A SOUND AND SOMETIMES COMPLETE QUERY EVALUATION ALGORITHM FOR RELATIONAL DATABASES WITH NULL VALUES

Raymond Reiter
Department of Computer Science
University of British Columbia
Vancouver, B.C. V6T 1W5
Canada

Technical Report 83-11

A SOUND AND SOMETIMES COMPLETE QUERY EVALUATION ALGORITHM FOR RELATIONAL DATABASES WITH NULL VALUES

Raymond Reiter
Department of Computer Science
University of British Columbia
Vancouver, B.C. V6T 1W5
Canada

June, 1983

ABSTRACT

This paper presents a sound and, in certain cases, complete method for evaluating queries in relational databases with null values where these nulls represent existing but unknown individuals. The soundness and completeness results are proved relative to a formalization of such databases as suitable theories of first order logic.

Key Words and Phrases:

relational databases, relational algebra, null values, query evaluation, first order logic, integrity constraints

CR Categories

A SOUND AND SOMETIMES COMPLETE QUERY EVALUATION ALGORITHM FOR RELATIONAL DATABASES WITH NULL VALUES

Raymond Reiter
Department of Computer Science
University of British Columbia
Vancouver, B.C. V6T 1W5
Canada

In [Reiter 1983] I proposed that a formal theory of databases can be formulated within the first order predicate calculus, and I presented a variety of arguments in favour of doing so. One such argument is that when data models are defined as first order theories there is a logical definition of the answers to a query, and this definition is independent of the data model (relational, entity-relationship, what have you) under consideration. As a consequence, a query evaluation algorithm for a particular data model may be proved sound - it returns only correct answers - and complete - it returns all correct answers - with respect to the logical semantics of that data model. Another argument in [Reiter 1983] is that the semantics of data models can be specified precisely and unambiguously by rendering them as first order theories. By way of an example, I showed how the relational data model extended to include the null value "an existing but unknown individual" may be defined in first order logic, thereby specifying a logical semantics for this null. I then posed the question whether there is an extension of the relational algebra for which there is a sound and complete query evaluation algorithm for relational databases with null values. My purpose in this paper is to partially answer this question.

Specifically, I shall propose a generalization of the relational algebra and a query evaluation algorithm based on its operators which is provably sound with respect to the logical semantics of null values. Unfortunately, the algorithm is not complete in general. However, for two classes of queries, namely positive queries and universal conjunctive queries, the algorithm is complete. Not too surprisingly, it also turns out to be complete for relational databases without null values.

2. *Motivation*

In this section we briefly recapitulate the argument of [Reiter 1983] which leads to the formalization of relational databases with null values as suitable theories of first order logic. The ideas are best conveyed by means of an example, so consider the following supplier - part database:

PART	SUPPLIER	SUPPLIES		SUBPART	
p_1	A	A	p_1	p_1	p_2
p_2	B	B	p_2		
p_3					

Here SUPPLIES and SUBPART are meant to be relation names and PART and SUPPLIER are the names of the domains of these relations.

Our objective is to reformulate the interpretation of such tables made by conventional relational database theory in terms of first order logic. One reason for doing so will become evident later when we extend the relational model to include null values. In order to carry out this logical reconstruction of relational database theory it is necessary to examine carefully the implicit *assumptions* underlying the relational model.

Assumption 1 - The Closed World Assumption [Reiter 1978]

This assumption has it that the entries in a table are all and only the tuples satisfying the relation. Thus SUPPLIES(A,p_1) holds, but SUPPLIES(A,p_2) is false because the tuple (A,p_2) is absent from the SUPPLIES table. Thus a tuple satisfies a relation (or domain) iff it is in the table for that relation (or domain). Let E(.,.) be the equality "relation" i.e. E(x,y) holds iff x equals y. Then we can formalize the closed world assumption for the example by the following logical equivalences:

$$(x)[PART(x) \equiv E(x,p_1) \vee E(x,p_2) \vee E(x,p_3)] \quad (2.1)$$

$$(x)[SUPPLIER(x) \equiv E(x,A) \vee E(x,B)] \quad (2.2)$$

$$(x)(y)[SUPPLIES(x,y) \equiv E(x,A) \wedge E(y,p_1) \vee E(x,B) \wedge E(y,p_2)] \quad (2.3)$$

$$(x)(y)[SUBPART(x,y) \equiv E(x,p_1) \wedge E(y,p_2)] \quad (2.4)$$

Assumption 2 - The Unique Name Assumption [Reiter 1980a]

This is the assumption that the individuals in the database, e.g. A, B, p_1 etc., are pairwise distinct. For example it is the assumption that A and B are distinct, together with the closed world assumption, which sanctions the conclusion that SUPPLIES(B,p_1) is false. We can formalize the unique name assumption for the example by the following formulae:

$$\neg E(A,B),\ \neg E(A,p_1),\ \neg E(p_1,p_2)\ \text{etc.} \quad (2.5)$$

Having explicitly introduced the equality predicate E, we must also formalize its properties. This can be done in the usual way by introducing axioms defining the reflexivity, symmetry and transitivity properties of equality, together with the principle of substitution of a term for another equal to it. These axioms will be given later.

To summarize, our claim is that what relational database theory means by the information in a set of relational tables can be formalized by a set of first order formulae consisting of

1. Logical equivalences like (2.1) - (2.4) which realize the closed world assumption.
2. Unique name axioms like those of (2.5).

3. Axioms for equality.

So far, we have not considered null values. The particular null of concern in this paper is that denoting an unknown individual about which certain properties are known, for example an unknown supplier who is known to supply part p_1. Such null values have been problematic for database theory ever since they were first proposed. The principal difficulties appear to be semantic. Although many approaches exist in the literature e.g. [Biskup 1981, Codd 1979, Walker 1980, Vassiliou 1979, Zaniolo 1977] there is no general agreement as to which of these, if any, provides a correct formal semantics for nulls. In [Reiter 1983] I argue that these semantic difficulties stem from an inappropriate theoretical foundation for database theory. This foundation holds that a database is some kind of model, in the logical sense of the word "model". Thus, from this model theoretic perspective relational calculus expressions are seen to have values true or false with respect to a database. In the presence of null values a third truth value - "unknown" - becomes necessary. The problem then becomes one of defining coherent truth tables for the resulting multi-valued logics. Typically such attempts fail, for example by assigning "unknown" to certain tautologies.

The perspective in [Reiter 1983] which this current paper pursues, is that databases are not models, but *theories* (i.e. sets of logical formulae). It is this perspective which informed our earlier formalization of a relational database without nulls as a suitable set of first order formulae. As we shall now see, this point of view can provide an intuitively correct semantics for null values, without appealing to multi-valued logics.

To focus the discussion, consider again our supplier-part database and its corresponding formulae (2.1) - (2.5). Suppose we wish to represent the fact "Some supplier supplies part p_3 but I don't known who it is. Moreover, this supplier may or may not be one of the known suppliers A and B." This fact may be represented by the first order formula

$$(Ex)SUPPLIER(x) \wedge SUPPLIES(x,p_3) \tag{2.6}$$

which asserts the existence of an individual x with the desired properties. Now we can choose to name this existing individual - call it ω - and instead of (2.6), ascribe these properties to ω directly:

$$SUPPLIER(\omega) \wedge SUPPLIES(\omega,p_3) \tag{2.7}$$

In database terminology, ω is a *null value.* It is called a *Skolem constant* by logicians. Skolem constants, or more generally Skolem functions, provide a technical device for the elimination of existential quantifiers in proof theory. (See, for example, [Chang and Lee 1973]).

The problem at hand is how to correctly integrate the facts (2.7) into our supplier and parts theory (2.1) - (2.5). Notice first that ω is a new constant, perhaps denoting the same individual as some known constant, perhaps not. So the unique name axioms (2.5) remain untouched. However, the SUPPLIER and SUPPLIES tables now should contain new tuples (ω) and (ω,p_3) respectively so that formulae (2.2) and (2.3) should be expanded to:

$$(x)[SUPPLIER(x) \equiv E(x,A) \vee E(x,B) \vee E(x,\omega)] \tag{2.8}$$

$$(x)(y)[SUPPLIES(x,y) \equiv E(x,A) \wedge E(y,p_1) \vee E(x,B) \wedge E(y,p_2) \vee E(x,\omega) \wedge E(y,p_3)] \tag{2.9}$$

The resulting set of first order formulae (2.1), (2.8), (2.9), (2.4) and (2.5) intuitively provides a correct representation of this new setting. Notice that in this resulting theory, *the only thing which distinguishes the null value ω from the "ordinary" constants A, B etc. is the absence of unique name axioms for ω;* there are no formulae $\neg E(\omega,A)$, $\neg E(\omega,B)$

etc. asserting that ω is distinct from the other individuals of the database. This, of course, is as it should be since ω's identity is unknown and hence cannot be assumed to be distinct from the other individuals.

Notice also that in this theory we can prove things like $\neg$SUPPLIES(A,p_2) and $\neg$SUPPLIES(B,p_1) but *not* $\neg$SUPPLIES(A,p_3) or $\neg$SUPPLIES(B,p_3). Intuitively, this is precisely what we want. For we known SUPPLIES(ω,p_3). Moreover, we *don't know* whether ω is the same as, or different than A or B. So if we could prove, say $\neg$SUPPLIES(A,p_3), we could also prove $\neg$E(ω,A) contradicting our presumed ignorance about the identity of ω.

