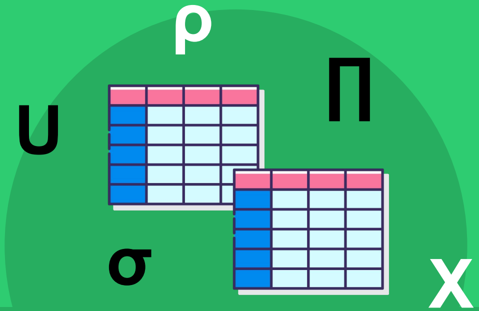
UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS
ILMU KOMPUTER

13

Relational Algebra
(part 1)

CSF2600700 - BASIS DATA



Acknowledgements

This slide is a modification to supplementary slide of
“Database System”, 6th edition, Elmasri/Navathe, 2011: **Chapter 8 The Relational Algebra and Relational Calculus** used in “Basis Data” course in academic years 2018/2019 in the Faculty of Computer Science, Universitas Indonesia and Stanford Database (CS145) Lecture 16 slides

Outline

1. Introduction to Relational Algebra (RA)

2. Unary Operations

3. Binary Operations

4. Query Tree

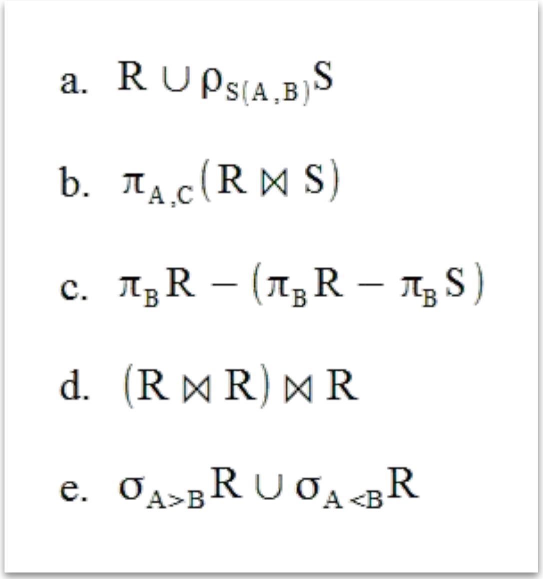
5. Additional RA Operations

Introduction

Relational Algebra (RA): The basic set of operations for relational model

Important for:

- ➔ Provides a **formal foundation** for relational model operations
- ➔ Used as basis for **implementing and optimizing queries** by RDBMS
- ➔ Incorporated into the SQL for RDBMS



a. $R \cup \rho_{S(A,B)} S$

b. $\pi_{A,C}(R \bowtie S)$

c. $\pi_B R - (\pi_B R - \pi_B S)$

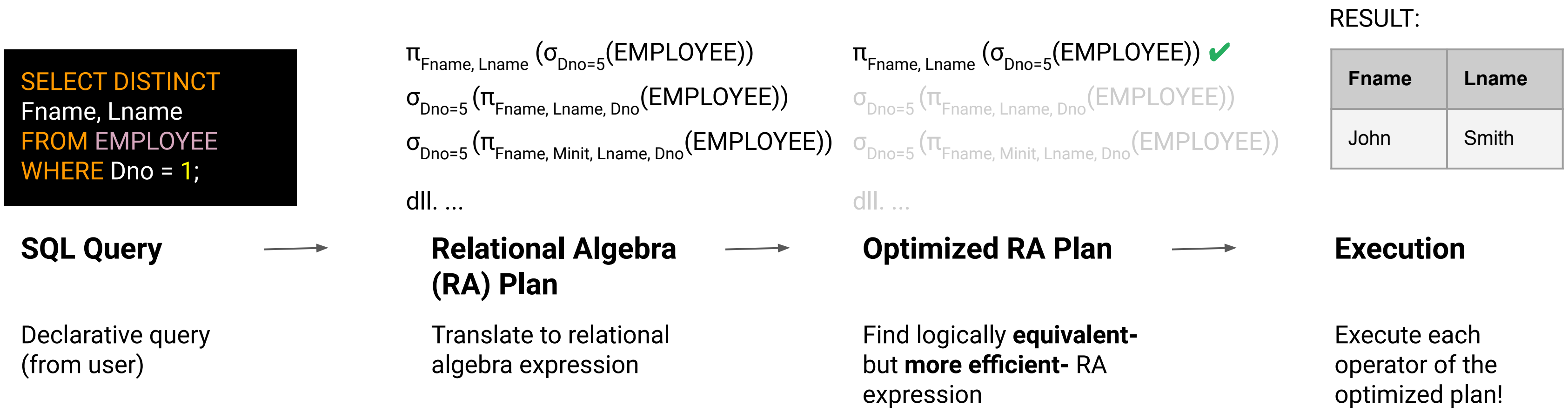
d. $(R \bowtie R) \bowtie R$

e. $\sigma_{A>B} R \cup \sigma_{A<B} R$

Illustration: http://openclassroom.stanford.edu/

RDBMS Architecture

How does a SQL engine work ?



```
SELECT DISTINCT
Fname, Lname
FROM EMPLOYEE
WHERE Dno = 1;
```

SQL Query

Declarative query (from user)

$\pi_{\text{Fname, Lname}}\ (\sigma_{\text{Dno=5}}(\text{EMPLOYEE}))$

$\sigma_{\text{Dno=5}}\ (\pi_{\text{Fname, Lname, Dno}}(\text{EMPLOYEE}))$

$\sigma_{\text{Dno=5}}\ (\pi_{\text{Fname, Minit, Lname, Dno}}(\text{EMPLOYEE}))$

dll. ...

