CS275 – Intro to Databases

ER Model to Relational Schema
Chap. 4.1 – 4.2

Roadmap

• Relational Algebra
• Selection
• Projection
• Joins

Query Languages

• Allow us to get information
• Allows us to build new tables
Query Languages

- Relational algebra
  - Procedural, define steps to get to result
- Relational calculus
  - Declarative, allows us to just specify conditions
  - In the reading but not covered in class

Relational Algebra

- Queries consist of a collection of operators
- Every operator accepts one or two relation instances as input
- Returns a relation as output
- Can combine to create complex queries

Relational Algebra

- Basic operators
  - Selection
  - Projection
  - Union
  - Cross-product
  - Difference
Relational Algebra

- Selection
  \( \sigma_{\text{GPA} > 3.5}(\text{Students}) \)

- Projection
  \( \pi_{\text{SID}, \text{Name}, \text{GPA}}(\text{Students}) \)

Composition

- Display the ID, Name, and GPA of students with a GPA > 3.5
  \( \pi_{\text{SID}, \text{Name}, \text{GPA}}(\sigma_{\text{GPA} > 3.5}(\text{Students})) \)
Set Operators

• Union of two sets of relation instances
• \( R \cup S \)

<table>
<thead>
<tr>
<th></th>
<th>R</th>
<th>S</th>
<th>R U S</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homer</td>
<td>123</td>
<td>Bart  22</td>
<td>Homer 123</td>
</tr>
<tr>
<td>Marge</td>
<td>546</td>
<td>Lisa 33</td>
<td>Marge 546</td>
</tr>
<tr>
<td>Bart</td>
<td>22</td>
<td></td>
<td>Bart 22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Lisa 33</td>
<td>Lisa 33</td>
</tr>
</tbody>
</table>
Set Operators

- Union of two sets of relation instances
- $R \cup S$

<table>
<thead>
<tr>
<th>$R$</th>
<th>$S$</th>
<th>$R \cup S$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homer 123</td>
<td>Moe's Tavern</td>
<td></td>
</tr>
<tr>
<td>Marge 546</td>
<td>Springfield</td>
<td></td>
</tr>
<tr>
<td>Bart 22</td>
<td>Nuclear Plane</td>
<td></td>
</tr>
</tbody>
</table>

Union-compatibility

- Must have the same number of attributes
- Corresponding attributes must have identical domains

Set Operators

- Intersection of two sets of relation instances
- $R \cap S$
- Requires union-compatible relations

<table>
<thead>
<tr>
<th>$R$</th>
<th>$S$</th>
<th>$R \cap S$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homer 123</td>
<td>Bart 22</td>
<td>Bart 22</td>
</tr>
<tr>
<td>Marge 546</td>
<td>Lisa 33</td>
<td></td>
</tr>
<tr>
<td>Bart 22</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Set Operators

- Difference of two sets of relation instances
- $R - S$
- Requires union-compatible relations

\[
\begin{array}{c|c|c}
 R & S & R - S \\
\hline
\text{Homer} & 123 & \text{Homer} \\
\text{Marge} & 546 & \text{Marge} \\
\text{Bart} & 22 & \text{Lisa} \\
\end{array}
\]

Set Operators

- Cross Product
  - All combinations of tuples

\[
\begin{array}{c|c|c|c|c}
 R & S & R \times S \\
\hline
\text{Homer} & 123 & \text{Moe's Tavern} \\
\text{Marge} & 546 & \text{Moe's Tavern} \\
\text{Bart} & 22 & \text{Moe's Tavern} \\
\text{Homer} & 123 & \text{Springfield Nuclear Plant} \\
\text{Marge} & 546 & \text{Springfield Nuclear Plant} \\
\text{Bart} & 22 & \text{Springfield Nuclear Plant} \\
\end{array}
\]

Renaming

- Takes a set of mappings of names and a relation, returning a renamed relation
- $\rho(R(F), E)$
Joins

- Can be created from just cross-products, projections and selections
- More often used than cross product
- Won't blow your stack as often

Joins

- Conditional Join
  \[ R \bowtie S = \sigma_{c} (R \times S) \]
- Example
  - S1 Sailors(sid:integer, sname:string, rating:integer, age:real)
  - S2 Boats(bid:integer, bname:string, color:string)
  - R1 Reserves(sid:integer, bid:integer, day:date)
  \[ S_{1} \bowtie \sigma_{s1.sid=R1.sid} R_{1} \]

Joins

- Example
  - S1 Sailors(sid:integer, sname:string, rating:integer, age:real)
  - S2 Boats(bid:integer, bname:string, color:string)
  - R1 Reserves(sid:integer, bid:integer, day:date)
  \[ S_{1} \bowtie \sigma_{s1.sid=R1.sid} R_{1} \]
  - Returns a new table with all attributes of S1 and R1 with rows from S1 joined together with rows from R1 where the sid are the same
Joins

- Equijoin
  - Special join where the conditions are only a set of equality conditionals
  - Removes redundantly named rows

Joins

- Natural join
  \[ S_1 \Join R_1 \]
  - Special case in which we look at all common fields between two relations
  - Just more shorthand

Joins

- S1 Sails(sid:integer, sname:string, rating:integer, age:real)
- S2 Boats(bid:integer, bname:string, color:string)
- R1 Reserves(sid:integer, bid:integer, day:date)

- Find the names of sailors who have reserved boat with bid 103
Joins

- S1 Sailors(sid:integer, sname:string, rating:integer, age:real)
- S2 Boats(bid:integer, bname:string, color:string)
- R1 Reserves(sid:integer, bid:integer, day:date)

- Find the names of sailors who have reserved boat with bid 103

$$\pi_{sname}\left(\sigma_{bid=103}(R_1) \bowtie S_1\right)$$

Joins

- S1 Sailors(sid:integer, sname:string, rating:integer, age:real)
- S2 Boats(bid:integer, bname:string, color:string)
- R1 Reserves(sid:integer, bid:integer, day:date)

- Find the names of sailors who have reserved a red boat

$$\pi_{sname}\left(\sigma_{color='red'}(S_2) \bowtie R_1 \bowtie S_1\right)$$
Joins

• S1 Sailors(sid:integer, sname:string, rating:integer, age:real)
• S2 Boats(bid:integer, bname:string, color:string)
• R1 Reserves(sid:integer, bid:integer, day:date)

• Find the names of sailors who have reserved a boat

\[ \pi_{sname}(R_1 \bowtie S_1) \]

Joins

• S1 Sailors(sid:integer, sname:string, rating:integer, age:real)
• S2 Boats(bid:integer, bname:string, color:string)
• R1 Reserves(sid:integer, bid:integer, day:date)

• Find the names of sailors who have reserved a red or green boat
Joins

- S1 Sailors(sid:integer, sname:string, rating:integer, age:real)
- S2 Boats(bid:integer, bname:string, color:string)
- R1 Reserves(sid:integer, bid:integer, day:date)

- Find the names of sailors who have reserved a red or green boat

\[
\rho(T_{\text{Tempboats}}, (\sigma_{\text{color} = \text{red}} \cup \sigma_{\text{color} = \text{green}} S_2)) \\
\pi_{\text{sname}}(T_{\text{Tempboats}} \bowtie R_1 \bowtie S_1)
\]