Suppose now that in addition we wish to represent the fact "Some supplier - possibly the same as A or B or ω, possibly not - supplies p_2"

$$(Ex)SUPPLIER(x) \wedge SUPPLIES(x,p_2)$$

we must choose a name for this supplier, say ω' , which must be distinct from the name of the previous unknown supplier ω. This, because we are not justified in assuming that ω and ω' are the same supplier. Moreover, the formulae (2.8) and (2.9) must be expanded to accommodate this new information:

$$(x)[SUPPLIER(x) \equiv E(x,A) \vee E(x,B) \vee E(x,\omega) \vee E(x,\omega')]$$

$$(x)(y)[SUPPLIES(x,y) \equiv E(x,A) \wedge E(y,p_1) \vee E(x,B) \wedge E(y,p_2)$$
$$\vee E(x,\omega) \wedge E(y,p_3) \vee E(x,\omega') \wedge E(y,p_2)]$$

Thus, in general, each time a new null value is introduced into the theory this null must be denoted by a fresh name, distinct from all other names of the theory. Therefore, we are dealing with so-called *indexed* nulls.

Let us summarize our approach to null values. Informally, the nulls we are considering represent unknown individuals which may, or may not, be the same as the other individuals (known or unknown) in the database; we simply don't know. New nulls are assigned names distinct from all other individual names in the database. A relational database with null values is formalized by a first order theory consisting of:

1. Logical equivalences like those of (2.8) and (2.9), one for each relation and domain name.
2. Unique name axioms like those of (2.5). Each pair of distinct known individuals a, b contributes ¬E(a,b) to these axioms. A null value contributes no such axiom.[1]
3. Axioms for equality.

3. Formal Preliminaries

In this section we formalize the intuitions of the previous section by defining first a suitable first order language, then an appropriate class of theories over this language, and finally a query language.

3.1 Relational Languages

A *first order language* is specified by a pair (ALPHA,WFFS) where ALPHA is an alphabet of symbols and WFFS is a set of syntactically well formed expressions called well formed formulae and which are constructed using the symbols of ALPHA. The rules for constructing the formulae of WFFS are the same for all first order languages; only the alphabet ALPHA may vary. ALPHA must contain symbols of the following kind, and only such symbols:

---

[1] This restriction will be relaxed in the formal development which follows. Thus if a null value ω is known to be different than another database individual α, null or not, then $\neg E(\omega,\alpha)$ will be permitted by the formalism.

Variables: x, y, z, x_1, y_1, z_1, ...,

There must be infinitely many of these.

Constants: a, b, c, p1, p_2, . . . ,

There may be 0 or more of these, possibly infinitely many.

Predicates: P, Q, R, SUPPLIES, SUPPLIER, PART, ...,

There must be at least one of these, possibly infinitely many. With each is associated an integer $n \geq 0$, its *arity,* denoting the number of arguments it takes.

Punctuation Signs: parentheses and comma.

Logical Constants: $\supset$ (implies), $\wedge$ (and), $\vee$ (or), $\neg$ (not), $\equiv$ (iff).

Notice that function symbols are not included in this alphabet.

With such an alphabet ALPHA in hand, we can construct a set of syntactically well formed expressions, culminating in a definition of the set WFFS of well formed formulae, as follows:

Terms

A variable or a constant of ALPHA is a *term.*

Atomic Formulae

If P is an n-ary predicate of ALPHA and $t_1, \ldots, t_n$ are terms, then $P(t_1, \ldots, t_n)$ is an *atomic formula.* $P(t_1, \ldots, t_n)$ is a *ground* atomic formulae iff $t_1, \ldots, t_n$ are all constants.

Well Formed Formulae

WFFS is the smallest set such that

(i) An atomic formula is a well formed formula (wff).

(ii) If W_1 and W_2 are wffs, so also are $(W_1 \wedge W_2)$, $(W_1 \vee W_2)$, $(W_1 \supset W_2)$, $(W_1 \equiv W_2)$, $\neg W_1$.

(iii) If x is a variable and W is a wff, then (x)(W) and (Ex)(W) are wffs. Here (x) is a *universal quantifier* and (Ex) an *existential quantifier.*

For the purpose of formally defining a relational database with null values, we won't require arbitrary first order languages; a suitable proper subset of these will do. Accordingly, define a first order language (ALPHA,WFFS) to be a *relational language* iff ALPHA has the following properties:

1. There are only finitely many constants in ALPHA.
2. There are but finitely many predicates in ALPHA.
3. Among the predicates of ALPHA there is a distinguished binary predicate E which will function for us as equality.
4. Among the predicates of ALPHA there is a distinguished non-empty subset of unary predicates. Such unary predicates are called *simple types.* Not all unary predicates of ALPHA need be simple types. Such simple types will, in part, model the concept of the domain of a relation as it arises in standard database theory.

For a relational language (ALPHA,WFFS) it is convenient to define appropriate syntactically sugared *abbreviations* for certain of the wffs of WFFS, as follows:

If τ is a simple type, then

$(x/\tau)(W)$ abbreviates $(x)(\tau(x) \supset W)$

$(Ex/\tau)(W)$ abbreviates $(Ex)(\tau(x) \wedge W)$

Here $(x/\tau)(W)$ should be read as "For all x which are τ, W is the case." and $(Ex/\tau)(W)$ as

"There is an x, which is a τ, such that W is the case." Thus these *type restricted quantifiers* are meant to restrict the possible x's to just those which belong to the class τ. Notice that quantifiers may be restricted only by types, not by arbitrary predicates.

Example 3.1

For our supplier-part example, the unary predicates PART, SUPPLIER are simple types. The following English statements translate naturally into type restricted quantified wffs:

"Every supplier supplies at least one part."

$$(x/SUPPLIER)(Ey/PART)SUPPLIES(x,y)$$

which abbreviates the ordinary wff

$$(x)[SUPPLIER(x) \supset (Ey)(PART(y) \wedge SUPPLIES(x,y))]$$

"Some supplier supplies all subparts of p_3."

$$(Ex/SUPPLIER)(y/PART)[SUBPART(y,p_3) \supset SUPPLIES(x,y)]$$

which abbreviates the ordinary wff

$$(Ex)SUPPLIER(x) \wedge (y)[PART(y) \supset [SUBPART(y,p_3) \supset SUPPLIES(x,y)]]$$

I have in this example omitted a lot of parentheses on the assumption that it is clear what these formulae *mean*. I shall continue this practise whenever no ambiguity will result.

More Abbreviations

If $\mathbf{x} = x_1,...,x_n$ is a sequence of distinct variables then $W(\mathbf{x})$ abbreviates $W(x_1,...,x_n)$ and $(\mathbf{x})W(\mathbf{x})$ abbreviates $(x_1)...(x_n)W(x_1,...,x_n)$. When E is the equality predicate and $\mathbf{s} = s_1,...,s_n$ and $\mathbf{t} = t_1,...,t_n$ are equal length sequences of terms then $E(\mathbf{s},\mathbf{t})$ abbreviates $E(s_1,t_1) \wedge ... \wedge E(s_n,t_n)$. $\neg E(\mathbf{s},\mathbf{t})$ abbreviates $\neg E(s_1,t_1) \vee ... \vee \neg E(s_n,t_n)$.

3.2 Extended Relational Theories

Let (ALPHA,WFFS) be a relational language. A finite subset R of WFFS is an *extended relational theory* iff R satisfies the following conditions:

1. For each n-ary predicate P of ALPHA distinct from E (but including the simple types), R contains *exactly one* formula of the form

$$(\mathbf{x})P(\mathbf{x}) \equiv E(\mathbf{x},\mathbf{c}^{(1)}) \vee \cdots \vee E(\mathbf{x},\mathbf{c}^{(r)})$$

   where the $\mathbf{c}^{(i)}$ are n-tuples of constants of ALPHA. The case $r = 0$ is permitted, in which case the corresponding formula is $(\mathbf{x}) \neg P(\mathbf{x})$. This formula is called the *extension axiom* of P in R.

2. R contains the following equality axioms:

   (i) Reflexivity

   $$(x)E(x,x)$$

   (ii) Symmetry

   $$(x)(y)E(x,y) \supset E(y,x)$$

   (iii) Transitivity

   $$(x)(y)(z)E(x,y) \wedge E(y,z) \supset E(x,z)$$

   (iv) Substitution of equal terms

   For each n-ary predicate P of ALPHA distinct from E

   $$(\mathbf{x})(\mathbf{y})P(\mathbf{x}) \wedge E(\mathbf{x},\mathbf{y}) \supset P(\mathbf{y}).$$

   These are called the *Leibnitz axioms* of R.

3. R contains 0 or more *unique name axioms* of the form $\neg E(c,c')$ for distinct constants c,c' of ALPHA.

4. Nothing else is in R.

Notice that extended relational theories provide no formal distinction between null values, which denote "unknown" individuals, and "ordinary" constants, which denote "known" individuals. Insofar as an extended relational theory is concerned there is simply available a collection of undistinguished constants certain pairs of which are unequal. Thus, the approach of this paper fails to capture fully the meaning of null values. For the purposes of representing null values, the only important feature of their "unknown" character is deemed to be the absence of some of their unique name axioms. In order to fully characterize their "unknown" property, it appears necessary to invoke a modal logic of belief [Levesque 1982]. As Levesque shows, it then becomes possible to express, and answer, queries like "Who are the *known* suppliers of p_3?" and "Which *unknown* suppliers are Canadian?". Such representational power transcends the approach of this paper.

