Umeå University
Department of Computing Science
5DV119 — Introduction to Database Management
Examination: November 07, 2011

Name (printed): _____

Swedish ID number: _____

Computer User-ID: _____

Signature: _____

Secret code number: _____

Instructions: / Instruktioner:

This examination will be graded anonymously. This page will be removed before the instructor receives the examination for grading. The secret code number given above must therefore be written on every answer page which you turn in to the examination proctor.

Denna skrivning rättas kodad. Detta blad kommer att avskiljas innan läraren får skrivningen för rättning. Ovanstående kod måste därför finnas på samtliga svarsblad när du lämnar skrivningen till skrivvakten.

To the proctor of the examination: / Till skrivningsbevakaren:

Detach this cover sheet from the examination and put it in the envelope which is addressed to Yvonne Löwstedt, Department of Computing Science.

Avskilj detta försättsblad och stoppa i kuvert som skickas till Yvonne Löwstedt, Datavetenskap.

Umeå University
Department of Computing Science
5DV119 — Introduction to Database Management
Examination: November 07, 2011

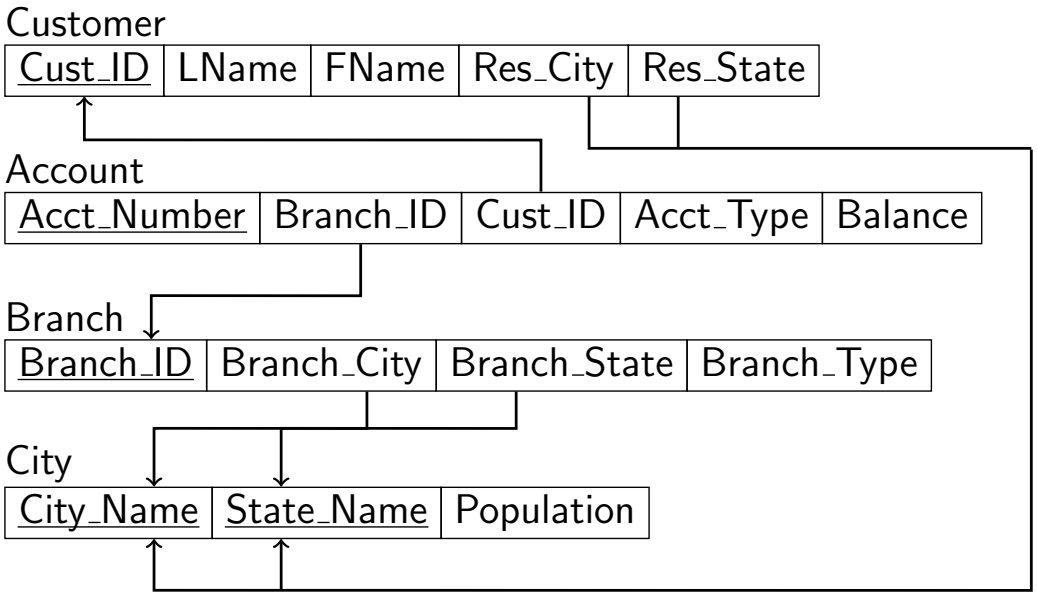
Secret code number: _____

1. Answers may be written in English or Swedish. However, all technical terms which do not have an absolutely standard representation in Swedish must be given in English.
2. An English/X - X/English dictionary may be used. No other help materials are allowed.
3. Answers must be written on the official university answer sheets which are provided the sheets in numerical order of the problems, and write on only one side of the paper. Write only the question number and your secret code number on these pages; do not write your name or ID.
4. Show your work wherever possible.
5. The examination has a total of 1000 points.
6. For each problem, you have the choice, for each part, to give a solution for that part, 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!

   For a problem with only one part, consider that part to be labelled by (a).

