Volume 5, Issue 3, March 2015 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Use of Ontology Concept Store Fuzzy Data

Geetanjali Tyagi*
M.tech CSE& SRM University
U.P., India

Megha Sharma
M.tech CSE& SRM University
U.P., India

Abstract— Database engineering has been progressed up to the Relational database stage. Fuzzy information administration in databases is a complex process in view of adaptable information nature and heterogeneous database frameworks. Relational Database Management System (RDBMS) can just handle fresh information but cannot handle precise data information. Structured Query Language (SQL) is a very powerful tool but can handle data which is crisp and precise in nature. It is not able to fulfill the requirements for information which is indeterminate, uncertain, inapplicable and imprecise and vague in nature. The goal of this work is to use Fuzzy technique in RDBMS. But, Fuzzy Relational Database Management System (FRDB) requires complex data structures, in most cases, are dependent on the platform in which they are implemented. A solution that involves representing an FRDB using an Ontology as an interface has been defined to overcome this problem. A new Fuzzy Query Ontology is proposed in this dissertation with implementation. The implementation layer, which is responsible for parsing and translating user requests into the corresponding DB implementations in transparent, is required to establish communication between the Ontology and the relational databases management system (RDBMS). This ontology defines a framework for storing fuzzy data by defining those using classes, slots, and instances. An Ontology is an explicit and formal specification of a conceptualization. Ontologies provides a shared understanding of a domain which allows interoperability between semantics

Keywords-SQL, Fuzzy Logic, Fuzzy SQL,Ontology

I. INTRODUCTION

The research area of fuzziness in Relational Data Base Management Systems (RDBMS) has resulted in a number of models aimed at the representation of imperfect information in DataBases (DB), or at enabling non-precise queries (often called flexible queries) on conventional database schemas.Classical data models often suffer from their incapability of representing and manipulating imprecise and uncertain information that may occur in many real world applications.The purpose of introducing fuzzy logic in databases is to enhance the classical models such that uncertain and imprecise information can be represented and manipulated. This resulted in numerous contributions, mainly with respect to the popular relational model or to some related form of it. Relational Database Management System(RDBMS) extension for representing fuzzy data is obviously not a new problem. Relational Database can store only crisp, precise data, but can able to store fuzzy values. Complexity normally arises from uncertainty in the form of ambiguity. The computerized system is not capable of addressing uncertain, imprecise data and ambiguous issues whereas, the human have the capacity to reason "approximately". As a result, human, when interacting with the database, want to make complex queries that have a lot of vagueness present in it. So, database which is in use cannot store uncertain data. But in real situations, these are not crisp and deterministic and therefore, cannot be described precisely.

FUZZY LOGIC

Fuzzy Logic is a form of many valued Logic. It deals with reasoning that is approximate rather than fixed and exact. Fuzzy Logic has been extended to handle the concept of partial truth where the truth value may range between completely true and completely false. In fuzzy Logic, membership function represents the degree of truth. For any set, membership function lies between [0,1]. The purpose of introducing fuzzy logic in databases is to enhance the classical models such that uncertain and imprecise information can be represented and manipulated. This resulted in numerous contributions, mainly with respect to the popular relational model or to some related form of it.

Relational Database Management System

A relational database is a database that has a collection of tables of data items, all of which is formally described and organized according to the relational model. A Data in a single table represents a relation, from which the name of the database type comes. In typical solutions, tables may have additionally defined relationships with each other. In the relational model, each table schema must identify a column or group of columns, called the primary key, to uniquely identify each row. A relationship can then be established between each row in the table and a row in another table by creating a foreign key, a column or group of columns in one table that points to the primary key of another table. The relational model offers various levels of refinement of table organization and reorganization called database normalization. The database management system (DBMS) of a relational database is called an RDBMS, and is the software of a relational database.

Fuzzy Set

The original interpretation of fuzzy sets arises from a generalization of the classic concept of a subset extended to embrace the description of ―"vague" and ―"imprecise" notions. This generalization is made in the following way:- 1. The membership of an element to a set becomes a ―"fuzzy" or ―"vague" concept. In the case of some elements, the issue of whether they belong to a set may not be clear. 2. The membership of an element may be measured by a degree, commonly known as the "membership degree" of that element to the set, and it takes a value in the interval [0,1] by agreement.

A linguistic label is the word, in natural language, that expresses or identifies a fuzzy set that may or may not be formally defined. Thus, the membership function μA(x) of a fuzzy set A expresses the degree in which x verifies the category specified by A. With this definition, we can assure that in our everyday life we use several linguistic labels for expressing abstract concepts such as young, old, cold, hot, cheap, expensive, and so forth. The intuitive definition of these labels not only varies from one person to another and depends on the moment, but also it varies with the context in which it is applied. For example, a "high" person and a "high" building do not measure the same.

Example 1.1: If we express the qualitative concept ―young‖ by means of a fuzzy set, where the x-axis represents the universe of discourse ―age‖ (in natural whole numbers) and the y-axis represents the membership degrees in the interval [0,1], then, following Equation , the fuzzy set that represents that concept could be expressed in the following way:-

Let us considering a discreet universe:- Young = 1/0 + ... + 1/25 + 0.9/26 + 0.8/27 + 0.7/28 + 0.6/29 + 0.5/30 + ... + 0.1/34

Figure:

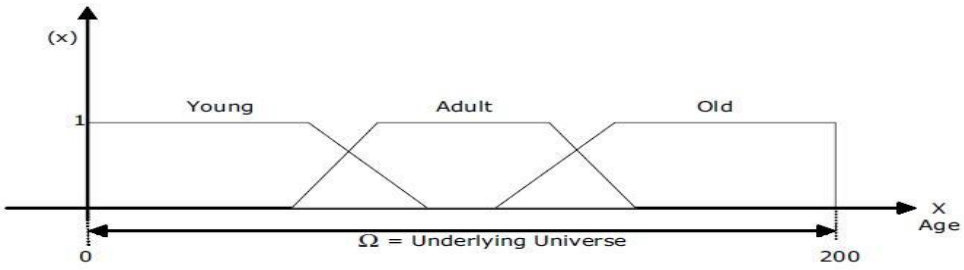


Figure 1.1. Three linguistic labels of Example 1.[1].

Types of Membership Functions

1. **Triangular (Figure 1.2)**: Defined by its lower limit *a***,** its upper limit *b*, and the model value *m*, so that *a* <*m* <*b*. We call the value *b-m margin* when it is equal to the value *m – a*.

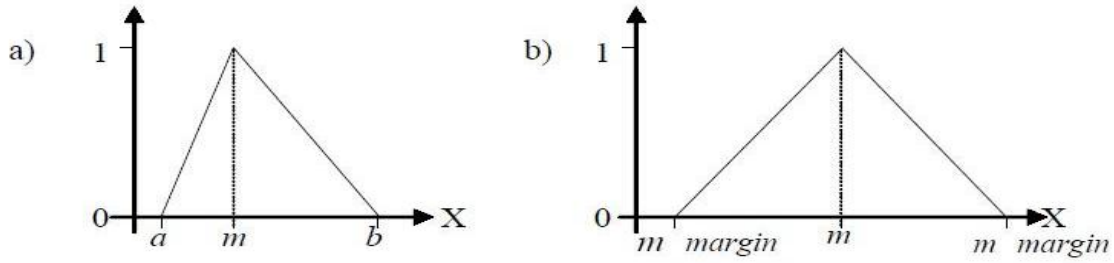
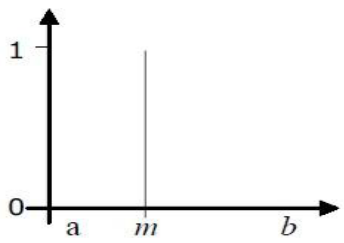


Figure 1.2. Triangular fuzzy sets: a) general and b) symmetrical.

$$A(x) = \begin{cases} 0 & \text{if } x \le a \\ (x-a)/(m-a) & \text{if } x \in (a, m] \\ (b-x)/(b-m) & \text{if } x \in (m, b) \\ 1 & \text{if } x \ge b \end{cases}$$

2. **Singleton (Figure 1.3)**: It takes the value 0 in all the universe of discourse except in the point *m*, where it takes the value 1. It is the representation of a crisp value.



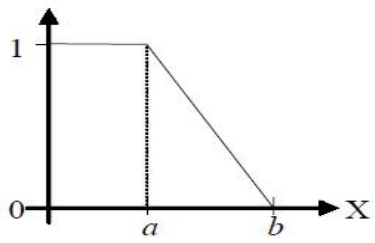


Figure 1.3 Singleton fuzzy set (left) and Figure 1.4. L fuzzy set (right)

$$SG(x) = \begin{cases} 0 & \text{if } x \neq m \\ 1 & \text{if } x = m \end{cases}$$

3. **L Function (Figure 1.4)**: This function is defined by two parameters **a** and**b**, in the following way:

$$L(x) = \begin{cases} 1 & \text{if } x \leq a \\ \dfrac{a-x}{b-a} & \text{if } a < x \leq b \\ 0 & \text{if } x > b \end{cases}$$