Notice also that extended relational theories permit some quite subtle distinctions to be represented, for example:

"Someone supplies p_3 but I don't know who. Whoever it is, it is neither A nor B".

$$(Ex/SUPPLIER)SUPPLIES(x,p_3) \wedge \neg E(x,A) \wedge \neg E(x,B)$$

which, after elimination of the existential quantifier, becomes

$$SUPPLIER(\omega) \wedge SUPPLIES(\omega,p_3) \wedge \neg E(\omega,A) \wedge \neg E(\omega,B)$$

"Someone supplies p_2 and someone supplies p_3. I don't know who they are but I do know they are not the same suppliers."

$$(Ex/SUPPLIER)(Ey/SUPPLIER)SUPPLIES(x,p_2) \wedge SUPPLIES(y,p_3) \wedge \neg E(x,y)$$

which becomes

$$SUPPLIER(\omega_1) \wedge SUPPLIER(\omega_2) \wedge SUPPLIES(\omega_1,p_2) \wedge SUPPLIES(\omega_2,p_3) \wedge \neg E(\omega_1,\omega_2)$$

Theorem 3.1

When R is an extended relational theory its Leibnitz axioms are dependent on (i.e. are provable from) the remaining axioms of R.

Proof: A typical Leibnitz axiom has the form

$$(x)(y)P(x) \wedge E(x,y) \supset P(y)$$

If P's extension axiom is $(x)\ \neg P(x)$ then the Leibnitz axiom is provable using P's extension axiom as sole premise. Otherwise, P's extension axiom is of the form

$$(x)P(x) \equiv E(x,c^{(1)}) \vee ... \vee E(x,c^{(r)})$$

so that the Leibnitz axiom is equivalent to

$$(x)(y)[E(x,c^{(1)}) \vee ... \vee E(x,c^{(r)})] \wedge E(x,y) \supset E(y,c^{(1)}) \vee ... \vee E(y,c^{(r)})$$

and this is provable using only the non Leibnitz equality axioms.

□

In view of Theorem 3.1, the Leibnitz axioms of R are irrelevant. Henceforth, we shall assume that extended relational theories contain no such axioms i.e. they contain only extension axioms, axioms specifying the reflexivity, symmetry and transitivity of equality, and unique name axioms.

Theorem 3.2

Any extended relational theory is consistent.

Proof: The proof is a simple consequence of [Reiter 1983, Theorem 4.2].

3.3 Queries and Their Answers

Following [Reiter 1983] we define a *query* for a relational language (ALPHA,WFFS) to be any expression of the form $<\mathbf{x}/\boldsymbol{\tau}|W(\mathbf{x})>$ where $\mathbf{x}/\boldsymbol{\tau}$ denotes $x_1/\tau_1,...,x_n/\tau_n$, each x_i is a distinct variable of ALPHA, each τ_i is a simple type of ALPHA, and $W(\mathbf{x}) \in$ WFFS is a wff whose free variables are among $x_1,...,x_n$ and whose quantifiers are all typed quantifiers. The case n=0 is permitted, in which case the query has the form $< |W>$ where W has no free variables.

Informally, a query $<\mathbf{x}/\tau|W(\mathbf{x})>$ is meant to denote the set of all n-tuples $\mathbf{x} = (x_1,...,x_n)$ such that each x_i satisfies the simple type τ_i and such that the database satisfies $W(\mathbf{x})$. A formal definition will follow the next example.

Example 3.2

The following are some queries for our supplier-part database:

1. Suppliers supplying more than one part.

$$<x/SUPPLIER \mid (Ey/PART)(Ez/PART)\neg E(y,z) \wedge SUPPLIES(x,y) \wedge SUPPLIES(x,z)>$$

2. Suppliers supplying all subparts of p_3.

$$<x/SUPPLIER \mid (y/PART)SUBPART(y,p_3) \supset SUPPLIES(x,y)>$$

3. Pairs of distinct suppliers supplying the same parts.

$$<x/SUPPLIER, y/SUPPLIER \mid \neg E(x,y) \wedge (z/PART)SUPPLIES(x,z) \equiv SUPPLIES(y,z)>$$

Let DB $\subseteq$ WFFS be any set of closed wffs. We view DB as defining a database, not necessarily relational. Elsewhere [Reiter 1980a] I have referred to any such DB as a *first order database*. An n-tuple $\mathbf{c} = (c_1,...,c_n)$ of constants of ALPHA is an *answer* to the query $<\mathbf{x}/\tau \mid W(\mathbf{x})>$ *with respect to* DB iff the formulae $\tau_i(c_i)$, $i = 1,...,n$, and $W(\mathbf{c})$ are all true in every model of DB. By the Godel Completeness Theorem for first order logic, this is equivalent to

1. $DB \vdash \tau_i(c_i) \quad i = 1,...,n$ and

2. DB $\vdash$ W(c)

where by S $\vdash$ w we mean that first order formula w is provable from the set of first order formulae S as premises. The set of all such answers is denoted $||<x/\tau \mid W(x)>||_{DB}$. When the first order database DB is clear from the context, the subscript "DB" will be omitted.

Notice the special case of this definition when n=0 i.e. when the query is $< |W>$. Then ( ) - the null tuple - is the sole answer to the query iff DB $\vdash$ W. Thus $||< |W>||$ is {( )} when DB $\vdash$ W and { } otherwise. Posing such a query corresponds to asking the database whether W holds. A response of {( )} denotes the answer "yes" while { } denotes "I don't know" - not "no". A response of {( )} to the query $< |\neg W>$ provides the answer "no" to the original query $< |W>$.

The purpose of this paper is to provide a sound and, in certain special cases, complete query evaluation algorithm for relational databases with null values. By "sound" and "complete" we mean the following:

A query evaluation algorithm is *sound* for extended relational theories iff for any such theory R and query Q the algorithm returns a subset of $||Q||_R$. Such an algorithm is *complete* iff it returns a superset of $||Q||_R$.

4. Sound Query Evaluation for Relational Databases with Null Values

The objective of this section is to provide a sound but, in general, incomplete query evaluation algorithm for relational databases with null values. Thus in general the algorithm will return some but not all answers to a query. The algorithm proceeds by recursively decomposing complex queries into suitable set theoretic operations on simpler queries. In most cases, the decomposition is equal to the original query, but in two cases the decomposition may be a strict subset of the original. It is these latter two cases which lead to the incompleteness of the method.

The first two such decompositions hold for arbitrary first order databases.

Theorem 4.1.

If DB is a first order database, then

$$||<x/\tau \mid W_1(x) \wedge W_2(x)>|| = ||<x/\tau \mid W_1(x)>|| \cap ||<x/\tau \mid W_2(x)>||$$

Proof. Follows from the simple fact that for any tuple **c** of constants

DB $\vdash W_1(c) \wedge W_2(c)$ iff DB $\vdash W_1(c)$ and DB $\vdash W_2(c)$.

Theorem 4.2.

If DB is a first order database, then

$$||<x/\tau \mid W_1(x)>|| \cup ||<x/\tau \mid W_2(x)>|| \subseteq ||<x/\tau \mid W_1(x) \vee W_2(x)>||$$

Proof: Follows from the simple fact that for any tuple **c** of constants, if DB $\vdash W_1(c)$ or DB $\vdash W_2(c)$ then DB $\vdash W_1(c) \vee W_2(c)$.

QED

Alas, the reverse inclusion of Theorem 4.2 does not hold, even for extended relational theories. For example, consider the theory with a single type τ, two constants a and b, a single predicate $P(\cdot)$, no unique name axioms, and extension axioms

$$(x)P(x) \equiv E(x,a)$$

$$(x)\tau(x) \equiv E(x,a) \vee E(x,b)$$

Then

$$||<x/\tau \mid P(x) \vee \neg P(x)>|| = \{a,b\}$$

but

$$||<x/\tau \mid P(x)>|| = \{a\}$$

and

$$||<x/\tau \mid \neg P(x)>|| = \{\ \}.$$

As we shall see, this failure of the reverse inclusion in Theorem 4.2 is one of two reasons preventing the completeness of the query evaluation algorithm of this paper.

In the subsequent development we shall require some preliminary concepts and lemmas.

If R is an extended relational theory, denote by E-AXIOMS_R the formulae of R which involve only the equality predicate i.e. R's unique name axioms, together with the reflexive, symmetric and transitive axioms for equality.

Lemma 4.3.

If R is an extended relational theory and W is a closed wff in which the only predicate that occurs is E, then $R \vdash W$ iff $\text{E-AXIOMS}_R \vdash W$.

Proof.

The necessity is trivial. To prove sufficiency we argue model-theoretically that for any model M of E-AXIOMS_R W is true in M, from which it follows that $\text{E-AXIOMS}_R \vdash W$.

