ON RELATIONAL ALGEBRA WITH MARKED NULLS

Preliminary Version

Witold Lipski, Jr.*

Laboratoire de Recherche en Informatique
E.R.A. 452 du C.N.R.S. "Al Khowarizmi"
Université de Paris-Sud, Centre d'Orsay
Bât. 490, 91405 Orsay Cédex, France

Abstract. We formulate some natural conditions which should be satisfied in an extension of the relational algebra from the usual relations to tables with null values, and we explain the motivation behind these conditions. Roughly speaking, our conditions say that the extended algebra makes it possible to correctly compute the "true tuples" in the result of applying a relational expression (query) to tables with nulls (database state), and that the computation can be carried out recursively, following the structure of the expression. We prove that these conditions are exactly equivalent to other conditions proposed earlier by the author and T. Imieliński. We give a simple proof of the correctness of the "naive" extension of the relational algebra to tables with marked nulls, where the nulls are treated as if they were regular values, and which supports the operations of projection, positive selection, union, join and renaming of attributes. We also show that the result of the naive evaluation of such an expression (query) is equal to the response to the query as defined -- in a proof-theoretic framework -- by Reiter.

1. INTRODUCTION

Handling incomplete information in database systems is undoubtedly one of the important issues in database theory and practice, and there exists an extensive literature on this subject (see e.g. the extensive bibliography in [Lip]). In the context of relational databases the key issue is to extend the usual relational algebra from relations to tables with null values (indicating unknown values), in a "semantically correct" way.

* On leave from the Institute of Computer Science, Polish Academy of Sciences, P. O. Box 22, 00-901 Warsaw PKiN.

A correctness criterion was proposed in [IL1,IL2], where it was embodied in the definition of a so-called representation system. In this paper we give different natural correctness conditions, and we explain the motivation behind these conditions. Roughly speaking, our conditions say that the extended algebra makes it possible to correctly compute the "true tuples" in the result of applying a relational expression (query) to tables with nulls (database state), and that the computation can be carried out recursively, following the structure of the expression. We prove that these conditions are in fact exactly equivalent to those given in [IL1, IL2]. We give a simple proof of the correctness of the "naive" extension of the relational algebra to tables with marked nulls, where the nulls are treated as if they were regular values, and which supports the operators of projection, positive selection, union, natural join, and renaming of attributes (for a more thorough discussion of this extension see [IL1,IL2]). We also show that the result of the naive evaluation of such an expression (query) is equal to the response to the query as defined -- in a proof-theoretic framework -- by Reiter [Rei].

Basic facts about relational databases can be found in [Ull] or [Mai]. Here we use the notation of [IL2], which we briefly recall below. Throught the paper $\mathcal{U}$ stands for a fixed set of attributes. Associated with every $A \in \mathcal{U}$ is an attribute domain $D(A)$, containing at least two elements. Elements of $D = \bigcup_{A \in \mathcal{U}} D(A)$ are called constants. A tuple of type X, where X is a finite subset of $\mathcal{U}$, is any mapping t which associates a value $t(A) \in D(A)$ with any $A \in X$. A relation of type X is any finite set of tuples of type X, and a multirelation of type $\langle X_1,\dots,X_n\rangle$ is any sequence $r = \langle r_1,\dots,r_n\rangle$ where r_i is a relation of type X_i, $1 \leq i \leq n$. For two multirelations $r = \langle r_1,\dots,r_n\rangle$, $s = \langle s_1,\dots,s_n\rangle$ of the same type, $r \subseteq s$ means that $r_i \subseteq s_i$ for $1 \leq i \leq n$. The set of all multirelations of type X is denoted by $\mathcal{R}(X)$. The type of multirelation r, tuple t, etc. is denoted by $\alpha(r)$, $\alpha(t)$, etc.

