Database Design for Storage of Fuzzy Information in Traditional Database

Pankaj Gupta
Assistant Professor
Vaish College of Engg.,
Rohtak , India

Rahul Rishi
Associate Professor
Technological Institute of Textile & Sciences, Bhiwani, India

Harish Mittal
Assistant Professor
Vaish College of Engg.,
Rohtak , India

ABSTRACT
Fuzzy set approach has been applied in the database and information retrieval areas for nearly 30 years. Here we give consideration to aspects of these areas that seem to afford the greatest potential for further development. Imprecise information can be stored in traditional database using fuzzy linguistic terms (e.g. young, big) that are used frequently in daily life. Although these words are ambiguous, community agreement can be reached as to their meaning. The paper shows a mechanism to implement fuzzy information in a Traditional Database.

Keywords
Database, Fuzzy sets, Fuzzy membership function, Fuzzy relational database;

1. INTRODUCTION
In the last few years, several authors have dealt with the problem of relaxing the relational model in order to represent imprecise information in databases[1,2]; this leads us to Database system that lay within the scope of Artificial Intelligence, as they allow us to manage information with a terminology that is very similar to natural language. Imprecision can be included in the system at two levels: The first level considers the possibility of making imprecise queries to the classic databases. The second one is related to the problem of adding fuzzy information to the system. In both cases, the fuzzy sets theory [3] provides a powerful tool to represent imprecision.

In this paper we consider the handling of the problems at the second level that gives rise to the fuzzy relational database model. The proposal is to include the capability of representing and handling fuzzy information in a traditional database. The thing to be considered is to adopt a particular criterion for fuzzy information representation, i.e., to find the most suitable representation for imprecise data and operators among those possible in GEFRED Model. In this paper, we propose a new database design which is based on the GEFRED model that could have a mechanism to store fuzzy information in traditional database.

This paper shows that the mechanisms adopted to implement all elements related to fuzzy information, and their validity is extended to implementations based on other criteria and illustrates the way in which fuzzy information is represented.

The rest of the paper is divided in sections. Section 2 introduces basic concept for storing fuzzy information. Section 3 describes implementation of storage of fuzzy information. Section 4 is concluding section and summarizes the work.

2. BASIC CONCEPT
In this section we introduce the basic elements of a fuzzy extension of the traditional model.

2.1 The GEFRED Model
The GEneralised model Fuzzy heart Relational Database (GEFRED) has been proposed in 1994 by Medina *et al*. [4]. One of the major advantages of this model is that it consists of a general abstraction that allows for the use of various approaches, regardless of how different they might look. In fact, it is based on the generalized fuzzy domain and the generalized fuzzy relation, which include respectively classic domains and classic relations. The data types supported by this model are showed in the Table 1.

Table 1 : Data types in the GEFRED model.

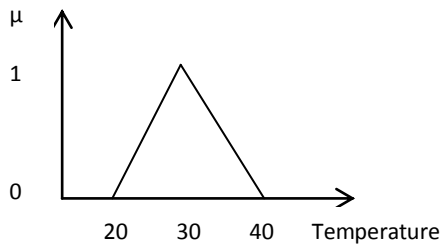
1. A single scalar (e.g., Behavior=good, represented by the possibility of distribution 1/good).
2. A single number (e.g., Age=28, represented by the possibility of distribution 1/28).
3. A set of mutually exclusive possible scalar assignations (e.g. Behavior={Bad, Good}, represented by {1=Bad, 1=Good}).
4. A set of mutually exclusive possible numeric assignations (e.g., Age={20, 21}, represented by {1/20, 1/21}).
5. A possibility distribution in a scalar domain (e.g., Behavior={0.6/Bad, 1.0/Regular}).
6. A possibility distribution in a numeric domain (e.g. Age= {0.4/23, 1.0/24, 0.8/25}, fuzzy numbers or linguistic labels).
7. A real number belonging to [0, 1], referring to the degree of matching (e.g., Quality=0.9).
8. An Unknown value with possibility distribution Unknown={1/u: u∈ U} on domain U, considered.
9. An Undefined value with possibility distribution Undefined={0/u: u ∈U} on domain U, considered.
10. A NULL value given by NULL={1/Unknown, 1/Undefined}.

