Umeå University
Department of Computing Science
TDBC86 — Database Concepts
Examination: December 19, 2003

Name and ID number: _____

Signature: _____

1. All answers must be written in English.
2. A pocket engineering calculator (miniräknare) and English/X - X/English dictionary may be used. No other help materials are allowed.
3. The examination consists of two parts. In Part I, which includes Problems 1–5, on pages 1–12, the answers **must** be written on the sheets provided. In general, only the answers will be checked for these questions; derivations need not be provided and will not be evaluated. In Part II, which includes Problems 6–11, the answers must be written on separate sheets. If appropriate for Problems 6–11, work showing how the answers were obtained will be evaluated, and in some cases is required.
4. You must turn in this cover sheet, the answer sheets for Part I (pages 1–12), and the pages containing your answers for Problems 6–11. Collate the sheets in numerical order of the problems, and write on only one side of the paper. Write your name and personnummer on the top of each sheet. For questions 6–11, write the question number on the top of each sheet as well.
5. The examination has a total of 800 points, which constitutes 80% of the course grade. The other 20% is based upon the obligatory exercises, as explained in the course syllabus.
5. For Problems 6–11, you have the choice, for each part, to do the problem, or to skip it for partial credit. In the table below, place an X in the position for any problem for which you have attempted a solution, and which you wish to have graded. It is extremely important that you fill in this table properly, because of the following option. For any box which is left blank, the associated question will not be graded, and you will instead be awarded 15% of the points for that question. Your decision to leave a box blank is definitive, so be very careful. For example, If you leave box 8(b) blank, your answer to that question will not be graded, even if it is completely correct. On the other hand, if you place an X in box 8(b), but provide no answer whatsoever to that question, you will not receive 15% of the points for that question. It is strongly recommended that you use a pencil, in case you change your mind! *Note that Problems 1–5 do not carry this option. No points are awarded for skipping them.*

Prob	6	7	8	9	10	11
(a)						
(b)	■					
(c)	■					
(d)	■	■	■	■		
(e)	■	■	■	■		
(f)	■	■	■	■	■	

Name and ID number: _____

(1: 80 points total) For each of the following statements, circle the T if the assertion is true, and the F if it is false. The point total will be computed as $4 \times C - 2 \times I$ with C = number of correct answers and I = number of incorrect answers; however, the score may never be less than zero. A blank answer is neither in C nor in I.

T F (a) The schema of a database in a modern relational system such as PostgreSQL is always in BCNF.

T F (b) If a single-relation schema which is constrained by functional dependencies is in 2NF, and it has only one candidate key, then it is in BCNF as well.

T F (c) If a single-relations schema which is constrained by functional dependencies is in 2NF, and each candidate key consists of a single attribute, then it is in BCNF as well.

T F (d) If a schedule is such that every transaction obeys a locking protocol which is both rigorous and conservative 2PL, then the schedule is necessarily serial.

T F (e) The current SQL standard, completed and released, is known as SQL:2003.

T F (f) A fundamental limitation of the relational model is that it does not support object identity.

T F (g) The OQL query language for object-oriented database systems admits stand-alone queries, just as SQL does.

T F (h) A major problem with a primary index is insertion and deletion of records.

T F (i) It is common for a relation to be stored such that there is both a primary index on the primary key and a clustering index on another attribute.

T F (j) It is usually the case that an index on a secondary key is dense.

T F (k) An advantage of extensible hashing over B^+-trees is that the index in extensible hashing is generally much smaller.

T F (l) In SQL schema declaration using **Create Table**, it is not possible to declare candidate keys which are not the primary key.

T F (m) In query optimization, it is generally a good idea to rearrange things so that select and project operations are performed before joins.

T F (n) A steal/no-force caching strategy is likely to be more efficient in terms of system resource use than a no-steal/force strategy.

Name and ID number: _____

T F (o) Requiring every query to return information on a sufficiently large number of records is an effective way to prevent tracking in statistical databases.

T F (p) Every relational schema which is constrained by functional dependencies has a dependency-preserving and lossless decomposition into 2NF.

T F (q) Every relational schema which is constrained by functional dependencies has a dependency-preserving and lossless decomposition into 3NF.

T F (r) For every query Q in the relational calculus, there is an equivalent query Q' which does not use the negation ($\neg$) operator.

T F (s) In ODBC, the database server and application client must run the same operating system.

T F (t) The system catalog is a database which provides information on the relations of the underlying schema.

Name and ID number: _____

(2: 60 points total; 6 points for each question) For each of the questions below, place an X between the parentheses to the left of the correct responses to the question. A question will have at least one correct choice, but it may have several. Mark all correct choices.

