Relational Algebra and Relational Calculus

406.426 Design & Analysis of Database Systems

Jonghun Park

jonghun@snu.ac.kr

Dept. of Industrial Engineering

Seoul National University

outline

- unary relational operations
  - SELECT, PROJECT, RENAME, operation sequences
- relational algebra operations from set theory
  - UNION, INTERSECTION, MINUS, Cartesian product
- binary relational operations
  - JOIN, DIVISION, EQUIJOIN, NATURAL JOIN, JOIN variations
- additional relational operations
  - aggregate functions, grouping, recursive closure, outer JOIN, outer UNION

SELECT

- used to select a **subset of the tuples** from a relation that satisfy a **selection condition**
- can be visualized as a **horizontal partition** of the relation into two sets of tuples
- notation: $\sigma_{<\text{selection condition}>}(R)$
  - the selection condition is a Boolean expression **specified on the attributes** of relation *R*
  - the Boolean expression is made up of a number of clauses of the form
    - ***<attribute name> <comparison op> <constant value>***
    - ***<attribute name> <comparison op> <attribute name>***
  - the clauses can be connected by the Boolean operators
  - *R* is a **relational algebra expression** whose result is a relation
- the relation resulting from the SELECT operation has **the same attributes** as *R*
- **commutative**: $\sigma_{<\text{cond1}>}(\sigma_{<\text{cond2}>}(R)) = \sigma_{<\text{cond2}>}(\sigma_{<\text{cond1}>}(R))$
- **cascade**: $\sigma_{<\text{cond1}>}(\sigma_{<\text{cond2}>}(\ldots(\sigma_{<\text{cond}n>}R))\ldots)) = \sigma_{<\text{cond1}>\text{AND}<\text{cond2}>\text{AND}\ldots<\text{cond}n>}(R)$

example

- $\sigma_{(DNO=4 \text{ AND } SALARY>25000) \text{ OR } (DNO=5 \text{ AND } SALARY>30000)}(EMPLOYEE)$

FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
Franklin	T	Wong	333445555	1955-12-08	638 Voss,Houston,TX	M	40000	888665555	5
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry,Bellaire,TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 FireOak,Humble,TX	M	38000	333445555	5

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
	Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
	Alicia	J	Zelaya	999887777	1968-07-19	3321 Castle, Spring, TX	F	25000	987654321	4
	Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
	Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
	Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
	James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1

PROJECT

- selects **certain columns** from the table
- can be visualized as a **vertical partition**
- notation: $\pi_{\text{<attribute list>}}(R)$
  - <attribute list> is the desired list of attributes from the attributes of relation *R*
- the result has only the attributes specified in <attribute list> in the **same order** as they appear in the list
- **duplicate elimination**
  - when the attribute list includes only nonkey attributes of *R*
- the # of tuples in a relation resulting from a PROJECT operation is always less than or equal to the # of tuples in *R*
- **subsumption**: $\pi_{\text{<list1>}}(\pi_{\text{<list2>}}(R)) = \pi_{\text{<list1>}}(R)$ if <list2> contains <list1>

example

- $\pi_{\text{LNAME,FNAME,SALARY}}(\text{EMPLOYEE})$

LNAME	FNAME	SALARY
Smith	John	30000
Wong	Franklin	40000
Zelaya	Alicia	25000
Wallace	Jennifer	43000
Narayan	Ramesh	38000
English	Joyce	25000
Jabbar	Ahmad	25000
Borg	James	55000

$\pi_{\text{SEX,SALARY}}(\text{EMPLOYEE})$

duplicate eliminated

SEX	SALARY
M	30000
M	40000
F	25000
F	43000
M	38000
M	25000
M	55000

sequence of operations

- $\pi_{FNAME,LNAME,SALARY}(\sigma_{DNO=5}(EMPLOYEE))$

FNAME	LNAME	SALARY
John	Smith	30000
Franklin	Wong	40000
Ramesh	Narayan	38000
Joyce	English	25000

