Towards Intensional Answers to OLAP Queries for Analytical Sessions

Patrick Marcel
University of Tours
3 Place Jean Jaurès
Blois, France
marcel@univ-tours.fr

Rokia Missaoui
Univ. of Quebec in Outaouais
101, Rue Saint-Jean-Bosco
Gatineau, Canada
rokia.missaoui@uqo.ca

Stefano Rizzi
University of Bologna
Viale Risorgimento, 2
Bologna, Italy
stefano.rizzi@unibo.it

ABSTRACT

One of the problems in analyzing large multidimensional databases through OLAP sessions is that decision makers can be overwhelmed by the size of query answers, while they need a concise summary of data. Intensional query answering can help by providing a concise description of extensional answers (i.e., the sets of retrieved facts), generally relying on knowledge like integrity constraints, taxonomies, or patterns discovered from data. This paper proposes a framework for computing an intensional answer to an OLAP query by leveraging on the previous queries in the current session. Such intensional answer is concise and semantically rich, and allows the size of the extensional answers returned to be reduced, so as to achieve an effective trade-off between conciseness and informational content. After describing the general framework, we propose a specific instantiation that relies on previous contributions in cube modeling and intensional query answering.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*theory and methods*; H.4.2 [**Information Systems Applications**]: Types of Systems—*decision support*

Keywords

OLAP, intensional query answering

1. INTRODUCTION

One of the key factors that rule the effectiveness of multidimensional analysis is the achievement of a satisfactory (from the users' viewpoint) compromise between the precision and the size of the information being analyzed. The OLAP paradigm gives a significant support in this direction by enabling users to interactively change the aggregation level of data by zooming in (with the drill-down operator) and out (with the roll-up operator), and to selectively view only a subset of data (with the slice-and-dice operator). But this is not always sufficient: more detail gives more information, but at the risk of missing the overall picture, while focusing on general trends of data may prevent users from observing specific small-scale phenomena.

Different approaches can be taken to cope with this issue. For instance, in preference-based OLAP personalization there is an attempt to avoid information flooding by considering the users' preferred aggregation levels, measures, and slices [7]. In approximate query answering, the focus is on quickly returning an answer at the price of some imprecision in the returned values [23]. Other approaches couple the OLAP paradigm with data mining techniques to create an OLAM approach where multidimensional data can be mined "on-the-fly" to extract concise patterns for user's evaluation, but at the price of an increased computational complexity and an overhead for analyzing the generated patterns [8].

Another interesting technique to tackle this problem is *intensional query answering*. In the context of databases, an intensional answer to a query q is an answer that, instead of providing the precise and complete set of tuples returned by q (which is called the *extensional answer* to q), summarizes this set with a concise description of the properties the tuples share [14]. Approaches for computing intensional answers to database queries have been classified according to three criteria [14]:

- *Purity*: the answer can be purely intensional or mixed with extensional information.
- *Completeness*: the intensional answer can fully cover the extensional answer or cover it only partially.
- *Dependency*: the computation of the intensional answer can be dependent on the database extension or use intensional information only.

In this paper we present a framework for intensional OLAP query answering that returns a concise, yet informative answer to a query posed by a user by leveraging on the queries that user has previously addressed during the current OLAP session. In our approach, an *expected cube* is created and updated based on the results of each query in the current session, as a representation of the user's understanding of the data. Each time a new query is posed, it is actually executed on both the real cube and the expected cube; the differences between the two answers express the relevant information for that query, i.e., the information that the user could not correctly extrapolate from the previous answers she got. These differences are effectively summarized using the dimension hierarchies of the cube, and presented to the user under the form of an intensional answer mixed with a partial extensional answer. More specifically, the idea is to use an intensional answer to concisely characterize the cube regions whose data approximately match with the expectation, and an extensional answer to describe in detail only the cube regions whose data significantly differ from the expectation.

Using Motro's criteria, our approach can be classified as mixed, partial, and dependent. While it is general —because it is independent of the particular method adopted for building the expected cube and deriving the intensional answer— we precisely describe an instance of it that relies on previous contributions in the domain of cube modeling and intensional answers.

The paper outline is as follows. Section 2 introduces the general framework we propose while Section 3 describes a specific instantiation. Section 4 discusses the related literature, and Section 5 draws the conclusion.

2. THE FRAMEWORK

This section motivates and then introduces the general framework we propose, starting with the definition of cubes and queries (based on a simplified formalization used in [1]).

2.1 Motivating Example

This simple example will give an intuition of our approach. We consider a cube of sales per city, product, and month. Three hierarchies are defined, namely LOCATION, PRODUCT, and TIME (see Figure 1). For example, a product (Redtab) belongs to a group (Levi's). A portion of the cube is shown below.

	Redtab		Silvertab	
	Jan.11	Feb.11	Jan.11	Feb.11
Queens	50	40	30	40
Brooklyn	10	20	10	0
Toronto	0	10	0	10
Ottawa	0	10	0	10

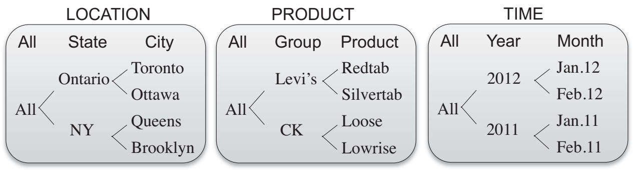

Figure 1: Dimension hierarchies

Consider an OLAP session consisting of three queries that investigate monthly sales of products. The first query simply asks for the grand total, i.e., the total sales for all products, all locations, all years. The system returns the total, say 640, to the user and updates the expected cube accordingly.

