CS411 Database Systems
Fall 2008

University of Illinois at Urbana-Champaign

Midterm Examination
October 14, 2008
Time Limit: 75 minutes

- Print your name and NetID below. In addition, print your NetID in the upper left corner of every page.

Name: _____ **NetID:** _____

- Closed notes; closed book; no sheet of formulas permitted.
- Please write your answers directly on the exam sheet. The space we left for your answers is often more than what you actually need. Please use the back side of the exam as scratch paper.
- In case you find a question ambiguous, please write down your assumption and answer the question accordingly.
- You may use temporary relations, if you like, for any of the queries below. If you use the same temporary relation for a second exam question, you must redefine the relation in your answer to the second question. In other words, your answer to each question should be self-contained.

Problem	1	2	3	4	5	Total
Points	10	20	15	20	35	100
Score						
Grader						

Turn over the page when instructed to do so. Good Luck!

I. [*True/False questions, 10 points*] For each of the following statements, indicate whether it is *TRUE* or *FALSE* by circling your choice. You will get *1 point* for each correct answer, *-1 point* for each incorrect answer, and *0 point* for each answer left blank.

1. **True** *False*
   Any candidate key of a relation is a super key for that relation.

2. **True** *False*
   All attributes of a relation together form a super key for the relation.

3. **True** *False*
   A relation decomposition is **not** always lossless.

4. *True* **False**
   One weakness of triggers is that they can only be activated for insertion and update operations.

5. *True* **False**
   The null value approach in translating subclass relationship in the ER model always saves space

6. *True* **False**
   The default policy for the foreign-key constraint sets the value of a foreign-key to null when the tuple of its referenced attribute gets deleted.

7. *True* **False**
   Join is one of the five basic operation in relational algebra.

8. *True* **False**
   In a relation $R(X, Y, Z, T)$, if $XY \rightarrow YZ$ then $X \rightarrow Z$.

9. *True* **False**
   The primary key of a relation may contain a NULL value as a value for their components.

10. *True* **False**
    The expression "(0 != NULL)" in the WHERE clause of a SQL query is evaluated to be true.

2. ER Diagram and Relational Schema (20 points; 10 points for (a) and (b))
Your task is to design a database for a banking system, which maintains information about customers and their accounts.

(a) Draw an ER diagram to model the application with the following assumptions:

- Each customer has a name, a permanent address, and a social security number.
- Each customer can have multiple phone numbers, and the same phone number may be shared by multiple customers.
- A customer can own multiple accounts, but each account is owned by a single customer.
- Each account has an account number, a type (such as saving, checking, etc), and a balance.
- The bank issues an account statement for each account and mails it to its account owner every month. As time goes on, there will be multiple statements of the same account.
- Each statement has an issued date and a statement ID. All the statements of the same account have different statement IDs, but two different accounts could have statements with the same statement ID. For example, it is possible that account A has a statement with ID '123', while account B has another statement with the same ID '123'.

If you need additional assumptions to complete your diagram, please state your assumptions here as well.

Solution:

The solution is one of the many possible solutions. See Figure 1.

Points breakdown:

- 2 points for reasonable entity sets
- 2 points for reasonable keys
- 2 points for correct relations
- 2 points for correct notations
- 2 points for recognizing statements is a weak entity (depending on your assumptions, phone may be a weak entity as well)

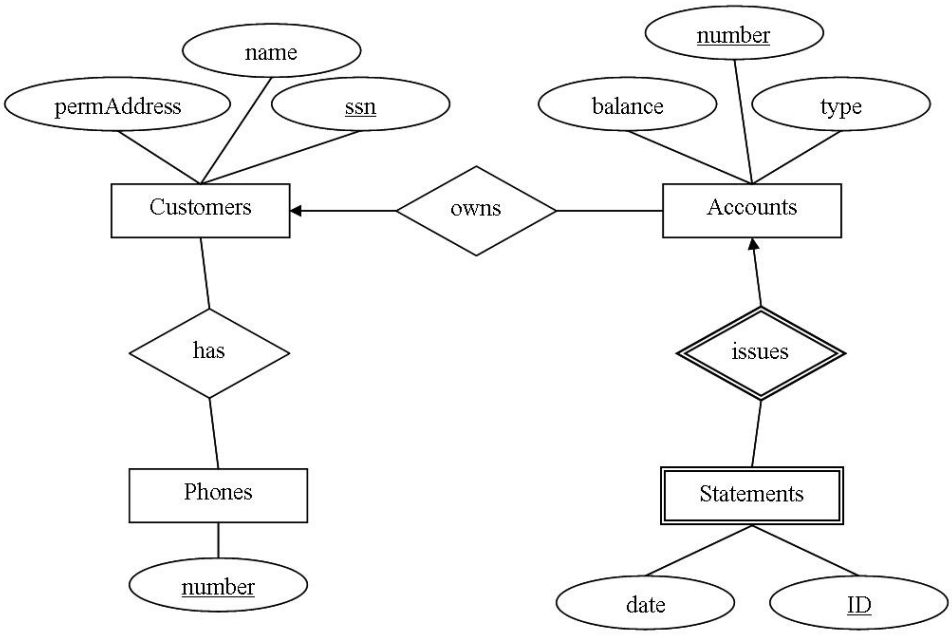


Figure 1: ER Diagram

(b) Translate your ER diagram into a set of relations. Select approaches that yield the fewest number of relations; merge relations where appropriate.

Solution:
Customers(ssn, name, permAddress)
Accounts(number, type, balance, ssn)
Has(ssn, number)
Statements(number, ID, date)
Points breakdown:

- 4 points for translating entity sets and relations correctly
- 4 points for merging various relations correctly and appropriately (ie. merging of many-to-many relations is bad)
- 2 points for correct key identification
- part b) is graded based on part a) answer. No points are taken off for propagation errors from part a).

3. Functional Dependencies and Normal Forms (15 points; 5 points for (a), 2 points for (b) and (c), and 6 points for (d))

Consider a relation $R(A, B, C, D, E)$ with FD's $AB \rightarrow C$, $CD \rightarrow E$, $C \rightarrow A$, $C \rightarrow D$, $D \rightarrow B$.

(a) Determine all the keys of relation R. Do not list superkeys that are not a key.

Solution:

Keys: AB, AD, C

To get the key AB, we can do the following:

- From $AB \rightarrow C$ and $C \rightarrow D$, we obtained $AB \rightarrow D$.
- From $AB \rightarrow C$ and $AB \rightarrow D$, we obtained $AB \rightarrow CD$.
- From $AB \rightarrow CD$ and $CD \rightarrow E$, we obtained $AB \rightarrow E$.

To get the key C, we can do the following:

- From $C \rightarrow A$ and $C \rightarrow B$, we obtained $C \rightarrow AB$.
- From AB, we can obtain the rest of the attributes.

To get the key AD, we can do the following:

- From $D \rightarrow B$, we can get $AD \rightarrow AB$.
- From AB, we can obtain the rest of the attributes.

Points breakdown:

- 3 points for correct keys.
- 2 points for work shown.

(b) List all the FD's that violate 3NF, if any.

Solution: No violations.

Points breakdown:

- no points are taken off for propagation errors.

(c) List all the FD's that violate BCNF.

Solution: Violation: $D \rightarrow B$

Points breakdown:

- no points are taken off for propagation errors.

(d) Decompose R using the BCNF decomposition algorithm. Show your work and summarize your final set of relations.

Solution:

Use the violation $D \rightarrow B$ to decompose the relation.

R1=DB, R2=ACDE

Points breakdown:

- 4 points for correct decomposition
- 2 points for work shown.
- no points are taken off for propagation errors.

4. Relational Algebra (20 points; 10 points for (a) and (b))
Tables 1, 2, and 3 on the next page show an example instance of corporate database for a coalition of retail stores. Those tables maintain information about stores, products, and the inventories of products in different stores. We assume that all the stores sell each product at the same price in Table 2.

Write down queries in relational algebra for the following questions. Do not use cartesian products if you can use joins instead. Please refer to the Store relation, Product relation, and Inventory relation as S, P, and I respectively in your solutions. We here assume that relational algebra supports the set semantics; that is, each operation in relational algebra eliminates duplicate tuples from the output relation.

(a) Return the name of the stores that have at least one product in their inventories whose unit price is greater than 2 USD.

$\pi_{StoreName}\ \sigma_{UnitPrice>2}$ (S $\bowtie$ I $\bowtie$ P)

(b) Return the name of the products with the maximum unit price.

We first find out the products that do not have the maximum price. Let's rename Product relation to P_1 and P_2:

$A = \pi_{ProductName}\ (P_1 \bowtie_{(P_1.ProductName!=P2.ProductName)and(P_1.UnitPrice<P_2.UnitPrie)}\ P_2)$

Then we get the answer by computing the difference between the complete list of the product names and the above relation.

Answer = $\pi_{ProductName}P_1$ - A.

Table 1: Store relation: S

StoreID	StoreName
1	Kmart
2	Walmart
3	Safeway
4	Meijer
5	Schnucks
6	Xmart
7	Ymart

Table 2: Product relation: P

ProductName	UnitPrice(in USD)
Bread	1.6
Cheese-Cheddar	2
Cheese-Feta	4
Cheese-Livarot	9
Cheese-Mozzarella	5
Cucumber	4
Lettuce	1
Onion	2
Potato	2.5
Tomato	2.1
Tuna	0.5
Yam	4

Table 3: Inventory relation: I

ProductName	StoreID
Bread	2
Cheese-Cheddar	4
Cheese-Cheddar	5
Cheese-Feta	7
Cheese-Livarot	4
Cucumber	1
Lettuce	4
Onion	1
Potato	5
Tomato	3
Tuna	5
Yam	3

5. SQL (35 points; 5 points for (a), 10 points for (b), (c), and (d))
Consider the relations given in problem 4. Write down the following queries in SQL. You do not have to worry about duplicate tuples in the output relations of your queries.

(a) Return the names of the stores that have Tomato and Lettuce in their inventories.

```
(Select StoreName
From Store,Inventory
Where Store.StoreID = Inventory.StoreID and Inventory.Product ='Tomato')
INTERSECT
(Select StoreName
From Store,Inventory
Where Store.StoreID = Inventory.StoreID and Inventory.Product ='Lettuce')
```

(b) Return the names of the products whose unit prices are below the average unit price of all the products.

```
Select ProductName
From Product
Where UnitPrice <
( Select Avg(UnitPrice)
  From Product)
```

(c) Each store has an *average unit price*. It is the average of the unit prices of the products available at the store. Return the average unit prices of all stores. Your query should produce pairs of store names and their average unit prices.

```
Select StoreName, Avg(UnitPrice)
From Store, Inventory,Product
Where Store.StoreID = Inventory.StoreID and Inventory.ProductName = Product.ProductName
GroupBy StoreName
```

(d) Return the product(s) that are available in at least three stores.

```
Select ProductName
From Inventory
GroupBy ProductName
Having  (Count(distinct StoreID) >= 3)
```

CS411 Database Systems

Spring 2010, Prof. Chang

Department of Computer Science
University of Illinois at Urbana-Champaign

Midterm Examination

March 2, 2010

Time Limit: 75 minutes

- Print your name and NetID below. In addition, print your NetID in the upper right corner of every page.

Name: _____ **NetID:** _____

- Including this cover page, this exam booklet contains **9** pages. Check if you have missing pages.
- The exam is closed book and closed notes. No calculators or other electronic devices are permitted. Any form of cheating on the examination will result in a zero grade.
- Please write your solutions in the spaces provided on the exam. You may use the blank areas and backs of the exam pages for scratch work. Please do not use any additional scratch paper.
- Please make your answers clear and succinct; you will lose credit for verbose, convoluted, or confusing answers. *Simplicity does count!*
- Generally, we think one minute per point is a reasonable allocation of time; so plan your time accordingly. *You should look through the entire exam before getting started, to plan your strategy.*

Problem	1	2	3	4	Total
Points	36	28	18	18	100
Score					
Grader					

Problem 1 (*36 points*) Misc. Concepts

For each of the following statements:

- for true/false choices, indicate whether it is *TRUE* or *FALSE* by circling your choice, and provide an explanation to justify;
- for short answer questions, provide a brief answer with clear explanation.

You will get *3point* for each correct answer with correct explanations, and ***no penalty* (of negative points) for wrong answers**.

Notice: To get full credit, you need to provide both correct answer, and correct explanations. If you simply choose True or False without explaining, you will automatically lose 1 point for the question. Moreover, since the questions here are true/false choices, or short answer questions, no partial credit is given for incomplete answers.

(1) *False*

An E-R diagram with m entities and n relationships will translate to $m+n$ tables.

$\Rightarrow$ *Explain*: Many-to-one relationships can often be merged and therefore reduce the overall number of tables needed.

(2) *True*

A table with two attributes, such as $R(A, B)$, must be in BCNF.

$\Rightarrow$ *Explain*: The only two non-trivial FDs for this relation will be $A \rightarrow B$ or $B \rightarrow A$, and both does not violate BCNF.

