UDC 004.4’2, DOI: 10.2298/csis0902127H

An Approach to Fuzzy Database Querying, Analysis and Realisation

Miroslav Hudec

INFOSTAT - Institute of Informatics and Statistics, Bratislava, Slovakia
hudec@infostat.sk

Abstract. Although the Structured Query Language (SQL) is a very powerful tool, it is unable to satisfy needs for data selection based on linguistic expressions and degrees of truth. The goal of the research whose results are presented in the paper is to capture these expressions and make them suitable for queries. For this purpose the fuzzy generalised logical condition for the WHERE part of SQL was developed. In this way, queries based on linguistic expressions are supported and are accessing relational databases in the same way as with the SQL. Fuzzy query is not only a querying tool; it improves the meaning of a query and extracts additional valuable information. Statistical data about districts of the Slovak Republic are used in the case study. Fuzzy approach has some limitations that would appear in a querying process. These limitations and ideas how to solve them are outlined in this paper.

Keywords: SQL, fuzzy queries, fuzzy generalised logical condition.

1. Introduction

This paper examines situations when database querying process by the two-valued realisation of Boolean algebra is not adequate and offers solution based on the fuzzy logic because the fuzzy logic is an approach to computing based on "degrees of truth" rather than the usual "true or false" logic. The fuzzy logic deals with reasoning that is approximate rather than precise to solve problems in a way that more resembles human logic.

This area of research is not new one but there are still many possibilities for the improvement of existing approaches and for creating new approaches. Fuzzy queries have emerged in the last 25 years to deal with the necessity to soften the two-valued Boolean logic in relational databases. A fuzzy query system is an interface to users to get information from database using (quasi) natural language sentences. Many fuzzy query implementations have been proposed, resulting in slightly different languages. Although there are some variations according to the particularities of different implementations, the answer to a fuzzy query sentence is generally a list of records, ranked by the degree of matching [2].

Organisations work with very large data collections mainly stored in relational databases. Linguistic expressions are interesting for data extraction, analysis, dissemination and decision making. In case of statistical data they are often collected with some errors and vagueness. If two-valued logic in selection process is used then the small error in data values or cases when user can not unambiguously define the criterion by crisps boundaries may involve some inadequate selected or non-selected data. To avoid this, the wanted scenario for the user is to determine what kind of data he wants to select by linguistic expressions and degrees of truth. These linguistic expressions have the logical meaning for user and describe a data selection process in the natural language. The conversion from human into computer language and all mathematical operations are implemented behind the user interface and user obtains the final solution of a task.

Issues and perspectives of fuzzy querying can be found in [8]. The fuzzy query approach based on the fuzzy Generalised Logical Condition (GLC) is presented and analysed in this paper. The GLC formula capable to capture linguistic expressions into the WHERE part of the SQL was initially created and presented in [5]. The implementation of the GLC for statistical databases is explained in [6]. Although fuzzy approach has significant advantages beyond the two-valued approach, there are situations when some inconsistency may occur. It is possible to avoid this problem by several ways [11], [12].

This paper is organised as follows: Classical SQL and its limitation are presented in section 2. Section 3 explains the fuzzy query approach and the GLC. This querying process is described by the case study. Section 4 analyses limitations of using fuzzy approach in querying process and suggests some ideas how to avoid them. Section 5 discusses some aspects of practical use of this solution and points out some additional usage of the GLC. Finally some conclusions and some suggestions for future research topics are drawn.

2. SQL and its limitation

Users search databases in order to obtain data needed for analysis, decision making or to satisfy their curiosity. The SQL is a standard query language for relational databases. The simply SQL query is as follows:

```
select attribute_1,…,attribute_n
from T                                          (0)
where attribute_p > P and attribute_r < R
```

The result of the query is shown in graphical mode in figure 1. Values P and R delimit the space of interesting data. Small squares in the graph show database records. In the graph it is obviously shown that three records are very close to meet the query criterion. These records could be potential customers and direct marketing could attract them or municipalities which almost meet the criterion for some financial support for example.