We consider the following relational operators: projection, selection (where the selection condition is any Boolean combination of conditions of the form $A = a$, $A = B$, with $a \in D(A)$, $D(A) = D(B)$), union, (natural) join, renaming of attributes, and difference (see [IL2]). A selection operator is called positive

if its selection condition does not contain negation. For any subset Ω of the relational operators, by a relational Ω-expression we mean any well formed expression built up from relation names and symbols for relational operators in Ω. Any relation name R has an associated type $\alpha(R) \subseteq \mathcal{U}$.
A (multirelational) Ω-expression is any sequence $f = \langle f_1,\ldots,f_n\rangle$ where the f_i are relational Ω-expressions. We assume that associated with any f is a sequence $\langle R_1,\ldots,R_m\rangle$ containing all relation names occurring in f, and we define $\alpha(f) = \langle \alpha(R_1), \ldots, \alpha(R_m)\rangle$. If $r = \langle r_1,\ldots,r_m\rangle$ is a multirelation of type $\alpha(f)$ then $f(r)$ denotes the multirelation s obtained by substituting r_i for R_i, $1 \leq i \leq m$, and performing all relational operations in f. We denote $\beta(f) = \alpha(s)$. If $\mathcal{X}$ is a set of multirelations of type $\alpha(f)$ then $f(\mathcal{X})$ denotes $\{f(r)\colon r \in \mathcal{X}\}$. If $\beta(g) = \alpha(f)$ then fg denotes the composition of f and g, i.e. the result of substituting g_i for the i-th relational symbol in f.

2. REPRESENTATION SYSTEMS: TWO EQUIVALENT DEFINITIONS

Let $\mathcal{T}$ be a set of abstract objects called multitables, and assume that associated with every $T \in \mathcal{T}$ is a type $\alpha(T) = \langle X_1,\ldots,X_n\rangle$ and a nonempty set Rep(T) of multirelations of type $\alpha(T)$. We think of multitable T as an "incompletely specified" multirelation, where Rep(T) gives the set of possibilities for this unknown multirelation. In other words, if T is a database state, then all we know about the real world is that it is exactly described by one of the multirelations in Rep(T).

Although the internal structure of a multitable will be immaterial in this section, one may think of a multitable as a "generalized multirelation" where some tuples may contain null values.

Suppose a user submits a query (Ω-expression) f and that the database state is T. Since the true state of the real world is represented by some $r^* \in \mathrm{Rep}(T)$, all we know about the "true response" $f(r^*)$ is that it is in the set f(Rep(T)). In general it is impossible to represent this set of possibilities by any multitable U so as to satisfy f(Rep(T)) = Rep(U). However, in many cases it is enough to "approximate" f(Rep(T)) by Rep(U). The notion of a "sufficiently good approximation" is formalized by the following definition, introduced in [IL1,IL2]:

Definition 1. $\langle \mathcal{T}, \mathrm{Rep}, \Omega\rangle$ is a representation system if for any Ω-expression f and for any T with $\alpha(f) = \alpha(T)$ there exists $U \in \mathcal{T}$ such that

$$(1) \qquad \mathrm{Rep}(U) \underset{\Omega}{\equiv} f(\mathrm{Rep}(T)),$$

where for any sets $\mathcal{X}, \mathcal{Y}$ of multirelations of the same type X,

$$\mathcal{X} \underset{\Omega}{\equiv} \mathcal{Y} \iff \text{for any } \Omega\text{-expression } g \text{ with } \alpha(g) = X,\ \bigcap g(\mathcal{X}) = \bigcap g(\mathcal{Y})$$

(the intersections are understood coordinatewise).

The intuition behind (1) is the following: U sufficiently well represents f(Rep(T)), since the fact that the two sets on both sides of (1) may be different cannot be found out by a user who has a query language based on Ω-expressions at his/her disposal and who is concerned only with the "true tuples" in a response; see [IL1,IL2] for a more detailed discussion.

We now give a different definition of a representation system.

Definition 2. $\langle \mathcal{T}, \mathrm{Rep}, \Omega\rangle$ is a representation system if there is a function which associates a multitable, denoted by f(T), with any Ω-expression f and any $T \in \mathcal{T}$ with $\alpha(T) = \alpha(f)$, in such a way that

$$(2) \qquad \bigcap \mathrm{Rep}(f(T)) = \bigcap f(\mathrm{Rep}(T))$$

and

$$(3) \qquad (gf)(T) = g(f(T))$$

for all Ω-expressions g with $\alpha(g) = \beta(f)$.

