INF-2700 DATABASE SYSTEMS

DATABASE PROGRAMMING

ALEXANDER EINSHØJ

21.09.16

PART I

Task 1:

The schema describes different aspects of a company which sells what seems like models of different sorts of vehicles (given the product scale column). The schema describes information about the products they sell, payments made, their customers, their customers' orders, the details of those orders, their offices and their employees.

Task 2:

a) This query extracts three columns from the table Customers.

b) This query retrieves all the entries in the table 'Orders' in the case that the 'shippedDate' column is NULL

c) This query retrieves two columns from two different tables. As the select statement essentially makes a new result table, it requires the tables chosen from to be connected. Orderdetails and Customers have no columns in common, therefore Orders need to be joined as well, because it has common columns with both tables. The result is grouped which means instead of listing each customer number entry multiple times, they're gathered as one entry. The ordering orders the resulting table in descending order given a column.

d) This query retrieves two columns from two tables which do have a common column. Since this is the case, they can be natural joined on that column. After the natural join statement, a nested statement is made, which produces its own table before joining with the other table. The requirement for an entry is stated in the WHERE clause.

PART II

Introduction

This assignment is split into two parts. In the first part, multiple SQL query problems are solved.

In the second part, the implementation of an application which runs a database "shell" is described. The resulting records are also to be formatted into JSON-format.

Technical Background

In terms of the SQL part of this assignment, the most important subject to know about is the SQL syntax. When retrieving information from multiple tables, a certain knowledge about the term ‘join’ is needed.

Natural Join
This join will merge two tables on an identically named column, and that column will only appear once in the resulting table. A natural join is very much like an inner join.

A natural join will implicitly join on all columns with identical names in both tables (given there are only two), whereas with an inner join, it has to be specified in the query what columns to be joined on. There are cases where a natural join can’t replace the inner join. An example of this is if one wants to join on columns with different names, or if the join relies on a difference in two columns.

Design

The sqlite library offers a couple solutions for doing queries against a database. In either case, the database will have to be initialized/opened before using it, and should also be closed when it’s not going to be used any longer.

As for executing the queries, a wrapper function is currently implemented. This function wraps around three other functions, so the amount of code needed shortens.

A callback function is needed to format each record as wished, before outputting.

The shell will read all input from the stream (stdin) until a terminating character is read, “;”. At this point the application delivers the SQL statement to the executing function, which decides whether the statement is valid or not.

Implementation

Part 2 is implemented in C.

Discussion

The current implementation does not suffice in terms of showing results in JSON-format. This is due to the wrapper function being used. The formatted output is fully dependent on the callback function, of which the application has no control to when it begins/end. The JSON-format requires a starting and closing bracket before and after the entire result statement is printed. Each entry should also be enclosed by braces, and each key-value pair inside the entry should be separated by a comma. There are also different rules on how to print strings and integers. The separation between each entry and whether a value is to be considered an int or not, is difficult using the callback function as done in this implementation. By not using the wrapper function for executing the SQL query, distinguishing every entry automatically becomes a simpler task, as you gain more control of the resulting statement.