(a) The write-ahead logging protocol:

( ) states that before images must be written to the log before database writes.
( ) states that after images must be written to the log before database writes.
( ) is applicable to the pure immediate update strategy.
( ) is applicable to the pure deferred update strategy.

(b) It is important that the re-do operation involved in recovery be idempotent because:

( ) a crash may occur during the recovery process.
( ) cascading rollback may force a transaction to be re-done more than once.
( ) it must be possible to re-do transactions in any order.
( ) more than one checkpoint may occur during the lifetime of a transaction

(c) Conflict-serializable schedules:

( ) always admit lock schedules which follow the 2PL protocol.
( ) have the property that the interaction of the transactions is equivalent to that of some view serial schedule.
( ) never involve blind writes.
( ) are always view serializable as well.

(d) View-serializable schedules:

( ) always admit lock schedules which follow the 2PL protocol.
( ) have the property that the interaction of the transactions is equivalent to that of some serial schedule.
( ) never involve blind writes.
( ) are always conflict-serializable as well.

(e) Unsafe queries:

( ) are impossible to express in the relational algebra.
( ) are impossible to express in SQL.
( ) are impossible to express in the relational domain calculus.
( ) do not have mathematically well-defined answers.

Name and ID number: _____

(f) A major advantage of the B^+-tree over the B-tree is that:

( ) it is easier to process the data items sequentially.
( ) a B^+-tree is more fully balanced.
( ) the algorithm to insert a new data item is much simpler.
( ) it is likely that fewer disk accesses will be necessary to retrieve a given record.

(g) When joining two relations of approximately equal size, one of which has a sequential index on the joining column:

( ) building a temporary common hash table is a useful technique.
( ) having a sequential index on the joining column of the other relation as well is of substantial additional advantage.
( ) having a hash index on the joining column of the other relation as well is of substantial additional advantage.
( ) a merging algorithm is generally the best choice.

(h) When a cache page is pinned:

( ) it contains correct data which has not yet been written to disk.
( ) it may contain incorrect data and so must not be written to the permanent database.
( ) it must be un-pinned at a checkpoint.
( ) the dirty bit is set.

(i) Livelock:

( ) cannot occur when a deadlock-free strategy is in place.
( ) may often be minimized by employing strategy in which transactions are restarted with their old timestamps.
( ) is likely to be an issue when the deadlock problem is resolved by aborting some transactions.
( ) is impossible 2PL schedules which are both rigorous and strict are used.

(j) An object-oriented approach is likely to be superior to a relational approach in an application which involves:

( ) persistent data.
( ) a great deal of concurrency.
( ) entities which have a natural hierarchy associated with them.
( ) many recursive queries.

Name and ID number: _____

(3: 100 points total; 10 points for each part) Important: The answers to this question must be indicated below; do not write the answers to this question on a blank sheet.

The following relational schema with relation $R[ABCDEFG]$ and functional dependencies $\mathcal{F} = \{AB \rightarrow C,\ CD \rightarrow E,\ E \rightarrow FG,\ F \rightarrow D,\ G \rightarrow F\}$ applies to all ten parts of this problem.

Given below are six projections of $R[ABCDEFG]$. Assume that each such projection is constrained by exactly the functional dependencies in $\mathcal{F}^+$ which embed in the projection. In the space provided, give all of the candidate keys for the projection, as well as a basis of the functional dependencies from $\mathcal{F}^+$ which embed in it. Also, circle *all* of the normal-form properties which the projection possesses (even if some are implied by others). Circle the word none just in case none of the normal-form properties holds.

(a) Projection: $R[ABCF]$

Candidate keys: _____

FD basis: _____

Normal forms: None 2NF 3NF BCNF

(b) Projection: $R[CDEG]$

Candidate keys: _____

FD basis: _____

Normal forms: None 2NF 3NF BCNF

(c) Projection: $R[CDEF]$

Candidate keys: _____

FD basis: _____

Normal forms: None 2NF 3NF BCNF

(d) Projection: $R[ACDG]$

Candidate keys: _____

FD basis: _____

Normal forms: None 2NF 3NF BCNF

Name and ID number: _____

(e) Projection: $R[ABDF]$

Candidate keys: _____

FD basis: _____

Normal forms: None 2NF 3NF BCNF

(f) Projection: $R[EFG]$

Candidate keys: _____

FD basis: _____

Normal forms: None 2NF 3NF BCNF

Shown below are four decompositions of the schema on the previous page. For each decomposition, indicate whether or not it has the properties of *lossless* and *dependency preserving* by circling the appropriate word(s). Circle the word *neither* just in case it is neither lossless nor dependency preserving.

