

Chapter 6

The Relational Algebra

Query

Query is a question or requesting information.

Query language is a language which is used to retrieve information from a database.

Query language is divided into two types –

Procedural language

Non-procedural language

Procedural language: Information is retrieved from the database by specifying the sequence of operations to be performed.

For Example – **Relational algebra.**

Structure Query language (SQL) is based on relational algebra.

Chapter 6 Outline

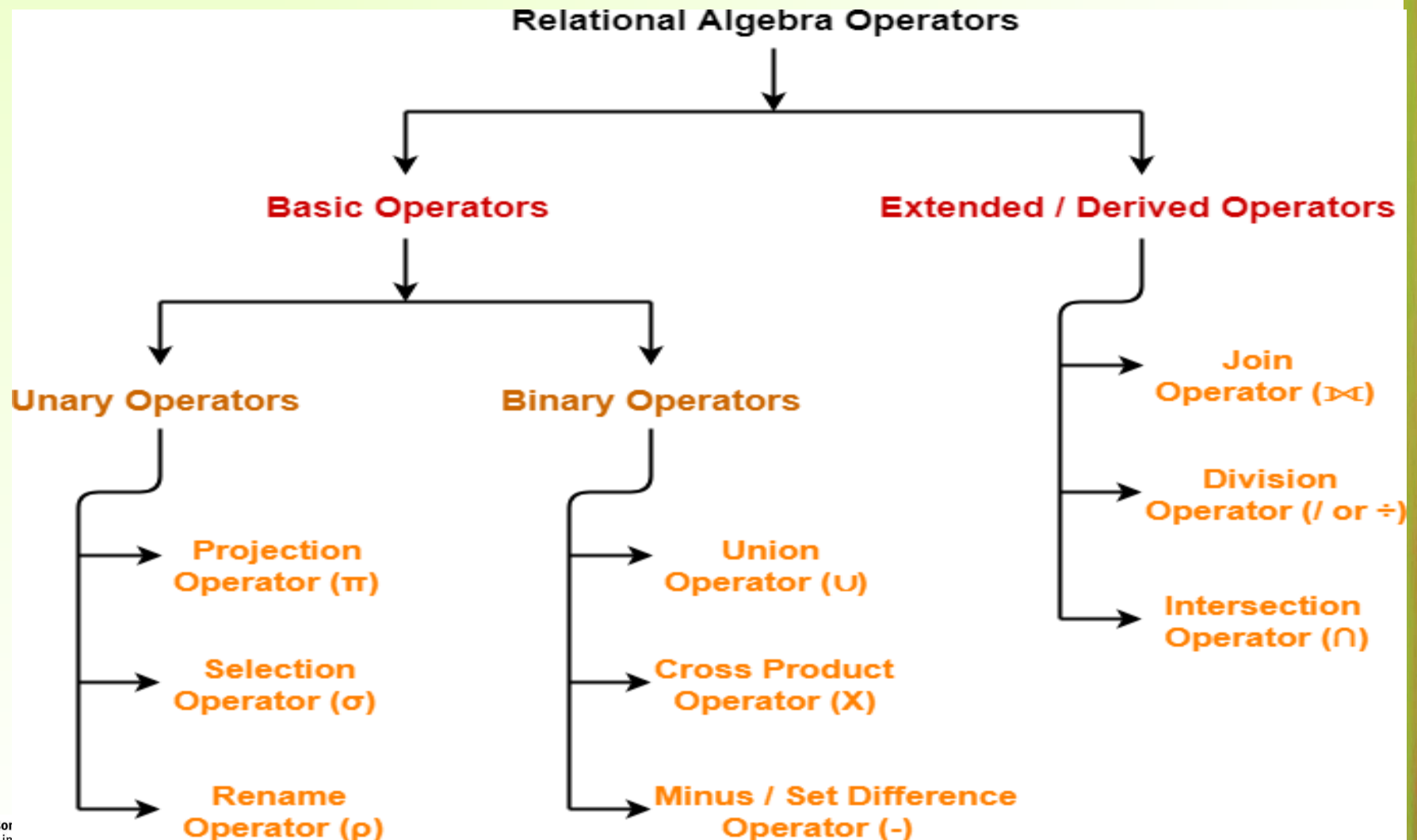
- Unary Relational Operations: SELECT and PROJECT
- Relational Algebra Operations from Set Theory
- Binary Relational Operations: JOIN and DIVISION
- Additional Relational Operations
- Examples of Queries in Relational Algebra

