International Conference on Control, Engineering & Information Technology (CEIT'13)
Proceedings Engineering & Technology - Vol.1, pp.175-180, 2013

Flexible SQLf query based on fuzzy linguistic summaries

Ines Benali Sougui[1], Minyar Sassi Hidri[2], Amel Grissa Touzi[3]
[1,2,3]Université Tunis El Manar
[1,2]Ecole Nationale d'Ingénieurs de Tunis
Laboratoire Signal, Images et Technologies de l'information
BP. 37, Le Belvèdère 1002, Tunis, Tunisia
[3]Faculté des Sciences de Tunis
Laboratoire d'Informatique, Programmation, Algorithmique et Heuristiques
Campus Universitaire, Tunis 1060, Tunisia
[1]ines.benali@gmail.com, {[2]minyar.sassi,[3]amel.touzi}@enit.rnu.tn

***Abstract*—Data is often partially known, vague or ambiguous in many real world applications. To deal with such imprecise information, fuzziness is introduced in the classical model. SQLf is one of the practical language to deal with flexible fuzzy querying in Fuzzy DataBases (FDB). However, with a huge amount of fuzzy data, the necessity to work with synthetic views became a challenge for many DB community researchers. The present work deals with Flexible SQLf query based on fuzzy linguistic summaries. We use the fuzzy summaries produced by our Fuzzy-SaintEtiq approach. It provides a description of objects depending on the fuzzy linguistic labels specified as selection criteria.**

Keywords-Fuzzy relational databases, Flexible querying, Data summarization, SQLf, Top k query, Fuzzy FCA.

I. Introduction

In recent years, a lot of attention has been attracted to Fuzzy DataBases (FDB) that generalize the classical relational data model by allowing uncertain and imprecise information to be represented and manipulated. Data is often partially known, vague or ambiguous in many real world applications. Fuzziness is introduced in the classical model to deal with such imprecise information and several extensions of the model which are available in literature.

We are confronted more and more with the situation where applications need to manage fuzzy data and to profit their users from flexible querying [1], [2], [3].

In the last decades, a relational database language for fuzzy querying, called SQLf (or SQL Fuzzy), has a big success for the description and the manipulation of the FDB (Fuzzy DataBases)[4]. It extends the SQL by allowing the user to construct queries regarding atomic conditions defined by fuzzy sets. Each atomic condition combines satisfaction $\mu \in [0, 1]$ to an attribute value.

In addition, SQlf limits the number of answers by using a quantitative calibration (the k best responses or the top k query) or qualitative calibration (the data that satisfy the query with an upper threshold α).

However, the massive data reached today make necessary a better exploitation of the last. Several solutions have been proposed to solve this problem and to contribute in database summarization. Formal approaches are ones that have been proposed to surround this problem [5], [6], [7].

In [8], we have proposed a fuzzy linguistic summarization approach called Fuzzy-SaintEtiq using concept lattice which is the core of Formal Concept Analysis (FCA) [9]. This approach is an extension of the SaintEtiQ model [10] to support the fuzzy data.It consists of two major steps: the first, called pre-processing step, considers a fuzzy clustering that permits the generation of fuzzy data partition associating the DB records or tuples to many clusters by means of memberships' degrees. This is a form of optimization as much in DB navigation as minimization of the domain expert risks compared to linguistic summarization proposed in [7]. The second step, called post processing, uses fuzzy concept lattice in order to generate conceptual hierarchy. Furthermore, querying summaries is crucial since it makes it possible to rapidly get a rough idea of the properties of tuples in a given relation.

Even that the interpretation of a summary is simple, it becomes difficult to predict a high number of summaries. So, we are faced to the problem of their use.

In this work, we propose to exploit the hierarchical summaries of fuzzy-SaintEtiq under flexible SQLf query. The theory of fuzzy sets [11] used in the summarization process, allows flexible querying [12].

The rest of the paper is organized as follows: section 2 presents the basic concept of fuzzy FCA and the theoretical modeling of fuzzy queries. Section 3 presents an overview of our Fuzzy-SaintEtiq approach presented in [8]. Section 4 describes our new approach for the flexible SQLf query based on Fuzzy-SaintEtiq approach. Section 5 concludes the paper and gives some future work.

II. Preliminaries

A. Fuzzy FCA

In this section, we discuss the Fuzzy FCA proposed by [13] which incorporates fuzzy logic into FCA to represent vague information.

