Chapter 8

The Relational Algebra (from E&N,Silberschatz and my editing)

Chapter Outline

- Relational Algebra
  - Unary Relational Operations
  - Relational Algebra Operations From Set Theory
  - Binary Relational Operations
  - Additional Relational Operations
  - Examples of Queries in Relational Algebra

- Relational Algebra is a Procedural Paradigm
  You Need to Tell What/How to Construct the Result
- Consists of a Set of Operators Which, When Applied to Relations, Yield Relations (Closed Algebra)

- Basic Relational Operations:
  - **Unary Operations**
    - **SELECT σ**
    - **PROJECT π or Π.**
  - **Binary Operations**
    - **Set operations:**
      - **UNION $\cup$**
      - **INTERSECTION $\cap$**
      - **DIFFERENCE $-$**
    - **CARTESIAN PRODUCT $\times$**
    - **JOIN operations**

$R \cup S$	union
$R \cap S$	intersection
$R - S$	set difference
$R \times S$	Cartesian product
$\pi_{A1, A2, ..., An}(R)$	projection
$\sigma_F(R)$	selection
$R \bowtie S$	natural join
$R\ \Theta\ S$	theta-join
$R \div S$	division
ρ [A1 B1,.., An Bn]	rename

Relational Algebra

- The basic set of operations for the relational model is known as the relational algebra. These operations enable a user to specify basic retrieval requests.

- The result of a retrieval is a new relation, which may have been formed from one or more relations. The **algebra operations** thus produce new relations, which can be further manipulated using operations of the same algebra.

- A sequence of relational algebra operations forms a **relational algebra expression**, whose result will also be a relation that represents the result of a database query (or retrieval request).

Selection

- Selects the Set of Tuples (Rows) From a Relation, Which Satisfy a Selection Condition
- General Form $\sigma_{<selection\ condition>}$ (R)
  - R is a Relation
  - Selection condition is a Boolean Expression on the Attributes of R
  - Resulting Relation Has the Same Schema as R
- Select Finds and Retrieves All Relevant Rows (Tuples) of Table/Relation R which Includes ALL of its Columns (Attributes)

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng.
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

σ $_{\text{TITLE='Elect. Eng.'}}$**(EMP)**

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng
E6	L. Chu	Elect. Eng.

σ $_{\text{TITLE='Elect. Eng.' OR TITLE='Mech.Eng'}}$**(EMP)**

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

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

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		Wallace	987654321	1941-06-20	291 Berry,Bellaire,TX	F	43000	888665555	4
Ramesh		Narayan	666884444	1962-09-15	975 FireOak,Humble,TX	M	38000	333445555	5

Select Condition

- A SELECT Condition is a Boolean Expression
  - Form F_1 Ψ F_2 Ψ ..., Ψ F_q (Q>=1), Where
  - F_i (I=1,…,q) are Atomic Boolean Expressions of the Form
  - aθc or aθb,
  - a, b are Attributes of R and c is a Constant.
  - The Operator θ is one of the Arithmetic Comparison Operators: <, >, =, <>, >=, <=
  - The Operator Ψ is one of the Logical Operators: ∧, ∨, ¬
- Nesting: ( )

Projection

- Select certain Columns (Attributes) Specified in an *Attribute List X* From a Relation R
- General Form $\pi_{<attribute\text{-}list>}(R)$
  - *R* is a Relation
  - *Attribute-list* is a Subset of the Attributes of *R* Over Which the Projection is Performed
- Project Retrieves Specified Columns of Table/Relation R which Includes ALL of its Rows (Tuples)

PROJ

PNO	PNAME	BUDGET
P1	Instrumentation	150000
P2	Database Develop.	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

$\pi_{PNO,BUDGET}(PROJ)$

PNO	BUDGET
P1	150000
P2	135000
P3	250000
P4	310000
P5	500000

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

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

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

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

Rel Algebra Expression

- Several Operations can be Combined to form a *Relational Algebra Expression* (query)
- Example: *Retrieve all Employee over age 60?*
  - Method 1:

    $\pi_{\text{CNAME, ADDRESS, AGE}}(\sigma_{\text{AGE>60}}(\text{CUSTOMER}))$
  - Method 2:

    Senior-CUST(C#, Addr, Age) $\leftarrow \pi_{\text{CNAME, ADDRESS, AGE}}(\sigma_{\text{AGE>60}}(\text{CUSTOMER}))$
  - Method 3:

    $\pi_{\text{CNAME, ADDRESS, AGE}}(C)$ where $C = \sigma_{\text{AGE>60}}(\text{CUSTOMER})$

Characteristics of Projection

- The PROJECT Operation Eliminates
- Duplicate Tuples in the Resulting Relation
  - Projection Must Maintain a Mathematical Set (No Duplicate Elements)

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng.
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

π_{TITLE}(PROJ)

TITLE
Elect.Eng
Syst.Anal
Mech.Eng
Programmer

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

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

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

Rename

- The RENAME operator gives a new schema to a relation.
- R1 := $\text{RENAME}_{R1(A1,\ldots,An)}$(R2) makes R1 be a relation with attributes A1,…,A*n* and the same tuples as R2.
- Simplified notation: R1(A1,…,A*n*) := R2.
- Other use (←) notation
  - R2 ← R1(A1,…,A*n*)

Sequence …

- Create temporary relation names.
- Renaming can be implied by giving relations a list of attributes.
- Example: R3 := R1 $JOIN_C$ R2 can be written:
  - R4 := R1 * R2
  - R3 := $SELECT_C$(R4)

  OR
  - R4 ← R1 * R2
  - R3 ← $SELECT_C$(R4)

Figure 7.9 Results of relational algebra expressions.
(a) $\pi_{LNAME,\ FNAME,\ SALARY}(\sigma_{DNO=5}(EMPLOYEE))$. (b) The same expression using intermediate relations and renaming of attributes.

(a)

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

(b)

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

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

Relational Algebra Operations From Set Theory

- **UNION Operation**

Example: To retrieve the social security numbers of all employees who either work in department 5 or directly supervise an employee who works in department 5, we can use the union operation as follows:

DEP5_EMPS ← $\sigma_{DNO=5}$ (EMPLOYEE)

RESULT1 ← π_{SSN}(DEP5_EMPS)

RESULT2(SSN) ← $\pi_{SUPERSSN}$(DEP5_EMPS)

RESULT ← RESULT1 ∪ RESULT2

The union operation produces the tuples that are in either RESULT1 or RESULT2 or both. The two operands must be "type compatible".

Union (U)

- Notation: $r \cup s$
- Defined as:
  - $r \cup s = \{t \mid t \in r \text{ or } t \in s\}$
- For $r \cup s$ to be valid.
  - 1. *r, s* must have the *same arity* (same number of attributes)
  - 2. The attribute domains must be *compatible* (e.g., 2nd column of *r* deals with the same type of values as does the 2nd column of *s*)

Example:

"find students registered for course C1 or C3"

$$\pi_{s\#}(\sigma_{CNO='C1'}(S\text{-}C)) \cup \pi_{s\#}(\sigma_{CNO='C3'}(S\text{-}C))$$

Union Compatibility

- Two Relations $R_1(A_1, A_2, ..., A_n)$ and $R_2(B_1, B_2, ..., B_n)$ are said Union-compatible If and Only If They Have
  - The Same Number of Attributes
  - The Domains of Corresponding Attributes are Compatible, i.e., $Dom(A_i)=dom(B_i)$ for i=1, 2, ..., N
  - Names Do Not Have to be Same!
- For Relational Union and Difference Operations, the Operand Relations Must Be *Union Compatible*
- The Resulting Relation for Relational Set Operations
  - Has the Same Attribute Names as the *First* Operand Relation R_1 (by Convention)

- Relations *r, s:*

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

$r \cup s$:

A	B
α	1
α	2
β	1
β	3

Set Difference (-)

- Notation $r - s$
  - Defined as:
  - $r - s = \{t \mid t \in r$ **and** $t \notin s\}$
- Set differences must be taken between *compatible* relations.
  - *r* and *s* must have the *same arity*
  - attribute domains of *r* and *s* must be compatible

