STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LVII**, Number 1, 2012

SPECIAL ATTRIBUTES FOR DATABASE NORMAL FORMS DETERMINATION

VITALIE COTELEA

Abstract. The article deals with the relational schemes defined on the reduced sets of functional dependencies divided into equivalence classes. The concepts of non-essential and recoverable attributes are introduced, and the algorithms for their computing are proposed. Some properties of schemes, based on these attributes, are presented as well.
Based on non-essential and recoverable attributes some conditions are introduced to make a relation be in the third or Boyce-Codd normal form. These conditions are just sufficient, because there could be a scheme that is in third or Boyce-Codd normal form but doesn't satisfy them. In addition, the article suggests a normalization algorithm that takes into account these conditions.

1. Introduction

A category of problems that may arise in the development of applications using a database is the incorrect designed relational schemes. Testing the correctness of a scheme can be made using functional (or other) dependencies attached to that scheme.

Many decision-making problems related to functional dependence schemes are difficult to calculate. Such issues include determining if an attribute is primary and testing if a scheme is in a certain normal form. Algorithms for these problems are required in design tools for databases. But these problems can be solved only by algorithms of exponential complexity for the time being. Even for schemes with a relatively small number of attributes, such algorithms can not be used for all design tasks.

Testing if a relational scheme is in Boyce-Codd normal form is an easy problem. In fact, it must be tested for all dependencies defined on the scheme, if their left sides are super keys [11]. This clearly is made in polynomial time,

Received by the editors: January 10, 2012.

2010 *Mathematics Subject Classification.* 68P05, 68P15.

1998 *CR Categories and Descriptors.* H.2.1 [**Database Management**]: Logical Design – *Normal forms* .

Key words and phrases. Database, Logical design, Non-essential attributes, Recoverable attributes, Third normal form, Boyce-Codd normal form.

in contrast to testing if the scheme is in third normal form, which is a NP-complete task, as testing the primeness of attributes is NP-complete [8].

It should be mentioned that the first statement is correct, only if the given set of dependencies is an exhaustive one. In case it does not represent the closure of the set of dependencies, the problem is of an exponential character. Although the Boyce-Codd normal form testing if the above condition is satisfied, is executed in polynomial time, to detect whether a subscheme of a scheme is in this form, is an NP-complete problem [2]. The reason of the increasing complexity lies in the following: For the Boyce-Codd problem the relational scheme is given. In other words, the set of functional dependencies is part of the input. And the only remaining thing is to test if the left side of each dependency is a superkey. But in the case of a subscheme, the set of dependencies is not known explicitly.

Worland [13] presents an algorithm that determines whether a scheme is in third normal form. The algorithm works, classifying scheme attributes in so-called dependent sets, which are based on the set of functional dependencies defined on the given scheme. To visualize the dependencies, a new type of dependency graph is introduced. The algorithm works faster than the designed algorithms for finding all candidate keys of the scheme, especially if there is more than one dependent set.

Obviously the question arises if recognizing the schema's normal form has a pre-requisite to determine the prime and nonprime attributes or finding of all keys. A solution would be to determine the equivalent characteristics of these entities, in such a way that it could be calculated in polynomial time.

This article introduces the notion of two types of attributes: essential and recoverable and it gives some of their properties that can be used to analyze database schemas. The algorithms for determining the essential and recoverable attributes are likely polynomial.

2. Preliminary notions

This section presents some known concepts from where we will proceed to describe the subject of this paper.

Let $Sch = (R, F)$ be a database schema, where F is a set of functional dependencies over a set R of attributes. The set F of functional dependencies may have a diverse structure. First, it can be a nonredundant set, secondly, it may be a reduced set, divided into equivalence classes or may be a minimum set [9, pp.71-85].

Two sets of functional dependencies F and G are equivalent (given as $F \equiv G$), if and only if $F^+ = G^+$, that is, if the closures of these sets are equal.