Definition 1. A fuzzy formal context is a triple $K_f = (G, M, I = \varphi(G \times M))$ where G is a set of objects, M is

a set of attributes, and I is a fuzzy set on domain $G \times M$. Each relation $(g, m) \in I$ has a membership value $\mu(g, m)$ in $[0, 1]$.

A fuzzy formal context can also be represented as a cross-table as shown in Table I. The context has three objects representing three documents, namely D_1, D_2 and D_3. In addition, it also has three attributes, *Data Mining (D)*, *Clustering (C)* and *Fuzzy Logic (F)* representing three researchable topics. The relationship between an object and an attribute is represented by a membership value between 0 and 1. A confidence threshold T can be set to eliminate relations that have low membership values [13]. Table I shows the cross-table of the fuzzy formal context with $T = 0.5$.

TABLE I
FUZZY FORMAL CONTEXT WITH $T = 0.5$

	D	C	F
D1	0.8	-	0.61
D2	0.9	0.85	-
D3	-	-	0.87

Each relationship between the object and the attribute is represented as a membership value in fuzzy formal context, then the intersection of these membership values should be the minimum of them, according to fuzzy theory [11].

Definition 2. Fuzzy formal concept: Given a fuzzy formal context $K_f = (G, M, I = \varphi(G \times M))$ and a confidence threshold T, we define $A* = \{m \in M | \forall g \in A : \mu(g, m) \geq T\}$ for $A \subseteq G$ and $B* = \{g \in G | \forall m \in B : \mu(g, m) \geq T\}$ for $B \subseteq M$.

A fuzzy formal concept (or fuzzy concept) of a fuzzy formal context $K_f = (G, M, I = \varphi(G \times M))$ with a confidence threshold T is a pair $(A_f = \varphi(A), B)$ where $A \subseteq G$, $B \subseteq M$, $A* = B$ and $B* = A$. Each object $g = \varphi(A)$ has a membership μ_g defined as $\mu_g = min(\mu(g, m))$ and $m \in B$ where $\mu(g, m)$ is the membership value between an object g and an attribute m, which is defined in I. Note that if $B = \{\}$ then $\mu_g = 1$ for every g.

Definition 3. Let (A_1, B_1) and (A_2, B_2) be two fuzzy concepts of a fuzzy formal context (G, M, I). $(\varphi(A_1), B_1)$ is the sub-concept of $(\varphi(A_2), B_2)$, denoted as $(\varphi(A_1), B_1) \leq (\varphi(A_2), B_2)$, if and only if $\varphi(A_1) \subseteq \varphi(A_2)$ $(\Longleftrightarrow B_2 \subseteq B_1)$. Equivalently, (A_2, B_2) is the super-concept of (A_1, B_1).

Definition 4. A fuzzy concept lattice of a fuzzy formal context K with a confidence threshold T is a set $F(K)$ of all fuzzy concepts of K with the partial order $\leq$ with the confidence threshold T.

Definition 5. The similarity of a fuzzy formal concept $K_1 = (\varphi(A_1), B_1)$ and its sub-concept $K_2 = (\varphi(A_2), B_2)$ is defined as:

$$E(K_1, K_2) = \frac{|\varphi(A_1) \cap \varphi(A_2)|}{|\varphi(A_1) \cup \varphi(A_2)|} \quad (1)$$

Figure 1 (a) gives the traditional concept lattice generated from table I, in which crisp values *Yes* and *No* are used instead of membership values. Figure 1 (b) gives the fuzzy concept lattice generated from the fuzzy formal context given in Table I.

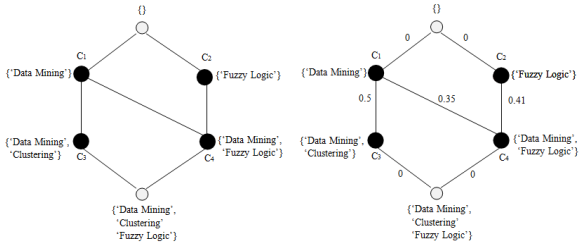

Fig. 1. (a) A concept lattice generated from traditional FCA. (b) A fuzzy concept lattice generated from Fuzzy FCA

B. Fuzzy queries' modeling

The SQLf [4] language extends the SQL language by allowing the user to construct queries on atomic conditions defined by fuzzy sets. Each atomic condition combines a satisfaction level $\mu \in [0, 1]$ with an attribute value. For all the attributes of an n-uplet, the semantics of degrees are the same which involve that all criteria are commensurable. SQLf query has the following syntax:
Select *[distinct] [k | α |k,α]* ⟨ *attribut* ⟩
From ⟨ *strict relation* ⟩
Where ⟨ *fuzzy condition* ⟩

where ⟨ fuzzy condition ⟩ can incorporate blocks of queries nested or partitioned. Parameters k and α from Select clause limit the number of answers by using a quantitative calibration(k best responses) or qualitative calibration(the data that satisfy the query with a threshold greater than α).

