**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:

𝜎<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:

𝜎 𝐷𝑛𝑜=4 AND 𝑆𝑎𝑙𝑎𝑟𝑦>2500 OR (𝐷𝑛𝑜=5 AND 𝑆𝑎𝑙𝑎𝑟𝑦>30000) EMPLOYEE

# SELECT OPERATOR (CONT’D.)

* Do not confuse this with SQL’s SELECT statement!
* Correspondence
  + Relational algebra

𝜎<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

𝜎𝐶2 𝜎𝐶1(𝑅) = 𝜎𝐶1 𝜎𝐶2(𝑅)

𝜎𝐶2𝜎𝐶1(𝑅)= 𝜎𝐶1 AND 𝐶2(𝑅)

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

𝜎𝐶(𝑅)

𝑅

# WHAT IS THE EQUIVALENT

**RELATIONAL ALGEBRA EXPRESSION?**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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 |

**Employee**

**8**

SELECT \*

FROM Employee

WHERE JobType = 'Faculty';

# PROJECT OPERATOR

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

𝜋<𝑎𝑡𝑡𝑟𝑖𝑏𝑢𝑡𝑒 𝑙𝑖𝑠𝑡> 𝑅

* **Duplicate elimination** 
  + Result of PROJECT operation is a set of distinct tuples Example:

**9**

𝜋𝐹𝑛𝑎𝑚𝑒,𝐿𝑛𝑎𝑚𝑒,𝐴𝑑𝑑𝑟𝑒𝑠𝑠,𝑆𝑎𝑙𝑎𝑟𝑦 EMPLOYEE

* Correspondence
  + Relational algebra

𝜋<𝑎𝑡𝑡𝑟𝑖𝑏𝑢𝑡𝑒 𝑙𝑖𝑠𝑡> 𝑅

* + SQL

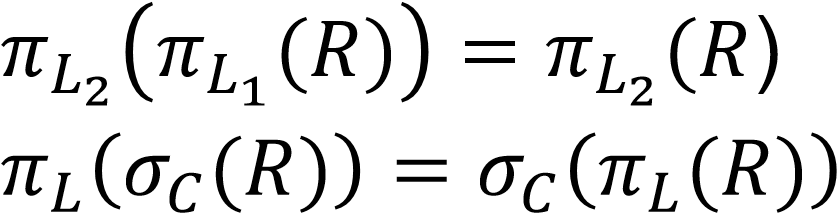
SELECT DISTINCT <attribute list>

FROM R

* + Note the need for DISTINCT in SQL

# PROJECT OPERATOR PROPERTIES

* 𝜋𝐿 𝑅 is defined only when L  *attr* (R )
* Equivalences



… 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?**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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 |

**Employee**

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:

𝜋Fname,Lname,Salary 𝜎Dno=5(EMPLOYEE) Equivalent sequence of operations:

DEP5

\_

EMPS

←

𝜎

Dno

=

5

EMPLOYEE

RESULT

←

𝜋

Fname

,

Lname

,

Salary

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*  *S*
  + Includes all tuples that are either in *R* or in *S* or in both *R* and *S*
  + Duplicate tuples eliminated
* INTERSECTION
  + *R* ∩ *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* × *S* 
  + Attributes of result is union of attributes in operands
  + deg(*R* × *S*) *=* deg(*R)* + deg(S)
  + Tuples in result are all combinations of tuples in operands
  + |*R* × *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

|  |  |
| --- | --- |
| **Student** |  |
| name | year |
| Ashley | 4 |
| Lee | 3 |
| Dana | 1 |
| Jo | 1 |
| Jaden | 2 |
| Billie | 3 |

