CS145 Lecture Notes #6
Relational Algebra

Steps in Building and Using a Database

1. Design schema
2. Create schema in DBMS
3. Load initial data
4. Repeat: execute queries and updates on the database

Database Query Languages

What is a query?

- Given a database, ask questions, get answers
  Example: get all students who are now taking CS145
  Example (from the TPC-D benchmark):

  "The Volume Shipping Query finds, for two given nations, the gross discounted revenues derived from lineitems in which parts were shipped from a supplier in either nation to a customer in the other nation during 1995 and 1996. The query lists the supplier nation, the customer nation, the year, and the revenue from shipments that took place in that year. The query orders the answer by supplier nation, customer nation, and year (all ascending)."

$\leadsto$ Some queries are easy to pose, some are not

$\leadsto$ Some queries are easy for DBMS to answer, some are not

Relational Query Languages

Formal: Relational Algebra, Relational Calculus, Datalog
Practical: SQL, Quel, QBE (Query-by-Example)
What is a relational query?

- Input: a number of relations in your database
- Output: one relation as the answer

Relational Algebra

- Basic operators: selection, projection, cross product, union, difference, and renaming
- Additional operators (can be defined using basic ones): theta-join, natural join, intersection, etc.
- Operands: relations
- Input relation(s) $\longrightarrow$ $\boxed{\text{operator}}$ $\longrightarrow$ output relation

Example:
```
Student(SID, name, age, GPA)
Take(SID, CID)
Course(CID, title)
```

Selection

Notation: $\sigma_p(R)$
Purpose: pick rows according to some criteria
Input: a table R
Output: has the same columns as R, but only the rows of R that satisfy p
Example: the student with SID 123

Example: students with GPA higher than 3.0

Example: straight-A students under 18 or over 21

$\leadsto$ The *selection predicate* p in general can include any columns of R, constants, comparisons such as $=$, $\leq$, etc., and Boolean connectives $\wedge$ (and), $\vee$ (or), $\neg$ (not)

Projection

Notation: $\pi_L(R)$
Purpose: pick columns to output
Input: a table R
Output: has only the columns of R listed in L
Example: SID's and names of all students

Example: SID's of students taking classes
$\leadsto$ Notice the elimination of duplicate rows

Example of composing σ and π: names of students under 18

Product and Joins

Cross Product

Notation: $R \times S$
Purpose: pair rows from two tables
Input: two tables R and S
Output: for each row r in R and each row s in S, output a row rs; the output table has the columns of R and the columns of S
Example: `Student` $\times$ `Take`
$\leadsto$ If column names conflict, prefix the names with the table name and a dot

$\leadsto$ Looks odd to glue unrelated tuples together; why use $\times$ then?
Example: names of students and CID's of the courses they are taking

Theta-Join

Notation: $R \bowtie_p S$
Purpose: relate rows from two tables according to some criteria
Shorthand for: $\sigma_p(R \times S)$
Example: names of students and CID's of the courses they are taking

Natural Join

Notation: $R \bowtie S$
Purpose: relate rows from two tables, and

- enforce equality on all common attributes
- eliminate one copy of common attributes

Shorthand for: $\pi_L(R \bowtie_p S)$, where $L = attrs(R) \cup attrs(S)$, and $p = \bigwedge_{A \in attrs(R) \cap attrs(S)} R.A = S.A$
Example: `Student` $\bowtie$ `Take`

Example: names of students taking calculus

Set Operators

Union: $R \cup S$
Difference: $R - S$
Intersection: $R \cap S$
Input: two tables R and S with identical schema
Output: has the same schema as R and S
$\rightsquigarrow$ Duplicate rows are eliminated (as usual) in union
$\rightsquigarrow$ $R \cap S$ is just a shorthand for $R - (R - S)$
Example of union:

    Student(SID, name, age, GPA)
    GradStudent(SID, name, age, GPA, advisor)
    Find all student SID's

Example of difference: CID's of the courses that nobody is taking

What if we also want course titles?

Renaming

Notation: $\rho_S(R)$, or $\rho_{S(A_1,A_2,\ldots)}(R)$
Purpose: rename a table and/or its columns
Example: SID's of all pairs of classmates

Summary of Relational Algebra

$$
\begin{array}{rcll}
E & ::= & R & \text{where } R \text{ is any table in the database} \\
 & | & \sigma_p(E) & \\
 & | & \pi_L(E) & \\
 & | & E_1 \times E_2 & \\
 & | & E_1 \cup E_2 & \\
 & | & E_1 - E_2 & \\
 & | & \rho_{R(A_1,A_2,\ldots)}(E) & \\
 & & \text{... plus additional ones defined in terms of the above:} & \\
 & | & E_1 \bowtie_p E_2 & \\
 & | & E_1 \bowtie E_2 & \\
 & | & E_1 \cap E_2 &
\end{array}
$$

One tricky example: which students have the highest GPA?
$\leadsto$ When an expressions gets too hairy, it helps to assign some intermediate result tables:

$\leadsto$ Or use an *expression tree*: