Position Papers of the Federated Conference on
Computer Science and Information Systems pp. 31–38

DOI: 10.15439/2015F1
ACSIS, Vol. 6

Processing Imprecise Database Queries by Fuzzy Clustering Algorithms

Anna Kowalczyk-Niewiadomy
Technical University of Lodz
Lodz, Poland
anna.kowalczyk-niewiadomy@p.lodz.pl

Adam Pelikant
Technical University of Lodz
Lodz, Poland
adam.pelikant@p.lodz.pl

***Abstract*—Nowadays database management systems are one of the most critical resources in every company. Despite advanced possibilities of SQL, relational database management systems do not support flexible query conditions.**

Main assumptions of this work were two facts. First, that real data not representing random distribution (white noise), but have natural trend to granularity. The second one, that in everyday contacts we do not using strict defined conditions. The second feature lead us to use fuzzy logic which closer representing natural communication. First gives us opportunity to automatically construct functions defining membership to discrete groups based only on data distribution.

The problem of extending database systems with natural language expressions is a matter of many research centers. The basic idea of presented research is to extend an existing query language and make database systems able to satisfy user needs more closely. This paper deals mostly with gaining imprecise information from relational database systems. Presented concept is based on fuzzy sets and automatic clustering techniques that allow to build membership function and fuzzy queries. Thanks to applied solutions, the relational database system is more flexible, and similar to natural way of communication.

***Index Terms*—fuzzy logic, fuzzy sets, fuzzy set theory, fuzzy systems, fuzzy clustering, FSQL, fuzzy queries, imprecise queries.**

I. Introduction

THE SECOND half of the twentieth century ushered in rapid development of technology, especially in information technology. The growing demand for storing and processing huge data sets has resulted in evolution of database management systems. Those systems are designed to ensure the cohesion and safety of stored data and their principal objective is to search large data sets efficiently. Despite advanced possibilities of SQL, it is restricted to the precise communication only. In most business applications, querying precise values or using standard sharp relationships and traditional methods of data aggregation is absolutely sufficient. However, in some cases a standard SQL language, which is based on three-value logic, is not flexible enough. For example, if one is looking for cheap accommodation, or wish to buy a house that costs around €100 000, it is impossible to get results that will satisfy him, by means of traditional precise query language. Both of presented queries use natural language features that are used in everyday life. Traditional SQL is not feasible to build a query that supports such imprecise expressions. Imprecision in such context should not be seen as a drawback, but on the contrary, it allows expressing true needs, preferences and evaluation.

II. The Current State Of Knowledge And Relevance Of Research

Over the years, traditional methods of searching for information based on the precise conditions are more often replaced by methods based on fuzzy logic elements. The first fuzzy query language was presented by Takahashi in 1991 [17]. Two years later he published the full theory of two languages: calculus query language and fuzzy algebra query language [18]. In the eighties the problem of fuzzy database were investigated by: Zamenkowa [23], Chang Ke [7], Buckles and Petry [4], [5].

In the early nineties, thanks to the rapid information technology development, we could notice first implementations of fuzzy query systems. In France, P.Bosc and O. Pivert designed SQLf – fuzzy language which allows getting imprecise information from database [3].

Almost at the same time, in Poland professor S. Zadrożny and professor J. Kasprzyk from Systems Research Institute at Polish Academy of Science (PAS), presented FQUERY system for Access database. FQUERY consists of tools that enable user building queries with fuzzy values, relations, linguistic modifiers

and quantifiers. Currently, scientists who were mentioned above, work on linguistic summaries of databases problems and publish their achievements together with P.Bosc and O. Pivert from Malaga [4].

After the year 2000, there were presented newer solutions based on today's leading database management systems. For example, the research team led by Dr. Jose Gomes Galino of the University of Malaga, have developed system FSQL for Oracle 7/8, available on the Internet [12]. In Poland, Technical University of Poznan [9] and Silesian University of Technology [11] designed their own fuzzy systems SQLf and Fuzzy Logic Management System.