To that end, assume $R \vdash W$ and that M is a model of E-AXIOMS_R. Define a structure M′ as follows:

1. M′ has the same domain as M.
2. M′ interprets the equality predicate E exactly as does M.
3. For each predicate P distinct from E, if P's extension axiom in R is $(\mathbf{x})\neg P(\mathbf{x})$, then P is false in M′ for all tuples **d** of domain elements. If P's extension axiom in R is

$$(x)P(x) \equiv E(x,c^{(1)}) \vee \cdots \vee E(x,c^{(r)})$$

then P is true in M′ on the tuple **d** of domain elements iff the right side of the equivalence is true in M for the tuple **d**.

Clearly M′ is a model of R. Since $R \vdash W$, W is true in M′. Since W involves only the equality predicate E, and since M and M′ interpret E in the same way, W is true in M.

QED

Definition

A first order formula is *Horn* iff all clauses of its clausal form are Horn. A clause is *Horn* iff it contains at most one positive literal[2] (but any number of negative literals). A first order theory is *Horn* iff all its formulae are.

Notice that E-AXIOMS$_R$ is a Horn theory whenever R is an extended relational theory, a fact we shall make use of shortly.

Lemma 4.4.

Suppose that H is a Horn first order theory, and that K_i is a conjunction of one or more ground atomic formulae for i=1,...,n. Then $H \vdash K_1 \vee \cdots \vee K_n$ iff $H \vdash K_i$ for some i.

Proof.

If H is an inconsistent theory the result is immediate. Hence, assume H consistent. In this case the necessity is obvious. To prove the sufficiency assume, with no loss of generality, that H is in clausal form. Then since $H \vdash K_1 \vee \cdots \vee K_n$, $H \cup \{\overline{K}_1, \ldots, \overline{K}_n\}$

[2] A *literal* is an atomic formula or the negation of an atomic formula.

is unsatisfiable where $\overline{K}_i$ is that clause obtained by negating the conjunct K_i. Notice that each literal of $\overline{K}_i$ is negative so that $\overline{K}_i$ is Horn. Since H is Horn, so is $H \cup \{\overline{K}_1, \ldots, \overline{K}_n\}$. By a theorem of [Henschen and Wos 1974] any unsatisfiable set of Horn clauses has a positive unit refutation i.e. a refutation by binary resolution in which one parent of each resolution operation is a positive unit clause. Since $H \cup \{\overline{K}_1, \ldots, \overline{K}_n\}$ is such a set of Horn clauses, it has a positive unit refutation. Since H is consistent then $\overline{K}_i$ enters into this refutation for some i. Moreover, from the positive unit property, no other negative clause can enter into this refutation so that $H \cup \{\overline{K}_i\}$ is unsatisfiable i.e. $H \vdash K_i$.

QED

Notation

Let R be an extended relational theory. If P is a predicate distinct from E whose extension axiom in R has the form

$$(\mathbf{x})P(\mathbf{x}) \equiv E(\mathbf{x},\mathbf{c}^{(1)}) \vee \cdots \vee E(\mathbf{x},\mathbf{c}^{(r)})$$

then $|P|_R$ denotes $\{\mathbf{c}^{(1)},\ldots,\mathbf{c}^{(r)}\}$. If P's extension axiom in R is $(\mathbf{x}) \neg P(\mathbf{x})$, then $|P|_R$ is $\{\ \}$. $|P|_R$ simply corresponds to the table of the relation P in the usual representation of a relational database. When the theory R is clear from context we simply write $|P|$. We extend this notion to the equality predicate E by defining

$$|E| = \{(c,c) \mid c \text{ is a constant of ALPHA}\}$$

Lemma 4.5

Suppose R is an extended relational theory and P is a predicate, possibly a simple type, possibly E. Then $R \vdash P(\mathbf{c})$ for some tuple of constants $\mathbf{c}$, iff $\mathbf{c} \in |P|$.

Proof.

$\Leftarrow$

Trivial.

$\Rightarrow$

Case 1 P is the equality predicate E.

Suppose $R \vdash E(c,c')$ for constants c,c'. By Lemma 4.3, $\text{E-AXIOMS}_R \vdash E(c,c')$, so that $\text{E-AXIOMS}_R \cup \{\neg E(c,c')\}$ is unsatisfiable. Notice that this is a set of Horn formulae. Now, by a theorem of [Henschen and Wos 1974], any such unsatisfiable set of Horn formulae has a positive unit refutation i.e. a refutation by binary resolution in which one parent of each resolution operation is a positive unit clause. Since E-AXIOMS_R is consistent (Theorem 3.2), $\neg E(c,c')$ must enter into this refutation. But the only positive unit clause of E-AXIOMS_R is $E(x,x)$ so this must be the parent of $\neg E(c,c')$ in the refutation, whence c and c' must be identical constants. Thus $(c,c') \in |E|$.

Case 2 P is distinct from the equality predicate.

Suppose $R \vdash P(\mathbf{c})$. We first prove $|P| \neq \{\ \}$. For if $|P|$ were empty then P's extension axiom in R would be $(\mathbf{x})\neg P(\mathbf{x})$. Since $R \vdash P(\mathbf{c})$, R must be inconsistent, contradicting Theorem 3.2.

Hence, P's extension axiom in R is of the form

$$(\mathbf{x})P(\mathbf{x}) \equiv E(\mathbf{x},\mathbf{c}^{(1)}) \vee \cdots \vee E(\mathbf{x},\mathbf{c}^{(r)}).$$

We must prove that $\mathbf{c}$ is identical to $\mathbf{c}^{(i)}$ for some i. Since $R \vdash P(\mathbf{c})$, then $R \vdash E(\mathbf{c},\mathbf{c}^{(1)}) \vee \ldots \vee E(\mathbf{c},\mathbf{c}^{(r)})$.

By Lemma 3.5

$$\text{E-AXIOMS}_R \vdash E(\mathbf{c},\mathbf{c}^{(1)}) \vee \ldots \vee E(\mathbf{c},\mathbf{c}^{(r)})$$

Since E-AXIOMS_R is a consistent Horn theory, then by Lemma 4.4 $\text{E-AXIOMS}_R \vdash E(\mathbf{c},\mathbf{c}^{(i)})$ for some i. Thus, if $\mathbf{c} = (c_1,\ldots,c_n)$ and $\mathbf{c}^{(i)} = (c_1',\ldots,c_n')$, then $\text{E-AXIOMS}_R \vdash E(c_1,c_1') \wedge \ldots \wedge E(c_n,c_n')$, i.e. $\text{E-AXIOMS}_R \vdash E(c_j,c_j')$ $j=1,\ldots,n$ so that by case 1, c_j and c_j'

are identical constants whence **c** and $\mathbf{c}^{(1)}$ are identical.

QED

The next two theorems provide for the elimination of quantifiers occurring in queries, in favour of the relational algebra operations of projection and division.

Definition

Let S be a set of (n+1)-tuples of constants, and R an extended relational theory. Suppose τ is a simple type. Then the ***division of S by τ (with respect to R)*** is defined only when $|\tau| \neq \{\ \}$ and is

$$\Delta_\tau S = \{\mathbf{a} \mid \mathbf{a}b \in S \text{ for all } b \in |\tau|\}$$

where, when **a** is the n-tuple $(a_1,...,a_n)$, **a**b denotes the (n+1)-tuple $(a_1,...,a_n,b)$.

For example, if

S={(c,d,a), (c,d,b), (c,d,c), (e,e,a), (e,e,b), (e,e,c), (d,e,a), (d,e,b), (c,e,a)}

and $|\tau| = \{a,b,c\}$ then $\Delta_\tau S = \{(c,d),(e,e)\}$.

The division operator defined above is a special case of that of [Reiter 1980a].

Notation

If $\tau = \tau_1,...,\tau_n$ is a sequence of simple types, then $|\tau|$ denotes $|\tau_1| \times ... \times |\tau_n|$. If n=0 then $|\tau|$ is $\{(\ )\}$.

The next theorem allows us to strip off leading typed universal quantifiers in queries.

Theorem 4.6.

If R is an extended relational theory and W(**x**,y) is a (possibly quantified) formula with free variables among $\mathbf{x} = x_1,...,x_n$ and y, then

(a) If $|\theta| = \{\ \}$

$||<x/\tau \mid (y/\theta)W(x,y)>|| = |\tau|$

(b) If $|\theta| \neq \{\ \}$

$$||<x/\tau \mid (y/\theta)W(x,y)>|| = \Delta_i ||<x/\tau, y/\theta \mid W(x,y)>|| \qquad (4.1)$$

Proof:

(a) If $|\theta| = \{\ \}$ then θ's extension axiom in R is $(x)\neg\theta(x)$. Suppose $c \in |\tau|$. Then $R \vdash \tau_i(c_i)$ $i=1,...,n$. Moreover, $R \vdash (y)\theta(y) \supset W(c,y)$ by θ's extension axiom. Hence $c \in ||<x/\tau \mid (y/\theta)W(x,y)>||$. On the other hand, if $c \in ||<x/\tau \mid (y/\theta)W(x,y)>||$ then $R \vdash \tau_i(c_i)$ $i=1,...,n$. By Lemma 4.5 $c_i \in |\tau_i|$ so that $c \in |\tau|$.

