Introduction to Database Design
BSc and MSc Exams

Björn Thór Jónsson

2019-03-21

Instructions

This file contains the questions from the March 2019 re-take exam that remain relevant in the current version of the course. Some of today's learning outcomes are not represented in this exam.

Database description

In this exam you will work with the database `imdb`. To start working with the database, import/run `idb-march-2019.sql` found in LearnIT using the PostgreSQL DBMS on your laptop. The database contains information on movies and people. Most of the data is actually part of the IMDb real database, but some information is made up. The following is the relevant part of the database schema (the relations `person` and `movie` relations have some additional attributes that are not used in the exam).

`person`(id, name, ...)
`movie`(id, title, year, ...)
`genre`(genre, category)
`movie_genre`(movieId, genre)
`role`(role)
`involved`(movieId, personId, role)

All relations have a defined primary key (all underlined attributes) except `movie_genre`. In particular, the last relation has a composite primary key consisting of all three attributes. Foreign keys are defined where appropriate. The `role` relation is a lookup table with a list of all possible roles. The meaning of other relations and attributes should be self-explanatory.

1 SQL (40 points)

Answer each of the following questions using a single SQL query on the `imdb` database:

(a) The `person` relation contains 573 entries with a registered height greater than 190 centimetres. How many entries do not have a registered height?

(b) In the database, there are 365 movies where the average height of all people involved is less than 165 centimetres (ignoring people where the height is not registered). For how many movies is the average height of all people involved greater than 190 centimetres?

(c) The `movie_genre` relation does not have a primary key, which can lead to a movie having more than one entry with the same genre. How many movies in `movie_genre` have such duplicate entries?

(d) According to the information in the database, 476 different persons acted in movies directed by 'Francis Ford Coppola'. How many different persons acted in movies directed by 'Steven Spielberg'?

(e) Of all the movies produced in 2002, there are 12 that have no registered entry in `involved`. How many movies produced in 1999 have no registered entry in `involved`?

(f) In the database, the number of persons who have acted in exactly one movie that they self directed is 603. How many persons have acted in more than one movie that they self directed?

(g) Of all the movies produced in 2002, there are 282 that have entries registered in `involved` for *all* roles defined in the `roles` relation. How many movies produced in 1999 have entries registered in `involved` for all roles defined in the `roles` relation? *Note: This is a relational division query which must work for any schema; you can not use the fact that currently there are only 2 different roles.*

(h) The number of persons who have had some role in some movie from *all* genres from the category 'Newsworthy' is 156. How many persons have had some role in some movie from all genres from the category 'Lame'?

Enter each query, as well as the numerical output of each query, in LearnIT. The query must work for any instance of the schema. The query should not return anything except the numerical answer; a query that returns more information will not receive full points, even if the correct answer is part of the returned result. In particular, a sequence of several queries that allow you to answer the question will not receive full points (but subqueries are fine). If you are unable to write a working query you can still submit your attempt, and it may be given partial points; in this case include a brief description of the problem with the query.