A set F of functional dependencies is nonredundant [12], if $\nexists G$, so that $G \subset F$ and $G \equiv F$.

Consider marking by $|F|$ and $||F||$ the cardinality of F and the number of attributes involved by F respectively (including the repeating ones).

The set F of functional dependencies is a minimum set [10], if $\nexists G$, such that $G \equiv F$ and $|G| < |F|$.

Let F be a set of functional dependencies over a scheme R and $X \to Y \in F$. The attribute A is extraneous in dependency $X \to Y$ with respect to F, if

$$A \in X \text{ and } F - \{X \to Y\} \cup \{(X - \{A\}) \to Y\} \equiv F$$

or

$$A \in Y \text{ and } F - \{X \to Y\} \cup \{X \to (Y - \{A\})\} \equiv F.$$

A set F of functional dependencies is left-reduced (right-reduced), if every functional dependency in F has no redundant attribute on the left (right) side. A left and right-reduced set of functional dependencies is a reduced set.

Let $X \to Y \in F$. It is defined as the equivalence class of functional dependencies which includes $X \to Y$ the set of dependencies, written $E_F(X)$:

$$E_F(X) = \{V \to W | V \to W \in F \wedge X \leftrightarrow V\}$$

So, $E_F(X)$ is the set of functional dependencies in F with left sides equivalent to X with respect to F.

The set of attributes $K \subseteq R_i$ is a key of a scheme $Sch_i = (R_i, F)$, if it satisfies the following conditions:

(1) $K \to R_i \in F^+$
(2) $\forall K' \subset K,\ K' \to R_i \notin F^+$.

The set $K \subseteq R_i$ of attributes which satisfy the condition (1) is called superkey.

The attribute A in R_i is prime with respect to F, if A is a part of key of Sch_i, otherwise A is nonprime in R_i.

Let $Sch_i = (R_i, F)$ be a scheme where V and W are nonempty subsets of R_i, and A is an attribute in R_i. Attribute A is transitively dependent on V via W if all the following conditions are satisfied:

(1) $V \to W \in F^+$,
(2) $W \to V \notin F^+$(i.e. V is not functionally dependent on W),
(3) $W \to A \in F^+$, and
(4) $A \notin VW$ (where VW is the union of V and W).

A relation scheme $Sch_i = (R_i, F)$ is in third normal form [4] with respect to a set F of functional dependencies, if it is in first normal form and there is no nonprime attribute in R_i that is transitively dependent upon a key of Sch_i scheme. The database scheme Sch is in third normal form with respect to F

if every relation scheme Sch_i in Sch is in third normal form with respect to F.

A relation scheme $Sch_i = (R_i, F)$ is in Boyce-Codd normal form [5] with respect to a set F of functional dependencies, if it is in first normal form and for every nontrivial dependency $V \to A \in F^+$ the dependency $V \to R_i \in F^+$ takes place. That is, the left side of every nontrivial functional dependency is a superkey for Sch_i. A database scheme Sch is in Boyce-Codd normal form with respect to F if every relation scheme Sch_i in Sch is in Boyce-Codd normal form with respect to F.

Let $X \to Y \in F^+$ and $< X_0, X_1, ..., X_n >$ be the maximal derivation [6] of the set X under F. Let X_i be the first element which contains the set Y. Then the subsequence $< X_0, X_1, ..., X_i >$ is considered to be the *derivation* (not necessarily the maximal one) of the functional dependency $X \to Y$ under F.

The expression $X \to Y \in F^+$ takes place if and only if the derivation of $X \to Y$ under F exists.

3. Non-essential and recoverable attributes

In this section, we introduce the notion of two types of attributes: non-essential and recoverable, some of their properties are given which can be used to analyze database schemas. It is shown that the algorithms for determining the non-essential and recoverable attributes have a polynomial complexity.

Further, it is assumed that the set F of functional dependencies is reduced and divided into equivalence classes $F = F_1 \cup ... \cup F_n$.

