CHALMERS UNIVERSITY OF TECHNOLOGY
Department of Computer Science and Engineering

Examination in Databases, TDA357/DIT620

Thursday 20 December 2012, 14:00-18:00

Examiner:	Graham Kemp (telephone 772 54 11, room 6475 EDIT) The examiner will visit the exam room at 15:00 and 17:00.
Results:	Will be published by the middle of January at the latest.
Exam review:	See course web page for time and place: http://www.cse.chalmers.se/edu/year/2012/course/TDA357/HT2012/
Grades:	Grades for Chalmers students (TDA357) are normally determined as follows: $\geq$ 48 for grade 5; $\geq$ 36 for grade 4; $\geq$ 24 for grade 3. Grades for GU students (DIT620) are normally determined as follows: $\geq$ 42 for grade VG; $\geq$ 24 for grade G.
Help material:	One A4 sheet with hand-written notes. You may write on both sides of that sheet. If you bring a sheet, it 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

An estate agent (Swedish: fastighetsmäklare) wants to use a database to manage information about the properties it sells and its clients. Each property that the agent is trying to sell is identified by a reference code. The address of each property should be stored in the database, and also the property's living area (in m^2) and guide price (in SEK).

Each of the estate agent's clients has a unique client identifier. The client's name and telephone number should be stored in the database. A client can be someone who is selling a property, or who is interested in buying a property or both (i.e. a seller of one property (or more than one property) and a prospective buyer for others). Information about who is selling a property should be stored in the database.

When selling a property, the estate agent arranges one or more viewings — occasions when prospective buyers can visit the property that is for sale. Each viewing of a property takes place on a different date (although different properties could have viewings on the same date). Clients can attend several viewings, and may view the same property more than once. The agent wants to record information about which clients attend each viewing.

The agent wants to store information about the bids that clients give on a property. The amount bid, and the date and time that the bid was given, should be stored in the database. During the selling process, each client may bid on the same property more than once.

When a property is sold, the buyer and the purchase price should be recorded in the database.

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 $R(A, B, C, D, E, F, G)$ and functional dependencies
10 p

$BC \to D$, $DE \to F$, $FA \to B$, $BC \to G$.

a) Relation R has three *keys*. State, with reasons, which two of the following are **not** keys of R.

$ABCD$, $ABCE$, $ACDE$, $ACDEG$, $ACEF$.

(2p)

b) Decompose relation R to BCNF. Show each step in the normalisation process, and at each step indicate which functional dependency is being used.
(3p)

c) State, with reasons, which FD(s) of relation R violate Third Normal Form (3NF).
(2p)

d) Decompose relation R to 3NF.
(3p)

Question 3. 11 p

A database system used by a hospital to record information about patients and wards has the following relations:

Wards(number, numBeds)
Patients(pid, name, year, gender)
PatientInWard(pid, ward)
Tests(patient, testDate, testHour, temperature, heartRate)

A ward is identified by its number. Attribute *numBeds* is the number of beds in that ward.

Patients are identified by their personal identification number. The name, year of birth and gender ('M' or 'F') of each patient is stored in the Patients relation.

The ward to which each patient is assigned is stored in relation PatientInWard.

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 are recorded, and for each of these tests the patient's temperature and heart rate are measured and recorded in the database. A patient will normally undergo these routine tests several times during their stay in hospital.

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

b) The number of patients in a ward cannot exceed the number of beds in that ward.
Write an assertion that checks this.
(3p)

c) If an attempt is made to insert a new row into relation *PatientInWard*, and that ward is already full, then the patient should instead be assigned to a ward that has an available bed. If there are several wards with available beds, then the patient should be assigned to the one with the lowest ward number.
Write a trigger that implements this.
(When writing the trigger you may assume that the view described in question 5(b) has already been defined.)
(4p)

Question 4. 7 p

Assume the same relations as in Question 3:

Wards(number, numBeds)
Patients(pid, name, year, gender)
PatientInWard(pid, ward)
Tests(pid, testTime, temperature, heartRate)

a) Write a relational algebra expression that finds the temperature and heart rate measured in each test carried out on patients born before 1950.
(2p)

b) Write a relational algebra expression that finds the years for which the number of male patients born in that year is higher than the number of female patients born in that year.
(For full marks your solution should deal with years for which there are male patients but no female patients.)
(5p)

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

Wards(number, numBeds)
Patients(pid, name, year, gender)
PatientInWard(pid, ward)
Tests(pid, testTime, temperature, heartRate)

a) Write an SQL query that finds the temperature and heart rate measured in each test carried out on patients born before 1950.
(3p)

b) Create a view *FreeBeds(ward, numBeds)* where ward is a ward number, and numBeds is the number of available beds in that ward.
(3p)

c) Write an SQL query that finds the years for which the number of male patients born in that year is higher than the number of female patients born in that year.
(For full marks your solution should deal with years for which there are male patients but no female patients.)
(4p)

Question 6. A system for booking cinema seats has a transaction T with the following three steps:
4 p

T_1: list available seats
T_2: book a seat
T_3: confirm the booking that was made in step T_2

In database transactions, what are *dirty reads*? Refer to transaction T in your answer.

Suppose users A and B both run transaction T at the same time (refer to these executing transactions as T_A and T_B).
What isolation level(s) of T_A and T_B could result in T_A performing dirty reads?

(4p)

Question 7. Consider the following piece of XML:

6 p

```
<Hospital>
  <Patients>
    <Patient pid="p001" name="smith" />
    <Patient pid="p002" name="jones" />
    <Patient pid="p003" name="green" />
  </Patients>
  <Tests>
    <Test pid="p001" time="07:00" temp="36.8" heartRate="75" />
    <Test pid="p001" time="11:00" temp="36.9" heartRate="77" />
    <Test pid="p001" time="15:00" temp="36.7" heartRate="74" />
    <Test pid="p002" time="10:00" temp="36.8" heartRate="66" />
    <Test pid="p003" time="17:00" temp="36.8" />
    <Test pid="p003" time="18:00" heartRate="60" />
  </Tests>
</Hospital>
```

a) Write a Document Type Definition (DTD) for the XML that is given above. (2p)

b) Assuming that the XML shown above is in file *hospital.xml*, write an XQuery expression that constructs the following result:

```
<Result>
  <Patient name="smith" pid="p001">
    <Test pid="p001" time="07:00" temp="36.8" heartRate="75"/>
    <Test pid="p001" time="11:00" temp="36.9" heartRate="77"/>
    <Test pid="p001" time="15:00" temp="36.7" heartRate="74"/>
  </Patient>
  <Patient name="jones" pid="p002">
    <Test pid="p002" time="10:00" temp="36.8" heartRate="66"/>
  </Patient>
  <Patient name="green" pid="p003">
    <Test pid="p003" time="17:00" temp="36.8"/>
    <Test pid="p003" time="18:00" heartRate="60"/>
  </Patient>
</Result>
```

(4p)