Prob	1	2	3	4	5	6	7	8
(a)								
(b)					■			
(c)					■			■
(d)					■	■		■
(e)					■	■	■	■

The following schema applies to Problems 1 through 4. In each case, the primary key is underlined, and the arrows run from foreign keys to their primary counterparts.
In answering the queries, it may be assumed that each relation is nonempty; that is, that each relation contains at least one tuple.



The following queries apply to Problems 1 through 3, and reference the schema defined above.

(a) Find the names (last and first) and customer ID of those customers who reside in a city with a population of at least 100000.

(b) Find the names (last and first) and customer ID of those customers who have an account of type `transaction` or an account of type `savings`.

(c) Find the names (last and first) and customer ID of those customers who have an account of type `transaction` and an account of type `savings`.

(d) Find the names (last and first) and customer ID of those customers who have at least two distinct accounts.

(e) Find the (city,state) pairs which house a branch of every type which is listed in the Branch relation.

Note: In the above schema, the attributes Customer_LName and Customer_FName identify the last and first names of the customer, respectively, while Res_City and Res_State identify the city and state of residence of the customer, respectively.

(1: 125 points total; 25 points for each part) Solve each of the five queries (a) - (e) on the preceding page using 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.)

(2: 125 points total; 25 points for each part) Solve each of the five queries (a) - (e) on the preceding page using the tuple relational calculus. Functional operators, such as count and average, may not be used.

(3: 125 points total; 25 points for each part) Solve each of the five queries (a) - (e) on the preceding page using SQL. To keep the solutions simple and consistent, the following conditions apply:

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

(ii) Standard SQL operations such as `INTERSECT` and `EXCEPT` may be used even though they are not supported by MySQL.

(iii) Uppercase and lowercase letters may be used interchangeably.

(iv) Subqueries involving SELECT are allowed only in the WHERE and HAVING fields of queries and subqueries, and in the SET fields of UPDATE commands.

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

(4: 125 points total; 25 points for each part) Provide solutions, in SQL, to the following additional queries. Conditions (i)-(iii) from Problem 3 above apply here as well. On the other hand, aggregate operations may be used, as may subqueries in the FROM clause.

(a) Find the total number of distinct branch types which are represented in the Branch relation.

