8/1 (2010), 121–132
tmcs@math.klte.hu
http://tmcs.math.klte.hu

Teaching Mathematics and Computer Science

# Normalization based on dependency diagram

Márta Czenky

*Abstract.* Normalization is an important database planning method, although the understanding and application of this method brings up the utmost problem during data modelling. That is why we were looking for alternative normalization methods, from which the normalization with dependency diagram proved to be the most efficient. This was also confirmed by the statistical estimation of the carried out survey.

*Key words and phrases:* data modelling, normalization, dependency diagram, survey.

*ZDM Subject Classification:* P20, Q50, Q60, R50.

## 1. Introduction

During the teaching of database management, the teaching of database planning method, the normalization raises the utmost problem. The method is a strict iterative planning method; its use requires strong abstraction. It demands from the students the conceptual understanding and practical application of functional dependencies and normal forms.

The functional dependency is the unambiguous definition of one of the table's columns by other columns of the table. In case of multivalued dependency the values of a table column define a set of values in another table column and these values are independent from the values of the other columns of the relation. The specifications of the normal forms are the followings:

Copyright © 2010 by University of Debrecen

| | |
|---|---|
| 1. NF | atomic value, primary key, no recurring group |
| 2. NF | no partial dependency |
| 3. NF | no transitive dependency |
| BCNF | all determinant attribute is super key |
| 4. NF | no multivalued dependency |

Figure 1 represents examples for functional dependencies which violate different normal forms. The multivalued dependencies are valid in the table remaining after taking out functional dependencies.

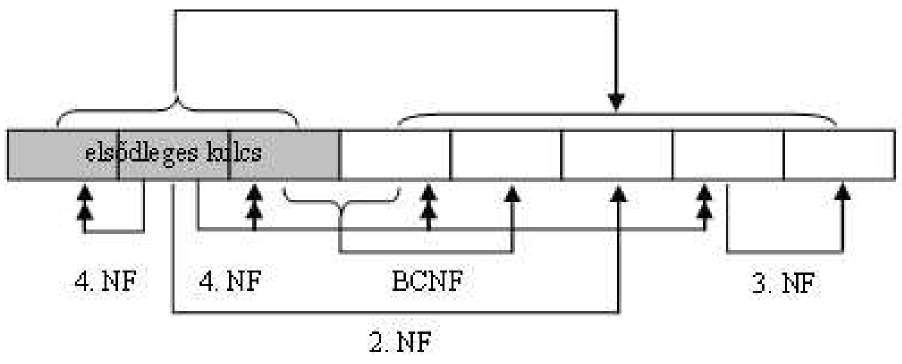


*Figure 1.* Types of functional dependencies and the unfulfilled normal forms

The essence of the method is to get tables that fulfil some of the higher normal forms (3. NF, BCNF or 4. NF) defined as goal, by the decomposition of the table based on the functional dependencies existing in the table and which violate one of the normal forms. The process of normalisation is presented in Figure 2.

The determination of the candidate key, the primary key and the functional dependencies, the identification of which normal form is violated by the functional dependencies, the strict adherence to the steps of planning and the consistent application of the table decomposition algorithm causes problems to the students.

In a survey we asked students to define what causes them problems in the sphere of data modelling. Table 1 shows the summary of the received answers to the activities connected to data modelling [1].

The entity-relationship modelling causes problem for 15–19% of the students. The teaching of normalization is unavoidable that is why we were looking for alternative solutions.

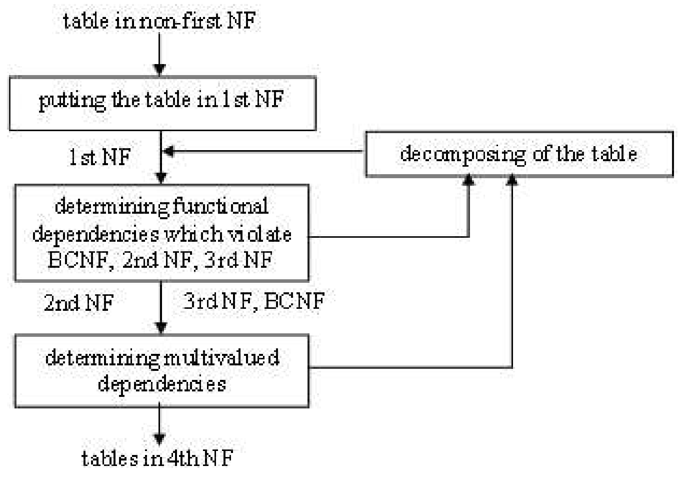


table in non-first NF
putting the table in 1st NF
1st NF
decomposing of the table
determining functional dependencies which violate BCNF, 2nd NF, 3rd NF
2nd NF
3rd NF, BCNF
determining multivalued dependencies
tables in 4th NF