Relational Algebra (RA) Plan

Translate to relational algebra expression

$\pi_{\text{Fname, Lname}}\ (\sigma_{\text{Dno=5}}(\text{EMPLOYEE}))$ ✔

$\sigma_{\text{Dno=5}}\ (\pi_{\text{Fname, Lname, Dno}}(\text{EMPLOYEE}))$

$\sigma_{\text{Dno=5}}\ (\pi_{\text{Fname, Minit, Lname, Dno}}(\text{EMPLOYEE}))$

dll. ...

Optimized RA Plan

Find logically **equivalent-** but **more efficient-** RA expression

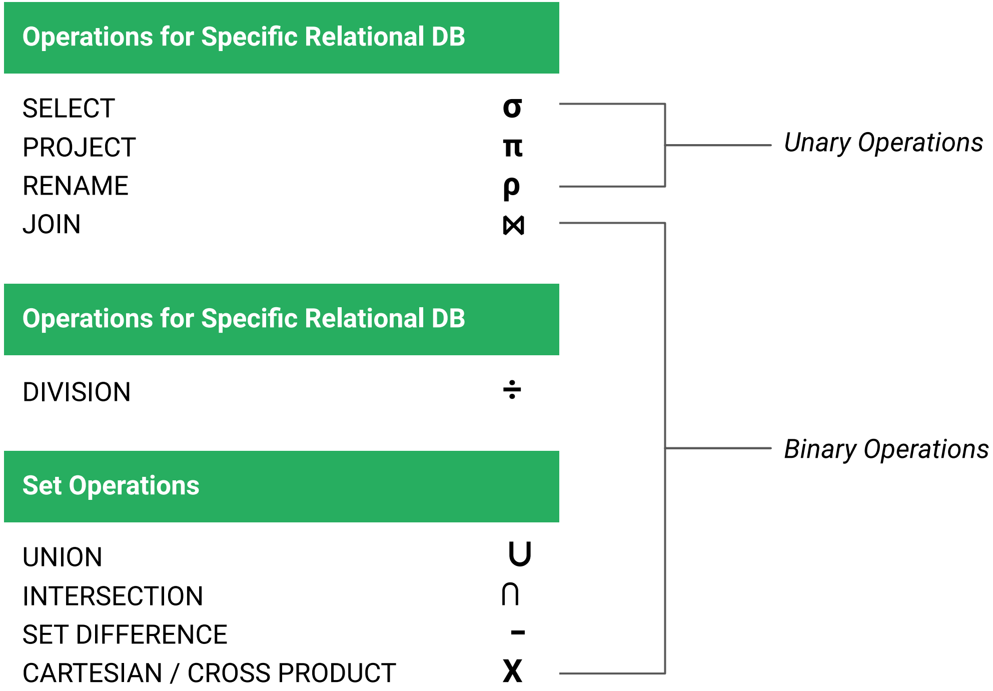
RESULT:

Fname	Lname
John	Smith

Execution

Execute each operator of the optimized plan!

Relational Algebra Operations



Operations for Specific Relational DB

SELECT	σ
PROJECT	π
RENAME	ρ
JOIN	⋈

Operations for Specific Relational DB

DIVISION	÷

Set Operations

UNION	∪
INTERSECTION	∩
SET DIFFERENCE	-
CARTESIAN / CROSS PRODUCT	X

Unary Operations

Binary Operations

Keep in mind: RA operates on sets!

RDBMSs use **multisets**, however in relational algebra formalism we will consider **sets**!

Also: we will consider the named perspective, where every attribute must have a **unique name**

➔ attribute **order does not matter**

Outline

1. Introduction to Relational Algebra (RA)

2. Unary Operations

3. Binary Operations

4. Query Tree

5. Additional RA Operations

➔ Select
➔ Project
➔ Rename

SELECT Operations

Returns all tuples which **satisfy a condition**

Notation: $\boldsymbol{\sigma}_{\textbf{<condition>}}$ **(R)**

Examples:

- ➔ $\sigma_{\text{Salary > 40000}}$ (EMPLOYEE)
- ➔ $\sigma_{\text{name = “Smith”}}$ (EMPLOYEE)

The <condition> can be:
<attr. name> <operator> <constant value>
or
<attr. name> <operator> <constant value>

Operator is one of:
=, <, ≤, > ≥, <>

Example:

STUDENTS(Sid,Sname,Gpa)

SQL:

```
SELECT *
FROM STUDENTS
WHERE Gpa > 3.5;
```

RA:

$\sigma_{\text{Gpa > 3.5}}$(STUDENTS)

SELECT Operations (Cntd.)

Another example:

Ssn	Name	Salary
1234545	John	200000
5423341	Smith	600000
4352342	Fred	500000

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

Ssn	Name	Salary
5423341	Smith	600000
4352342	Fred	500000

SELECT Operations (Cntd.)

Clauses can be connected by the **standard Boolean operators** AND, OR, and NOT to form a general selection condition

For example,
to select the tuples for all employees who either work in department 4 and make over $25,000 per year, or work in department 5 and make over $30,000, we can specify the following SELECT operation:

$$\sigma_{\text{(Dno=4 AND Salary>25000) OR (Dno=5 AND Salary>30000)}}\text{(EMPLOYEE)}$$

SELECT Operations (Cntd.)

Based on that nature, a sequence of SELECTs **can be applied in any order and any cascade** of SELECT operations can then be combine into a single SELECT operations with conjunctive (AND) condition.

$$\sigma_{<cond1>}(\sigma_{<cond2>}(\ldots(\sigma_{<cond3>}(R))\ldots)) \;=\; \sigma_{<cond1>\text{ AND}<cond2>\text{ AND }\ldots\text{ AND}<cond3>}(R)$$

The nature of SELECT Operation is **commutative**:

$$\sigma_{<cond1>}(\sigma_{<cond2>}(R)) \;=\; \sigma_{<cond2>}(\sigma_{<cond1>}(R))$$

SELECT Operations (Cntd.)

