



NOCAT - a normalisation CASE tool

P. Pouyioutas

*School of Computing, University of North London,
2-16 Eden Grove, London N7 8EA, UK*

Abstract

In this paper we present the first version of the NOCAT Case Tool for Data Normalisation. A prototype version was presented in SEHE 1994 [1]. NOCAT was developed using the object-based language ObjectPal of Paradox for Windows [2]. Herein, we explain the main functionality of NOCAT and its expected use and usefulness. The NOCAT case tool provides support for teaching the database design technique of Data Normalisation. In this paper we explain how NOCAT provides tutorial support through the on-line tutorial and the various multiple-choice questions. We also explain how the interactive built-in and user-defined examples and exercises support the learning of this design technique. Finally, we illustrate how the "automatic normalisation" facility to be provided by the tool will enable students to check the answers to their examples and exercises.

1 Introduction

The development of self study packs, computer-based materials and generally open learning materials is very important for the School of Computing, the faculty and the University at large. The use of Information Technology in teaching and learning is therefore essential in the Teaching and Learning Strategy of the School [3]. The normalisation CASE tool NOCAT was developed in line with the said strategy. The availability of a computer-based Software Engineering tool will provide the students with the facility to work interactively, follow the tutorials, answer multiple choice questions and obtain feedback from the system.

The quality of the students' learning experience will be improved as the students will study at their own pace, will be able to obtain immediate feedback

which will help them to consolidate their knowledge and monitor their own progress. The tool will be evaluated by the lecturers involved in teaching database design techniques. The feedback given by the lecturers will be taken into consideration and changes may have to be made. The final version will be available for student use in the second semester of the 1995-1996 academic year.

2 The design and implementation of NOCAT

The author's background is in the design and implementation of structured menu-driven database applications rather than the design and implementation of hypertext systems. In these days there is an emerging demand and supply of hypertext and multimedia educational packages. Moreover, most other software packages provide on-line help in hypertext form. These hypertext systems allow access to a vast amount of information and enable users to locate and cross-reference the information required. This is done in a non-sequential and non-structured way and the author felt that such an approach would not be appropriate in providing support for learning the technique for Data Normalisation. For example, there is no point in learning Third Normal Form in data normalisation without having learnt Second Normal Form. Thus a more structured menu-driven interface was implemented to allow students a step-by-step learning experience. The menu structure of the tool is given in Figure 1 and each part of the tool is then explained. The tool mainly consists of three parts. The NORMALISATION TUTORIAL, the NORMALISATION PROCESS and the AUTOMATIC NORMALISATION.

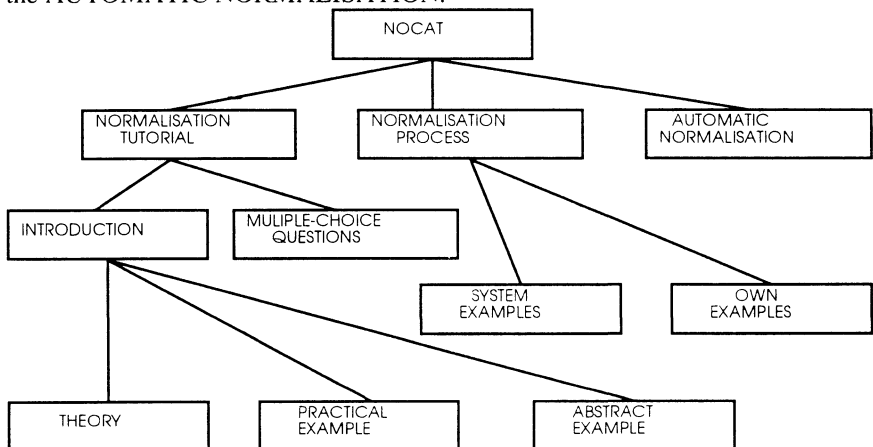


Figure 1

3 The NORMALISATION TUTORIAL

The NORMALISATION TUTORIAL (NT) introduces the students to the concepts of Data Normalisation and provides a number of multiple-choice questions to test students. NT is thus divided into two parts; INTRODUCTION and MULTIPLE-CHOICE QUESTIONS. INTRODUCTION is also divided into three parts; THEORY, PRACTICAL EXAMPLE and ABSTRACT EXAMPLE.

