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: ∪

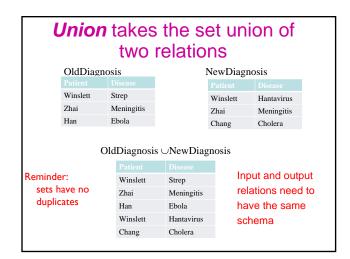
-Set difference: -

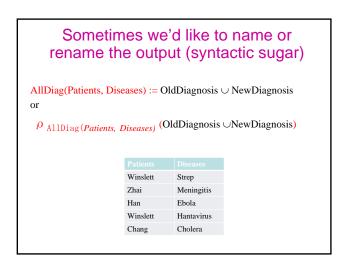
–Selection: σ

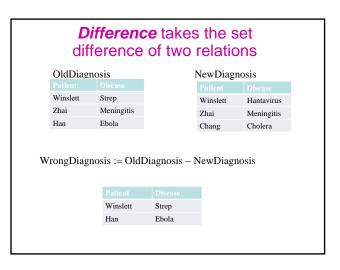
-Projection: π

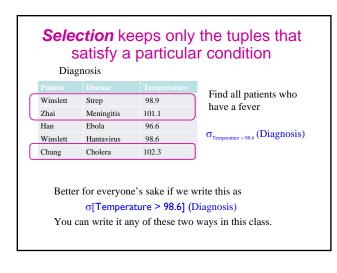
-Cartesian product: ×

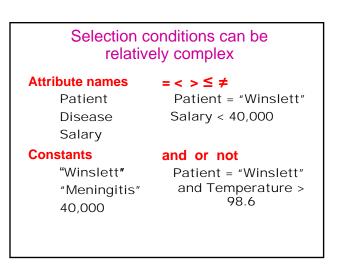
Can add some syntactic sugar and/or define new operators in terms of these









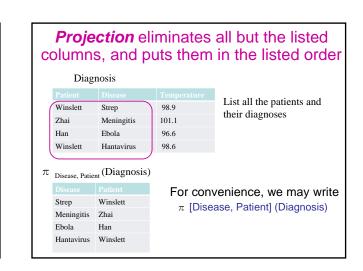


Selection Example Employee SSN Name DepartmentID Salary 99999999 John 1 30,000 77777777 Tony 32,000 88888888 Alice 45,000 Find all employees with salary more than \$40,000. $\sigma_{_{\textit{Salary}\,>\,40000}}(Employee)$ SSN Name DepartmentID Salary

45,000

88888888

Alice



The columns you project onto have to actually exist

 π [Salary, Town] Diagnosis

 π [Disease] Employee

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

Projection Example

Employee

SSN	Name	DepartmentID	Salary
99999999	John	1	30,000
77777777	Tony	1	32,000
88888888	Alice	2	45,000

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

SSN	Name
99999999	John
77777777	Tony
88888888	Alice

The *cartesian product* of two relations is usually enormous

DiagnosisPatientDiseaseWinslettStrepZhaiMeningitis

Ebola

Diagnosis × RareDiseases

Han

RareDiseases

Disease

Ebola

Hantavirus

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

Winslett Strep Ebola

Zhai Meningitis Ebola

Han Ebola Ebola

Patient Diagnosis.Disease RareDiseases.Disease

Winslett Strep Hantavirus
Zhai Meningitis Hantavirus
Han Ebola Hantavirus

(may need to rename some attributes)

relation

Cartesian Product Example

 Employee

 Name
 SSN

 John
 99999999

 Tony
 77777777

Dependents

 EmployeeSSN
 Dname

 999999999
 Emily

 777777777
 Joe

Employee x Dependents

Zimproj ee ii z ependento					
Name	SSN	EmployeeSSN	Dname		
John	99999999	99999999	Emily		
John	99999999	77777777	Joe		
Tony	77777777	99999999	Emily		
Tony	77777777	77777777	Joe		

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

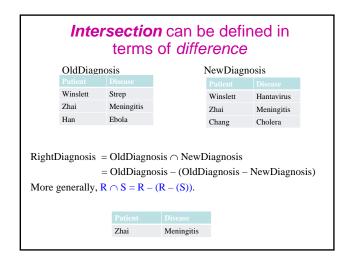
Any relation name is a relational algebra expression.

