Chapter 13: Query Optimization
Basic Steps in Query Processing

1. Parsing and translation
2. Optimization
3. Evaluation
Query Optimization

- **Alternative ways** of evaluating a given query
  - Equivalent expressions
    - E.g., $\sigma_{\text{salary}<75000}(\Pi_{\text{salary}}(\text{instructor}))$ is equivalent to $\Pi_{\text{salary}}(\sigma_{\text{salary}<75000}(\text{instructor}))$
  - Different algorithms for each operation
    - E.g., to find instructors with salary < 75000,
      - can use an index on salary,
      - or can perform complete relation scan and discard instructors with salary $\geq$ 75000

- **Query optimization**
  - The process of selecting the most efficient strategies (query evaluation plan) for processing a given query
Equivalent Expression

- Two relational-algebra expressions are equivalent if, on every legal database instance, the two expressions generate the same (multi)set of tuples.
  - Discussion in this chapter is based on the set version of the relation algebra.
    - In SQL, the inputs and outputs are multisets of tuples, and the multiset version of the relational algebra is used for evaluating SQL queries.

- Example
  (a) \( \Pi_{name, title}(\sigma_{dept\_name=\text{Music}}(\text{instructor} \bowtie (\text{teaches} \bowtie \text{course})) ) \)
  (b) \( \Pi_{name, title}(\sigma_{dept\_name=\text{Music}}(\text{instructor}) \bowtie (\text{teaches} \bowtie \text{course}) ) \)
Query Evaluation Plan

- An **evaluation plan** defines exactly what algorithm is used for each operation, and how the execution of the operations is coordinated.

```
Π_{\text{name}, \text{title}} \text{(sort to remove duplicates)}
```

```
\times \text{(hash join)}
```

```
\times \text{(merge join)}
```

```
\sigma_{\text{dept\_name} = \text{Music}} \text{(use index 1)}
```

```
\sigma_{\text{year} = 2009} \text{(use linear scan)}
```

```
instructor
```

```
teaches
```

```
course
```
Cost-Based Query Optimization

- **Cost-based query optimization**
  - Amongst all equivalent evaluation plans choose the one with lowest cost

- Generating query evaluation plan in cost-based query optimization
  1. Generate logically equivalent expressions using equivalence rules
  2. Annotate resultant expressions to get alternative query plans
  3. Choose the cheapest plan based on estimated cost

- **Estimation of plan cost** based on:
  - Statistical information about relations.
    - Examples: number of tuples, number of distinct values for an attribute
  - Statistics estimation for intermediate results
    - to compute cost of complex expressions
  - Cost formulae for algorithms, computed using statistics
Equivalence Rules #1~4

1. Conjunctive selection operations can be deconstructed into a sequence of individual selections.
   \[ \sigma_{\theta_1 \land \theta_2} (E) = \sigma_{\theta_1} (\sigma_{\theta_2} (E)) \]

2. Selection operations are commutative.
   \[ \sigma_{\theta_1} (\sigma_{\theta_2} (E)) = \sigma_{\theta_2} (\sigma_{\theta_1} (E)) \]

3. Only the last in a sequence of projection operations is needed, the others can be omitted.
   \[ \prod_{L_1} (\prod_{L_2} (\ldots (\prod_{L_n} (E))\ldots)) = \prod_{L_1} (E) \]
   - \( L_i \) = lists of attributes

4. Selections can be combined with Cartesian products and theta joins.
   a. \( \sigma_\theta (E_1 \times E_2) = E_1 \bowtie_\theta E_2 \)
   b. \( \sigma_{\theta_1} (E_1 \bowtie_{\theta_2} E_2) = E_1 \bowtie_{\theta_1 \land \theta_2} E_2 \)
5. **Theta-join operations (and natural joins) are commutative.**

\[ E_1 \Join_{\theta} E_2 = E_2 \Join_{\theta} E_1 \]

6. (a) **Natural join operations are associative:**

\[ (E_1 \Join E_2) \Join E_3 = E_1 \Join (E_2 \Join E_3) \]

(b) **Theta joins** are associative in the following manner:

\[ (E_1 \Join_{\theta_1} E_2) \Join_{\theta_2 \land \theta_3} E_3 = E_1 \Join_{\theta_1 \lor \theta_3} (E_2 \Join_{\theta_2} E_3) \]

where \( \theta_2 \) involves attributes from only \( E_2 \) and \( E_3 \).
### Example Relations for Equivalence Rules

#### instructor

<table>
<thead>
<tr>
<th>ID</th>
<th>name</th>
<th>dept_name</th>
<th>salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>10101</td>
<td>Srinivasan</td>
<td>Comp. Sci.</td>
<td>65000</td>
</tr>
<tr>
<td>12121</td>
<td>Wu</td>
<td>Finance</td>
<td>90000</td>
</tr>
<tr>
<td>15151</td>
<td>Mozart</td>
<td>Music</td>
<td>40000</td>
</tr>
<tr>
<td>22222</td>
<td>Einstein</td>
<td>Physics</td>
<td>95000</td>
</tr>
<tr>
<td>32343</td>
<td>El Said</td>
<td>History</td>
<td>60000</td>
</tr>
<tr>
<td>33456</td>
<td>Gold</td>
<td>Physics</td>
<td>87000</td>
</tr>
<tr>
<td>45565</td>
<td>Katz</td>
<td>Comp. Sci.</td>
<td>75000</td>
</tr>
<tr>
<td>58583</td>
<td>Califieri</td>
<td>History</td>
<td>62000</td>
</tr>
<tr>
<td>76543</td>
<td>Singh</td>
<td>Finance</td>
<td>80000</td>
</tr>
<tr>
<td>76766</td>
<td>Crick</td>
<td>Biology</td>
<td>72000</td>
</tr>
<tr>
<td>83821</td>
<td>Brandt</td>
<td>Comp. Sci.</td>
<td>92000</td>
</tr>
<tr>
<td>98345</td>
<td>Kim</td>
<td>Elec. Eng.</td>
<td>80000</td>
</tr>
</tbody>
</table>