The user then combines a drill-down and a slice operator to ask for the 2011 monthly sales per product per state. Knowing the grand total, she might expect that the distribution of sales is the following:

		Redtab	Silvertab	Loose	Lowrise
Ontario	Jan.11	20	20	20	20
	Feb.11	20	20	20	20
NY	Jan.11	20	20	20	20
	Feb.11	20	20	20	20

These values are indeed those currently stored in the expected cube. However, the actual (extensional) answer to the query is as follows:

		Redtab	Silvertab	Loose	Lowrise
Ontario	Jan.11	0	0	10	10
	Feb.11	20	20	10	10
NY	Jan.11	60	40	20	20
	Feb.11	60	40	20	20

Parts of this answer match the user's current understanding of data while others do not; to reduce the overall size of the answer, the system compares the extensional answer with the data in the expected cube, and it only returns the "unexpected" facts:

		Redtab	Silvertab	Loose	Lowrise
Ontario	Jan.11	0	0	10	10
	Feb.11			10	10
NY	Jan.11	60	40		
	Feb.11	60	40		

integrated with an intensional answer that summarizes the "expected" facts:

⟨Ontario, Levi's, Feb.11⟩: *as expected*
⟨NY, CK, 2011⟩: *as expected*

In this example, the "as expected" value is used to inform the user that the facts not reported in the extensional answer do not deviate at all from her expectation. A more sophisticated form of intensional answer is discussed in Section 3. Finally, the system uses the complete extensional answer to update the expected cube.

Let the third query in the session be a drill-down to city. The complete extensional answer is twice the size of the previous one (because in this example we only have 2 cities per state), but again the system may represent it concisely using intensional information:

		Redtab	Silvertab
Jan.11	Queens	50	30
	Brooklyn	10	10
Feb.11	Queens	40	40
	Brooklyn	20	0

⟨Ontario, All, 2011⟩: *as expected*
⟨NY, CK, 2011⟩: *as expected*

Importantly, the "as expected" value is now to be interpreted with respect to the user understanding of data after the previous answer. This means, for instance, that the sales of Redtab in Toronto for Feb. 2011 is 10 (the Feb. 2011 sales for Redtab in Ontario is 20, which is expected to be fairly distributed between Toronto and Ottawa).

2.2 Preliminary Definitions

2.2.1 Cubes

Our formalization of cubes involves hierarchies; however, to keep the formalism simpler, and without actually restricting the validity of our approach, we will consider hierarchies without branches, i.e., consisting of chains of levels.

Definition 2.1 (Multidimensional Schema). *A multidimensional schema (or, briefly, a* schema*) is a couple $\mathcal{M} = \langle L, H \rangle$ where:*

- *L is a finite set of* levels*, each level $l \in L$ being defined on a categorical domain $Dom(l)$;*

- $H = \{h_1, \ldots, h_n\}$ *is a finite set of* hierarchies*, each characterized by (1) a subset $L_i \subseteq L$ of levels (all L_i's are disjoint); (2) a* roll-up *total order $\succeq_{h_i}$ of L_i; and (3) a family of* roll-up functions *including a function $Dom(l_k) \rightarrow Dom(l_j)$ for each pair of levels l_k and l_j such that $l_k \succeq_{h_i} l_j$.*

For each hierarchy h_i, the top level of the order is called a *dimension*, denoted by DIM_i, and determines the finest aggregation level for h_i. Conversely, the bottom level (denoted ALL_i) has a single value in its domain ($Dom(ALL_i) = \{\text{All}\}$ for each i) and determines the coarsest aggregation level. Roll-up functions allow for values of fine-grained levels to be mapped into values of coarse-grained levels.

A group-by set includes one level for each hierarchy, and defines a possible way to aggregate data. A coordinate of a group-by set is a point in the n-dimensional space defined by the levels in that group-by set.

Definition 2.2 (Group-by Set). *Given schema $\mathcal{M}$, let $Dom(H) = L_1 \times \ldots \times L_n$; each $G \in Dom(H)$ is called a* group-by set *of $\mathcal{M}$. Let $G = \langle l_{k_1}, \ldots, l_{k_n} \rangle$ and $Dom(G) = Dom(l_{k_1}) \times \ldots \times Dom(l_{k_n})$; each $g \in Dom(G)$ is called a* coordinate *of G.*

Let $\succeq$ denote the product order[1] of the roll-up orders of the hierarchies in H. Then, $(Dom(H), \succeq)$ is a lattice, that we will call *group-by lattice*, whose top and bottom elements are $G^\top = \langle DIM_1, \ldots, DIM_n \rangle$ and $G^\perp = \langle ALL_1, \ldots, ALL_n \rangle$, respectively. Given two group-by sets G and G' such that $G \succeq G'$ and two coordinates $g \in Dom(G)$ and $g' \in Dom(G')$, we write $g \succeq g'$ to denote that, for each hierarchy h_i, the value of level l_{k_i} in g rolls-up to the value of level $l_{k'_i}$ in g' through the roll-up function from l_{k_i} to $l_{k'_i}$. The domain of $G^\perp$ includes a single coordinate, $g^\perp = \langle \text{All}, \ldots, \text{All} \rangle$, such that $g \succeq g^\perp$ for any coordinate g of any group-by set.

Example 2.1. *In our example, $n = 3$ and* City $\succeq_{\text{LOCATION}}$ State $\succeq_{\text{LOCATION}}$ All*; examples of group-by sets are:*

$$G^\top = \langle \mathsf{City}, \mathsf{Product}, \mathsf{Month} \rangle$$
$$G_1 = \langle \mathsf{State}, \mathsf{Product}, \mathsf{Month} \rangle$$
$$G_2 = \langle \mathsf{All}, \mathsf{All}, \mathsf{Year} \rangle$$

It is $G^\top \succeq G_1 \succeq G_2$. Examples of coordinates of these group-by sets are, respectively,

$$g^\top = \langle \text{Toronto}, \text{Loose}, \text{Jan.11} \rangle$$
$$g_1 = \langle \text{Ontario}, \text{Loose}, \text{Jan.11} \rangle$$
$$g_2 = \langle \text{All}, \text{All}, 2011 \rangle$$

with $g^\top \succeq g_1 \succeq g_2$ because Toronto *rolls-up to* Ontario*,* Jan.11 *rolls-up to* 2011*, and everything rolls-up to* All *as shown in Figure 1.*

A schema is populated with facts, each characterized by a group-by set G that defines its aggregation level, a coordinate of G, and a numerical value for a measure. To keep the notation simple and without restricting the validity of our approach, we will consider cubes having a single measure.