It will be denoted by R_i the set of attributes of dependencies involved in class F_i, and by $PS(F_i)$ the set of left sides of dependencies F_i, and $F - F_i(C)$ will denote by the expression $(F - F_i) \cup \{X \to (Y - \{C\}) | X \to Y \in F_i\}$, for $i = \overline{1, n}$. Then, the non-essential attribute is defined as follows:

Definition 1. *Let F_i be an equivalence class, where $|F_i| > 1$ and $X \to YC \in F_i$. The attribute C is non-essential in the scheme $Sch_i = (R_i, F)$, if for every two left sides V and Z, where $V, Z \in PS(F_i)$, $V \to Z \in (F - F_i(C))^+$ takes place.*

It is not difficult to see that the non-essential attribute C is in the right side of only one dependency of equivalence class F_i. Otherwise, the set F would not be a reduced set of functional dependencies.

It should be mentioned that if C_1 and C_2 are non-essential attributes in the scheme Sch_i, then their union $C_1 \cup C_2$ is not necessarily non-essential.

Example 1. *Let $F = F_1 \cup F_2 \cup F_3$ be a given set of functional dependencies divided into three equivalence classes $F_1 = \{C_1 \to D\}$, $F_2 = \{C_2 \to D\}$ and*

$F_3 = \{AD \to B,\ AB \to C_1C_2\}$. It can be verified that both C_1 and C_2 are non-essential attributes in the scheme $Sch_3 = (\{A, B, C_1, C_2, D\},\ F)$, but their union $C_1 \cup C_2$ is not non-essential, because $AB \to AD \notin (F - F_3(C_1C_2))^+$.

Here and below, F is written in scheme Sch_i because the introduction of the set F_i would not be correct, due the set F_i^+ is a subset of a set of functional dependencies defined over the set $R_i$of attributes. In this context and considering the hypothesis of universal schema [7], the set $F$is presented in Sch_i, but it is meant the set of dependencies in F^+ satisfied by relations defined over the set R_i of attributes.

Definition 2. *The attribute A, where $A \in R_i$, is recoverable in $Sch_i = (R_i, F)$, if $(R_i - A) \to A \in (F - F_i)^+$ takes place.*

Unfortunately, likewise non-essential attributes, the union of recoverable attributes is not always recoverable.

From the definition of non-essential attribute, it appears a remarkable property of it: within the class of equivalence the free navigation of non-essential attributes on the right sides of the dependencies it is allowed. However, the closure of the set of dependencies remains intact. But, in the new set, the dependencies can become non-reduced.

Example 2. *Let $F = F_1 \cup F_2$ be a set of functional dependencies divided in two equivalence classes $F_1 = \{C \to B\}$ and $F_2 = \{AD \to B,\ AB \to DC\}$ of dependencies. The attribute C is non-essential in the scheme $Sch_2 = (\{A, B, C, D\},\ F)$, because $AB \leftrightarrow AD$ under $F - F_i(C)$. The set F is equivalent to G, where $G = G_1 \cup G_2$ and $G_1 = F_1$, but $G_2 = \{AD \to BC,\ AB \to D\}$. However, the dependency $AD \to BC$ is not reduced, because its substitution with $AD \to C$ does not affect the set G^+. It is easy to verify that the removed attribute B is recoverable in Sch_2.*

Therefore, the moving of non-essential attributes can be used to obtain an equivalent set of dependencies, but with fewer attributive symbols.

The algorithm for calculation of non-essential attributes is based on the following theorem:

Theorem 1. *Let $F = F_1 \cup ... \cup F_n$ be a minimum and reduced set of functional dependencies, and let $f : X \to YC \in F_i$ be a dependency. The attribute C is non-essential in Sch_i, if and only if for every left side $Z \in PS(F_i)$ the following $X \to Z \in (F - F_i(C))^+$ holds.*

Proof. (Necessity) The veracity of this statement follows directly from the definition of non-essential attribute. □

Proof. (Sufficiency) For the reverse statement is enough to show that for every left side $Z \in PS(F_i)$ the expression $Z \to X \in (F - f)^+$ takes place.