In checking the lossless property, use only pairwise reconstructions (*i.e.*, join trees). Do not test for cyclic decompositions (*i.e.*, decompositions which are lossless, but the join can not be reconstructed pairwise). [It was not explained in the course how to test for cyclic decompositions, so if you do not understand what this means, just use the techniques which were presented in the course.]

(g) Decomposition: $\{R[ABCF], R[ABDF], R[CDEG]\}$.
Neither Lossless Dependency preserving

(h) Decomposition: $\{R[ABCF], R[CDEG], R[CDEF]\}$.
Neither Lossless Dependency preserving

(i) Decomposition: $\{R[ABCF], R[ABDF], R[CDEG], R[EFG]\}$.
Neither Lossless Dependency preserving

(j) Decomposition: $\{R[ABCF], R[CDEF], R[EFG]\}$.
Neither Lossless Dependency preserving

Name and ID number: _____

(4: 80 points) A database contains three data objects, A, B, and C. Six transactions are given on these objects, specified as follows.

T_1: Read(A)	T_2: Read(A)	T_3: Write(C)	T_4: Write(C)	T_5: Read(A)	T_6: Read(C)
Write(A)	Write(A)	Read(B)	Read(B)	Write(A)	Write(C)
Write(B)	Read(B)	Write(B)	Write(B)	Read(B)	Read(A)
	Write(B)			Write(B)	Write(A)

Shown on each of the next two pages is (the same) table, the first two columns of which gives the initial part of a schedule for these transactions, up to the point of a system crash. The table on page 8 regards the pure immediate-update protocol, while that on page 9 regards the pure deferred-update protocol. Assume that both steal and no-force cache-writing strategies are used, and that the cache contents are lost with the crash. For each read and each write operation, indicate the *minimal* information which must be written into the system log for support of the protocol by drawing a circle around the correct items. The following notation is used for the entries in this column:

- Log:Before(T_i,X) denotes a log entry which contains the information that T_i executed a write on data item X, together with the value of that data item before the write was executed.
- Log:After(T_i,X) denotes a log entry which contains the information that T_i executed a write on data item X, together with the value of that data item after the write was executed.
- Log:Read(T_i,X) denotes a log entry which contains the information that T_i executed a read on data item X, together with the value of that data item at the time of the read.

Circle None just in case none of the other listed item need be entered into the log. **It is important that only the minimal necessary items be circled.** It is clearly sufficient to circle every Log:Read, Log:Before, and Log:After entry, but this is in general not necessary and not the minimal information, and will not be regarded as a correct answer. Note that nothing need be circled in this column for Begin, Commit, and Checkpoint operations, since the only reasonable log entry has already been included.

Also, answer the questions on pages 10 and 11 concerning the schedule given in these tables.

Name and ID number: _____

(a)

	Action	**Log information for pure immediate update**		
T_1	Begin	Log:Begin(T_1)		
T_1	Read(A)	Log:Read(T_1, A)	None	
T_1	Write(A)	Log:Before(T_1, A)	Log:After(T_1, A)	None
T_3	Begin	Log:Begin(T_3)		
T_3	Write(C)	Log:Before(T_3, C)	Log:After(T_3, C)	None
T_1	Write(B)	Log:Before(T_1, B)	Log:After(T_1, B)	None
T_1	Commit	Log:Commit(T_1)		
T_5	Begin	Log:Begin(T_5)		
T_5	Read(A)	Log:Read(T_5, A)	None	
	Checkpoint	Log:Checkpoint		
T_5	Write(A)	Log:Before(T_5, A)	Log:After(T_5, A)	None
T_4	Begin	Log:Begin(T_4)		
T_4	Write(C)	Log:Before(T_4, C)	Log:After(T_4, C)	None
T_2	Begin	Log:Begin(T_2)		
T_2	Read(A)	Log:Read(T_2, A)	None	
T_2	Write(A)	Log:Before(T_2, A)	Log:After(T_2, A)	None
T_3	Read(B)	Log:Read(T_3, B)	None	
T_3	Write(B)	Log:Before(T_3, B)	Log:After(T_3, B)	None
T_3	Commit	Log:Commit(T_3)		
T_4	Read(B)	Log:Read(T_4, B)	None	
T_4	Write(B)	Log:Before(T_4, B)	Log:After(T_4, B)	None
T_4	Commit	Log:Commit(T_4)		
T_2	Read(B)	Log:Read(T_2, B)	None	
T_2	Write(B)	Log:Before(T_2, B)	Log:After(T_2, B)	None
T_2	Commit	Log:Commit(T_2)		
T_6	Begin	Log:Begin(T_6)		
T_6	Read(C)	Log:Read(T_6, C)	None	