The THEORY part of NOCAT introduces the students to the basic concepts of Data Normalisation. We assume that the students are familiar with relational databases. We only explain the concepts of Functional Dependencies, Keys, Partial Functional Dependencies and Transitive Functional Dependencies. We also explain the steps of converting a non-normalised relation into a number of well structured Third Normal Form relations. We thus, explain how to go from non-normalised relations to First Normal Form (1NF) relations, Second Normal Form relations (2NF) and Third Normal Form (3NF) relations. The introduction is supported by a general theoretical example that illustrates all the aforesaid concepts (Figure 2). The PRACTICAL EXAMPLE part of NOCAT explains the concepts of Data Normalisation using a College database application. The Universal relation is given with some data. The disadvantages of data duplication are illustrated in terms of insertion/deletion/update of data. The said relation is then step-by-step decomposed into a set of 2NF and finally 3NF relations. The process of Data Normalisation is thus illustrated through the use of a real-life example and the elimination of the update anomalies problems are also demonstrated (Figure 3). The ABSTRACT EXAMPLE is used to illustrate the Data Normalisation process through a general abstract example. This example illustrates that this process is a formal process with very simple rules which when followed will guarantee a good database design. It is the author's personal view that if students understand the abstract example, they will be able to apply this process in real-life database applications.

The MULTIPLE-CHOICE QUESTIONS part of NOCAT provides a number of multiple-choice questions that test the students and provide appropriate feedback on their answers. The questions cover the theoretical concepts of Data Normalisation and test whether the students can apply this technique to various database examples. It is very important for the students not only to get feedback from the tool in terms of correct/wrong answer but also to get explanations why their answer is wrong or correct. Thus, NOCAT provides appropriate feedback to all answers (Figures 4,5).



TRANSITIVE FUNCTIONAL DEPENDENCY

Given a relation ONE(A,B,C,D,E) and the Functional Dependencies $A \rightarrow B, C, D, E$ and $D \rightarrow E$, we say that $A \rightarrow E$ is a Transitive Functional Dependency since it can be derived (by transitivity) from the fact that $A \rightarrow D$ and $D \rightarrow E$.

When converting 2NF relations to 3NF relations we eliminate such dependencies by creating for each such dependency a new relation containing all the attributes of the said dependency and eliminating the right hand side of each such dependency from the original relation. In this example, we create a relation TWO with attributes D and E and we eliminate attribute E from the relation ONE. Thus, we get

ONE(A,B,C,D) and
TWO(D,E)

EXIT NOCAT

EXIT THEORY

PREVIOUS

Figure 2

2NF relations still give rise to similar problems as the ones described for the 1NF relations. To illustrate these problems we consider the 2NF relations derived for this example with the sample data of this example.

UNIT-LECT

Uno	Uname	Lname	Lroom
ACC1	accounts	Jim	eg12
IS12	dbase1	Phil	ca104
IS13	dbase2	Phil	ce104
IS14	P	Tricia	ce205
WIK18	marketing	Ellie	eg14

STUDENT

Sno	Sname	Cno
s1	Tom	IS1
s4	Ann	ACC1

UR'

Sno	Uno	Mark
s1	IS12	62
s1	IS14	55
s4	ACC1	90
s4	IS13	45
s4	WIK18	61

1. Insertion Anomaly: a lecturer cannot be inserted (in the UNIT-LECT relation) unless there is a unit that he/she teaches

2. Deletion Anomaly: deleting the unit IS14 (in the UNIT-LECT relation) results in losing information about the lecturer Tricia and her room ce205

3. Modification Anomaly: changing room for lecturer Phil (in the UNIT-LECT relation) results in changes in more than one row

EXIT NOCAT

EXIT PRACTICAL

PREVIOUS

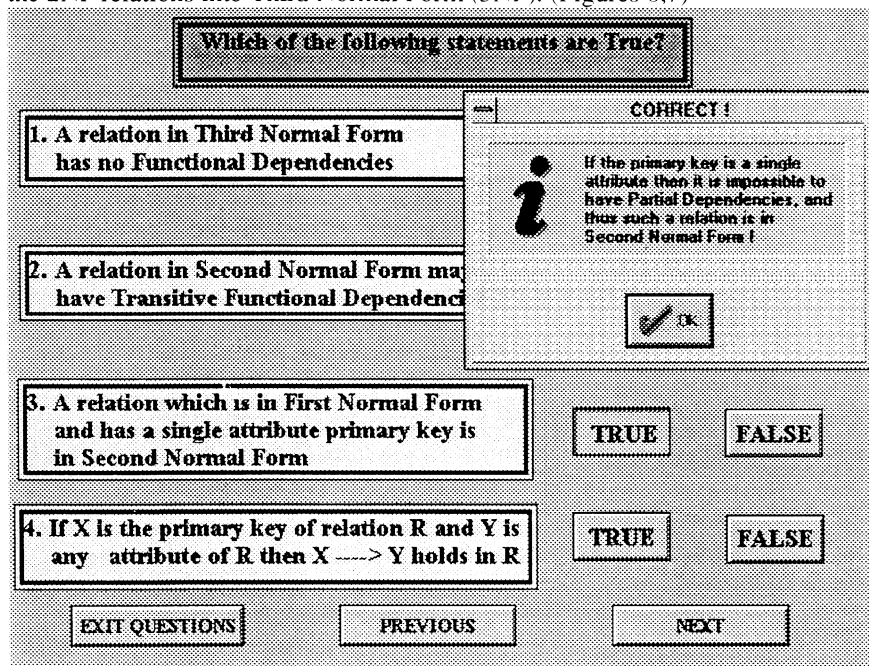
NEXT

Figure 3

4 The NORMALISATION PROCESS

The NORMALISATION PROCESS part of NOCAT provides students with interactive sessions that allow them to normalise existing system-provided examples or their own examples. Thus, this part consists of two parts: SYSTEM EXAMPLES and OWN EXAMPLES.