* + Rename relation

𝜌𝑆 𝑅

* + Rename attributes

𝜌(𝐵1,𝐵2,…𝐵𝑛) 𝑅

* + Rename relation and its attributes

𝜌𝑆(𝐵1,𝐵2,…,𝐵𝑛) 𝑅

* Example: pairing upper year students with freshmen

𝜌Mentor(senior,class) 𝜎year>2 Student × 𝜎year=1 Student

# JOIN OPERATOR

* Binary operator
* *R* ⋈<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 ⋈<𝑗𝑜𝑖𝑛 𝑐𝑜𝑛𝑑𝑖𝑡𝑖𝑜𝑛> 𝑆 *=* 𝜎<𝑗𝑜𝑖𝑛 𝑐𝑜𝑛𝑑𝑖𝑡𝑖𝑜𝑛> 𝑅 × 𝑆

* aka **-join** or **inner join** 
  + Join condition expressed as *A*  *B*, where   {=,,>,,<,}
  + 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?

𝜋Fname,Lname,Salary DEPARTMENT ⋈𝑀𝑔𝑟\_𝑠𝑠𝑛=𝑆𝑠𝑛 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** ⋈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 |*R* ⋈𝐶 *S|*

|*R*| ∗ |*S|*

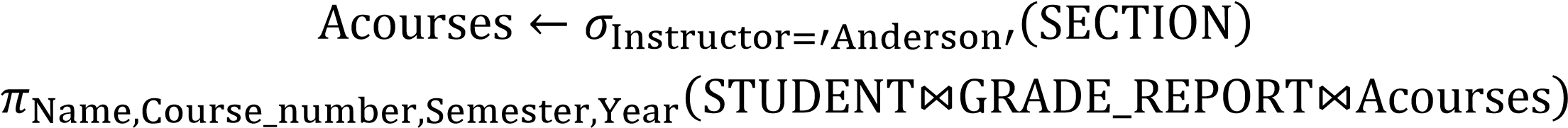
* Common case (as in examples above): **equijoin**

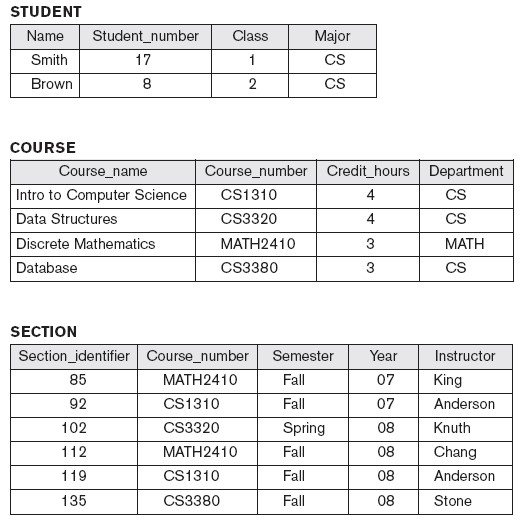
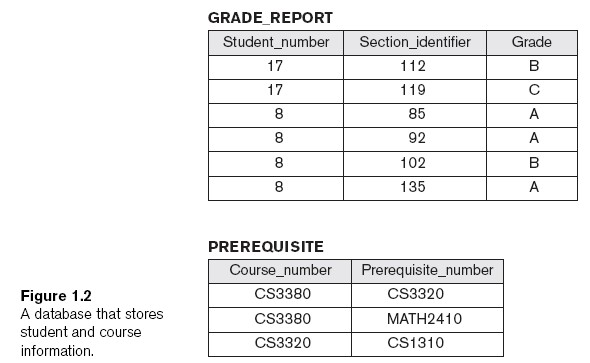
# NATURAL JOIN

* *R* ⋈*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?





