DBM1
IST Semester

Vincent Primault

vincent.primault@insa-lyon.fr

Who am I?

Vincent Primault

- 3rd year PhD student at the LIRIS laboratory
- Office: 501.315, in CS department
- Email: vincent.primault@insa-lyon.fr

Course organization

- 20h in total, from now to October, 20th.
- Typical french organisation:
  - 6h of lectures (rather theoretical)
  - 4h of exercises (to apply abstract notions)
  - 8h of labs (practical experiments)
  - 2h of final exam
- Balance between theory and practice.

- Final mark: 50% exam, 50% labs.

Course organization (cont’d)

- Take notes!
- Ask questions!

Course outline

- Databases fundamentals
- Relational algebra
- SQL language
- Query processing & optimization
- Distributed databases & NoSQL

Bibliography

- A huge amount of resources on the Web, but be careful…

- J.D. Ullman. **Database and Knowledge-base Systems**. 1988.
- J. Paredaens, P. De Bra, M. Gyssens, D. Van Gucht. **The Structure of the Relational Model**. 1989.
- H. Garcia-Molina, J. D. Ullman, J. Widom. **Database Systems**. 1999.
- P. O'Neil. **Database: Principles, Programming, Performance**. 1994.
- S. Abiteboul, R. Hull, V. Vianu. **Foundations of Databases**. 1995.

DBM1
Part 1: Introduction

Vincent Primault

vincent.primault@insa-lyon.fr

From files…

- Data can always be stored within ordinary files
  - Various formats: plain text, Excel, CSV, JSON, YAML, XML…
  - More or less structured

Last name	First name	Birthdate
Doe	John	1992-05-13
Smith	Jane	1993-12-01

Students.xls

Last name	First name	Course	Mark
Doe	John	DBM1	13
Doe	John	OPS	15
Smith	Jane	DBM1	12
Smith	Jane	OPS	17

Marks.xls

From files… (cont'd)

- Data is duplicated (first/last name)
  - What happens if one of this information changes?
  - What happens if there is a mistake while typing a name?
- Nothing prevents to put inconsistent data (whether it is on purpose or not). E.g, a mark more than 20.
- How to uniquely identify student (thing about homonyms)?
- How to compute the average mark?

… to databases

- A database is a structured way to store data.
- A database management system (DMBS) is a software that acts as an interface between users or other applications and a database. It includes creating, modifying, deleting and querying data.

- Databases still store information into text and/or binary files!
  - But this is an implementation detail that differ from one DBMS to the other.
  - DBMS hides this complexity by providing a clean interface.
  - DBMS rely on standards (like SQL) to ensure interoperability.

The promises of databases

- **Atomicity:** guarantee that a set of operations can be processed in a row.
- **Consistency:** possibility to put constraints on data.
- **Isolation:** it is possible for several users to interact with the same database.
- **Durability:** result of operations are stored permanently, and resilient to software and hardware errors.

