Fuzzy Querying based Tool for Building Courses Evaluation Tests

Livia Borjas[1], Josué Ramírez[1], Rosseline Rodríguez[2] and Leonid Tineo[2]
[1] *Departamento de Informática, IUT Federico Rivero Palacios, Caracas, Venezuela*
[2] *Departamento de Computación, Universidad Simón Bolívar, Caracas, Venezuela*

Keywords: Computer Aided Education, e-Learning, Fuzzy Querying.

Abstract: In this paper, we present a tool intended for helping in exam configuration based on the reutilization of questions according to user preferences. It is a real life application of fuzzy querying that fulfils an actual need of academic personal at a high studies institution of Venezuela. This application uses the fuzzy querying language SQLf, on top of an existing relational DBMS by means a logic layer named SQLfi. Final users of our application are professors of any area without knowledge of fuzzy sets and databases. We use criteria for the test preparation that support fuzzy terms. These terms can be adjusted to user preferences. Graphic user interfaces are provided in order to perform such adjusts as well as exam configuration and any other fuzzy querying operation. We present here the Database Design, the process test construction and the management of preferences.

1 INTRODUCTION

Professors throughout their carriers have to measure the performance of their students basically through the application of written tests. The design of these tests might become sometimes a discouraging process given its repetitive nature. Furthermore, some professors, due to their work as researchers and because of the administrative burden, do not have enough time to dedicate to the preparation of tests using all the creativity and awareness required.

By virtue thereof, we propose the creation of a software tool for supporting tests' preparation process. In connection therewith, the needs of professors will be considered the following criteria for the preparation of a test or question: difficulty (for example: low, moderate/low, moderate, high and very high), time to solve it (for example: long, moderate and short), correction time (for example: long, moderate and short). The aforementioned criteria are not accurate; therefore the definition of fuzzy terms is required. These terms are subjective since, for each term, each professor will differently conceive the range of values. Also, these terms do show the present imprecision at this application, as well as the feasibility of the fuzzy logic use in its queries.

Additionally, the preparation of tests is commonly based on existent tests to benefit from previous experiences. Consequently, a database storing this information is required. The use of these data might require the definition of searches based on preferences including fuzzy terms.

Another important factor is the consideration of the grades or results obtained in previous tests (for example: outstanding, excellent, good, regular and poor), which allows professors to project the expected performance of their students in a test.

This might be considered through the generation of statistics based on non-accurate gradual requirements defined by professors. It is also important for professors to have a scenario providing potential different tests based on established criteria, allowing for making the decision of choosing which tests better fit to the needs of the students being evaluated at a given time.

We present here a fuzzy querying based application for support professor's work of preparing evaluation tests. This application is made in the frame of an effort of the Venezuelan IUT Federico Rivero Palacio of providing professors some computer aided support tools for academic work.

This paper has the following structure: Section 2 presents Fuzziness in Databases. Section 3 briefly introduces the Fuzzy Querying Language SQLf that is used in this development research. Section 4 describes main System Features such as the Database Design, the functionality of Examination

Tests Construction with fuzzy queries expressing user desires and the Management of Preferences. Section 5 is devoted to giving some Concluding Remarks and Future works.

2 FUZZINESS IN DATABASES

The growing ambitious use of information systems in the amount of data volume and data recovery mechanisms has fostered the accelerated development of technology. In this regard, query specification mechanisms that resembling natural language expression and human thought is sought. Therefore, it would be convenient to provide flexible query capabilities allowing users to express requirements involving preferences.

A powerful tool used in the management of databases for the purpose of making queries in a more flexible way has been the fuzzy sets (Zadeh, 1965; Cox, 1995; Galindo, 2008; Pivert and Bosc, 2012), which vests traditional databases with a more intuitive approach closed to the natural language when retrieving information from data sources. This type of sets has been applied to several knowledge areas, such as Control Systems, Decision-making Systems, and Expert Systems. In (Ma and Yan, 2010), it is stated that the theory of fuzzy sets has been widely applied to extend several models and management systems database, which has resulted in many contributions. In this regard, different research projects have been conducted worldwide focused on including fuzziness in databases (Bosc and Pivert, 1995a; Kacpryzyk and Zadrozny, 1995; Goncalves and Tineo, 2008; Galindo, 2008; Yager, 2010; Ma and Ya, 2010; Pivert and Bosc, 2012). Nevertheless, there are few known developed applications that enjoy the benefits of fuzzy databases (Galindo et al, 2006; Goncalves and Tineo, 2008; Fernandez et al, 2008; Delgado et al, 2008).

We spoke of imprecision in database when the information is incomplete or uncertain. Thus, an approximate value or a term describing it is stored instead of an exact value. In general, we could be talking about data inaccuracies or imprecision in queries, i.e., imprecise terms may be stored as data values or could be used in the query criteria. Thus there are mainly two different approaches that address the vagueness of the databases. Those focused on the data (Galindo, 2008) and those focused on queries (Pivert and Bosc, 2012).

The results produced by an application system, based on conventional DBMS, strictly adjust to the compliance with the search criteria. But, they are not considering other results that approximate in certain degree to the response; these could be presented to user as alternate results. Search criteria in the systems based on classic DBMS are accurate, while Fuzzy DBMS allow for defining more flexible criteria that better adjust to human thought. Classical databases use all the results obtained, while Fuzzy Querying Systems only use those results that meet the minimum level of satisfaction defined by users. Classical queries limit their results since are based on the use of Boolean Logic, where a proposition only accepts two values: true or false.

Systems based on fuzzy databases are closer to human through. The use of fuzzy logic better adapts to real world and it even is able to understand and deal with linguistic terms or expressions in natural language, such as "it is really easy", "he is not too long", “he is too short”, etc.

3 FUZZY QUERYING LANGUAGE

Different fuzzy query languages have been proposed (Bosc and Pivert, 1995a; Galindo, 2005; Kacpryzyk and Zadrozny, 1995; Nakajima et al, 1995). On of the most remarkable efforts is SQLf (Bosc and Pivert, 1995a). It is an extended SQL with the application of the Fuzzy Set Theory aimed at expressing flexible queries on Relational Databases. This query language has been updated with different versions of SQL (Goncalves et al, 2009). SQLf allows the use of different fuzzy elements in constructions of SQL:2003 where a condition is involved.

For the purpose of defining the preferences of each user, SQLf counts on a series of sentences belonging to the DDL (Data Definition Language) for fuzzy components. In this (SQLf - DDL) terms such as predicates, modifiers, quantifiers and comparators are involved, which have a meaning that is specific to a group of users or individuals.

Fuzzy predicates are the atomic components of fuzzy logic. These correspond to the class of terms known as "linguistic labels". In SQLf, fuzzy predicates are defined through the following sentence:

```
CREATE FUZZY PREDICATE <name> ON
     <domain> AS <fuzzyset>
```

Where `<domain>` is a range of scale characters defined by the user, `<fuzzyset>` is a specification of the fuzzy set, which membership function be defined by

- a trapezium with the syntax: *(<value1>,<value2>,<value3>,<value4>)* using the "infinite" special value, the semantics of which is infinite;
- extension with the syntax: *{<d_1>/<v_1>, ... , <d_n>/<v_n>}* where <d_i> is a degree and <v_i> is a value.

Example: Predicate of Figure 1 is defined in SQLf by means the sentence:

```
CREATE FUZZY PREDICATE excellent
ON 0..20 AS (15,18,20,20)
```

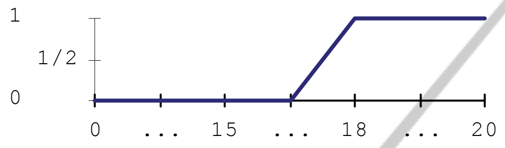


Figure 1: Membership function for the fuzzy predicate *excellent*, defined as (15,18,20,20).

An example for a predicate defined by extension is the following:

```
CREATE FUZZY PREDICATE favorite
ON topicName AS {
   1.0/design,
   0.3/querying,
   0.5/implementation,
   0.9/programming
}
```

As ever, SQLf has different querying structures for data manipulation. The main query structure in SQLf is the multirelational basic block. The main difference with respect to SQL is that the condition of the clause WHERE is a fuzzy condition. SQLf also includes the selection of the best answers according to their satisfaction degree that is named the calibration. Another difference is that the result of the query is not a bag but a fuzzy bag. The syntax of basic block is:

```
SELECT [DISTINCT] <attributes>
FROM <relations>
WHERE <fuzzy condition>
[WITH CALIBRATION k | α | k, α]
```

For obtaining, the results with excellent score, the fuzzy predicate *excellent* defined earlier by the user, is used in the following statement:

```
SELECT * FROM RESULT
WHERE score = excellent;
```

A significant difference between fuzzy querying and traditional Boolean querying is as follows: In a Boolean query, we establish crisp filtering criteria. Selected rows may be after ordered according sorting criteria or attributes. Selection and ordering are made on independent crisp criteria. On the other hand, in a fuzzy query, we establish flexible selection criteria. Such criteria are based on fuzzy logic involving linguistic terms. They express user preferences. Fuzzy criteria are combined giving a global satisfaction degree for each retrieved tuple. This degree is a measure of user preference fulfilment. Results should be automatically sorted in decreasing ordered of satisfaction degrees. It would be helpful in decision-making process.

In SQLf, it is possible to express the calibration of the solution, i.e. the selection of the best answers, in two senses. First, the qualitative sense where we indicate a minimum level of tolerance α. Only tuples with satisfaction degree greater or equal to α are retrieved. Second, the quantitative senses indicating a maximum number of desired answers k. Best k tuples according their satisfaction degree are retrieved. It is also possible to combine both senses in the querying statement. This is the effect of the clause WITH CALIBRATION.

4 SYSTEM FEATURES

We have developed an Automated System for the Preparation and Configuration of Tests named SAECE for its words in Spanish language. Remark that this is an application for a Venezuelan educational institute, the IUT Federico Rivero Palacios, therefore interfaces and linguistic terms are also given in Spanish that is the official language in Venezuela. Through the SAECE, professors might express their requests by using terms that are more similar to the commonly used natural language. Thus, we avoid inflexible queries that do not include their preferences. Additionally, the system facilitates the preparation and configuration of evaluations and/or tests based on existent tests, as well as the definition of search preferences and generation of statistics based on queries with non-accurate gradual requirements defined by users. SQLfi (Goncalves and Tineo, 2008), a server of SQLf, was used for the implementation of SAECE. SQLfi works as a layer that connects to the Oracle database manager and applies the Derivation Principle (Bosc and Pivert, 1995b) to evaluate a fuzzy query over the result of a classical query avoiding extra rows access in querying processing and keeping low the added cost of fuzzy querying features.

With this system, professors may create new tests based on existent tests; defining therefore parameters such as difficulty, duration, level of difficulty, time of response, time of correction.

4.1 Database Design

SAECE has a database that allows for retrieving tests created that might be configured in all their parameters. The idea is to have a test questions bank to be reused in different evaluations. The configuration of new evaluations would be made with this system aid. A simplified version of the ER diagram for this database is in Figure 2.

The PROFESSOR entity contains the user information as login, password, name, gender, and address. The QUESTION entity covers information over tests' questions or specific topic questions, as type, description, time answer, time of correction, difficulty, and references. The EVAL TEST entity contains information such as the evaluation name, date of elaboration, rating scale, percentage and description. The TOPIC SUBJET entity contains the different topics that a test has or the topics that a course covers. This entity keeps the topic name, time, references and description. The RESULT entity saves the test results to generate statistics that user wish. This information includes date, description and score. The COURSE entity contains relevant information of courses such as name and description. There is a relationship (evaluates) for indicate the topics including in a test and the different tests that evaluate a topic. Also, there is a relationship (about) to indicate the questions that deal a topic. There is a relationship to show the questions that compound a test. Also, it is possible that a question can to belong different tests. This relationship contains an attribute to indicate weighting of questions. There are relationships to indicate the evaluation tests that belongs to a professor, the results associates to an evaluation test, the professors that teaches the courses, the topics associates to the courses and the topics subordinate to another topics.

4.2 Examination Tests Construction

Tests might be partially recovered (certain questions) or totally recovered through search criteria based on fuzzy logics. A professor might also add questions to an existent test. SAECE allows for conducting different statistics such as: outstanding, excellent, good, bad and poor grades for the results obtained from a test application.

Difficulty: This is modelled in a rank of values between 0 and 5, being 5 the highest difficulty and 0 the lowest. This search will be made based on the criteria that the user has defined for each difficulty. For example the system has like predetermined configuration that the low difficulty is from 0 to 0.5, but this can be modified and the professor can place the values to him that consider appropriate for that difficulty.

Time of Correction: Possibly when the professors were created the new evaluations, to each question of the evaluation, they assign a time of correction. As well as the criterion of the difficulty, the system also has a configuration predetermined for this criterion that the professor can modify.

Time to solve it: For this criterion it is applied just like the Time of Correction.

Precision of the search: The number of questions that the system throws will depend on the precision of the search with which it is desired to collect the data. While upper it is the degree of the precision, there are possibilities that less results are obtained. The 100% of precision is the highest exigency. One is due to consider that the search can be done by the three criteria or by those which the user wishes. For example, a search can be made of: number of questions: 3; difficulty: medium; time of correction:

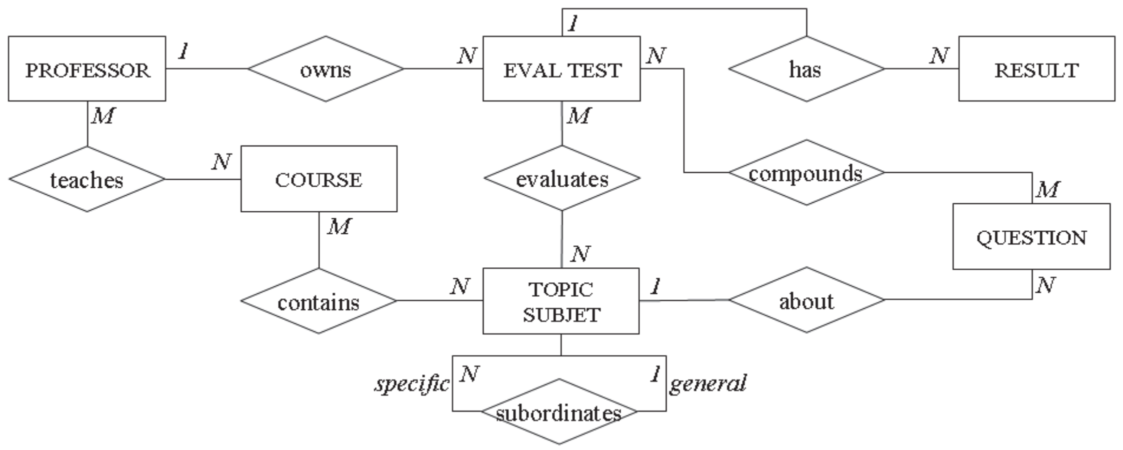


Figure 2: SAECE database Entity-Relationship diagram.

medium; time to solve it: medium; and precision: 20%.

SAECE brings a single interface for user choice of criteria (Figure 3). From this selection, SAECE builds the SQLf statement for being processed in SQLfi fuzzy querying system. In this case, the fuzzy query in SQLf should be the following one:

```
SELECT q.questionId, q.wording,
       q.type, q.difficulty,
       q.correctionTime,
       q.answerTime
FROM QUESTION q, TOPIC t,
     COURSE m,
     ABOUT tq
WHERE m.login = 'josueramirez' AND
m.courseId = 'programacion' AND
t.topicId = 1
AND tq.topicId = t.topicId
AND tq.questionId = q.questionId AND
q.difficulty = high
AND q.correctionTime = medium
AND q.answerTime = medium
WITH CALIBRATION 3, 0.2
```

This fuzzy query in SAECE should return a result as that shown in Figure 4.