(3) *True* or *False*

An SQL query can often be translated into *multiple* relational algebra expressions.

$\Rightarrow$ *Explain*: The answer to this question can be true or false based on different assumptions. Our grading takes your assumption into consideration, and grades accordingly.

(4) Give a relation that is in 3NF but not in BCNF.

$\Rightarrow$ *Explain*: A relation R(A,B,C) with two FDs: $AB \rightarrow C$ and $C \rightarrow A$. The second FD violates BCNF, since C is not a superkey. However, it does not violate 3NF as A is part of a key (AB).

(5) *True*
In relational algebra, join is a *derived* operator.

$\Rightarrow$ *Explain*: It can be derived from Cartesian product and selection.

(6) *False*
Schema normalization often contributes to enhancement of query processing efficiency.

⇒ *Explain*: Schema normalization is mainly for reducing redundancy, rather than for improving query efficiency. In fact, it often introduces more query processing, as decomposed tables have to be joined online for answering queries.

(7) *True*
For relation $R(A, B, C)$, if A is a key, then the decomposition into $R(A, B)$ and $S(A, C)$ is lossless.

⇒ *Explain*: You can always reconstruct the original table (A,B,C) by joining (A,B) with (A,C), since A is a key. (many of your wrong answers give examples where A is not a key)

(8) *True*
In the following relation $R(A, B, C, D)$, the F.D. $AC \rightarrow B$ is not violated.

A	B	C	D
1	3	2	2
2	3	2	4
3	1	3	6
3	1	1	12

⇒ *Explain*: In the 4 tuples given, $AC \rightarrow B$ is not violated. You will also get full credit, if you choose False and argue that FD dependence can not be inferred from a set of given tuples (and should rather be derived based on the property of the relation).

(9) In the above relation $R(A, B, C, D)$, what are *possible* keys of the table?

⇒ *Explain*: Possible keys are D, or AC. You will also get full credit, if you argue we can not inferred keys from the set of 4 tuples.

(10) Provide a lossless-join decomposition of the above relation $R(A, B, C, D)$ (into to two relations). If there is no such decomposition, explain why.

⇒ *Explain*: You can decompose it into (ABC) and (ACD), just based on the 4 tuples given in the table. You will also get full credit, if you argue no such decomposition is possible, since we do not know the FDs for the relation.

(11) For the above relation $R(A, B, C, D)$, write a relational algebra expression to return the lowest value of D.

⇒ *Explain*: $\pi_D R - \pi_{R1.D}(R1 \bowtie_{R1.D>R2.D} R2)$ where R1 and R2 are just renames of R.

(12) What is the result of the following query, for the above relation $R(A, B, C, D)$?

```
select A, B
from R
where C >
      (select D from R where A = 3)
```

⇒ *Explain*: This query will report error. The reason is the subquery returns a scalar, and therefore can not be compared against a single data value. You would need to add "ANY" or "ALL" before the subquery for the query to run.

Problem 2 (*28 points*) Database Design

You have been asked to design a database for the university administration, which records the following information:

1. All students necessarily have a unique student ID, a name, and a university email address. Each student is also either an undergraduate or a graduate student.
2. Each graduate student has an advisor.
3. Each undergraduate student has a major.
4. Students take courses. A student may take one course, multiple courses, or no courses.
5. Each course has a course number, course name, and days of the week the course is scheduled.
6. Each course has exactly one head TA, who is a graduate student.
7. Every head TA has an office where he or she holds office hours.

(a) Draw an ER diagram for this application. Be sure to mark the multiplicity of each relationship of the diagram. Decide the key attributes and identify them on the diagram. Please state all assumptions you make in your answers. (*16 points*)

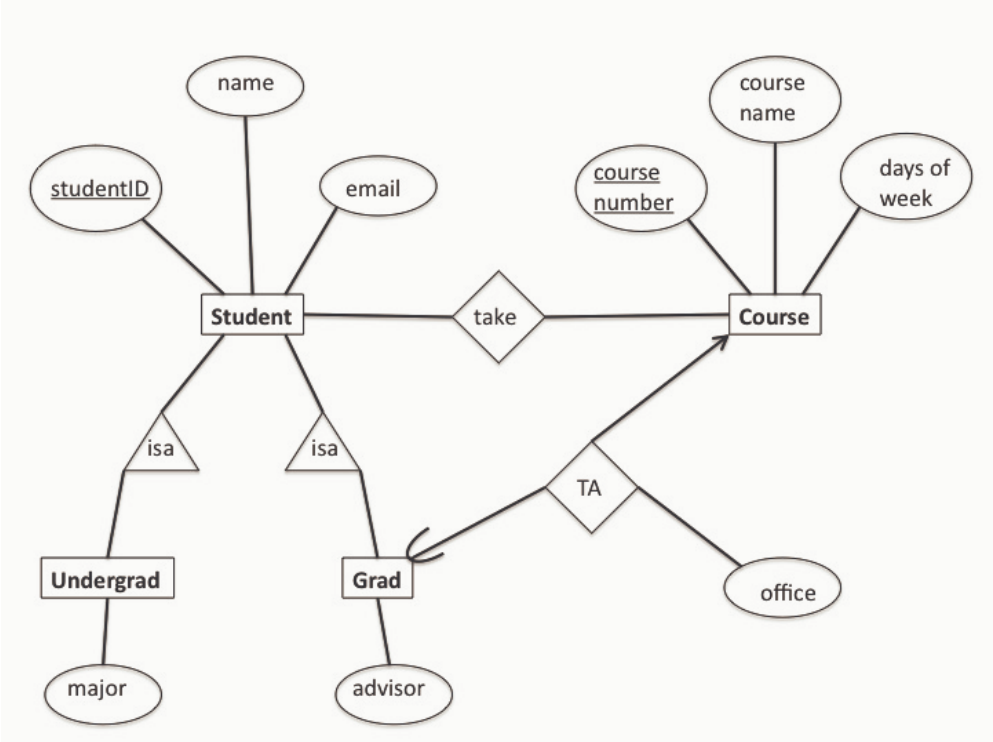


Figure 1: ER Diagram

(b) Translate your ER diagram into a relational schema. Select approaches that yield the fewest number of relations; merge relations where appropriate. Specify the key of each relation in your schema. (*12 points*)

Undergrad(<u>studentID</u>, name, email, major)
Grad(<u>studentID</u>, name, email, advisor)
TakeCourse(<u>studentID</u>, <u>course_number</u>)
Course(<u>course_number</u>, <u>studentID</u>, course_name, days_of_week, office)

Problem 3 (*18 points*) Relational Algebra

Nowadays, web search engine is widely used by people all over the world to find useful information. To analyze users' search behaviors, a *query log* stores the history of users' queries and clicked URLs.

There are two relations in a query log:

- $Request(Time, UserID, Query, Results)$, which lists the time, the userID, the query he/she requested, the number of search results for one search process.
- $Click(Time, UserID, Query, URL)$, which lists the time, the userID, the query he/she requested and the URL clicked on after request this query.

For both *Request* and *Click* relation, the attributes $(Time, UserID)$ forms the only key which implies one user can request only one query or click on only one URL each time.

Example instances of these two relations are given here:

Request Relation

Time	UserID	Query	Results
2010-02-12 19:00:00	U001	NBC	100
2010-02-12 19:01:00	U001	NBC Olympics	50
2010-02-12 18:54:00	U002	NBC 2010	80
2010-02-13 09:00:05	U001	Olympics wiki	70
2010-02-15 19:27:08	U003	Olympics 2010 medals	5
2010-02-16 13:24:45	U004	NBC Olympics	50
2010-02-16 13:25:55	U004	Vancouver 2010	95
2010-02-16 17:11:56	U002	NBC 2010	80
2010-02-20 20:13:45	U005	Olympics ski	54
2010-02-20 20:45:34	U005	Olympics Austria medals	7

Click Relation

QueryID	UserID	Query	URL
2010-02-12 19:01:06	U001	NBC Olympics	www.nbcolympics.com
2010-02-12 19:01:34	U001	NBC Olympics	www.nbcolympics.com/video
2010-02-12 18:54:01	U002	NBC 2010	www.nbcolympics.com
2010-02-15 19:27:22	U003	Olympics 2010 medals	www.vancouver2010.com
2010-02-15 19:29:01	U003	Olympics 2010 medals	www.vancouver2010.com/olympic-medals/
2010-02-16 13:25:58	U004	Vancouver 2010	www.vancouver2010.com
2010-02-16 17:12:56	U002	NBC 2010	www.nbcolympics.com
2010-02-20 20:14:00	U005	Olympics ski	www.usskiteam.com
2010-02-20 20:45:40	U005	Olympics Austria medals	en.wikipedia.org/wiki/Austria_at_the_Olympics
2010-02-20 20:45:50	U005	Olympics Austria medals	www.vancouver2010.com

Note that not all the queries have corresponding clicked URLs since a user might not be interested in any returned URL. Sometimes user may click on multiple URLs using one query.

Though different from the actual case, you can assume that the number of returned search results associated with the same query doesn't change no matter when and who request this query.

Answer the following queries using relational algebra. Your answer should work for any instance of the database, not just this one.

(a) List all the queries that do not have any corresponding URL. (*8 points*)

Solution:
$\pi_{Query}Request - \pi_{Query}Click$

(b) List all the queries that have been requested by *different* users. (*10 points*)
Solution:
$\pi_{Request_1.Query}(Request_1 \bowtie_C Request_2)$
where, $Request_1$ and $Request_2$ are both type of $Request$
$C = (Request_1.Query = Request_2.Query$ AND $Request_1.UserID <> Request_2.UserID)$

Problem 4 (*18 points*) SQL

Consider the same schema as Problem 3. Answer the following queries using SQL. Your statements should work for any instance of the database, not just this one. You do not need to remove duplications in your results.

(a) List all the users (userID) who either requested the query "NBC 2010" or clicked on the URL "www.nbcolympics.com." (*8 points*)

Solution:

```
select UserID from Request where Query = "NBC 2010"
union
select UserID from Click where URL = "www.nbcolympics.com"
```

(b) Return the query whose corresponding URLs contain "www.vancouver2010.com" and, among these queries, has the minimum number of returned search results. (*10 points*)

Solution:

```
select Request.Query from Request, Click
where Request.Query = Click.Query
and Click.URL = "www.vancouver2010.com"
and Results <= All
(select Results from Request, Click
where Request.Query = Click.Query
and Click.URL = "www.vancouver2010.com")
```

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:_____

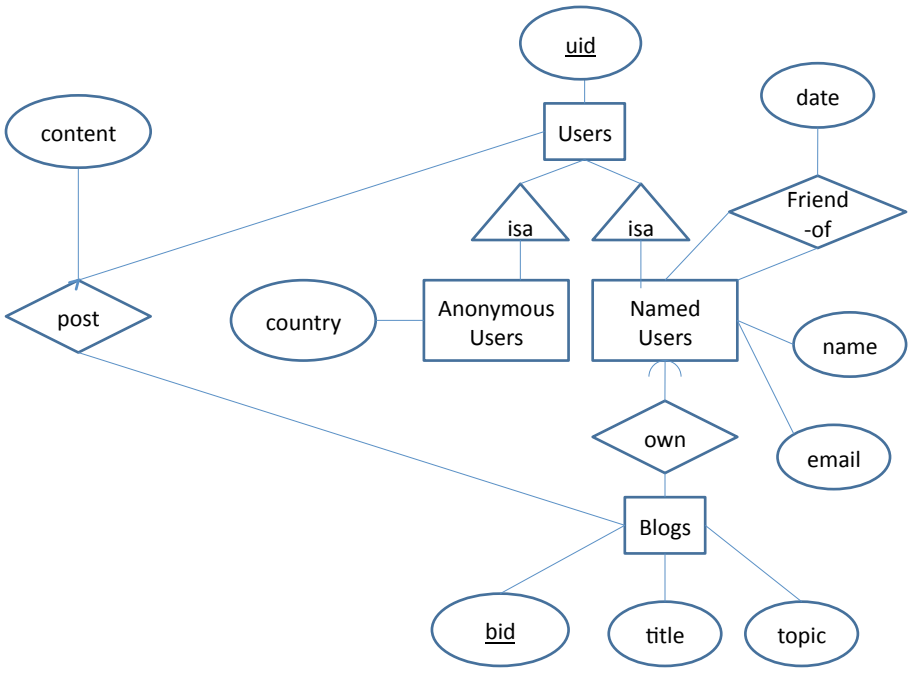
2. (**30** points) **Conceptual Design**

   (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 $\rightarrow$ E and A $\rightarrow$ 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(A,B) and R2(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(C, D, E) and R4 (A, C, D)

End result: R1(A,B), R3(C, D, E), and R4 (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.

