Overview of Data Fuzzificator Concept

***Abstract**: In this overview, we present Data Fuzzificator concept using an example from care dental services. We specially take an example from care dental services since it is known for most of us who visit from time-to-time dental practices. The same Data Fuzzificator concept can be applied for any other similar examples from banking, management, insurance, data mining, etc.*

INTRODUCTION

Generally, dental practices aim to providing the highest quality services to their patients. To achieve this, it is important that dentists are able to collect opinions of patients about their experiences at the dental practice and are able to evaluate patients' feedback most precisely. We propose to use a Data Fuzzificator concept to combine various assessment criteria into one general measure to assess patients' satisfaction with care dental services. The proposed framework can be used, for example, in information systems of dental practices and easily integrated with conventional information systems already used in dental practices. The benefits of using the proposed Data Fuzzificator approach include more flexible and accurate analysis of patients' feedback. To confirm the theory, a prototype was developed based on the Microsoft SQL Server database management system for two criteria used in dental practices, namely, making an appointment with a dentist and waiting time for care dental services.

Attempts to classify patients' feedback for provided care dental services in dental practices are often not easy to address by standard mathematical and statistical approaches, like moving average, straight-line interpolation and many others. First, this is because there are many kinds of assessment criteria that can be used, and these criteria may not provide a consistent picture of patient satisfaction with particular dentists. Second, a number of patient feedback criteria are qualitative and, hence, are difficult to incorporate into standard classification schemes designed to produce one general measure to evaluate patients' satisfaction with particular care dental services. Fuzzy set theory (Zadeh, 1989) offers a way to address such problems. Fuzzy sets provide mathematical meanings to the natural language statements and become an effective solution for dealing with uncertainty. An important feature of fuzzy sets is that they provide formalism for incorporating ambiguity and lack of quantitative data in a typical classification scheme.

In most practical data classification situations, more than one criterion (attribute) must be considered simultaneously. Criteria are usually measured with different scales and should be combined into one general measure. The approach of fuzzy set theory with its membership functions is a unique approach widely used to form a realistic general measure of the assessment incorporating ambiguity and lack of quantitative data in various criteria (Chen, 1998; Han and Kamber, 2001; Zadeh, 1989). In fuzzy set theory, various criteria can be relatively easily combined into a joint response measure called the aggregated value of the membership (Zimmermann, 1992), which can be used as the general measure.

A number of various schemes and tools for the implementation of fuzzy sets in database management systems have been proposed in recent years, such as fuzzy querying (Bellma and Vojdani, 2000), fuzzy extension of SQL (Structured Query Language) (Bosc and Pivert, 2000; Kacprzyk and Zadrozny, 2000) and fuzzy object oriented database schemes (Borgodna et al., 2000; Dubois, Nakata and Prade, 2000). There are also commercial tools available on the market, like FIDE (Fuzzy Inference Development Environment for the development of fuzzy logic-based systems) of Aptronix Inc., FLINT (Fuzzy Logic Interfacing Toolkit that makes fuzzy logic technology and fuzzy rules available within a sophisticated programming environment) of Logic Programming Associates Co. and FQS (Fuzzy Query System that allows the user to conduct database queries using the semantic flexibility of Fuzzy SQL) of Sonalysts Inc.

The majority of the above-mentioned fuzzy methods and systems for data mining as well as commercial tools require to change the conventional relational database structure or add special features to the database management tools, for example, the modification of the conventional SQL (Structured Query Language) functionality (Bosc and Pivert, 2000; Kacprzyk and Zadrozny, 2000) or the extraction of fuzzy functionality into a separate application as it was implemented in FQS of Sonalysts Inc.

Fuzzy methods are not widely used in relational database systems in practice and the main reason is that most of database system users do not want to switch to fuzzy database structures or use third-party applications. As a result, they usually do not use fuzzy querying in their information systems. To attract more attention to fuzzy applications from database users we propose to use a framework based on the fuzzy classification and conventional SQL queries (Data Fuzzificator concept). The main benefit of our approach is that there is no need to modify the functionality of conventional relational databases and SQL. All manipulations can be done as an extension of the database schema by applying fuzzy data classification. Benefits of using fuzzy sets and fuzzy classification in data mining, like user-friendly data presentation, precision of the data classification, use of linguistic variables instead of numeric values and easy-to-use facilities for querying the extended database schema become available for users of relational databases as well.