Name and ID number: _____

(b)

	Action	**Log information for pure deferred update**		
T_1	Begin	Log:Begin(T_1)		
T_1	Read(A)	Log:Read(T_1, A)	None	
T_1	Write(A)	Log:Before(T_1, A)	Log:After(T_1, A)	None
T_3	Begin	Log:Begin(T_3)		
T_3	Write(C)	Log:Before(T_3, C)	Log:After(T_3, C)	None
T_1	Write(B)	Log:Before(T_1, B)	Log:After(T_1, B)	None
T_1	Commit	Log:Commit(T_1)		
T_5	Begin	Log:Begin(T_5)		
T_5	Read(A)	Log:Read(T_5, A)	None	
	Checkpoint	Log:Checkpoint		
T_5	Write(A)	Log:Before(T_5, A)	Log:After(T_5, A)	None
T_4	Begin	Log:Begin(T_4)		
T_4	Write(C)	Log:Before(T_4, C)	Log:After(T_4, C)	None
T_2	Begin	Log:Begin(T_2)		
T_2	Read(A)	Log:Read(T_2, A)	None	
T_2	Write(A)	Log:Before(T_2, A)	Log:After(T_2, A)	None
T_3	Read(B)	Log:Read(T_3, B)	None	
T_3	Write(B)	Log:Before(T_3, B)	Log:After(T_3, B)	None
T_3	Commit	Log:Commit(T_3)		
T_4	Read(B)	Log:Read(T_4, B)	None	
T_4	Write(B)	Log:Before(T_4, B)	Log:After(T_4, B)	None
T_4	Commit	Log:Commit(T_4)		
T_2	Read(B)	Log:Read(T_2, B)	None	
T_2	Write(B)	Log:Before(T_2, B)	Log:After(T_2, B)	None
T_2	Commit	Log:Commit(T_2)		
T_6	Begin	Log:Begin(T_6)		
T_6	Read(C)	Log:Read(T_6, C)	None	

Name and ID number: _____

In answering the questions below, use exactly the partial schedule shown on the previous two pages, with only the single checkpoint indicated. *In all cases, assume that the system is restored to the state at the checkpoint as the initial step of the recovery process.* All transactions must be completed at the end of the recovery process; it is not acceptable to "discard" a transaction.

(c) To recover from the system crash in the context of the *pure immediate-update protocol*, for each transaction, indicate which of the following four actions is necessary by placing an $\times$ in the appropriate $\square$, as so $\boxtimes$.

T_1: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_2: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_3: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_4: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_5: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_6: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

(d) To recover from the system crash in the context of the *pure deferred-update protocol*, for each transaction, indicate which of the following four actions is necessary by placing an $\times$ in the appropriate $\square$, as so $\boxtimes$.

T_1: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_2: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_3: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_4: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_5: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_6: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

Name and ID number: _____

In answering the questions below, assume that a second checkpoint is executed after T_6 reads C, just before the crash. *Assume further that the system is restored to the state at the* ***second*** *checkpoint as the initial step of the recovery process.* All transactions must be completed at the end of the recovery process; it is not acceptable to "discard" a transaction.

(e) To recover from the system crash in the context of the *pure immediate-update protocol*, for each transaction, indicate which of the following four actions is necessary by placing an $\times$ in the appropriate ☐, as so ☒.

T_1: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_2: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_3: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_4: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_5: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_6: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

(f) To recover from the system crash in the context of the *pure deferred-update protocol*, for each transaction, indicate which of the following four actions is necessary by placing an $\times$ in the appropriate ☐, as so ☒.

T_1: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_2: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_3: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_4: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_5: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

T_6: ☐ No further action necessary ☐ Re-do from log entries ☐ Rollback and re-run ☐ Re-run, no rollback necessary

Name and ID number: _____

(5: 60 points total) Shown below is the sequence of read and write operations of a transaction T for a system with four data objects, a, b, c, and d.

$$\mathsf{Begin}(T) \quad \mathsf{r}(a)\mathsf{w}(a)\mathsf{r}(b)\mathsf{w}(d)\mathsf{r}(c)\mathsf{w}(c) \quad \mathsf{Commit}(T)$$

The following notation is employed:

$\mathsf{r}(x)$	Read data item x	$\mathsf{w}(x)$	Write data item x
$\mathsf{RL}(x)$	Read lock data item x	$\mathsf{WL}(x)$	Write lock data item x
$\mathsf{UL}(x)$	Unlock data item x		

For each of the augmented sequences shown below, indicate whether it obeys the *two-phase locking protocol* (2PL), the *conservative* two-phase locking protocol, the *strict* two-phase locking protocol, or the *rigorous* two-phase locking protocol. Do this by circling the appropriate key words above the sequence. Note that, in a given instance, more than one of these properties, or none at all, may apply. **Circle *all* applicable properties, even if some are implied by others.** If none of the choices applies, then circle the word *none*. You must submit your answers on this sheet.

