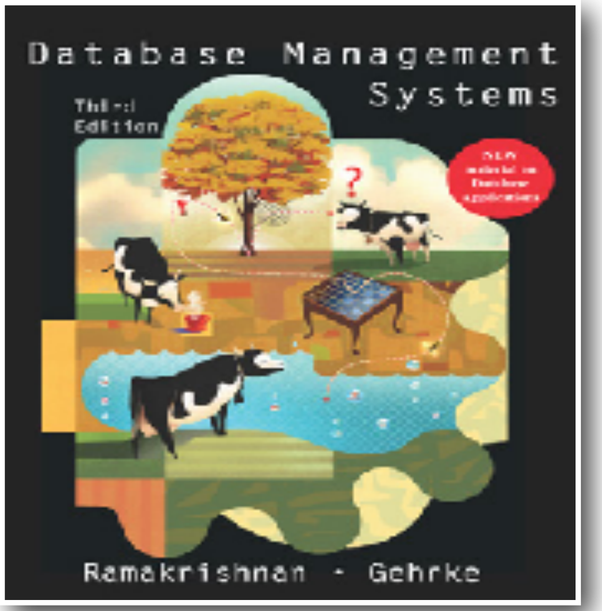
TÉCNICO LISBOA

Bases de Dados

Aula 08: Álgebra Relacional

Prof. Paulo Carreira

Bibliografia



Capítulo 4

Relational Algebra

https://www.youtube.com/watch?v=yPVIBjf-Ilk

Relação: Definição Formal (revisitada)

- Dado um **esquema de relação** R com atributos $A_1, A_2, \ldots, A_n$ e domínios $D_1, \ldots D_n$ uma **relação** *r* é um conjunto

  $$r \subseteq D_1 \times \ldots \times D_n$$

portanto

- Um **exemplar da relação** *r (ou simplesmente* ***relação****)* é um conjunto de *n*-tuplos

  $$(a_1, \ldots, a_n) \text{ onde cada } a_i \in D_i$$

Consulta, Interrogação ou Query

- Uma base de dados é uma **coleção de relações**
- Uma **consulta** à base de dados resume-se a aplicar operações a relações para obter uma relação com o resultado

Query Languages

- Domain Specific Language to express information requests (from a database.)
- Categories of languages
  - **Imperative**
  - Non-procedural, or **declarative**
- "Pure" relational languages:
  - **Relational Algebra**
  - **Tuple Relational Calculus**
  - **Domain Relational Calculus**
- Pure languages form underlying basis of query languages used in Computer Science

Consultas (*queries*)

- Uma consulta é uma expressão relacional expressa na seguinte gramática:

$E ::=$

$\varnothing \mid \{\langle v_1,\ldots,v_i\rangle, \ldots\}$	Relação vazia, Relação literal
$\mid r$	Nome de relação
$\mid \sigma_c(E) \mid \pi_{A1,..,An}(E)$	Seleção, Projeção
$\mid E_1 \cup E_2 \mid E_1 \cap E_2 \mid E_1 - E_2$	União, Intersecção e Diferença
$\mid E_1 \times E_2 \mid E_1 \div E_2$	Produto Cartesiano, Divisão
$\mid E_1 \bowtie E_2$	Junção Natural (Cruzamento)
$\mid \rho_{A1\mapsto B1,\ldots,Am\mapsto Bm}(E) \mid \pi_{F1,..,Fk}(E)$	Renomeação, Projeção Generalizada
$\mid G_{L,F}(E)$	Agrupamento

Interrogação (*query*)

- A Álgebra Relacional tem a propriedade do **fecho** em que
  - **Entrada:** uma ou várias relações
  - **Saída**: relação
- Avaliada sobre instâncias das relações (tabelas) de entrada e produz uma relação (tabela) de saída
- A escrita de uma interrogação é
  - dependente do esquema de entrada e saída
  - mas é independente das instâncias existentes na relação

Bank Database

Account(account_number, branch-name, balance

Depositor(customer_name, account_number)

- account-number: FK(Account)

Loan(loan_number, branch_name, amount)

Borrower(customer_name, loan_number)

- loan-number: FK(Loan)

Fundamental Relational Set Operations

Union

Notation: $r \cup s$

Defined as:

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

For $r \cup s$ to be valid:

- r and s must have the same arity (i.e., same number of attributes)
- The attribute domains must be compatible (e.g., the nth column of r deals with the same type of values as does the nth column of s)

Example: *Find all customers with either an account or a loan*

$$\pi_{customer\_name}(depositor) \cup \pi_{customer\_name}(borrower)$$

Union - Example

Relation *depositor*

customer_name	*account_number*
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

