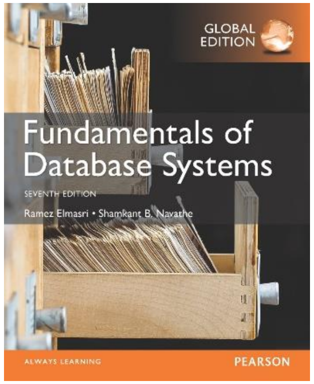
Database Systems

Department of Computer Science
Chungbuk National University

Seo-Young Noh

CHAPTER 8:
The Relational Algebra and The Relational Calculus



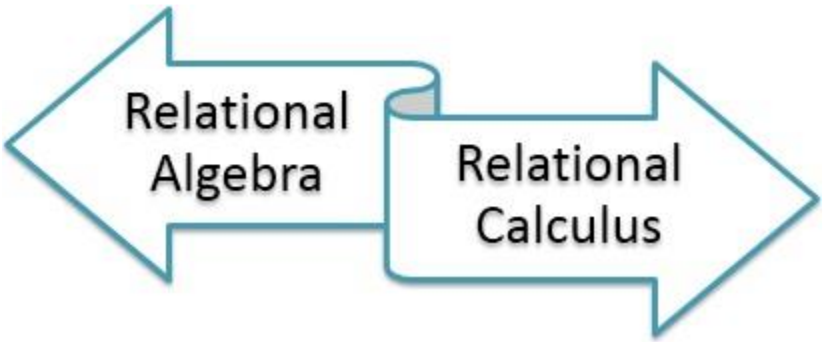
Chapter Outline

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

- **Relational Calculus**
  - Tuple Relational Calculus
  - Domain Relational Calculus

- **Example Database Application (COMPANY)**
- ~~**Overview of the QBE language (appendix D)**~~

Relational Algebra vs. Relational Calculus



- Relational Algebra and Relational Calculus are the formal query languages for a relational model.
- Both form the base for the SQL language which is used in most of the relational DBMSs.
- Relational Algebra is a procedural language.
- On the other hands, Relational Calculus is a declarative language.

https://techdifferences.com/difference-between-relational-algebra-and-relational-calculus.html

Relational Algebra vs. Relational Calculus

Relational Algebra

`{ Select (σ), Project (π), Union (∪ ), Set Difference (-), Cartesian product (×) and Rename (ρ)}`

Relational Calculus

Tuple Relational Calculus

list tuples `{ t | P(t) }`

Domain Relational Calculus

list attributes `{<X1, X2, X3, . . . Xn> | P(X1, X2, X3, . . . Xn)}`

https://techdifferences.com/difference-between-relational-algebra-and-relational-calculus.html

Relational Algebra vs. Relational Calculus

BASIS FOR COM PARISON	RELATIONAL ALGEBRA	RELATIONAL CALCULUS
Basic	Relational Algebra is a Procedural language.	Relational Calculus is Declarative language.
States	Relational Algebra states how to obtain the result.	Relational Calculus states what results we have to obtain.
Order	Relational Algebra describes the order in which operations have to be performed.	Relational Calculus does not specify the order of operations.
Domain	Relational Algebra is not domain dependent.	Relation Calculus can be domain dependent.
Related	It is close to a programming language.	It is close to the natural language.

https://techdifferences.com/difference-between-relational-algebra-and-relational-calculus.html

Unary Relational Operations: `SELECT` and `PROJECT`

Relational Algebra Overview

- **Relational algebra is <u>the basic set of operations</u> for the relational model**

- **These operations enable a user to <u>specify basic retrieval requests</u> (or queries)**

- **The result of an operation is a *new relation*, which may have been formed from one or more *input* relations**
  - <u>This property makes the algebra "closed" (all objects in relational algebra are relations)</u>

Relational Algebra Overview (continued)

- **The algebra operations thus produce new relations**
  - These can be further manipulated using operations of the same algebra

- **A sequence of relational algebra operations forms a relational algebra expression**
  - The result of a relational algebra expression is also a relation that represents the result of a database query (or retrieval request)

Brief History of Origins of Algebra

- **Muhammad ibn Musa al-Khwarizmi (800-847 CE) – from Morocco wrote a book titled al-jabr about arithmetic of variables**
  - Book was translated into Latin.
  - Its title (al-jabr) gave Algebra its name.

- **Al-Khwarizmi called variables "shay"**
  - "Shay" is Arabic for "thing".
  - Spanish transliterated "shay" as "xay" ("x" was "sh" in Spain).
  - In time this word was abbreviated as x.

- **Where does the word Algorithm come from?**
  - Algorithm originates from "al-Khwarizmi"
  - Reference: PBS (http://www.pbs.org/empires/islam/innoalgebra.html)