To avoid any confusion about whether initial locks are performed before or after the beginning of the transaction, and whether final unlocks are performed before or after the commit, assume that the sequences which are underlined are executed as single, indivisible operations.

(a) The sequence below is: none 2PL conservative strict rigorous

$\underline{\mathsf{Begin}(T)\ \mathsf{WL}(a)}\ \mathsf{r}(a)\ \mathsf{w}(a)\ \mathsf{UL}(a)\ \mathsf{RL}(b)\ \mathsf{r}(b)\ \mathsf{UL}(b)\ \mathsf{WL}(d)\ \mathsf{w}(d)\ \mathsf{UL}(d)\ \mathsf{WL}(c)\ \mathsf{r}(c)\ \mathsf{w}(c)\ \underline{\mathsf{UL}(c)\ \mathsf{Commit}(T)}$

(b) The sequence below is: none 2PL conservative strict rigorous

$\underline{\mathsf{Begin}(T)\mathsf{WL}(a)\ \mathsf{WL}(b)\ \mathsf{WL}(c)\ \mathsf{WL}(d)}\ \mathsf{r}(a)\ \mathsf{w}(a)\ \mathsf{UL}(a)\ \mathsf{r}(b)\ \mathsf{w}(d)\ \mathsf{UL}(d)\ \mathsf{UL}(b)\ \mathsf{r}(c)\ \mathsf{w}(c)\ \underline{\mathsf{UL}(c)\ \mathsf{Commit}(T)}$

(c) The sequence below is: none 2PL conservative strict rigorous

$\underline{\mathsf{Begin}(T)\ \mathsf{WL}(a)}\ \mathsf{r}(a)\ \mathsf{w}(a)\ \mathsf{RL}(b)\ \mathsf{r}(b)\ \mathsf{UL}(b)\ \mathsf{WL}(d)\ \mathsf{w}(d)\ \mathsf{WL}(c)\ \mathsf{r}(c)\ \mathsf{w}(c)\ \underline{\mathsf{UL}(c)\ \mathsf{UL}(a)\ \mathsf{UL}(d)\ \mathsf{Commit}(T)}$

(d) The sequence below is: none 2PL conservative strict rigorous

$\underline{\mathsf{Begin}(T)\ \mathsf{WL}(a)}\ \mathsf{r}(a)\ \mathsf{w}(a)\ \mathsf{RL}(b)\ \mathsf{r}(b)\ \mathsf{WL}(d)\ \mathsf{w}(d)\ \mathsf{WL}(c)\ \mathsf{r}(c)\ \mathsf{w}(c)\ \underline{\mathsf{UL}(c)\ \mathsf{UL}(a)\ \mathsf{UL}(d)\ \mathsf{UL}(b)\ \mathsf{Commit}(T)}$

(e) The sequence below is: none 2PL conservative strict rigorous

$\underline{\mathsf{Begin}(T)\ \mathsf{WL}(a)}\ \mathsf{WL}(d)\ \mathsf{WL}(c)\ \mathsf{r}(a)\ \mathsf{w}(a)\ \mathsf{RL}(b)\ \mathsf{r}(b)\ \mathsf{UL}(b)\ \mathsf{w}(d)\ \mathsf{r}(c)\ \mathsf{w}(c)\ \underline{\mathsf{UL}(c)\ \mathsf{UL}(a)\ \mathsf{UL}(d)\ \mathsf{Commit}(T)}$

(f) The sequence below is: none 2PL conservative strict rigorous

$\underline{\mathsf{Begin}(T)\ \mathsf{WL}(a)\ \mathsf{RL}(b)\ \mathsf{WL}(d)\ \mathsf{WL}(c)}\ \mathsf{r}(a)\ \mathsf{w}(a)\ \mathsf{r}(b)\ \mathsf{UL}(b)\ \mathsf{w}(d)\ \mathsf{r}(c)\ \mathsf{w}(c)\ \underline{\mathsf{UL}(c)\ \mathsf{UL}(a)\ \mathsf{UL}(d)\ \mathsf{Commit}(T)}$