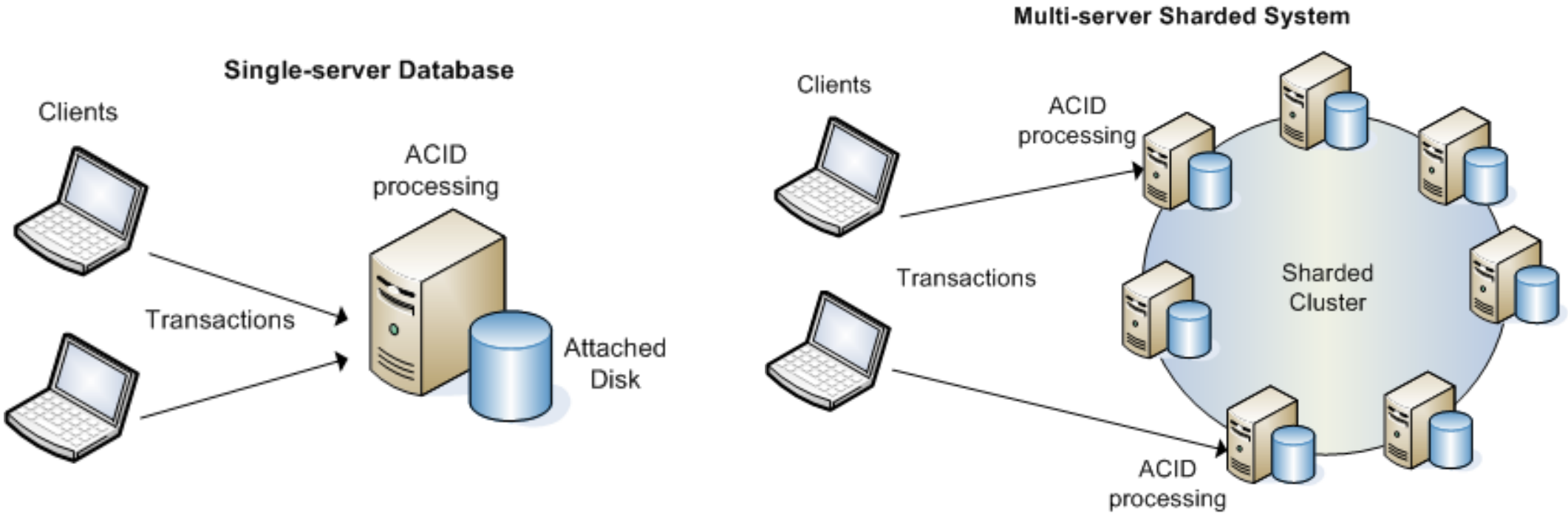
- And more:
  - **Security:** access rights can be enforced with a fine granularity.
  - Powerful query language: **SQL**.

A lot of players

Relational and centralized DBMS

"NoSQL" and distributed DBMS

Centralized vs distributed

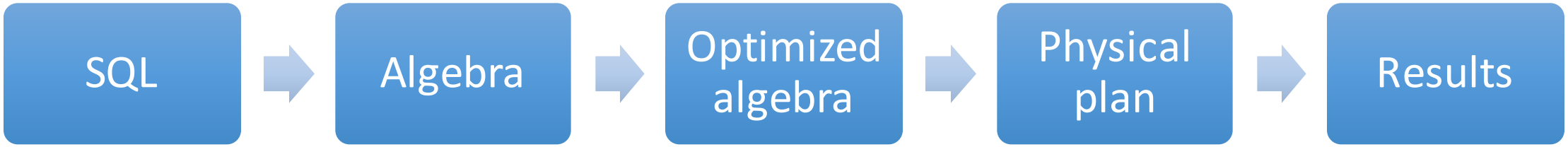


DBM1
Part 2: Relational algebra

Vincent Primault

vincent.primault@insa-lyon.fr

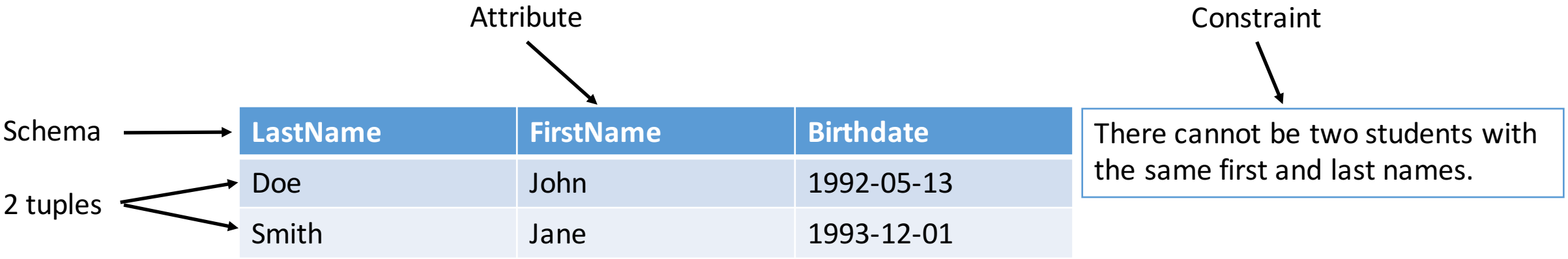
Why learn relational algebra?



The life of a typical SQL query

The relational model

- A *relation* is a set of *tuples*, bound to a *schema*.
- A schema contains *attributes* and constraints on them.
- Each attribute is associated with a *domain* of values.