#### teaches

<table>
<thead>
<tr>
<th>ID</th>
<th>course_id</th>
<th>sec_id</th>
<th>semester</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>10101</td>
<td>CS-101</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
</tr>
<tr>
<td>10101</td>
<td>CS-315</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>10101</td>
<td>CS-347</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
</tr>
<tr>
<td>12121</td>
<td>FIN-201</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>15151</td>
<td>MU-199</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>22222</td>
<td>PHY-101</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
</tr>
<tr>
<td>32343</td>
<td>HIS-351</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>45565</td>
<td>CS-101</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>45565</td>
<td>CS-319</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>76766</td>
<td>BIO-101</td>
<td>1</td>
<td>Summer</td>
<td>2009</td>
</tr>
<tr>
<td>76766</td>
<td>BIO-301</td>
<td>1</td>
<td>Summer</td>
<td>2010</td>
</tr>
<tr>
<td>83821</td>
<td>CS-190</td>
<td>1</td>
<td>Spring</td>
<td>2009</td>
</tr>
<tr>
<td>83821</td>
<td>CS-190</td>
<td>2</td>
<td>Spring</td>
<td>2009</td>
</tr>
<tr>
<td>83821</td>
<td>CS-319</td>
<td>2</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>98345</td>
<td>EE-181</td>
<td>1</td>
<td>Spring</td>
<td>2009</td>
</tr>
</tbody>
</table>

#### course

<table>
<thead>
<tr>
<th>course_id</th>
<th>title</th>
<th>dept_name</th>
<th>credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIO-101</td>
<td>Intro. to Biology</td>
<td>Biology</td>
<td>4</td>
</tr>
<tr>
<td>BIO-301</td>
<td>Genetics</td>
<td>Biology</td>
<td>4</td>
</tr>
<tr>
<td>BIO-399</td>
<td>Computational Biology</td>
<td>Biology</td>
<td>3</td>
</tr>
<tr>
<td>CS-101</td>
<td>Intro. to Computer Science</td>
<td>Comp. Sci.</td>
<td>4</td>
</tr>
<tr>
<td>CS-190</td>
<td>Game Design</td>
<td>Comp. Sci.</td>
<td>4</td>
</tr>
<tr>
<td>CS-315</td>
<td>Robotics</td>
<td>Comp. Sci.</td>
<td>3</td>
</tr>
<tr>
<td>CS-319</td>
<td>Image Processing</td>
<td>Comp. Sci.</td>
<td>3</td>
</tr>
<tr>
<td>CS-347</td>
<td>Database System Concepts</td>
<td>Comp. Sci.</td>
<td>3</td>
</tr>
<tr>
<td>EE-181</td>
<td>Intro. to Digital Systems</td>
<td>Elec. Eng.</td>
<td>3</td>
</tr>
<tr>
<td>FIN-201</td>
<td>Investment Banking</td>
<td>Finance</td>
<td>3</td>
</tr>
<tr>
<td>HIS-351</td>
<td>World History</td>
<td>History</td>
<td>3</td>
</tr>
<tr>
<td>MU-199</td>
<td>Music Video Production</td>
<td>Music</td>
<td>3</td>
</tr>
<tr>
<td>PHY-101</td>
<td>Physical Principles</td>
<td>Physics</td>
<td>4</td>
</tr>
</tbody>
</table>
Example for Equivalence Rule #6

- Example: \((instructor \bowtie teaches) \bowtie course\)

<table>
<thead>
<tr>
<th>ID</th>
<th>name</th>
<th>dept_name</th>
<th>salary</th>
<th>course_id</th>
<th>sec_id</th>
<th>semester</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>10101</td>
<td>Srinivasan</td>
<td>Comp. Sci.</td>
<td>65000</td>
<td>CS-101</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
</tr>
<tr>
<td>10101</td>
<td>Srinivasan</td>
<td>Comp. Sci.</td>
<td>65000</td>
<td>CS-315</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>10101</td>
<td>Srinivasan</td>
<td>Comp. Sci.</td>
<td>65000</td>
<td>CS-347</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
</tr>
<tr>
<td>12121</td>
<td>Wu</td>
<td>Finance</td>
<td>90000</td>
<td>FIN-201</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>15151</td>
<td>Mozart</td>
<td>Music</td>
<td>40000</td>
<td>MU-199</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>22222</td>
<td>Einstein</td>
<td>Physics</td>
<td>95000</td>
<td>PHY-101</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
</tr>
<tr>
<td>32343</td>
<td>El Said</td>
<td>History</td>
<td>60000</td>
<td>HIS-351</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>45565</td>
<td>Katz</td>
<td>Comp. Sci.</td>
<td>75000</td>
<td>CS-101</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>45565</td>
<td>Katz</td>
<td>Comp. Sci.</td>
<td>75000</td>
<td>CS-319</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>76766</td>
<td>Crick</td>
<td>Biology</td>
<td>72000</td>
<td>BIO-101</td>
<td>1</td>
<td>Summer</td>
<td>2009</td>
</tr>
<tr>
<td>76766</td>
<td>Crick</td>
<td>Biology</td>
<td>72000</td>
<td>BIO-301</td>
<td>1</td>
<td>Summer</td>
<td>2010</td>
</tr>
<tr>
<td>83821</td>
<td>Brandt</td>
<td>Comp. Sci.</td>
<td>92000</td>
<td>CS-190</td>
<td>1</td>
<td>Spring</td>
<td>2009</td>
</tr>
<tr>
<td>83821</td>
<td>Brandt</td>
<td>Comp. Sci.</td>
<td>92000</td>
<td>CS-190</td>
<td>2</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>83821</td>
<td>Brandt</td>
<td>Comp. Sci.</td>
<td>92000</td>
<td>CS-319</td>
<td>2</td>
<td>Spring</td>
<td>2010</td>
</tr>
<tr>
<td>98345</td>
<td>Kim</td>
<td>Elec. Eng.</td>
<td>80000</td>
<td>EE-181</td>
<td>1</td>
<td>Spring</td>
<td>2009</td>
</tr>
</tbody>
</table>

- \((instructor \bowtie teaches) \bowtie course\)
Example for Equivalence Rule #6 (Cont.)