Example

"Find the students who registered course C1 but not C3"

$$\pi_{S\#}(\sigma_{CNO='C1'}(S\text{-}C)) - \pi_{S\#}(\sigma_{CNO='C3'}(S\text{-}C))$$

- Relations *r, s:*

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

$r - s$:

A	B
α	1
β	1

Intersection (∩)

- Notation: $r \cap s$
  - Defined as:
  - $r \cap s = \{ t \mid t \in r \textbf{ and } t \in s \}$
- Assume:
  - *r*, *s* have the *same arity*
  - attributes of r and s are compatible
- Note: $r \cap s = r - (r - s)$

- Relation r, s:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- $r \cap s$

A	B
α	2

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

FN	LN
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

FN	LN
Susan	Yao
Ramesh	Shah

FNAME	LNAME
John	Smith
Ricardo	Browne
Francis	Johnson

Cartesian (x)

- Notation $r \times s$
  - Defined as:
  - $r \times s = \{t\,q \mid t \in r \textbf{ and } q \in s\}$
- Assume that attributes of r(R) and s(S) are disjoint. (That is, $R \cap S = \emptyset$).
- If attributes of *r(R)* and *s(S*) are not disjoint, then renaming must be used.

Relations *r, s*:

A	*B*
α	1
β	2

r

C	*D*	*E*
α	10	a
β	10	a
β	20	b
γ	10	b

s

r x *s*:

A	*B*	*C*	*D*	*E*
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

R

A	B	C
a1	b1	c3
a2	b1	c5
a3	b4	c7

S

E	F
e1	f1
e2	f5

R × S

A	B	C	E	F
a1	b1	c3	e1	f1
a1	b1	c3	e2	f5
a2	b1	c5	e1	f1
a2	b1	c5	e2	f5
a3	b4	c7	e1	f1
a3	b4	c7	e2	f5

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

SAL

TITLE	SAL
Elect. Eng.	40000
Syst. Anal.	34000
Mech. Eng.	27000
Programmer	24000

EMP × SAL

ENO	ENAME	EMP.TITLE	SAL.TITLE	SAL
E1	J. Doe	Elect. Eng.	Elect. Eng.	40000
E1	J. Doe	Elect. Eng.	Syst. Anal.	34000
E1	J. Doe	Elect. Eng.	Mech. Eng.	27000
E1	J. Doe	Elect. Eng.	Programmer	24000
E2	M. Smith	Syst. Anal.	Elect. Eng.	40000
E2	M. Smith	Syst. Anal.	Syst. Anal.	34000
E2	M. Smith	Syst. Anal.	Mech. Eng.	27000
E2	M. Smith	Syst. Anal.	Programmer	24000
E3	A. Lee	Mech. Eng.	Elect. Eng.	40000
E3	A. Lee	Mech. Eng.	Syst. Anal.	34000
E3	A. Lee	Mech. Eng.	Mech. Eng.	27000
E3	A. Lee	Mech. Eng.	Programmer	24000
E8	J. Jones	Syst. Anal.	Elect. Eng.	40000
E8	J. Jones	Syst. Anal.	Syst. Anal.	34000
E8	J. Jones	Syst. Anal.	Mech. Eng.	27000
E8	J. Jones	Syst. Anal.	Programmer	24000

The Join Operation

- Used to combine related tuples from two relation into single tuples with some conditions
- Cartesian product all combination of tuples are included in the result, meanwhile in the join operation, only combination of tuples satisfying the join condition appear in the result

Theta Join (Θ)

- General Form
  - $R \Theta S$
  - Where *R*, *S* are Relations,
  - θ is a Boolean Expression, called a Join Condition.
- A Derivative of Cartesian Product
  - $R \Theta S = \sigma_\theta (R \times S)$
- $R(A_1, A_2, ..., A_m, B_1, B_2, ..., B_n)$ is the Resulting Schema of a Θ-Join over R_1 and R_2: $R_1(A_1, A_2, ..., A_m) \Theta R_2 (B_1, B_2, ..., B_n)$