CONVENTIONAL DATA CLASSIFICATION IN RELATIONAL DATABASES

To show the difference in the data presentation between the conventional data classification and fuzzy data classification using Data Fuzzificator concept, the following simple example from dentistry practices can be considered. An exemplary data set (this is a selected data set from continuous tracking of care dental service quality based on patient's feedback) includes OFFERS table that contains dentist surname (*Dentist* column), provided care dental service (*Service* column), waiting time of patients until they are invited to start their treatment in the dentist room (*Waiting Time* column) and the number of days that patients had to wait for their first suitable appointment proposed by a dentist (*Appointment* column). An exemplary data relation is presented in Table 1.

According to conventional data classification, we will have to define classes for our exemplary OFFERS table. To define classes, we introduce atomic values for *Appointment* and *Waiting Time* attributes. We define "acceptable" and "unacceptable" atomic values for *Appointment* attribute. The atomic value of "acceptable" is assigned to the interval of (1-5) and the atomic value of "unacceptable" is assigned to the interval of (6 - 10). Similarly, we define "short" and "long" atomic values for *Waiting Time* attribute. The atomic value of "short" is assigned to "short" and "average" *Waiting Time* attribute values. "Long" atomic value is assigned to "above average" and "long" *Waiting Time* attribute values. As soon as the definition of atomic values is finished, it is easy to define all classes for dentists:

C1 – "*Award a bonus*" class with atomic values ("acceptable", "short");
C2 – "*Complain about appointment*" class with atomic values ("unacceptable", "short");
C3 – "*Complain about waiting time*" class with atomic values ("acceptable", "long");
C4 – "*Interfere for improvements*" class with atomic values ("unacceptable", "long").

Table 1

OFFERS table

Dentist	Service	Waiting time [subjective patients' feeling]	Appointment [number of days]
Berg	Consultation	short	7
Berg	Tooth treatment	above average	6
Berg	Tooth treatment	short	2
Crow	Consultation	short	2
Gate	Consultation	average	4
Gate	Tooth removal	short	3
Gate	Tooth removal	above average	8
Gate	Tooth treatment	short	9
Host	Consultation	above average	2
Host	Tooth treatment	above average	4
Merk	Consultation	average	9
Merk	Consultation	long	7
Merk	Tooth removal	above average	8
Roy	Consultation	long	4
Roy	Tooth removal	long	2
Thon	Tooth treatment	long	2
Thon	Tooth treatment	average	2

A distribution of dentists from OFFERS table among the introduced classes is graphically presented in Figure 1. The most common method of querying relational database tables is through SQL (Structured Query Language). To find all members of "*Complain about appointment*" class **C2** (see Figure 1), one can query initial OFFERS table with simple SQL query:

```
SELECT Dentist FROM Offers WHERE ([Waiting Time] = "short" or [Waiting Time] = "average") and
        (Appointment > 5).
```

Using above-presented and other similar SQL queries one can classify all dentists. For example, dentist Gate belongs two times to class **C1**, one time to class **C2** and one time to class **C4** (see Figure 1). Hence, one can say that he most likely belongs to class **C1** (2 occurences for class **C1** (50 percent) and one occurrence for each of classes **C2** and **C4** (25 percent for each)). Dentist Thon belongs one time to class **C1** and one time to class **C3**.

A (Appointment)

	10	**C2**			**C4**	
	9	Gate	Merk			
unacceptable	8			Merk, Gate		
	7	Berg			Merk	
	6			Berg		
	5	**C1**			**C3**	
	4		Gate	Host	Roy	
acceptable	3	Gate				
	2	Crow, Berg	Thon	Host	Thon, Roy	
	1					
		short	average	above average	long	**W (Waiting time)**
		short		**long**		

Figure 1. Conventional non-fuzzy classification of dentists

This means that he belongs with 50 percent to class **C1** and 50 percent to class **C3**. One can continue in the same way with other dentists. Let us state the following important questions and see that it is not easy to find the answers on them using conventional data classification:

- With the introduction of atomic values the attributes become less accurate since different values are aggregated into atomic values (in our example, “acceptable” and “unacceptable”, “short” and “long” atomic values are used). How to prevent deterioration of data accuracy with the introduction of atomic values?
- Who among the dentists belong to which class more (the above-presented calculation scheme based on occurences is not accurate enough since with the introduction of atomic values for classes the attributes become less accurate)?
- How to get answers on first two questions using conventional easy-to-use SQL queries?