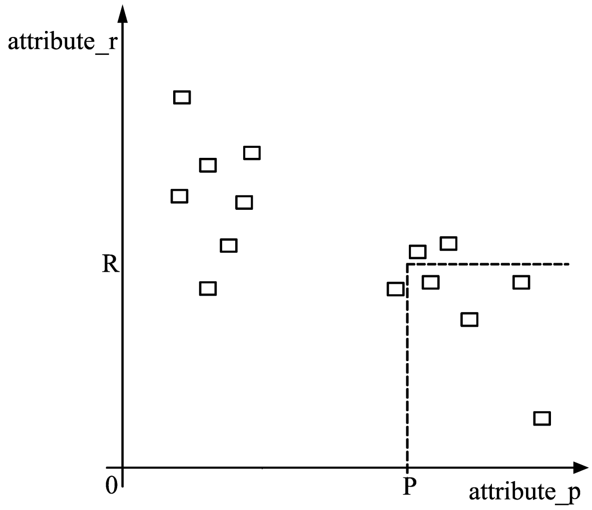

Fig. 1. The result of the classical query

The SQL uses the crisp logic in querying process that causes crisp selection. It means that the record would have not been selected even if it is extremely close to the intent of the query criterion. As the criterion becomes more and more complex, the set of records selected by the WHERE statement becomes more and more crisp.

If the classical SQL is used for solving this problem, the SQL relaxation would have to be used in the following way:

```
select attribute_1,…,attribute_n
from T                                                    (0)
where ateribute_p > P-p and attribute_r < R+r
```

where p and r are used to expand the initial query criteria to select records that almost meet the query criteria. This approach has two disadvantages [3]. First, the meaning of the initial query is diluted in order to capture adjacent records. The meaning of a query: “where attribute_p is more than P” is changed and adjacent records satisfy a query in the same way as initial ones. More precisely, the difference between original and adjacent data (caught records along the “edge” of interesting space) does not exist. Secondly problem rises from the question: what about records that are very close to satisfy the new expanded query and it is useful to make another expanding of a query. In this way more data from the database is selected, but the user has lost the accuracy of his query.

Many applications have created uncountable accesses to wide variety of data. The data and the classical access to data are simply not enough in many cases. In cases when the user can not unambiguously separate interesting data from not interesting by sharp boundaries or when the user wants to obtain data that is very close to meet the query criterion and to know the index of distance to full query satisfaction, it is necessary to adapt the SQL to these requirements.

The SQL was initially developed in [7]. Since then the SQL has been used in many relational databases and information systems for data selection. The use of SQL may be regarded as one of the mayor reasons for the success of relational databases in the commercial world [13]. In this research the core of SQL remains intact and the extension is done to improve the selection process. Adding some flexibility to the SQL meets above mentioned requirements and increases effectiveness and comprehensibility of the whole querying process.

3. Queries based on fuzzy logic

The GLC for the WHERE part of the SQL based on linguistic expressions has been created in [5]. For the further reading it is important to define the Query Compatibility Index (QCI). The QCI is used to indicate how the selected record satisfies a query criterion. The QCI has values from the [0, 1] interval with the following meaning: 0 - record does not satisfy a query, 1 - record fully satisfies the query, interval (0, 1) - record partially satisfies a query with the distance to the full query satisfaction.

The GLC has the following structure:

$$\text{where} \bigotimes_{i=1}^{n} (a_i \circ L_{xi}) \ . \qquad (\mathbf{0})$$

where n denotes the number of attributes with fuzzy constraints in a WHERE clause of a query,

$$\otimes = \begin{cases} and \\ or \end{cases} ,$$

where *and* and *or* are fuzzy logical operators,
and

$$a_i \circ L_{xi} = \begin{cases} a_i > L_{di}, & \text{a}_\text{i} \text{ is Big} \\ a_i < L_{gi}, & \text{a}_\text{i} \text{ is Small} \\ a_i > L_{di} \text{ and } a_i < L_{gi}, a_i \text{ is About} \end{cases} ,$$

where a_i is a database attribute, L_{di} is the lower bound and L_{gi} is upper bound of a linguistic expression described by fuzzy sets shown in figure 2.

Conditions in queries contain these comparison operators: >, <, =, ≠ and *between* when numerical attributes are used. These crisp logical comparison operators are adapted for fuzzy queries in the following way: operator >

(greater than) was improved with fuzzy set „Big value“ (figure 2a), operator < (less than) was improved with fuzzy set „Small value“ (figure 2b) and operator = (equal) was improved with fuzzy set „About value“ (figure 2c). Operator ≠ is the negation of the operator = so this operator is not further analysed. Analogous statement is valid for the operator *between* because it is similar to the operator = from the fuzzy point of view. Other types of fuzzy sets could be added in the future to catch other linguistic expressions.