Some domains allow additional types of comparison operators, such as in domain of character string may allow the comparison operator **SUBSTRING_OF**.

$$\sigma_{\text{Fname SUBSTRING\_OF “Shoto Todoroki”}}(\text{EMPLOYEE})$$

PROJECT Operations

Select certain columns and **eliminates the other columns**, then **removes duplicates**

Notation: $\boldsymbol{\pi}_{\textbf{<attribute>}}$ **(R)**

Examples:

- ➔ $\pi_{\text{Sname, Gpa}}$ (STUDENTS)
- ➔ $\pi_{\text{Name, Salary}}$ (EMPLOYEE)

Example:

STUDENTS(Sid,Sname,Gpa)

SQL:

```
SELECT DISTINCT Sname, Gpa
FROM STUDENTS;
```

↓

RA:

$\pi_{\text{Sname, Gpa}}$ (STUDENTS)

PROJECT Operations (Cntd.)

Another example:

Ssn	Name	Salary
1234545	John	200000
5423341	Smith	600000
4352342	Fred	500000

$\pi_{\text{Name, Salary}}$ (EMPLOYEE)

Name	Salary
John	200000
Smith	600000
Fred	500000

Note that RA Operators are Compositional!

STUDENTS(Sid,Sname,Gpa)

SQL:

```
SELECT DISTINCT Sname, Gpa
FROM STUDENTS
WHERE Gpa > 3.5;
```

RA:

$\pi_{\text{Sname, Gpa}} (\sigma_{\text{Gpa > 3.5}}(\text{STUDENTS}))$ …. (1)

$\sigma_{\text{Gpa > 3.5}} (\pi_{\text{Sname, Gpa}}(\text{STUDENTS}))$ …. (2)

Are these logically equivalent?

Logical Equivalence of RA Plans

Given relations: R(A, B) and S(B, C)

Here, selection and projection commute: $\sigma_{A=5}\ (\pi_A(R)) = \pi_A\ (\sigma_{A=5}(R))$

What about here?

$\sigma_{A=5}\ (\pi_B(R)) = \pi_B\ (\sigma_{A=5}(R))$

RENAME Operation

RENAME: **Changes the schema, not the instance**

Notation: $\rho_{S(B1, \ldots, Bn)}$ **(R)** or ρ_{S} **(R)** or $\rho_{(B1, \ldots, Bn)}$ **(R)**

Notes:

- **S**: New relation name for R
- **B1, …, Bn**: R's new attribute names (**order matters**)

Note that the above notation is a shorthand version for the following proper form:

Notation: $\rho_{A1 \text{ -> } B1, \ldots, An \text{ -> } Bn}$ **(R)**

Notes:

- **A1, …, An**: Former R's attribute names to be changed (**order does not matter**)

Example:

STUDENTS(Sid,Sname,Gpa)

SQL:

```
SELECT
Sid AS studId,
Sname AS name,
Gpa AS gradePtAvg
FROM STUDENTS;
```

RA:

$\rho_{(\text{studId, name, gradePtAvg})}$ (STUDENT)

We care about this operator because we are working in a named perspective

RENAME Operation (Cntd.)

Another example:

$\rho_{(\text{studId, name, gradePtAvg})}$ (STUDENT)

STUDENTS

Sid	Sname	Gpa
001	John	200000
002	Bob	600000

→

STUDENTS

studId	name	gradePtAvg
001	John	200000
002	Bob	600000

$\rho_{\text{Stud-Rel(A, B, C)}}$ (STUDENT)

STUDENTS

Sid	Sname	Gpa
001	John	200000
002	Bob	600000

→

Stud-Rel

A	B	C
001	John	200000
002	Bob	600000

(Back to:) Note that RA Operators are Compositional!

> We can also **break down a sequences/compositional RA** by specifying intermediate result relations and renaming the relation and its attributes.

STUDENTS(Sid,Sname,Gpa)

SQL:

```sql
SELECT DISTINCT Sname, Gpa
FROM STUDENTS
WHERE Gpa > 3.5;
```

RA:

$\pi_{\text{Sname, Gpa}}\ (\sigma_{\text{Gpa > 3.5}}(\text{STUDENTS}))$ .... (1)

$\text{TEMP} \leftarrow \sigma_{\text{Gpa > 3.5}}(\text{STUDENTS})$

$\text{R(Sname, Gpa)} \leftarrow \pi_{\text{Sname, Gpa}}\ (\text{TEMP})$

$\sigma_{\text{Gpa > 3.5}}\ (\pi_{\text{Sname, Gpa}}(\text{STUDENTS}))$ .... (2)

$\text{TEMP} \leftarrow \pi_{\text{Sname, Gpa}}(\text{STUDENTS})$

$\text{R(Sname, Gpa)} \leftarrow \sigma_{\text{Gpa > 3.5}}\ (\text{TEMP})$

Note that in the RA expression (1), the inner RA expression of ($\sigma_{\text{Gpa > 3.5}}$(STUDENTS)) will includes Gpa column in its results.
Thus we can project the results using Sname and Gpa because Gpa column is included in the results of the inner RA expression.