The Relational Algebra and Relational Calculus

- **Relational algebra**
 - Basic set of operations for the relational model
- **Relational algebra expression**
 - Sequence of relational algebra operations
- **Relational calculus**
 - Higher-level declarative language for specifying relational queries

Relational algebra consists of a set of operations that take one or two relations as an input and produces a new relation as output.

RELATIONAL ALGEBRA OPERATORS



EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno
-------	-------	-------	------------	-------	---------	-----	--------	-----------	-----

DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date
-------	----------------	---------	----------------

DEPT_LOCATIONS

<u>Dnumber</u>	<u>Dlocation</u>
----------------	------------------

PROJECT

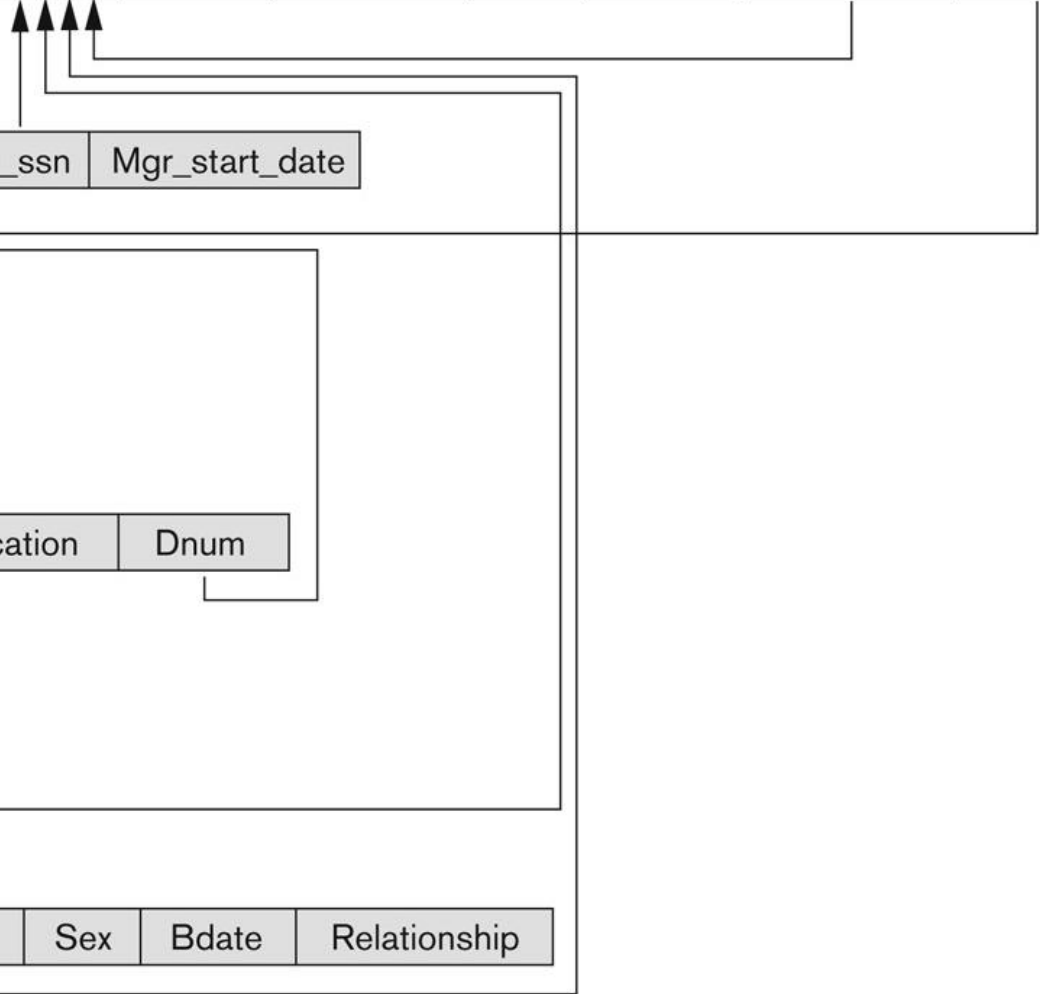
Pname	<u>Pnumber</u>	Plocation	Dnum
-------	----------------	-----------	------