[1]The product order of n total orders is a partial order of the Cartesian product of the n totally ordered sets, such that $\langle x_1, \ldots, x_n \rangle \succeq \langle y_1, \ldots, y_n \rangle$ iff $x_i \succeq y_i$ for $i = 1, \ldots, n$.

Definition 2.3 (Cubes and Facts). *Given a schema $\mathcal{M}$, an instance of $\mathcal{M}$, called a* cube*, is a (partial) function $C : \bigcup_{G \in Dom(H)} \rightarrow \mathbb{R}$ such that, for each $G \neq G^\top$ and $g \in Dom(G)$, the value of $C(g)$ is an aggregation of the values of $C(g_i^\top)$ for all the $g_i^\top$'s such that $g_i^\top \succeq g$. We call a* fact *of C any couple $f = \langle g, C(g) \rangle$ for which $C(g)$ is defined.*

For simplicity, we will consider a cube as a set of facts, so we will write $f \in C$ to denote that f is a fact of C. Also, to transparently deal with the cube sparseness issues, in the following we will denote with $Dom_C(G)$ the *active* domain of G in cube C, i.e., the subset of coordinates g of G for which $C(g)$ is defined.

Intuitively, a slice is a subset of facts of a cube that have a given group-by set G and satisfy a given selection predicate (expressed on hierarchy levels not finer than those in G). It is formally defined as follows.

Definition 2.4 (Slice). *Let $\mathcal{M}$ be a schema and C be an instance of $\mathcal{M}$. A* slice schema *of $\mathcal{M}$ is a couple $\Sigma = \langle G, G_{sel} \rangle$ where G and G_{sel}, with $G \succeq G_{sel}$, are two group-by sets of $\mathcal{M}$. Given a coordinate $g_{sel} \in Dom(G_{sel})$, the* slice *of C according to schema Σ over g_{sel} is the subset of facts of C defined as follows:*

$$\sigma_{\Sigma, g_{sel}}(C) = \{\langle g, C(g) \rangle | g \in Dom_C(G), g \succeq g_{sel}\}$$

Example 2.2. *With reference to Example 2.1, an example of fact is $f = \langle g_1, 10 \rangle$ which indicates that the amount of sales in Ontario for product Loose in Jan.11 is 10. A possible slice schema is $\Sigma = \langle G_1, G_2 \rangle$, and the slice according to Σ over g_2 is the set of monthly sales per state and product in 2011.*

2.2.2 Queries and Answers

The queries we consider are basic (GPSJ) OLAP queries characterized by a group-by set and a selection predicate. The extensional answer to a query is the set of facts at the required group-by set that satisfy the selection predicate. An OLAP session is a sequence of correlated queries formulated by a user on a single cube; typically (but not necessarily), each query in a session is derived from the previous one by applying an OLAP operator (such as roll-up, drill-down, and slice-and-dice).

Definition 2.5 (Query and Session). *A* query *over schema $\mathcal{M}$ is a couple $q = \langle \Sigma, g_{sel} \rangle$ where $\Sigma = \langle G, G_{sel} \rangle$ is a slice schema and $g_{sel} \in Dom(G_{sel})$ is a coordinate. An* OLAP session *is a sequence s of queries on schema $\mathcal{M}$.*

Definition 2.6 (Extensional Answer). *Let C be a cube with schema $\mathcal{M}$ and $q = \langle \Sigma, g_{sel} \rangle$ be a query over $\mathcal{M}$. The* extensional answer *to q over C is the slice $Ext_q(C) = \sigma_{\Sigma, g_{sel}}(C)$.*

We now define the type of intensional answer we consider, that is close to the one adopted in [20]. An intensional answer is a set of couples, each associating a coordinate with a quantification of predictability.

Definition 2.7 (Intensional Answer). *Let C be a cube with schema $\mathcal{M}$ and $q = \langle \Sigma, g_{sel} \rangle$ be a query over $\mathcal{M}$, with $\Sigma = \langle G, G_{sel} \rangle$. An* intensional answer *to q over C is any set of couples $Int_q(C) = \{\langle g', d \rangle\}$ such that*

- *g' is a coordinate of a group-by set G' such that $G \succeq G' \succeq G_{sel}$, $g' \succeq g_{sel}$, and $g' \in Dom_C(G')$;*
- *d is an attribute characterizing how close is the actual information for g' to the user's expectations. The smaller d, the higher the similarity between actual data and user's expectations.*

The coverage *of $Int_q(C)$ is the subset of coordinates $g \in Dom_C(G)$ such that $Int_q(C)$ includes a couple $\langle g', d\rangle$ with $g \succeq g'$.*

Note that, with this definition, couples at different group-by sets can be mixed in an intensional answer. Besides, the coverage of an intensional answer is always a subset of the set of coordinates belonging to the extensional answer. In particular, an intensional answer $Int_q(C)$ is *complete* if all the coordinates in $Ext_q(C)$ are included in the coverage of $Int_q(C)$; it is *non redundant* if, for each $g \in Dom(G)$, there is at most one couple $\langle g', d\rangle \in Int_q(C)$ such that $g \succeq g'$.

Example 2.3. *With reference to Example 2.1 and given slice schema $\Sigma = \langle G_1, G_2\rangle$, an example of query is $q = \langle \Sigma, g_2\rangle$ (monthly sales per state and product in 2011). Consistently with the cube shown in Section 2.1, two sample facts in the extensional answer to q are*

$$f_1 = \langle\langle \text{NY}, \text{Loose}, \text{Jan.11}\rangle, 20\rangle$$
$$f_2 = \langle\langle \text{Ontario}, \text{Silvertab}, \text{Feb.11}\rangle, 20\rangle$$

while a possible intensional answer could be

$$\langle\langle \text{Ontario}, \text{Levi's}, \text{Feb.11}\rangle, 0\rangle$$
$$\langle\langle \text{All}, \text{Levi's}, 2011\rangle, 80\rangle$$

where the d part of each tuple measures the absolute difference between the actual data and the expected ones. Note that this intensional answer is redundant (fact f_2 is covered by both tuples in the intensional answer) and non-complete (fact f_1 is not covered). Moreover, the first entry of this intensional answer states that the Feb. 2011 sales for the Levi's products in Ontario are exactly as expected.

