Midterm Exam

March 10th, 2016
1:50-3:05

CS520 - Data Integration, Warehousing, and Provenance
Instructions

• Try to answer all the questions using what you have learned in class. Keep hard questions until the end.

• When writing a query, write the query in a way that it would work over all possible database instances and not just for the given example instance!

• The exam is closed book and closed notes! No calculator, smartphones, or similar allowed!

Consider the following database schema and example instance about music albums:

<table>
<thead>
<tr>
<th>product</th>
<th>pid</th>
<th>version</th>
<th>title</th>
<th>category</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>8.3c</td>
<td>VCleaner</td>
<td>antivirus</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>6.0</td>
<td>VCleaner</td>
<td>antivirus</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.3</td>
<td>EncM</td>
<td>music</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.4</td>
<td>EncM</td>
<td>music</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>supporter</th>
<th>name</th>
<th>salary</th>
<th>location</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Bob</td>
<td>40,000</td>
<td>Chicago</td>
</tr>
<tr>
<td></td>
<td>Alice</td>
<td>54,000</td>
<td>Austin</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bug</th>
<th>bugNumber</th>
<th>product</th>
<th>version</th>
<th>description</th>
<th>status</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>8.3c</td>
<td>Does not start on windows</td>
<td>resolved</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>6.0</td>
<td>Crashes after scan</td>
<td>open</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2</td>
<td>0.4</td>
<td>Does not play mp3</td>
<td>open</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>bugAssignment</th>
<th>name</th>
<th>bug</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Bob</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Bob</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Alice</td>
<td>2</td>
</tr>
</tbody>
</table>

Hints:

• Attributes with black background form the primary key of a relation (e.g., name for relation supporter).

• The attributes product and version of relation bug are a foreign key to relation product.

• The attribute name of relation bugAssignment is a foreign key to relation supporter.

• The attribute bug of relation bugAssignment is a foreign key to relation bug.
Part 1.1 Datalog (Total: 25 Points)

Recall that Datalog applies set semantics.

Question 1.1.1 (4 Points)
Write a Datalog program that returns product titles (attribute title of relation product).

Question 1.1.2 (6 Points)
Write a Datalog program that returns the description and status of bugs for product “VCleaner”.

Question 1.1.3  (7 Points)

Write a Datalog program that returns all products (attribute title) that belong to category *antivirus* or *office* (attribute category).

Question 1.1.4  (8 Points)

Write a Datalog program that returns the names of supportes that are not assigned to any open bugs (attribute status).
Part 1.2  Constraints (Total: 30 Points)

Question 1.2.1  Expressing Constraints in First-Order Logic (15 Points)

Recall the logical representation of constraints introduced in class. Write down the logical definition for the following constraints over the example schema:

- The foreign key from attributes `product` and `version` of relation `bug` to relation `product`.
- The primary key of relation `product`.
- The following functional dependency for relation `supporter`: `location → salary`
Question 1.2.2  Creating Denial Constraints (15 Points)

Create denial constraints over the example schema based on the following descriptions.

- All supporters earn less than $20,000.
- Resolved bugs (attribute `status`) should not be assigned to any supporter
- Each bug is assigned to at most one supporter
Part 1.3 Query Containment And Equivalence (Total: 27 Points)

Question 1.3.1 (27 Points)

Consider the 3 queries shown below. Check all possible containment relationships. If there exists a containment mapping from $Q_i$ to $Q_j$ then write down the mapping.

\[
Q_1(X, Y) : -R(X, X), R(X, Y).
\]
\[
Q_2(X, Y) : -R(X, X), R(Y, Y).
\]
\[
Q_3(X, Y) : -R(X, X), R(Z, Y).
\]
Part 1.4 Virtual Data Integration (Total: 18 Points)

Question 1.4.1 (9 Points)
Check all correct statements below. You have to answer the question (incorrect blanks are considered errors)

- GLAV mappings can be expressed as tuple-generating dependencies.
- Both the inverse rule algorithm and the Minicon algorithm compute maximally contained rewritings.
- Maximally contained rewritings are independent of the query language used for expressing rewritings.
- If there exists a maximally contained rewriting for $Q$ given a set of views then there has to exist an equivalent rewriting for a query $Q$ using the same set of views.
- The open world assumption is the same as the closed world assumption.
- $Q_G(X) : -\text{Person}(X,Y) \supseteq Q_L(X) : -\text{P}(X,Y,Z)$ is a GAV mapping.

Question 1.4.2 (9 Points)
Rewrite the following query using the inverse rules algorithm.

$$Q(X, A, Y, B) : -G(X, A, Y, B)$$

The available views are:

- $V_1(X,Y) : -G(X, A, Y, B)$
- $V_2(X, A) : -G(X, A, Y, B)$
- $V_3(Y, B) : -G(X, A, Y, B)$