An Approach to Intensional Query Answering at Multiple Abstraction Levels Using Data Mining Approaches

Suk-Chung Yoon
Dept. of Computer Science
Widener University
Chester, PA 19013

E. K. Park
Dept. of Software Architecture
University of Missouri
Kansas City, MO 64110

ABSTRACT

In this paper, we introduce a partially automated method for generating intensional answers at multiple abstraction levels for a query, which can help database users find more interesting and desired answers. Our approach consists of three phases: preprocessing, query execution, and answer generation. In the preprocessing phase, we build a set of concept hierarchies constructed by generalization of data stored in a database and a set of virtual hierarchies to provide a global view of relationships among high-level concepts from multiple concept hierarchies. In the query execution phase, we receive a user's query, process the query, collect an extensional answer, and select a set of relevant attributes to be generalized in the extensional answer. In the answer generation phase, we find the general characteristics of those relevant attribute values at multiple abstraction levels with the concept hierarchies and the virtual hierarchies by using data mining methods. The main contribution of this paper is that we apply and extend data mining methods to generate intensional answers at multiple abstraction levels, which increases the relevance of the answers. In addition, we suggest strategies to avoid meaningless intensional answers, which substantially reduces the computational complexity of the intensional answer generation process. Keywords: Attribute-Oriented Induction, Data Mining, Extensional Answer, Intensional Answer

1. INTRODUCTION

Intensional query answering refers to a mechanism which answers a query by extracting the conditions and/or the characteristics that justify an extensional answer directly retrieved from a database. Previous studies have shown that there are many advantages in intensional answers. For example, intensional answers can provide a more compact and intuitive form than an extensional answer in the form of an enumeration of objects. Summarizing the general features of an extensional answer, an intensional answer can be considered as a kind of interpretation or explanation of the extensional answer. Most of previous studies on intensional query answering have been focused on generating intensional answers at a single level based on integrity constraints and/or rules. However, it is often desirable to generate answers at multiple abstraction levels, which can help database users find more interesting and desired answers. As the size of databases is rapidly growing, an intensional answer at a single level might be either too general or too specific to contain enough informative and meaningful information. We believe we need much more sophisticated answer-finding schemes in intensional query answering in order to achieve a goal of intensional query answering to provide a significant and meaningful description of an extensional answer to users.

In recent years, there has been an emerging area, called data mining or knowledge discovery, that addresses the problems in finding implicit, previously unknown, and potentially useful patterns from large databases [1]. Data mining or knowledge discovery has been the subject of intense research and development. Several tools and methods have been developed. We believe some of those techniques can be applied to intelligent query answering mechanisms in database systems. This motivates the development of mechanisms for intensional query answering at multiple abstraction levels using data mining techniques. In this paper, we introduce a partially automated method for generating intensional answers at multiple abstraction levels by applying current data mining techniques. So, we can generate intensional answers at different levels for a query in database systems, which enhances the usefulness and flexibility of the database systems.

Our approach consists of three phases: preprocessing, query execution, and answer generation. In the preprocessing phase, we build a

set of concept hierarchies [2] constructed by generalization of data stored in a database and a set of virtual hierarchies to provide a global view of relationships among high-level concepts from multiful concept hierarchies. In the query execution phase, we receive a user's query, process the query, collect an extensional answer, and select a set of relevant attributes to be generalized in the extensional answer. In the answer generation phase, we characterize the features of those relevant attribute values at multiple abstraction levels with the concept hierarchies and the virtual hierarchies by using data mining methods, especially the inductive and attribute-oriented method introduced in [2]. In our approach, users can follow certain interesting paths to generate more generalized intensional answers or more specialized intensional answers. In addition, we suggest strategies to avoid certain meaningless intensional answers, which substantially reduces the computational complexity of the intensional answer generation process. The main contribution of this paper is that we developed a framework for generating intensional answers at multiple abstraction levels by applying data mining methods, which increases the relevance of the answers. Our approach provides a simple and efficient way of generating such intensional answers.