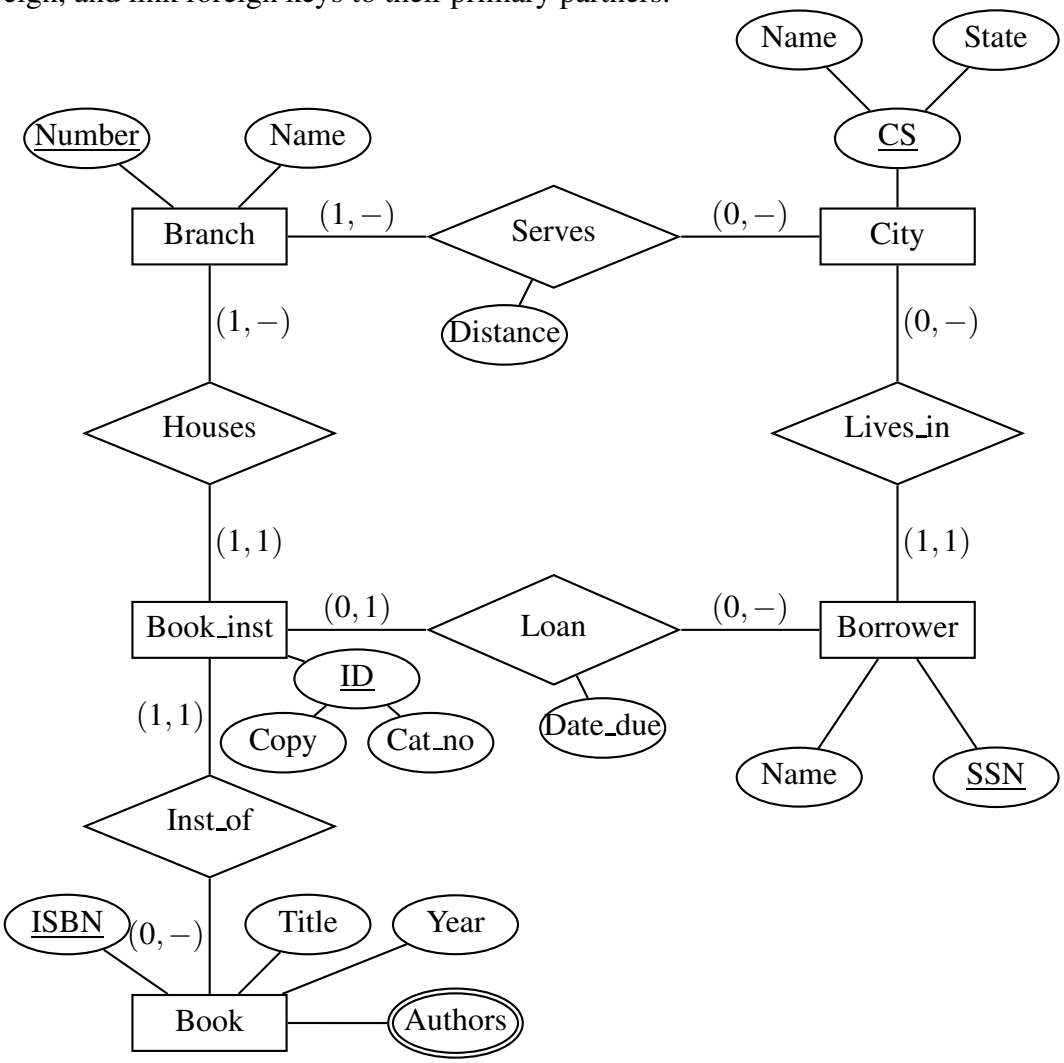
(g) The sequence below is: none 2PL conservative strict rigorous

$\underline{\mathsf{Begin}(T)\ \mathsf{WL}(a)}\ \mathsf{r}(a)\ \mathsf{w}(a)\ \mathsf{RL}(b)\ \mathsf{r}(b)\ \mathsf{WL}(d)\ \mathsf{w}(d)\ \mathsf{WL}(c)\ \mathsf{UL}(a)\ \mathsf{r}(c)\ \mathsf{w}(c)\ \underline{\mathsf{UL}(c)\ \mathsf{UL}(b)\ \mathsf{UL}(d)\ \mathsf{Commit}(T)}$

(h) The sequence below is: none 2PL conservative strict rigorous

$\underline{\mathsf{Begin}(T)\ \mathsf{WL}(a)\ \mathsf{RL}(b)\ \mathsf{WL}(d)\ \mathsf{WL}(c)}\ \mathsf{r}(a)\ \mathsf{w}(a)\ \mathsf{r}(b)\ \mathsf{w}(d)\ \mathsf{r}(c)\ \mathsf{w}(c)\ \underline{\mathsf{UL}(c)\ \mathsf{UL}(b)\ \mathsf{UL}(a)\ \mathsf{UL}(d)\ \mathsf{Commit}(T)}$

(6: 60 points) Shown below is an ER-diagram for a library database. Using the techniques developed in the course, map this diagram into an equivalent relational schema. Show all keys, primary and foreign, and link foreign keys to their primary partners.