4. **Gamma Function (Figure 1.5)**: It is defined by its lower limit *a* and the value $k > 0$. The two definitions of Gamma function are as follows:

$$\Gamma(x) = \begin{cases} 0 & \text{if } x \leq a \\ \dfrac{k(x-a)^2}{1+k(x-a)^2} & \text{if } x > a \end{cases}$$

$$\Gamma(x) = \begin{cases} 0 & \text{if } x \leq a \\ \dfrac{k(x-a)^2}{1+k(x-a)^2} & \text{if } x > a \end{cases}$$

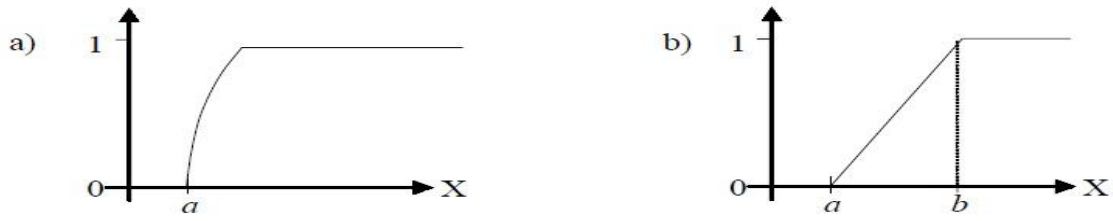


Figure 1.5 Gamma fuzzy sets: a) general and b) linear

5. **Trapezoid Function (Figure 1.6)**: Defined by its lower limit **a** and its upper limit **d,** and the lower and upper limits of its nucleus, **b** and **c** respectively.

$$T(x) = \begin{cases} 0 & \text{if } (x \leq a) \text{ o } (x \geq d) \\ (x-a)/(b-a) & \text{if } x \in (a,b] \\ 1 & \text{if } x \in (b,c) \\ (d-x)/(d-c) & \text{if } x \in (b,d) \end{cases}$$

Figure 1.6 Trapezoidal fuzzy set (left) and Figure 1.7. S fuzzy set (right)

6. **S Function (Figure 1.7)**: Defined by its lower limit *a*, its upper limit *b*, and the value *m* or point of inflection so that *a* <*m* <*b***.** A typical value is *m* = (*a* + *b*) / 2**.** Growth is slower when the distance *a* – *b* increases. 7.

7. **Gaussian Function (Figure 1.8)**: This is the typical Gauss bell, defined by its midvalue **m** and the value of *k* >0. The greater *k* is**,** the narrower the bell. $G(x) = e^{-k(x-m)2}$

Figure 1.8 Gaussian fuzzy set (left) and Figure 1.9. Pseudo-exponential fuzzy set (right

8. Pseudo-Exponential Function (Figure 1.9): Defined by its midvalue*m* and the value *k* >1. As the value of *k* increases, the rate of growth increases, and the bell becomes narrower.

Fuzzy Database Modeling

On occasion, the term imprecision embraces several meanings that we should differentiate. For example, the information you have may be incomplete or fuzzy (diffuse or vague), you may not know whether it is certain (uncertainty), perhaps you are totally ignorant of the information (unknown), you may know that the information cannot be applied to a specific entity (undefined), or you may not even know whether the data can be applied to the entity in question (total ignorance or a value of null) [1]. Each of these terms depends on the context in which it is applied. The management of uncertainty 13 in database systems is a very important problem, as the information is often vague. Motro states that fuzzy information is content-dependent and he classifies it as follows:

• **Uncertainty**: It is impossible to determine whether the information is true or false. For example, "John may be 38 years old."

• **Imprecision**: The information available is not specific enough. For example,"John may be between 37 and 43 years old, "John is 34 or 43 years old" (disjunction),"John is not 37 years old" (negative), or even a simple unknown.

• **Vagueness**: The model includes elements (predicates or quantifiers) that are inherently vague, for example,"John is in his early years" or "John is at the end of his youth." However, after these concepts have been defined, this case would match the previous one (imprecision).

• **Inconsistency**: It contains two or more pieces of information that cannot be true at the same time. For example, "John is 37 and 43 years old, or he is 35 years old"; this is a special case of disjunction.

• **Ambiguity**: Some elements of the model lack complete semantics (or a complete meaning). For example, "It is unclear whether the salaries are annual or monthly."