θ-Join Condition

- A θ-Join Condition is a Boolean Expression of the form F_1 ψ_1 F_2 ψ_2 ..., ψ_{n-1} F_q (q>=1), where
  - F_i (i=1,…,q) are Atomic Boolean Expressions of the form
    $A_i \theta B_j$,
  - A_i, B_j are Attributes of R_1 and R_2 Respectively
  - θ is one of the Algorithmic Comparison Operators
    =, <>, >, <. >=, <=
  - The Operator ψ_i (*i=1,…,n-1*) is Either a Logical AND operator $\wedge$ or a logical OR operator $\vee$

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

SAL

TITLE	SAL
Elect. Eng.	40000
Syst. Anal.	34000
Mech. Eng.	27000
Programmer	24000

EMP ϴ $_{\text{E.TITLE=SAL.TITLE}}$ SAL

ENO	ENAME	TITLE	SAL.TITLE	SAL
E1	J. Doe	Elect. Eng.	Elect. Eng.	40000
E2	M. Smith	Analyst	Analyst	34000
E3	A. Lee	Mech. Eng.	Mech. Eng.	27000
E4	J. Miller	Programmer	Programmer	24000
E5	B. Casey	Syst. Anal.	Syst. Anal.	34000
E6	L. Chu	Elect. Eng.	Elect. Eng.	40000
E7	R. Davis	Mech. Eng.	Mech. Eng.	27000
E8	J. Jones	Syst. Anal.	Syst. Anal.	34000

Equijoin

- The θ Expression only Contains one or more *Equality Comparisons* Involving Attributes from R_1 and R_2
- The most common use of join involves join conditions with equality comparisons only. Such a join, where the only comparison operator used is =, is called an EQUIJOIN. In the result of an EQUIJOIN we always have one or more pairs of attributes (whose names need not be identical) that have *identical values* in every tuple.

Natural Join ($\bowtie$)

- Notation: r $\bowtie$ s
- Let *r* and *s* be relations on schemas *R* and *S* respectively.
  Then, r $\bowtie$ s is a relation on schema $R \cup S$ obtained as follows:
  - Consider each pair of tuples t_r from *r* and t_s from *s*.
  - If t_r and t_s have the same value on each of the attributes in $R \cap S$, add a tuple *t* to the result, where
  - *t* has the same value as t_r on *r*
  - *t* has the same value as t_s on *s*

- Relations r, s:

A	B	C	D
α	1	α	a
β	2	γ	a
γ	4	β	b
α	1	γ	a
δ	2	β	b

r

B	D	E
1	a	α
3	a	β
1	a	γ
2	b	δ
3	b	∈

s

A	B	C	D	E
α	1	α	a	α
α	1	α	a	γ
α	1	γ	a	α
α	1	γ	a	γ
δ	2	β	b	δ

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

SAL

TITLE	SAL
Elect. Eng.	70000
Syst. Anal.	80000
Mech. Eng.	56000
Programmer	60000

EMP $\bowtie$ SAL

ENO	ENAME	E.TITLE	SAL
E1	J. Doe	Elect. Eng.	70000
E2	M. Smith	Syst. Anal.	80000
E3	A. Lee	Mech. Eng.	56000
E4	J. Miller	Programmer	60000
E5	B.Casey	Syst.Anal	80000
E6	L. Chu	Elect.Eng	70000
E7	R.Davis	Mech.Eng	56000
E8	J. Jones	Syst. Anal.	80000

R

A	B	C
a1	b1	c3
a2	b1	c5
a3	b4	c7

S

B	E
b1	e1
b5	e2

EQUIJOIN

$R \bowtie_{R.B=S.B} S$

A	R.B	S.B	C	E
a1	b1	b1	c3	e1
a2	b1	b1	c5	e1

Natural Join

$R \bowtie S$

A	R.B	C	E
a1	b1	c3	e1
a2	b1	c5	e1

Division (÷)

