Study of Imperfect Information Representation and FSQL processing

Kamal Singh, Kundan Prasad, Mahendra Kumar, A.K. Sharma

Abstract--Structured Query Language (SQL) is a very powerful too, but it is unable to satisfy needs for data selection based on linguistic expressions and degree of truth. As we know that the information in real-world applications is often vague, imprecise and uncertain. Fuzzy logic breaks the rigid crisp logic boundaries and allows decision to be taken in a more realistic manner. In database context, several fuzzy database (DB) models have been proposed. In these works, fuzziness is introduced at different levels. Common to all these proposals is the support of fuzziness at the attribute level and introducing fuzziness in querying permits the realistic querying on the crisp data as well as on fuzzy databases. Fuzzy query is not only a querying tool; it improves the meaning of querying extracts additional valuable information. The goal of research whose result presented in paper is to understand the implementation of imprecise data and explain FSQL (Fuzzy Structured Query Language).

Keywords- SQL, Fuzzy SQL, Fuzzy Logic, Fuzzy Database.

1. INTRODUCTION

Most of today's database management system consists of the crisp information. The primary goal of DBMS is to provide precise information at the time of retrieval. The conventional database management system does not handle imprecise, incomplete or vague information such as a very high, approximately some values. To triumph over this problem, the fuzzy database system has been introduced. In the real world scenario, every time the information we need cannot be precisely stated but rather than that there is a need to deal with natural language and retrieving the pertinent information. The paper has been divided in four parts. The essential idea is understand the implementation of imprecise data and FSQL. Section 1 and 2 presents introduction and concept of fuzzy logic correspondingly. Section 3 gives the introduction of imperfect information representation. Section 4 describe fuzzy query processing, main extensions added to FSQL and practical implementation of fuzzy data in database and perform queries on that database and section 5 portray conclusion respectively.

2. FUZZY CONCEPTS OVERVIEW

Traditional databases operate on large samples of precisely defined data. In real world, a lot of uncertainty exists which creates a need to represent imprecise information. Fuzzy logic breaks the rigid crisp boundaries and allows decisions to be taken in a more realistic manner. The concept was conceived by Zadeh in 1964 when he was trying to contemplate how to program software for handwriting recognition [1]. Human beings manipulate decisions based on pieces of information and generally reason them with vague terms like "The boy is young", or "He is running too fast". The manipulation needs additional processing on the computers part. The term "young" would cause a lot of errors when linked with relational databases which are tuned to incorporate numerical values. Fuzzy logic extends computer capabilities to deal with imprecise and vague information. Inconsistency, imprecision, vagueness, uncertainty, and ambiguity are five kinds of imperfect information in databases [2] inconsistency explain a property where the same attribute may hold different values. An example would be a person who may maintain multiple statuses of both a student and a part time employee. Imprecision and vagueness relate to attribute values picked from a range set rather than a defined numerical content. As an example, 'young' relates to a value in a numerical range like [10, 25]. Uncertainty 5 is the degree of truth of an attribute value. Ambiguity leads to several interpretations of the content. Furthermore the different kinds of imperfect information can co-exist with each other [2]. Fuzzy set theory [1] is a generalization of the crisp sets and proposes methods to resolve the different forms of information representation. The membership function in a fuzzy set is not a matter of true or false but a matter of degree [1]. Consider an example, "The temperature is hot". The temperature is called a fuzzy variable or a linguistic variable, which takes values (hot), which are in turn connected to numerical values through membership function.

3. Introduction to Imperfect Information

The objective of fuzzy databases is primarily to handle imperfect information in databases. In [6], the author distinguishes five types of imperfect information:

- Inconsistency: is a kind of semantic conflict that holds when some aspects of real-world is irreconcilably represented more than once in the database (e.g. when the age of a person is stored as 34 and 37).
- Imprecision: is relevant to the content of an attribute value and means that a choice must be made from a given range (interval or set) values (e.g. the age of a person is the set {17, 18, 19, 20} or the height is in interval [1.00-1.95]);
- Vagueness: is like imprecision but which is generally represented with linguistic terms (e.g. the age of a person is the linguistic "young");
- Uncertainty: is related to the degree of truth of attribute value, and I means that we can apportion some, but not all, of our belief to a given value or groups of values. It results from a lack of information and is that related to the designer and not to the object/concept being modeled. (e.g. the possibility that the age of a person is 35 right now should be about 90%);
- Ambiguity: it means that some elements of the model lack complete semantics leading to several possible interpretations.

