COMP9311: DATABASE SYSTEMS

Term 1 2024

Week 2 – Relational Algebra

By Xiaoyang Wang, CSE UNSW

Disclaimer: the course materials are sourced from previous offerings of COMP9311 and COMP3311

Motivation

We know how to store data ... i.e., we modelled our data in relational data model -> then created tables to store them into a relational database.

How do we manipulate or retrieve (interesting) the data?

- We needed a formal language to specify data (tuples) from the relational model

The basic set of operations for the relational model is **Relational Algebra**

- Edgar F. Codd (1970): Relational Algebra, mathematical foundation for relational data management
- supports basic retrieval requests (queries) -> the result of a query is also a **relation**
- A sequence of relational algebra operations form a relational algebra expression -> results in a **relation**

Motivation

The relational algebra is very important for several reasons.

- First, it provides a formal foundation for relational model operations.
- Second, and perhaps more important, it is used as a basis for implementing and optimizing queries in the query processing and optimization modules that are integral parts of relational database management systems (RDBMSs),
- Third, some of its concepts are incorporated into the SQL, the standard query language for relational database management systems

Characteristics of an Algebra

An algebra expression:

- is constructed with operators from atomic operands (constants, variables, ...)
- can be evaluated
- can be equivalent to another expression
  - ...if they return the same result for all values of the variables

This equivalence concept gives rise to an algebraic identity between expressions

An **algebraic identity** is an equality that holds for any values of its variables.

The value of an expression is independent of its context

- e.g.,5+3 has the same value, no matter whether it occurs as

  10 - (5 + 3) or 4 × (5 + 3)

Atomic expressions:

numbers and variables

Operators: +, -, ×, /

Identitities: $x+y=y+x$
$x \times (y + z) = x \times y + x \times z$
... and so on

Consequence: subexpressions can be replaced by equivalent expressions without changing the meaning of the entire expression

Relational Algebra: Principles

Atoms are relations

Operators are defined for arbitrary instances of a relation

The following two results have to be defined for each operator:

- result schema
- result instance

It specifies operations on relations to define new relations:

- **Unary Relational Operations**: Select, Project
- **Operations from Set Theory**: Union, Intersection, Difference, Cartesian Product
- **Binary Relational Operations**: Join, Divide.

"Equivalent" to SQL query language ... Relational Algebra concepts reappear in SQL

Used inside a DBMS, to express query plans

Projection and Selection

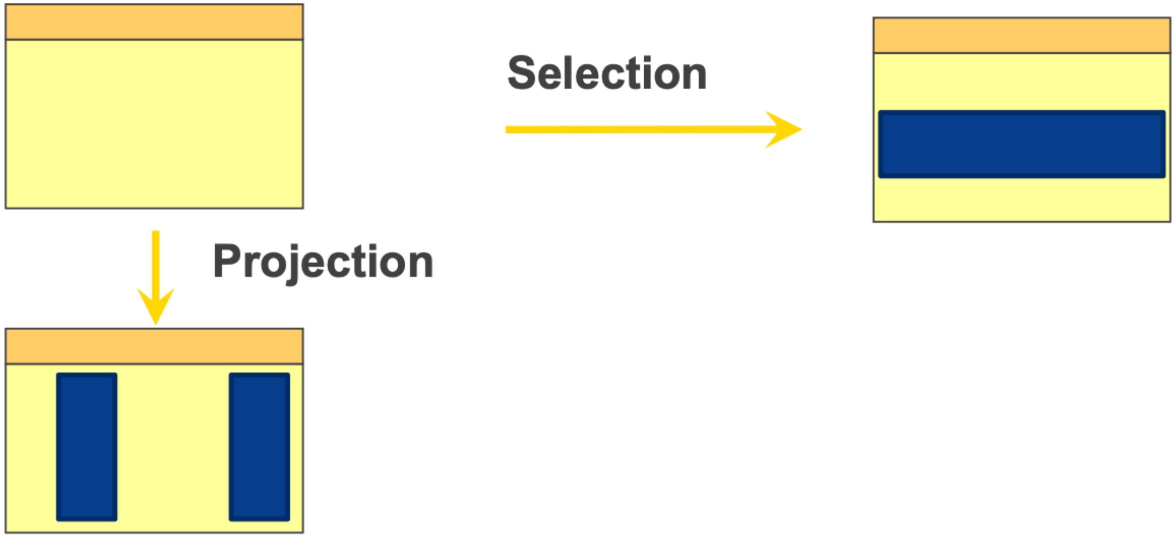
Two “orthogonal” operators

Selection:

- horizontal decomposition

Projection:

- vertical decomposition



Selection

The SELECT operation/predicate is used to select a subset of the tuples of a relation R, satisfying some condition C.

