CSE 344 Midterm

Wednesday, November 7, 2012, 9:30-10:20

Name: _____

Question	Points	Score
1	45	
2	30	
3	25	
Total:	100	

- This exam is open book and open notes but NO laptops or other portable devices.
- You have 50 minutes; budget time carefully.
- Please read all questions carefully before answering them.
- Some questions are easier, others harder. Plan to answer all questions, do not get stuck on one question. If you have no idea how to answer a question, write your thoughts about the question for partial credit.
- Good luck!

1 SQL and Physical Tuning

1. (45 points)

   You have been analyzing the data from a social networking site and have derived the following relation, which captures topics discussed by various users.

   `Discussion(user1,user2,topic)`

   The relation contains a tuple `(u1,u2,t)` every time a user `u1` discussed a topic `t` with user `u2`. To avoid duplicate entries, `user1` always precedes `user2` in alphabetical order.

   (a) (15 points) Write a SQL query that returns all topics discussed by `Alice` and `Bob` but not discussed by `Alice` and `Chuck`.

Solution:

```
select topic
from Discussion
where user1='Alice'
and user2='Bob'
and topic not in
    (select topic
     from Discussion
     where user1='Alice'
     and user2='Chuck')
```

Discussion(user1,user2,topic)

(b) (15 points) Write a SQL query that returns the number of topics discussed by more than 10 pairs of users.

Solution:

```
select count(*)
from
  ( select topic
    from Discussion
    group by topic
    having( count(*) > 10 ) as X
  )
```

(c) (5 points) Give two reasons why database administrators typically do **NOT** create an index on every single attribute of every single relation. You do not need to discuss the reasons. Just state them.

Solution:

1. Reason 1: Indexes take-up space
2. Reason 2: Indexes can slow-down updates (inserts, deletes, updates)

(d) (10 points) Explain how a database administrator should proceed in order to select a good set of indexes for a relational database. Note that more complete answers will receive more points.

Solution: The DBA should first talk to the developers and users to determine the **workload**, in the form of a set of **queries, updates, and their frequencies**, that needs to execute on the database. The DBA should then consider which indexes have the potential to speed-up the queries in the workload. He or she should consider the most frequent (or otherwise most important) queries first. When selecting the indexes, the DBA should consider the **trade-off between slowing down updates, using space, and accelerating queries**. The DBA should also consider which indexes should be **clustered and which ones should be unclustered**. For each relation, only one index can be clustered.

2 Relational Algebra, Datalog, and Relational Calculus

2. (30 points)

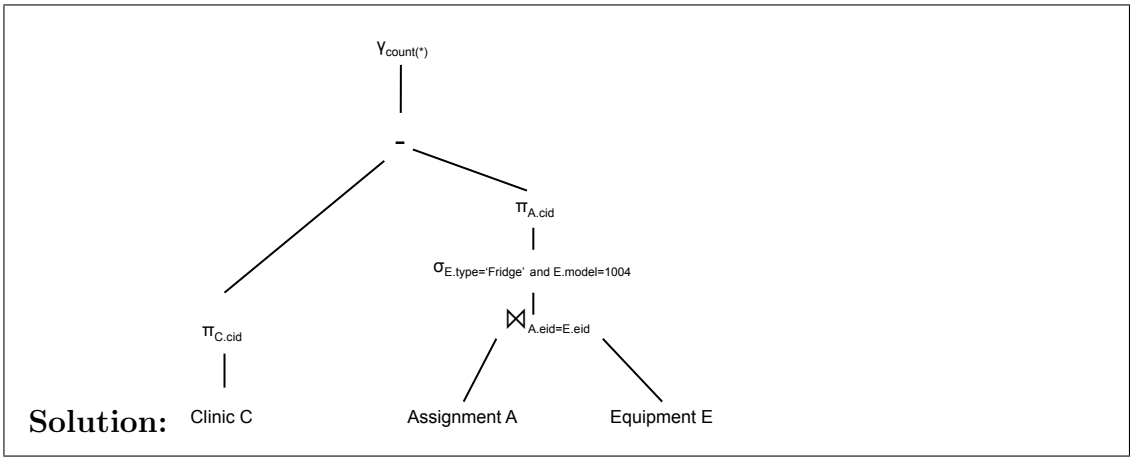
Consider the following database schema. Relation `Clinic` lists medical clinics with their unique identifiers, names, street addresses, and states. Relation `Equipment` lists the unique identifiers, types, and models of various pieces of equipment. Finally, relation `Assignment` indicates the equipment available in each clinic.

```
Clinic(cid, name, street, state)
Equipment(eid, type, model)
Assignment(cid, eid)
```