WORKS_ON

<u>Essn</u>	<u>Pno</u>	Hours
-------------	------------	-------

DEPENDENT

<u>Essn</u>	<u>Dependent_name</u>	Sex	Bdate	Relationship
-------------	-----------------------	-----	-------	--------------



Unary Relational Operations: SELECT and PROJECT

- The SELECT Operation
 - Subset of the tuples from a relation that satisfies a selection condition:

$$\sigma_{\langle \text{selection condition} \rangle}(R)$$

- Boolean expression contains clauses of the form
<attribute name> <comparison op> <constant value>
or
- <attribute name> <comparison op> <attribute name>

Unary Relational Operations: SELECT and PROJECT (cont'd.)

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

- **<selection condition>** applied independently to each individual tuple t in R
 - If condition evaluates to TRUE, tuple selected
- Boolean conditions **AND**, **OR**, and **NOT**
- **Unary**
 - Applied to a single relation

Unary Relational Operations: SELECT and PROJECT (cont'd.)

■ Selectivity

- Fraction of tuples selected by a selection condition

The degree (number of attributes) of resulting relation from a Selection operation is same as the degree of the Relation given.

■ SELECT operation commutative

$$\sigma_{c_1}(\sigma_{c_2}(R)) = \sigma_{c_2}(\sigma_{c_1}(R))$$

■ Cascade SELECT operations into a single operation with **AND** condition

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

$$\sigma_{Dno=5}(\sigma_{Salary>30000}(EMPLOYEE))$$

Select Operation - Example

■ Relation r

<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

■ $\sigma_{A=B \text{ AND } D > 5}$
(r)

<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>
α	α	1	7
β	β	23	10

Selection: General form $\sigma_F(R)$

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.

$\sigma_{\text{TITLE}='Elect. Eng.'}(\text{EMP})$

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

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

The PROJECT Operation

- Selects columns from table and discards the other columns:

$$\pi_{\langle \text{attribute list} \rangle}(R)$$

- **Degree**
 - Number of attributes in $\langle \text{attribute list} \rangle$
- **Duplicate elimination**
 - Result of PROJECT operation is a set of distinct tuples

Project Operation - Example

Relation r :

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

$\Pi_{A,C}(r)$

A	C
α	1
α	1
β	1
β	2

=

A	C
α	1
β	1
β	2

Projection General form $\pi_X(R)$

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

$PROJNAME = \pi_{PNAME}(PROJ)$

PNAME
Instrumentation
Database Develop.
CAD/CAM
Maintenance

SELECTION & PROJECTION Example

Person

ID	Name	Address	Hobby
1123	John	123 Main	Stamps
1123	John	123 Main	Coins
5556	Mary	7 Lake Dr	Hiking
9876	Bart	5 Pine St	Stamps

σ Hobby = 'stamps'(Person)

ID	Name	Address	Hobby
1123	John	123 Main	Stamps
9876	Bart	5 Pine St	Stamps

Π Name, Hobby(Person)

Name	Hobby
John	Stamps
John	Coins
Mary	Hiking
Bart	Stamps

Sequences of Operations and the RENAME Operation

- **In-line expression:**

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

- **Sequence of operations:**

$$\begin{aligned}\text{DEP5_EMPS} &\leftarrow \sigma_{\text{Dno}=5}(\text{EMPLOYEE}) \\ \text{RESULT} &\leftarrow \pi_{\text{Fname, Lname, Salary}}(\text{DEP5_EMPS})\end{aligned}$$

- **Rename** attributes in intermediate results

- **RENAME** operation

$$\rho_{S(B_1, B_2, \dots, B_n)}(R) \quad \text{or} \quad \rho_S(R) \quad \text{or} \quad \rho_{(B_1, B_2, \dots, B_n)}(R)$$

Relational Algebra Operations from Set Theory

■ UNION, INTERSECTION, and MINUS

- Merge the elements of two sets in various ways
 - Binary operations
 - Relations must have the same type of tuples
- union compatibility:

for $R(A_1, A_2, \dots, A_n)$ and $S(B_1, B_2, \dots, B_m)$ $n=m$ and
 $\text{dom}(A_i) = \text{dom}(B_i)$

■ UNION

- $R \cup S$
- Includes all tuples that are either in R or in S or in both R and S
- Duplicate tuples eliminated

Union and intersection Operation - Example

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

Relational Algebra Operations from Set Theory (cont'd.)