# 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 ÷ Critical**) × **Critical** | | |
| enum | pnum |  |
| E45 | P15 |  |
| E45 | P10 |  |
| E35 | P15 |  |
| E35 | P10 |  |

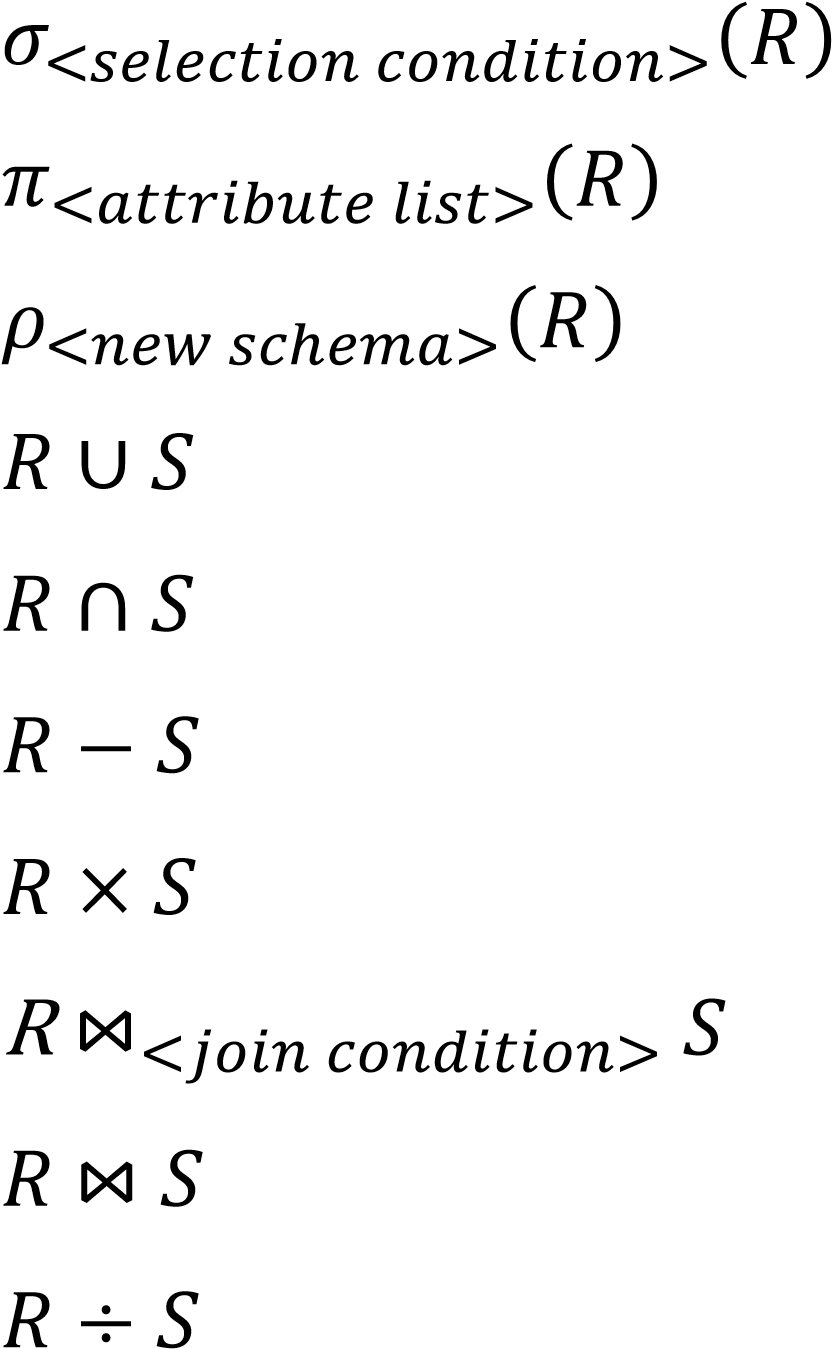
|  |  |
| --- | --- |
| **Works** | |
| enum | pnum |
| E35 | P10 |
| E45 | P15 |
| E35 | P12 |
| E52 | P15 |
| E52 | P17 |
| E45 | P10 |
| E35 | P15 |

|  |  |  |
| --- | --- | --- |
| pnum |  | enum |
| P15 | E45 |
| P10 | E35 |

**Critical Works ÷ Critical**

* “Inverse” of cross product

# REVIEW OF OPERATORS

* Select
* Project
* Rename
* Union
* Intersection
* Difference
* Cross product
* Join
* Natural join
* Division