Relation *borrower*

customer_name	*loan_number*	*amount*
Adams	L-16	1300
Curry	L-93	500
Hayes	L-15	1500
Jackson	L-14	1500
Jones	L-17	1000
Smith	L-23	2000
Smith	L-11	900
Williams	L-17	1000

$\pi_{customer\ name}(depositor) = ?$

$\pi_{customer\ name}(borrower) = ?$

$\pi_{customer\_name}(depositor) \cup \pi_{customer\_name}(borrower) = ?$

Union - Example

$r =$

A	B
a	1
a	2
b	1

$s =$

A	B
a	2
b	3

$r \cup s =$

A	B
a	1
a	2
b	1
b	3

Difference

Notation: $r - s$

Defined as:

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

For $r - s$ to be valid:

- r and s must have the same arity
- The attribute domains must be compatible

Example: *Find all customers with an account but not a loan*

$$\pi_{customer\ name}(depositor) - \pi_{customer\ name}(borrower)$$

Difference - Example

customer_name	account_number
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

customer_name	loan_number	amount
Adams	L-16	1300
Curry	L-93	500
Hayes	L-15	1500
Jackson	L-14	1500
Jones	L-17	1000
Smith	L-23	2000
Smith	L-11	900
Williams	L-17	1000

$\pi_{customer\ name}(depositor) = ?$

$\pi_{customer\ name}(borrower) = ?$

$\pi_{customer\ name}(depositor) - \pi_{customer\ name}(borrower) = ?$

Difference - Example

$r =$

A	B
a	1
a	2
b	1

$s =$

A	B
a	2
b	3

$r - s =$

A	B
a	1
b	1

Intersection

Notation: $r \cap s$

Note: $r \cap s = r - (r - s)$

Defined as:

$$r \cap s = \{t \mid t \in r \text{ and } t \in s\}$$

For $r \cap s$ to be valid:

- r and s must have the same arity
- The attribute domains must be compatible

Example: *Find all customers with both an account and a loan*

$$\pi_{customer\_name}(depositor) \cap \pi_{customer\_name}(borrower)$$

Intersection - Example

$r =$

A	B
a	1
a	2
b	1

$s =$

A	B
a	2
b	3

$r \cap s$ =?

A	B
a	2

Fundamental Relational Operations

Select

Notation:

$$\sigma_c(r)$$

C is called the selection condition (or predicate)

Returns a relation **with the same schema** as the input such that:

$$\sigma_c(r) = \{t | t \in r \text{ and } c(t)\}$$

- Where c is a formula in predicate calculus over the attributes of r consisting of terms connected by : $\wedge$ (and), $\vee$ (or), $\neg$ (not)
- c is evaluated for each tuple $t \in r$

Select - Example

Relation *account*

account_number	*branch_name*	*balance*
A-101	Downtown	500
A-215	Mianus	700
A-102	Perryridge	400
A-305	Round Hill	350
A-201	Brighton	900
A-222	Redwood	700
A-217	Brighton	750

$\sigma_{branch\_name="Brighton"}(account)$

Select - Example

Relation *r*

A	B	C	D
a	a	1	7
a	b	5	7
b	b	12	3
b	b	23	10

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

A	B	C	D
a	a	1	7
b	b	23	10

Project

Notation:

$$\pi_{A1,\ldots An}(r)$$

Where $A_1, \ldots, A_n$ are attribute names and r is a relation name.

Returns a relation with schema $A_1 \ldots A_n$ such that:

$$\pi_{A1,\ldots An}(r) = \{t[A_1 \ldots A_n] \mid t \in r\}$$

- The result is defined as the relation of n columns obtained by erasing the columns that are not listed
- $A_1, \ldots A_n$ must all exist in the schema of r
- Will <u>result in collapsing duplicate rows from result</u>, since relations are sets

Project - Example

Relation *account*

account_number	*branch_name*	*balance*
A-101	Downtown	500
A-215	Mianus	700
A-102	Perryridge	400
A-305	Round Hill	350
A-201	Brighton	900
A-222	Redwood	700
A-217	Brighton	750

$\pi_{account\_number, balance}(account) = ?$

Project - Example

Relation *r*

A	B	C
a	10	1
a	20	1
b	30	1
b	40	2

$\pi_{A,C}(r) =$

A	C
a	1
a	1
b	1
b	2

Renaming

Notation:

$$\rho_{A1\mapsto B1,\dots,Am\mapsto Bm}(E)$$

Where $A_i \mapsto B_i$ is a mapping of an attribute name

Returns a relation **with schema** $B_1 \dots B_m$ such that:

$$\rho_{A1\mapsto B1,\dots,Am\mapsto Bm}(E) = \{t \mid \exists\, u \in r,\ t[B_i] = u[A_i]\ \forall_{1\le i\le m}\}$$