*Figure 2.* The process of the normalization

*Table 1.* Result of the first survey

| **Task** | **Ratio of those students who raised problems** |
|---|---|
| Defining the table in 0th normal form | 41, 3% |
| Defining the primary key | 39, 1% |
| Recognition of functional dependencies | 28, 3% |
| Recognition of multivalued dependencies | 19, 6% |
| Understanding of normal forms, recognition of in which normal form the table is | 69, 6% |
| Decomposition rule of the tables | 26, 1% |
| Drawing the dependency diagram | 30, 4% |

It is common to represent the functional dependencies on dependency diagrams. Based on this diagram the decomposition of the tables can be carried out, according to our experience students understand and apply it better than the traditional method. Furthermore we introduced a third method, namely the Cookbook method [6] to the students, although this method – despite of its simplicity – did not result better solutions than the application of the traditional one.

## 2. Drawing of dependency diagrams

It is common to illustrate the functional dependencies on diagrams where the attributes are arranged horizontally and the arrows are placed above and below the attributes. If more attributes are determinant or dependent then we connect them with lines or arrows. The dependency diagram of Figure 1 or the diagrams showed by [5] are also arranged this way. The disadvantage of the horizontal arrangement is that due to lack of space it can happen that the attributes have to be illustrated in several lines, which breaks the continuous representation of the arrows.

The other way of arrangement is, when we represent vertically, under each other the attributes that are connected to the primary key and to the functional dependency's dependent attributes. A dependency diagram like this has to be drawn based on the rules below [7], [3], [8]:

- We include the attributes in boxes, arrow goes from the determinant attribute to the dependent attribute.
- The graphic illustration of A → B functional dependency is: $\boxed{\text{A}} \rightarrow \boxed{\text{B}}$
- The graphic illustration of C → D multivalued dependency is: $\boxed{\text{C}} \twoheadrightarrow \boxed{\text{D}}$
- Each attribute appears only once in the diagram.
- If on the left side of the functional dependency a set of irreducible attributes stand, than we include these attributes in one box, the arrow representing the dependency starts from the box.
- One arrow leads to one attribute. This means that based on the Armstrong decomposition rule we decompose those functional dependencies, which have several attributes on their right side to several functional dependencies, which have only one attribute on the right side.
- If between the attributes of A, B, C of one of the relations of $A \rightarrow B$, $B \rightarrow C$ functional dependencies exist then because of the transitivity the $A \rightarrow C$ functional dependency also exists, however in the diagram the latter one is not shown.
- We do not represent the functional dependency from alternative keys on the diagram.

## 3. Definition of the closure of a set of attributes by dependency diagram

See the following relation:

$$\mathrm{R(A, B, C, D, E, F, G, H, I, J, K)}$$

In the relation the following functional and multivalued dependencies exist:

$$\begin{array}{ll}
\mathrm{A \to B, C, D, E} & (1)\\
\mathrm{J \to F, G} & (2)\\
\mathrm{A, F, H \to I} & (3)\\
\mathrm{H, J \to K} & (4)\\
\mathrm{F \to G} & (5)\\
\mathrm{D \to E} & (6)\\
\mathrm{H \to\to A} & (7)\\
\mathrm{H \to\to J} & (8)
\end{array}$$

In the table A, H, and J are candidate keys, because $\{\mathrm{A, H, J}\}^+$ closure on the given dependency set includes every attribute of the relation.

$\{\mathrm{A, H, J}\}^+_1 = \{\mathrm{A, H, J}\}$ due to reflexivity

$\{\mathrm{A, H, J}\}^+_2 = \{\mathrm{A, H, J, B, C, D, E}\}$ due to $\mathrm{A \to B, C, D, E}$ dependency

$\{\mathrm{A, H, J}\}^+_3 = \{\mathrm{A, H, J, B, C, D, E, F, G}\}$ due to $\mathrm{J \to F, G}$ dependency

$\{\mathrm{A, H, J}\}^+_4 = \{\mathrm{A, H, J, B, C, D, E, F, G, I}\}$ due to $\mathrm{A, F, H \to I}$ dependency

$\{\mathrm{A, H, J}\}^+_5 = \{\mathrm{A, H, J, B, C, D, E, F, G, I, K}\}$ due to $\mathrm{H, J \to K}$ dependency

The remaining functional dependencies do not enlarge the closure set with additional attributes. The $\{\mathrm{A, H, J}\}$ set of attributes is candidate key, because it is true that none of the closures of its real subsets includes all attributes of the relation.