Some notations

Relation schema: (Ω, Δ, dom, SC)

- Ω: set of attributes
- Δ: set of domains
- dom: $\Omega \rightarrow \Omega$
- SC: set of constraints

Tuple t

- t: $\Omega \rightarrow U\delta \in \Delta$, $t(A) \in dom(A)$
- It is a function!

Relation instance

Schema of Students = (Ω, Δ, dom, SC)

Ω = {LastName, FirstName, Birthdate}

Δ = {set of dates, set of strings}

dom(LastName) = set of strings

dom(FirstName) = set of strings

dom(Birthdate) = set of dates

LastName	FirstName	Birthdate
Doe	John	1992-05-13
Smith	Jane	1993-12-01

An instance of Students

SC contains the following constraint:

1. There are no two different tuples with the same (FirstName, LastName) pair.

A relational database

A relational database is a finite set of relations SR.

If two relations i and j both have an attribute A such that

$$A \in \Omega_i \cap \Omega_j$$

then

$$dom_i(A) = dom_j(A)$$

and they have the exact same meaning.

What is a relational algebra?

- An algebra is a mathematical system consisting of:
  - Operands : variables or values from which new values can be constructed.
  - Operators: symbols denoting procedures that construct new values from given values.

- A relational algebra is an algebra whose operands are relations or variables that represent relations.
- Operators are designed to do the most common things that we need to do with relations in a database. The result is an algebra that can be used as a query language for relations.

Relational algebra operators

- Set operations: **union**, **intersection** and **difference**.
- **Selection**: picking only certain tuples.
- **Projection**: picking only certain attributes.
- **Renaming** of attributes.
- **Products** and **joins**: composition of relations.
- **Division**.

Intersection, union, difference

- These are the classical set operators, except that...
- **… operands need to have the same schema.**
- Notations:
  - $R \cap S$
  - $R \cup S$
  - $R - S$

Intersection, union, difference (cont'd)

Students2014

LastName	FirstName	Birthdate
Doe	John	1992-05-13
Smith	Jane	1993-12-01

Students2015

LastName	FirstName	Birthdate
Brown	Emily	1991-11-18
Doe	John	1992-05-13

Students2014 ∩ Students 2015

LastName	FirstName	Birthdate
Doe	John	1992-05-13

Students2014 ∪ Students 2015

LastName	FirstName	Birthdate
Doe	John	1992-05-13
Smith	Jane	1993-12-01
Brown	Emily	1991-11-18

Students2014 − Students 2015

LastName	FirstName	Birthdate
Smith	Jane	1993-12-01

Intersection, union, difference (cont'd)

- Schema of R is (Ω, Δ, dom).
- Schema of S is (Ω, Δ, dom).
  - Schema of $R \cap S$ is (Ω, Δ, dom)
  - Schema of $R \cup S$ is (Ω, Δ, dom)
  - Schema of $R - S$ is (Ω, Δ, dom)

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

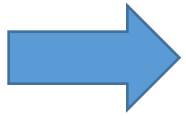
Projection

- Eliminates columns, consequently removing duplicates across remaining attributes.
- Notation: $\pi_{A1,\ldots,As}(R)$

Projection (cont'd)

Students

LastName	FirstName	Birthdate
Doe	John	1992-05-13
Smith	Jane	1993-12-01
Doe	Anna	1994-03-01
Smith	Lara	1993-12-01
Jones	John	1993-01-16



$\pi_{LastName,FirstName}(Students)$

LastName	FirstName
Doe	John
Smith	Jane
Doe	Anna
Smith	Lara
Jones	John

$\pi_{LastName,Birthdate}(Students)$

LastName	Birthdate
Doe	1992-05-13
Smith	1993-12-01
Doe	1994-03-01
Jones	1993-01-16

$\pi_{LastName}(Students)$

LastName
Doe
Smith
Jones

Projection (cont'd)

- Schema of R is (Ω, Δ, dom).
- Schema of $\pi_{A1,\ldots,As}(R)$ is $(\{A_1,\ldots,A_s\}, dom(\{A_1,\ldots,A_s\}), dom_{|\{A1,\ldots,As\}})$.

