A web-based tool for teaching and learning SQL

Josep Soler, Ferran Prados, Imma Boada and Jordi Poch

Abstract—**Structured query language (SQL) is one of the main topics of introductory database courses. In this paper, we present a web-based tool developed to automatically correct SQL statements. This tool has been integrated in a more general framework that supports, amongst others, the automatic correction of relational database schemas, entity-relationship diagrams and normalization exercises. The system assigns a personalized workbook composed by different SQL queries to each student. The student has to solve the exercises and enter the solution into the system. The tool corrects the exercises giving information about the correction. The student has the change to correct them and send a new solution. These steps can be repeated as many times as required until a correct solution is obtained. Student work is recorded in the database of the system providing the information required for supporting continuous assessment. Currently, the tool is used to reinforce teaching and learning in an introductory database course of our university with very promising results.**

Index Terms—**e-learning, database, learning SQL, teaching SQL, on-line assessment.**

I. Introduction

A fundamental tool to define, manipulate and retrieve information from relational databases is the Structured Query Language (SQL). This is the standard database language and one of the main topics of database courses in higher education. Despite the simplicity of SQL syntax, it is a difficult and a complex language to learn. In most of database courses after introducing main SQL concepts to the students they are recommended to practice, since the best way to learn this language is by practicing.

In our university, SQL is introduced in laboratory sessions of database courses. The teacher introduces main SQL concepts and then, exercises are proposed to the students to acquire practice. These exercises have to be solved on a database provided by the teacher. Crucial to success on this learning process is the student individual work and the teacher personalized attention. Unfortunately, the large amount of students of the majority of courses difficulties teachers to carry out an individual tracking of the work.

From our experience, we detected that if students have support when practicing, they feel more confident and they success. Motivated by this fact and by the difficulties we detected in laboratory sessions, we decided to develop a web environment that supports teaching and learning of SQL. The designed SQL tool is integrated in a more general e-learning framework denoted ACME (the acronym ACME stands for the catalan ”Continuous Assessment and Improvement of Skills”). ACME supports the automatic correction of different kind of problems, amongst them, problems related with main topics of database design courses such as the design of entity-relationship diagrams [8], the design of relational database schemas [7] and normalization exercises. In this paper, we present this SQL tool, how has been integrated in the ACME environment and how it is applied in introductory database courses.

The paper is structured as follows. In Section 2 related work is presented. In Section 3, the ACME environment is described. In Section 4, we present the SQL-ACME tool and in Section 5, how it has been integrated in our e-learning platform. In Section 6, we describe how it is used in introductory database courses. Finally, in Section 7 conclusions and future work are given.

II. Related Work

The goal of an SQL learning environment is to provide a tool where the student can write a SQL statement, visualize its execution, obtain immediate feedback about the correction and also modify the query until the correct result is obtained.

In the last years, different packages have been designed for teaching SQL. Dietrich et al. [2] propose WinRDBI, an educational tool that provides students with the capability to test their understanding of the formal relational query languages and also of SQL. Gove [4] proposes WebSQL, an interactive environment for executing SQL statements against a DBMS displaying the results via Internet. The system only provides the result of the query and the user has to analyze it to determine if it is correct or not. Sadiq et al. [9] propose SQLator, an online learning workbench that provides a collection of databases, each one with a pool of queries. The learner selects a query to work on, he enters the SQL statement into the system and an evaluation engine, based on heuristic algorithms, corrects it. SQL-Tutor, proposed by Mitrovic [5], is an intelligent tutor system with a set of problems for specified database and the ideal solutions to them. Student solutions are compared with the ideal one. Coleman [1] presents AsseSQL an online tool to test SQL formulation skills.

Departament Informàtica i Matemàtica Aplicada.
Universitat de Girona. Spain
({josep.soler, ferran.prados, imma.boada, jordi.poch}@udg.es)

1-4244-0406-1/06/$20.00 ©2006 IEEE.

Although, all these environments can be used as reinforcement to SQL lecture sessions, from our point of view, most of them have a main limitation. They have been designed considering only the role of the student, i.e. teachers work has not been taken into account. They do not support lecturer tasks such as, continuous assessment, tracking of students work to detect weak points, abilities to obtain statistics of common errors, number of attempts required to solve a query, etc. For a lecturer all these abilities are very valuable since they provide information of student progress and also which topics need further work.

