CSE 344 Midterm

Monday, February 8, 2016, 2:30-3:20

Name: _____

Question	Points	Score
1	50	
2	20	
3	30	
Total:	100	

- This exam is CLOSED book and CLOSED devices.
- You are allowed ONE letter-size page with notes (both sides).
- 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 Indexing

1. (50 points)

   Consider the following database about picture tagging Website:

   ```
   Member(mid, name, age)
   Picture(pid, year)
   Tagged(mid, pid)
   ```

Solution:

```
drop table if exists Tagged;
drop table if exists Member;
drop table if exists Picture;

create table Member(mid int, name text, age int);
create table Picture(pid int, year int);
create table Tagged(mid int, pid int);

insert into Member values(1, 'Alice', 10);
insert into Member values(2, 'Bob', 20);
insert into Member values(3, 'Carol', 30);

insert into Picture values(10, 2011);
insert into Picture values(20, 2012);
insert into Picture values(30, 2013);
insert into Picture values(40, 2014);

insert into Tagged values(1,10);
insert into Tagged values(1,20);
insert into Tagged values(2,20);
insert into Tagged values(3,10);
insert into Tagged values(3,20);
insert into Tagged values(3,30);
```

Relation `Member` contains a tuple for each registered member of the Website; `Picture` contains the metadata of the pictures (its key `pid`, and the year when the picture was taken), and `Tagged` says which user was tagged in what picture. Primary keys are underlined. Attributes values may be null.

(a) (10 points) Write a SQL query that retrieves the names of all members that were tagged both in the year 2011 and in 2014. Return them sorted in lexicographic order.

Solution:

```
select x.name
from Member x, Tagged y1, Picture z1, Tagged y2, Picture z2
```

```
where x.mid = y1.mid and y1.pid = z1.pid and z1.year = 2011
  and x.mid = y2.mid and y2.pid = z2.pid and z2.year = 2014
order by x.name;
```

Common mistakes:

1. Joining both Picture Tables to the same Tagged Table.

```
select M.name
from Member M, Tagged T, Picture P1, Picture P2
where M.mid = T.mid and P1.pid = T.pid and P2.pid = T.pid and
P1.year = 2011 and P2.year = 2014
order by M.name;
```

`T.pid = P1.pid and T.pid = P2.pid => P1.pid = P2.pid => P1.year = P2.year` since pid is the primary key of Picture

Students received 5 points for this answer.

2. Selecting Members who were tagged in 2011 or 2014 instead of both.

```
select M.name
from Member M, Tagged T, Picture P,
where M.mid = T.mid and P.pid = T.pid and
(P.year = 2011 or P.year = 2014)
order by M.name;
```

Students received 3 points for this answer.

```
Member(mid, name, age)
Picture(pid, year)
Tagged(mid, pid)
```

(b) (10 points) For each query below, indicate if the query correctly returns the `name` of all users who were never tagged in 2015. Anywhere between zero and all queries compute the correct answer. Each query is syntactically correct.

```
Q1 = select distinct x.name
     from Member x, Tagged y
     where x.mid = y.mid
       and not exists
                  (select *
                   from Picture z
                   where y.pid = z.pid
                     and z.year = 2015);

Q2 = select distinct x.name
     from Member x
     where not exists
                 (select *
                  from Tagged y, Picture z
                  where x.mid = y.mid
                    and y.pid = z.pid
                    and z.year = 2015);

Q3 = select distinct x.name
     from Member x
     where not exists
             (select *
              from Tagged y
              where x.mid = y.mid
               and not exists
                   (select *
                    from Picture z
                    where y.pid = z.pid
                      and z.year = 2015));

Q4 = select distinct x.name
     from Member x, Tagged y, Picture z
     where x.mid = y.mid and y.pid = z.pid and z.year = 2015
     group by x.name
     having count(z.pid) = 0;
```

(b) **Q2**

Answer with any subset of Q1, Q2, Q3, Q4:

```
Member(mid, name, age)
Picture(pid, year)
Tagged(mid, pid)
```

(c) (10 points) Write a SQL query that returns for each member the total number of times that member was tagged in 2015. Order the results by the members' names. You should include all members, even those who were not tagged that year.