The main problem of the presented conventional data classification is that it has a discrete boundary definition of atomic values (see Figure 1). Thus, the classification reports based on such conventional data classification are usually not accurate enough. To improve a precision of data classification, we will use fuzzy data classification methodology (Data Fuzzificator concept).

FUZZY DATA CLASSIFICATION IN RELATIONAL DATABASES

To get more detailed information about dentist membership in classes, one can introduce fuzzy classification of data (Schindler, 1998; Zimmermann, 1992) and linguistic variables (Schindler, 1998). As an example, we can consider *Appointment* attribute as a linguistic variable. Area of definition of the linguistic variable is domain **A**(Appointment). Similarly to the assignment of the atomic value, *Appointment* linguistic variable possesses a set of terms **A**(Appointment) = {“acceptable”, “unacceptable”} with the verbal terms “acceptable” and “unacceptable” which define appropriate equivalence classes (1 - 5) and (6 - 10).

The most important feature of linguistic variables is that every term of a linguistic variable represents a fuzzy set. The membership function of the fuzzy set is defined over the domain of the corresponding attribute. For instance, as it is shown in Figure 2 for our example, the appointment with dentist within the next five days is at the same time acceptable and unacceptable; in both cases the membership function μ has value of 0.5 (i.e. $\mu_{acceptable} = 0.5$ and $\mu_{unacceptable} = 0.5$). The membership of an object in a specific class can be calculated by an aggregation over all terms of the linguistic variables that define the class. Terms "acceptable" and "short", for instance, describe class **C1**. The membership is a conjunction of the corresponding values of the membership functions $\mu_{acceptable}$ and μ_{short}.

There exist a number of operators that can be used to calculate conjunctions of membership function values (Zimmermann, 1992). For example, one can apply the γ-operator which is used as a “compensatory AND” and was empirically tested in (Zimmermann, 1992). The membership $\mu_{\tilde{A}_i,\mathrm{comp}}(x)$ of *x* object with *m* linguistic variables to the given classes can be calculated based on the following equation (Zimmermann, 1992):

$$\mu_{\tilde{A}_i,comp}(x) = \left(\prod_{i=1}^{m} \mu_i(x)\right)^{(1-\gamma)} \left(1 - \prod_{i=1}^{m}(1-\mu_i(x))\right)^{\gamma}, \quad x \in X, \quad 0 \le \gamma \le 1$$

where γ is control parameter with default value of 0.5 (Zimmermann, 1992), $\mu_i(x)$ is the membership value of *x* object to a particular linguistic variable, *m* is the number of linguistic variables. Based on γ-operator, values of dentist membership are calculated and presented in Figure 2 for our example.

The main benefit of fuzzy data classification becomes clear when one compares the presentation of classes **C1**, **C2**, **C2** and **C4** in Figures 1 and 2. As it is shown in Figure 2 with fuzzy data classification, one can not only determine members of the given class but also compare the level of the membership to particular classes. For example, based on the fuzzy classification, one can say that dentist Gate belongs more to class **C1** than to classes **C2** or **C4** (see Figure 2). Membership values can be very useful for searching the most representative dentists in the given class. They become important criteria for evaluating dentists; one only needs to compare membership values of dentists to a particular class. For example, if two dentists have the same values of *Appointment* and *Waiting Time* for a particular care dental service, one can compare their membership values to the given class. The dentist with a higher membership value to **C1** class can be considered to be the best.

Membership value is similar to the deviation in the statistics and shows the distribution of the dentist membership to various classes. For example, to get dentists that belong to class **C4** (see Figure 2), one could get data output as database view (see Table 2) after fuzzy classification. Using given data view, one can say that dentist Merk belongs to class **C4** more than dentist Berg or dentist Gate because of higher value of membership function.