III. ACME, OUR E-LEARNING PLATFORM

In 1998, we started to develop ACME, an e-learning platform to improve both teaching and learning at the technical/engineering degrees of our university [6,7,8]. Besides the functionalities of a common e-learning platform our environment includes the following main features:

- *Support automatic generation and correction of problems*. In most of engineering courses there is a lot of practice. Students have acquired the theoretic concepts when they are able to solve the problems related with these. Due to the importance of practice the platform integrates the modules required to automatically generate and correct exercises of different subjects.
- *Provide students with a friendly scenario to solve practical problems.* The possibility to automatically generate and correct different exercises allows us to give personalized attention to each student. This means exercises specifically designed for the student and continuous feedback from the platform.
- *Support continuous assessment.* All the student solutions are stored in the database of the platform providing teachers the information required to perform continuous assessment.

ACME is composed of different modules. Below we describe the main modules of the platform with their functionalities:

- A *repository of problems* maintains all problems entered by teachers in the system. This repository allows teachers to share material.
- The *workbook generation module* generates a personalized workbook for each student. A workbook is composed of exercises grouped into topics. The exercises are selected from the repository of problems following the teacher indications. Problems can be added to the workbook at any time.
- The *correction module* provides an environment to correct on-line a solution designed for a given problem. This module has a specific component for each typology of problems supported by the platform. The correction module is the core of the platform.
- The *system database* stores all student work. The teacher selects a set of problems related to each topic from the repository. From these exercises the system automatically generates a different workbook for each student. Students have to solve the problems, using the correction environment, and send the solutions to the system before a fixed deadline. All the student solutions, correct and incorrect, are stored in the system database.
- The *continuous assessment module* collects from the data base quantitative data about student work such as number of errors, types of errors, time taken to complete the problem, etc. The students' progress through their personal exercise book provides the basis for the continuous assessment of their skills and gives valuable information about their difficulties. This information can be used to guide the student through those topics which present a greater difficulty for them.
- The *communication channel* establishes a virtual communication channel between the teacher of a group and all the students that compose this group. This channel can be used by the teacher as a virtual tutoring system.

The system supports different type of users: students, teachers and administrator. Interface windows are specifically designed for each role and also for each subject. It has to be taken into account that the functionalities required for a specific subject can be different from the required for another one.

IV. THE SQL-ACME TOOL

In this section, we present the proposed SQL-ACME tool. First, we present the main design decisions. Then, we describe the tool from a technical perspective giving information of main modules that compose it. These modules are the SQL student interface, the structure of a SQL problem and the SQL correction module.

A. Design decisions

Our goal is to develop a SQL-web environment that provides support to both students and teachers. The main requirements and decisions we take into account for the design of this web environment are:

- *Easy access.* A first requirement of the tool is that it has to be easily accessible, either for teachers and students. Any kind of installation has to be required, only a web browser.
- *SQL training environment.* The system has to provide SQL exercises and all the facilities required to write a SQL statement and obtain its correction.
- *Support any SQL statement.* Most of the existing SQL packages only support SELECT queries. Our environment has to support any kind of SQL statement.
- *Support different databases.* To improve training, our tool has to support more than one database, in this manner we can diversify the set of exercises that can be solved through the tool.
- *Online-correction.* When one student enters a solution of a SQL exercise, the correction has to be immediate. In case of errors, the student has to have the change to enter new

solutions.

- *Record of students work.* The system has to record all the students work, i.e. all attempts entered for a question until the correct one has been obtained.
- *Continuous assessment.* The system has to provide teachers all information and abilities required to carry out continuous assessment.
- *Enhance the student-teacher contact.* The system has to provide a communication channel that enhances the student-teacher contact, either to ask for doubts or to give hints of how to solve queries.

To develop our SQL-ACME tool, the starting point is the ACME e-learning platform described in the previous section. ACME provides some of the required features, such as easy access, record of student work, continuous assessment and the student-teacher communication channel. Therefore, in order to satisfy all our requirements, our objective is to design and integrate in the platform the modules that support SQL exercises. These modules are the SQL-student interface and the SQL correction module.

