CS411
Database Systems

05: Relational Algebra

Kazuhiro Minami

How do we query (specify what info we want from) the database?

Find all the employees who earn more than $50,000 and pay taxes in Champaign-Urbana.

- Could write in C++/Java, but who would want to?
- Instead use *high-level query languages:*
  - Theoretical: Relational algebra
  - Practical: SQL

Relational algebra has 5 operations

Input = relation(s), output = relations

- –Set union: $\cup$
- –Set difference: –
- –Selection: σ
- –Projection: π
- –Cartesian product: $\times$

Can add some syntactic sugar and/or define new operators in terms of these

Union takes the set union of two relations

OldDiagnosis

Patient	Disease
Winslett	Strep
Zhai	Meningitis
Han	Ebola

NewDiagnosis

Patient	Disease
Winslett	Hantavirus
Zhai	Meningitis
Chang	Cholera

OldDiagnosis $\cup$NewDiagnosis

Patient	Disease
Winslett	Strep
Zhai	Meningitis
Han	Ebola
Winslett	Hantavirus
Chang	Cholera

Reminder: sets have no duplicates

Input and output relations need to have the same schema

Sometimes we'd like to name or rename the output (syntactic sugar)

AllDiag(Patients, Diseases) := OldDiagnosis ∪ NewDiagnosis

or

$\rho_{\texttt{AllDiag}(Patients,\ Diseases)}$ (OldDiagnosis ∪NewDiagnosis)

Patients	Diseases
Winslett	Strep
Zhai	Meningitis
Han	Ebola
Winslett	Hantavirus
Chang	Cholera

Difference takes the set difference of two relations

OldDiagnosis

Patient	Disease
Winslett	Strep
Zhai	Meningitis
Han	Ebola

NewDiagnosis

Patient	Disease
Winslett	Hantavirus
Zhai	Meningitis
Chang	Cholera

WrongDiagnosis := OldDiagnosis – NewDiagnosis

Patient	Disease
Winslett	Strep
Han	Ebola

Selection keeps only the tuples that satisfy a particular condition

Diagnosis

Patient	Disease	Temperature
Winslett	Strep	98.9
Zhai	Meningitis	101.1
Han	Ebola	96.6
Winslett	Hantavirus	98.6
Chang	Cholera	102.3

Find all patients who have a fever

$\sigma_{\text{Temperature} > 98.6}$ (Diagnosis)

Better for everyone's sake if we write this as

σ[Temperature > 98.6] (Diagnosis)

You can write it any of these two ways in this class.

Selection conditions can be relatively complex

Attribute names

Patient

Disease

Salary

= < > ≤ ≠

Patient = "Winslett"

Salary < 40,000

Constants

"Winslett"

"Meningitis"

40,000

and or not

Patient = "Winslett" and Temperature > 98.6

Selection Example

Employee

SSN	Name	DepartmentID	Salary
999999999	John	1	30,000
777777777	Tony	1	32,000
888888888	Alice	2	45,000

Find all employees with salary more than $40,000.

$\sigma_{Salary > 40000}$ (Employee)

SSN	Name	DepartmentID	Salary
888888888	Alice	2	45,000

Projection eliminates all but the listed columns, and puts them in the listed order

Diagnosis

Patient	Disease	Temperature
Winslett	Strep	98.9
Zhai	Meningitis	101.1
Han	Ebola	96.6
Winslett	Hantavirus	98.6

List all the patients and their diagnoses

$\pi_{\text{Disease, Patient}}$ (Diagnosis)

Disease	Patient
Strep	Winslett
Meningitis	Zhai
Ebola	Han
Hantavirus	Winslett

For convenience, we may write

π [Disease, Patient] (Diagnosis)

The columns you project onto have to actually exist

π [Salary, Town] Diagnosis

π [Disease] Employee

Formally, $\pi_{A1, \ldots, An}(R)$ is a legal relational algebra expression if each of *A1*, …, *An* is an attribute of *R*

Projection Example

Employee

SSN	Name	DepartmentID	Salary
999999999	John	1	30,000
777777777	Tony	1	32,000
888888888	Alice	2	45,000

$\Pi_{\text{SSN, Name}}$ (Employee)