(b) Begin by observing that if θ's extension axiom in R has the form

$$(x)\theta(x) \equiv E(x,a_1) \vee ... \vee E(x,a_r)$$

then for a tuple c of constants

$R \vdash (y)\theta(y) \supset W(c,y)$ iff

$R \vdash (y)[E(y,a_1) \vee ... \vee E(y,a_r) \supset W(c,y)]$ iff

$R \vdash (y)E(y,a_i) \supset W(c,y)$ $i=1,...,r$ iff

$R \vdash W(c,a_i)$ $i=1,...,r$ iff

$R \vdash W(c,a)$ for all $a \in |\theta|$.

1. Now suppose c is an element of the left hand side of (4.1). Then

$R \vdash \tau_i(c_i)$ $i=1,...,n$ and

$R \vdash (y)\theta(y) \supset W(c,y)$

This latter implies, by the preamble of this proof, that $R \vdash W(c,a)$ for all $a \in |\theta|$. We must prove c is an element of the right hand side, i.e. that for all $a \in |\theta|$

$ca \in ||<x/\tau, y/\theta \mid W(x,y)>||$ i.e. we must prove

$R \vdash \tau_i(c_i)$ $i=1,...,n$ which is known, and for all $a \in |\theta|$

$R \vdash \theta(a)$ and $R \vdash W(\mathbf{c},a)$, both of which are known.

2. Suppose $\mathbf{c}$ is an element of the right hand side of (4.1). Then for all $a \epsilon \ |\theta|$,

   $\mathbf{c}a \ \epsilon \ ||<\mathbf{x}/\tau, y/\theta \mid W(\mathbf{x},y)>||$ i.e.

   $R \vdash \tau_i(c_i)$ $i=1,...,n$ and for all $a \ \epsilon \ |\theta|$

   $R \vdash W(\mathbf{c},a)$ so that by the preamble of this proof,

   $R \vdash \tau_i(c_i)$ $i=1,...,n$ and

   $R \vdash (y)\theta(y) \supset W(\mathbf{c},y)$ i.e.

   $\mathbf{c}$ is an element of the left hand side.

   QED

Our next task is to strip off leading typed existential quantifiers in queries.

Definition

Let S be a set of (n+1)-tuples of constants. Then the ***projection*** of S is

$$\Pi S = \{\mathbf{a} \mid \mathbf{a}b \ \epsilon \ S \text{ for some constant } b\}$$

For example, if

$$S = \{(a,b,c),(a,b,d),(a,a,c)\}$$

then

$$\Pi S = \{(a,b),(a,a)\}$$

This projection operator is a special case of that of [Reiter 1980a].

Theorem 4.7.

If R is an extended relational theory and $W(\mathbf{x},y)$ is a (possibly quantified) formula with free variables among $\mathbf{x} = x_1,...,x_n$ and y, then

$$\Pi||<\mathbf{x}/\tau, y/\theta \mid W(\mathbf{x},y)>|| \subseteq ||<\mathbf{x}/\tau \mid (Ey/\theta)W(\mathbf{x},y)>||$$

Proof:

Suppose c is a tuple of constants in the left hand side. Then for some constant a, $ca \in ||<x/\tau, y/\theta \mid W(x,y)>||$. Thus

$R \vdash \tau_i(c_i)$ i=1,...,n and

$R \vdash \theta(a)$ and

$R \vdash W(c,a)$

Hence $R \vdash \theta(a) \wedge W(c,a)$ so that $R \vdash (Ey/\theta)W(c,y)$. Hence c is an element of the right hand side.

QED

Unfortunately, the reverse inclusion of Theorem 4.7 fails, as the following example shows:

Example 4.1.

Let R be the extended relational theory having two simple types τ and θ and a binary predicate P, where

$|\tau| = \{\alpha\}$ $\quad |\theta| = \{a,b,c\}$ $\quad |P| = \{(\alpha,a),(\alpha,b)\}$

Moreover, R has a single unique name axiom $\neg E(a,b)$. Consider the query

$$Q = <x/\tau \mid (Ey/\theta)P(x,y) \wedge \neg E(y,c)>$$

Then $||Q|| = \{\alpha\}$ since

$$R \vdash (Ey)\theta(y) \wedge P(\alpha,y) \wedge \neg E(y,c).$$

This is so since

$$R \vdash \theta(a) \wedge P(\alpha,a) \wedge \neg E(a,c) \vee \theta(b) \wedge P(\alpha,b) \wedge \neg E(b,c)$$

even though neither of $\theta(a) \wedge P(\alpha,a) \wedge \neg E(a,c)$ and $\theta(b) \wedge P(\alpha,b) \wedge \neg E(b,c)$ is provable separately.

On the other hand

$$||\langle x/\tau, y/\theta \mid P(x,y) \wedge \neg E(y,c)\rangle|| = \{\ \}$$

The failure of the reverse inclusions of Theorems 4.2 and 4.7 will be seen to be the two sources of the incompleteness of the query evaluation technique of this paper.

Definition:

Suppose for $n \geq 1$ that S is a set of n-tuples of constants. Let $i_1,...,i_k$ be distinct integers in the range [1,n]. Then the ***projection*** of S onto components $i_1,...,i_k$ is

$$\Pi_{i_1,...,i_k} S = \{(a_1,a_2,...,a_k) \mid \text{for some tuple } (b_1,b_2,...,b_n) \in S,\ a_j = b_{i_j} \text{ for } j=1,...,k\}$$

Π, the previous projection operator, abbreviates $\Pi_{1,2,...,n-1}$.

Theorem 4.8.

Suppose R is an extended relational theory, and the variable y does not occur free in the formula W(**x**). Then

(a) $||\langle y/\theta, \mathbf{x}/\boldsymbol{\tau} \mid W(\mathbf{x})\rangle|| = |\theta| \times ||\langle \mathbf{x}/\boldsymbol{\tau} \mid W(\mathbf{x})\rangle||$

(b) If for $n \geq 1$ $\mathbf{x}/\boldsymbol{\tau} = x_1/\tau_1,...,x_n/\tau_n$ and for $k \geq 0$ $\mathbf{z}/\boldsymbol{\varphi} = z_1/\varphi_1,...,z_k/\varphi_k$ then

$$||\langle \mathbf{x}/\boldsymbol{\tau}, y/\theta, \mathbf{z}/\boldsymbol{\varphi} \mid W(\mathbf{x},\mathbf{z})\rangle|| = \Pi_{2,...,n+1,1,n+2,...,n+k}(|\theta| \times ||\langle \mathbf{x}/\boldsymbol{\tau}, \mathbf{z}/\boldsymbol{\varphi} \mid W(\mathbf{x},\mathbf{z})\rangle||).$$

Proof:

Case (b) follows trivially from case (a) by observing that

$$||\langle \mathbf{x}/\boldsymbol{\tau}, y/\theta, \mathbf{z}/\boldsymbol{\varphi} \mid W(\mathbf{x},\mathbf{z})\rangle|| = \Pi_{2,...,n+1,1,n+2,...,n+k}||\langle y/\theta, \mathbf{x}/\boldsymbol{\tau}, \mathbf{z}/\boldsymbol{\varphi} \mid W(\mathbf{x},\mathbf{z})\rangle||$$

Accordingly, we now prove case (a). If $\mathbf{x}/\boldsymbol{\tau} = x_1/\tau_1,...,x_n/\tau_n$ then an (n+1)-tuple of constants $a\mathbf{c} \in ||\langle y/\theta, \mathbf{x}/\boldsymbol{\tau} \mid W(\mathbf{x})\rangle||$ iff

$R \vdash \theta(a)$ and

$R \vdash \tau_i(c_i)$ $i=1,...,n$ and

$R \vdash W(\mathbf{c})$

By Lemma 4.5, $R \vdash \theta(a)$ iff $a \in |\theta|$. Hence

$\mathbf{ac} \in ||<y/\theta,\mathbf{x}/\tau \mid W(\mathbf{x})>||$ iff

$\mathbf{ac} \in |\theta| \times ||<\mathbf{x}/\tau \mid W(\mathbf{x})>||$

QED

Suppose given a query $<\mathbf{x}/\tau \mid W(\mathbf{x})>$. Then we can apply to $W(\mathbf{x})$ the usual validity preserving transformations which replace subformulae of the form $\alpha \equiv \beta$ by $\alpha \supset \beta \wedge \beta \supset \alpha$, and subformulae of the form $\alpha \supset \beta$ by $\neg\alpha \vee \beta$, and which distribute negation inward until the scope of each negation sign is an atomic formula. Hence, with no loss of generality, we can consider only queries of the form $<\mathbf{x}/\tau \mid W(\mathbf{x})>$ where $W(\mathbf{x})$ contains only the connectives $\wedge$, $\vee$ and $\neg$, and where, moreover, the scope of each negation sign is an atomic formula.

By a *primitive* query is meant a query of the form $<\mathbf{x}/\tau \mid P(\mathbf{r})>$ or $<\mathbf{x}/\tau \mid \neg P(\mathbf{r})>$ where $P(\mathbf{r})$ is an atomic formula and all of the variables of $\mathbf{x}$ occur in $\mathbf{r}$.

Using Theorems 4.1, 4.2, 4.6, 4.7 and 4.8 we can now decompose arbitrary queries into appropriate algebraic operations on primitive queries.

Example 4.2.