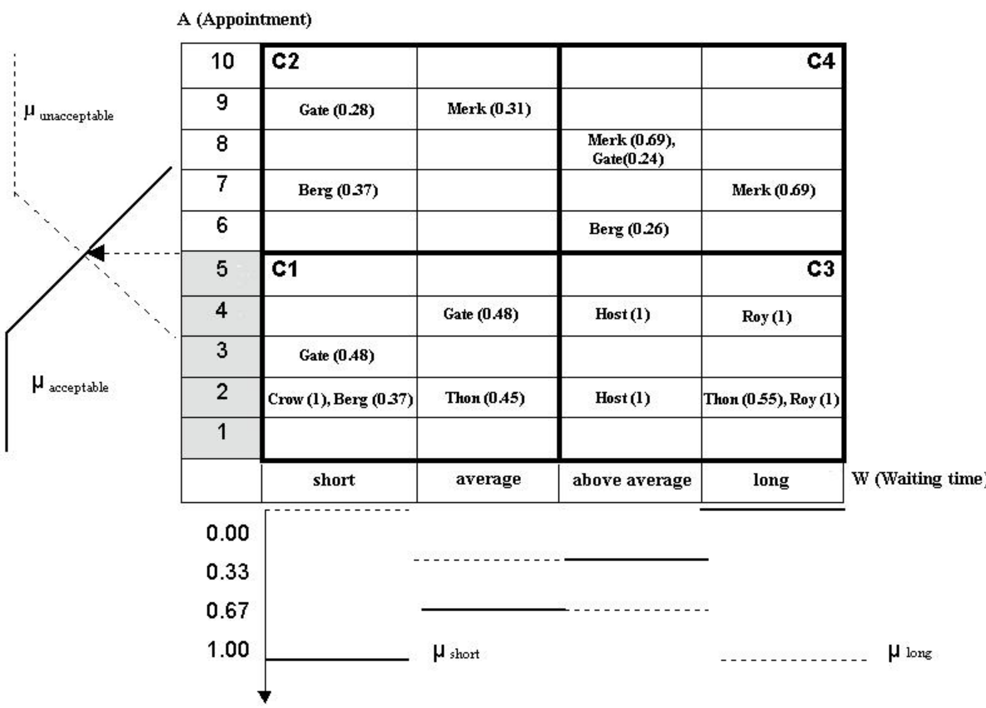


Figure 2. Fuzzy classification of dentists with appropriate membership values to the classes in brackets

The use of fuzzy classification provides higher accuracy of data presentation and gives a more precise description of data in generated reports. The drawback is that membership values of dentists have to be calculated with rather complicated mathematical formulas making SQL querying process a bit complicated.

Table 2

Database view for members of class **C4** ('*Interfere for improvements*')

Dentist	Class	Waiting time	Availability of appointment	Membership Value
Merk	C4	long	unacceptable	0.69
Berg	C4	long	unacceptable	0.25
Gate	C4	long	unacceptable	0.23

There are different approaches for implementing the presented fuzzy classification in practice. One of the approaches is to use fuzzy querying languages as an extension of the conventional SQL (Chen, 1998; Kacprzyk and Zadrozny, 2000). However, an introduction of new clauses into the SQL syntax changes conventional SQL functionality. To attract more users' attention to fuzzy applications, we propose to use only conventional SQL functionality for querying data with fuzzy classification. Database users do not need to add any new clauses to the SQL syntax and can continue using a conventional syntax of SQL, as if there is no fuzzy classification behind the data. This approach provides an added value to the conventional SQL and becomes very attractive for users of existing relational databases in dental practices.

SQL FOR FUZZY CLASSIFIED DATA QUERYING

The goal of SQL querying of data with fuzzy classification is to provide database views and/or reports of fuzzy classified data, similar to that presented in Table 2, using easy-to-use and familiar SQL queries. To implement the functionality of SQL for fuzzy classified data querying in relational databases, the best solution is to develop an interpreter as stored procedure that will translate conventional SQL commands into native SQL queries of a particular database. Users can formulate SQL queries with well-defined and familiar terms and do not need to know definitions of equivalence classes in details or fuzzy classification details behind the data. In addition, all complex mathematical formulas for calculating membership functions are hidden from users; the interpreter will take care for them. In the developed prototype, the following basic functionality of SQL for fuzzy classified data was implemented:

select <*Object*>
[**into**] <*View*>
from <*Relation*>
[**where**] <*Classification_condition*>

For example, SQL query

select Dentist **into** [FUZZY CLASSIFIED DENTISTS] **from** OFFERS

performs a classification (see Figure 2) of all dentists from OFFERS table (see Table 1) into [FUZZY CLASSIFIED DENTISTS] view. [FUZZY CLASSIFIED DENTISTS] view content is presented in Table 3.