2.2 Information representation

The elements related to fuzzy data handling can have different representations (i.e. using different membership functions), here we will use a triangular representation. The same can be said about the way we are going to model fuzzy relational operators as well as for the rest of fuzzy items to appear in the system. In this section, we show the representation criteria used in our implementation. This criterion is not exclusive for a concrete representation but represents the base on which the system is built according to the designed scheme for a Fuzzy database.

We will use the representation provided by the traditional database. The model considers two different groups with different representation for imprecise data.

- "Imprecise data over ordered underlying domain." This type of data contains possibility distributions defined on continuous or discrete but ordered domains. Type 7 of Table 1 belongs to this group. Each data of this type is associated with a membership function. For the sake of simplicity in the representation and computing efficiency, we will adopt the representation shown in Figure ….



- "Data with correlation over discrete domain." This group of data is built over discrete domains on which there are close relations defined between its values. In this case, we will have to store the data representation as well as the representation of the close relations defined on the domain values. The different data which we can represent in this group are the following.

  - Simple scalars. These data are represented using the representation scheme of the traditional database. We need to provide the system with the information for it to handle the close relation defined on the underlying domain.

  - Possibility distribution over discrete domain. An imprecise data of this type is associated with a representation in which the domain values that constitute it are described together with the respective possibility values for each of them. $((P_1,d_1), \ldots ,(P_n,d_n))$.

2.3 Fuzzy Operators

The different comparison operators used to relate data base relations are the relational ones. To operate on fuzzy information these operators are extended. The representation adopted in our model for the different relational operators is as follows.

"=", this operator models equality concept for imprecise data.
"≈", this operator provides the degree in which two "crisp" numeric values are approximately equal.
"≥", It is defined on ordered domains. This operator membership function is given by the fuzzy relation. This operator can solve the following comparisons.

- Degree to which a "crisp" number is "≥" than a possibility distribution.
- Degree to which a possibility distribution is "≥" than a "crisp" number.
- Degree to which a possibility distribution is "≥" than another possibility distribution.

"≤", It is defined on ordered domains. This operator membership function is given by the fuzzy relation, where A and B are imprecise data over ordered domain or "crisp" numeric data. "≤", is defined as the complement of the operator "≥"

2.4 Querying Database

When we query an imprecise database we are imposing some conditions that the result must satisfy. Given the imprecise nature of data and operators we work on, there is a degree of matching for any condition involved in a query. This degree of matching is in the interval [0, 1]. Using a minimal threshold for this degree of matching we can control the precision with which the conditions of the query are satisfied. Such a threshold will be named "qualifier". This "qualifier", a value between 0 and 1, can be represented through a linguistic value; for example, if we state that the degree to which a condition is matched is "low", it means that we will accept all tuples whose degree of matching is less than or equal to 0.2, i.e., we can associate linguistic values with qualifiers. The threshold value we associate with each linguistic label must be stored in the system and has, as well as linguistic labels, a subjective meaning.

2.5 Limits of the Existing Works

As per above discussion, the solutions currently proposed are restricted to the modelling of the flexible queries in the Database. The fuzzy information is already supposed manually implemented by the user. It returns these solutions limited to some simple examples. We propose in the continuation a new approach that allows the user to descript and manipulate the fuzzy information directly with traditional database.

3. IMPLEMENTATION OF THE FUZZY INFORMATION

Fuzzy Information is implemented in three levels.

- At the level of the Fuzzy database. The system holds knowledge about the treatment of the available fuzzy operations.
- At the level of the Database. The database consists of precise data. As we are concerned with the

representation of imprecise data, we must determine how we can store it. So, the data representation must be extended in order to deal with this kind of information.

— At the level of the metadata. There is a part of the database where all the information the system must know about the presented data is stored i.e. data about the data.

