CS-A1150 Databases, Summer 2020
Project, part 2 (SQL)

1. Introduction

The purpose of the project is to learn to design databases in practice. The project consists of two parts. The project is designed for groups of 2-3 students. Both project rounds must be done with same group.

2. Instructions

This part will be made on the basis of the first part. In this part, you create and use a SQL database based on the plans of the first part. You test your database in SQLiteStudio environment. Include the document of your first part in the submission of this part. If you find mistakes in the first part, you must correct them at first (however, the grading of the first part will not be changed because of the corrections). In that case, add a chapter "Changes made after the submission of the first part" in your document after the document of the first part. You have to correct only the UML diagram and relation schemas, not other parts of Project, part 1.

Tasks

- Define the relation schema in SQL. The schema must contain the same information as the project part 1. Write the CREATE statements to the document. Use data types that are reasonable. Justify your solutions. In addition, insert enough initial data into your tables such that you can test the SQL queries described below.
- Check the keys and other constraints in your database. Verify that the primary keys and foreign keys are well defined.
- Which kind of SQL queries might be typical for the database? Create reasonable indexing which supports the purposeful use of the database. In addition, add at least one useful view definition to your database.
- Create some typical use cases for the database. Write a description of the use case, explain which information needs to be selected/updated/inserted and write the SQL queries and possible other commands that are needed to fulfill the use case. The number of SQL queries (SQL statements which begin with the word SELECT and end with semicolon) written in the document must be at least 15. (One use case may contain several queries.) Try to find at least one use case where you need to use a little bit more complicated SQL query, for example including join of several tables, grouping and calculating a value of an aggerate function. In addition to queries, many of your use cases probably need other SQL commands, like inserts or deletes.
- Run the SQL statements (creating the tables, inserting example data in them and the SQL queries) in SQLiteStudio environment. See Exercise Round 2 in A+ System for instructions how to use SQLiteStudio. Attach the .sql-file to your submission. Insert the listing of the SQL commands you run and their outputs in your document (another file).

Justify all your solutions carefully!

Note that **you are not asked** to write a user interface for the database. You can create the database and perform all your queries by just writing the SQL statements.

You are not asked to write triggers. You can write several SQL statements which are all supposed to be performed for one use case. (The user interface program would take care of it, but you are not asked to write it.)

3. Principles of Grading

All members of the group will get the same grade of the project. The project consists of two parts. The parts are worth 20 points each.

- To pass the project, you must get at least 20 points in total and submit solutions to both parts.
- **Writing SQL statements is compalsory for the second round. Your solution is not considered as submitted, if it does not contain at least some SQL statements.**
- If you get at least 30 points, your course grade will be improved by one full grade (however, you must pass the exam!)

The project grade is valid exams arranged in the academic year 2020-2021.

You can get up to 20 points for this part:

- Converting the database to SQL and constraints (primary and foreign keys, etc) 5 p. (If your model is too constricted or contains errors, you may lose points even if the conversion is done correctly.)
- Indexing (including justifications) and views 3 p.
- Use cases and SQL-queries (If you do not have enough SQL-queries starting with word SELECT and ending with semicolon, your points are decreased. The complexity of the queries is also taken into account. If almost all queries are just simple queries involving only one table, you lose points. At least 8 queries must involve more than one table, and one one of them must be a complicated query as described above.) 8 p.
- The quality of documentation 4 p.

4. Submitting the Project

The deadline of the second part is August 28th 2020 at 20:00. Submit a zip-file containing the two files described below in A+-System. If your submission is late at most 7 days, your submission will be graded, but you will lose 5 points, if the delay is 3-7 days and 3 points, if the delay is under 3 days. If the delay is over 7 days, your project will not be graded except if you have an exceptional reason (for example, a long sick leave signed by a doctor).

Your submission must be a zip-file containing two separate files:

1. Text file (.sql file) which includes all SQL commands which are used to create the tables, insert the initial data into them, and the SQL queries and other commands used in the use cases. **The teaching assistant must be able to create and test the database in SQLiteStudio by just running the commands in the file as they are given.** If you want, you may split the .sql file into two separate files, one of them containing the statements which create the database and the other the actual queries. The SQL statements used to insert the initial data to the tables may be in either of the files.
2. The report as a PDF file. The structure of the report is explained below. You may write in English, Finnish or Swedish.

The report must consist of the following parts:

- Cover page, which includes the names and the e-mail addresses of all members of the group.
- Solution of the first part.
- Modifications and supplements to the solution of the first part (if there are any).
- Solution for the tasks given above.
- Justification of the solutions.
- Explanations of the use cases and SQL statements run in SQLiteStudio environment (for example, the purpose of the various queries) and listings of the outputs of the statements. If some of the SQL statements causes purposely an error (for example, insertion does not succeed because it violates some consistency conditions), explain it in your documentation. Note that the SQL commands are listed both in the document and in the .sql-file.