ICS 321 Spring 2011
Algebraic and Logical Query Languages (ii)

Asst. Prof. Lipyeow Lim
Information & Computer Science Department
University of Hawaii at Manoa
**Datalog : Database Logic**

**Atom**

- A (relational) **atom**
  - Consists of a **predicate** and a list of **arguments**
  - Arguments can be **constants** or **variables**
  - Takes on Boolean value (true or false)

**Relation R**

- A relation R can be represented as a predicate R
  - A tuple \(<a,b,c,d,e,f,g>\) is in R iff the atom R\((a,b,c,d,e,f,g)\) is true.
Example: tables in datalog

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

Datalog

- \( R(1,2) \)
- \( R(3,4) \)

True by default.

\( R(1,4) \) would be false
Arithmetic Atoms

\[ x < y \]
\[ x + 1 \geq y + 4z \]

Can contain both constants and variables.
Datalog Rules

LongMovie(t,y) :- Movies(t,y,l,g,s,p), l >=100

(t,y) is a tuple of LongMovie
IF (t,y,l,g,s,p) is a tuple of Movies and length of movie is at least 100

These two “t,y” have to match
These two “l” have to match

Anonymous variables

Aka “subgoal” Can be preceded by negation operator “NOT” or “~”
Safety Condition for Datalog Rules

Every **variable** that appears anywhere in the rule **must** appear in some **nonnegated, relational subgoal** of the body.

- Without the safety condition, rules may be underspecified, resulting in an infinite relation (not allowed).
- **Examples**
  - `LongMovie(t,y) :- Movies(t,y,l,_,_,_,) , l >=100`
  - `P(x,y) :- Q(x,z), NOT R(w,x,z), x<y`
Alternative Interpretation: Consistency

For each consistent assignment of nonnegated, relational subgoal,

Check the negated, relational subgoals and the arithmetic subgoals for consistency

<table>
<thead>
<tr>
<th>Q(x,z)</th>
<th>R(z,y)</th>
<th>Consistent?</th>
<th>NOT Q(x,y)</th>
<th>Head</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,2)</td>
<td>(2,3)</td>
<td>Yes</td>
<td>false</td>
<td></td>
</tr>
<tr>
<td>(1,2)</td>
<td>(3,1)</td>
<td>No, z=2,3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(1,3)</td>
<td>(2,3)</td>
<td>No, z=2,3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(1,3)</td>
<td>(3,1)</td>
<td>Yes</td>
<td>true</td>
<td>P(1,1)</td>
</tr>
</tbody>
</table>
Intensional vs Extensional

- **Extensional** predicates – relations stored in a database
- **Intensional** predicates – computed by applying one or more datalog rules

Datalog:

- \( Q(1,2) \)
- \( Q(1,3) \)
- \( R(2,3) \)
- \( R(3,1) \)
- \( P(x,y) : \neg Q(x,z), R(z,y), \neg Q(x,y) \)
What about bag semantics?

- Datalog still works if there are no negated, relational subgoals.
- Treat duplicates like non-duplicates

Datalog

<table>
<thead>
<tr>
<th>R(1,2)</th>
<th>R(1,2)</th>
<th>S(2,3)</th>
<th>S(4,5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>H(x,z) :- R(x,y), S(y,z)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>R(x,y)</th>
<th>S(y,z)</th>
<th>Consistent?</th>
<th>Head</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,2)</td>
<td>(2,3)</td>
<td>Yes</td>
<td>H(1,3)</td>
</tr>
<tr>
<td>(1,2)</td>
<td>(4,5)</td>
<td>No, y=2,4</td>
<td></td>
</tr>
<tr>
<td>(1,2)</td>
<td>(4,5)</td>
<td>No, y=2,4</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Example 1

### Datalog

<table>
<thead>
<tr>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Answer(x,y) :- A(x,y)</code></td>
</tr>
<tr>
<td><code>Answer(x,y) :- B(x,y)</code></td>
</tr>
</tbody>
</table>
Example 2

Datalog

Answer(x,y) :- A(x,y), B(x,y)
Example 3

Datalog

Answer(x,y) :- A(x,y), NOT B(x,y)
Example 4

Datalog

Answer(x,y) :- A(x,y), x > 10, y = 200
Example 5

**Datalog**

\[
\text{Answer}(x) : - A(x, y)
\]
Example 6

Datalog

Answer(w,x,y,z) :- A(w,x), B(y,z)
Example 7

Datalog

\[
\text{Answer}(w,x,y) \leftarrow A(w,x), B(x,y)
\]
Example 8

**Datalog**

\[
\text{Answer}(w,x,z) : \neg A(w,x), \neg B(y,z), x>y
\]
Example 9

\[
\text{Path}(x,y) :\text{-} \text{Edge}(x,y) \\
\text{Path}(x,z) :\text{-} \text{Edge}(x,y), \text{Edge}(y,z)
\]