2.3 Approach Overview

Given a query q over cube C, the basic idea of our work is to return a hybrid answer that combines an extensional part and an intensional part to achieve a satisfactory trade-off between conciseness and precision. This trade-off is based on the history of the data the user has seen so far during the current OLAP session. The overall process can be sketched as follows:

Startup: The user formulates her first query q_1 over C; the extensional answer $Ext_q(C)$ is computed, returned to the user, and used to initialize an *expected cube EC*, sharing the same schema of C and storing a model of the user's expected values according to the data included in $Ext_q(C)$.

Iteration: For each subsequent query q_i formulated by the user:

0. **Execute**. Query q_i is executed against C.
1. **Predict**. Query q_i is executed against EC.
2. **Improve**. The extensional answer $Ext_{q_i}(C)$ is used to update EC.
3. **Build**. An intensional answer $Int_{q_i}(C)$ is built based on the result of the comparison between $Ext_{q_i}(EC)$ and $Ext_{q_i}(C)$, and returned to the user together with the subset of $Ext_{q_i}(C)$ that complements the coverage of $Int_{q_i}(C)$.

The generic structure of the framework is represented in Figure 2. Besides query execution against the actual cube (*Execute*), there are three components that are sequentially processed as many times as there are queries in a session. While *Improve* aims to continuously improve the quality of the estimated values in the expected cube based on the generated extensional answer, *Predict* allows the computation of the expected extensional answer that is further used by *Build* together with the actual extensional answer to produce an intensional answer. Such response is obtained by capturing the differences between the two extensional answers at some levels of aggregation.

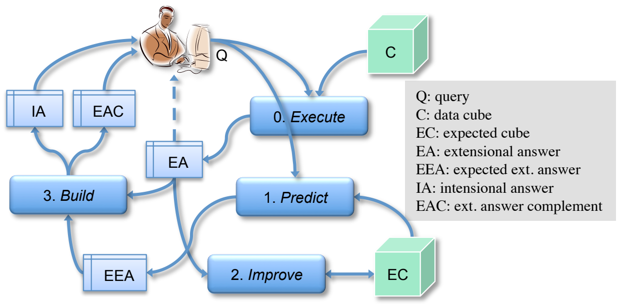

Figure 2: Steps of the approach

Next section will describe an instantiation of this framework.

3. AN INSTANCE OF THE FRAMEWORK

In this section we describe a particular instance of the framework, that relies on past contributions in the domain of cube modeling [18, 16] and intensional answers [20, 17]. We first describe how the expected cube is maintained, then how it is queried, and finally how the intensional answer is built.

3.1 Improve

The expected cube, EC, can be seen as a representation of the user's understanding of the data in the actual cube. After each query, appropriate parts of EC are compared with the extensional answer in order to compute the intensional answer (*Predict*); then, EC is updated using the extensional answer. EC not only stores estimates for the measure values, but also a confidence $conf(f)$ attached to each fact f that indicates how reliable the estimated value is. It is $conf(f) = 1$ if fact f is precisely known to the user, i.e., if it has been shown to her as a part of the extensional answer for a query of the current OLAP session.

We first explain how EC is updated, then how the confidence is computed.

3.1.1 Estimating Facts

The Maximum Entropy principle has been successfully used in the OLAP context for estimating a data cube's fact values [18, 16]. According to the maximum entropy principle, the best estimated values are those that maximize the uniformity of data values, while maintaining the value of known aggregates. In the context of OLAP, the collection of known facts used for modeling purposes is the set of past extensional answers.

We use the description given in [16] to define our estimation principle. In this work, the authors were concerned with reconstructing an n-dimensional cube instance based on some of its aggregates, and they showed that this problem is analogous to that of reconstructing an n-dimensional probability distribution from a number of its marginal distributions.

Given a slice schema $\Sigma = \langle G, G_{sel}\rangle$ and a coordinate $g_{sel} \in Dom_{EC}(G_{sel})$, the fact values for slice $\sigma = \sigma_{\Sigma,g_{sel}}(EC)$ are estimated as those that maximize the *entropy* H of σ using the subset S of the facts of EC that aggregate facts in σ and have confidence 1. More precisely, the problem is to find the fact values that maximize

$$H(\sigma) = \sum_{\langle g, EC(g)\rangle \in \sigma} EC(g) \times log(EC(g))$$

while respecting the aggregation constraints

$$\sum_{\langle g, EC(g)\rangle \in \sigma, g \succeq g'} EC(g) = EC(g'),\ \forall \langle g', EC(g')\rangle \in S$$

where

$$S = \{f' \in EC | f' = \langle g', EC(g')\rangle, G \succeq G', conf(f') = 1\}$$

These estimates can be efficiently computed using Iterative Proportional Fitting (IPF), the main method used in log-linear modeling to model multi-way frequency tables.

EXAMPLE 3.1. *Consider slice schema $\Sigma = \langle G_1, G_2\rangle$, and the slice according to Σ over g_2 which is the set of monthly sales per state and product in 2011. If the only known aggregate is 640 (the grand total), the estimated value for each fact of this slice is $\frac{640/2}{16} = 20$, since this value maximizes the entropy of the slice. If it is also known that the overall sale of CK products for 2011 is 120, then the estimated values for this slice will be 120/8=15 for the facts related to CK products, and 200/8=25 for those related to Levi's products.*

This solution for creating and maintaining the expected cube suffers from the following drawbacks: 1) only aggregates are considered for estimating fact values, and 2) it relies on the assumption that data is generally uniformly distributed, which may not be consistent with the user's view and with the actual data. Point (1) is easily dealt with through Algorithm 1: each time a new query is posed to the data cube C, each fact f in its extensional answer (line 1) is used to update the facts of EC that are either in the same slice (line 2), at a coarser granularity (using aggregation, lines 3-4), or at a finer granularity than f (using IPF, lines 5-6). Note that the very first call to Algorithm 1 is done with $EC = \emptyset$. Point (2) is discussed in Section 3.2.

