HOA SEN UNIVERSITY

Lecture 11
The Relational Algebra and Relational Calculus – 2

ĐẠI HỌC HOA SEN

Objectives

- Aggregate Functions and Grouping
- Relational Calculus
  - Tuple Relational Calculus
  - Domain Relational Calculus

- Ref.: Chapter 8

Unary Relational Operations: SELECT Operation Properties

- 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*:
    - $\sigma_{<\text{condition1}>}(\sigma_{<\text{condition2}>}(R)) = \sigma_{<\text{condition2}>}(\sigma_{<\text{condition1}>}(R))$
    - $\sigma_{<\text{cond1}>}(\sigma_{<\text{cond2}>}(\sigma_{<\text{cond3}>}(R))) = \sigma_{<\text{cond2}>}(\sigma_{<\text{cond3}>}(\sigma_{<\text{cond1}>}(R)))$
  - Conjunction of all the conditions:
    - $\sigma_{<\text{cond1}>}(\sigma_{<\text{cond2}>}(\sigma_{<\text{cond3}>}(R))) = \sigma_{<\text{cond1}> \text{ AND } <\text{cond2}> \text{ AND } <\text{cond3}>}(R)$
  - 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

ĐẠI HỌC HOA SEN

Unary Relational Operations: PROJECT Operation Properties

- PROJECT Operation Properties
  - The number of tuples in the result of projection $\pi_{<list>}(R)$ 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
    - $\pi_{<list1>}(\pi_{<list2>}(R)) = \pi_{<list1>}(R)$ as long as <list2> contains the attributes in <list1>

ĐẠI HỌC HOA SEN

Some properties of UNION, INTERSECT, and DIFFERENCE

- Notice that both union and intersection are *commutative* operations; that is
  - $R \cup S = S \cup R$, and $R \cap S = S \cap R$
- Both union and intersection can be treated as n-ary operations applicable to any number of relations as both are *associative* operations; that is
  - $R \cup (S \cup T) = (R \cup S) \cup T$
  - $(R \cap S) \cap T = R \cap (S \cap T)$

- The minus operation is *not commutative*; that is, in general
  - $R - S \neq S - R$

ĐẠI HỌC HOA SEN

Some properties of JOIN

- Consider the following JOIN operation:
  - $R(A_1, A_2, . . ., A_n) \bowtie_{R.Ai=S.Bj} S(B_1, B_2, . . ., B_m)$
  - Result is a relation Q with degree n + m attributes:
    - $Q(A_1, A_2, . . ., A_n, B_1, B_2, . . ., B_m)$, 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* $r[A_i]=s[B_j]$
  - 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

ĐẠI HỌC HOA SEN

Some properties of JOIN (2)

- The general case of JOIN operation is called a Theta-join: R θ S
  - The join condition is called ***theta***

- *Theta* can be any general boolean expression on the attributes of R and S; for example:
  - $R.A_i < S.B_j$ AND ($R.A_k = S.B_l$ OR $R.A_p < S.B_q$)

- Most join conditions involve one or more equality conditions "AND"ed together; for example:
  - $R.A_i = S.B_j$ AND $R.A_k = S.B_l$ AND $R.A_p = S.B_q$

Binary Relational Operations: DIVISION

- DIVISION Operation
  - The division operation is applied to two relations
  - $r \div s = \{t: \forall u \in s \ \& \ <t,u> \in r\}$
  - $R(Z) \div S(X)$, 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.
  - 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.
  - 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.

ĐẠI HỌC HOA SEN

- Give 2 relations

R (A B C D)

A	B	C	D
a	b	c	d
a	b	e	f
b	c	e	f
c	d	c	d
c	d	e	f
a	b	c	d

S (C D)

C	D
c	d
e	f

- Division:

R ÷ S = Q (A B)

A	B
a	b
c	d

SQL statement:

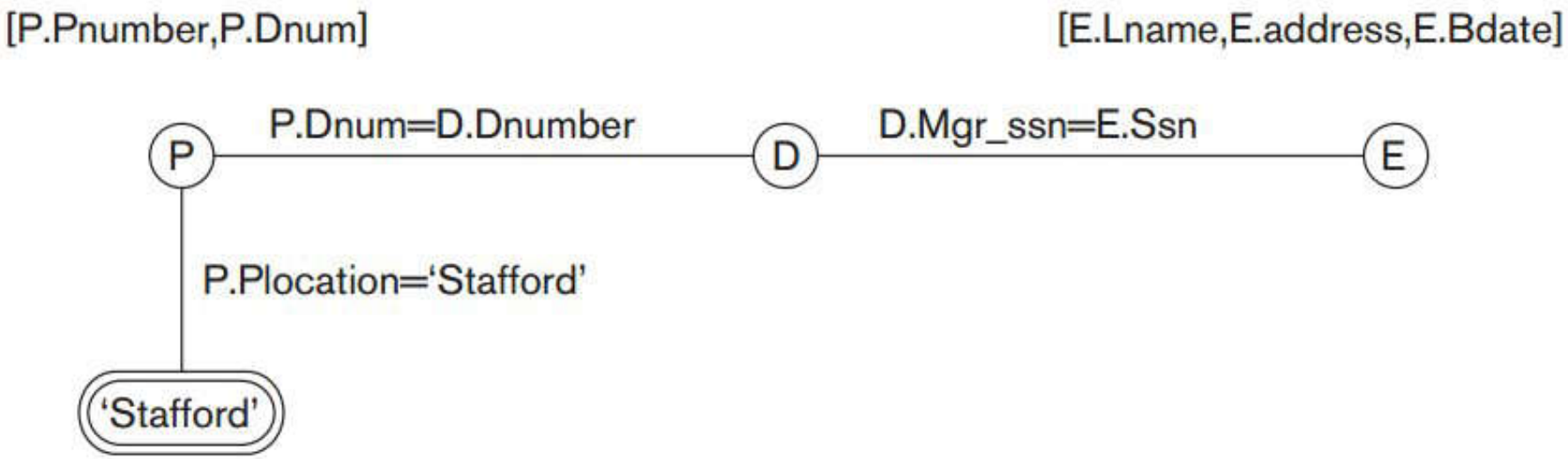
```
Select <attr.>
    From <tab_1> AS a
    Where Not Exists
        (Select *
        From <tab_2> b
        Where Not Exists
            (Select *
            From <tab_1> AS c
            Where (a.<attr._1> = c. <attr._1>)
            And (c. <attr._2> = b. <attr._2>)))
```

Other Example

- For every project located in ‘Stafford’, retrieve the project number, the controlling department number, and the department manager’s last name, address, and birth date:

$$\pi_{\text{Pnumber, Dnum, Lname, Address, Bdate}}(((\sigma_{\text{Plocation='Stafford'}}(\text{PROJECT})) \bowtie_{\text{Dnum=Dnumber}} (\text{DEPARTMENT})) \bowtie_{\text{Mgr\_ssn=Snn}} (\text{EMPLOYEE}))$$

Query Graph


[P.Pnumber,P.Dnum]
[E.Lname,E.address,E.Bdate]
P.Dnum=D.Dnumber
D.Mgr_ssn=E.Ssn
P
D
E
P.Plocation='Stafford'
'Stafford'

ĐẠI HỌC HOA SEN

Additional Relational Operations: Aggregate Functions and Grouping

- A type of request that cannot be expressed in the basic relational algebra is to specify mathematical **aggregate functions** on collections of values from the database.
- Examples of such functions include retrieving the average or total salary of all employees or the total number of employee tuples.
  - 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 $\mathcal{F}$
  - $\mathcal{F}_{\text{MAX Salary}}$ (EMPLOYEE) retrieves the maximum salary value from the EMPLOYEE relation
  - $\mathcal{F}_{\text{MIN Salary}}$ (EMPLOYEE) retrieves the minimum Salary value from the EMPLOYEE relation
  - $\mathcal{F}_{\text{SUM Salary}}$ (EMPLOYEE) retrieves the sum of the Salary from the EMPLOYEE relation
  - $\mathcal{F}_{\text{COUNT Ssn, AVERAGE Salary}}$ (EMPLOYEE) computes the count (number) of employees and their average salary
    - Note: count just counts the number of rows, without removing duplicates

ĐẠI HỌC HOA SEN

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
- Grouping can be combined with Aggregate Functions
  - Example: For each department, retrieve the DNo, COUNT SSN, and AVERAGE SALARY
- A variation of aggregate operation $\mathcal{F}$ allows this:
  - Grouping attribute placed to left of symbol
  - Aggregate functions to right of symbol
  - $_{\text{DNO}}\mathcal{F}_{\text{COUNT (SSN), AVERAGE (Salary)}}$ (EMPLOYEE)
- Above operation groups employees by DNO (department number) and computes the count of employees and average salary per department

ĐẠI HỌC
HOA SEN

```
Select Dno, count(SSN)
        From Employee
        Where Sal > 20000
        Group by Dno
        Having count(SSN) > 2;
```

π Dno, count(SSN)
$(\sigma_{COUNT(SSN) > 2}\ (_{DNO}\mathcal{F}_{COUNT(SSN)}(\sigma_{Sal > 20000}(EMPLOYEE))$

Additional Relational Operations (2)

- 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.