The Fuzzy database must contain information about the elements in the Database with imprecise data as well as their nature and representation. We will call Fuzzy metadata as extension of the metadata which captures all the necessary information about the imprecise data in the database. The criteria used in order to represent several aspects about the fuzzy information is shown above. Now, we proceed to describe the imprecise data representation in the database and metadata. There are three types of attributes which can be treated in an imprecise manner as follows.

1. Attributes which are totally crisp (traditional), but they have some linguistic labels defined on them, which allow us to make the query conditions for these attributes more flexible. They are called Type 1 attributes. The representation of these data attributes is similar to that of precise data. In addition, information about the linguistic labels along with information about the attribute type is stored in the Metadata.

2. The attributes which admit crisp data as well as possibility distributions over an ordered underlying domain, called Type 2 attributes. The data types allowed are those described in Table 2. Incomplete information such as UNKNOWN, UNDEFINED and NULL, and precise information can also be represented. We must store the following information in the database:

   — The data type. These can be: UNKNOWN, UNDEFINED, NULL, "CRISP", LABEL, INTERVAL, APROX., FUZZY.
   — The parameter description defining each data, depending on the corresponding type data. For instance, there three parameters describing a FUZZY type: α,m,β.

   For each type 2 an attribute F is created, an attribute F_TYPE with the type code and attributes Fuzzy_1, Fuzzy_2, Fuzzy_3 representing the parameters for each data. For the LABEL type the FUZZY_ID code represents an identifier for the linguistic label defined in the metadata. "Margin" is another parameter stored in the metadata.

Table 2: Type 2 Attributes Representation

Data type	F_TYPE	Fuzzy_1	Fuzyy_2	Fuzzy_3
UNKNOWN	0	NULL	NULL	NULL
UNDEFINED	1	NULL	NULL	NULL
NULL	2	NULL	NULL	NULL
CRISP	3	V	NULL	NULL
LABEL	4	FUZZY_ID	NULL	NULL
INTERVAL[A, B]	5	A	0	B
APROX(d)	6	d-margin	Margin	d+margin
FUZZY	7	α	m	β

3. The attributes which do not have an ordered underlying domain, for instance, hair color, are called Type 3 attributes. These attributes are used for scalar data or possibility distributions on scalar domains: It also accepts UNKNOWN, UNDEFINED and NULL.

For these kind of attributes the type and the representation associated with each data needs to be stored in the database. It is stored in the metadata which are the type 3 attributes as well as the definition for the close relations associated with the underline domain.

3.1. The fuzzy metadata

The metadata organizes [5, 6] all the information concerning the imprecise nature of attributes which must be stored in an accessible manner by the System. We consider the metadata as an extension to the catalog system; so, we will organize the information using tables or relations. The elements stored in the metadata are the following.

- The attributes in the database with imprecise treatment?
- The type of these attributes: type 1, 2 or 3.
- Elements defined in the database scope, i.e., query fuzzy quantifiers
- The fuzzy objects defined on each attribute:
  - — Linguistic variables
  - — Query threshold qualifiers
  - — Approximate values
  - — Close relations

The organization of the tables of fuzzy metadata is shown in Fig. 2. In the following section we are going to describe the data structure of each table of fuzzy metadata.

