CSE 444 Midterm Exam

July 28, 2010

Name _____Sample Solution_____

Question 1	/ 28
Question 2	/ 20
Question 3	/ 16
Question 4	/ 20
Question 5	/ 16
Total	/ 100

The exam is open textbook and open lecture notes, including any marginal or other notes you have made during lecture. Please put away all other materials including projects, homework, old exams, and sample solutions. No computers, electronics, communications, or other devices are permitted.

Please wait to turn the page until everyone has their exam and you are told to begin.

Question 1. SQL (28 points, 7 each part) We have a small database with three tables to keep track of mountains and the people who climb them. The tables and their attributes are as follows:

Mountain(name, height, country, state)
Climber(cid, name, sex)
Ascent(cid, mountain, month, year)

- Mountain: name (string) is assumed to be unique for this problem; height in feet (integer); country and state (or province, district, etc.) where the mountain is located (strings)
- Climber: cid (unique integer), name (string), sex (string, either 'M' or 'F', and never null). Different climbers may have the same name, but they will have different cid numbers.
- Ascent: record of a single climb by a single climber. cid is a foreign key referencing Climber, mountain is a foreign key referencing a name in Mountain, month and year are integers. We assume that no climber climbs a single mountain more than once in a given month and year.

(a) Write a SQL query that gives the names of all climbers who have climbed the tallest mountain. You may assume that there is only one mountain that is taller than all of the others. If two or more climbers have the same name, it is your choice whether that name appears more than once.

```
select c.name
from climber c, ascent a, mountain m
where c.cid = a.cid and
    a.mountain = m.name and
    m.height >= all (select height from mountain)
```

(b) Write a SQL query that returns the percentage of women among the climbers that ascended the mountain named 'Rainier' in July 2007. You may assume there are at least one man and one woman in the list of climbers that ascended Rainier during that month.

```
select 100 * women.nbr / total.nbr
from    (select count(*) as nbr
         from ascent a, climber c
         where a.cid = c.cid and a.mountain = 'Rainier' and
         a.month = 7 and  a.year = 2007)
         as total,
         (select count(*) as nbr
         from ascent a, climber c
         where a.cid = c.cid and a.mountain = 'Rainier' and
         a.month = 7 and a.year = 2007 and c.sex = 'F')
         as women
```

(continued next page)

Question 1. (cont) Schemas repeated for convenience:

Mountain(name, height, country, state)
Climber(cid, name, sex)
Ascent(cid, mountain, month, year)

(c) Write a SQL query that returns a list of every mountain in the database and the number of different climbers who have ascended each mountain, sorted in descending order by number of climbers. Climbers may have climbed the same mountain many times, but each unique climber should only be counted once in the total for each mountain. If several mountains have the same number of climbers, those mountains with the same number may be listed in any order.

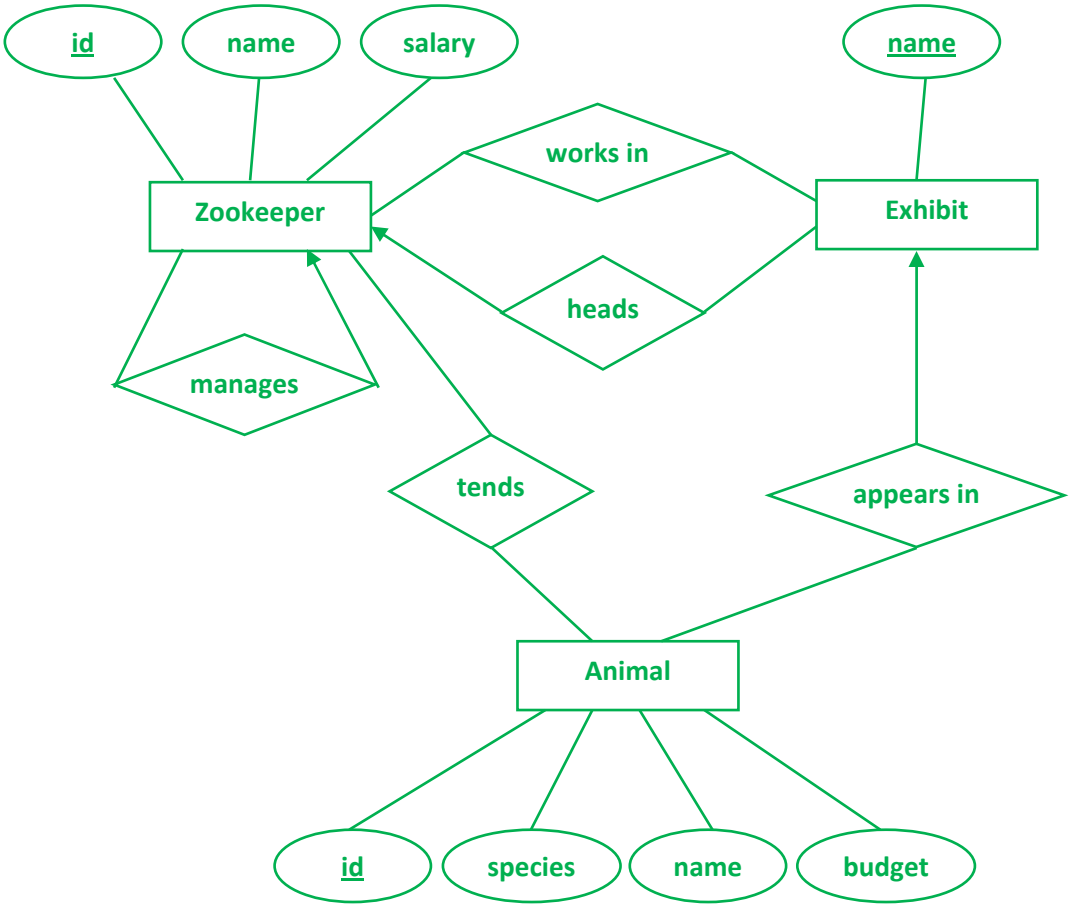
```
select count (distinct a.cid) as c_climbers
from mountain m left outer join ascent a
    on m.name = a.mountain
group by m.name
order by c_climbers desc;
```