Generalized PROJECT Operation

Also allows **functions of attributes** to be included in the projection list

Example:

EMPLOYEE(Ssn, Salary, ,Deduction, Years_service)

A report may be required to show:

Net Salary = **Salary - Deduction**
Bonus = **2000 * Years_service**, and
Tax = **0.25 * Salary**

Then, a generalized projection combined with renaming can be used as follows:

REPORT ← $\rho_{(\text{Ssn, Net\_salary, Bonus, Tax})}$ $(\pi_{\text{Ssn, } \textbf{Salary - Deduction}\text{, } \textbf{2000 * Years\_service}\text{, } \textbf{0.25 * Salary}}(\text{EMPLOYEE}))$

Outline

1. Introduction to Relational Algebra (RA)

2. Unary Operations

3. Binary Operations

4. Query Tree

5. Additional RA Operations

Binary Operations

Two relations $R_1(A_1, A_2, .., A_n)$ and $R_2(B_1, B_2, \ldots, B_n)$ are said to be **union compatible** if they have the same degree n and if $\mathrm{dom}(A_i) = \mathrm{dom}(B_i)$ for $1 \le i \le n$

Need union compatibility:

UNION

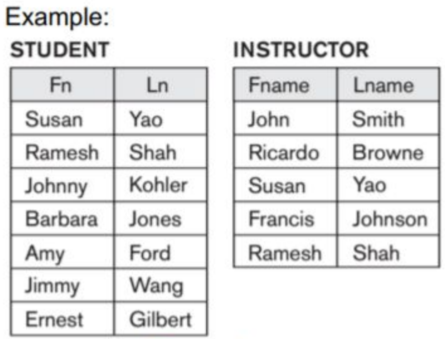
INTERSECTION

SET DIFFERENCE

Do not need union compatibility:

CARTESIAN PRODUCT

JOIN



Example:

STUDENT

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR

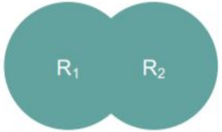
Fname	Lname
John	Smith
Ricardo	Browne
Susan	Yao
Francis	Johnson
Ramesh	Shah

Both have two attributes with the same domain

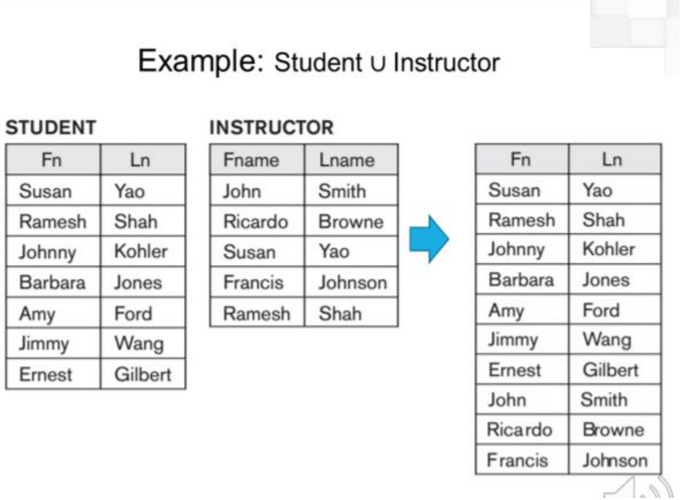
UNION Operation

Includes **all tuples** that are either in R_1 and R_2 or both **without duplicate** tuples

Notation: $\mathbf{R_1 \cup R_2}$



- ➔ Commutative
- ➔ Associative



Example: Student ∪ Instructor

STUDENT

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR

Fname	Lname
John	Smith
Ricardo	Browne
Susan	Yao
Francis	Johnson
Ramesh	Shah

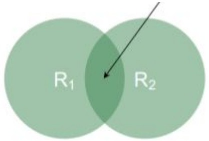
➔

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert
John	Smith
Ricardo	Browne
Francis	Johnson

INTERSECTION Operation

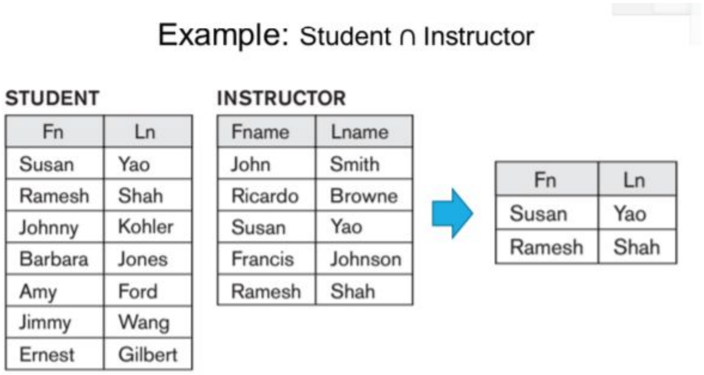
Includes **all tuples** that are **both in** R_1 and R_2

Notation: $\mathbf{R_1 \cap R_2}$



$R_1 \cap R_2 = R_1 - (R_1 - R_2)$

- ➔ Commutative
- ➔ Associative



Example: Student ∩ Instructor

STUDENT

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR

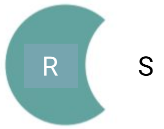
Fname	Lname
John	Smith
Ricardo	Browne
Susan	Yao
Francis	Johnson
Ramesh	Shah

Fn	Ln
Susan	Yao
Ramesh	Shah

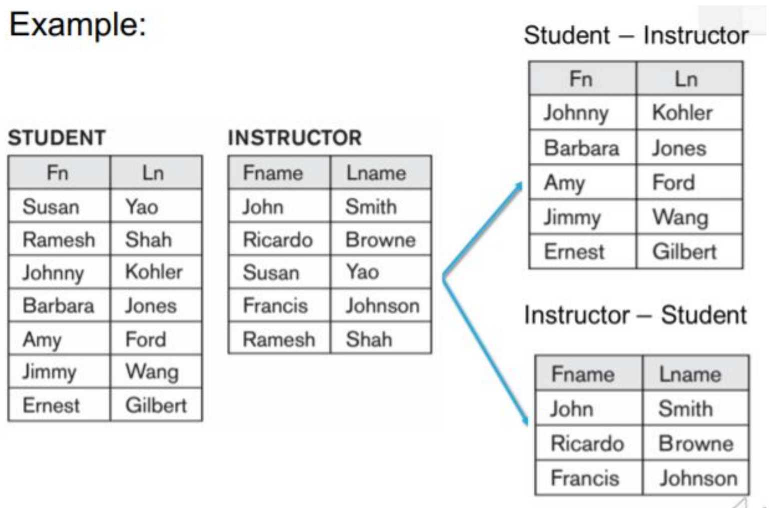
DIFFERENCE/MINUS/EXCEPT Operation

Includes **all tuples** that are **in R but not in S**

Notation: **R - S**



R - S = ((R ∪ S) - (R - S)) - (S - R)