ĐẠI HỌC HOA SEN

Additional Relational Operations: Outer Join

- 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.

ĐẠI HỌC HOA SEN

Additional Relational Operations: Outer Join (2)

- 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.

ĐẠI HỌC HOA SEN

Additional Relational Operations: Outer Union

- 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).

Examples of Queries in Relational Algebra

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

$$RESEARCH\_DEPT \leftarrow \sigma_{DNAME='Research'} (DEPARTMENT)$$

$$RESEARCH\_EMPS \leftarrow (RESEARCH\_DEPT \bowtie_{DNUMBER= DNO} EMPLOYEE)$$

$$RESULT \leftarrow \pi_{FNAME, LNAME, ADDRESS} (RESEARCH\_EMPS)$$

Or

$$\pi_{FNAME, LNAME, ADDRESS} (\sigma_{DNAME='Research'} (DEPARTMENT) \bowtie_{DNUMBER= DNO} (EMPLOYEE))$$

Examples of Queries in Relational Algebra (2)

Retrieve the names of employees who have no dependents.

$ALL\_EMPS \leftarrow \pi_{SSN}(EMPLOYEE)$

$EMPS\_WITH\_DEPS(SSN) \leftarrow \pi_{ESSN}(DEPENDENT)$

$EMPS\_WITHOUT\_DEPS \leftarrow (ALL\_EMPS - EMPS\_WITH\_DEPS)$

$RESULT \leftarrow \pi_{LName, FName}(EMPS\_WITHOUT\_DEPS * EMPLOYEE)$

Or

$\pi_{LName, FName}((\pi_{SSN}(EMPLOYEE) - \pi_{ESSN}(DEPENDENT)) * EMPLOYEE)$

ĐẠI HỌC HOA SEN

Relational Calculus

- A **relational calculus** expression creates a new relation, which is specified in terms of variables that range over rows of the stored database relations (in **tuple calculus**) or over columns of the stored relations (in **domain calculus**).
- In a calculus expression, there is *no order of operations* to specify how to retrieve the query result—a calculus expression specifies only what information the result should contain.
  - This is the main distinguishing feature between relational algebra and relational calculus.

ĐẠI HỌC HOA SEN

Relational Calculus (2)

- Relational calculus is considered to be a **nonprocedural** language.

- This differs from relational algebra, where we must write a *sequence of operations* to specify a retrieval request; hence relational algebra can be considered as a **procedural** way of stating a query.

ĐẠI HỌC HOA SEN

Tuple Relational Calculus

- The tuple relational calculus is based on specifying a number of tuple variables.
- Each tuple variable usually ranges over a particular database relation, meaning that the variable may take as its value any individual tuple from that relation.
- A simple tuple relational calculus query is of the form

  {t | COND(t)}

  - where t is a tuple variable and COND (t) is a conditional expression involving t.
  - The result of such a query is the set of all tuples t that satisfy COND (t).

Tuple Relational Calculus (2)

- Example: To find the first and last names of all employees whose salary is above $50,000, we can write the following tuple calculus expression:

  {t.FName, t.LName | EMPLOYEE(t) $\wedge$ t.Salary>50000}

  {t | EMPLOYEE $\wedge$ t.Salary>50000}

  - The condition EMPLOYEE(t) specifies that the **range relation** of tuple variable t is EMPLOYEE.
  - The first and last name (PROJECTION $\pi_{FName, LName}$) of each EMPLOYEE tuple t that satisfies the condition t.SALARY>50000 (SELECTION $\sigma_{Salary >50000}$) will be retrieved

The Existential and Universal Quantifiers

- Two special symbols called quantifiers can appear in formulas; these are the universal quantifier (∀) and the existential quantifier (∃).
- Informally, a tuple variable t is bound if it is quantified, meaning that it appears in an (∀ t) or (∃ t) clause; otherwise, it is free.

