ANSWERING QUERIES IN RELATIONAL DATABASES

Alessandro D'Atri*
Marina Moscarini**
Nicolas Spyratos***

ABSTRACT

This paper concerns query answering in relational databases. We assume a universe U of attributes and a set of values associated with each attribute. A database scheme is a given collection R = {1,2,...,n} of subsets of U, called relation schemes. A query in R is any subset of U. We call "context" any joinable subset of R. An unambiguous context is one in which a query receives the same answer independently of the subcontext used for the computation. A formal treatment of unambiguous contexts is presented and it is shown that they provide a suitable basis for discussing the universal relation assumption, the relationship uniqueness assumption, maximal objects, and other related concepts.

1. INTRODUCTION

This paper concerns query answering in relational databases. In the present section the basic definitions and notation are introduced and the motivations are discussed.

We assume a *universe*, i.e. a set U of *attributes*, that we denote A,B,C,... . A *set of values*, or a *domain*, is associated with each attribute in U. Following the usual practice, we denote subsets of U by juxtaposition of attributes, e.g. BCD denotes the subset {B,C,D} of U. A *relation scheme* on U is any nonempty subset of U. A *database scheme* on U is a specified set of relation schemes, say R = {1,2,...,n}, such that 1 ∪ 2... ∪ n = U. Figure 1 shows a database scheme R = {1,2,3,4} on the universe U = ABCL, where 1 = AB, 2 = BL, 3 = LC, 4 = AC.

* Ist. di Elettrotecnica, Università dell'Aquila, 67100 L'Aquila, Italy
** IASI-CNR, Via Buonarroti 12, 00185 Roma, Italy
*** Université Paris-Sud, Bât. 508, 91405 Orsay, France.

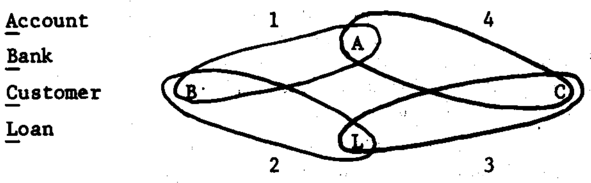

Figure 1. A database scheme.

An *i-tuple*, i ∈ R, is a function that associates with each attribute in i a value in the corresponding domain and an *instance of* i is a set of i-tuples. Following the usual abuse of notation we let i denote both the relation scheme and its current instance. Usually it is clear from the context what i really denotes. The current *instance of* R is the set of current instances of i's, where i ∈ R. Again we let R denote both the database scheme and its current instance. We shall often refer to the set of current instances of the relation schemes in R as the *base relations*.

A *query* on U is any relation scheme on U whose current instance (or current "answer") is computed from the base relations using relational operations. We restrict our attention to two such operations: natural join and projection. This implies that only joinable subsets of R can be used for computing the answers to queries. Such joinable subsets of R will be called contexts.

DEFINITION 1. A subset c of R is a *context* iff for all i and j in c,

- either $i \cap j \neq \emptyset$
- or, if $i \cap j = \emptyset$ then there exist subsets c' and c" of c such that $c' \cap c'' \neq \emptyset$, $i \in c'$, $j \in c''$ and c' and c" are contexts.

Subsets of a context c that are themselves contexts are called *subcontexts*. ■

In Figure 1, the set c = 123 is a context, whereas the set c' = 13 is not (we use juxtaposition of relation schemes to denote subsets of R). In the remaining of this paper we assume that the whole database scheme R is a context. Given a context c, we use the symbol *attr*(c) to denote the set of all attributes appearing in c. For example, in Figure 1, if we take c = 12 we have attr(c) = ABL.

DEFINITION 2. Let c be a context of R and let q be a query. We say that q *is covered by* c (c *covers* q) iff $q \subseteq attr(c)$. We denote C(q,c) the set of all subcontexts of c covering q. ■

To answer a query q we proceed as follows:

(1) select a context c covering q

(2) join the relations in c and then project over q, i.e. $\pi_q(*c)$. Where "*c" denotes the natural join of the base relations in c, and "π_q" denotes projection over q.