4.3 Management of Preferences

SAECE has a set of linguistic terms for the expression of user preferences. These terms are used to describe imprecise conditions on questions attributes such as: difficulty, time of correction, time to solve it, and score. The set contains different terms for different attributes. Moreover, each user may give terms own interpretation and change it arbitrary at anytime.

The specification of these terms semantics is based on fuzzy sets. Nevertheless, SAECE provides a graphic user interface for establish preferences. This interface generates corresponding DDL statements in SQLf.

For example, for the score (nota) attribute, SAECE has the linguistic terms *deficient* (deficiente), regular, *good* (buena), *excellent* (excelente) and *outstanding* (sobresaliente). These terms are interpreted as fuzzy predicates defined by trapezium. Figure 5 shows the interface for these terms specification. This interface is rather intuitive to be handled by a final user not familiarized with fuzzy sets and fuzzy databases.

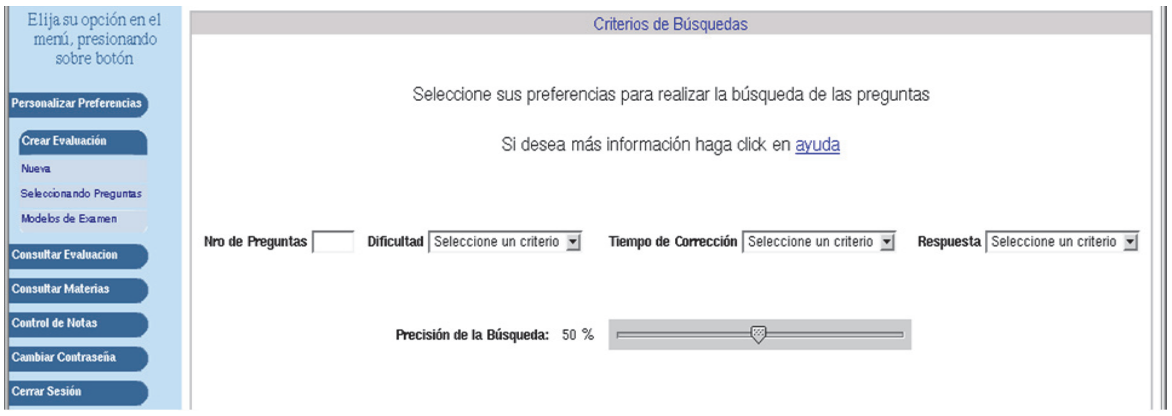
These linguistic terms of Figure 5 would be created with the SQLf statements:

```
CREATE FUZZY PREDICATE deficient
ON 0..20 AS (0,0,7,10);
CREATE FUZZY PREDICATE regular
ON 0..20 AS (7,10,12,20);
CREATE FUZZY PREDICATE good
ON 0..20 AS (12,15,16,20);
CREATE FUZZY PREDICATE excellent
ON 0..20 AS (16,17,18,20);
CREATE FUZZY PREDICATE outstanding
ON 0..20 AS (18,19,20,20);
```

It is also possible to change the scale range of test grades. If so, the built-in values are automatically adjusted to fit the scale.

5 CONCLUDING REMARKS

In the presented tool, user is able to define and use qualitative and quantitative selection criteria based on linguistic terms expressing user preferences. If a similar system were built with a classic querying language, it would impose more rigid selection criteria, limiting thus the expression of user