- The result is a relation with the same tuples as the input but with distinct attribute names
- Useful to avoid clashes of names
- Alternatively, Ai can also specify the position of an attribute
- Example:

$$\rho_{branch\_name \mapsto branch}(account)$$

Cartesian Product

Notation: $r \times s$

Defined as:

$$r \times s = \{ t_r t_s \mid t_r \in r, \ t_s \in s \}$$

- Preconditions:
  - Attributes of $r(R)$ and $s(S)$ are disjoint (i.e., $R \cap S = \emptyset$).
- If attributes of $r(R)$ and $s(S)$ are not disjoint, then the Renaming operation should be used.

Cartesian Product - Example

$r =$

A
a
b

$s =$

X
1
2
3

$r \times s = ?$

A	X
a	1
a	2
a	3
b	1
b	2
b	3

Cartesian Product - Example

$r =$

A	B
a	1
b	2

$s =$

C	D	E
a	10	x
b	10	x
b	20	y
c	10	y

$r \times s =$

A	B	C	D	E
a	1	a	10	x
a	1	b	10	x
a	1	b	20	y
a	1	c	10	y
b	2	a	10	x
b	2	b	10	x
b	2	b	20	y
b	2	c	10	y

Chaining of Relational Expressions

Composition of operations

Expressions can be built using multiple operations

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

$r =$

A	B
a	1
b	2

$s =$

C	D	E
a	10	x
b	10	x
b	20	y
c	10	y

$r \times s$

A	B	C	D	E
a	1	a	10	x
a	1	b	10	x
a	1	b	20	y
a	1	c	10	y
b	2	a	10	x
b	2	b	10	x
b	2	b	20	y
b	2	c	10	y

$\sigma_{A=C}$

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

A	B	C	D	E
a	1	a	10	x
b	2	b	10	x
b	2	b	20	y

Assignment

Notation: $r \longleftarrow E$

- Assigns the result of evaluating the expression E to a relation name r
- Useful to break down complex relational expressions

- NB.: An **assignment** is a statement (not a relational expression!)
- Example:

$$customer \longleftarrow \pi_{customer\_name}(depositor) \cup \pi_{customer\_name}(borrower)$$

$$\sigma_{customer\_name='Smith'}(costumer)$$

Renomeação com atribuição (cf/ livro)

Notação:

$\rho(r(F), E)$

Onde

- E: é uma expressão relacional (input)
- r: é o nome da nova relação (output)
- F: é um mapeamento de nomes na forma

nome_antigo ↦ nome

posição ↦ nome

NB. Viola o principio do fecho!

- O resultado é uma nova relação com mesmos campos de E, excepto para os que são renomeados.
- F pode ser vazio

Additional Operations

Natural Join

Notation:

$r \bowtie s$

Where r and s be relations on schemas R and S respectively.

Returns a relation with schema $R \cup S$ such that:

$$r \bowtie s = \{t_r t_s \mid t_r \in r,\ t_s \in s,\ and\ t_r[R \cap S] = t_s[R \cap S]\}$$

- All combinations of tuples t_r from r and t_s from s that agree on the values of the attributes common to r and r (i.e., of the attributes of R ∩ S

Natural Join - Example 1

$r =$

A	B
a	1
b	2
c	3

$s =$

B	C
1	x
2	y
3	z

One-to-one mapping of values

$r \bowtie s =$

A	B	C
a	1	x
b	2	y
c	3	z

Natural Join - Example 2

$r =$

A	B
a	1
b	2
c	2

$s =$

B	C
1	x
2	y
3	z

One-to-many mapping of values

$r \bowtie s =$

A	B	C
a	1	x
b	2	y
c	2	y

Natural Join - Example 3

$r =$

A	B
a	1
b	2
c	2

$s =$

B	C
1	x
2	y
2	z

Many-to-many mapping of values

$r \bowtie s =$

A	B	C
a	1	x
b	2	y
b	2	z
c	2	y
c	2	z

Natural Join - Example 4

employee

employee_name	*street*	*city*
Coyote	Toon	Hollywood
Rabbit	Tunnel	Carrotville
Smith	Revolver	Death Valley
Williams	Seaview	Seattle

works

employee_name	*branch_name*	*salary*
Coyote	Mesa	1500
Rabbit	Mesa	1300
Gates	Redmond	5300
Williams	Redmond	1500

employee ⋈ *works*

"Perde informação"!