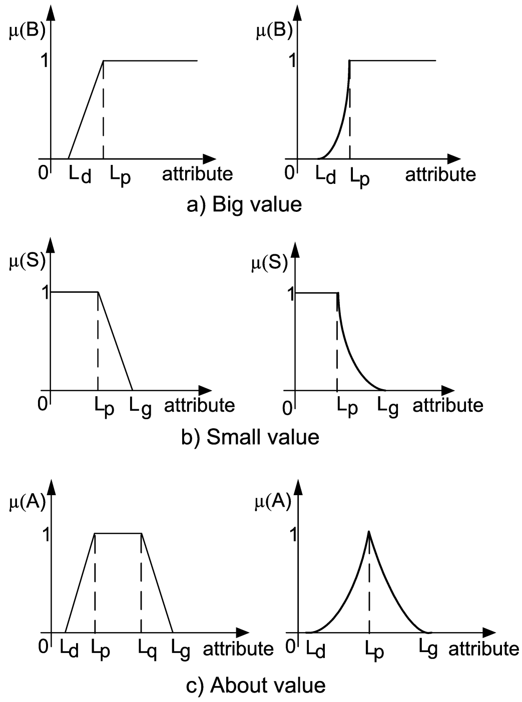

Fig. 2. Fuzzy sets

In the figure 2 two often used fuzzy sets are shown and implemented in the fuzzy query approach. It is possible to add more types of fuzzy sets for each of linguistic expressions. A useful theoretical overview about fuzzy sets and fuzzy operators can be found in [9] and a useful practical overview can be found in [3].

If a query in the WHERE clause contains fuzzy as well as classical constraints, these classical constraints could be easy added to the WHERE clause (3) in the following way:

where $\bigotimes_{i=1}^{n} (a_i \circ L_{xi})$ [and/or][attribute_m LIKE “*String”] [and/or] ... (**0**)

where LIKE is the SQL comparison operator and String is an arbitrary string variable.

The querying process consists of the two steps. In the first step lower and/or upper bounds of linguistic expressions (fuzzy sets) are used as parameters for database queries. It means that all records that have QCI greater than zero are selected only. In the second step the chosen analytical form of the fuzzy set is used to calculate the membership degree of each selected record to appropriate fuzzy set. Finally, appropriate t-norms or t-conorms are used to calculate QCI values for all retrieved records.

In case when two fuzzy constraints are in a WHERE clause all t-norms or t-conorms can be used as an aggregation function. However, for fuzzy SQL it is required to use more than two attributes in the WHERE clause of a query. From the associative rule of t-norms and t-conorms, the following functions can be easy aggregated for cases when more than two attributes are used [12]:

t-norms for *and* operator:

- min:

$$\mathrm{QCI} = \min(\mu_i(\mathrm{a_i})), \quad \mathrm{i=1,..., n} . \qquad (\mathbf{0})$$

- product:

$$\mathrm{QCI} = \prod_{i=1}^{n} (\mu_i(\mathrm{a_i})), \qquad (\mathbf{0})$$

- bounded difference (BD):

$$\mathrm{QCI} = \max(0, \sum_{i=1}^{n} \mu_i(a_i) - n + 1) . \qquad (\mathbf{0})$$

t-conorms for *or* operator:

- max:

$$\mathrm{QCI} = \max(\mu_i(\mathrm{a_i})), \quad \mathrm{i=1,..., n} . \qquad (\mathbf{0})$$

- bounded sum (BS):

$$\mathrm{QCI} = \min(1, \sum_{i=1}^{n} \mu_i(a_i)) . \qquad (\mathbf{0})$$

where $\mu_i(a_i)$ denotes the membership degree of the attribute a_i to the i-th fuzzy set.

These functions are implemented into the query language and cover usual expected cases. Min t-norm (5) takes into account the lowest value of membership degrees to fuzzy sets only; product t-norm (6) takes into account all membership degrees and balances the query truth membership value across each of the membership degrees. In case when the correlation is unknown or does not exist, these two norms are used. In other cases, the BD t-norm (7) is used. From above mentioned t-norms, only the BD t-norm satisfies the non-contradiction and excluded middle laws. The similar discussion is valid for t-conorms (8) and (9).