CHALMERS UNIVERSITY OF TECHNOLOGY
Department of Computer Science and Engineering

Examination in Databases, TDA357/DIT620

Thursday 20 December 2012, 14:00-18:00

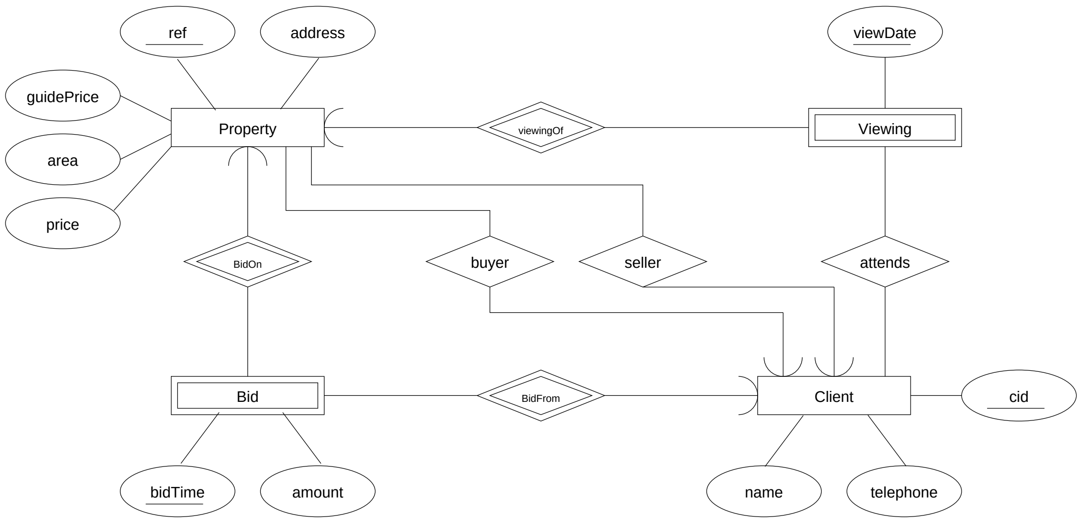
Solutions

Updated 2012-12-22

Question 1. 12 p

a) (Here is one suggestion. Several other designs are also accepted. For example, modelling 'SoldProperties' as a subclass of property, or modelling 'Buyers' and 'Sellers' as subclasses of client, to model the different roles that clients can have.)

E-R diagram:



b) $Clients(\underline{cid}, name, telephone)$

$Properties(\underline{ref}, address, guidePrice, area, price, seller, buyer)$
$seller \rightarrow Clients.cid$
$buyer \rightarrow Clients.cid$

$Viewings(\underline{property}, \underline{viewDate})$
$property \rightarrow Properties.ref$

$Attends(\underline{property}, \underline{viewDate}, \underline{client})$
$(property, viewDate) \rightarrow Viewings.(property, viewDate)$

$Bids(\underline{property}, \underline{client}, \underline{bidTime}, amount)$
$property \rightarrow Properties.ref$
$client \rightarrow Clients.cid$

Question 2. 10 p

a) $ABCD$ — does not identify all attributes.
$ACDEG$ — this is a superkey but not a key, since attribute G can be removed and the resulting set of attributes is a key.

b)
```
Decompose on BC -> D
{BC}+ = {BCDG}

        R1(_B,_C,D,G)
        R2(B,C,A,E,F)
            B,C -> R1.(B,C)

Decompose R2 on FA -> B
{FA}+ = {FAB}

        R21(_F,_A,B)
        R22(F,A,C,E)
            F,A -> R21.(F,A)

Key of R22 is FACE
```

c) $BC \rightarrow G$
Left side is not a superkey of R, and G is not prime in R.

d) $R1(B, C, D, G)$
$R2(D, E, F)$
$R3(F, A, B)$
$R4(F, A, C, E)$

Question 3. 11 p

a) $Wards(\underline{number}, numBeds)$
$Patients(\underline{pid}, name, year, gender)$
$PatientInWard(\underline{pid}, ward)$
$pid \rightarrow Patients.pid$
$ward \rightarrow Wards.num$
$Tests(\underline{patient}, \underline{testDate}, \underline{testHour}, temperature, heartRate)$
$patient \rightarrow Patients.pid$

```
CREATE TABLE Wards (
    num         INT PRIMARY KEY,
    numBeds     INT
);

CREATE TABLE Patients (
    pid        CHAR(10) PRIMARY KEY,
    name       VARCHAR(30),
    year       INT,
    gender     CHAR(1) CHECK (gender IN ('F','M'))
);

CREATE TABLE PatientInWard (
    pid        CHAR(10),
    ward       INT,
    PRIMARY KEY (pid),
    FOREIGN KEY (pid) REFERENCES Patients(pid)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
    FOREIGN KEY (ward) REFERENCES Wards(num)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

CREATE TABLE Tests (
    patient     CHAR(10),
    testDate    DATE,
    testHour    INT,
    temperature REAL,
    heartRate   INT,
    PRIMARY KEY (patient, testDate, testHour),
    FOREIGN KEY (patient) REFERENCES Patients(pid)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);
```

b)

```
CREATE ASSERTION NotOverFullWard CHECK
  ( NOT EXISTS (
        SELECT    num
        FROM      Wards JOIN PatientInWard ON num=ward
        GROUP BY  num, numBeds
        HAVING    numBeds < COUNT(pid)
  ) );
```

c)

```
CREATE TRIGGER WardFull
BEFORE INSERT ON PatientInWard
REFERENCING NEW AS new
FOR EACH ROW
DECLARE numAvailable INT;
        availableWard INT;
BEGIN
    SELECT  numBeds INTO numAvailable
    FROM    FreeBeds
    WHERE   ward = :new.ward;

    IF numAvailable = 0 THEN
        SELECT MIN(ward) into availableWard
        FROM FreeBeds
        WHERE numBeds > 0;

        :new.ward := availableWard;
    END IF;
END;
```