- INTERSECTION

- $R \cap S$
- Includes all tuples that are in both R and S

- SET DIFFERENCE (or MINUS)

- $R - S$
- Includes all tuples that are in R but not in S

Example

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

■ $r \cap s$:

A	B
α	2

The CARTESIAN PRODUCT (CROSS PRODUCT) Operation

- **CARTESIAN PRODUCT**
 - **CROSS PRODUCT** or **CROSS JOIN**
 - Denoted by \times
 - Binary set operation
 - Relations do not have to be union compatible
 - Useful when followed by a selection that matches values of attributes

Cartesian Product: $R \times S$

R

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

S

E	F
e1	f1
e2	f5

$R \times S$

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

Join Operation


- ❑ Join operation is essentially a Cartesian product followed by a selection criterion.
- ❑ Join operation denoted by \bowtie .
- ❑ JOIN operation also allows joining variously related tuples from different relations.
- ❑ Types of JOIN:

Various forms of join operation are:

- Inner Joins:
 - Theta join
 - EQUI join
 - Natural join
- Outer join:
 - Left Outer Join
 - Right Outer Join
 - Full Outer Join

Binary Relational Operations: JOIN and DIVISION

■ The **JOIN** Operation

- Denoted by 
- Combine related tuples from two relations into single “longer” tuples
- General join condition of the form **<condition> AND <condition> AND...AND <condition>**
- Example:

$$\text{DEPT_MGR} \leftarrow \text{DEPARTMENT} \bowtie_{\text{Mgr_ssn}=\text{Ssn}} \text{EMPLOYEE}$$
$$\text{RESULT} \leftarrow \pi_{\text{Dname, Lname, Fname}}(\text{DEPT_MGR})$$

Binary Relational Operations: JOIN and DIVISION (cont'd.)

■ THETA JOIN

- Each <condition> of the form $A_i \theta B_j$
- A_i is an attribute of R
- B_j is an attribute of S
- A_i and B_j have the same domain
- θ (theta) is one of the comparison operators:
 - $\{=, <, \leq, >, \geq, \neq\}$

θ -Join Example

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.

EMP  SAL E.TITLE=SAL.TITLE

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

SAL

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

<i>sid</i>	<i>sname</i>	<i>rating</i>	<i>age</i>
22	Dustin	7	45.0
31	Lubber	8	55.5
58	Rusty	10	35.0

Figure 4.1 Instance *S1* of Sailors

<i>sid</i>	<i>bid</i>	<i>day</i>
22	101	10/10/96
58	103	11/12/96

Figure 4.3 Instance *R1* of Reserves

- The condition join $S1 \bowtie_{S1.sid < R1.sid} R1$ yields

<i>(sid)</i>	<i>sname</i>	<i>rating</i>	<i>age</i>	<i>(sid)</i>	<i>bid</i>	<i>day</i>
22	Dustin	7	45.0	58	103	11/12/96
31	Lubber	8	55.5	58	103	11/12/96

Figure 4.12 $S1 \bowtie_{S1.sid < R1.sid} R1$

Variations of JOIN: The EQUIJOIN and NATURAL JOIN

■ EQUIJOIN

- Only = comparison operator used
- Always have one or more pairs of attributes that have identical values in every tuple

■ NATURAL JOIN

- Denoted by *
- Removes second (superfluous) attribute in an EQUIJOIN condition

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 EQUIJOIN.

NATURAL JOIN Operation

Because one of each pair of attributes with identical values is superfluous, a new operation called natural join—denoted by *—was created to get rid of the second (superfluous) attribute in an EQUIJOIN condition.

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.

