A Web-based Problem-Solving Environment for Database Normalization

Josep Soler, Imma Boada, Ferran Prados and Jordi Poch

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

Abstract. ACME is a web-based problem-solving environment developed at the University of Girona to reinforce teaching and learning of technical/engineering degrees. ACME assigns different exercises to the students and allows them to enter solutions, obtaining on-line the correction and feedback about the correctness. The kernel of the environment is the set of correction modules specifically designed for each type of problems supported by the system. In this paper, we present the module developed to support and automatically correct database normalization problems. We present how this new ACME functionality has been applied in an introductory database course of our university and the first results and impressions.

1 Introduction

Database design is a common subject in undergraduate database courses. In these courses students are taken through all stages of database development (analysis, conceptual, logical and physical design) and are introduced to main database topics. Amongst these topics database normalization is an important one. Normalization is a technique used in the designing of relational database schemas. Its purpose is the creation of relations with no redundant data and that can be modified in a consistent and correct manner. The basis of normalization theory is the normal forms. There are five normal forms and a relation is said to be in a particular normal form if it satisfies the set of constrains imposed by it.

As most of the database concepts, the best way to learn and understand normalization is by practicing. For this reason, in the majority of database courses after introducing in the lectures all the concepts related with normalization theory, exercises are proposed to the students to acquire practice. 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. To overcome this limitation we design a set of web-based problem-solving modules able to support database normalization problems. These modules are integrated in a more general e-learning tool, denoted ACME, developed at the University of Girona to reinforce teaching and learning of technical/engineering degrees [1, 6, 9]. The resulting e-learning environment assigns to each student a set of database normalization problems selected from a common repository. Each problem describes a database in terms of one or more relations with their attributes and the functional dependencies they have to satisfy. The student, following the problem constrains, has to design a normalized relational schema and enter it into the system through a user friendly interface. The correction tool corrects the design and shows detected errors giving advice of how to solve them. In case of a non-correct solution there is a chance to send a new design. These steps can be repeated as many times as required until the problem is solved satisfactorily. An important feature of the proposed environment is that all the student work is stored in the database of the system, providing teacher the information required to carry out continuous assessment and student tracking. Currently, such an environment is used in our university as a complement (blended e-learning) to traditional classroom database courses with very promising results.

The paper has been structured as follows. In Section 2 related work is presented. In Section 3 a general view of the problem-solving environment is given. The kernel of our system is described in Section 4. Experimental results are collected in Section 5. Finally, in Section 6 conclusions and future work are given.

2 Related Work

Most of existing tools for database normalization have not been conceived as web-based environments but as software applications that automate some step of the normalization process [2, 4]. To best of our knowledge there are very few web environments that support database normalization learning. Mitrovic [5] propose NORMIT a web-enabled tutor for database normalization where the student enters and selects a problem. Then, he goes through a number of steps to analyze the quality of the database following a fixed sequence of actions: determine the candidate keys, the closure of a set of attributes and prime attributes, simplify functional dependencies, etc., with an specific web page for each task. Student solutions are analyzed by the system receiving feedback. The environment is a question-answer based approach since student does not propose a solution, he answers if the displayed solution is correct or not. Fong et al. [3] present a methodology of using a set of human computer interface rules to develop a computer aided instruction for database normalization. The development of the normalization environment is presented as a case study.

3 A General View of the Developed Problem-Solving Environment

The web-based tool we propose has been conceived to reinforce teaching and learning on introductory database courses and also with the idea of going one step further giving support to the teacher. On the one hand, our system has to support database normalization exercises and, on the other hand, it has to provide teachers with a continuous assessment tool and all the facilities required to track student work. To reach our objective we develop ACME an e-learning system that supports different kind of exercises typical of technical degrees, such as mathematical, programming, etc. [1, 6, 8–10].

The main components of ACME are described below. A repository maintains the problems entered by teachers into the system. A work-book generation module automatically generates personalized work-books for each student by using problems of the repository. Problems can be added to the work-book at any time. A correction module specifically designed for each kind of problem. When the student enters into the system he selects a problem from the work-book and enters a solution. The correction module corrects on-line student solutions giving feed-back about the errors. In case of erroneous solutions the student has the change to enter a new one. All solutions entered by the students are stored in the database of the system. A continuous assessment module collects from this database 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. A set of tools, such as forum, chat, mailing, etc. provides a communication channel between the teacher of a group and all the students that compose it. This channel can be used by the teacher as a virtual tutoring system.

