CS 411a/433a/538a — Databases II
Midterm, Nov. 5, 2004
50 Minutes

Answer all questions on the exam page

No aids; no electronic devices.

The marks total 60

<table>
<thead>
<tr>
<th>Question</th>
<th>Maximum</th>
<th>Your Mark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>59</td>
<td></td>
</tr>
</tbody>
</table>
1. (16 marks) For each of the following, state whether the term applies to one, two or all three of: relational databases, object-oriented databases or XML databases. Give a BRIEF reason for your answer.

(a) attributes  
   All three

(b) sharing  
   All three

(c) closed query language  
   All three - XQuery and OQL by design, relational queries always produce relations.

(d) nesting of data to arbitrary levels  
   OODB and XML

(e) always has a schema  
   Relational and OODB. XML DDTs or Schemas are optional

(f) tags in the data  
   only XML

(g) identity comparisons  
   Only OODB. XML if you know about IDREFs

(h) path expressions  
   OODB and XML
2. (23 marks) Consider the following O2 schema information:

Class Person public type
tuple(Name : string,
    Address : string) end;

Class Singer inherit Person public type
tuple(Repertoire : unique set(Song));

Class Song public type
tuple(Title : string,
    Composer : Person,
    Genre : string) end;

Class Performance public type
tuple(Date : string,
    Performer : Singer,
    Performed : list(Song)) end;

Name Singers : unique set (Singer);
Name Songs : unique set (Song);
Name Performances : unique set (Performance);

Give OQL queries corresponding to the following:

(a) Give all singer names, song titles such that the song is in the singer’s repertoire.
    Select s.Name, o.Title
    From s in Singers, o in s.Repertoire

(b) Give the names of all singers who have a song whose title is "Yesterday" in their repertoire.
    Select s.Name
    From s in Singers
    Where exists o in s.Repertoire:(o.Title = "Yesterday")
(c) Give names of singers who have a song in their repertoire such that the composer's name is the same as the singer's name.
   Select s.Name
   From s in Singers, o in s.Repertoire
   Where o.Composer.Name = s.Name

(d) Give the names of all singers who have not performed.
   Select s.Name
   From s in Singers
   except
   (Select g.Performer.Name
    From g in Performances)

(e) Write an O2C run body, containing an embedded query, which, for each song, finds and prints out the song title and then the names of all the singers who have performed it in a Performance.
   Idea is:

   run body{
     declare all your o2 variables;
     for s in Songs {
       copy the song title to an o2 string variable v;
       o2query(Answ, "Select p.Performer.Name from p in Performances where
                   exists s in p.Performed:(p.Title = $1)", v);
       print out s.Title;
       for a in Answ {print out a};
     }
   }
3. (16 marks) Consider the following relations for a relational database:

- Singer(SingerID, Name, Town) primary key: {SingerID}
- Song(SongID, Title, Composer, Genre) primary key {SongID}
- Performs(SingerID, SongID, Date, Position) primary key {SingerID, SongID}

Which of the following pairs of queries are equivalent (i.e. are guaranteed to produce the same answer)? If they are not equivalent, say why.

(a) \( \sigma_{\text{Name} = \text{"Twain"}}(\text{Singer}) \bowtie \text{Performs} \)
   and
   \( \sigma_{\text{Name} = \text{"Twain"}}(\text{Singer} \bowtie \pi_{\text{SingerID}, \text{Date}}(\text{Performs})) \)
   No, the second version has fewer attributes.

(b) \( \pi_{\text{Title}}(\text{Singer} \bowtie (\text{Song} \bowtie \text{Performs})) \)
   and
   \( \pi_{\text{Title}}(\text{Performs} \bowtie (\text{Song} \bowtie \text{Singer})) \)
   yes

(c) \( \pi_{\text{Town}}(\sigma_{\text{Name} = \text{"Twain"}}(\text{Singer})) \)
   and
   \( \sigma_{\text{Name} = \text{"Twain"}}(\pi_{\text{Town}}(\text{Singer})) \)
   No, \( \pi \) in the second version gets rid of name, so query won’t work.
Assume the following statistics for some of the above data:

For Singer:
- No. of tuples in the Singer relation: 10
- No. of bytes in SingerID: 8
- No. of bytes per tuple in Singer: 50
- Distinct values in SingerID: 10

For Performs:
- No. of tuples in Performs: 50
- No. of bytes in SingerID: 8
- No. of bytes per tuple: 30
- Distinct values in SingerID: 8

(d) How many tuples are in Singer $\bowtie$ Performs?

8

(e) How many bytes per tuple are in Singer $\bowtie$ Performs?

50

(f) How many bytes are in $\pi_{SingerID}$ (Performs)?

8 tuples $\times$ 8 bytes $= 64$ bytes.
4. (8 marks) Answer **TWO** of the following questions. (If you answer more than two, only the first two will be marked.)

(a) What is the main goal of performing the algebraic query optimization for a relational database query?

(b) What is the main issue in having operations like \( \pi \) in relational algebra, in an object-oriented query language?

(c) What is one disadvantage of hash joins?

(d) What is meant by “identical \( \Rightarrow \) shallow equal”?

(e) Why does the execution of the \( \pi \) operation in relational algebra have an \( O(n \log_2 n) \) run time rather than \( O(n) \)?
(this page is for rough work)