Despite many implemented systems for query languages, this branch of science is still being investigated and requires extensions to cope with the growing demands. So far, solutions based on the fuzzy sets theory contain strong constraints on the design stage of fuzzy sets. The developed systems are based on rigidly defined membership functions, and therefore require cooperation with expert's knowledge.

The basic idea of our research is to redesign a fuzzy structured query language system by extending traditional SQL (Oracle 11g) with condition definition similar to natural language. Due to the fact that few fuzzy query systems already exist in Polish as well as foreign research centers, it is worth to emphasize that the innovative element of this work is development of universal, multi-dimensional algorithms, which automatically generate fuzzy sets, based on the real data distribution. There is no need to use expert knowledge while original algorithms based on fuzzy clustering methods are implemented. In addition, conducting a comprehensive analysis of standardization issues and the labeling process enabled implementation of an intelligent module responsible for the allocation of labels according to the automatically generated fuzzy sets. There are some arguments to build such solution. The work of branch experts generates additional high costs. In many cases the meaning of label changes as a result of data distribution changes. For example prices of apartments usually grow up, so that meaning of cheap and expensive flat changes as well. Additionally in times of crisis prices can rapidly drop. Any of these states requires the help of experts, which can be avoided if the proposed solution is used. The approach (in more detail discussed in Chapter VI) is a completely new idea in the fuzzy SQL language issues.

III. Fuzzy Clustering Algorithms

Data clustering is a process of assigning a set of objects into groups (called clusters) so that the objects in the same cluster are more similar (in some sense or another) to each other than to those in other clusters. Each data point belongs to a cluster of a degree of membership grade. This paper reviews three of the most representative clustering techniques: Fuzzy C Means, Fuzzy C Medoids clustering and Mountain clustering. All techniques were implemented and tested against automatic fuzzy sets generation problem. Fuzzy clustering methods together with the author's algorithm and the trapezoidal membership function allowed generating fuzzy sets based on the actual data distribution.

A. Equations

Let X be a set of n patterns described by $X = \{x_1, x_2, \dots, x_n\}$. Let c be an assumed number of clusters. $C = \{c_j | \ 1 \le j \le c\}$ is the set of centers. The notation $u_{ij}(1 \le i \le n, 1 \le j \le c)$ indicates the degree of membership of the i-th sample to the j-th prototype. The membership matrix *U* is limited to values between 0 and 1. However, the summation of degrees of belongingness of a data point to all clusters is always equal to unity (1).

$$\sum_{j=1}^{c} u_{ij} = 1; \ 1 \le i \le n \tag{1}$$

The Fuzzy C-means method was proposed in 1973 by Dunn [10] and modified in 1981 by Bezdek [1]. The algorithm is based on clusters search in a data set, such that an objective function (2) of distance measure is minimized. The squared distance is weighted by the m-th power of the membership in cluster j.

$$J_m(U,c) = \sum_{i=1}^{n}\sum_{j=1}^{c} u_{ij}^m \left\|x_i - c_j\right\|^2 \tag{2}$$
$$1 \le m \le \infty$$

Distance measure can be expressed by one of specific forms general Minkowski (3) norm [20]

$$d_n(x_i, x_j) = \left(\sum_{k=1}^{d} \left(|x_{i,k} - x_{j,k}|\right)^n\right)^{\frac{1}{n}} \tag{3}$$

One should remember that the result of clustering depends on kind of selected metric. The (4) and (5) are mandatory conditions for equation (2) to reach its minimum.

$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|}\right)^{\frac{2}{m-1}}} \tag{4}$$

$$c_j = \frac{\sum_{i=1}^{n} u_{ij}^m x_i}{\sum_{i=1}^{n} u_{ij}^m} \tag{5}$$

The algorithm works iteratively through the preceding two conditions until there is no more improvement noticed. FCM calculates cluster centers and the membership matrix U using the steps presented at figure Fig 1.

The main advantage of the FCM algorithm is high performance and low hardware requirements. Unfortunately, this algorithm has three major drawbacks. First, the final distribution of objects between clusters strongly depends on the assumed number of clusters. Second, the performance of FCM depends on the initial membership matrix values. It is advised to run the algorithm for several times, every time starting with new values of membership grades for data points. Third, the algorithm is sensitive to disrupted data (e.g. singular point).

