Cube query answering via the results of previous cube queries

Panos Vassiliadis
Univ. Ioannina
Ioannina, Greece
pvassil@cs.uoi.gr

ABSTRACT

In this paper, we come with a model for multidimensional spaces with hierarchically structured dimensions in several layers of abstractions, data cubes and cube queries. The model serves as the basis to offer the main contribution of this paper which includes a theorem and an algorithm for being able to decide and facilitate the computation of the contents of a new cube query from a previous one, defined at a different level of abstraction.

1 INTRODUCTION

Multidimensional spaces with hierarchically structured dimensions over several levels of abstraction, along with data cubes, (i.e., structured collections of data points at the same level of detail in the context of such spaces), provide a paradigm for data management whose simplicity is hard to match.

The problem that this paper addresses can be summarized as the principled answering of the question: how can we compute the contents of a new cube query, by reusing the existing contents of the result of a previous cube query? Traditionally, related work has handled the problem of query containment and view usability for the relational case (see [14] for the general problem of answering queries using views, a survey of aggregate query containment in [4] and two lemmas [5] and [21]); also works on Query Containment [1], [18], [7], [8]) [9]. View usability [16], [10], [14]; Query rewriting [17], [2], [13], [3], [9], [12], [11], [1], [6], [7]). However, the existence of hierarchically structured dimensions in the case of multidimensional spaces with different possible levels of aggregations as a context for the determination of cube usability, has not been extensively dealt with by the database community (see [23], [20] for two early attempts). We attempt to fill this gap by providing a comprehensive rigorous basis and the respective theorems and algorithms for being able to solve the problem of cube usability for a very powerful class of queries.

Contribution. In this paper, we start in Section 2, with a brief presentation of the core components of a model for multidimensional hierarchical spaces, cubes and cube queries. Based on the intrinsic property of the model that all query semantics are defined with respect to the most detailed level of aggregation in the hierarchical space, and in contrast to all previous models of multidimensional hierarchical spaces, in Section 3, we accompany the proposed model with definitions of equivalent expressions at different levels of granularity. We introduce the necessary terminology and notation, too, to solidify these concepts in the vocabulary of multidimensional modeling. Specifically, we introduce (a) proxies, i.e., equivalent expressions at different levels of abstraction, (b) signatures, i.e., sets of coordinates specifying a “border” in the multidimensional space that specifies a sub-space pertaining to a model’s construct, and, (c) areas, i.e., set of cells enclosed within a signature.

In Section 5, we address the usability problem of computing a new cube query $c^h$ from the cells of a previous one, $c^b$, defined at a different level of abstraction; we introduce the respective test as well as a rewriting algorithm. As a pre-requisite to address the problem, which comes with the complexity of having to deal with grouper dimensions where selections have also been posed, in Section 4 we introduce the notion of rollability which refers to the property of the combination of a filter and a grouper level at the same dimension to produce result coordinates that are fully covering the respective subspace at the most detailed level. Section 6 provides issues for future work.

We encourage the reader to refer to the long version of the paper [22] that comes with (1) a comprehensive model for hierarchical multidimensional spaces and query expressions in them (including all the typical OLAP operations), (2) more explanations, rigorous definitions and proofs for the current paper, and, (3) a principled set of tests and algorithms for the problem of cube query containment at various level of detail (specifically: foundational, same-level, and, different level containment), query intersection (at various levels of detail), and query distance.

2 FORMALIZING DATA, DIMENSION HIERARCHIES CUBES AND CUBE QUERIES

In this Section, we give the background of our modeling concerning multidimensional databases, hierarchies and queries (see [19] for a survey of models). We assume data in a multidimensional space, where dimensions provide a context for facts [15]. Each dimension comes with a hierarchy of levels. Each dimension (e.g., TaxDate) is a lattice of levels (e.g., Day, Week, Month, Year). Each level comes with a domain of values; values in different levels are mapped via an anc() function for higher levels and and desc mapping for lower levels (e.g., anc_Country[Athens] = Greece). Each dimension includes a single most detailed level, and, a single top level ALL with a single-valued domain [all].

Facts are structured in cubes. A cube is defined over the Cartesian Product of several levels from discrete dimensions along with a number of measures to hold the measurable aspects of its facts. Each cell is a point in the multidimensional space of the cube’s dimensions hosting a set of measures. A detailed cube is a cube having all its dimensions fixed at the lowest possible level.

A conjunctive selection condition is a conjunction of atoms, each of the form $D.L \in \{v_1, \ldots, v_n\}$, where the values $v_i$ belong to the domain of $L$.