The same author adds that imprecision, vagueness and uncertainty are the main types of imperfect information. Fuzziness comes from the impossibility to define sharp or precise borders, and therefore it is often associated with vagueness. In fuzzy database literature, uncertainty and imprecision are often represented through fuzzy set theory, similarity, proximity and/or resemblance relations. The combination of several approaches is also frequent. In addition to fuzzy, imprecise and uncertain, values of attributes may be unknown, undefined or null-valued. To facilitate data manipulation and for computing efficiency while giving the maximum flexibility to the users, the different types of attributes values (crisp, imprecise, uncertain, fuzzy, unknown, undefined or null) will be uniformly represented through possibility distribution. Figure 1.0 represents the different data types which we think permit to model almost all kinds of imperfect information.

3.1 Fuzzy Attributes

A fuzzy database needs a special data dictionary in order to store the information related to the inexact nature or context of each fuzzy attribute. We call it the Fuzzy Meta knowledge Base (FMB). In order to model fuzzy attributes we distinguish between two classes of fuzzy attributes: Fuzzy attributes whose fuzzy values are fuzzy sets and fuzzy attributes whose values are fuzzy degrees. Each class includes some different fuzzy data types:

3.1.1 Fuzzy Sets as Fuzzy Values

These fuzzy attributes may be classified in four data types. This classification is performed taking into account the type of referential or underlying domain. In all of them the values Unknown, Undefined, and Null are included:

Type 1: These are attributes with "*precise data*", classic or crisp (traditional, with no imprecision). However, we can define linguistic labels in its domain and we can use them in fuzzy queries. This type is useful for extending traditional databases allowing fuzzy queries to be made about classic data. For example: "Give me employees that earn a lot more than the minimum salary" (salary must be a well known attribute).

Type 2: These are attributes that gather "*imprecise data over an ordered referential*". These attributes admit both crisp and fuzzy data, in the form of possibility distributions over an underlying ordered dominion (fuzzy sets). It is an extension of the Type 1 that does, now, allow the storage of imprecise information, such as: "he is approximately 2 meters tall". The most complex of these fuzzy sets are the so-called *extended trapezoidal*. This novel type of fuzzy set is composed by linear functions in some pieces. It allows great flexibility (even for non-convex sets). For the sake of simplicity the simple trapezoidal functions (Figure 1) are more common. Note that this underlying domain is continuous and ordered (usually it will be the real number dominion or the time).

Type 3: They are attributes over "*data of discreet non-ordered dominion with analogy*". In these attributes some labels are defined (e.g. "blond", "ginger", "brown", etc.) that are scalars with a similarity (or proximity) relationship defined over them, so that this relationship indicates to what extent each pair of labels resemble each other. They also allow possibility distributions (or fuzzy sets) over this dominion, like for example, the value {1/dark, 0.4/ginger}, which expresses that a certain person is more likely to be dark than ginger. Note that underlying domain of these fuzzy sets are the set of labels and this set is discreet and non-ordered.

Type 4: These attributes are original and they are defined in the same way as Type 3 attributes, without it being

necessary for a similarity relationship to exist between the labels.

3.1.2 Fuzzy Degrees as Fuzzy Values

The domain of these degrees can be found in the interval [0,1], although other values are also permitted, such as a possibility distribution (usually over this unit interval). In order to keep it simple, we will only use degrees in the interval [0,1], because the other option offers no great advantages. The meaning of these degrees is varied and depends on their use. The processing of the data will be different depending on the meaning. The most important possible meanings of the degrees used by some authors are [12][13]: Fulfillment degree ,Uncertainty degree, Possibility degree and Importance degree. Of course, we can define and use other meanings. In this paper we do not aim to demonstrate the usefulness of these degrees and their different meanings. Several authors who have used these degrees have already done so. The ways of using these fuzzy degrees are classified in two families: Associated and non associated degrees.

