Washington University in St. Louis

Engineering

Database Management Systems

- Relational Operations

Doug Shook

Retrieving information

- SELECT
- FROM
- WHERE
- ORDER BY
- GROUP BY

Relational Algebra

- All of these queries can be broken down using relational algebra
  - Operates on sets
- Unary
  - Select
  - Project
  - Rename
- Relational
  - Union
  - Intersection
  - Difference
  - Cartesian Product

Select

- Careful!
  - Not like SELECT

- Specifies which tuples to keep from a relation based on a condition:

$$\sigma_{\text{hire date after 1-1-1990}}\ (\text{EMPLOYEES})$$

Select

- Result set will have the same attributes as the relation
- Its commutative!
- Bound on number of tuples?

Project

- Keeps certain columns, discards the rest

$$\Pi_{\text{LNAME, FNAME, SALARY}}\ (\text{EMPLOYEE})$$

Roles of a DBMS

- Duplicate tuples are removed (why?)
- Bound on tuples?
- Commutative?

Rename

- Applies a different name to attribute(s) of a relation
  - Necessary in some cases

$$\rho_{(A1,\ A2,\ A3,\ A4,\ A5)}\ (R)$$

Union

R $\cup$ S includes all tuples in R, in S, or in both R and S

Relations must contain the same number of columns

Columns must have the same data type

SQL Syntax:

```
SELECT *
 FROM dept_emp
UNION
SELECT *
 FROM dept_manager
```

Intersection and Difference

R $\cap$ S includes all tuples that are in R and S
  * INTERSECT

R – S includes all tuples that are in R but not in S
  Not often used

- Commutative?

Cartesian Product

- Combine tuples from two relations combinatorially
  - R x S

- Number of attributes in the result?

- Number of tuples in the result?

Cartesian Product

- Under what conditions is this a useful operation?
  - What do we have at our disposal to help us with this operation?

JOIN

- Cartesian Product combined with a select
- Example:

```
SELECT *
FROM employees JOIN salaries ON
    employees.emp_no = salaries.emp_no
```

- Can join more than 2 tables if necessary

Outer JOINs

- Three types
  - LEFT
  - RIGHT
  - FULL

- What are “missing” values replaced with?

- What could this be used for?

Combinations

- The six operations (select, project, union, difference, rename, cartesian product) are a complete set
  - Any other expression can be expressed using these operations
- Question: how can we express intersection using these operations?
- What order are the SQL clauses that we've been using executed in?

Aggregates

- "outside" the realm of relational algebra

$$\mathcal{F}_{\text{AVERAGE SALARY}}\ (\text{SALARIES})$$

$$_{\text{emp\_no}}\mathcal{F}_{\text{AVERAGE SALARY}}\ (\text{SALARIES})$$

Query Trees

- Data structure used to organize the operations to be performed

- Example: Employee names and salaries

Relational Calculus

- Relational algebra is procedural
- Relational calculus is declarative
  - No order of operations
- Tuple Calculus:
  {t.first_name | EMPLOYEE(t) AND t.birth_date > 1960-01-01}
- Domain Calculus:
  { uv | (∃r) (∃s) (∃t)(EMPLOYEE(rstuv) and r>1960-01-01)}

Quantifiers

- ∃ is called the existential qualifier. To satisfy the condition, a tuple must exist within the specified domain of tuples.

- ∀ is called the universal qualifier - every tuple must conform to the condition.

Practice Problems

- Write a query to show the names and birthdays of all employees
  - What relational operations did you use?
  - Write the query using relational algebra
  - Write the query using relational (tuple) calculus

Practice Problems

- Write a query to show who the managers are of each employee
  - Hint: you'll need a join (maybe more than 2 tables?)
  - What relational operations did you use?
  - Write the query using relational algebra
  - Write the query using relational (tuple) calculus