Notation: $\sigma_C(R)$

Intuition: Filters out all tuples that do not satisfy select condition C

Result:

- Schema: the schema of R
- Instance: the set of all tuples that satisfy C

Selection: Example

STUDENT

studno	name	hons	tutor	year
s1	jones	ca	bush	2
s2	brown	cis	kahn	2
s3	smith	cs	goble	2
s4	bloggs	ca	goble	1
s5	jones	cs	zobel	1
s6	peters	ca	kahn	3

$\sigma_{name=\text{'bloggs'}}(STUDENT)$ =

STUDENT

studno	name	hons	tut or	year
s4	bloggs	ca	goble	1

Selection Condition

Elementary conditions

<attr> op <val>, <attr> op <attr>, <expr> op <expr>

where op is “=”, “<”, “<=”, (on numbers and strings)

“LIKE” (for string comparisons),...

Example:

- age < 24
- phone LIKE ‘0039%’
- salary + commission < 24000

Combined conditions (using Boolean connectives)

C1 and C2 or C1 or C2 or not C

Selection: Example

STUDENT

studno	name	hons	tutor	year
s1	jones	ca	bush	2
s2	brown	cis	kahn	2
s3	smith	cs	goble	2
s4	bloggs	ca	goble	1
s5	jones	cs	zobel	1
s6	peters	ca	kahn	3

$$\sigma_{((\text{hons}=\text{'cs'}) \text{ or } (\text{hons} = \text{'ca'})) \text{ and } (\text{tutor}=\text{'goble'})}(\text{STUDENT}) =$$

STUDENT

studno	name	hons	tut or	year
s3	smiths	cs	goble	2
s4	bloggs	ca	goble	1

Properties of Selection

For Condition C1 and C2

Selection splitting

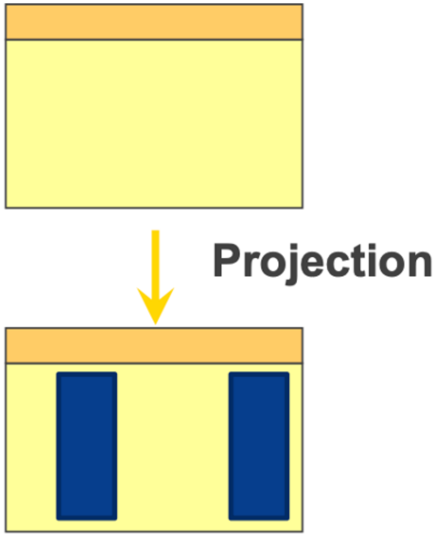
$$\sigma_{C1\ and\ C2}\ (R) = \sigma_{C1}(\sigma_{C2}(R))$$

Also, selection is commutative

$$\sigma_{C1}\big(\sigma_{C2}(R)\big) = \sigma_{C2}(\sigma_{C1}(R))$$

Projection

The PROJECT operation is used to project a subset of the attributes (column) of a relation, denoted by:



General form: $\pi_{A1,\ldots,Ak}(R)$

where R is a relation and A1,...,Ak are attributes of R

Result:

- Schema: (A1,...,Ak)
- Instance: the set of all subtuples t[A1,...,Ak] where t belongs to R

Intuition: “removes” all attributes that are not in projection list

Projection: Example

STUDENT

studno	name	hons	tutor	year
s1	jones	ca	bush	2
s2	brown	cis	kahn	2
s3	smith	cs	goble	2
s4	bloggs	ca	goble	1
s5	jones	cs	zobel	1
s6	peters	ca	kahn	3

$\pi_{tutor}(\text{STUDENT})$ =

tutor
bush
kahn
goble
zobel

Relational Algebra is based on *sets*, so no duplicates are allowed.

- The PROJECT operation *removes any duplicate tuples*, so the result of the PROJECT operation is a set of distinct tuples, and this is known as **duplicate elimination**.

Operators Can Be Nested

Who is the tutor of the student named “Bloggs”?

