Assignment 3 – Relational database, views, user-defined functions, and stored procedures

This assignment IS graded
Deadline: 2020-11-22

Problem statement

Create a relational database for the Relational Model for which the assignment 2 was created. Add some test data into it and make 2 queries that fetch information from 2 or more joined tables (if there are customers involved, join in the customer table, and return the customer's name, NOT his/her ID). As the data models were very different, use your own common sense for designing these queries, but here are some examples:

- Mail order database
    - Query all orders in a specified time frame, returning order number, customer name, number of ordered items and their total sum, as well as expected shipping date and actual shipping date (when the order has been shipped, or NULL when it has not)
    - Select a single order and list both order header info, as well as all ordered items and their ordered quantities
- Movie database
    - List all movies where that have come out within a specified year range. Return movie title, movie's release year, number of actors
    - List all quotes for selected actor, listing the quote, the movie in which it was said, a year when the movie came out
- Conference review database
    - List all papers submitted in specified date range. Return the name of the paper's author, title and abstract of the paper, number of reviewers who have reviewed it, average grade across all 4 technical merits
    - List all reviews for a specified paper by all reviewers and in all 4 categories (technical merits). Return name of the paper, name of its author, name of the reviewer, all 4 merits as separate columns. There will be as many rows of data, as there are reviewers. If desired, a MARS (Multiple Active Results Sets) style query can be made that divides the problem into 2 sub-queries – one for paper header, and the other for each individual reviewer.

Query descriptions

Once the database is created, filled with data, and the queries are done, add a view, user-defined function, and a stored procedure into the mix.

NOTE: Please keep in mind that the queries created above must also be included, even though they may be incorporated into the view, function, or a stored procedure!

- View shall return data for the first query
- The function should calculate the number of items the first query may require
- The stored procedure should contain the second query