Question 4. 7 p

a) Unfortunately the attributes names in Questions 3 and 4 are inconsistent, so we accept either:

$\pi_{temperature,heartRate}(\sigma_{year<1950}(Patients) \bowtie Tests)$

or:

$\pi_{temperature,heartRate}(\sigma_{year<1950}(Patients) \bowtie_{patient=pid} Tests)$

(Similarly, we accept alternative solutions for Question 5(a).)

b)

$$
\begin{aligned}
&(\pi_{year}\\
&\quad(\sigma_{m>f}\\
&\qquad(\\
&\qquad\quad\gamma_{year,COUNT(pid)ASm}(\sigma_{gender='M'}(Patients))\\
&\qquad\bowtie\\
&\qquad\quad\gamma_{year,COUNT(pid)ASf}(\sigma_{gender='F'}(Patients))\\
&\qquad)\\
&\quad)\\
&)\\
&\cup\\
&(\\
&\quad(\pi_{year}(\sigma_{gender='M'}(Patients)))\ \setminus\ (\pi_{year}(\sigma_{gender='F'}(Patients)))\\
&)
\end{aligned}
$$

Question 5. 10 p

a)
```
SELECT  temperature, heartRate
FROM    Patients, Tests
WHERE   pid = patient and year < 1950
```

b)
```
CREATE VIEW FreeBeds AS
  SELECT   num as ward, numBeds - COUNT(pid) AS numBeds
  FROM     Wards LEFT OUTER JOIN PatientInWard ON ward = num
  GROUP BY num, numBeds
```

c)
```
WITH
  R1 AS
      ( SELECT    year, COUNT(pid) AS m
        FROM      Patients
        WHERE     gender = 'M'
        GROUP BY  year ),
  R2 AS
      ( SELECT    year, COUNT(pid) AS f
        FROM      Patients
        WHERE     gender = 'F'
        GROUP BY  year ),
  R3 AS
      ( SELECT    year
        FROM      R1 NATURAL JOIN R2
        WHERE     m > f ),
 R4 AS
      ( SELECT    year
        FROM      Patients
        WHERE     gender = 'M' ),
  R5 AS
      ( SELECT    year
        FROM      Patients
        WHERE     gender = 'F' )

SELECT year FROM R3 UNION (SELECT year FROM  R4 MINUS SELECT year FROM R5)
```

Using Oracle's NVL function, we could have:

```
WITH
  R1 AS
      ( SELECT    year, COUNT(pid) AS m
        FROM      Patients
        WHERE     gender = 'M'
        GROUP BY  year ),
  R2 AS
      ( SELECT    year, COUNT(pid) AS f
        FROM      Patients
        WHERE     gender = 'F'
        GROUP BY  year ),
  R3 AS
      ( SELECT year, m, NVL(f,0) AS f
        FROM R1 NATURAL LEFT OUTER JOIN R2 )

SELECT year FROM R3 WHERE m>f;
```

Question 6. 4 p

A dirty read can occur when one transaction reads a data value that has been modified by another, before that other transaction commits the change. For example, if T_B modifies the cinema seat bookings by booking a seat (in step T_2), and this modified value is read by T_A (in step T_1), and then transaction T_B rolls back, undoing the change, then transaction T_A will have performed a dirty read.

In this case, a dirty read can occur when T_A runs at isolation level READ UNCOMMITTED, and T_B runs at any isolation level.

Question 7. 6 p

a)

```
<!DOCTYPE Hospital [

<!ELEMENT Hospital (Patients, Tests) >

<!ELEMENT Patients (Patient*) >
  <!ELEMENT Patient EMPTY >
    <!ATTLIST Patient
      pid       ID    #REQUIRED
      name      CDATA #REQUIRED >

<!ELEMENT Tests (Test*) >
  <!ELEMENT Test EMPTY >
    <!ATTLIST Test
      pid       IDREF #REQUIRED
      time      CDATA #REQUIRED
      temp      CDATA #IMPLIED
      heartRate CDATA #IMPLIED >

]>
```

b)

```
<Result>
  {
    let $h := doc("hospital.xml")
    for $p in $h//Patient
    let $tests := $h//Test[@pid = $p/@pid]
    return <Patient pid="{$p/@pid}" name="{$p/@name}">{$tests}</Patient>
  }
</Result>
```