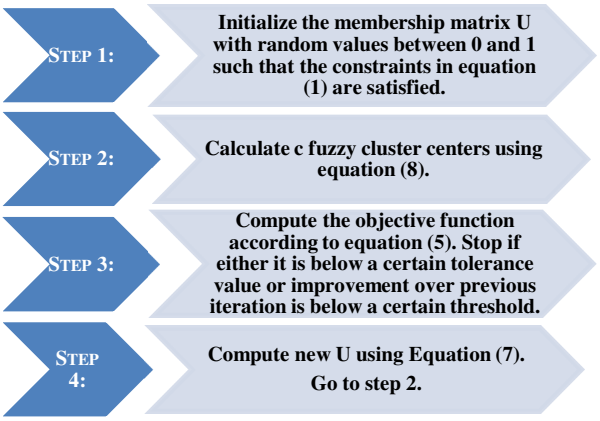

Fig 1. FCM – processing steps.

In order to solve this problem, instead of calculating mean we can search the most centrally located cluster point called medoid. In this way, the occurrence of the singular point in the cluster will not cause significant disruptions.

B. *Fuzzy C-Medoids Clustering (FCMdd)*

Fuzzy C-Medoids Clustering [8], relies on the basic idea of Fuzzy C-means clustering (FCM) with the difference of calculating cluster centers. The change has a significant influence on the efficiency of the algorithm. Instead of searching for means (calculated as a simple arithmetic formula) we need to process several steps over the neighbor points to find medoids. The improvement (minimization) of the criterion function (6) is much more complex and expensive. The notation $r(x_i, v_j)$ indicates dissimilarity between the x_i sample and v_j medoid.

$$J_m(V;X) = \sum_{i=1}^{n}\sum_{j=1}^{c} u_{ij}^{m} r(x_i, v_j) \qquad (6)$$

Membership matrix (u) is calculated according to (7):

$$u_{ij} = \frac{\left(\frac{1}{r(x_j,v_i)}\right)^{\frac{1}{m-1}}}{\sum_{k=1}^{c}\left(\frac{1}{r(x_j,v_k)}\right)^{\frac{1}{m-1}}} \qquad (7)$$

Figure 2 presents basic steps of the FCMdd algorithm.

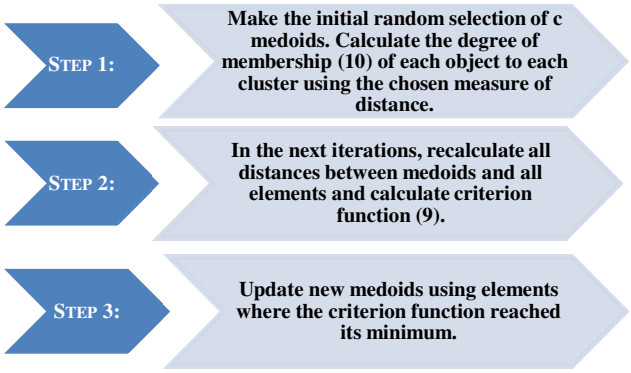

Fig 2. FCMdd-processing steps

C. *Mountain Clustering*

The mountain clustering method, proposed by Yager and Filev [22] is a simple and effective algorithm based on a density measure called the mountain function. It is based on three main steps. The first one involves forming a grid on the data space, where the intersections of the grid lines constitute the potential cluster centers. The second step entails construction of a mountain function representing data density measure. The height of the mountain function at a point $v \in V$ is equal to (8):

$$m(v) = \sum_{i=1}^{N} e^{-\left(\frac{\|v-x_i\|^2}{2\sigma^2}\right)} \qquad (8)$$

where x_i is the i-th data point and σ is an application specific constant. The third step involves selection of the cluster centers by sequentially reducing the mountain function. This is done by modification of the mountain function to the form represented by equation (9):

$$m_{new}(v) = m(v) - m(c_1)e^{-\left(\frac{\|v-c_1\|^2}{2\beta^2}\right)} \qquad (9)$$

IV. CLUSTER VALIDATION PROBLEM

The problem of data clustering is quite complex, what is mainly caused by wide potential of methods usage. Depending on the situation there is a need to use different types of algorithms, so it is difficult to impose a universal method. One of the main subjects in data clustering is evaluation of the result of clustering algorithms (cluster validation). More precisely, the

cluster validation problem is to find an objective criterion to determine how good a partition generated by a clustering algorithm is. Since most clustering algorithms require a pre-assumed number of clusters, a validation criterion to find an optimal number of clusters would be very beneficial.