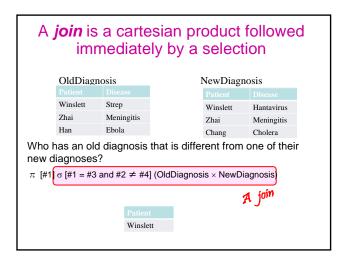
- If R and S are relational algebra expressions, then so are R S, $R \cup S$ and $R \times S$.
- If ${\pmb R}$ is a relational algebra expression and θ is a selection condition, then $\sigma[\theta]{\pmb R}$ is a relational algebra expression.
- If ${\pmb R}$ is a relational algebra expression and L is a list of attributes of ${\pmb R}$, then π [L] ${\pmb R}$ is a relational algebra expression.

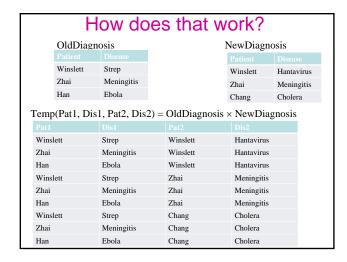
Nothing else is a relational algebra expression.

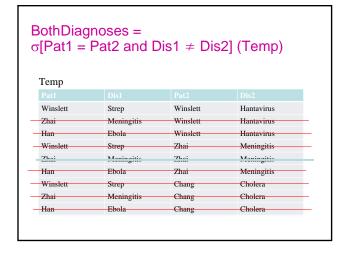
Derived RA Operations

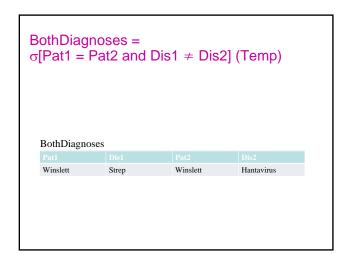
Intersection, join

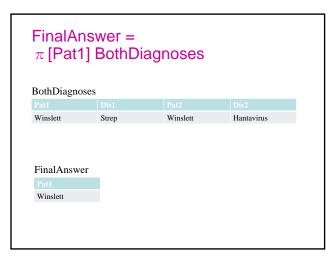


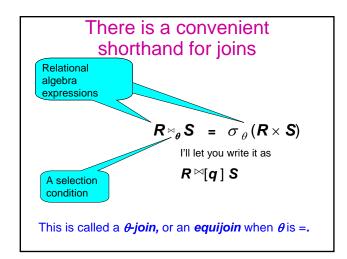


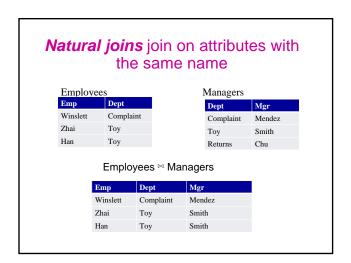


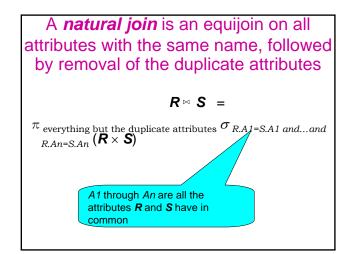








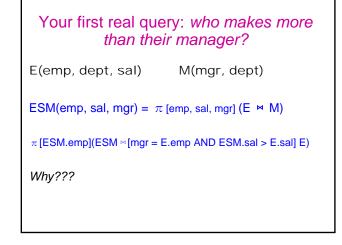


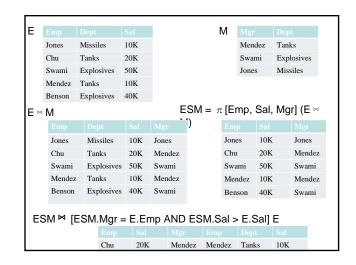


Natural joins don't always make sense

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

Emp ⋈ Dept is nonsensical





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,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:

$$\begin{array}{c} \sigma \left[C \right] \left(E \right) \\ \pi \left[A_{1},...,A_{n} \right] \left(E \right) \\ E1 \times E2 \\ E1 \cup E2 \\ E1 - E2 \\ \rho [S(A_{1},...,A_{n})] \left(E \right) \\ Abbreviations: \\ E1 \bowtie E2 \end{array}$$

E1 ⋈ E2 E1 ⋈ _C E2 E1 ∩ E2