Example. Let be consider the employee DB relation presented in table II and the following query: *Finding employees about 40 years and having a high income*.

TABLE II
RELATIONSHIP EMPLOYEE

Id	Name	Age	Income
1	Pierre	38	2900
2	Jean	37	2800
3	Yvette	42	2700

About 40 and high income are the selection criteria defined by fuzzy sets. In particular, for the n-uplets of the relation employees we have used *High income=0.9/2900, 0.8/2800, 0.7/2700*, *About 40=0.8/38, 0.6/37, 0.8/42*.

Each n-uplet is associated with a vector that represents its position regarding the atomic conditions. Thus, the final result is evaluated and calculated using a standard triangular (eg min) to express a conjunction between the criteria. We then obtain: $Pierre(0.8) > Yvette(0.7) > Jean(0.6)$ means that the tuple 1 is preferred to tuple 3 which is preferred to tuple 2.

With SQLf language, preferences are considered only as constraints and are taken into account through the expression of fuzzy predicates commensurable, modeled using fuzzy sets of values more or less satisfactory.

Consequently, the results are totally ordered. Such predicates can be combined using a rich platform operators of logical fuzzy, with some reflecting the effects of such compensation or relative importance between criteria, have no counterpart in Boolean logic. Unlike other approaches (such as Preference SQL), data selection and order preferences operation are processed simultaneously.

Besides, *top-k query* has attracted much interest in many different areas such as network and system monitoring [14], information retrieval [15], sensor networks[16], [17], large databases [18], multimedia databases [19], spatial data analysis [20], [21], P2P systems [22], data stream management systems [23] etc.

The main reason for such interest is that they avoid overwhelming the user with large numbers of uninteresting answers which are resource-consuming.

The problem of answering top k queries can be modeled as follows [24]. Suppose that we have m lists of n data items such that each data item has a local score in each list and the lists are sorted according to the local scores of their data items. And each data item has an overall score which is computed based on its local scores in all lists using a given scoring function. Then the problem is to find the k data items whose overall scores are the highest.

III. Fuzzy data summarization

A. Overview of Fuzzy-SaintEtiq approach

In [8], we have proposed a fuzzy linguistic summarization approach Fuzzy-SaintEtiq which is based on FCA-based Summary model [25]. It takes the database records and provides knowledge. Figure 2 gives the system architecture.

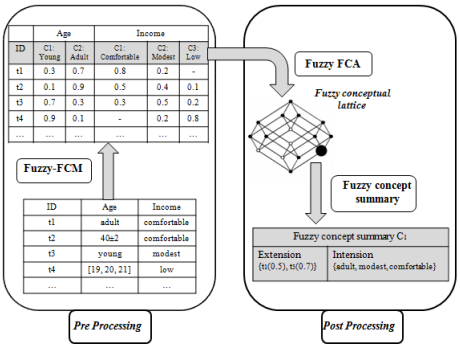

Fig. 2. The Overall process of Fuzzy-SaintEtiq

The summarization act considered like a process of knowledge discovery from database, in the sense that it is organized according to two following principal steps. The preprocessing step which organizes the FDB records in homogeneous clusters having common properties. This step gives a certain number of clusters for each attribute. Each tuple has values in the interval [0..1] representing these membership degrees according to the formed clusters. Linguistic labels, which are fuzzy partitions, will be assigned on attribute's domain. For the classification on these fuzzy data, a new algorithm, called Fuzzy-FCM [26] has been proposed. It's an extension of FCM algorithm in order to support different types of data represented by GEFRED model [1]. The second step, called the post processing, takes into account the result of the fuzzy clustering on each attribute, visualizes by using the fuzzy concepts lattices. Then, it nests them in a fuzzy nested lattice. Finally, it generalizes them in a fuzzy lattice associating all records in a simple and hierarchical structure. Each lattice node is a fuzzy concept which represents a concept summary. This structure defines summaries at various hierarchical levels.