Table 3

FUZZY CLASSIFIED DENTISTS view

Dentist	Class	Waiting time	Availability of appointment	Membership Value
Crow	C1	short	acceptable	1
Gate	C1	short	acceptable	0.48
Thon	C1	short	acceptable	0.45
Berg	C1	short	acceptable	0.37
Merk	C2	short	unacceptable	0.37
Gate	C2	short	unacceptable	0.31
Host	C2	short	unacceptable	0.28
Host	C3	long	acceptable	1
Roy	C3	long	acceptable	1
Thon	C3	long	acceptable	0.55
Merk	C4	long	unacceptable	0.69
Berg	C4	long	unacceptable	0.26
Gate	C4	long	unacceptable	0.24

Data from [FUZZY CLASSIFIED DENTISTS] view can be again queried using conventional SQL for further data analysis and report generation. An extended SQL query

select Dentist **from** OFFERS **where** (Appointment="unacceptable" and WaitingTime="long")

will generate the data presented in Table 2. SQL for fuzzy classified data fully complies with the conventional SQL. The only difference is that one needs to use an interpreter that will translate above presented SQL queries into the native SQL queries of the given database management system, for example, into Transact-SQL of the Microsoft SQL Server. To implement the SQL for querying of fuzzy classified data, appropriate database management system extensions will be added.

IMPLEMENTATION OF DATA FUZZIFICATOR IN RELATIONAL DATABASES

Implementation scheme of SQL for fuzzy classified data querying includes the following steps:

1. Design of database tables or views to query them later using SQL for fuzzy classified data. This step has to be carried out by database owners.
2. Design of database extensions (additional tables that contain linguistic variables, membership values and descriptions of atomic values). This step should be carried out by an expert in the given application area.
3. Design and implementation of interpreter for SQL transformation into native SQL for the given relational database management system using lexical and syntactical analysis of queries. This step should be carried out by software developer that will develop an interpreter in the form of the stored procedure for the given database management system.
4. Generation of database reports and views using SQL querying of fuzzy classified data formed on Steps 1 and 2.

The implementation scheme of Data Fuzzificator for conventional relational databases is shown in Figure 3.