STUDENT

studno	name	hons	tutor	year
s1	jones	ca	bush	2
s2	brown	cis	kahn	2
s3	smith	cs	goble	2
s4	bloggs	ca	goble	1
s5	jones	cs	zobel	1
s6	peters	ca	kahn	3

$$\pi_{tutor}\left(\ \sigma_{name=\text{'bloggs'}}(STUDENT)\ \right)$$

=

tutor
goble

Properties of Projection

Consider $\pi_{list1}(\pi_{list2}(R))$

If list2 contains all the attributes in list1

then $\pi_{list1}\left(\pi_{list2}(R)\right) = \pi_{list1}(R)$

Else the operation is not well defined.

Question: Is projection operator commutative with selection?

$$\pi_{A1,\dots,Am}\left(\sigma_C(R)\right) = \sigma_C(\pi_{A1,\dots,Am}(R))$$

Set Operators

Observations:

Instances of relations are sets

-> we can form **unions**, **intersections**, and **differences**

Set algebra operators can only be applied to relations with identical attributes,

- same number of attributes
- same attribute names
- same domains
- (i.e., set operation compatibility)

Union

CS-Student

Studno	Name	Year
s1	Egger	5
s3	Rossi	4
s4	Maurer	2

Master-Student

Studno	Name	Year
s1	Egger	5
s2	Neri	5
s3	Rossi	4

CS-Student ∪ Master-Student

Studno	Name	Year
s1	Egger	5
s2	Neri	5
s3	Rossi	4
s4	Maurer	2

Intersection

CS-Student

Studno	Name	Year
s1	Egger	5
s3	Rossi	4
s4	Maurer	2

Master-Student

Studno	Name	Year
s1	Egger	5
s2	Neri	5
s3	Rossi	4

CS-Student ∩ Master-Student

Studno	Name	Year
s1	Egger	5
s3	Rossi	4

Difference

CS-Student

Studno	Name	Year
s1	Egger	5
s3	Rossi	4
s4	Maurer	2

Master-Student

Studno	Name	Year
s1	Egger	5
s2	Neri	5
s3	Rossi	4

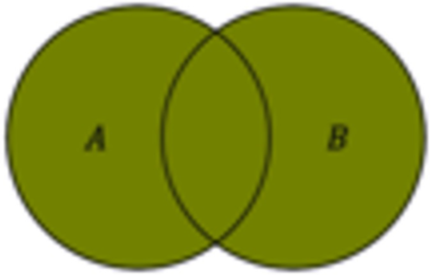
CS-Student - Master-Student

Studno	Name	Year
s4	Maurer	2

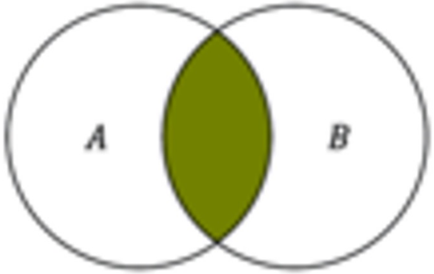
Summary

Operations on Relations

- Union: $A \cup B$



- Intersection: $A \cap B$



- Difference: $A - B$

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

The names of persons who are either a student or a researcher?

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

The names of persons who are either a student or a researcher?

$\pi_{\{name\}}(STUDENT \cup RESEARCHER)$

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

The names of persons who are a student and a researcher?

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

The names of persons who are a student and a researcher?

$$\pi_{\{name\}}(STUDENT \cap RESEARCHER)$$

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

The names of persons who are a student but not a researcher?

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

The names of persons who are a student but not a researcher?

$\pi_{\{name\}}(STUDENT - RESEARCHER)$

Cartesian Product

General form:

$$R \times S$$

where R and S are arbitrary relations

Result

- Schema: (A1,...,Am,B1,...,Bn), where (A1,...,Am) is the schema of R and (B1,...,Bn) is the schema of S.
- (If A is an attribute of both, R and S, then $R \times S$ contains the disambiguated attributes R.A and S.A.)
- Instance: the set of all concatenated tuples (t,s) where $t \in R$ and $s \in S$

Cartesian Product: Example

STUDENT

studno	name	hons	tutor	year
s1	jones	ca	bush	2
s2	brown	cis	kahn	2
s3	smith	cs	goble	2
s4	bloggs	ca	goble	1
s5	jones	cs	zobel	1
s6	peters	ca	kahn	3