TEMP <- $\sigma_{DNO=5}(EMPLOYEE)$

R(FIRSTNAME, LASTNAME, SALARY) <-

$\pi_{FNAME,LNAME,SALARY}(TEMP)$

TEMP	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John	B	Smith	123456789	1965-01-09	731 Fondren,Houston,TX	M	30000	333445555	5
	Franklin	T	Wong	333445555	1955-12-08	638 Voss,Houston,TX	M	40000	888665555	5
	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak,Humble,TX	M	38000	333445555	5
	Joyce	A	English	453453453	1972-07-31	5631 Rice,Houston,TX	F	25000	333445555	5

R	FIRSTNAME	LASTNAME	SALARY
	John	Smith	30000
	Franklin	Wong	40000
	Ramesh	Narayan	38000
	Joyce	English	25000

RENAME

- renames either the **relation name** or the **attribute names**, or **both**
- $\rho_{S(B1, B2, \ldots, Bn)}(R)$
  - S is the new relation name
  - $B_1, \ldots, B_n$ are the new attribute names
- $\rho_S(R)$
  - renames the relation only
- $\rho_{(B1, B2, \ldots, Bn)}(R)$
  - renames the attributes only

UNION, INTERSECTION, SET MINUS

- **union compatibility**
  - 2 relations on which any of three operations are applied must have the **same type of tuples**
  - $R(A_1, A_2, ..., A_n)$ and $S(B_1, B_2, ..., B_n)$ are said to be union compatible if they have the same degree n and if **dom(A_i) = dom(B_i) for all *i***
- union: $R \cup S$
  - a relation that includes all tuples that are either in R or in S or in both R and S
  - duplicate tuples are eliminated
- intersection: $R \cap S$
  - a relation that includes all tuples that are in both R and S
- set minus: $R – S$
  - a relation that includes all tuples that are in R but not in S
- UNION and INTERSECTION are **commutative** and **associative**

examples

(a) 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

(b)

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

(c)

Fn	Ln
Susan	Yao
Ramesh	Shah

(d)

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

(e)

Fname	Lname
John	Smith
Ricardo	Browne
Francis	Johnson

Figure 6.4
The set operations UNION, INTERSECTION, and MINUS. (a) Two union-compatible relations.
(b) STUDENT ∪ INSTRUCTOR. (c) STUDENT ∩ INSTRUCTOR. (d) STUDENT – INSTRUCTOR.
(e) INSTRUCTOR – STUDENT.

Cartesian product

- aka cross product, **cross join**
- used to combine tuples from two relations in a combinatorial fashion
- notation: $R(A_1, A_2, ..., A_n) \times S(B_1, B_2, ..., B_m)$
  - relation Q with degree $n + m$ attributes $Q(A_1, A_2, ..., A_n, B_1, B_2, ..., B_m)$
  - if $|R| = n_R$ and $|S| = n_S$, then $|R \times S| = n_R * n_S$

example

- to retrieve a list of names of each female employee’s dependents

FEMALE_EMPS <- $\sigma_{SEX='F'}$(EMPLOYEE)

FEMALE_ EMPS	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	Alicia	J	Zelaya	999887777	1968-07-19	3321 Castle,Spring,TX	F	25000	987654321	4
	Jennifer	S	Wallace	987654321	1941-06-20	291 Berry,Bellaire,TX	F	43000	888665555	4
	Joyce	A	English	453453453	1972-07-31	5631 Rice,Houston,TX	F	25000	333445555	5

EMPNAMES <- $\pi_{FNAME,\ LNAME,\ SSN}$(FEMALE_EMPS)

EMPNAMES	FNAME	LNAME	SSN
	Alicia	Zelaya	999887777
	Jennifer	Wallace	987654321
	Joyce	English	453453453

example (cont.)