The SYSTEMS EXAMPLES allows students to apply the Data Normalisation process to existing system-provided examples. Students are given a description of a database application and the Universal relation of the application. They are then asked to identify all the Functional Dependencies of the application and the Primary Key with the system providing them with appropriate feedback on their answers. Once they have successfully completed this, they are asked to identify the Partial Functional Dependencies and split the Universal relation into a number of Second Normal Form (2NF) relations. We note again that throughout all these steps the system provides appropriate feedback to students. The last step is to identify the Transitive Functional Dependencies and convert the 2NF relations into Third Normal Form (3NF). (Figures 6,7)



Which of the following statements are True?

1. A relation in Third Normal Form has no Functional Dependencies
2. A relation in Second Normal Form may have Transitive Functional Dependencies
3. A relation which is in First Normal Form and has a single attribute primary key is in Second Normal Form
4. If X is the primary key of relation R and Y is any attribute of R then $X \twoheadrightarrow Y$ holds in R

CORRECT!

i If the primary key is a single attribute then it is impossible to have Partial Dependencies, and thus such a relation is in Second Normal Form!

☒ TRUE ☐ FALSE

☐ TRUE ☐ FALSE

☐ TRUE ☐ FALSE

☐ TRUE ☐ FALSE

Figure 4

In an orders database application, cutstomers place orders for parts. Each customer has a unique customer number (cno), a name (cname) and an address (caddress). A customer can place various orders. Each order is for one customer and has a unique order number (ono) and date (odate). Within an order a customer orders various parts with some quantity (qty). Each part has a unique part number (pno), a name (pname) and a price (pprice). The universal relation is given by

U (cno, cname, address, ono, odate, pno, pname, pprice, qty)

The primary key of the above relation (U) is

A. cno,ono,pno B. ono, cno
C. pno, cno D. ono,pno

WRONG !

This is actually a superkey ! Try Again!

The correct answer is: A B

Please insert the functional dependencies in the table below. If the LHS of a dependency consists of more than one attribute then list these separated by a SINGLE space. DO NOT leave any spaces in front or after the LHS and the RHS of the dependencies. Use the BEGIN EDIT button to start inserting and changing data and the END EDIT button to finish. To delete a functional dependency highlight the corresponding row in the table and click on the DELETE button. Once you finish click on the VERIFY DEPENDENCIES button to check whether your answer is correct. Then insert the primary key in the area provided in this screen. If the key is composite then list all attributes separated by a SINGLE space (NO spaces at start or end of the key). Click on the VERIFY KEY button to check your answer. When you have correctly identified all functional dependencies and primary key click on the 2ND NORMAL FORM button to convert the Universal relation into 2nd Normal Form.

LHS OF DEPENDENCY	
pno	pname
ono	odate

WRONG

You have missed some dependencies

Figures 5.6

The 2nd Normal Form of this example results in three tables. Please insert the appropriate attributes in the table, below. Then click on the **VERIFY TABLES** button to check your answer. If your answer is correct click on the **PROCEED 3NF** button to go to 3rd Normal Form. Just to remind you, the universal relation U is given by

$U (eno, ename, saddr, eval, dno, dname, pno, pname, pbudget, weekno, hours)$

the primary key is

"eno pno weekno"