(d) Write a SQL query that determines the state in the country 'USA' that has the most mountains at least 12,000 feet high, and returns the name of that state and a list of those mountains, sorted by mountain name. You may assume there is only one state with this "largest number of tall mountains". You may repeat the name of the state along with each mountain name in the output if that is more convenient.

```
select name, state
from mountain
where country = 'USA' and
   state = (
          select state
          from mountain
          where country = 'USA' and height >= 12000
          group by state
          having count(*) >= all (
                    select count(*)
                    from mountain
                    where country = 'USA' and height >= 12000
                    group by state
                    )
          )
   and height >= 12000
order by name;
```

Question 2. Conceptual design (20 points) We would like to design a database to keep track of a zoo.
(a) Give an E/R diagram that captures the following entities and relationships:

- The zoo employees are known as zookeepers. Each zookeeper has a name, a unique employee ID number, and a salary. He or she may work in any number of exhibits, or none at all (administrative staff do not work in exhibits, for example). A zookeeper does not have to work in an exhibit to head it.
- The zoo has several exhibits. Each one has an exhibit name and a head zookeeper who is in charge of it. A zookeeper may head many exhibits, but each exhibit has at most one head.
- Each zookeeper has at most one manager, who is another zookeeper (some zookeepers, such as the director of the zoo, have no manager). A zookeeper can manage any number of employees, including none.
- The zoo has many animals. Each has a name, species, unique ID number, and a weekly budget for its care. Each animal is tended by at least one zookeeper, and may appear in at most one exhibit. Some animals, such as those housed in the veterinary clinic, are not in any exhibit.



(continued next page)

Question 2 (cont.) (b) Give SQL CREATE TABLE statements for tables that implement your E/R diagram from part (a). You should give the attribute names and types for each table, and clearly indicate which attributes are keys, which others are unique, and which are foreign keys referencing other tables.

```
CREATE TABLE Zookeeper (
        id INTEGER PRIMARY KEY,
        name VARCHAR(100),
        salary_cents INTEGER,
        manager_id INTEGER REFERENCES Zookeeper(id)
);

CREATE TABLE Exhibit (
        name VARCHAR(100) PRIMARY KEY,                  -- PRIMARY KEY is optional here
        head_id INTEGER REFERENCES Zookeeper(id)
);

CREATE TABLE Animal (
        id INTEGER PRIMARY KEY,
        species VARCHAR(100),
        name VARCHAR(100),
        budget_cents INTEGER,
        appears_in VARCHR(100) REFERENCES Exhibit(name)
);

CREATE TABLE WorksIn (
        zkid INTEGER REFERENCES Zookeeper(id),
        exname VARCHAR(100) REFERENCES Exhibit(name),
        PRIMARY KEY(zkid, exname)
);

CREATE TABLE Tends (
        zkid INTEGER REFERENCES Zookeeper(id),
        aid INTEGER REFERENCES Animal(id),
        PRIMARY KEY(zkid, aid)
);
```

Note: If we could do it over again, it would have been better to just ask for the table schemas instead of the full CREATE TABLE SQL statements in order to cut down on the amount of writing needed.