II. REPRESENTATION OF FUZZY KNOWLEDGE IN RELATIONAL DATABASES

The Relational Model was developed by E.F. Codd of IBM and published in 1970 [1]. It is currently the most used and has been a milestone in the history of databases, revolutionizing the market. In fact, relational databases have been the most widespread of all databases. On a theoretical level, many Fuzzy Relational Database models, which are based on the relational model, extend this so that vague and uncertain information can be stored and/ or treated with or without fuzzy logic.

The FuzzyEER Model [1] is an extension of the EER Model for creating conceptual schemas with fuzzy semantics and notations. This extension is a good eclectic synthesis between different models and provides new and useful definitions: fuzzy attributes, fuzzy entities, fuzzy relationships, fuzzy specializations, and so forth.

For each fuzzy attribute type, it is necessary to clarify two aspects:

1. How to represent the values (which the attribute can store).
2. What information needs to be stored in the Fuzzy Metaknowledge Base (FMB) to process it, and how this information should be organized.

FMB (Fuzzy Metaknowledge Base):

Fuzzy Metaknowledge Base [1] organizes all the information related to the imprecise or vague nature of these attributes.

1. Attributes with fuzzy processing: Attributes of the database that receive fuzzy processing and the type of these attributes (from Type 1 to 8).
2. Information about these attributes: Depending on its type, different information is stored for each attribute:
   - ✓ Types 1 and 2:
   - Linguistic labels: name and definition (trapezoidal fuzzy set) for each one.
   - The margin value for approximate values.
   - The much value M: minimum distance M to consider two values as ―very‖ separated (called ―much‖). This last value is used in comparisons such as ―much greater than‖ and ―much less than‖.
   - ✓ Types 3 and 4:
   - Maximum length for possibility distributions in values of these types (value n used in the fuzzy attributes Type 3 and Type 4 sections).
   - Linguistic labels: name.
   - Similarity relations between these labels (only for fuzzy attributes Type 3).
   - ✓ Types 5 and 6:
   - Degree significance (or meaning).
   - Attribute or attributes to which the degree is associated (in the Type 5 case, there is only one attribute).
   - ✓ Types 7 and 8:
   - Degree significance (or meaning), which is optional in the Type 8 (fuzzy degree with its own meaning).

III. ONTOLOGY DESCRIPTION

The ontology that describes a Fuzzy Database Schema as defined previously consists of a fuzzy Database schema and the fuzzy data stored in the Database (the tuples). This ontology, however, represent schemas and data simultaneously as ontology class cannot be instantiable twice, therefore two ontologies is defined, one of which describes fuzzy schemas as instances of a Database catalog ontology and the other which describes the same schema as a domain ontology which will allow the data instatiating it to be defined.

Ontologies vs. Databases

Ontologies allow representing knowledge of any domain in a formal and consensuatedway . On the other hand, relational DBs also represent knowledge of any domain but following certain rules specified by ANSI Standard SQL . Nowadays, both technologies coexist together and they can exchange information to take advantage of the information that they represent. Several proposals have been developed for establishing the communication between ontologies and databases. Some of them consist of creating ontologies from a database structures, others populate ontologies using database data, and there are another which represent databases as ontologies.

Fuzzy Catalog Ontology

Fuzzy Catalog Ontology has been defined to represent the fuzzy RDBMS Catalog. This Ontology contains all the relational concepts defined in the ANSI Standard SQL, for example, the concepts of schema, Table, Column, Constraints, Data Types, etc, and thefuzzy structures extension to manage flexible information: Fuzzy Column, fuzzy labels, fuzzy discrete values and fuzzy data types. Moreover, the following fuzzy structures have been added to this ontology to complete the fuzzy RDBMS description:-

- **Fuzzy Constraints:** These restrictions, which are described in table can only be applied to fuzzy domains and are used either alone or in combination to generate domains such as, for example, not known, undefined, or null values are allowed, or only labels are allowed
- **Fuzzy Domains:** These represent a set of values that can be used by one or more attributes. They are defined by a fuzzy data type, one or more fuzzy constraints, and those labels or discrete values that describe this domain

Fuzzy Schema Ontology A fuzzy schema ontology is a domain ontology that represents a specific FDB schema. This ontology is generated using the previously defined fuzzy schema as instances of the Fuzzy Catalog Ontology. The generation process for this ontology has been described previously and consists of translating the Table instances into classes, and the attributes instances into properties. The constraints restrictions establish the connections between the attributes properties and the fuzzy data structures. For example, the object property of the Tall attributes allows Trapezoidal, Approximate, Crisp, Interval, Labels values to be defined but not Null, Unknown or Undefined ones. Moreover, the previously defined structures, namely Fuzzy Labels, and Discrete Tags, are included in the Schema Ontology. The result is a mixed ontology containing the Fuzzy Schema Ontology and the Fuzzy Catalog Ontology instances or a new ontology where the fuzzy values are imported as a separate ontology.