Associated degrees are associated to a specific value to which they incorporate imprecision. These degrees may be associated to different concepts [13]:

Degree in each value of an attribute (we will call it as **Type 5):** Some attributes may have a fuzzy degree associated to them. This implies that each value of this attribute (in every tuple or instance) has an associated degree, measuring the level of fuzziness in that value. In order to interpret it, we need to know the meaning of the degree and the meaning of the associated attribute.

Degree in a set of values of different attributes (Type 6): Here, the degree is associated to some attributes. It joins the fuzziness of some attributes in only one degree.

Degree in the whole instance of the relation (Type 7): This degree is associated to the whole tuple of the relation and not exclusively to the value of a specific attribute of the tuple (or instance). Usually, it can represent something like the "membership degree" of this tuple (or instance) to the relation (or table) of the database. This degree represents the fuzzy degree of a fuzzy relation. **Non-associated degrees (Type 8):** There are cases in which the imprecise information, which we wish to express, can be represented by using only the degree, without associating this degree to another specific value or values. For example, the dangerousness of a medicine may be expressed by a fuzzy degree. The first version of FSQL only includes the fuzzy attributes Type 1, 2 and 3.

4. CONCEPT OF FUZZY QUERY

Fuzziness is introduced in database query language to allow imprecise querying on conventional data. If we aim for conventional crisp querying language it is precise and certain querying. Precision assumes that the effect will be exactly our perception and certainty assumes the structure and parameter are exactly identified. But for factual database there may be the understated complications: Actual situations are very often not crisp and deterministic and cannot be described precisely i.e. Real situations are very often uncertain or vague in a number of ways. Complete description of a real system would entail far more detailed data than a human being could ever be acquainted with and process at the same time. To get rid of these complications we need to be concerned about the notion of uncertainty. An analogous example would be our everyday SQL selects. Let's say a company is searching for young employees with great sales records, to consider for promotions.

Select from employee

Where age>20 and sales>100000

This statement will miss the 21 year-old with $90,000 in sales and the 31 year-old with $500,000, although those people may be bright young stars as well. Widening the search parameters waters down the results and the black & white nature of it will always miss those on the cusp. What the company actually wants is to do a SQL statement

Select young employee with great sales record. One solution would be fuzzy logic. They want employees that fall into two sets-

1) Young and

2) Good sales

The fuzzy solution would say, ok, every employee selling over $100,000 is 0.6. Also, anyone less than 30 years old is a

member of the Young Employee set with membership 1.0. 31 years old is 0.8. 35 years old is 0.5. Once you define those parameters, by definition the membership of an employee in the two sets is the lowest membership he has in either set. Our precocious 21 year-old would be 0.8(he has 1.0 in Young, and 0.8 in Good Sales) and our older but productive 31 year old would also be 0.8(0.8 Young and 0.8 Good Sales).

FSQL

The FSQL language [7, 8, 9] is an extension of SQL which allows the use of flexible queries. It means that all valid queries in SQL are as valid in FSQL. This paper will only

provide a summary of the main extensions added to this command:

a) **Linguistic Labels**: If an attribute is able of fuzzy treatment then linguistic labels can be defined on it. These labels will be preceded with the symbol $ to distinguish them easily. There are two types of labels and they will be used in different fuzzy attribute types: (1) Labels for attributes with an ordered underlined fuzzy domain (Fuzzy Attributes Type 1 and 2) and (2) Labels for attributes with a non-ordered fuzzy domain (Fuzzy Attributes Type 3 and 4).

b) **Fuzzy Comparators**: Besides the typical comparators (=,>, etc.), FSQL includes fuzzy comparators. There are many comparators, the most used are: FEQ (Fuzzy Equal than), (NFEQ: Necessarily FEQ), FGT (Fuzzy Greater than), (NFGT: Necessarily FGT), FGEQ (Fuzzy Greater or Equal than), (NFGEQ: Necessarily FGEQ), FLT (Fuzzy Less Than), (NFLT: Necessarily FLT), FLEQ (Fuzzy Less or Equal than), (NFLEQ: Necessarily FLEQ), MGT (Much Greater Than), (NMGT: Necessity MGT), MLT (Much Less than), (NMLT: Necessarily MLT), FINCL (Fuzzy INCLuded in), INCL, FDIF (Fuzzy DIFrent), (NFDIF: Necessary FDIF) [10, 11]. Like in SQL, fuzzy comparators compare one column with one constant or two columns of the same type.

c) **Fuzzy constant:** Besides the typical constants (NULL), FSQL included many such as $[a,b,c,d], #n, $label[n,m], UNKNOWN, UNDEFINED, etc.

d) **Fuzzy qualifiers**: they are of two natures, absolute and relative [11].

