Relational Model

Dr. Odelu Vanga

Indian Institute of Information Technology Sri City

http://www.iiits.ac.in/people/regular-faculty/dr-odelu-vanga/

Evaluation Components

- 2 Lab Assignments – 10+10 (2-3rd week of Sept, Nov)
- 3 Quizzes (10+10+10) – 2-3rd week of Sept, Oct, Nov
- Mid – 15
- End – 25
- Class Participation - 10

Project Operation

- Notation:

$$\prod_{A1,\, A2,\, \ldots,\, Ak} (r)$$

where A_1, A_2 are attribute names and r is a relation name.

- The result is defined as the relation of *k* columns obtained by erasing the columns that are not listed
- Duplicate rows removed from result, since relations are sets
- E.g. To eliminate the *branch-name* attribute of *account*

$$\prod_{account\text{-}number,\, balance} (account)$$

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

■ $\Pi_{A,B}\,(r)$

A	B
α	10
α	20
β	30
β	40

- Relation *r*

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

- $\sigma_{A=B \wedge D > 5}\,(r)$

A	B	C	D
α	α	1	7
β	β	23	10

- $\Pi_{A,C}(\sigma_{A=B \wedge D > 5}\,(r))$

A	C
α	1
β	23

Example Queries

- Find all loans of over $1200

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

- Find the loan number for each loan of an amount greater than $1200

$$\Pi_{loan\text{-}number}\ (\sigma_{amount>1200}\ (loan))$$

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:
  - $\text{DEP5\_EMPS} \leftarrow \sigma_{\text{DNO=5}}(\text{EMPLOYEE})$
  - $\text{RESULT} \leftarrow \pi_{\text{FNAME, LNAME, SALARY}}(\text{DEP5\_EMPS})$

Basic Structure

- SQL is based on set and relational operations with certain modifications and enhancements
- A typical SQL query has the form:

  select $A_1, A_2, \ldots, A_n$
  from $r_1, r_2, \ldots, r_m$
  where P

  - A_i - represent attributes
  - r_i - represent relations
  - P - a predicate.
- This query is equivalent to the relational algebra expression.

$$\Pi_{A1, A2, \ldots, An}(\sigma_P (r_1 \text{ x } r_2 \text{ x } \ldots \text{ x } r_m))$$

- The result of an SQL query is a relation.

The select Clause

- The **select** clause corresponds to the projection operation of the relational algebra. It is used to list the attributes desired in the result of a query.
- Find the names of all branches in the *loan* relation

  select *branch-name*
  from *loan*

  In the "pure" relational algebra syntax, the query would be:

  $$\prod_{\text{branch-name}}(loan)$$
- An asterisk in the select clause denotes "all attributes"

  select *
  from *loan*

- NOTE: SQL does not permit the '-' character in names, so you would use, for example, *branch_name* instead of *branch-name* in a real implementation.
- NOTE: SQL names are case insensitive, meaning you can use upper case or lower case. You may wish to use upper case in places where we use bold font.

The select Clause (Cont.)

- SQL allows duplicates in relations as well as in query results.
- To force the elimination of duplicates, insert the keyword **distinct** after **select.**
- Find the names of all branches in the *loan* relations, and remove duplicates

  select distinct *branch-name*
  from *loan*

- The keyword **all** specifies that duplicates not be removed.

  select all *branch-name*
  from *loan*

The select Clause (Cont.)

- The **select** clause can contain arithmetic expressions involving the operation, +, –, *, and /, and operating on constants or attributes of tuples.
- The query:

  select *loan-number, branch-name, amount* * 100
  from *loan*

  would return a relation which is the same as the *loan* relations, except that the attribute *amount* is multiplied by 100.

The where Clause

- The **where** clause corresponds to the selection predicate of the relational algebra. If consists of a predicate involving attributes of the relations that appear in the **from** clause.
- The find all loan number for loans made a the Perryridge branch with loan amounts greater than $1200.

  select *loan-number*
  from *loan*
  where *branch-name* = 'Perryridge' **and** *amount* > 1200
- Comparison results can be combined using the logical connectives **and, or,** and **not.**
- Comparisons can be applied to results of arithmetic expressions.

The where Clause (Cont.)

- SQL Includes a **between** comparison operator in order to simplify **where** clauses that specify that a value be less than or equal to some value and greater than or equal to some other value.
- Find the loan number of those loans with loan amounts between \$90,000 and \$100,000 (that is, ≥\$90,000 and ≤\$100,000)

```
select loan-number
from loan
where amount between 90000 and 100000
```

Set Operations

- The set operations **union, intersect,** and **except** operate on relations and correspond to the relational algebra operations $\cup, \cap, -$.
- Each of the above operations automatically eliminates duplicates; to retain all duplicates use the corresponding multiset versions **union all, intersect all** and **except all.**
- Suppose a tuple occurs *m* times in *r* and *n* times in *s,* then, it occurs:
    - $m + n$ times in *r* **union all** *s*
    - min(*m*,*n*) times in *r* **intersect all** *s*
    - max(0, *m* – *n*) times in *r* **except all** *s*

Union Operation

- Notation: $r \cup s$
- Defined as:

$$r \cup s = \{t \mid t \in r \text{ or } t \in s\}$$

- For $r \cup s$ to be valid.

1. r, s must have the *same arity* (same number of attributes)
2. The attribute domains must be *compatible* (e.g., 2nd column of r deals with the same type of values as does the 2nd column of s)

- E.g. to find all customers with either an account or a loan

$$\Pi_{customer\text{-}name}\,(depositor) \;\cup\; \Pi_{customer\text{-}name}\,(borrower)$$

Union Operation – 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*
α	*2*
β	*1*
β	*3*

Find all customers who have a loan, an account, or both:

(select *customer-name* **from** *depositor*)
union
(select *customer-name* **from** *borrower)*

THANK YOU