For a more formal expression of a query, let be consider:

- $A = \{A_1, A_2, ..., A_k\}$ is a set of attributes,
- A_k is the k^{th} attribute which appears is query Q,
- $R(A)$ is the relation whose tuples are summarized,
- t_i is the i^{th} tuple, $i \in 1...N$,
- $L_k = \{l_{1k}, ..., l_{jk}\}$ is a set of linguistic terms of attribute A_k,
- μ_{ijk} is a membership degree of tuple t_i (record) to the linguistic term l_{jk} of attribute A_k,
- R_{Z_f} is the sub set of involved tuples in summary z_m,
- I_{Z_f} is the sub set of linguistic terms which appears in summary z_f,
- $Z_f = (R_{z_f}, I_{Z_f})$ is the concept summary, R_{zm} and I_{Z_f} are respectively the intension and the extension of the concept,
- $level$ is the level of the concept summary in the concept lattice,
- $|R_{Z_f}|$ is the number of candidates tuples in R_{Z_f}.

B. Illustrative example

Let consider a relation R =($IdTuple$, Age, $Income$, $ProfessionalBackground$) from an FDB Employees table. Table III gives a sample of FDB employees table.

TABLE III
Data sample

IdTuple	Age	Income	ProfessionalBackground
t_1	[38,39,40]	950	10
t_2	Adult	650	5
t_3	Young	700 ± 20	3
t_4	Adult	Poor	20
t_5	38	Poor	7
t_6	40 ± 2	Comfortable	12

Each cluster of a partition is labeled by linguistic descriptor provided by a domain expert. For example, the fuzzy label *Young* belongs to a partition built on the domain of age attribute. Linguistic variables associated with the attributes of R. These linguistic variables constitute the new attribute domains used for the rewriting of tuples in the summarization process. For clusters generation, we carry out a fuzzy clustering [27] while benefiting from fuzzy logic. This operation makes it possible to generate, for each attribute, a set of membership degrees. In fact, several fuzzy clustering algorithms have been proposed in the literature [28], [29].

Table IV presents the results of fuzzy clustering applied to *Age* and *Income* attributes. For *Age* attribute, fuzzy clustering

generates two clusters whereas, for Income attribute, there are three clusters. The minimal value (resp. maximal) of each cluster corresponds to the lower (resp. higher) interval terminal of the values of this last. Each cluster of a partition is labeled with a linguistic descriptor provided by the user or a domain expert. For this, the following abbreviations are used:

- For *Age* attribute: *YA* (Young Age) and *AA* (Adult Age).
- For Income attribute: *PI* (Poor Income), *MI* (Modest Income) and *CI* (Comfortable Income).
- For Professional background attribute: *A* (Associate), *E* (Expert) and *S* (Senior).

Table IV gives the result of fuzzy clustering from data in table III.

TABLE IV
DATA CLUSTERING

IdTuple	Age	Income	ProfessionalBackground
t_1	$YA^{0.5}$, $AA^{0.5}$	$MI^{0.6}$, $CI^{0.4}$	$A^{0.7}$, $E^{0.3}$
t_2	$YA^{0.4}$, $AA^{0.6}$	$PI^{0.4}$, $MI^{0.6}$	$A^{0.5}$, $E^{0.5}$
t_3	$YA^{0.7}$, $AA^{0.3}$	$MI^{0.8}$	$A^{0.8}$
t_4	$YA^{0.2}$, $AA^{0.8}$	$PI^{0.6}$, $MI^{0.4}$	$E^{0.3}$, $S^{0.7}$
t_5	$YA^{0.6}$, $AA^{0.4}$	$PI^{0.7}$	$A^{0.7}$, $E^{0.3}$
t_6	$YA^{0.5}$, $AA^{0.5}$	$CI^{0.8}$	$E^{0.4}$, $S^{0.6}$

IV. AN SQLF-BASED FLEXIBLE SUMMARY QUERYING

A flexible querying process of a DB can be divided into three steps [30]: extension of criteria, selection of results and ordering. The first step uses similarities between values to extend the criteria. It allows graded semantics for any criterion, which can now express *around 20* instead of being limited to the binary semantics of *equal to 20* or *between 18 and 22*. The second step, namely the selection of results, determines which data will participate in the answer to the query. The last step (ordering) follows the extension of criteria. It discriminates the results on the basis of their relative satisfaction to the graded semantics: a value of 20 is better ranked than a value of 18.

The following works, which are the research of flexible query in FDB, exemplify the use of fuzzy sets. They are essentially characterized by a tuple-oriented processing, the possibility to define new terms and especially the use of satisfaction degrees to extract the top-k query.

