CHALMERS UNIVERSITY OF TECHNOLOGY
Department of Computer Science and Engineering

Examination in Databases, TDA357/DIT620

Saturday 19 December 2009, 08:30-12:30

Examiner:	Graham Kemp (telephone 772 5411, room 6475 EDIT) The examiner will visit the exam room at 09:30 and 11:30.
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/course/TDA357/
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. That sheet must be handed in with your answers to the exam questions. English language dictionaries are allowed.

If you have not yet completed the course evaluation form, please do this after the exam.

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 science research council processes grant proposals that are sent in by researchers. The research council wants to use a database to record information about the proposals that it receives. Each proposal is allocated to one of the research council's panels (e.g. the Computing panel or the Biology panel); the panels have unique names. Each panel allocates a serial number to each proposal it receives (integers starting from one) in each year (so Computing panel will have a proposal with serial number "1" in 2008 and another with serial number "1" in 2009, and the Biology panel will also have a proposal with serial number "1" in 2009). Within each panel, a proposal is uniquely identified by the proposal's serial number, and the year in which it is processed.
>
> Researchers are identified by unique personal identification numbers, and the research council wants to store each researcher's name and the name of the organisation where they work. Researchers play several roles: they can be applicants and/or panel members and/or reviewers. Each panel consists of several researchers, but a researcher can only be a member of at most one panel. Each proposal will be reviewed by several researchers (who might or might not be panel members), and a researcher might be asked to review several proposals.
>
> Each proposal has a title (but the titles might not be unique). Each proposal also has a main applicant. There are two kinds of research proposal that are processed by the research council. The first kind is *project proposals*. In addition to having a main applicant, a project proposal can have several co-applicants. The second kind is *travel proposals*. Travel proposals do not have co-applicants, but the destination of the planned travel should be recorded with these.

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 $R(A, B, C, D, E, F, G)$ with keys ABE, BCE and BEF, and functional dependencies $AB \to C$, $B \to D$, $CE \to F$, $ABE \to F$, $ABE \to G$, $D \to G$, $F \to A$.
10 p

a) i) State, with reasons, which FDs violate BCNF. (1p)

ii) Decompose relation R to BCNF. Show each step in the normalisation process, and at each step indicate which functional dependency is being used. Indicate keys and references for the resulting relations. (4p)

b) i) Which attributes of R are prime? (1p)

ii) State, with reasons, which FDs violate 3NF. (1p)

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

Question 3. A database system used by a company's personnel department has the following relations:
9 p

Employees(empId, name, year, salary, entitlement, branch)
ParentalLeave(employee, startDay, startYear, endDay, endYear)

Employee identifiers (*empId*) are unique. Attribute *year* is the employee's year of birth. Attribute *entitlement* is the number of annual vacation days to which the employee is entitled. Employees have 30 days of annual vacation entitlement by default. *Branch* is the name of the city where the branch is located (assume that there is only one branch in each city). The personnel department keeps a record of all periods of parental leave taken by employees. The attributes *startDay* and *endDay* are integers in the range 1-366 that represent the day within the year. For each period of parental leave, the start date must be before the end date.

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

b) It is the company's policy that the salary difference between the highest paid and lowest paid employees in the company should not be more than 10000. Write an assertion that checks this.
(2p)

c) The company's policy on vacation days changes so that all employees who were born before 1970 have at least 35 days annual vacation entitlement.
Write a trigger that, when a new employee is added to the Employees relation, sets the vacation entitlement for that employee to be 35, if they were born before 1970 and the given entitlement is less than 35.
(3p)

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

Employees(empId, name, year, salary, entitlement, branch)
ParentalLeave(employee, startDay, startYear, endDay, endYear)

a) Write a relational algebra expression that finds the names of employees who had a period of parental leave that started in 2007.
(2p)

b) Write a relational algebra expression that finds the employee(s) in each branch who have the highest salary in their branch. The result should contain the employee's name, the branch and the salary, and the result should be sorted by branch name.
(3p)

