Name:____

CSE 444, Fall 2010, Midterm Examination
10 November 2010

Rules:

- Open books and open notes.
- No laptops or other mobile devices.
- Please write clearly.
- Relax! You are here to learn.

Question	Max	Grade
1	30	
2	30	
3	40	
Total	100	

Initials:____

1. (**30** points) **SQL**

   Consider a database with the following three relations.

```
Sensor( sid, type, location, description )
Temperature ( sid, time, tempvalue )
Pressure ( sid, time, pressvalue )
```

   `Temperature.sid` is a foreign key that references `Sensor.sid`.

   `Pressure.sid` is a foreign key that references `Sensor.sid`.

   This database holds a set of sensor readings (temperature and pressure) recorded by a set of sensors deployed at various locations. Some sensors record only temperature readings. Others record only pressure readings. Some record both. Each sensor produces a reading every minute (pressure reading, temperature reading, or both). Each sensor is associated with a given type, location, and description.

   (a) (**8** points) Write a SQL query that returns the type and description of all sensors that have produced both temperature readings and pressure readings. Each unique combination of type and description should appear only once in the output.

   Solution:

```
SELECT DISTINCT S.type, S.description
FROM Temperature T, Pressure P, Sensor S
WHERE T.sid = S.sid AND P.sid = S.sid
```

Initials:____

(b) (**8** points) Write a SQL query that computes the highest ever pressure reading among all sensors that recorded both pressure and temperature readings.

Solution:

```
SELECT MAX(PI.pressvalue)
FROM   Pressure PI, Sensor SI, Temperature TI
WHERE  PI.sid = SI.sid AND TI.sid = SI.sid
```

Initials:____

(c) (**14** points) Consider the set of sensors that record BOTH temperature and pressure (your query will have to compute that set). Write a SQL query that computes, for these sensors only, the `sid` and average `tempvalue` for all sensors that recorded the highest ever pressure reading. Note that more than one sensor could have recorded the same, highest pressure reading.

Solution:
We can use our query from the previous question in a subquery to find the `sid` and average `tempvalue` for all sensors that produced such a high pressure reading:

```
SELECT S.sid, AVG(T.tempvalue)
FROM   Pressure P, Sensor S, Temperature T
WHERE  P.sid = S.sid AND S.sid = T.sid
AND    P.pressvalue >= (SELECT MAX(PI.pressvalue)
                        FROM   Pressure PI, Sensor SI, Temperature TI
                        WHERE  PI.sid = SI.sid AND TI.sid = SI.sid)
GROUP BY S.sid
```

Alternate solution:

```
SELECT S.sid, AVG(T.tempvalue)
FROM   Pressure P, Sensor S, Temperature T
WHERE  P.sid = S.sid AND S.sid = T.sid
AND    P.pressvalue >= ALL (SELECT PI.pressvalue
                        FROM   Pressure PI, Sensor SI, Temperature TI
                        WHERE  PI.sid = SI.sid AND TI.sid = SI.sid)
GROUP BY S.sid
```