To implement this a fuzzy query interpreter, which transforms fuzzy queries to the classical SQL structure, was developed. In this way, queries based on linguistic expressions on client side are supported and are accessing relational databases in the same way as with the classical SQL. Figure 3 shows this model. The first step of querying process (to select records that have QCI>0) is situated in parts 1, 2 and 3. Lower and/or upper limits of linguistic expressions are calculated and converted into SQL query in the part 1. Thus created SQL query selects data from database (part 2) and saves it into the temporary table (part 3). The second step uses data from part 3 for further calculations. Firstly, the chosen analytical form of the fuzzy set (from part 1) is used to determine the membership degree of each selected record to appropriate fuzzy set. Secondly, t-norm and/or t-conorm function also defined in the part 1 are used to calculate the QCI value. These calculations are situated in the part 4. Finally the result of fuzzy query is displayed in part 5.

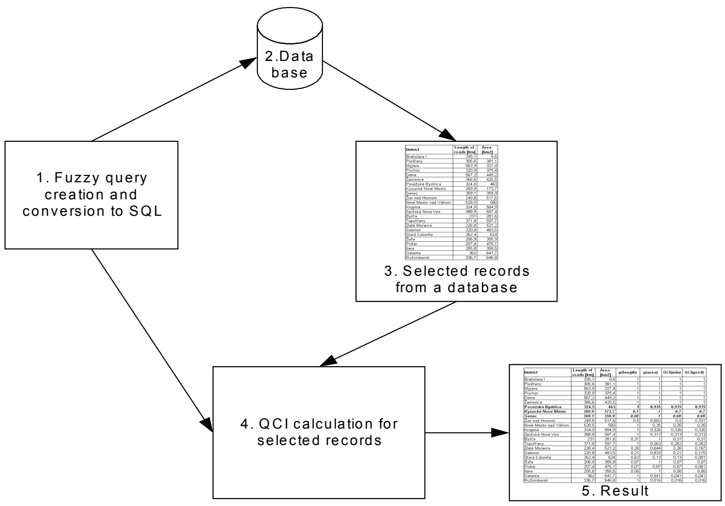

Fig. 3. Structure of the fuzzy SQL

3.1. Case study

This system is tested on data from the Urban and Municipality Statistics database used in the Statistical Office of the Slovak Republic [1]. In this case study, districts with high length of road and small area size are sought. The big road infrastructure density is analysed as an illustrative example. The query has the following form (The simplification is done for better comprehensibility. The names of attributes are simplified and the names of tables and join conditions in the WHERE and FROM part of the query are omitted.):

```
select district, roads, area
from T
where roads is Big and area is Small
```

The length of road indicator is represented by „Big value“ fuzzy set with these parameters L_d=200 km and L_p =300 km and the shape as from figure 2a). The „Small value“ fuzzy set with parameters L_p=450 km^2 and L_g =650 km^2 and the shape as from figure 2b) describes the area attribute.

According to these parameters the query has the following form:

```
select district, roads, area
from T
where roads > 200 and area < 650
```

The result of the fuzzy query is shown in Table 1. The value of min t-norm (5) is used for district ranking. The Table 1 shows six districts fully satisfying the query; one district is extremely close to satisfying the query and another two districts are close to the query criterion. It means for example that even small changes in districts' attributes could imply that another records fully satisfies the query. If SQL was used, this additional information would remain hidden.

In cases when the user uses SQL and wants to obtain similar results like result presented in Table 1 it is needed to make small changes in criterion parameters and to execute larger number of queries. The WHERE clause from the query (1) could be modified as follows:

- **(attribute_p > P and attribute_r < R)** extracts records that satisfy initial conditions.
- **(attribute_p > P-p_1 and attribute_p < P) and (attribute_r < R+r_1 and attribute_r > R)** selects records that meet query criterion with value of e.g. 0.9 or almost meet the query criterion (where p_1 and r_1 are small real values greater than zero),
- **(attribute_p > P-p_2 and attribute_p < P-p_1) and (attribute_r < R+r_2 and attribute_r > R+r_1)** selects records that meet query criterion with value of e.g. 0.8 or records that are very close to query criterion (where p_1>p_2 and r_2>r_1),
- etc.

The following conclusion appears: for the very soft gradation, the infinite number of SQL queries has to be used. In case of fuzzy queries, one query is sufficient.