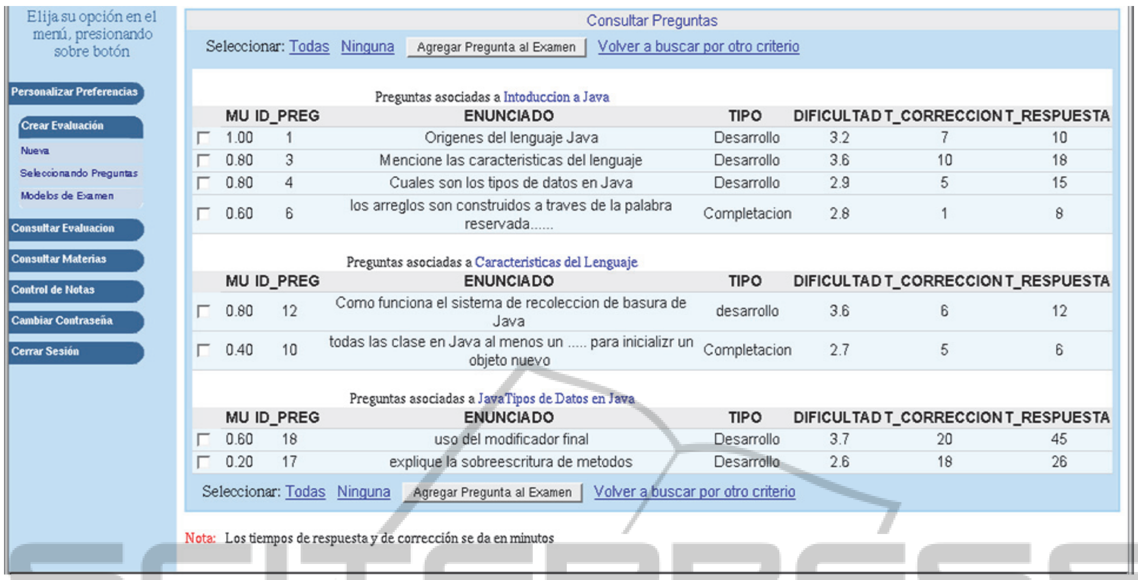
Seleccione sus preferencias para realizar la búsqueda de las preguntas

Si desea más información haga click en ayuda

Nro de Preguntas | Dificultad: Seleccione un criterio | Tiempo de Corrección: Seleccione un criterio | Respuesta: Seleccione un criterio

Precisión de la Búsqueda: 50 %

Figure 3: End user fuzzy querying interface. Here the user can settle down the criteria search for the questions that wish to assign to the examination according to criteria difficulty (dificultad), time of correction (tiempo de corrección), and time to solve it (respuesta).



Preguntas asociadas a Intoduccion a Java

	MU	ID_PREG	ENUNCIADO	TIPO	DIFICULTAD	T_CORRECCION	T_RESPUESTA
☐	1.00	1	Origenes del lenguaje Java	Desarrollo	3.2	7	10
☐	0.80	3	Mencione las caracteristicas del lenguaje	Desarrollo	3.6	10	18
☐	0.80	4	Cuales son los tipos de datos en Java	Desarrollo	2.9	5	15
☐	0.60	6	los arreglos son construidos a traves de la palabra reservada......	Completacion	2.8	1	8

Preguntas asociadas a Caracteristicas del Lenguaje

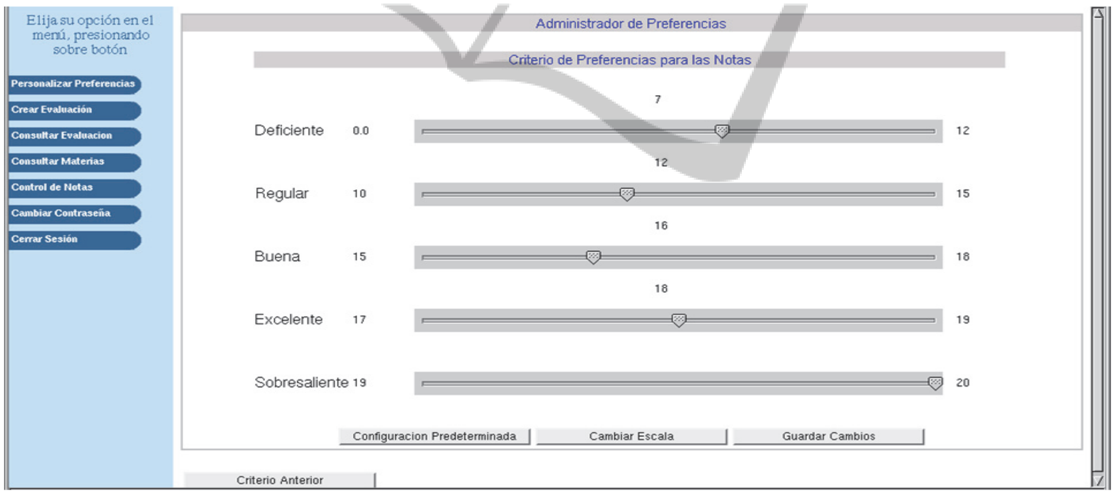
	MU	ID_PREG	ENUNCIADO	TIPO	DIFICULTAD	T_CORRECCION	T_RESPUESTA
☐	0.80	12	Como funciona el sistema de recoleccion de basura de Java	desarrollo	3.6	6	12
☐	0.40	10	todas las clase en Java al menos un ..... para inicializr un objeto nuevo	Completacion	2.7	5	6