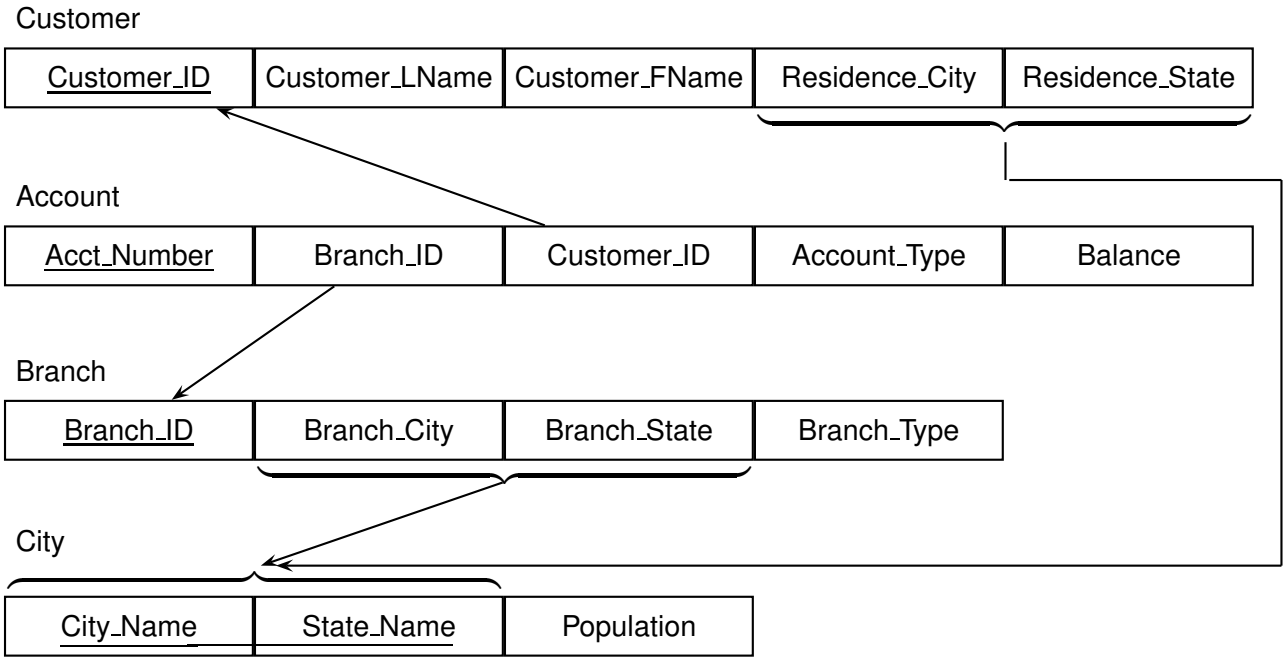


- If a primary and/or foreign key consists of more than one attribute, make sure that your notation identifies and links these composite keys as groups.
- Use design decisions which produce the simplest relational schema. If there is a choice between creating an additonal relation and adding attributes to an existing one, choose the latter. Similarly, a foreign key which cannot take on null values is to be preferred to one which can.

Notes:

- The (x,y) notation gives the minimum and maximum number of times that a given instance of the entity may participate in the relationship. Thus, $(1,1)$ means exactly one, and $(0,-)$ means any number.
- Book_inst = book instance; Inst_of = instance of; cat_no = catalog number; CS = city state.
- A *book* is an entity with an ISBN; *e.g.*, the course textbook.
- A *book instance* is a physical book; *e.g.*, your copy of the course textbook.

The following schema applies to Problems 7 through 10. In each case, the primary key is underlined, and the arrows run from foreign keys to their primary counterparts.



The following queries apply to Problems 5 through 7, and reference the schema defined above.

(a) Find the names (last and first) of those customers who have an account in a branch which is located in a city named Burlington, and also an account in a branch located in the state named Vermont. These may be, but need not be, the same account.

(b) Find the Branch_IDs of those branches which have at least two distinct customers with the same last name.

(c) Find the branch IDs of those branches in which every customer who is listed in the Customer relation and whose first name is Bob has an account.

Note: In the above schema, the attributes Customer_LName and Customer_FName identify the last and first names of the customer, respectively.

(7: 60 points total; 20 for each part) Solve each of the three queries (a) - (c) in the relational algebra. Functional operators, such as count and average, may not be used. Use the join notation described in the course notes: (The symbol $\bowtie$ is used to denote join; in the absence of subscripts it denotes the natural join; any other join conditions must be specified explicitly using subscripts.)

