About Top-k Flexible Queries in Large Databases

Khaoula Mabrouki
Informatique
FST
Tunisia
mab.khaoula@gmail.com

Amel Grissa Touzi
Technologies of Information and Communications
ENIT
Tunisia
amel.touzi@enit.rnu.tn

Habib Ounalli
Informatique
FST
Tunisia
habib.ounelli@fst.rnu.tn

***Abstract*—The problem of obtaining efficient answers to top-k queries has attracted a lot of research attention. Unfortunately, current top-k query processing techniques focus on Boolean queries, and cannot be applied to the large Data Bases (DB) seen the gigantic number of data. In this paper, we propose a new approach for top-k flexible queries taking into account another degree of granularity in the process of the evaluation of the query. We start by generating a Meta-DB formed by a set of clusters resulting of a preliminary fuzzy classification on the data. This set represents a reduced view of the initial DB and permits to deduct the semantics of the initial DB. We prove that our approach permits an optimal search of the relevant data sources and generate automatically the better k answers while proposing a new operator called stratified operator for taking into account the user's preferences.**

Keywords-Large databases; flexible query; preference; formal concept analysis; Top k.

I. INTRODUCTION

An issue in extending databases is to increase the expressiveness of query languages. Based on fuzzy set theory, flexible querying enables users to express preferences inside requirements. Particularly, at the level of the requests addressed to large databases, the integration of the preferences allows to obtain more relevant answers. Besides, the user can be interested to receive a limited number of answers, in mind of "Top k queries ", k represents an ideal number of answers that is recommended to reach [1]. Unfortunately, current top-k query processing techniques focus on Boolean queries, and cannot be applied to large DB seen the gigantic number of data. Indeed, the majority of these systems use a function of score which remains difficult to establish seen the voluminous number of data. In addition, the approaches presented in this context present several limits, in particular in the hold in account of the dependencies between the search criteria that permit to detect the unrealizable queries (having an empty answer) with the user and in the generation and the scheduling of the turned over approximate answers.

In [2], a flexible and cooperative database flexible querying approach within the fuzzy theory framework has been proposed. This approach contributes two promising shares compared to the similar approaches. The first, taking into account the semantic dependencies between the query search criteria to determine its realizability or not. The second contribution related to its cooperative aspect in the flexible querying.

To ensure these functionalities, they have proposed to construct mono-attribute Type Abstraction Hierarchy (TAH) and a Multi-attribute Type Abstraction Hierarchy (MTAH) [3]. Problems lie in: 1) The generation of TAH's and MTAH from relieving attributes, 2) The storage and the indexing of such structures, and 3) The update of HATM.

In this paper, we propose a new approach for top-k flexible queries taking into account another degree of granularity in the process of the evaluation of the query. Seen that before, the interrogation of flexible queries was applied to raw data (the tuples of the base), the idea is to change this level of granularity and apply the clustering operation, so the interrogation will focus necessarily on clusters. Thus, we start by generating a Meta-DB formed by a set of clusters resulting of a preliminary fuzzy classification on data. This set represents a reduced view of the initial BD and permits to deduct semantics of the initial DB. The data classification is to divide a data set into subsets, called classes, so that all data in the same class are similar and data from different classes are dissimilar. Thus:

- The number of clusters generated by a classification algorithm is always less than the number of objects starting on which we apply the classification algorithm.
- All objects belonging to the same cluster have the same properties.

In this context, the query is modelled knowing the set of clusters modelling the meta-DB. To generate the meta-DB, we use the concepts of Clustering and Formal Concept Analysis (FCA). The use of these methods is justified by 1) fuzzy clustering has been a very successful data analysis technique as demonstrated in different domains [4]. These techniques allow data to belong to several groups (or clusters) simultaneously, with different membership degrees; 2) FCA is a method for knowledge representation that takes advantage of the features of formal concepts [5].

 ISBN: 978-1-61208-185-4

The rest of the paper is organized as follows. Section II makes a review of flexible querying and Formal Concept Analysis. Section III presents the problems of the other approaches and the contributions of this paper. Section IV describes our database flexible querying approach. Section V presents an example of relieving query. Section VI presents the experimental study we conducted to validate our approach. Section VII studies the complexity of our approach. Finally, Section IX concludes the paper and gives some future works.

II. BASIC CONCEPTS

In this section, we present the basic concepts of flexible queries and Formal Concept Analysis (FCA).

A. Flexible queries

The traditional systems of interrogation distinguish two categories of data: those which satisfy the search criteria and those which do not satisfy them. The principle of the flexible interrogation aims at extending this bipolar behaviour by introducing the concept of approximate pairing. Thus, an element returned by a request will be at least relevant according to its satisfaction degree to the constraints of interrogation. Four principal approaches have been proposed to express and evaluate the flexible queries: 1) Use of the secondary criteria [6][7], 2) Use of the distance and the similarity [8] [9], 3) Expression of the preferences with linguistic terms [10] and 4) Modelling of the inaccuracy by the fuzzy subsets theory [11][12]. A comparative study of the systems of flexible interrogation has been achieved in [13][14].

The problem of the expression of the users' preferences in the flexible queries received much attention these last years [11][15][16][17][18].In general, it is possible to distinguish two families of approaches for the expression of the preferences: implicit and explicit.

In the implicit approach, mechanisms of numerical scores, commensurable or not, are used to represent the preferences. In the first case, the values of preferences can be aggregated to deliver a total value and to define a total order on the answers. In the second case, when there is not commensurability, only a partial order of the answers, based on the order of Pareto [19], is possible for the incomparable classes of answers are built. This approach is detailed in [20] and is illustrated by the Skyline operator [21] or in PreferenceSQL [22].