Example: instructor $\Join (\text{teaches} \times \text{course})$

<table>
<thead>
<tr>
<th>ID</th>
<th>course_id</th>
<th>sec_id</th>
<th>semester</th>
<th>year</th>
<th>title</th>
<th>dept_name</th>
<th>credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>10101</td>
<td>CS-101</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
<td>Intro. to Computer Science</td>
<td>Comp. Sci.</td>
<td>4</td>
</tr>
<tr>
<td>10101</td>
<td>CS-315</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Robotics</td>
<td>Comp. Sci.</td>
<td>3</td>
</tr>
<tr>
<td>10101</td>
<td>CS-347</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
<td>Database System Concepts</td>
<td>Comp. Sci.</td>
<td>3</td>
</tr>
<tr>
<td>12121</td>
<td>FIN-201</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Investment Banking</td>
<td>Finance</td>
<td>3</td>
</tr>
<tr>
<td>15151</td>
<td>MU-199</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Music Video Production</td>
<td>Music</td>
<td>3</td>
</tr>
<tr>
<td>22222</td>
<td>PHY-101</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
<td>Physical Principles</td>
<td>Physics</td>
<td>4</td>
</tr>
<tr>
<td>32343</td>
<td>HIS-351</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>World History</td>
<td>History</td>
<td>3</td>
</tr>
<tr>
<td>45565</td>
<td>CS-101</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Intro. to Computer Science</td>
<td>Comp. Sci.</td>
<td>4</td>
</tr>
<tr>
<td>45565</td>
<td>CS-319</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Image Processing</td>
<td>Comp. Sci.</td>
<td>3</td>
</tr>
<tr>
<td>76766</td>
<td>BIO-101</td>
<td>1</td>
<td>Summer</td>
<td>2009</td>
<td>Intro. to Biology</td>
<td>Biology</td>
<td>4</td>
</tr>
<tr>
<td>76766</td>
<td>BIO-301</td>
<td>1</td>
<td>Summer</td>
<td>2010</td>
<td>Genetics</td>
<td>Biology</td>
<td>4</td>
</tr>
<tr>
<td>83882</td>
<td>CS-190</td>
<td>1</td>
<td>Spring</td>
<td>2009</td>
<td>Game Design</td>
<td>Comp. Sci.</td>
<td>4</td>
</tr>
<tr>
<td>83882</td>
<td>CS-190</td>
<td>2</td>
<td>Spring</td>
<td>2009</td>
<td>Game Design</td>
<td>Comp. Sci.</td>
<td>4</td>
</tr>
<tr>
<td>83882</td>
<td>CS-319</td>
<td>2</td>
<td>Spring</td>
<td>2010</td>
<td>Image Processing</td>
<td>Comp. Sci.</td>
<td>3</td>
</tr>
<tr>
<td>98345</td>
<td>EE-181</td>
<td>1</td>
<td>Spring</td>
<td>2009</td>
<td>Intro. to Digital Systems</td>
<td>Elec. Eng.</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>name</th>
<th>dept_name</th>
<th>salary</th>
<th>course_id</th>
<th>sec_id</th>
<th>semester</th>
<th>year</th>
<th>title</th>
<th>credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>10101</td>
<td>Srinivasan</td>
<td>Comp. Sci.</td>
<td>65000</td>
<td>CS-101</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
<td>Intro. to Computer Science</td>
<td>4</td>
</tr>
<tr>
<td>10101</td>
<td>Srinivasan</td>
<td>Comp. Sci.</td>
<td>65000</td>
<td>CS-315</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Robotics</td>
<td>3</td>
</tr>
<tr>
<td>10101</td>
<td>Srinivasan</td>
<td>Comp. Sci.</td>
<td>65000</td>
<td>CS-347</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
<td>Database System Concepts</td>
<td>3</td>
</tr>
<tr>
<td>12121</td>
<td>Wu</td>
<td>Finance</td>
<td>90000</td>
<td>FIN-201</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Investment Banking</td>
<td>3</td>
</tr>
<tr>
<td>15151</td>
<td>Mozart</td>
<td>Music</td>
<td>40000</td>
<td>MU-199</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Music Video Production</td>
<td>3</td>
</tr>
<tr>
<td>22222</td>
<td>Einstein</td>
<td>Physics</td>
<td>95000</td>
<td>PHY-101</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
<td>Physical Principles</td>
<td>4</td>
</tr>
<tr>
<td>32343</td>
<td>El Said</td>
<td>History</td>
<td>60000</td>
<td>HIS-351</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>World History</td>
<td>3</td>
</tr>
<tr>
<td>45565</td>
<td>Katz</td>
<td>Comp. Sci.</td>
<td>75000</td>
<td>CS-101</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Intro. to Computer Science</td>
<td>4</td>
</tr>
<tr>
<td>45565</td>
<td>Katz</td>
<td>Comp. Sci.</td>
<td>75000</td>
<td>CS-319</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Image Processing</td>
<td>3</td>
</tr>
<tr>
<td>76766</td>
<td>Crick</td>
<td>Biology</td>
<td>72000</td>
<td>BIO-101</td>
<td>1</td>
<td>Summer</td>
<td>2009</td>
<td>Intro. to Biology</td>
<td>4</td>
</tr>
<tr>
<td>76766</td>
<td>Crick</td>
<td>Biology</td>
<td>72000</td>
<td>BIO-301</td>
<td>1</td>
<td>Summer</td>
<td>2010</td>
<td>Genetics</td>
<td>4</td>
</tr>
<tr>
<td>83882</td>
<td>Brandt</td>
<td>Comp. Sci.</td>
<td>92000</td>
<td>CS-190</td>
<td>1</td>
<td>Spring</td>
<td>2009</td>
<td>Game Design</td>
<td>4</td>
</tr>
<tr>
<td>83882</td>
<td>Brandt</td>
<td>Comp. Sci.</td>
<td>92000</td>
<td>CS-190</td>
<td>2</td>
<td>Spring</td>
<td>2009</td>
<td>Game Design</td>
<td>4</td>
</tr>
<tr>
<td>83882</td>
<td>Brandt</td>
<td>Comp. Sci.</td>
<td>92000</td>
<td>CS-319</td>
<td>2</td>
<td>Spring</td>
<td>2010</td>
<td>Image Processing</td>
<td>3</td>
</tr>
<tr>
<td>98345</td>
<td>Kim</td>
<td>Elec. Eng.</td>
<td>80000</td>
<td>EE-181</td>
<td>1</td>
<td>Spring</td>
<td>2009</td>
<td>Intro. to Digital Systems</td>
<td>3</td>
</tr>
</tbody>
</table>
Equivalence Rules #7

