RELATIONAL ALGEBRA

CHAPTER 6

LECTURE OUTLINE

- Unary Relational Operations: SELECT and PROJECT
- Relational Algebra Operations from Set Theory
- Binary Relational Operations: JOIN and DIVISION
- Query Trees

THE RELATIONAL ALGEBRA

- **Relational algebra**
  - Basic set of operations for the relational model
  - Similar to algebra that operates on numbers
    - Operands and results are relations instead of numbers
- **Relational algebra expression**
  - Composition of relational algebra operations
  - Possible because of *closure* property
- Model for SQL
  - Explain semantics formally
  - Basis for implementations
  - Fundamental to query optimization

SELECT OPERATOR

- Unary operator (one relation as operand)
- Returns subset of the tuples from a relation that satisfies a selection condition:

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

where <selection condition>

- may have Boolean conditions **AND**, **OR**, and **NOT**
- has clauses of the form:

  <attribute name> <comparison op> <constant value>

  or

  <attribute name> <comparison op> <attribute name>

- Applied independently to each individual tuple *t* in operand
  - Tuple selected iff condition evaluates to TRUE
- Example:

$$\sigma_{(Dno=4\ \mathrm{AND}\ Salary>2500)\ \mathrm{OR}\ (Dno=5\ \mathrm{AND}\ Salary>30000)}(\mathrm{EMPLOYEE})$$

SELECT OPERATOR (CONT'D.)

- Do not confuse this with SQL's `SELECT` statement!
- Correspondence
  - Relational algebra

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

```
SELECT *
FROM R
WHERE <selection condition>
```

SELECT OPERATOR PROPERTIES

- Relational model is set-based (no duplicate tuples)
  - Relation *R* has no duplicates, therefore selection cannot produce duplicates.
- Equivalences

$$\sigma_{C_2}\left(\sigma_{C_1}(R)\right) = \sigma_{C_1}\left(\sigma_{C_2}(R)\right)$$
$$\sigma_{C_2}\left(\sigma_{C_1}(R)\right) = \sigma_{C_1 \text{ AND } C_2}(R)$$

- **Selectivity**
  - Fraction of tuples selected by a selection condition

$$\frac{|\sigma_C(R)|}{|R|}$$

WHAT IS THE EQUIVALENT RELATIONAL ALGEBRA EXPRESSION?

Employee

ID	Name	S	Dept	JobType
12	Chen	F	CS	Faculty
13	Wang	M	MATH	Secretary
14	Lin	F	CS	Technician
15	Liu	M	ECE	Faculty

```
SELECT  *
FROM    Employee
WHERE   JobType = 'Faculty';
```

PROJECT OPERATOR

- Unary operator (one relation as operand)
- Keeps specified attributes and discards the others:

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

- **Duplicate elimination**
  - Result of PROJECT operation is a set of distinct tuples
- Example:

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

- Correspondence
  - Relational algebra

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

```
SELECT DISTINCT <attribute list>
FROM R
```

  - Note the need for `DISTINCT` in SQL

PROJECT OPERATOR PROPERTIES

- $\pi_L(R)$ is defined only when $L \subseteq$ *attr* (R)
- Equivalences

$$\pi_{L_2}\left(\pi_{L_1}(R)\right) = \pi_{L_2}(R)$$
$$\pi_L(\sigma_C(R)) = \sigma_C(\pi_L(R))$$

… as long as all attributes used by C are in L

- **Degree**
  - Number of attributes in projected attribute list

WHAT IS THE EQUIVALENT RELATIONAL ALGEBRA EXPRESSION?

Employee

ID	Name	S	Dept	JobType
12	Chen	F	CS	Faculty
13	Wang	M	MATH	Secretary
14	Lin	F	CS	Technician
15	Liu	M	ECE	Faculty

```
SELECT  DISTINCT Name, S, Department
FROM    Employee
WHERE   JobType = 'Faculty';
```

WORKING WITH LONG EXPRESSIONS

- Sometimes easier to write expressions a piece at a time
  - Incremental development
  - Documentation of steps involved
- Consider in-line expression:

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

- Equivalent sequence of operations:

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

$$\text{RESULT} \leftarrow \pi_{\text{Fname,Lname,Salary}}(\text{DEP5\_EMPS})$$

OPERATORS FROM SET THEORY

- Merge the elements of two sets in various ways
  - Binary operators
  - Relations must have the same types of tuples (*union-compatible*)
- UNION
  - $R \cup S$
  - Includes all tuples that are either in *R* or in *S* or in both *R* and *S*
  - Duplicate tuples eliminated
- INTERSECTION
  - $R \cap S$
  - Includes all tuples that are in both *R* and *S*
- DIFFERENCE (or MINUS)
  - $R – S$
  - Includes all tuples that are in *R* but not in *S*