Suppose

$$Q = <x/\tau \mid (y/\theta)[P(x,y,a,y) \vee (Ez/\varphi)E(y,z) \wedge \neg R(b,z,x)]>$$

Then, provided $|\theta| \neq \{\ \}$

$$||Q|| = \Delta_i||<x/\tau,y/\theta \mid P(x,y,a,y) \vee (Ez/\varphi)E(y,z) \wedge \neg R(b,z,x)>||$$

$$\supseteq \Delta_i\{||<x/\tau,y/\theta \mid P(x,y,a,y)>|| \cup ||<x/\tau,y/\theta \mid (Ez/\varphi)E(y,z) \wedge \neg R(b,z,x)>||\}$$

$$\supseteq \Delta_i\{||<x/\tau,y/\theta \mid P(x,y,a,y)>|| \cup \Pi||<x/\tau,y/\theta,z/\varphi \mid E(y,z) \wedge \neg R(b,z,x)>||\}$$

$$= \Delta_i\{||<x/\tau,y/\theta \mid P(x,y,a,y)>|| \cup \Pi[||<x/\tau,y/\theta,z/\varphi \mid E(y,z)>||$$

$$\cap ||<x/\tau,y/\theta,z/\varphi \mid \neg R(b,z,x)>||]\}$$

$$= \Delta_r\{\|<x/\tau,y/\theta \mid P(x,y,a,y)>\| \cup \Pi[\,|\tau| \times \|<y/\theta,z/\varphi \mid E(y,z)>\|$$

$$\cap\, \Pi_{2,1,3}(\,|\theta| \times \|<x/\tau,z/\varphi \mid \neg R(b,z,x)>\|)]\}$$

Our remaining task, therefore, is to determine algebraic operators for evaluating primitive queries. We focus first on primitive queries of the form $<\mathbf{x}/\tau \mid P(\mathbf{r})>$.

Definitions.

If $\mathbf{r}$ is an n-tuple of variables and/or constants, and S is a set of n-tuples of constants, then

$\Sigma_{\mathbf{r}}(S) = \{\mathbf{t} \in S \mid$ For i=1,...,n, if r_i is a constant, then $t_i = r_i$, and if r_i is a variable, say x, and if $r_{i_1},\ldots,r_{i_k}$ are all of the components of $\mathbf{r}$ such that $r_{i_1} = r_{i_2} = \cdots = r_{i_k} = x$ then $t_{i_1} = t_{i_2} = \cdots = t_{i_k}\}$

$\Sigma_{\mathbf{r}}$ is realizable by the standard selection operator σ of the relational algebra. For example

$$\Sigma_{x,y,a,x,z,y} = \sigma_{1=4 \wedge 2=6 \wedge 3=a}$$

If $\mathbf{r}$ is as above, and if $\mathbf{x} = x_1,\ldots,x_m$ where each x_i is a variable occurring in r, and if $\mathbf{c} = c_1,\ldots,c_m$ where each c_i is a constant, then $\mathbf{r}_{\mathbf{c}|\mathbf{x}}$ is that n-tuple obtained from $\mathbf{r}$ by substituting c_i for each occurrence of x_i in $\mathbf{r}$ for i=1,...,m. For example $(x,y,a,x,z,y)_{(b,c,d) \mid (y,x,z)} = (c,b,a,c,d,b)$.

Theorem 4.9

Suppose that R is an extended relational theory and $<\mathbf{x}/\tau \mid P(\mathbf{r})>$ is a primitive query where $\mathbf{x} = x_1,\ldots,x_n$ and where $\mathbf{r} = r_1,\ldots,r_m$ is a tuple of constants and/or variables. Suppose further for j=1,...,n that r_{i_j} is the first occurrence of x_j in $\mathbf{r}$. Then

$$\|\langle \mathbf{x}/\tau \mid P(\mathbf{r})\rangle\| = |\tau| \cap \Pi_{i_1,\ldots,i_n}\Sigma_{\mathbf{r}}(|P|)$$

Proof:

$\mathbf{c} \in \|Q\|$ iff $R \vdash \tau_i(c_i)$ $i=1,\ldots,n$ and $R \vdash P(\mathbf{r}_{\mathbf{c}|\mathbf{x}})$

iff, by Lemma 4.5, $\mathbf{c} \in |\tau|$ and $\mathbf{r}_{\mathbf{c}|\mathbf{x}} \in |P|$

But $\mathbf{r}_{\mathbf{c}|\mathbf{x}} \in |P|$ iff $\mathbf{r}_{\mathbf{c}|\mathbf{x}} \in \Sigma_{\mathbf{r}}(|P|)$ iff $\mathbf{c} \in \Pi_{i_1,\ldots,i_n}\Sigma_{\mathbf{r}}(|P|)$.

QED

Example 4.9.

$$\|\langle x/\tau, y/\theta \mid P(a,y,y,x)\rangle\| = |\tau| \times |\theta| \cap \Pi_{4,2}\Sigma_{a,y,y,x}(|P|)$$

$$= |\tau| \times |\theta| \cap \Pi_{4,2}\sigma_{1=a \wedge 2=3}(|P|).$$

Definition

Let R be an extended relational theory. An n-tuple $\mathbf{a}$ of constants *disagrees with* an n-tuple $\mathbf{b}$ of constants *(with respect to R)* iff $R \vdash \neg E(\mathbf{a},\mathbf{b})$.

Theorem 4.10

Suppose R is an extended relational theory and $\langle \mathbf{x}/\tau \mid \neg P(\mathbf{r})\rangle$ is a primitive query. Then $\mathbf{c} \in \|\langle \mathbf{x}/\tau \mid \neg P(\mathbf{r})\rangle\|$ iff $\mathbf{c} \in |\tau|$ and $\mathbf{r}_{\mathbf{c}|\mathbf{x}}$ disagrees with every tuple of $|P|$.

Proof:

$\Rightarrow$

Suppose $\mathbf{c} \in \|\langle \mathbf{x}/\tau \mid \neg P(\mathbf{r})\rangle\|$. Then $R \vdash \tau_i(c_i)$ so by Lemma 4.5 $\mathbf{c} \in |\tau|$. Moreover $R \vdash \neg P(\mathbf{r}_{\mathbf{c}|\mathbf{x}})$. But, for any $\mathbf{t} \in |P|$, $R \vdash P(\mathbf{t})$ so that $R \vdash \neg E(\mathbf{r}_{\mathbf{c}|\mathbf{x}},\mathbf{t})$ i.e. $\mathbf{r}_{\mathbf{c}|\mathbf{x}}$ disagrees with $\mathbf{t}$.

$\Leftarrow$

Suppose $\mathbf{c} \in |\tau|$. Then $R \vdash \tau_i(c_i)$. Moreover, if $|P| = \{\,\}$ so that P's completion axiom is $(y)\neg P(y)$ then $R \vdash \neg P(r_{c|x})$ and the result follows. Otherwise $|P| = \{t^{(1)},...,t^{(r)}\}$ for $r \geq 1$ so that P's completion axiom is $(y)P(y) \equiv E(y,t^{(1)}) \vee ... \vee E(y,t^{(r)})$ and since $r_{c|x}$ disagrees with each $t^{(i)}$, $R \vdash \neg E(r_{c|x},t^{(i)})$ whence $R \vdash \neg P(r_{c|x})$ and the result follows.

QED

Definition

Theorem 4.10 suggests the definition of a new algebraic operator, as follows:

Suppose

1. S is a set of m-tuples of constants.
2. x is a tuple of m distinct variables.
3. r is a tuple of $n \geq m$ constants and/or variables where the variables of r are identical to those of x.
4. T is a set of n-tuples of constants.

Define

$D_{x,r}(S,T) = \{c \in S \mid r_{c|x}$ disagrees with every tuple of T$\}$

Then we have the following simple corollary of Theorem 4.10:

Corollary 4.11

Suppose R is an extended relational theory, and $\langle x/r \mid \neg P(r)\rangle$ is a primitive query. Then $||\langle x/r \mid \neg P(r)\rangle|| = D_{x,r}(|r|, |P|)$.

It remains only to specify how to compute $D_{x,r}(S,T)$, i.e. how to determine whether or not two equal length tuples of constants a and b disagree. This is the question whether or not $R \vdash \neg E(a,b)$ for R an extended relational theory. By Lemma 4.3, this is equivalent to determining whether or not E-AXIOMS$_R \vdash \neg E(\mathbf{a},\mathbf{b})$, i.e. whether or not E-AXIOMS$_R \cup \{E(\mathbf{a},\mathbf{b})\}$ is unsatisfiable.

Now $E(\mathbf{a},\mathbf{b})$ abbreviates $E(a_1,b_1) \wedge .. \wedge E(a_n,b_n)$. Moreover, E-AXIOMS$_R$ contains only the reflexive, symmetric and transitive axioms for equality, which define E as an equivalence relation, together with formulae of the form $\neg E(c,c')$ for constants c,c'. Hence there is a simple decision procedure for the unsatisfiability of E-AXIOMS$_R \cup \{E(\mathbf{a},\mathbf{b})\}$ as follows:

1. Determine the equivalence classes under E of $\{a_1,...,a_n,b_1,...,b_n\}$.
2. E-AXIOMS$_R \cup \{E(\mathbf{a},\mathbf{b})\}$ is unsatisfiable (and hence $\mathbf{a}$ disagrees with $\mathbf{b}$) iff some equivalence class contains a pair of constants c,c' such that $\neg E(c,c')$ is one of the unique name axioms of R.