A. Fuzzy query expression

In the query SQLf we have two parameter α and k. As previously said parameters k and α from Select clause limits the number of answers by using a quantitative calibration(k best responses) or qualitative calibration(the data that satisfy the query with a threshold greater than α) .

The parameter k is given by the user and the parameter α is calculated depending on the number of clusters which involved in condition of Select query.

$$\alpha = \frac{1}{max(NClus)} \tag{2}$$

where $NClus$ is the number of clusters involved in the condition of select clause.

Let us consider the example in the table III. The queries are as follows:

Q$_1$
Select *3 0.5 Income, ProfessionalBackground*
From *Employees* ***Where*** *Age IN (Young);*

and

Q$_2$
Select *3 0.3 ProfessionalBackground*
From *Employees*
Where *Income IN (Comfortable) AND Age IN (Young);*

In a query, descriptors like Young, Comfortable in Q_1 and Q_2 are called *required characteristics* and embody the properties that a record must consider them as an element of the answers. A query also defines the attributes for which required characteristics exist. The set of these input attributes is denoted by $Inputs(A_Q)$. The expected answer is a description over a set of other attributes, denoted by $Outputs(A_Q)$. It is the complement of $Inputs(A_Q)$ relatively to A_Q (the set of attributes appears in the query Q):

$$Inputs(A_Q) \cup Outputs(A_Q) = A \tag{3}$$

and

$$Inputs(A_Q) \cap Outputs(A_Q) = \emptyset \tag{4}$$

Hence a query Q defines not only a set $Inputs(A_Q)$ of input attributes but also for each attribute A_k, the set $L_{A_k}(Q)$ of its required characteristics which define the set of linguistic terms of attribute A_k appears query Q. The set of sets $L_{A_k}(Q)$ is denoted by $L(Q)$.

For example, for Q_2, this set is determined as follows:

- $Inputs(A_{Q_2}) = \{Income, Age\}$;
- $Outputs(A_{Q_2}) = \{ProfessionalBackground\}$;
- $L_{Income}(Q_2) = \{RC\}$, $L_{Age}(Q_2) = \{AJ\}$;
- $L(Q_2) = \{L_{Income}(Q_2, L_{Age}(Q_2)\}$.
- The degree of membership to this query is 0.3 and the number of the desired result is 5.

B. Fuzzy query rewriting

The query rewriting in a logical proposition $P_f(Z_f, Q)$ used to qualify the link between the fuzzy summary Z_f and the query Q. $P_f(Z_f, Q)$ is in a conjunctive form in which all descriptors are literals. Then, each set of descriptors yields one corresponding clause.Thereafter we will apply an α-cut on this new form with the parameters α is calculated previously. This form is defined as follows:

$min(l_{11} \vee l_{21} \vee ... \vee l_{j1})(x), min(l_{12} \vee l_{22} \vee ... \vee l_{j2})(x), ...,$
$min(l_{1k} \vee l_{2k} \vee ... \vee l_{jk})(x) \geq \alpha$

$\iff (l_{11} \vee l_{21} \vee ... \vee l_{j1})(x) \geq \alpha,$
$(l_{12} \vee l_{22} \vee ... \vee l_{j2})(x) \geq \alpha$,... ,$(l_{1k} \vee l_{2k} \vee ... \vee l_{jk})(x) \geq \alpha$

$\iff l_{11}(x) \geq \alpha$ or $l_{21}(x) \geq \alpha$ or $l_{j1}(x) \geq \alpha$, $l_{12}(x) \geq \alpha$ or $l_{22}(x) \geq \alpha$ or $l_{j2})(x) \geq \alpha$, $l_{1k}(x) \geq \alpha$ or $l_{2k}(x) \geq \alpha$ or $l_{jk}(x) \geq \alpha$

$\implies P(Q) = (\alpha - cut(l_{11}) \vee \alpha - cut(l_{21}) \vee ... \vee \alpha - cut(l_{j1}))$
$\wedge(\alpha - cut(l_{12}) \vee \alpha - cut(l_{22}) \vee ... \vee \alpha - cut(l_{j2})) \wedge ... \wedge$
$(\alpha - cut(l_{1k}) \vee \alpha - cut(l_{2k}) \vee ... \vee \alpha - cut(l_{jk}))$

Example: Let be consider the query Q_3:
Select *3 0.3 ProfessionalBackground*
From *Employees*
Where *Age IN (Young, Adult)*
And *Income IN (Poor, Modest);*

We have then:

- $Inputs(A_{Q_1}) = \{Age, Income\}$;
- $L_{Age} = \{YA, AA\}$;
- $L_{Income} = \{PI, MI\}$;
- The degree of membership to this query is 0.3 and the number of the desired result is 3;
- $P(Q_3) = (0.3 - cut(YA) \vee 0.3 - cut(AA)) \wedge (0.3 - cut(PI) \vee 0.3 - cut(MI))$.