STAFF

lecturer	roomno
kahn	IT206
bush	2.26
goble	2.82
zobel	2.34
watson	IT212
woods	IT204
capon	A14
lindsey	2.10
barringer	2.125

Brings all information from relations into one without applying any conditions

What's the point of this?

studno	name	hons	tutor	year	lecturer	roomno
s1	jones	ca	bush	2	kahn	IT206
s1	jones	ca	bush	2	bush	2.26
s1	jones	ca	bush	2	goble	2.82
s1	jones	ca	bush	2	zobel	2.34
s1	jones	ca	bush	2	watson	IT212
s1	jones	ca	bush	2	woods	IT204
s1	jones	ca	bush	2	capon	A14
s1	jones	ca	bush	2	lindsey	2.10
s1	jones	ca	bush	2	barringer	2.125
s2	brown	cis	kahn	2	kahn	IT206
s2	brown	cis	kahn	2	bush	2.26
s2	brown	cis	kahn	2	goble	2.82
s2	brown	cis	kahn	2	zobel	2.34
s2	brown	cis	kahn	2	watson	IT212
s2	brown	cis	kahn	2	woods	IT204
s2	brown	cis	kahn	2	capon	A14
s2	brown	cis	kahn	2	lindsey	2.10
s2	brown	cis	kahn	2	barringer	2.125
s3	smith	cs	goble	2	kahn	IT206
s3	smith	cs	goble	2	bush	2.26
s3	smith	cs	goble	2	goble	2.82
s3	smith	cs	goble	2	zobel	2.34
s3	smith	cs	goble	2	watson	IT212
s3	smith	cs	goble	2	woods	IT204
s3	smith	cs	goble	2	capon	A14
s3	smith	cs	goble	2	lindsey	2.10
s3	smith	cs	goble	2	barringer	2.125
s4	bloggs	ca	goble	1	kahn	IT206

Where are the Tutors of Students?

To answer the query

- "For each student, identified by name and student number, return the name of the tutor and their office number"

We have to

- combine tuples from Student and Staff
- that satisfy "Student.tutor=Staff.lecturer"
- and keep the attributes studno, name, (tutor or lecturer), and roomno.

In relational algebra:

$$\pi_{studno,name,lecturer,roomno}(\sigma_{tutor=lecturer}(Student \times Staff))$$

Join

The most used operator in the relational algebra.

- Allows us to establish connections among data in different relations, taking advantage of the ”data-based" nature of the relational model.
- Join is used to combine related tuples from two relations into single "longer" tuples.

Three main versions of the join:

- "natural" join: takes attribute names into account;
- "theta" join.
- “equi” join (a special form of theta join)
- all denoted by the symbol $\bowtie$

Natural Join

Student ⋈ Enrol

Implicit join based on **common** attributes

The tuples in the resulting relation are obtained by combining tuples in the operands with equal values on the common attributes

Common attributes appear once in the results

STUDENT

studno	name	hons	tutor	year
s1	jones	ca	bush	2
s2	brown	cis	kahn	2
s3	smith	cs	goble	2
s4	bloggs	ca	goble	1
s5	jones	cs	zobel	1
s6	peters	ca	kahn	3

ENROL

stud no	course no	lab mark	exam mark
s1	cs250	65	52
s1	cs260	80	75
s1	cs270	47	34
s2	cs250	67	55
s2	cs270	65	71
s3	cs270	49	50
s4	cs280	50	51
s5	cs250	0	3
s6	cs250	2	7

Natural Join

STUDENT

studno	name	hons	tutor	year
s1	jones	ca	bush	2
s2	brown	cis	kahn	2
s3	smith	cs	goble	2
s4	bloggs	ca	goble	1
s5	jones	cs	zobel	1
s6	peters	ca	kahn	3