B. *The SQL-Student Interface*

The SQL student interface is the first window that appears when the student enters into the system and selects a SQL problem. The problem that appears in the student interface is a problem assigned by the workbook generation module to the student. This module selects the exercises from the repository following the teacher indications. The structure of an SQL problem is determined by a set of rules (described in Section C) and only if the problem satisfies them can be stored in the repository.

In Fig.1 the SQL-Student interface with a SQL example from the popular Elmasri and Navate database text [3] is shown. From top to down, the main components of this window are:

1) The *problem descriptor area* that contains the query that has to be answered by the student and the set of tables required to answer this question. The tables correspond to a database stored in the system and created using the specifications of the problem. Selecting one of the tables, the student can access to all the information related to it. For example, if we select the PROJECT table represented in Fig.1, we obtain the information of Fig.2 with all the details of the attributes, primary key, etc.
2) The *answering area* where the student enters his proposed solution. Since the solution is a text sequence to enter the solution no special buttons are required.
3) The *correct button* sends the student solution to the correction module. This module, as will be seen in next section D, corrects the solution and returns a feedback message about the correction to the student.
4) A *tabular representation of all student solutions* with a direct access to each one of them. All the solutions are stored in the database of the system and they can be accessed, either by the student or by the teacher.

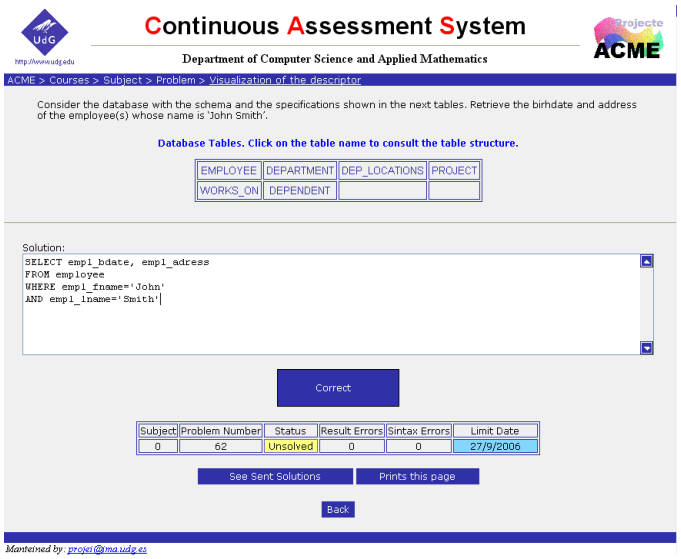

Fig. 1. SQL Student Interface

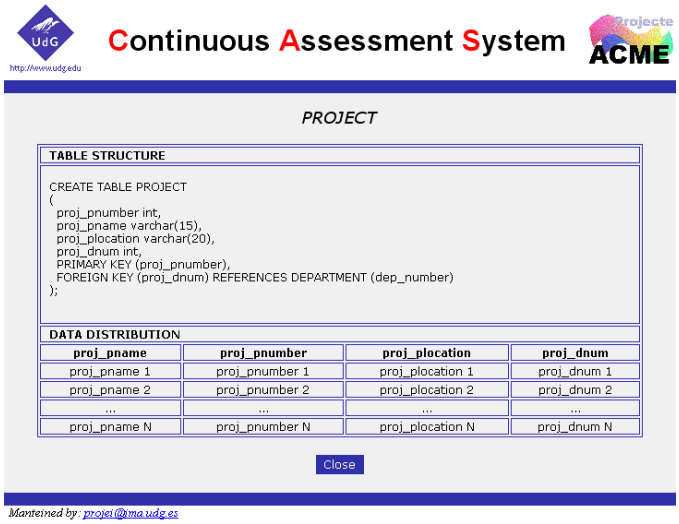

Fig. 2. Detailed information of a table

C. *Structure of a SQL-problem*

SQL problems are created by teachers and are stored in the problem repository of the system. The problem assigned to the student is automatically generated by the problem generation module using the information stored in the SQL problem.