- Let r be an instance of R.
  - $\pi_{A1,\ldots,As}(r) = \{t_{|\{A1,\ldots,As\}} \mid t \in r\}$

- Properties:
  - $card(\pi_{A1,\ldots,As}(r)) \leq card(r)$
  - $\pi_{\Omega}(r) = r$
  - $\pi_{\varnothing}(r)$ = singleton with empty tuple

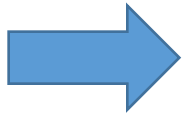
Selection

- Filters tuples to keep only those satisfying a condition.
- Notation: $\sigma_C(R)$
  - C is a logical condition.
  - The condition must be of the form A op B or A op b, with op being one of <, >, ≤, ≥, =, ≠.
  - Multiple conditions can be combined with AND, OR and NOT conjonctions.

Selection (cont'd)

Students

LastName	FirstName	Birthdate
Doe	John	1992-05-13
Smith	Jane	1993-12-01
Doe	Anna	1994-03-01
Smith	Lara	1993-12-01
Jones	John	1993-01-16



$\sigma_{LastName = Doe}(Students)$

LastName	FirstName	Birthdate
Doe	John	1992-05-13
Doe	Anna	1994-03-01

$\sigma_{LastName = Doe\ AND\ FirstName=John}(Students)$

LastName	FirstName	Birthdate
Doe	John	1992-05-13

$\sigma_{Birthdate=1992-05-14}(Students)$

LastName	FirstName	Birthdate

Selection (cont'd)

- Schema of R is (Ω, Δ, dom).
- Schema of $\sigma_C(R)$ is (Ω, Δ, dom).

- Let r be an instance of R.
  - $\sigma_C(r) = \{t \mid t \in r \text{ and } C(t)\}$

- Properties:
  - $card(\sigma_C(r)) \leq card(r)$
  - $\sigma_{A=A}(r) = r$
  - $\sigma_{NOT\ A=A}(r) = \varnothing$

- $\sigma_{C\ AND\ C'}(r) = \sigma_C(R) \cap \sigma_{C'}(R)$
- $\sigma_{C\ OR\ C'}(r) = \sigma_C(R) \cup \sigma_{C'}(R)$
- $\sigma_{NOT\ C}(r) = r - \sigma_C(R)$

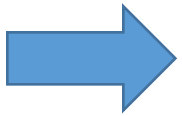
Renaming

- Gives a new schema to a relation but…
- **It does not change the instance.**
- Notation: $\rho_{\gamma}(R)$
  - γ is a one-to-one function, $\gamma : \Omega \rightarrow \Omega'$

Renaming (cont'd)

Students

LastName	FirstName	Birthdate
Doe	John	1992-05-13
Smith	Jane	1993-12-01
Doe	Anna	1994-03-01
Smith	Lara	1993-12-01
Jones	John	1993-01-16



ρ_{γ}(Students)

Nom	Prénom	DateDeNaissance
Doe	John	1992-05-13
Smith	Jane	1993-12-01
Doe	Anna	1994-03-01
Smith	Lara	1993-12-01
Jones	John	1993-01-16

γ(LastName) = Nom
γ(FirstName) = Prénom
γ(Birthdate) = DateDeNaissance

Renaming (cont'd)

- Schema of R is (Ω, Δ, dom).
- Schema of $\rho_{\gamma}(R)$ is ($\gamma(\Omega)$, Δ, dom.γ^{-1}).

- Let r be an instance of R.
  - $\rho_{\gamma}(r) = \{t.\gamma^{-1} \mid t \in r\}$

- Property:
  - card(Ω) = card(Ω')

Cartesian product

- Associates every tuple in R with every tuple in S.
- Notation: R x S
- It is very rare in practice, mainly used to express joins.