Example 4.4.

Suppose $|P|$ and $|\tau|$ are given by the following tables:

$\|P\|$			$\|\tau\|$
ω	b	ω'	a
b	a	b	b
b	a	a	ω
			ω'

and there is a single unique name axiom $\neg E(a,b)$. Then

$$\|<x/\tau,y/\tau \mid \neg P(x,y,a)>\| = D_{(x,y),(x,y,a)}(|\tau| \times |\tau|, |P|)$$
$$= \{(a,a),(a,\omega),(a,\omega'),(\omega',\omega')\}$$

5. *Complete Query Evaluation: Some Special Cases*

This section treats three special cases for which the query evaluation technique of this paper is not only sound but complete: universally quantified conjunctive queries, positive queries, and databases without nulls.

Notice that there are but two sources of incompleteness of the evaluation algorithm of Section 4. There are Theorems 4.2 and 4.7 which treat disjunction and existential quantification respectively. Hence, in order to prove completeness of the methods of Section 4 in certain special cases, it is sufficient to prove either that Theorems 4.2 and 4.7 are irrelevant to the special case, or that the set inclusion of these theorems may be reversed.

5.1. *Universally Quantified Conjunctive Queries.*

These are queries of the form

$$< x/\tau \mid (y_1/\theta_1)...(y_m/\theta_m) L_1 \wedge \cdots \wedge L_r >$$

where each L_i is an atomic formula, or the negation of an atomic formula. The case $m=0$ is permitted, in which case there are no typed universal quantifiers. It is easy to see that the methods of Section 4 are complete for these queries since such queries involve neither disjunction nor existential quantification.

5.2. *Positive Queries.*

Define the class of ***positive*** wffs as follows:

1. An atomic wff is positive.
2. If K is a positive wff and τ a simple type then $(x/\tau)K$ and $(Ex/\tau)K$ are positive.
3. If K_1 and K_2 are positive wffs, then so also are $K_1 \vee K_2$ and $K_1 \wedge K_2$.
4. A wff is positive only by virtue of 1, 2 and 3.

A query <x/τ | K> is *positive* iff K is positive.

Our objective is to show that the set inclusions of Theorems 4.2 and 4.7 may be reversed in the case of positive queries, from which completeness follows. To that end we shall require

Lemma 5.1.

Suppose that R is an extended relational theory and that W is a wff with a single free variable x. Then

(a) $R \vdash (Ex/\theta)W(x) \equiv \bigvee_{a \in |\theta|} W(a)$

When $|\theta| = \{\ \}$ the right side of this equivalence is the identically false proposition.

(b) $R \vdash (x/\theta)W(x) \equiv \bigwedge_{a \in |\theta|} W(a)$

When $|\theta| = \{\ \}$ the right side of this equivalence is the identically true proposition.

Proof:

(a) Recall that $(Ex/\theta)W(x)$ abbreviates $(Ex)\theta(x) \wedge W(x)$. The result is trivial if $|\theta| = \{\ \}$. Otherwise, suppose θ's extension axiom in R is $(x)\theta(x) \equiv E(x,a_1) \vee \cdots \vee E(x,a_n)$. Then

$$R \vdash (Ex/\theta)W(x) \equiv (Ex)[E(x,a_1) \vee \cdots \vee E(x,a_n)] \wedge W(x)$$

so that

$$R \vdash (Ex/\theta)W(x) \equiv \bigvee_{a \in |\theta|} (Ex)E(x,a) \wedge W(x).$$

By standard properties of equality,

$$R \vdash (Ex)E(x,a) \wedge W(x) \equiv W(a)$$

from which the result follows.

(b) This follows from (a) by noting first that $(x/\theta)W(x)$ is logically equivalent to $\neg(Ex/\theta)\neg W(x)$. Hence, by (a) $R \vdash (x/\theta)W(x) \equiv \neg \bigvee_{a \in |\theta|} \neg W(a)$ so by de Morgan's law

$$R \vdash (x/\theta)W(x) \equiv \bigwedge_{a \in |\theta|} W(a).$$

QED

Lemma 5.2.

Suppose R is an extended relational theory, and K_1 and K_2 are closed positive wffs all of whose quantifiers are typed quantifiers. Suppose further that $R \vdash K_1 \vee K_2$. Then $R \vdash K_1$ or $R \vdash K_2$.

Proof:

Using Lemma 5.1 we can eliminate all typed quantifiers in K_1 and K_2 in favour of disjunctions and conjunctions, to yield quantifier free wffs K_1' and K_2' respectively. Hence $R \vdash K_1' \vee K_2'$, and both K_1' and K_2' are positive since K_1 and K_2 were. Now for each predicate P use P's extension axiom in R to replace every occurrence of P in K_1' and K_2' by equalities, to yield E_1 and E_2 respectively. Thus $R \vdash E_1 \vee E_2$ where both E_1 and E_2 are positive quantifier free formulae in the equality predicate E. By Lemma 4.3, $\text{E-AXIOMS}_R \vdash E_1 \vee E_2$. Without loss of generality, assume E_1 and E_2 are both in disjunctive normal form. Since E_1 is positive then so also is each conjunct C_i in E_1's disjunctive normal form $C_1 \vee \cdots \vee C_m$. Similarly for each conjunct D_i in E_2's disjunctive normal form $D_1 \vee \cdots \vee D_n$. Thus

$$\text{E-AXIOMS}_R \vdash C_1 \vee \cdots \vee C_m \vee D_1 \vee \cdots \vee D_n.$$

Hence by Lemma 4.4, $\text{E-AXIOMS}_R \vdash C_i$ say, for some i. Hence $\text{E-AXIOMS}_R \vdash E_1$ so that $R \vdash E_1$. Since E_1 was obtained from K_1 by a series of logical equivalences, $R \vdash K_1$.

QED

Corollary 5.3.

If R is an extended relational theory and $Q = <\mathbf{x}/\tau \mid W_1(\mathbf{x}) \vee W_2(\mathbf{x})>$ is a positive query then

$$||Q|| \subseteq ||<\mathbf{x}/\tau \mid W_1(\mathbf{x})>|| \cup ||<\mathbf{x}/\tau \mid W_2(\mathbf{x})>||$$

This establishes the reverse inclusion of Theorem 4.2 in the case of positive queries. It remains to prove the reverse inclusion of Theorem 4.7, as follows:

Theorem 5.4.

If R is an extended relational theory and $W(\mathbf{x},y)$ is a positive formula with free variables among $\mathbf{x} = x_1,\ldots,x_n$ and y then

$$||<\mathbf{x}/\tau \mid (Ey/\theta)W(\mathbf{x},y)>|| \subseteq \Pi||<\mathbf{x}/\tau,y/\theta \mid W(\mathbf{x},y)>||$$

Proof:

Suppose $\mathbf{c}$ is an element of the left hand side. Then $\mathbf{c} \in |\tau|$ and $R \vdash (Ey/\theta)W(\mathbf{c},y)$. We must prove $\mathbf{c},a \in ||<\mathbf{x}/\tau,y/\theta \mid W(\mathbf{x},y)>||$ for some $a \in |\theta|$ i.e. that $R \vdash W(\mathbf{c},a)$ for some $a \in |\theta|$. Now since $R \vdash (Ey/\theta)W(\mathbf{c},y)$ then by Lemma 5.1(a), $R \vdash \bigvee_{a \in |\theta|} W(\mathbf{c},a)$. Hence by Lemma 5.2, $R \vdash W(\mathbf{c},a)$ for some $a \in |\theta|$.

Results analogous to ours on the soundness and completeness of algebraic techniques for positive queries are described in [Imilienski 1983] although for him such queries may not involve universal quantifiers or disjunctions. Imilienski has independently adopted a logical framework for addressing the problem of query evaluation over databases with null values. While his notion of a null value agrees with ours - they arise from existential statements - he proposes a different class of first order theories than ours as a formalization of relational databases with null values. Specifically, he provides no representation of the closed world assumption. Accordingly, it is difficult to compare his results with ours.

5.3. Relational Databases Without Null Values

A minimal requirement on the query evaluation method of Section 4 is that it be complete in the absence of null values. This section establishes such a result.

Let us call an extended relational theory R a ***relational theory*** iff for each pair of distinct constants c,c' , $\neg E(c,c') \in R$ or $\neg E(c',c) \in R$. Thus, for relational theories distinct constants are known to denote distinct individuals. This is a standard assumption underlying conventional relational database theory in the absence of null values. Thus relational theories formalize the information content of conventional null-free relational databases. The completeness of the algebraic operators of Section 4 for relational theories will thus be our version of the completeness of the relational algebra in standard relational database theory. We prove this completeness result by proving the reverse inclusions of Theorems 4.2 and 4.7.

Lemma 5.5.

Suppose R is a relational theory, and K_1 and K_2 are closed wffs all of whose quantifiers are typed quantifiers. Suppose further that $R \vdash K_1 \vee K_2$. Then $R \vdash K_1$ or $R \vdash K_2$.

Proof:

The proof is very like that of Lemma 5.2. As in that proof, use Lemma 5.1 to eliminate all typed quantifiers of K_1 and K_2, yielding quantifier free wffs K_1' and K_2' , for which $R \vdash K_1' \vee K_2'$. Then use the completion axioms of R to transform K_1' and K_2' to yield E_1 and E_2 respectively, both of which are quantifier free wffs in the predicate E, and for which $R \vdash E_1 \vee E_2$. By Lemma 4.3, $\text{E-AXIOMS}_R \vdash E_1 \vee E_2$.