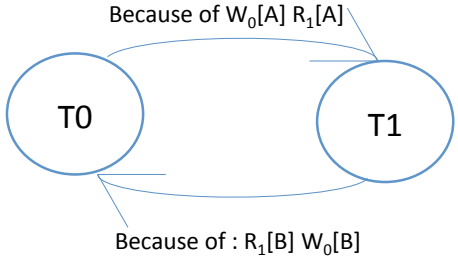
Solution:
The UNDO pass scans backward from the end of the log. The pass undoes the updates by all transactions that had not committed by the time of the crash. These transactions can be found in the Transactions Table rebuilt during the analysis pass. All updates are undone unconditionally (since the 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
$L_0[A]$	
$r_0[A]$	
$w_0[A]$	
	$L_1[A] \rightarrow blocks$
$L_0[B]$	
$r_0[B]$	
$w_0[B]$	
$U_0[A]$	
$U_0[B]$	
c_0	
	$L_1[A] \rightarrow granted$
	$r_1[A]$
	$L_1[B]$
	$r_1[B]$
	$U_1[A]$
	$U_1[B]$
	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	

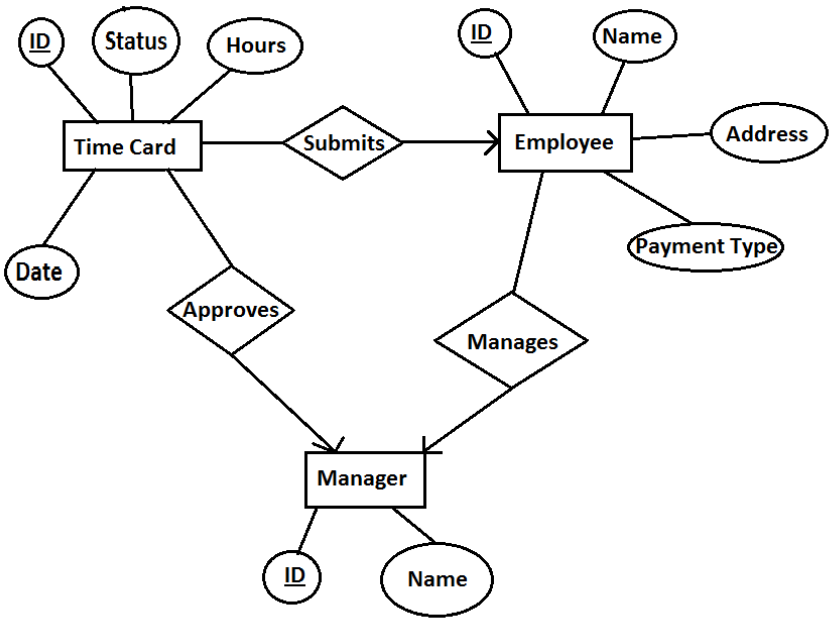
Problem 1: ER Diagram Design

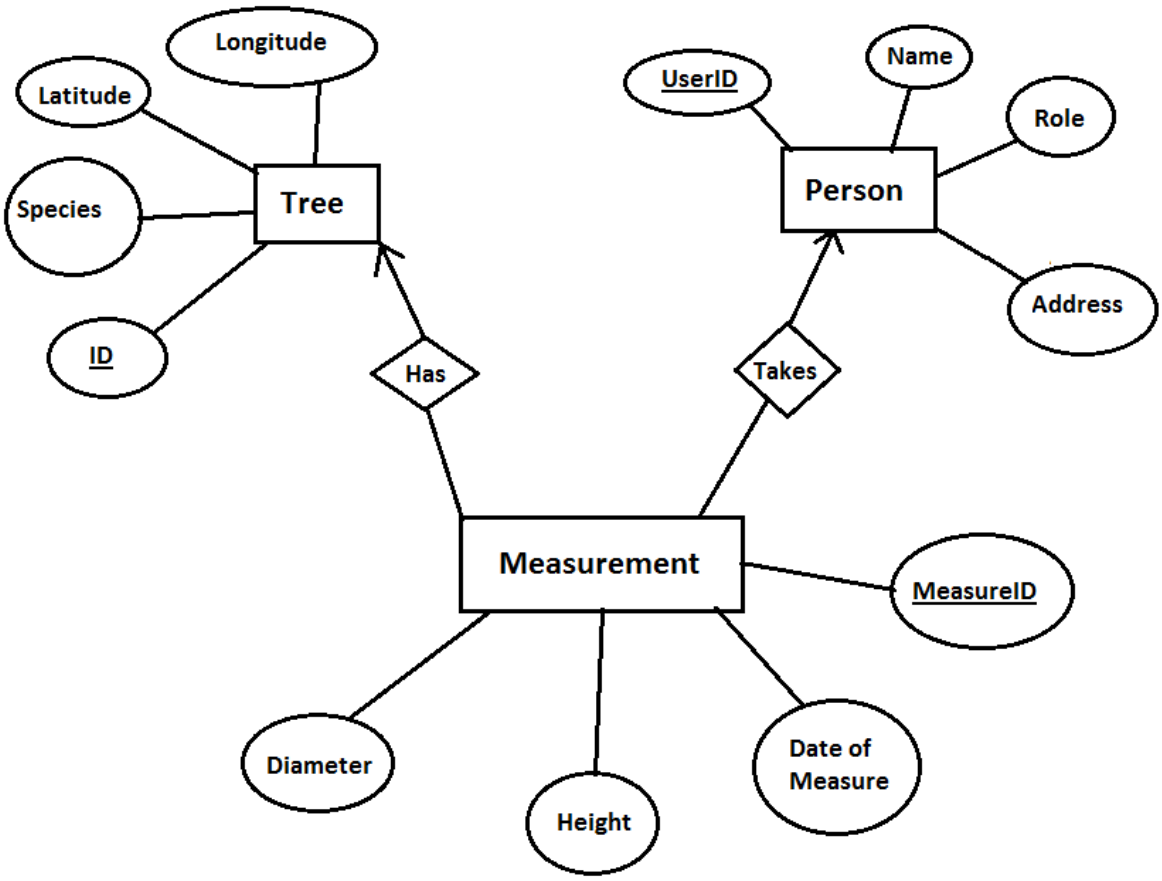
The company you work for wants to digitize their time cards. You have been asked to design the database for submitting and approving time cards. Draw the database ER diagram with the following information:

- A *timecard* should have hours worked and date submitted
- Each *timecard* is associated with exactly one *employee*
- Each *timecard* should have a unique id
- Each *timecard* has a status: it is either approved, not approved, or pending
- Each *employee* has a unique id
- Each *employee* has a name and address.
- Each *employee* submits a time card every pay period. i.e. In 1 year, they will submit multiple time cards
- Each *employee* either has direct deposit or physical check as their method of payment
- Each *employee* is associated with exactly one *manager*
- Each *manager* has a unique id and a name
- Each *manager* is in charge of multiple employees
- Each *manager* approves time cards for multiple employees

If you feel that you must make some assumptions, please state them clearly so that they are easily understood by the graders. Remember to indicate the key for each entity, as well as the multiplicity of each relationship (e.g. one-to-many) using the appropriate notation.

Solution:





Use the above diagram for the following questions!

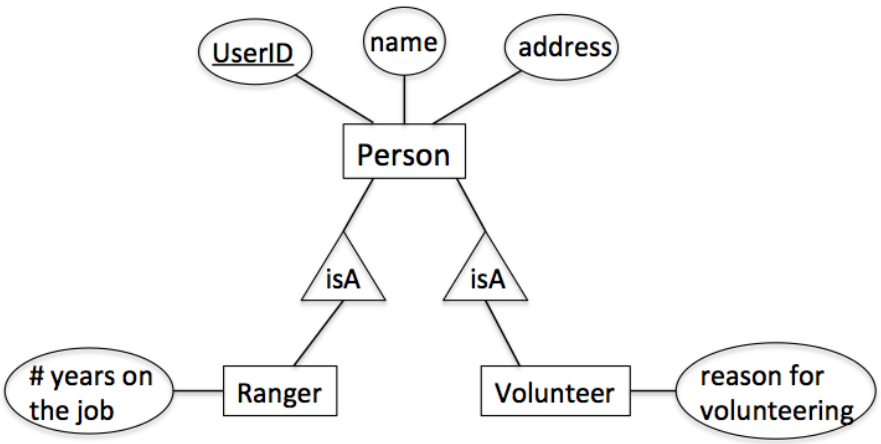
Problem 2: More ER Fun

Answer the following questions based on the original ER Diagram in Figure 1:

1. Could the "Date of Measure" attribute of the *Measurement* entity be the key for the entity, instead of the "MeasureID"? Why or Why Not?
2. We want to add a "Tools Used" attribute, which will store the tool(s) used to make measurements, but are not sure where this attribute belongs. We do know that one person might use multiple tools for different trees, and also that more than one tool may be used to measure the same tree (for example, different tools may be necessary to measure the same tree in the summer, than in the winter.) Where could we add this attribute? Choose *one or more* answers from the 3 entities and 2 relationships (we would add the attribute only once, but if you believe there are multiple possible places to add the attribute, we ask you to identify all potential candidates):
    a. the entity Person
    b. the relationship Has
    c. the entity Measurement
    d. the relationship Takes
    e. the entity Tree
3. There are two roles that people can have, a Ranger and a Volunteer. Using what you know of subclassing, add these as two new entities in the ER Diagram. Add a reasonable attribute to each of the new entities. Just redraw the relevant part of the diagram that needs to change. Can we eliminate the **role** attribute from*Person*?

Solution

1. It would not be a good key because there may be many measurements made on the same day. Therefore this attribute is not unique, and would not make a good key.
2. The relationship Has, the entity Measurement, and the relationship Takes are all correct answers (b, c, d). The entity Person and the entity Tree are not (a, e).
3. See Diagram. Any appropriate attributes to the subclasses can be accepted. You can eliminate the role type attribute. However, we may not want to, as keeping the payment type attribute may facilitate some queries, and may make creating new payment types easier.



Problem 3: ER Diagram Translation

Use the original ER Diagram in Figure 1 for this problem (without any changes you may have made during your work in problem 2).

1. Translate the ER diagram to a relational design. Try to minimize the number of relations your solution has, and merge relations where appropriate. Don't forget to specify the keys.
2. Assume now that we have changed the ER diagram so that the Takes relationship is now many-many, with the interpretation that it might take several people to determine a single measurement. How will this change the relational design?

Solution:

1. Person(UserID, Name, Role, Address), Tree(TreeID, Latitude, Longitude, Species), Measurement(MeasureID, Date_of_Measure, Height, Diameter, TreeID, UserID) (because Measurements had a 1-many relationship with both Person and Tree, the Measurement table can store those relationships).
2. Now the many-many relationship needs its own table, Takes(UserID, MeasureID) and the UserID column would be removed from the Measurement table. *6 points*

Problem 4: SQL Statements

Grading: 25 points

Use the original ER Diagram in Figure 1 for this problem (without any changes you may have made during your work in problems 2 or 3).

1. Write the SQL command that would define each table. Your definition must include correct data types with correct sizes for each field, key and unique declarations, and NULL constraints for fields. Please assume the following data types and lengths:
    a. the three ID attributes are all of type char, length 10
    b. Latitude and Longitude are of type decimal, precision value 10, scale value 6
    c. Height and Diameter precision value 10, scale value 6
    d. DateOfMeasure is of type date
    e. the rest of the attributes are all of type varchar, length 50
2. Write the SQL Command to insert one tuple of data into each table. The data you insert can be of your choosing, but must adhere to the data types and constraints of the tables.
3. We have decided that the Address property is no longer relevant. We do however wish to be able to email staff members. Change the Person table to remove the Address field and add an Email field.
4. Write a SQL Statement to add an Email address to the person you inserted into the Person table, now that the table has an Email field and not an Address field.

Solution:

1. CREATE TABLE Person(UserID char(10) PRIMARY KEY, Name varchar(50) NOT NULL, Address varchar(50) NOT NULL, Role varchar(50));
CREATE TABLE Tree(ID char(10) PRIMARY KEY, Species varchar(50) NOT NULL, Latitude Decimal(10,6) NOT NULL, Longitude Decimal(10,6) NOT NULL);
CREATE TABLE Measurement(MeasureID char(10) PRIMARY KEY, DateOfMeasure date NOT NULL, Height Decimal(10,2) NOT NULL, Diameter Decimal(10,2) NOT NULL);
9 points, 3 points per Create statement. Only the primary keys need to be specified as NOT NULL, the rest are ok either way

2. INSERT INTO Person (UserID, Name, Address, Role) VALUES ('12345', 'John Smith', '123 Science Lane, Urbana, IL', 'Ranger');
INSERT INTO Tree(ID, Species, Latitude, Longiture) VALUES ('987', 'Fir', 9.353234, 39.234562);
INSERT INTO Measurement(MeasureID, DateOfMeasure, Height, Diameter) VALUES ('432', '2011-01-30', 15.50, 3.10);
6 points, 2 points per Insert Statement

3. ALTER TABLE Person DROP Address;
ALTER TABLE Person ADD Email varchar(50);
5 points

4. UPDATE Person SET Email = 'smithj@treeplace.com' WHERE UserID = '12345';

Problem 1: ER Diagram

You have been hired to develop an account management system for a company. The company sells packages that allow consumers to download books in either text or audio format. Draw the database ER diagram with the following information:

- A *package* allows the consumer to download a given number of books or audiobooks a month, depending on the package type.
- Each *package* has a unique *id*.
- Each *package* also must have a *type*, noting what kind of downloads it provides: either books or audiobooks but not both.
- Each *package* also must have a *tier*, representing the number of monthly downloads allowed by this package (e.g. “Tier10” means 10 monthly downloads are allowed, "Tier50" means 50 monthly downloads are allowed, etc.)
- Each *account* has a unique *id*.
- Each *account* has one or more *packages*.
- Each *account* is associated with one or more *credit cards*
- Each *credit card* is associated with exactly one *account*.
- Each *credit card* must have a short nickname to identify it to its account (e.g. “card1”, “visa”), a *credit card number*, and an *expiration date*
- Each *account* is owned by exactly one *customer*
- Each *customer* must have a customer *id*, a *name*, and one contact *email*, and may also specify one contact *phone number*.