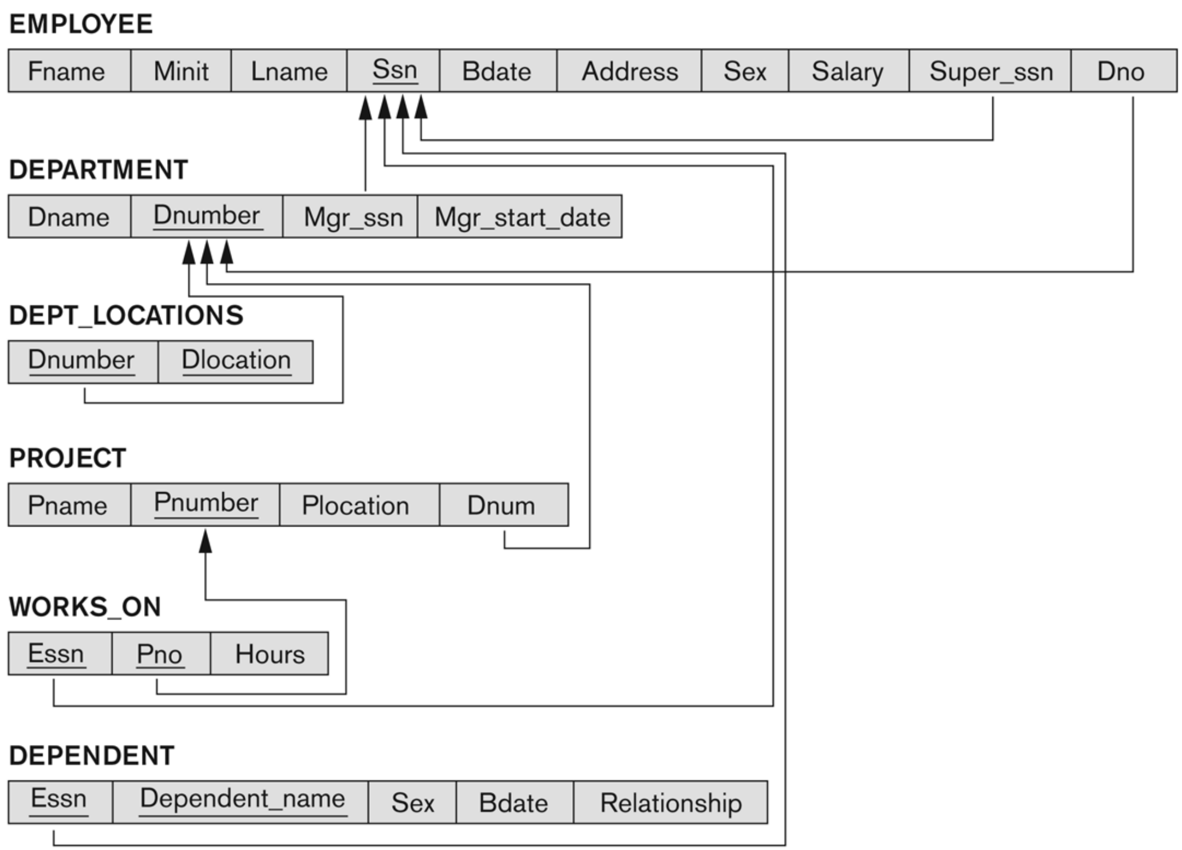
Relational Algebra Overview

- **Relational Algebra consists of several groups of operations**
  - Unary Relational Operations
    - SELECT (symbol: σ (sigma))
    - PROJECT (symbol: π (pi))
    - RENAME (symbol: ρ (rho))
  - Relational Algebra Operations from Set Theory
    - UNION ($\cup$)
    - INTERSECTION ($\cap$)
    - DIFFERENCE (or MINUS, **–**)
  - Binary Relational Operations
    - CARTESIAN PRODUCT (**x**)
    - JOIN (several variations of JOIN exist)
    - DIVISION
  - Additional Relational Operations
    - OUTER JOINS, OUTER UNION
    - AGGREGATE FUNCTIONS (These compute summary of information: for example, SUM, COUNT, AVG, MIN, MAX)

Database State for COMPANY

- **All examples discussed below refer to the COMPANY database shown here.**

Figure 5.7
Referential integrity constraints displayed on the COMPANY relational database schema.



Unary Relational Operations: SELECT

- **The SELECT operation (denoted by σ (sigma)) is used to select a *subset* of the tuples from a relation based on a selection condition.**
  - The selection condition acts as a **filter**
  - Keeps only those tuples that satisfy the qualifying condition
  - Tuples satisfying the condition are *selected* whereas the other tuples are discarded (*filtered out*)

- **Examples:**
  - Select the EMPLOYEE tuples whose department number is 4:

$$\sigma_{DNO = 4}\ (EMPLOYEE)$$

  - Select the employee tuples whose salary is greater than $30,000:

$$\sigma_{SALARY > 30{,}000}\ (EMPLOYEE)$$

Unary Relational Operations: SELECT

- **In general, the *select* operation is denoted by**

$$\sigma_{\text{<selection condition>}}(R)$$

where,

- the symbol σ (sigma) is used to denote the *select* operator
- the selection condition is a Boolean (conditional) expression specified on the attributes of relation R
- tuples that make the condition **true** are selected
  - appear in the result of the operation
- tuples that make the condition **false** are filtered out
  - discarded from the result of the operation

Unary Relational Operations: SELECT (continued)

- **SELECT Operation Properties**
  - The SELECT operation $\sigma_{\text{<selection condition>}}(R)$ produces a relation S that has the same schema (same attributes) as R
  - SELECT σ is commutative:
    - $\sigma_{\text{<condition1>}}(\sigma_{\text{< condition2>}}(R)) = \sigma_{\text{<condition2>}}(\sigma_{\text{<condition1>}}(R))$
  - Because of commutativity property, a cascade (sequence) of SELECT operations may be applied in any order:
    - $\sigma_{\text{<cond1>}}(\sigma_{\text{<cond2>}}(\sigma_{\text{<cond3>}}(R)) = \sigma_{\text{<cond2>}}(\sigma_{\text{<cond3>}}(\sigma_{\text{<cond1>}}(R)))$
  - A cascade of SELECT operations may be replaced by a single selection with a conjunction of all the conditions:
    - $\sigma_{\text{<cond1>}}(\sigma_{\text{< cond2>}}(\sigma_{\text{<cond3>}}(R)) = \sigma_{\text{<cond1> AND <cond2> AND <cond3>}}(R)))$
  - The number of tuples in the result of a SELECT is less than (or equal to) the number of tuples in the input relation R

The following query results refer to this database state

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	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	1968-01-19	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

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

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

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Unary Relational Operations: PROJECT

- **PROJECT Operation is denoted by π (pi)**

- **This operation extracts certain *columns* (attributes) from a relation and discards the other columns.**
  - PROJECT creates a vertical partitioning
    - The list of specified columns (attributes) is kept in each tuple
    - The other attributes in each tuple are discarded

- **Example: To list each employee's first and last name and salary, the following is used:**

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

Unary Relational Operations: PROJECT (cont.)

- **The general form of the *project* operation is:**

$$\pi_{\text{<attribute list>}}(R)$$

  - π (pi) is the symbol used to represent the *project* operation
  - <attribute list> is the desired list of attributes from relation R.

- **The project operation *removes any duplicate tuples***
  - This is because the result of the *project* operation must be a *set of tuples*
    - Mathematical sets *do not allow* duplicate elements.

Unary Relational Operations: PROJECT (contd.)

- **PROJECT Operation Properties**
  - The number of tuples in the result of projection $\pi_{\text{<list>}}(R)$ is always less or equal to the number of tuples in R
    - If the list of attributes includes a *key* of R, then the number of tuples in the result of PROJECT is *equal* to the number of tuples in R
  - PROJECT is *not* commutative
    - $\pi_{\text{<list1>}} (\pi_{\text{<list2>}} (R)) = \pi_{\text{<list1>}} (R)$ as long as <list2> contains the attributes in <list1>

Examples of applying SELECT and PROJECT operations

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

(a)

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	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 Fire Oak, Humble, TX	M	38000	333445555	5

(b)

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

(c)

Sex	Salary
M	30000
M	40000
F	25000
F	43000
M	38000
M	25000
M	55000

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

Figure 8.1
Results of SELECT and PROJECT operations.

$\pi_{\text{Lname, Fname, Salary}}$(EMPLOYEE)

Relational Algebra Expressions

- **We may want to apply several relational algebra operations one after the other**
  - Either we can write the operations as a single **relational algebra expression** by nesting the operations, or
  - We can apply one operation at a time and create **intermediate result relations**.