(8: 60 points total; 20 for each part) Solve each of the three queries (a) - (c) in the tuple relational calculus. Functional operators, such as count and average, may not be used.

(9: 60 points total; 20 for each part) Solve each of the three queries (a) - (c) using SQL. To keep the solutions simple and consistent, the following restrictions apply:

(i) Directives involving the keyword JOIN may not be used. Express the join conditions directly using conditions in the WHERE clause.

(ii) The solution must consist of a single SQL directive. Creation and subsequent use of temporary tables is not permitted.

(iii) Aggregate operators, such as COUNT, SUM, AVG, MAX, and MIN may **not** be used.

(10: 100 points total; 20 points for each part) Provide solutions, in SQL, to the following additional queries. Restrictions (i) and (ii) from Problem 8 apply here as well. On the other hand, aggregate operations may be used, but only in the SELECT and HAVING clauses of queries and subqueries.

(a) For each state having at least three distinct cities listed in the City relation, find the total population of all cities listed in the City relation for that state. List both the name of the state and the total population.

(b) For each value of Customer_LName which is listed in the Customer relation, give the total number of accounts held by customers with that last name, as well as the average balance of those accounts. If the total number of accounts for a listed last name is zero, then give the average balance as zero as well.

(c) Display the name and customer ID of those customers whose total balance in all accounts which are housed in branches located in the state of Massachusetts is greater than the balance of any single account in the entire database, regardless of where it is located.

(d) Increase by 100 the balance of each account whose owner resides in the same city as the branch housing that account. (This means the *same* city, *i.e.*, the same city and state.)

(e) Grant user George the privilege of reading (but not writing) the Customer and Account relations, and give George the right to pass on this privilege to other users as well.

(11: 80 points total) The following parameters apply to a relation which is to be stored in a B^+-tree. Assume that each node, interior or leaf, is stored in one page, and that each tuple is stored as one record.

Number of tuples in the relation	6×10^9
Maximum number of tuples per leaf node	13
Size of one tuple	See part (b)
Size of pointer to a B^+-tree node	4 bytes
Size of internal B^+-tree key	16 bytes
Page size	2048 bytes
Sequential-access pointers in the leaf nodes	8 bytes total

Answer the following. To receive credit, you must show your work, including the formulas used. (If you show just numerical computations, without an underlying formula, full credit will not be awarded.)

(a: 12 points) Compute the maximum number of indices per non-leaf node.

(b: 12 points) Suppose that a maximum of 13 records per leaf node is allowed. Compute the maximum tuple size which may be accommodated. Round this value down to the nearest integer. (Note: This is not difficult. Instead of computing the maximum number of records from the record size, as in the obligatory exericse, compute the record size from the maximum number of records. Use the same formula.)

(c: 12 points) Compute the maximum depth that such a B^+-tree may have. (The depth is defined as the length of a path from the root to a leaf.)

(d: 12 points) Compute the minimum depth that such a B^+-tree may have.

(e: 20 points) For a B^+-tree of the maximum depth computed in (c), compute the average number of indices in the interior nodes other than the root, assuming that the root contains the minimum possible number of indices, and that the leaf nodes contain the minimum possible number of records. State whether or not this value is consistent with the constraints on the minimum and maximum number of indices in an interior node of a B^+-tree.

(f: 12 points) Compute the number of interior nodes and the number of leaf nodes for the B^+-tree with the parameters found in part (e). Do this even if the values obtained in (e) are not consistent with the constraints on the minimum and maximum number of indices in an interior node of a B^+-tree.

Appendix

A useful formula for uniform (m,r,d) B-trees:

$$R(m,r,d) = (m+1) \times (r+1)^d - 1$$

with

$$\begin{aligned} m &= \text{the total number of indices at the root node.} \\ r &= \text{the total number of records in each non-root node.} \\ d &= \text{the depth of the tree; } \textit{i.e.}, \\ &\quad \text{the length of a path from the root to a(ny) leaf node.} \\ R(m,r,d) &= \text{the total number of records in the tree.} \end{aligned}$$

A useful formula for uniform (m,q,r,d) B^+-trees:

$$R(m,q,r,d) = (m+1) \times (q+1)^{d-1} \times r$$

with

$$\begin{aligned} m &= \text{the total number of indices at the root node.} \\ q &= \text{the total number of indices in each non-root, non-leaf node.} \\ r &= \text{the total number of records in each leaf node.} \\ d &= \text{the depth of the tree; } \textit{i.e.}, \\ &\quad \text{the length of a path from the root to a(ny) leaf node.} \\ R(m,q,r,d) &= \text{the total number of records in the tree.} \end{aligned}$$