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

Employees(empId, name, year, salary, entitlement, branch)
ParentalLeave(employee, startDay, startYear, endDay, endYear)

a) Write an SQL query that finds the employee identifier and name of each employee who had a period of parental leave that started in one year and ended in another year. The result should not contain duplicate rows.
(2p)

b) Write an SQL query that finds the names of pairs of employees who have the same year of birth and vacation entitlement, but have different salaries. Each pair of employees who satisfy the condition should be listed only once in the result.
(2p)

c) Create a view $V(branch, year, numParents)$ which contains the number of employees (the third column in the view) at that *branch* who has a period of parental leave that started in that *year*.
(3p)

Question 6. Suppose relation $Lectures(course, day, hour, room)$ contains the following data:
3 p

course	**day**	**hour**	**room**
TDA357	Monday	8	HB3
TDA357	Thursday	8	HB3
TIN090	Tuesday	13	VR

Suppose transactions T1 and T2 have the following steps:

T1:

$Step\ T1_A :\ get\ count\ of\ lectures\ in\ HB3\ into\ X$
$Step\ T1_B :\ get\ count\ of\ lectures\ in\ HB3\ into\ Y$
$Step\ T1_C :\ print\ X, Y$

T2:

$Step\ T2_A :\ delete\ tuple\ (TDA357, Monday, 8, HB3)$
$Step\ T2_B :\ insert\ tuple\ (TIN090, Friday, 13, HB3)$

a) What are the possible outcomes if both T1 and T2 are run as serializable transactions? Explain your answer. (1p)

b) What are the possible outcomes if both T1 and T2 are run with isolation level `REPEATABLE READ`? Explain your answer. (2p)

Question 7. 6 p

Suppose we have relation $PassedCourses(courseId, studentId, grade)$, and that this relation is stored in 30 disc blocks.

Each course has records stored in 10 disc blocks, on average. Similarly, on average, each student has records stored in five disc blocks.

Suppose that three kinds of task are performed on this relation:

- task 1: inserting a new row;
- task 2: finding the *courseId* and *grade* for a given *studentId*;
- task 3: finding the *studentId* and *grade* for a given *courseId*.

a) For each of these tasks, state how many disc block transfers will be needed if:

   i) there are no indexes
   (1p)

   ii) there is an index on *courseId* (assume that this index fits into a single disc block)
   (1p)

   iii) there is an index on *studentId* (assume that this index fits into a single disc block)
   (1p)

   iv) both of the indexes mentioned in (ii) and (iii) are used.
   (1p)

b) Suppose that 50% of the operations performed on this relation are inserting new rows (task 1), that 40% of the operations are finding information for a given student (task 2) and that the remaining 10% are finding information for a given course (task 3).

   In this case, is it better to have no indexes, an index on only *courseId*, an index on only *studentId*, or both of those indexes?
   Show the calculations that support your answer.
   (2p)

Question 8. Suppose relation Employees is as follows:
8 p

name	**branch**	**salary**
Andersson	3	35000
Jonsson	3	25000
Larsson	2	32000

and relation Branches is as follows:

number	**city**
1	Stockholm
2	Paris
3	London

Instead of using a relational database to store this information, suppose we now want to store the same information in an XML format.

a) Write a piece of XML that contains the same information as the relations that are given above.
(3p)

b) Give a suitable DTD for the XML in your answer to part (a).
(2p)

c) Assuming the XML in your answer to part (a), Write an XPath expression that finds elements that contain the city name "Paris".
(1p)

d) Assuming the XML in your answer to part (a), write an XQuery expression that finds the names of employees whose salary is more than 30000, and the cities where they work, i.e.

```
<Result>Andersson: London</Result>
<Result>Larsson: Paris</Result>
```

(2p)