The user can submit cube queries to the system. A cube query specifies (a) the detailed data set over which it is imposed, (b) the selection condition that isolates the records that qualify for further processing, (c) the aggregator levels, that determine the level of coarseness for the result, and (d) an aggregation over the measures of the underlying cube that accompanies the aggregator levels in the final result. More formally, a cube query, is an expression of the form:
A proxy is an equivalent expression at a different level of detail to a given expression. It is the set of cells that the expression produces at its own level of detail. More precisely, given an expression \( q \) over a database schema \( S \), the proxy of \( q \) at level \( L \) is the set of cells \( q(L) \) that \( q \) produces at level \( L \). A proxy is a subset of the space of all possible values that \( q \) can produce, depending on the levels of detail available. The concept of a proxy is crucial in multidimensional modeling, as it allows expressing queries at any level of detail without necessarily knowing the exact values of the data at that level. In this sense, a proxy captures the essence of the query, independent of the specific data details. This is particularly useful in scenarios where data is stored at different levels of aggregation and where querying across these levels is required.
Definition 4.2. Given a multidimensional schema $S$ and a simple selection condition $\phi$ to which it participates, a dimension $D$ with a grouper level $D.\mathcal{L}^\gamma$ at the schema level and a filter level $D.\mathcal{L}^\sigma$ at $\phi$, is characterized as follows:

- **unbound**, if $D.\mathcal{L}^\sigma = D.\mathcal{A}LL$ and the atom of $\phi$ is $D.\mathcal{A}LL \in \{\mathcal{A}ll\}$ (equiv., true)
- **pinned grouper**, if both $D.\mathcal{L}^\gamma$ and $D.\mathcal{L}^\sigma \neq D.\mathcal{A}LL$
- **pinned non-grouper**, if $D.\mathcal{L}^\gamma = D.\mathcal{A}LL$ and $D.\mathcal{L}^\sigma \neq D.\mathcal{A}LL$

**Example.** Assume a cube over Date, Education, Workclass with TaxPaid as measure. Assume that two queries roll-up Geography at level ALL, and report sales per month and product family. Let $q^\phi = \{DS^\phi, \phi^\sigma, [Month, W.\mathcal{L}1, E.\mathcal{A}LL, sum(TaxPaid), \{\text{sum(TaxPaid)}\}]\}$ be a query over this cube.

Then, Month and $W.\mathcal{L}1$ are groupers and Education is a non-grouper.

Concerning Education:

- if the atom $E.\mathcal{A}LL \in \{\mathcal{A}ll\}$ is part of $\phi$ then the dimension is a pinned non-grouper

Concerning Date:

- if the atom $Date.\mathcal{A}LL \in \{\mathcal{A}ll\}$ is part of $\phi$ then the dimension is unbound
- if an atom like $D.Year \in \{2019, 2020\}$ is part of $\phi$, then the dimension is a pinned grouper

Definition 4.3 (Perfectly Rollable Dimension / Perfectly Rollable atom). Assume a grouper level $D.\mathcal{L}^\gamma$ and an atom $\alpha:D.\mathcal{L}^\sigma \in V$, $V = \{v_1, \ldots, v_k\}$. Then, the dimension $D$ is **perfectly rollable** with respect to the tuple $(\mathcal{L}^\gamma, \mathcal{L}^\sigma, V)$, or, equivalently, $\alpha$ is **perfectly rollable** with respect to $\mathcal{L}^\gamma$, if one of the following two conditions holds:

(a) $\mathcal{L}^\gamma \leq \mathcal{L}^\sigma$ (which implies that every grouper value of $\mathcal{L}^\gamma$ that qualifies is entirely included, as the selection condition is put at a higher level that the grouping, e.g., group by month, for year = 2020)

(b) $\mathcal{L}^\sigma < \mathcal{L}^\gamma$, and for each value $v_i \in \text{dom}(\mathcal{L}^\gamma)$: $v_i = \text{and}_{\mathcal{L}^\gamma}(v_i), all \ \text{desc}^{\mathcal{L}^\gamma}_{\mathcal{L}^\sigma}(v_i) \in V$ (i.e., the entire set of children of a grouper value $v_i$ is included in the computation of $v_i$).
Definition 4.4 (Perfectly Rollable Schema / Perfectly Rollable simple selection condition). Assume a schema $S = [D_1, L_1, \ldots, D_n, L_n]$ with each group level belonging to a different dimension and a simple selection condition $\phi$. Let $\alpha_i$ be an atom of $S$ of the form $D.L \in V$, $V = \{v_1, \ldots, v_k\}$ with exactly one atom per dimension. Then, the schema $S$ is perfectly rollable with respect to the tuple $(S, \phi)$, or equivalently, $\phi$ is perfectly rollable with respect to $S$, if each atom $\alpha_i$ is perfectly rollable with respect to its respective grouper level $L_i$.

The perfectly rollable condition is a "clean" characterization stating that if we group by a level $L$ on any possible data set, then, the resulting grouper values of $L$ will be produced by the entire population of their descendants at lower levels (in fact, as far as the semantics are concerned: the most detailed one see Figure 2). Perfect rollability guarantees that, given a simple selection condition on a dimension and a grouper level, there are no grouper cells in the result of a cube that could be computed on the basis of only a subset of their detailed descendants, but rather, the entire range of descendant values are taken into consideration for their computation.