Example:

STUDENT

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR

Fname	Lname
John	Smith
Ricardo	Browne
Susan	Yao
Francis	Johnson
Ramesh	Shah

Student – Instructor

Fn	Ln
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

Instructor – Student

Fname	Lname
John	Smith
Ricardo	Browne
Francis	Johnson

CARTESIAN/CROSS PRODUCT

Combining each tuple in R_1 with each tuple in R_2

Notation: $\mathbf{R_1 \text{ x } R_2}$

Example:

- ➔ Employee × Dependents
- ➔ People × Students

Rare in practice; mainly used to express joins

Useful when followed by a selection condition (see example in the next page)

Example:

People(ssn,pname,address)
Students(sid,sname,gpa)

SQL:

```
SELECT  *
FROM PEOPLE, STUDENTS;
```

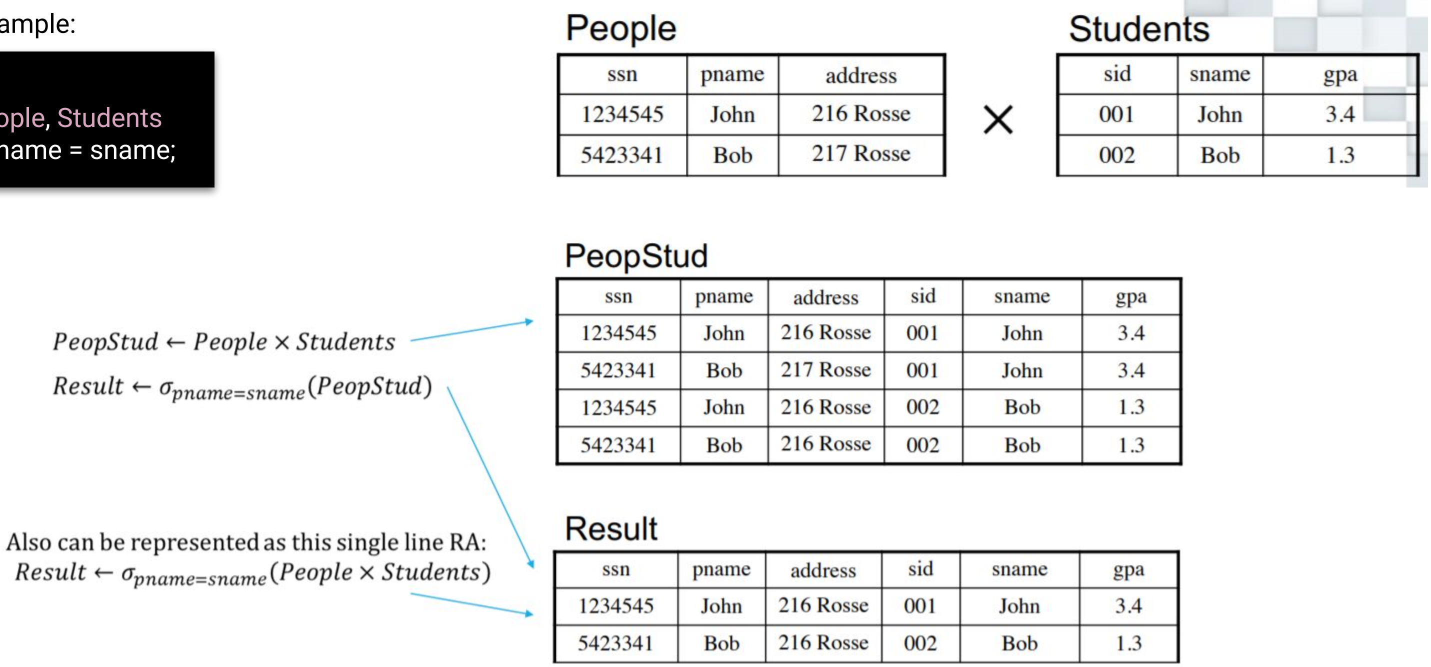
RA:

PEOPLE x STUDENTS)

CARTESIAN/CROSS PRODUCT (Cntd.)

Another example:

```
SELECT *
FROM People, Students
WHERE pname = sname;
```



People

ssn	pname	address
1234545	John	216 Rosse
5423341	Bob	217 Rosse

×

Students

sid	sname	gpa
001	John	3.4
002	Bob	1.3

$PeopStud \leftarrow People \times Students$

$Result \leftarrow \sigma_{pname=sname}(PeopStud)$

PeopStud

ssn	pname	address	sid	sname	gpa
1234545	John	216 Rosse	001	John	3.4
5423341	Bob	217 Rosse	001	John	3.4
1234545	John	216 Rosse	002	Bob	1.3
5423341	Bob	216 Rosse	002	Bob	1.3

Also can be represented as this single line RA:

$Result \leftarrow \sigma_{pname=sname}(People \times Students)$

Result

ssn	pname	address	sid	sname	gpa
1234545	John	216 Rosse	001	John	3.4
5423341	Bob	216 Rosse	002	Bob	1.3

(Theta) JOIN

To **combine related tuples** from two relations **into single tuples**. In database JOIN operation is important because it allows us to process relationship among relations

Notation: $\mathbf{R_1 \bowtie_{<join\ condition>} R_2}$

<join condition> = <condition> AND <condition> AND … AND <condition>

Each <condition> in the form A_i θ B_j and θ is one of =, <, <=, >, >=, <>
A_i is attribute in R_1, B_j is attribute in R_2.