Preguntas asociadas a JavaTipos de Datos en Java

	MU	ID_PREG	ENUNCIADO	TIPO	DIFICULTAD	T_CORRECCION	T_RESPUESTA
☐	0.60	18	uso del modificador final	Desarrollo	3.7	20	45
☐	0.20	17	explique la sobreescritura de metodos	Desarrollo	2.6	18	26

Nota: Los tiempos de respuesta y de corrección se da en minutos

Figure 4: Here are the results obtained according to the selected criteria. As it is possible to be observed the thrown questions, satisfaction degrees are associated to each one of the subjects selected according to the selected criteria of searches.


Elija su opción en el menú, presionando sobre botón
Personalizar Preferencias
Crear Evaluación
Consultar Evaluacion
Consultar Materias
Control de Notas
Cambiar Contraseña
Cerrar Sesión
Administrador de Preferencias
Criterio de Preferencias para las Notas
Configuracion Predeterminada
Cambiar Escala
Guardar Cambios
Criterio Anterior

Deficiente	0.0	7	12
Regular	10	12	15
Buena	15	16	18
Excelente	17	18	19
Sobresaliente	19		20

Figure 5: Fuzzy predicates configuration interface for *deficient* (deficiente), *regular*, *good* (buena), excellent (excelente) and outstanding (sobresaliente) fuzzy predicates on the attribute score.

preferences.

As a database system with fuzzy logic is used, queries might be more flexible and close to natural language. Furthermore, system lists the results obtained with their respective associated satisfaction degree; so users might know how the result meets the query criteria made, allowing among other things enhance their search criteria for future queries. Moreover, it would be very useful in decision-making support.

The presented tool helps in exam configuration based on the reutilization of questions according to user preferences. This problem has been proposed because it is in fact a real life need at the educational institute IUT Federico Rivero Palacio.

Provided tool was built using the fuzzy querying language SQLf. For the development, we use SQLfi. That is a fuzzy logic layer implementing SQLf on top of an existing relational DBMS. There are some SQLf features that have not been used in this application. We might improve it with such features.

It would also be useful to explore the need and the way for the incorporation of vague information in the database. SQLf does not currently support it, so is matter of more research in future works.

The application of fuzzy querying is relatively new; therefore, it gives an added value to this research area showing practical benefits.

This system shows the feasibility of building real applications using fuzzy queries. As future work, we will evaluate the acceptance of teachers in terms of usefulness. To do this, we will do the steps for implementing the system in the IUT Federico Rivero Palacios at the academic department of informatics. This evaluation should include how the teacher's ability and experience affects the tests construction. This allows validate the benefits expected from the developed system.

Another future work is to evaluate the system performance compared with other similar systems, in the context of a significant database size. Since SQLfi is developed as a logical layer on a relational DBMS, it adds overhead time to the queries response time. We have performed studies on the time overhead resulting in some cases excessive. At present time, we plan to develop a new version of SQLfi to decrease the time overhead.

We hope to get a good system acceptance by users. Thus, the database will grow to the extent of their use, so it is essential to improve the performance of SQLfi.