With no loss of generality assume E_1 is in disjunctive normal form $C_1 \vee \cdots \vee C_m$ and E_2 is in disjunctive normal form $D_1 \vee \cdots \vee D_n$. Thus $\text{E-AXIOMS}_R \vdash C_1 \vee \cdots \vee C_m \vee D_1 \vee \cdots \vee D_n$. With no loss of generality, assume that $\{C_1,\ldots,C_m, D_1,\ldots,D_n\}$ is a minimal set of conjuncts such that $\text{E-AXIOMS}_R \vdash C_1 \vee \cdots \vee C_m \vee D_1 \vee \cdots \vee D_n$. If $m=0$ it follows that $\text{E-AXIOMS}_R \vdash E_2$ and hence $R \vdash K_2$ and we are done. Hence assume $m \geq 1$. Consider the conjunct C_1. Suppose one of its literals has the form $\neg E(c,c)$ for some constant c, or the form $E(c,c')$ for distinct constants c,c'. Then since R is a relational theory, $\text{E-AXIOMS}_R \vdash \neg C_1$, from which it follows that $\text{E-AXIOMS}_R \vdash C_2 \vee \cdots \vee C_m \vee D_1 \vee \cdots \vee D_n$ contradicting the minimality of $\{C_1,\ldots,C_m, D_1,\ldots,D_n\}$. Thus, no literal of C_1 has the form $\neg E(c,c)$ or $E(c,c')$. This means that every literal of C_1 has the form $E(c,c)$ or $\neg E(c,c')$. But then $\text{E-AXIOMS}_R \vdash C_1$ from which it follows that $R \vdash K_1$.

QED

Corollary 5.6.

If R is a relational theory, then

$$||<x/\tau \mid W_1(x) \vee W_2(x)>|| \subseteq ||<x/\tau \mid W_1(x)>|| \cup ||<x/\tau \mid W_2(x)>||$$

This establishes the reverse inclusion of Theorem 4.2 in the case of relational theories. The reverse inclusion of Theorem 4.7 is the following:

Theorem 5.7.

If R is a relational theory and $W(x,y)$ is a formula with free variables among $x = x_1,\ldots,x_n$ and y then

$$||<x/\tau \mid (Ey/\theta)W(x,y)>|| \subseteq \Pi||<x/\tau, y/\theta \mid W(x,y)>||$$

Proof:

As in the proof of Theorem 5.4, using Lemmas 5.1(a) and 5.5.

Notice that for relational theories the D operator for evaluating negative primitive queries assumes a simpler form:

$$D_{x,r}(S,T) = \{c \in S \mid r_{c|x} \notin T\}$$

This is so because for relational theories two tuples of constants $a_1,\ldots,a_n$ and $b_1,\ldots,b_n$ disagree iff a_i and b_i are distinct constants for some i. We omit the simple proof.

6. *Discussion and Conclusions*

I believe that the main point of this paper is not so much its soundness and completeness results, but rather the *methodology* by which these results were obtained. We began with an abstract logical *specification,* provided by the notion of an extended relational theory, of the semantics of null values for the relational data model. In addition we provided a logical specification of what it means to be an answer to a query. All of this was entirely non procedural. The specification was concerned exclusively with *meaning*. What does a null value mean? What does it mean to be an answer? With such a specification in hand, we could then focus on the problem of *realizing* the specification which in this case was the problem of *computing* answers. This implementation concern lead to the definitions of various algebraic operators, and ultimately to an algorithm for query evaluation based on these operators. Finally this realization was proved sound and sometimes complete with respect to the original logical specification. The most important feature of this approach was having a logical specification to begin with; the rest was more or less routine. The importance for conceptual modelling of such logical specifications is discussed at some length in [Reiter 1983, Section 6].

Now as a methodology for the theory of databases this approach is very general and may be invoked in a wide variety of settings:

1. Database Integrity

We can view any closed first order formula as an integrity constraint. Given the specification of a database as a set DB of first order formulae, we can specify what we mean by the database satisfying its integrity constraints $\{I_1,...,I_n\}$ as follows:

DB *satisfies* $\{I_1,...,I_n\}$ iff $I_1 \wedge \cdots \wedge I_n$ is true in all models of DB which, by the Godel Completeness Theorem is equivalent to $DB \vdash I_1 \wedge \cdots \wedge I_n$ [Reiter 1980b, 1983].

Using this specification one can propose, and prove the correctness of, algorithms for detecting violations of database integrity and for maintaining integrity.

In this connection notice that the results of this paper have a direct application to the detection of integrity violations in relational databases with null values. Suppose R is an extended relational theory and I an integrity constraint. Then $R \vdash I$ iff $\| < \mid I > \| = \{(\,)\}$. (See Section 3.3). Thus the algebraic methods of this paper may be used as follows:

If $\| < \mid I > \| = \{(\,)\}$ then the database satisfies I.

If $\| < \mid I > \| = \{\ \}$ then the database may or may not satisfy I, this because of the incompleteness of the algebraic methods of this paper. However, if I is a universally quantified conjunct, or is positive, then the algebra is complete and the database satisfies I iff $\| < \mid I > \| = \{(\,)\}$.

2. Query Optimization

This paper did not consider methods for query optimization i.e. ways of transforming queries prior to their evaluation in order to improve the efficiency of the evaluation process. Clearly many such transformations are possible, and desirable. For example, when c is a constant one should replace formulae of the form $(Ex/\tau)W(x) \wedge E(x,c)$ by W(c) when $c \in |\tau|$.

When the database has been logically specified, one can then prove the correctness of such optimizing transformations on queries. Specifically, a transformation mapping a query Q to a query Q′ is *correct* iff $||Q|| = ||Q'||$. Such correctness proofs can be particularly important when subtle query transformations are invoked, for example transformations exploiting integrity constraints as in [King 1981].

3. Other Data Models

This paper is concerned with the relational data model. But there are many other data and conceptual models of interest to the database community. The same methodology of this paper may be applied to these other models with the same attendant advantages. Thus the same kinds of results on query evaluation and optimization, and on integrity constraints can be obtained provided these data models are given logical specifications. Moreover, a valuable side effect of such logical formalizations is that the semantics of the corresponding data model is precisely and unambiguously given by the logic [Reiter 1980a, 1983].

Acknowledgement

This work was done with the financial assistance of the National Science and Engineering Research Council of Canada under grant A7642 and the US National Science Foundation under grant MCS-8203954. Many thanks to Ms. Theresa Fong for her patient and careful typing of the manuscript and to David Etherington for unearthing some of the more arcane features of troff for Theresa's benefit.

REFERENCES

Biskup, J. (1981). Null values in database relations, in H. Gallaire, J. Minker and J.M. Nicolas (eds.), *Advances in Data Base Theory, Vol. 1,* Plenum Press, New York, 299-341.

Chang, C.L. and Lee, R.C.T. (1973). *Symbolic Logic and Mechanical Theorem Proving,* Academic Press, New York.

Codd, E.F. (1979). Extending the database relational model to capture more meaning. *ACM Trans. Database Syst. 4,* 4 (December), 397-434.

Henschen, L. and Wos, L. (1974). Unit refutations and Horn sets. *J.ACM 21, 4* (Oct.), 590-605.

Imielinski, T. (1983). Extending relational algebra for logical databases. Technical report, School of Computer Science, McGill University, Montreal.

King, J. (1981). QUIST: A system for semantic query optimization in relational databases. *Proc. VLDB-7,* Sept., 510-517.

Levesque, H.J. (1982). A Formal Treatment of Incomplete Knowledge Bases. FLAIR Tech. Report 3, Fairchild Laboratory for Artificial Intelligence Research, Palo Alto, CA, February.

Reiter, R. (1978). On closed world data bases, in H. Gallaire and J. Minker (eds.), *Logic and Data Bases,* Plenum Press, New York, 55-76.

Reiter, R. (1980a). Equality and domain closure in first order data bases. *J.ACM. 27,* 2, 235-249.

Reiter, R. (1980b). Data bases: a logical perspective. *Proc. ACM Workshop on Data Abstraction, Databases and Conceptual Modelling,* M.L. Brodie and S.N. Zilles (eds.), Pingree Park, Colorado, June 23-26, 174-176.

Reiter, R. (1983). Towards a logical reconstruction of relational database theory. In *On Conceptual Modelling: Perspectives from Artificial Intelligence, Databases, and Programming Languages,* M.L. Brodie, J. Mylopoulos and J. Schmidt (eds.), Springer-Verlag, to appear.

Vassiliou, Y. (1979). Null values in database management: A denotational semantics approach. *Proc. ACM SIGMOD 1979 Int. Conf. on Manage. of Data,* Boston, Mass., May 30-June 1, 162-169.

Walker, A. (1980). Time and space in a lattice of universal relations with blank entries. *XP1 Workshop on Relational Database Theory,* Stony Brook, NY, June/July.

Zaniolo, C. (1977). Relational views in a database system; support for queries. *Proc. IEEE Computer Applications and Software Conf.,* Chicago, Ill., Nov. 8-11, 267-275.