The creation of the set of attribute's closure can happen with the help of drawing of the dependency diagram. We draw the candidate key attributes. In each following step we enlarge the diagram by one more column for the dependent attributes of the functional dependencies of which the determinant attributes are already included in the diagram. On Figure 3 we marked with continuous line the functional dependencies which enlarge the closure set by new attributes and with dotted-broken line those which do not. If the diagram includes all attributes of the relation, the left outermost column or column-combination forms a candidate key.

Another process can also be applied, for example we can draw the dependency diagram then we colour each attribute which is included in the closure set. In details, at first because of the reflexivity we colour the candidate keys. Afterwards we colour the attributes that are on the right side of a functional dependency where the determinant attributes are already coloured. At the end of the process those attributes will be coloured, which form the closure [7].

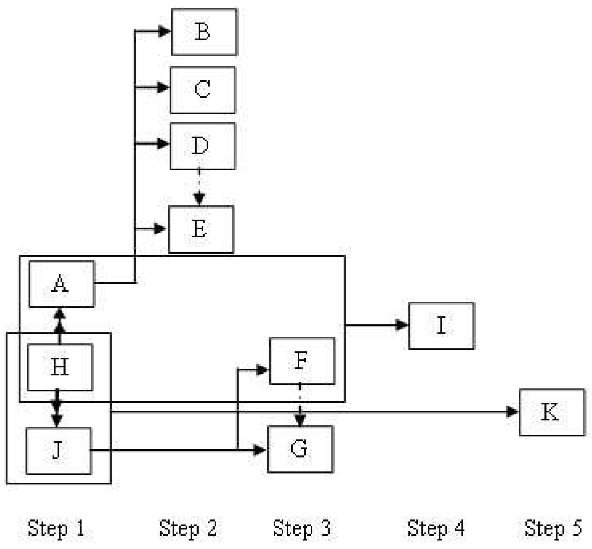


*Figure 3.* Definition of the set of attribute's closure

## 4. Normalization based on dependency diagram

The process of normalization (see Figure 2) does not change even if we do it based on a dependency diagram. We take out the functional dependencies which violate the normal forms into an independent table and the determinant attribute will be the primary key in the new tables. The dependent attributes have to be deleted from the diagram, what we indicate by crossing the attributes, so the diagram shows always how the original relation changes. The determinant attributes remain on the diagram as foreign keys.

In relation R the $\{A, H, J\}$ set of attributes will be the primary key. From the functional dependencies shown in Figure 3 the functional dependencies (1),

(2) and (4) violate the 2nd normal form, the functional dependency (3) violates the Boyce–Codd normal form and the functional dependencies (5) and (6) violate the 3rd normal form. If we follow the usual steps in normalization, which means that we take out first the functional dependencies that violate the 2nd normal form, than we get to the following tables:

T1 (**A**, B, C, D, E)
T2 (**J**, F, G)
T3 (**H, J**, K)

And the relation R is:

R(***A***, ***H***, I, ***J***)

In the tables we indicated the primary keys in bold, the foreign keys in italics. The functional dependency (3) is not included in any tables so we can not continue the normalization this way.

Consequently we have to start the normalization by taking out the functional dependency (3) which violates the Boyce–Codd normal form. It is easy to see that the 2nd NF-3rd NF take out order can be inverted to 3rd NF-2nd NF takeout order, the result will be the same in both cases. Therefore we continue the taking out with the 3rd NF-2nd NF-4th NF order and we get to the following data model. T6 and R relations can be contracted.

T1(***A***, ***F***, ***H***, I)
T2 (**D**, E)
T3 (**F**, G)
T4 (**A**, B, C, *D*)
T5 (**J**, *F*)
T6 (**H,** ***J***, K)
T7 (***A***, ***H***)
R(***H***, ***J***)

We have drawn the dependency diagram in vain as it did not help in simplifying the normalization since attention also had to be paid to the sequence of taking-outs. This means an extra task to students. To eliminate it we’ve drawn the dependency diagram in another way: to the left column we draw not only the attributes of the primary key, but the attributes of the super key too that includes the determinant attributes of the functional dependency that violates the Boyce–Codd normal form. (see Figure 4 [2])

The taking-outs can be classified in two groups, first we take out functional dependencies, where the dependent attributes do not belong to the super key, in this case we do not have to pay attention to the sequence it can be for example

3rd NF-2nd NF-BCNF, then we take out the dependencies that exist between the elements of the super key.

