associated to each record de S.

Given R=PRODUCTS and S =PRODUCERS

reference	designation	unit_price	producer
139	Lamp rhapsody	30.00	1623
248	Female shoes camargue	75.00	1623
258	Male shoes camargue	87.00	1623
416	Backpack dolpo	100.00	1369
426	Backpack nepal	155.00	1369
765	tent	300.00	1502

Producer_id	Name	address
1369	denecker sarl	Lyon
1370	chen'alpe diffussion	Lyon
1502	rodenas francois	Toulouse
1623	adidas	Dettwiller

Beware of the attributes which share the same name !!!!

CARTESIAN PRODUCT result

Producer_id	Name	address	reference	designation	unit_price	producer
1369	denecker sarl	Lyon	139	Lamp rhapsody	30.00	1623
1369	denecker sarl	Lyon	248	Female shoes camargue	75.00	1623
1369	denecker sarl	Lyon	258	Male shoes camargue	87.00	1623
1369	denecker sarl	Lyon	416	Backpack dolpo	100.00	1369
1369	denecker sarl	Lyon	426	Backpack nepal	155.00	1369
1369	denecker sarl	Lyon	765	tent	300.00	1502
1370	chen'alpe diffussion	Lyon	139	Lamp rhapsody	30.00	1623
1370	chen'alpe diffussion	Lyon	248	Female shoes camargue	75.00	1623
1370	chen'alpe diffussion	Lyon	258	Male shoes camargue	87.00	1623
1370	chen'alpe diffussion	Lyon	416	Backpack dolpo	100.00	1369
1370	chen'alpe diffussion	Lyon	426	Backpack nepal	155.00	1369
1370	chen'alpe diffussion	Lyon	765	tent	300.00	1502
1502	rodenas francois	Toulouse	139	Lamp rhapsody	30.00	1623
1502	rodenas francois	Toulouse	248	Female shoes camargue	75.00	1623
1502	rodenas francois	Toulouse	258	Male shoes camargue	87.00	1623
1502	rodenas francois	Toulouse	416	Backpack dolpo	100.00	1369
1502	rodenas francois	Toulouse	426	Backpack nepal	155.00	1369
1502	rodenas francois	Toulouse	765	tent	300.00	1502
1623	adidas	Dettwiller	139	Lamp rhapsody	30.00	1623
1623	adidas	Dettwiller	248	Female shoes camargue	75.00	1623
1623	adidas	Dettwiller	258	Male shoes camargue	87.00	1623
1623	adidas	Dettwiller	416	Backpack dolpo	100.00	1369
1623	adidas	Dettwiller	426	Backpack nepal	155.00	1369
1623	adidas	Dettwiller	765	tent	300.00	1502

Composition of Cartesian product and selection: INNER JOIN

- *CARTESIAN PRODUCT + SELECTION*

  R1= PRODUCTS x PRODUCER

  R2 = $\sigma_{producer=producer\_id}$ (R1)

Producer_id	Name	Address	Reference	designation	Unit_price	producer
1369	denecker sarl	Lyon	416	backpack dolpo	100.00	1369
1369	denecker sarl	Lyon	426	backpack nepal	155.00	1369
1502	rodenas francois	Toulouse	765	tent	300.00	1502
1623	adidas	Dettwiller	139	lamp rhapsody	30.00	1623
1623	adidas	Dettwiller	248	Female shoes camargue	75.00	1623
1623	adidas	Dettwiller	258	Male shoes camargue	87.00	1623

INNER JOIN: cartesian product, then selection based on the equality of 2 attributes (here the foreign key of PRODUCTS and the primary key of PRODUCER)

Composition of Cartesian product and selection: INNER JOIN

- *EQUALITY JOIN (or join)*

R= Products (producer_id) ▶◀ Producers (producer)

where ▶◀ is the join between 2 atttributes.

The attributes must share the same type…

•But they are not necessarily linked by a foreign key

Join is a fundamental operation of the queries, for:
- creating links between different tables
- projecting attributes from different tables
- selecting records using criteria concerning different tables

Composition of INNER JOIN and projection

- *JOIN + PROJECTION*

What are the references and unit prices of the products, for each producer?

raison_sociale	ref_produit	prix_unitaire
denecker sarl	416	100.00
denecker sarl	426	155.00
rodenas francois	765	300.00
adidas	139	30.00
adidas	248	75.00
adidas	258	87.00

R1= Products (producer) ▶◀ Producers (producer_id)

R2= $\pi_{\text{name, reference,unit\_price}}$(R1)

Extracts only the chosen attributes from the result of the join.

Composition of INNER JOIN and selection

- *JOIN + SELECTION*

Example: What are the products produced by Adidas?

Producer_id	Name	Address	Reference	designation	Unit_price	producer
1623	adidas	Dettwiller	139	lamp rhapsody	30.00	1623
1623	adidas	Dettwiller	248	Female shoes camargue	75.00	1623
1623	adidas	Dettwiller	258	Male shoes camargue	87.00	1623

R1= Products (producer) ▶◀ Producers (producer_id)

R2 = $\sigma_{\text{Producer.name="Adidas"}}$ (R1)

Only the records for which the producer’s name is «Adidas » are extracted.

Union

- *UNION*

Example: What are the products which unit price is superior to 155€, or which are produced by Adidas?

Union is an INCLUSIVE OR.

Constraint: both tables in the UNION must share the same scheme:

- same number of attributes,
- in the same order,
- same types (definition domains)

Union

- *UNION*

In our example, the schemes of the two tables are different => we first have to build two tables with the same scheme

- Products which price >=155: *SELECTION*.