Example

FSQL introduces a new way of retrieving information from a relational database. Using this query facility you tell the computer what you want based on concepts rather than numbers and text strings. We want in this example to model an worker described by the following information: name and age. The attributes age described as follows:Age is of FTYPE2 type having like linguistic labels *BABY*(0, 0, 0, 15), *YOUNG*(0 ,15 ,15 ,30) , *MIDDLE*(15, 30,30, 45), MATURE(30,45,45,60) and OLD(45,60,60,60) defined by trapezoidal possibility distributions.

4.2.1 Implementation

Step-1 Design of DB tables

Before the composing of queries, DB administrator or DB designer should have designed DB tables to query them. For given example DB table given in table 2.

Name	age
Deepesh	23
Rohan	27
Mohan	55
Rahul	38
Devid	16
Ram	40

Table 2. Worker

Step-2 Design of Fuzzy Meta-Knowledgebase:

Meta-Knowledgebase means that the additional DB table and activities to support the fuzzy classification based on fuzzy interpreter. Additional tables contain linguistic variables, membership values and descriptions of atomic values. For the given example Fuzzy Meta-Knowledgebase gave in table 3 and table 4.

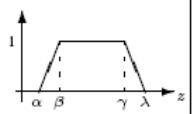
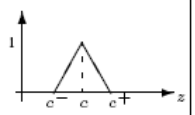
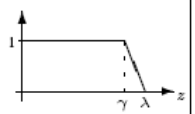
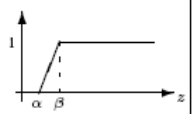
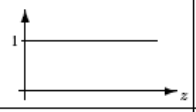
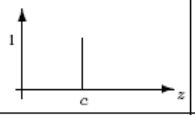
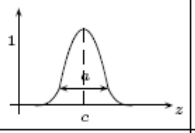
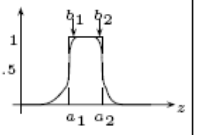
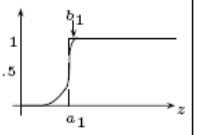
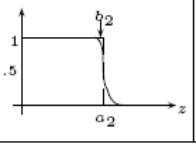
Data type A	Model	Representation	Parameters	$\mu_A(z)$
Fuzzy range label e.g. *age*= more or less between 20 and 30	I.1		$\alpha, \beta, \gamma, \lambda$	$\mu_A(z) = \begin{cases} 1, & \text{if } \beta \le z \le \gamma; \\ \frac{\lambda - z}{\lambda - \gamma}, & \text{if } \gamma < z < \lambda; \\ \frac{z - \alpha}{\beta - \alpha}, & \text{if } \alpha < z < \beta; \\ 0, & \text{Otherwise.} \end{cases}$
Approximate value e.g. *age*=about 35	I.2		c, c^-, c^+	$\mu_A(z) = \begin{cases} 1, & \text{if } z = c; \\ \frac{c^+ - z}{c^+ - c}, & c < z < c^+; \\ \frac{z - c^-}{c - c^-}, & c^- < z < c; \\ 0, & \text{Otherwise.} \end{cases}$
Interval e.g. $age \in [25, 35]$	I.3		α, β	$\mu_A(z) = \begin{cases} 1, & \text{if } \alpha \le z \le \beta; \\ 0, & \text{Otherwise.} \end{cases}$
Less than value e.g. age= less than 35	I.4		γ, λ	$\mu_A(z) = \begin{cases} 1, & \text{if } z \le \gamma; \\ 0, & \text{if } z < \lambda; \\ \frac{\lambda - z}{\lambda - \gamma}, & \text{if } \gamma \le z \le \lambda. \end{cases}$
More than value e.g. age= more than 35	I.5		α, β	$\mu_A(z) = \begin{cases} 1, & \text{if } z \ge \beta; \\ 0, & \text{if } z \le \alpha; \\ \frac{z - \alpha}{\beta - \alpha}, & \text{if } \alpha < z < \beta. \end{cases}$
Unknown	I.6			$\mu_A(z) = 1 ; z \ge 0$
Undefined	I.7			$\mu_A(z) = 0 ; z \ge 0$
Real number e.g. *age*=30	I.8		c	$\mu_A(z) = \begin{cases} 1, & \text{if } z = c; \\ 0, & \text{Otherwise.} \end{cases}$
Linguistic label e.g. *age*=young	II.1		a, c	$\mu_A(z) = \frac{1}{(1 + (a(z-c)^2}; \ z \ge 0$
Linguistic label e.g. *age*=young	II.2		a_1, a_2, b_1, b_2	$\mu_A(z) = \begin{cases} \frac{1}{1 + \frac{z - a_1 - b_1}{b_1}}, & \text{if } z < a_1 + b_1; \\ 1, & \text{if } a_1 + b_1 \le z \le a_2 - b_2; \\ \frac{1}{1 + \frac{z - a_2 + b_2}{b_2}}, & \text{if } z > a_2 - b_2. \end{cases}$
Linguistic label *age*=very old	II.3		a_1, b_1	$\mu_A(z) = \begin{cases} \frac{1}{1 + \frac{z - a_1 - b_1}{b_1}}, & \text{if } z < a_1 + b_1; \\ 1, & \text{if } a_1 + b_1 \le z; \end{cases}$
Linguistic label e.g. *age*=very young	II.4		a_2, b_2	$\mu_A(z) = \begin{cases} 1, & \text{if } z \le a_2 - b_2; \\ \frac{1}{1 + \frac{z - a_2 + b_2}{b_2}}, & \text{if } z > a_2 - b_2. \end{cases}$