# COMPLETE SET OF OPERATIONS

* Some operators can be expressed in terms of others

•

e.g.,

𝑅

∩

𝑆

=

𝑅

∪

S

−

𝑅

−

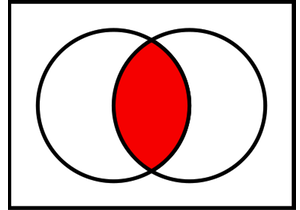
𝑆

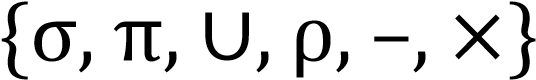
∪

𝑆

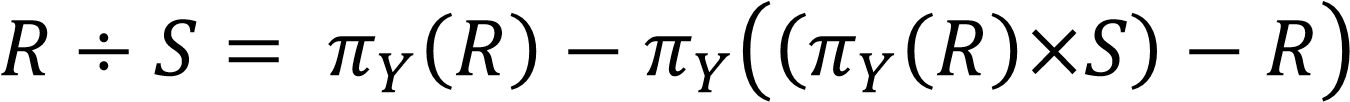
−

𝑅



* Set of relational algebra operations  is complete

• Other four relational algebra operation can be expressed as a sequence of operations from this set.

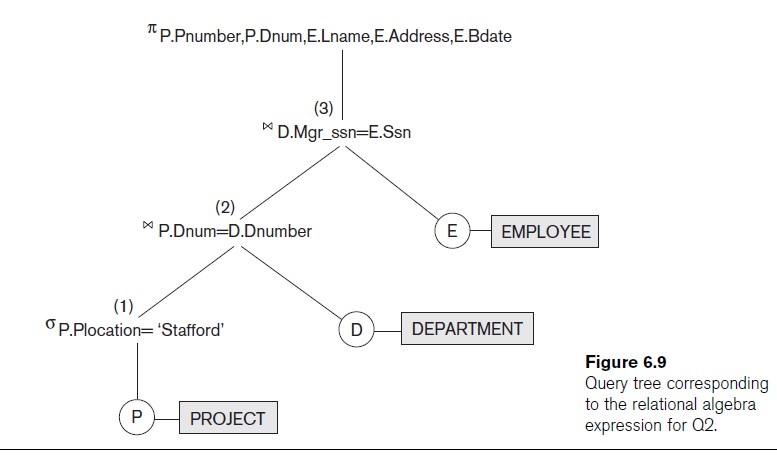
* 1. *Intersection*, as above
  2. *Join* is cross product followed by select, as noted earlier *3. Natural join* is rename followed by join followed by project *4. Division*: 

where  are attributes in  and not in 

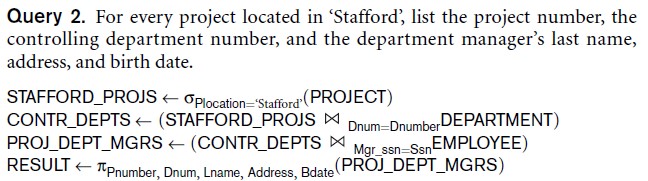
# NOTATION FOR QUERY TREES

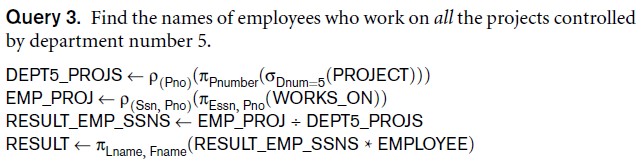
Representation for computation

* cf. arithmetic trees for arithmetic computations
* Leaf nodes are base relations
* Internal nodes are relational algebra operations



# SAMPLE QUERIES





# 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