Let be consider v_f the valuation function. It is obvious that the valuation of $P_f(Q)$ depends on the summary Z_f. Thus $v_f(P_f(Q)_{Z_f})$ denotes the valuation of $P_f(Q)$ in the context of Z_f. $L_{A_i}(Z_f)$ the set of linguistic terms that appear in Z_f. We can distinguish between three assumptions:

- $Coresp(Z_f, Q) = Exact : v_f(P_f(Q)_{Z_f}) = true$ and $L_{A_i}(Z_f) \subseteq L(Q)$: All tuples including in Z_f verify the query Q;
- $Coresp(Z_f, Q) = False : v_f(P_f(Q)_{Z_f}) = false : L_{A_i}(Z_f) \neq L(Q)$: Linguistic terms appear in Z_f do not correspond to terms in query Q;
- $Coresp(Z_f, Q) = Indecision : \exists i,\ L_{A_i}(Z_f) - L(Q) \neq \emptyset$: There are some tuples in Z_f satisfying Q.

These situations reflect a global view of the matching of a fuzzy summary Z_f with a query Q.

C. Fuzzy k-query

The idea of the Fuzzy k-query algorithm is to search using the summary concept; which summary responds to the query and calculate their satisfaction degree. Then we will insert them in a list order by satisfaction degree. We will repeat these steps until ensure that there is not a branch in the summary concept that satisfies the query. Finally we can display the top k $\alpha - summary$ from the list of results. Figure 3 shows the principle of our approach.

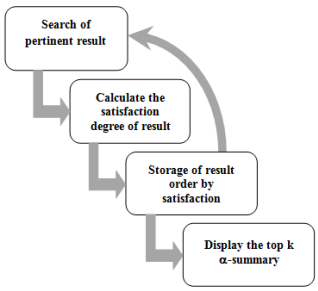

Fig. 3. An SQLf-based flexible summary querying approach step

Definition 6. An $\alpha - summary$, denoted as $\alpha - Z_f$, is a fuzzy summary $Z_f = (R_{Z_f}, I_{Z_f})$ in which R_{Z_f} is a collection of candidate records $R_{Z_f} = \{t_1, t_2, ..., t_N\}$ which represents the extent and I_{Z_f} is the intent. Each tuple t_i of R_{Z_f} existing in the $\alpha - summary$ has a membership value $\mu(t_i) \geq \alpha$. It can be formulated as follows:

$$\alpha - Z_f = \{\forall\ t_i \in Z_f | \mu(t_i) \geq \alpha\},\ with\ \ \mu(t_i) \in [0, 1] \quad (5)$$

D. Query evaluation

In this section, we try to evaluate the proposed approach. For this, the searching procedure should take into account all fuzzy summaries in the concept lattice that correspond to the query Q. The evaluation procedure is based on a generalization search and relies on the property of the concept lattice hierarchy. The algorithme 1 describes the different steps of the *fuzzy k-query* function where k is the number of answers, Z_{result} is a list of all $\alpha - summary$ responding to a query Q.

Algorithm 1 Fuzzy k-query(Z_f, k, α, Q)

Require: Z_f the fuzzy concept summary, k the number of answers, α the threshold greater data to satisfy the query Q.
Ensure: Result list of the top k $\alpha - summary$ responding to the query Q.

```
Result ⇐ ∅ List of summary with their satisfaction degrees
Z_result ⇐ PertinentResult(Z, 0, α, Q)
Z_result.first()
for i = 1 → k do
    Result.info() ⇐ Z_result.info()
    Z_result.next()
    Result.next()
end for
```

The algorithm 2 describes the different steps of the *PertinentResult* procedure.

