Intensional Answers to Database Queries

Amihai Motro

Abstract—**In addition to data, database systems store various kinds of information about their data. Examples are class hierarchies, to define the various data classes and their relationships; integrity constraints, to state required relationships among the data; and inference rules, to define new classes in terms of known classes. This information is often referred to as intensional information (the data are referred to as extensional information). Recently, there have been several independent research works that suggested ways by which intensional information may be used to improve the conventional (extensional) database answers. Although each of these efforts developed its own specific methods, they all share a common belief: Database answers would be improved if accompanied by intensional statements that describe them more abstractly. In this paper, we study and compare the various approaches to intensional answers by using various classifications; we examine their relative merits with regard to key aspects; we discuss remaining issues; and we offer new research directions.**

Index Terms— **Database, database extension, database intension, query, cooperative answer, extensional answer, intensional answer**

I. Introduction

WHEN presented with questions, the responses of humans often go beyond simple, direct answers. For example, a person asked a question may prefer to answer a related question, or this person may provide additional information that justifies or explains the answer. The emulation of human cooperative behavior in man–machine interfaces has been the subject of many studies in artificial intelligence [25].

Traditionally, database systems have been concerned only with providing direct answers to queries, with most efforts being aimed at ensuring such properties as correctness, efficiency, reliability, and convenience. In recent years, however, various research works have demonstrated how to achieve some of the goals of intelligent man–machine interfaces within the framework of database systems.

An important constraint that characterizes these works is that the interface must rely only on information that is normally stored in conventional databases. By "conventional databases," we mean databases that adhere to general purpose data models, such as the relational, logic-based, semantically rich, or object-oriented models. Standard extensions, incorporated into these models for other purposes (e.g., information on keys, integrity constraints, class hierarchies), are often assumed as well, but little else is assumed.

Notable examples of this approach are the various attempts to provide user interfaces to relational databases that achieve logical data independence; i.e., interpret queries that specify only a list of attributes and a condition, without naming the specific relations to which the attributes belong and how the relations should be joined (e.g., [12], [24]). Another example is interfaces that avoid returning empty answers by automatically broadening all queries whose answers are empty [7], [16]. A system based in part on these ideas is FLEX [15], a formal language interface to relational databases designed to service satisfactorily users with different levels of expertise. Using only the definition of the database and various data indices, FLEX interprets each and every input that is presented to it, regardless of its formal correctness. FLEX is also cooperative: It never delivers empty answers without explanation or assistance.

Recently, there have been several independent efforts aimed at enhancing interfaces to conventional databases with yet another intelligent feature, which we shall refer to as the ability to compute *intensional answers*. An intensional answer is a complement of the conventional answer, comprising either a terse description of the answer or various useful statements that concern the answer.

The term "intensional answer" comes from a distinction often made between the *intension* and the *extension* of a database. The intension of a database is the set of definitions of the data structures for the particular database (also called *schema*). The extension of the database is the set of database values that populate these data structures. The term intension also refers to various other elements of the definition of the database, such as statements that express relationships that must be satisfied by every extension (integrity constraints), or statements that define new data structures and their extensions in terms of the basic structures (views or inference rules). Specifically, the intension of a relational database includes the definitions of the base relations, the definitions of views, and the integrity constraints. The intension of a logic-based database includes the definitions of the base predicates, the inference rules, and the integrity constraints. In semantically rich or object-oriented models, the intension includes the definition of the various classes and their associated hierarchies.

A database query is an intensional statement (for example, in the relational model, it is a view). Its answer is the extension of this intensional statement. Although the intensional information in the database is utilized in the processing of each query, answers are composed entirely of extensional information. Consequently, the query itself is often the only intensional

Manuscript received November 19, 1990; revised October 10, 1991. This work was supported in part by the National Science Foundation under Grant IRI-9007106, and in part by the AT&T Affiliates Research Program.

The author is with the Department of Information and Software Systems Engineering, George Mason University, Fairfax, VA 22030–4444 USA; e-mail: ami@kodkod.gmu.edu.

IEEE Log Number 9213352.

characterization of the retrieved set of values of which the user is aware. Still, the intensional information in the database may include additional characterizations of the extensional answer. If this intensional information is derived and retrieved, database answers would gain additional meaning.

Several researchers have recently addressed themselves to the issue of intensional answers. Although all share a common goal, to respond to queries more abstractly by using the intension of the database, the individual approaches are often very dissimilar: They adopt different frameworks (i.e., the data model and its intensional information), they define their intensional answers differently, and they develop their own specific methods for computing them.

The purpose of this paper is to study and compare these recent results. We offer classifications that enable us to place these different works in one general setting, and we examine their performance relative to several key criteria. This evaluation model (the classifications and the criteria) helps us elucidate important distinctions and similarities among the independent works, leading to better understanding of what has already been accomplished, and what still needs to be addressed.

We begin by providing a simple classification method, which allows us to distinguish among the different kinds of intensional answers, and by establishing several key criteria for evaluating the effectiveness of the various approaches (Section II). This general discussion is followed by a closer look at the individual research works, how they fit into the classification, and how they address the key issues (Section III). We then resume the general discussion. We consider various remaining issues, and we suggest several new research directions (Section IV). We conclude with a brief summary (Section V).

II. The Evaluation Model

To evaluate and compare the different works in this survey, we establish a classification method and a set of effectiveness criteria.

A. Classification

The various approaches to intensional answers may be classified according to four fundamental aspects:

1) Data model and intensional information employed,
2) Inclusion of extensional information in intensional answers,
3) Completeness of intensional characterization, and
4) Independence from extensional information.

The first aspect separates the various approaches into three groups. One group (four separate efforts) works within a logic-based model. The intensional information employed consists of the definitions of the base predicates and the inference rules (one work also uses integrity constraints). Another group (two efforts) works within the relational model. The intensional information employed consists of the definitions of the base relations and the integrity constraints. A third group (three efforts) does not adhere to a specific data model. The researchers assume only the availability of a generalization hierarchy of classes and its extension. Such a hierarchy is an essential component of every semantically rich or object-oriented model.

The second aspect distinguishes between two kinds of intensional answers: those that consist of *pure* intensional information, and those that *mix* intensional and extensional information.

Before discussing the other two classifications, we introduce several simple definitions. Let D denote a database, and let P and Q be two queries. Each query is an intensional statement that for a given extension of D, specifies a set of values. We define several relationships among queries.

A query P *contains* a query Q (for a given extension of D) if the extension of P contains the extension of Q. The queries P and Q are *extension-equivalent* (for a given extension of D) if their extensions are equal. For example, in a particular extension, it is possible that a query on the employees who earn over \$30 000 contains a query on the employees who are engineers.

A query Q *implies* a query P if, in every extension of the database D, the extension of P contains the extension of Q. The queries P and Q are *intension-equivalent* if each implies the other. For example, in a database with an integrity constraint that all engineers earn over \$40 000, a query on the employees who are engineers implies a query on the employees who earn over \$30 000.

The third aspect distinguishes between intensional answers that provide a *complete* characterization of the extensional answer, and intensional answers that provide *partial* characterization. A complete characterization is an alternative specification of the given query, whereas a partial characterization provides only additional insights into the nature of the extensional answer.

In our formalization, intensional answers that are complete characterizations include intensional statements that are related to the query via relationships of extension equivalence or intension equivalence, and intensional answers that are partial characterizations include intensional statements that are related to the query via relationships of containment or implication.

The fourth aspect distinguishes between intensional answers that are *independent* of the database extension, and intensional answers that *depend* on the extension. In other words, it distinguishes between intensional answers that are computed only from the intension, and intensional answers that consider the extension as well. Obviously, mixed answers are dependent on the extension. The opposite, however, is not true: Pure intensional answers are not necessarily independent of the extension.

In our formalization, intensional answers that are independent of the extension include intensional statements that are related to the query via relationships of implication or intension equivalence, and intensional answers that are dependent on the extension include intensional statements that are related to the query via relationships of containment or extension equivalence.

Using the last three aspects, the various research works can be classified into six categories of intensional answers (recall that an intensional answer cannot be mixed and independent):

1) Pure-complete-independent,
2) Pure-complete-dependent,
3) Pure-partial-independent,
4) Pure-partial-dependent,
5) Mixed-complete-dependent, and
6) Mixed-partial-dependent.

The survey of the individual works in Section III will be organized according to the first classification and will refer to the six categories derived from the last three classifications. As we shall see, there will be works in four of these categories.

B. Effectiveness

To determine the effectiveness of any method that computes intensional answers, we propose to examine it from five key aspects: completeness, optimality, non-redundancy, relevance, and efficiency. These aspects are discussed below. Note that all of the methods reviewed here are *sound*: They involve terminating algorithms that compute finite answers that are correct with respect to the particular definition of intensional answer.

Completeness: A method is *complete* if it discovers all of the intensional answers that exist. Note the difference between completeness of a method for generating intensional answers (a complete method generates all intensional answers) and completeness of an intensional answer (a complete intensional answer is equivalent to the query).[1]

Nonredundancy: Nonredundancy is concerned with avoiding intensional statements that provide no additional information over the query itself, or its extensional answer, or other intensional statements. For example, an intensional statement that is a rephrasing of the query is redundant. Similarly, an (extension-dependent) intensional statement that is a disjunction of terms of the kind $X = a$, where a is a value of the extensional answer, is redundant. To avoid a conflict between completeness and nonredundancy, completeness may be interpreted as the computation of all nonredundant answers.

Optimality: When multiple intensional answers exist, it is sometimes sensible to define a measure that describes the "goodness" of each answer. A method is *optimal* if it generates the best answer according to the measure. The multiple intensional answers may be simply syntactic variants (i.e., a situation involving redundancies), in which case, an optimal method selects the most desirable (canonical) variant.

Relevance: Relevance is concerned with avoiding intensional statements that have little or no value to the user. For example, a user who inquires about the programmers proficient in Ada may be uninterested in finding out that they all have medical insurance from Prudential. As we shall see, the issue of relevance presents one of the most difficult challenges to the effectiveness of intensional answers. Although nonredundancy and relevance are both concerned with undesirable answers, we shall deal with them separately. As we shall see, redundant answers can be identified accurately and unambiguously, whereas relevance is often a matter of opinion or degree.

Efficiency: Efficiency is concerned with the cost of deriving intensional answers. Although the volume of intensional information is usually much smaller than the volume of extensional information, the processing of intensional information involves more complex algorithms. Efficiency often conflicts with the other four criteria, as attempts to satisfy these criteria may contribute to the complexity of the method.

Note that completeness and optimality are often alternative criteria. When intensional answers are partial characterizations, each intensional answer may contribute a different characterization, and finding all such answers may be important. On the other hand, when intensional answers are complete characterizations, one answer is usually sufficient, and finding the most desirable answer may be important.

The analysis of the individual works in the following section refers to these five criteria.

III. Survey and Analysis

In this section, we review nine research works. These works are discussed in three groups, according to their formal framework. In some examples, we shall use typeface to distinguish between intensional information such as classes, relations, attributes and predicates (e.g., EMPLOYEE), and extensional information (e.g., `John_Smith`). Occasionally, when the example is informal, we shall use normal typeface (e.g., the employee John Smith).

A. Research Within Semantically Rich or Object-Oriented Frameworks

The first group includes works by Corella [6] and Shum and Muntz [21], [22]. As mentioned earlier, these works assume a data model that includes a generalization hierarchy of classes (also referred to as a taxonomy of concepts). Because this structure is an essential component of the semantically rich [9], [18] and object-oriented [11] approaches, the results may be applied in data models that adhere to these approaches.

The following basic definitions are assumed by this group of researchers[2]. Let D be a finite domain of objects. A *concept* is a unary predicate over D (i.e., a subset of D). A *taxonomy* is a tree whose nodes are labeled by concepts. Each concept is subsumed by (i.e., contained in) its parent concept, and the union of all sibling concepts is equal to the parent concept. A taxonomy is *strict* if sibling concepts are all mutually exclusive. A concept (set of objects) is *classifiable* by a taxonomy if it is contained in the root concept.

Corella: Corella [6] notes that though research on knowledge representation produced much work on the derivation of taxonomies of concepts, at times, concepts are also essential in responses to queries. The application is assumed to be *catalogs*: taxonomies of concepts without any extensional information. We assume that the subject of a query is always a concept of the taxonomy, and define an intensional answer to be the labels of the *maximal* concepts that are subsumed by its subject, but are not equal to it. Thus, a query about the concept EMPLOYEE could retrieve the concepts ENGINEER, PROGRAMMER, FEMALE, and so on.

[1] The latter was discussed in Section II-A.

[2] These definitions simplify somewhat those assumed in [6].

Clearly, because there is no extension, all answers are extension-independent and purely intensional. In addition, because of the nature of catalog taxonomies, where concepts are *defined* by their siblings, completeness is guaranteed. This suggests classifying these answers in category 1. One may argue, however, that the bottom level of a catalog should be regarded as the extension of the database, and that in non-catalog applications, concepts are not necessarily "covered" by their siblings. Under these more general assumptions, the intensional answers would be partial, pure, and extension-dependent, and would thus belong to category 4. For the purpose of this survey, where we are considering general purpose databases, and for a meaningful comparisons with other methods, we shall apply Corella's approach to general databases, and therefore use the latter classification.

Shum and Muntz (1): Shum and Muntz also note that answers that are exhaustive enumerations of individual objects are not always the most efficient or most effective means of information exchange. In [21], they are concerned with implicit representation of answers through concise expressions that involve both concepts and individuals. An expression may include concepts and individuals as either *positive* or *negative* terms (i.e., they are either added to the answer or subtracted from it). For example, an acceptable answer to the query, "Who earns over $30 000?" is, "All engineers except John Smith," or, "All engineers and all managers except junior managers." Because they contain extensional information, these answers are mixed and extension-dependent. Obviously, they are complete characterizations. Altogether, they belong in category 5.

The authors note that a query may be answered with several different intensional answers, and the main issue they consider is how to determine which answer is "best." They define an answer as being optimal if it has the smallest number of terms. Among answers with the same number of terms, answers with the maximal number of positive terms are preferred. For taxonomies that are strict, they prove that all optimal answers use the same set of terms, differing only in their order (though not any order constitutes a correct answer), and they describe an algorithm based on postorder traversal of the tree that generates an optimal answer (from which all other optimal answers may be derived via certain permutations). The set of optimal answers is reduced further by considering only answers whose terms are sorted in an order induced by the given taxonomy. All remaining answers are considered equally satisfactory, and an arbitrary answer is presented to the user. For taxonomies that are not strict, optimal answers no longer share the same set of terms, and no efficient algorithm for obtaining such answers can be found (the problem is shown to be NP-complete).

Shum and Muntz (2): In another study [22], Shum and Muntz are concerned with a different kind of intensional answers, based on aggregate expressions. An aggregate expression is a sequence of terms of the kind $r/t\ C$, where C is a concept, t is its total number of individuals, and r is the number of these individuals who belong to the answer. For example, an acceptable answer to the query, "Who earns over $30 000?" is, "90/120 engineers + 20/30 managers." An intensional answer must "cover" the extensional answer, but an individual may be covered by more than one term. This requirement provides some kind of "completeness" of characterization, but it is important to note that this is not quite the same as the completeness of characterization defined in Section II-A, which compared the extension of the intensional characterization with the actual extensional answer. It is not obvious how to evaluate the extension of these aggregate expressions, because each expression corresponds to many different extensional answers.[3] If we relaxed the definition of complete characterization to include situations where *one* of the possible extensions of the characterization was equal to the extensional answer, then these aggregate expressions would be complete characterizations. Although the computation of these answers depends on the extension, the statements themselves (sums of fractions of concepts) may be considered purely intensional. Altogether, these answers may be classified in category 2.

Again, a query may be answered with several different intensional answers, and the main issue considered is how to determine which answer is "best." Two criteria for optimality are recognized: conciseness and preciseness. Conciseness is simply the number of terms in the answer. Preciseness measures the amount of information encapsulated in the expression, and is based on the concept of enthropy, known from information theory. For example, because each extensional answer is always covered by the root concept, a possible answer to the previous query is "110/480 employees." Although this answer is more concise than the former answer to this query, it is less precise (conveys less information). To handle these often conflicting criteria, the authors consider the problem of finding the most precise answer (answer with the least amount of enthropy) for a given expression length. They show an efficient solution to this problem for the restricted case of one-level taxonomies with equal cardinalities for all leaf concepts, and they suggest an algorithm for the general case that appears to give reasonable answers.

Analysis: All three works in this group define intensional answers that are extension-dependent, but whereas Corella's intensional answers are partial characterizations, the two kinds of intensional answers defined by Shum and Muntz are complete characterizations. Of the latter two works, the first achieves completeness by allowing terms that are individuals; the second achieves completeness by allowing terms that are "fractions" of concepts.

The *efficiency* of the methods developed by Shum and Muntz has already been discussed. Because the first work does not discuss specific algorithms for generating intensional answers, the issue of efficiency cannot be addressed.

The set of maximal concepts that are subsumed by a given extensional answer is well defined. Therefore, in the first work, each query has exactly one intensional answer. Thus, any sound method for generating intensional answers is necessarily *complete*. In contradistinction, the other two works allow for multiple intensional answers. Both of these works define

[3] This is also the case with other kinds of statistical summaries.

measures of the goodness of answers, and attempt to achieve *optimality*.

Because each method generates a single intensional statement, the possibility of *redundancy* within the answer does not exist. Another form of redundancy is avoided by insisting that retrieved concepts are strictly subsumed by the subject, thus preventing intensional answers that simply restate the query. It is possible that the optimal intensional answer computed by the second work is simply a sequence of positive terms, each describing an individual. Such answers are redundant, because they simply restate the extensional answer.

How *relevant* are these intensional answers? Corella assumes that the subjects of queries are always drawn from the concepts of the taxonomy. Thus, each intensional answer describes the concept stated in the query with concepts from the same taxonomy. Hence, the degree of relevance is related to the coherence of the taxonomy. In other words, if it can be assumed that taxonomy concepts are all mutually relevant, then intensional answers are always relevant.

From the examples they discuss, it is apparent that Shum and Muntz assume that each concept of the taxonomy has associated attributes, which can be used to define the subjects of queries. Consider this example of a taxonomy that consists of the concept PERSON with the attributes SALARY and HEALTH, and descendant concepts COLLEGE_GRADUATE and VEGETARIAN. The query, "Who earns over $30 000?" could then have two intensional answers: "All vegetarians" and "All college graduates except John Smith." Because the former is more concise, it will be preferred over the latter. However, because education is more relevant to salary than dietary habits are, one may argue that the less relevant answer was preferred.

B. Research Within a Relational Framework

The second group includes works by this author [17] and Chu, Lee, and Chen [5]. In the former work, the formal framework is that of the conventional model of relational databases (including integrity constraints). The concept of *view* is central to this work: A view is an expression in the relation schemes of the database that defines a new relation scheme, and for each database instance, a unique extension. Views are used for expressing queries (customary), and also for expressing integrity constraints (see below). In both cases, the views are defined with selection-projection-product expressions. In the latter work, the relational model is only the ground level; using knowledge acquisition techniques, additional intensional information is inferred from the extension (e.g., generalization relationships and rules), and this information is then used to generate intensional answers.

Motro: The intensional answers described in [17] are derived from known integrity constraints, and characterize extensional answers in two ways: with constraints that apply to the extensional answer, and with database views that are contained entirely in the extensional answer. Consider, for example, a database with relation EMPLOYEE (NAME, TITLE, SALARY, DEPARTMENT), and two constraints: One states that all employees of the design department earn over $30 000, and the other states that all employees in research positions are in the design department. The query, "Who are the employees of the design department?" will be answered extensionally with a list of individuals, and intensionally with two characterizations: "All employees retrieved earn over $30 000," and "All employees in researcher positions retrieved."

The generation of intensional answers is treated as an application of the following more general problem, called the *view inference problem*: *Given a query and a set of database views that possess a particular property, what views of the answer possess this property?* Consider the property of being *empty*. The problem then becomes: Given a query and a set of empty views, what views of the answer are empty? Empty views are statements of constraints. This follows from the fact that every constraint of the form $(\forall x_1)\cdots(\forall x_n)$ $(\alpha(x_1,\cdots,x_n) \longrightarrow \beta(x_1,\cdots,x_n))$, where x_i are domain variables and α and β are safe relational calculus expressions with these free variables, may be rewritten as an empty view: $\{x_1,\cdots,x_n|\alpha(x_1,\cdots,x_n) \wedge \neg\beta(x_1,\cdots,x_n)\} = \emptyset$. Thus, the problem becomes: Given a query and a set of constraints, what are the constraints that apply to the answer?

The author's solution to the general view inference problem is to represent the definitions of the given database views in special relations, using the concept of *meta-tuples*. A metatuple defines a selection-projection view of a single relation, and several metatuples can be used together to define general views (i.e., views with product). All metatuples that define views of the same relation are stored together in one *metarelation* whose structure mirrors the actual relation. Standard algebraic operators (product, selection, and projection) are extended to these metarelations. When a query is presented to the database system, it is performed *both* on the actual relations, resulting in an extensional answer, and on the metarelations, resulting in a *meta-answer*: definitions of views of the answer that inherit the particular properties of the given views.

In this case, where the property is emptiness, the above process discovers views of the answer that are empty. A simple extension to this process infers also views of the database that are *contained* entirely in the answer. Altogether, the intensional answers characterize the extensional answers in two ways: with *constraints*, i.e., views of the extensional answer that are empty, and with *containments*, i.e., views that are contained entirely in the extensional answer. Referring to the classification of Section II-A, these intensional answers are pure, partial, and extension-independent (category 3).

For presentation, a meta-answer is converted into intensional statements about constraints and containments, whose syntax resembles other statements in the query language. For example, the previous query to retrieve the employees of the design department will return an extensional answer in the form of a relation (NAME, POSITION, SALARY), and an intensional answer in the form of two statements:

constrained SALARY $>$ $30 000

contains POSITION= `researcher`

Chu, Lee, and Chen: Chu, Lee, and Chen describe a method by which intensional information is gathered from

the extension of a plain relational database [5]. Among the works surveyed here, this work is therefore unique in that the intensional information used for generating intensional answers is dependent on the extension. This new information allows the system to view its data as being structured in accordance with a model that is an extension of the entity-relationship model [3], with entity sets, one-to-many relationships, generalization relationships, and rules.

For example, assume a relation EMPLOYEE with attributes NAME and POSITION, and a relation POSITION with attributes RANK and LEVEL. Using knowledge of the key attributes, it is possible to infer one-to-many relationships between entity-sets (relations); for example, between POSITION and EMPLOYEE. Also, by selecting specific values for nonkey attributes, it is possible to define new entity-sets that would be related through generalization relationships to existing entity sets; for example, the entity set SENIOR_EMPLOYEE of the employees for whom LEVEL = `senior`. Finally, by observing the behavior of the data, it is possible to infer *rules* that express relationships between the values of attributes; for example, RANK $> 6 \longrightarrow$ LEVEL = `senior`.

A typical query specifies a set of output attributes and a condition on related attributes, for example, "List the names of employees with rank 8," or, "List the names of the employees who are senior." In the first query it can be concluded from the example rule that each employee in the answer is senior. In the second query, it can be concluded from the same rule that the employees with rank greater than 6 are all included in the answer. Thus, rules can be applied in both forward direction (deduction) and backward direction (abduction) to infer intensional statements, such as, "The answer is *contained in* the set of senior employees," or, "The answer *contains* the set of employees with rank greater than 6."[4] Note that a particular query may require the "chained" application of numerous rules, some deductively and some abductively.

These partial characterizations are purely intensional, but are dependent on the extension, and are therefore in category 4. The authors then consider the addition of "rules" that apply only to individuals to "complete" the definitions of subsets, for example, "John is also senior" (though John's rank may be lower than 6). Clearly, if such "rules" are available to complete the definition of all subsets, it is possible to generate intensional answers that are complete; however, these answers would then be mixed (category 5).

A final note on the classification of this work. The use of logic (e.g., the rules and the induction/abduction process) may support classifying this work as "logic-based" (Section III-C). Similarly, the posterior view of the database using generalization relationships may suggest that this work should have been discussed in Section III-A. We prefer, however, to consider the discovery of intensional information as part of the method. Consequently, the results should be viewed as obtained in the framework of conventional relational databases.

Analysis: Although both works start with the relational model, their approach is very different. We consider first the intensional answers generated by manipulating definitions of empty views.

With respect to *efficiency*, the duality with regular query processing guarantees that the cost of deriving intensional answers is essentially the cost of processing the query on the metarelations. Although the method is shown to be sound, it is not necessarily *complete*: There may be additional intensional answers that are not generated by this method.

By definition, meta-answers include only views that can be expressed with the attributes of the extensional answer. Therefore, this method implements the following definition of *relevance*: An intensional statement (constraint or containment) is relevant to a query if it can be expressed with the output attributes. Thus, the constraint that all employees in researcher positions are in the design department is relevant only to queries that inquire about both POSITION and DEPARTMENT. Although this solution may not be entirely satisfactory, it is extremely simple and is usually effective.

Metaprocessing could generate one form of *redundancy*. Although identical property views are removed from the meta-answer as replicated metatuples, the meta-answer may include property views, which are related through containment. This yields intensional statements that are implied by other intensional statements.

The method described by Chu *et al.* involves two computational processes: *inducing* the intensional information (e.g., rules) from the data, and *inferring* the intensional answers from these rules. The authors do not discuss the complexity of either process, so it is difficult to comment on the *efficiency* of their method, but we note that the first process cannot be considered a one-time effort, because induced rules may need to be updated quite often to reflect changes to the database extension. The method is *complete*, in the sense that the inference engine used for deduction and abduction could generate all possible conclusions (intensional statements) from the induced set of rules.

The authors do not address directly the problem of *relevance* of intensional statements to user queries. This problem is particularly crucial here, because of the additional need to determine the intensional information worth inducing. The process of inducing intensional information must also address issues of *redundancy*, because redundancies in the induced intensional information could result in redundancies in intensional answers.

C. Research Within a Logic-Based Framework

The third group includes works by Cholvy and Demolombe [4], Pirotte and Roelants [20], Andreasen [1], and Imielinski [10]. The formal framework is first-order logic. Although the basic definitions differ somewhat from one work to another, they are roughly equivalent to the following model [23].[5]

An *atomic formula* is a predicate name followed by a list of arguments (variables and constants). A *fact* is a predicate name followed by a list of constants. A *rule* is a formula of the form $B_1 \wedge \cdots \wedge B_n \longrightarrow A$, where A and each B_i are atomic formulas

[4] These characterizations are essentially the same as the characterizations by constraints and containments described in Section III-B.

[5] Significant deviations from these assumptions will be noted in the individual surveys.

(A is the *head* of the rule and $B_1 \wedge \cdots \wedge B_n$ is its *body*; variables appearing only in the body are quantified existentially, and all other variables are quantified universally). An *integrity constraint* is a formula of the form $\neg(B_1 \wedge \cdots \wedge B_n)$, where each B_i is an atomic formula (all variables are quantified universally). A *database* $\mathcal{D}$ consists of the following:

- A set $\mathcal{P}$ of *base* predicates and, for each predicate, an associated set of facts of that predicate;
- A set $\mathcal{Q}$ of *built-in* predicates (their associated sets of facts are assumed to be known);
- A set $\mathcal{R}$ of *derived* predicates, and for each predicate, an associated set of rules (each predicate is the head of each of its associated rules); and
- A set $\mathcal{S}$ of *integrity constraints*.

The predicates in $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ are disjoint. The first two sets are referred to as the *extensional database* (*EDB*), and the last two sets are referred to as the *intensional database* (*IDB*). The entire database is understood as collection of axioms (it must be consistent), and the resolution principle is established as the rule of inference. A *query* is a rule whose head predicate is always called Q. The variables that appear only in its head are free. Assuming that Q has free variables $X = (X_1, \cdots, X_n)$, a tuple of constants $a = (a_1, \ldots, a_n)$ belongs to the (extensional) answer to Q, if the substitution of a_i for X_i $(i = 1, \cdots, n)$ yields a theorem.

Cholvy and Demolombe: Cholvy and Demolombe [4] assume a somewhat simpler model consisting of a single set of first-order formulas over a given set of predicates. The formulas are regarded as axioms, and the set must be consistent. Axioms express information considered "invariant." For example, an axiom may declare that "All managers earn over $40 000," but not that "Smith is a manager." Hence, all extensional information is excluded (i.e., the sets of facts associated with the predicates in $\mathcal{P}$ and $\mathcal{Q}$ above). Another important difference is that formulas are not limited to the forms defined earlier.

An intensional answer to a query $Q(X)$ is a set of formulas $A(X)$ such that $(\forall X)A(X) \longrightarrow Q(X)$ is a theorem. For example, the query, "Who earns over $40 000?" is answered intensionally, "All managers." Because an intensional answer must derive the query, but not vice versa, it provides a partial characterization. Obviously, answers are purely intensional and extension-independent. Altogether, they are in category 3.

The authors then sketch the following method for generating intensional answers. By definition, $A(X)$ is an intensional answer, if and only if $(\forall X)A(X) \longrightarrow Q(X)$ is a theorem, or, equivalently, if and only if the negation of $(\forall X)A(X) \longrightarrow Q(X)$ is inconsistent with the axioms. Thus, $A(X)$ is an answer if and only if $(\exists Y)A(Y) \wedge \neg Q(Y)$ is inconsistent with the axioms, or, alternatively, if and only if for some Y, $A(Y)$ is inconsistent with the set comprising the axioms and $\neg Q(Y)$. Assume now that resolution is applied to the set consisting of the axioms and $\neg Q(X)$, and let $R(X)$ be a resolvent. Clearly, for some Y, $\neg R(Y)$ is inconsistent with the set comprising the axioms and $\neg Q(Y)$ (or else $\neg R(X)$ would also be a resolvent). Hence, $\neg R(X)$ is an answer. In summary, the intensional answers generated are negations of resolvents obtained by applying resolution to the axioms and the negation of the query.

Pirotte and Roelants: Pirotte and Roelants [20] follow the general approach of Cholvy and Demolombe. The model they adopt adheres more closely to the model described at the beginning of this section, with two notable exceptions. First, rules are assumed to be nonrecursive. Second, a derived predicate may also have an additional rule associated with it (of a different form), that guarantees that the definition of this predicate is *complete* (i.e., facts not generated by its defining rules are inconsistent with the database). The authors adopt the same definition of intensional answers as Cholvy and Demolombe, and therefore their work, too, is in category 3.

The main thrust of this work is the use of integrity constraints for improving intensional answers. Specifically, intensional answers may be identified as inconsistent (and discarded), or they may be simplified considerably. For example, consider a constraint stating, "All employees earn under $80 000." Assume that the resolution process described earlier generates the intensional answer, "All employees who earn over $90 000." The constraint can be used to identify this answer as inconsistent (i.e., always empty). Consider the intensional answer, "All employees who earn under $90 000." The constraint can be used to transform this answer to the simpler answer, "All employees."

The method developed by the authors begins by generating additional constraints from the constraints in $\mathcal{S}$ (creating some kind of closure). When a constraint and a formula (a rule or an answer) can be resolved successfully, the resolvent is a constraint that is considered relevant to the formula; it expresses a simpler version of the original constraint as it applies to this formula (relevant constraints are similar to the *constraint residues* defined by Chakravarthy, Fishman and Minker [2]; see also Section III-C below). Initially, each rule is associated with a set of relevant constraints. The resolution process for generating intensional answers is then expanded to compute for each answer also the set of relevant constraints. These constraints are then applied to the answer to identify it as inconsistent or to simplify it (as in the above examples). Note that the intensional answers manipulated by this method are always conjunctions of atomic formulas.

Andreasen: Andreasen [1] adapts the basic assumptions and guidelines described by Motro [17] to the logic-based framework defined at the beginning of this section. Again, the derived predicates $\mathcal{R}$ and the integrity constraints $\mathcal{S}$ must be expressed with the base predicates $\mathcal{P}$ or the built-in predicates $\mathcal{Q}$, thus disallowing any recursive definitions.

Like Pirotte and Roelants, Andreasen transforms the given integrity constraints to *constraint residues* [2] that are attached to the predicates of $\mathcal{P}$ or $\mathcal{R}$. Intuitively, a residue is a true statement about the predicate, expressed with the predicate variables. For example, given a predicate *employee(Name, Title, Salary, Department)* and a constraint *employee(Name, Title, Salary, design)* $\longrightarrow$ *Salary* $>$ 30 000 (employees of the design department earn over $30 000), the *employee* predicate is attached the residue *Department* $=$ *design* $\longrightarrow$ *Salary* $>$ 30 000. The computation of these residues results in a so-called *compiled* version of the database.

When a query is presented to the database system, the system:

1) forms the set of all residues attached to the predicates mentioned in the query,
2) expands the set using a theorem prover, and
3) prunes the expanded set for residues considered "relevant" to the query.

The final set of residues is represented as an intensional answer consisting of *constraints* and *containments* (as in [17]): Residues of the kind $R \longrightarrow true$ correspond to containments; all other residues correspond to constraints. As discussed in Section III-B, such intensional answers are pure, partial and extension-independent (category 3).

Imielinski: The model adopted by Imielinski [10] is similar to the model described at the beginning of this section, with three notable exceptions. First, the predicates $\mathcal{R}$ are taken from the predicates $\mathcal{P}$; thus, rules are used to *augment* base relations. Second, though rules are allowed to be recursive, mutual recursion is disallowed. Third, queries are expressed in the relational algebra.

The author argues that rules should be allowed to occur in answers, and defines an answer as a set of facts that satisfy the query, and a set of rules that may be applied to these facts to generate additional facts that satisfy the query. Hence, the structure of an answer is identical to the structure of database itself, with an extensional part and an intensional part. Such answers have both conceptual and computational advantages. As an example, assume a rule that states that employees in the same department must have the same skills, and consider the query, "Who are the employees with programming skills?" This query would be answered by a set of persons (facts) and a rule specifying all those in their departments. The facts in the answer were either present in $\mathcal{P}$ or were derived by the application of other rules. Exhaustive enumeration of this answer may be performed upon request.

The general approach is to "apply" the query to the rule base $\mathcal{R}$, and *transform* the rules that are applicable to the query (would have been involved in the traditional evaluation of the query). The transformed rules form two sets: rules that must be applied "immediately," and rules that may be "postponed." The rules in the first set are applied to the facts $\mathcal{P}$, yielding the extensional part of the answer. The rules in the second set constitute the intensional part of the answer; they may be applied later to this extensional part, to yield the full extensional answer. It should be noted that rule transformation is not always feasible. Hence, for some queries and for some sets of applicable rules, the intensional part of the answer would be empty (i.e., the answer would be purely extensional).

Obviously, these answers are complete (i.e., their extensions are identical to the extensional answers). Because of their extensional component, these answers are mixed and extension-dependent. Note, however, that only the extensional part of the answer depends on the extension; the intensional part (the transformed rules) is computed only from the database intension. In other words, when the database extension changes, only the extensional part of answers needs to be evaluated anew.[6] Altogether, these answers belong in category 5.

Analysis: The work of Imielinski differs significantly from the preceding three works, and we shall discuss it separately at the end. The main difference between the intensional answers generated by Cholvy and Demolombe or by Pirotte and Roelants and those generated by Andreasen is that the former answers are concluded essentially from the inference rules $\mathcal{R}$, whereas the latter answers are concluded essentially from the integrity constraints $\mathcal{S}$.

Although the method for generating intensional answers from rules (described in Section III-C) is sound, it is not necessarily *complete*, because it generates only answers that are in the form of resolvents.[7] The intensional answers generated by Andreasen are complete in the sense that the set of residue constraints for the predicate mentioned in the query had been closed under resolution.

The two methods for generating intensional answers from rules may yield a large number of *redundant* answers. Indeed, both research teams consider their methods as only the first step in the computation of intensional answers, which should be followed by various pruning steps. For example, removal of answers that are syntactic variants of other answers (e.g., answers differing only in their variable names or their order of atomic formulas), removal of inconsistent answers (as described earlier), and removal of answers that are subsumed by other answers. The presence of syntactic variants raises the question of the particular answer that is most desirable. This issue is not addressed directly. On the other hand, the simplification of answers described earlier indicates that in the presence of equivalent answers, shorter answers are preferred. Similar redundancies may also be introduced into the intensional answers generated by Andreasen (e.g., answers that are subsumed by other answers), but this issue is not considered.

With respect to *efficiency*, the method sketched by Cholvy and Demolombe is fairly expensive, because it involves generating all possible resolvents from a given set of formulas. In addition, as each candidate answer is generated, it must be checked for redundancy. Pirotte and Roelants improve the situation through several techniques. First, integrity constraints are excluded from the inference process. Second, the closure of the constraints is generated a priori and stored. Third, the checks for redundancy are not performed anew for each new answer, but the outcome of the checking of an answer is used in the checking of answers generated from it. Similarly, a priori compilation of constraints used by Andreasen reduces the cost of his process (note that recompilation is needed when the rules or the constraints are updated). Most probably, Andreasen's resolution-based theorem proving would be more expensive than Motro's algebraic metaprocessing (though the former may generate answers with additional statements).

Cholvy and Demolombe, and also Pirotte and Roelants, acknowledge that the problem of *relevance* remains largely

[6] This is in contrast with the mixed answers of Shum and Muntz, where the entire answer depends on the extension.

[7] A conjecture is raised in [4] that the answers that are not generated by this method are "not interesting."

TABLE I
CLASSIFICATION OF INTENSIONAL ANSWERS

Authors	Framework	Purity	Completeness	Dependence	Category
Corella	semantic/object	pure	partial	dependent	4
Shum and Muntz (1)	semantic/object	mixed	complete	dependent	5
Shum and Muntz (2)	semantic/object	pure	complete	dependent	2
Motro	relational	pure	partial	independent	3
Chu, Lee, and Chen	relational	pure	partial	dependent	4
Cholvy and Demolombe	logic-based	pure	partial	independent	3
Pirotte and Roelants	logic-based	pure	partial	independent	3
Andreasen	logic-based	pure	partial	independent	3
Imielinski	logic-based	mixed	complete	dependent	5

unsolved, and sketch some possible ways to approach it. Both teams suggest a solution in which a language is defined for each user (by specifying a set of predicates), and only answers expressible in that language are considered relevant to that user. An alternative solution, suggested by Pirotte and Roelants, is to organize the answers in layers according to their level of detail, and present to the user only the most general answers. When the user rejects an answer, it will be used in the generation of additional, more specific answers; when the user approves an answer, that particular avenue will not be pursued any further. Andreasen adopts the definition of relevance used by Motro.

Because the method described by Imielinski defines (whenever feasible) a unique complete answer, issues of method *completeness* or *optimality* are irrelevant. With regard to *efficiency*, the transformation of the rule base does not appear to be very costly. When considering this cost, it must be remembered that in each of the other methods we surveyed, intensional answers are generated separately from extensional answers, whereas in this method, intensional answers may be regarded as an intermediate step toward extensional answers. With regard to *redundancy*, the rule transformation has some commonality with the view manipulations defined by Motro [17] (both are driven by the structure of a relational algebra query), and it could similarly generate rules that subsume one another. The intensional portion of an answer may include rules that involve predicates whose *relevance* is questionable. Imielinski's approach is that predicates that appeared in the query are always relevant (similar to [1], [17]), and that users should specify a priori any other relevant predicates (similar to [4], [20]). A problem related to relevance is *comprehensibility*. Occasionally, the method generates transformed rules that are extremely complex and codified, resulting in intensional statements that do not convey intelligible concepts (such answers may still have computational advantages).

IV. DISCUSSION

The classification of the nine research works is summarized in Table I. Recall that *purity* refers to the absence of any extensional information from the intensional answers, *dependence* refers to the dependence of intensional answers on the database extension, and *completeness* refers to the extent to which intensional answers characterize extensional answers. Thus, intensional answers may be mixed-dependent, pure-dependent, or pure-independent (but not mixed-independent), and each of these may be either partial or complete. Table I inspires several observations.

Although the term "intensional answer" seems to imply an answer that is pure and complete, the near-absence of works in category 1 or 2 suggests that such intensional answers may be unattainable. The only exception is the second work of Shum and Muntz, which was classified in category 2, but only after the definition of completeness was relaxed significantly.

One possible compromise is to abandon completeness, and settle for partial characterizations that are pure (category 3 or 4). Six works have taken this approach. The other possible compromise is to abandon purity, and settle for complete characterizations that are mixed (category 5). The remaining two works have taken this approach. Note that none of the works abandons both purity and completeness (i.e., there are no works in category 6).

In addition to purity and completeness, one may argue that the "ideal" intensional answer should be independent of the database extension. We note that only four methods are extension-independent, and none generates complete answers.

The other five methods are dependent on the database extension, but note that this dependence results from any of three possible causes.

1) The intensional information is gathered from the extension.
2) The intensional answers are derived by locating the extensional answers on the generalization hierarchy.
3) The intensional answers incorporate extensional information to handle "exceptions".

In the latter case, answers are always mixed, whereas in the first two cases, answers may be purely intensional.

A. Relevance

Perhaps the biggest obstacle to the usability and effectiveness of intensional answers is their relevance. Whereas criteria such as completeness, nonredundancy, optimality, and efficiency are usually well defined and quantifiable, relevance is a more elusive criterion.

The intensional answers provided by any of the methods discussed in this paper can be regarded as statements in

a *language* whose *basic vocabulary* is a set of intensional concepts. Thus, the problem of determining whether an answer is relevant to a query is transformed into the problem of selecting the set of relevant intensional concepts. Overall, we have seen three general approaches to this issue. One approach is to assume that the set of relevant concepts includes every concept of the database intension; thus, every intensional answer is relevant. Another approach is to assume that the relevant concepts are those mentioned in the query. A third approach is to assume that the relevant concepts are supplied by the user (either in a predefined "user profile" or through a dialogue). The advantage of the first approach is its ultimate simplicity, but in many respects, it evades the central problem. The third approach provides more accuracy, but also demands user involvement.

Clearly, there is advantage in judging the relevance of answers "automatically" (i.e., without user involvement). But a problem with the second approach is that it is often too restrictive; though it is reasonable to assume that a concept mentioned in the query is relevant, other concepts may be relevant as well. One possibility for addressing this problem, though still avoiding the need to consult the user, is to define a priori *relevance dependencies* among the intensional concepts. The set of concepts relevant to a query would then be the *closure* of the set of concepts mentioned in the query, according to the predefined relevance dependencies. Formally, a relevance dependency $X \longleftrightarrow Y$ implies that whenever the concepts X are relevant, the concepts Y are also relevant, and vice versa. For example, assume the intensional concepts NAME, TITLE, SALARY, ADDRESS and PHONE, and the relevance dependencies TITLE $\longleftrightarrow$ SALARY and ADDRESS $\longleftrightarrow$ PHONE. Thus, salaries are relevant to queries that mention titles, and telephone numbers are relevant to queries that mention addresses. This approach is similar to the concept of *topics*, which are predefined sets of related attributes used in automatic broadening of queries, as part of a cooperative answering mechanism [8]. It is also reminiscent of the concept of *objects*, which are sets of related attributes used in the design of a universal relation interface [13].

B. Inferring Intensional Statements from the Extension

As discussed in Section III-A, Shum and Muntz are concerned with compact representations of the extensional answers through the available hierarchy of classes. Considering only *predefined* classes is somewhat limiting, because *ad-hoc* classes, created through any of the attributes, could be just as effective for intensional answers. For example, the query, "Who earns more than \$30 000?" could be answered intensionally by, "All the employees assigned to project 3382 and Betty." Here the assignment of employees to projects is assumed to be information that is not represented in the class hierarchy, and, if partial characterizations are used, then an intensional answer to the same query would include the observation, "All employees assigned to project 3382."

Thus, their approach could be generalized to discover ad-hoc classes that are related (through containment or equality) to the result. In other words, the intensional answers computed by Shum and Muntz are expressions that contain only the unary predicates that define classes; the intensional answers we propose would be expressions that involve *any* of the database predicates.

In either approach, the answers are dependent on the extension; but in the more general approach, the search is much less restricted. Clearly, the intensional answers should create only those ad-hoc classes that appear to be relevant to the query. (The challenge here is similar to that discussed in Section IV-A.)

This possibility of inferring intensional answers from purely extensional information recalls the work of Chu, Lee, and Chen, discussed earlier. The fundamental difference is that Chu *et al.* discover intensional characterizations of the *entire* database extension, and then proceed to conclude the characterizations that apply to particular queries. (The latter process is similar to most other methods.) The possibility discussed here is to discover characterizations of *specific* extensional answers.

This problem of discovering intensional characterizations in the extension of the database can be stated as follows: Given an extensional database and an extensional answer, find an intensional characterization that holds only on this answer. This is a problem of *knowledge discovery* (or *knowledge mining*), an area that has been attracting much attention recently [19], and is related to issues of machine learning [14].

Finally, the intensional characterizations sought may also be statements that describe any behavior of the data in the answer, which is markedly *different* from their behavior in the entire domain. For example, if the proportion of female employees is in general 40%, but only 10% among the employees who earn over \$30 000, then an intensional answer to the same query would include the observation, "Only 10% of the female employees."

C. Presentation

An issue that we have avoided so far is the communication of the answer to the user. Relatively little effort is required for adequate presentation of extensional answers (e.g., tabulation, sorting, grouping). This is because extensional information is relatively simple, and all users may be assumed to be familiar with its form and meaning. Intensional information, however, is more complex (e.g., rules, constraints, hierarchies, views), and users may not always be assumed to be familiar with its form and meaning. Hence, the presentation of intensional answers may require more effort.

It is reasonable to assume that the user is familiar with the query language that he is using. Therefore, the syntax and semantics of the query language should be adopted for the presentation of intensional answers. However, as we observed in Imielinski's method, this by itself does not guarantee comprehensibility. Some intensional answers may benefit from visual representations. For example, for answers that are essentially new ad hoc classes, the class hierarchy may be displayed, showing the ad hoc class in its proper location; upon request, the user will be presented with either the intensional definition of this class or with its extensional enumeration.

V. Summary

Intensional answers are abstract characterizations of conventional (extensional) answers that are computed from the intension of the database. In this paper, we studied and compared various independent methods for computing intensional answers. We classified the works according to their principal characteristics, and we analyzed their effectiveness by several criteria. Finally, we discussed several remaining issues, and suggested new research directions.

References

[1] T. Andreasen, "Semantic query answering," in N. Prakash, Ed., *Proc. COMAD 90*. New Delhi, India: McGraw-Hill, 1990.

[2] U. S. Chakravarthy, D. H. Fishman, and J. Minker, "Semantic query optimization in Expert systems and database systems," in *Proc. 1st Int. Workshop on Expert Database Syst.*, 1984, pp. 326–341.

[3] P. P. Chen, "The entity-relationship model: Toward a unified view of data," *ACM Trans. Database Syst.*, vol. 1, no. 1, pp. 9–36, Jan. 1976.

[4] L. Cholvy and R. Demolombe, "Querying a rule base," in *Proc. 1st Int. Conf. Expert Database Syst.*, 1986, pp. 365–371.

[5] W. W. Chu, R.-C. Lee, and Q. Chen, "Using type inference and induced rules to provide intensional answers," in *Proc. IEEE Comput. Soc. 7th Int. Conf. Data Eng.* Washington, DC: IEEE Computer Society, 1991.

[6] F. Corella, "Semantic retrieval and levels of abstraction," in *Proc. 1st Int. Workshop on Expert Database Syst.*, 1984, pp. 91–114.

[7] F. Corella, S. J. Kaplan, G. Wiederhold, and L. Yesil, "Cooperative responses to Boolean queries," in *Proc. IEEE Comput. Soc. 1st Int. Conf. Data Eng.*, 1984, pp. 77–85.

[8] F. Cuppens and R. Demolombe, "Cooperative answering: A methodology to provide intelligent access to databases," in *Proc. 2nd Int. Conf. Expert Database Syst.*, 1988, pp. 333–353.

[9] R. Hull and R. King, "Semantic database modeling: survey, applications and research issues," *Computing Surv.*, vol. 19, no. 3, pp. 201–260, Sept. 1987.

[10] T. Imielinski, "Intelligent query answering in rule based systems," *J. Logic Programming*, vol. 4, no. 3, pp. 229–257, Sept. 1987.

[11] W. Kim and F. H. Lochovsky, Eds., *Object-Oriented Concepts, Databases and Application*, Frontier Series. New York: ACM Press, 1989.

[12] D. Maier, D. Rozenshtein, S. Salveter, J. Stein, and D. S. Warren, "Towards logical data independence: a relational query language without relations," in *Proc. ACM-SIGMOD Int. Conf. Management of Data*, 1982, pp. 51–60.

[13] D. Maier and J. D. Ullman, "Maximal objects and the semantics of universal relation databases," *ACM Trans. Database Syst.*, vol. 8, no. 1, pp. 1–14, Mar. 1983.

[14] R. S. Michalski, "A theory and methodology of inductive learning," in R. S. Michalski, T. M. Mitchell, and J. Carbonell, Eds., *Machine Learning: An Artificial Intelligence Approach*. Los Altos, CA: Morgan Kaufmann, 1983.

[15] A. Motro, "FLEX: A tolerant and cooperative user interface to databases," *IEEE Trans. Knowl. Data Eng.*, vol. 2, pp. 231–246, June 1990.

[16] A. Motro, "SEAVE: A mechanism for verifying user presuppositions in query systems," *ACM Trans. Office Inform. Syst.*, vol. 4, no. 4, pp. 312–330, Oct. 1986.

[17] A. Motro, "Using integrity constraints to provide intensional responses to relational queries," in *Proc. 15th Int. Conf. Very Large Data Bases*, Los Altos, CA: Morgan Kaufmann, 1989, pp. 237–246.

[18] J. Peckham and F. Maryanski, "Semantic data models," *Computing Surv.*, vol. 20, pp. 153–190, Sept. 1988.

[19] G. Piatetsky-Shapiro and W. Frawly, Eds. *Proc. IJCAI-89 Workshop on Knowledge Discovery in Databases*, Detroit, MI, 1989.

[20] A. Pirotte and D. Roelants, "Constraints for improving the generation of intensional answers in a deductive database," in *Proc. IEEE Comput. Soc. 5th Int. Conf. Data Eng.*, 1989, pp. 652 659.

[21] C. D. Shum and R. Muntz, "Implicit representation for extensional answers," in *Proc. 2nd Int. Conf. Expert Database Syst.*, 1988, pp. 257–273.

[22] C. D. Shum and R. Muntz, "An information-theoretic study on aggregate responses," in *Proc. 14th Int. Conf. Very Large Data Bases*. Los Altos, CA: Morgan Kaufmann, 1988, pp. 479–490.

[23] J. D. Ullman, *Database and Knowledge-Base Systems*, vol. I. Rockville, MD: Computer Science Press, 1988.

[24] J. A. Wald and P. G. Sorenson, "Resolving the query inference problem," *ACM Trans. Database Syst.*, vol. 9, no. 3, pp. 348–368, Sept. 1984.

[25] B. L. Webber, "Questions, answers and responses: interacting with knowledge-base systems," in *On Knowledge Base Management Systems*. Berlin, Germany: Springer-Verlag, 1986, pp. 353–402.

A. Motro received the B.Sc. degree in mathematical sciences from Tel-Aviv University, Tel-Aviv, Israel, in 1972, the M.Sc. degree in computer science from the Hebrew University, Jerusalem, Israel, in 1976, and the Ph.D. degree in computer and information science from the University of Pennsylvania, Philadelphia, in 1981.

He is currently an Associate Professor of Information Systems at George Mason University, Fairfax, VA, USA. From 1981 until 1990, he was on the faculty of the Department of Computer Science, University of Southern California, Los Angeles, CA, USA. His main research area is data management; in particular, intelligent user interfaces to data and knowledge bases, query languages for data and knowledge, knowledge discovery in databases, uncertainty and imprecision in databases, data integrity and consistency, and integration of multiple information sources.

Dr. Motro is a member of the Association for Computing Machinery and the IEEE Computer Society.