Natural Join Operation - Example

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	ϵ

s

■ $r * s$

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

Complete Set of Relational Operations

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

For example:

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

$$\mathbf{R} \bowtie_{\langle \text{join condition} \rangle} \mathbf{S} = \sigma_{\langle \text{join condition} \rangle} (\mathbf{R} \times \mathbf{S})$$

The DIVISION Operation

- Denoted by \div
- Example: retrieve the names of employees who work on all the projects that 'John Smith' works on
- Apply to relations $R(Z) \div S(X)$
 - Attributes of R are a subset of the attributes of S

$$q = r \div s$$

Then q is the largest relation satisfying $q \times s \subseteq r$

Division Operation - Example

■ Relations r, s :

A	B
α	1
α	2
α	3
β	1
γ	1
δ	1
δ	3
δ	4
ϵ	6
ϵ	1
β	2

r

B
1
2

s

■ $r \div s$:

A
α
β

Another Division Example

- 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

Outerjoin Operator

Left outer join

The left outer join is written as $R \text{ ⋈ } S$ where R and S are relations. The result of the left outer join is the set of all combinations of tuples in R and S that are equal on their common attribute names, in addition to tuples in R that have no matching tuples in S .

Right outer join

The right outer join is written as $R \text{ ⋈ } S$ where R and S are relations. The result of the right outer join is the set of all combinations of tuples in R and S that are equal on their common attribute names, in addition to tuples in S that have no matching tuples in R .

Left Outerjoin Example

For an example consider the tables *Employee* and *Dept* and their left outer join:

Employee

Name	EmpID	DeptName
Harry	3415	Finance
Sally	2241	Sales
George	3401	Finance
Harriet	2202	Sales

Dept

Sales	Harriet
-------	---------

Employee ⋈ Dept

Name	EmpID	DeptName	Mgr
Harry	3415	Finance	NULL
Sally	2241	Sales	Harriet
George	3401	Finance	NULL
Harriet	2202	Sales	Harriet

Right Outerjoin Example

For an example consider the tables *Employee* and *Dept* and their right outer join:

Employee

Name	EmpID	DeptName
Harry	3415	Finance
Sally	2241	Sales
George	3401	Finance
Harriet	2202	Sales

Dept

DeptName	Mgr
Sales	Harriet
Production	Charles

Employee ⋈_⊆ Dept

Name	EmpID	DeptName	Mgr
Sally	2241	Sales	Harriet
Harriet	2202	Sales	Harriet
NULL	NULL	Production	Charles

Full Outer join Example

The **outer join** or **full outer join** in effect combines the results of the left and right outer joins.

For an example consider the tables *Employee* and *Dept* and their full outer join:

Employee

Name	EmpID	DeptName
Harry	3415	Finance
Sally	2241	Sales
George	3401	Finance
Harriet	2202	Sales

Dept

DeptName	Mgr
Sales	Harriet
Production	Charles

Employee  Dept

Name	EmpID	DeptName	Mgr
Harry	3415	Finance	NULL
Sally	2241	Sales	Harriet
George	3401	Finance	NULL
Harriet	2202	Sales	Harriet
NULL	NULL	Production	Charles

Operations of Relational Algebra

Table 6.1 Operations of Relational Algebra

OPERATION	PURPOSE	NOTATION
SELECT	Selects all tuples that satisfy the selection condition from a relation R .	$\sigma_{\langle \text{selection condition} \rangle}(R)$
PROJECT	Produces a new relation with only some of the attributes of R , and removes duplicate tuples.	$\pi_{\langle \text{attribute list} \rangle}(R)$
THETA JOIN	Produces all combinations of tuples from R_1 and R_2 that satisfy the join condition.	$R_1 \bowtie_{\langle \text{join condition} \rangle} R_2$
EQUIJOIN	Produces all the combinations of tuples from R_1 and R_2 that satisfy a join condition with only equality comparisons.	$R_1 \bowtie_{\langle \text{join condition} \rangle} R_2$, OR $R_1 \bowtie_{(\langle \text{join attributes 1} \rangle), (\langle \text{join attributes 2} \rangle)} R_2$
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.	$R_1 \star_{\langle \text{join condition} \rangle} R_2$, OR $R_1 \star_{(\langle \text{join attributes 1} \rangle), (\langle \text{join attributes 2} \rangle)} R_2$ OR $R_1 \star R_2$

Operations of Relational Algebra (cont'd.)