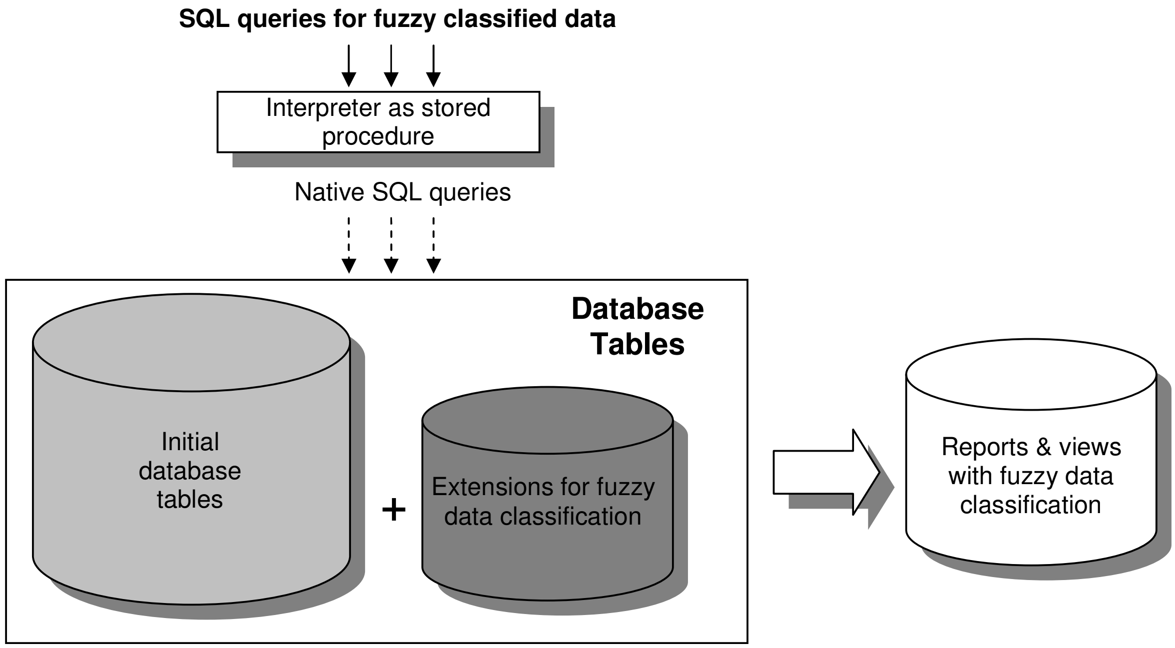


Figure 3. Scheme of Data Fuzzificator implementation in relational database

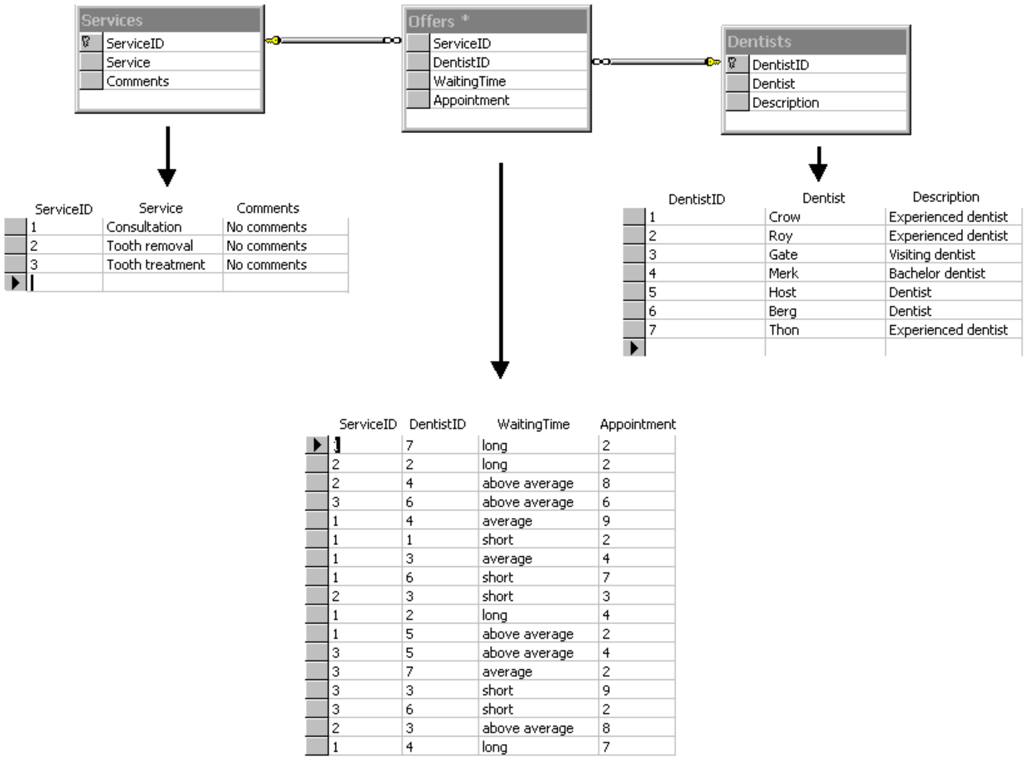
Processing of SQL queries by the interpreter includes two main stages:

- Syntactical analysis of SQL string clauses for fuzzy classified data querying;
- Execution of native SQL sub-queries (grouping of objects into classes, calculation of "compensatory AND" membership of objects to the classes and calculation of normalized membership of objects to classes) with parameters from SQL clauses for fuzzy classified data querying.

The fuzzy SQL interpreter was developed with the assumption that similar interpreter could be further developed for other platforms, like Oracle or SyBase, using their embedded SQL versions. The interpreter was realized as a stored procedure on the Microsoft SQL Server that could translate SQL queries for fuzzy classified data into normal Transact-SQL queries. The syntactical analysis (typical extraction of clauses from query string) of SQL queries for fuzzy classified data was implemented using Transact-SQL functions (SUBSTRING, LTRIM, RTRIM, LEFT and PATINDEX) for strings and embedded Microsoft SQL Server stored procedure (SP_EXECUTESQL).

