OLAP query suggestion and discovery driven analysis

Patrick Marcel

Université François Rabelais de Tours, LI campus de Blois, France

Abstract. Interactive analysis of datacube, in which a user navigates a cube with a sequence of queries to find and understand unexpected data, is often tedious. To better support this process, we propose in this paper to connect two techniques proposed earlier in this domain. These techniques are, on the one hand, discovery driven analysis, that guides the user towards regions of the cube they will find of interest, and on the other hand, query recommendation, that takes advantage of what the other users did during former analyses. Benefiting from these techniques we propose a framework for recommending OLAP queries to the user by taking into account what previous users found of interest and the explanation they worked out.

1 Introduction

Context Interactive analysis of a data cube has often be described as a tedious process whereby users interactively navigate a cube by launching a sequence of queries over a datawarehouse, what we call an analysis session (or session for short) in the following.

In a typical session, one of the queries detects something surprising and then the subsequent queries are used to navigate the cube to explain what has been detected. But designing these subsequent queries is often difficult since the user may have no idea of what part of the cube he should navigate.

To cope with this, Sarawagi et al. proposed in Sarawagi et al. (1998) and subsequent work new operators to guide the user towards unexpected data in the cube or to propose to explain an unexpected result. We remark that these operators are applied only on query results and they do not take into account what other users might have discovered.

On the other hand, in Giacometti et al. (2008) it is proposed to consider what other users did during former sessions on the cube to suggest (or recommend) a query to the user. But in this work no emphasis is put on what former users may have found interesting.

Problem In this present paper, we propose to bridge the gap between these two domains, that is to answer the question: How to suggest to the user queries that lead him to interesting parts of the cube, by taking advantage of what other users found interesting?

Contribution To answer this question, we propose a framework for recommending OLAP queries that is based on the following principle:

- Every analytical session on a datacube is logged and each of these sessions is associated with a *context*, which represents the problem, that is the unexpected data, the session investigates, and with a *goal*, which represents the outcome of the session. This is done especially by an offline preprocessing of the log that uses some of the operators proposed in the area of discovery driven analysis of datacubes.

- A context is assigned to the current session (that is to the user for whom the system has to recommend a query) that allows to find in the log the sessions that tried to investigate a problem of a similar nature.

- If there exists in the log a session with a context similar to the current session, then the outcome of this logged session is proposed as a recommendation. Otherwise, it is proposed to the user to see the context of a logged session that is similar to his session in terms of the sequence of queries composing it.

This framework can be seen as an extension of our previous work on OLAP query recommendation Giacometti et al. (2008) Note that this paper is a position paper in the sense that its goal is to propose and discuss an original idea for enhancing interactive analysis of datacube. Further research and experimentation need to be conducted in order to validate the idea. This is briefly discussed in the conclusion.

Outline A short motivating example is given in the next section. Section 3 reviews the works underlying the framework we propose, namely discovery driven analysis of datacube and OLAP query suggestion, and discusses some related work done in information retrieval that we have used to connect discovery driven analysis with OLAP query recommendation. Preliminary definitions are given in Section 4 and the contribution is detailed in Section 5. Section 6 concludes the paper and exposes future work.

2 Motivating example

In this section we illustrate with a simple example the basic idea under our framework. It is assumed that the reader is familiar with the MDX query language (Microsoft Corporation (2008)).

Consider an OLAP server used by several users. Each user can open a session on the server to navigate the cube by launching a sequence of queries. The server logs these sessions, i.e., logs the sequences of queries launched during each session.

Imagine now a new session, called the *current session*, is opened by a user, called the *current user*. For instance, the user is analyzing the sales of wines in various countries. At some point, the user is issuing the following query to find the sales of Bordeaux and Bourgognes in US:

```
SELECT   {[French wines].[Bordeaux].Children,[French wines].[Bourgognes].Children}
         ON ROWS,
         [US].[All Members] ON COLUMNS
FROM     [SalesCube]
WHERE    [Measures].[Sales]
```

Based on the result of this query, the system can search the log to find among the sessions similar to the current session the queries whose result reveals something unexpected w.r.t. what the user is currently observing. One of these queries can then be recommended to the user.

For instance, suppose the systems finds the sessions of some other users who analyzed earlier the sales of wines. It suggests the following query, that is present in one of these sessions, that displays the sales of french wines in US in 2004 and 2005:

```
SELECT   [French Wines].[All Members] ON ROWS,
         crossjoin([US].[All Members],{[2004],[2005]}) ON COLUMNS
FROM     [SalesCube]
WHERE    ([Measures].[Sales])
```

The system suggests this query because it displays a significant drop of sale results from 2004 to 2005.