Indeed, the fact that $Z, X \in PS(F_i)$, shows that $Z \to X \in F^+$. But for the dependency f to be non-redundantly used in the derivation of dependence $Z \to X$ under $F - f$ it is necessary $Z \to X \in (F - f)^+$ to take place. □

The algorithm to calculate the non-essential attributes:

ATR_NEES(F,ClasEchivDep)
    $MatrAtrNees := 0$;
    for each $f : X \to Y \in F$ *do;*
        $i := ClasEchivDep(f)$;
        for each $C \in Y$ *do;*
            $m := 0$; $j := 0$;$X_j := X$;
            repeat
                $j := j + 1$;$X_j := X_{j-1}$;
                for each $V \to W \in (F - F_i(C))$ *do;*
                    if $V \subseteq X_j$ *then* $X_j := X_j \cup W$;
                    if $V \in PS(F_i)$ *then* $m := m + 1$;
                endfor;
            until $X_j = X_{j-1}$;
            if $|F_i| = m$ *then* $MatrAtrNees(i, C) := 1$;
        endfor;
    endfor;
end ATR_NEES.

In the described algorithm, $ClasEchivDep$ is an array showing the equivalence class of each dependency in F, and $MatrAtrNees$ is a two-dimensional array where for each class non-essential attributes are fixed. The closure of the set X of attributes under the set of dependencies $F - F_i(C)$ is calculated (*repeat* loop) in time $O(||F||)$ [2]. In the same loop it is determined whether the conditions of Theorem 1 are satisfied. Therefore, the *ATR_NEES* procedure takes $O(||F||^2)$ time.

It is not difficult to note that the algorithm for calculating the recoverable attributes has a temporal complexity similar to the procedure for determining non-essential attributes.

These constructions and statements will be used to determine the degree of normalization of the database schema. It is known that the problem of determining the degree of normalization is of exponential nature. First, the definitions of normal forms (second, third and Boyce-Codd normal forms) contain the key notion. But it is known that a relation can have an exponential

number of keys according to the number of attributes of the scheme. Second, the definitions of normal forms appeal to notions of prime and nonprime attributes, the concepts related again to the notion of key.

4. Sufficient conditions for normalization

In this section, it is shown that the sets of non-essential and recoverable attributes of a relational scheme are disjoint. In addition, it is demonstrated that any attribute that transitively depends on a key of scheme is recoverable in this scheme. New features in terms of non-essential and recoverable attributes for the schemes in third normal form are presented.

The main result of this section is to present and demonstrate sufficient conditions for a relational scheme to be in Boyce-Codd normal form. The proposed algorithm for testing whether a scheme is in Boyce-Codd normal form is of polynomial complexity.

Let $Sch_i = (R_i, F)$ be a relation scheme, and S and T are the sets of non-essential and recoverable attributes, respectively. Then the following assertion can be formulated:

Lemma 1. $S \bigcap T = \emptyset$.

Proof. It is supposed the opposite: $S \bigcap T \neq \emptyset$, that is, there is in the set R_i of attributes an attribute C, which is both non-essential and recoverable in the Sch_i. Let the attribute C be found in the right side of dependence $X \to YC$ of equivalence class F_i. Taking into account the non-essential attribute definition, the expression $X \to Z \in (F - F_i(C))^+$ holds for every left side Z of the set $PS(F_i)$ of left sides. Therefore, $X \to (R_i - C) \in (F - F_i(C))^+$. From the definition of recoverable attribute it follows that $(R_i - C) \to C \in (F - F_i)^+$ and from the fact that $(F - F_i) \subseteq (F - F_i(C))$, it occurs that $(R_i - C) \to C \in (F - F_i(C))^+$. Of $X \to (R_i - C) \in (F - F_i(C))^+$ and $(R_i - C) \to C \in (F - F_i(C))^+$, it follows that $X \to C \in (F - F_i(C))^+$. But this contradicts the assumption that the dependency $X \to YC$ is reduced. □