Figure 1.0: Different Data types

Table	column	Type
Worker	Age	2

Table 3: fuzziness.

Table 3 describes what types of fuzzy attributes are used. In above example we consider type 2 attribute (Age).

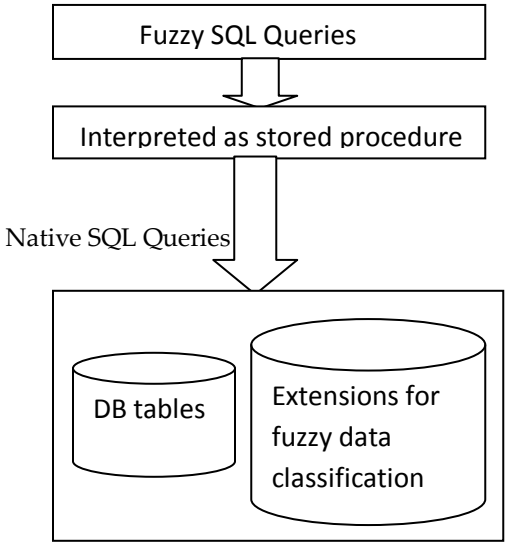
Table	Column	LV	A	B	C	D
Employee	Age	Baby	0	0	0	15
Employee	Age	Young	0	15	15	30
Employee	Age	Middle	15	30	30	45
Employee	Age	Mature	30	45	45	60
Employee	Age	Old	45	60	60	60

Table 4: Possibility

Age **is** of type 2 having linguistic labels defined by trapezoidal possibility distributions. We represent this in database using possibility table 3. (LV means linguistic variable).

Step-3 Design and implementation of interpreter:
In this step, fuzzy interpreter should be designed on the frame of given RDBMS using lexical and syntactical analysis of queries. The interpreter transforms the fuzzy SQL into a native SQL. This step should be carried out by software developer or DB programmer. In Veryha's (2005) research the interpreter was developed in the form of the stored procedure for the given RDBMS.

Step-4 Generation of DB reports and views:
As a result of fuzzy classification by using conventional SQL queries and fuzzy interpreter, fuzzy classified data is presented. The scheme of fuzzy classification framework implementation in RDB (Veryha, 2005) is shown in figure 2.0.