EXAMPLE 3.2. *Consider the following sample of EC:*

$\langle\langle$All, CK, 2011$\rangle$, 160$\rangle$	$\langle\langle$Queens, CK, 2011$\rangle$, 40$\rangle$
$\langle\langle$Ontario, CK, 2011$\rangle$, 80$\rangle$	$\langle\langle$Brooklyn, CK, 2011$\rangle$, 40$\rangle$
$\langle\langle$NY, CK, 2011$\rangle$, 80$\rangle$	$\langle\langle$Ottawa, CK, 2011$\rangle$, 40$\rangle$
	$\langle\langle$Toronto, CK, 2011$\rangle$, 40$\rangle$

and suppose that a query is posed returning the following extensional answer:

Algorithm 1 Improve EC

```
Input: $\overline{\sigma} = \sigma_{\langle G, G_{sel}\rangle, g_{sel}}(EC)$: an extensional answer; $EC$: the
    expected cube
Output: $EC$: the updated expected cube
1: for each fact $f = \langle g, C(g)\rangle \in \overline{\sigma}$ do
2:     let $EC(g) = C(g)$
3:     for each group-by set $G'$ such that $G \succeq G'$ do
4:         recompute $\sigma_{\langle G', G_{sel}\rangle, g_{sel}}(EC)$
5:     for each group-by set $G'$ such that $G' \succeq G$ do
6:         maximize $H(\sigma_{\langle G', G_{sel}\rangle, g_{sel}}(EC))$
7: return $EC$
```

$\langle\langle$Ontario, CK, 2011$\rangle$, 40$\rangle$
$\langle\langle$NY, CK, 2011$\rangle$, 80$\rangle$

EC can be updated based on this extensional answer, and the sample above becomes:

$\langle\langle$All, CK, 2011$\rangle$, 120$\rangle$	$\langle\langle$Queens, CK, 2011$\rangle$, 40$\rangle$
$\langle\langle$Ontario, CK, 2011$\rangle$, 40$\rangle$	$\langle\langle$Brooklyn, CK, 2011$\rangle$, 40$\rangle$
$\langle\langle$NY, CK, 2011$\rangle$, 80$\rangle$	$\langle\langle$Ottawa, CK, 2011$\rangle$, 20$\rangle$
	$\langle\langle$Toronto, CK, 2011$\rangle$, 20$\rangle$

3.1.2 *Scoring the Estimates*

The confidence $conf(f)$ associated with each fact $f \in EC$ is a real in [0,1], with 1 assigned to facts whose measure value is precisely known to the user. The estimates for EC are obtained using known (i.e., with confidence 1) aggregates, and it has been shown that the more precise the aggregate used for estimation, the more accurate the estimate [16]. So, let $Agg(f) \subseteq EC$ be the set of the aggregates that have been used to estimate $f = \langle g, EC(g)\rangle$, with $g \in Dom(G)$:

$$Agg(f) = \{f' \in EC | f' = \langle g', EC(g')\rangle, g \succeq g', conf(f') = 1\}$$

The confidence $conf(f)$ is computed as the maximum similarity between G and the group-by sets of the aggregates in $Agg(f)$, where the similarity $\delta(G, G')$ between two group-by sets G and G' is computed from the distance between G and G' in the group-by lattice in such a way that $\delta(G, G) = 1$ for any G and $\delta(G^{\top}, G^{\perp}) = 0$. The confidence of a slice of the expected cube is the average of the confidences of its facts.

EXAMPLE 3.3. *Consider the fact f with coordinate* $\langle$NY, CK, Jan.11$\rangle$*, and let*

$$Agg(f) = \{\langle\langle \text{All,All,All}\rangle, 640\rangle, \langle\langle \text{All,CK,2011}\rangle, 120\rangle\}$$

Aggregate $\langle\langle$All,CK,2011$\rangle$, 120$\rangle$ *is the one whose group-by set* $\langle$All, Group, Year$\rangle$ *is the closest to the one of f, that is,* $\langle$State, Group, Month$\rangle$*. These group-by sets have distance 2 on the lattice, so the confidence for f in EC will be 0.66 (in this case the maximum distance on the lattice is 6).*

3.2 Predict

Given query $q = \langle \Sigma, g_{sel}\rangle$, with $\Sigma = \langle G, G_{sel}\rangle$, the goal of this step is to extract from EC the expected extensional answer to q, i.e., the user's understanding of the data asked by q based on the data she saw so far during the current session.

A basic way to achieve this goal is to define the expected extensional answer as the exact slice obtained by posing q against EC: $Ext_q(EC) = \sigma_{\Sigma,g_{sel}}(EC)$. However, more elaborated approaches can be devised to deal with the cases in which this slice has low confidence. In the approach we present here, we try to derive a more reliable prediction based on other slices that share the same slice schema Σ but were estimated more accurately, i.e., by using aggregates at closer group-by sets. In particular, we propose to define $Ext_q(EC)$ by using, besides $\sigma_{\Sigma,g_{sel}}(EC)$, the slice(s) of EC with highest confidence among those having schema Σ.

EXAMPLE 3.4. *Consider the following sample of EC, where each fact is associated with the aggregate used to estimate it as well as with the fact's confidence.*

f	$Agg(f)$	$conf(f)$
$\langle\langle$Ontario,CK,2012$\rangle, 80\rangle$	$\langle\langle$All,All,2012$\rangle, 320\rangle$	*0.4*
$\langle\langle$NY,CK,2012$\rangle, 80\rangle$	$\langle\langle$All,All,2012$\rangle, 320\rangle$	*0.4*
$\langle\langle$Ontario,CK,2011$\rangle, 80\rangle$	$\langle\langle$All,CK,2011$\rangle, 120\rangle$	*0.9*
$\langle\langle$NY,CK,2011$\rangle, 80\rangle$	$\langle\langle$NY,All,2011$\rangle, 280\rangle$	*0.9*

The first two facts form slice $\sigma_{2012} = \sigma_{\Sigma,\langle \text{All,CK,2012}\rangle}$ where $\Sigma = \langle\langle \mathsf{State, Group, Year}\rangle, \langle \mathsf{All, Group, Year}\rangle\rangle$ while the last two facts form slice $\sigma_{2011} = \sigma_{\Sigma,\langle \text{All,CK,2011}\rangle}$. Now let q ask for the 2012 sales for CK by states. It turns out that the estimates for 2011 (slice σ_{2011}) have higher confidence than those for 2012 (slice σ_{2012}). Therefore, the former will be used to adjust the latter.