The structure of the SQL problem entered by the teacher has four different parts:

- The first part is an introductory sentence common to all problems. In the example of Fig.1 the introductory sentence is

  Consider the database with the schema and the specifications shown in the next tables.

  For a same problem different introductory sentences can be recorded. In this manner, the system is capable to generate different versions of the same problem [6].

- The second part of the problem is a list of possible questions where each question indicates the information it has to be retrieved from the database. Each one of these questions has an identifier and for each one, the teacher has to enter at least one correct solution. Since more than one correct solution is possible, when entering them into the system they have to be identified. Therefore, for each question we stored the following information:

```
<Question_1><Solution1><Solution2>…
<Question_2><Solution1><Solution2>…
…
<Question_i><Solution1><Solution2>…
```

For the problems represented in Fig.1 and Fig.3 the information recorded in the repository of problems is:

```
<Question_1 – Retrieve the birthday and address
of the employee whose name is John Smith>

<Solution_1 – select emp_bdate,empl_address from
EMPLOYEE where empl_fname='John' and
empl_lname='Smith'>

<Question_2 – Retrieve the name (first name and
last name) of all employees who work for the
'Research' department>

<Solution_1 – select empl_fname, empl_lname from
EMPLOYEE,DEPARTMENT where dep_name='Research' and
dep_number =empl_dep_num>

<Solution_2 – select empl_lname, empl_fname from
(EMPLOYEE join DEPARTMENT on dep_number =
empl_dep_num) where dep_name='Research'>
```

Observe that in this case, there are two different questions. For the first, the teacher has entered one correct solution and for the second question two solutions.

- The third part of the SQL problem is the DBMS that has to be applied to solve the problem. The system supports different DBMS such as Oracle, SQLServer, Postgres, etc.. For this reason, in the problem we have to identify the DBMS that have to be used to solve it.

- The last part of the problem is the set of instructions required to create the tables in the selected DBMS and also, the instructions to enter data in these tables. i.e. the corresponding CREATE TABLE and INSERT instructions. Observe in Fig.1 that below the problem descriptor there are the names of the tables created with the instructions stored in this last part. The database creation is done before assigning the problem to the student.

All the information recorded in a SQL problem is used by the problem generation module to define the student workbook. To generate the problems the teacher has to determine the number of SQL exercises he wants to assign to each student.

D. The SQL-correction module

The correction module evaluates the student solution and determines if it is correct or not. To carry out this process it uses the correct solutions entered by the teacher and stored in the repository of problems.

The correction is based on a matching process that analyzes if there is a matching between the solution entered by the student and one of the correct solutions of the problem. This process has to consider all possible combinations of attributes, table names, etc. Note that two solutions can be correct, even though the order of their elements is different. In the next example, the teacher solution is

```
<select empl_fname, empl_lname from EMPLOYEE,
DEPARTMENT where dep_name='Research' and
dep_number =empl_dep_num>
```

and the student solution is

```
<select empl_lname, empl_fname from DEPARTMENT,
EMPLOYEE where dep_number =empl_dep_num and
dep_name='Research'>.
```

In this case, the student' solution is correct, but it is different from the solution entered by the teacher.

To describe the correction strategy we consider the three different situations that can be given.

1. The first situation is given when there is a matching between the student solution and the teacher solution. In this case, the solution is correct. This case is illustrated in Fig.3 and Fig.4. The first figure, Fig.3, shows the student interface with the entered solution. Fig. 4 is the result of the correction that appears after pressing the correct button.