Table 6.1 Operations of Relational Algebra

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

Banking Example (relational algebra queries)

branch (*branch name*, *branch_city*, *assets*)

customer (*customer name*, *customer_street*,
customer_city)

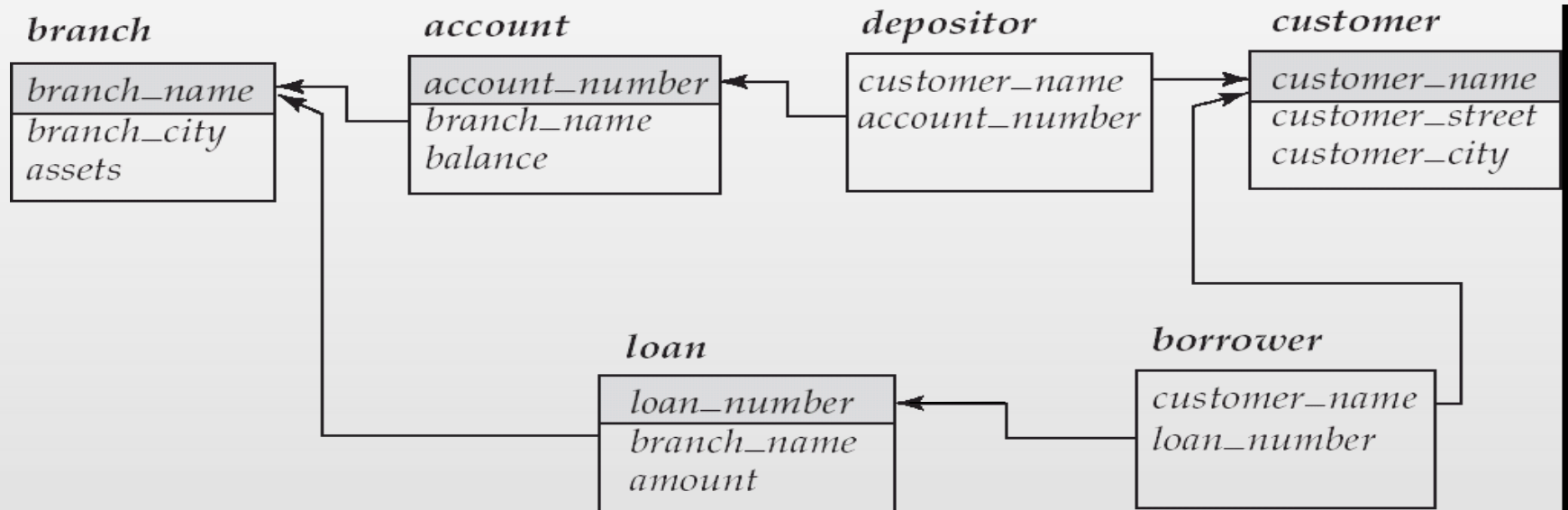
account (*account number*, *branch_name*,
balance)

loan (*loan number*, *branch_name*, *amount*)

depositor (*customer name*,
account number)

borrower (*customer name*, *loan number*)

Example Queries



Example Queries

1. Find all loans of over \$1200

$$\sigma_{amount > 1200} (loan)$$

2. Find the loan number for each loan of an amount greater than \$1200

$$\Pi_{loan_number} (\sigma_{amount > 1200} (loan))$$

3. Find the names of all customers who have a loan, an account, or both, from the bank

$$\Pi_{customer_name} (borrower) \cup \Pi_{customer_name} (depositor)$$

branch (*branch_name*, *branch_city*, *assets*)

customer (*customer_name*, *customer_street*, *customer_city*)

account (*account_number*, *branch_name*, *balance*)

loan (*loan_number*, *branch_name*, *amount*)

depositor (*customer_name*, *account_number*)

borrower (*customer_name*, *loan_number*)

branch (branch_name, branch_city, assets)

customer (customer_name, customer_street, customer_city)

account (account_number, branch_name, balance)

loan (loan_number, branch_name, amount)

depositor (customer_name, account_number)

borrower (customer_name, loan_number)