In the explicit approach, the preferences are specified by binary relations of preferences and in the majority of the cases, a partial order is obtained on the tuples. In addition, the preferences can be regarded as being constraints (preferences obligatory) and wishes (optional preferences). This reveals that this bipolar vision [23] of the preferences makes it possible to bring a refinement of the set of the answers: to satisfy the constraints, then, if possible, wishes. The preferences of the users can also be expressed by criteria of selection based on fuzzy sets. The predicates are not then any more in *"all or nothing"* but can be more or less satisfied. Other researchers used charts to model the preferences on a great number of alternatives. As an example, we can quote the Conditional Preferences Networks (CP-Nets)[24], which constitute a chart appraisal for modelling the preferences.

Bosc et al. [25] suggest the introduction of the preferences in the form of subsets of n-uplets (stratified divisor). Thus, they used the terms of ***"stratified divisor"*** and ***"stratified division"***. Consequently, an element x of the dividend will be more acceptable as it will be associated with a large number of subsets (S_i) defining the divisor. Three types of requests studied by Bosc et al. are expressed in SQL language, where the dividend can be an intermediate relation and the stratified divisor is given explicitly by the user or is the result from sub queries.

As example of principal systems of interrogation with preferences, we can quote, the systems PreferenceSQL [22] and Preference Queries [16] which are based on a partial order, consequently, they deliver to the user the not dominating tuples. Preference SQL also incorporates a concept of bipolarity in the Preferring clause. The system top-K queries [15][24]. uses an ad-hoc score function f and delivers the k better answers of the total order obtained by f. However, this score function remains difficult to establish. The SQLf language uses the fuzzy set theory to define the preferences and makes the assumption of commensurability. It offers a framework founded to combine obligatory preferences.

B. Fuzzy Conceptual Scaling and FCA

Conceptual scaling theory is the central part in Formal Concept Analysis (FCA). It allows to embed the given data into a much more general scale than the usual chains and direct products of chains. In the direct products of the concept lattices of these scales, the given data can be embedded. FCA starts with the notion of a formal context specifying which objects have which attributes and thus a formal context may be viewed as a binary relation between the object set and the attribute set with the values 0 and 1.

In [26], an ordered lattice extension theory has been proposed: Fuzzy Formal Concept Analysis (FFCA), in which uncertainty information is directly represented by a real number of membership value in the range of $[0, 1]$. This number is equal to similarity defined as follows:

 ISBN: 978-1-61208-185-4

Definition 1: The similarity of a fuzzy formal concept $C_1 = (\varphi(A_1), B_1)$ and its sub-concept $C_2 = (\varphi(A_2), B_2)$ is defined as:

$$S(C1, C2) = \frac{|\varphi(A_1) \cap \varphi(A_2)|}{|\varphi(A_1) \cup \varphi(A_2)|}$$

where $\cap$ and $\cup$ refer intersection and union operators on fuzzy sets, respectively;
φ is the relation which associates degrees to the elements of a fuzzy set $I = X \times V$ (X is the set of objects and V is the set of attributes). Each pair $(x_i, v_j) \in I$ has a membership degree $\mu(x_i, v_j) \in [0, 1]$.

In [27][28], we showed how these FFCA are very powerful as well in the interpretation of the results of the fuzzy clustering and in optimization of the flexible query.

Example: Let a relational database describing travel, means of transport and hotels, which a passenger can reserve for business trips or for pleasure. The primary key of each relation is underlined:

Travel (idV, price, stay, date, typeStay, offer , idT, idH) **V**
Transport (idT, means, route, comfort) **T**
Hotel (idH, category, name, region, city, restaurant) **H**

Price $\in [100, 10000]$, Stay $\in [1, 12]$, Category $\in [1, 5]$ and Comfort $\in [1, 8]$. The result of fuzzy clustering (using Fuzzy C-Means [29]) and the application of the α-*Cut* are shown in Table I and Table II.
We use a Cut of a fuzzy context U, noted, α-*Cut*, and defined as the inverse of the number of clusters obtained. It is given by the following expression:

$$\alpha\text{-}Cut(U) = (c)^{-1}$$

For Stay attribute (respectively Price, Category and Comfort), fuzzy clustering generates three clusters (C1, C2 and C3) (respectively three clusters (C4, C5 and C6), two clusters (C7 and C8) and three clusters (C9, C10 and C11)). In our example, α-*Cut*(Stay) = 0.3, α-*Cut*(Price) = 0.3, α-*Cut*(Category) = 0.5 and α-*Cut*(Comfort) = 0.3.

Table I
FUZZY CONCEPTUAL SCALES FOR STAY AND PRICE ATTRIBUTES WITH α-*Cut*

	Stay			Price		
	C1	C2	C3	C4	C5	C6
T1	-	0.67	-	0.86	-	-
T2	-	0.94	-	0.94	-	-
T3	0.98	-	-	0.47	0.47	0.47
T4	-	0.94	-	0.35	-	0.35
T5	-	0.95	-	0.49	-	0.45
T6	0.83	-	-	0.67	0.27	0.05

Table II
FUZZY CONCEPTUAL SCALES FOR CATEGORY AND COMFORT ATTRIBUTES WITH α-*Cut*

	Category		Comfort		
	C7	C8	C9	C10	C11
T1	0.98	-	-	0.81	-
T2	0.92	-	-	0.88	-
T3	0.62	-	-	-	0.85
T4	-	0.93	0.61	-	0.34
T5	-	0.9	-	-	0.89
T6	0.53	-	0.51	0.43	-

III. PROBLEMS AND CONTRIBUTIONS

The majority of the current approaches presented to support flexible queries have several limits, in particular 1) in the consideration of the dependencies between the search criteria that permit to detect the unrealizable requests (having an empty answer) with the user 2) and in the generation and the scheduling of the turned over approximate answers.