The initial database scheme of our example in the normalized form is presented in Figure 4. To extend the current database scheme for working with the fuzzy classified data, database schema extensions (i.e. tables and relationships), were provided for fuzzy classification, as it is shown in Figure 5. In addition to initial OFFERS, SERVICES and DENTISTS tables (see Figure 5), the following classifications tables were added (see Figure 5): fAPPOINTMENT, fWAITINGTIME, fCLASSES, fCLASSESDEFINITION and fLINGUISTICVARIABLES. These additional tables include the definition of classes, linguistic variables and their dependencies. To query the given database and receive a fuzzy classified data report similar to that shown in Table 3, one may have to execute very large native Transact-SQL query without interpreter.

Instead of this, one can execute a simple SQL query with the help of the developed fuzzy SQL interpreter and receive the same results, as it is shown in Figure 6.



ServiceID	Service	Comments
1	Consultation	No comments
2	Tooth removal	No comments
3	Tooth treatment	No comments

DentistID	Dentist	Description
1	Crow	Experienced dentist
2	Roy	Experienced dentist
3	Gate	Visiting dentist
4	Merk	Bachelor dentist
5	Host	Dentist
6	Berg	Dentist
7	Thon	Experienced dentist

ServiceID	DentistID	WaitingTime	Appointment
1	7	long	2
2	2	long	2
2	4	above average	8
3	6	above average	6
1	4	average	9
1	1	short	2
1	3	average	4
1	6	short	7
2	3	short	3
1	2	long	4
1	5	above average	2
3	5	above average	4
3	7	average	2
3	3	short	9
3	6	short	2
2	3	above average	8
1	4	long	7

Figure 4. Initial database schema with content and relationships in our example

Thus, use of SQL with the developed fuzzy SQL interpreter is much easier and user-friendlier than the direct use of native Transact-SQL in Microsoft SQL Server. The drawback of the presented solution is that it is not flexible enough for possible database structure changes and may require additional programming to implement other operators (currently, it supports only one “compensatory AND” operator for computing membership function).

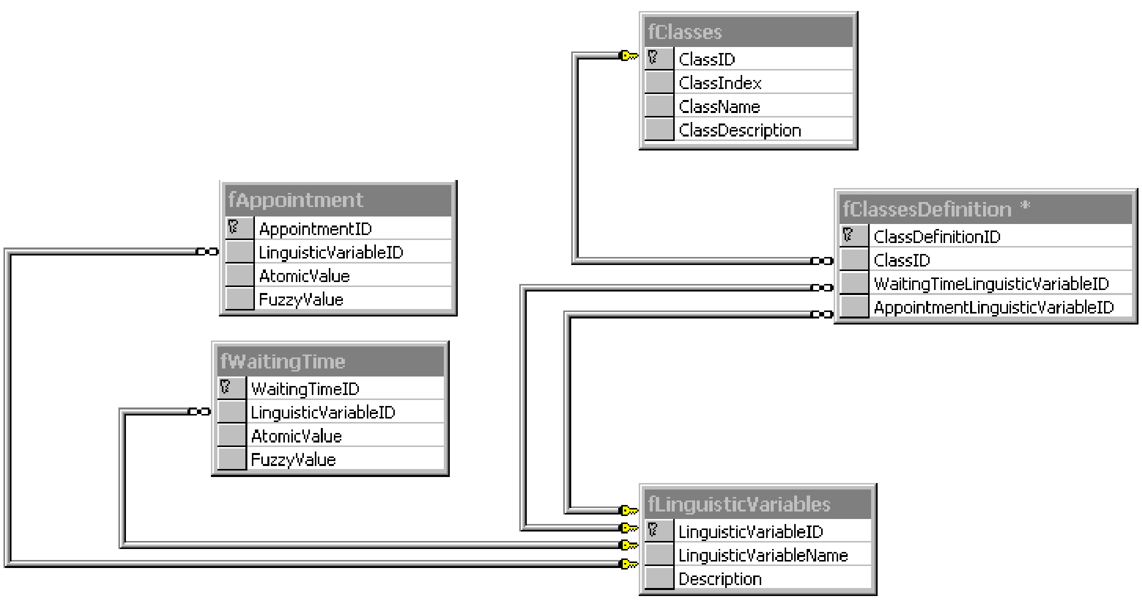
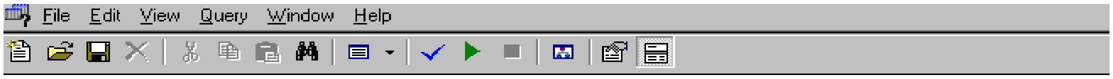