7. The selection operation distributes over the theta join operation under the following two conditions:

(a) When all the attributes in $\theta_0$ involve only the attributes of one of the expressions ($E_1$) being joined.

$$\sigma_{\theta_0}(E_1 \bowtie_{\theta} E_2) = (\sigma_{\theta_0}(E_1)) \bowtie_{\theta} E_2$$

(b) When $\theta_1$ involves only the attributes of $E_1$ and $\theta_2$ involves only the attributes of $E_2$.

$$\sigma_{\theta_1 \land \theta_2}(E_1 \bowtie_{\theta} E_2) = (\sigma_{\theta_1}(E_1)) \bowtie_{\theta} (\sigma_{\theta_2}(E_2))$$
Example for Equivalence Rule #7

Example: \( \sigma_{dept\_name=\text{"Music"}} (\text{instructor} \times (\text{teaches} \times \text{course})) \)

<table>
<thead>
<tr>
<th>ID</th>
<th>course_id</th>
<th>sec_id</th>
<th>semester</th>
<th>year</th>
<th>title</th>
<th>dept_name</th>
<th>credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>10101</td>
<td>CS-101</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
<td>Intro. to Computer Science</td>
<td>Comp. Sci.</td>
<td>4</td>
</tr>
<tr>
<td>10101</td>
<td>CS-315</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Robotics</td>
<td>Comp. Sci.</td>
<td>3</td>
</tr>
<tr>
<td>10101</td>
<td>CS-347</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
<td>Database System Concepts</td>
<td>Comp. Sci.</td>
<td>3</td>
</tr>
<tr>
<td>12121</td>
<td>FIN-201</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Investment Banking</td>
<td>Finance</td>
<td>3</td>
</tr>
<tr>
<td>15151</td>
<td>MU-199</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Music Video Production</td>
<td>Music</td>
<td>3</td>
</tr>
<tr>
<td>22222</td>
<td>PHY-101</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
<td>Physical Principles</td>
<td>Physics</td>
<td>4</td>
</tr>
<tr>
<td>32343</td>
<td>HIS-351</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>World History</td>
<td>History</td>
<td>3</td>
</tr>
<tr>
<td>45565</td>
<td>CS-101</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Intro. to Computer Science</td>
<td>Comp. Sci.</td>
<td>4</td>
</tr>
<tr>
<td>45565</td>
<td>CS-319</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Image Processing</td>
<td>Comp. Sci.</td>
<td>3</td>
</tr>
<tr>
<td>76766</td>
<td>BIO-101</td>
<td>1</td>
<td>Summer</td>
<td>2009</td>
<td>Intro. to Biology</td>
<td>Biology</td>
<td>4</td>
</tr>
<tr>
<td>76766</td>
<td>BIO-301</td>
<td>1</td>
<td>Summer</td>
<td>2010</td>
<td>Genetics</td>
<td>Biology</td>
<td>4</td>
</tr>
<tr>
<td>83821</td>
<td>CS-190</td>
<td>2</td>
<td>Spring</td>
<td>2009</td>
<td>Game Design</td>
<td>Comp. Sci.</td>
<td>4</td>
</tr>
<tr>
<td>83821</td>
<td>CS-190</td>
<td>2</td>
<td>Spring</td>
<td>2009</td>
<td>Game Design</td>
<td>Comp. Sci.</td>
<td>4</td>
</tr>
<tr>
<td>98345</td>
<td>EE-181</td>
<td>1</td>
<td>Spring</td>
<td>2009</td>
<td>Intro. to Digital Systems</td>
<td>Elec. Eng.</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>name</th>
<th>dept_name</th>
<th>salary</th>
<th>course_id</th>
<th>sec_id</th>
<th>semester</th>
<th>year</th>
<th>title</th>
<th>credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>10101</td>
<td>Srinivasan</td>
<td>Comp. Sci.</td>
<td>65000</td>
<td>CS-101</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
<td>Intro. to Computer Science</td>
<td>4</td>
</tr>
<tr>
<td>10101</td>
<td>Srinivasan</td>
<td>Comp. Sci.</td>
<td>65000</td>
<td>CS-315</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Robotics</td>
<td>3</td>
</tr>
<tr>
<td>10101</td>
<td>Srinivasan</td>
<td>Comp. Sci.</td>
<td>65000</td>
<td>CS-347</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
<td>Database System Concepts</td>
<td>3</td>
</tr>
<tr>
<td>12121</td>
<td>Wu</td>
<td>Finance</td>
<td>90000</td>
<td>FIN-201</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Investment Banking</td>
<td>3</td>
</tr>
<tr>
<td>15151</td>
<td>Mozart</td>
<td>Music</td>
<td>40000</td>
<td>MU-199</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Music Video Production</td>
<td>3</td>
</tr>
<tr>
<td>22222</td>
<td>Einstein</td>
<td>Physics</td>
<td>95000</td>
<td>PHY-101</td>
<td>1</td>
<td>Fall</td>
<td>2009</td>
<td>Physical Principles</td>
<td>4</td>
</tr>
<tr>
<td>32343</td>
<td>El Said</td>
<td>History</td>
<td>60000</td>
<td>HIS-351</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>World History</td>
<td>3</td>
</tr>
<tr>
<td>45565</td>
<td>Katz</td>
<td>Comp. Sci.</td>
<td>75000</td>
<td>CS-101</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Intro. to Computer Science</td>
<td>4</td>
</tr>
<tr>
<td>45565</td>
<td>Katz</td>
<td>Comp. Sci.</td>
<td>75000</td>
<td>CS-319</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Image Processing</td>
<td>3</td>
</tr>
<tr>
<td>76766</td>
<td>Crick</td>
<td>Biology</td>
<td>72000</td>
<td>BIO-101</td>
<td>1</td>
<td>Summer</td>
<td>2009</td>
<td>Intro. to Biology</td>
<td>4</td>
</tr>
<tr>
<td>76766</td>
<td>Crick</td>
<td>Biology</td>
<td>72000</td>
<td>BIO-301</td>
<td>1</td>
<td>Summer</td>
<td>2010</td>
<td>Genetics</td>
<td>4</td>
</tr>
<tr>
<td>83821</td>
<td>Brandt</td>
<td>Comp. Sci.</td>
<td>92000</td>
<td>CS-190</td>
<td>1</td>
<td>Spring</td>
<td>2009</td>
<td>Game Design</td>
<td>4</td>
</tr>
<tr>
<td>83821</td>
<td>Brandt</td>
<td>Comp. Sci.</td>
<td>92000</td>
<td>CS-190</td>
<td>2</td>
<td>Spring</td>
<td>2009</td>
<td>Game Design</td>
<td>4</td>
</tr>
<tr>
<td>83821</td>
<td>Brandt</td>
<td>Comp. Sci.</td>
<td>92000</td>
<td>CS-319</td>
<td>2</td>
<td>Spring</td>
<td>2010</td>
<td>Image Processing</td>
<td>3</td>
</tr>
<tr>
<td>98345</td>
<td>Kim</td>
<td>Elec. Eng.</td>
<td>80000</td>
<td>EE-181</td>
<td>1</td>
<td>Spring</td>
<td>2009</td>
<td>Intro. to Digital Systems</td>
<td>3</td>
</tr>
</tbody>
</table>
Example for Equivalence Rule #7 (Cont.)