Now suppose that the user is observing something surprising in the result of the current query (this may be so because he has chosen to evaluate the query recommended in the case described above). The system can search in the log for sessions having tried to explain something similar. Among these sessions, the system can look for the query that explains this surprising result the best and recommend it to the user.

Continuing with our running example, suppose that the current user has evaluated the query recommended by the systems and is observing this drop of sale results from 2004 to 2005. Then the system suggests to the user the following query, found in one of the sessions that analyzed the sales of wines. This query displays the unit sold of white Bordeaux for the fourth quarters of both 2005 and 2004 in the south of the US:

```
SELECT   {[2005].[Q4],[2004].[Q4] } ON ROWS,
         [US].[South].Children ON COLUMNS
FROM     [Sales]
WHERE    ([Measures].[Unit sold],[French Wines].[Bordeaux].[White wines])
```

The systems suggests this query because it displays that very few white Bordeaux were sold in the fourth quarter of 2005 in the southern states. This can explain the drop of sale results the user observed earlier.

3 Related work

In this section we present the salient features of the works that inspired the framework described in the next section.

3.1 Recommending or anticipating OLAP query

To the best of our knowledge, Giacometti et al. (2008) is the only work that proposes a framework and a system for recommending OLAP queries to a user by taking advantage of former analytical sessions. The framework relies on the following principle, illustrated in Figure 1:

1. A preprocessing step: The query log of the OLAP server is preprocessed in order to cope with sparsity. As, for most of the queries in this log it is unlikely that this query is asked many times, queries that are close to one another are grouped. This preprocessing is done offline.

2. A matching step: The query log is examined in order to find a set of sessions that match (i.e., that are closest to) the current session.

3. A prediction step: The set of sessions matching the current session is used to suggest what the forthcoming query of the current session could be. A set of candidate queries is obtained.

4. A ranking step: The candidate queries are ranked so as to present to the user the most relevant queries first.

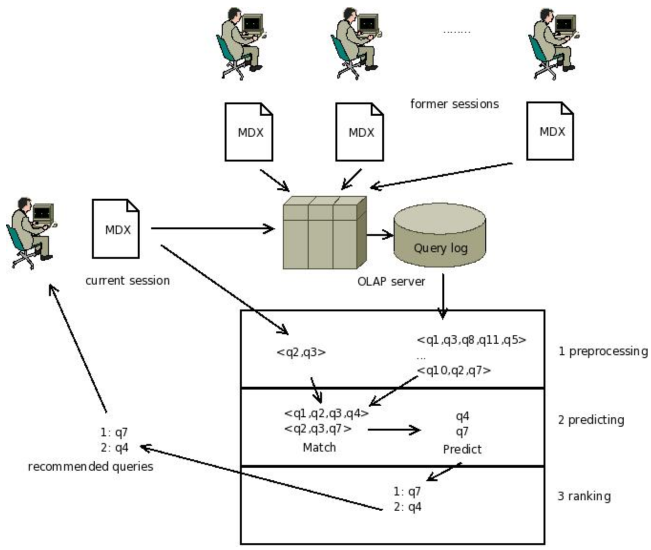

FIG. 1 – *Overview of the generic framework*

This framework is generic in the sense that each step can be parametrized by a specific function to preprocess, match, predict or rank. An instantiation of this framework is given in Giacometti et al. (2008) that has been used to implement a prototypical OLAP query recommender system. A drawback of this system is that it does not take into account what former users might have found of interest, which can be either unexpected data or the explanation of some unexpected data.

In what follows, we will use this generic framework and propose a particular instantiation where this is taken into account, both for session matching and query prediction.

Note that the work of Sapia (1999, 2000) shares with our work the goal of predicting the forthcoming OLAP query. However the main concern of this work is to prefetch data, not to guide the user towards interesting parts of the cube.

3.2 Discovery driven analysis of OLAP data

Sarawagi et al. introduced discovery driven analysis of OLAP cube in Sarawagi et al. (1998). This and subsequent works resulted in the definition of various OLAP operators to support interactive exploration of data cubes. We present some of these operators next.

The DIFF operator The DIFF operator proposed in Sarawagi (1999) explores the reasons why an aggregate is lower (or higher) in one cell compared to another. It takes as parameter two cells v_a and v_b, and looks into the two isomorphic subcubes C_a and C_b that detail the two cells (i.e., that are aggregated to form the observed v_a and v_b). As a result, it summarizes the differences in these two subcubes by providing the top N informative cells from the unvisited part of the cube.