Previous CROSS PRODUCT’s example can be replaced by this JOIN operation.

Result = (PEOPLE) $\bowtie_{\mathbf{Pname=Sname}}$ **(STUDENTS)**

➔ This JOIN example is called EQUIJOIN = a JOIN operation uses only equality comparison operator

(Natural) JOIN

A theta JOIN + PROJECT operation.

- ➔ Join two relations, then use projection to select desired columns

An EQUIJOIN with the removal of superfluous attribute.

- ➔ See example in the next page.

Notation: $\mathbf{R_1 * R_2}$

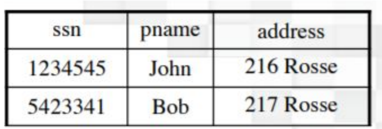
or

Notation: $\mathbf{R_1} \bowtie \mathbf{R_2}$ without join condition

The attributes to be joined **must have the same name** in both relations, otherwise we have to rename one of them first

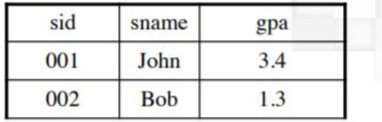
(Natural) JOIN (Cntd.)

PEOPLE

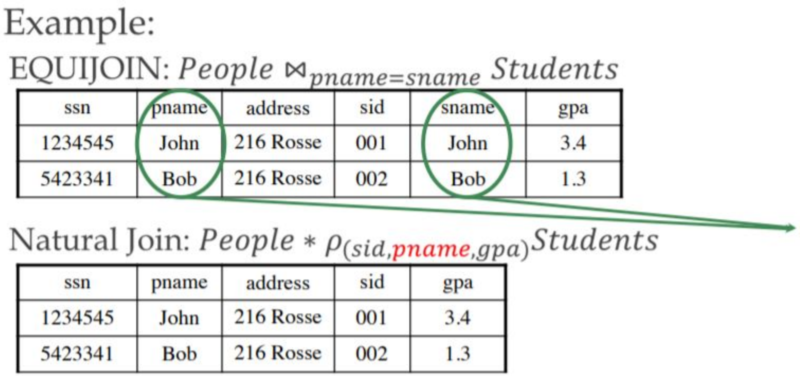


ssn	pname	address
1234545	John	216 Rosse
5423341	Bob	217 Rosse

STUDENTS



sid	sname	gpa
001	John	3.4
002	Bob	1.3



Example:

EQUIJOIN: $People \bowtie_{pname=sname} Students$

ssn	pname	address	sid	sname	gpa
1234545	John	216 Rosse	001	John	3.4
5423341	Bob	216 Rosse	002	Bob	1.3

Natural Join: $People * \rho_{(sid,pname,gpa)} Students$

ssn	pname	address	sid	gpa
1234545	John	216 Rosse	001	3.4
5423341	Bob	216 Rosse	002	1.3

Superfluous Attribute

If the two joined relations already have the same attribute name as join attribute, then renaming is not necessary

(Semi) JOIN

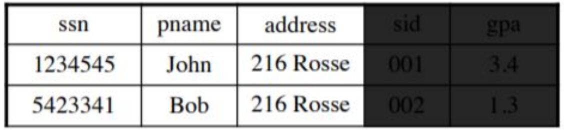
Notation: $\mathbf{R \ltimes_{<join\ condition>} S}$

$R \ltimes_{<\text{join condition}>} S = \pi_{A1,\ A2,\ \ldots,\ An} (R \bowtie S)$

Where A1, A2, …, An are attributes in R

Example:

$\text{People} \ltimes_{\text{pname = sname}} \text{Students}$



ssn	pname	address	sid	gpa
1234545	John	216 Rosse	001	3.4
5423341	Bob	216 Rosse	002	1.3

Example:

People(ssn,pname,address)
Students(sid,sname,gpa)

SQL:

```
SELECT DISTINCT ssn, pname, address
FROM People, Students
WHERE pname = sname;
```

RA:

$\text{People} \ltimes_{\text{pname = sname}} \text{Students}$

Example: Converting Select-From-Where Query -> RA

Example:

```
People(ssn,pname,address)
Students(sid,sname,gpa)
```

SQL:

```
SELECT DISTINCT gpa, address
FROM Students S, People P
WHERE gpa > 3.5 AND sname = pname;
```

→

$$\Pi_{gpa,address}(\sigma_{gpa>3.5} \\ ((\rho_S\,(\mathrm{STUDENTS})) \bowtie_{Sname\ =\ Pname} \\ (\rho_P\,(\mathrm{PEOPLE})))$$

How do we represent this query in RA?