Example: \((\sigma_{\text{dept\_name= "Music"}}(\text{instructor})) \bowtie (\text{teaches } \bowtie \text{ course})\)

\[ \sigma_{\text{dept\_name= "Music"}}(\text{instructor}) \]

\[(\text{teaches } \bowtie \text{ course})\]

\[(\sigma_{\text{dept\_name= "Music"}}(\text{instructor})) \bowtie (\text{teaches } \bowtie \text{ course})\]

---

Database System Concepts - 6th Edition

©Silberschatz, Korth and Sudarshan
Equivalence Rules #8

8. The projection operation distributes over the theta join operation as follows:

(a) if \( \theta \) involves only attributes from \( L_1 \cup L_2 \):
\[
\Pi_{L_1 \cup L_2} (E_1 \Join_{\theta} E_2) = (\Pi_{L_1} (E_1)) \Join_{\theta} (\Pi_{L_2} (E_2))
\]

(b) Consider a join \( E_1 \Join_{\theta} E_2 \).

- Let \( L_1 \) and \( L_2 \) be sets of attributes from \( E_1 \) and \( E_2 \), respectively.
- Let \( L_3 \) be attributes of \( E_1 \) that are involved in join condition \( \theta \), but are not in \( L_1 \cup L_2 \), and
- let \( L_4 \) be attributes of \( E_2 \) that are involved in join condition \( \theta \), but are not in \( L_1 \cup L_2 \).

\[
\Pi_{L_1 \cup L_2} (E_1 \Join_{\theta} E_2) = \Pi_{L_1 \cup L_2} ((\Pi_{L_1 \cup L_3} (E_1)) \Join_{\theta} (\Pi_{L_2 \cup L_4} (E_2)))
\]
Example: $\Pi_{name, title}(\sigma_{dept\_name= "Music"}(instructor) \bowtie teaches) \bowtie course$

$\sigma_{dept\_name= "Music"}(instructor)$

<table>
<thead>
<tr>
<th>ID</th>
<th>name</th>
<th>dept_name</th>
<th>salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>15151</td>
<td>Mozart</td>
<td>Music</td>
<td>40000</td>
</tr>
</tbody>
</table>

$\sigma_{dept\_name= "Music"}(instructor) \bowtie teaches$

<table>
<thead>
<tr>
<th>ID</th>
<th>name</th>
<th>dept_name</th>
<th>salary</th>
<th>course_id</th>
<th>sec_id</th>
<th>semester</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>15151</td>
<td>Mozart</td>
<td>Music</td>
<td>40000</td>
<td>MU-199</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
</tr>
</tbody>
</table>

$(\sigma_{dept\_name= "Music"}(instructor) \bowtie teaches) \bowtie course$

<table>
<thead>
<tr>
<th>ID</th>
<th>name</th>
<th>dept_name</th>
<th>salary</th>
<th>course_id</th>
<th>sec_id</th>
<th>semester</th>
<th>year</th>
<th>title</th>
<th>credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>15151</td>
<td>Mozart</td>
<td>Music</td>
<td>40000</td>
<td>MU-199</td>
<td>1</td>
<td>Spring</td>
<td>2010</td>
<td>Music Video Production</td>
<td>3</td>
</tr>
</tbody>
</table>

$\Pi_{name, title}(\sigma_{dept\_name= "Music"}(instructor) \bowtie teaches) \bowtie course$
Example for Equivalence Rule #8 (Cont.)