- **In the latter case, we must give names to the relations that hold the intermediate results.**

Single expression versus sequence of relational operations (Example)

- **To retrieve the first name, last name, and salary of all employees who work in department number 5, we must apply a select and a project operation**

- **We can write a *single relational algebra expression* as follows:**
  - $\pi_{\text{FNAME, LNAME, SALARY}}(\sigma_{\text{DNO=5}}(\text{EMPLOYEE}))$

- **OR We can explicitly show the *sequence of operations*, giving a name to each intermediate relation:**
  - $\boxed{\text{DEP5\_EMPS}} \leftarrow \sigma_{\text{DNO=5}}(\text{EMPLOYEE})$
  - $\text{RESULT} \leftarrow \pi_{\text{FNAME, LNAME, SALARY}}(\boxed{\text{DEP5\_EMPS}})$

Unary Relational Operations: RENAME

- **The RENAME operator is <u>denoted by ρ (rho)</u>**

- **In some cases, <u>we may want to *rename* the attributes</u> of a relation or the relation name or both**
  - Useful when a query requires multiple operations
  - Necessary in some cases (see JOIN operation later)

Unary Relational Operations: RENAME (continued)

- **The general RENAME operation ρ can be expressed by any of the following forms:**
  - $\rho_{S\ (B1,\ B2,\ \ldots,\ Bn\ )}(R)$ changes both:
    - <u>the relation name to S, *and*</u>
    - <u>the column (attribute) names to B1, B1, …..Bn</u>
  - $\rho_S(R)$ changes:
    - <u>the *relation name* only to S</u>
  - $\rho_{(B1,\ B2,\ \ldots,\ Bn\ )}(R)$ changes:
    - <u>the *column (attribute) names* only to B1, B1, …..Bn</u>

Unary Relational Operations: RENAME (continued)

- **For convenience, we also use a *shorthand* for renaming attributes in an intermediate relation:**
  - If we write:
    - RESULT $\leftarrow \pi_{\text{FNAME, LNAME, SALARY}}$ (DEP5_EMPS)
    - RESULT will have the *same attribute names* as DEP5_EMPS (same attributes as EMPLOYEE)
  - If we write:
    - RESULT (F, M, L, S, B, A, SX, SAL, SU, DNO) $\leftarrow \rho_{\text{RESULT (F,M,L,S,B,A,SX,SAL,SU, DNO)}}$(DEP5_EMPS)
    - The 10 attributes of DEP5_EMPS are *renamed* to F, M, L, S, B, A, SX, SAL, SU, DNO, respectively

Note: the $\leftarrow$ symbol is an assignment operator

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	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	1968-01-19	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

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

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

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Example of applying multiple operations and RENAME

(a)

Fname	Lname	Salary
John	Smith	30000
Franklin	Wong	40000
Ramesh	Narayan	38000
Joyce	English	25000

Figure 8.2 Results of a sequence of operations.

(a) $\pi_{\text{Fname, Lname, Salary}}(\sigma_{\text{Dno}=5}(\text{EMPLOYEE}))$.
(b) Using intermediate relations and renaming of attributes.

(b)
TEMP

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	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

First_name	Last_name	Salary
John	Smith	30000
Franklin	Wong	40000
Ramesh	Narayan	38000
Joyce	English	25000

Relational Algebra Operations from Set Theory: UNION

- **UNION Operation**
  - Binary operation, denoted by $\cup$
  - The result of $R \cup S$, is a relation that includes all tuples that are either in R or in S or in both R and S
  - Duplicate tuples are eliminated
  - The two operand relations R and S must be "type compatible" (or UNION compatible)
    - R and S must have same number of attributes
    - Each pair of corresponding attributes must be type compatible (have same or compatible domains)

Relational Algebra Operations from Set Theory: UNION

- **Example:**
  - To retrieve the social security numbers of all employees who either *work in department 5* (RESULT1 below) or *directly supervise an employee who works in department 5* (RESULT2 below)

  - We can use the UNION operation as follows:

    $\textbf{DEP5\_EMPS} \leftarrow \sigma_{\textbf{DNO=5}}\ \textbf{(EMPLOYEE)}$

    $\textbf{RESULT1} \leftarrow \pi_{\textbf{SSN}}\textbf{(DEP5\_EMPS)}$

    $\textbf{RESULT2(SSN)} \leftarrow \pi_{\textbf{SUPERSSN}}\textbf{(DEP5\_EMPS)}$

    $\textbf{RESULT} \leftarrow \textbf{RESULT1} \cup \textbf{RESULT2}$

  - <u>The union operation produces the tuples that are in either RESULT1 or RESULT2 or both</u>

Relational Algebra Operations from Set Theory: UNION

$\text{DEP5\_EMPS} \leftarrow \sigma_{\text{DNO=5}}\ (\text{EMPLOYEE})$

$\text{RESULT1} \leftarrow \pi_{\text{SSN}}(\text{DEP5\_EMPS})$

$\text{RESULT2(SSN)} \leftarrow \pi_{\text{SUPERSSN}}(\text{DEP5\_EMPS})$

$\text{RESULT} \leftarrow \text{RESULT1} \cup \text{RESULT2}$

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	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	1968-01-19	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

duplicate

RESULT1

Ssn
123456789
333445555
666884444
453453453

RESULT2

Ssn
333445555
888665555

RESULT

Ssn
123456789
333445555
666884444
453453453
888665555

Figure 8.3 Result of the UNION operation RESULT ← RESULT1 ∪ RESULT2.

Relational Algebra Operations from Set Theory

Relational Algebra Operations from Set Theory

- **Type Compatibility of operands is required for the binary set operation UNION $\cup$, (also for INTERSECTION $\cap$, and SET DIFFERENCE –, see next slides)**

- **R1(A1, A2, ..., An) and R2(B1, B2, ..., Bn) are type compatible if:**
  - they have the same number of attributes, and
  - the domains of corresponding attributes are type compatible (i.e. dom(Ai) = dom(Bi) for i=1, 2, ..., n).

- **The resulting relation for R1$\cup$R2 (also for R1$\cap$R2, or R1–R2, see next slides) has the same attribute names as the *first* operand relation R1 (by convention)**

Relational Algebra Operations from Set Theory: INTERSECTION

- **INTERSECTION is <u>denoted by ∩</u>**

- **The result of the operation R ∩ S, is a relation that includes all tuples that are in both R and S**
  - The attribute names in the result will be the same as the attribute names in R

- **The two operand relations R and S <u>must be "type compatible"</u>**

Relational Algebra Operations from Set Theory: SET DIFFERENCE (cont.)

- **SET DIFFERENCE (also called MINUS or EXCEPT) <u>is denoted by –</u>**
- **The result of R – S, is a relation that includes <u>all tuples that are in R but not in S</u>**
  - The attribute names in the result will be the same as the attribute names in R
- **The two operand relations R and S <u>must be "type compatible"</u>**

Example to illustrate the result of UNION, INTERSECT, and DIFFERENCE

STUDENT ∪ INSTRUCTOR

(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

Figure 8.4 The set operations UNION, INTERSECTION, and MINUS.
(a) Two union-compatible relations.
(b) STUDENT ∪ INSTRUCTOR.
(c) STUDENT ∩ INSTRUCTOR.
(d) STUDENT – INSTRUCTOR.
(e) INSTRUCTOR – STUDENT.