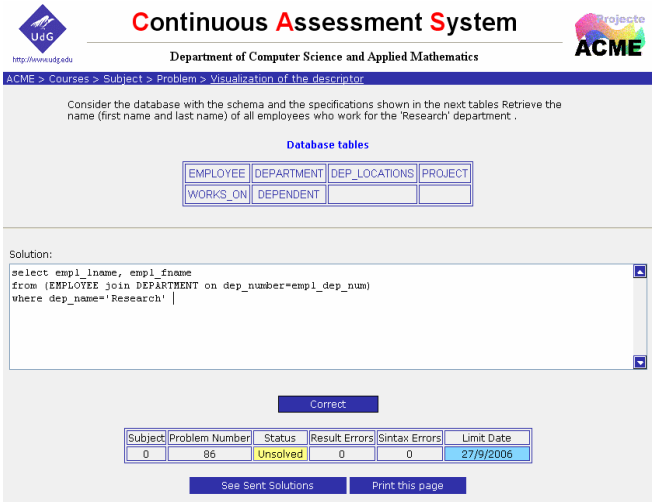

Fig. 3. Student solution

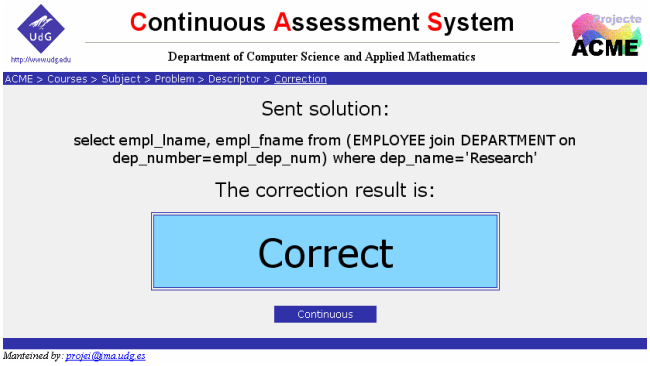

Fig. 4. Result of the correction process.

2. The second situation is given when: (i) there is no matching between the student solution and the teacher solutions and (ii) the statement is a SELECT. SELECT is the most common statement of this type of exercises. In this case, both the student solution and the teacher solution are executed on the DBMS and then, retrieved data is compared. If retrieved data is the same, the student solution is correct. On the contrary, it is incorrect. Note that to compare if all retrieved data is the same, a new comparison process is required. The strategy applied in this situation is only possible for SELECT statement, since the execution of other statements such as INSERT, DELETE, UPDATE,…, will modify the problem database and this is not allowed by our system.

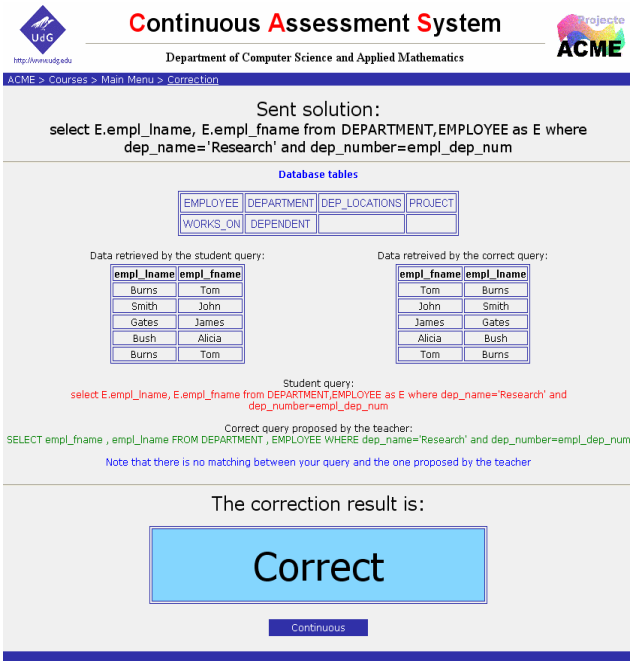

Fig. 5. Correct student solution but different to the teacher solution.

In Fig. 5, we illustrate a case in which the solution entered by the student is correct but, different from the solutions stored in the system. In this case, the matching process fails, but since the solution is a SELECT we are in the second situation and hence both, the student and the teacher solutions, are executed against the DBMS. Observe that retrieved data, presented in Fig.5, is the same but in a different order. Although the student solution is correct, in this case we also present the solution entered by the teacher since we consider that this is the best one.

In Fig.6 and Fig.7 we illustrate two different incorrect solutions. In Fig.6 the system has detected a syntax error. In the second case, Fig.7 the system has executed the student and the teacher solutions and retrieved data is different.