Example: \( \Pi_{\text{name, title}}((\Pi_{\text{name, course\_id}}(\sigma_{\text{dept\_name}=\text{“Music”}}(\text{instructor})) \bowtie\text{teaches}) \bowtie \Pi_{\text{course\_id, title}}(\text{course})) \)

\[\]

\( \sigma_{\text{dept\_name}=\text{“Music”}}(\text{instructor}) \)

\( (\sigma_{\text{dept\_name}=\text{“Music”}}(\text{instructor})) \bowtie \text{teaches} \)

\( \Pi_{\text{name, course\_id}}(\sigma_{\text{dept\_name}=\text{“Music”}}(\text{instructor})) \bowtie \text{teaches} \)

\( \Pi_{\text{course\_id, title}}(\text{course}) \)
Equivalence Rules for Set Operations

9. The set operations **union and intersection** are commutative

\[
E_1 \cup E_2 = E_2 \cup E_1 \\
E_1 \cap E_2 = E_2 \cap E_1
\]

(set difference is not commutative).

10. Set **union and intersection** are associative.

\[
(E_1 \cup E_2) \cup E_3 = E_1 \cup (E_2 \cup E_3) \\
(E_1 \cap E_2) \cap E_3 = E_1 \cap (E_2 \cap E_3)
\]

11. The **selection** operation distributes over \( \cup, \cap \) and \(-\).

\[
\sigma_\theta (E_1 - E_2) = \sigma_\theta (E_1) - \sigma_\theta(E_2)
\]

and similarly for \( \cup \) and \( \cap \) in place of \(-\).

Also:
\[
\sigma_\theta (E_1 - E_2) = \sigma_\theta(E_1) - E_2
\]

and similarly for \( \cap \) in place of \(-\), but not for \( \cup \).

12. The **projection** operation distributes over **union**

\[
\Pi_L(E_1 \cup E_2) = (\Pi_L(E_1)) \cup (\Pi_L(E_2))
\]
Transformation Example: Pushing Selections

- **Performing the selection as early as possible** reduces the size of the relation to be joined.

- **Example query:** Find the names of all instructors in the Music department, along with the titles of the courses that they teach.

  \[
  \pi_{name, title}(\sigma_{dept\_name=\text{``Music''}}(\text{instructor} \bowtie (\text{teaches} \bowtie \pi_{course\_id, title}(\text{course}))))
  \]

  - **Transformation using rule 7a**

  \[
  \pi_{name, title}((\sigma_{dept\_name=\text{``Music''}}(\text{instructor})) \bowtie (\text{teaches} \bowtie \pi_{course\_id, title}(\text{course})))
  \]
Transformation Example: Pushing Projections

- Performing the projection as early as possible reduces the size of the relation to be joined

- Example query:

\[
\Pi_{\text{name, title}}\left(\left(\sigma_{\text{dept\_name}=\text{"Music"}}\left(\text{instructor}\Join\text{teaches}\right)\Join\Pi_{\text{course\_id, title}}\left(\text{course}\right)\right)\right)
\]

- When we compute \(\left(\sigma_{\text{dept\_name}=\text{"Music"}}\left(\text{instructor}\Join\text{teaches}\right)\right)\), we obtain a relation whose schema is:

\((ID, \text{name, dept\_name, salary, course\_id, sec\_id, semester, year})\)

- Push projections using equivalence rules 8a and 8b; eliminate unneeded attributes from intermediate results to get:

\[
\Pi_{\text{name, title}}\left(\left(\Pi_{\text{name, course\_id}}\left(\sigma_{\text{dept\_name}=\text{"Music"}}\left(\text{instructor}\Join\text{teaches}\right)\Join\Pi_{\text{course\_id, title}}\left(\text{course}\right)\right)\right)\right)
\]
Example with Multiple Transformations

- Query: Find the names of all instructors in the Music department who have taught a course in 2009, along with the titles of the courses that they taught

\[ \Pi_{name, title}(\sigma_{dept\_name= "Music" \land year=2009}(instructor \Join (teaches \Join \Pi_{course\_id, title}(course)))) \]

- Transformation using \textit{join associatively} (Rule 6a):

\[ \Pi_{name, title}(\sigma_{dept\_name= "Music" \land year=2009}((instructor \Join teaches) \Join \Pi_{course\_id, title}(course))) \]

- Second form provides an opportunity to apply the “perform selections early” rule

\[ \Pi_{name, title}((\sigma_{dept\_name = "Music"}(instructor) \Join \sigma_{year = 2009}(teaches)) \Join \Pi_{course\_id, title}(course)) \]
Multiple Transformations (Cont.)

(a) Initial expression tree

(b) Tree after multiple transformations
Join Ordering

- For all relations \( r_1, r_2, \) and \( r_3 \), \((r_1 \Join r_2) \Join r_3 = r_1 \Join (r_2 \Join r_3)\) (Rule 6a)

- Choose the expression that will yield smaller temporary result
  - If \( r_2 \Join r_3 \) is quite large and \( r_1 \Join r_2 \) is small, we choose \( (r_1 \Join r_2) \Join r_3 \)
    so that we compute and store a smaller temporary relation

- Example

  \[
  \Pi_{name, title}
  ((\sigma_{dept\_name= \text{ "Music"}} (instructor) \Join teaches) \Join \Pi_{course\_id, title} (course))
  \]

  - Which join expression is it better to compute first?
    1. Compute \( teaches \Join \Pi_{course\_id, title} (course) \) first,
       and join result with \( \sigma_{dept\_name= \text{ "Music"}} (instructor) \)
       - The result of the first join is likely to be a large relation
    2. Compute \( \sigma_{dept\_name= \text{ "Music"}} (instructor) \Join teaches \) first
       - Only a small fraction of the university’s instructors are likely to be from the Music department – This would be better
Enumeration of Equivalent Expressions