Figure 5. Introduced database extension tables and relationships for Data Fuzzificator




```
Execute fSQLInterpreter 'SELECT Dentists INTO FuzzyClassifiedDentists FROM Offers'
```

Class	Dentist	Waiting Time	Appointment	Fuzzy Membership
C1	Berg	short	acceptable	0.373273
C1	Crow	short	acceptable	1
C1	Gate	short	acceptable	0.480053
C1	Thon	short	acceptable	0.450107
C2	Berg	short	unacceptable	0.373273
C2	Gate	short	unacceptable	0.285915
C2	Merk	short	unacceptable	0.310396
C3	Host	long	acceptable	1
C3	Roy	long	acceptable	1
C3	Thon	long	acceptable	0.549893
C4	Berg	long	unacceptable	0.253453
C4	Gate	long	unacceptable	0.234032
C4	Merk	long	unacceptable	0.689604

Figure 6. Fuzzy classified data querying using SQL and fuzzy SQL interpreter as the stored procedure

CONCLUSION

We have shown how Data Fuzzificator concept can be used in dental care services, in particular, we developed a general and easy-to-use computer aided assessment method based on fuzzy classification to help dental practices in the assessment of patients' feedback. In comparison to conventional classification approaches, use of Data Fuzzificator provided a more accurate evaluation of patients' feedback, which is important for assuring the highest quality care dental services.

The fuzzy classification and use of conventional SQL queries (Data Fuzzificator concept) in our computer aided assessment method provided a much needed functionality of more accurate data extraction and analysis comparing to the conventional non-fuzzy classification and SQL querying. The main practical benefits of using Data Fuzzificator include the following:

1. Good integration with conventional databases and high flexibility for data analysis.
2. Data presentation with linguistic variables and fuzzy values in the report generation stage (such data presentation is more descriptive for users, since one does not have to think "in numbers" – one can use descriptive linguistic variables instead).
3. Users of Data Fuzzificator concept do not have to do large changes in their existing data structures (which may be quite large) – they will just have to extend them with a few additional tables that define fuzzy values and linguistic variables.

The drawback of the our approach is that it may require additional programming to implement other operators (currently, it supports only one "compensatory AND" operator for computing membership function).

REFERENCES

1. Bellma, M., and Vojdani, N. 2000. Fuzzy Prototypes for Fuzzy Data Mining. Studies in fuzziness and soft computing. Vol. 39: 175 – 186.
2. Borgodna, G., Leporati, A., Lucarella D., and Pasi, G. 2000. The Fuzzy Object-Oriented Database Management System. Studies in fuzziness and soft computing. Vol. 53: 209 – 236.
3. Bosc, P., and Pivert, O. 2000. SQLf Query Functionality on Top of a Regular Relational Database Management System. Studies in fuzziness and soft computing. Vol. 39: 171 – 191.
4. Brai, A., Vibert, J., and Koutlidis R. 1994. An expert system for the analysis and interpretation of evoked potentials based on fuzzy classification: application to brainstem auditory evoked potentials. Computers and Biomedical Research. Vol. 27(5): 351-366.
5. Chen, G. 1998. Fuzzy Logic in Data Modeling - Semantics, Constraints and Database Design. London: Kluwer.
6. Dubois, D., Nakata, M., and Prade, H. 2000. Extended Divisions for Flexible Queries in Relational Databases. Studies in fuzziness and soft computing. Vol. 39: 105 – 121.
7. Han, J., and Kamber, M. 2001. Data Mining: Concepts and Techniques. New-York: Morgan Kaufmann Publishers.
8. Kacprzyk, J., and Zadrozny, S. 2000. On Combining Intelligent Querying and Data Mining Using Fuzzy Logic Concepts. Studies in fuzziness and soft computing. Vol. 53: 67 – 84.
9. Schindler, G. 1998. Fuzzy Datenanalyse durch kontextbasierte Datenbankanfragen. Wiesbaden, Germany: DUV/Gabler.
10. Zadeh L. 1989. Knowledge representation in fuzzy logic. IEEE Trans. on Knowledge and Data Engineering. Vol. 1: 89 – 100.
11. Zimmermann, H. 1992. Fuzzy Set Theory - and Its Applications. London: Kluwer.