Problem 2: ER Diagram continued

Answer the following questions based on your diagram from Problem 1:

1. Would expiration date be a good key for the Credit Card entity? Why or why not?
2. For the Package entity, which attributes must not be null? What about for the Customer entity?
3. There are two types of packages: book and audiobook. Using your knowledge of subclassing, introduce these as two new entities in the ER diagram. Add a reasonable attribute to each of the new entities. Can we eliminate the type attribute of the Package entity? Just redraw the relevant part of the diagram that needs to change.

Problem 3: ER Diagram translation

Answer the following questions based on your original diagram from Problem 1:

1. Translate the ER diagram from Problem 1 to a relational design. Try to minimize the number of relations your solution has, and merge relations where appropriate. Don't forget to specify the keys.
2. Translate the ER diagram from Problem 2 question 3 (only the Package entity and the two new entities) to a relational design using OO, ER, and NULL value approaches. Don't forget to specify the keys.

Problem 4: SQL commands - table and field definitions

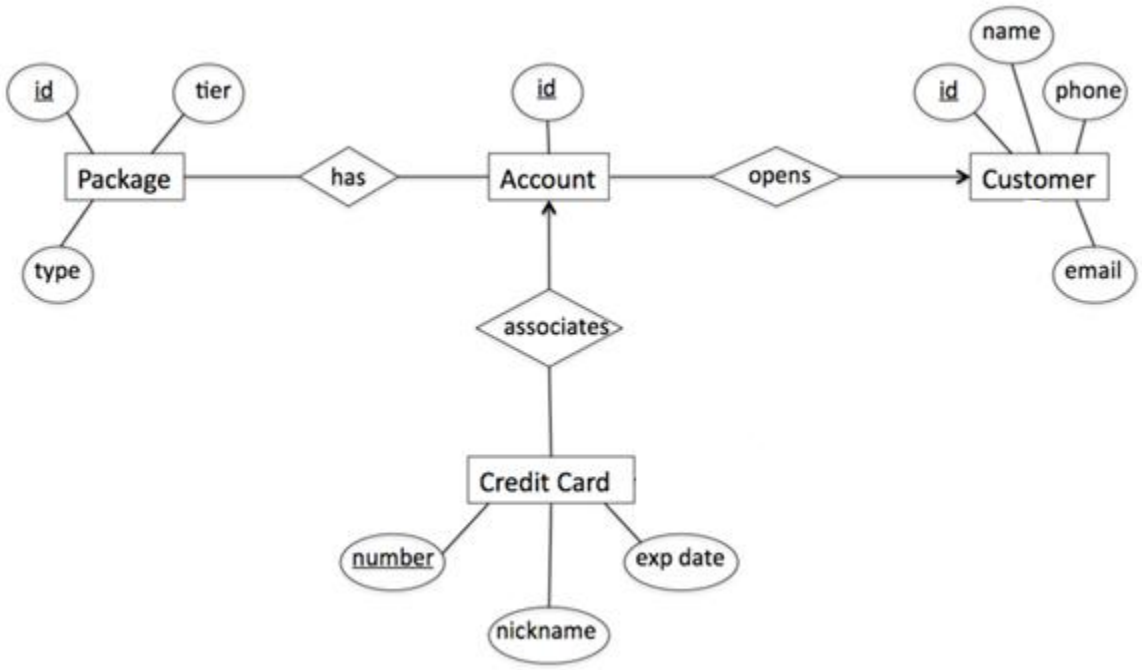
Answer the following questions based on your relational design from Problem 3. The response to each question should be a valid SQL command:

1. Write the SQL command that would define each table. Your definition must include appropriate data types with reasonable sizes for each field, key and unique declarations, and NULL constraints for fields:
    1. Customer
    2. Account
    3. Credit Card
    4. Package
2. Insert one tuple into the Customer table. You can choose arbitrary values for the data fields of your tuple but they must conform to the table's defined constraints.
3. Rename the nickname field in the Credit Card table to cardID.
4. Change the type field in the Package table to have a default value of book, represented by the value 'B'.

Homework 1 Solution

Problem 1: ER Diagram

Solution



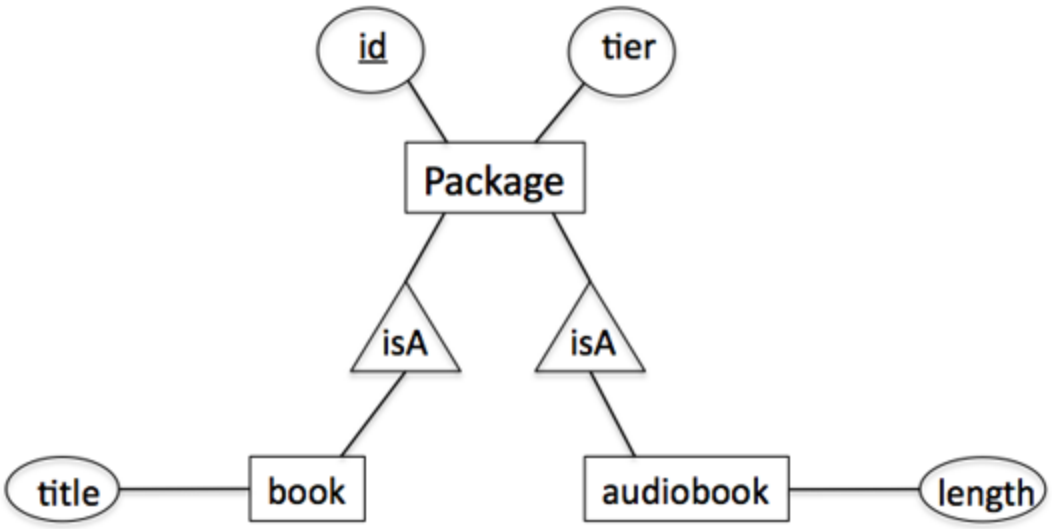
Problem 2: ER Diagram continued

Solution

1. No - multiple credit cards may have the same expiration date.

2. Package: id, type, tier. Customer: id, name, email.

3. See diagram. Yes, we could eliminate the type attribute. However, we may not want to, as keeping the type attribute may facilitate some queries, and may make creating new types easier.



Reasonable attributes for book: title, author, genre, number of pages, etc; for audiobook: title, author, genre, length, etc. Any attribute that makes sense is acceptable.

Problem 3: ER Diagram translation

Solution

1. Relational design: Package(id, type, tier), Card(nickname, number, exp, account_id), Account(id, customer_id), Customer(id, name, email, phone), HasPackage(account_id, package_id)

2. If you chose to remove the type attribute:

- OO: Package(id, tier), PackageBook(id, tier, title), PackageAudioBook(id, tier, length)
- ER: Package(id, tier), PackageBook(id, title), PackageAudioBook(id, length)
- Null: Package(id, tier, title, length)

If you chose not to remove the type attribute:

- OO: Package(id, type, tier), PackageBook(id, type, tier, title), PackageAudioBook(id, type, tier, length)
- ER: Package(id, type, tier), PackageBook(id, title), PackageAudioBook(id, length)
- Null: Package(id, type, tier, title, length)

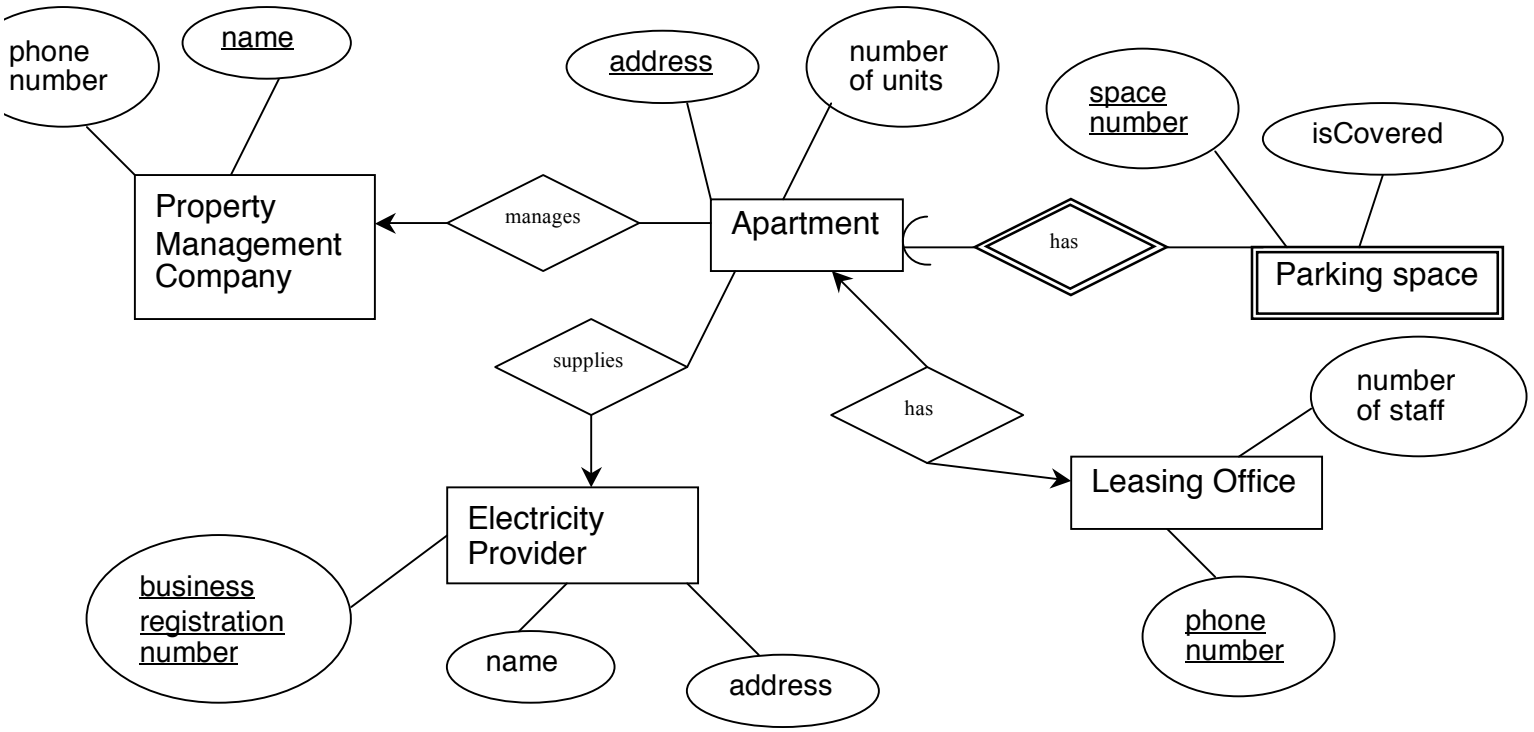
(title and length were the attributes chosen for book and audiobook for this solution. If you chose different attributes, just replace title and/or length in the answers with the attributes you chose)