Table 1. Result of the fuzzy query

District	Length of roads [km]	Area [km^2]	μ (length)	μ (area)	QCI (min)	QCI (prod)
Bratislava I	335,1	9,6	1	1	1	1
Piešťany	305,6	381,1	1	1	1	1
Myjava	563,9	327,4	1	1	1	1
Púchov	320,9	375,4	1	1	1	1
Detva	567,2	449,2	1	1	1	1
Žarnovica	366,6	425,5	1	1	1	1
Považská Bystrica	324,5	463	1	0,935	0,935	0,935
Kysucké N. M.	269,9	173,7	0,7	1	0,7	0,7
Senec	269,1	359,9	0,69	1	0,69	0,69
Žiar nad Hronom	249,8	517,6	0,5	0,662	0,5	0,331
N. Mesto n/Váhom	528,5	580	1	0,35	0,35	0,35
Krupina	334,9	584,9	1	0,326	0,326	0,326
Spišská Nová Ves	388,9	587,4	1	0,313	0,313	0,313
Bytča	231	281,6	0,31	1	0,31	0,31
Topoľčany	371,8	597,7	1	0,262	0,262	0,262
Zlaté Moravce	226,4	521,2	0,26	0,644	0,26	0,167
Sabinov	220,8	483,5	0,21	0,833	0,21	0,175
Stará Ľubovňa	262,4	624	0,62	0,13	0,13	0,081
Šaľa	206,9	355,9	0,07	1	0,07	0,07
Poltár	207,4	476,1	0,07	0,87	0,07	0,061
Ilava	205,8	358,5	0,06	1	0,06	0,06
Galanta	362	641,7	1	0,041	0,041	0,041
Ružomberok	336,7	646,8	1	0,016	0,016	0,016

4. The GLC and the Boolean algebra

The GLC is developed on axioms of fuzzy logic. Fuzzy logic is based on same principle as classical logic, the principle of truth-functionality. Logic is truth functional if the truth value of a compound sentence depends only on the truth values of the constituent atomic sentences [14], not on their meaning or structure [11]. In the two-valued logic the mentioned principle is enough for all axioms. In case of all many-valued logics, including fuzzy logic, this principle is not sufficient and as a consequence these logics are not in the Boolean frame. More precisely, fuzzy logic is a precise many-valued logic where axioms of non-contradiction and excluded middle are not satisfied. It is obvious on following example: “WHERE attribute_p >5 and attribute_p <=5” (contradiction). In classical query it is obvious that criterion retrieves no record from database. In case of fuzzy query when the “WHERE attribute_p is Big and attribute_p is not Big” criterion is used it could be expected that no record is retrieved because of non-contradiction axiom existence but min t-norm (5) retrieves some records with QCI≤0.5. This is the consequence of not satisfied non-contradiction axiom.

First way how to use the gradation in mathematics is to leave these axioms as non adequate and accept the principle of truth functionality with all consequences. When the first way is chosen, it is possible to avoid this problem by selecting adequate t-norm or t-conorm function for each query. A new t-norm T^ and a new t-conorm C^, which depend on a parameter r in [-1, 1] or the correlation between the truth values of the operands are explained in [12].

The second way is to go to the source of Boolean algebra and find the principle for gradation to be in the frame of the Boolean algebra. New approach to treating fuzziness or gradation in logic is based on the Interpolative Realisation of Boolean Algebra (IBA) [11]. The IBA ensures that the whole selection process will be in the frame of Boolean algebra and avoids theoretically possible situations when inappropriate functions are chosen.

5. Discussion about proposed model

The comparison between SQL and fuzzy query performance can not be unambiguously determined because of different nature of these two querying concepts. SQL has faster performance because of non existence of additional calculation of lower bounds of fuzzy sets, membership degrees and QCIs for selected records as in the fuzzy counterpart. On the other side fuzzy query provides more information than classical one and gives the user more freedom for creating of a selection task. In cases when user does not have ambiguities and uncertainties concerning data, the SQL solves all user needs and requirement for fuzzy queries does not exist.

The WHERE clause of a query usually contains comparison operators with crisp values such as:

```
where unemployment_rate > 5 and population_density >
200 and migration_level < 1.
```

The meaning and purpose of the previous query is understandable for the user who creates and uses it. In case of reuse of the same query in e.g. different time period or by different user the meaning and purpose of query is not obviously clear at the first glance. Fuzzy query contains logical conditions defined by linguistic expressions whereby the query becomes easy understandable and applicable. The meaning of a query remains the same, only the parameters and shapes of fuzzy sets are changeable to allow the query adaptation to new situations or requirements.