STUDENT – INSTRUCTOR

(c)

Fn	Ln
Susan	Yao
Ramesh	Shah

STUDENT ∩ INSTRUCTOR

(d)

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

(e)

Fname	Lname
John	Smith
Ricardo	Browne
Francis	Johnson

INSTRUCTOR – STUDENT

Some properties of UNION, INTERSECT, and DIFFERENCE

- **Notice that both union and intersection are <u>*commutative* operations</u>; that is**
  - $R \cup S = S \cup R$, and $R \cap S = S \cap R$
- **Both <u>union and intersection</u> can be treated as n-ary operations applicable to any number of relations as both are <u>*associative* operations</u>; that is**
  - $R \cup (S \cup T) = (R \cup S) \cup T$
  - $(R \cap S) \cap T = R \cap (S \cap T)$
- **The <u>minus operation is not commutative</u>; that is, in general**
  - $R - S \neq S - R$

Relational Algebra Operations from Set Theory: CARTESIAN PRODUCT

- **CARTESIAN (or CROSS) PRODUCT Operation**
  - This operation is used to combine tuples from two relations in a combinatorial fashion.
  - Denoted by R(A1, A2, . . ., An) x S(B1, B2, . . ., Bm)
  - Result is a relation Q with degree n + m attributes:
    - Q(A1, A2, . . ., An, B1, B2, . . ., Bm), in that order.
  - The resulting relation state has one tuple for each combination of tuples—one from R and one from S.
  - Hence, if R has n_R tuples (denoted as $|R| = n_R$), and S has n_S tuples, then R x S will have $n_R * n_S$ tuples.
  - The two operands do NOT have to be "type compatible"

```
mysql> desc salaries;
+-----------+---------+------+-----+---------+-------+
| Field     | Type    | Null | Key | Default | Extra |
+-----------+---------+------+-----+---------+-------+
| emp_no    | int(11) | NO   | PRI | NULL    |       |
| salary    | int(11) | NO   |     | NULL    |       |
| from_date | date    | NO   | PRI | NULL    |       |
| to_date   | date    | NO   |     | NULL    |       |
+-----------+---------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> desc employees;
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| emp_no     | int(11)       | NO   | PRI | NULL    |       |
| birth_date | date          | NO   |     | NULL    |       |
| first_name | varchar(14)   | NO   |     | NULL    |       |
| last_name  | varchar(16)   | NO   |     | NULL    |       |
| gender     | enum('M','F') | NO   |     | NULL    |       |
| hire_date  | date          | NO   |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql>
mysql> select * from employees  E, salaries S limit 5;
+--------+------------+------------+-----------+--------+------------+--------+--------+------------+------------+
| emp_no | birth_date | first_name | last_name | gender | hire_date  | emp_no | salary | from_date  | to_date    |
+--------+------------+------------+-----------+--------+------------+--------+--------+------------+------------+
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 |  10001 |  60117 | 1986-06-26 | 1987-06-26 |
|  10002 | 1964-06-02 | Bezalel    | Simmel    | F      | 1985-11-21 |  10001 |  60117 | 1986-06-26 | 1987-06-26 |
|  10003 | 1959-12-03 | Parto      | Bamford   | M      | 1986-08-28 |  10001 |  60117 | 1986-06-26 | 1987-06-26 |
|  10004 | 1954-05-01 | Chirstian  | Koblick   | M      | 1986-12-01 |  10001 |  60117 | 1986-06-26 | 1987-06-26 |
|  10005 | 1955-01-21 | Kyoichi    | Maliniak  | M      | 1989-09-12 |  10001 |  60117 | 1986-06-26 | 1987-06-26 |
+--------+------------+------------+-----------+--------+------------+--------+--------+------------+------------+
5 rows in set (0.44 sec)

mysql>
```

Relational Algebra Operations from Set Theory: CARTESIAN PRODUCT (cont.)

- **Generally, <u>CROSS PRODUCT is not a meaningful operation</u>**
  - Can become meaningful when followed by other operations

- **Example (not meaningful):**
  - FEMALE_EMPS $\leftarrow \sigma_{\text{SEX='F'}}$(EMPLOYEE)
  - EMPNAMES $\leftarrow \pi_{\text{FNAME, LNAME, SSN}}$ (FEMALE_EMPS)
  - EMP_DEPENDENTS $\leftarrow$ EMPNAMES x DEPENDENT

- **EMP_DEPENDENTS will contain every combination of EMPNAMES and DEPENDENT**
  - whether or not they are actually related

Relational Algebra Operations from Set Theory: CARTESIAN PRODUCT (cont.)

- **To keep only combinations where the DEPENDENT is related to the EMPLOYEE, we add a SELECT operation as follows**

- **Example (meaningful):**
  - FEMALE_EMPS $\leftarrow \sigma_{\text{SEX='F'}}$(EMPLOYEE)
  - EMPNAMES $\leftarrow \pi_{\text{FNAME, LNAME, SSN}}$ (FEMALE_EMPS)
  - EMP_DEPENDENTS $\leftarrow$ EMPNAMES x DEPENDENT
  - ACTUAL_DEPS $\leftarrow \sigma_{\text{SSN=ESSN}}$(EMP_DEPENDENTS)
  - RESULT $\leftarrow \pi_{\text{FNAME, LNAME, DEPENDENT\_NAME}}$ (ACTUAL_DEPS)

- **RESULT will now contain the name of female employees and their dependents**

Figure 8.5 The CARTESIAN PRODUCT (CROSS PRODUCT) operation.

FEMALE_EMPS $\leftarrow \sigma_{\text{SEX='F'}}$(EMPLOYEE)
EMPNAMES $\leftarrow \pi_{\text{FNAME, LNAME, SSN}}$ (FEMALE_EMPS)

FEMALE_EMPS

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
Alicia	J	Zelaya	999887777	1968-07-19	3321Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291Berry, Bellaire, TX	F	43000	888665555	4
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5

EMPNAMES

Fname	Lname	Ssn
Alicia	Zelaya	999887777
Jennifer	Wallace	987654321
Joyce	English	453453453

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	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	1968-01-19	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

Figure 8.5 The CARTESIAN PRODUCT (CROSS PRODUCT) operation.

FEMALE_EMPS

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
Alicia	J	Zelaya	999887777	1968-07-19	3321Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291Berry, Bellaire, TX	F	43000	888665555	4
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5

EMPNAMES

Fname	Lname	Ssn
Alicia	Zelaya	999887777
Jennifer	Wallace	987654321
Joyce	English	453453453

EMP_DEPENDENTS ← EMPNAMES x DEPENDENT

Results are shown in next slide

Dependent

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Figure 8.5 The CARTESIAN PRODUCT (CROSS PRODUCT) operation.

EMP_DEPENDENTS ← EMPNAMES x 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	. . .

Figure 8.5 The CARTESIAN PRODUCT (CROSS PRODUCT) operation.

ACTUAL_DEPS $\leftarrow \sigma_{\text{SSN=ESSN}}$(EMP_DEPENDENTS)
RESULT $\leftarrow \pi_{\text{FNAME, LNAME, DEPENDENT\_NAME}}$ (ACTUAL_DEPS)

ACTUAL_DEPENDENTS

Fname	Lname	Ssn	Essn	Dependent_name	Sex	Bdate	...
Jennifer	Wallace	987654321	987654321	Abner	M	1942-02-28	...

RESULT

Fname	Lname	Dependent_name
Jennifer	Wallace	Abner

Binary Relational Operations JOIN and DIVISION

Binary Relational Operations: JOIN

- **JOIN Operation (denoted by $\bowtie$)**
  - The sequence of CARTESIAN PRODUCT followed by SELECT is used quite commonly to identify and select related tuples from two relations
  - A special operation, called JOIN combines this sequence into a single operation
  - This operation is very important for any relational database with more than a single relation, because it allows us *combine related tuples* from various relations
  - The general form of a join operation on two relations R(A1, A2, . . ., An) and S(B1, B2, . . ., Bm) is:

$$R \bowtie_{<\text{join condition}>} S$$

  - where R and S can be any relations that result from general *relational algebra expressions*.