The GEFRED model

The GEFRED model (**GE**neralised model **F**uzzy heart **R**elational **D**atabase) has been proposed in 1994 by Medina et al. It is based on the generalized fuzzy domain and generalized fuzzy relation, which include classic domains and classic relations, respectively.

To model the flexible queries and the concept of the fuzzy attributes, Medina et al. introduced an extension of SQL language called FSQL (Fuzzy SQL). The GEFRED model is based on the generalized fuzzy domain (D) and generalized fuzzy relation (R), which include classic domains and classic relations, respectively.

Ontology Database Description

An Fuzzy Catalog ontology work [3] which describes the ANSI SQL 2003 specification is defined by Martinez-Cruz et al. This ontology models fuzzy data specification presented in the GEFRED model, that is a fuzzy relational database catalog. 28

Henceforth, this metaontology, called Fuzzy Catalog Ontology [3], models all fuzzy relational database structures (the system catalog) such as fuzzy data types, fuzzy columns, fuzzy constraints, etc. and their relations with classical SQL elements. Thus, a database schema, that manages fuzzy or classic data, is viewed as an ontology regardless of any RDBMS implementation constraint. This representation presents a fuzzy database schema from two different perspectives: The first one, a fuzzy schema is presented as a set of instances of the Fuzzy Catalog Ontology [3]. The second one, a fuzzy schema is presented as a domain ontology where a set of classes, properties and constraints defines the reality and data (or tuples) are defined as instances of this ontology. This ontology is generated automatically from the instances of the Fuzzy Catalog Ontology.

IV. PROBLEM DESCRIPTION

•RDBMS is used to store crisp data and SQL Language is used to intact with database.Now we want to store two types of data in RDBMS.

1. Crisp data
2. Fuzzy data

• Now problem is SQL language does not work with fuzzy values and how fuzzy values is stored in database.

EXISTING PROBLEM

Fuzzy RDBMS requires complex data structures, in most cases, are dependent on the platform in which they are implemented.FRDB systems are poorly portable and scalable, even when implemented in standard RDBs. The solution is to use Ontology which makes system independent of platform used.

All the previous work which uses Ontology are used for web application, none of the work is carried out for window application. Here, a new Ontology is proposed for the above problem.

PROPOSED SOLUTION

The proposed Ontology, whose definition is extended in this project, provides a frame where fuzzy data are defined in a platform-independent manner.

An implementation layer, which is responsible for parsing and translating user requests into the corresponding DB implementations in transparent is required to establish communication between the Ontology and the relational databases management system(RDBMS).

Fuzzy Query Ontology

An ontology, called from now Fuzzy Query Ontology (FQO), represents all the basic fuzzy relational database query constituents to get a flexible Select clause definition. After the instantiation of this ontology, the query process can start. This Ontology is specially designed for localhost application. Database queries can be viewed as ontologies from two different 30

perspectives: The first one, a query is a set of descriptions, conditions and rules defined as a set of instances of a Fuzzy Query ontology. In this sense, this query can be viewed as a SQLf [3] statement where any element of the sentence is modeled in the ontology. The second one, a query is described as aa reduced domain ontology where a set of classes, properties and axioms represent a query specification and the ontology instances represent the resulting tuples. A diagram of this ontology is shown in figure and it is described below