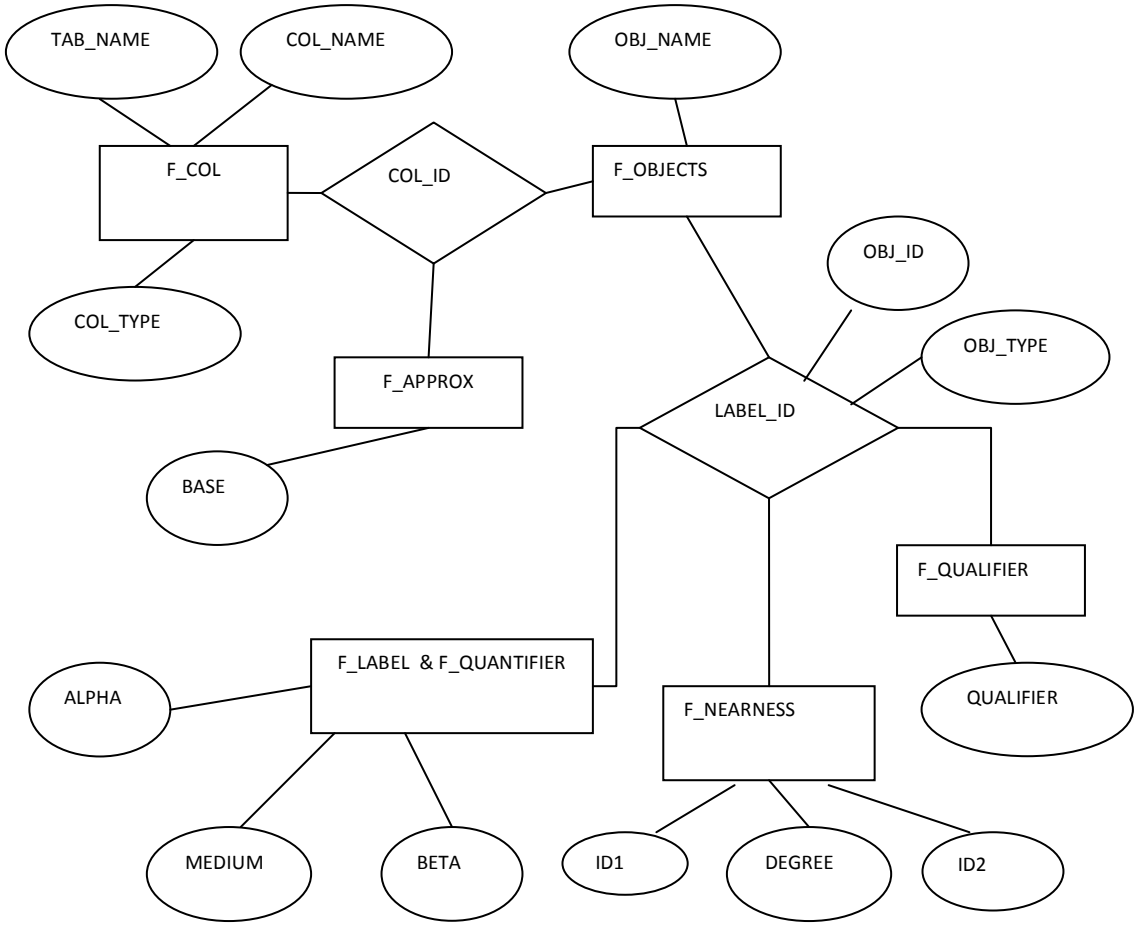


Figure 2: Organization of Fuzzy Meta Knowledge Base

F-COL

This table contains a description of the system items that permit a fuzzy handling and establishes a classification either for the fuzzy data types that can appear in an item or for the type of handling that the Fuzzy database will use with them. The table is initially formed by four items in which the previous characteristics are reflected. The items are the following.

-TAB_NAME: The type of this item is "character"; the rules used to name the tables will be the same as those used by the system. This item contains the name of the table which the fuzzy item belongs to.

- COL_NAME: It is also a character type item and has the same considerations about the identifiers type that may be involved. The identifiers contained in this item refer to those columns which will receive fuzzy handling, because they contain fuzzy information or because they may be involved in a fuzzy querying process.

-COL_ID: It is a positive number whose range depends on the implemented system. It is, in addition, primary key of this table and associates a numeric identifier with the item definition.

-COL-TYPE: This contains information about the data types that the columns designed by COL_ID will receive.

Column Types. Apart from the "crisp" data of a database, we consider those added under the new fuzzy conception of the database. The classification of the different columns is the same as that adopted for the attributes.

Type 1. Those columns involving data that, though crisp, are susceptible to fuzzy handling and thus can appear in the description in the F_COL table. The treatment of this type of data, as well as the conventional functions, is determined by the linguistic labels defined over them and appropriately coupled in the F_ OBJECTS table.