EMP_DEPENDENTS <- EMPNAMES × DEPENDENT

EMP_DEPENDENTS	FNAME	LNAME	SSN	ESSN	DEPENDENT_NAME	SEX	BDATE	• • •
	Alicia	Zelaya	999887777	333445555	Alice	F	1986-04-05	• • •
	Alicia	Zelaya	999887777	333445555	Theodore	M	1983-10-25	• • •
	Alicia	Zelaya	999887777	333445555	Joy	F	1958-05-03	• • •
	Alicia	Zelaya	999887777	987654321	Abner	M	1942-02-28	• • •
	Alicia	Zelaya	999887777	123456789	Michael	M	1988-01-04	• • •
	Alicia	Zelaya	999887777	123456789	Alice	F	1988-12-30	• • •
	Alicia	Zelaya	999887777	123456789	Elizabeth	F	1967-05-05	• • •
	Jennifer	Wallace	987654321	333445555	Alice	F	1986-04-05	• • •
	Jennifer	Wallace	987654321	333445555	Theodore	M	1983-10-25	• • •
	Jennifer	Wallace	987654321	333445555	Joy	F	1958-05-03	• • •
	Jennifer	Wallace	987654321	987654321	Abner	M	1942-02-28	• • •
	Jennifer	Wallace	987654321	123456789	Michael	M	1988-01-04	• • •
	Jennifer	Wallace	987654321	123456789	Alice	F	1988-12-30	• • •
	Jennifer	Wallace	987654321	123456789	Elizabeth	F	1967-05-05	• • •
	Joyce	English	453453453	333445555	Alice	F	1986-04-05	• • •
	Joyce	English	453453453	333445555	Theodore	M	1983-10-25	• • •
	Joyce	English	453453453	333445555	Joy	F	1958-05-03	• • •
	Joyce	English	453453453	987654321	Abner	M	1942-02-28	• • •
	Joyce	English	453453453	123456789	Michael	M	1988-01-04	• • •
	Joyce	English	453453453	123456789	Alice	F	1988-12-30	• • •
	Joyce	English	453453453	123456789	Elizabeth	F	1967-05-05	• • •

example (cont.)

ACTUAL_DEPENDENTS <- $\sigma_{SSN=ESSN}$(EMP_DEPENDENTS)

ACTUAL_DEPENDENTS	FNAME	LNAME	SSN	ESSN	DEPENDENT_NAME	SEX	BDATE	• • •
	Jennifer	Wallace	987654321	987654321	Abner	M	1942-02-28	• • •

RESULT <- $\pi_{FNAME, LNAME, DEPENDENT\_NAME}$(ACTUAL_DEPENDENTS)

RESULT	FNAME	LNAME	DEPENDENT_NAME
	Jennifer	Wallace	Abner

JOIN

- to combine **related tuples** from two relations into single tuples
- notation: $R(A_1, A_2, ..., A_n) \bowtie_{\text{<join condition>}} S(B_1, B_2, ..., B_m)$
  - relation Q with $n + m$ attributes $Q(A_1, A_2, ..., A_n, B_1, B_2, ..., B_m)$
  - Q has one tuple for each combination of tuples (consists of one from R and one from S) **whenever the combination satisfies the join condition**
  - i.e., CARTESIAN PRODUCT followed by SELECT
  - the join condition
    - <condition> AND <condition> AND ... AND <condition>
    - each condition is of the form $\boldsymbol{A_i \ \theta \ B_j}$, where A_i is an attribute of R, B_j is an attribute of S, A_i and B_j have the same domain, θ is one of the comparison operators
- if $|R| = n_R$ and $|S| = n_S$, then $0 \leq |R \bowtie_{\text{<join condition>}} S| \leq n_R * n_S$
- if there is no join condition, JOIN degenerates into a CARTESIAN PRODUCT -> called CROSS JOIN

Example

- DEPT_MGR <- DEPARTMENT $\bowtie_{MGRSSN=SSN}$ EMPLOYEE

DEPT_MGR	DNAME	DNUMBER	MGRSSN	• • •	FNAME	MINIT	LNAME	SSN	• • •
	Research	5	333445555	• • •	Franklin	T	Wong	333445555	• • •
	Administration	4	987654321	• • •	Jennifer	S	Wallace	987654321	• • •
	Headquarters	1	888665555	• • •	James	E	Borg	888665555	• • •

