CHALMERS UNIVERSITY OF TECHNOLOGY
Department of Computer Science and Engineering

Examination in Databases, TDA357/DIT620

Thursday 12 April 2012, 08:30-12:30

Examiner:	Graham Kemp (telephone 772 54 11, room 6475 EDIT) The examiner will visit the exam room at 09:30 and 11:30.
Results:	Will be published by the end of April at the latest.
Exam review:	See course web page for time and place: http://www.cse.chalmers.se/edu/year/2011/course/TDA357/HT2011/
Grades:	Grades for Chalmers students (TDA357) are normally determined as follows: ≥ 48 for grade 5; ≥ 36 for grade 4; ≥ 24 for grade 3. Grades for GU students (DIT620) are normally determined as follows: ≥ 42 for grade VG; ≥ 24 for grade G.
Help material:	One A4 sheet with hand-written notes. You may write on both sides of that sheet. That sheet must be handed in with your answers to the exam questions. English language dictionaries are allowed.

Specific instructions:

- Please answer in English where possible. You may clarify your answers in Swedish if you are not confident you have expressed yourself correctly in English.
- Begin the answer to each question on a new page.
- Write clearly; unreadable = wrong!
- Fewer points are given for unnecessarily complicated solutions.
- Indicate clearly if you make any assumptions that are not given in the question.
- Write the page number and question number on every page.

Question 1. Consider the following domain description.
12 p

> A hospital wants to use a database to manage information about its patients. The hospital contains several wards, and many of the wards contain several beds. Each ward is identified by its number, and each bed is identified by its number within a ward (e.g. bed number 4 in ward 32). The floor on which a ward is located should be stored in the database. A patient is identified by their personal identification number. The name, gender ('M' or 'F') and year of birth of each patient should be represented in the database. The bed assigned to each patient should be represented in the database, and the date on which the patient was assigned to that bed should be recorded. During their stay in hospital, a patient might be moved from one bed to another, possibly moving several times. During their stay in hospital, patients will undergo routine tests. The date and hour of each occasion when these tests are performed on a patient should be recorded, and for each of these tests the patient's temperature and blood pressure will be measured and recorded in the database. A patient will normally undergo these routine tests several times during their stay in hospital.

a) Draw an E-R diagram that correctly models this domain.
(6p)

b) Translate this E-R diagram into a set of relations, clearly marking all references and keys.
(6p)

Question 2. Suppose we have relation $R(A, B, C, D)$ and functional dependencies $AC \to D$, $B \to C$.
9 p

a) By considering the closures of all subsets of attributes, find **all** non-trivial FDs, superkeys and keys. State which FDs violate BCNF.
(4p)

b) Decompose relation R to BCNF in two different ways:
— by first decomposing on $AC \to D$
— by first decomposing on $B \to C$.

Show each step in the normalisation process, and at each step indicate which functional dependency is being used.
(3p)

c) State, with reasons, whether the FDs $AC \to D$ and $B \to C$ violate Third Normal Form (3NF).
(2p)

Question 3. A database system used to record seat bookings at a concert hall has the following relations:

11 p

Seats(row, number)
Bookings(ref, person, day, month, year, paid)
BookedSeats(booking, row, number)

A seat is identified by its number within a row; rows are identified by a single letter ('A', 'B', etc.).
Booking reference numbers (attribute *ref*) are unique integers.
Attribute *person* is the person identification number of the individual making a booking.
Attributes *day*, *month* and *year* identify the date of the concert to which the booking applies.
Attribute *paid* indicates whether payment has been made for this booking. This attribute either has the value 'Y' (yes) or 'N' (no). 'N' is the default value for this attribute.
Each booking can include several booked seats (e.g. booking 303 might include seats 1, 2 and 3 in row 'B').

a) Suggest keys and references for these relations.
Write SQL statements that create these relations with constraints in a DBMS.
(4p)

b) A seat cannot be booked more than once for the same concert.
Write an assertion that checks this.
(3p)

c) If an attempt is made to insert a new row into relation *BookedSeats*, this can only be done if that booking has not yet been paid. If the booking has already been paid, then a new booking must be created (with a new booking reference number) and the seat should be booked under this new booking number.
Write a trigger that implements this.
(4p)

Question 4. Assume the same relations as in Question 3:

6 p

Seats(row, number)
Bookings(ref, person, day, month, year, paid)
BookedSeats(booking, row, number)

a) Write a relational algebra expression that finds which seats have been paid for, for the concert on 12 April 2012 (the result should be a list of seat rows and numbers).
(2p)

b) Write a relational algebra expression that finds the person identification number of the person(s) who has booked the most seats in total (these seats might have been booked under different booking reference numbers, and could be for concerts on different dates).
(4p)

Question 5. Assume the same relations as in Question 3:

9 p

Seats(row, number)
Bookings(ref, person, day, month, year, paid)
BookedSeats(booking, row, number)

a) Write an SQL query that finds how many seats have been paid for, for the concert on 12 April 2012.
(3p)

b) Write an SQL query that finds the seats that are available (i.e. not booked) for the concert on 12 April 2012 (the result should be a list of seat rows and numbers).
(3p)

c) Write an SQL query that finds the person identification number of the person(s) who has booked the most seats in total (these seats might have been booked under different booking reference numbers, and could be for concerts on different dates).
(3p)

Question 6. Suppose we have relation *Prices(item,price)* and the following PSM procedure:

4 p

```
CREATE PROCEDURE DoublePrice(IN i VARCHAR(20))
DECLARE x INTEGER;
BEGIN
  -- Step 1
    SELECT price INTO x FROM Prices WHERE item = i;
  -- Step 2
    UPDATE Prices SET price = x * 2 WHERE item = i;
END;
```

a) What privileges are needed by the author of procedure DoublePrice?
What privileges are needed by users who want to call procedure DoublePrice?
(2p)

b) Suppose relation *Prices* contains the tuple (i222,100).
Suppose users A and B issue the command

```
CALL DoublePrice('i222')
```

at the same time, and that the procedures are run as transactions with isolation level READ COMMITTED.
Give all possible interleavings of the steps (refer to the steps of A's transaction as A_1 and A_2, and the steps of B's transaction as B_1 and B_2), and state the corresponding outcomes.
(2p)

Question 7. Consider the following piece of XML:

9 p

```
<ConcertHall>
  <Seats>
    <Seat row="A" num="1" />
    <Seat row="A" num="2" />
    <Seat row="A" num="3" />
    <Seat row="A" num="4" />
    <Seat row="B" num="1" />
    <Seat row="B" num="2" />
    <Seat row="B" num="3" />
    <Seat row="B" num="4" />
  </Seats>
  <Bookings>
    <Booking ref="301" person="700707-7777" day="12" month="April" year="2012">
      <BookedSeat row="A" num="1" />
      <BookedSeat row="A" num="2" />
    </Booking>
    <Booking ref="302" person="800808-8888" day="12" month="April" year="2012">
      <BookedSeat row="A" num="3" />
      <BookedSeat row="A" num="4" />
    </Booking>
    <Booking ref="303" person="900909-9999" day="1" month="May" year="2012">
      <BookedSeat row="B" num="1" />
      <BookedSeat row="B" num="2" />
      <BookedSeat row="B" num="3" />
    </Booking>
  </Bookings>
</ConcertHall>
```

a) Write a Document Type definition (DTD) for the XML that is given above.

(3p)

b) Write an XPath expression that finds the booking elements that include a booked seat with number 1, i.e.

```
<Booking ref="301" person="700707-7777" day="12" month="April" year="2012">
  <BookedSeat row="A" num="1"/>
  <BookedSeat row="A" num="2"/>
</Booking>
<Booking ref="303" person="900909-9999" day="1" month="May" year="2012">
  <BookedSeat row="B" num="1"/>
  <BookedSeat row="B" num="2"/>
  <BookedSeat row="B" num="3"/>
</Booking>
```

(1p)

c) Write an XQuery expression that finds the total number of seats in the concert hall. The result of this query should look as follows:

```
<Result>8</Result>
```

(2p)

d) Write an XQuery expression that finds the person identifiers of people who have booked seats for the concert on 12 April 2012, together with information about the seats that they have booked. The result of this query should look as follows:

```
<Result>
  <BookedSeat person="700707-7777" seat="1" row="A"/>
  <BookedSeat person="700707-7777" seat="2" row="A"/>
  <BookedSeat person="800808-8888" seat="3" row="A"/>
  <BookedSeat person="800808-8888" seat="4" row="A"/>
</Result>
```

(3p)