Binary Relational Operations: JOIN (cont.)

- **Example: Suppose that we want to retrieve the name of the manager of each department.**
  - To get the manager's name, we need to combine each DEPARTMENT tuple with the EMPLOYEE tuple whose SSN value matches the MGRSSN value in the department tuple.
  - We do this by using the join $\bowtie$ operation.

  - $DEPT\_MGR \leftarrow DEPARTMENT \bowtie_{MGRSSN=SSN} EMPLOYEE$

- <u>**MGRSSN=SSN is the join condition**</u>
  - Combines each department record with the employee who manages the department
  - The join condition can also be specified as DEPARTMENT.MGRSSN= EMPLOYEE.SSN

Binary Relational Operations: JOIN (cont.)

Figure 8.6
Result of the JOIN operation
DEPT_MGR ← DEPARTMENT $\bowtie_{\text{Mgr\_ssn=Ssn}}$ EMPLOYEE.

DEPT_MGR

Dname	Dnumber	Mgr_ssn	. . .	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	. . .

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	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	1968-01-19	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

Some properties of JOIN

- **Consider the following JOIN operation:**
  - R(A1, A2, . . ., An) $\underset{R.Ai=S.Bj}{\bowtie}$ S(B1, B2, . . ., Bm)
  - Result is a relation Q with degree n + m attributes:
    - Q(A1, A2, . . ., An, B1, B2, . . ., Bm), in that order.
  - The resulting relation state has one tuple for each combination of tuples
    - r from R and s from S, but *only if they satisfy the join condition* r[Ai] = s[Bj]
  - Hence, if R has n_R tuples, and S has n_S tuples, then the join result will generally have *less than* $n_R * n_S$ tuples.
  - Only related tuples (based on the join condition) will appear in the result

```
mysql> desc salaries;
+-----------+---------+------+-----+---------+-------+
| Field     | Type    | Null | Key | Default | Extra |
+-----------+---------+------+-----+---------+-------+
| emp_no    | int(11) | NO   | PRI | NULL    |       |
| salary    | int(11) | NO   |     | NULL    |       |
| from_date | date    | NO   | PRI | NULL    |       |
| to_date   | date    | NO   |     | NULL    |       |
+-----------+---------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> desc employees;
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| emp_no     | int(11)       | NO   | PRI | NULL    |       |
| birth_date | date          | NO   |     | NULL    |       |
| first_name | varchar(14)   | NO   |     | NULL    |       |
| last_name  | varchar(16)   | NO   |     | NULL    |       |
| gender     | enum('M','F') | NO   |     | NULL    |       |
| hire_date  | date          | NO   |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql> select * from employees  E, salaries S where E.emp_no = S.emp_no limit 5;
+--------+------------+------------+-----------+--------+------------+--------+--------+------------+------------+
| emp_no | birth_date | first_name | last_name | gender | hire_date  | emp_no | salary | from_date  | to_date    |
+--------+------------+------------+-----------+--------+------------+--------+--------+------------+------------+
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 |  10001 |  60117 | 1986-06-26 | 1987-06-26 |
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 |  10001 |  62102 | 1987-06-26 | 1988-06-25 |
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 |  10001 |  66074 | 1988-06-25 | 1989-06-25 |
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 |  10001 |  66596 | 1989-06-25 | 1990-06-25 |
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 |  10001 |  66961 | 1990-06-25 | 1991-06-25 |
+--------+------------+------------+-----------+--------+------------+--------+--------+------------+------------+
5 rows in set (0.06 sec)
```

Some properties of JOIN

- **The <u>general case of JOIN operation</u> is <u>called a Theta-join</u>:**

$$R \underset{theta}{\bowtie} S$$

- **The <u>join condition is called *theta*</u>**

- ***<u>Theta</u>* <u>can be any general boolean expression</u> on the attributes of R and S; for example:**
  - R.Ai < S.Bj AND (R.Ak = S.Bl OR R.Ap < S.Bq)

- **Most join conditions involve <u>one or more equality conditions “AND”ed together</u>; for example:**
  - R.Ai = S.Bj AND R.Ak = S.Bl AND R.Ap = S.Bq

Binary Relational Operations: EQUIJOIN

- **EQUIJOIN Operation**

- **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.
  - The JOIN seen in the previous example was an EQUIJOIN.

R.Ai = S.Bj AND R.Ak = S.Bl AND R.Ap = S.Bq

Binary Relational Operations: NATURAL JOIN Operation

- **NATURAL JOIN Operation**
  - Another variation of JOIN called NATURAL JOIN *— denoted by * —* was created to get rid of the second (superfluous) attribute in an EQUIJOIN condition.
    - because one of each pair of attributes with identical values is superfluous
  - The standard definition of natural join requires that the two join attributes, or each pair of corresponding join attributes, *have the same name* in both relations
  - If this is not the case, a renaming operation is applied first.

```
mysql> desc salaries;
+-----------+---------+------+-----+---------+-------+
| Field     | Type    | Null | Key | Default | Extra |
+-----------+---------+------+-----+---------+-------+
| emp_no    | int(11) | NO   | PRI | NULL    |       |
| salary    | int(11) | NO   |     | NULL    |       |
| from_date | date    | NO   | PRI | NULL    |       |
| to_date   | date    | NO   |     | NULL    |       |
+-----------+---------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> desc employees;
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| emp_no     | int(11)       | NO   | PRI | NULL    |       |
| birth_date | date          | NO   |     | NULL    |       |
| first_name | varchar(14)   | NO   |     | NULL    |       |
| last_name  | varchar(16)   | NO   |     | NULL    |       |
| gender     | enum('M','F') | NO   |     | NULL    |       |
| hire_date  | date          | NO   |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql> select * from employees  E natural join salaries S limit 5; // 결과 릴레이션에 조인 속성 emp_no가 한번만 출력
+--------+------------+------------+-----------+--------+------------+--------+------------+------------+
| emp_no | birth_date | first_name | last_name | gender | hire_date  | salary | from_date  | to_date    |
+--------+------------+------------+-----------+--------+------------+--------+------------+------------+
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 |  60117 | 1986-06-26 | 1987-06-26 |
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 |  62102 | 1987-06-26 | 1988-06-25 |
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 |  66074 | 1988-06-25 | 1989-06-25 |
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 |  66596 | 1989-06-25 | 1990-06-25 |
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 |  66961 | 1990-06-25 | 1991-06-25 |
+--------+------------+------------+-----------+--------+------------+--------+------------+------------+
5 rows in set (0.00 sec)
```

Binary Relational Operations NATURAL JOIN (continued)

- **Example: To apply a natural join on the DNUMBER attributes of DEPARTMENT and DEPT_LOCATIONS, it is sufficient to write:**
  - DEPT_LOCS ← DEPARTMENT * DEPT_LOCATIONS

- **<u>Only attribute with the same name is DNUMBER</u>**

- **An <u>implicit join condition is created</u> based on this attribute:**
  DEPARTMENT.DNUMBER=DEPT_LOCATIONS.DNUMBER

- **Another example: Q ← R(A,B,C,D) * S(C,D,E)**
  - The implicit join condition includes *each pair* of attributes with the same name, "AND"ed together:
    - R.C=S.C AND R.D.S.D
  - Result keeps only one attribute of each such pair:
    - Q(A,B,C,D,E)

Example of NATURAL JOIN operation

Figure 8.7 Results of two natural join operations. (a) proj_dept ← project * dept. (b) dept_locs ← department * dept_locations.

