ACTA DIDACTICA NAPOCENSIA
Volume 8, Number 3, 2015

SOFTWARE APPLICATION FOR SUPPORTING THE EDUCATION OF DATABASE SYSTEMS

Anikó Vágner

Abstract: The article introduces an application which supports the education of database systems, particularly the teaching of SQL and PL/SQL in Oracle Database Management System environment. The application has two parts, one is the database schema and its content, and the other is a C# application. The schema is to administrate and store the tasks and the solutions, and it has a stored procedure which performs a syntactic verification of the solutions. The C# program helps the teacher organize the tasks and give feedback to the students. The main goal of the software application was to support the active learning method for database systems. With its help the students can practise alone, and they can get feedback from the teacher on their solutions.

Key words: education of database systems, database, software-aided education, education of programming

1. Introduction

In the Software Information Technology BSc program at the University of Debrecen in Hungary, SQL and PL/SQL are basic level curriculums. It is very important that the students practise SQL and PL/SQL not only with the teachers but also alone. The teacher can help them practise by giving tasks and feedback about their solutions.

For the teacher it is a hard work to examine many tasks and their solutions, which they get in the form of simple text. In this case they cannot be sure that the students execute their solutions. It is a good idea to use a software system which helps the teacher examine the solutions with automatic syntax verification. The Oracle DBMS is a very good tool to administrate the students' tasks and solutions and automatically verify the syntax of them.

In this article I introduce a software product that supports the education of SQL and PL/SQL in Oracle environment. With this software system the teacher can apply the active learning method in their lessons.

2. Literature Review

Classic didactic writers like (Skinner, 1968) and (Polya, 1957) say that the learning process of students is more efficient if they learn in an active way instead of a passive one. The students should solve tasks and problems and acquire experience instead of observing the tasks and their solutions of the teacher. The tasks are to guide and help the students in the learning process. The "learning by doing" concept also appears in new didactic books such as (DuFour et al, 2010) and (Schank et al, 1999). The latter gives an experimental education framework to the teacher and shows how teachers can organize work in the school and in the classroom.

The "learning by doing" concept also works in the education of computer science. (Gogoulou et al, 2009) used a software application for exploratory and collaborative learning in the education of programming. (Moore et al, 2002) describes a relational database management system course at Texas A&M University Corpus Christi that uses experimental learning. They received a very good feedback from the participating students. (Mason, 2013) also presents experimental learning for teaching

Received January 2015.

database administration and software development at Regis University. His students indicated that the course was a successful experience that helps them fine-tune their technical skills and develop new soft skills.

In the area of database systems (Ramakrishna, 2000) describes an experimental education survey in undergraduate education. His results show that his students prefer the experimental learning over the traditional tutorials.

There are applications which help the teacher verify the solutions of the tasks like the evaluating tool of (Kósa et al, 2005). Their application can examine C, Pascal, C++, and Java programs in such a way that the application executes the solutions for a lot of test cases, and if the result is the same as the predetermined output, the solution is correct. They don't check the program code itself, only the result. I cannot use this method, because an SQL or PL/SQL code results in changes in the database.

3. The software system

The first goal of the software system is to administrate the tasks given to the students and the solutions made by them. The second is to check whether the syntax of an uploaded solution is correct or not.

The administration helps both the teacher and the students to follow to performance of the students. The second goal helps the teacher to give feedback. It's not easy to find syntax errors in the program code without execution. The system throws an exception when a solution with incorrect syntax is uploaded. The teacher can set deadlines for the tasks with the application, which can also enforce these deadlines.

3.1. The database objects of the software system

The students have to use a database management system because they learn database management. So, it is obvious that the administration of the tasks and their solutions is done in the same database. The students find the tasks in the database which they can use to practise, upload their solutions and get the feedback for their solutions from the same database.

Additionally, storing the tasks and the solutions in a database has another advantage: the students are forced to use the database. This is the main goal of the courses. The students write a SELECT statement when they search for the new tasks and their deadline in the database and then execute a stored procedure when they upload a solution. The procedure sometimes throws an exception, which the student has to understand and correct the mistake. So, the students practise how to use the database.