SSN	Name
999999999	John
777777777	Tony
888888888	Alice

The *cartesian product* of two relations is usually enormous

Diagnosis

Patient	Disease
Winslett	Strep
Zhai	Meningitis
Han	Ebola

RareDiseases

Disease
Ebola
Hantavirus

Diagnosis × RareDiseases

Patient	Diagnosis.Disease	RareDiseases.Disease
Winslett	Strep	Ebola
Zhai	Meningitis	Ebola
Han	Ebola	Ebola
Winslett	Strep	Hantavirus
Zhai	Meningitis	Hantavirus
Han	Ebola	Hantavirus

Take each possible combination of one tuple from the first relation and one tuple from the second relation

(may need to rename some attributes)

Cartesian Product Example

Employee

Name	SSN
John	999999999
Tony	777777777

Dependents

EmployeeSSN	Dname
999999999	Emily
777777777	Joe

Employee x Dependents

Name	SSN	EmployeeSSN	Dname
John	999999999	999999999	Emily
John	999999999	777777777	Joe
Tony	777777777	999999999	Emily
Tony	777777777	777777777	Joe

Relational algebra = every expression you can make using these 5 operators
(plus renaming)

Any relation name is a relational algebra expression.

If ***R*** and ***S*** are relational algebra expressions, then so are ***R* – *S***, ***R*** $\cup$ ***S*** and ***R*** $\times$ ***S***.

If ***R*** is a relational algebra expression and θ is a selection condition, then $\sigma[\theta]$***R*** is a relational algebra expression.

If ***R*** is a relational algebra expression and L is a list of attributes of ***R***, then π [L]***R*** is a relational algebra expression.

Nothing else is a relational algebra expression.

Derived RA Operations

Intersection, join

Intersection can be defined in terms of *difference*

OldDiagnosis

Patient	Disease
Winslett	Strep
Zhai	Meningitis
Han	Ebola

NewDiagnosis

Patient	Disease
Winslett	Hantavirus
Zhai	Meningitis
Chang	Cholera

RightDiagnosis = OldDiagnosis ∩ NewDiagnosis

= OldDiagnosis – (OldDiagnosis – NewDiagnosis)

More generally, R ∩ S = R – (R – (S)).

Patient	Disease
Zhai	Meningitis

A ***join*** is a cartesian product followed immediately by a selection

OldDiagnosis

Patient	Disease
Winslett	Strep
Zhai	Meningitis
Han	Ebola

NewDiagnosis

Patient	Disease
Winslett	Hantavirus
Zhai	Meningitis
Chang	Cholera

Who has an old diagnosis that is different from one of their new diagnoses?

π [#1] $\boxed{\sigma \text{ [\#1 = \#3 and \#2} \neq \text{\#4] (OldDiagnosis} \times \text{NewDiagnosis)}}$

A join

Patient
Winslett

How does that work?

OldDiagnosis

Patient	Disease
Winslett	Strep
Zhai	Meningitis
Han	Ebola

NewDiagnosis

Patient	Disease
Winslett	Hantavirus
Zhai	Meningitis
Chang	Cholera

Temp(Pat1, Dis1, Pat2, Dis2) = OldDiagnosis × NewDiagnosis

Pat1	Dis1	Pat2	Dis2
Winslett	Strep	Winslett	Hantavirus
Zhai	Meningitis	Winslett	Hantavirus
Han	Ebola	Winslett	Hantavirus
Winslett	Strep	Zhai	Meningitis
Zhai	Meningitis	Zhai	Meningitis
Han	Ebola	Zhai	Meningitis
Winslett	Strep	Chang	Cholera
Zhai	Meningitis	Chang	Cholera
Han	Ebola	Chang	Cholera

BothDiagnoses = σ[Pat1 = Pat2 and Dis1 ≠ Dis2] (Temp)

Temp

Pat1	Dis1	Pat2	Dis2
Winslett	Strep	Winslett	Hantavirus
~~Zhai~~	~~Meningitis~~	~~Winslett~~	~~Hantavirus~~
~~Han~~	~~Ebola~~	~~Winslett~~	~~Hantavirus~~
~~Winslett~~	~~Strep~~	~~Zhai~~	~~Meningitis~~
~~Zhai~~	~~Meningitis~~	~~Zhai~~	~~Meningitis~~
~~Han~~	~~Ebola~~	~~Zhai~~	~~Meningitis~~
~~Winslett~~	~~Strep~~	~~Chang~~	~~Cholera~~
~~Zhai~~	~~Meningitis~~	~~Chang~~	~~Cholera~~
~~Han~~	~~Ebola~~	~~Chang~~	~~Cholera~~