The first validation was associated with the FCM partition coefficient proposed by Bezdek [1], defined by (10):

$$I_{PC} = \frac{1}{n}\sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^2 \tag{10}$$

To produce a better clustering performance we find optimal cluster numbers for $\max_{2\le c\le n-1} I_{PC}$.

Partition entropy was also proposed by Bezdek for the Fuzzy C-Means algorithm and it is defined by the following equation (11):

$$I_{PE} = -\frac{1}{n}\sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}\log_2 u_{ij} \tag{11}$$

To produce better clustering performance we find optimal cluster numbers for $\min_{2\le c\le n-1} I_{PE}$.

In 1991 Xie and Beni [19] proposed a validation index based on compactness and separation defined as (12):

$$I_{XB} = \frac{\sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^m \left\|x_j - v_i\right\|^2}{n \min\limits_{i\ne j} i,j \left\|x_j - v_i\right\|^2} \tag{12}$$

In 2011 Rubio, Castillo and Melin [16] compared the most commonly used indices such as I_{PC}, I_{PE}, I_{XB} and proposed its own (13) proving its greatest effectiveness.

$$I_{RCM} = I_{MPE} + D_M \tag{13}$$

where

$$I_{MPE} = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}{}^2 \log_2 u_{ij} \tag{14}$$

$$D_{M_k} = \sum_{\substack{i,j=1 \\ i\ne j}}^{k} \left\|M_i - M_j\right\|^2, k = 1,\dots,c \tag{15}$$

During the research **all presented above validity indexes were implemented** and used in the process of fuzzy sets generation. In order to evaluate the quality of the indexes we used multi-dimensional data structures from the actual Machine Learning Repository maintained by the Center for Machine Learning and Intelligent Systems, University of California, Irvine [6]. Detailed experiments revealed that validity indexes based on compactness and separation are most effectiveness, that is why in our fuzzy SQL system the Rubio and Xie-Beni indexes were most important.

V. FUZZY SQL SYSTEM DESIGN

The main idea of our research was to design and implement system, which extends traditional SQL with condition build in the way similar to natural language. The great difference between proposed solution and already existing similar systems is fully automated generation of membership functions and fuzzy sets [13]. In addition to this, the automatic labeling module is also novelty[14], [15]. The project consists of three main modules described below.

The basic problem of testing is to gather representative data set. It should provide the actual distribution of the data and their continuous growth. The tests were carried out on several public available datasets. Finally we decided to build and positioned the dedicated WWW portal with tutorials for some programing and database platforms. Because the data set is one of the most important factor for all tests realized for implementing algorithms we decided to use selected Google Analytics statistics of the website traffic (fig. 3):

- number of visits (total and unique users);
- number of page views;
- percentage of rejections (input of 1 page views);
- average time spent on the site;
- loyalty (percent of new visits).

Fig 3. Exemplary overview of Google Analytics for defined data period.

Results describing portal users activity store in Google Analytics were exported to database server. This step concerns gathering an input data, and preparing database model for the main processing are presented at Fig 4.

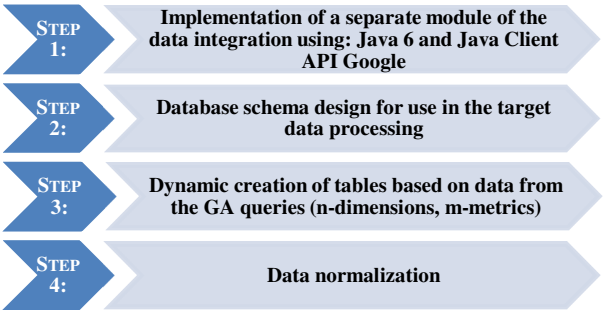

Fig 4. Data preparation diagram.