EQUIJOIN & NATURAL JOIN

- EQUIJOIN
  - the only comparison operator used is **=**
  - we always have one or more **pairs of attributes** that have **identical values** in every tuple
- NATURAL JOIN (*)
  - created to **get rid of the superfluous attribute** in an EQUIJOIN condition
  - requires that the two join attributes (or each pair of join attributes) have the **same name** in both relations

example

- DEPT <- $\rho_{(\text{DNAME},\mathbf{DNUM},\text{MGRSSN},\text{MGRSTARTDATE})}$(DEPARTMENT)
- PROJ_DEPT <- PROJECT * DEPT

PROJ_DEPT	PNAME	PNUMBER	PLOCATION	DNUM	DNAME	MGRSSN	MGRSTARTDATE
	ProductX	1	Bellaire	5	Research	333445555	1988-05-22
	ProductY	2	Sugarland	5	Research	333445555	1988-05-22
	ProductZ	3	Houston	5	Research	333445555	1988-05-22
	Computerization	10	Stafford	4	Administration	987654321	1995-01-01
	Reorganization	20	Houston	1	Headquarters	888665555	1981-06-19
	Newbenefits	30	Stafford	4	Administration	987654321	1995-01-01

PROJECT	PNAME	PNUMBER	PLOCATION	DNUM
	ProductX	1	Bellaire	5
	ProductY	2	Sugarland	5
	ProductZ	3	Houston	5
	Computerization	10	Stafford	4
	Reorganization	20	Houston	1
	Newbenefits	30	Stafford	4

DEPARTMENT	DNAME	DNUMBER	MGRSSN	MGRSTARTDATE
	Research	5	333445555	1988-05-22
	Administration	4	987654321	1995-01-01
	Headquarters	1	888665555	1981-06-19

complete set of relational algebra operations

- the set of relational algebra operations {σ, π, ∪, -, x} is a **complete** set -> any of the other original relational algebra operations can be expressed as a sequence of operations from this set
- example
  - INTERSECTION
    - $R \cap S = (R \cup S) - ((R - S) \cup (S - R))$
  - JOIN
    - $R \bowtie_{<condition>} S = \sigma_{<condition>}(R \times S)$

DIVISION

- notation: $R(Z) \div S(X)$
  - $Z \supseteq X$
  - let Y be the set of attributes of R that are not attributes of S (i.e., $\boldsymbol{Y = Z - X}$)
  - the result is a relation $T(Y)$ that includes a tuple t if tuples t_R appear in $\boldsymbol{R}$ with $\boldsymbol{t_R[Y] = t}$ and with $\boldsymbol{t_R[X] = t_S}$ for **every tuple** $\boldsymbol{t_S}$ in S
  - for a tuple t to appear in the result T, the values in t must appear in R in combination with every tuple in S
- the reverse of cross join
- example
  - $Z = \{A, B\}$, $X = \{A\}$, $Y = \{B\}$

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

Example

- SSNS(SSN) <- SSN_PNOS ÷ SMITH_PNOS

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

SSNs of employees who work on all the projects that Smith works on!

aggregate functions and grouping

- to specify mathematical aggregate functions on collections of values from the DB
  - e.g., SUM, AVERAGE
- notation: $_{\text{<grouping attributes>}} \mathfrak{I}_{\text{<function list>}} (R)$
  - <grouping attributes> is a list of attributes of the relation specified in R
  - <function list> is a list of (<function> <attribute>) pairs
  - in each pair, <function> is one of the allowed functions, such as SUM, AVERAGE, MAXIMUM, MINIMUM, COUNT, and <attribute> is an attribute of R
  - the resulting relation has the grouping attributes plus one attribute for each element in the <function list>
- if no grouping attributes are specified, the functions are applied to all the tuples in the relation

example

Figure 6.10
The aggregate function operation.

(a) $\rho_{R(\text{Dno, No\_of\_employees, Average\_sal})}(_{\text{Dno}}\mathfrak{I}_{\text{COUNT Ssn, AVERAGE Salary}}(\text{EMPLOYEE}))$.

