Databases Exam

TDA357 (Chalmers), DIT620 (University of Gothenburg)

18 March 2016 at 8:30–12:30, HA, HB, HC, Hörsalsvägen

Department of Computer Science and Engineering

Examiner: Aarne Ranta tel. 031 772 10 82. Aarne will visit the exam rooms at around 9:30 and 11:30 and can be reached by phone at other times.

Results: Will be published 5 April at the latest.

Exam review: 13 April at 10-12 in Aarne's office (ED-huset, 6106)

Grades: Chalmers: 24 for 3, 36 for 4, 48 for 5. GU: 24 for G, 42 for VG.

Help material: One cheat sheet, which is an 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. One English language dictionary is also allowed.

Specific instructions: You can answer in English, Danish, Dutch, Finnish, French, German, Italian, Norwegian, Russian, Spanish, or Swedish (in this exam; next time it can be another set of languages ;-) Begin the answer to each question (numbers 1 to 6) on a new page. The a,b,c,... parts with the same number can be on the same page. If you need many pages for one question, number the pages as well, for instance, Question 3 p. 2. Write the question number on every 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. In particular: in SQL questions, use standard SQL or PostgreSQL. If you use any other variant (such as Oracle or MySQL), say this; but full points are not guaranteed since this may change the nature of the question.

1 E-R modelling (12p)

The domain to model is mountains and countries:

- A mountain has a summit position (the latitude and longitude coordinates), which identifies it uniquely.
- A mountain can have many names: e.g. *Mount Everest* is also known as *Chomolungma* and *Sagarmatha*.
- There can be many mountains with the same name (e.g. many *Stortinden* in Norway), but not with the same summit position.
- A mountain can lie in many countries: e.g. Mont Blanc is on the border of France and Italy, and hence lies in both countries.
- Every country has a highest mountain: e.g. Mont Blanc is the highest mountain of both France and Italy. (We assume that every country has exactly one highest mountain, even though there might be borderline cases with countries that have no mountains and countries with many equally high highest mountain).
- Every country has a different name.

a. Draw an Entity-Relationship diagram for this domain. **Do not use multivalued attributes**. (7p).

b. Show the corresponding database schema (4p).

c. Modify the schema so that it enforces the following constraint:

- The highest mountain of a country is a mountain that lies in that country.

It is enough to do this in the schema; it would be more difficult in the E-R diagram. (1p)

2 Functional dependencies and normal forms (8p)

Here is a part of a country fact box in Wikipedia:

- name: Sweden
- capital: Stockholm
- area: 450,295 sq km
- population: 9,816,431
- population density: 21.8/sq km
- GDP (Gross Domestic Product) total: $483,724,270,387
- GDP per capita (i.e. per person): $49,277
- currency: SEK
- time zone: CET
- time zone difference from UTC (Greenwich time): +1

We assume that all the attributes have a single value, even the time zone (which in the real world is not true of all countries). We also assume that all countries have different names and that all capitals have different names.

Now, consider the task of creating a database from country fact boxes. We start with a big table with the schema

```
Countries(name,capital,area,population,density,
          gdp,gdpCapita,currency,timeZone,timeZoneDiff)
```

a. What functional dependencies can you find? (You don't need to list dependencies that follow from other ones you list.) (3p)

b. What keys are suggested by the functional dependencies, and why? (1p)

c. Which functional dependencies violate the Boyce Codd Normal Form (BCNF), and why? (2p)

d. How would you decompose the table to bring it to BCNF?

- What are the new tables, their keys, and their functional dependencies?
- Which dependencies could/should be treated in another way than by new tables?

(2p)

3 SQL tables and queries (12p)

Consider the relation `Countries(name,population,area,timeZoneDiff)`, which is a part of the Wikipedia fact boxes shown in Question 2.

a. Write an SQL table definition with reasonable types and constraints (n.b. the time zone difference varies from -12 to +12). (4p)

b. Write an SQL query that returns the population density (i.e. population divided by area) for the countries east of Greenwich, i.e. with a strictly positive time zone difference. (4p)

c. Write an SQL query that lists country names with the attribute `hemisphere`, which has the value `western` if the country's time zone difference is 0 or negative, and `eastern` if it is strictly positive. (4p)

4 Relational algebra (8p)

a. Given the relation `Countries(name,population,area,timeZone)`, write a relational algebra query that returns, for each time zone, the sum of populations of countries in that time zone. The query should return tuples of the form `(timeZone,totalPopulation)`. (4p)

b. Consider the operators δ (distinct) and π (projection). Show an example in which the order of applying the operators (i.e. δ before π vs. π before δ) makes a difference in the resulting multisets of tuples. (4p)

5 Constraints and triggers (12p)

Constraints and triggers are techniques that can be used to control properties of databases. As a rule of thumb, constraints are the simplest solution, but not always possible. In the following cases, use either constraints (added to a table) or triggers (on a table), whichever you find to be the simplest solution.

a. Assume you want to create a table that should have at most one row. How can you guarantee this? Write an example `CREATE TABLE` statement, together with a trigger if you need one.

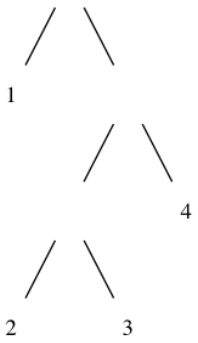
b. Assume you have a table `Teachers(name,phoneNumber)` that lists the teachers currently in charge for some class. Assume the table already has some rows. How can you guarantee that this table will never become empty i.e. never have 0 rows? Show the exact `CREATE TABLE` statement and/or trigger.

c. Consider a table with the schema `Distances(fromCity,toCity,distance)` that stores distances between cities. Since the distance from Y to X is always the same as the distance from X to Y, it would be redundant to store them both. How can you guarantee that the table never stores the distance from Y to X if it already has the distance from X to Y? Show the exact `CREATE TABLE` statement and/or trigger.

(4p for each of a,b,c. Less points for solutions that are more complicated than necessary.)

6 XML (8p)

A **binary tree** is a tree whose every node either branches to two binary trees or is a **leaf**, i.e. contains a value. Here is an example of a binary tree:



a. Design a DTD for representing binary trees and nothing but binary trees. The branching nodes should not carry any information, whereas every leaf should carry a value that can be any string (`#PCDATA`). (3p)

b. Show an XML element representing the above example tree, and which is valid according to your DTD. (3p)

c. Write an XPath query that returns all leaf elements of a binary tree. For the above example, it should return 1,2,3,4 (without any XML tags). (2p)