Prof. Dr. Stefan Brass
Institut für Informatik
MLU Halle-Wittenberg

November 7, 2019

Databases IIB: DBMS-Implementation — Exercise Sheet 3 —

Please read Part a) and b) and mark questions which you want to discuss in class. You only have to submit Part i) to l). Please upload your solution (a pure ASCII text file with the four SQL statements) into the StudIP file folder called "Hausaufgabe_3" in the StudIP entry of the lecture (with course ID `06fb9210013971fbdbb78ff1746b744e`). The deadline is November 12 (the day before the next lecture).

It is permitted to form groups of up two members, but please make sure that both members can fully explain all homeworks submitted by the group. The intention is certainly not that each member solves only half of the exercises. You must do the insertion required in Part l) for each group member separately. Please upload only one file per group. Name the file such that it contains the names of all group members.

Not all submitted homeworks will be corrected, but all homework exercises will be discussed in class. If you should have questions about your homework, please ask! A precondition for getting credit for this course is that you submit solutions to two thirds of the homeworks. Obviously wrong or very incomplete submissions do not count.

Some Feedback on Homework 1

- Please make sure that the program can be compiled on Linux as well as on Windows. E.g. include files like `stdafx.h`, `conio.h` and `pch.h` do not exist under Linux.
- If one executes a console program under Windows not inside a shell, a console is created for the program, but immediately closed when the program finishes. So one has to wait for the user pressing "Return" (only under Windows). My program contains the following code which reads characters until a line end:

```
#ifdef VER_PRESS_RETURN
        cout << "\n(Please press Return/Enter to finish)\n";
        for(char c = ' '; !cin.fail() && !cin.eof() && c!='\n'; )
                cin.get(c);
#endif
```

  If one wants this version, the symbol `VER_PRESS_RETURN` must be defined (e.g. at the top, or in an include file `ver.h`):

```
#define VER_PRESS_RETURN
```

  One can also do this on the command line with the option `-DVER_PRESS_RETURN` when calling the `g++` compiler. If one wants code that is automatically chosen on Windows, one can try


```
#if defined(_MSC_VER) || defined(_WINDOWS_) || defined(WIN32)
```

  Some check also for `_WIN32`.

  Note that `system("pause");` is *not* a good solution: It creates a new process with a shell (command prompt) to execute this command, which is only available under Windows, and the shell might even find a different command than you expected! The function `system` should be used if great care (or better not at all) if you want to write safe programs.

- Please avoid lines longer than 80 characters. In my editor, the tabulator size is 8 characters (classical UNIX standard).
- Note that it is ok if a loop body is executed 0 times. Some students had a special condition for the input 2 to the prime number test. This is not necessary.
- An advantage of `for` loops over `while` loops is that intialization, loop condition, and the update of the loop variable are all together in one place. In my view, a loop with a single loop variable that has all of these parts should be written as a `for`-loop, not as a `while`-loop. (Personally, I use `for`-loops also when some of the control parts are missing.) If the loop variable is not used outside the loop, you can declare it in the `for`. This reduction of the scope of the variable is not possible with `while`.
- Your code looks more professionally if you avoid comparisons with boolean constants, e.g. if `isPrim` is a variable declared as `bool`, instead of `if(isPrim == true)` simply write `if(isPrim)`.
- Note that if a function/method is declared as returning `int`, then every possible execution path should end a `return`-statement specifying a value. Otherwise, there is no error message (it is formally legal in C++), but the returned value is undefined (not initialized) otherwise. Switch on warnings (`-Wall`) to be informed in this case.

  However, there is one exception for the function `main`: It may terminate without a `return` and will return `0` in this case.

- Avoid duplicating code! E.g. do not write one loop that checks for the prime number condition and then another loop that prints the divisors. As a master student, you should be well beyond "Copy&Paste" programming.

Repetition Questions

a) What would you answer to the following questions in an oral exam?

- Sketch data dictionary tables for the database schema, i.e. tables, columns, and constraints (keys, foreign keys). If you know the basic structure of the Oracle tables, you could use that (of course, you should concentrate on the most important columns). But you could also design your own tables (the concepts are important, you should not learn the Oracle tables by heart).