In Sathe and Sarawagi (2001) a RELAX operator has been proposed that can be thought of as opposite of the DIFF operator, in that RELAX summarizes exceptions by rolling-up whereas DIFF summarizes differences by drilling down.

The INFORM operator In Sarawagi (2000), an INFORM operator is used to find parts of the cube a user will find most surprising based on what the user already knows about the data. The idea in that paper was to adapt the Maximum Entropy Principle to OLAP. The user's expectation is computed by finding a model of what the user has not visited yet that is consistent with all the facts already known (i.e., the cells viewed earlier in the session) and otherwise as uniform as possible. Then the differences between the expected result and the actual result can guide the user towards problematic cases hidden in detailed data.

Note that both DIFF and INFORM are slightly different than the other classical OLAP operators, in the sense that they do not produce a cube nor a cross-tab as a result. Instead, they provide what can be thought of as a list of cells. In what follows we consider that the signature of these operators are: DIFF is applied on a set of cells and on two particular cells of this set. INFORM is applied on a set of cells. Both operators output a set of cells.

Two recent works use a data mining approach to inform the user of potentially interesting regions of a cube by either automaticaly detecting interesting cells Cariou et al. (2007) or proposing interesting drill paths Cariou et al. (2008). In the former case, the goal is simply to highlight in a given query result the cells whose measure deviates the most from a theoretical value computed under independence model hypothesis. In the latter case however, the goal is very similar to that of Sarawagi (2000) and can be seen as recommending drill down queries to the user. This approach does not take into account former explorations and thus is a promising candidate to be used instead of the INFORM operator in the framework we propose in Section 5.

It is to be noted that discovery driven analysis has been adapted to document search, like in e.g., Dash et al. (2008). In this work, facets (i.e., attributes) instances (i.e., attribute values) are associated with documents and documents are searched with queries using both keywords and facet instances. A model of the distribution of facets in the documents is computed and this expectation is compared to the actual search result. If the search result is large, the system presents to the user the part of the result that is found the most surprising w.r.t. the expectation.

3.3 Session properties used in Information Retrieval

The idea of using former sessions to improve current search is very popular in Information Retrieval (Adomavicius and Tuzhilin (2005)) and Web Usage Mining (Spiliopoulou et al. (2000)).

In recent works, properties of the session can be inferred to support subsequent searches. For instance, in Downey et al. (2008), the *information goal* of a session is defined as the last URL visited during the session or alternatively the last click on a search engine result page.

In Parikh and Sundaresan (2008) in the domain of E-commerce, the session goal is a particular event occurring in the session. In this case of the Ebay site, the goal of a session is a buy event. This allows to enrich all the sessions (and especially the queries of the sessions) with the description of the item bought, which is called the *context* of the session. The authors show how defining the context of a session helps recovering from null result in subsequent searches, provides a better understanding of the queries in the session, or helps generating recommendations.

Note that there is a recent interest for trying to combine information retrieval and OLAP. For instance, in Wu et al. (2007) the authors propose to query a datacube with only a set of keywords. Among the potential answers to the query, only the subcubes that are the most surprising are presented to the user.

To the best of our knowledge, no work has been done on how to compute properties of former OLAP session and take advantage of this properties to improve the current analysis.

3.4 Problem statement

This paper proposes to connect the ideas stemming from these different areas. More precisely, we will try to answer the following questions:

- Can we identify the goal or context of an OLAP session?
- Can we use this information to generate recommendations?

4 Basic definitions

In this section we give the basic definitions and notations underlying our framework.

Cubes, dimensions, members, cells, cell references An N-dimensional cube C is defined as the classical $N+1$ relation instances of a star schema, one relation instance for each of the N dimensions and one relation instance for the fact table.

Given a particular dimension table, a member is a value in this table. Given a fact table, a cell is a tuple of this table.

Given an N-dimensional cube C, a cell reference (or reference for short) is an N-tuple $\langle r_1, \ldots, r_N \rangle$ where r_i is a member of the ith dimension, for all $i \in [1, N]$.

Query In this paper we consider as in Giacometti et al. (2008) that a query is a set of references. We denote by $res(q)$ the set of cells being the result of the query q and if Q is a set of queries, then $res(Q)$ denotes the union of the $res(q)$ such that $q \in Q$.