Division

Notation: **R ÷ S**

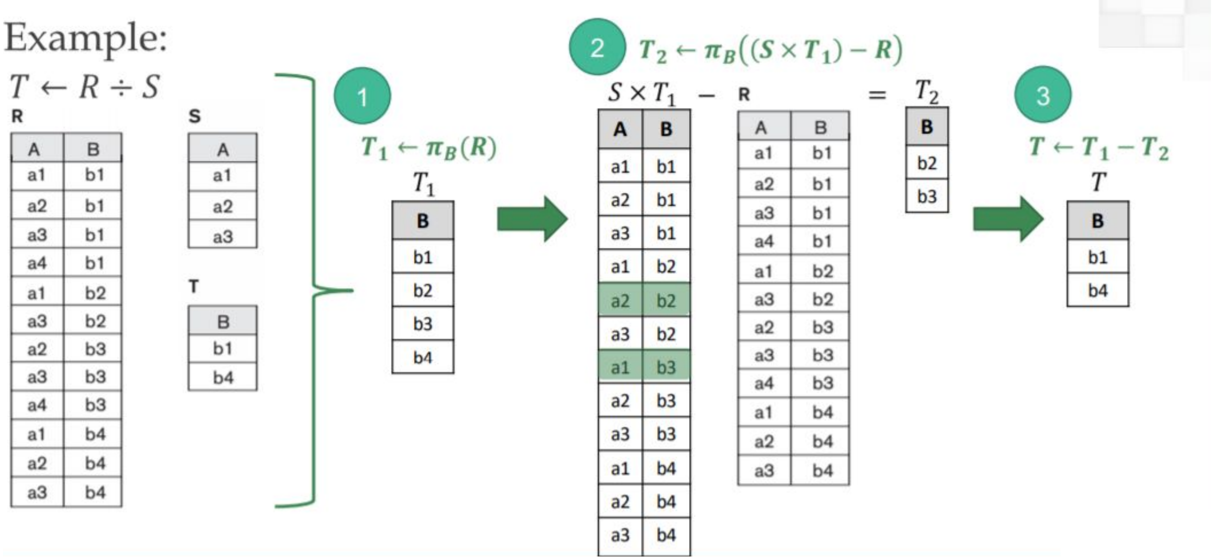
Division .is used when we wish **to express queries with “all”**.

For example:

- ➔ “Which persons have a loyal customer's card at ALL the clothing boutiques in town X?”
- ➔ “Which persons have a bank account at ALL the banks in the country?”
- ➔ “Which students are registered on ALL the courses given by Soni”
- ➔ “Which boys are registered on those courses that are taken by ALL the girls?”

Division (Cntd.)

Division **T ⟵ R ÷ S** can be expressed as a sequence of these operations.



Example:

$T \leftarrow R \div S$

R

A	B
a1	b1
a2	b1
a3	b1
a4	b1
a1	b2
a3	b2
a2	b3
a3	b3
a4	b3
a1	b4
a2	b4
a3	b4

S

A
a1
a2
a3

T

B
b1
b4

1

$T_1 \leftarrow \pi_B(R)$

T_1

B
b1
b2
b3
b4

2

$T_2 \leftarrow \pi_B((S \times T_1) - R)$

$S \times T_1$ − R = T_2

$S \times T_1$

A	B
a1	b1
a2	b1
a3	b1
a1	b2
a2	b2
a3	b2
a1	b3
a2	b3
a3	b3
a1	b4
a2	b4
a3	b4

R

A	B
a1	b1
a2	b1
a3	b1
a4	b1
a1	b2
a3	b2
a2	b3
a3	b3
a4	b3
a1	b4
a2	b4
a3	b4

T_2

B
b2
b3

3

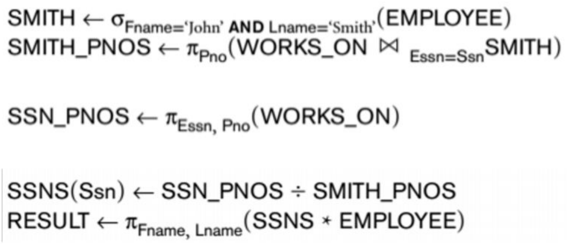
$T \leftarrow T_1 - T_2$

T

B
b1
b4

Division (Cntd.)

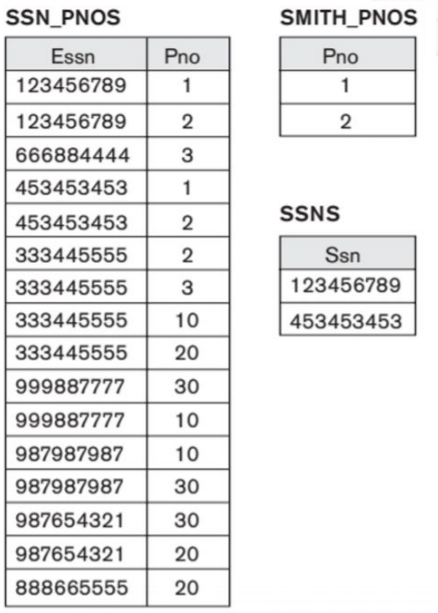
Example:
Retrieve the names of employees who work on all projects that 'John Smith' works on



$$\text{SMITH} \leftarrow \sigma_{\text{Fname='John' } \textbf{AND} \text{ Lname='Smith'}}(\text{EMPLOYEE})$$
$$\text{SMITH\_PNOS} \leftarrow \pi_{\text{Pno}}(\text{WORKS\_ON} \bowtie_{\text{Essn=Ssn}} \text{SMITH})$$

$$\text{SSN\_PNOS} \leftarrow \pi_{\text{Essn, Pno}}(\text{WORKS\_ON})$$

$$\text{SSNS(Ssn)} \leftarrow \text{SSN\_PNOS} \div \text{SMITH\_PNOS}$$
$$\text{RESULT} \leftarrow \pi_{\text{Fname, Lname}}(\text{SSNS} * \text{EMPLOYEE})$$



SSN_PNOS

Essn	Pno
123456789	1
123456789	2
666884444	3
453453453	1
453453453	2
333445555	2
333445555	3
333445555	10
333445555	20
999887777	30
999887777	10
987987987	10
987987987	30
987654321	30
987654321	20
888665555	20