Initials:____

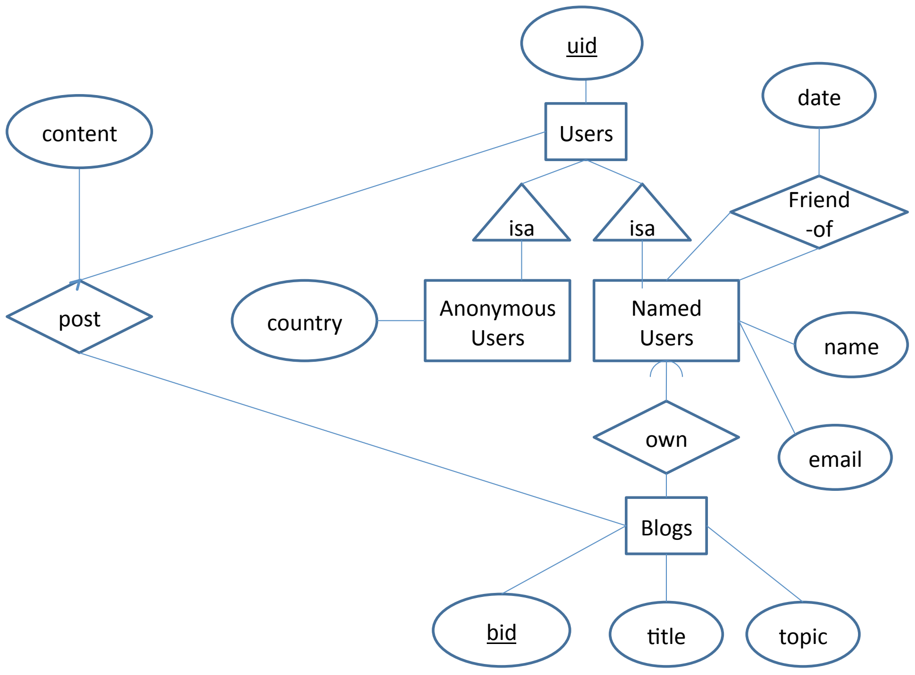
2. (**30** points) **Conceptual Design**

   (a) (**12** points) Draw an E/R diagram describing the following domain:

   - **Users** have a single attribute **uid** (key).
   - **AnonymousUsers** are a type of **Users** with attribute **country**.
   - **NamedUsers** are a type of **Users** with attributes **name** and **email**.
   - **Blogs** have attributes **bid**(key), **title**, and **topic**.
   - A **NamedUser** can be a **friend-of** zero or more other **NamedUsers**. Each friend-of relationship has an associated start **date**.
   - A **NamedUser** can **own** many **Blogs**, but each blog is owned by exactly one user.
   - A **User** can **post** to zero or more **Blogs**. Each post has a given **content**. Many users can post to a blog.

   Your answer should consist of an E/R diagram, which includes entity sets, attributes, relationships, ISA relations. Indicate the type of each relationship with appropriate arrows (one-one, one-many, etc.).

   Solution:



Initials:____

(b) (**8** points) Consider the following relational schema and set of functional dependencies. (a) List **all** superkey(s) for this relation. (b) Which of these superkeys form a key (i.e., a minimal superkey) for this relation? Justify your answer in terms of functional dependencies and closures.

R(A,B,C,D,E) with functional dependencies CD → E and A → B.

Solution:
(a) A superkey is a set of attributes X s.t. X^+ = all attributes.

From the FDs above, we can derive:

$$\{A, B, C, D, E\}^+ = \{A, B, C, D\}^+ = \{A, C, D, E\}^+ = \{A, C, D\}^+ = \{A, B, C, D, E\}$$

Hence,
$\{A, B, C, D, E\}, \{A, B, C, D\}, \{A, C, D, E\}$, and $\{A, C, D\}$ are all superkeys.

(b) A key is a set of attributes which form a superkey and for which no subset is a superkey. In our example, $\{A, C, D\}$ is the only key.

Initials:____

(c) (**10** points) Decompose R into BCNF. Show your work for partial credit. Your answer should consist of a list of table names and attributes and an indication of the keys in each table (underlined attributes).

Solution:
Both functional dependencies violate BCNF.
Try $\{A\}^+ = \{A, B\}$. Decompose into R1($\underline{\text{A}}$,B) and R2($\underline{\text{A,C,D}}$,E).

R1 has two attributes, so it is necessarily in BCNF.

R2 is not in BCNF, since $\{C, D\}$ is not a key and we have CD $\rightarrow$ E.

Try $\{C, D\}^+ = \{C, D, E\}$. Decompose into R3($\underline{\text{C, D}}$, E) and R4 ($\underline{\text{A, C, D}}$)

End result: R1($\underline{\text{A}}$,B), R3($\underline{\text{C, D}}$, E), and R4 ($\underline{\text{A, C, D}}$)

Initials:____

3. (**40** points) **Transactions**

   (a) (**6** points) In the ARIES method, assuming NO checkpoints have been used, explain what happens during the ANALYSIS pass of recovery. Your answer should indicate at least (1) what part of the log the system reads and in what direction and (2) what data structures the system rebuilds.