A disseminated number without meaning (without explanatory metadata) does not tell very much to user. A number in a WHERE clause does not explain the purpose of a query in many situations. As metadata are used to explain the meaning of the figures, the linguistic expressions are used to explain the meaning of a query.

Fuzzy query enables also simplified and easy to use distance measurement of records around selected value. For this purposes normalised (fuzzy set is normalised if $\exists x, \mu_A(x) = 1$) and symmetric fuzzy sets shown in figure 4 are used. The criterion "WHERE attribute is approximately 5" can be described with "about" fuzzy set. The query retrieves all records that have attribute value equal to 5 and all records that have QCI>0 and QCI value is a distance value of each selected record.

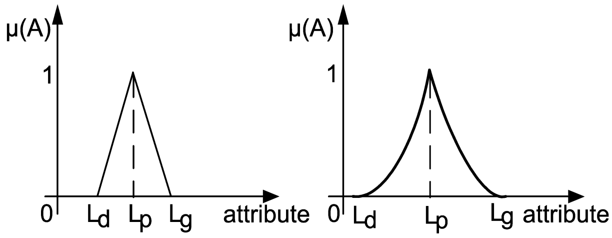

Fig. 4. Approximately or about value fuzzy sets

Case-based Reasoning (CBR) is very interesting field for the research and applications. Using relational databases for storing and representing cases in CBR reduces effort needed to develop a CBR system. The approximate reasoning is not possible with the SQL. A proposal where the case retrieval is implemented by using a fuzzy version of SQL is presented in [10].

Fuzzy query implementation FSQL in [4] deals with fuzzy databases. The FSQL's syntax permits to use fuzzy comparisons with fuzzy and non-fuzzy database attributes. The FSQL supports possibility and necessity comparators and supports fuzzy hedges, thresholds, distributions, quantifiers in WHERE and HAVING clause of a query. This is a complex solution for including fuzzy concepts into database queries.

However, many statistical databases are developed in Relational Database Management Systems (RDBMS) and this trend dominates in development of new statistical information systems and databases. The fuzzy SQL for traditional relational databases has the significant perspective for usage in many areas including statistics. The approach presented in this paper is a way how to create an easy to use fuzzy querying tool for end users. Further improvement of the proposed fuzzy query approach could be in avoiding the inconsistency of fuzzy logic caused by the principle of truth-functionality.

The goal of fuzzy querying is not to obtain more data but better data for users. Data selection depends on degree of satisfaction that implies that a fuzzy query retrieves some data in cases when traditional SQL retrieves no data. Additional information for users could be as follows: although no records fully satisfy the intent of a query, there are some records that are very close to meet a query criterion and it may be interesting to analyse this data.

6. Conclusion

When users work with usual software tools they have to change their many-valued logical thinking (approximate reasoning) into the two-valued computer logic. The SQL requires the crisp specification of a query criterion, while for users a query is best described in terms of a natural (or quasi) natural language with ambiguities and uncertainties. The gradation is immanent for many-valued logic and one sentence based on many-valued logic is as powerful as infinite number of sentences based on two-valued logic. With this approach the user is given a powerful and easy to use data mining tool which allows him to query data from databases by using linguistic expressions in order to improve the quality of selection process.

This system is under programming for a statistical information system. Expressions like: high rate of unemployment or high migration level etc, are very often used in statistics. The goal of this research is to capture these expressions and make them suitable for database queries. In many cases (not only for statistic data) the user (analyst, decision maker,…) can not unambiguously separate data he is interested in from data he is not interested in by sharp boundaries or user can not expressly prove, why the chosen boundary value is the best one. Users also want to obtain data that are very close to satisfy queries and to know the index of distance to full query satisfaction.

The SQL is used in all major RDBMS and also in all major information systems. The SQL is optimised to work with RDBMS. A fuzzy query interpreter was developed to transform fuzzy queries to the classical ones. In this way, queries based on linguistic expressions on client side are supported and are accessing relational databases in the same way as with the classical SQL. No modification of databases has to be undertaken. Another advantage is that the user does not need to learn a new query language.

