CPS216: Advanced Database Systems

Notes 02: Query Processing (Overview)

Shivnath Babu
Query Processing

Declarative SQL Query → Query Plan

NOTE: You will not be tested on how well you know SQL. Understanding the SQL introduced in class will be sufficient (a primer follows). SQL is described in Chapter 6, GMUW.

Focus: Relational System (i.e., data is organized as tables, or relations)
SQL Primer

We will focus on SPJ, or Select-Project-Join Queries

Select  <attribute list>
From    <relation list>
Where   <condition list>

Example Filter Query over R(A,B,C):
Select  B
From    R
Where    R.A = “c” ∧ R.C > 10
We will focus on SPJ, or Select-Project-Join-Queries

```
Select  <attribute list>
From    <relation list>
Where   <condition list>
```

Example Join Query over R(A,B,C) and S(C,D,E):
```
Select   B, D
From     R, S
Where    R.A = “c” ∧ S.E = 2 ∧ R.C = S.C
```
<table>
<thead>
<tr>
<th>R</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>S</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1</td>
<td>10</td>
<td></td>
<td>10</td>
<td>x</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>1</td>
<td>20</td>
<td></td>
<td>20</td>
<td>y</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>2</td>
<td>10</td>
<td></td>
<td>30</td>
<td>z</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>d</td>
<td>2</td>
<td>35</td>
<td></td>
<td>40</td>
<td>x</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>3</td>
<td>45</td>
<td></td>
<td>50</td>
<td>y</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Select B,D  
From R,S  
Where R.A = “c”  ∧  
S.E = 2  ∧  R.C=S.C  

Answer | B | D  
| 2 |  x |
How do we execute this query?

Select B,D
From R,S
Where R.A = “c” ∧ S.E = 2 ∧ R.C=S.C

One idea
- Do Cartesian product
- Select tuples
- Do projection
<table>
<thead>
<tr>
<th>R X S</th>
<th>R.A</th>
<th>R.B</th>
<th>R.C</th>
<th>S.C</th>
<th>S.D</th>
<th>S.E</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select B,D From R,S Where R.A = “c” (\wedge) S.E = 2 (\wedge) R.C=S.C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>x</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>1</td>
<td>10</td>
<td>20</td>
<td>y</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

Bingo! Got one...
Relational Algebra - can be used to describe plans

Ex: Plan I

\[ \Pi_{B,D} \left( \sigma_{R.A = "c" \land S.E = 2 \land R.C = S.C} (R \times S) \right) \]
Relational Algebra Primer
(Chapter 5, GMUW)

Select: $\sigma_{R.A=\text{"c"} \land R.C=10}$
Project: $\Pi_{B,D}$
Cartesian Product: $R \times S$
Natural Join: $R \Join S$
Relational Algebra - can be used to describe plans

Ex: Plan I

\[ \Pi_{B,D} \left( \sigma_{R.A="c" \land S.E=2 \land R.C=S.C} (R \times S) \right) \]

OR: \[ \Pi_{B,D} \left[ \sigma_{R.A="c" \land S.E=2 \land R.C=S.C} (R \times S) \right] \]
Another idea:

Plan II

$\Pi_{B,D}$

$\sigma_{R.A = "c"}$

$\sigma_{S.E = 2}$

$R(A,B,C)$

$S(C,D,E)$

Select $B,D$

From $R,S$

Where $R.A = "c" \land S.E = 2 \land R.C = S.C$
Select B,D
From R,S
Where R.A = “c” ∧
S.E = 2 ∧ R.C = S.C
Plan III

Use R.A and S.C Indexes

(1) Use R.A index to select R tuples with R.A = “c”

(2) For each R.C value found, use S.C index to find matching tuples

(3) Eliminate S tuples S.E ≠ 2

(4) Join matching R,S tuples, project B,D attributes, and place in result
The image contains a diagram illustrating a process involving two tables, R and S, and a check function. The table R has columns A, B, and C with rows a, b, c, d, e, and c. The table S has columns C, D, and E with rows containing values for C ranging from 10 to 50, D ranging from x to y, and E ranging from 2 to 3.

The diagram includes nodes labeled I1 and I2 with edges connecting them. The process involves checking for the condition A="c", and outputs a tuple <2,x> for the appropriate row in S.

For instance, for the tuple <c,2,10> from R, it is checked against S, and the output is <2,x>. The next tuple to be considered is <c,7,15> from R.
Overview of Query Processing

- SQL query
  - parse
  - parse tree
  - Query rewriting
    - logical query plan
      - statistics
      - Physical plan generation
        - physical query plan
          - execute
            - result

Query Optimization

Query Execution
Example Query

Select B,D
From R,S
Where R.A = “c”  ∧  R.C=S.C
Example: Parse Tree

Select B,D
From R,S
Where R.A = "c" ∧ R.C=S.C
Along with Parsing …

• Semantic checks
  – Do the projected attributes exist in the relations in the From clause?
  – Ambiguous attributes?
  – Type checking, ex: R.A > 17.5

• Expand views
Query rewriting

parse tree

parse

Initial logical plan

Rewrite rules

“Best” logical plan

Logical plan

Physical plan generation

statistics

physical query plan

execute

result

SQL query

logical query plan
Initial Logical Plan

Relational Algebra: \( \Pi_{B,D} \left[ \sigma_{R.A = "c" \land R.C = S.C} (R \times S) \right] \)
Apply Rewrite Rule (1)

\[\pi_{B,D} \left[ \sigma_{R.A = "c" \land R.C = S.C} (R \times S) \right] \sigma_{R.A = "c"}(R \times S)\]

\[\pi_{B,D} \left[ \sigma_{R.C = S.C} \right]\]

\[\Pi_{B,D} \left[ \sigma_{R.C = S.C} [\sigma_{R.A = "c"}(R \times S)] \right]\]
Apply Rewrite Rule (2)

\[ \pi_{B,D} \exists \sigma_{R.C = S.C} \exists \sigma_{R.A = "c"} \begin{array}{c} X \\ \sigma_{R.C = S.C} \end{array} X \begin{array}{c} R \\ S \end{array} \] \implies \[ \pi_{B,D} \exists \sigma_{R.C = S.C} \exists \sigma_{R.A = "c"}(R) \] 

\[ \Pi_{B,D} [ \sigma_{R.C = S.C} [\sigma_{R.A = "c"}(R)] X S ] \]
Apply Rewrite Rule (3)

\[ \prod_{B,D} \left[ \left[ \sigma_{R.A = "c"}(R) \right] \bowtie S \right] \]

\[ \sigma_{R.C = S.C} \]

\[ \pi_{B,D} \]

Natural join

\[ \sigma_{R.A = "c"}(S) \]

\[ R \]

\[ S \]
Query rewriting

- SQL query
  - parse
    - parse tree
  
  Query rewriting

- statistics
  - logical query plan
    - Physical plan generation
      - physical query plan
        - execute
          - result

- Initial logical plan
  - Logical plan
    - Rewrite rules
      - “Best” logical plan