The second element of created system is a set of tools responsible for fuzzy queries processing consisting of the following components:

- query parser, which decomposes the fuzzy query into understandable by the system and database elements;
- fuzzy sets generator – a complex module using fuzzy clustering methods, novel fuzzy sets algorithms, validity indexes, T/S norms implementations etc.;
- labels assignment process – each fuzzy set is addressed by appropriate label or labels;
- fuzzy conditions, operators and aggregate functions executor.

The third element of the project – the client application is responsible for construction fuzzy queries (tree view and text) and presentation of the results.

A. *Data preparation*

In order to collect a sufficient data set, there was a need to prepare (design, implement and publish) an exemplary website which provides tutorials for web graphics (3dStudioMax, After Effects, Photoshop). Website traffic statistics are a good source of natural input data. In our project the Google Analytics (GA) was used as a statistical data warehouse. GA is a free, online tool mainly used to analyze websites statistics. GA is a powerful tool with variety of functionalities accessible via web browser and what was very useful via Java API.

One of the most important elements of the research was to prepare software in order to communicate easily with the GA. By use Java programming language, Java GA API (JGA), and JDBC the integration with the Oracle database was provided. Figure 4 presents basic GA integration steps.

It is worth to emphasize that data prepared for fuzzy clustering algorithms are normalized to the <0, 100> range in order to eliminate the problem of scale, negative values and to ensure the integrity of the generated fuzzy sets. Standardization refers to both input data as well as a range of labels. In this way, it is possible to assign labels by percentage match in order to avoid context dependency. For example query using "high temperature" expression in context of the weather, boiling water or melting metals is completely different. Data normalization eliminates this problem.

GA enables lots of important statistics about website traffic. It is impossible to discuss all of them, so we focused on most commonly used like metrics: visits, pageviews, visit duration, avg time on page, bounce rate, %new visits. Such data can be analyzed in the following dimensions: the date (hour, day, month, year), location - source of visits (continent, country, city), the type, version and parameters of a web browser (IE, FF,) language etc.

B. *Data processing*

Our fuzzy SQL interpreter enables natural language expressions, so labeling module was designed and implemented. Firstly, it was necessary to define labels with the gradation of "strength" of each label (appropriate thresholds was required). Labels of the same type (e.g. short, average, tall) are combined into sets. Each label set is assigned to the attribute e.g. the attribute “time on site” can be short or long and the attribute “number of visits” can be small, average or big. The process of assigning labels to fuzzy set causes some difficulties connected to following issues [16]:

- each attribute may have different number of labels;
- each clustering process can generate different number of clusters. Presented issues were coped in the implemented algorithm.

Figure 5 presents an exemplary data distribution used in a labeling process. The naive approach is to evenly split the range of variation attribute according to the number of labels. However, it ignores the variable distribution and fuzzification process.

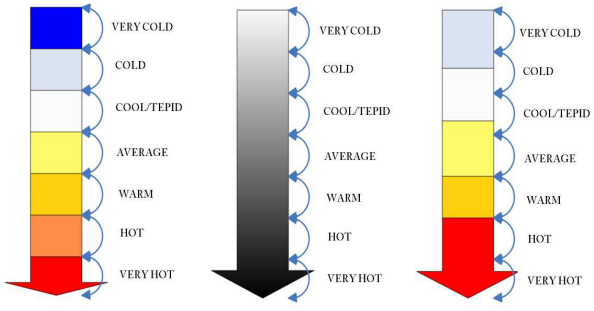

Fig 5. Exemplary data distribution used in a labeling process.

In the case of same number of clusters and assigning labels stems from the method of ordering. If number of clusters and labels are different assignment is performed by comparing the position of the centers of gravity. The last of these methods was used during creation of fuzzy query processing system.