3 ER Diagrams and DDL (25 points)

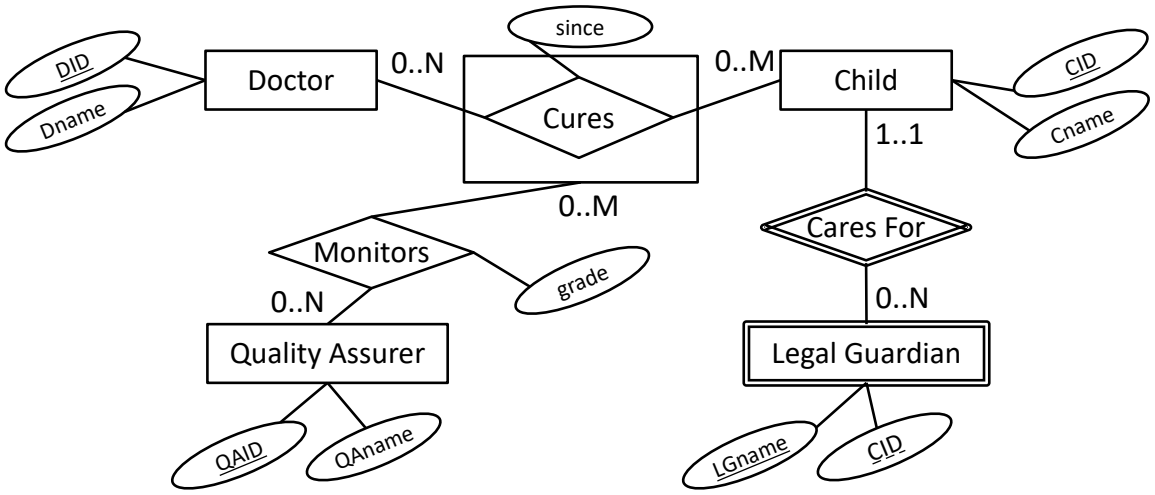

Figure 1: ER Diagram for medial database.

a) Write MySQL DDL to create a medical database based on the ER diagram in Figure 1. The relations must include all key and foreign key constraints. Make reasonable assumptions on the domains of attributes in the ER diagram. If some aspects of the ER diagram are not captured by the DDL commands, that should be noted.

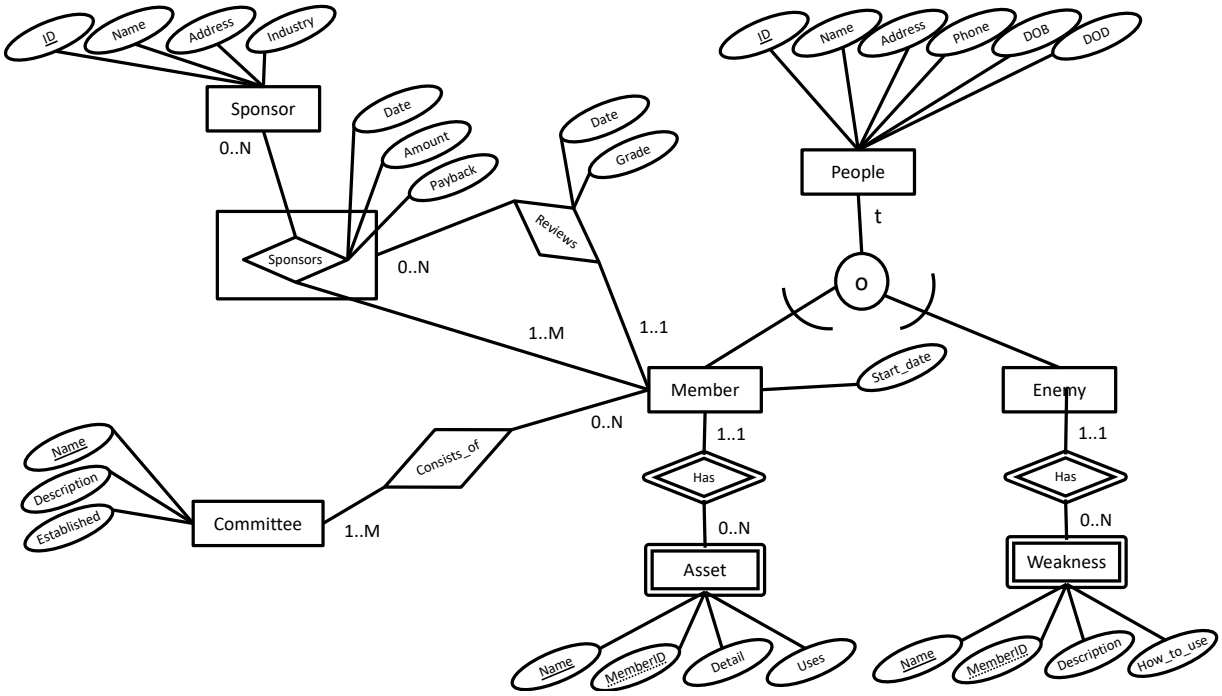

Figure 2: ER Diagram for a database for a political party.

b) Figure 2 shows an ER diagram for a database designed for a political party. Considering ONLY the design presented in the ER diagram, select the true statements:

(a) All people are either a member or an enemy.

