Chapter 6: Relational Algebra

Content:

• Relational algebra: querying the relational schema

Next:

• SQL
Relational Algebra

π Projection
σ Selection
ρ Renaming
X Cartesian Product
⊗ Join

∪ Union
∩ Intersection
Relational Algebra

Each relational algebra operator takes a number of relations as an input and outputs one relation.
Projection $\pi$

- Takes one relation as an input.
- Alters the schema of the input relation by only producing the specified attributes $A_1, A_2,.. A_n$ in the output relation.

Example: Find the Name of all professors.
Solution: $\Pi_{\text{Name}}(\text{Professors})$
Selection $\sigma$

- Takes one relation as an input.
- The output relation only contains tuples fulfilling the specified condition.

Example: Find all professors with level ‘C4’.
Solution: $\sigma_{\text{Level}= \text{‘C4’}} (\text{Professors})$

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Professors Table" /></td>
<td><img src="image" alt="Professors Table" /></td>
</tr>
</tbody>
</table>

$\sigma_{\text{Level}= \text{‘C4’}}$ | Professors
Renaming $\rho$

- Takes one relation as an input.
- Outputs the same relation with a different name.

Example: Output all C4 professors in a relation named C4Profs.
Solution: $\rho_{\text{C4Profs}}(\sigma_{\text{Level} = 'C4'}(\text{Professors}))$

\[
\begin{array}{|c|c|c|}
\hline
\text{PersNr} & \text{Name} & \text{Level} \\
\hline
2125 & Sokrates & C4 \\
2126 & Russel & C4 \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|c|}
\hline
\text{PersNr} & \text{Name} & \text{Level} \\
\hline
2125 & Sokrates & C4 \\
2126 & Russel & C4 \\
\hline
\end{array}
\]
Cross/Cartesian Product $X$

- Takes $n$ relations as an input.
- Outputs the cartesian product of the input relations.
- $A \times B = \{(a, b) \mid a \in A, b \in B\}$
Join

- Takes n relations as an input.
- Cross product is very common in combination with a selection, the join is a short cut for that:
- \( \sigma_{\text{given}_\text{by} = \text{persNr}} (\text{Professors} \times \text{Lectures}) = \text{Professors} \bowtie_{\text{given}_\text{by} = \text{persNr}} \text{Lectures} \)

---

**Input**

<table>
<thead>
<tr>
<th>Professors</th>
</tr>
</thead>
<tbody>
<tr>
<td>PersNr</td>
</tr>
<tr>
<td>------------</td>
</tr>
<tr>
<td>2125</td>
</tr>
<tr>
<td>2126</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lectures</th>
</tr>
</thead>
<tbody>
<tr>
<td>LectNo</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>2125</td>
</tr>
<tr>
<td>2126</td>
</tr>
</tbody>
</table>

---

**Output**

<table>
<thead>
<tr>
<th>???</th>
</tr>
</thead>
<tbody>
<tr>
<td>PersNr</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>2125</td>
</tr>
<tr>
<td>2126</td>
</tr>
</tbody>
</table>
Relational Algebra: Example

Find the name all lectures the student named Jonas is attending.

\[ \text{Students}.\text{studNr} = \text{attend}.\text{studNr} \]
\[ \text{attend.lectureNr} = \text{Lectures}.\text{lectureNr} \]
\[ \sigma_{\text{Name} = \text{‘Jonas’}} \text{Students} \]

Students: \{[\text{StudNr}, \text{Name}, \text{Semester}]\}
Lectures: \{[\text{LectureNr}, \text{Title}, \text{WeeklyHours}]\}
attend: \{[\text{StudNr}, \text{LectureNr}]\}
Other Relational Languages

- Relational Algebra
- Relational Tuple Calculus
- Relational Domain Calculus
Relational Tuple Calculus (Example)

Students who attend at least one lecture of Curie

\{ s \mid s \in \text{Students} \\
\quad \land \exists h \in \text{attend}(s.\text{StudNr}=h.\text{StudNr}) \\
\quad \land \exists v \in \text{Lectures}(h.\text{LectureNr}=v.\text{LectureNr}) \\
\quad \land \exists p \in \text{Professors}(p.\text{PersNr}=v.\text{Given_by} \\
\quad \land p.\text{Name} = 'Curie') \}\}
Relational Domain Calculus (Example)

Query in the domain calculus is of the form:

\{[v1, v2, \ldots, vn] \mid P(v1, \ldots, vn)\}

with \(v1, \ldots, v2\) domain variables and \(P\) predicate.

Example:

StudNr and Name of the testees of Sokrates

\{[m, n] \mid \exists ([m, n, s] \in Students \land \exists p, v, g ([m, p, v, g] \in test \land \exists a, r, b ([p, a, r, b] \in Professors \land a = ‘Sokrates’))\}
Expressive Power

The three languages

• Relational Algebra
• Tuple Relational Calculus (restricted to safe expressions)
• Domain Relational Calculus (restricted to safe expressions)

are equal in their expressive power

\{ n \mid \neg (n \in Professors) \} \text{ e.g. is not safe, as the result is infinite}