C. *Fuzzy query processing*

In order to process the fuzzy query and generate fuzzy sets automatically, the fuzzy clustering methods, label-attribute and operator logic were implemented (Fig 6). For each distinct attribute to get most relevant results, the algorithms run several times, each time starting with new clustering parameters (different number of clusters, distance measures and start points). After the validity indexes are calculated the cluster count is adjusted on the basis of Xie-Beni and Rubio calculation for selected attribute/attributes. Next, the results of clustering method are processed by novel algorithms [13] based on membership functions in order to generated fuzzy sets with triangle and trapezoid membership functions.

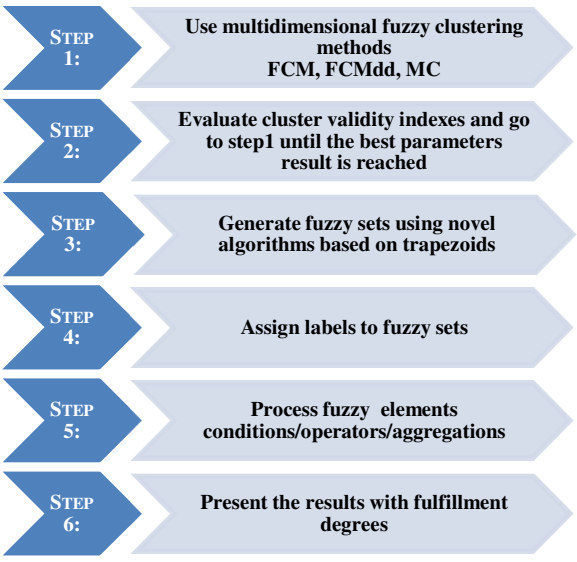

Fig 6. Multidimensional data processing.

With such prepared data the intelligent module responsible for label allocation according to the automatically generated fuzzy sets and labeling criteria, generates a set of pair-value assignments. Finally generated data collection contains fuzzy set to label matches eg. label_small » fuzzy_set_#1, label_medium » {fuzzy_set_#2 and fuzzy_set_#3} etc.

After that all the operations for current attribute based on T Norms and S Norms are processed according to query logic with respect to supplied order and operators priorities. In case of query with multiple arguments, output of processing of single arguments are joined with respect to order and operators and the final result is built from the composition of each attributes output. The mid results are combined by the use of primary keys or ROWID from primary flat SQL query. Finally the client application presents query results ordered by membership degree.

VI. EXAMPLES

SQL queries containing imprecise conditions look quite similar to traditional queries. SQL query is extended by FuzzyWhere, FuzzyHaving clauses. Figure 7 presents fuzzy query syntax, containing standard and fuzzy conditions, logical operators (or/and/not) and fuzzy aggregation conditions.

In this section, the authors present the difference between traditional SQL queries and imprecise FSQL. average number of page views during a single visit. To illustrate the possibilities of application, shown simultaneously all of available imprecise filters on all levels we can set following problem.

It is necessary to find the countries from which comes a large number of visitors, in which the average time spent on the site by users is significant.

Let's consider classical SQL syntax query solved this problem that may look like listed on figure 8. The result for query is presented on fig 9.

Similar query in FSQL language may look like listed on figure Fig 10. The result set is not restricted by crisp conditions so it contains more values Fig 11.

```
SELECT {EXPRESSIONS, AGGREGATE FUNCTIONS AGG } FROM {TABLENAME/S, VIEW/S}
WHERE {CONDITIONS}
FUZZYWHERE MINIMUMMEMBERSHIP FUZZYLABEL1(ATTRIBUTE1) AND
FUZZYLABEL2(ATTRIBUTE1)OR FUZZYLABEL1(ATTRIBUTE2) AND NOT
FUZZYLABEL3(ATTRIBUTE3)
GROUP BY {GROUP CONDITION} HAVING <CONDITIONS>
FUZZYHAVING MINIMUMMEMBERSHIP FUZZYLABEL1(AGG)
```

Fig 7. Imprecise query syntax.

```
SELECT COUNTRY, AVG (PERCENTNEWVISITS), AVG (AVGTIMEONPAGE)
FROM WEB_DATA_TST
WHERE PERCENTNEWVISITS > 90 AND AVGTIMEONPAGE > 200
GROUP BY COUNTRY
```

Fig 8. An exemplary traditional query.