The motivation behind Definition 2 is the following: Suppose we want to design a system which correctly evaluates the set of "true tuples"

$$(4) \qquad \bigcap f(\mathrm{Rep}(T)),$$

i.e. those tuples which are in the "true response" $f(r^*)$, no matter which element of Rep(T) turns out to be the true state r^* of the real world. Moreover, assume that our system computes (4) by first evaluating a multitable U = f(T) and then determining the set of "true tuples" $\bigcap \mathrm{Rep}(U)$. Note that (2) is exactly the statement that such a system is correct. Notice however that if any multirelation is (or can be identified with) a multitable then (2) can trivially be satisfied by putting

$$f(T) = \bigcap f(\mathrm{Rep}(T))$$

((3) is then, in general, not satisfied). What we really would like to have is not just a system which computes f(T) individually for any f and T, but a true extension of the relational algebra from relations to tables, where for any relational operator in Ω we have a rule how to perform this operator over tables. Such a system provides a uniform recursive method to compute f(T), and is clearly characterized by condition (3).

The main result of this section is

Theorem 1. Definition 1 is equivalent to Definition 2. $\square$

3. REPRESENTATION SYSTEM BASED ON TABLES WITH MARKED NULLS

In this section we give a simple proof of the fact that the system based on tables with marked nulls (V-tables) considered in [IL1,IL2] is indeed a representation system. The proof of this fact given in [IL1,IL2] relied on a rather complicated notion of a so-called conditional table. Here we in addition indicate a connection of the representation system based on marked nulls with the approach of Reiter [Rei].

For any attribute $A \in \mathcal{U}$, let V(A) be an infinite set of variables, and let $V = \bigcup_{A \in \mathcal{U}} V(A)$. The elements in V will also be called (marked) nulls. We shall assume that $V \cap D = \emptyset$, that $V(A) = V(B)$ if $D(A) = D(B)$, and that $V(A) \cap V(B) = \emptyset$ if $D(A) \neq D(B)$. A V-tuple of type $X \subseteq \mathcal{U}$ is any mapping associating a value $t(A) \in D(A) \cup V(A)$ with every $A \in X$, a V-table of type X is any finite set of V-tuples of type X, and a V-multitable of type $\langle X_1,\ldots,X_n\rangle$ is any sequence $T = \langle T_1,\ldots,T_n\rangle$ of V-tables, where T_i is of type X_i. A valuation is any mapping $v\colon V \to D$ such that for all $A \in \mathcal{U}$ and $x \in V$, $x \in V(A)$ implies $v(x) \in D(A)$. The set of all valuations is denoted by Val. For any V-multitable T we define

$$\mathrm{rep}(T) = \{v(T)\colon v \in \mathrm{Val}\}$$

$$\mathrm{Rep}(T) = \{r \in \mathcal{R}(\alpha(T))\colon \text{for some } v \in \mathrm{Val},\ v(T) \subseteq r\}.$$

Here $v(T)$ denotes the result of substituting, for all $x \in V$, all occurrences of x in T by $v(x)$ (the meaning of the notation $v(t)$ used later is similar).

Given a multirelational expression f and a multitable T, we can define $f^{CWA}(T)$ and $f^{OWA}(T)$ -- the CWA (Closed World Assumption) and OWA (Open World Assumption) responses, respectively, for query f in database state T -- in the following way:

$$f^{CWA}(T) = \{t\colon (\forall v)\ v(t) \in f(v(T))\}$$

$$f^{OWA}(T) = \{t\colon (\forall v,r)(v(T) \subseteq r \Rightarrow v(t) \in f(r))\}.$$

(Here v and r range over all valuations, and multirelations of type $\alpha(T)$, respectively. If f is a multirelational expression then we should make precise the notion of "tuple t belonging to a multitable", in the obvious way).

In fact, these definitions should be treated as different versions of the definition given -- in terms of the predicate calculus -- by Reiter [Rei]. Reiter defines the response to query $W(\vec{x})$, expressed by a formula $W(\vec{x})$ of the predicate calculus with free variables $x_1,\ldots,x_n$, as the set of all V-tuples t (called answers in [Rei]), such that for any model M of his theory DB, $M \models W(t)$, i.e. that the interpretation of t belongs to the interpretation of W in M. In the case of f^{CWA}, any model (up to isomorphism) is given by $v(T)$ for a valuation v, the interpretation of t is $v(t)$, and the interpretation of W is computed algebraically as $f(v(T))$ (according to "Codd Completeness Theorem"). The meaning of f^{OWA} can be interpreted in a similar way.