At the level of the requests addressed to large databases, the current top-k query processing techniques focus on Boolean queries, and cannot be applied to the large DB seen the gigantic number of data. The majority of these systems [15][24] uses an ad-hoc score function f and delivers the k better answers of the total order obtained by f. However, this score function remains difficult to establish seen the voluminous number of data.

In this paper, we propose a new approach for top-k flexible queries taking into account another degree of granularity in the process of the evaluation of the query.
The advantage of this approach is that it can be applied to the large DB and that it does not require modifying the SQL language. The contributions of our approach are (1) the extraction of the dependencies between the search criteria to detect the unrealizable query; (2) an optimal search of the relevant data sources for a given query; (3) the automatic generation of the better k answers while proposing a new operator called stratified operator for taking into account the user's preferences.

IV. THE PROPOSED APPROACH

In this section, we propose a relieving approach within the fuzzy set framework. We consider a relational database containing relieving attributes i.e. attributes which the users can use in a predicate of comparison containing a linguistic term.

In this paper, we limit ourselves to the relieving numerical attributes. Figure 1 shows the proposed approach.
This approach consists on three principal steps:

 ISBN: 978-1-61208-185-4

1) Generate the meta-DB: Apply a fuzzy algorithm of classification (example FCM): this allows generating clusters which overlap. Each cluster represents a set of the data verifying the same properties.
2) Deduct the semantic of the data: Represent the matrix obtained in the first step under the form of fuzzy concept lattice (FCA).
3) Generate the k better answers: Apply a stratified operator.

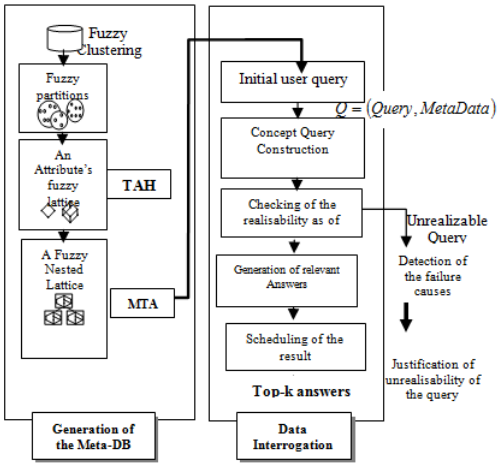

Figure 1. Proposed approach

In this part, we present the theoretical foundations of the proposed approach based on the following properties:

- The number of clusters generated by a classification algorithm is always lower than the number of starting objects to which one applies the classification algorithm.
- All objects belonging to one same cluster have the same proprieties. These characteristics can be deduced easily knowing the center and the distance from the cluster.
- The size of the lattice modelling the properties of the clusters is lower than the size of the lattice modelling the properties of the objects.
- The management of the lattice modelling the properties of the clusters is optimum than the management of the lattice modelling the properties of the objects since the number of clusters is less than to the number of database objects.

To model the expression of the users' preferences, and generate the top-k answers we define **the stratified operator** as follows:

***Definition** 2:* The **stratified operator** r whose schema is R (A, X) by the relation s whose schema is S(B) is expressed according to the mechanism of partitioning and a similar expression is suggested here, as follows [25]:

select top k X from r [where condition] group by X having set(A) contains $v_{1,1}, \ldots, v_{1,j_1} and - or \ldots and - or v_{n,1}, \ldots, v_{n,j_n}$.
This expression infers an order on the elements of the divisor namely $S = (S_1 = v_{1,1}, \ldots, v_{1,j_1}) \succ \ldots \succ (S_n = v_{n,1}, \ldots, v_{n,j_n})$ where a $\succ$ b denotes the preference of a over b.

An ordinal scale L with labels l_i (such as $l_1 > \ldots > l_n > l_{n+1}$) is associated with this relation and it is used to attribute levels of satisfaction to the elements of the result of a stratified division (l_1 corresponds to the maximal satisfaction and l_{n+1} express the refusal); they are the counterpart of 1 and 0 in the unit interval.

The principle of interpretation of these queries is to involve all levels for which the association is completely valid. An element is the more preferred as it is associated with a set S_i so strongly preferred. The degree of satisfaction of x is expressed by a vector $V(x)$ of dimension n where $V(x)[i] \in]0..1]$ if x is associated with the fuzzy values of S_i, 0 otherwise. The classification of elements means comparing these vectors according to the lexicographical order($\succ_{lex}$):

$$x \succ y \Leftrightarrow V(x) \succ_{lex} V(y) \Leftrightarrow \exists k \in [1, n]$$
$$that \ \forall j < k, V_j(x) = V_j(y) and V_k(x) > V_k(y)$$

The scale L is not used then directly, but we notice that the order obtained reflects it in the sense that if $i < j$, $V_i(x)$ is more important than $V_j(x)$ quite as $l_i > l_j$. It would nevertheless be possible to use a symbolic scale to make the comparison of elements within the framework of this request. The scale in question have 2^n levels. This would be made by means of a function transforming a vector V into a whole score as follows:

$$SAT(x) = \sum_{i=1}^{n} (V_i(x) * 2^{(n-i)}) \quad (1)$$

It is easy to show that the preference of x on y so defined is equivalent to $SAT(x) > SAT(y)$.