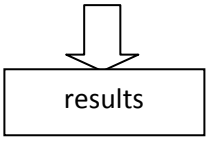


Figure 2.0: Fuzzy Classification Framework

2. SELECT name, Age FROM worker WHERE Age = 'mature';

Result shown in table 6

Name	Age	Fuzzy_degree
Ram	40	0.666666667
Rahul	38	0.533333333
Mohan	55	0.333333333

Table 6.

3. SELECT name, Age FROM worker WHERE Age = 'middle' and Age = 'mature'

Result shown in table 5

name	Age	Fuzzy_degree
Roy	38	0.4666666666
Dayal	40	0.3333333333

Table 6.

The FSQL approach strikes a balance in the process by including records that meet, to some degree, the intent of our query. Selecting the minimum or average result analysis then allows us to choose records that have some truth in each of the selection criteria

5. CONCLUSION

Information in real-world applications is often vague, imprecise and uncertain. In database context, several fuzzy database models have been proposed. In these works, fuzziness is introduced at different levels. Common to all these proposals is the support of fuzziness at the attribute level. This paper proposes first a rich set of data types devoted to model the different kinds of imperfect information. To facilitate data manipulation and for computing efficiency, the different types of attributes values are uniformly represented through possibility distribution. The FSQL language is an extension of the SQL language which permits us to handle fuzzy information in fuzzy or crisp databases. The first version of FSQL was implemented for Oracle databases [13]. In this paper we discuss a formal approach to implement imprecise information and FSQL processing.

6. REFERENCES

[1] .L. A. Zadeh, "Fuzzy Sets," Information and Control, vol.8, Academic Press, New York, pp. 338-353, 1965.

[2] Z. Ma, "Fuzzy Database Modeling of Imprecise and Uncertain Engineering Information", 1st Edition, Springer, 2006.

[3] M. Umano, S. Fukami, M. Mizumoto, K. Tanaka "Retrieval Processing from Fuzzy Databases". Technical Reports of IECE of Japan, Vol. 80, No 204, (on Automata and Languages), 1980, pp. 45-54, AL80-50. .

[4] Yoshikane Takahashi, "Fuzzy Database Query Languages and Their Relational Completeness Theorem" IEEE Transaction On Knowledge and Data Engineering, vol 5, N° 1, February 1993.

[5] M. Zemankova-Leech, A. Kandel, "Implementing Imprecision in Information Systems". Information Sciences, 37, 1985, pp. 107-141.

[6] Z. M. Ma, *A conceptual design methodology for fuzzy relational databases*. Journal of Database Management, Vol. 16, No. 2, pp. 66-83, 2005.

[7] J. Galindo, J. M. Medina, O. Pons, and J. C. Cubero, "A server for fuzzy SQL queries," in Flexible Query Answering Systems, T. Andreasen, H. Christiansen, and H. L. Larsen, Eds. New York: Springer-Verlag, 1998, pp. 164–174. Lecture Notes in Artificial Intelligence (LNAI) 1495.

[8] J. Galindo, A. Urrutia, M. Piattini, *Fuzzy Databases: Modeling, Design and Implementation*. Eds. Idea Group Publishing Hershey, USA, 2005.

[9] J. Galindo, "New Characteristics in FSQL, a Fuzzy SQL for Fuzzy Databases". WSEAS Transactions on Information Science and Applications 2, Vol. 2, 2005, pp. 161-169. [11] J. M. Medina, O. Pons, M. A. Vila.

[10] J. Galindo, M. Medina, M. A Vila, J. C. Cubero, "Fuzzy Comparators for Flexible Queries to Databases". In: Proc. of IBERAMIA'98. Lisbona (Portugal), 1998.

[11] .J. Galindo, "New Characteristics in FSQL, a Fuzzy SQL for Fuzzy Databases". WSEAS Transactions on Information Science and Applications 2, Vol. 2, 2005, pp. 161-169.

[12] Urrutia A., Galindo J., Piattini M., "Modeling Data Using Fuzzy Attributes". Proc. IEEE Computer Soc. Press XXII Int. Conf. of the Chilean Computer Science Soc. (SCCC 2002), pp. 117-123. Chile, 2002.

[13] Galindo J., Medina M., Pons O., Cubero J. C., "A Server for Fuzzy SQL Queries". In "Flexible Query Answering Systems". Lecture Notes in Artificial Intelligence 1495, pp. 164-174. Ed. Springer, 1998.