- Suited to queries that include the phrase "for all".
- Let *r* and *s* be relations on schemas R and S respectively where
  - $R = (A_1, \ldots, A_m, B_1, \ldots, B_n)$
  - $S = (B_1, \ldots, B_n)$
- The result of r ÷ s is a relation on schema
  $R - S = (A_1, \ldots, A_m)$
- $r \div s = \{ t \mid t \in \Pi_{R-S}(r) \wedge \forall u \in s \ (tu \in r) \}$

Relations *r, s*:

A	B
α	1
α	2
α	3
β	1
γ	1
δ	1
δ	3
δ	4
$\in$	6
$\in$	1
β	2

r

B
1
2

s

$r \div s$:

A
α
β

Relations *r, s*:

A	*B*	*C*	*D*	*E*
α	a	α	a	*1*
α	a	γ	a	*1*
α	a	γ	b	*1*
β	a	γ	a	*1*
β	a	γ	b	*3*
γ	a	γ	a	*1*
γ	a	γ	b	*1*
γ	a	β	b	*1*

r

D	*E*
a	*1*
b	*1*

s

$r \div s$:

A	*B*	*C*
α	a	γ
γ	a	γ

R

ENO	PNO	PNAME	BUDGET
E1	P1	Instrumentation	150000
E2	P1	Instrumentation	150000
E2	P2	Database Develop.	135000
E3	P1	Instrumentation	150000
E3	P4	Maintenance	310000
E4	P2	Instrumentation	150000
E5	P2	Instrumentation	150000
E6	P4	Maintenance	310000
E7	P3	CAD/CAM	250000
E8	P3	CAD/CAM	250000

Find the employees who work for both project P1 and project P4?

S

PNO	PNAME	BUDGET
P1	Instrumentation	150000
P4	Maintenance	310000

R ÷ S

ENO
E3

- *"list the S# of students that have taken all of the courses listed in SC"*

$$\pi_{S\#}(SC \div \pi_{C\#}(SC))$$

- *"list the S# of students who have taken all of the courses taught by instructor Smith"*

$$\pi_{S\#}(SC \div \pi_{C\#}(\sigma_{Instructor='Smith'}(SC)))$$

- Selection
- Projection
- Union
- Difference
- Cartesian Product

Fundamental Operators

- Intersection
- Join, Equi-join, Natural Join
- Quotient (Division)

Derivable from the fundamental operators

Relational Algebra Operations From Set Theory

- Notice that both union and intersection are *commutative operations;* that is

  $$\mathbf{R \cup S = S \cup R, \text{ and } R \cap S = S \cap R}$$

- Both union and intersection can be treated as n-ary operations applicable to any number of relations as both are *associative operations;* that is

  $$\mathbf{R \cup (S \cup T) = (R \cup S) \cup T, \text{ and } (R \cap S) \cap T = R \cap (S \cap T)}$$

- The minus operation is *not commutative;* that is, in general

  $$\mathbf{R - S \neq S - R}$$

Complete Set of Relational Operations

- The set of operations including **select σ, project π , union ∪, set difference - , and cartesian product X** is called a complete set because any ⋈ other relational algebra expression can be expressed by a combination of these five operations.

- For example:

$$\mathbf{R \cap S = (R \cup S) - ((R - S) \cup (S - R))}$$

$$R \bowtie_{\text{<join condition>}} S = \boldsymbol{\sigma}_{\text{<join condition>}} \mathbf{(R \times S)}$$

All Rel Algebra Operation

- A Set of Relational Algebra Operations Is Called a Complete Set, If and Only If
  - Any Relational Algebra Operator in the Set Cannot be Derived in Terms of a Sequence of Others in Set
  - Any Relational Algebra Operator Not in the Set Can Be Derived in Terms of a Sequence of Only the Operators in the Set

Additional ...