Let us note that whereas the current answer to q is computed from the base relations in c, the relation schemes in c provide the interpretation of that answer. For example, consider the query q = BC in Figure 1. The context c = 123 covers q and therefore it can be used to answer q. The (current) answer to q is the relation $\pi_q(1*2*3)$, computed from the base relations 1,2 and 3. The interpretation of the answer, provided by the relation schemes 1,2 and 3, can be worded as follows: "list of banks and customers, such that: each customer has at least one loan in the bank *and* the bank has at least one account".

A context may also be considered as a view, i.e. as a subset of R used by a user as environment for his queries. Under such an interpretation a desirable property of contexts is unambiguity. That is, once a context c is fixed (by the user or by the system) we would like to have unique answers to all queries covered by c, independently of the subcontext used for computing the answer. Let us see an example. In Figure 1, suppose that we fix the context c = 124. The query q = CL is covered by c and receives as answer the relation $\pi_q(1*2*4)$. Note that there is no proper subcontext of c covering q. Consider next the query q' = BC. This query is covered by c and receives as answer the relation $\pi_{q'}(1*2*4)$. But it is also covered by the context c' = 14, a subcontext of c, and receives as answer the relation $\pi_{q'}(1*4)$. In the absence of constraints on R, we may have that $\pi_{q'}(1*4) \neq \pi_{q'}(1*2*4)$. That is the answer to q' depends on the subcontext of c used for the computation. Therefore, the context c is ambiguous with respect to q'.

In the next section a definition of unambiguous context is given and necessary and sufficient conditions for unambiguity are provided.

2. UNAMBIGUOUS CONTEXTS

Unambiguous contexts play a central role in our discussion. So we present first a formal definition.

DEFINITION 3. A context c of a database scheme R is *unambiguous w.r.t. query* q, where $q \subseteq attr(c)$, iff for every c' in C(q,c) and for every instance of R we have: $\pi_q(*c) = \pi_q(*c')$.

A context c is *weakly unambiguous* iff there exists q such that c is unambiguous w.r.t. q; otherwise c is *ambiguous*.

A context c is *unambiguous* iff is unambiguous w.r.t. every $q \subseteq attr(c)$. ■

An immediate consequence of this definition is that all subcontexts of an unambiguous context are also unambiguous. Also, if the database scheme R is unambiguous, then every query $q \subseteq U$ receives a unique answer independently of the context used for computing the answer. In other words the attributes in q are in a "unique relationship" according to the terminology of [2].

First we will consider unambiguity in the case of a scheme without integrity constraints. In order to give a characterization of the set of queries with respect to which the scheme is unambiguous, we need the following definition:

DEFINITION 4. Let c be a context and $q \subseteq attr(c)$; we say that c is a *minimal context* w.r.t. q iff there exists no proper subcontext c' of c covering q. ■

THEOREM 1. Let c be a context in a database scheme R without integrity constraints and $q \subseteq attr(c)$; c is unambiguous w.r.t. q iff c is a minimal context for q.

PROOF. The *if part* is trivial. In order to prove the *only if part* we will show that if there exists a proper subcontext c' of c covering q then c is not unambiguous w.r.t. q (a contradiction). Let us consider, without loss of generality, a subcontext c' = c - {i}; we will prove that there exists an instance of R such that $\pi_q(*c') \supsetneq \pi_q(*c)$. In particular we define an instance s.t. $\pi_q(*c') \neq \emptyset$ and $\pi_q(*c) = \emptyset$. Due to the fact that R is a database scheme without integrity constraints an instance of R exists s.t. $*c' \neq \emptyset$ and the base relation i is empty hence $\pi_q(*c) = \emptyset$. Q.E.D.

In Figure 1, the context c = 123 is ambiguous w.r.t. query BC but it is unambiguous w.r.t. query ABC. Also, the whole of R is ambiguous w.r.t. every query. The following corollaries are immediate consequences of Theorem 1.

COROLLARY 1. Let c be a context in database scheme R without integrity constraints; c is weakly unambiguous iff c is a minimal context w.r.t. attr(c).

COROLLARY 2. Let c be a context in a database scheme R without integrity constraints; c is unambiguous iff c is a singleton.