The fuzzy SQL is in this approach an independent module and it can be used when the user wants to use a linguistic expression in queries. The research done in this work would be continued in following directions:

- Implementation of the IBA in querying process is very interesting topic for further research. The IBA in data selection will cause that the whole querying process would be in the frame of the Boolean algebra.
- The web application with a fuzzy module for data dissemination is another way of improvement of this fuzzy query approach. For example, statistical institutions put vast amount of data onto their websites. Providing a selection criterion by linguistic expression gives natural way for data selection and sites with big amount of data and metadata would become more user friendly in processes of data selection.

7. References

1. Benčič, A., Hudec, M.: MOŠ/MIS–Urban and municipal statistics project and information system of the Slovak Republic .In Proceedings of the SYM-OP-IS. Vuletić Print, Tara, Serbia, XXI-32--XXI-35. (2002)
2. Branco, A., Evsukoff, A., Ebecken, N.: Generating Fuzzy Queries from Weighted Fuzzy Classifier Rules. In Proceedings of the ICDM workshop on Computational Intelligence in Data Mining. IOS Press, Huston, USA, 21-28. (2005)
3. Cox, E.: Fuzzy modeling and genetic algorithms for data mining and exploration. Morgan Kaufman, San Francisco, USA. (2005)
4. Galindo, J., Urrutia, A., Piattini, M., Fuzzy Databases: Modeling, Design and Implementation, Idea Group Publishing, Hershey, USA. (2006)
5. Hudec, M.: Fuzzy improvement of the SQL. In Proceedings of the Balkan Conference on Operational Research (Balcor). Beograd, Serbia, 255-265 (2007).
6. Hudec, M.: Fuzzy SQL for statistical databases. In MSIS, Meeting on the Management of Statistical Information Systems. Luxembourg. (2008) [Online]. Available: http://www.unece.org/stats/documents/ece/ces/ge.50/2008/wp.12.e.pdf (January 2009)
7. Chamberlin, D., Boyce R.: SEQUEL: A Structured English Query Language. In Proceedings of the ACM SIGMOD Workshop on Data Description, Access and Control. ACM Press, Ann Arbor, USA, 249-264. (1974)
8. Kacprzyk, J., Pasi, G., Vojtáš, P., Zadrozny, S.: Fuzzy querying: Issues and perspectives. Kybernetika, Vol. 36, No. 6, 605-616. (2000)
9. Klir, G., Yuan, B.: Fuzzy sets and fuzzy logic, theory and applications. Prentice Hall, New Jersey, USA. (1995)
10. Portinale, L., Verrua, A.: Exploiting Fuzzy-SQL in Case-Based Reasoning In: Fourteenth International Florida Artificial Intelligence Research Society Conference (FLAIRS). AAAI Press, Key West, USA, 103-107. (2001)
11. Radojević, D.: Interpolative Realization of Boolean Algebra as a Consistent Frame for Gradation and/or Fuzziness. In: Nikravesh, M., et al. (eds.): Forging New Frontiers: Fuzzy Pioneers II Studies in Fuzziness and Soft Computing, Vol. 218. Springer-Verlag Berlin and Heidelberg , Germany, 295-318. (2008)
12. Siler, W., Buckley, J.: Fuzzy expert systems and fuzzy reasoning. John Wiley & Sons, Inc., New Jersey, USA. (2005)
13. Urrutia, A., Pavesi, L.: Extending the capabilities of database queries using fuzzy logic. In Proceedings of the Collecter-LatAm, Santiago, Chile (2004). [Online]. Available: http://www.collecter.org/archives/2004_October/06.pdf (current January 2009)
14. Zadeh, L.: Fuzzy Sets. Information and Control, No. 8, 338-353. (1965)

Miroslav Hudec received the Dipl.Ing. degree in information systems in 1996, and the M. Sc. degree in operations research in 2005, booth from the University of Belgrade, Serbia. He is currently preparing Ph.D dissertation in operations research at the same university. Since 2000 he has been employed in Institute of Informatics and Statistics in Bratislava, Slovakia. In this institute he works as a designer of information systems and researcher. His current research area is the fuzzy logic and its applicabilities. Since 2005 he is participant on the UNECE/Eurostat/OECD Meeting on the Management of Statistical Information Systems where he regularly presents his contributions.

Received: March 02, 2009; Accepted: October 01, 2009.