The correction module is the kernel of the e-learning environment. The integration of a new type of problems requires the development of its specific correction module and also the set of interfaces required to enter a solution of the problem into the system.

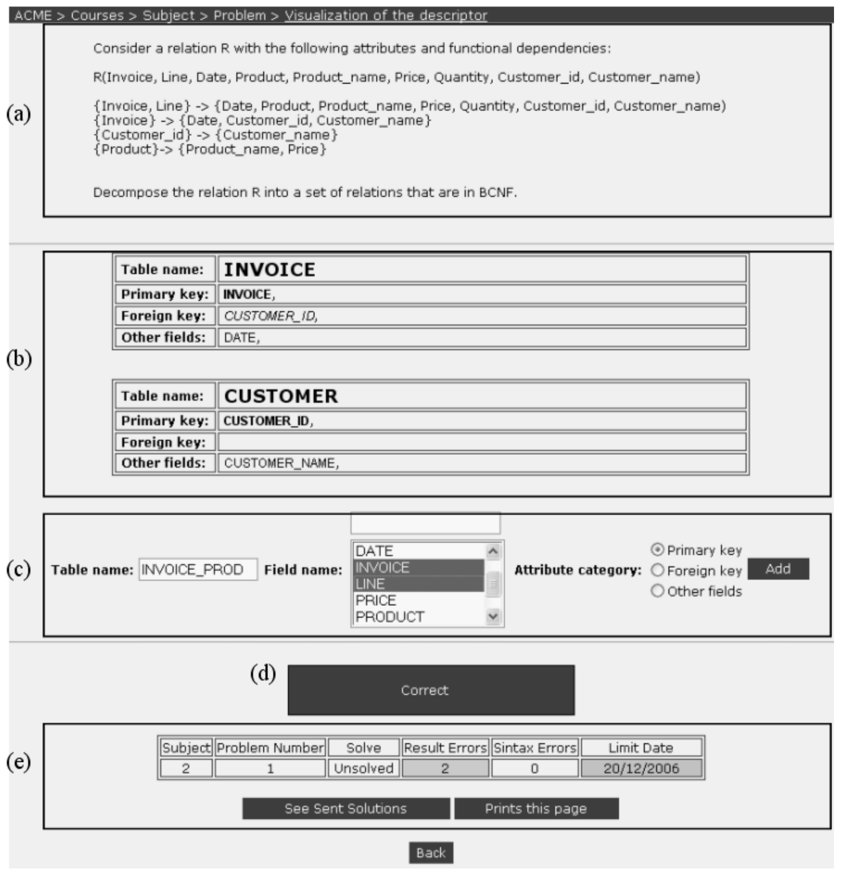
4 The Correction Module

In this section, we present the module developed to support and automatically correct database normalization problems. To describe how the correction module proceeds we have to consider: (i) how is a normalization database exercise; (ii) how the student solution is entered in the system, and (iii) how correction and feedback are carried out.

4.1 Normalization database exercises

The basis of normalization theory is the normal forms. Normal forms represent a sequence of steps that leads to a database structure that is well designed. First normal form (1NF) ensures that no relation within the database contains any repeating groups. In second normal form (2NF), database relations are examined in order to identify whether any non-key attributes are partially dependent upon the keys of the relations. When one is found it is decomposed. Finally, relations of the database are examined to identify whether any non key attributes are transitively dependent upon the keys of the relations. When one is found a decomposition process is carried out to remove these attributes from their relations and re-group them. Although there are five normal forms the goal of the majority of our exercises is to have all the tables in the third normal form (3NF) or the Boyce-Codd normal form (BCNF). The first two normal forms are intermediate steps to achieve the 3NF or BCNF. The concept of functional dependencies is the basis for these normal forms. It has to be remarked that our system supports normalization until anyone of the five normal forms.

Taking into account all these considerations, a normalization exercise consists in a problem descriptor composed of two main elements: a set of attributes represented in the database and a set of functional dependencies that these attributes have to satisfy (see Figure 1(a)).