COUNTRY	AVG(PERCENTNEWVISITS)	AVG(AVGTIMEONPAGE)
USA	97	225
India	96	325
Philippines	93	218
United Kingdom	92	308
Germany	92	243
Romania	90	224

Fig 9. Traditional query's results

```
SELECT COUNTRY, AVG(PERCENTNEWVISITS), AVG(AVGTIMEONPAGE)
FROM WEB_DATA_TST
FUZZYWHERE 0,5 LARGE(PERCENTNEWVISITS)AND LONG(AVGTIMEONPAGE)
GROUP BY COUNTRY
```

Fig 10. Imprecise query.

COUNTRY	AVG(PERCENTNEWVISITS)	AVG(AVGTIMEONPAGE)	SATIS
USA	97	225	0.97
India	96	325	0.96
Romania	90	224	0.95
Philippines	96	207	0.95
Germany	92	217	0.94
UK Kingdom	95	216	0.94
Brazil	94	198	0.94
Myanmar	90	196	0.93
Poland	94	196	0.93
Argentina	90	211	0.92
Canada	89	209	0.90
Tanzania	90	186	0.82

Fig 11. Imprecise query's results.

In the presented realization of query besides the definition of fuzzy labels for expressions components in FUZZYWHERE clause is possible to define the minimum cluster membership degree. This allows to control the number of elements in resulting records set. In the resulting table presents an additional column describing the degree of fulfillment of fuzzy expressions. According to the same principles we achieved the results in the case of clause FUZZYHAVING or a combination of both types of filters fuzzy.

In a situation where the traditional precise search for information is insufficient, the user has the option to take advantage of the system to define inaccurate query. Presented solution allows the user to extract information according to defined expectations and preferences. These preferences are determined at the stage of defining linguistic terms and modifications in the database. Dictionaries of labels are available during the formulation of queries. The client application enables the construction and implementation of inaccurate statements, as well as the presentation of the results of that are arranged according to the degree of fulfilment of the query.

Presented solution allows you to get fuzzy answers for the imprecise query. The fuzzy conditions can be defined in the FUZZYWHERE and FUZZYHAVING. The task is carried out in several stages, which allows

for a detailed analysis of the results of the individual phases of the query execution. Through the write access to the files mechanism, we can see both the shape of membership functions obtained as well as the result of action on individual fuzzy sets. Queries may contain aggregate functions as well as complex conditional statements and fuzzy conjunction operators and alternatives. In addition, conditions may relate to both single and multiple attributes, which is associated with the multidimensional data processing.

VII. CONCLUSION

This paper, presents a novel approach to the problem of imprecise information retrieval from database systems. SQL standard does not provide any mechanism for solving such task. Recently, fuzzy SQL languages have become a very interesting scope of research. Most of current implementations are based on strictly defined membership function and require expert knowledge about threshold degree for specific data type. This article presents an idea of gaining imprecise and incomplete information from database by novel algorithms. Most important points of the carried out research are: natural data set preparation based on real website traffic; fuzzy query processing algorithms located on database server implementation and front-end application implementation.

Main idea of the proposed algorithms is fuzzy clustering mechanisms with automatically generated fuzzy sets. All the processing is done by use of smart clustering combined with validity indexes to reach the best results. The process of generation membership function can be executed on demand, triggered by events or executed by scheduler. That feature gives opportunity to adopt to data distribution changes dynamically. In addition to this, the intelligent labeling mechanism together with own parser assigns labels defined in natural language to generated fuzzy sets. The designed system is able to execute fuzzy conditions and aggregations and can be combined with standard SQL. Currently the integration with SQL is based on Java frontend client application but in future it can be provided as an extension of standard SQL. As a summary it can be said that presented idea of fuzzy sets generator together with query parser and intelligent labeling mechanism enables retrieval of data for query written in meta-natural language (fuzzy query with smart labels).

REFERENCES

[1] Bezdek, J. (1981). Pattern Recognition with Fuzzy Objective Function Algoritms. Plenum Press.