- Important Concepts:
  - The Set of Algebra Operations {$\boldsymbol{\sigma}$, $\boldsymbol{\Pi}$, ∪, −, ×} is a *Complete Set* of Relational Algebra Operations
  - Any Query Language *Equivalent to* These Five Operations is Called **Relationally Complete**

Use of the Functional operator $\mathcal{F}$

$\mathcal{F}_{\text{MAX } Salary}$ (**Employee**) retrieves the maximum salary value from the Employee relation

$\mathcal{F}_{\text{MIN } Salary}$ (**Employee**) retrieves the minimum Salary value from the Employee relation

$\mathcal{F}_{\text{SUM } Salary}$ (**Employee**) retrieves the sum of the Salary from the Employee relation

${}_{\text{DNO}}\ \mathcal{F}_{COUNT\ SSN,\ AVERAGE\ Salary}$ (**Employee**) groups employees by DNO (department number) and computes the count of employees and average salary per department.[Note: count just counts the number of rows, without removing duplicates]

Additional Relational Operations (cont.)

(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

Additional Relational Operations (cont.)

- **Recursive Closure Operations**
  - Another type of operation that, in general, cannot be specified in the basic original relational algebra is **recursive closure.** This operation is applied to a **recursive relationship**.
  - An example of a recursive operation is to retrieve all SUPERVISEES of an EMPLOYEE e at all levels—that is, all EMPLOYEE e' directly supervised by e; all employees e'' directly supervised by each employee e'; all employees e''' directly supervised by each employee e''; and so on .
  - Although it is possible to retrieve employees at each level and then take their union, we cannot, in general, specify a query such as "retrieve the supervisees of 'James Borg' at all levels" without utilizing a looping mechanism.
  - The SQL3 standard includes syntax for recursive closure.

Additional Relational Operations (cont.)

(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 ∪ RESULT2)

Additional Relational Operations (cont.)

- **The OUTER JOIN Operation**

  – In NATURAL JOIN tuples without a *matching* (or *related*) tuple are eliminated from the join result. Tuples with null in the join attributes are also eliminated. This amounts to loss of information.

  – A set of operations, called outer joins, can be 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.

  – The left outer join operation keeps every tuple in the *first* or *left* relation R in R $⟕$ S; if no matching tuple is found in S, then the attributes of S in the join result are filled or "padded" with null values.

  – A similar operation, right outer join, keeps every tuple in the *second* or right relation S in the result of R $⟖$ S.

  – A third operation, full outer join, denoted by $⟗$ keeps all tuples in both the left and the right relations when no matching tuples are found, padding them with null values as needed.

Additional Relational Operations (cont.)

- **OUTER UNION Operations**
  - The outer union operation was developed to take the union of tuples from two relations if the relations are *not union compatible.*
  - This operation will take the union of tuples in two relations R(X, Y) and S(X, Z) that are **partially compatible**, meaning that only some of their attributes, say X, are union compatible.
  - The attributes that are union compatible are represented only once in the result, and those attributes that are not union compatible from either relation are also kept in the result relation T(X, Y, Z).
  - **Example:** An outer union can be applied to two relations whose schemas are STUDENT(Name, SSN, Department, Advisor) and INSTRUCTOR(Name, SSN, Department, Rank). Tuples from the two relations are matched based on having the same combination of values of the shared attributes—Name, SSN, Department. If a student is also an instructor, both Advisor and Rank will have a value; otherwise, one of these two attributes will be null.

    The result relation STUDENT_OR_INSTRUCTOR will have the following attributes:

    STUDENT_OR_INSTRUCTOR (Name, SSN, Department, Advisor, Rank)

Examples of Queries in Relational Algebra

- **Q1: Retrieve the name and address of all employees who work for the 'Research' department.**

RESEARCH_DEPT ← σ $_{DNAME='Research'}$ (DEPARTMENT)

RESEARCH_EMPS ← (RESEARCH_DEPT ⋈ $_{DNUMBER=DNO}$ EMPLOYEE)

RESULT ← π $_{FNAME, LNAME, ADDRESS}$ (RESEARCH_EMPS)

- **Q6: Retrieve the names of employees who have no dependents.**

ALL_EMPS ← π $_{SSN}$(EMPLOYEE)

EMPS_WITH_DEPS(SSN) ← π $_{ESSN}$(DEPENDENT)

EMPS_WITHOUT_DEPS ← (ALL_EMPS - EMPS_WITH_DEPS)

RESULT ← π $_{LNAME, FNAME}$ (EMPS_WITHOUT_DEPS * EMPLOYEE)