ACKNOWLEDGEMENTS

This work is supported in part by the Governmental Venezuelan Foundation for Science, Innovation and Technology FONACIT Grant G-2005000278. We would also give thanks to the King of Kings who as inspired us. We acknowledge that the intelligence and the knowledge are gifts of Him Hands by Him Love.

REFERENCES

Bosc, P., Pivert, O., 1995a. SQLf: A Relational Database Language for Fuzzy Querying. In: *IEEE Transactions on Fuzzy Systems*, Vol 3, No. 1.

Bosc, P., Pivert, O., 1995b, On the Efficiency of the Alpha-cut Distribution Method to Evaluate Simple Fuzzy Relational Queries. In: *Advances in Fuzzy Systems-Applications and Theory*, Vol. 4, Fuzzy Logic and Soft Computing. Bouchon-Meunier, B., Yager, R. and Zadeh, L. (Eds.), Wold Scientific, pp. 251-260.

Cadenas, J., Aguilera, A., Tineo, L., 2011. Rendimiento de consultas SQLf en arquitecturas débil y fuertemente acopladas. In: *Multiciencias*. Vol. 11, pp. 410-415.

Cox, E., 1995. Relational Database Queries using Fuzzy Logic. In: *Artificial Intelligent Expert*, pp 23-29.

Delgado, G., Aranda, V., Calero, J., Sánchez-Marañón, M., Serrano, J. M., Sánchez, D. and Vila, M. A., 2008. Building a fuzzy logic information network and a decision-support system for olive cultivation in Andalusia. In: *Spanish Journal of Agricultural Research*, Vol. 6, No. 2, pp. 252 – 263.

Fernández, A., Tineo, L., Ortiz, J., Borjas, L., 2008. FBMS Application for a Survey on Educational Performance. In: *LNCS*. Vol. 5181, pp. 753-760.

Galindo, J., 2005. New Characteristics in FSQL, a Fuzzy SQL for Fuzzy Databases. In: *WSEAS Transactions on Information Science and Applications 2*. Vol. 2, pp. 161-169.

Galindo, J., 2008. *Handbook of Research on Fuzzy Information Processing in Databases*. Hershey, PA, USA: Information Science.

Galindo, J., Urrutia, A., Piattini, M., 2006. Some Applications of Fuzzy Databases with FSQL, In: *Fuzzy Databases: Modeling, Design and Implementation*, Idea Group Inc.

Goncalves, M., Tineo L., 2008. SQLfi and its Applications. In: *Revista Avances en Sistemas e Informática*, Vol. 5, No. 2, pp. 33-40.

González, C., Goncalves, M., Tineo, L., 2009. A New Upgrade to SQLf: Towards an Standard in Fuzzy Databases. In: *20th International Workshop on Database and Expert Systems Application Proceeding*. pp. 442-446.

Kacpryzyk, J., Zadrozny, S., 1995. Fuzzy Queries in Microsoft AccessTM v.2. In: *Proc. of Fuzzy IEEE'95 Workshop on Fuzzy Database Systems and Information Retrieval*.

Ma, Z. M., Yan, L., 2010. A Literature Overview of Fuzzy Conceptual Data Modeling. In: *Journal of Information Science and Engineering*, Vol. 26, No. 2, pp. 427-441.

Nakajima H. et al, 1983. Fuzzy Database Language and Library-Fuzzy Extension to SQL. In: *Proc. of Second IEEE International Conference on Fuzzy Systems*. pp. 477-482.

Pivert, O., Bosc, P., 2012. *Fuzzy Preference Queries to Relational Databases*. Imperial College Press.

Yager, R., 2010. Soft Querying of Standard and Uncertain Databases. In: *IEEE Transactions on Fuzzy Systems.* Vol. 18, No. 2, pp. 336-347.

Zadeh, L.A., 1965. Fuzzy sets. In: *Information and Control 8*. pp 338-353.

Zhao, F.X., Ma, Z.M., 2009. Vague Query Based on Vague Relational Model. In: Yu W. and Sanchez (Eds.). *Advances in Computational Intelligence*, Vol. 61, pp. 229 – 238.