V. Example of relieving query

For better explaining this step, we consider a relational database table describing travel, means of transport and hotels. Let's the following query:

 ISBN: 978-1-61208-185-4

Select	V.idV, V.price, T.means, T.comfort , H.idH, H.name	
From	TRAVEL **V**, TRANSPORT **T**, HOTEL **H**	
Where	V.idH = H.idH	
And	V.idV = T.idT	
And	T.means = 'plane'	(A_1)
And	T.route = 'direct'	(A_2)
And	V.offer <> 'circuit'	(A_3)
And	V.typeStay = 'full-board'	(A_4)
And	***V.stay*** = 7	$(Pref_1)$
And	***V.price*** = 800	$(Pref_2)$
And	***H.category*** = 3	$(Pref_3)$
And	***T.comfort*** = 3	$(Pref_4)$

In this query, the user wishes that his preferences be considered according to the descending order: Stay, Price, Category and Comfort with Top-k=3. In other words, returned data must be ordered and presented to the user according to these preferences. Without this flexibility, the user must refine these search keys until obtaining satisfaction if required since it does not have precise knowledge on the data which it consults.

According to the criteria of the query Q , only the ($Pref_1$, $Pref_2$, $Pref_3$ and $Pref_4$) criteria correspond to relaxable attributes. Initially, we determine, starting from the DB, the tuples satisfying the non relaxable criteria A_1,A_2,A_3 and A_4.

Select	V.idV, V.price, T.means, T.comfort , H.idH, H.nom	
From	TRAVEL **V**, TRANSPORT **T**, HOTEL **H**	
Where	V.idH = H.idH	
And	V.idV = T.idT	
And	T.means = 'plane'	(A_1)
And	T.route = 'direct'	(A_2)
And	V.offer <> 'circuit'	(A_3)
And	V.typeStay = 'full-board'	(A_4)

These tuples, answering this request, are broken up into clusters according to labels of the relaxable attributes Stay, Price, Category and Comfort.

Following this operation of clustering, the expert can assign linguistic terms to the clusters generated for each relaxable attribute. The minimum value (resp. maximum) of each cluster corresponds to the lower (resp. higher) interval terminal of its values (of this cluster). For example, the linguistic terms *"Short, Medium and High"* (respectively *"Low, Medium and High"* , *"Medium and High"* and *"Low, Medium and High"*) will be associated to the relaxable attribute Stay (respectively Price, Category and Comfort).

A. Construction of the query concept

We define a query concept $Q = (Q_A, Q_B)$ where Q_A is a name to indicate a required extension and $Q_B$is the set of clusters describing the data reached by the query. The set Q_B of clusters is determined by the following procedure:

Procedure Construction of the query concept
Input: Vector $V(A) = \{v_j : j = 1, \ldots, C(A)\}$ of cluster
centers of relaxable attribute A and
the value of Q associated to this last.
Output: Query concept $Q = (Q_A, Q_B)$.
Begin
Step 1**:** Calculate the membership degrees of the specified
clusters for each value of the criterion of Q associated to the
relaxable attribute A.
Step 2**:** Apply $\alpha - Cut$ to generate the fuzzy context.
Step 3**:** Form the set Q_B of clusters whose membership is higher than the $\alpha - Cut$ value.
End Procedure

Table III and Table IV present the membership degrees associated to the query. These degrees are obtained while basing on memberships matrix obtained by a fuzzy clustering algorithm. Then, we apply the $\alpha - Cut$ for each attribute to minimize the number of concepts.

Table III
QUERY MEMBERSHIPS DEGREES (STAY AND PRICE)

Stay			Price		
C1	C2	C3	C4	C5	C6
0.353	-	0,331	0,333	0,333	0,333

Table IV
QUERY MEMBERSHIPS DEGREES (CATEGORY AND COMFORT)

Category		Comfort		
C7	C8	C9	C10	C11
0.516	-	0,334	0,351	-

According to our example, the query Q seek the data sources having the metadata $Q_B = \{C1, C3, C4, C5, C6, C7, C9, C10\}$.

B. Generation of the approximate answers

In our example, the request Q is realizable then we can build his conceptlattice. Figure 2 represents the concept

lattice associated with this request. The concepts are either new concepts, or concepts modified following construction of the request Q knowing the HATM. These concepts are the only ones which divide clusters with the request and which can thus contain relevant answers.

Let given a query $Q = (Q_A, Q_B)$, all the relevant data sources are in the extension of Q and of its subsumers in the concepts lattice since the intention of each one of these concepts are included in Q_B(the intention of the query concept).

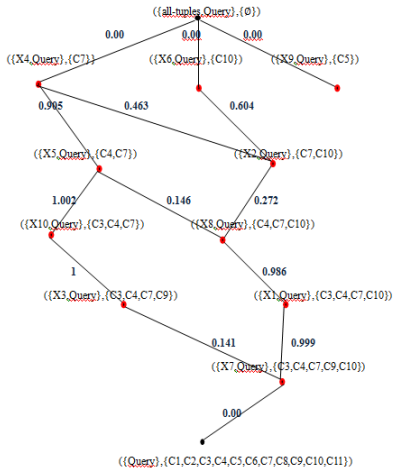

Figure 2. Concept lattice associated to the query

C. Generation of the Top-K answers

Relevant data sources can be sorted according to the distance separating the concepts in the lattice. This step consists of ordering the n-uplets obtained according to their satisfaction degrees of the initial query.