The rest of the paper is organized as follows. Section 2 introduces motivating examples to show the advantages of intensional answers at multiple abstraction levels. Section 3 surveys related works on both intensional query answering and data mining. Section 4 presents our approach to derive intensional answers at multiple levels using data mining techniques. Section 5 discusses our conclusions and possible extensions of our work for future research.

2. MOTIVATING EXAMPLES

The following examples illustrate the advantages of intensional answers at multiple levels.

Example 1 Suppose there is a database of car sale transactions. Now, we may have a query asking ``who are typical buyers of sports cars ?''. The query may be answered with an intensional answer as follows: ``typical buyers of sports cars are people whose age is between 25 and 30, whose income is greater than $55000, whose address contains a zip code between 19015 and 19025''. Such an intensional answer might be relevant to a user like a marketing manager. However, the answer may be uninteresting to a user like a senior executive who needs more general information. If a database system can support intensional query answering at multiple levels, the system can generate another intensional answer as follows: ``typical buyers of sports cars are *young* people with a *high* income who live in *suburb* areas'', which is more informative and meaningful to the senior executive.

Example 2 Let us consider a database of staffs, faculty, and students in a university. A manager at the computing center may issue a query asking ``what are the popular web sites ?''. An intensional answer to the query might be a long list of web sites such as: ``the popular web sites are entertainment, ..., music, and investment''. Assume that the manger wants a more detailed intensional answer. If a database system can support intensional query answering at multiple levels, the system can derive another intensional answer which may be more specific as follows: ``movies and films, home videos, television..., modern music, classical music, stocks, and mutual funds'' , which is more informative and meaningful to the manager.

Example 3 A user may ask a query like ``what kinds of cars can be bought with the price range between $15,000 and $20,000 ?''. The query may be answered with an intensional answer such that ``Any Van''. If a database system can support intensional query answering at multiple levels, the system can generate another intensional answer which distinguishes the features of the answer from some other concepts in the database as follows: ``no sports car, all kinds of mini van, all kinds of full-size van, no luxurious sedan, etc.''.

As you see in the above examples, intensional query processing at a single level may not be very useful in some cases. Therefore, we need to generate intensional answers at multiple levels to provide database users with more informative and meaningful intensional answers. In this paper, we suggest a method to generate more generalized or more specialized intensional answers. We believe intensional answers at multiple levels may be used in many ways. For example, the generalized intensional answer in the example 1 may be a valuable resource used for a competitive advantage in decision support and market strategy. The specialized intensional answer in the example 2 may be useful in designing web pages for people in the university.

3. RELATED WORK

In this section, we survey both areas of intensional query answering and data mining, which are related to our work.

3.1 Research on Intensional Answers

Recently, there have been several research works to suggest ways to generate intensional answers in various data models including relational and logic-based models [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 18, 19] . Those works share a common goal to provide an abstract description of extensional answers. Most works have been focused on using integrity constraints and/or rules. In this section, we introduce some approaches directly related to our work. Cholvy and Demolombe [3] studied the idea of having a set of formulas as an answer set in logic-based databases. They provided answers, which were independent of facts and valid in all the states, by negating resolvents obtained by applying resolution to axioms and the negation of a query. Their answers were a set of formulas defining the conditions that a given tuple must satisfy to be an answer for a given query. Their approach allowed only nonrecursive rules to ensure that the algorithm for generating answers was terminated. Based on the research by Cholvy and Demolombe [3], Pascual and Cholvy [8] developed an improved algorithm to deal with only Horn clauses to avoid numerous tests while generating the answer set. Thus, the latter approach accelerated the resolution process. However, the detailed steps to remove redundant resolution steps or meaningless answers were not addressed in Pascual and Cholvy [8]. These are important steps to reduce intensional query processing time. Song and Kim [11] suggested ideas to solve the problems left on Pascual and Cholvy's approach. They discussed a three-step intensional query processing scheme based on resolution and used the notions of relevant literal and relevant clauses to avoid generating certain meaningless intensional answers. Yoon and Song [12] generalized Song and Kim's approach and made their ideas efficient by using a graph structure, called a rule graph, as a compiled structure facilitated search through the intensional database rules in the resolution step by avoiding repetition each time a query was given. Pirotte et al. [10] followed Cholvy and Demolombe's approach, but used integrity constraints to filter out inadequate answers. Imielinski [4] introduced a new concept of an answer for a query. The intensional answers in his approach were expressed in terms of a set of facts and a set of rules built from projection, join and selection, so that they could be incorporated into the answer of a query. Motro [5] suggested ideas to derive intensional answers from known integrity constraints in relational databases.