4. Find the names of all customers who have a loan at the Perryridge branch.

$$\Pi_{customer_name} (\sigma_{branch_name = "Perryridge"} \\ (\sigma_{borrower.loan_number = loan.loan_number} (borrower \times loan)))$$

5. Find the names of all customers who have a loan at the Perryridge branch but do not have an account at any branch of the bank.

$$(\Pi_{customer_name} (\sigma_{branch_name = "Perryridge"} \\ (\sigma_{borrower.loan_number = loan.loan_number} (borrower \times loan)))) - \\ \Pi_{customer_name} (depositor)$$

branch (*branch_name*, *branch_city*, *assets*)
customer (*customer_name*, *customer_street*, *customer_city*)
account (*account_number*, *branch_name*, *balance*)
loan (*loan_number*, *branch_name*, *amount*)
depositor (*customer_name*, *account_number*)
borrower (*customer_name*, *loan_number*)

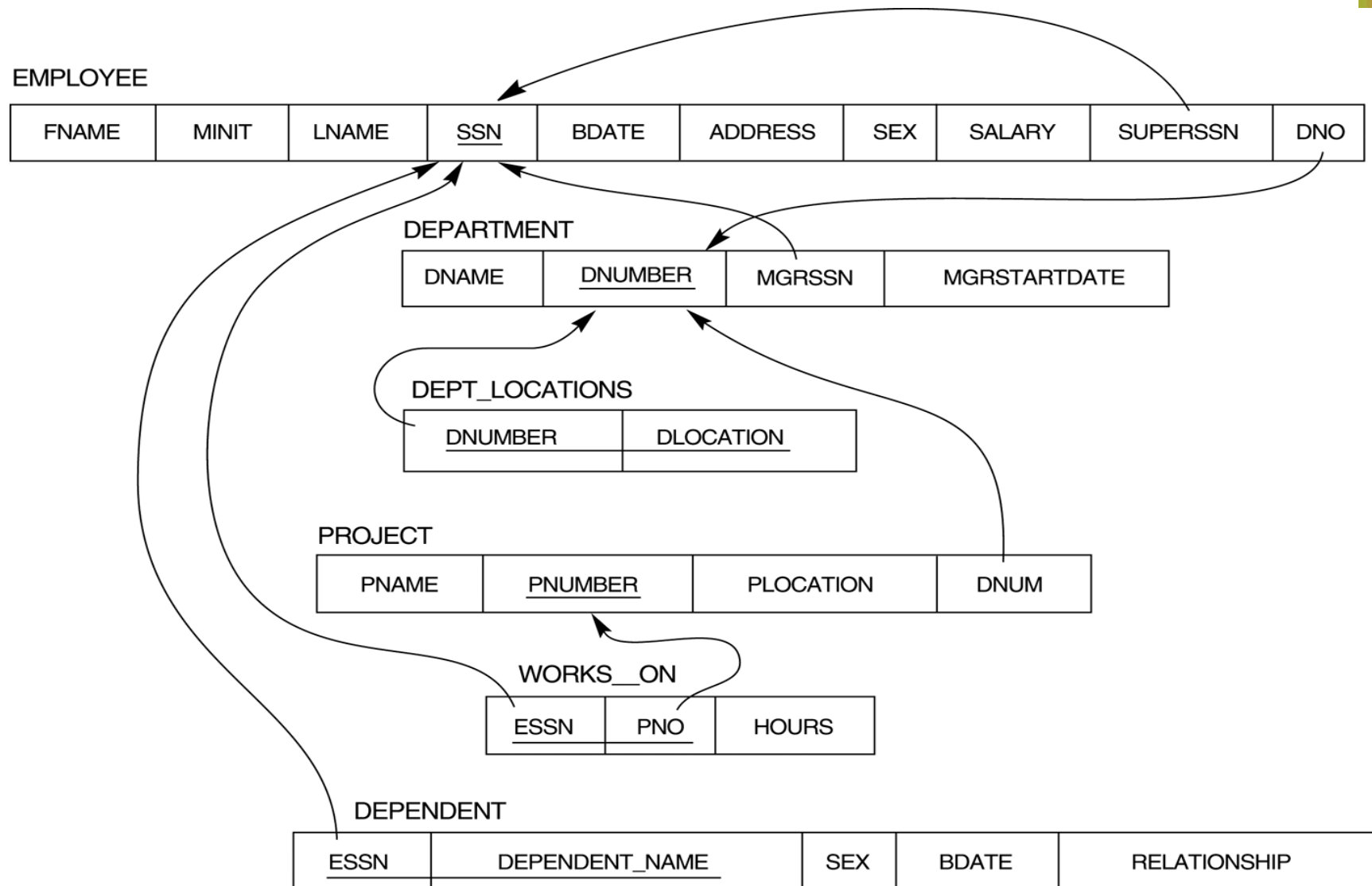
6. Find the names of all customers who have a loan at the Perryridge branch.