(a) (10 points) Write a Relational Algebra expression in the form of a logical query plan that is equivalent to the SQL query below:

```
select count(*)
from Clinic C
where not exists
   (select *
    from Assignment A, Equipment E
    where C.cid = A.cid
    and   A.eid  = E.eid
    and   E.type = 'Fridge'
    and   E.model = 1004
   )
```

Solution:



```
Clinic(cid, name, street, state)
Equipment(eid, type, model)
Assignment(cid, eid)
```

(b) (10 points) Write a Datalog query equivalent to the following SQL query:

```
select C.name
from Clinic C
where not exists
   (select *
    from Assignment A, Equipment E
    where C.cid = A.cid
    and   A.eid  = E.eid
    and  E.type = 'Fridge'
    and  E.model = 1004
   )
```

Solution:
AllClinics(x,y) :- Clinic(x,y,_,_)
NonAnswers(x) :- Assignment(x,z),Equipment(z,'Fridge',1004)
Answer(y) :- AllClinics(x,y), NOT NonAnswers(x)

```
Clinic(cid, name, street, state)
Equipment(eid, type, model)
Assignment(cid, eid)
```

(c) (10 points) Write a relational calculus query that returns the types of equipment assigned to clinics in the state of `WA`:

Solution:

$Q(t) = \exists c.\exists n.\exists s.\exists e.\exists m.\,(\texttt{Clinic}(c, n, s, \text{"}WA\text{"}) \wedge \texttt{Assignment}(c, e) \wedge \texttt{Equipment}(e, t, m))$

3 XML and XPath

3. (25 points)

   (a) (15 points) Consider the following XML document stored in a file called trips.xml:

```
<trips>
  <business reason='Meeting at CompanyX' destination='Baltimore'>
    <airline>American</airline>
    <stops>
      <location>Houston</location>
      <location>Boston</location>
    </stops>
  </business>

  <personal destination='Boston'>
    <airline>American</airline>
    <stops>
      <location>Chicago</location>
    </stops>
  </personal>

  <personal destination='Hawaii'>
    <airline>Alaska</airline>
    <stops>
    </stops>
  </personal>

</trips>
```

Write an XQuery expression that will transform it into the following document:

```
<trips>
  <airline>
    <name>American</name>
    <trip destination="Baltimore">
      <stops>2</stops>
    </trip>
    <trip destination="Boston">
      <stops>1</stops>
    </trip>
  </airline>
  <airline>
    <name>Alaska</name>
    <trip destination="Hawaii">
      <stops>0</stops>
    </trip>
  </airline>
</trips>
```

Solution:

```
<trips>
{
  for $d in doc("trips.xml")/trips
  for $x in distinct-values( $d//airline/text())

  return
    <airline>
      <name> {$x} </name>
      {
      for $y in $d/personal | $d/business
      where $y/airline/text() = $x
      return
           <trip> { $y/@destination }
             <stops>{ count($y/stops/location) }</stops>
           </trip>
      }
   </airline>
}
</trips>
```

(b) (10 points) Write a possible DTD for the document used as **input** above:

Solution:

```
<!DOCTYPE trips [
    <!ELEMENT trips  (business|personal)*>
    <!ELEMENT business (airline, stops)>
    <!ATTLIST business reason CDATA #REQUIRED >
    <!ATTLIST business destination CDATA #REQUIRED >
    <!ELEMENT personal (airline, stops)>
    <!ATTLIST personal destination CDATA #REQUIRED >
    <!ELEMENT airline  (#PCDATA )>
    <!ELEMENT stops    (location*)>
    <!ELEMENT location (#PCDATA )>
]>
```