CHALMERS UNIVERSITY OF TECHNOLOGY
Department of Computer Science and Engineering

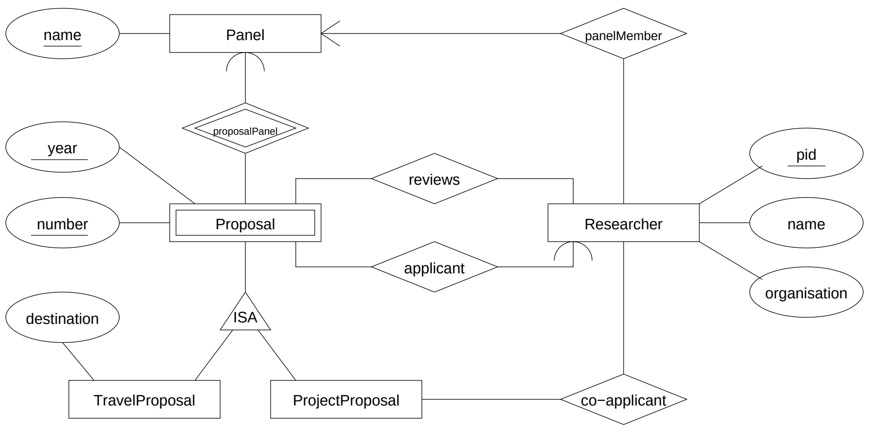
Examination in Databases, TDA357/DIT620

Saturday 19 December 2009, 08:30-12:30

Solutions

Updated 2013-12-13

Question 1. a) E-R diagram:
12 p



b) $Researchers(\underline{pid}, name, organisation)$

$Panels(\underline{name})$

$Proposals(\underline{panel}, \underline{year}, \underline{number}, applicant)$
$\quad panel \rightarrow Panels.name$
$\quad applicant \rightarrow Researchers.pid$

$TravelProposals(\underline{panel}, \underline{year}, \underline{number}, destination)$
$\quad (panel, year, number) \rightarrow Proposals.(panel, year, number)$

$ProjectProposals(\underline{panel}, \underline{year}, \underline{number})$
$\quad (panel, year, number) \rightarrow Proposals.(panel, year, number)$

$CoApplicants(\underline{panel}, \underline{year}, \underline{number}, \underline{researcher})$
$\quad (panel, year, number) \rightarrow ProjectProposals.(panel, year, number)$
$\quad researcher \rightarrow Researchers.pid$

$Reviews(\underline{panel}, \underline{year}, \underline{number}, \underline{researcher})$
$\quad (panel, year, number) \rightarrow Proposals.(panel, year, number)$
$\quad researcher \rightarrow Researchers.pid$

$PanelMembers(\underline{researcher}, panel)$
$\quad researcher \rightarrow Researchers.pid$
$\quad panel \rightarrow Panels.name$

Question 2. a) i)
10 p

```
AB -> C
B -> D
CE -> F
D -> G
F -> A
```

ii)

```
Decompose on AB -> C
{AB}+ = {ABCDG}

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

Decompose R1 on B->D
{B}+ = {BDG}

        R11(_B,D,G)
        R12(_A,_B,C)
                B -> R11.B

Decompose R11 on  D->G
{D}+ = {DG}

        R111(_D,G)
        R112(_B,D)
                D -> R111.D

Decompose R2 on F -> A
{F}+ = {AF}

        R21(A,_F)
        R22(_B,_E,_F)
                F -> R21.F
```

b) i)

```
A,B,C,E and F are prime
```

ii)

```
B -> D
D -> G
```

iii)

```
Compute the minimal closure of F
        (Remove A->C if we have A->B and B->C, etc.)

Group together FDs with the same LHS

        ABE -> FG

For each group, create a relation with the LHS as the key.

        R1(A,B,C)
        R2(B,D)
        R3(C,E,F)
        R4(A,B,E,F,G)
        R5(D,G)
        R6(A,F)

If no relation contains a key of R, add one relation containing only
a key of R.

        Relation R4 contains a key of R.
```

Question 3. a) $Employees(\underline{empId}, name, year, salary, entitlement, branch)$
9 p
$ParentalLeave(\underline{employee}, \underline{startDay}, \underline{startYear}, endDay, endYear)$
$employee \rightarrow Employees.empId$

```
CREATE TABLE Employees (
    empId              INT PRIMARY KEY,
    name               VARCHAR(40),
    year               INT,
    salary             INT,
    entitlement        INT DEFAULT 30,
    branch             VARCHAR(40),
);

CREATE TABLE ParentalLeave (
    employee           INT,
    startDay           INT,
    startYear          INT,
    endDay             INT,
    endYear            INT,
    PRIMARY KEY (employee, startDay, startYear),
    FOREIGN KEY (employee) REFERENCES Employees(empId)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
    CONSTRAINT CheckStartDay CHECK (StartDay BETWEEN 1 and 366),
    CONSTRAINT CheckEndDay CHECK (EndDay BETWEEN 1 and 366),
    CONSTRAINT CheckStartBeforeEnd (startYear < endYear OR
                 (startYear = endYear AND startDay < endDay)),
);
```