To administrate the tasks, its deadlines, and the solutions of the students, schema objects are created in the database.

3.1.1. Schemas

Every student gets their own database account. A schema belongs to each database account. In this schema a student can do everything that he or she wants. (Of course the database administrator gives privileges to students, but not all the privileges. So they can do everything for what they have privilege.) In this schema the students can explore how they can use the database tools, statements, etc. The students can use their own schemas not only at the university but also at home.

The tables which have to be used by the students when they solve the tasks are stored in one or more schemas. The tables are ready for selecting since they were previously populated. The students execute their solutions (as program codes) on these tables. The students have only read privilege on these schemas. If they want to update or delete the contents of the tables in these schemas as a part of the task, they have to make a copy of the required tables in their own schema.

In the database there is a schema named ADBMS for the description of the tasks to be solved and their solutions. When the structure of objects in this schema was designed, it had to be taken into consideration that the students attend the course in more than one section in a semester. Basically the schedules of the sections can be the same, but in one section there can be more lessons than in the other in each semester because of the public holidays. Thus the structure of the database tables in the

ADBMS schema have to give the opportunity to create groups of students for each section in a semester and to assign the tasks separately for each group. It is worth dividing the tasks into groups. This way the tasks given for each lesson can easily be queried from the database. The students in each section get the same deadline for uploading the accepted solution for each task in a group.

Figure 1 shows the tables of the ADBMS schema and the relationships among them.