(a) proj_dept ← project * dept

PROJ_DEPT

Pname	Pnumber	Plocation	Dnum	Dname	Mgr_ssn	Mgr_start_date
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	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

Example of NATURAL JOIN operation

Figure 8.7 Results of two natural join operations. (a) proj_dept ← project * dept. (b) dept_locs ← department * dept_locations.

dept_locs ← department * dept_locations

(b)

DEPT_LOCS

Dname	Dnumber	Mgr_ssn	Mgr_start_date	Location
Headquarters	1	888665555	1981-06-19	Houston
Administration	4	987654321	1995-01-01	Stafford
Research	5	333445555	1988-05-22	Bellaire
Research	5	333445555	1988-05-22	Sugarland
Research	5	333445555	1988-05-22	Houston

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Complete Set of Relational Operations

- **The set of operations including SELECT σ, PROJECT π , UNION $\cup$, DIFFERENCE – , RENAME ρ, 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 (sequence of operations from this set).**

- **For example:**
  - $R \cap S = (R \cup S) - ((R - S) \cup (S - R))$
  - $R \bowtie_{\text{<join condition>}} S = \sigma_{\text{<join condition>}} (R \times S)$

Binary Relational Operations: DIVISION

- **DIVISION Operation**
  - The division operation is applied to two relations
  - $R(Z) \div S(X)$, where X subset Z. Let $Y = Z - X$ (and hence $Z = X \cup Y$); that is, let Y be the set of attributes of R that are not attributes of S.

attributes

  - The result of DIVISION is a relation T(Y) that includes a tuple t if tuples t_R appear in R with $t_R [Y] = t$, and with
    - $t_R [X] = t_s$ *for every tuple* t_s in S.

tuples

  - For a tuple t to appear in the result T of the DIVISION, the values in t must appear in R in combination with *every* tuple in S.

Example of DIVISION

Figure 8.8 The DIVISION operation. (a) Dividing SSN_PNOS by SMITH_PNOS. (b) T ← R ÷ S.

(a)

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

(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

Table 8.1 Operations of Relational Algebra

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 $R_1 * R_2$

continued on next slide

Table 8.1 Operations of Relational Algebra

Table 8.1 Operations of Relational Algebra

OPERATION	PURPOSE	NOTATION
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 $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$

Query Tree Notation

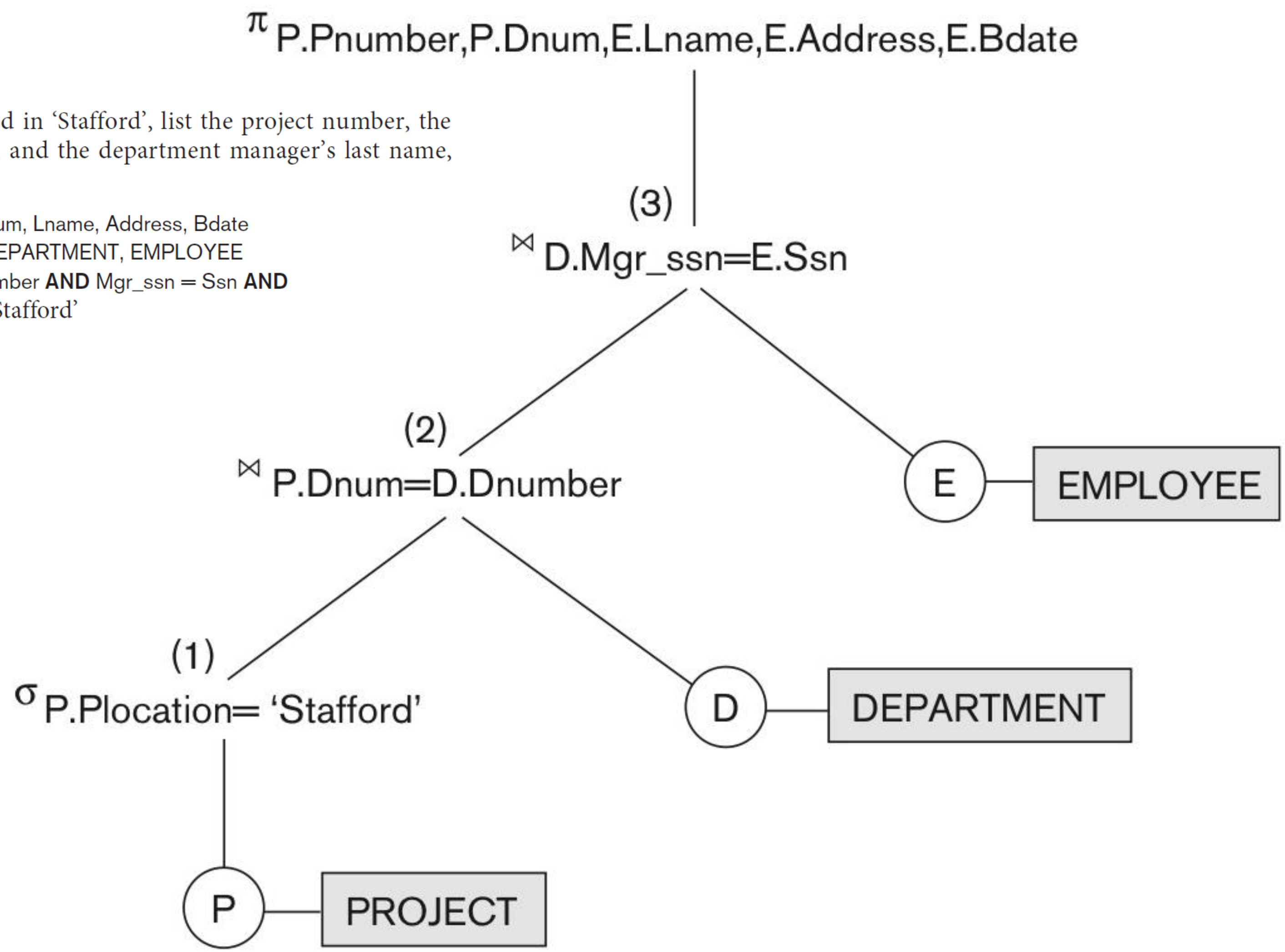
- **Query Tree**
  - An internal data structure to represent a query
  - Standard technique for estimating the work involved in executing the query, the generation of intermediate results, and the optimization of execution
  - Nodes stand for operations like selection, projection, join, renaming, division, ….
  - Leaf nodes represent base relations
  - A tree gives a good visual feel of the complexity of the query and the operations involved
  - Algebraic Query Optimization consists of rewriting the query or modifying the query tree into an equivalent tree. **(see Chapter 15)**

Example of Query Tree

Figure 8.9 Query tree corresponding to the relational algebra expression for Q2.