Our approach takes a fundamentally different approach from those discussed so far. Most of previous studies on intensional query answering have been focused on generating intensional answers at a single level whereas our approach can generate answers at multiple abstraction levels. To provide intensional answers, our approach considers extensional answers whereas their approaches consider integrity constraints and/or deduction rules. That is, we generate intensional answers based on extensional answers. In that sense, our approach is a data-driven approach while other approaches are integrity constraints and/or rules-driven ones. Previous work uses resolution to compute intensional answers, whereas our approach uses data mining tools.

3.2 Research on Data Mining

Data mining or knowledge discovery, which facilitates understanding large amounts of data by discovering interesting patterns, has received a great deal of attention over the past few years. Many methods for the discovery of various kinds of knowledge, including association, characterization, and classification, have been proposed in the context of relational database systems [1, 2, 13, 14, 15, 16, 17]. Most of data mining techniques have been applied in the areas of decision support and market strategies. In addition to those applications, there are other applications that would benefit from the use of the data mining techniques. One of those applications is intelligent query answering. In this paper, we propose a new application area of data mining techniques for intelligent query answering. We discuss some work related to our approach. Han et al. [2] introduced a method for discovering knowledge by building the generalization hierarchy of attribute values. The attribute domains were defined in the given generalization hierarchy and, in turn, used to generalize the attribute values in a table. The strong assumption was the user's predefined domain hierarchy, called conceptual hierarchy, on which the generalization depends. Their approach starts from the assumption that the hierarchy is existed. We apply and extend their approach developed for data mining in relational

databases to the process of generating intensional answers at multiple levels. In addition, we suggest strategies to avoid certain meaningless intensional answers.

4. OUR APPROACH

In this section, we present our approach to generate intensional answers at multiple abstraction levels. We divide our approach into three phases: preprocessing, query execution, and answer generation. The preprocessing phase is done statically once while the remaining two phases have to be performed at run time. We now consider each phase in detail.

4.1 Preprocessing Phase

In this phase, we perform two steps. First, for each attribute defined in a database schema, we build a concept hierarchy to define generalized relationships for the values in the attribute domain. It may not be possible to build a concept hierarchy for every attribute. Especially, if an attribute satisfies one of the following two cases, we can not build a concept hierarchy for the attribute.

Case 1: there is a large set of distinct nonnumeric values of an attribute, but any meaningful high-level concepts for those values are not existed.

Case 2: there is a large set of distinct numerical values, but finite number of meaningful intervals[a range of values] based on the distribution of the attribute values are not existed.

A concept hierarchy can be constructed for an attribute if the attribute domain can be divided into groups and each group can be represented by a meaningful concept. The concept hierarchy can be constructed with knowledge provided by domain experts. The concept hierarchy forms a taxonomy of concepts in a set of layers from most specific value to most general concept. In each layer except bottom layer, there are nodes labeled by concepts. The bottom level of the hierarchy has the specific values of an attribute stored in a database. The nodes in different layers contain concepts at different levels of abstraction. A node in a higher layer stores general information extracted from nodes in lower layers. That is, a node in a high layer represents more general concept and covers more cases than a node in a lower layer. The concept in a node is subsumed by the concept of its parent node.

Example 4 Suppose that there is a database of car sale transactions. In the database, we have an attribute, say buyer-name. We can not find any high-level concepts for the values in the attribute. So, there is no concept hierarchy for the attribute. On the other hand, we can easily build a concept hierarchy of the attribute Model as shown in Figure 1. For example, Dodge Caravan belongs to Mini van, then Van, and then Any Car, which forms a concept hierarchy.

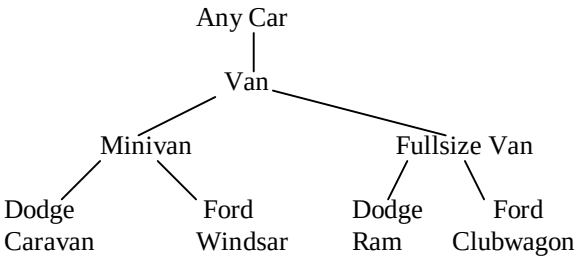
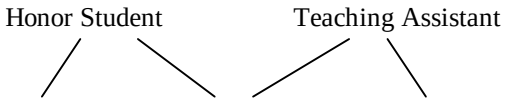


Figure 1. Concept Hierarchy for the Attribute Model

When we build a concept hierarchy, we may use a threshold value to control the number of levels in the hierarchy. The threshold value can be specified by a user or a domain expert to provide the upper bound on the number of levels.

Second, we build a set of virtual hierarchies to provide a global view of generalization relationships among high-level concepts from multiple concept hierarchies. The relationships might be meaningful and interesting to users. The relationships are not explicitly present in concept hierarchies. The node in the virtual hierarchy can be formed by joining two or more high-level concepts from concept hierarchies.

Example 5 Suppose that there is a database of students in a university and there exist a concept hierarchy for the attribute Class and a concept hierarchy for the attribute GPA. We can build a virtual hierarchy to show interesting relationships between high-level concepts in those two concept hierarchies [see Figure 2]. For example, we can define a node whose label is *Honor Student* by combining the concept *Undergraduate* in the Class concept hierarchy and the concept *Excellent* in the GPA concept hierarchy. The node says that a honor student is an undergraduate student whose GPA is excellent. Similarly, we can define another node *Teaching Assistant* in the virtual hierarchy by combining the concept *Graduate* in the Class concept hierarchy and the concept *Excellent* in the GPA concept hierarchy.



Undergraduate Excellent Graduate

Figure 2. A Virtual Hierarchy

The concept hierarchies and the virtual hierarchies include background information that is used to direct the generalization process performed in the answer generation phase to generate intensional answers at multiple level. This phase is independent of queries posed to a database and hence is computed once prior to the processing of any query. Our approach tries to minimize the number of operations performed at run time.

4.2 Query Execution Phase

In this phase, we perform three steps. First, we receive a user's query, process the query, and collect the extensional answer for the query. The extensional answer consists of a set of tuples that satisfy specific conditions in the query. When we process a query, we use available query optimization techniques in databases. Thus, this step can be done efficiently.

Second, we identify attributes that are not relevant to our task in the extensional answer and then eliminate those attributes. We apply the following strategy to find those attributes removed.

Strategy: if an attribute contained in an extensional answer does not have its corresponding concept hierarchy, we eliminate the attribute from the extensional answer. We believe the attribute does not play a role in forming intensional answers because the attribute does not provide any additional information over the extensional answer. For example, if a key attribute is included in an extensional answer, it should be removed. Our purpose of this step is to eliminate meaningless attributes in forming intensional answers.

Third, if the extensional answer still contains a large set of attributes, we select a set of the relevant attributes to be generalized in the answer generation phase among those attributes. The selection can be based on a user's preference or some selection standards as follow:

Strategy 1: we choose attributes that have meaningful correlations. For example, when we form an intensional answer, information showing relationship between experience and salary is more useful than information showing relationship between experience and name.

Strategy 2: we choose attributes that have occurred frequently in the past queries. Frequently referenced attributes can be determined by the analysis of the statistics of query history. We believe that the values in those attributes can be informative and meaningful in forming intensional answers.

Strategy 3: we choose infrequently updated attributes. We believe intensional answers derived from those attributes are valid for a longer period of time. The extensional answer in which nonrelevant attributes are eliminated is passed to the next phase for deriving intensional answers.

4.3 Answer Generation Phase

In this phase, we perform three steps. First, we perform the substitution process which is described as follows.

For each relevant attribute,

1. we generalize each specific value in the attribute into its corresponding higher-level concept by ascension of its concept hierarchy one level. That is, we replace the value by its immediately corresponding concept in the concept hierarchy.
2. check if there exists a high-level concept in the concept hierarchy which replaces all values of the attribute. If any, terminate the substitution process for the attribute. Otherwise, generalize the high-level concepts in the attribute into much higher-level concepts.
3. repeat the step 2.

During this process, a user can provide a threshold value to control the number of distinct concepts in each relevant attribute. In that case, we repeat the substitution process until we find the maximal concept that subsumes all values in the attribute or the number of distinct concepts is not greater than the threshold value. That is, for each relevant attribute contained in an extensional answer, we repeatedly substitute the lower-level concept with its corresponding high-level concept until we find a common concept that is satisfied by all values in the attribute or the number of distinct concepts is less than or equal to the threshold value. This step transforms a less generalized concept into a more generalized concept. The concept hierarchies assist in generalizing lower-level concepts to higher-level concepts. The attribute values are replaced by the concepts which they belong. If there are identical tuples, then merge them into one tuple and store the total count of such merged tuples. The count can be used to calculate the number of the tuples in an extensional answer which are contained in the intensional answer versus the number of tuples in the extensional answer, if necessary. After this process, we may delete tuples with weak support to prevent the system from generating an intensional answer to characterize exceptional cases.

Second, we generate an intensional answer. If there is only one tuple in the final result, the intensional answer is represented as the conjunction of maximal concepts in the tuple. We may not generate an intensional answer which have 100% confidence because there may exist some different cases. In this case, each tuple is transformed into a conjunction of maximal concepts in the tuple. Then, each conjunction is combined into a disjunction which represents the intensional answer. The intensional answer subsumes its corresponding extensional answer and summarizes the general characteristics of the extensional answer.

Third, if a user is not satisfied with the intensional answer, the user can adjust the level of the answer to find more interesting and desired answers. Our approach makes it possible to follow certain interesting paths to generate more generalized answers or more specialized ones. That is, a user can get another intensional answers at different abstraction levels. Our approach can derive those intensional answers as follow:

Case 1: a user requests more general and abstract intensional answers.

Check if there exist any concepts in the normal intensional answer which can be generalized to slightly higher concepts in their concept hierarchies. If any, we generalize the concepts in the intensional answer to slightly higher concepts and generate an intensional answer with the higher concepts. Otherwise, check if there exist any concepts in the virtual hierarchies that are matched to conjunctions of maximal concepts in the normal intensional answer. If any, we replace those maximal concepts by the concepts found in the virtual hierarchies and generate an intensional answer with the concepts.

Case 2: a user requests less general and more specific intensional answers.

We perform back substitution, which is a process of replacing maximal concepts in the normal intensional answer by their corresponding specific conditions that define those concepts. That is, we specialize each concept in each attribute into its corresponding lower-level concepts by descending its concept hierarchy one level at a time.

Our approach can generate intensional answers at different abstraction levels, which allow users to find interesting answers according to their needs.

Example 6 This example shows the application of our approach and algorithm introduced in previous sections. Suppose we have the following database of car buyers [Table 1].

Name	Type-of-Car	Age	Income
Miller	sports	28	58000
Fisher	mini van	37	45000
Johnson	sports	33	60000
:	:	:	:
Smith	truck	58	70000
Wallace	sports	30	65000

Table 1. Car-Buyers

Suppose that we have a query, ``who are typical buyers of sports cars ?''. From the given database, the query is answered extensionally with the tuples in the table given below. The extensional answer is collected into one table [Table 2].

Name	Type-of-Car	Age	Income
Miller	sports	28	58000
Johnson	sports	33	60000
Wallace	sports	30	65000

Table 2.

Our later steps are focused on the table. Now, we identify attributes that do not have concept hierarchies. The attribute Name doesn't have the concept hierarchy. So, the values in the attribute can not be replaced by higher-level concepts. It implies that we can not derive a common high-level concept that subsumes all the values of the attribute Name. So, we eliminate the attribute. Assume that we have the following two concept hierarchies[see figures 3 and 4] for Age and Income, respectively.