5 CUBE Usability: Computing a Cube from Another Cube Whose Result Is Available

In this section, we address the main problem for this paper: assume we have computed a previous query $q^b$ and we want (a) to check whether the contents of a new query $q^a$ are derivable from the cells of $q^b$, and (b) if this is the case indeed, to actually perform the computation. To support our discussion, in the sequel, we assume two queries, to which we refer to as $q^b$ (with the hidden implicitness of "broad" in terms of selection condition, "below" in terms of the level of the grouping, and, "before" in terms of creation) and $q^a$ (with the hidden implicitness of "narrow" in terms of selection condition, "not lower" in terms of the level of the grouping, and, "new" in terms of creation). As in all other cases, we will assume that the selection conditions are simple selection conditions, including $n$ atoms, each of the form $D.L \in \{v_1, \ldots, v_k\}$. We also assume that all aggregate functions are distributive (e.g., sum, max, min, count). To simplify the presentation even more, we assume that there is a one-to-one mapping between the measures of the two queries and the respective aggregate functions that produce them (thus, the two queries differ only at the levels of their schema and their selection conditions). See the long version of the paper [22] for the proof and the formalization of distributive functions.

Theorem 5.1 (Cube Usability). Assume the following two queries:

$$q^b = \langle DS^b, \phi^b, [L^n_1, \ldots, L^n_m, M_1, \ldots, M_m], [agg_1(M^b_1), \ldots, agg_m(M^b_m)] \rangle$$

and

$$q^a = \langle DS^a, \phi^a, [L^n_1, \ldots, L^n_m, M_1, \ldots, M_m], [agg_1(M^a_1), \ldots, agg_m(M^a_m)] \rangle$$

The query $q^a$ is usable for computing, or simply, usable for query $q^b$, meaning that Algorithm 1 correctly computes $q^a$.cells from $q^b$.cells, if the following conditions hold:

1. both queries have exactly the same underlying detailed cube DS,

Algorithm 1: Answer Cube Query from a Pre-Existing Query Result

Input: A new query expression $q^a$ and a previously computed query $q^b$ along with its result $q^b$.cells

Output: The result of $q^a$, $q^a$.cells

1 begin
2 $q^a$.cells ← compute $q^a$ and for every coordinate, create a new cell with all measures initialized to $\emptyset$
3 if $q^b$ and $q^a$ satisfy all conditions of Theorem 5.1 then
4 forall dimensions $D_i$ do
5 $\alpha_i \leftarrow$ transformed atom of the new query at the schema grouper level $L_i^b$ of $q^b$
6 $\phi^a_{\alpha_i} = \land$ $\phi_{\alpha_i}$ \(\text{cells} \leftarrow\) apply $q^a_{\alpha_i}$ to $q^b$.cells
7 $q^a_G = \text{group the cells of } q^a_{\alpha_i}.\text{cells according to } q^a$
8 forall measures $M_j$ do
9 $q^a$.cells$M_j \leftarrow$ apply $agg_j^a$ to the $j$-th measure of the members of the groups of $q^a_G$
10 end
11 return $q^a$.cells
12 end
13
(2) both queries have exactly the same dimensions in their schema and the same aggregate measures $agg_j(M^a_i)$, $i \in 1...m$ (implying a 1:1 mapping between their measures), with all $agg_j$ belonging to a set of known distributive functions. To simplify notation, we will assume that the two queries have the same measure names,

(3) both queries have exactly one atom per dimension in their selection condition, of the form $D.L \in \{v_1, \ldots, v_k\}$ and selection conditions are conjunctions of such atoms,

(4) both queries have schemata that are perfectly rollable with respect to their selection conditions, which means that grouper levels are perfectly rollable with respect to the respective atom of their dimension,

• (for convenience) for both queries, for all dimensions $D$ having $D.L^b$ as a grouper level and $D.L^a$ as the level involved in the selection condition’s atom for $D$, we assume $D.L^b \leq D.L^a$, i.e., the selection condition is defined at a higher level than the grouping,

(5) all schema levels of query $q^a$ are ancestors (i.e, equal or higher) of the respective levels of $q^b$, i.e., $D.L^b \leq D.L^a$, for all dimensions $D$, and,

(6) for every atom of $\phi^a$, say $\alpha^a$, if (i) we obtain $\alpha^a_{bob}$ (i.e., its detailed equivalent at the respective schema level of the previous query $q^b$, $L^b$) to which we simply refer as $\alpha^a_{bob}$, and, (ii) compute its signature $\alpha^a_{bob}$, then (iii) this signature is a subset of the grouper domain of the respective dimension at $q^b$ (which involves the respective atom $\alpha$ and the grouper level $L^b$), i.e., $\alpha^a_{bob} \leq \text{gdom}(\alpha^a, L^b)$.
6 CONCLUSIONS

In this paper we have provided a method for computing a new cube from a previous one, defined at a different level of abstraction. The basis of the method is perfect rollability, a property characterizing the combination of selection conditions and groupers that guarantees the correct computation of aggregate measures. Future work can also target operators comparing cubes for intrinsic properties of their cells (e.g., hidden correlations, predictions, classifications) that have to be decided via the application of knowledge extraction operators to the results, or the detailed areas of the contrasted cubes.

ACKNOWLEDGMENTS

The author has been co-financed by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call Research - Create - Innovate (prj code:T2EDK-02848).

REFERENCES


Figure 3: An example of cube usability