Solution:

```
select x.name, count(z.pid)
from Member x left outer join Tagged y on x.mid = y.mid
            left outer join Picture z on y.pid = z.pid and z.year = 2015
group by x.mid, x.name
order by x.name;
```

Common mistake:

```
SELECT m.name, COUNT(*)  -- incorrect count
FROM Member m LEFT OUTER JOIN Tagged t ON (m.mid = t.mid), Picture p
         -- incorrect join
WHERE p.pid = t.pid and p.year=2015
group by m.name
order by m.name
```

Received 7 because of the incorrect join and needing a COUNT(p.pid)

```
SELECT m.name, COUNT(*)
FROM Member m, Tagged t, Picture p -- need outer join!
WHERE m.mid = t.mid and p.pid = t.pid and p.year=2015
group by m.name
order by m.name
```

Received 6 because no outer joining

A lot of students missed 1 point because they put `count(*)` instead of `count(p.pid)`

```
Member(mid, name, age)
Picture(pid, year)
Tagged(mid, pid)
```

(d) (10 points) Write a SQL query that returns the mid's and names of all members that were tagged only in pictures were Alice was also tagged. Hint: you may want to first answer question 2.b.

Solution: Datalog:

```
aliceTagged(pid) :- Member(mid, 'Alice',-),Tagged(mid, pid)
nonAnswer(mid) :- Tagged(mid,pid) not aliceTagged(pid)
answer(mid,name) :- Member(mid,name,-), not nonAnswer(mid)
```

```
select x.mid, x.name
from Member x
where x.mid not in
           (select y.mid
            from Tagged y
            where y.pid not in
               (select v.pid from Member u,Tagged v
                where u.name='Alice' and u.mid=v.mid));
```

Common mistakes:

1. Have a single level of negation.
2. Have a single level of negation combined with `u.name!='Alice'`

Students received 4 points for such solutions.

```
Member(mid, name, age)
Picture(pid, year)
Tagged(mid, pid)
```

(e) Consider a workload consisting of many queries of this kind:

```
select x.name
from Member x, Tagged y, Picture z
where x.mid = y.mid
  and y.pid = z.pid
  and z.year = ?;
```

For each index below, indicate if it can potentially speed up the workload, if it is the only index that exists. You will assume that the `Member` and `Tagged` are very large relations, and that only a very small number of pictures are in any given year.

i. (1 point) Index on `Picture(year)`. Yes or no? i. **yes**

ii. (1 point) Index on `Picture(pid)`. Yes or no? ii. **no**

iii. (1 point) Index on `Picture(pid,year)`. Yes or no? iii. **no**

iv. (1 point) Index on `Picture(year,pid)`. Yes or no? iv. **yes**

v. (1 point) Index on `Tagged(mid)`. Yes or no? v. **no**

vi. (1 point) Index on `Tagged(pid)`. Yes or no? vi. **yes**

vii. (1 point) Index on `Tagged(pid,mid)`. Yes or no? vii. **yes**

viii. (1 point) Index on `Member(mid)`. Yes or no? viii. **yes**

ix. (1 point) Index on `Member(name)`. Yes or no? ix. **no**

x. (1 point) Index on `Member(age)`. Yes or no? x. **no**

2 Relational Algebra, Datalog, and Relational Calculus

2. (20 points)