The satisfaction degree corresponds to the similarity of a fuzzy formal concept and its sub-concept defined in Definition 1. In our example, these degrees are given in Table V. To model the expression of the users' preferences,

Table V
SATISFACTION DEGREE OF THE GENERATED ANSWERS

Data sources	Meta data
E1	C3,C4,C7,C10
E2	C7,C10
E3	C3,C4,C7,C9
E4	C7
E5	C4,C7
E6	C10
E7	C3,C4,C7,C9,C10
E8	C4,C7,C10
E9	C5
E10	C3,C4,C7

and generate the top-k answers, we use the Fuzzy stratified divisor, our query is rewritten in this form:

Select	V.idV, V.price, T.means, T.comfort , H.idH, H.name
From	TRAVEL **V**, TRANSPORT **T**, HOTEL **H**
Where	V.idH = H.idH
And	V.idV = T.idT
And	T.means = 'plane'
And	T.route = 'direct'
And	V.offer <> 'circuit'
And	V.typeStay = 'full-board'
having set	(HOTEL)
contains	{Short_Stay, Long_Stay} **and-or** {Low_Price, Medium_Price, High_Price} **and-or** { Medium_Category } **and-or** { Low_Comfort , Medium_Comfort}.

In our example, the scale $L = l_1 > l_2 > l_3 > l_4$ with $l_1 = \{C1, C3\}, l_2 = \{C4, C5, C6\}, l_3 = \{C7\}$ and $l_4 = \{C9, C10\}$. We have in this case four strata. Each line of the table represents a set of answers E1,E2,...,E10. Let S the divisor l1 $\succ$ l2 $\succ$ l3 $\succ$ l4 and the dividend: r = {(E1, {C3,C4,C7,C10}), (E2, {C7,C10}), (E3, {C3,C4,C7,C9}), (E4, {C7}), (E5, {C4,C7}), (E6, {C10}), (E7,{C3,C4,C7,C9,C10}), (E8, {C4,C7,C10}), (E9, {C5}), (E10,{C3,C4,C7})}.

The degree of satisfaction of x is expressed by a vector $V(x)$ of dimension n where V(x)[i] $\in$ $]0 \dots 1]$, if x is associated with all the values of S_i, it represent the Query Memberships degrees of the associate cluster in the query (Table III and Table IV), 0 otherwise.
$V(X1) = (0.331, 0.333, 0.516, 0.351)$
$V(X2) = (0, 0, 0.516, 0.351)$
$V(X3) = (0.331, 0.333, 0.516, 0.334)$
$V(X4) = (0, 0.516, 0, 0)$
$V(X5) = (0, 0.333, 0.516, 0)$
$V(X6) = (0, 0, 0, 0.351)$
$V(X7) = (0.331, 0.333, 0.516, 0.685)$
$V(X8) = (0, 0.333, 0.516, 0.351)$
$V(X9) = (0, 0.333, 0, 0)$
$V(X10) = (0.331, 0.333, 0.516, 0)$

Using Expression 1, Table III, Table IV and Table V, we obtain:

$SAT(E1) = 0.331 * 2^3 + 0.333 * 2^2 + 0.516 * 2^1 + 0.351 * 2^0 = 5.363$
$SAT(E2) = 0 * 2^3 + 0 * 2^2 + 0.516 * 2^1 + 0.351 * 2^0 = 1.383$
$SAT(E3) = 0.331 * 2^3 + 0.333 * 2^2 + 0.516 * 2^1 + 0.334 * 2^0 = 5.346$
$SAT(E4) = 0 * 2^3 + 0 * 2^2 + 0.516 * 2^1 + 0 * 2^0 = 1.032$

 ISBN: 978-1-61208-185-4

$SAT(E5) = 0*2^3+0.333*2^2+0.516*2^1+0*2^0 = 2.346$
$SAT(E6) = 0*2^3+0*2^2+0*2^1+0.351*2^0 = 0.351$
$SAT(E7) = 0.331*2^3+0.333*2^2+0.516*2^1+0.658*2^0 = 5.67$
$SAT(E8) = 0*2^3+0.333*2^2+0.516*2^1+0.351*2^0 = 2.715$
$SAT(E9) = 0*2^3+0.333*2^2+0*2^1+0*2^0 = 1.332$
$SAT(E10) = 0.331*2^3+0.333*2^2+0.516*2^1+0*2^0 = 5.012$
Thus, E7 $\succ$ E1 $\succ$ E3 $\succ$ E10 $\succ$ E8 $\succ$ E5 $\succ$ E2 $\succ$ E9 $\succ$ E4 $\succ$ E6;
Now, suppose that the user wishes to have the 30 Top answers (30 best answers). In that case, the tuples returned are the first 30 tuples starting with all $E7$ and so on.

VI. Experimentations

A. Context

The general principle used to implement the previous queries is based on the use of SQL queries to access data encapsulated and calculate the satisfaction degree (denoted by SAT) assigned to each element of the result. The proposed method has the following characteristics:

1) It is based on the usual way of expressing a division with the counting function and
2) It benefited from the stratification of the divisor to access primarily to user preferences with the highest priority. The strata are traversed in decreasing order of importance (S_1 to S_n), which has a real impact for those requests.