Clearly, $f^{OWA}(T) \subseteq f^{CWA}(T)$, and if f is monotone, $f^{OWA} = f^{CWA}$.

If we assume that no variable can occur in two different columns in T, or that the attribute domains are infinite, then we can prove the following lemma.

Lemma 1. For any multirelational expression f

$$\bigcap \mathrm{rep}(f^{CWA}(T)) = \bigcap f(\mathrm{rep}(T))$$

(preservation of "true tuples", cf. (2)). □

For any multirelational expression f and any V-multitable T with $\alpha(T) = \alpha(f)$, $f(T)$ will denote the result of evaluating f over T, treating variables as if they were constants (different from all constants in D) -- this will be referred to as the "naive evaluation". For any T and f, let $\mathrm{Val}^*(T,f)$ denote the set of all valuations v such that $v(x)$ does not occur in either T or in (selection conditions in) f, and $v(x) \neq v(y)$ for all x,y, $x \neq y$.

Theorem 2. If the attribute domains are sufficiently large (e.g. infinite; more exactly, if $V^*(T,f) \neq \emptyset$) then the naive evaluation is complete, i.e.

$$f^{CWA}(T) \subseteq f(T).$$

Note 1. Here, similarly as in Lemma 1, f is any multirelational expression (possibly involving difference).

Note 2. In general, the naive evaluation is not sound, i.e. the inverse inclusion does not hold.

The main fact used in the proof is that $f(v(T)) = v(f(T))$ for all $v \in V^*(T,f)$. □

In the rest of the paper we shall be concerned with Ω-expressions with Ω consisting of projection, positive selection, union, join, and renaming of attributes, denoted $\Omega = PS^+UJR$.

Theorem 3. The naive evaluation is sound for PS^+UJR-expressions, i.e.

$$f(T) \subseteq f^{CWA}(T)$$

for any PS^+UJR-expression f.

The main fact used in the proof is that $v(f(T)) \subseteq f(v(T))$ for any PS^+UJR-expression f. □

Combining Theorem 2 with Theorem 3 we conclude that for any PS^+UJR-expression f, $f^{CWA}(T)$ is given by the naive evaluation of f over T. Finally, taking into account that the naive evaluation clearly satisfies (3), and using Theorem 1, we obtain

Theorem 4. [IL1,IL2] The set of V-multitables with the funcion rep (or Rep) define a representation system supporting $\Omega = PS^+UJR$ (under the infinite attribute domains assumption). □

It may be noted that Theorem 4 -- together with the results of [IL3] -- can be used to prove the correctness of the query evaluation method based on representative instances in the context of the universal relation model [MUV]: T plays then the role of a representative instance.

REFERENCES

[IL1] T. Imieliński, W. Lipski, On representing incomplete information in a relational database. Proc. 7th Internat. Conf. on Very Large Data Bases, Cannes, France, Sept. 9-11, 1981, pp. 388-397.

[IL2] T. Imieliński, W. Lipski, Incomplete information in relational databases. Institute of Computer Science PAS Report 475, Warsaw, May 1982, submitted for publication.

[IL3] T. Imieliński, W. Lipski, Incomplete information and dependencies in relational databases. Proc. ACM SIGMOD Internat. Conf. on Management of Data, San Jose, CA, May 23-26, 1983, pp. 178-184.

[Lip] W. Lipski, Logical problems related to incomplete information in databases. Presented at Colloquium on Algebra, Combinatorics and Logic in Computer Science, Győr, Hungary, Sept. 12-16, 1983.

[Mai] D. Maier, The Theory of Relational Databases. Computer Science Press, Rockville, MD, 1983.

[MUV] D. Meier, J. D. Ullman, M. Y. Vardi, On the foundations of the universal relation model. Manuscript, 1983.

[Rei] R. Reiter, A sound and sometimes complete query evaluation algorithm for relational databases with null values. Techn. Rep. 83-11, Dept. of Computer Science, The Univ. of British Columbia, Vancouver, BC, Canada, June 1983.

[Ull] J. D. Ullman, Principles of Database Systems, Second Edition. Computer Science Press, Potomac, MD, 1982.