Problem 4: SQL commands - table and field definitions

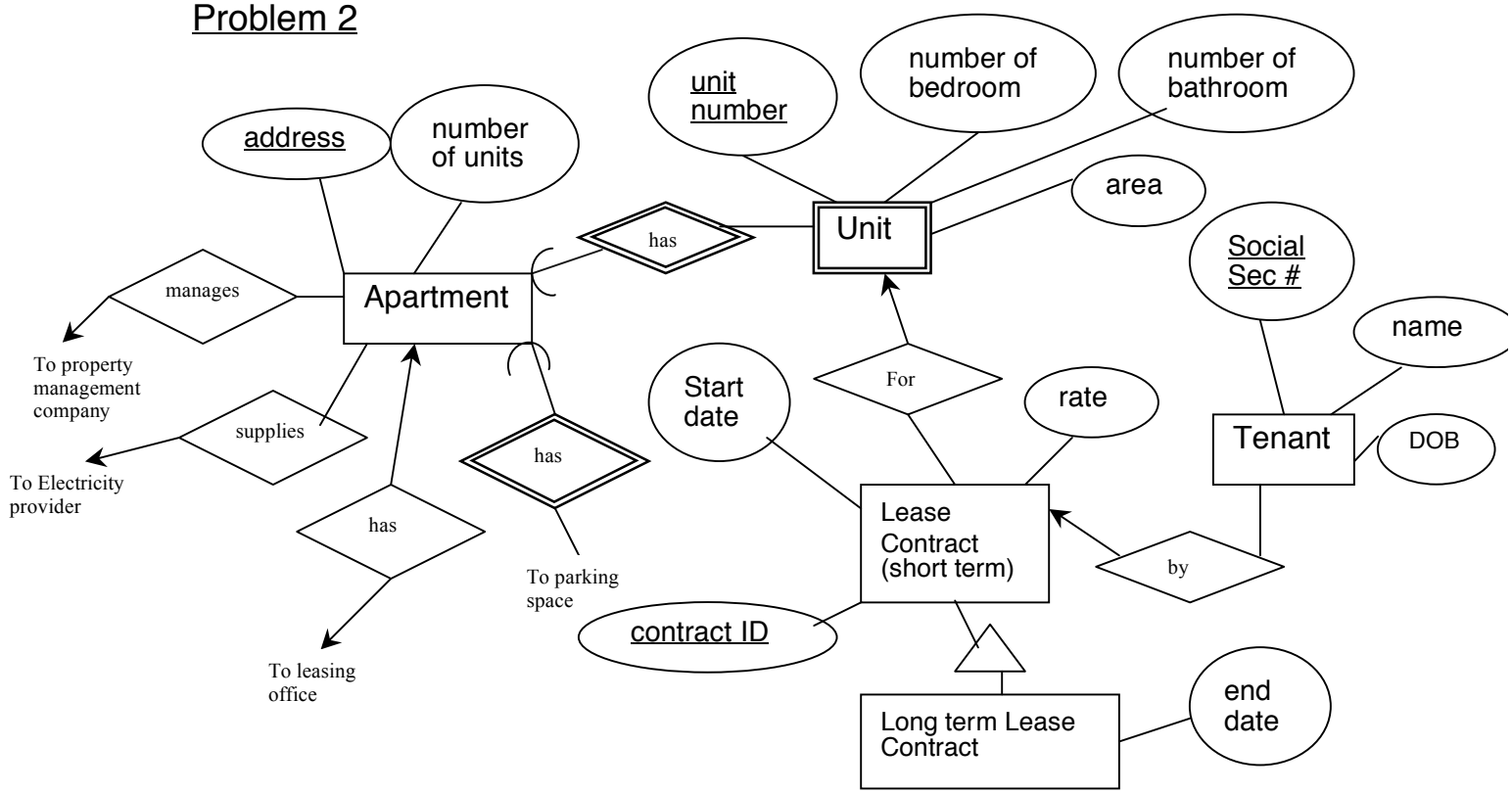
Solution

1.
    1. Create table Customer(id char(10) Primary Key, name varchar(50) not null, email varchar(50) not null, phone char(10));
    2. Create table Account(id char(10) Primary Key);
    3. Create table CreditCard(number char(16) Primary Key, nickname varchar(20) not null, expDate date not null);
    4. Create table Package(id char(10) Primary Key, type char(1) not null, tier varchar(10) not null);
2. Insert Into Customer(id,name,email,phone) Values('1234567890', 'John Doe', 'veryoldemail@hotmail.com', '9876543210');
3. Alter table CreditCard Drop nickname; Alter table CreditCard Add cardID varchar(20);
4. Alter table Package Drop type; Alter table Package Add type char(1) Default 'B';

Problem 1



Problem 2



Problem 3

a) Convert the E/R diagram of Problem 1 to a relational database schema. Don't forget to indicate the keys (by underlining them) for each relation. Note that you don't need to combine the relations for removing duplications in this problem. (4 points)

ANSWER:

Property Management Company(name, phone number)
Apartment(address, company name, electricity provider reg num, number of units, leasing office phone number, leasing office number of staff)
Parking space(apartment address, space number, isCovered)
Electricity Provider(Business reg num, name, address)

b) Convert the Leasing contract, long term contract, and monthly contract (Note you may use different names other than ones given here, as long as you make it clear for graders to understand) relationships in Problem 2 into relational schema, using the following three approaches: 1. ER approach, 2. Object oriented approach, 3. and Null approach. (6 points, 2 points for each approach)

ANSWER:

ER approach:
Short term lease contract(contract ID, rate, start date)
Long term lease contract(contract ID, end date)

OO approach:
Short term lease contract(contract ID, rate, start date)
Long term lease contract(contract ID, rate, start date, end date)

Null approach:
Lease contract(contract ID, rate, start date, end date)

Problem 4 Basic Concepts of Keys (25 points, part b 10 and other parts 5 points)

a) Given a relation R, explain how you can use the closure to test if a subset of attributes A_1, A_2, ..., A_n is a candidate key for R.

ANSWER: By first checking that the closure of A_1, A_2, ..., A_n (i.e. { A_1, A_2, ..., A_n}+ is all attributes of the relation R, and then checking that no subset of A_1, A_2, ..., A_n is all attributes of R.

b) Suppose R is a relation with attributes A_1, A_2, ..., A_n. As a function of n, tell how many superkeys R has, if
(1) the only keys are A_1 and A_2, and

ANSWER: Number of combinations of attributes that contains {A1}, {A2} or {A1,A2}: 3 * 2^{n-2}

c) Show that each of the following are not valid rules about FD's by giving example relations that satisfy the given FD's (following the "if") but not the FD that allegedly follows (after the "then"). Explain your answer and state any assumption you make.
1. If A → B then B → A
2. If AB → C then A → C or B → C

ANSWER:

1.

StudentID	Surname	First name
12335	Jones	James
56932	Jones	Bridget

Surname is functionally dependent on StudentID, but more than one student may share a same surname.

2.

SupplierID	ProdID	Quantity
1	4902	300
1	9088	10
2	4902	500

Multiple suppliers (of some product) may supply the same product with varying quantity. SupplierID and ProdID are both required to uniquely identify the quantity supplied.

d) Consider a relation with schema R(A,B,C,D) with functional dependencies (FD's): BC → A, AD → B, CD → B, AC → D.
Find all the candidate keys of R.

ANSWER: BC, CD, AC

Problem 5 Functional Dependency (20 points, 5 points each)

Consider a relation with schema R(A,B,C,D) and FDs AC → B, B → A, BD → C and D → A.
a) Describe the concept of Functional Dependency and Dependency-preserving decomposition.

ANSWER:

Functional dependency can be thought of as a generalization of the idea of a key for a relation. It states that if 2 tuples of a relation agrees on the attributes $A_1A_2…A_n$, then they must also agree on attributes $B_1B_2…B_n$, written $A_1A_2…A_n$ -> $B_1B_2…B_n$. Simply put, it means attributes $A_1A_2…A_n$ functionally determines attributes $B_1B_2…B_n$.

Dependency-preserving decomposition is a decomposition of a relation R into R1, R2…,Rn according to some FD where all FDs that hold on R also hold on all "sub-relations" R1, R2…,Rn.

b) Find the closures for subsets A, BD and BC respectively.
ANSWER:
{A}+ = {A}
{BD}+ = {ABCD}
{BC}+ = {ABC}

c) List all nontrivial FD's that follow from the given FD's.
ANS: BD -> A, CD -> A, CD -> B, CD -> AB, ABD -> C, BCD -> A, ACD -> B, BC -> A, BD -> AC

d) Find all the candidate keys for this relation (you don't need to list superkeys that are not keys). Explain your answer.
ANSWER: BD, CD since {B,D} and {C,D} are both minimal set of attributes whose closure contains all attributes {A,B,C,D,E}

Problem 6 Normalization and Schema Design (20 points, 5 points each)
Consider a relation R(A, B, C, D), with FDs AB → C , BC → D, CD → A.
a) Is R in BCNF? Explain why or why not.
ANSWER: No, R is not in BCNF since the functional dependency CD->A violates the condition; CD is not a superkey of R.

b) We are considering to decompose R into R1 (A, B, C) and R2 (A, C, D). Is this a lossless decomposition?
ANSWER: No.
Assume R has the following tuples to start with:

A	B	C	D
1	2	3	4
1	4	3	2

After decomposition, R1 and R2 will have the following tuples:

R1:

A	B	C
1	2	3
1	4	3

R2:

A	C	D
1	3	4
1	3	2

Joining R1+R2 results in R' as follows:

R':

A	B	C	D
1	2	3	4
1	2	3	2
1	4	3	4
1	4	3	2

Since R is not the same as R' (2nd and 3rd tuples are introduced after join), this decomposition is not lossless.

c) Provide the BCNF normal form for this relation
ANSWER:
Decomposing using CD->A
R1(ACD), R2(BCD)

d) Provide the 3rd normal form for this relation.
ANSWER: R(ABCD) is in 3NF

Name:_____ **NetID:** _____

MIDTERM EXAM, March 9, 2006
CS 411 Database Systems
Department of Computer Science
University of Illinois at Urbana-Champaign

Exam Rules:

1) Close book and notes, **75 minutes**, scratch papers are allowed.

2) Please write down your name and NetID number NOW.

3) Please wait until being told to start reading and working on the exam.

4) No question can be asked during the exam (due to departmental regulations, to be fair to off-campus students). If you think a problem is ambiguous, write down your assumptions, argue that they are reasonable, then work on the problem using those assumptions.

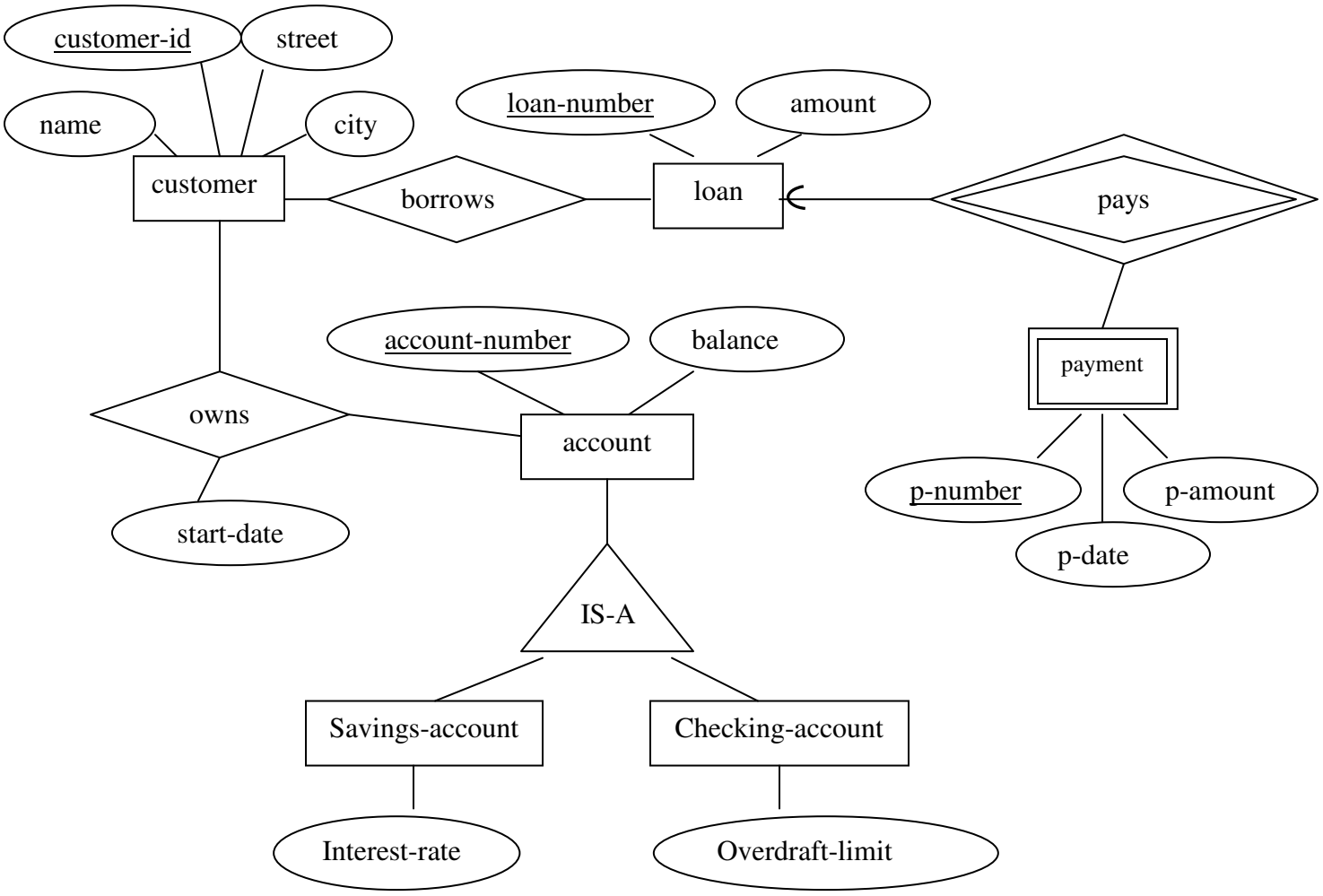
5) No electronic devices are permitted.