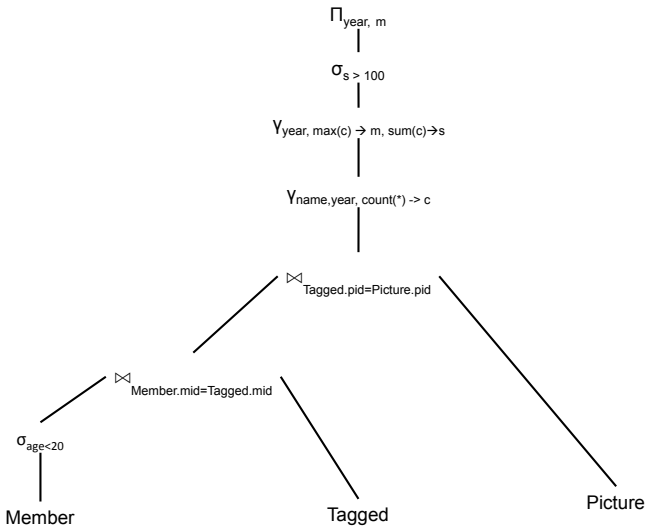
Consider the same relational schema as before:

```
Member(mid, name, age)
Picture(pid, year)
Tagged(mid, pid)
```

(a) (5 points) Write a Relational Algebra expression in the form of a logical query plan (i.e., draw a tree) that is equivalent to the SQL query below. Your query plan does not have to be necessarily "optimal": however, points will be taken off for overly complex solutions.

```
select w.year, max(w.c) as m
from (select x.name, z.year, count(*) as c
      from Member x, Tagged y, Picture z
      where x.mid = y.mid
        and y.pid = z.pid
        and age < 20
      group by x.name, z.year) w
group by w.year
having sum(w.c) > 100;
```

Solution:



```
Member(mid, name, age)
Picture(pid, year)
Tagged(mid, pid)
```

(b) (10 points) Write a query in datalog with negation that returns the mid's and names of all members that were tagged only in pictures were Alice was also tagged.

Solution:

```
aliceTagged(pid) :- Member(mid, 'Alice',-),Tagged(mid, pid)
nonAnswer(mid) :- Tagged(mid,pid) not aliceTagged(pid)
answer(mid,name) :- Member(mid,name,-), not nonAnswer(mid)
```

```
Member(mid, name, age)
Picture(pid, year)
Tagged(mid, pid)
```

(c) (5 points) This question is about matching relational calculus queries to English statements. Consider the relational calculus queries below:

(a)

$$P1(m,x) = \exists a.\texttt{Member}(m,x,a) \wedge \\ \forall p(\texttt{Tagged}(m,p) \Rightarrow (\texttt{Picture}(p,2015) \vee \texttt{Picture}(p,2016)))$$

(b)

$$P2(m,x) = \exists a.\texttt{Member}(m,x,a) \wedge \\ \forall p((\texttt{Picture}(p,2015) \vee \texttt{Picture}(p,2016)) \Rightarrow \texttt{Tagged}(m,p))$$

(c)

$$P3(m,x) = \exists a.\texttt{Member}(m,x,a) \wedge \\ [\forall p(\texttt{Picture}(p,2015) \Rightarrow \texttt{Tagged}(m,p)) \vee \forall p(\texttt{Picture}(p,2016) \Rightarrow \texttt{Tagged}(m,p))]$$

(d)

$$P4(m,x) = \exists a.\texttt{Member}(m,x,a) \wedge \\ \forall p1(\texttt{Tagged}(m,p1) \wedge \texttt{Picture}(p1,2015) \Rightarrow \exists p2(\texttt{Tagged}(m,p2) \wedge \texttt{Picture}(p2,2016)))$$

(e)

$$P5(m,x) = \exists a.\texttt{Member}(m,x,a) \wedge \\ \forall p \forall n((\texttt{Picture}(p,2015) \vee \texttt{Picture}(p,2016)) \wedge \texttt{Tagged}(n,p) \Rightarrow m = n)$$

(continued on the next page)

For each English statement below, indicate which relational calculus query answers it:

(i) Find all members that were tagged in 2015 only if they were also tagged in 2016.

(c) **P4**

Choose one of P1,P2,P3,P4,P5,none:

(ii) Find all members that were tagged in all pictures in 2015 and in all pictures in 2016.

(c) **P2**

Choose one of P1,P2,P3,P4,P5,none:

(iii) Find the member that was the only person tagged in 2015 and 2016.