B. Experimental Results

The aim of the experiments is to evaluate the additional cost due to the inclusion of preferences in the division queries. Queries are evaluated with dividend relations of different size (300, 1000, 3000, 5000, 10000 and 15000 tuples), with a divisor composed of four strata (i.e four preferences). The results obtained are reported in the tables below, where:

- Each instance was executed 10 times to avoid the variable load of the machine used,
- The size of the result (K) is 10 (respectively 30, 100) for a dividend of 300 (respectively 1000, 3000, 5000, 10000 and 15000) tuples,

Table VI
The experimental results for the operation of the Division

	Processing time(s)			Used Memory(MB)
Number of tuples	K=10	K=30	K=100	
300	3.236	3.487	4.210	0.168
1000	4.052	4.356	4.923	0.215
3000	4.573	4.789	5.002	0.275
5000	4. 590	4.706	5.403	2.808
10000	4. 661	4.776	5.624	3.636
15000	4.789	5.129	5.942	3.779

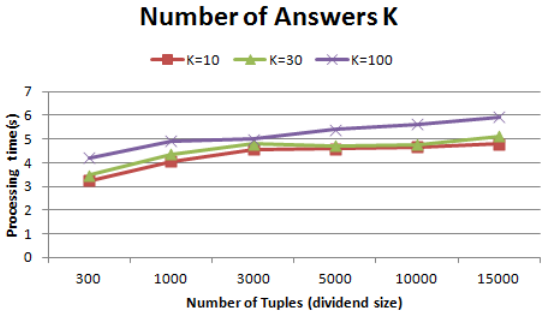

Figure 3. Variation of the processing time according to the variation of K

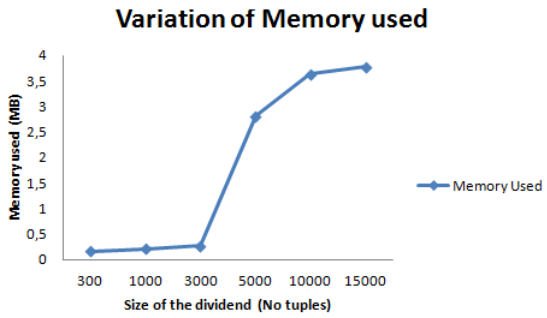

Figure 4. Variation of the memory used for the Division Operation

In Table VI and as shown in both figures (3 and 4), we varied the number of answers requested by the user (Top K answers) to assess its impact on the processing time and the memory used. Thus, we note that the number K does not have a major effect on the processing time since the division operation retrieves tuples (answering the query) that are already listed and highly ranked in a text file.

1) The cost of the division operation depends linearly, as expected, on the size of the dividend,
2) The number of answers (K) required by the user does not have a major influence on the execution time,

These first results are quite encouraging, even if they must be completed to achieve more definitive conclusions about the best way to develop tools over an existing DBMS or otherwise intervening in the kernel of the DBMS to run these queries.

VII. Study of Complexity

A study of spacial and temporal complexities of the proposed approach is presented in this section.

- Space complexity: In the field of space complexity, we store only XML files. The clusters of the relaxable

 ISBN: 978-1-61208-185-4

attributes are not stored any more in the KBAR(Knowledge Base of Relaxable Attributes). What constitutes an asset for the practical application of this approach.

- Temporal complexity: It includes the following costs: a) construction of the clusters of the relaxable attributes, b) construction of lattice (with the algorithm of Ganter) and c) scheduling of the approximate answers.

For the construction of the clusters of the relaxable attributes , we calculated the theoretical complexity of the approaches of clustering suggested. It is equal to $O(Nc^2)$, where N corresponds to the number of data and c is the maximum number of clusters. For the construction of the lattice, temporal complexity depends on the method of adopted construction. In our approach, we were interested in the method of Ganter [30] where its complexity is $O((max(|N|, |n|)).(|N|.|n|))$.

Thus, the total complexity is equal to $O(Nc^2) + O((max(|N|, |M|)).(|N|.|M|)) + O(n * level)$, where N corresponds to the number of data (the objects of the DB), M is the number of attributes, c is the number of clusters, n is the number of concepts from the lattice and $level$ corresponds to the number of levels present in the lattice.
We present in Table VII a study of the complexity of some algorithms of construction of the lattices.

Table VII
STUDY OF TEMPORAL COMPLEXITY OF LATTICE CONSTRUCTION ALGORITHMS

Algorithm	**Temporal Complexity**
Bordat [31]	$O(n.\|N\|.(\|N\| + \|M\|))$, n is the number of concepts
Nourine et Raynaud [32]	$O(n.\|N\|.(\|N\| + \|M\|))$, n is the number of concepts
Ganter [30]	$O((max(\|N\|, \|M\|)).(\|N\|.\|M\|))$
Godin [33]	Quadratic compared to the number of elements in the lattice of concepts.

VIII. COMPARISON WITH OTHER APPROACHES

In this section, we present the essential idea of the principal existing flexible querying approaches closest to our approach. Those differ primarily by the manner used to find the values closest to those required by the user and the used formalism to model the uncertainty and the imperfection of the real world.

The literature on the flexible querying and the co-operative systems abounds. We can distinguish three principal categories. The first category, indicated by C1, includes ”ad hoc” approaches specific to particular systems. The objective of such approaches is the introduction of flexibility by the use of linguistic terms and the specification of the preferences of the users between the various search keys from the desired data. Among the approaches of C1, we can quote the systems ARES [9], MULTOS [11], SEAVE [34], FLEX [35]. Second category approaches indicated by C2, use the formalism of sets and fuzzy logic to model in addition to the imperfection and the uncertainty of the real world, the evaluation of the query known as vague or fuzzy.

The principal common point between these approaches is the modification of the query language, generally. This modification consists in introducing vague linguistic terms, like ”accessible price” or ”large budget”, and of the operators of approached comparison like ”Near-to” and ”similar-to” of the system CoBase [4]. To not modify the DBMS system, these systems add an additional layer charged to transform a fuzzy query into a traditional one known as ”wraps query”. This one is subjected to the target DBMS for evaluation. Its result is then filtered according to preferences of the user before being presented to him. This process of transformation and filtering is based on established properties of the sets and fuzzy logic.