employee_name	*street*	*city*	*branch_name*	*salary*
Coyote	Toon	Hollywood	Mesa	1500
Rabbit	Tunnel	Carrotville	Mesa	1300
Williams	Seaview	Seattle	Redmond	1500

Natural Join vs Product

Natural Join and Product are similar in their definition and in their notation

$$r \times s = \{ t_r t_s \mid t_r \in r,\ t_s \in s \}$$

$$r \bowtie s = \{ t_r t_s \mid t_r \in r,\ t_s \in s, \text{ and } t_r[R \cap S] = t_s[R \cap S] \}$$

- Join is a "specialization" of the Cartesian product
  - Because a join can be defined algebraically in terms of a projection, a
    and Cartesian product

Example:

$R = (A, B, C, D)$, $S = (E, B, D)$

Result schema = (A, B, C, D, E)

$$r \bowtie s = \pi_{r.A, r.B, r.C, r.D, s.E}(\sigma_{r.B = s.B \wedge r.D = s.D}\ (r \times s))$$

Natural Join - Example 5

r =

A	B	C	D
a	1	a	x
b	2	c	x
c	4	b	y
a	1	c	x
d	2	b	y

s =

B	D	E
1	x	a
3	x	b
1	x	c
2	y	d
3	y	e

$r \bowtie s$ =

A	B	C	D	E
a	1	a	x	a
a	1	a	x	c
a	1	c	x	a
a	1	c	x	c
d	2	b	y	d

Division

Notation:

$r \div s$

Where *r* and *s* be relations on schemas *R* and *S* respectively.

Returns a relation **with schema** ***R-S*** such that:

$$r \div s = \{t[R - S] \mid t \in r \text{ and } s \subseteq \{u[S] \mid u \in r \text{ and } u[R - S] = t[R - S]\}\}$$

- Determines the subset of *r* that is combined (i.e. covers) all the tuples of *s*.

Example:

Which suppliers (r) can supply all parts (s)

Division

A
1
2

×

B
□
△
○

=

A	B
1	□
1	△
1	○
2	□
2	△
2	○

÷

A
1
2

=

B
□
△
○

A
1
2

×

B
□
△
○

=

A	B
1	□
1	△
1	○
2	□
2	△
~~2~~	~~○~~

÷

A
1
2

=

B
□
△

Division: Example 1

$r =$

A	X
a	1
a	2
a	3
b	1
b	2

$s =$

X
1
2

$r \div s =$

A
a
b

Division - Example 1

$r =$

A	B
a	1
a	2
a	3
b	1
c	1
d	1
d	3
d	4
e	6
e	1
b	2

$s =$

B
1
2

$r \div s = ?$

A
a
b

Division - Example 2

r =

A	B	C	D	E
a	x	a	x	1
a	x	c	x	1
a	x	c	y	1
b	x	c	x	1
b	x	c	y	3
c	x	c	x	1
c	x	c	y	1
c	x	b	y	1

s =

D	E
x	1
y	1

$r \div s$ =

A	B	C
a	x	c
c	x	c

TÉCNICO LISBOA

Bases de Dados

Aula 09: Álgebra Relacional

Prof. Paulo Carreira

Extended Operations

Generalised Projection

Generalised Projection

Notation:

$\pi_{F1,..,Fk}(r)$

Where $F_1,\ldots,F_k$ are functions (or expressions) over the attributes of r

Returns a relation **with schema** $F_l \ldots F_k$ such that:

$\pi_{F1,..,Fk}(r) = \{<F_1(t), \ldots, F_k(t)>| \ t \in r\}$

- The result is the tuples resulting from of evaluating the expressions $F_1,\ldots,F_k$ over each input tuple t of r.

Example:

- Given the relation `credit_info(customer_name, credit_limit amount_spent)`, find how much each person can spend.