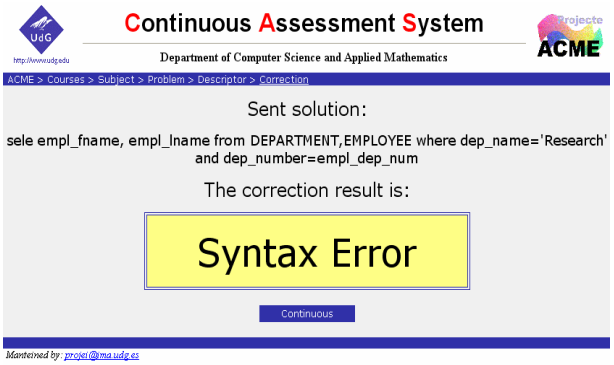

Fig. 6. Student solution with a syntax error

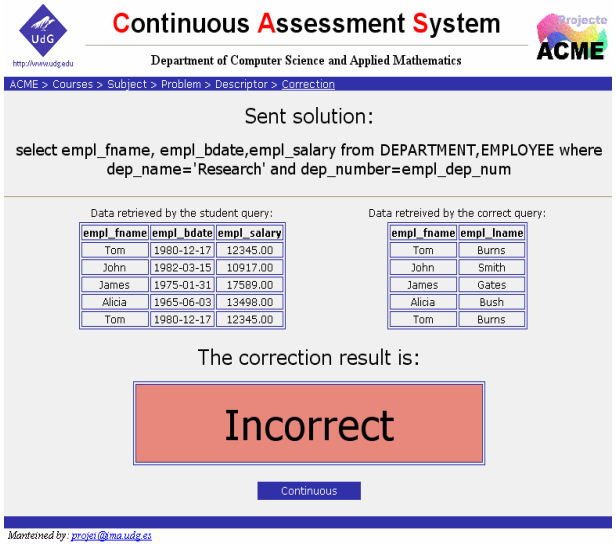

Fig. 7. Incorrect student solution since retrieved data is different

3. The third situation is given when: (i) there is no matching between the student solution and the teacher one and (ii) the statement is not a SELECT. In this case the student solution is considered incorrect.

V. Integration of the SQL-tool in our E-learning platform

The core of our e-learning platform is the correction module which has been designed in such a way that different type of problems can be supported and easily integrated. Therefore, to satisfy all our requirements the last phase consists in the integration of the SQL-ACME tool into the ACME e-learning platform.

In Fig.8 we describe, at high level, the structure of the ACME e-learning platform and the modifications that have been carried out to integrate the SQL-ACME tool. The modules that have been integrated are represented in grey.

On a first layer, there is the web browser required to enter into the e-learning platform. To enter into the system a username and a password is required. Then, according to the user role (student, teacher or administrator) and the selected subject, the different abilities of the ACME e-learning platform can be accessed. In this layer we have integrated the SQL-student interface. The main modules of the platform compose the second layer. These modules are: the workbook generation module, the continuous assessment module, the communication channel and the correction tool, composed by different correction modules specific of each kind of problems. It is in this second level where we have to integrate the SQL correction module. Since the module supports different DBMS (Oracle, SQLServer, MySQL,…), to have the application independent of these DBMS we have used the Pear libraries. The database specific of each problem is created and stored in the system. Only one database per problem is created and it can not be modified.

The last level of our environment maintains the repository of problems, with all the problems of all the supported subjects, and the database of the system where all students work is recorded.