The third category, indicated by C3 comprises approaches which lie within the scope of the artificial intelligence techniques and aims at determining tacit knowledge starting from the explicit data. Several systems like DBLEARN [36], DB-Discover [37] and GBDR [38], belong to this category. Generated knowledge is in the form of rules or of hierarchy of concepts.
The results obtained by these approaches, in particular within clustering, are of a great utility for this work.

The contributions of approaches of C2, such as for example those of CoBase [4], are significant, in particular the concepts of TAH and MTAH to model generalization and specialization by hierarchies of concepts. However, we estimate these systems remain demanding with respect to the end-users. For example, in CoBase, the operators used require a precise knowledge of the contents of the database,. It does not detect the realizability of a query only after its execution. CoBase can also generate false answers. The users must also know the organization of the database since they must specify the attributes which they must release or not as well as the level of relieving of each attribute.

In [2], no modification of SQL is necessary, which constitutes an asset for the practical application of this approach. The user is not solicited to make choices during relieving, which can be hazardous, as it is the case in several systems such as Flex, Vague [10] and CoBase, to

 ISBN: 978-1-61208-185-4

quote only those.

In this approach, the relieving attributes are fixed by the administrator of the database. This is the more significant, since the approach suggested is addressed to end-users not having the knowledge precise and detailed on the organization and the data which they consult. It is easier to an expert to specify that a price attribute of a table of the database is relaxable and than it can be used with the terms "weak" or "accessible". This is easier than to use the operator "Within" $(100, 120, 150, 300)$ of CoBase. However, this approach present limits at the level of the structures which it uses. We quote:

a. The incremental maintenance of the base of knowledge of the relaxable attributes (KBAR),
b. The clustering of the relaxable attributes without fixing a priori the number of clusters; and
c. The problem of storage of the clusters and indexing of the MTAH.

In the proposed approach, the clusters generated for each relaxable attribute are not stored any more in the catalogue of the DBMS. So, the maintenance of this meta-base does not pose any problem. Indeed, to be able to trace the lattices, it is quite simply necessary to charge an XML file which makes it possible to recover all information necessary to the tracing of these lattices. XML parsers recover information and recall the lattice starting from the methods of constructions of these structures. In this file are backed up:

- The title of the lattice.
- Identifiers of the concepts, their positions with the styles of the labels of the objects and attributes of the concept.
- The set of data and attributes of each concept.
- The set of the arcs and the concepts which they bind.

This parser also allows curing the problem of storage of the clusters and indexing of the MTAH.

The problem of clustering does not arise with this approach since the approaches of clustering suggested allow, in addition to the optimization of the number of clusters, the evaluation of the quality of the latter. Finally, at the level of the requests addressed to large databases, the current top-k query processing techniques focus on Boolean queries, and cannot be applied to the large DB seen the gigantic number of data. The majority of these systems uses an ad-hoc score function f and delivers the k better answers of the total order obtained by f. However, this score function remains difficult to establish seen the voluminous number of data.

IX. Conclusion

Several algorithms of the top-k retrieval problem have been introduced in recent years. Unfortunately, these techniques cannot be applied to the large DB seen the gigantic number of data. The majority of these systems uses a score function which remains difficult to establish seen the voluminous number of data.

In this paper, we have proposed a new approach for top-k flexible queries taking into account another degree of granularity in the process of the evaluation of the query. The proposed approach consists of the following steps: 1) Generation the meta-DB, for this we apply a fuzzy algorithm of clustering. Each cluster represents a set of data verifying the same properties; 2) Deduction of the semantic of the data, we represent the matrix obtained in the first step under the form of fuzzy concept lattice (FCA); and 3) Generation of the k better answers while proposing a new operator called stratified operator for taking into account the user's preferences.

The contributions of this approach are (1) the extraction of the dependencies between the search criteria to detect the unrealizable query; (2) the optimal search of the relevant data sources for a given query; (3) the automatic generation of the k better answers. This work can be spread while proposing to adapt our method to the large fuzzy DB.

References

[1] R. Fagin and E. Wimmers, *Incorporating User Preferences in Multimedia Queries*,ICDT '97 Proceedings of the 6th International Conference on Database Theory Delphi, Greece, pp. 247-261, January 1997.

[2] H. Ounalli and R. Belhadj, *Interrogation flexible et coopérative d'une BD par abstraction conceptuelle hiérarchique*, INFORSID 2004, pp. 41-56.

[3] W.W. Chu, K. Chiang, C. Hsu, and H. Yau, *An Error-based Conceptual Clustering Method for Providing Approximate Query Answers*, Communications of ACM, 1996.

[4] W.W. Chu, H. Yang, K. Chiang, M. Minock, G. Chow, and C. Larson, *CoBase: A Scalable and Extensible Cooperative Information System*, Journal of Intelligence Information Systems, Boston, 1996.

[5] K. Uri and Z. Jianjun, *Fuzzy Clustering Principles, Methods and Examples*, Technical Report ,Technical University of Denmark,. Department of Control and Engineering Design (IKS), 13p, December 1998.

[6] R. Wille, *Lattices in data analysis: how to draw them with a computer*, In: I.Rival (ed.): Algorithms and order, Kluwer, Dordrecht-Boston, pp. 33-58, 1989.

[7] M. Lacroix and P. Lavency, *Preferences : Putting more knowledge into queries*.In Proceedings of the 13th International Conference on Very Large Data Bases, Morgan Kaufmann Publishers Inc., pp. 217-225, 1987.

[8] CL. Chang, *Decision support in an imperfect world*. Trends and applications on automating intelligent behavior: applications and frontiers, 25, 1983.

 ISBN: 978-1-61208-185-4

[9] T. Ichikawa and M. Hirakawa, *ARES: A relational database with the capability of performing flexible interpretation of queries*. IEEE Transactions on Software Engineering, 12(5), pp. 624-634, 1986.