CROSS PRODUCT OPERATOR

- Binary operator
- aka CARTESIAN PRODUCT or CROSS JOIN
- $R \times S$
  - Attributes of result is union of attributes in operands
    - $\deg(R \times S) = \deg(R) + \deg(S)$
  - Tuples in result are all combinations of tuples in operands
    - $|R \times S| = |R| * |S|$
- Relations do *not* have to be union compatible
- Often followed by a selection that matches values of attributes

Course

dept	cnum	instructor	term
CS	338	Jones	Spring
CS	330	Smith	Winter
STATS	330	Wong	Winter

TA

name	major
Ashley	CS
Lee	STATS

Course × TA

dept	cnum	instructor	term	name	major
CS	338	Jones	Spring	Ashley	CS
CS	330	Smith	Winter	Ashley	CS
STATS	330	Wong	Winter	Ashley	CS
CS	338	Jones	Spring	Lee	STATS
CS	330	Smith	Winter	Lee	STATS
STATS	330	Wong	Winter	Lee	STATS

- *What if both operands have an attribute with the same name?*

RENAMING RELATIONS & ATTRIBUTES

- Unary RENAME operator
  - Rename relation

$$\rho_S(R)$$

  - Rename attributes

$$\rho_{(B1,B2,\ldots Bn)}(R)$$

  - Rename relation and its attributes

$$\rho_{S(B1,B2,\ldots,Bn)}(R)$$

Student

name	year
Ashley	4
Lee	3
Dana	1
Jo	1
Jaden	2
Billie	3

- Example: pairing upper year students with freshmen

$$\rho_{\text{Mentor(senior,class)}}\left(\sigma_{\text{year}>2}(\text{Student})\right) \times \sigma_{\text{year}=1}(\text{Student})$$

JOIN OPERATOR

- Binary operator
- $R \bowtie_{<join\ condition>} S$

  where **join condition** is a Boolean expression involving attributes from both operand relations
- Like cross product, combine tuples from two relations into single "longer" tuples, but only those that satisfy matching condition
  - Formally, a combination of cross product and select

$$R \bowtie_{<join\ condition>} S = \sigma_{<join\ condition>}(R \times S)$$

- aka **θ-join** or **inner join**
  - Join condition expressed as $A\ \theta\ B$, where $\theta \in \{=,\neq,>,\geq,<,\leq\}$
  - as opposed to *outer joins*, which will be explained later

JOIN OPERATOR (CONT'D.)

- Examples:
  - What are the names and salaries of all department managers?

$$\pi_{\text{Fname,Lname,Salary}}(\text{DEPARTMENT} \bowtie_{Mgr\_ssn=Ssn} \text{EMPLOYEE})$$

  - Who can TA courses offered by their own department?

Course

dept	cnum	instructor	term
CS	338	Jones	Spring
CS	330	Smith	Winter
STATS	330	Wong	Winter

TA

name	major
Ashley	CS
Lee	STATS

Course $\bowtie_{\text{dept=major}}$ **TA**

dept	cnum	instructor	term	name	major
CS	338	Jones	Spring	Ashley	CS
CS	330	Smith	Winter	Ashley	CS
STATS	330	Wong	Winter	Lee	STATS

- Join selectivity
  - Fraction of number tuples in result over maximum possible

$$\frac{|R \bowtie_C S|}{|R| * |S|}$$

- Common case (as in examples above): **equijoin**

NATURAL JOIN

- $R \bowtie S$
  - No join condition
  - Equijoin on attributes having identical names followed by projection to remove duplicate (superfluous) attributes
- Very common case
  - Often attribute(s) in foreign keys have identical name(s) to the corresponding primary keys

NATURAL JOIN EXAMPLE

- Who has taken a course taught by Anderson?

$$\text{Acourses} \leftarrow \sigma_{\text{Instructor='Anderson'}}(\text{SECTION})$$

$$\pi_{\text{Name,Course\_number,Semester,Year}}(\text{STUDENT} \bowtie \text{GRADE\_REPORT} \bowtie \text{Acourses})$$

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

DIVISION OPERATOR

- Binary operator
- *R* ÷ *S*
  - Attributes of *S* must be a subset of the attributes of *R*
  - attr(*R* ÷ *S*) = attr(*R*) – attr(*S*)
  - *t* tuple in (*R* ÷ *S*) iff (*t* × *S*) is a subset of *R*
- Used to answer questions involving *all*
  - e.g., Which employees work on *all* the critical projects?