Scores:

Problem 1:	out of	15 points
Problem 2:	out of	15
Problem 3:	out of	20
Problem 4:	out of	30
Problem 5:	out of	20
Total:	out of	100 points

Problem 1. (15 points) ER and translation to relation model

Translate the following ER diagram into a relational schema. For each relation in your schema, **specify the key** of that relation. In translating a subclass hierarchy, **use the ER style translation**.



customer(customer-id, name, street, city)
loan(loan-number, amount)
payment(p-number,loan-number, p-date, p-amount)
borrows(customer-id, loan-number)
account(account-number, balance)
owns(customer-id,account-number, start-date)
Savings-account(account-number, Interest-rate)
Checking-account(account-number, Overdraft-limit)

Grading Scheme:
- 4pts for the "payment" relation
- 3pts for the "Savings-account" relation
- 3pts for the "Checking-account" relation
- 1pt for each other relation

Problem 2. (15 points) Schema Refinement

Consider the relation R(A,B,C,D,E) with the following functional dependencies:

$$A \rightarrow B,\ A \rightarrow C,\ BC \rightarrow A,\ D \rightarrow E.$$

Is R in BCNF or 3NF or neither? If R is not in BCNF, decompose R into a collection of BCNF relations. Show each step of the decomposition process.

[NOTE] FDs = $\{A \rightarrow B, A \rightarrow C, BC \rightarrow A, D \rightarrow E\} = \{A \rightarrow BC, BC \rightarrow A, D \rightarrow E\}$

It is neither in BCNF nor in 3NF.
The keys are AD, BCD.
All the FDs violate the BCNF condition as none of their LHS are superkeys.

There are two possible decompositions: {ABC, DE, AD} or {BCA, BCD, DE}

<u>Decomposition 1</u>
Decomposing on $A \rightarrow BC$:
$R_1 = (ABC)$, $FD_1 = \{A \rightarrow BC, BC \rightarrow A\}$:
This is in BCNF as A and BC are keys and LHS of both FDs are super keys.
$R_2 = (ADE)$, $FD_2 = \{D \rightarrow E\}$:
This is not in BCNF because the key is AD where as LHS of the FDs is not a superkey.

Decomposing R_2 on $D \rightarrow E$:
$R_{21} = (DE)$, $FD_{21} = \{D \rightarrow E\}$
This is in BCNF as the key is D and LHS of the FD a superkey.
$R_{22} = (AD)$, $FD_{22} = \{\}$
This is in BCNF as there are no FDs

Therefore the decomposed schema is:
$R_1 = (ABC)$, $FD_1 = \{A \rightarrow BC, BC \rightarrow A\}$, $R_{21} = (DE)$, $FD_{21} = \{D \rightarrow E\}$, $R_{22} = (AD)$, $FD_{22} = \{\}$

<u>Decomposition 2</u>
Decomposing on $BC \rightarrow A$:
$R_1 = (BCA)$, $FD_1 = \{A \rightarrow BC \text{ or } BC \rightarrow A\}$
This is in BCNF as A and BC are the keys and hence LHS of all FDs are superkeys.
$R_2 = (BCDE)$, $FD_2 = \{D \rightarrow E\}$
This is not in BCNF as the key is BCD and hence LHS of FD $D \rightarrow E$ is not a superkey.

Decomposing R_2 on $D \rightarrow E$:
$R_{21} = (DE)$, $FD_{21} = \{D \rightarrow E\}$
This is in BCNF as D is the key and the LHS of the only FD.
$R_{22} = (BCD)$, $FD_{22} = \{\}$
This is in BCNF because there is no FD.

Therefore the decomposed schema is:
$R_1 = (BCA)$, $FD_1 = \{A \rightarrow BC, BC \rightarrow A\}$, $R_{21} = (DE)$, $FD_{21} = \{D \rightarrow E\}$, $R_{22} = (BCD)$, $FD_{22} = \{\}$

Grading Scheme:
- 5 pts for the first part and 10 pts for the decomposition.

Problem 3. (20 points) Relational Algebra

This problem has 2 parts.

Consider the following database schema:

course (<u>course#</u>, dept-name)
enroll (<u>studentID, course#</u>, status)

(a) (10 points) Write a **relational algebra** expression to find the course# of all the courses offered by CS department.

Project_{course#}(Select_{dept-name==”CS”}(Course))

Grading Scheme:
- No partial credits here.

(b) (10 points) Write a **relational algebra** expression to find the studentIDs of all the students who are not enrolled in course# 411.

Project_{studentID}(enroll) – Project_{studentID}(Select_{course# == “411”}(Enroll))

Some students answered:
Project_{studentID}(Select_{course# <> “411”}(Enroll))

This answer is incorrect. Suppose there is a student who does not register for 411, however, the above query would still return the studentID for that student if he/she is registered for any course other than 411.

And a few students answered:
Project_{studentID}(enroll – Select_{course# == “411”}(Enroll))

This was a good attempt at removing above problem, however, still suffers from that problem.

A good way to verify your relational algebra expression or sql query is to try on a small example.

Grading Scheme:
- For the two incorrect answers, we award 5/10 points.

Problem 4. (30 points) SQL

This problem has 3 parts.

Again consider the following database schema:

course (<u>course#</u>, dept-name)
enroll (<u>studentID, course#</u>, status)

(a) (10 points) Write a **SQL** query that finds the studentID of all the students who are enrolled in at least one course offered by CS department and are not enrolled in any courses offered by EE department.

{SELECT studentID
FROM course, enroll
WHERE course.course# = enroll.course# AND dept-name = 'CS' }
EXCEPT
{SELECT studentID
FROM course, enroll
WHERE course.course# = enroll.course# AND dept-name = 'EE' }

(b) (10 points) Write a **SQL** query that lists the course# of all the courses for which more than 50 students are enrolled.

SELECT course#
FROM enroll
GROUP BY course#
HAVING count(studentID)>50

(c) (10 points) Write a **SQL** query that finds the studentID of all the students who are enrolled in the maximum number of courses. (For example, suppose there are 3 students, each is enrolled in 5 courses, and all other students are enrolled in fewer than 5 courses. Then you are expected to return the studentID of these 3 students.)

CREATE VIEW v AS
SELECT student ID, count(course#) as NumOfCourses
FROM enroll
GROUP BY studentID

SELECT studentID
FROM v
WHERE NumOfCourses=max(NumOfCourses)

Problem 5. (20 points) Constraints and Assertions

In this problem, you are given the schema for a bank and will be required to **write an assertion**. The relation *Account-owners* is a many-to-many relationship between the relation *Customers* and the relation *Accounts*. The SQL definitions for these relations are as follows:

create table *Customers*
(*customer-name* **char(20),**
customer-address **char(20),**
primary key *(customer-name))*

create table *Accounts*
(*account-number* **char(10),**
balance **integer,**
primary key *(account-number))*

create table *Account-owners*
(*customer-name* **char(20),**
account-number **char(10),**
primary key (*customer-name, account-number),*
foreign key (*customer-name)* **references** *Customers(customer-name),*
foreign key (*account-number)* **references** *Accounts(account-number))*

Write an assertion such that for every customer at least one of the following conditions holds true:

- He (or she) is owner of at most 5 accounts
- The sum of the balance of various accounts he (or she) owns is greater than $50,000.

```
CREATE ASSERTION mid-term-assert AS CHECK
    (NOT EXISTS (SELECT O.customer-name
                 FROM Accounts A, Account-Owners O
                 WHERE A.account-number == O.account-number
                 GROUP BY O.customer-name
                 HAVING COUNT(A.account-number) > 5
                        and SUM(A.balance) < 50000))
```

Alternative answer:

```
CREATE ASSERTION mid-term-assert2 AS CHECK
    (NOT EXISTS (SELECT * FROM Customers C
                 WHERE (SELECT count(*) FROM Account-Owners O
                        WHERE O.customer-name == C.customer-name) > 5
                       AND (SELECT SUM(balance)
                            FROM Accounts A, Account-Owners O2
                            WHERE A.account-number == O2.account-number
                                  AND A.customer-name == C.customer-name) < 50000
                ))
```

Grading Scheme:

- checking the two conditions: 8 each
- combining the two conditions: 2 points
- syntax of assertion: “CREATE ASSERTION name CHECK (some Boolean condition)”: 2 points

Name:_____

CSE 444, Winter 2011, Midterm Examination
9 February 2011

Rules:

- Open books and open notes.
- No laptops or other mobile devices.
- Please write clearly.
- Relax! You are here to learn.
- An extra page is provided in case you run out of space, but make sure to put a forward reference.

Question	Max	Grade
1	30	
2	30	
3	18	
4	22	
Total	100	

Initials:_____

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

   For a given quarter, the table `Student` lists all students registered at the University, the table `Course` lists all courses offered, and the table `Enrollment` encodes the students that attend any course in the current quarter.

```
Student ( sid, sname )
Course ( cid, cname )
Enrollment ( cid, sid )
```

   `Enrollement.sid` is a foreign key that references `Student.sid`.

   `Enrollement.cid` is a foreign key that references `Course.cid`.

   (a) (**8** points) Assume that no two courses have the same name. Call a course "big" if the number of students enrolled is at least 50. Write a SQL query that returns the names of all big courses in decreasing order of their enrollment size.

   Solution:

```
SELECT cname
FROM Course C, Enrollment E
WHERE C.cid = E.cid
GROUP BY cname
HAVING COUNT(*) >= 50
ORDER BY COUNT(*) DESC
```

Initials:_____

(b) (**8** points) A "classmate" of a student is another student who is enrolled in at least one same class. Write a SQL query that returns all pairs of classmates. You should return their student IDs, as their names do not uniquely identify them.

Solution:

```
SELECT DISTINCT E1.sid AS sid1, E2.sid AS sid2
FROM Enrollment E1, Enrollment E2
WHERE E1.cid = E2.cid AND E1.sid != E2.sid
```

Initials:_____

(c) (**14** points) Write a SQL query that computes for each student id in the `Student` table the total number of different classmates that this student has across all courses. Note that a student may be so busy with research (or parties ⌣̈) that he or she may choose not to enroll to any course that quarter, and hence has no classmates. Such students should still appear in the result, with 0 as the number of their classmates.

Hint: this is not required, but it may help you to make use of an earlier query.

Solution:
We can use our query from the previous question in a subquery:

```
SELECT S.sid, COUNT(T.sid2)
FROM Student S  LEFT OUTER JOIN  (SELECT DISTINCT E1.sid AS sid1, E2.sid AS sid2
                                  FROM Enrollment E1, Enrollment E2
                                  WHERE E1.cid = E2.cid AND E1.sid != E2.sid) T
                ON S.sid = T.sid1
GROUP BY S.sid
```

Slightly different alternative:

```
SELECT S.sid, COUNT(DISTINCT T.sid2)
FROM Student S  LEFT OUTER JOIN  (SELECT DISTINCT E1.sid AS sid1, E2.sid AS sid2
                                  FROM Enrollment E1, Enrollment E2
                                  WHERE E1.cid = E2.cid) T
                ON S.sid = T.sid1 AND S.sid != T.sid2
GROUP BY S.sid
```

Initials:_____

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

(a) (**12** points) A friend is impressed by your skills in CSE 444, and asks you to help her design a database to model doctors' offices across Seattle. She started the E/R diagram below which already contains all the entity sets, their attributes and relationships she wants you to consider. You need to finish the diagram and define a number of constraints to ensure that you model the semantics of an office as closely as possible.

Make sure that you do not impose additional constraints not defined by the model.