$\pi_{customer\_name,\ credit\_limit\ -\ amount\_spent}(credit\_info)$

Generalized Projection - Example

$r =$

A	B	C
a	10	1
a	20	1
b	30	1
b	40	2

$\pi_{A,C*1.10}(r) =$

A	C*1.1
a	1.1
b	1.1
b	2.2

Aggregation

Função de Agregação

Definição

Uma **função de agregação** recebe um **conjunto de valores** como entrada e devolve um **valor** como resultado

- Exemplos
  - Min, Max
  - Sum, Count, Avg

Aggregate Operations

Notation:

$$_{L}G_{F}(r)$$

Where L={A_1,…,A_n} is a subset of attributes of the schema of *r* and F={F_1,…,F_k} if a set of aggregate functions

Returns a relation **with schema** $A_1 \ldots A_n F_1 \ldots F_k$ such that:

$$_{L}G_{F}(r) = \{t \mid t[A_1,\ldots,A_n] \in \pi_{A1,\ldots,An}(r) \text{ and } t[F_i]=F_i(\{u \mid u[A_1,\ldots,A_n] = t[A_1,\ldots,A_n]\}) \ \forall_{1\leq i\leq k}\}$$

- The result is the tuples resulting from of evaluating the expressions $F_1,\ldots,F_k$ over set of tuples of each formed group
- *L={A1,…, An} is a set of attributes on which to group (can be empty)*
- *F={F1 … Fk} is a set of aggregate functions*

Aggregate Operation: Example

Vendas de produto (vendas)

Nome	Balcao	Saldo
Cajó	Chelas	250
Manel	Damaia	300
Quim	Venda Nova	200
Tó	Damaia	400
Xico	Damaia	500
Zé	Venda Nova	100

Consulta

Apurar total de vendas por Balcão

Expressão Relacional na forma:

$_{L}G_{F}$

Passo 1: Formar grupos por balcão

$_{balcao}G_{F?}(vendas)$

Aggregate Operation: Example

Quantas linhas terá resultado?

Vendas de produto

Nome	Balcao	Saldo
Cajó	Chelas	250
Manel	Damaia	300
Tó	Damaia	400
Xico	Damaia	500
Zé	Venda Nova	100
Quim	Venda Nova	200

$$_{balcao}G_{SUM(Saldo)}(vendas)$$

Balcao	Saldo
Chelas	250
Damaia	1200
Venda Nova	300

Passo 2: Somar (agregar) sub-totais por grupo

Aggregate Operation: Example

Vendas de produto

Nome	Balcao	Saldo
Cajó	Chelas	250
Manel	Damaia	300
Tó	Damaia	400
Xico	Damaia	500
Zé	Venda Nova	100
Quim	Venda Nova	200

O que acontece se adicionarmos mais atributos?

$$_{balcao,\ nome}\mathcal{G}_{SUM(Saldo)}(vendas)$$

Nome	Balcao	Saldo
Cajó	Chelas	250
Manel	Damaia	300
Tó	Damaia	400
Xico	Damaia	500
Zé	Venda Nova	100
Quim	Venda Nova	200

Aggregate Operation: Example

Vendas de produto

Nome	Balcao	Saldo
Cajó	Chelas	250
Manel	Damaia	300
Tó	Damaia	400
Xico	Damaia	500
Zé	Venda Nova	100
Quim	Venda Nova	200

O que acontece se retirarmos atributos?

$$G_{SUM(Saldo)}(vendas)$$

Saldo
1750

Aggregate Operation: Example

Vendas de produto

Nome	Balcao	Saldo
Cajó	Chelas	250
Manel	Damaia	300
Tó	Damaia	400
Xico	Damaia	500
Zé	Venda Nova	100
Quim	Venda Nova	200

Como contar os vendedores por balcao?

$$_{balcao}G_{count()}(vendas)$$

Balcao	Count
Chelas	1
Damaia	3
Venda Nova	2

Aggregate Operation: Example

Vendas de produto

Nome	Balcao	Saldo
Cajó	Chelas	250
Manel	Damaia	300
Tó	Damaia	400
Xico	Damaia	500
Zé	Venda Nova	100
Quim	Venda Nova	200

Como contar os vendedores e a soma por balcao?

$$ {}_{balcao}G_{count(),\, sum(Saldo)}(vendas) $$

Balcao	Count	Saldo
Chelas	1	250
Damaia	3	1200
Venda Nova	2	300

Aggregate Operation: Example

Relation account grouped by branch name

branch name	account number	balance
Lisboa	A-102	400
Lisboa	A-201	900
Oporto	A-217	750
Oporto	A-215	750
Vila Real	A-222	700

$_{branch\ name}G_{sum(balance)}(account)$

branch name	sum(balance)
Lisboa	1300
Oporto	1500
Vila Real	700

Aggregate Operation (cont.)

- The attributes corresponding to aggregation functions do not have a name. However:
  - We can use the **Rename** operation to give it a name
  - For convenience, we permit **inline renaming** after each aggregation function using '**as**' or '↦'.

$$_{branch\_name}G_{sum(balance) \mapsto sum\_balance}(account)$$

branch name	sum balance
Damaia	1300
Chelas	1500
Buraca	700