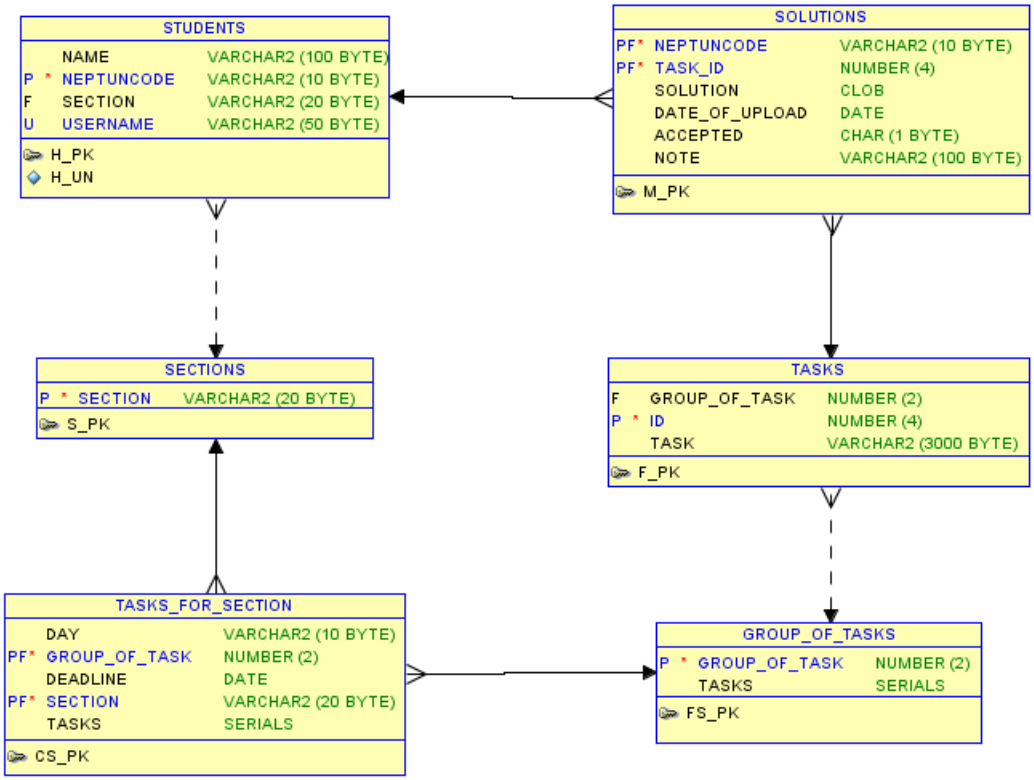

Figure 1. *Tables of the ADBMS schema*

Each student can only read data from the ADBMS schema which belongs to them. This means that each student gets privilege for selecting two views in the schema: MYTASKS and MYSOLUTIONS. The result of selecting the MYTASKS view gives the tasks (the number and the description of the task and the deadline) which the student has to solve. This is a subset of the rows in the TASKS table. The students can check their uploaded solutions in the MYSOLUTIONS view. The view contains not only the accepted solutions but all the solutions they uploaded. The students can check in this view whether their solution of each task is accepted, has to be corrected, or is waiting for examination. In the view there is a comment column. The teacher can write comments into this column about the reason why they did not accept the solution. Based on this comment, the student has to correct it.

Students upload their solutions into the database by executing the UPLOAD_SOLUTION procedure of the ADBMS schema. The students have got execution privilege for this procedure. The procedure has two parameters: the first is the number of the task which the solution belongs to; the second is the solution as source code. The UPLOAD_SOLUTION procedure checks the deadline and throws an exception if the deadline of the task has expired. Of course the teacher can modify the deadline for the task.

3.1.2. Syntactic verification of the solutions

The UPLOAD_SOLUTION procedure performs the syntactic verification. The students often make a mistake in the program code and may be too lazy to execute their solutions. Without syntactic verification it can happen that a solution is accepted by the teacher, but it contains syntax errors.

The database management system can perform the syntactic verification automatically. The solution which the student wants to upload is executed by the UPLOAD_SOLUTION procedure in the schema of the student. The procedure throws an exception if the program code cannot be executed, and in that case the solution will not be uploaded. The students are obliged to correct the syntax errors in their work.

The Oracle PL/SQL has a very good tool named native dynamic SQL, which I used to realize syntactic verification. The native dynamic SQL can execute one SQL statement without semicolon at the end or one PL/SQL unit.

In a solution the students can upload more than one statement or PL/SQL unit. It doesn't matter whether each of them is SQL or PL/SQL code, however at the end of the SQL code the students cannot use semicolon. Every statement or PL/SQL code has to be closed with a "/" sign. The UPLOAD_SOLUTION procedure splits the uploaded solutions into SQL statements and PL/SQL units based on the "/" sign. Then it executes the parts of the solutions one after the other in the schema of the student who executed the UPLOAD_SOLUTION procedure. If one of them throws an exception, the others will not be executed. This also means that the whole solution is wrong, so it will not be uploaded to the ADBMS schema.

The students have to make sure that the statements work in their schema. Let's see an example. If a student wants to upload a CREATE TABLE solution, and the name exists in their schema, the uploading of the CREATE TABLE statement will give error despite the fact that the statement is correct. So the student has to drop the table first or has to choose another table name.

Moreover the student may clean their schema because the UPLOAD_SOLUTION procedure, which executed their codes in their schema, can make a lot of schema objects, like tables, views, procedures, etc.

3.2. The C# program

It is not enough to verify whether the program code has syntactic errors or not. The semantic mistakes also have to be verified. However this cannot be automatically done by the database management system. The teacher verifies the semantic correctness. If the teacher works in the database management system, this job is not easy. A C# program has been written to support the teacher's work. It can help with many tasks of the teacher.

The first task with which the C# program can help the teacher is to administrate the students in the system. This means that the name and other data of the students have to be inserted into the appropriate table, and the groups have to be created for each section. The program has an import function, so the teacher can load the data into the table with a few clicks.

The next important task with which the C# program can help the teacher is to publish the descriptions of the tasks from week to week. The program has a window where the teacher can create groups for the tasks and another window where they can write the descriptions of the new tasks. In this window they can arrange the tasks into groups. Of course later the descriptions of the tasks can be modified because the teacher can also make a mistake in the text. A task can only be deleted until a student submits a solution for this task. The teacher has to give a deadline for each group of tasks. Later the teacher can modify the deadline if needed. The reasons of the modification can be unusual cases, for example, the database management system does not work, or a lesson finishes earlier for some reason, etc.