Type 2. This classification permits the system to recognize the fuzzy data in the item, referred to such as linguistic labels and possibility distributions as well as the most suitable handling, according to the information in the F_OBJECTS table.

Type 3. These columns require that close relations over the elements of each item are defined.

F_OBJECTS
This table contains a list of fuzzy objects defined in the database columns. Furthermore, it contains a classification of these objects through the OBJ_TYPE item. The information is structured as detailed in the following.

-COL_ID: Contains the number that identifies the column over which the object named in the OBJ_NAME item of this table is defined. It constitutes a foreign key to the F_COL table.
-OBJ_NAME: Contains the name of an object among the types involved in the OBJ_TYPE item.
-OBJ_ID: Assigns a number to each object; this number is used to identify the object in the rest of the tables. This item, together with the COL_ID item constitutes the primary key of this table.
-OBJ_TYPE: Specifies the object type identified by the OBJ_ID item; it therefore tells the system which table to look for in the suitable definition of the object involved. The values allowed are:

-0 for linguistic label
-1 for scalar related to close relations handling
-2 for qualifier labels defined over the matching degree of the query
-3 for quantifier labels over query.

F-LABEL
This table contains the points that determine the membership function corresponding to the linguistic labels of trapezoidal type. The items in this table are:

-LABEL_ID: Contains the number that identifies the label through the OBJ_ID item in the F_ OBJECTS table. This item constitutes a foreign key to this table. The LABEL_ID item is the primary key of the F_ LABEL table.

ALPHA=inf{x:x∈support(label)}
BETA=inf{x:x∈support(label)}
MEDIUM: =sup{x:x∈kernel(label)}

F_APPROX
This table contains all the information about the membership function of the "approximately" type labels. These functions are triangular, with membership value 1 for the point over which the approximation is considered and can be characterized by this point, that will be given in the query, and by the width of the triangle base. Therefore, the table is composed of a column identifier, COL_ID (the primary key) and by the item (BASE) that describes the boundaries.

F-NEARNESS
This table represents the close or similarity measures between the different domains values allowed for type "3" items of the F_COL table. The information is structured in this way: Two items establish the relation between two domain values, and the third one contains their degree of close. We use the concept of close because it is less restrictive in the sense that it only demands symmetric and reflexive properties. Thus, it is not necessary to give information about those pairs whose values can be deduced from these properties R(x, x) = 1, R(x, y) = R(y, x). The items are structured as follows:

-ID1: Contains the first value of the related couple.
-ID2: Contains the second value.
-DEGREE: number ([0,1]). Contains the degree of close of the related concepts of ID1 and ID2.

F_QUALIFIERS
This table contains the qualifier associates to the linguistic value given in the table F_OBJECTS. The items in this table are:

-LABEL_ID: Is the primary key and identifies the label over which the definition is established.
-QUALIFIER: Qualifier value for the label referenced in the LABEL_ID item and will take values between 0 and 1.

F_QUANTIFIER
This table contains the label definitions of the quantifiers used in the query. These labels identify triangular possibility distributions over [0, 1] domain.

-LABEL_ID: (number). Is the primary key and identifies the label over which the definition is established.
- ALPHA: number [0,1]
  α=inf{x:x∈support(label)}
- MEDIUM: number [0,1]
  β=inf{x:x∈kernel(label)}
- BETA: number [0,1]
  δ=sup{x:x∈support(label)}

3.2 Example of implementation of fuzzy information

To illustrate the complete mechanism of the fuzzy information implementation in a conventional database, we give an example of how the information related to a table involving both crisp and fuzzy data is expressed. Our example is based on a group of employees as shown in Table 3. The columns NAME and ADDRESS contain crisp information, while AGE and SALARY contains both crisp and fuzzy information. The Figure 3, shows the label definition for SALARY for example shown in Table 3. Similarly Label definition for AGE can also be represented.