Question 3. BCNF (16 points) Suppose we have a relational schema R(A,B,C,D,E) with the following functional dependencies:

- A -> E
- C -> D
- A,B -> C,D

Decompose this relation, if needed, into collections of relations that are in BCNF. At each step, show your work and explain which dependency violation(s) you are correcting. Be sure the steps in the decomposition are clear and that it is clear which tables are the final ones. Also, identify the keys of each table by underlying the attribute(s) that make up the key.

Hint: there may be more than one correct solution to the problem.

The closures of the various dependencies are:

{AB}+ = {ABCDE}
A+ = {AE}
C+ = {CD}

The first one is not a problem, the other two violate BCNF.

Solution I. Use A->E to decompose the table into

R1(A,E)
R2(A,B,C,D)

R1 is in BCNF. R2 has a bad dependency C->D. Use that to decompose and we get

R21(C,D)
R22(A,B,C)

The final tables are R1, R21, and R22, which are in BCNF.

Solution II. If we use C->D for the first decomposition we get

R1(C,D)
R2(A,B,C,E)

R1 is in BCNF, R2 has a bad dependency A->E. Decompose R2 to get

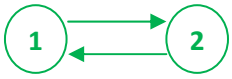
R21(A,E)
R22(A,B,C)

The final tables are R1, R21, and R22.

Question 4. Serialization (20 points). For each of the following schedules,

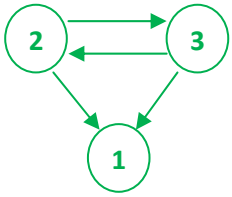
i. Draw the precedence graph for the schedule.
ii. If the schedule is conflict-serializable, give the equivalent serial schedule. If the schedule is not conflict-serializable, explain why not.
iii. If the schedule is not conflict-serializable, but there still is an equivalent serial schedule, give that schedule and explain why it is equivalent.

(a) r2(X) r1(X) r2(Z) w2(Z) w2(X) r1(Z) w1(X)



This is not conflict-serializable because there is a cycle in the precedence graph. Further, there is no equivalent serial schedule, since T1 must read X before T2 writes it, but T2 must write Z before T1 reads it.

(b) r2(X) r3(Y) w3(X) r1(Y) w2(X) w1(Y) w1(X)



This is not conflict serializable because there is a cycle in the graph.

However, the schedule r2(X) w2(X) r3(Y) w3(X) r1(Y) w1(Y) w1(X) is serial and is view equivalent to the original schedule. The writes to X by T2 and T3 are superseded by w1(X) at the end of the schedule before any other transactions read X.

Question 5. (16 points) Assorted short questions.

(a) In a system with a simple **undo** log, when a transaction wants to commit it must first wait until all of its data pages have been written to disk. True or false? ___**true**_____

(b) In a system with a simple **redo** log, changes to the transactions data pages can be written before or after they are written to the log. True or false? ___**false**_____
(In a redo log, outputs must be done after they are logged)

(c) in a system with a simple **redo** log using non-quiescent checkpointing, an <END CKPT> record in the log indicates that all transactions identified in the corresponding <START CKPT> record have either committed or aborted. True or false? ___**false**_____
(END CKPT only indicates that all dirty pages belonging to transactions that committed before the <START CKPT> are now on disk)

(d) In the ARIES recovery protocol, after a crash, the analysis pass examines the log and determines two things. One is the list of active transactions that did not finish at the time of the crash and will need to be undone. The other is a number known as FirstLSN that identifies an entry (LSN) in the log. What is the significance of this number?

This identifies the earliest entry in the log that must be examined on the redo pass. (i.e., the first log entry whose changes to the database might have been lost in the crash)

(e) In the ARIES recovery protocol, if a crash occurs during the undo phase of the recovery, then after a restart, all of the undo operations from before the crash will be repeated because they are idempotent.
True or false? ___**false**_____
(Each ARIES undo operation is logged with a CLR. The CLR will be redone if there is a crash before the undo phase finishes; no undo operation happens more than once.)

(f) In a system using the ARIES recovery protocol, changes made to transaction data pages may be written to disk before or after the transaction's commit record is written to the log.
True or false? ___**true**_____

(g) In a scheduler that uses **timestamps** for concurrency control, each database element X has a read timestamp RT(X). Whenever a transaction reads a database element X, the element's RT(X) timestamp is updated by recording the transaction's timestamp in RT(X) for that database element.
True or false? ___**false**_____
(RT(X) = max(old RT(X) value, timestamp on current transaction))

(h) In a scheduler that uses **validation** for concurrency control, the committed transactions have the same effect as they would in a serial schedule where the transactions are executed in the order in which they started (i.e., transaction start times determine the serialization order).
True or false? ___**false**_____
(Serialization order = transaction validation order)