It follows from Corollary 1 that every non-reduced context is ambiguous. (A context is *reduced* if it does not contain relation schemes i and j such that $i \subseteq j$). It follows from Corollary 2 that database schemes without integrity constraints allow only intrarelational queries to be answered unambiguously (i.e. queries whose context is a single relation scheme of R).

We proceed now to see how our concept of unambiguity is related to the concept of acyclicity. The concept of acyclic database schemes has been introduced in [4] and several properties of such schemes have been studied in the literature. In [3] and [6] properties of a particular kind of acyclicity the γ-acyclicity, have been discussed. We recall briefly the definition. Let c be a context in a database scheme. We say that c is γ*-cyclic* iff there exists:

(1) a subcontext c' of c that contains a pair of relation schemes i,j and

(2) a relation scheme k in c not belonging to c' that contains a pair A,B of attributes such that:
   - A is in i but in no other relation scheme of c'
   - B is in j but in no other relation scheme of c'.

A context c is said γ-*acyclic* if it is not γ-cyclic.

For instance, in the database scheme of Figure 2, the contexts 1245 and 135 are γ-acyclic, whereas the contexts 123 and 2345 are γ-cyclic.

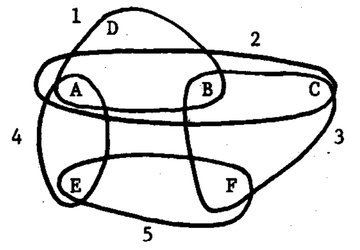

Figure 2. database scheme with γ-acyclic and γ-cyclic contexts.

THEOREM 2. Let c be a γ-acyclic context in a database scheme without integrity constraints; c is weakly unambiguous iff c is reduced.

PROOF. The *only if part* trivially follows from Corollary 1. In order to prove the *if part* let us suppose that c is reduced and that it is not a minimal context w.r.t. attr(c) [hence $|c| \geq 3$].Then there exists a relation scheme i such that $c-\{i\}$ is a context w.r.t. attr(c). Since c is reduced,there is no $j \neq i$ such that $i \subseteq j$. Hence there are attributes A and B in i and relation schemes k and m in c such that:

i) A is in k and A is not in m

ii) B is in m and B is not in k.

It follows that c is a γ-cyclic context, a contradiction, and this completes the proof of the theorem. Q.E.D.

Let us consider now unambiguity in a database scheme with integrity constraints. The existence of constraints may turn an ambiguous context to an unambiguous one. For instance, the context c = 124 of Figure 1, is ambiguous w.r.t. query BC, when there is no constraint. If we introduce the constraint $\pi_B(1) \subseteq \pi_B(2)$ then the context c becomes unambiguous w.r.t. query BC. In particular we are interested in the following question: under what condition a set of integrity constraints guarantees the unambiguity of a given context.

A desirable property of a database instance especially interesting with respect to query answering is the global consistency (*). In [1] an algorithm is described that provides a "minimal" query weakly equivalent with a given query (that is the answers to such queries are equal if they are computed from a globally consistent instance). However if we consider the global consistency as a constraint, we can see that this constraint is not sufficient to guarantee the unambiguity of a scheme, as it is shown in the following example.

(*) *Global consistency* was introduced in [11] in order to characterize database instances R enjoying the following property; there exists a "universal relation" such that all relations in R are obtained by projection from the universal relation.

Let R = 1234 be a database scheme where 1 = ABC, 2 = ABD, 3 = BCE and 4 = DE. Suppose that for every instance of R the set of relations {1,2,3} is globally consistent. Then the instance given in Figure 3 is a legal instance of R, but $\pi_{AC}(1) \neq \pi_{AC}(2*3)$; so the context c = 123 is not unambiguous w.r.t. query AC.

A B C	A B D	B C E	D E
a1 b1 c2	a1 b1 d1	b1 c1 e1	d1 e2
a2 b1 c1	a2 b1 d1	b1 c2 e1	

Figure 3.

We introduce now a stronger concept of consistency and we prove that it provides a necessary and sufficient condition for the unambiguity of a context; then we will relate this concept to the global consistency and pairwise consistency(*) concepts.