ENROL

stud no	course no	lab mark	exam mark
s1	cs250	65	52
s1	cs260	80	75
s1	cs270	47	34
s2	cs250	67	55
s2	cs270	65	71
s3	cs270	49	50
s4	cs280	50	51
s5	cs250	0	3
s6	cs250	2	7

```
 stuno |  name  | hons | tutor | year | courseno | labmark | exammark
-------+--------+------+-------+------+----------+---------+----------
 s1    | jones  | ac   | bush  |    2 | cs250    |      65 |       52
 s1    | jones  | ac   | bush  |    2 | cs260    |      80 |       75
 s1    | jones  | ac   | bush  |    2 | cs270    |      47 |       34
 s2    | brown  | is   | kahn  |    2 | cs250    |      67 |       55
 s2    | brown  | is   | kahn  |    2 | cs270    |      65 |       71
 s3    | smith  | cs   | goble |    2 | cs270    |      49 |       50
 s4    | bloggs | ac   | goble |    1 | cs250    |      50 |       51
 s5    | jones  | cs   | zobel |    1 | cs250    |       0 |        3
 s6    | peters | ac   | kahn  |    3 | cs250    |       2 |        7
(9 rows)
```

Theta-Join and Equi-Join

Theta-Join:

- The most general form of JOIN ...
- Theta join combines tuples from different relations provided they satisfy the theta condition. The join condition is denoted by the symbol θ.
- Theta join can use comparison operators and common attributes are not required.

$$Student \bowtie_{\theta} Enrol$$

- The results include the 'joined' attributes from both relations
- The attribute names do not have to match (but their domains have to be compatible)

Equi-Join:

- A special form of Theta-join, and the most common form of JOIN ...
- with a join condition containing an equality operator (i.e., explicitly stating the joining
- attributes)

$$Student \bowtie_{STUDENT.stuno=ENROL.stuno} Enrol$$

Theta-Join and Equi-Join

STUDENT

studno	name	hons	tutor	year
s1	jones	ca	bush	2
s2	brown	cis	kahn	2
s3	smith	cs	goble	2
s4	bloggs	ca	goble	1
s5	jones	cs	zobel	1
s6	peters	ca	kahn	3

STAFF

lecturer	roomno
kahn	IT206
bush	2.26
goble	2.82
zobel	2.34
watson	IT212
woods	IT204
capon	A14
lindsey	2.10
barringer	2.125

$\text{Student} \bowtie_{\text{tutor=lecturer}} \text{Staff}$

stud no	name	hons	tutor	year	lecturer	roomno
s1	jones	ca	bush	2	bush	2.26
s2	brown	cis	kahn	2	kahn	IT206
s3	smith	cs	goble	2	goble	2.82
s4	bloggs	ca	goble	1	goble	2.82
s5	jones	cs	zobel	1	zobel	2.34
s6	peters	ca	kahn	3	kahn	IT206

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

COURSE

Depart	Degree
EE	PhD
CS	PhD
EE	MSc
CS	MSc

ENROLMENT:

Enrol#	Supervisee	Supervisor	Depart	Degree
1	1	2	EE	PhD
2	3	1	CS	PhD
3	4	1	CS	MSc
4	5	1	CS	MSc

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

COURSE

Depart	Degree
EE	PhD
CS	PhD
EE	MSc
CS	MSc

ENROLMENT:

Enrol#	Supervisee	Supervisor	Depart	Degree
1	1	2	EE	PhD
2	3	1	CS	PhD
3	4	1	CS	MSc
4	5	1	CS	MSc

The name of supervisor who supervises student with ID 3

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

COURSE

Depart	Degree
EE	PhD
CS	PhD
EE	MSc
CS	MSc

ENROLMENT:

Enrol#	Supervisee	Supervisor	Depart	Degree
1	1	2	EE	PhD
2	3	1	CS	PhD
3	4	1	CS	MSc
4	5	1	CS	MSc

The name of supervisor who supervises student with ID 3

$$\pi_{\{name\}}(\sigma_{supervisee=3}ENROLMENT \bowtie_{supervisor=person\#} RESEARCHER)$$

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

COURSE

Depart	Degree
EE	PhD
CS	PhD
EE	MSc
CS	MSc

ENROLMENT:

Enrol#	Supervisee	Supervisor	Depart	Degree
1	1	2	EE	PhD
2	3	1	CS	PhD
3	4	1	CS	MSc
4	5	1	CS	MSc

What are the names of students who are studying MSc in computer science

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

COURSE

Depart	Degree
EE	PhD
CS	PhD
EE	MSc
CS	MSc

ENROLMENT:

Enrol#	Supervisee	Supervisor	Depart	Degree
1	1	2	EE	PhD
2	3	1	CS	PhD
3	4	1	CS	MSc
4	5	1	CS	MSc