Query similarity In what follows, $sim(q, q')$ is any function computing a similarity score for q and q'. We do not propose a specific algorithm for computing such a similarity here. The reader is redirected to Giacometti et al. (2008) or Negre (2009) for two examples of functions computing a distance between MDX queries that can be used to obtain a similarity score between OLAP queries. In both of these works, as queries are considered to be sets of cell references, the distance between two queries is computed by using the Hausdorff distance between sets Hausdorff (1914). This distance allows to compare two sets based on a distance between the elements of the sets. Informally, two sets are closed if every element of either set is closed to some element of the other set. In Negre (2009), it is proposed that the distance between elements is a distance between cell references that relies on a distance between members. The distance between two members in a hierarchy is the length of the shortest path between these members in the hiearchy.

OLAP session and log An OLAP session is a sequence of queries. We note $q \in s$ the fact that a query q appears in a session s. An OLAP query log, or log for short, is a set of OLAP sessions. We note $q \in L$ the fact that a query q appears in some session of a log L.

OLAP session context and OLAP session goal The context of a session is an OLAP query that appears in the session. If s is a session, we denote its context by $cxt(s)$.

The goal of a session is an OLAP query that appear in the session. The goal of a session must be different from its context. If s is a session, we denote its goal by $goal(s)$.

5 Framework

We now present the framework for recommending OLAP query using discovery driven analysis operators. We start by giving the principle and then detail every step.

5.1 Principle

The basic idea behind our framework is to infer two properties of each session. These two properties are:

- A context: The context of a session is the query that leads to the *most surprising* data when *compared to the other queries of the session*.
- A goal: Given a session and its context, the goal of the session is the query that leads to the *best explanation of the context*.

These properties are used for recommending queries in the following way: If the current session has a context similar to that of some sessions in the log, then the recommended query is chosen among the goals of these sessions. If there is no session in the log that shows a context similar to the current session, then a query of the log which result is surprising w.r.t. the former queries in the current session is proposed.

5.2 Preprocessing

We now explain how a context and a goal are associated with each session.

Preprocessing the log The log are preprocessed in order to associate with each session a context and a goal. Note that this preprocessing is done offline (whereas the computation of recommendation is done on-line).

First, candidate contexts are detected. Every query q of the log is associated with a score $unexpected(q)$ computed as follows:

1. for each query q of a session s
2. let Q be the set of queries of s preceding q
3. compute the set $C = INFORM(res(Q))$
4. $unexpected(q)$ is the number of cell references of C in $res(q)$

Then, candidate goals are detected. Every pair (q, q') of queries of the session is associated with a score $explains(q', q)$ computed as follows:

1. for each query q of a session s
2. for each pair of cells (c, c') of q such that the difference exceeds a given threshold
3. compute the sets $C_{c,c'} = DIFF(res(q), c, c')$
4. for each successors q' of q in s
5. $explains(q', q)$ is the maximal number of cell references of the $C_{c,c'}$ in $res(q')$

All queries can be ranked according to how unexpected their result is w.r.t. the former queries. The context is then the most unexpected query q for which there exists a query q' in the session that explains it (i.e., such that there is a non null $explains(q', q)$ score). The goal being then the query that maximizes this score. Formally, for a session s,

- $cxt(s) = argmax(\{unexpected(q) | q \in s \wedge \exists q' \in s, explains(q', q) \neq 0\})$ and
- $goal(s) = argmax_1(\{explains(q', cxt(s)) | q' \in s\})$.

Obviously the scores can be 0. In that case:

- If for all queries q, $unexpected(q) = 0$ but there exists a pair (q, q') such that $explains(q, q')! = 0$, then we use the pair (q, q') that maximizes that score for goal and context respectively.
- If there exists a query q such that $unexpected(q)! = 0$ but for all pairs (q, q'), $explains(q, q') = 0$ then the context is the query q maximizing $unexpected(q)$ and the goal is a query artificially constructed from $INFORM(res(q))$. For instance, it can be the query which result is the most surprising cell of $INFORM(res(q))$.
- If all scores are null, then the session cannot be used with the matching and recommending functions described below. Thus it is simply not taken into consideration.

Preprocessing the current session A context is assigned to the current session by using the same principle as above. As long as the session grows in term of queries, the context is updated.