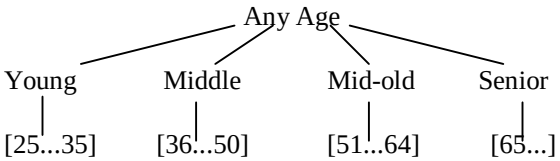


Figure 3. Concept Hierarchy for Age

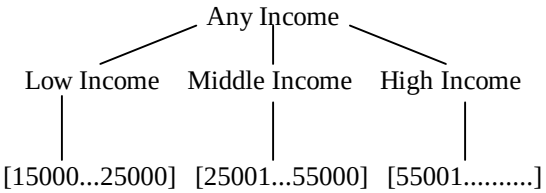


Figure 4. Concept Hierarchy for Income

Now, we start generalizing the values of the remaining attributes with the help of concept hierarchies. In our example, we need to generalize the values of those two attributes, Age and Income.

We don't need to generalize the values of the attribute Type-of-Car contained in the query because all tuples in the table have already the same value in the attribute. The generalization process is performed attribute by attribute. Now, we may receive an input from a user for the number of distinct concepts for each attribute. Assume that the user does not provide the number. For each remaining attribute in the table, we derive a concept that subsumes all the values in the attribute by ascending its concept hierarchy repeatedly one level at a time. The concept is the label of a node in the concept hierarchy. Then, we replace all the values in the attribute by the concept. During this process, distinct values may correspond to the same concept. After we generalize the values of the attribute Age, we get table 3.

Type-of-Car	Age	Income
Sports	Young	58000
Sports	Young	60000
Sports	Young	65000

Table 3.

Similarly, after we generalize the values of the attribute Income, we get the table 4 in which all tuples are identical.

Type-of-Car	Age	Income
Sports	young	high
Sports	young	high
Sports	young	high

Table 4.

Now, all of the tuples have the same values. Identical tuples are merged together. We store the total count of such merged tuples [table 5].

Type-of-Car	Age	Income	Count
sports	young	high	3

Table 5.

To generate an intensional answer, we select the tuple and construct a conjunction of maximal concepts. In our example, we get the following intensional answer: ``typical buyers of sports cars are young people with a high income.''.

In our approach, we can support intensional answers at multiple abstraction levels. If a user wants to get more specific intensional answers, we perform back substitution. We can replace the concept young with ``the age between 25 and 35'' and replace the concept high income with ``income greater than $55,000''. Then, we get the following intensional answer that ``typical buyers of sports cars are people whose age is between 25 and 35 and whose income is greater than $55,000''. If a user wants to get more general intensional answers, we try to find concepts which can be generalized to slightly higher concepts or concepts in virtual hierarchies that match conjunctions of the maximal concepts in the normal intensional answer. For example, we have a node in a virtual hierarchy, saying that if a person is young with a high income, then the person is called ``Yuppy''. We replace the conjunction of the maximal concepts, young and high income, by the concept Yuppy in the virtual hierarchy. Then, we can get a more general intensional answer, such as ``the typical buyers of sports cars are Yuppies''. By allowing intensional answers at multiple levels, we can generate not only a higher-level intensional answer but also a lower-level intensional answer.

5. CONCLUSIONS

Conventional database answers to queries, usually given as lists of objects, are not always the best means of efficient and effective communications between users and the database systems, especially when the answers include a large number of objects. Large sets of objects need to be summarized or reduced to descriptions of a form. Thus, it is necessary to enhance user-machine interfaces to conventional databases with additional features.

In this paper, we have presented a partially automated method for deriving intensional answers at multiple levels from large sets of extensional answers. We have applied and extended the inductive and attribute-oriented data mining method [2] for our task that provides more meaningful intensional answers to queries. An intensional answer characterizes an extensional answer and can provide more insight into the nature of the extensional answer. The capability to generate intensional answers at multiple levels enhances the man-machine interface to databases. The intensional answers may be used in many applications, including promotion and marketing.

In our approach, first, we build a concept hierarchy for each attribute and virtual hierarchies based on the concept hierarchies. Second, after a query is presented to the system, we find an extensional answer. Third, we derive an intensional answer from the extensional answer by characterizing the features of the extensional answer with the concept hierarchies and the virtual

hierarchies. Finally, we derive a more generalized answer or a more specialized answer according to a user's interest. Our approach provides a simple and reasonable way of providing intensional answers at multiple levels in database systems. Also, our approach is complete in the sense that it discovers all of the nonredundant intensional answers. The possibility of redundancy within intensional answers does not exist.