- Query optimizers use equivalence rules to systematically generate expressions equivalent to the given expression.
- Can generate all equivalent expressions as follows:
  - Repeat
    - apply all applicable equivalence rules on every subexpression of every equivalent expression found so far
    - add newly generated expressions to the set of equivalent expressions
  Until no new equivalent expressions are generated above
- The above approach is very expensive in space and time
  - Two approaches
    - Optimized plan generation based on transformation rules – avoid examining some of the expressions by considering the estimated cost
    - Heuristic-based transformation: special case approach for queries with only selections, projections and joins
Cost Estimation

- Cost of each operator computer as described in Chapter 12
  - Need statistics of input relations
    - E.g. number of tuples, sizes of tuples
- Inputs can be results of sub-expressions
  - Need to estimate statistics of expression results
  - To do so, we require additional statistics
    - E.g. number of distinct values for an attribute
Statistical Information for Cost Estimation

- $n_r$: number of tuples in a relation $r$
- $b_r$: number of blocks containing tuples of $r$
  - $b_r = \lceil n_r / f_r \rceil$, if tuples of $r$ are stored together physically in a file
- $l_r$: size of a tuple of $r$
- $f_r$: blocking factor of $r$ — i.e., the number of tuples of $r$ that fit into one block

- $V(A, r)$: number of distinct values that appear in $r$ for attribute $A$ ($= \text{size of } \Pi_A(r)$)
- $SC(A, r)$: selection cardinality of attribute $A$ of relation $r$
  - Average number of records that satisfy equality on $A$

- $f_i$: average fan-out of internal nodes of index $i$, for $B^+$-trees
- $HT_i$: number of levels in index $i$ (i.e., the height of $i$) & on attribute $A$ of relation $r$
  - For a $B^+$-tree index, $HT_i = \lceil \log_{f_i}(V(A, r)) \rceil$
  - For a hash index, $HT_i = 1$
- $LB_i$: number of lowest-level index blocks in $i$ (i.e., the # of blocks at the leaf level)
Histograms

- Histogram on attribute *age* of relation *person*

- **Equi-width** histograms – the size of each range is equal
- **Equi-depth** histograms – each range has the same number of values
Selection Size Estimation

- **Equality selection** \( \sigma_{A=v}(r) \)
  - \( SC(A, r) \): number of records that will satisfy the selection
    - = 1, if A is a key attribute
    - = \( n_r / V(A, r) \), otherwise

- \( \sigma_{A \leq v}(r) \) (case of \( \sigma_{A \geq v}(r) \) is symmetric)
  - Let \( c \) denote the estimated number of tuples satisfying the condition
  - If \( \min(A, r) \) and \( \max(A, r) \) are available in catalog
    - \( c = 0 \) if \( v < \min(A, r) \)
    - \( c = n_r \cdot \frac{v - \min(A, r)}{\max(A, r) - \min(A, r)} \)
  - If histograms available, can refine above estimate
  - In absence of statistical information \( c \) is assumed to be \( n_r / 2 \)
Size Estimation of Complex Selections

- **Selectivity** of a condition $\theta_i$: the probability that a tuple in the relation $r$ satisfies $\theta_i$
  - If $s_i$ is the number of satisfying tuples in $r$, the selectivity of $\theta_i$ is given by $s_i/n_r$

- **Conjunction:** $\sigma_{\theta_1 \land \theta_2 \land \ldots \land \theta_n}(r)$.
  Assuming independence, estimate of tuples in the result is:
  $$n_r \cdot \frac{s_1 \cdot s_2 \cdot \ldots \cdot s_n}{n_r^n}$$

- **Disjunction:** $\sigma_{\theta_1 \lor \theta_2 \lor \ldots \lor \theta_n}(r)$.
  Estimated number of tuples:
  $$n_r \cdot \left(1 - \left(1 - \frac{s_1}{n_r}\right) \cdot \left(1 - \frac{s_2}{n_r}\right) \cdot \ldots \cdot \left(1 - \frac{s_n}{n_r}\right)\right)$$

- **Negation:** $\sigma_{\neg \theta}(r)$.
  Estimated number of tuples: $n_r - \text{size}(\sigma_{\theta}(r))$
Join Operation: Running Example

Running example: student \( \bowtie \) takes

Catalog information for join examples:

- \( n_{\text{student}} = 5,000 \)
- \( f_{\text{student}} = 50 \), which implies that \( b_{\text{student}} = 5000/50 = 100 \)
- \( n_{\text{takes}} = 10,000 \)
- \( f_{\text{takes}} = 25 \), which implies that \( b_{\text{takes}} = 10000/25 = 400 \)
- \( V(ID, \text{takes}) = 2500 \), which implies that on average, each student who has taken a course has taken 4 courses.
  - Attribute \( ID \) in \( \text{takes} \) is a foreign key referencing \( \text{student} \).
- \( V(ID, \text{student}) = 5000 \) (primary key!)
Join Size Estimation

- If $R \cap S = \emptyset$, $r \bowtie s = r \times s$
  - $r \times s$ contains $n_r \cdot n_s$ tuples
  - Each tuple occupies $s_r + s_s$ bytes

- If $R \cap S$ is a key for $R$,
  - A tuple of $s$ will join with at most one tuple from $r$
    - The number of tuples in $r \bowtie s$ is no greater than the number of tuples in $s$
  - If $R \cap S$ is a foreign key in $S$ referencing $R$, then the number of tuples in $r \bowtie s$ is exactly the same as the number of tuples in $s$.

- In the example query $\text{student} \bowtie \text{takes}$,
  - $ID$ in $\text{takes}$ is a foreign key referencing $\text{student}$
  - hence, the result has exactly $n_{\text{takes}}$ tuples, which is 10,000
Estimation of the Size of Joins (Cont.)