Lemma 2. *If an attribute A in R_i transitively depends on a key of the scheme $Sch_i = (R_i, F)$, then, A is recovered Sch_i.*

Proof. Because A transitively depends on a key of the scheme $Sch_i = (R_i, F)$, let it be X, then, there are a set $V \subseteq R_i$ of attributes, such that $X \to V \in F^+$, $V \to A \in F^+$, $V \to X \notin F^+$ and $A \notin XV$. Then there is the derivation $H =< V_0, ..., V_m >$ for dependency $V \to A$ under F. Since $V \to X \notin F^+$, then $X \subseteq V_m$ and the left side of each dependency used in the construction of H is not equivalent to X. Thus, $V \to A \in (F - F_i)^+$. But $V \subseteq (R_i - A)$ and hence $(R_i - A) \to A \in (F - F_i)^+$. □

Theorem 2. *Let $Sch_i = (R_i, F)$ be a relation scheme. Then*

a) if every nonprime attribute is non-essential, the scheme Sch_i is in third normal form.

b) if every prime attribute is recoverable, the scheme Sch_i is in third normal form.

Proof. Veracity of statements a) and b) arise from the definition of scheme in third normal form and lemmas 1 and 2. □

The statements given in Theorem 2 contain the concepts of prime and nonprime attributes. But, as mentioned in [1], the problem of determining whether the attributes are prime or not is an NP-complete one.

In terms of applicability, the assertion set out by the next theorem is more acceptable:

Theorem 3. *If every attribute A in R_i is not recoverable in scheme $Sch_i = (R_i, F)$, then $Sch_i = (R_i, F)$ is in Boyce-Codd normal form.*

Proof. Let the scheme $Sch_i = (R_i, F)$ not be in Boyce-Codd normal form. That is, there is at least one functional dependency $V \to A \in F^+$, where $VA \subseteq R_i$, $A \notin V$ and V is not a superkey for $Sch_i = (R_i, F)$. But in this case A transitively depends on a key of the scheme Sch_i and according to Lemma 2 A is recoverable. We have obtained a contradiction. □

Corollary 1. *If every attribute A in R_i that does not belong to any left side of dependencies in equivalence class F_i, is not recoverable in $Sch_i = (R_i, F)$, then the scheme $Sch_i = (R_i, F)$ is in third normal form.*

It should be mentioned that conditions of Theorem 3 and Corollary 1 guarantee the existence of schemes in Boyce-Codd normal form and third normal form, respectively. But not every scheme that is in Boyce-Codd normal form (third normal form) satisfies the conditions of Theorem 3 (Corollary 1).

For example, let $F = F_1 \cup F_2 \cup F_3 \cup F_4$ be a set o functional dependencies where $F_1 = \{KL \to B\}$, $F_2 = \{A \to K\}$, $F_3 = \{CE \to L\}$ and $F_4 = \{ADE \to B, AB \to CDE\}$. Although the attribute B is recoverable in $Sch_4 = (\{A, B, C, D, E\}, F)$, however the schema Sch_4 is in Boyce-Codd normal form.

Nevertheless the "fast" algorithms based on the Theorem 3 and Corollary 1 can be useful for database schemas analyzing.

Moreover, the corollary 1 permits application of a fairly simple algorithm for the synthesis of database schema. Indeed let it be given a minimum set of functional dependencies, divided into equivalence classes $F = F_1 \cup ... \cup F_n$. It is know, that such a set can be achieved by $O(||F||^2)$ operations [9].

Step 1. The set Z_i of all attributes is built, which are located in the right and not in the left sides of dependencies in the current equivalence class F_i.