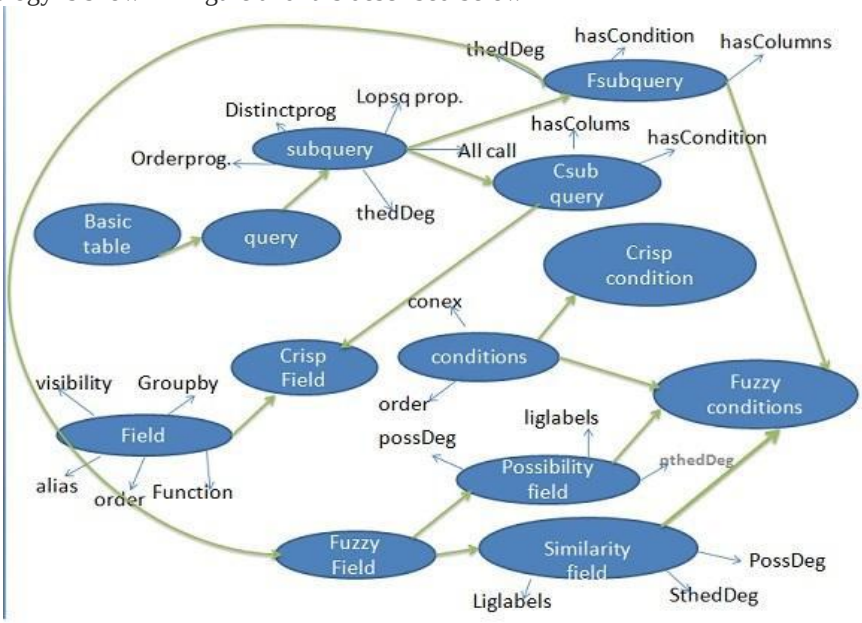


Fig 4. Architecture of Fuzzy Query Ontology

Query class: This class represents a query composed of one or more subqueries joined by logical connectors.

• **Subquery class**: It is a simple query composed of a list of ordered columns. This class has two subclasses: classic and fuzzy subqueries. Subquery properties are: – distinct prop: establishes if the result set allows duplicated registers. – order prop: establishes the order of a subquery in the general query structure. – LOpsq prop: establishes the logical operator to integrate this subquery in the main query structure. The most common operators are: Union and Intersection (but they can be extended). – thedDeg: establishes the threshold to accomplish the logic operation. – allColl: establishes if all the columns are displayed in the answer.

• **Crispquery class**: It is a subquery that only uses classical data. A classic subquery involves only classic columns. This representation can be made as a SQL sentence as well. – hasColumns: establishes which columns are involved in a subquery. – hasCondition: establishes the filtering conditions to get results. – having: establishes the filtering conditions to get results in an aggregation operation.

• **FuzzyQuery class**: Subquery that uses fuzzy columns. This kind of queries can also use classical data. This class is represented by the properties: – thedDeg prop: represents a threshold that the resulting records of the fuzzy query must accomplish. – hasColumns prop: establishes the columns involved in a subquery. The range of this property is Field class. – hasCondition prop: establishes the filtering condition to get results. This property goes to conditions class because conditions involved in this subquery can be fuzzy or not. – having prop( ): establishes the filtering condition to get result in an aggregation operation.

• **Field class**: represents any column involved in an specific subquery, no matter if they are visible or not. These columns are related with columns defined in the Schema Ontology Description modeled in . It has two subclasses: fuzzyField and crispField. These classes have the following properties: – visibility: determines if this column will be in the resulting set. – alias: establishes the final name of the column. – order: establishes the order of the column in the result set. – function( ): establishes the operation to do in this column. This can be a simple operation or an aggregation function (AVG, MAX, MIN, COUNT, Other). – groupBy( ): establishes if this column is used to make a group. – name: establishes the relation between this column and the one defined in the schema definition.

• **FuzzyField class**: represents any column involved in an fuzzy subquery, no matter if they are visible or not. It has two subclasses: PossibiliticField and Similarity Field class

• **PossibiliticField class**: this class deals with type 2 of fuzzy attributes[1] which is discussed above

- possDeg: gives a degree of possibilitic distribution approach
- liglabels: deals with linguistic labels with its degree
- pthedDeg: represents a threshold that the resulting records of the fuzzy query must accomplish. • **SimiliarityField class**: this class deals with fuzzy attributes type 3 [1] which is already discussed.
- possDeg: gives a degree of similarity distribution approach
- liglabels: deals with linguistic labels with its degree
- sthedDeg: represents a threshold that the resulting records of the fuzzy query must accomplish.

• **Condition class**: This class represents any subquery condition to filter data. All conditions are related with a column or a set of them, one operator, and a condition object. This class has two subclasses depending if there are fuzzy: fuzzyConditions and Classic conditions. Condition class properties are: – order: establishes the order in the list of conditionsin a subquery. – conex: establishes the logical operator that join this condition in a subquery: AND, OR. – not: establishes if the condition is negated or not.

• **ClassicCondition class**: This class represents any classic condition, and the property origin defines the list of columns to be compared with.

• **FuzzyConditions class**: this class represents any fuzzy conditions, and the property origins that defines the list of columns to be compared with.

V. DESIGN & IMPLEMENTATION OF FUZZY QUERY ONTOLOGY

An implementation layer, which is responsible for parsing and translating user requests into the corresponding DB implementations in transparent, is required to establish communication between the Ontology and the relational databases management system (RDBMS). This implementation is designed with respect to corporate.