- If \( R \cap S = \{A\} \) is not a key for \( R \) or \( S \),
  - If we assume that every tuple \( t \) in \( r \) produces tuples in \( r \bowtie s \),
    the number of tuples in \( r \bowtie s \) is estimated to be:
      \[
      \frac{n_r \times n_s}{V(A,s)}
      \]
  - If the reverse is true, the estimate obtained will be:
    \[
    \frac{n_r \times n_s}{V(A,r)}
    \]
  - The lower of these two estimates is probably the more accurate one

- Can improve on above if histograms are available
  - Use formula similar to above, for each cell of histograms on the two relations

- Example: \textit{students} \( \bowtie \) \textit{takes} without using information about foreign keys
  - \( V(ID, takes) = 2500 \), and \( V(ID, student) = 5000 \)
  - The two estimates are
    \[
    5000 \times \frac{10000}{2500} = 20,000 \]
    \[
    and \ 5000 \times \frac{10000}{5000} = 10,000
    \]
Size Estimation for Other Operations

- **Projection**: estimated size of $\Pi_A(r) = V(A, r)$
  - Projection eliminates duplicates

- **Aggregation**: estimated size of $A g_F(r) = V(A, r)$
  - There is one tuple for each distinct value of A

- **Set operations**
  - For operations on different relations:
    - estimated size of $r \cup s = \text{size of } r + \text{size of } s$
    - estimated size of $r \cap s = \text{minimum size of } r \text{ and size of } s$
    - estimated size of $r - s = r$
  - All the three estimates may be quite inaccurate, but provide upper bounds on the sizes

  - For unions/intersections of selections on the same relation: rewrite and use size estimate for selections
    - E.g. $\sigma_{01}(r) \cup \sigma_{02}(r)$ can be rewritten as $\sigma_{01} \sigma_{02}(r)$
Estimation of Number of Distinct Values

Selections: \( \sigma_\theta (r) \)

- If \( \theta \) forces \( A \) to take a specified value: \( V(A, \sigma_\theta (r)) = 1 \)
  - e.g., \( A = 3 \)

- If \( \theta \) forces \( A \) to take on one of a specified set of values:
  
  \[ V(A, \sigma_\theta (r)) = \text{number of specified values} \]
  
  - e.g., \( (A = 1 \lor A = 3 \lor A = 4) \)

- If the selection condition \( \theta \) is of the form \( A \ op \ r \)
  
  estimated \( V(A, \sigma_\theta (r)) = V(A, r) \times s \), where \( s \) is the selectivity of the selection

- In all the other cases: use approximate estimate of
  
  \[ \min(V(A, r), n_{\sigma_\theta (r)}) \]
  
  - More accurate estimate can be got using probability theory, but this one works fine generally
Size Estimation of Distinct Values (Cont.)

Joins: $r \bowtie s$

- If all attributes in $A$ are from $r$
  estimated $V(A, r \bowtie s) = \min(V(A, r), n_{r \bowtie s})$

- If $A$ contains attributes $A_1$ from $r$ and $A_2$ from $s$,
  estimated $V(A, r \bowtie s) = \min(V(A_1, r) \ast V(A_2 - A_1, s), V(A_1 - A_2, r) \ast V(A_2, s), n_{r \bowtie s})$
  - More accurate estimate can be got using probability theory, but this one works fine generally

- **Projection**: Estimation of distinct values are straightforward for projections
  - They are the same in $\prod_{A} r$ as in $r$

- **Aggregation**: The same holds for grouping attributes of aggregation
  - For aggregated values
    - For $\min(A)$ and $\max(A)$, the number of distinct values can be estimated as $\min(V(A, r), V(G, r))$ where $G$ denotes grouping attributes
    - For other aggregates, assume all values are distinct, and use $V(G, r)$
Choice of Evaluation Plans

- Must consider the interaction of evaluation techniques when choosing evaluation plans
  - Choosing the cheapest algorithm for each operation independently may not yield best overall algorithm, e.g.,
    - Merge-join may be costlier than hash-join, but may provide a sorted output which reduces the cost for an outer level aggregation
    - Nested-loop join may provide opportunity for pipelining

- Practical query optimizers incorporate elements of the following two broad approaches:
  1. Search all the plans and choose the best plan in a cost-based fashion
  2. Uses heuristics to choose a plan
Cost-Based Optimization

- Consider finding the best join-order for \( r_1 \Join r_2 \Join \ldots \Join r_n \).
- There are \( (2(n - 1))!/(n - 1)! \) different join orders for above expression (see Practice Exercise 13.10):
  - with \( n = 7 \), the number is 665280
  - with \( n = 10 \), the number is greater than 176 billion!

- Can reduce search space using dynamic programming:
  - Using dynamic programming, the least-cost join order for any subset of \( \{r_1, r_2, \ldots, r_n\} \) is computed only once and stored for future use.
  - Time complexity: \( O(3^n) \), with bushy trees (see Practice Exercise 13.11):
    - with \( n = 10 \), the number is 59,000 (instead of 176 billion!)
  - Space complexity: \( O(2^n) \)
Heuristic Optimization

- Cost-based optimization is expensive, even with dynamic programming

- Heuristic optimization transforms the query-tree by using a set of rules that typically (but not in all cases) improve execution performance:
  - Perform selection early (reduces the number of tuples)
  - Perform projection early (reduces the number of attributes)
  - Perform most restrictive selection and join operations (i.e. with smallest result size) before other similar operations

- Some systems use only heuristics, others combine heuristics with partial cost-based optimization
In **left-deep join trees**, the right-hand-side input for each join is a relation, not the result of an intermediate join.

If only left-deep trees are considered, time complexity of finding best join order is $O(n \ 2^n)$ (see Practice Exercise 13.12):
- with $n = 10$, the number of join orders is 10,000 (c.f., 59,000 or 176 billion)
- Space complexity remains at $O(2^n)$

Left-deep join orders are convenient for pipelined evaluation: the right operand is a stored relation and only one input to each join is pipelined.

Many optimizers consider only left-deep join orders.
End of Chapter 13