5.3 Session matching

Two sessions can be considered similar if they are similar sequences of queries. In Giacometti et al. (2008) we use a technique stemming from approximate string matching (see e.g., Navarro (2001)) to compute a similarity score between two sessions. Given two sequences s and s', Approximate String Matching is the problem of matching the sequences allowing errors. The matching relies on the computation of a distance between the sequences, which is the minimal cost of the sequences of operations transforming s into s'. The classical Levenshtein (or edit) distance Levenshtein (1966) is commonly used, that can be thought of as the minimal number of insertions, deletions or substitutions to make the two sequences equal. Let's call this score $s_{seq}(s, s')$ for two sessions s and s'.

Now, even though two sessions s and s' are not similar in the sense that $s_{seq}(s, s')$ does not exceed a given threshold, s and s' can have a similar context, in the sense that in both s and s' the user tries to explain a particular surprising observation.

Thus we can compute another similarity score s_{cxt} for the two sessions s and s' in the following way: $s_{cxt}(s, s') = sim(cxt(s), cxt(s'))$ where sim is the function computing the similarity of two queries.

5.4 Query suggestion

We now explain how recommendations are computed.

Operators for query suggestion We introduce two operators for recommending a query. Let L be a set of sessions, s be a session and q be a query,

- $understand(q, s, L) = q' \in L$ such that q' explains q the best
- $surprise(q, s, L) = q' \in L$ such that q' shows the most unexpected result w.r.t. q

Note that $understand$ is not DIFF and $surprise$ is not INFORM since $understand$ and $surprise$ produce a query as output, and this output has to belong to a set of existing queries.

Recommendation Given a preprocessed log L and a session s with the current query q, recommendations are computed with the following algorithm:

if $unexpected(q)$ exceeds a given threshold then

recommend $understand(q, s, L)$

otherwise

recommend $surprise(q, s, L)$

Implementing the operators There are many ways of implementing the two operators *understand* and *surprise*. We propose here a simple implementation. Let L be a log, s be the current session and q be the current query:

- For $understand(q, s, L)$: Let S be the set of sessions s' from L such that $s_{cxt}(s, s')$ exceeds a given threshold. Let G be the set of goals of the sessions in S, i.e., $G = \{goal(s') | s' \in S\}$.

  Then $understand(q, s, L) = argmax_1(\{explains(q', q) | q' \in G\})$.

- For $surprise(q, s, L)$: Let S be the set of sessions s' from L such that $s_{seq}(s, s')$ exceeds a given threshold. Let C be the set of contexts of the sessions in S, i.e., $C = \{cxt(s') | s' \in S\}$.

  Then $surprise(q, s, L) = argmax_1(\{sim(q', q) | q' \in C\})$.

5.5 Presenting the suggested queries to the user

We now explain why and how the recommender system presents more than one recommended queries to the user.

Advantage of the approach The main advantage of using what other users did to improve the current analysis is that the current user is not tied to a specific set of operations for issuing his forthcoming queries.

For instance, suppose the user is observing something unexpected in the answer of his current query. For his next query, he may use the DIFF operator to find an explanation for what he is observing. In that case, he gets one and only one explanation, that is the output of the DIFF operator. Alternatively, the user may choose to issue the query recommended by the system, that is a query whose result another user have obtained to explain the interesting observation. If the current user is not happy with the result of this recommended query, the system can then suggest another query that has been computed using the same process.

Ranking recommendations Suppose that the output of the *understand* or *surprise* operators consists of more than one query. Then, instead of proposing only one recommendation to the user, we can rank these queries so that they all can be proposed in a given order. Ranking functions are proposed in Giacometti et al. (2008) and Negre (2009). In Giacometti et al. (2008), recommendations are ranked according to how similar they are to the current queries. In that case again the Hausdorff distance is used to compute the similarity. In Negre (2009), the ranking is computed w.r.t. a user profile by using the ordering on MDX queries proposed in Bellatreche et al. (2005).

6 Conclusion

In this paper we propose a framework for recommending OLAP queries to the user by taking into account what other users found of interest (the context of their session) and the explanation they worked out (the goal of their session). This framework is designed by adapting the framework proposed in Giacometti et al. (2008) and incorporating the operators proposed in the domain of discovery driven analysis of datacube in Sarawagi (1999) and Sarawagi (2000).

Our future work include:

- The implementation of the proposed framework by adapting the prototype used in Giacometti et al. (2008). This implementation will allow to validate the approach in the two following ways:
    1. Firstly by conducting experimentation questionning the impact of the various thresholds needed as parameters of our algorithms (cf. Section 5).
    2. Secondly by using the prototype on real data and obtaining feedback from OLAP users. In that sense, we are currently connecting our prototype to the Mondrian OLAP server (Pentaho Corporation (2009)) for suggesting MDX queries (Microsoft Corporation (2008)).
- The investigation of the importance of the path followed by the user during a session. More precisely, we would like to answer the following questions: What would be the impact of having more than one context or goal for a session? Are the intermediate results found in a session before the goal relevant for computing recommendations?