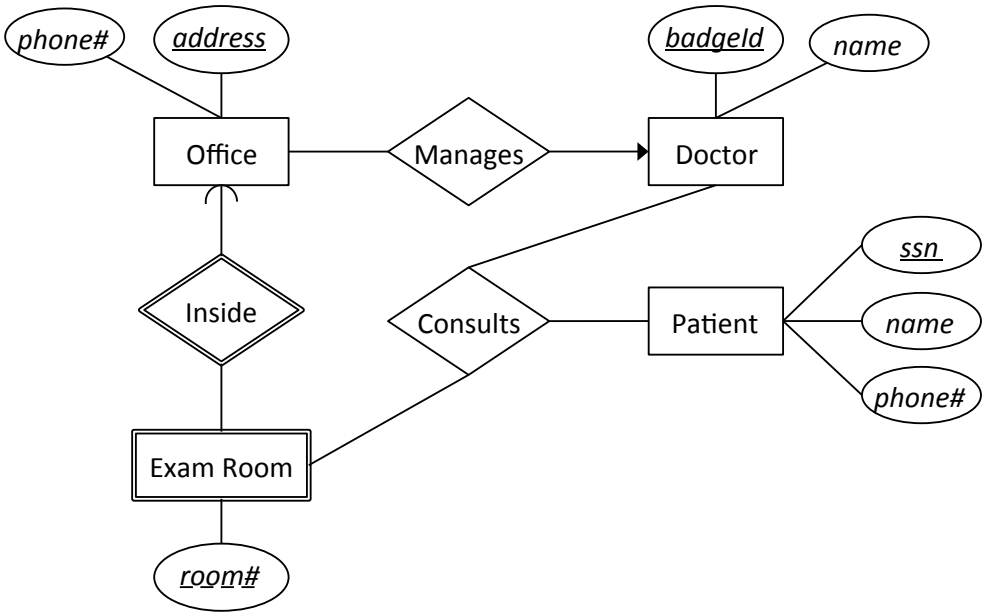
i. (**4** points) An office may be managed by at most one doctor. A doctor is uniquely identified by their badgeId, and may manage more than one office. Draw the necessary constraints to capture this requirement.

ii. (**5** points) An office is identified by its address, and contains one or more exam rooms. A room can be identified by its room number, and the office that it is in. Capture these constraints by drawing on or modifying the diagram where necessary.

iii. (**3** points) When a patient visits an office, he or she has a consultation with a doctor in an examination room. Each patient is uniquely identified by their SSN.

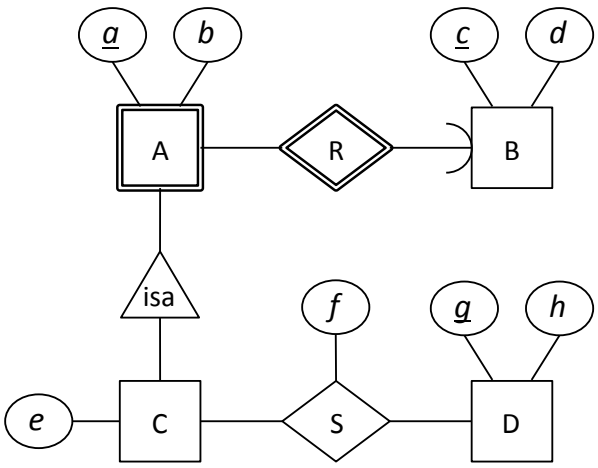
Draw your answer directly onto the figure below. Make sure that any arrows, lines, double lines etc, are clearly visible.

Solution:



Initials:_____

(b) (**12** points) Convert the E/R diagram below to a database schema. Indicate the keys for each table in your answer. You do not need to write SQL DDL commands.



Solution:

```
A ( c , a , b )
B ( c , d )
C ( a , c , e )
D ( g , h )
S ( a , c , g , f )
```

Explanation:
Entity A is weak and dependent on B. Hence, the key of A needs to include the key of B.
C is a refinement of A and, hence, needs to include the key of A.

Initials:_____

(c) (**6** points) Suppose we are told that `R(A, B, C, D)` is in BCNF, and that 3 out of the 4 FDs listed below hold for R. Choose the FD that R does not satisfy, and explain your reasoning.

$$1: A \to BCD$$
$$2: BC \to A$$
$$3: CD \to B$$
$$4: D \to C$$

Solution:
The problem states that only 3 out of the given 4 FDs hold for R. So, there are 4 possibilities:

- If 2, 3, 4 are the ones that hold, then D is the only key, and therefore 2 would violate BCNF, so it's not a possibility.
- If 1, 3, 4 are the ones that hold, then the only key is A. But then both 3 and 4 would be in violation of BCNF, so this is not a good choice either.
- If 1, 2, 4 are the ones that hold, then the keys are A, and BC. But then both 4 would violate BCNF.
- If 1, 2, 3 are the ones that hold, then A, BC and CD are all keys, so none of them violates BCNF. Therefore, this is the right choice. So, 4 is the FD that doesn't hold based on the problem description.

(Note that all 4 FDs could hold without violation of BCNF, but the problem definition tells you that only 3 hold, and under this assumption, there is only one possibility as listed above)

Initials:_____

3. (**18** points) **Logging and Recovery**

Your database server crashed due to a power outage. After rebooting, you find the following log and checkpoint information on disk, and begin the recovery process. We assume that STEAL/NO FORCE policy is used, and we use the ARIES method for recovery.

LSN	**Record**	**prevLSN**	**undoNextLSN**
30	update: T3 writes P5	null	-
40	update: T4 writes P1	null	-
50	update: T4 writes P5	40	-
60	update: T2 writes P5	null	-
70	update: T1 writes P2	null	-
80	Begin Checkpoint	-	-
90	update: T1 writes P3	70	-
100	End Checkpoint	-	-
110	update: T2 writes P3	60	-
120	T2 commit	110	-
130	update: T4 writes P1	50	-
140	T2 end	120	-
150	T4 abort	130	-
160	update: T5 writes P2	Null	-
180	CLR: undo T4 LSN 130	150	50

Transaction Table at time of checkpoint

Transaction ID	**lastLSN**	**Status**
T1	70	Running
T2	60	Running
T3	30	Running
T4	50	Running

Dirty Page Table at checkpoint

Page ID	**recLSN**
P5	50
P1	40

(a) (**2** points) The log record at LSN 60 denotes an update to page P5 by transaction T2. Was this update to page P5 successfully written to disk? If yes, at what point could that have happened?

Solution:
The update at LSN 60 may have been written to disk; the log entry was flushed before the write itself. It was not yet flushed at the time of the checkpoint, but may have been flushed later.

(b) (**2** points) The log record at LSN 70 denotes an update to page P2 by transaction T1. Was this update successfully written to disk? If yes, at what point? Briefly explain your answers.

Solution:
The update at LSN 70 was flushed to disk before the Begin Checkpoint (LSN 80). We know this because it is not in the dirty page table at the time of the checkpoint.

Initials:_____

(c) (**6** points) At the end of the Analysis phase, what will the Transaction and Dirty Page Tables look like? Populate your answers in the tables below.

Solution:

Transaction ID	**lastLSN**	**Status**
T1	90	Running
T3	30	Running
T4	180	Aborting
T5	160	Running

Page ID	**recLSN**
P1	40
P2	160
P3	90
P5	50

(d) (**6** points) At which LSN in the log will REDO begin? Which log records will be redone? List their LSNs, and briefly justify your answers.

Solution:

Redo should begin at LSN 40, the smallest of the recLSNs in the dirty page table. The following log records should be redone:

40, 50, 60, 90, 110, 130, 160, 180

30 is skipped because it precedes LSN 40. 70 is skipped because $P2.recLSN = 160 > 70$. Entries that are not updates are skipped. The CLR record is not skipped, nor is the LSN that it undoes.

Initials:_____

(e) (**2** points) For each of the following questions circle the right answers.

i. In an UNDO only logging scheme, what buffer management policies apply? (circle one of STEAL or NO STEAL, and one of FORCE or NO FORCE)

STEAL / NO STEAL FORCE / NO FORCE

ii. During recovery with REDO only logging, we cannot use a checkpoint for which we see a <START CKPT($\ldots T_i \ldots$)> record but no corresponding <END CKPT> record.

TRUE / FALSE

Solution:
i. STEAL and FORCE, ii. TRUE

Initials:_____

4. (**22** points) **Concurrency Control**

In the schedules given below, the label $R_i(X)$ indicates a read of element X by transaction T_i, and $W_i(X)$ indicates a write of element X by transaction T_i.

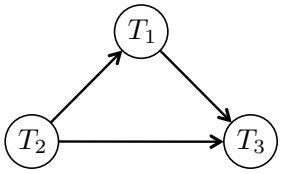
(a) (**4** points) Draw the precedence graph for schedule 1. Is schedule 1 conflict-serializable? If so, what order of the three transactions defines a conflict-equivalent serial schedule?

Schedule 1

$R_2(A)\ R_1(C)\ R_2(B)\ W_2(B)\ R_3(B)\ R_1(A)\ R_3(C)\ W_3(C)\ W_1(A)$

Solution:

Schedule 1 is conflict-serializable because the precedence graph has no cycles. There is an arrow $T_2 \to T_1$ because of the conflict $R_2(A) \ldots W_1(A)$. There is an arrow $T_2 \to T_3$ because of the conflict $W_2(B) \ldots R_3(B)$. There is an arrow $T_1 \to T_3$ because of the conflict $R_1(C) \ldots W_3(C)$. The only possible conflict-equivalent serial schedule is (T_2, T_1, T_3)



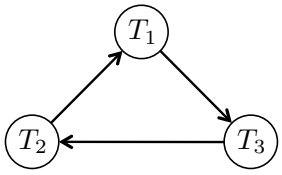
(b) (**4** points) Draw the precedence graph for schedule 2. Is schedule 2 conflict-serializable? If so, what order of the three transactions defines a conflict-equivalent serial schedule?

Schedule 2

$R_2(A)\ R_1(C)\ R_2(B)\ R_3(B)\ W_2(B)\ R_1(A)\ R_3(C)\ W_3(C)\ W_1(A)$

Solution:

Schedule 2 is not conflict-serializable because the precedence graph has a cycle. There is an arrow $T_2 \to T_1$ because of the conflict $R_2(A) \ldots W_1(A)$. There is an arrow $T_3 \to T_2$ because of the conflict $R_3(B) \ldots W_2(B)$. There is an arrow $T_1 \to T_3$ because of the conflict $R_1(C) \ldots W_3(C)$.



Initials:_____

(c) (**9** points) Can the two schedules from (a) and (b) occur under (non-strict) 2 Phase Locking? If yes, then add the proper lock/unlock actions to the corresponding schedule(s) in a way compliant with (non-strict) 2PL. Use $L_i(X)$ to denote transaction i locking element X, and $U_i(X)$ to denote transaction i releasing the lock on X. Assume only exclusive locks are used, and make sure that no locks remain held at the end of the schedule.

The schedules below are the same ones as in the previous page, and repeated here for convenience with dotted spaces in-between the actions. You can use this dotted space to add the proper lock and unlock actions, or if you prefer you can rewrite the schedules including the locking actions in the empty space further down the page.

If you believe a schedule cannot occur with 2PL, give a brief explanation.

Solution:

There are many correct answers to this question. As long as each piece of data is locked before used, each lock is not simultaneously held by multiple transactions, and no transaction locks data after it begins unlocking, the answer is correct.

Schedule 1

$L_2(A)\ \ R_2(A)\ \ L_1(C)\ \ R_1(C)\ \ L_2(B)\ \ R_2(B)\ \ W_2(B)\ \ U_2(B)\ \ U_2(A)\ \ L_3(B)\ \ R_3(B)$

$L_1(A)\ \ R_1(A)\ \ U_1(C)\ \ L_3(C)\ \ R_3(C)\ \ W_3(C)\ \ U_3(C)\ \ W_1(A)\ \ U_1(A)\ \ U_3(B)$

Schedule 2 cannot be produced by 2PL, as it is not conflict-serializable.

Initials:_____

(d) (**5** points) Circle TRUE or FALSE to reflect the validity of the following statements.

i. For any schedules S_1 and S_2, if S_1 and S_2 are conflict serializable, then S_1 and S_2 are conflict equivalent.[1]

TRUE / FALSE

ii. A SIX lock is compatible with IS and IX locks, i.e. if a transaction holds a SIX lock on an object, then another transaction can take an IS or IX lock on the same object.

TRUE / FALSE

iii. An IX lock is compatible with an IS lock, i.e. if a transaction holds an IS lock on an object, then another transaction can take an IX lock on the same object.

TRUE / FALSE

iv. Strict 2PL prevents deadlocks.

TRUE / FALSE

v. In timestamp-based concurrency control, if a transaction gets aborted, it will be restarted with a new timestamp.

TRUE / FALSE

Solution:

i. FALSE, ii. FALSE, iii. TRUE, iv. FALSE, v. TRUE

[1] Two schedules are conflict equivalent, if every pair of conflicting actions appears in the same order in both schedules.

Initials:_____

EXTRA PAGE

CSE 444 Midterm Exam

November 13, 2009

Name _____Sample Solution_____

Question 1	/ 24
Question 2	/ 22
Question 3	/ 22
Question 4	/ 12
Question 5	/ 20
Total	/ 100

Question 1. SQL (24 points, 8 each part) With the holiday gift-giving season upon us, we'd like to set up a database to keep track of different products and stores that stock them. Our database has the following tables:

PRODUCT(pid, pname, manufacturer, color)
STORE(sid, sname, address, city, phone)
INVENTORY(pid, sid, price, quantity)

Table PRODUCT has a unique product id number, pid, for each product, and gives the name, manufacturer, and color of that product. Table STORE has a unique id number for each store, sid, and gives the store name, address, city, and phone number. Note that there may be more than one store with the same name in the same city – for example, there may be several 'Costco' stores in 'Seattle' – but the stores will have different addresses and store id numbers. The INVENTORY table gives a list of which stores carry which products, how many of each product are currently in stock at each store, and the product price at each different store. The pid and sid entries in this table are foreign keys referring to the PRODUCT and STORE tables respectively. The quantity in an INVENTORY table entry might be 0 if a particular store carries a product but currently has none in stock. Also, the same product may have different prices at different stores.