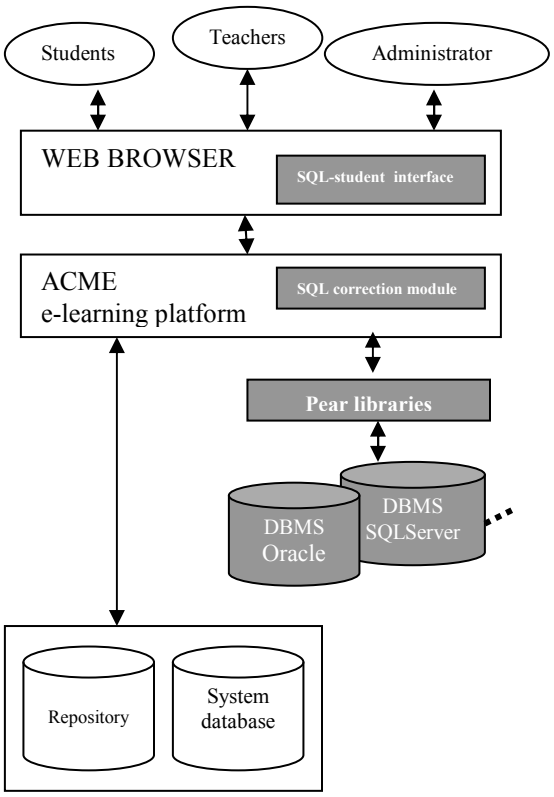

Fig. 8. Structure of the ACME environment. In grey the modules integrated in order to support SQL exercises.

VI. Experimental Results

The evaluation of the SQL-ACME tool has been carried out on a group of 62 students of an introductory database course of the University of Girona (Spain). The experiment aims to compare the classical methodology with the new methodology based on the SQL-ACME tool. With this purpose we divide the group into two groups of 31 students. The first group, denoted group A, follows the classical SQL teaching methodology and the second one, denoted group B, uses the proposed SQL-ACME tool for teaching and learning. In both groups, eight laboratory sessions of two hours are spent to teach SQL.

In Group A, in each SQL laboratory session the teacher introduces main SQL concepts and then exercises are proposed to the students to acquire practice. These exercises are solved in laboratory sessions using a database provided by the teacher.

In each laboratory session, the teacher presents two or three examples and then students have to solve the others. The teacher is in the laboratory to solve possible student questions. At the end of each session, each student has to submit the finished exercises. The teacher uses this information to track students work. If student have not finished all the laboratory exercises, the teacher proposes them to solve as homework and in case of questions, to contact with him, by e-mail or going to his office.

At the end of each laboratory session, the teacher of group A has to correct all the exercises of the students. The teacher complains that he is not able to know if the student has finished homework, since a low percentage of students go to his office asking for help. The student-teacher communication is reduced to the laboratory communication.

In Group B, the teacher also introduces main SQL concepts in each laboratory session by using SQL-ACME tool. Then, a personalized workbook with different SQL exercises is assigned to each student. The exercises of the session have to be solved during the session and are the same exercises that have been assigned to students of group A. If not all the exercises are solved in the class, we give the chance to solve at home using the SQL-ACME tool. The number of SQL exercises assigned to students of group A and B is the same.

In this group at the end of each laboratory session, the teacher only has to retrieve information from the ACME database to track student work. The teacher has exact information of the number of solved exercises, he know the exercises the student has to solve as homework and also if the student have solved them or not. The student-teacher

communication channel provided by the system allows teacher to send mails to the student giving help and hints of how to solve errors. It is not necessary that the student goes to the teacher office. The communication channel of system strengthens the student-teacher relationship.

At the end of the SQL sessions, students of both groups have to pass a test exam. The test is composed of ten questions similar to the exercises solved in laboratory sessions. The test is the same for Group A and Group B. In Table I, we collect the results of the test for the two groups where, A represent one or zero errors in the test, B represents 2 or 3 errors, C represents 4 or 5 errors and D more than 5 errors. Observe that the results of Group B are better than the results of Group A.

TABLE I
RESULTS OF THE FINAL SQL TEST

	GROUP A	GROUP B
A	6	8
B	13	16
C	9	4
D	3	3

At the end of the SQL sessions, we also give a questionnaire to the student with the next three questions:

(1) Do you feel motivated to do SQL exercises?
(2) Do you feel supported when practicing SQL?
(3) Do you consider that you have acquired SQL skills?

They have to answer with a number from 1 to 5, being 1 the minimum and 5 the maximum. In Tables II and III, we collect the results obtained for each one of the groups. Note that the students of group B feel more motivated, more supported when practicing and also, more student consider that SQL skills have been acquired. These results are also corroborated by the results of the test exam.