(b) $_{\text{Dno}}\mathfrak{I}_{\text{COUNT Ssn, AVERAGE Salary}}(\text{EMPLOYEE})$.

(c) $\mathfrak{I}_{\text{COUNT Ssn, AVERAGE Salary}}(\text{EMPLOYEE})$.

(a) **R**

Dno	No_of_employees	Average_sal
5	4	33250
4	3	31000
1	1	55000

(b)

Dno	Count_ssn	Average_salary
5	4	33250
4	3	31000
1	1	55000

(c)

Count_ssn	Average_salary
8	35125

recursive closure

- applied to a recursive relationship between tuples of the same type
- example: retrieval of all supervisees of ‘James Borg’ up to two levels
  - $BORG\_SSN \leftarrow \pi_{SSN}(\sigma_{FNAME=\text{‘JAMES’ AND } LNAME=\text{‘BORG’}}(EMPLOYEE))$
  - $SUPERVISION(SSN1, SSN2) \leftarrow \pi_{SSN,\ SUPERSSN}(EMPLOYEE)$
  - $RESULT1(SSN) \leftarrow \pi_{SSN1}(SUPERVISION \bowtie_{SSN2=SSN} BORG\_SSN)$
  - $RESULT2(SSN) \leftarrow \pi_{SSN1}(SUPERVISION \bowtie_{SSN2=SSN} RESULT1)$
  - $RESULT \leftarrow RESULT2 \cup RESULT1$
- in general, the implementation of recursive closure using the basic relational algebra **requires a looping** mechanism

(Borg’s SSN is 888665555)

SUPERVISION	SSN1 (SSN)	SSN2 (SUPERSSN)
	123456789	333445555
	333445555	888665555
	999887777	987654321
	987654321	888665555
	666884444	333445555
	453453453	333445555
	987987987	987654321

RESULT 1	SSN
	333445555
	987654321

(Supervised by Borg)

RESULT 2	SSN
	123456789
	999887777
	666884444
	453453453
	987987987

(Supervised by Borg’s subordinates)

RESULT	SSN
	123456789
	999887777
	666884444
	453453453
	987987987
	333445555
	987654321

($RESULT1 \cup RESULT2$)

OUTER JOIN

- used when we want to keep all the tuples in *R*, or all those in *S*, or all those in both relations in the result of the JOIN, regardless of **whether or not they have matching tuples** in the other relation
- LEFT OUTER JOIN: ⟕
  - keeps every tuple in the left, in R⟕S
- RIGHT OUTER JOIN: ⟖
  - keeps every tuple in the right in R⟖S
- FULL OUTER JOIN: ⟗
  - keeps all tuples in both the left and right relations when no matching tuples are found

example

- a list of all **employee names** and also the **name of the departments they manage** if they happen to manage a department; if they do not manage any, an indication is made with a null value
- TEMP <- (EMPLOYEE $\bowtie_{SSN=MGRSSN}$ DEPARTMENT)
- RESULT <- $\pi_{FNAME,\ MINIT,\ LNAME,\ DNAME}$ (TEMP)

RESULT	FNAME	MINIT	LNAME	DNAME
	John	B	Smith	null
	Franklin	T	Wong	Research
	Alicia	J	Zelaya	null
	Jennifer	S	Wallace	Administration
	Ramesh	K	Narayan	null
	Joyce	A	English	null
	Ahmad	V	Jabbar	null
	James	E	Borg	Headquarters

OUTER UNION

- to take the union of tuples from two relations if the relations are not union compatible but **partially compatible**, meaning that only some of their attributes, *X*, are union compatible
- two tuples t_1 in *R* and t_2 in *S* are said to **match** if $t_1[X] = t_2[X]$, and are considered to represent the same relationship instance -> unioned into a single tuple in the result, *T*
- tuples in either relation that have no matching tuple in the other relation are **padded with null values**
- example
  - STUDENT(Name, SSN, Department, Advisor) and INSTRUCTOR(Name, SSN, Department, Rank)
  - the result: STUDENT_OR_INSTRUCTOR(Name, SSN, Department, Advisor, Rank)