[10] A. Motro, *VAGUE : A User Interface to Relational Databases that Permits Vague Queries*. ACM Transactions on Office Information Systems, 6(3), pp. 187-214, 1988.

[11] F. Rabitti and P. Savino, *Retrieval of multimedia documents by imprecise query specification*. Advances in Database Technology-EDBT'90, pp. 203-218, 1990.

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

[13] V. Tahani, *A conceptual framework for fuzzy query processing-A step toward very intelligent database systems*. Information Processing and Management, 13(5), pp. 289-303, 1977.

[14] O. Pivert, *Contribution à l'interrogation flexible de bases de données : expression et évaluation de requêtes floues*. PhD thesis, Université de Rennes 1, 1991.

[15] P. Bosc, L. Liétard, and O. Pivert, *Databases and flexibility : gradual queries*. TSI. Technique et science informatiques, 17(3), pp. 355-378, 1998.

[16] P. Bosc and L. Liétard, *Aggregates computed over fuzzy sets and their integration into sqlf*. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 16(6), pp. 761-792, 2008.

[17] J. Chomicki, *Preference formulas in relational queries*. ACM Trans. Database Syst., 28(4), pp. 427-466, 2003.

[18] W. Kießling, *Foundations of preferences in database systems*. In Very Large Data Base (VLDB) Endowment Inc, pp. 311-322, 2002.

[19] L. Liétard, D. Rocacher, and S.E. Tbahriti, *Preferences and bipolarity in query languages*. Fuzzy Information Processing Society, New York City, NY, pp. 1-6, 2008.

[20] L. Liétard and D. Rocacher, *On the definition of extended norms and co-norms to aggregate fuzzy bipolar conditions*. In the European Society of Fuzzy Logic and Technology Conference, pp. 513-518, 2009.

[21] S. Börzsönyi, D. Kossmann, and K. Stocker, *The skyline operator*. In International Conference on Data Engineering (ICDE), pp. 421-430, 2001.

[22] W. Kießling and G. Köstler, *Preference sql - design, implementation, experiences*. In Very Large Data Base (VLDB) Endowment Inc, pp. 990-1001, 2002.

[23] D. Dubois and H. Prade, *Bipolarity in Flexible Querying*. Proceedings of the 5th International Conference on Flexible Query Answering Systems, FQAS '02, London, UK, pp. 174–182, 2002.

[24] C. Domshlak, H. Hoos, C. Boutilier, R. Brafman, and D. Poole, *Cp-nets : A tool for representing and reasoning with conditional ceteris paribus preference statements*. Journal of Artificial Intelligence Research, 21 , pp. 135-191, 2004.

[25] P. Bosc and O. Pivert, *A propos de requêtes à préférences et diviseur stratifié*, In INFORSID, pp. 311-326, 2010.

[26] C. Li, MA. Soliman, CK. Chang, and IF. Ilyas, *Ranksql : Supporting ranking queries in relational database management systems*. In Very Large Data Base (VLDB) Endowment Inc, pp. 1342-1345, 2005.

[27] T. Thanh, H.S. Cheung, and C.Tru Hoang, *A Fuzzy FCA-based Approach to Conceptual Clustering for Automatic Generation of Concept Hierarchy on Uncertainty Data*. Proceedings of the CLA 2004 International Workshop on Concept Lattices and their Applications, Ostrava, Czech Republic, pp. 1-12, 2004.

[28] A. Grissa Touzi, M. Sassi, and H. Ounelli, *An innovative contribution to flexible query through the fusion of conceptual clustering, fuzzy logic, and formal concept analysis*; International Journal of Computers and Their Applications. Vol. 16, N 4, pp. 220-233, December 2009.

[29] M. Sassi, A. Grissa Touzi, and H. Ounelli, *Clustering Quality Evaluation based on Fuzzy FCA*, 18th International Conference on Database and Expert Systems Applications, (DEXA'07), Regensburg, Germany, pp. 62-72, LNCS, Springer 2007.

[30] B. Ganter, *Two Basic Algorithms in Concept Analysis*. Technical report, Darmstadt, 1984.

[31] J. Bordat, *Calcul pratique du treillis de Galois dune correspondance* , Mathmatique, Informatique et Sciences Humaines, pp. 31-47, 1986.

[32] L. Nourine and O. Raynaud, *A fast algorithm for building lattice*, Information Processing Letters, vol. n 71, pp. 199-204, 1999.

[33] R. Godin and R. Missaoui, *An incremental concept formation approach for learning from databases*, Theoretical Computer Science, vol. n 133, pp. 387-419, 1994.

[34] A. Motro, *SEAVE: A Mechanism for verifying User Presuppositions in Query Systems*, ACM Transactions on Information Systems, pp. 312-330 ,1986.

[35] A. Motro, *FLEX: Tolerant and Cooperative User Interface to Database*, IEEE Transactions on Knowledge and Data Engineering, 4(4), pp. 231-246 , 1990.

[36] H.J. Hamilton and D.F. Fudger, *Estimating DBLEARNs Potential for knowledge Discovery in Databases*, Computational Intelligence, 11(2), 1995.

[37] C.B. Rivera and C.L. Carter, *A tutorial Guide to DB-Discovrer*, Version 2.0, Technical Report CS-95-05, University of Regina, pp. 280-296, 1995.

[38] C.L. Carter, H.J. Hamilton, W.B. Hase, and C. Rivera, *GDBR: An Optimal Relation Generalization Algorithm for Knowledge Discovery from Databases*, Department of Computer Science, University of Regina, 1998.

 ISBN: 978-1-61208-185-4