b)
```
CREATE ASSERTION PayRangeLimit CHECK
   ( NOT EXISTS (
         SELECT A.empId
         FROM   Employees A, Employees B
         WHERE  A.salary - B.salary > 10000 ) )
```
or
```
CREATE ASSERTION PayRangeLimit CHECK
   ( ( (SELECT MAX(salary) FROM Employees)
         - (SELECT MIN(salary) FROM Employees) ) <= 10000);
```

c)
```
CREATE TRIGGER IncreaseEntitlement
BEFORE INSERT ON Employees
REFERENCING NEW ROW AS new
FOR EACH ROW
WHEN (new.year < 1970 AND new.entitlement < 35)
SET  new.entitlement = 35;
```

Question 4. 5 p

a) $\pi(Employees \bowtie_{empId=employee} (\sigma_{startYear=2007}(ParentalLeave)))$

b) $\tau_{branch}(\pi_{name,branch,salary}(\sigma_{salary=maxSal}($
$Employees \bowtie (\gamma_{branch,MAX(salary)\rightarrow maxSal}(Employees)))))$

Question 5. 7 p

a)
```
SELECT DISTINCT empId, name
FROM   Employees JOIN (
             SELECT employee
             FROM   ParentalLeave
             WHERE  startYear < endYear ) ON employee=empId;
```

b)
```
SELECT A.name, B.name
FROM   Employees A, Employees B
WHERE  A.empId <> B.empId
       AND A.year = B.year
       AND A.entitlement = B.entitlement
       AND A.salary < B.salary;
```

c)
```
CREATE VIEW V AS
SELECT   branch, startYear, COUNT(empId) AS numParents
FROM     Employees JOIN ParentalLeave on empId = employee
GROUP BY branch, startYear
```

Question 6. 3 p

a) Either T1 runs completely before T2, or vice versa. In either case, the output is: 2,2

b) In addition to the above, several other orderings are now possible.
T1B will see at least the same tuples as T1A.
T2A,T1A,T1B,T2B gives 1,1
T2A,T1A,T2B,T1B gives 1,2
T1A,T2A,T1B,T2B gives 2,2
T1A,T2A,T2B,T1B gives 2,3

Question 7. 6 p

a)
i) task 1: 2, task 2: 30, task 3: 30.
ii) task 1: 4, task 2: 30, task 3: 11.
iii) task 1: 4, task 2: 6, task 3: 30.
iv) task 1: 6, task 2: 6, task 3: 11.

b) (iv) has lowest cost of 6.5, with (iii) next with cost of 7.4.

Question 8. 8 p

a) (Alternative tags and structures are also acceptable.
Note: Branch numbers have been modified so that they start with a letter.
We'll accept solutions where these begin with digits.)

```
<?xml version="1.0" standalone="yes" ?>

<!DOCTYPE Question8 [

<!ELEMENT Question8 (Employees, Branches)>

<!ELEMENT Employees (Employee*)>
  <!ELEMENT Employee EMPTY>
    <!ATTLIST Employee
      name   ID    #REQUIRED
      branch IDREF #REQUIRED
      salary CDATA #REQUIRED >

<!ELEMENT Branches (Branch*)>
  <!ELEMENT Branch EMPTY>
    <!ATTLIST Branch
      number ID    #REQUIRED
      city   CDATA #REQUIRED >

]>
```

b)

```
<Question8>
 <Employees>
  <Employee name="Andersson" branch="b3" salary="35000" />
  <Employee name="Jonsson"   branch="b3" salary="25000" />
  <Employee name="Larsson"   branch="b2" salary="32000" />
 </Employees>
 <Branches>
  <Branch number="b1" city="Stockholm" />
  <Branch number="b2" city="Paris" />
  <Branch number="b3" city="London" />
 </Branches>
</Question8>
```

c) (Alternatives are acceptable. Two suggestions are given here.)

```
/Question8/Branches/Branch[@city="Paris"]

//Branch[@city="Paris"]
```

d) (Alternatives are acceptable.)

```
FOR $e IN /Question8/Employees/Employee[@salary>30000]
LET $place := $e/@branch => Branch/@city
RETURN <Result>{$e/name}: {$place}</Result>
```