Finding the slices of EC to compute the expected extensional answer to q is done with Algorithm 2, whose explanation is given below. Let

$$Same(G, g_{sel}) = \{g'_{sel} \in Dom_{EC}(G_{sel}) | \\ \forall i \in \{1,\ldots,n\}, G.h_i \succ G_{sel}.h_i \Rightarrow g'_{sel}.h_i = g_{sel}.h_i\}$$

where $G.h_i$ denotes the level of group-by set G in hierarchy h_i and $g.h_i$ denotes the value of coordinate g in h_i. Importantly, for each $g'_{sel} \in Same(G, g_{sel})$, and whatever the hierarchy, the level value used in g'_{sel} rolls up to the same level value as the one used in g_{sel}. Among the slices over the coordinates in $Same(G, g_{sel})$, the ones achieving the highest confidence are retained only (*candidate slices*, line 1 of Algorithm 2):

$$Cand(q) = argmax_{g'_{sel} \in Same(G,g_{sel})} conf(\sigma_{\Sigma,g'_{sel}}(EC))$$

If $\sigma_{\Sigma,g_{sel}}(EC) \in Cand(q)$ (i.e., the slice precisely requested by q has highest confidence) then it is returned as the expected extensional answer (lines 2-3). Otherwise, the other slices in $Cand(q)$ will be used as well (lines 4-18). Let g be the coordinate of one of the facts in $\sigma_{\Sigma,g_{sel}}(EC)$ to be predicted (line 7); let σ^* be a slice in $Cand(q)$ and g^* be the homologous coordinate to g in σ^* (line 11). For each σ^* (line 10), the proportion of $EC(g^*)$ to the closest aggregate that was used to estimate it (line 13) is computed and weighted with the confidence of $EC(g^*)$ (line 14). This weighted ratio is applied to the aggregate that is used to predict $EC(g)$. Finally, the prediction for $EC(g)$ is the weighted average over all slices in $Cand(q)$ (line 16). Note that this principle is consistent with the way IPF estimates values from aggregates. Note also that the predicted slice as output by Algorithm 2 is not used to update the expected cube, since, as mentioned above, only actual extensional answers are used to that end.

EXAMPLE 3.5. *Going on with Example 3.4, it is*

$$Same(\langle \mathsf{State, Group, Year}\rangle, \langle \text{All,CK,2012}\rangle) = \\ \{\langle \text{All,CK,2011}\rangle, \langle \text{All,CK,2012}\rangle, \\ \{\langle \text{All,Levi's,2011}\rangle, \langle \text{All,Levi's,2012}\rangle\}$$

and

$$Cand(q) = \{\sigma_{2011}\}$$

The expected extensional answer will include fact $\langle\langle$Ontario, CK,2012$\rangle, 172.3\rangle$ because

$$\frac{320 \times \frac{80}{120} \times 0.9 + 320 \times \frac{80}{320} \times 0.4}{0.9 + 0.4} = 172.3$$

Algorithm 2 Predict

```
Input: $q = \langle \Sigma = \langle G, G_{sel}\rangle, g_{sel}\rangle$: the current query; $EC$: the expected cube
Output: $Ext_q(EC)$: the expected extensional answer
 1: let $Cand(q) = argmax_{g'_{sel} \in Same(G,g_{sel})} conf(\sigma_{\Sigma,g'_{sel}}(EC))$
 2: if $\sigma_{\Sigma,g_{sel}}(EC) \in Cand(q)$ then
 3:     return $\sigma_{\Sigma,g_{sel}}(EC)$
 4: else
 5:     $Ext_q(EC) = \emptyset$
 6:     $Cand(q) = Cand(q) \cup \{\sigma_{\Sigma,g_{sel}}(EC)\}$
 7:     for each fact $\langle g, EC(g)\rangle \in \sigma_{\Sigma,g_{sel}}(EC)$ do
 8:         let $v_{exp} = 0$
 9:         let $sum_{coef} = 0$
10:         for each slice $\sigma^* \in Cand(q)$ do
11:             let $f^* = \langle g^*, v^*\rangle \in \sigma^*$
12:             let $\langle g^{ex}_{ag}, v^{ex}_{ag}\rangle = argmin_{agg(\langle g,EC(g)\rangle)} \delta(G, G')$
13:             let $\langle g^*_{ag}, v^*_{ag}\rangle = argmin_{agg(f^*)} \delta(G, G')$
14:             $v_{exp} = v_{exp} + (v^{ex}_{ag} \times \frac{v^*}{v^*_{ag}} \times conf(f^*))$
15:             $sum_{coef} = sum_{coef} + conf(f^*)$
16:         $v_{exp} = \frac{v_{exp}}{sum_{coef}}$
17:         $Ext_q(EC) = Ext_q(EC) \cup \{\langle g, v_{exp}\rangle\}$
18: return $Ext_q(EC)$
```

3.3 Build

The aim of this step is to derive an intensional answer from the extensional answer obtained from C and from the expected extensional answer obtained from EC. We propose an approach loosely inspired by that of [17] and [20], in the sense that hierarchies are used to derive the intensional answer in the spirit of [17], and an information theoretic characterization is used in the spirit of [20].

More precisely, the extensional answer $Ext_q(C)$ and the expected extensional answer $Ext_q(EC)$ are compared by computing, for each coordinate, the deviation of the fact value in $Ext_q(C)$ from the one in $Ext_q(EC)$, using the estimation error, normalized with the standard deviation of all estimation errors, as in [16].

A first basic intensional answer is formed with these deviations, as the set of couples $\{\langle g, d\rangle\}$ where $d = \frac{|C(g) - EC(g)|}{\gamma}$, $\langle g, C(g)\rangle$ is a fact of the extensional answer, $\langle g, EC(g)\rangle$ is a fact in the expected extensional answer, and γ is the standard deviation of all estimation errors.

The final intensional answer is derived from the basic intensional answer by looking for regions of the basic intensional answer where deviations are homogeneous enough (using a threshold α) and the mean deviation is low (using a threshold β). These regions are delimited using the granularity levels that are coarser than that of the basic intensional answer.