and the partial functional dependencies are

$eno \rightarrow ename, saddr, eval, dno, dname$

$pno \rightarrow pname, pbudget$

no

TABLE 1

ATTRIBUTE

DELETE ATR

TABLE 2

ATTRIBUTE

DELETE ATR

THE ONLY CORRECT TABLES ARE THE RED ONES

☒ **OK**

DELETE ATR

BACK **BEGIN EDIT** **END EDIT** **VERIFY TABLES** **PROCEED**

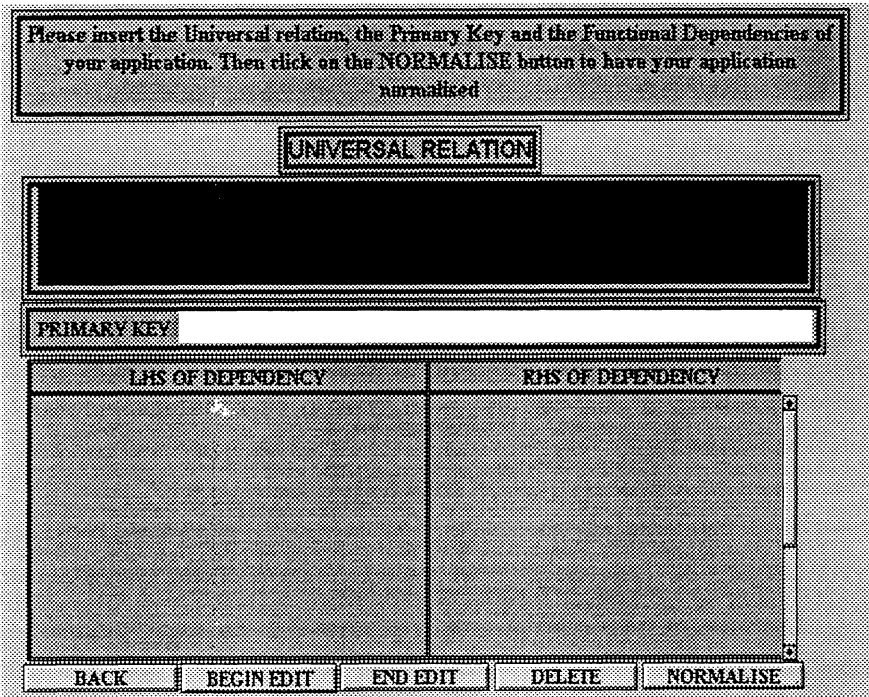
Figure 7

The OWN EXAMPLES part of NOCAT allows students to normalise their own examples. Students give the Universal relation, the Functional Dependencies and the Primary Key and then identify the Partial Functional Dependencies. The system then based on the provided information converts the Universal relation into a set of 2NF relations. The student then identifies for each 2NF relation the primary key and any Transitive Functional Dependencies. Finally, the system based on the new information converts all relations into 3NF.

5 The AUTOMATIC NORMALISATION

The normalisation process used in all the aforesaid parts of NOCAT is based on the *recognise and split* technique [4] whereby the dependencies of attributes are recognised and the logical groupings/relations are formed. In addition NOCAT will provide a facility to cross check the student's design. The *Bernstein algorithm* [5,6] will be implemented in the AUTOMATIC NORMALISATION

part of the tool. This part will provide an alternative design technique which requires only information about the Universal relation, all the Functional dependencies and the Primary Key (i.e. no information about Partial and Transitive Dependencies and thus avoids all the intermediate steps to 3NF). This part can be used as an automatic design checker for the *recognise and split* results. (Figure 8)



The screenshot shows a graphical user interface for a database normalization tool. At the top, a text box contains the instruction: "Please insert the Universal relation, the Primary Key and the Functional Dependencies of your application. Then click on the NORMALISE button to have your application normalised". Below this is a button labeled "UNIVERSAL RELATION". Underneath is a large, empty rectangular box for the universal relation. Below that is a text field labeled "PRIMARY KEY". The main part of the interface is a table with two columns: "LHS OF DEPENDENCY" and "RHS OF DEPENDENCY". The table is currently empty. At the bottom of the interface is a row of five buttons: "BACK", "BEGIN EDIT", "END EDIT", "DELETE", and "NORMALISE".

LHS OF DEPENDENCY	RHS OF DEPENDENCY
-------------------	-------------------

Figure 8

6 Conclusions and Future Plans

The development of NOCAT forms part of a larger activity for the creation of self study packs within the School of Computing. This particular application arose from the identification of the need to provide additional support to an increasingly large number of students. Another benefit of this approach will be the standardisation of methods via the adoption of the automated CASE tools which in turn will enhance the flexibility of the teaching staff and the effective use of the existing resources (laboratories and software). The tool will be

evaluated during the first semester of the 1995-1996 academic year by the lecturers teaching database design techniques and it will be used by students during the second semester of that academic year. Feedback will be given by both lecturers and students and the tool will be improved accordingly.

References

1. Pouyioutas, P. & Georgiadou, E. Teaching and Learning - CASE Tool for Data Normalisation, *First International Conference on Software Engineering in Higher Education*, pp. 191-197, Southampton, UK, 1994.
2. Borland *Paradox for Windows*, User Manuals, 1994.
3. University of North London Corporate Plan (92/93 -95/96), London, 1992.
4. Date, C.J. *Database Systems*, Sixth Edition. Addison-Wesley, 1994.
5. Bernstein, P.A Synthesizing Third Normal Form Relations From Functional Dependencies, *ACM Transactions on Database Systems Vol. 1, No. 4*, pp. 277-298, 1976.
6. Salzberg, B. Third Normal form Made Easy, *SIGMOD RECORD, Vol. 15, No.4*, pp. 2-17, 1986.