Introduction to Data Management

Lecture #12
(Relational Languages II)

Instructor: Mike Carey
mjcarey@ics.uci.edu

It’s time again for....

Friday Nights
With Databases

Brought to you by...
Announcements

- Homework notes
  - HW #3 was due today (tomorrow if one day late)
  - HW #4 will come out on Monday (after the exam)
- Exam notes (time flies!)
  - **Midterm #1 is on Monday (in class)!**
  - We’ll use assigned seating – come early!
  - Bring an 8.5” x 11” (2-sided) cheat sheet if you want!
  - Don’t bank on last-minute Piazza answers…
- Today’s lecture plan:
  - Relational languages continued…
  - **Note:** Today’s info won’t be on Monday’s exam! 😊

Ex: Wisconsin Sailing Club Database

<table>
<thead>
<tr>
<th>Sailors</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>sid</td>
<td>sname</td>
<td>rating</td>
</tr>
<tr>
<td>22</td>
<td>Dustin</td>
<td>7</td>
</tr>
<tr>
<td>29</td>
<td>Brutus</td>
<td>1</td>
</tr>
<tr>
<td>31</td>
<td>Lubber</td>
<td>8</td>
</tr>
<tr>
<td>32</td>
<td>Andy</td>
<td>8</td>
</tr>
<tr>
<td>58</td>
<td>Rusty</td>
<td>10</td>
</tr>
<tr>
<td>64</td>
<td>Horatio</td>
<td>7</td>
</tr>
<tr>
<td>71</td>
<td>Zorba</td>
<td>10</td>
</tr>
<tr>
<td>74</td>
<td>Horatio</td>
<td>9</td>
</tr>
<tr>
<td>85</td>
<td>Art</td>
<td>4</td>
</tr>
<tr>
<td>95</td>
<td>Bob</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reserves</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>sid</td>
<td>bid</td>
<td>date</td>
</tr>
<tr>
<td>22</td>
<td>101</td>
<td>10/10/98</td>
</tr>
<tr>
<td>22</td>
<td>102</td>
<td>10/10/98</td>
</tr>
<tr>
<td>22</td>
<td>103</td>
<td>10/8/98</td>
</tr>
<tr>
<td>22</td>
<td>104</td>
<td>10/7/98</td>
</tr>
<tr>
<td>31</td>
<td>102</td>
<td>11/10/98</td>
</tr>
<tr>
<td>31</td>
<td>103</td>
<td>11/6/98</td>
</tr>
<tr>
<td>31</td>
<td>104</td>
<td>11/12/98</td>
</tr>
<tr>
<td>64</td>
<td>101</td>
<td>9/5/98</td>
</tr>
<tr>
<td>64</td>
<td>102</td>
<td>9/8/98</td>
</tr>
<tr>
<td>74</td>
<td>103</td>
<td>9/8/93</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Boats</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>bid</td>
<td>bname</td>
<td>color</td>
</tr>
<tr>
<td>101</td>
<td>Interlake</td>
<td>blue</td>
</tr>
<tr>
<td>102</td>
<td>Interlake</td>
<td>red</td>
</tr>
<tr>
<td>103</td>
<td>Clipper</td>
<td>green</td>
</tr>
<tr>
<td>104</td>
<td>Marine</td>
<td>red</td>
</tr>
</tbody>
</table>
**REVIEW: Find names of sailors who’ve reserved a red boat**

- Sailors(sid, sname, rating, age)  Reserves(sid, bid, day)
  Boats(bid, bname, color)

  - Information about boat color only available in Boats; so need to do two joins:
    \[ \pi_{sname}((\sigma_{\text{color} = \text{red}} \text{Boats}) \bowtie \text{Reserves} \bowtie \text{Sailors}) \]

  - A more “efficient” solution:
    \[ \pi_{sname}(\pi_{\text{sid}}((\pi_{\text{bid}}\sigma_{\text{color} = \text{red}} \text{Boats}) \bowtie \text{Reserves} \bowtie \text{Sailors})) \]

  *A query optimizer will find the latter, given the 1st query!*

---

**Find sailors who’ve reserved a red or a green boat**

- Sailors(sid, sname, rating, age)  Reserves(sid, bid, day)
  Boats(bid, bname, color)

  - Can identify all red or green boats, then find sailors who’ve reserved one of these boats:
    \[ \rho (\text{Tempboats}, (\sigma_{\text{color} = \text{red}} \lor \text{color} = \text{green} \text{Boats})) \]

    \[ \pi_{sname}(\text{Tempboats} \bowtie \text{Reserves} \bowtie \text{Sailors}) \]