It may be interesting to use different computational methodologies to get intensional answers. Currently, we are building a prototype to perform experimentation on the proposed method. We are extending our approach with some modification to other data models, such as object-oriented and deductive ones.

REFERENCES

[1] Frawley, et. al., ``Knowledge Discovery in Databases: An Overview'', G. Piatetsky-Shapiro and W.J. Frawley[editors], *Knowledge Discovery in Databases*, pp. 1-27, AAAI/MIT Press, 1991

[2] J. Han, et.al., ``Data-Driven Discovery of Quantitative Rules in Relational Databases'', *IEEE Transactions on Knowledge and Data Engineering*, vol. 5, no. 1, pp. 29-40, 1993

[3] L. Cholvy and R. Demolombe, ``Querying a Rule Base'', *Proceedings of the First International Conference on Expert Database* Systems, pp. 365-371, 1986

[4] T. Imielinski, ``Intelligent Query Answering in Rule Based Systems'', Journal of Logic Programming, Vol. 4, No. 3, pp.229-258, 1987

[5] A. Motro, ``Using Integrity Constraints to Provide Intensional Answers to Relational Queries'', *Proceedings of 15th VLDB Conference*, pp. 237-246, 1989

[6] A. Motro and Q. Yuan, ``Querying Database Knowledge'', *Proceedings of the International Conference on Management of Data*, pp. 173-183, 1990

[7] A. Motro, ``Intensional Answers to Database Queries'', IEEE Transactions on Knowledge and Data Engineering, Vol.6. No. 3, pp. 444-454, 1994

[8] E. Pascual and L. Cholvy, ``Answering Queries Addressed to The Rule Base of a Deductive Database'', *Procceedings of the Second International Conference on Uncertainty in Knowledge-based Systems*, pp. 138-145, 1988

[9] A. Pirotte and D. Roelants, ``Constraints for Improving The Generation of Intensional Answers in A Deductive Database'', *Proceedings of 5th International Conference on Data Engineering*, pp.652-659, 1989

[10] A. Pirotte, et,al., ``Controlled Generation of Intensional Answers'', IEEE Transactions on Knowledge and Data Engineering, Vol.3, No. 2, pp. 221-236, 1991

[11] I.Y. Song and H.J. Kim, ``Design and Implementation of a Three-Step Intensional Query Processing Scheme'', Journal of Data Administration, Vol 2, No 2, pp.23-25, 1991

[12] S.C. Yoon and I.Y. Song, ``A General Method for Generating Intensional Answers in an Intelligent Information System'', *Proceedings of the 1994 ISCA International Conference on Computers and Their Applications*, pp. 94-98, 1994

[13] R. Agrawal, et.al., ``Mining Association Rules between Sets of Items in Large Databases, *Proceedings of ACM SIGMOD*, pp.207-216, 1993

[14] V. Dhar and A. Tuzhilin, ``Abstract-Driven Pattern Discovery in Databases'', *IEEE Transactions on Knowledge and Data Engineering*, vol. 5, no. 6, pp. 926-938, 1993

[15] J.S. Park, et.al., ``An Effective Hash-Based Algorithm for Mining Association Rules, *Proceedings of ACM SIGMOD*, pp.175-186, 1995

[16] G. Piatetsky-Shapiro, and W.J. Frawley, eds., *Knowledge Discovery in Databases*, AAAI/MIT Press, 1991

[17] G. Piatetsky-Shapiro, ``Discovery, Analysis and Presentation of Storong Rules'', *Knowledge Discovery in Databases*, pp. 229-248, AAAI/MIT Press, 1991

[18] S. C. Yoon, et.al., ``Intelligent Query Answering in Deductive and Object-Oriented Databases'', *Proceedings of the Third ACM*

International Conference on Information and Knowledge Management, pp.244-251, 1994

[19] S. C. Yoon, et.al. ``Semantic Query Processing in Deductive Object-Oriented Databases'', *Proceedings of the Fourth ACM International Conference on Information and Knowledge Management*, pp.150-157, 1995