References

Adomavicius, G. and A. Tuzhilin (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng. 17*(6), 734–749.

Bellatreche, L., A. Giacometti, P. Marcel, H. Mouloudi, and D. Laurent (2005). A personalization framework for olap queries. In *DOLAP*, pp. 9–18.

Cariou, V., J. Cubillé, C. Derquenne, S. Goutier, F. Guisnel, and H. Klajnmic (2007). Built-in indicators to automatically detect interesting cells in a cube. In *DaWaK*, pp. 123–134.

Cariou, V., J. Cubillé, C. Derquenne, S. Goutier, F. Guisnel, and H. Klajnmic (2008). Built-in indicators to discover interesting drill paths in a cube. In *DaWaK*, pp. 33–44.

Dash, D., J. Rao, N. Megiddo, A. Ailamaki, and G. M. Lohman (2008). Dynamic faceted search for discovery-driven analysis. In *CIKM*, pp. 3–12.

Downey, D., S. T. Dumais, D. J. Liebling, and E. Horvitz (2008). Understanding the relationship between searchers' queries and information goals. In *CIKM*, pp. 449–458.

Giacometti, A., P. Marcel, and E. Negre (2008). A framework for recommending OLAP queries. In *DOLAP*, pp. 73–80.

Hausdorff, F. (1914). *Grundzüge der Mengenlehre*. Von Veit.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady 10*(8), 707–710.

Microsoft Corporation (2008). Multidimensional expressions (MDX) reference. Available at http://msdn.microsoft.com/en-us/library/ms145506.aspx.

Navarro, G. (2001). A guided tour to approximate string matching. *ACM Comput. Surv. 33*(1), 31–88.

Negre, E. (2009). Recommandations personnalisées de requêtes MDX. In *Cinquième journées francophones sur les Entrepôts de Données et l'Analyse en ligne*.

Parikh, N. and N. Sundaresan (2008). Inferring semantic query relations from collective user behavior. In *CIKM*, pp. 349–358.

Pentaho Corporation (2009). Mondrian open source OLAP engine. Available at http://mondrian.pentaho.org/.

Sapia, C. (1999). On modeling and predicting query behavior in OLAP systems. In *DMDW*, pp. 2.1–2.10.

Sapia, C. (2000). Promise: Predicting query behavior to enable predictive caching strategies for OLAP systems. In *DaWaK*, pp. 224–233.

Sarawagi, S. (1999). Explaining differences in multidimensional aggregates. In *VLDB*, pp. 42–53.

Sarawagi, S. (2000). User-adaptive exploration of multidimensional data. In *VLDB*, pp. 307–316.

Sarawagi, S., R. Agrawal, and N. Megiddo (1998). Discovery-driven exploration of OLAP data cubes. In *EDBT*, pp. 168–182.

Sathe, G. and S. Sarawagi (2001). Intelligent rollups in multidimensional OLAP data. In *VLDB*, pp. 531–540.

Spiliopoulou, M., J. Srivastava, R. Kohavi, and B. M. Masand (2000). Webkdd 2000 - web mining for e-commerce. *SIGKDD Explorations 2*(2), 106–107.

Wu, P., Y. Sismanis, and B. Reinwald (2007). Towards keyword-driven analytical processing. In *SIGMOD Conference*, pp. 617–628.

Résumé

L'analyse interactive de cube de données, où l'utilisateur navigue à l'aide d'une séquence de requêtes un cube pour trouver et expliquer des données inattendues, est souvent fastidieuse. Pour améliorer ce processus, nous proposons dans ce papier de rapprocher deux techniques proposées précédemment dans ce domaine. Ces techniques sont d'une part l'analyse pilotée par la découverte, où l'utilisateur est guidé vers des régions du cube supposées intéressantes, et d'autre part la recommandation de requêtes, qui tire profit de ce que les autres utilisateurs on fait lors d'analyses précédentes. A l'aide de ces techniques nous proposons un cadre pour la recommendation de requêtes OLAP à un utilisateur en prenant en compte ce que les utilisateurs précédents auront trouvé intéressant et la manière qu'ils auront eu de l'expliquer.