ACME > Courses > Subject > Problem > Visualization of the descriptor
(a)
(b)
(c)
(d)
(e)
Correct
Add
See Sent Solutions
Prints this page
Back

Consider a relation R with the following attributes and functional dependencies:

R(Invoice, Line, Date, Product, Product_name, Price, Quantity, Customer_id, Customer_name)

{Invoice, Line} -> {Date, Product, Product_name, Price, Quantity, Customer_id, Customer_name)
{Invoice} -> {Date, Customer_id, Customer_name}
{Customer_id} -> {Customer_name}
{Product}-> {Product_name, Price}

Decompose the relation R into a set of relations that are in BCNF.

Table name:	**INVOICE**
Primary key:	**INVOICE,**
Foreign key:	*CUSTOMER_ID,*
Other fields:	DATE,

Table name:	**CUSTOMER**
Primary key:	**CUSTOMER_ID,**
Foreign key:	
Other fields:	CUSTOMER_NAME,

Table name: INVOICE_PROD **Field name:** DATE, INVOICE, LINE, PRICE, PRODUCT **Attribute category:** [✓ Primary key] Foreign key Other fields

Subject	Problem Number	Solve	Result Errors	Sintax Errors	Limit Date
2	1	Unsolved	2	0	20/12/2006

Fig. 1. Student Interface. (a)Problem descriptor. (b)Graphical representation of the student solution. (c)Relation dialog. (d)Correct button. (e)Index to all entered student solutions.

4.2 Student Solution

Solving a database normalization problem consists in grouping attributes into relation schemas that are in a normal form. A solution of a normalization problem can be represented as a set of relations that group the attributes of the database satisfying the imposed functional dependencies. Therefore, to enter the solution into the system we have defined a friendly interface that allows entering tables and all the required information. This interface is similar to the one we propose in [7] for solving relational database problems. The main parts of the interface are presented in Figure 1(c). From left to right we can distinguish three items. The first item is a text box where the name of the relation has to be entered. The second item is a list with all the attributes presented in the problem descriptor. For each relation entered in the system the student selects from this list the set of attributes that define it. The third item is a list of attribute categories. Each time an attribute is selected we have to determine if it is a primary key, a foreign key or a normal attribute.

In order to facilitate student interaction, the system also provides a graphical representation of the different relations that have been entered (see Figure 1(b)). Modifications on the entered solution can be done by selecting the item to be modified on this graphical representation.

Once the solution has been entered, to obtain the correction the student has to press the correct button (see Figure 1(d)). All solutions are stored in the database of the system. These solutions can be consulted by the student selecting one of them from the tabular representation displayed on the screen (see Figure 1(e)).

4.3 Correction and Feedback

The correction strategy is based on a comparison strategy which compares the student's solution with the possible correct solutions entered by the teacher and stored in the repository jointly with the normalization problem descriptor. To describe this process first of all, we have to consider how solutions are represented in the system. Even though, in most of the cases, the solution of a normalization problem is unique, the system is able to maintain several solutions.

The correct solutions are entered by the teacher and each one consists of a set of relations with the corresponding attributes including primary key, foreign keys and other attributes. Each solution has assigned an identifier. For each solution relation the name of the relation and the three groups of attributes that compose it are recorded. An example of this representation is given below.

```
Solution_identifier
    <TABLE NAME>,<PRIMARY KEY>,<FOREIGN KEYS>,<ATTRIBUTES>
    <TABLE NAME>,<PRIMARY KEY>,<FOREIGN KEYS>,<ATTRIBUTES>
    ...
```

The solutions entered by the student are stored in the database of the system and represented in the same manner as the correct solution.

4.4 The Correction Strategy

The correction process is a four step based comparison strategy that ends when a complete coincidence between a correct solution and the student's one is found. The comparison process stops when an error is detected and according to the step and the kind of error a feedback message is sent to the student. The message gives some hints of how the error can be corrected. Possible examples of error message are: "More tables are required", "You have to reduce the number of tables", "The table X is not correct", etc. When the student enters a solution into the system he has not any restriction on the name of the tables, therefore to carry out the correction process only the name of the attributes and its category (primary key, foreign key or others) are taken into account. The four phases of the correction strategy can be defined as follows:

– Compare number of tables. On a first step, the process compares if the number of tables entered by the student is equal to the number of tables of one of the correct solutions stored in the repository. In case that more than one solution coincidence exists all of them have to be evaluated. When a coincidence is found the second phase of the process starts, otherwise an error message appears on the screen.
– Compare primary keys. We compare the primary keys of the student's tables with the solution selected in the previous phase. If there is one solution with the same primary keys we start the next phase processing this solution. If there is an error in this phase, the process is applied to another of the solutions previously selected. In case that none of the solutions matches, the corresponding error message appears on the screen.
– Compare foreign keys. As in the previous case, an once each table has been identified by means of its primary key, the process looks for the coincidence between all the foreign keys. If there is not, a message appears, and otherwise we apply the last phase.
– Compare other attributes. Finally, the attributes classified as others are compared proceeding as in the previous phases.