Algorithm 2 PertinentResult(Z_f, $level$, α, Q)

Require: Z_f the fuzzy concept summary, $level$ the current level of summary, α the threshold greater data to satisfy the query Q.
Ensure: Z_{result} list of all $\alpha - summary$ responding to the query Q.

```
Z_result ⇐ ∅
if Coresp(Z, Q) = Exact then
    x.degree ⇐ Calcul − SD(Z, level)
    x.sum ⇐ α − summary(Z, α)
    if Z_result.Empty() then
        Z_result.Insertprevious(x)
    else
        Z_result.first()
    end if
    while not(Z_result.offlist()) do
        if Z_result.degree < x.degree then
            Z_result.Insertprevious(x)
        end if
        Z_result.next()
    end while
else
    if Coresp(Z, Q) = indecision) then
        for all Fuzzy summary z_f of Z_f do
            Level ⇐ level + 1
            Z_result ⇐ Z_result + PertinentResult(Z_f, level, α, Q)
        end for
    end if
end if
```

$Calcul-SD$ is the function to calculate the satisfaction degree SD of fuzzy summary Z_f.This degree is the max of the road that lets us find Z_f. It is determined as follows:

$$SD = max(\sum Fuzzy - score(Z_j, Z_{j+1})) \quad (6)$$

with $j = 1..p$, p the current level of Z_f and

$$Fuzzy - score(Z_p, Z_{p+1}) = \frac{|(Z_p) \cap (Z_{p+1})|}{|(Z_p) \cup (Z_{p+1})|} \quad (7)$$

$Coresp(Z, Q)$ is the function that allows to test the correspondence between the summary Z_f and the query Q that we have seen previously.

$\alpha - summary(Z, \alpha)$ is the function that uses the definition of $\alpha - summary$; the result is the $\alpha - Z = \{\forall t \in Z, \mu(t) \geq \alpha\}$.

The result of applying the algorithm on the concept lattice for queries Q_1, Q_2 and Q_3 is given in table V.

TABLE V
TOP k $\alpha - summary$

Query	$\alpha - Summary$
Q_1	$\alpha - z_{13}$=$\{t_1^{0.5}, t_3^{0.7}, t_5^{0.6}, t_6^{0.5}\}$
Q_2	$\alpha - z_{42} = \{t_1^{0.4}\}$, $\alpha - z_{34}$=$\{t_1^{0.4}, t_6^{0.8}\}$
Q_3	$\alpha - z_{21}$=$\{t_1^{0.5}, t_2^{0.6}, t_4^{0.4}\}$, $\alpha - z_{22}$=$\{t_1^{0.5}, t_2^{0.4}, t_3^{0.7}\}$, $\alpha - z_{23}$=$\{t_2^{0.4}, t_4^{0.6}, t_5^{0.7}\}$

V. CONCLUSION

In this paper we have presented a flexible SQLf query based on fuzzy linguistic summaries. Because, the SQLf query limits the number of answers by using a quantitative calibration or qualitative calibration, the mechanism of Fuzzy k-query explores a hierarchy of fuzzy summaries. Each fuzzy summary, represented by a fuzzy concept, performs a comparison with set-based query descriptors from a predefined vocabulary and applies an $\alpha - cut$. The result of this comparison determines whether the abstract will be part of the answer but it also conditions the exploration of a part of the hierarchy. Then using the satisfaction degree that will be calculated for each result allows us to return the *top k result*.

In brief, our objective is to found a result even in the case of the absence of summaries corresponding strictly to the query. To accommodate this target we will introduce a new kind of repaired query. Reparation is based on the idea that there could be a result semantically close to the query. To find these results, the query is modified using the best fuzzy summaries which is the near value answering the query.

REFERENCES

[1] J. Galindo, A.Urrutia and M. Piattini, *Fuzzy databases: modeling, design and implementation*, SA: Idea Group Publishing Hershey, 2006.

[2] M.A. Ben Hassine, A. Grissa Touzi, J. Galindo and H. Ounelli, *How to Achieve Fuzzy Relational Databases*, Handbook of Research on Fuzzy Information Processing in Databases, Information Science Reference, pp. 351-380, 2008.

[3] P. Bosc, L. Litard and O. Pivert, *Bases de données et Flexibilité: Les requtes Graduelles*, Techniques et Sciences informatiques, vol. 7(3), pp. 355-378, 1998.

[4] P. Bosc and O. Pivert, *SQLf: a relational database language for fuzzy querying*, IEEE Transactions on Fuzzy Systems, vol(3), pp.1-17, 1995.

[5] P. Bosc, D. Dubois, O. Pivert and H. Prade, *Résumés de données et ensembles flous - principes d'une nouvelle approche*, In LFA, pp. 333-340, 2000.

[6] D.H. Lee and M.H. Kim, *Database summarization using fuzzy ISA hierarchies*, IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics, vol. 27, pp. 68-78, 1997.