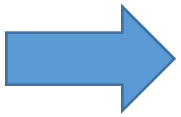
Cartesian product (cont'd)

Students

LastName	FirstName	Birthdate
Doe	John	1992-05-13
Smith	Jane	1993-12-01

FirstNames

FirstName	Gender	Popularity
John	Male	60
Anna	Female	75
Lara	Female	30



Students x FirstNames

$LastName_1$	$FirstName_1$	Birthdate	$FirstName_2$	Gender	Popularity
Doe	John	1992-05-13	John	Male	60
Doe	John	1992-05-13	Anna	Female	75
Doe	John	1992-05-13	Lara	Female	30
Smith	Jane	1993-12-01	John	Male	60
Smith	Jane	1993-12-01	Anna	Female	75
Smith	Jane	1993-12-01	Lara	Female	30

Cartesian product (cont'd)

- Schema of R is $(\Omega_R, \Delta_R, dom_R)$.
- Schema of S is $(\Omega_S, \Delta_S, dom_S)$.
- Schema of R x S is $(\Omega', \Delta_R \cup \Delta_S, dom')$.
  - $\Omega' = \{A_1 \mid A \in \Omega_R\} \cup \{A_2 \mid A \in \Omega_S\}$
  - $dom' = dom_R.\gamma_R \cup dom_S.\gamma_S, \gamma_R(A_1) = A, A \in \Omega_R, \gamma_S(A_2) = A, A \in \Omega_S$
- Let r, s be an instances of R and S respectively.
  - $r \times s = \{t.\gamma_R \cup t'.\gamma_s \mid t \in r \text{ and } t' \in s\}$

- Properties:
  - $card(r \times s) = card(r) \times card(s)$
  - $r \times \varnothing = \varnothing$

Natural join

- Associates two relations by their common attributes.
- Notation: $R \bowtie S$
- Meaning: $R \bowtie S = \pi_A(\sigma_C(R \times S))$ where
  - C checks equality of common attributes
  - A is the set of common attributes

- This is a reason why it is important that two attributes with the same name in two different relations must represent the same notion!

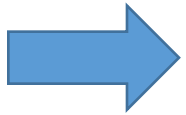
Natural join (cont'd)

Students

LastName	FirstName	Birthdate
Doe	John	1992-05-13
Smith	Jane	1993-12-01
Doe	Anna	1994-03-01
Smith	Lara	1993-12-01
Jones	John	1993-01-16

FirstNames

FirstName	Gender	Popularity
John	Male	60
Jane	Female	25
Lara	Female	30
James	Male	28
Anna	Female	43



Students ⋈ FirstNames

LastName	FirstName	Birthdate	Gender	Popularity
Doe	John	1992-05-13	Male	60
Smith	Jane	1993-12-01	Female	25
Doe	Anna	1994-03-01	Female	43
Smith	Lara	1993-12-01	Female	30
Jones	John	1993-01-16	Male	60

FirstNames ⋈ Students

FirstName	Gender	Popularity	LastName	Birthdate
John	Male	60	Doe	1992-05-13
John	Male	60	Jones	1993-01-16
Jane	Female	25	Smith	1993-12-01
Lara	Female	30	Smith	1993-12-01

Natural join (cont'd)

- Schema of R is $(\Omega_R, \Delta_R, dom_R)$.
- Schema of S is $(\Omega_S, \Delta_S, dom_S)$.
  - $dom_R|_{\Omega R} \cap \Omega_s = dom_S|_{\Omega R} \cap \Omega_S$
- Schema of $R \bowtie S$ is $(\Omega_R \cup \Omega_S, \Delta_R \cup \Delta_S, dom_R \cup dom_S)$.

- Let r, s be an instances of R and S respectively.
  - $r \bowtie s = \{t \cup t' \mid t \in r \text{ and } t' \in s\}$

- Properties:
  - $card(r \bowtie s) \leq card(r) \times card(s)$
  - $r \bowtie \varnothing = \varnothing$
  - $r \times s = r \bowtie s$ if $\Omega_R \cap \Omega_S = \varnothing$
  - $r \cap s = r \bowtie s$ if $\Omega_R = \Omega_S$

Theta join

- A join that involves a predicate.
- Notation: $R \bowtie_{\theta} S$
  - θ is a logical condition.
  - The condition must be of the form A op B or A op b, with op being one of <, >, ≤, ≥, =, ≠.
  - Multiple conditions can be combined with AND, OR and NOT conjonctions.