Parsing Technique Used

A parser generator which generates fully featured object-oriented frameworks for building ompilers, interpreters and other text parsers. In particular, generated frameworks include intuitive strictlytyped abstract syntax trees and treewalkers. SableCC [8] also keeps a clean separation betweenmachinegenerated code and user-written code which leads toa shorter development cycle. An object-oriented framework that generates compilers(and interpreters) in the Java programming language. This

Framework is based on two fundamental design decisions. Firstly, the framework uses object-oriented techniques to automatically build a strictly-typed abstract syntax tree thatmatches the grammar of the compiled language and simplifies debugging. Secondly, the framework generates tree-walker classes

Using an extended version of the visitor design pattern whichenables the implementation of actions on the nodes of theabstract syntax tree using inheritance. These two designdecisions lead to a tool that supports a shorter developmentcycle for constructing compilers.

Fuzzy Query Translator

The queries are written in SQLf which is an extension toSQL. Fuzzy Query Executer works as a translator, thattranslates fuzzy queries to standard, SQL queries, andexecutes them with RDBMS.

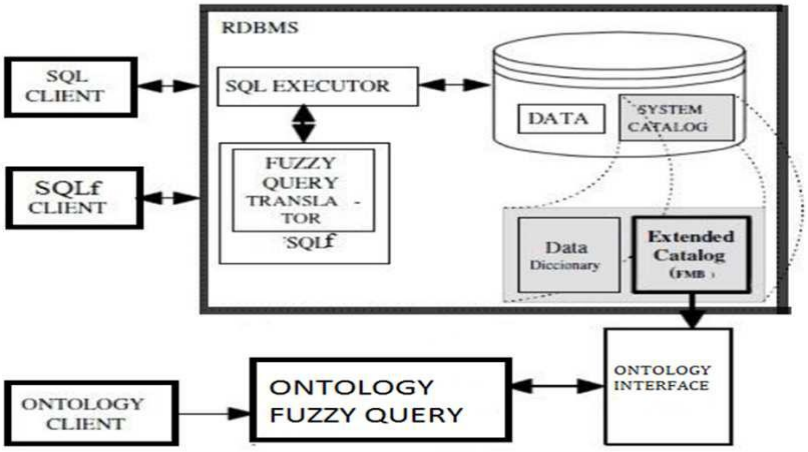


Figure 5. Block Diagram of Fuzzy Query Ontology

SQL Client directly interact with Relational DatabaseManagement System through SQL Executer and do not needany translation in between. SQLf is language used in theproject tO retrieve data which is fuzzy in nature. SQLf Client needs translation because Database can understand only SQLLanguage. So, Fuzzy Query Translator is used to convertSQLf query into SQL query. Ontology which is used in thisproject is independent of the platform used. Figure showsontology is outside of RDBMS. Fuzzy Query Ontology isimplemented in this project through various classes and itsproperties and behavior. A FMB is needed. In relational database, The FMB will be responsible for organizing all theinformation related to the inexact nature or context of theseattributes. The FMB is contemplated as an extension of the catalogue of the

system (Data Dictionary),FMB is created as shown in Block Diagram figure 2.
FMB contains few tables in database which stores the fuzzyattributes with their parameters to calculate fuzzy degreewhich handles the fuzzy database. This Block Diagram showshow data is carried in database. An ontology for fuzzyinformation representation is proposed in this paper. Theontology includes knowledge about how to manage andrepresent uncertain and imprecise data. Figure 2 shows howthe system catalog is related to the ontology modeling it. TheOntology Client module carries out the same operationsthrough the Ontology than the DBMS Clients. The connectionbetween the ontology and the database needs an interface, theOntology Interface, which establishes the communication andrefreshes the data. The fuzzy model representation comprisestwo well-differentiated parts. Firstly, the ontology must definethe necessary classes and slots to represent the metadata. Secondly, the ontology will be able to represent classical orfuzzy information as instances of the relations. In thisontology, metadata establish how the fuzzy information willbe stored.

Verification of Implemented Fuzzy Query Ontology

This project is implemented in context of some workers who are working for a various organization like Academics, government, Industry. The job expertise for organizations are AI, Statistics, Robotics, Expert System. There are various fuzzy attributes like age, salary, expertise and there are various linguistic labels for fuzzy attributes like age there are BABY,YOUNG, MIDDLE, MATURE, OLD and for salary labels are SMALL, MIDDLE, HIGH

Examples:

1. Select name of worker, location of worker and age of worker from table name workers where fuzzy attribute age is ~young‘;

```
SQL_FQE>select fname,lname, age from workers where ~age is 'young';
+---------+-------------+-----+---------------+
| fname   | lname       | age | fuzzy_degree  |
+---------+-------------+-----+---------------+
| neha    | ghaziabad   |  16 |  0.933333333  |
| mandeep | meerut      |  23 |  0.466666667  |
| rajesh  | muradnagar  |  27 |  0.200000000  |
+---------+-------------+-----+---------------+
3 rows in set
OK
SQL_FQE>
```

Figure 6.Shows the result of example 1.