The homogeneity is formally computed as the entropy, i.e., for a given slice σ, it is $H(\sigma) = \sum_{\langle g,d\rangle \in \sigma} d \times log(d)$, and the final intensional answer is computed by Algorithm 3. Each couple $\langle g, d\rangle$ of the final intensional answer is interpreted as the facts covered by $\langle g, d\rangle$ being close to the prediction. This intensional answer is then complemented with the facts of the extensional answer that are not covered by the intensional answer, since those facts deviate too much from the prediction; note that, by doing so, the final intensional answer is complete.

Algorithm 3 Build IA

```
Input: $Ext_q(C)$: the extensional answer to $q = \langle\langle G, G_{sel}\rangle, g_{sel}\rangle$;
    $Ext_q(EC)$: the expected extensional answer to $q$; $\alpha, \beta$: two
    thresholds
Output: $Int_q(C)$: the intensional answer
Variables: $BIA_q$: the basic intensional answer
 1: let $BIA_q = \emptyset$
 2: for each coordinate $g$ such that $\langle g, C(g)\rangle \in Ext_q(C)$ do
 3:     $BIA_q = BIA_q \cup \{\langle g, \frac{|C(g)-EC(g)|}{\gamma}\rangle\}$
 4: return $Int_q(C) = FindRegion(BIA_q, G, \alpha, \beta)$
```

Function 4 FindRegion

```
Input: $BIA_q$: an intensional answer to $q$; $G$: a group-by set; $\alpha, \beta$:
    two thresholds
Output: $Int_q$: an intensional answer to $q$
Variables: $IA_{G_m}, BIA_{G_m}$: intensional answers
 1: if $H(BIA_q) < \alpha$ then                     ▷ $BIA_q$ is homogeneous enough
 2:     let $d = \frac{\sum_{\langle g,v\rangle \in BIA_q} v}{|BIA_q|}$
 3:     if $d < \beta$ then                          ▷ mean deviation is low enough
 4:         return $\{\langle g_{sel}, d\rangle\}$
 5:     else                                         ▷ mean deviation is too high
 6:         return $\emptyset$
 7: else                                             ▷ $BIA_q$ is not homogeneous enough
 8:     for each $G_m \in max_{\succeq}(\{G' \in Dom(H) | G \succeq G'\})$ do
 9:         let $BIA_{G_m} = \{\langle g, d'\rangle \in BIA_q | g \succeq g', g' \in Dom(G_m)\}$
10:         let $IA_{G_m} = FindRegion(BIA_{G_m}, G_m, \alpha, \beta)$
11:     return $\bigcup_{G^m} IA_{G_m}$
```

EXAMPLE 3.6. *Consider the motivating example of Section 2.1, and the second query asking for 2011 monthly sales per product per state. Suppose Algorithm 3 is called over the extensional answer and the expected extensional answer shown in Section 2.1, with* α *and* β *as low as possible. The basic intensional answer includes only 6 couples where deviation is 0, corresponding to the facts where the expected value and the actual value are the same. Then Function 4 is called and only two regions are detected as both homogeneous and with mean deviation 0, corresponding to coordinates* ⟨Ontario, Levi's, Feb.11⟩ *and* ⟨NY, CK, 2011⟩. *The intensional answer is thus the set of couples*

$$\{\langle\langle \text{Ontario, Levi's, Feb.11}\rangle, 0\rangle, \langle\langle \text{NY, CK, 2011}\rangle, 0\rangle\}$$

where 0 represents the mean deviation of the region and, in this particular case, can be interpreted as "as expected".

4. RELATED WORK

Substantial work has been conducted in order to query and explore data cubes in a more meaningful manner. This includes (but is not limited to) outlier detection in multidimensional data [11, 19, 21], cubegrade generation [10], constrained gradient analysis [6], cube mining [22, 13], cube modeling and compression [2, 3, 12, 19], and query answer approximation [2, 4, 16].

Since our framework relies both on intensional query answering (IQA) and cube modeling and approximation, this section will briefly recall the meaning of IQA as well as provide an overview about the second topic.

4.1 Intensional Query Answering

Non-conventional query answering includes a variety of answering mechanisms and happens when either the user has no clear formulation of his needs (e.g., he does not know what he really wants) or has a good understanding of his needs but is flexible enough to accept an alternate, approximate or intensional answer. This covers the production of intensional query answers, i.e., the intent behind the answer as well as other kinds of non-conventional answers. According to [15], an intensional query answer complements the extensional one by including either a concise description of the answer or some useful facts about it. Motro considers that the effectiveness of an intensional answer can be expressed through the following criteria: completeness, non-redundancy and optimality. Another kind of query answering mechanism is query relaxation which aims to relax the query conditions in order to provide a non-empty (but approximate) answer while associative query answering gives additional (and potentially useful) information about the initial query.

To illustrate intensional query answering, let us assume that the user needs to retrieve the sales of products per month and city that are higher than $100,000 in the second quarter. The intensional query answering system could inform the user that the produced extensional answer corresponds to all products of a given group sold in the city of Lyon, except product P_1.

IQA relies generally on knowledge like integrity constraints, inference rules (in knowledge-based systems), ontology (and more frequently a taxonomy), and user's preferences to either provide more insight about the extensional answer or give an approximate answer. The increasing popularity of data mining technology [9] makes it possible to exploit developed tools and techniques to generate patterns from either original data or the extensional answer to user's query. Such patterns could be association rules, classification/discrimination rules, clusters, trends, and outliers. For example, in the presence of the following implication: *if monthly sales per product and city for the second quarter are higher than $100,000, then the population of the city is more than three million persons and the product is a sport item*, then the intensional answer to the previous query is very concise and more appealing to the user than a display of many facts. It will indicate the specificity of the cities and products involved in the extensional answer.