T1 (**D**, E)
T2 (**F**, G)
T3 (**A**, B, C, *D*)
T4 (**H, J**, K)
T5 (***A***, ***F***, ***H***, I)
T6 (**J**, *F*)
T7 (***A***, ***H***)
R(***H***, ***J***)

T4 and R relations of the data model can be contracted.

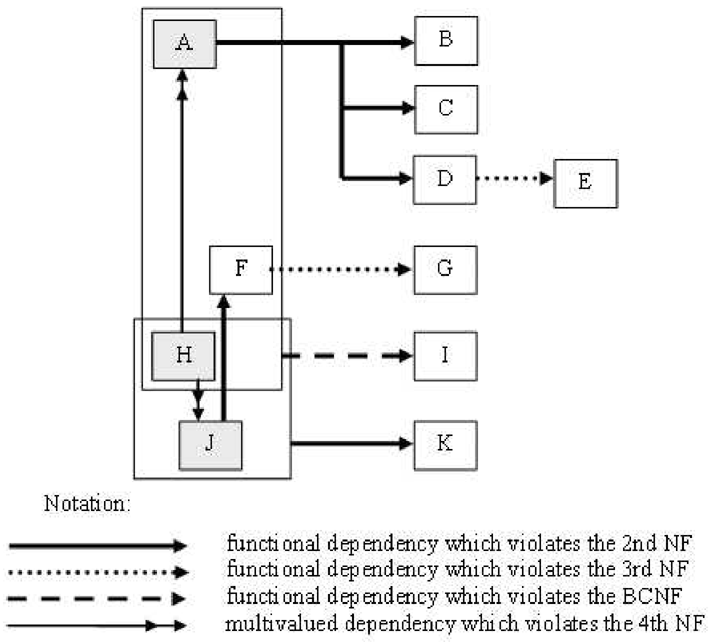
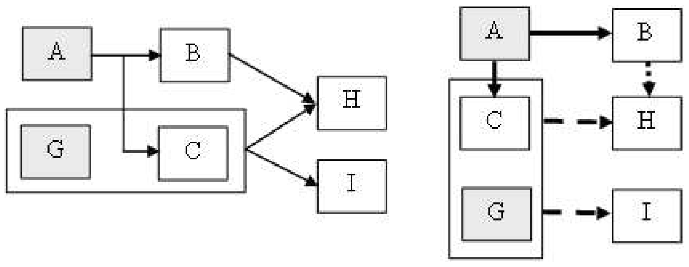

A
B
C
D
E
F
G
H
I
J
K
Notation:
functional dependency which violates the 2nd NF
functional dependency which violates the 3rd NF
functional dependency which violates the BCNF
multivalued dependency which violates the 4th NF

*Figure 4.* Modified dependency diagram

Our representation, that we draw on the left side the super key of all functional dependencies which includes the determinant attributes and the modified decomposition method does not always result such data model that keeps all functional dependencies. We show as an example two dependency diagrams, see

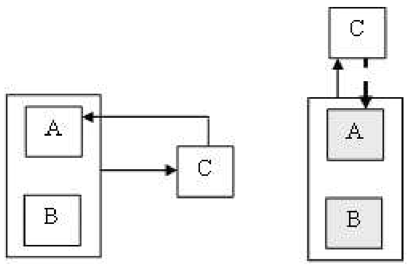
Figure 5 and 6. On Figure 5 since attribute H is dependent attribute in two functional dependencies, no decomposition exists that includes each dependencies of B $\rightarrow$ H and $C, G \rightarrow$ H.



a) original diagram b) modified diagram

*Figure 5.* First example for a diagram that results a decomposition which does not keep the dependency

The diagram b) of Figure 6 does not prove, that after decomposition the relations include all C $\rightarrow$ A and A, B $\rightarrow$ C functional dependencies.



a) original diagram b) modified diagram

*Figure 6.* Second example for a diagram that results a decomposition that does not keep the dependency

## 5. Survey

In the first semester of the school year 2008/2009 we made a survey among students to find out which normalization method helps them more efficiently in modelling.