2. Select, name of worker and age of worker from table name workers where fuzzy attribute age is ~’age’;

```
SQL_FQE>select fname ,age ,~age from workers order by age;
+-----------+------+-----------------------------+----------------+
| fname     | age  | p_l_linguistic_variable     | fuzzy_degree   |
+-----------+------+-----------------------------+----------------+
| neha      |  16  | YOUNG                       |  0.933333333   |
| neha      |  16  | MIDDLE                      |  0.066666667   |
| mandeep   |  23  | MIDDLE                      |  0.533333333   |
| mandeep   |  23  | YOUNG                       |  0.466666667   |
| rajesh    |  27  | MIDDLE                      |  0.800000000   |
| rajesh    |  27  | YOUNG                       |  0.200000000   |
| jogi      |  38  | MATURE                      |  0.533333333   |
| jogi      |  38  | MIDDLE                      |  0.466666667   |
| abhishek  |  40  | MATURE                      |  0.666666667   |
| abhishek  |  40  | MIDDLE                      |  0.333333333   |
| ravikant  |  55  | OLD                         |  0.666666667   |
| ravikant  |  55  | MATURE                      |  0.333333333   |
```

Figure 7.Shows the result of example 2.

3. Select threshold value , name of worker, location of worker and age of worker from table name workers where fuzzy attribute age is ~’age’,’salary’;

```
SQL_FQE>select threshold 0.5 FName, LName, Age, Salary from workers where ~Age is 'YOUNG' and ~Salary is 'MIDDLE' or ~Salary is 'HIGH';

+----------+------------+-----+--------+--------------+
| fname    | lname      | age | salary | fuzzy_degree |
+----------+------------+-----+--------+--------------+
| ravikant | delhi      |  55 |   6500 |  1.000000000 |
| jogi     | delhi      |  38 |   6200 |  1.000000000 |
| neha     | ghaziabad  |  16 |   2400 |  0.933333333 |
| rajesh   | muradnagar |  27 |   3800 |  0.520000000 |
+----------+------------+-----+--------+--------------+
4 rows in set
OK
SQL_FQE>
```

Figure 8.Shows the result of example3.

VI. CONCLUSION AND FUTURE WORK

The above defined Ontology is specific to data retrievalfrom database. It is specifically designed for a flexible Selectclause which represents all the basic fuzzy relational databasequery constituents. In this proposal of an ontology whichrepresents a query structure regardless of any RDBMSimplementation is defined. This ontology allows generatingand executing any query on fuzzy or classical data accordingto the system where it is executed. The use of ontologies hasprovided of an independent layer where data can be modeledregardless of any RDBMS implementation

particularity. Thus,the client interaction is performed through this ontology andan interface between the ontology and a real RDBMS is incharge of establishing the communication. A new Ontology isdefined in this dissertation called Fuzzy Query Ontology.

REFERENCES

[1] Geetanjalityagi, kumarkaushik, Arnika Jain, Manish Bhardwaj, Ontology Based Fuzzy Query Execution, American Journal of Networks and Communications. Special Issue:Ad Hoc Networks. Vol. 4, No. 3-1, 2015, pp. 16-21. doi: 10.11648/ j.and.s.2015040301.14

[2] Carmen Martinez-Cruz..“An Ontology to represent Queries inFuzzy Relational Databases”. IEEE 2011

[3] J. Galindo, A. Urrutia, and M. Piattini, “Fuzzy DatabaseModeling, Design and Implementation”. Idea Group Publishing, 2006

[4] “Describing Fuzzy Database DB Schemas as Ontologies: ASystem Architecture View”. Springer 2010. 13 th InternationalConference, IPMU july 2010.

[5] Using search engine optimization technique increasing website traffic and online visibility.http://www.ijarcsse.com/docs/papers/Volume_5/1_January2015/V5I1-0338.pdf

[6] Protege, tool for creating and editing ontologies.http://protege.stanford.edu/, 2011.

[7] M. Neunerdt, B. Trevisan, T. C. Teixeira, R. Mathar, and E.- M.Jakobs, “Ontology-based corpus generation for web commentanalysis,” in ACM conference on Hypertext and hypermedia(HT 2011), (Eindhoven), 05 2011.