4.2 Cube Modeling and Exploration

As pointed out earlier, some studies were concerned with cube modeling and exploration. For example, the work in [19] presents an approach based on log-linear modeling to identify exceptions in data cubes by comparing anticipated cell values against actual values while in [3], log-linear modeling is also used for data compression. In [12] the authors apply non-negative multiway array factorization (NMF) for approximating, modelling and exploring data cubes, and then compare this technique against log-linear modeling for cube approximation and compression purposes. They also show how NMF can provide a good approximation to OLAP queries by exploring the generated model rather than the actual data cube. This is particularly true for queries involving selection and/or roll-up on dimensions. The work by Xi et al. [24] presents an asymptotically lossless technique to compress data cubes to further execute OLAP queries and generate logistic regression models.

Our present work is close to the studies conducted independently by [16, 18] since cell estimation is involved in each one of the three studies (including our present work) using maximum entropy principle. Indeed, the objective in [18] is to estimate the values of non visited parts of a data cube based on the previously seen parts during an analytical session while in [16] the objective is to provide approximate answers and identify exceptions. In our case, we use the same principle for the second step of (an instance of) our framework to update the expected cube to further produce intensional answers by comparing extensional answers against expected extensional answers and computing aggregates using dimension hierarchies.

5. CONCLUSION

In this paper we have presented a framework for intensional query answering in OLAP applications that returns a concise, yet informative answer to a user's query by exploiting the queries previously formulated during the current OLAP session. In our approach, an expected cube is created and updated based on the extensional answer to each query in the current session, as a representation of the user's understanding of the data. For each new query, an intensional answer concisely characterizing the data that approximately match with the user's understanding is returned, coupled with a partial extensional answer that describes in detail the data that significantly differ from the user's understanding.

We believe that our framework is generic and flexible enough to allow different instantiations for each of the three steps (*Improve*, *Predict* and *Build*) involved in the generation of the intensional answer. For example, the *Build* step can accommodate different perspectives (i.e., different kinds of intensional answers) and use a variety of knowledge patterns other than dimension hierarchies (e.g., user's profile).

6. REFERENCES

[1] J. Aligon, M. Golfarelli, P. Marcel, S. Rizzi, and E. Turricchia. Mining preferences from OLAP query logs for proactive personalization. In *Proc. ADBIS 2011*, pages 84–97, Vienna, Austria, 2011.

[2] B. Babcock, S. Chaudhuri, and G. Das. Dynamic sample selection for approximate query processing. In *Proc. SIGMOD*, pages 539–550, San Diego, California, 2003.

[3] D. Barbara and X. Wu. Loglinear-based quasi cubes. *Journ. Intell. Inf. Syst.*, 16(3):255–276, 2001.

[4] A. Cuzzocrea and W. Wang. Approximate range-sum query answering on data cubes with probabilistic guarantees. *Journ. Intell. Inf. Syst.*, 28(2):161–197, 2007.

[5] W. Deming and F. Stephan. On a least squares adjustment of a sampled frequency table when the expected marginal total are known. *The Annals of Mathematical Statistics*, 11(4):427–444, 1940.

[6] G. Dong, J. Han, J. M. W. Lam, J. Pei, and K. Wang. Mining multi-dimensional constrained gradients in data cubes. In *Proc. VLDB*, pages 321–330, San Francisco, USA, 2001.

[7] M. Golfarelli, S. Rizzi, and P. Biondi. myOLAP: An approach to express and evaluate OLAP preferences. *IEEE Trans. Knowl. Data Eng.*, 23(7):1050–1064, 2011.

[8] J. Han. OLAP mining: Integration of OLAP with data mining. In *Proc. Conf. on Database Semantics*, pages 3–20, 1997.

[9] J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques (3rd edition)*. Morgan Kaufmann, 2011.

[10] T. Imielinski, L. Khachiyan, and A. Abdulghani. Cubegrades: Generalizing association rules. *Data Min. Knowl. Discov.*, 6(3):219–257, 2002.

[11] E. M. Knorr, R. T. Ng, and V. Tucakov. Distance-based outliers: algorithms and applications. *The VLDB Journal*, 8(3-4):237–253, 2000.

[12] R. Missaoui, C. Goutte, A. K. Choupo, and A. Boujenoui. A probabilistic model for data cube compression and query approximation. In *Proc. DOLAP*, pages 33–40, 2007.

[13] R. Missaoui and L. Kwuida. Mining triadic association rules from ternary relations. In *Proc. ICFCA*, pages 204–218, 2011.

[14] A. Motro. Intensional answers to database queries. *IEEE Trans. Knowl. Data Eng.*, 6(3):444–454, 1994.

[15] A. Motro. Cooperative database systems. *Encyclopedia of Library and Information Science*, 66:79–97, 2000.

[16] T. Palpanas, N. Koudas, and A. O. Mendelzon. Using datacube aggregates for approximate querying and deviation detection. *IEEE Trans. Knowl. Data Eng.*, 17(11):1465–1477, 2005.

[17] E. K. Park and S.-C. Yoon. An approach to intensional query answering at multiple abstraction levels using data mining approaches. In *Proc. HICSS*, 1999.

[18] S. Sarawagi. User-adaptive exploration of multidimensional data. In *Proc. VLDB*, pages 307–316, Cairo, Egypt, 2000.

[19] S. Sarawagi, R. Agrawal, and N. Megiddo. Discovery-driven exploration of OLAP data cubes. In *Proc. EDBT*, pages 168–182, London, UK, 1998. Springer-Verlag.

[20] C.-D. Shum and R. R. Muntz. An information-theoretic study on aggregate responses. In *Proc. VLDB*, pages 479–490, Los Angeles, USA, 1988.

[21] Z. Tang and R. Li. A novel regression mining algorithm based on multi-dimensional data. *Journ. of Computational Inf. Syst.*, 6(5):1459–1465, May 2010.

[22] H. C. Tjioe and D. Taniar. Mining association rules in data warehouses. *IJDWM*, 1(3):28–62, 2005.

[23] J. S. Vitter and M. Wang. Approximate computation of multidimensional aggregates of sparse data using wavelets. In *Proc. SIGMOD*, pages 193–204, Philadelphia, USA, 1999.

[24] R. Xi, N. Lin, and Y. Chen. Compression and aggregation for logistic regression analysis in data cubes. *IEEE Trans. Knowl. Data Eng.*, 21(4):479–492, 2009.