   Solution:
   In the absence of checkpoints, the analysis pass reads the log from the beginning to the end. It rebuilds the Dirty Pages Table and the Transactions Table to determine the state of the system as of the time of the crash. It rebuilds these data structures by updating them according to the log records that it encounters during the forward scan.

   (b) (**6** points) After the analysis pass, the protocol performs a REDO pass. Explain (1) where does REDO start reading the log and in what direction it reads the log, (2) what happens during the REDO pass.

   Solution:
   The REDO pass begins at the log record whose LSN corresponds to the earliest recoveryLSN of all the entries in the Dirty Page Table. From that point, the REDO pass redoes the updates of all transactions (committed or otherwise). At the end of this pass, the database is in the same state as it was right before the crash.

Initials:____

(c) (**6** points) The last pass during recovery is the UNDO pass. Explain (1) where does the UNDO start reading the log and in what direction it reads the log, (2) what happens during the UNDO pass.

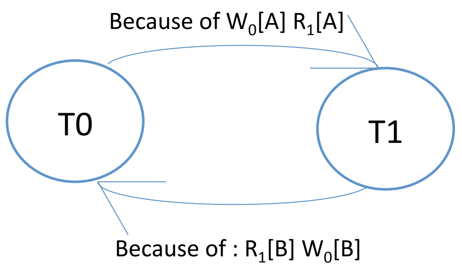
Solution:
The UNDO pass scans backward from the end of the log. The pass undoes the updates by all transactions that had not committed by the time of the crash. These transactions can be found in the Transactions Table rebuilt during the analysis pass. All updates are undone unconditionally (since the REDO pass ensured that all logged updates have been applied to affected pages). For each update that is undone, the undo operation is logged with a Compensation Log Record (CLR).

Initials:____

(d) (**7** points) Consider the following two transactions and schedule (time goes from top to bottom). Is this schedule conflict-serializable? Explain why or why not.

Transaction T_0	Transaction T_1
$r_0[A]$	
$w_0[A]$	
	$r_1[A]$
	$r_1[B]$
	c_1
$r_0[B]$	
$w_0[B]$	
c_0	

Solution:
The schedule is not conflict serializable because the precedence graph contains a cycle. The graph has an edge $T_0 \rightarrow T_1$ because the schedule contains $w_0[A] \rightarrow r_1[A]$. The graph has an edge $T_1 \rightarrow T_0$ because the schedule contains $r_1[B] \rightarrow w_o[B]$.



Initials:____

(e) (**7** points) Show how 2PL can ensure a conflict-serializable schedule for the same transactions above. Use the notation $L_i[A]$ to indicate that transaction i acquires the lock on element A and $U_i[A]$ to indicate that transaction i releases its lock on A.

Solution:

Multiple solutions are possible.

Transaction T_0	Transaction T_1
$L_0[A]$	
$r_0[A]$	
$w_0[A]$	
	$L_1[A] \rightarrow \textit{blocks}$
$L_0[B]$	
$r_0[B]$	
$w_0[B]$	
$U_0[A]$	
$U_0[B]$	
c_0	
	$L_1[A] \rightarrow \textit{granted}$
	$r_1[A]$
	$L_1[B]$
	$r_1[B]$
	$U_1[A]$
	$U_1[B]$
	c_1

Initials:____

(f) (**8** points) Show how the use of locks without 2PL can lead to a schedule that is NOT conflict serializable.

Solution:

Transaction T_0	Transaction T_1
$L_0[A]$	
$r_0[A]$	
$w_0[A]$	
$U_0[A]$	
	$L_1[A]$
	$r_1[A]$
	$U_1[A]$
	$L_1[B]$
	$r_1[B]$
	$U_1[B]$
	c_1
$L_0[B]$	
$r_0[B]$	
$w_0[B]$	
$U_0[B]$	
c_0	