DEFINITION 5. A context c is *consistent w.r.t. query* q, where $q \subseteq$ attr(c), iff for every i in c and for every subcontext c' of $c-\{i\}$, such that c' covers q and $p \triangleq i \cap \mathrm{attr}(c') \neq \emptyset$, we have $\pi_p(*c') \subseteq \pi_p(i)$ for every instance of R. A context c is *weakly consistent* iff there exists a query q such that c is consistent w.r.t. q; c is *consistent* iff c is consistent w.r.t. every query $q \subseteq$attr(c).■

LEMMA 1. Let c be a context, consistent w.r.t. query q, and let c' be a subcontext of c covering q. Then:

(1) c' is consistent w.r.t. q

(2) $\pi_q(*c') = \pi_q(*c)$. ■

PROOF. We shall prove the lemma assuming that $c' = c - \{i\}$, for some i in c. Then, in order to prove the lemma for an arbitrary c', all we have to do is to remove the members of c-c', one by one, in a judicious way, so that what is left in each step is a single context. For example, if c = 12345 and c' = 23 (see Figure 4) then we shall remove

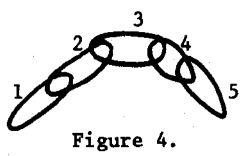

Figure 4.

the members of c-c' = 145, in one of the following orderings: (1,5,4), (5,1,4), or (5,4,1)

(1) Let $c' = c - \{i\}$. If $|c'| = 1$ then c' is con-

(*) A database instance R is *pairwise consistent* iff for all pairs i,j in R we have: $\pi_{i \cap j}(i) = \pi_{i \cap j}(j)$.

sistent. Suppose then that $|c'| > 1$ and that c' is not consistent w.r.t. q. It follows from Definition 5 that there exist j in c', a subcontext c_1 of c' - {j} covering q and an instance of R such that: $\pi_{p_1}(*c_1) \not\subseteq \pi_{p_1}(j)$, where $p_1 = j \cap attr(c_1)$. As c is consistent w.r.t. q and c_1 is a subcontext of c - {j}, it follows that: $\pi_{p_1}(*c_1) \subseteq \pi_{p_1}(j)$ for every instance of R, a contradiction. Therefore c' is consistent w.r.t. q.

(2) It is enough to show that: $(*c') = \pi_{attr(c')}(*c)$ for every instance of R. As c is consistent w.r.t. q, it follows that: for every $i \in c$ and for every subcontext c' of c - {i} covering q and such that $p = i \cap attr(c') \neq \emptyset$, we have $\pi_p(*c') \subseteq \pi_p(i)$ for every instance of R. Therefore, $(*c') = \pi_{attr(c')}(*(c' \cup \{i\})) = \pi_{attr(c')}(*c)$, for every instance of R. Q.E.D.

It follows from the corresponding definitions that every context which is unambiguous w.r.t. a given query q is also consistent w.r.t. q. It follows from Lemma 1 that the inverse also holds, i.e. every context which is consistent w.r.t. a given query q is also unambiguous w.r.t. q. Hence the following theorem and corollaries state that consistency is a necessary and sufficient condition for a context to be unambiguous.

THEOREM 3. Let c be a context and q a query covered by c; c is unambiguous w.r.t. q iff c is consistent w.r.t. q.

COROLLARY 3. A context c is weakly unambiguous iff c is weakly consistent.

COROLLARY 4. A context c is unambiguous iff c is consistent.

But to verify that a given context c is unambiguous, one has to consider all queries covered by c, i.e. all subsets of attr(c). In what follows we show that there is a way to verify unambiguity of c by examining only a proper subset of the set of queries covered by c.

THEOREM 4. Let c be a context; c is consistent (hence unambiguous) iff for every i in c and for every subcontext c' of c-{i}, such that $p \triangleq i \cap attr(c') \neq \emptyset$, we have $\pi_p(*c') = \pi_p(i)$, for every instance of R.

PROOF. The *if part* is trivial.
Only if part. Since c is consistent it follows from Definition 5 that for every $i \in c$ and for every subcontext c' of c - {i}, such that $p \triangleq i \cap attr(c') \neq \emptyset$, we have:

a) $\pi_p(*c') \subseteq \pi_p(i)$ for every instance of R.

