Relational Algebra

Dr Paolo Guagliardo

dbs-lecturer@ed.ac.uk

THE UNIVERSITY of EDINBURGH
informatics

Fall 2018

Data model

Relations (tables) are sets of **records** of the same length

Schema

- Set of **relation names**
- Set of distinct **attributes** for each table
  Note that columns are not ordered

Instance

- Actual data (that is, the records in each relation)
- Each **record** is a function from attributes to values

Relational algebra

Procedural query language

A relational algebra expression

- takes as input one or more relations
- applies a **sequence of operations**
- returns a relation as output

Operations:

Projection	(π)	Union	($\cup$)
Selection	(σ)	Intersection	($\cap$)
Product	($\times$)	Difference	($-$)
Renaming	(ρ)		

The application of each operation results in a new relation that can be used as input to other operations

Projection

- **Vertical operation**: choose some of the columns
- Syntax: $\pi_{\text{set of attributes}}(\text{relation})$
- $\pi_{A_1,\ldots,A_n}(R)$ takes only the values of attributes $A_1, \ldots, A_n$ for each tuple in R

Customer

CustID	Name	City	Address
cust1	Renton	Edinburgh	2 Wellington Pl
cust2	Watson	London	221B Baker St
cust3	Holmes	London	221B Baker St

$\pi_{\text{Name,City}}(\text{Customer})$

Name	City
Renton	Edinburgh
Watson	London
Holmes	London

Selection

- **Horizontal operation**: choose rows satisfying some condition
- Syntax: $\sigma_{\text{condition}}(\text{relation})$
- $\sigma_\theta(R)$ takes only the tuples in R for which θ is satisfied

$$\begin{aligned} \text{term} &:= \text{attribute} \mid \text{constant} \\ \theta &:= \text{term } \textbf{op} \text{ term with } \textbf{op} \in \{=, \neq, >, <, \geqslant, \leqslant\} \\ &\mid \theta \wedge \theta \mid \theta \vee \theta \mid \neg\theta \end{aligned}$$

Example of selection

Customer

CustID	**Name**	**City**	**Age**
cust1	Renton	Edinburgh	24
cust2	Watson	London	32
cust3	Holmes	London	35

$\sigma_{\text{City} \neq \text{'Edinburgh'} \wedge \text{Age} < 33}(\text{Customer})$

CustID	**Name**	**City**	**Age**
cust2	Watson	London	32

Efficiency (1)

Consecutive selections can be combined into a single one:

$$\sigma_{\theta_1}\big(\sigma_{\theta_2}(R)\big) = \sigma_{\theta_1 \wedge \theta_2}(R)$$

Example

$Q_1 = \sigma_{\text{City}\neq\text{'Edinburgh'}}\big(\sigma_{\text{Age}<33}(\text{Customer})\big)$

$Q_2 = \sigma_{\text{City}\neq\text{'Edinburgh'} \wedge \text{Age}<33}(\text{Customer})$

$Q_1 \equiv Q_2$ but Q_2 **faster** than Q_1 in general

Efficiency (2)

Projection can be pulled in front of selection

$$\sigma_{\theta}\big(\pi_{\alpha}(R)\big) = \pi_{\alpha}\big(\sigma_{\theta}(R)\big)$$

only if all attributes mentioned in θ appear in α

Example

$Q_1 = \pi_{\text{Name,City,Age}}\big(\sigma_{\text{City}\neq\text{'Edinburgh'} \wedge \text{Age}<33}(\text{Customer})\big)$

$Q_2 = \sigma_{\text{City}\neq\text{'Edinburgh'} \wedge \text{Age}<33}\big(\pi_{\text{Name,City,Age}}(\text{Customer})\big)$

Question: Which one is more efficient?

Cartesian product

$R \times S$ **concatenates** each tuple of R with all the tuples of S

Example

R	A	B
	1	2
	3	4

$\times$

S	C	D
	1	a
	2	b
	3	c

$=$

$R \times S$	A	B	C	D
	1	2	1	a
	1	2	2	b
	1	2	3	c
	3	4	1	a
	3	4	2	b
	3	4	3	c

Expensive operation:

- $\mathrm{card}(R \times S) = \mathrm{card}(R) \times \mathrm{card}(S)$
- $\mathrm{arity}(R \times S) = \mathrm{arity}(R) + \mathrm{arity}(S)$

Joining relations

Combining Cartesian product and selection

Customer: ID, Name, City, Address
Account: Number, Branch, CustID, Balance

We can join customers with the accounts they own as follows

$$\sigma_{\mathsf{ID}=\mathsf{CustID}}(\mathsf{Customer} \times \mathsf{Account})$$