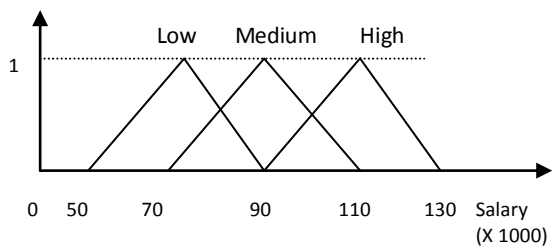


Figure 3: Label Definition for SALARY

Table 3: Employees

NAME	ADDRESS	AGE	SALARY
RAM	MODEL TOWN, ROHTAK	30	HIGH
MOHAN	DLF COLONEY, ROHTAK	YOUNG	LOW
SHYAM	ROHINI, DELHI	MIDDLE	40000
DEEPAK	SOUTH EX, DELHI	50	MEDIUM
ROHIT	GURGAON	OLD	120000

4. CONCLUSIONS

This paper presents a new idea for storing fuzzy and non-fuzzy information in traditional database. It put forward the adoption of a series of approaches for representing fuzzy information in databases and implementing this representation through the use of the available mechanisms in the traditional database.

With these approaches it has been pursued to satisfy the following general objectives.

- To provide representation to an extensive range of fuzzy information for storage in database.
- To center our approach on the mechanisms provided by the conventional database at the moment of deciding the way in which we can represent this information.
- To select those approach which support the efficiency in the treatment of the fuzzy information in the framework of these systems?

To implement the fuzzy information the structure of data available in any database (domain, attributes, relations and catalogue of system) is used. The mechanism used to extend the capacity of representation of a conventional database is integrated perfectly in the system. It establishes an adequate implement scheme in order that the operations of fuzzy manipulation allow obtaining satisfactory results and reaching a high grade of efficiency.

5. REFERENCES

[1] Bosc P. and Pivert O., "SQLf: A Relational Database Language for Fuzzy Querying," Computer Journal of IEEE Transactions on Fuzzy Systems, vol. 3, no. 1, pp. 80-89, 1995.

[2] Bosc P. and Pivert O., "SQLf Query Functionality on Top of a Regular Relational Database management," in Proceedings of Knowledge Management in Fuzzy Databases, Heidelberg, pp. 171-190, 2000.

[3] L.A. Zadeh, Fuzzy sets as a basis for a theory of possibility, Fuzzy Sets and Systems 1 (1978) 3-28. José Galindo, Handbook of Research on Fuzzy Information Processing in Databases, Information science reference, Hershey, New York, 2008

[4] Medina M., Pons O., and Vila A., "GEFRED: A Generalized Model of Fuzzy Relational Data Bases", Computer Journal of Information Sciences, vol. 76, no. 1, pp. 87-109, 1994.

[5] Galindo J., Urrutia A., and Piattini M., Representation of Fuzzy Knowledge in Relational Databases, IEEE Computer Society, 2004.

[6] Galindo J., Urrutia A., and Piattini M., Fuzzy Databases: Modeling, Design and Implementation, Idea Group Publishing, Hershey, 2006.

[7] Medina M., Pons O., and Vila A., "FIRST: A Fuzzy Interface for Relational Systems," in Proceedings of VI International Fuzzy Systems Association World Congress (IFSA), Sao Paulo, pp. 29-31, 1995.

[8] M. Umano and S. Fukami, Perspectives of fuzzy data-bases, Japanese J. Fuzzy Theory and Systems 3 (1991) 75-91.

[9] Pankaj Gupta and Rahul Rishi, " Recent trends of fuzzy databases and the roadmap ahead ".RMS Journal of Management and IT " , vol.1,no. 2 pp.54-60,2009.

[10] Urrutia A., Galindo J., and Piattini M., "Modeling Data Using Fuzzy Attributes," in Proceedings of the 22nd International Conference of the Chilean Computer Science Society (SCCC'02), UK, pp. 117-123, 2002.

[11] Zemankova M. and Kandel A., "Implementing Imprecision in Information Systems," Computer Journal of Information Sciences, vol. 37, no. 1, pp. 107-141, 1985.