- If F is a formula, then so are (∃t)(F) and (∀t)(F), where t is a tuple variable.
  - The formula (∃t)(F) is true if the formula F evaluates to true for some (at least one) tuple assigned to free occurrences of t in F; otherwise (∃t)(F) is false.
  - The formula (∀t)(F) is true if the formula F evaluates to true for every tuple (in the universe) assigned to free occurrences of t in F; otherwise (∀t)(F) is false.

The Existential and Universal Quantifiers (2)

- ∀ is called the universal or “for all” quantifier because every tuple in “the universe of” tuples must make F true to make the quantified formula true.
  - (∀d)(d.MgrSSN = ‘333445555’)

- ∃ is called the existential or “there exists” quantifier because any tuple that exists in “the universe of” tuples may make F true to make the quantified formula true.
  - (∃t)(d.Dnumber = t.Dno)

ĐẠI HỌC HOA SEN

Example

- List the name and address of all employees who work for the 'Research' department.

  {*t*.Fname, *t*.Lname, *t*.Address | EMPLOYEE(*t*) $\wedge$ ($\exists d$)(DEPARTMENT(*d*) $\wedge$ *d*.Dname='Research' $\wedge$ *d*.Dnumber=*t*.Dno)}

- For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, birth date, and address
  - {p.Pnumber, p.Dnum, m.Lname, m.Bdate, m.Address | PROJECT(p) $\wedge$ EMPLOYEE(m) $\wedge$ p.Plocation='Stafford' $\wedge$ (($\exists$d)(DEPARTMENT(d) $\wedge$ p.Dnum=d.Dnumber $\wedge$ d.Mgr_ssn=m.Ssn))}

ĐẠI HỌC HOA SEN

Languages Based on Tuple Relational Calculus

- The language **SQL** is based on tuple calculus. It uses the basic block structure to express the queries in tuple calculus:

  SELECT <list of attributes>

  FROM <list of relations>

  WHERE <conditions> ;

- SELECT clause mentions the attributes being projected, the FROM clause mentions the relations needed in the query, and the WHERE clause mentions the selection as well as the join conditions.
  - SQL syntax is expanded further to accommodate other operations.

ĐẠI HỌC HOA SEN

The Domain Relational Calculus

- Another variation of relational calculus called the domain relational calculus, or simply, domain calculus is equivalent to tuple calculus and to relational algebra.
- The language called QBE (Query-By-Example) that is related to domain calculus was developed almost concurrently to SQL at IBM Research, Yorktown Heights, New York.
  - Domain calculus was thought of as a way to explain what QBE does.
- Domain calculus differs from tuple calculus in the type of variables used in formulas:
  - Rather than having variables range over tuples, the variables range over single values from domains of attributes.
- To form a relation of degree n for a query result, we must have n of these domain variables — one for each attribute.

The Domain Relational Calculus (2)

- An expression of the domain calculus is of the form

$\{ x_1, x_2, \ldots, x_n \mid COND(x_1, x_2, \ldots, x_n, x_{n+1}, x_{n+2}, \ldots, x_{n+m})\}$

  - where $x_1, x_2, \ldots, x_n, x_{n+1}, x_{n+2}, \ldots, x_{n+m}$ are domain variables that range over domains (of attributes)
  - and COND is a condition or formula of the domain relational calculus.

Example Query Using Domain Calculus

- Retrieve the birthdate and address of the employee whose name is ‘John B. Smith’.
- Query :

  {uv | (∃ q) (∃ r) (∃ s) (∃ t) (∃ w) (∃ x) (∃ y) (∃ z)

  (EMPLOYEE(qrstuvwxyz) and q=’John’ and r=’B’ and s=’Smith’)}
- Ten variables for the employee relation are needed, one to range over the domain of each attribute in order.
  - Of the ten variables q, r, s, . . ., z, only u and v are free.
- Specify the *requested attributes*, Bdate and Address, by the free domain variables u for Bdate and v for Address
- Specify the condition for selecting a tuple following the bar (|) —
  - namely, that the sequence of values assigned to the variables qrstuvwxyz be a tuple of the employee relation and that the values for q (Fname), r (Minit), and s (Lname) be ‘John’, ‘B’, and ‘Smith’, respectively.

ĐẠI HỌC
HOA SEN

Q & A