  - Can also define Tempboats using union! (Q: How?)
  - What happens if \( \lor \) is replaced by \( \land \) in this query?
Find sailors who’ve reserved a red and a green boat

Sailors(sid, surname, rating, age)  Reserves(sid, bid, day)  Boats(bid, bname, color)

- Previous approach won’t work! Must identify sailors who’ve reserved red boats and sailors who’ve reserved green boats, then find their intersection (notice that sid is a key for Sailors!):
  \[
  \rho (\text{Tempred}, \pi_{\text{sid}}((\sigma_{\text{color}=\text{red}} \text{Boats}) \bowtie \text{Reserves}))
  \]
  \[
  \rho (\text{Tempgreen}, \pi_{\text{sid}}((\sigma_{\text{color}=\text{green}} \text{Boats}) \bowtie \text{Reserves}))
  \]
  \[
  \pi_{\text{surname}}((\text{Tempred} \cap \text{Tempgreen}) \bowtie \text{Sailors})
  \]

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke
Find the names of sailors who’ve reserved all boats.

Uses division; schemas of the input relations feeding the / operator must be carefully chosen:

\[ \rho (\text{Tempsids}, (\pi_{\text{.sid}} \text{Reserves}) / (\pi_{\text{bid}} \text{Boats})) \]

\[ \pi_{\text{fname}} (\text{Tempsids} \bowtie \text{Sailors}) \]

To find sailors who’ve reserved all ‘Interlake’ boats:

\[ \ldots / \pi_{\text{bid}} (\sigma_{\text{bname} = \text{Interlake}} \text{Boats}) \]

Find the names of sailors who’ve reserved all boats.

Uses division; schemas of the input relations feeding the / operator must be chosen:

\[ \rho (\text{Tempsids}, (\pi_{\text{sid,bid}} \text{Reserves}) / (\pi_{\text{bid}} \text{Boats})) \]

\[ \pi_{\text{fname}} (\text{Tempsids} \bowtie \text{Sailors}) \]

To find sailors who’ve reserved all ‘Interlake’ boats:

\[ \ldots / \pi_{\text{bid}} (\sigma_{\text{bname} = \text{Interlake}} \text{Boats}) \]
Find the names of sailors who’ve reserved all boats:

To find sailors who’ve reserved all boats:

\[
\begin{align*}
\text{Sailors} & \rightarrow \text{sid, sname, rating, age} \\
\text{Reserves} & \rightarrow \text{sid, bid, day} \\
\text{Boats} & \rightarrow \text{bid, bname} \\
\end{align*}
\]

Using division; schemas of the input relations feeding the / operator must be carefully chosen:

\[
\begin{align*}
\text{Sailors} & \bowtie \text{Reserves} / \text{Boats} \\
\text{Sailors.sname} & \rightarrow \text{Sailors.sname} \\
\end{align*}
\]

\[
\begin{align*}
\text{Sailors} & \rightarrow \text{sid, sname, rating, age} \\
\text{Reserves} & \rightarrow \text{sid, bid} \\
\text{Boats} & \rightarrow \text{bid, bname} \\
\end{align*}
\]

RelaX - relational algebra:

\[
\begin{align*}
\text{Sailors} & \rightarrow \text{sid, sname} \\
\text{Reserves} & \rightarrow \text{sid, bid} \\
\text{Boats} & \rightarrow \text{bid, bname} \\
\end{align*}
\]

**PS: RelaX Renaming Example...**
Relational Algebra Summary

- The relational model has (several) rigorously defined query languages that are both simple and powerful in nature.
- Relational algebra is more “operational”; very useful as an internal representation for query evaluation plans. (SQL “EXPLAIN”)
- Several ways of expressing a given query; a query optimizer should choose the most efficient version. (Take CS122C...! 😊)
- We’ll add a few more operators later on…
- Next up for now: Relational Calculus

NEXT: Relational Calculus!

- Comes in two flavors: Tuple relational calculus (TRC) and Domain relational calculus (DRC).
- Calculus has variables, constants, comparison ops, logical connectives and quantifiers.
  - TRC: Variables range over (i.e., get bound to) tuples.
  - DRC: Variables range over domain elements (= field values).
  - Both TRC and DRC are simple subsets of first-order logic.
- Expressions in the calculus are called formulas. An answer tuple is essentially an assignment of constants to variables that make the formula evaluate to true.
- TRC is the basis for various query languages (Quel, SQL, OQL, XQuery, ...), while DRC is the basis for example-based relational query UIs. We’ll study TRC!
Tuple Relational Calculus

- **Query** in TRC has the form:
  \[
  \{ t(\text{attrlist}) \mid P(t) \}
  \]

- **Answer** includes all tuples t with (optionally) specified schema (attrlist) that cause formula \( P(t) \) to be true.