Furthermore, from Lemma 1 we know that for every $i \in c$ and for every subcontext c' of c-{i}, such that $p \triangleq i \cap attr(c') \neq \emptyset$, we have:

b) $i = \pi_i(*(c' \cup \{i\}))$, for every instance of R.

Since b) implies that $\pi_p(i) \subseteq \pi_p(*c')$, the theorem is proved. Q.E.D.

We will relate now the concept of unambiguity with the concept of pairwise and global consistency of a relational database.

It follows from Theorem 4 that unambiguity of a context implies global consistency of all its instances. But the example of Figure 3 shows that the reverse implication is not always true. In fact, it is shown in [6] that γ-acyclicity is a necessary and sufficient condition for the following statement to be true: if a database instance is globally consistent then it satisfies the "relationship uniqueness assumption" [2]. As mentioned earlier this assumption simply means that the database scheme R is unambiguous. Furthermore, in [5] it is shown that global consistency is equivalent to pairwise consistency if and only if the database scheme is acyclic(*).
Therefore we can state the following proposition:

PROPOSITION 1. Let c be a context. Then

i) c is unambiguous $\Rightarrow$ c is globally consistent $\Rightarrow$ c is pairwise consistent
ii) c is acyclic iff:
c is pairwise consistent $\Rightarrow$ c is globally consistent
iii) c is γ-acyclic iff:
c is globally consistent $\Rightarrow$ c is unambiguous.

where a context c is said *pairwise* (resp. *globally consistent*) iff every instance of c is pairwise consistent (resp. globally consistent).

3. ANSWERING QUERIES IN AMBIGUOUS CONTEXTS

In this section we discuss query answering in ambiguous contexts. For instance, how do we answer query BC in the context c = 1234 of Figure 1. A possible approach is that the system or the designer provides a unique "interpretation", i.e. a unique context, for the query. Osborn in [10] suggests that when different answers can be obtained by different join sequences (i.e. by different subcontexts), then the union of all valid answers be given as the unique answer. Following this suggestion, query BC in Figure 1 receives as unique answer all pairs of banks and customers such that the customer has either a loan or an account in the bank.

Maier and Ullman in [9] use functional and multivalued dependencies in order to identify a set of contexts that they call "maximal objects". Using these maximal objects they answer a query q as follows. One answer to q is computed from each maximal object covering q. The union of the answers thus computed is *the* answer to q. The problem is that, in the absence of constraints, this method may not produce an answer at all! Indeed, in our previous example of query BC (without integrity constraints) the method produces four maximal objects, namely the four relations themselves. As BC is not contained in any of these maximal objects, there is no way to answer BC!

(*) Let us recall that a context c is *acyclic* iff c can be reduced to the empty set by a sequence of the following operations:
(i) delete an attribute if it belongs to only one relation scheme
(ii) delete a relation scheme i if i is empty or if there exists another relation scheme containing i.
Other equivalent definitions can be found in [5].

Another approach is to assume that the answer to a query q is always obtained by means of projection over q of the join of all database relations [8]. Unfortunately it may be necessary to compute the join of *all* relations [1], evenif we assume the "universal instance assumption" [7] possibly with null values. In fact, as it has been shown in the previous section, this assumption guarantees that for every query q and every context c covering q, the expression $\pi_q(*c)$ provides always the same answer only if the scheme is γ-acyclic. Thus only in this case minimal contexts can be used to answer the query.

All the above approaches to the ambiguity problem assume that there is only one meaningful interpretation of the query. This is not the general case! In fact a query may have more than one meaning. For instance, in Figure 1, the query q = C ("list the customers") may have the following meanings (among others):

(1) "list the customers", having a loan [answer: $\pi_C(3)$]

(2) "list the customers", having an account [answer: $\pi_C(4)$]

(3) "list the customers", having a loan and an account [answer: $\pi_C(3*4)$]

(4) "list the customers", having a loan and an account both in the same bank [answer: $\pi_C(1*2*3*4)$].

In order to introduce our approach let us consider the set C(q,R) of all contexts in R covering q (see Definition 2). Define c $\overset{q}{\equiv}$ c', for all c and c' in C(q,R), iff $\pi_q(*c) = \pi_q(*c')$. That is, c is equivalent to c' w.r.t. q iff q receives the same answer in c and c'. This equivalence relation partitions C(q,R) into equivalence classes. Every equivalence class may be partially ordered with respect to set inclusion. This gives rise to maximal and minimal objects.

DEFINITION 6. Let q be a query on the scheme R. We say that a context c in C(q,R) is a *maximal object* [resp. *minimal object*] w.r.t. q in R iff there exists no c' $\supsetneq$ c [resp. c' $\subsetneq$ c] such that c' $\overset{q}{\equiv}$ c. We denote Max(q,R) the set of all maximal objects w.r.t. q in R [resp., Min(q,R) for the set of all minimal objects].■

As the database scheme R is assumed to be a context we have the following proposition.

PROPOSITION 2.

(i) The database scheme R belongs to Max(q,R), and every minimal context of q belongs to Min(q,R)

(ii) If R is unambiguous w.r.t. q then Min(q,R) coincides with the set of minimal contexts w.r.t. q (in general the reverse implication is not true)

(iii) R is unambiguous w.r.t. q iff Max(q,R) = {R}.

Minimal and maximal objects can be used by the system during the process of query answering in an ambiguous context, in the following way:

1. First, the set of maximal objects w.r.t. query q is proposed to the user (possibly giving also equivalences among them).
2. Once a maximal object is selected (by the user or by the system), the system computes the answer to query q using one of the minimal objects equivalent to the selected maximal object. (The use of minimal objects is done for economy in computation).

4. CONCLUSIONS

We have introduced the concept of a context for query answering. A context determines both an *interpretation* for the query and a *value* (i.e. an answer). If this answer is independent of the subcontext used for the computation then the context is *unambiguous*. Assuming unambiguity of the database scheme itself is equivalent to the relationship uniqueness assumption.

For contexts which are ambiguous w.r.t. a query, we have proposed a systematic way for classifying the unambiguous subcontexts into equivalence classes. Selecting a maximal context of an equivalence class is tantamount to imposing an interpretation to the query. Once the interpretation is fixed, a minimal context of the class can be used for the "economical" computation of the answer.

REFERENCES

[1] Aho A.V., Sagiv Y., Ullman J.D., "Efficient optimization of a class of relational expressions", ACM Trans. on Database Systems, 4; 4, pp. 435-454, December 1979.

[2] Atzeni P., Parker D.S.,"Assumptions in relational database theory",ACM SIGACT-SIGMOD Principles of Database Systems, Los Angeles, March 1982.

[3] Ausiello G., D'Atri A., Moscarini M.,"Minimal coverings of acyclic database schemata",Proc. Workshop on Logical Bases for Data Bases, Toulouse, December 1982.

[4] Beeri C., Fagin R., Maier D., Mendelzon A.O., Ullman J.D., Yannakakis M.,"Properties of acyclic database schemes", Proc. Thirteenth ACM Symposium on the Theory of Computing, pp. 355-362, 1981.

[5] Beeri C., Fagin R., Maier D., Yannakakis M., "On the desirability of acyclic database schemes", Research Report RJ3131, IBM San Jose, California, May 1981, to appear on J. ACM.

[6] Fagin R., "Types of acyclicity for hypergraphs and relational database schemes", Research Report RJ3330, IBM, San Jose, California, September 1982, to appear on J. ACM.

[7] Fagin R., Mendelzon A.O., Ullman J.D., "A simplified Universal Relation Assumption and its properties",ACM Trans. on Database System, 7, 3, pp. 343-360, September 1982.

[8] Korth H.F., Ullman J.D., "SYSTEM/U:A database system based on the Universal Relation Assumption", Proc. XP1 Conf., Stony Brook, New York, June 1980.

[9] Maier D., Ullman J.D., "Maximal objects and the semantics of Universal Relation database", Technical Report 80-016, Stanford University, California, November 1980.

[10] Osborn S.L., "Towards a Universal Relation interface", Proc. Conf. on Very Large Data Bases,Rio de Janeiro,Brasil,pp.52-60,Oct.1979.

[11] Rissanen J., "Theory of joins for relational databases",Proc. Seventh Symp. on Mathematical Foundations of Computer Science, Lecture Notes in Computer Science,Springer-Verlag,pp.537-551, 1978.