(a) Write a SQL query that gives a list of store names, addresses, and cities of all stores that sell iPods (pname is 'iPod') that have the color 'red'. The list should include all such stores, even if they currently have none in stock. The list should be sorted by city; beyond that, the other information about stores in a given city may appear in any order.

```
SELECT s.sname, s.address, s.city
FROM PRODUCT p, STORE s, INVENTORY i
WHERE p.pid = i.pid AND s.sid = i.sid AND p.pname = 'iPod' AND p.color = 'red'
ORDER BY s.city
```

(continued next page)

Question 1. (cont) Schemas repeated for convenience:

PRODUCT(pid, pname, manufacturer, color)
STORE(sid, sname, address, city, phone)
INVENTORY(pid, sid, price, quantity)

(b) Write a SQL query that gives the minimum available price of the product named 'Droid' from any store that has one or more 'Droid's in stock – don't consider any store that doesn't have at least one 'Droid' in its current inventory. You only need to produce the minimum price; you don't need to identify the store.

```
SELECT min(i.price)
FROM INVENTORY i, PRODUCT p
WHERE p.pname = 'Droid' AND i.pid = p.pid AND i.quantity > 0
```

(c) Write a SQL query that produces a table showing the availability by city of products with the name 'Xbox'. The results should have one entry for each city where an 'Xbox' is available. The entry for each city should show the city name, the total number of 'Xbox's currently available in all stores in that city and the average selling price of a 'Xbox' in that city (just compute the average of the individual store prices – don't consider the number of units in stock at each store). The results should be sorted by city.

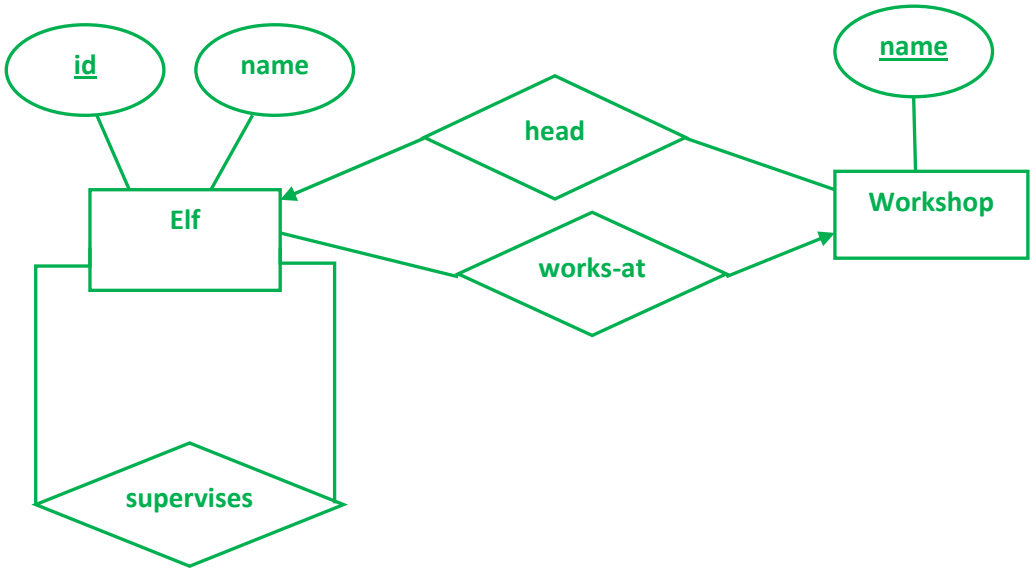
```
SELECT s.city, SUM(s.quantity), AVG(i.price)
FROM PRODUCT p, STORE s, INVENTORY i
WHERE p.pid = i.pid AND i.sid = s.sid AND p.pname = 'Xbox'
GROUP BY s.city
ORDER BY s.city
```

Question 2. Database design (22 points) Continuing with our holiday theme, Santa has been having trouble keeping track of all of his workshops and elves. He has decided that he needs a database to organize all this information and has asked you to design it.

(a) Give an E/R diagram that captures the following information.

- Santa has several workshops. Each workshop has a unique name (North Pole, Northwest Branch, Toy Factory, Hoboken …), and a head elf (the elf in charge of the workshop). A single elf may be the head of more than one workshop. Although all workshops normally have a head elf, these positions are vacant from time to time.
- Santa has many elves working for him. An elf only works at one particular workshop, but some elves are between jobs and are not assigned to any particular workshop at the moment. Each elf has a name and a unique employee number.
- Some elves supervise other elves, and those elves might themselves have a supervisor who is a different elf. Most elves are just workers who don't supervise anyone. An individual elf may have more than one supervisor if that is appropriate for the particular elf's job. There are a few elves that don't have a supervisor. An elf who is the head of a workshop (see the first paragraph above) does not necessarily supervise other elves, but he/she might do so.

Draw your E/R diagram below:



(continued next page)

Question 2. (cont.) (b) Give a relational schema that captures your E/R diagram from part (a). You should give the table and attribute names and clearly indicate which attributes are keys and foreign keys in the various tables. Your tables should be in BCNF (i.e., contain no bad anomalies or functional dependencies), but you do not need to demonstrate this or prove that your design is in BCNF. You do **not** need to give SQL CREATE TABLE statements for your tables.

Notation: tbl.key means a foreign key in the given table.

Elf(id, name, Workshop.name)

Workshop(name, Elf.id)

Supervises(Elf.id, Elf.id)

Question 3. BCNF (22 points) Consider the relation R(A, B, C, D, E) that has the following functional dependencies:

AB $\rightarrow$ C
CD $\rightarrow$ E
DE $\rightarrow$ B

Decompose this relation, if necessary, into a collection of relations that are in BCNF. For full credit, show your work, the intermediate decomposition steps, and show which dependency violations you are correcting at each step.

There are three different ways to get a BCNF decomposition, depending on which dependency is used first.

I. Use AB+ = {A, B, C} to decompose R into R1(A, B, C) and R2(A, B, D, E). Table R2 is not in BCNF because DE+ = {B, D, E} ≠ {A, B, D, E}. Use this to decompose R2 into R21(B, D, E) and R22(A, D, E)

II. Use CD+ = {B, C, D, E} to decompose R into R1(B, C, D, E) and R2(A, C, D). R1 is not in BCNF because DE+ = {B, D, E} ≠ {B, C, D, E}, so decompose it to get R11(B, D, E) and R12(C, D, E).

III. Use DE+ = {B, D, E} to decompose R into R1(B, D, E) and R2(A, C, D, E). R2 is not in BCNF beause of CD+ = {C, D, E}, so decompose it to get R21(C, D, E) and R22(A, C, D).

Question 4. Concurrency Control / Serialization (12 points) We have looked at several ways a database scheduler can manage transactions to either ensure that their operations are consistent with some conflict-serializable schedule, or detect operations that would violate serialization and either delay or roll back one or more transactions to prevent the problem.

Each of the different concurrency control schemes determines the effective serial order of transactions based on some operation(s) issued by the transactions. For example, one possibility is to ensure that the effective serial order is the same order in which the transactions issue their first read operation. Or the serial order might reflect the order in which the transactions commit or abort.

The significant events that occur during a transaction include some or all of the following:

(a) Start transaction
(b) First read operation issued by the transaction
(c) Last read operation issued by the transaction
(d) First write operation issued by the transaction
(e) Last write operation issued by the transaction
(f) All write operations have completed (i.e., are written to stable storage)
(g) First lock acquired by transaction
(h) Last lock acquired by transaction
(i) First lock released by transaction
(j) Last lock released by transaction
(k) Beginning of validation phase
(l) End of validation phase
(m) Transaction commit or abort (rollback) issued

Write the correct letter in front of each concurrency control scheme below to indicate the event that determines the effective serial order of transactions using that scheme. If a scheme doesn't absolutely guarantee to produce a serializable schedule, use the answer that best determines the effective order of the transactions.

_____**i**_____ Two Phase Locking (2PL)

_____**a**_____ Timestamps

_____**l**_____ Validation

_____**a**_____ Snapshot Isolation

Question 5. ARIES (20 points) After a crash, an ARIES log contains the following entries. There are no checkpoints in the log, so this is all of the information that is available.

LSN	prevLSN	Transaction ID	Type	pageID	Data
1	0	T1	U	P2	Y
2	0	T2	U	P2	Y
3	2	T2	U	P1	X
4	3	T2	C	--	--
5	0	T3	U	P2	Y

(Note: the log entries in this question go from top to bottom, unlike the example in class where they ran from left to right.)

(a) Using the information in this log, show the results of the analysis pass that is the first part of the ARIES crash recovery. Fill in the transaction table and dirty page table below with the information discovered during the analysis pass. If an entry in a table is created and then later changed or deleted during the analysis pass, show the entry and draw a line through all or part of it as appropriate to indicate the changes or deletions.

Transaction table

Transaction	lastLSN
T1	**1**
~~**T2**~~	~~**2, 3, 4**~~
T3	**5**

Dirty page table

Page	RecoveryLSN
P2	**1**
P1	**3**

(b) What is the FirstLSN, i.e., the LSN of the first log entry that needs to be redone during the redo phase of the crash recovery that follows the analysis?

1

(c) Which transaction or transactions are "loser" transactions that need to be undone during the undo phase of the crash recovery that follows the redo phase?

T1, T3

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:_____

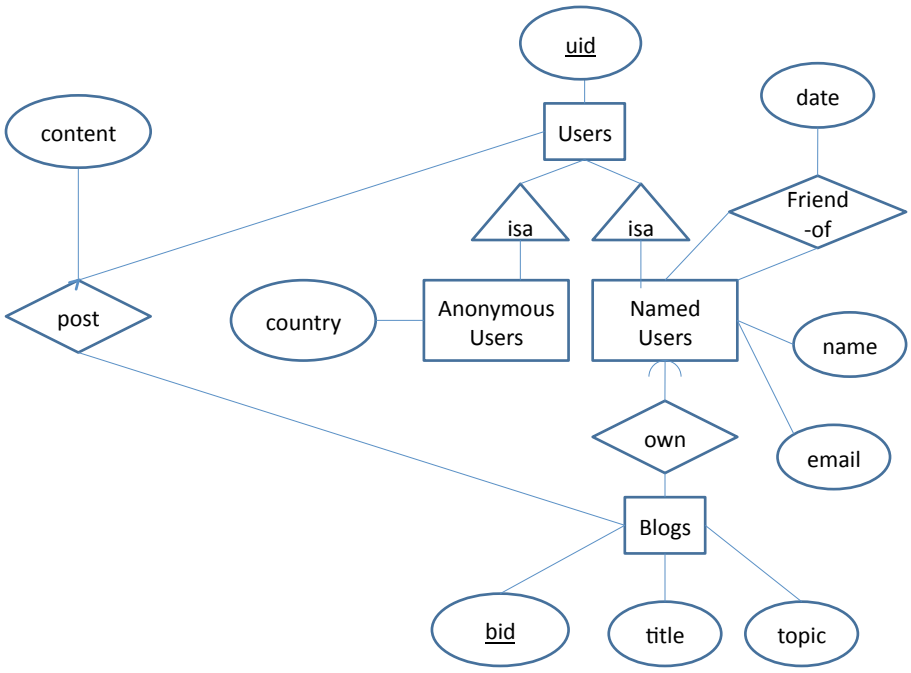
2. (**30** points) **Conceptual Design**

   (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 $\rightarrow$ E and A $\rightarrow$ 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(A,B) and R2(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(C, D, E) and R4 (A, C, D)

End result: R1(A,B), R3(C, D, E), and R4 (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.

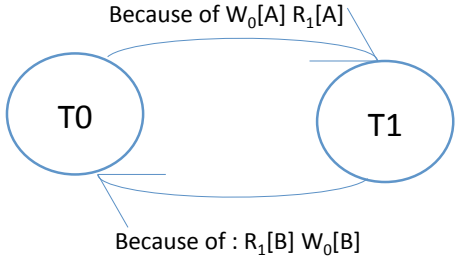
Solution:
The UNDO pass scans backward from the end of the log. The pass undoes the updates by all transactions that had not committed by the time of the crash. These transactions can be found in the Transactions Table rebuilt during the analysis pass. All updates are undone unconditionally (since the 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
$L_0[A]$	
$r_0[A]$	
$w_0[A]$	
	$L_1[A] \rightarrow blocks$
$L_0[B]$	
$r_0[B]$	
$w_0[B]$	
$U_0[A]$	
$U_0[B]$	
c_0	
	$L_1[A] \rightarrow granted$
	$r_1[A]$
	$L_1[B]$
	$r_1[B]$
	$U_1[A]$
	$U_1[B]$
	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	