Renaming

Gives a new name to some of the attributes of a relation

Syntax: $\rho_{\text{replacements}}(\text{relation})$,
where a replacement has the form $A \to B$

$$\rho_{A\to A', C\to D}\left(\begin{array}{ccc} \textbf{A} & \textbf{B} & \textbf{C} \\ \hline \text{a} & \text{b} & \text{c} \\ 1 & 2 & 3 \end{array}\right) = \begin{array}{ccc} \textbf{A}' & \textbf{B} & \textbf{D} \\ \hline \text{a} & \text{b} & \text{c} \\ 1 & 2 & 3 \end{array}$$

Example

Customer: **CustID**, Name, City, Address

Account: Number, Branch, **CustID**, Balance

$$\sigma_{\text{CustID}=\text{CustID}'}\left(\text{Customer} \times \rho_{\text{CustID}\to\text{CustID}'}(\text{Account})\right)$$

Natural join

Joins two tables on their **common attributes**

Example

Customer: **CustID**, Name, City, Address

Account: Number, Branch, **CustID**, Balance

Customer $\bowtie$ Account $=$

$$\pi_{X\cup Y}\left(\sigma_{\text{CustID}=\text{CustID}'}\left(\text{Customer} \times \rho_{\text{CustID}\to\text{CustID}'}(\text{Account})\right)\right)$$

where X = { all attributes of Customer }
Y = { all attributes of Account }

From SQL to relational algebra

$$\begin{aligned} \texttt{SELECT} &\mapsto \text{projection } \pi \\ \texttt{FROM} &\mapsto \text{Cartesian product } \times \\ \texttt{WHERE} &\mapsto \text{selection } \sigma \end{aligned}$$

$$\begin{array}{ll} \textbf{SELECT} & A_1, \ldots, A_m \\ \textbf{FROM} & T_1, \ldots, T_n \\ \textbf{WHERE} & \langle \text{condition} \rangle \end{array} \quad \mapsto \quad \pi_{A_1, \ldots, A_m} \left(\sigma_{\langle \text{condition} \rangle} (T_1 \times \cdots \times T_n) \right)$$

Common attributes in $T_1, \ldots, T_n$ must be renamed

Set operations

Union

R	A	B
	a1	b1
	a2	b2

$\cup$

S	A	B
	a1	b1
	a3	b3

$=$

R $\cup$ S	A	B
	a1	b1
	a2	b2
	a3	b3

Intersection

R	A	B
	a1	b1
	a2	b2

$\cap$

S	A	B
	a1	b1
	a3	b3

$=$

R $\cap$ S	A	B
	a1	b1

Difference

R	A	B
	a1	b1
	a2	b2

$-$

S	A	B
	a1	b1
	a3	b3

$=$

R $-$ S	A	B
	a2	b2

The relations must have the same set of attributes

Union and renaming

R	Father	Child
	George	Elizabeth
	Philip	Charles
	Charles	William

S	Mother	Child
	Elizabeth	Charles
	Elizabeth	Andrew

We want to find the relation **parent-child**

$\rho_{\text{Father}\to\text{Parent}}(\text{R}) \cup \rho_{\text{Mother}\to\text{Parent}}(\text{S}) \quad =$

Parent	Child
George	Elizabeth
Philip	Charles
Charles	William
Elizabeth	Charles
Elizabeth	Andrew

Full relational algebra

Primitive operations: π , σ , $\times$, ρ , $\cup$, $-$

Removing any of these results in a **loss of expressive power**

Derived operations

$\bowtie$ can be expressed in terms of π , σ , $\times$, ρ

$\cap$ can be expressed in terms difference:

$$R \cap S = R - (R - S)$$

Other derived operations

Theta-join	$R \bowtie_\theta S = \sigma_\theta(R \times S)$
Equijoin	$\bowtie_\theta$ where θ is a **conjunction of equalities**
Semijoin	$R \ltimes_\theta S = \pi_X\big(R \bowtie_\theta S\big)$ where X is the set of attributes of R
Antijoin	$R \,\overline{\ltimes}_\theta\, S = R - (R \ltimes_\theta S)$

Why use these operations?

- to write things more succintly
- they can be optimized independently

Division

R over set of attributes X

S over set of attributes $Y \subset X$

Let $Z = X - Y$

$$R \div S = \left\{ \bar{r} \in \pi_Z(R) \mid \forall \bar{s} \in S \,.\, \bar{r}\bar{s} \in R \right\}$$

$$= \left\{ \bar{r} \in \pi_Z(R) \mid \{\bar{r}\} \times S \subseteq R \right\}$$

$$= \pi_Z(R) - \pi_Z\big(\pi_Z(R) \times S - R\big)$$

Division: Example

Exams

Student	Course
John	Databases
John	Networks
Mary	Programming
Mary	Math
Mary	Databases

DPT

Course
Databases
Programming

Exams $\div$ DPT $=$

Student
Mary

$$= \pi_{\textbf{Student}}(\text{Exams}) - \pi_{\textbf{Student}}\big(\pi_{\textbf{Student}}(\text{Exams}) \times \text{DPT} - \text{Exams}\big)$$