BothDiagnoses =
σ[Pat1 = Pat2 and Dis1 ≠ Dis2] (Temp)

BothDiagnoses

Pat1	Dis1	Pat2	Dis2
Winslett	Strep	Winslett	Hantavirus

FinalAnswer =
π [Pat1] BothDiagnoses

BothDiagnoses

Pat1	Dis1	Pat2	Dis2
Winslett	Strep	Winslett	Hantavirus

FinalAnswer

Pat1
Winslett

There is a convenient shorthand for joins

Relational algebra expressions

$$\boldsymbol{R} \bowtie_{\theta} \boldsymbol{S} \;=\; \sigma_{\theta}(\boldsymbol{R} \times \boldsymbol{S})$$

A selection condition

I'll let you write it as

$$\boldsymbol{R} \bowtie [\boldsymbol{q}]\ \boldsymbol{S}$$

This is called a ***θ-join,*** or an ***equijoin*** when *θ* is =.

Natural joins join on attributes with the same name

Employees

Emp	Dept
Winslett	Complaint
Zhai	Toy
Han	Toy

Managers

Dept	Mgr
Complaint	Mendez
Toy	Smith
Returns	Chu

Employees ⋈ Managers

Emp	Dept	Mgr
Winslett	Complaint	Mendez
Zhai	Toy	Smith
Han	Toy	Smith

A ***natural join*** is an equijoin on all attributes with the same name, followed by removal of the duplicate attributes

$\boldsymbol{R} \bowtie \boldsymbol{S}$ =

$\pi_{\text{everything but the duplicate attributes}}\ \sigma_{R.A1=S.A1 \text{ and} \ldots \text{and } R.An=S.An}\ (\boldsymbol{R} \times \boldsymbol{S})$

A1 through *An* are all the attributes ***R*** and ***S*** have in common

Natural joins don't always make sense

Emp(name, dept)   Dept (name, mgr)

Emp $\bowtie$ Dept is nonsensical

Your first real query: *who makes more than their manager?*

E(emp, dept, sal)   M(mgr, dept)

ESM(emp, sal, mgr) = π [emp, sal, mgr] (E $\bowtie$ M)

π [ESM.emp](ESM $\bowtie$[mgr = E.emp AND ESM.sal > E.sal] E)

Why???

E

Emp	Dept	Sal
Jones	Missiles	10K
Chu	Tanks	20K
Swami	Explosives	50K
Mendez	Tanks	10K
Benson	Explosives	40K

M

Mgr	Dept
Mendez	Tanks
Swami	Explosives
Jones	Missiles

E $\bowtie$ M

Emp	Dept	Sal	Mgr
Jones	Missiles	10K	Jones
Chu	Tanks	20K	Mendez
Swami	Explosives	50K	Swami
Mendez	Tanks	10K	Mendez
Benson	Explosives	40K	Swami

ESM = π [Emp, Sal, Mgr] (E $\bowtie$ M)

Emp	Sal	Mgr
Jones	10K	Jones
Chu	20K	Mendez
Swami	50K	Swami
Mendez	10K	Mendez
Benson	40K	Swami

ESM $\bowtie$ [ESM.Mgr = E.Emp AND ESM.Sal > E.Sal] E

Emp	Sal	Mgr	Emp	Dept	Sal
Chu	20K	Mendez	Mendez	Tanks	10K

You can define relational algebra on bags instead of sets (closer match to SQL)

- Union: {a,b,b,c} U {a,b,b,b,e,f,f} = {a,a,b,b,b,b,b,c,e,f,f}
  - *add* the number of occurrences
- Difference: {a,b,b,b,c,c} – {b,c,c,c,d} = {a,b,b}
  - subtract the number of occurrences
- Intersection: {a,b,b,b,c,c}   {b,b,c,c,c,c,d} = {b,b,c,c}
  - minimum of the two numbers of occurrences
- Selection: preserve the number of occurrences
- Projection: preserve the number of occurrences (no duplicate elimination)
- Cartesian product, join: no duplicate elimination

More detail in the book (Chapter 5.1)

Summary of relational algebra

Basic primitives:

σ [C] (E)
π [A_1, …, A_n] (E)
E1 $\times$ E2
E1 $\cup$ E2
E1 – E2
ρ[S(A_1, …, A_n)] (E)

Abbreviations:

E1 $\bowtie$ E2
E1 $\bowtie_C$ E2
E1 $\cap$ E2