When all the phases succeed the student's solution is considered correct. Note that the matching process required to perform all the comparison is quite complex because there are a lot of possible combinations of the attributes and also, the order in which the tables are entered. Moreover, the process always has to maintain all the correct solution candidates to be matched with the student solution.

5 Experimental Results

Currently, the system is being used in an experimental group of an introductory database course with 62 students. In lecture sessions theoretical concepts of database normalization and some example exercises are given. After the two first sessions a personalized work-book with three exercises has been assigned to each student. Exercises are not compulsory but we strongly propose student to solve them to reinforce theoretical concepts. Only 9 students do not solve the exercises. Therefore the degree of participation has been of 86%.

Statistics of the obtained results are collected in Table 1. We show for each exercise the number of solutions the student has to sent until the correct one is obtained. Although the first exercise was the easiest one, students require more times to solve it. This was an expected result since student were not familiar with the system. However, more than a half of the students solve the exercise in two or less times. Although the second and third exercises were more difficult than the first one, the obtained results were better since student were familiar with the environment.

Exercise	One solution	Two solutions	Between 3 and 5 solutions	more than 5 solutions	Unsolved
1	24	12	11	4	2
2	27	11	8	4	3
3	28	14	6	1	4

Table 1. Statistics from an experimental group.

During the different sessions students were asked to comment on the problems they faced while using the system. The responses were 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. The students' impressions have also been positive since to access the system they only need an internet connection and a browser.

From the teacher's first impressions, we can remark that the environment is easy to use. More importantly, it provides gains with respect to the classical teaching methodology in the sense that it offers a system for the continuous assessment of the student's progress, makes personalized attention to the student easier and assesses the degree of participation of the students.

6 Conclusions and Future Work

In this paper we have presented a web-based environment for teaching database normalization in introductory database courses. The proposed system corrects on-line database normalization problems that have been automatically assigned to students. The system also maintains all the information required to carry out continuous assessment and student tracking. The system is being used in our university with very promising results.

Our future work will be centered on the definition of new modules to support the correction of relational algebra exercises. Our goal is to integrate in a unique e-learning environment the modules required to correct normalization problems, relational database schemas, entity-relationship diagrams and SQL queries [7, 8, 10]. In this manner, our web-based tool will support main topics of introductory database courses.

References

1. I.Boada, F.Prados, J.Soler and J.Poch. A teaching/learning support tool for introductory programming courses. Proceedings IEEE International Conference on Information Technology based higher Education and Training 2004, ITHET pp.604-609.
2. R.Fangury and B.A.Kleen. normalization shootout: a computing game that impacts student learning. Issues in Information Systems Volume IV, n.1 2005, pp.21-25.
3. J.Fong, I.Kwan, M.Nig and K.L. Lau. Fifth Normal Form Made Easy with novel Web-Based CAI HCI. Proceedings of ICWl 2002, LNCS 2436, pp.398-410.
4. M.Kolp and E. Zimanyi. Prolog-based algorithms for database design. Proceedings of the 6th International Conference on Practical Applications of Prolog, PAP'98
5. A. Mitrovic. NORMIT: a Web-Enabled Tutor for Database Normalization. International Conference on Computers in Education ICCE 2002.
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, Madrid, 2005.
7. F.Prados, I.Boada. J.Soler and J.Poch. An Automatic correction tool for relational database schemas. Proceedings of 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. LNCS 3980, pp 364-372 Glasgow.
9. J.Soler, J.Poch, E.Barrabés, D.Juher and J.Ripoll. A tool for the continuous assessment and improvement of the student's skills in a mathematics course. Technology of Information and Communication in education for engineering and industry. Proceedings of the Symposium. TICE 2002, pp-105-110.
10. J.Soler, F.Prados, I. Boada and J.Poch. A web-based tool for teaching and learning SQL. Proceedings of IEEE International Conference on Information Technology based higher Education and Training, ITHET 2006 (to appear).