Step 2. From the set Z_i the next attribute A is selected.
Step 3. It is checked if X $(R_i - A) \to A \in (F - F_i)^+$ takes place. If so, then the $R_i := R_i - \{A\}$ and $Z_i := Z_i - \{A\}$ are set. Steps 2-3 are executed until all attributes in Z_i are examined.
Step 4. The right sides of the dependencies of equivalence class F_i are substituted with R_i. Steps 1-4 are executed for all equivalence classes of functional dependencies.
Step 5. The set of functional dependencies is reduced.

Each equivalence class will represent a relational scheme and the left sides of dependencies in the class will be the possible keys.

Since all recoverable attributes in the right sides of the dependencies are removed, the database schema obtained by the application of described steps will consist of relational schemes in third normal form.

It is not difficult to show that the complexity of this algorithm is the same as the complexity of the algorithm proposed by Bernstein [3] - $O(||F||^2)$.

5. Conclusions

It should be noted that the problem of relational schemes analysis is the most poorly studied. It is more difficult to identify the properties of existing databases rather than building a database with such features.

Researches are needed to determine various classes of attributes that directly affect the quality properties of the database. Obviously the question arises, if in order to recognize a normal form of a scheme is essential the determination of the prime and nonprime attributes or the determination of all keys.

It must be also investigated whether the identification of prime attributes implies searching all the keys. It is obvious, that the acceptable case, in terms of complexity of the problem is the non-affirmative one. Thus, the investigations are required to determine the equivalent characteristics of these entities, but that could be calculated in polynomial time.

Another aspect of the analysis problem lies in determining the normalization degree of a scheme. The data presented in the literature that touch this aspect is not studied enough that can be used for database testing. Investigations are needed to improve algorithms and development of new classes of models that would present features and entities at least sufficient, if not necessary and sufficient for normal forms.

References

[1] Beeri C., Bernstein Philip A., *Computational Problems Related to the Design of Normal Form Relational Database.* ACM Trans. Database Syst., V.301, N. 4, pp.752-766, 1983.

[2] Beeri C., Bernstein Philip A., *Computational Problems Related to the design of Normal Form Relations Schemes.* ACM Trans. Database Syst., V.4, N 1, p.30-59, 1979.

[3] Bernstein Philip A., *Synthesizing Third Normal Form Relations from Functional Dependencies.* ACM Trans. Database Syst., V.1, N. 4, p.277-298, 1976.

[4] Codd E.F., *Further Normalization of the data base relational model.* Data Base Systems, R. Rustin (ed), Prentice Hall, p. 33-64, 1972.

[5] Codd E.F., *Recent Investigation in Relation Data Base Systems.* IFIP Congress, p.1017-1021, 1974.

[6] Cotelea Vitalie, *An inference model for functional dependencies in database schemas.* Meridian Ingineresc, N.3, p. 89-92, 2009.

[7] Kent W., *The Universal Relation Revised.* ACM Trans. Database Syst., V.8, N 4, p.644-648, 1983.

[8] Lucchesi C.L., Osborn S.L., *Candidate Keys for Relations.* Jour. of Comput. and Syst. Sci., N.17, p.270-279, 1978.

[9] Maier D., *The Theory of Relational Database.* Computer Science Press, 637 p., 1983.

[10] Maier D., *Minimum Cover in the Relational Database Model.* Jour. of ACM, V.27, N. 4, p 664-674, 1980.

[11] Osborn Sylvia L., *Testing for Existance of a Covering Boyce-Codd Normal Form.* Information Processing Letters, V.8, N.1, p. 11-14, 1979.

[12] Paredaens J., *About Functional Dependencies in a Database Structure and their Coverings.* PhillipsMBLE Lab. Report 342, 1977.

[13] Worland Peter B., *An Efficient Algorithm for 3NF Determination.* Information Science, V.167, N.1-4, p.177-192, 2004.

Faculty of Cybernetics, Statistics and Economic Informatics, Academy of Economic Studies of Moldova, 59 Banulescu-Bodoni street, Chisinau, Republic of Moldova

E-mail address: vitalie.cotelea@gmail.com