Works(enum,pnum)   Critical(pnum)

Works

enum	pnum
E35	P10
E45	P15
E35	P12
E52	P15
E52	P17
E45	P10
E35	P15

Critical

pnum
P15
P10

Works ÷ Critical

enum
E45
E35

(**Works ÷ Critical**) × **Critical**

enum	pnum
E45	P15
E45	P10
E35	P15
E35	P10

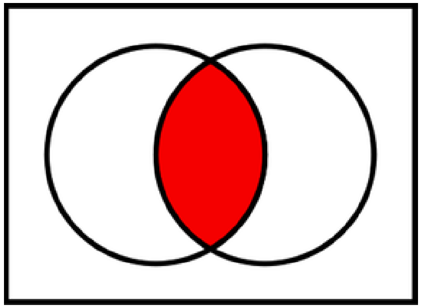
- “Inverse” of cross product

REVIEW OF OPERATORS

Select	$\sigma_{<selection\ condition>}(R)$
Project	$\pi_{<attribute\ list>}(R)$
Rename	$\rho_{<new\ schema>}(R)$
Union	$R \cup S$
Intersection	$R \cap S$
Difference	$R - S$
Cross product	$R \times S$
Join	$R \bowtie_{<join\ condition>} S$
Natural join	$R \bowtie S$
Division	$R \div S$

COMPLETE SET OF OPERATIONS

- Some operators can be expressed in terms of others
  - e.g., $R \cap S = (R \cup S) - \big((R - S) \cup (S - R)\big)$



- Set of relational algebra operations $\{\sigma, \pi, \cup, \rho, -, \times\}$ is complete
  - Other four relational algebra operation can be expressed as a sequence of operations from this set.

1. *Intersection*, as above
2. *Join* is cross product followed by select, as noted earlier
3. *Natural join* is rename followed by join followed by project
4. *Division*: $R \div S = \pi_Y(R) - \pi_Y\big((\pi_Y(R) \times S) - R\big)$

   where Y are attributes in R and not in S

NOTATION FOR QUERY TREES

- Representation for computation
  - cf. arithmetic trees for arithmetic computations
  - Leaf nodes are base relations
  - Internal nodes are relational algebra operations

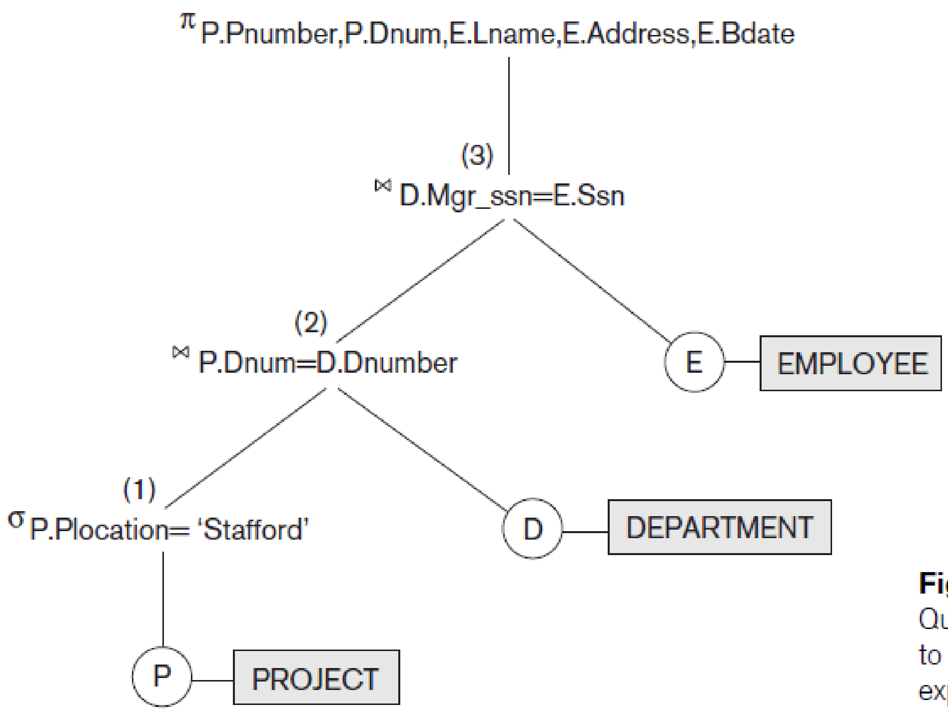


Figure 6.9
Query tree corresponding to the relational algebra expression for Q2.

SAMPLE QUERIES

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.

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

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

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

LECTURE SUMMARY

- Relational algebra
  - Language for relational model of data
  - Collection of unary and binary operators
  - Retrieval queries only, no updates
- Notations
  - Inline
  - Sequence of assignments
  - Operator tree