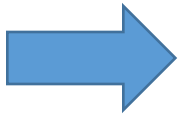
Theta join (cont'd)

Students

LastName	FirstName	Birthdate
Smith	Jane	1993-12-01
Doe	Anna	1994-03-01
Smith	Lara	1993-12-01

Celebrities

LastName	FirstName	Birthdate
Bieber	Justin	1994-03-01
Smith	Will	1968-09-25
Firth	Colin	1960-09-10



Students $\bowtie_{LastName1=LastName2}$ Celebrities

$LastName_1$	$FirstName_1$	$Birthdate_1$	$LastName_2$	$FirstName_2$	$Birthdate_2$
Smith	Jane	1993-12-01	Smith	Will	1968-09-25
Smith	Lara	1993-12-01	Smith	Will	1968-09-25

Students $\bowtie_{Birthdate2 \leq Birthdate1}$ Celebrities

$LastName_1$	$FirstName_1$	$Birthdate_1$	$LastName_2$	$FirstName_2$	$Birthdate_2$
Smith	Jane	1993-12-01	Smith	Will	1968-09-25
Smith	Jane	1993-12-01	Firth	Colin	1960-09-10
Doe	Anna	1994-03-01	Bieber	Justin	1994-03-01
Doe	Anna	1994-03-01	Smith	Will	1968-09-25
Doe	Anna	1994-03-01	Firth	Colin	1960-09-10
Smith	Lara	1993-12-01	Smith	Will	1968-09-25
Smith	Lara	1993-12-01	Firth	Colin	1960-09-10

Theta join (cont'd)

- Schema of R is $(\Omega_R, \Delta_R, dom_R)$.
- Schema of S is $(\Omega_S, \Delta_S, dom_S)$.
- Schema of $R \bowtie_\theta S$ is $(\Omega', \Delta_R \cup \Delta_S, dom')$.
  - $\Omega' = \{A_1 \mid A \in \Omega_R\} \cup \{A_2 \mid A \in \Omega_S\}$
  - $dom' = dom_R.\gamma_R \cup dom_S.\gamma_S$, $\gamma_R(A_1) = A$, $A \in \Omega_R$, $\gamma_S(A_2) = A$, $A \in \Omega_S$

- Let r, s be an instances of R and S respectively.
  - $r \bowtie_\theta s = \{t \mid t \in r \times s \text{ and } \theta(t)\}$

- Properties:
  - $card\ (r \bowtie_\theta s) = card(\sigma_\theta(R \times S))$
  - $R \bowtie_\theta S = \sigma_\theta(R \times S)$

Division

- Not a primitive operator. Used to express queries such as "*find students that attended all lessons*".
- Notation: R ÷ S

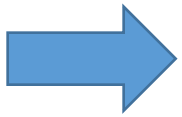
Division (cont'd)

Attendance

LastName	FirstName	Teaching
Smith	Jane	DBM1
Smith	Jane	DBM2
Smith	Jane	OPS
Doe	John	DBM1
Doe	John	OPS
Jones	John	DBM1
Jones	John	DBM2
Jones	John	OPS

Teachings

Teaching
DBM1
DBM2
OPS



Attendance ÷ Teachings

LastName	FirstName
Smith	Jane
Jones	John

It gives all tuples over (LastName, FirstName) that in combination of every tuple of Teachings results in a tuple of Attendance.

Division (cont'd)

- Schema of R is $(\Omega_R, \Delta_R, dom_R)$.
- Schema of S is $(\Omega_S, \Delta_S, dom_S)$.
- $\Omega_S \subseteq \Omega_R$, $dom_R|_{\Omega S} = dom_S$.

- Schema of $R \div S$ is $(\Omega_R - \Omega_S, \Delta_R, dom_R|_{\Omega R - \Omega R})$.