Query 2. For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, address, and birth date.

```
Q2:  SELECT  Pnumber, Dnum, Lname, Address, Bdate
     FROM    PROJECT, DEPARTMENT, EMPLOYEE
     WHERE   Dnum = Dnumber AND Mgr_ssn = Ssn AND
             Plocation = 'Stafford'
```



Additional Relational Operations

Additional Relational Operations: Aggregate Functions and Grouping

- **A type of request that cannot be expressed in the basic relational algebra is to <u>specify mathematical aggregate functions on collections of values from the database</u>.**

- **Examples of such functions include <u>retrieving the average or total salary of all employees or the total number of employee tuples.</u>**
  - These functions are used in simple statistical queries that summarize information from the database tuples.

- **Common functions applied to collections of numeric values include**
  - SUM, AVERAGE, MAXIMUM, and MINIMUM.

- **The COUNT function is used for counting tuples or values.**

Aggregate Function Operation

- **Use of the Aggregate Functional operation $\mathfrak{I}$**
  - $\mathfrak{I}_{\text{MAX Salary}}$ (EMPLOYEE) retrieves the maximum salary value from the EMPLOYEE relation
  - $\mathfrak{I}_{\text{MIN Salary}}$ (EMPLOYEE) retrieves the minimum Salary value from the EMPLOYEE relation
  - $\mathfrak{I}_{\text{SUM Salary}}$ (EMPLOYEE) retrieves the sum of the Salary from the EMPLOYEE relation
  - $\mathfrak{I}_{\text{COUNT SSN, AVERAGE Salary}}$ (EMPLOYEE) computes the count (number) of employees and their average salary
    - Note: count just counts the number of rows, without removing duplicates

Using Grouping with Aggregation

- **The previous examples all summarized one or more attributes for a set of tuples**
  - Maximum Salary or Count (number of) Ssn

- **<u>Grouping can be combined with Aggregate Functions</u>**

- **Example: For each department, retrieve the DNO, COUNT SSN, and AVERAGE SALARY**

- **A variation of aggregate operation ℑ allows this:**
  - <u>Grouping attribute placed to left of symbol</u>
  - <u>Aggregate functions to right of symbol</u>
  - $_{\text{DNO}}\ \Im\ _{\text{COUNT SSN, AVERAGE Salary}}$ (EMPLOYEE)

- **Above operation <u>groups employees by DNO</u> (department number) and <u>computes the count of employees and average salary per department</u>**

Figure 8.10 The aggregate function operation.

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

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	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	1968-01-19	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

Fname	Minit	Lname	Ssn	. . .	Salary	Super_ssn	Dno
John	B	Smith	123456789		30000	333445555	5
Franklin	T	Wong	333445555		40000	888665555	5
Ramesh	K	Narayan	666884444		38000	333445555	5
Joyce	A	English	453453453	. . .	25000	333445555	5
Alicia	J	Zelaya	999887777		25000	987654321	4
Jennifer	S	Wallace	987654321		43000	888665555	4
Ahmad	V	Jabbar	987987987		25000	987654321	4
James	E	Bong	888665555		55000	NULL	1

Grouping EMPLOYEE tuples by the value of Dno

Dno	Count (*)	Avg (Salary)
5	4	33250
4	3	31000
1	1	55000

Result of Q24

R

(a)

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

Figure 8.10 The aggregate function operation.

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

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	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	1968-01-19	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

Fname	Minit	Lname	Ssn	. . .	Salary	Super_ssn	Dno
John	B	Smith	123456789		30000	333445555	5
Franklin	T	Wong	333445555		40000	888665555	5
Ramesh	K	Narayan	666884444		38000	333445555	5
Joyce	A	English	453453453	. . .	25000	333445555	5
Alicia	J	Zelaya	999887777		25000	987654321	4
Jennifer	S	Wallace	987654321		43000	888665555	4
Ahmad	V	Jabbar	987987987		25000	987654321	4
James	E	Bong	888665555		55000	NULL	1

Grouping EMPLOYEE tuples by the value of Dno

Dno	Count (*)	Avg (Salary)
5	4	33250
4	3	31000
1	1	55000

Result of Q24

(b)

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

Figure 8.10 The aggregate function operation.

c. $\Im_{\text{COUNT Ssn, AVERAGE Salary}}$(EMPLOYEE).

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	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	1968-01-19	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

(c)

Count_ssn	Average_salary
8	35125

Additional Relational Operations (continued)

- **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.

Additional Relational Operations (continued)

- **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.

Figure 8.11 A two-level recursive query.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	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	1968-01-19	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

SUPERVISION

(Borg's Ssn is 888665555)

Ssn1 (Ssn)	Ssn2 (Super_ssn)
123456789	333445555
333445555	888665555
999887777	987654321
987654321	888665555
666884444	333445555
453453453	333445555
987987987	987654321
888665555	null

RESULT1

Ssn
333445555
987654321

(Supervised by Borg)

RESULT2

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 (continued)

- **The OUTER JOIN Operation**
  - In NATURAL JOIN and EQUIJOIN, 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.

Additional Relational Operations (continued)

- 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.

Cross Join vs. Full Outer Join

Table 1

Column 1	Column2
A	X
B	Y
C	Z
D	W

Table 2

Column2	Column3
X	XName
Y	YName
P	PName
O	OName

Cross Join = Cartesian Product

Column 1	Column2	Column2	Column3
A	X	X	XName
A	X	Y	YName
A	X	P	PName
A	X	O	OName
B	Y	X	XName
B	Y	Y	YName
B	Y	P	PName
B	Y	O	OName
C	Z	X	XName
C	Z	Y	YName
C	Z	P	PName
C	Z	O	OName
D	W	X	XName
D	W	Y	YName
D	W	P	PName
D	W	O	OName

Full Outer Join

Column 1	Column2	Column2	Column3
A	X	X	XName
B	Y	Y	YName
C	Z	NULL	NULL
D	W	NULL	NULL
NULL	NULL	P	PName
NULL	NULL	O	OName

Figure 8.12
The result of a LEFT OUTER JOIN operation.

$$TEMP \leftarrow (EMPLOYEE ⟕_{Ssn=Mgr\_ssn} DEPARTMENT)$$
$$RESULT \leftarrow \pi_{Fname,\ Minit,\ Lname,\ Dname}(TEMP)$$

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	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	1968-01-19	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

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

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

Additional Relational Operations (continued)

- **OUTER UNION Operations**
  - The outer union operation was developed to take the union of tuples from two relations if the relations are *not type 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 type compatible.
  - The attributes that are type compatible are represented only once in the result, and those attributes that are not type compatible from either relation are also kept in the result relation T(X, Y, Z).