(c) **P5**

Choose one of P1,P2,P3,P4,P5,none:

(iv) Find all members that were tagged only in pictures in 2015 or 2016.

(c) **P1**

Choose one of P1,P2,P3,P4,P5,none:

(v) Find all members that were tagged in all pictures in 2015, or were tagged in all pictures in 2016.

(c) **P3**

Choose one of P1,P2,P3,P4,P5,none:

3 Relational Data Model and Query Evaluation

3. (30 points)

Answer each question below.

(a) (2 points) In the relational data model an attribute of a relation cannot be another relation.

(a) **True**

True or false?

(b) (2 points) Consider the relation `Product(productName, manufacturer, price, weight)` where the key is `(productName, manufacturer)`. Can there be two different products with the same value of `manufacturer`?

(b) **Yes**

Yes or no?

(c) (2 points) If an attribute in a table is a foreign key, then the table cannot contain two tuples with the same value of that attribute.

(c) **False**

True or false?

(d) (2 points) In this and the following question we will assume that the relational algebra has set semantics. If the relation $R(A, B)$ has m tuples, and the relation $S(B, C)$ has n tuples, what is the largest possible size of the output of the natural join $R \bowtie S$? Choose one of the following:

(a) m
(b) n
(c) mn
(d) $m + n$
(e) $\max(m, n)$
(f) $(m + n)/2$
(g) None of the above.

(d) **c**

Answer a-g:

(e) (2 points) If the relation $R(A, B)$ has m tuples, and the relation $S(B, C)$ has n tuples, what is the largest possible size of the output of $\Pi_{AB}(R \bowtie S)$? Here $\bowtie$ represents the natural join. Choose from the same options as in the previous question.

(e) **c**

Answer a-g:

(f) Consider two relations $R(A, B), S(C, D)$, where all attributes are integers and cannot be NULL. For each identity below, indicate whether it is true or false.

i. (2 points) $R \bowtie_{B=C} S = S \bowtie_{B=C} R$

i. **True**

True or false?

ii. (2 points) $(\sigma_{B<1000}(R)) \bowtie_{B \neq C} S = (\sigma_{B<1000}(R)) \bowtie_{B \neq C} (\sigma_{C \geq 1000}(S))$

ii. **False**

True or false?

iii. (2 points) $R \times S - R \bowtie_{B=C} S = R \bowtie_{B \neq C} S$

iii. **True**

True or false?

iv. (2 points) $R - \Pi_{AB}(R \bowtie_{B=C} S) = \Pi_{AB}(R \bowtie_{B \neq C} S)$

iv. **False**

True or false?

v. (2 points) $\gamma_{A,\text{count}(*) \to F}(R \bowtie_{B=C} S) = \gamma_{A,\text{sum(E)} \to F}(R \bowtie_{B=C} (\gamma_{C,\texttt{count(*)} \to E}(S)))$

v. **True**

True or false?

(g) (2 points) If a relation is stored in 1000 blocks on disc, then it cannot have more than 1000 records.

(g) **False**

True or false?

(h) (2 points) Alice is a database administrator. She wants to add indexes to a table that is accessed frequently by her applications. She knows that a clustered index delivers better performance than an unclustered index, but most indexes she creates are unclustered. Why? Choose one of the following:

(a) Because clustered indexes take more space than unclustered indexes and Alice's database servers has little space.

(b) Because a relation can have only one clustered index, but many unclustered indexes.

(c) Because clustered indexes will slow down updates and Alice's application has many updates.

(d) None of the above.

(h) **b**

Choose one of a-d:

(i) (2 points) There is an important distinction between logical operators and physical operators. Indicate which of the operators below are physical operators:

(a) Theta join
(b) Hash join
(c) Natural join
(d) Eq join
(e) Index-based join
(f) Merge join

(i) **b,e,f**

Answer a-f (choose all that apply):

(j) (2 points) Did the speed of sequential reads from disk increase significantly over the years?

(j) **Yes**

Yes or no?

(k) (2 points) Did the speed of random reads from disk increase significantly over the years?

(k) **No**

Yes or no?