[2] Bosc, P., Pivert, O. (1995, luty). SQLf: A Relational Database Language for Fuzzy Querying. IEEE Transactions on Fuzzy Systems, 3(1), pp. 1-17.

[3] Bosc, P., Tré, G. D., Dujmovic, J. J., HadjAli, A., Pivert, O., Ribeiro, R. (2012). On advances in soft computing applied to databases and information systems. Fuzzy Sets and Systems 196: 1-3.

[4] Buckles, B. P., Petry, F. E. (1982). A fuzzy reprezentation of data fo relational database. Fuzzy Sets and Systems(7), pp. 213-226.

[5] Buckles, B. P., Petry, F. E., Sachar, H. S. (1989). A domain calculus for fuzzy relational databases. Fuzzy Sets and Systems(29), pp. 327-340.

[6] Center for Machine Learning and Intelligent Systems, U. o. (2007). Retrieved from Machine Learning Repository: http://archive.ics.uci.edu/ml/

[7] Chang, S., Ke, J. (1979). Translation of fuzzy queries for relational database systems. IEEE Transactions on Pattern Analysis and Machine Inteligence PAMI-1, pp. 281-294.

[8] Chu, S.-C., Roddick, J. F., Pan, J. S. (2002). An Incremental Multi-Centroid, Multi-Run Sampling Scheme for k-medoids-based Algorithms – Extended Report. Knowledge Discovery and Management Laboratory; Technical Report KDM-02-003.

[9] Dembczyński, K., Przybył, D., Kalinowski, P. (2006). Retrieved from SQLf_j: http://calypso.cs.put.poznan.pl/projects/sqlf_j/pl/index.php?page=intro

[10] Dunn, J. (1973). A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. Journal of Cybernetics, 3, pp. 32-57.

[11] Dziedzic, B., Małysiak, B., Mrozek, D. (2008). Interpreter wyrażeń rozmytych stosowanych w składni języka SQL. BDAS. Ustron.

[12] Galindo, J. (n.d.). Retrieved from A Fuzzy Query Language: http://www.lcc.uma.es/~ppgg/FSQL/

[13] Pelikant, A., Kowalczyk, A. (2007). Implemntation of automatically generated membership functions based on grouping algorithms . The International Conference on "Computer as a Tool". Warsaw.

[14] Pelikant, A., Kowalczyk-Niewiadomy, A. (2009). Fuzzy queries in relational databases. System Modelling and Control.

[15] Pelikant, A., Kowalczyk-Niewiadomy, A. (2011). Algorytm etykietowania analizujący rozmyte zapytania w metajęzyku naturalnym. Bazy Danych Aplikacje i Systemy. Ustron.

[16] Rubio, E., Castillo, O., Melin, P. (2011). A new validation index for fuzzy clustering and its comparisons with other methods. Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference., (pp. 301-306). Anchorage, AK.

[17] Takahashi, Y. (1991). A fuzzy query language for relational databases. IEEE Transactions on Systems, Man and Cybernetics, 21, pp. 1576-1579.

[18] Takahashi, Y. (1993). Fuzzy database query languages and their relational completeness theorem. IEEE Transactions on Knowledge and Data Engineering, 5, pp. 122-125.

[19] [Xie, X. L., Beni, G. (1991, Aug). A validity measure for fuzzy clustering. Pattern Analysis and Machine Intelligence, IEEE Transactions on(13, 8), 841-847.

[20] Xiong, H., Zhan, G., Wu, J., Shi, Z. (2009, September). Distance Measures for Clustering Validation: Generalization and Normalization. Knowledge and Data Engineering, 21(9), 1249-1262.

[21] Yager RR, F. D. (1994). Approximate clustering via the mountain method. IEEE Trans Syst Man Cybern 24, (pp. 1279–1284).

[22] Zadeh, L. A. (1965). Fuzzy sets. Information and Control, 8(3), 338-353.

[23] Zamenkova, M., & Kendel, A. (1985). Implementing imprecision in information systems. Information Sciences (37(1-3)), pp. 107-141.