SMITH_PNOS

Pno
1
2

SSNS

Ssn
123456789
453453453

Summary

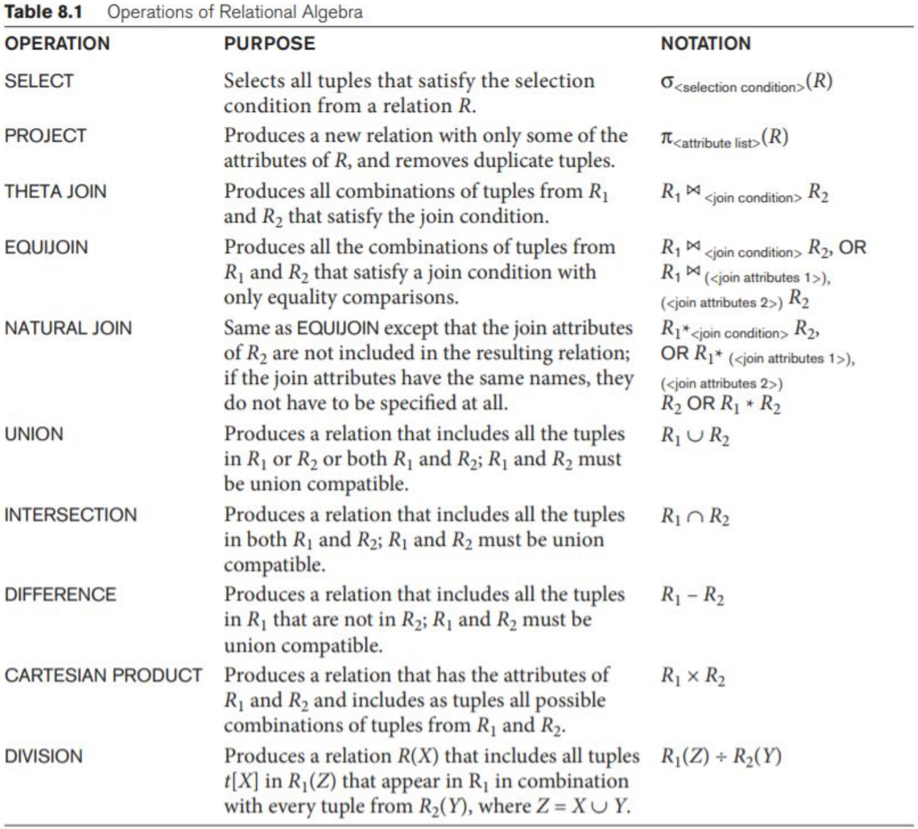


Table 8.1 Operations of Relational Algebra

OPERATION	PURPOSE	NOTATION
SELECT	Selects all tuples that satisfy the selection condition from a relation R.	$\sigma_{\text{<selection condition>}}(R)$
PROJECT	Produces a new relation with only some of the attributes of R, and removes duplicate tuples.	$\pi_{\text{<attribute list>}}(R)$
THETA JOIN	Produces all combinations of tuples from R_1 and R_2 that satisfy the join condition.	$R_1 \bowtie_{\text{<join condition>}} R_2$
EQUIJOIN	Produces all the combinations of tuples from R_1 and R_2 that satisfy a join condition with only equality comparisons.	$R_1 \bowtie_{\text{<join condition>}} R_2$, OR $R_1 \bowtie_{(\text{<join attributes 1>}),\ (\text{<join attributes 2>})} R_2$
NATURAL JOIN	Same as EQUIJOIN except that the join attributes of R_2 are not included in the resulting relation; if the join attributes have the same names, they do not have to be specified at all.	$R_1 *_{\text{<join condition>}} R_2$, OR $R_1 *_{(\text{<join attributes 1>}),\ (\text{<join attributes 2>})} R_2$ OR $R_1 * R_2$
UNION	Produces a relation that includes all the tuples in R_1 or R_2 or both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cup R_2$
INTERSECTION	Produces a relation that includes all the tuples in both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cap R_2$
DIFFERENCE	Produces a relation that includes all the tuples in R_1 that are not in R_2; R_1 and R_2 must be union compatible.	$R_1 - R_2$
CARTESIAN PRODUCT	Produces a relation that has the attributes of R_1 and R_2 and includes as tuples all possible combinations of tuples from R_1 and R_2.	$R_1 \times R_2$
DIVISION	Produces a relation $R(X)$ that includes all tuples $t[X]$ in $R_1(Z)$ that appear in R_1 in combination with every tuple from $R_2(Y)$, where $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$

Exercise

1. Retrieve the name and address of all employees who work for the ‘Research’ department.
2. Retrieve the names of all employees in department 5 who work more than 10 hours per week on the ProductX project.
3. List the names of all employees who have a dependent with the same first name as themselves.
4. Find the names of all employees who are directly supervised by ‘Franklin Wong’.
5. Find the names of employees who work on all the projects controlled by department number 5.

Exercise for Discussion

1. Diberikan schema R(A, B, C, D) dan S(A, C, E). Berikan hasil keluaran skema dari expression $R \bowtie S$.
2. Diberikan schema R(A, B, C) dan S(D, E). Berikan hasil keluaran skema dari expression $R \bowtie S$.
3. Diberikan schema R(A, B) dan S(C, D). Berikan hasil keluaran skema dari expression $R \ltimes_{B=C} S$.
4. Tampilkan fname, lname semua employee yang menjadi supervisor.
5. Tampilkan fname, lname semua employee yang menjadi supervisor, namun tidak menjabat sebagai manager.
6. Tampilkan fname dan lname manager dari department yang mengontrol project dan fname dan lname supervisor dimana department tempat supervisor bekerja mengontrol project.
7. Tampilkan pname project yang dikerjakan oleh semua employee.
8. Tampilkan fname dan lname dari employee yang mengerjakan semua project yang dikerjakan oleh supervisor dari employee tersebut.

Q&A