  $R1 = \sigma_{unit\_price>=155}(PRODUCTS)$

- Products which are produced by Adidas: *JOIN + SELECTION*

  R3=Products (producer) ⋈ Producers (producer_id)

- Selection of the products produced by Adidas

  $R4 = \sigma_{name='Adidas'}(R3)$

- Building the same scheme:

  $R5 = \pi_{reference,designation}(R1)$

  $R6 = \pi_{reference,designation}(R4)$

- Union of the two tables:

  $R7 = R5 \cup R6$

Difference

- *DIFFERENCE*

Like union, difference is applied on two tables with the same schema.
The result of R-S id the set of records from R which are not in S.

<u>Difference is the only operator capable of expressing queries including a NEGATION.</u>

Difference

- *DIFFERENCE*

Example: Which products have never been ordered.

We consider that the table ORDER_LINES contains all the ordered products.

- Building the list of products:

  R1 = $\pi_{\text{reference}}$ (PRODUCTS).

- Building the list of the ordered products:

  R2 = $\pi_{\text{reference}}$ (ORDER_LINES)

- Building the difference on identical schemes:

  R3 = R1 - R2

Intersection

- *INTERSECTION*

Example: Which products in our procduct table have been ordered. We consider that the table ORDER_LINES contains all the ordered products.

- Building the list of products:

  $R1 = \pi_{\text{reference}}(\text{PRODUCTS})$.

- Building the list of the ordered products:

  $R2 = \pi_{\text{reference}}(\text{ORDER\_LINES})$

- Building the intersect on identical schemes:

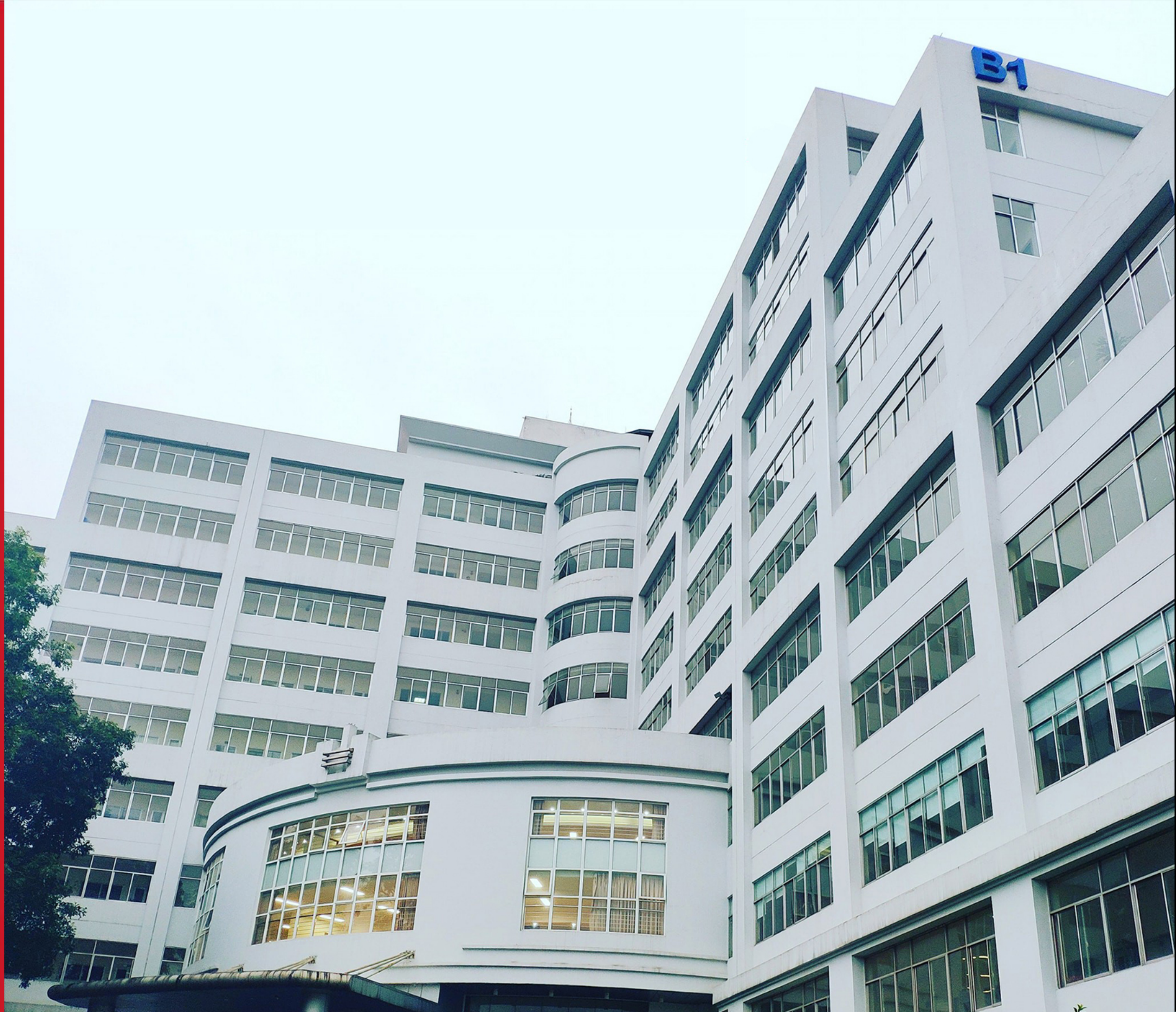
  $R3 = R1 \cap R2$

Exercises

- *Exercises: exercise list n°1*

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Thank you for your attention!



Questions