(b) The total sponsorship amount for each member can be computed.

(c) A sponsor can give many sponsorships to the same member.

(d) A sponsorship can be reviewed by many members.

(e) A sponsor can choose to give no sponsorships.

(f) No two children can have the same name.

(g) An enemy can have at most a single weakness.

(h) Two enemies can have a weakness with the same name, but with a different description.

(i) All sponsorships given by a single sponsors must have the same amount.

(j) A member can review many sponsorships.

(k) A member can only be part of one committee.

(l) A sponsor can be linked with at least one committee via the members the sponsor.

c) Write an ER diagram for a for a database of a wholesale dealer. The diagram should show the relations and constrains described below:

- The main entities are Salesmen, Products, Customers, and Auditors. Assume that Salesmen have SID as key. Otherwise, their attributes are not important here.
- Each salesman has taken zero, one, or more courses. Each course has a name and hours. For each salesman, the name of the course is unique.
- Salesmen can supervise products. Each product has exactly one supervisor.
- Salesmen can sell all products to all customers.
- Each salesman can sell the same product to the same customer multiple times, but only once per day. Aside from the date of the sale, the quantity and total price must be registered.
- Sales are sometimes reviewed by auditors. Each sale can at most be reviewed by one auditor.

If you need to make additional assumptions put them in the box below.

4 Normalisation (10 points)

a) Consider a table $R(A, B, C, D, E)$ with the following dependencies:

$$\begin{aligned} A &\rightarrow C \\ C &\rightarrow B \\ D &\rightarrow E \\ A &\rightarrow BCDE \end{aligned}$$

Select the true statements:

(a) A is the only key of R.

(b) $BCD \rightarrow D$ is a trivial functional dependency.

(c) Normalizing to BCNF results in exactly two relations.

(d) The relation $Z(A, B, C, D)$ is in BCNF.

b) Consider a table $R(A, B, C, D, E)$ with the following dependencies:

$$\begin{aligned} AB &\rightarrow C \\ A &\rightarrow E \\ A &\rightarrow D \\ C &\rightarrow A \end{aligned}$$

Select the true statements:

(a) AB is the only key of R.

(b) $AB \rightarrow C$ is an unavoidable functional dependency.

(c) Normalizing to 3NF (but not to BCNF) results in exactly two relations.

(d) The relation $Z(A, B, E, D)$ is in BCNF.

c) Consider a table $R(A, B, C, D, E)$ with the following dependencies:

$$\begin{aligned} AB &\rightarrow C \\ DE &\rightarrow E \\ C &\rightarrow DE \\ A &\rightarrow C \end{aligned}$$

Normalize R to BCNF and write down the resulting relations here:

6 Index Selection (10 points)

Consider the following two relations with information on professors and departments:

`Prof` (id, name, dept_id, c, ...)
`Dept` (id, code, mgr_id, ...)

Assume that all attributes are required (`not null`) and that salary is at least 10000. For each of the queries below, select the index(es) that a good query optimiser is likely to use to process the query. You may select 1 or more options (including "no index"). Note that we only consider indexes for the `Prof` relation; you may assume that no useful index exists for the `Dept` relation. Also assume that all indexes are unclustered.

(a) Prof(id)

(b) Prof(dept_id)

(c) Prof(salary)

(d) Prof(dept_id, salary)

(e) Prof(dept_id, salary, id)

(f) No index

The queries are:

Query 1

```
select P.dept_id, P.name
from Prof P;
```

Query 2

```
select P.id, P.name
from Prof P
    join Dept D on P.id = D.mgr_id;
```

Query 3

```
select avg(P.salary)
from prof P
where dept_id = 23;
```

Query 4

```
select P.name
from Prof P
where P.salary > 0;
```