- Let r, s be an instances of R and S respectively.
  - $r \div s = \{t \mid \forall t' \in s, t \cup t' \in r\}$

Building complex expressions

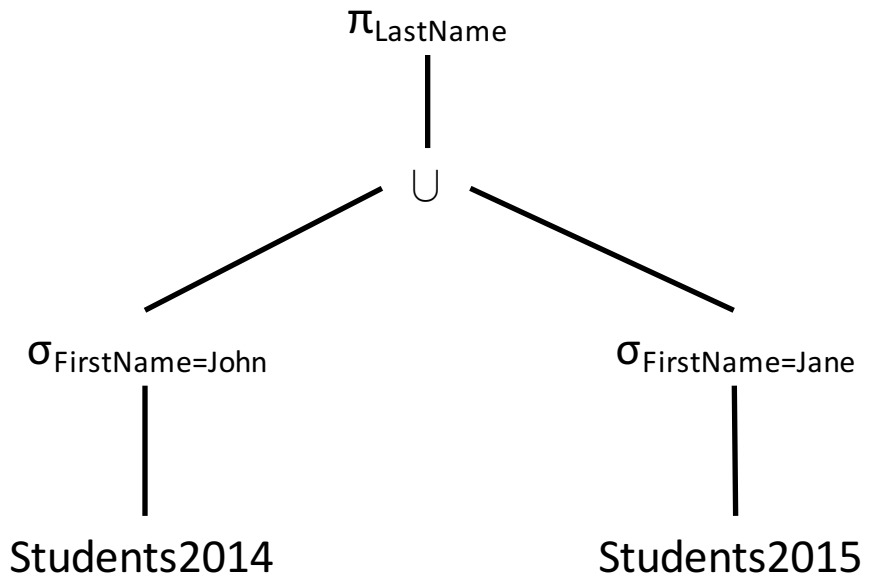
- Combine operators w.r.t. precedence rules of operators:
  1. Projection, selection, renaming (highest)
  2. Cartesian product, joins
  3. Difference
  4. Union, intersection (lowest)
- Use parentheses to force a different order, or just to clarify.

- Example: find the last name of 2014 students named John and those of 2015 named Jane.

$$\pi_{\text{LastName}}(\sigma_{\text{FirstName=John}}(\text{Students2014}) \cup \sigma_{\text{FirstName=Jane}}(\text{Students2015}))$$

Expression trees

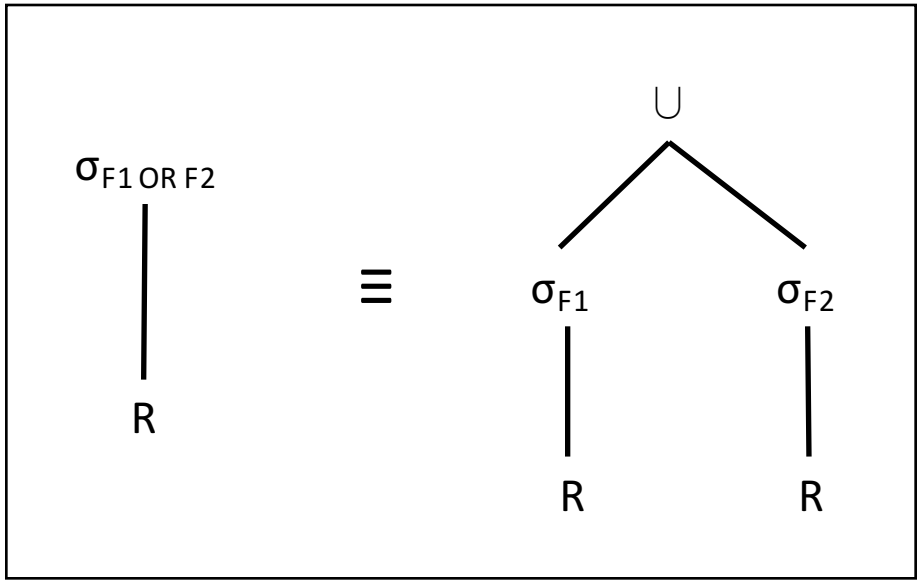
- Another representation of any relational algebra expression.
- Leaves are operands, usually variables standing for relations.
- Nodes are operators, applied to their child or children.



Equivalent expressions

- $E_1(R_1,...,R_k) \equiv E_2(R_1,...,Rk)$ if for all instances $r_1,...,r_k$ of $R_1,...,R_k$ we have $E_1(r_1,...,r_k) = E_2(r_1,...,r_k)$
- It is independent of a given database.

- Example: $\sigma_{F1\ OR\ F2}(R) = \sigma_{F1}(R) \cup \sigma_{F2}(R)$



Exercise