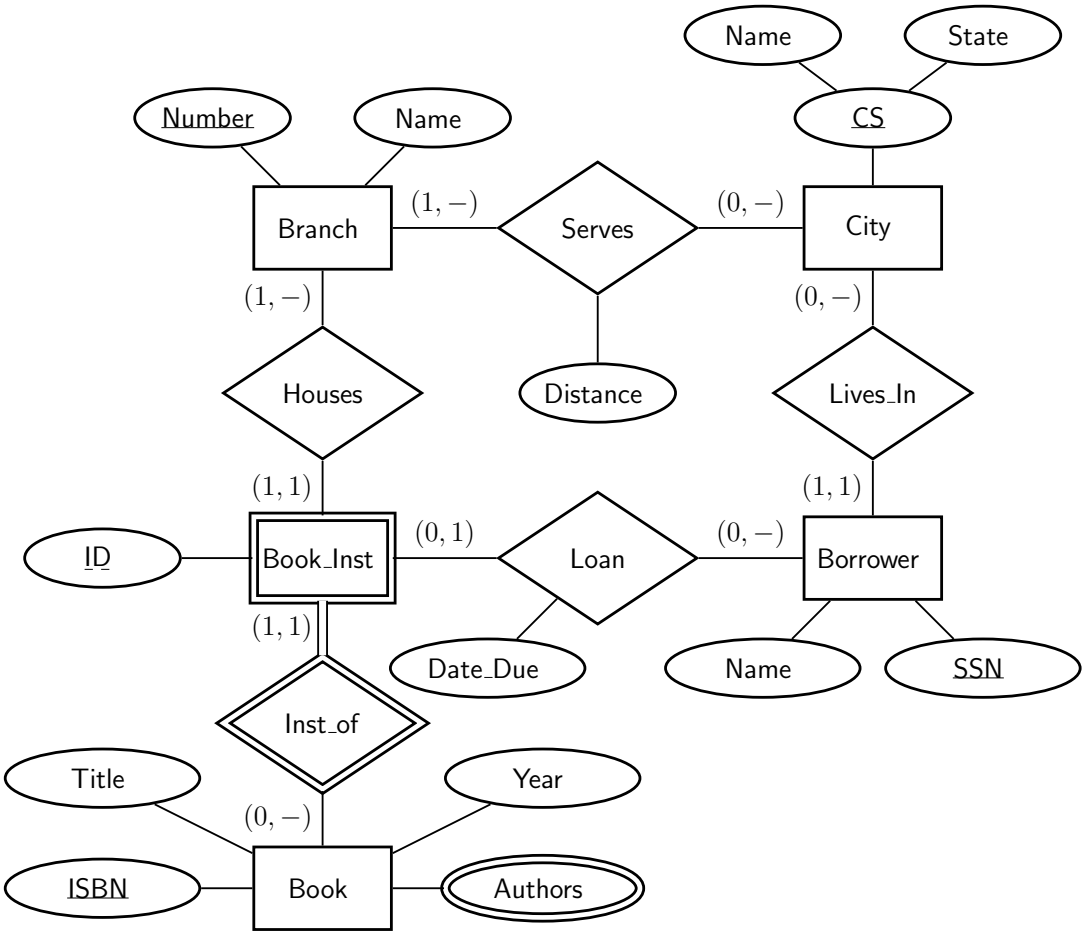
(b) For each customer, find the number of accounts held, as well as the average. minimum, and maximum balance over those accounts. Provide answers even for customers with no accounts (in which case the average is represented as zero.) Give the last and first names of the customer, as well as the customer ID.

(c) List the customers (ID and names) who have a total balance of at least 30000 in all accounts owned.

(d) Find the customer (ID and names) who has the account with the greatest balance. In case of a tie, list all such customers.

(e) Add 5% to the balance of all accounts of type `transaction`.

(5: 125 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. Although it is not strictly required to show the steps in the translation process, you will receive far more partial credit in the case of an error if you document your solution process carefully.



- 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 additional 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; 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.

(6: 125 points) Let $\mathbf{E}_0$ be the relational schema with the single relation $R[ABCDE]$, constrained by the FDs $\mathcal{F}_0 = \{A \to BC,\ BC \to D,\ D \to E\}$.

(a: 50 points) Find a lossless BCNF decomposition of $\mathbf{E}_0$ which is also dependency preserving. Draw the join tree and explain clearly why the decomposition is dependency preserving.

(b: 50 points) Find a lossless BCNF decomposition of $\mathbf{E}_0$ which is not dependency preserving. Draw the join tree and explain clearly why the decomposition is not dependency preserving.

(c: 25 points) Determine whether or not the dependency-preserving decomposition found for part (a) is independent. You must justify your answer in order to receive credit.

(7: 125 points) Let $\mathbf{E}_1$ be the relational schema with the single relation $R[ABCDE]$, constrained by the FDs $\mathcal{F}_1 = \{AB \to CD,\ C \to AD,\ E \to B\}$.

(a: 30 points) Give a canonical cover for $\mathcal{F}_1$.

(b: 30 points) Identify the candidate keys of $\mathbf{E}_1$.

(c: 35 points) Apply the synthesis algorithm to this schema to obtain a *lossless* and dependency-preserving 3NF decomposition.

(d: 30 points) Determine whether or not the decomposition identified in (c) is in BCNF. You must justify your answer with a clear explanation in order to receive credit.

(8: 125 points) Answer the following questions about security.

(a: 75 points) Explain briefly what *SQL injection* is and illustrate with a simple example.

(b: 50 points) Describe the best means of preventing SQL injection within the context of ODBC programs.