Basics of database systems

**Project – Database design**

Lappeenranta-Lahti University of Technology LUT

Software Engineering

Basics of database systems

Spring 2022

Table of contents

[Table of contents 1](#_Toc92697309)

[1 Definition 2](#_Toc92697310)

[2 modeling 4](#_Toc92697311)

[2.1 Concept model 4](#_Toc92697312)

[2.2 Relational model 5](#_Toc92697313)

[3 database implementation 6](#_Toc92697314)

[4 discussion 7](#_Toc92697315)

# Definition

Describe the need of your work, for what the database is developed. Define the userbase. Describe the problem area in such detail that it can be modelled into a database and it can be used to critically compare the concept analysis during evaluation. Note that the database will be used through another application, not with raw SQL queries. You can define needs for the user interface and software functionalities.

**Example: N-division**

In project ‘N-division,’ database is developed for a customer who manages the amateur football league. Because the user is a small association/club and activity is small-scale, the database should be as small and lightweight as possible. The database should allow the administration of teams and their players participating in the league. Important player information are name, player number, and contact information. The database also keeps track of the points players have gained during the league (score, assist). In addition to players, teams have a coach and a manager, and their information needs to be readily available. Teams will play against each other, and the results should be put to the database. Each match have voluntary referees whose contact information needs to be stored in the database as well as who refereed what match. Finally, the location of matches (address, phone number of the person in charge, when matches can be arranged, etc.) needs to be stored in the database.

Possible database users: Administrator, coaches, secretary of the association. Administrator can alter all the tables, queries, views and create new data and structures. Coaches and team representatives can read all information and some special queries made for them. They cannot add, delete or change any data. Secretary can edit the following (list...) and read the following data (list...). To help the secretary to do their work, some queries or views are required and they are described below.

The following database queries have to be implemented: (1) List the information of a specific player, their points and warnings. (2) List all matches where person X has been the referee. (3) List the match results of team X. (4) Show the contact information of the coaches that were on field X on day Y during a match. (5) Check if field X is reserved on a specific day for a match and if so, return the contact information of the managers.

**The queries do not have to be complicated but there has to be a query that requires two JOIN clauses in a SELECT query (M:N -relation).**

# modeling

## Concept model

Database design begins with concept analysis. The target of the analysis is the whole activity domain or a subpart of the domain. The result of the concept analysis is the logical model (conceptual model) of the domain. The conceptual model is represented graphically and filled with data descriptions. The target domain description is simplified for the database. The conceptual model includes integrity constraints.

The use of concept analysis leads to database design decisions that are independent of the data and implementation.

At this point of the project, the aim is to describe the conceptual model of the database that has been developed with the help of concept analysis. Use the ER-modeling notation.

Represent at least: Entities (concepts), relationships (the connection between concepts) and the cardinalities of the relationships (one-to-one 1:1, one-to-many 1:N, many-to-many M:N), and properties (attributes).

**Example:** Check the ER-model lectures and material.

## Relational model

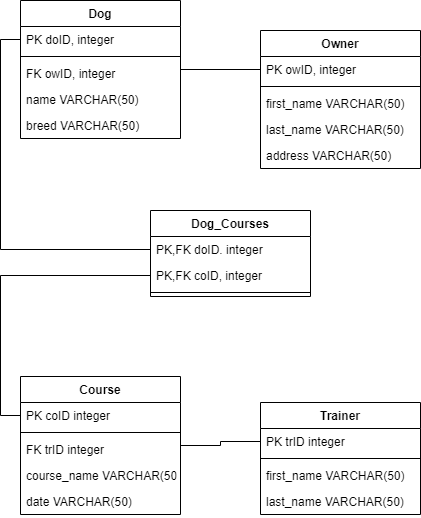
After the ER-modelling, transform it to a relational form using the appropriate transformation rules. Represent the result either as a relational diagram or a freeform UML-style table structure. Describe the used transformation steps and rules for transformation.

Each entity should have the necessary primary key, alternate/secondary keys (other candidate keys), and foreign keys. Primary keys and secondary keys have their own integrity constraints that are not automatically enforced when following the relational model.

The definition of accepted value range should be done for each attribute: length, type, uniqueness, NULL values, and default values. The definition needs to be done for keys. Primary key cannot be NULL but secondary keys can be NULL.

Relationships have their own integrity constraints when necessary. If a relationship has attributes, it is described as an entity.

**Example:** Check the transformation rules from the lecture material in topic 3, Relational model. Here is an example of a freeform UML style table structure (picture below).



# database implementation

Report here what kind of changes, compromises, surprises, etc., the transformation to relational model caused. Describe how you did the SQL queries with Python.

**Example:**

**In general:**

During the transformation of concept model to relational model, it was found out that address and phone number attributes repeating in multiple tables could have their own data type, that would standardize the information across the database, as they are susceptible to errors. I also noticed that it helps to have a constant “zero” team where a new player without a team could be added by default. For this, a new integrity constraint was developed. Third issue that rose was the need to re-evaluate the relationship of Team, Coach and Manager. Coach and Manager are not subtypes of Organization because the same person can be a coach and a manager for the same team. As a result, they were given separate tables that was originally removed from the relational model. No changes were made to views and queries during implementation.

**Implementation:**

I wrote a simple Python program that showed the example queries, asked the user to update data, wrote one new row to the database and demonstrated reading by showing the changes.

**Note during testing:** Nothing special to note.

# discussion

If you want to mention something that has not been discussed in the previous chapters, you can discuss them here.