Additional Relational Operations (continued)

- **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

Examples of Queries in Relational Algebra : Procedural Form

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

$$\text{RESEARCH\_DEPT} \leftarrow \sigma_{\text{Dname}=\text{'Research'}}(\text{DEPARTMENT})$$
$$\text{RESEARCH\_EMPS} \leftarrow (\text{RESEARCH\_DEPT} \bowtie_{\text{Dnumber}=\text{Dno}} \text{EMPLOYEE})$$
$$\text{RESULT} \leftarrow \pi_{\text{Fname, Lname, Address}}(\text{RESEARCH\_EMPS})$$

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

$$\text{ALL\_EMPS} \leftarrow \pi_{\text{Ssn}}(\text{EMPLOYEE})$$
$$\text{EMPS\_WITH\_DEPS(Ssn)} \leftarrow \pi_{\text{Essn}}(\text{DEPENDENT})$$
$$\text{EMPS\_WITHOUT\_DEPS} \leftarrow (\text{ALL\_EMPS} - \text{EMPS\_WITH\_DEPS})$$
$$\text{RESULT} \leftarrow \pi_{\text{Lname, Fname}}(\text{EMPS\_WITHOUT\_DEPS} * \text{EMPLOYEE})$$

Examples of Queries in Relational Algebra – Single expressions

As a single expression, these queries become:

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

$$\text{RESEARCH\_DEPT} \leftarrow \sigma_{\text{Dname}=\text{'Research'}}(\text{DEPARTMENT})$$
$$\text{RESEARCH\_EMPS} \leftarrow (\text{RESEARCH\_DEPT} \bowtie_{\text{Dnumber}=\text{Dno}} \text{EMPLOYEE})$$
$$\text{RESULT} \leftarrow \pi_{\text{Fname, Lname, Address}}(\text{RESEARCH\_EMPS})$$

$$\pi_{\text{Fname, Lname, Address}}\,(\sigma_{\text{Dname}=\text{'Research'}}(\text{DEPARTMENT} \bowtie_{\text{Dnumber}=\text{Dno}}(\text{EMPLOYEE}))$$

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

$$\text{ALL\_EMPS} \leftarrow \pi_{\text{Ssn}}(\text{EMPLOYEE})$$
$$\text{EMPS\_WITH\_DEPS(Ssn)} \leftarrow \pi_{\text{Essn}}(\text{DEPENDENT})$$
$$\text{EMPS\_WITHOUT\_DEPS} \leftarrow (\text{ALL\_EMPS} - \text{EMPS\_WITH\_DEPS})$$
$$\text{RESULT} \leftarrow \pi_{\text{Lname, Fname}}(\text{EMPS\_WITHOUT\_DEPS} * \text{EMPLOYEE})$$

$$\pi_{\text{Lname, Fname}}((\pi_{\text{Ssn}}(\text{EMPLOYEE}) - \rho_{\text{Ssn}}(\pi_{\text{Essn}}(\text{DEPENDENT}))) * \text{EMPLOYEE})$$

Examples of Queries in Relational Algebra

Query 2. For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, address, and birth date.

$$\text{STAFFORD\_PROJS} \leftarrow \sigma_{\text{Plocation}=\text{'Stafford'}}(\text{PROJECT})$$
$$\text{CONTR\_DEPTS} \leftarrow (\text{STAFFORD\_PROJS} \bowtie_{\text{Dnum}=\text{Dnumber}} \text{DEPARTMENT})$$
$$\text{PROJ\_DEPT\_MGRS} \leftarrow (\text{CONTR\_DEPTS} \bowtie_{\text{Mgr\_ssn}=\text{Ssn}} \text{EMPLOYEE})$$
$$\text{RESULT} \leftarrow \pi_{\text{Pnumber, Dnum, Lname, Address, Bdate}}(\text{PROJ\_DEPT\_MGRS})$$

Query 3. Find the names of employees who work on *all* the projects controlled by department number 5.

$$\text{DEPT5\_PROJS} \leftarrow \rho_{(\text{Pno})}(\pi_{\text{Pnumber}}(\sigma_{\text{Dnum}=5}(\text{PROJECT})))$$
$$\text{EMP\_PROJ} \leftarrow \rho_{(\text{Ssn, Pno})}(\pi_{\text{Essn, Pno}}(\text{WORKS\_ON}))$$
$$\text{RESULT\_EMP\_SSNS} \leftarrow \text{EMP\_PROJ} \div \text{DEPT5\_PROJS}$$
$$\text{RESULT} \leftarrow \pi_{\text{Lname, Fname}}(\text{RESULT\_EMP\_SSNS} * \text{EMPLOYEE})$$

Examples of Queries in Relational Algebra

Query 4. Make a list of project numbers for projects that involve an employee whose last name is 'Smith', either as a worker or as a manager of the department that controls the project.

$\text{SMITHS(Essn)} \leftarrow \pi_{\text{Ssn}} (\sigma_{\text{Lname='Smith'}}(\text{EMPLOYEE}))$
$\text{SMITH\_WORKER\_PROJS} \leftarrow \pi_{\text{Pno}}(\text{WORKS\_ON} * \text{SMITHS})$
$\text{MGRS} \leftarrow \pi_{\text{Lname, Dnumber}}(\text{EMPLOYEE} \bowtie_{\text{Ssn=Mgr\_ssn}} \text{DEPARTMENT})$
$\text{SMITH\_MANAGED\_DEPTS(Dnum)} \leftarrow \pi_{\text{Dnumber}} (\sigma_{\text{Lname='Smith'}}(\text{MGRS}))$
$\text{SMITH\_MGR\_PROJS(Pno)} \leftarrow \pi_{\text{Pnumber}}(\text{SMITH\_MANAGED\_DEPTS} * \text{PROJECT})$
$\text{RESULT} \leftarrow (\text{SMITH\_WORKER\_PROJS} \cup \text{SMITH\_MGR\_PROJS})$

Query 5. List the names of all employees with two or more dependents.

Strictly speaking, this query cannot be done in the *basic* (*original*) *relational algebra*. We have to use the AGGREGATE FUNCTION operation with the COUNT aggregate function. We assume that dependents of the *same* employee have *distinct* Dependent_name values.

$T1(\text{Ssn, No\_of\_dependents}) \leftarrow {}_{\text{Essn}} \Im_{\text{COUNT Dependent\_name}}(\text{DEPENDENT})$
$T2 \leftarrow \sigma_{\text{No\_of\_dependents}>2}(T1)$
$\text{RESULT} \leftarrow \pi_{\text{Lname, Fname}}(T2 * \text{EMPLOYEE})$

Examples of Queries in Relational Algebra

Query 7. List the names of managers who have at least one dependent.

$\text{MGRS(Ssn)} \leftarrow \pi_{\text{Mgr\_ssn}}(\text{DEPARTMENT})$
$\text{EMPS\_WITH\_DEPS(Ssn)} \leftarrow \pi_{\text{Essn}}(\text{DEPENDENT})$
$\text{MGRS\_WITH\_DEPS} \leftarrow (\text{MGRS} \cap \text{EMPS\_WITH\_DEPS})$
$\text{RESULT} \leftarrow \pi_{\text{Lname, Fname}}(\text{MGRS\_WITH\_DEPS} * \text{EMPLOYEE})$

Chapter Summary

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

- ~~**Relational Calculus**~~
  - ~~Tuple Relational Calculus~~
  - ~~Domain Relational Calculus~~

- ~~**Overview of the QBE language (appendix C)**~~