[7] G. Raschia, *SaintEtiq: Une approche floue pour la génération de résumés à partir de bases de données relationnelles*, Thèse de doctorat, Université de Nantes, 2001.

[8] I. BenAli-Sougui, M. Sassi-Hidri and A. Grissa-Touzi, *About Summarization in Large Fuzzy Databases*, The Fifth International Conference on Advances in Databases, Knowledge, and Data Applications, pp. 87-94, 2013.

[9] R. Wille, *Restructuring lattice theory: An approach based on hierarchies of concepts*, In Ivan Rival, editor, Ordered sets, Reidel, Dordrecht-Boston, pp. 445-470, 1982.

[10] G. Raschia and N. Mouaddib, *SaintEtiQ: A fuzzy set-based approach to database summarization*, Fuzzy Sets and Systems, vol.129, no. 2, pp. 137162, 2002.

[11] L.A. Zadeh, *Fuzzy Sets*, Journal of Information and Control, Vol. 8, pp.338-353, 1965.

[12] W.A. Voglozin, G. Raschia, L. Ughetto and N. Mouaddib, *Querying the SaintEtiq summaries - a first attempt*, FQAS, 2004.

[13] T.T. Quan, S.C. Hui,and T.H. Cao, *A Fuzzy FCA-based Approach to Conceptual Clustering for Automatic Generation of Concept Hierarchy on Uncertainty Data*, Concept Lattices and Their Applications Workshop, pp. 1-12, 2004.

[14] B. Babcock and C. Olston, *Distributed top-k monitoring*, SIGMOD Conference, 2003.

[15] B. Kimelfeld and Y. Sagiv, *Finding and approximating top-k answers in keyword proximity search*, PODS Conference, 2006.

[16] A. Silberstein, R. Braynard, C.S. Ellis and K. Munagala, *A sampling-based approach to optimizing top-k queries in sensor networks*, IEEE International Conference on Data Engineering, 2006.

[17] M. Wu, J. Xu, X. Tang and W.-C Lee, *Monitoring top-k query in wireless sensor networks*, IEEE International Conference on Data Engineering, 2006.

[18] A. Grissa-Touzi and H. Ounalli, *A New Approach For Top-k Flexible Queries In Large Database Using The Knowledge Discovered*, The Fourth International Conference on Advances in Databases, Knowledge, and Data Applications, pp.103-111, 2012.

[19] S. Chaudhuri and L. Gravano and A. Marian, *Optimizing top-k selection queries over multimedia repositories*, IEEE Transactions on Knowledge and Data Engineering 16(8), 2004.

[20] P. Ciaccia and M. Patella, *Searching in metric spaces with user-defined and approximate distances*, ACM Transactions on Database Systems, vol. 27(4), 2002.

[21] G.R. Hjaltason and H. Samet, *Index-driven similarity search in metric spaces*, ACM Transactions on Database Systems, vol. 28(4), 2003.

[22] R. Akbarinia, E. Pacitti and P. Valduriez, *Processing top-k queries in distributed hash tables*. Euro-Par Conf., 2007.

[23] K. Mouratidis, S. Bakiras and D. Papadias, *Continuous monitoring of top-k queries over sliding windows*, SIGMOD Conference, 2006.

[24] R. Fagin, J. Lotem and M. Naor, *Optimal aggregation algorithms for middleware*. J. of Computer and System Sciences 66(4), 2003.

[25] M. Sassi, A. Grissa-Touzi, H. Ounelli and I. Aissa, *About Database Summarization*, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, vol. 18(2), pp. 133-151, 2010.

[26] A. Grissa-Touzi, *An Alternative Extension of the FCM Algorithm for Clustering Fuzzy Databases*, The Second International Conference on Advances in Databases, Knowledge, and Data Applications, pp.135-142, 2010.

[27] D. Vanisri and Dr.C. Loganathan, *Survey on Fuzzy Clustering and Rule Mining*, International Journal of Computer Science and Information Security 8(4), pp. 183-187, 2010.

[28] M. Sassi, *Towards Fuzzy-Hard Clustering Mapping Processes*, Advances in Fuzzy Sets and Systems 9(1), pp. 37-63, 2011.

[29] R. Pradeep and S. Shubha, *A Survey of Clustering Techniques*, International Journal of Computer Applications 7(12), pp. 1-5, 2010.

[30] H.L. Larsen, *An approach to flexible information access systems using soft computing*, 32^{nd} Annual Hawaii International Conference on System Sciences 6042, 1999.