TABLE II
RESULTS OF THE QUESTIONNAIRE OF GROUP A

GROUP A	**1**	**2**	**3**	**4**	**5**
Question 1	4	10	8	7	2
Question 2	3	5	13	10	-
Question 3	2	5	15	6	3

TABLE III
RESULTS OF THE QUESTIONNAIRE OF GROUP B

GROUP B	**1**	**2**	**3**	**4**	**5**
Question 1	2	3	7	14	5
Question 2	2	1	11	11	6
Question 3	2	3	9	14	3

Although a more complete experiment has to be done, from the teacher point of view, before evaluating the results of the exam, our first impressions are positive. The tool provides gains with respect to the classical teaching methodology, since it:

- Offers a system for the continuous assessment of the student's progress. We can retrieve statistics on different aspects of the problems, for instance the number of attempts required to solve a problem, the main errors, main difficulties detected by students, etc. This information can be collected before a laboratory session and can be used to guide the new session, since we have information of student weak-points.
- Makes personalized attention to the student easier. Time required to correct SQL exercises is reduced considerably since the system provides an automatic correction. The teacher has more time for personalized attention. We can track student work and send messages to them through the communication channel enhancing the student-teacher relationship. The degree of participation in the classes increases with respect to Group A. Student feel supported all the time.
- Assesses the degree of participation of the students. We have accurate information of the state of each exercise.

From the student point of view, students of group B feel more motivated to practice SQL than the other group. They consider as the main advantage of the system the fact of having immediate feedback about the solution proposed to a given exercise. During the different sessions students were asked to comment on the problems they found while using the system. The responses are very positive. The students feel motivated to solve the proposed problems. The possibility to correct a problem in real time encourages them to work until a correct solution is found. They also consider that their relationship with the teacher has been enhanced and now they contact or go to the teacher office when they have doubts.

VII. CONCLUSION

We have presented a SQL-ACME tool developed to support teaching and learning of SQL. Main feature of this tool is that different to other e-learning environments it has been designed to support both teacher and student tasks. SQL-ACME has been integrated in a more general framework that supports problems of main topics of database design courses such as the design of entity-relationship diagrams, the design of relational database schemas and normalization exercises. Moreover, the tool supports any kind of SQL statements. No DBMS installation is required, only a web browser is needed.

The tool has been used in an experimental group of students with very promising results. Future work will be centered on the definition of relational algebra problems. The idea is to develop a web environment that supports main topics of any introductory database course.

REFERENCES

[1] J.Coleman, Online Assessment of SQL Query Formulation Skills. Fifth Australasian Computing Education Conference (ACE 2003), Adelaide, Australia, 4-7 February 2003, 247-256

[2] S.W. Dietrich, E.Eckert, K.Piscator: WinRDBI: a Windows-based relational database educational tool. Proceedings of the 28th SIGCSE Technical Symposium on Computer Science Education, 1997, San Jose, California, USA, February 27 - March 1, 1997. ACM 1997, ISBN 0-89791-889-4 SIGCSE 1997: 126-130

[3] R.Elmasri and B. Navathe. Fundamentals of DataBase Systems. 3rd edition.Addison-Wesley, 2000.

[4] A.Gove. "WebSQL: An Interactive Web Tool for Teaching Structured Query Language," Proceedings of the 2000 Americas Conference on Information Systems, Long Beach, CA, August 10-13, 2000.

[5] A.Mitrovic: Learning SQL with a computerized tutor. Proceedings of the 29th SIGCSE Technical Symposium on Computer Science Education, 1998, Atlanta, Georgia, USA, February 26 - March 1, 1998, 307-31

[6] F.Prados, I.Boada. J.Soler and J.Poch.. *"Automatic generation and correction of technical exercices"*. International Conference on Engineering and Computer Education ICECE 2005.

[7] F.Prados, I.Boada. J.Soler and J.Poch., *"An Automatic correction tool for relational database schemas"*. Proc. IEEE International Conference on Information Technology based higher Education and Training ITHET 2005, S3C, 9-14 .

[8] F.Prados, I.Boada. J.Soler and J.Poch., *"A web-based tool for entity-relationship modeling"*. International Conference on Computational Science and its Applications ICCSA 2006.

[9] S.Sadiq, M.orlowska, W.Sadiq, J.Lin, "SQLator-An Online SQL Learning Workbench" ITiCSE. Annual Conference on Innovation and Technology in Computer Science Education 2004