They had to carry out the steps of the traditional method in the Moodle course management system by solving an electronic test. The interpretation of dependency diagrams, the normalization and the drawing of diagrams were tested in writing. The results of the survey are summarized in Table 2.

*Table 2.* Result of the second survey

| **Task** | **Ratio of correct solutions** | |
|---|---|---|
| | **Traditional method** | **Diagram based method** |
| Determination of the primary key | 30% | |
| Determination of the functional dependencies | 50% | |
| Recognition of the functional dependencies that violate the 2nd NF | 60% | 100% |
| Recognition of the functional dependencies that violate the 3rd NF | 30% | 70% |
| The given table is in which normal form | 30% | |
| 2nd NF decomposition | 40% | 90% |
| 3rd NF decomposition | | 40% |
| Creation of the table that remains after decomposition | | 30% |
| Marking of the primary and foreign key | 40% | 50% |

In interpretation of the dependency diagrams we asked which normal forms are violated by the represented functional dependencies, whether transitive dependency exists between two attributes and into how many tables could the diagram be decomposed. In normalization the data model had to be created following the correct decomposition sequence, also indicating the primary and foreign keys. In diagram drawing it was not an exercise to determine the primary key and functional dependencies in the table, we have given them in advance, the students had to draw only the diagram. In case of both types of exercises the students had to work with functional dependencies that violate the 2nd and 3rd normal forms.

It is visible on Table 2 that the normalization based on dependency diagram resulted better solutions. The three problematic areas are: recognition of functional dependencies that violate the 3rd normal form, putting the table that remains after the taking-outs into the data model and the correct creation of the foreign keys.

We carried out an independency test to decide if the result is significantly better. Our null hypothesis was [4]:

H0: *the data series are independent, the result does not depend from the applied solution method*

The data of Table 3 shows that, although the number of elements of the sample was low, we can state with 95% probability that the null hypothesis is not true, the solutions were significantly better when students carried out the normalization based on dependency diagrams.

*Table 3.* Results of the independency test

| | |
|---|---|
| calculated probability | 0.04692402 |
| $\chi^2$ | 11.23448511 |
| degree of freedom | 5 |
| significance level | 0.05 |
| critical value | 11.07049775 |

## 6. Summary

Students understand the normalization which requires strong abstraction better, if we present the dependency relations between tables graphically on dependency diagrams and the decomposition of the tables based on a diagram. It helps the process of decomposition if we use the modified dependency diagram which does not require the analysis of the sequence of taking-out.

The carried out independency test shows, that the solutions of the students are significantly better when applying the presented process. To reinforce the result the survey should be carried out in the following years as well.

## References

[1] M. Czenky, *Adatbázis-kezelés oktatás hallgatói szemmel*, Informatika a felsőoktatásban 2008 Konferencia, Debrecen.

[2] M. Czenky, *Adatmodellezés, SQL és Access alkalmazás, SQL Server és ADO*, ComputerBooks Kiadó, Budapest, 2005, 445.

[3] C. J. Date, *Introduction to Database Systems*, Vol. I.–II., Addison-Wesley Publishing Company, New York, 1990, 854.

[4] I. Falus and J. Ollé, *Az empirikus kutatások gyakorlata*, Nemzeti Tankönyvkiadó, Budapest, 2008, 341.

[5] D. Kennedy, *Database Design and the Reality of Normalization*, 13th Annual NACCQ Conference, 2000,
`http://www.in-site.co.nz/misc_links/papers/kennedy167.pdf`.

[6] H. J. Kung and H. L. Tung, An alternative approach to teaching database normalization: A simple algorithm and an interactive e-learning tool, *Journal of Information Systems Education* **17**, no. 3, 315–326.

[7] R. K. Moniot, *Data Base Systems, Functional Dependency Diagrams*,
`http://www.dsm.fordham.edu/ moniot/Classes/DatabaseSp04/FD-diagrams.html`.

[8] H. C. Smith:, Database design: composing fully normalized tables from a rigorous dependency diagram, *Communications of the ACM* **28**, no. 8 (1985), 826–838.

MÁRTA CZENKY
SZENT ISTVÁN UNIVERSITY
FACULTY OF MECHANICAL ENGINEERING
DEPARTMENT OF INFORMATICS
1084 BUDAPEST, DÉRI 15 3/5

*E-mail:* `marta.czenky@t-online.hu`

*(Received July, 2009)*