The main reason for developing the C# program is the next task of the teacher: they have to verify the solutions uploaded by the students.

The program can list the source code of the solutions. In the ADBMS schema there is a VERIFIER view, which selects not only the solutions but also the name and the section of the student who has uploaded the solution, the identifier of the task, and the identifier of the group of tasks. The view also shows for each solution whether the solution is accepted, refused, or the teacher has not yet dealt with it. The C# program uses this view to list the solutions. The program can filter the rows for these values. This list is very long at the end of the semester, so filtering is very important for the teacher.

When a teacher selects a solution from the previous list, the program shows the teacher the source code of the solution, the description of the task, and the name of the student in another window. This window can be easily read by the teacher. The teacher can set in this window whether he accepts or refuses the solution and can write a comment on the solution. The semantic verification of the solution uploaded by the student is done by the teacher with the help of this window.

In the ADBMS schema there are other views to assist the work of the teacher. The program can list and filter the content of the views. One of them lists the students who have not uploaded solutions for the given tasks and the identifier of the tasks and task groups. Another view lists the name of the students and shows how many per cent of the tasks each student has solved. The third view helps the teacher to examine how many per cent of the published tasks in each group of tasks are not solved by each student. With these three views the teacher can exactly follow the performance of the students.

4. Conclusion

I have used the software system in the teaching of PL/SQL in the course name Advanced DBMS 1 in the Software Information Technology BSc program at the University of Debrecen. (Vágner, 2014) made a survey about the active learning method which uses this software system. She find that the active learning method forces the students to work with the database management system independently; moreover, the teacher can give personalized support for the learning of the students. The results of her voluntary survey demonstrate that the students like the active learning method. The software system worked well, and I, as a teacher, liked the C# program. In the future, I plan to test the software system in the practice of Database Systems course with active learning method.

References

DuFour, R., DuFour, R., Eaker, R., Many, T. (2010). *Learning by Doing: A Handbook for Professional Communities at Work*, United States of America: Solution Tree Press

Gogoulou, A., Gouli, E. & Grigoriadou, M. (2009). Teaching programming with ECLiP didactical approach, *International Conference on Cognition and Exploratory Learning in Digital Age*. 204–211.

Kósa, M., Pánovics, J., Gunda, L. (2005). An Evaluating Tool for Programming Contests. *Teaching Mathematics and Computer*, 3/1 103–119.

Mason, R. T. (2013). A Database Practicum for Teaching Database Administration and Software Development at Regis University, *Journal of Information Technology Education: Innovations in Practice*, 12, 159–168.

Moore, M., Binkerd, C. & Fant, S. (2002). Teaching web-based database application development: an inexpensive approach, *United States of America: Journal of Computing Sciences in Colleges*, 58–63.

Polya, G. (1957): *How to solve it*, United States of America: Princeton University Press

Ramakrishna, M. V. (2000). A Learning by Doing Model for Teaching Advanced Databases. *Proceedings of the Australasian conference on Computing education*, 203–207.

Schank, R. G., Berman T.R. & Macpherson, K. A. (1999). Learning by Doing, *Instructional Design Theories and Models*, United States of America: Lawrence Erlbaum Associates 161–182.

Skinner, B. F. (1968):*The technology of teaching*, New York: Meredith Corporation

Vágner, A. (2014) Let's Learn Database Programming in an Active Way, *Teaching Mathematics and Computer* 12/2, 213–228

Authors

Anikó Vágner, Faculty of Informatics, University of Debrecen, Debrecen, Hungary, e-mail: vagner.aniko@inf.unideb.hu

Acknowledgement

The author thanks to her student, József Kányási for his technical support of development of the software application.

The publication was supported by the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project. The project has been supported by the European Union, co-financed by the European Social Fund.