- What do the Oracle data types `NUMBER` and `VARCHAR2` mean? These are not part of the SQL standard.
- Table and column names are not case-sensitive in most SQL systems (sometimes this depends on DBMS options or settings specified at database creation time). Explain the situation when accessing the Oracle data dictionary.
- Why is it useful to specify constraint names in the `CREATE TABLE` statement?
- Is it possible to store a short explanation about tables inside an Oracle database? If you use a system that has no special command for this, what could you do?
- Name some features that make constraints in real systems like Oracle more complicated than the simple view of the first database course.
- Explain "Object Privileges" (the standard SQL access rights). Give an example for a `GRANT` command in SQL. Explain the clause `WITH GRANT OPTION`. What should be the effect of the `REVOKE` command?
- How could a data dictionary table for managing object privileges look like?
- What problem is solved with "system privileges" in Oracle? What would be an alternative solution, used in other systems?
- What is the purpose of "roles" in Oracle? How can they be used?

b) Check your knowledge of C++ by answering these questions:

- You use a static variable in the class ("class attribute") to count the number of objects of the class. This variable is incremented in the constructor and decremented in the destructor. After running some tests, you find that the number has become negative. What happened? How can this problem be avoided?
- Suppose you define a class without constructor. What does the constructor do that the C++ compiler generates in this case? Which attributes will be initialized?
- Suppose you declare a class `C` in file `c.h` that has an attribute of class `D` (declared in `d.h`). How can you simplify the usage of your class, by making sure that the user does not have include `d.h`? Would the situation change if the attribute is only a pointer to class `D`?
- Suppose you wrote `d.h` yourself and want to make sure that there is no problem if `d.h` is included two times. How is this done?
- Suppose that C is a class. What is the difference between the following functions?
  - `void f(C x)`
  - `void f(C* x)`
  - `void f(const C* x)`
  - `void f(C& x)`
  - `void f(const C& x)`

  How would one call these functions for an object declared as "`C obj;`"?

In-Class Exercises

c) Please download the following file which creates the Oracle example tables:

[http://www.informatik.uni-halle.de/˜brass/dbi19/sql/empdept.sql]

If you use SQL*Plus, you can execute the script with `@empdept`. If you use Adminer, use "Importieren" (Import) in the menu on the left side. You can also try the SQL Developer, create a database connection and open an SQL Worksheet.

d) Now print name and type of all database objects you own (not only tables, but also indexes).

e) Create a small test table with one column and one row. Grant read access to this test table to your neighbour in class. Ask him/her to use an SQL query to access the data item in your table. Conversely, access his/her table.

Note that if you do not have read access to the table, Oracle will tell you that the table does not exist (even if it actually does exist). In this way, no information about the existence of tables can be gained from the error message.

f) Which roles were granted to you?

g) Which system privileges does the role `CONNECT` contain? Note that the corresponding data dictionary view shows only the information about roles you have, even if you are logged in as DBA (e.g., `SYSTEM`).

h) Print a list of all system privileges that you have, either directly granted or via a role. You can ignore role-to-role grants.

Homework Exercise 3

i) Schreiben Sie eine Anfrage, die Ihnen alle Tabellen listet, an denen Ihnen direkt Zugriffsrechte gegeben wurden (also nicht über eine Rolle). Drucken Sie den Besitzer der Tabelle (`OWNER`), den Tabellen-Namen (`TABKE_NAME`) und das Recht (`PRIVILEGE`).

j) Unter diesen Tabellen müsste auch eine Tabelle des Nutzers `BRASS` sein, für die Sie sogar Einfüge-Rechte haben. Schreiben Sie eine SQL-Anfrage, die Ihnen die Spalten dieser Tabelle (`COLUMN_NAME`, `COLUMN_ID`) mit Datentyp (`DATA_TYPE`, `DATA_LENGTH`, `DATA_PRECISION`) anzeigt. Sortieren Sie die Spalten in der deklarierten Reihenfolge (also nach der `COLUMN_ID`).

k) Sie sollen eine Zeile in diese Tabelle einfügen, bei denen die erste Spalte Ihren Nutzernamen enthält (die Pseudospalte `USER` ist immer der Name des aktuellen Nutzers). Damit die Sache nicht so einfach ist, sind auf der Tabelle Integritätsbedingungen formuliert. Schreiben Sie eine Anfrage, die die Integritätsbedingungen anzeigt.

l) Anschließend suchen Sie passende Werte aus, die Sie einfügen können, und fügen die Zeile ein als Beleg Ihres Erfolgs.