What are the names of students who are studying MSc in computer science

$$\pi_{\{name\}}(\sigma_{(degree=MSc\ and\ Depart=CS)}\ ENROLMENT \bowtie_{supervisee=person\#} Student)$$

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

COURSE

Depart	Degree
EE	PhD
CS	PhD
EE	MSc
CS	MSc

ENROLMENT:

Enrol#	Supervisee	Supervisor	Depart	Degree
1	1	2	EE	PhD
2	3	1	CS	PhD
3	4	1	CS	MSc
4	5	1	CS	MSc

The IDs of students who are supervised by Dr C.C.Chen

Pracs

STUDENT:

Person#	Name
1	Dr C.C.Chen
3	Ms K.Juliff
4	Ms J.Gledill
5	Ms B.K.Lee

RESEARCHER:

Person#	Name
1	Dr C.C.Chen
2	Dr R.G.Wilkinson

COURSE

Depart	Degree
EE	PhD
CS	PhD
EE	MSc
CS	MSc

ENROLMENT:

Enrol#	Supervisee	Supervisor	Depart	Degree
1	1	2	EE	PhD
2	3	1	CS	PhD
3	4	1	CS	MSc
4	5	1	CS	MSc

The names of supervisor who supervises both MSc and PhD students

Divide

The DIVISION operation is applied to two Relations R and S, where the attributes of S are a subset of the attributes of R.

The relation returned by division operator will have attributes = All attributes of R – All Attributes of S

Return all tuples from relation R, which are associated to every S's tuple.

R

A	B
a_1	b_1
a_1	b_2
a_2	b_1
a_3	b_2
a_4	b_1
a_5	b_1
a_5	b_2

S

B
b_1
b_2

$R \div S =$

A
a_1
a_5

Divide

Typical use: The departments which offer all degrees?

$$Course \div (\pi_{Degree} Course)$$

COURSE

Depart	Degree
EE	PhD
CS	PhD
EE	MSc
CS	MSc

Exercise

R:

A	B	C
a_1	b_1	c_1
a_1	b_1	c_2
a_1	b_1	c_3
a_1	b_2	c_2
a_2	b_1	c_1
a_2	b_2	c_2
a_3	b_1	c_1
a_3	b_2	c_1
a_3	b_2	c_2

S:

B	C
b_1	c_1
b_1	c_2

Write relational algebra that retrieve:

1. Find A of R that contains all S.
2. Find (A, B) of R that contains all C of S.

Rename Operator

- The rename operator ρ changes the name of one or more attributes
- Change the names in a schema
- Does not affect instance of the target relation

Family

Father	Child
Adam	Abel
Adam	Cain
Abraham	Isaac

$\rho_{\text{(Parent, Child)}}\text{(Family)}$

Parent	Child
Adam	Abel
Adam	Cain
Abraham	Isaac

- Why might this be useful? To be included in relational algebra?

Why RENAME Operator?

- To unify schemas for set operators
- For disambiguation in “self-join”

Basic vs Extended Operators

- Note, $\{\sigma, \pi, \cup, -, \times\}$ and rename are sufficient to define all these operations: this is a relationally complete set of operators. These are the basic operators of the Relational Algebra.
- What about JOIN, INTERSECTION and DIVIDE? They are extended operators because they can be derived by the basic operators.
- e.g.,

$$\text{Student} \bowtie_{\text{tutor=lecturer}} \text{Staff} \qquad \sigma_{\text{tutor=lecturer}}(\text{Student} \times \text{Staff})$$

Aggregate Operators

- What if we want a relation with information about "sum of salaries" of employees, or the "average age" of students.
- We need more expressive power. We can use aggregation functions that to summarize information from multiple tuples into aggregate values.
- We can use an aggregation operator γ and a function such as SUM, AVG, MIN, MAX, or COUNT.

If $R =$

A	B
1	2
3	4
3	5
1	1

, then $\gamma_{SUM(A)}(R) =$

SUM(A)
8

and $\gamma_{AVG(B)}(R) =$

AVG(B)
3

Aggregate Operators

- We can also retrieve aggregate values for groups, like the "sum of employee salaries" per department, or the "average student age" per faculty.
- We give γ additional arguments to specify that the aggregation behavior should be based on groups (defined by a set of attributes).

If $R =$

a	b
1	2
3	4
3	5
1	3

, then $\gamma_{a,SUM(b)}(R) =$

a	SUM(b)
1	5
3	9