- $\Pi_{\text{customer_name}} (\sigma_{\text{branch_name} = \text{"Perryridge"}} (\sigma_{\text{borrower.loan_number} = \text{loan.loan_number}} (\text{borrower} \times \text{loan})))$

OR

- $\Pi_{\text{customer_name}} (\sigma_{\text{loan.loan_number} = \text{borrower.loan_number}} (\sigma_{\text{branch_name} = \text{"Perryridge"}} (\text{loan}) \times \text{borrower}))$

COMPANY database relational schema.



Additional Relational Operations

- **Generalized projection**

- Allows functions of attributes to be included in the projection list

$$\pi_{F_1, F_2, \dots, F_n}(R)$$

- **Aggregate functions and grouping**

- Common functions applied to collections of numeric values
- Include SUM, AVERAGE, MAXIMUM, and MINIMUM

Additional Relational Operations (cont'd.)

- Group tuples by the value of some of their attributes
 - Apply aggregate function independently to each group

$$\langle \text{grouping attributes} \rangle \mathfrak{J} \langle \text{function list} \rangle (R)$$

Populated Database--Fig.5.6

EMPLOYEE	FNAME	MINIT	LNAME	<u>SSN</u>	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

DEPT_LOCATIONS	<u>DNUMBER</u>	DLOCATION
	1	Houston
	4	Stafford
	5	Bellaire
	5	Sugarland
	5	Houston

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

WORKS_ON	<u>ESSN</u>	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	<u>PNUMBER</u>	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	<u>ESSN</u>	<u>DEPENDENT_NAME</u>	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 6.10

The aggregate function operation.

- a. $\rho_{R(Dno, No_of_employees, Average_sal)}(Dno \int COUNT Ssn, AVERAGE Salary(EMPLOYEE)).$
- b. $Dno \int COUNT Ssn, AVERAGE Salary(EMPLOYEE).$
- c. $\int COUNT Ssn, AVERAGE Salary(EMPLOYEE).$

R

(a)

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

⁸Note that this is an arbitrary notation we are suggesting. There is no standard notation.

Examples of Queries in Relational Algebra

Query 1. Retrieve the name and address of all employees who work for the 'Research' department.

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

As a single in-line expression, this query becomes:

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

Examples of Queries in Relational Algebra (cont'd.)

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.

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

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

```
DEPT5_PROJS  $\leftarrow \rho_{(Pno)}(\pi_{Pnumber}(\sigma_{Dnum=5}(PROJECT)))$   
EMP_PROJ  $\leftarrow \rho_{(Ssn, Pno)}(\pi_{Essn, Pno}(WORKS\_ON))$   
RESULT_EMP_SSNS  $\leftarrow EMP\_PROJ \div DEPT5\_PROJS$   
RESULT  $\leftarrow \pi_{Lname, Fname}(RESULT\_EMP\_SSNS * EMPLOYEE)$ 
```

The Tuple Relational Calculus

- Declarative expression
 - Specify a retrieval request nonprocedural language
- Any retrieval that can be specified in basic relational algebra
 - Can also be specified in relational calculus

Summary

- Formal languages for relational model of data:
 - Relational algebra: operations, unary and binary operators
 - Some queries cannot be stated with basic relational algebra operations
 - But are important for practical use
- Relational calculus
 - Based predicate calculus