- **Formula** is recursively defined, starting with simple atomic formulas (getting tuples from relations or making comparisons of values), and building up bigger and better Boolean formulas using logical connectives.

TRC Formulas

- **Atomic formula:**
  - \( r \in R \), or \( r \notin R \), or \( r.a \ op \ s.b \), or \( r.a \ op \text{ constant} \)
  - \( \text{op} \) is one of \(<\), \(>, \leq\), \(\geq\), \(\neq\), \(=\)

- **Formula:**
  - an atomic formula, or
  - \( \neg P \), \( P \land Q \), \( P \lor Q \), where \( P \) and \( Q \) are formulas, or
  - \( \exists r \in R (P(r)) \), where variable \( r \) is free in \( P(\ldots) \), or
  - \( \forall r \in R (P(r)) \), where variable \( r \) is free in \( P(\ldots) \), or
  - \( P \Rightarrow Q \) (pronounced “implies”, equivalent to \( (\neg P) \lor Q \))
Free and Bound Variables

- The use of a quantifier such as $\exists t \in T$ or $\forall t \in T$ in a formula is said to bind $t$.
  - A variable that is not bound is free.
- Now let us revisit the definition of a TRC query:
  - $\{ t(a_1, a_2, \ldots) \mid P(t) \}$
- There is an important restriction: the variable $t$ that appears to the left of the $\mid$ (“such that”) symbol must be the only free variable in the formula $P(\ldots)$.
- Let’s look at some examples!

Find sailors with a rating above 7

$\{ s \mid s \in \text{Sailors} \land s.\text{rating} > 7 \}$

- This is equivalent to the more general form:
  - $\{ t(\text{id}, \text{nm}, \text{rtg}, \text{age}) \mid \exists s \in \text{Sailors} \ \text{such that}
    \begin{align*}
    & t.\text{id} = s.\text{sid} \land t.\text{nm} = s.\text{name} \\
    & t.\text{rtg} = s.\text{rating} \land t.\text{age} = s.\text{age} \\
    & s.\text{rating} > 7
    \end{align*}
    \}$

(Q: See how each one specifies the answer’s schema and values...? Note that the second one’s schema is different, as we’ve specified it.)
Find ids of sailors who are older than 30.0 or who have a rating under 8 and are named “Horatio”

\[
\{ (s.id) \mid \exists s \in \text{Sailors} \ ( (s.age > 30.0 \\
\quad \lor (s.rating < 8 \land s.sname = \text{“Horatio”}) ) \\
\quad \land t.id = s.id ) \}
\]

- Things to notice:
  - Again, how result schema and values are specified
  - Use of Boolean formula to specify the query constraints
  - Highly declarative nature of this form of query language!

---

**Ex: TRC Query Semantics**

<table>
<thead>
<tr>
<th>sid</th>
<th>sname</th>
<th>rating</th>
<th>age</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>Dustin</td>
<td>7</td>
<td>45.0</td>
</tr>
<tr>
<td>29</td>
<td>Brutus</td>
<td>1</td>
<td>33.0</td>
</tr>
<tr>
<td>31</td>
<td>Lubber</td>
<td>8</td>
<td>55.5</td>
</tr>
<tr>
<td>32</td>
<td>Andy</td>
<td>8</td>
<td>25.5</td>
</tr>
<tr>
<td>58</td>
<td>Rusty</td>
<td>10</td>
<td>35.0</td>
</tr>
<tr>
<td>64</td>
<td>Horatio</td>
<td>7</td>
<td>35.0</td>
</tr>
<tr>
<td>71</td>
<td>Zorba</td>
<td>10</td>
<td>16.0</td>
</tr>
<tr>
<td>74</td>
<td>Horatio</td>
<td>9</td>
<td>35.0</td>
</tr>
<tr>
<td>85</td>
<td>Art</td>
<td>4</td>
<td>25.5</td>
</tr>
<tr>
<td>95</td>
<td>Bob</td>
<td>3</td>
<td>63.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bid</th>
<th>bname</th>
<th>color</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>Interlake</td>
<td>blue</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>sid</th>
<th>sname</th>
<th>rating</th>
<th>age</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>Dustin</td>
<td>7</td>
<td>45.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>nid</th>
<th>nname</th>
<th>nvalue</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Pi</td>
<td>3.14159...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>sid</th>
<th>sname</th>
<th>rating</th>
<th>age</th>
</tr>
</thead>
<tbody>
<tr>
<td>58</td>
<td>Rusty</td>
<td>10</td>
<td>35.0</td>
</tr>
</tbody>
</table>
To Be Continued...