Chapter IX

Querying Multidimensional Data

Leonardo Tininini
Italian National Institute of Statistics, Italy

ABSTRACT

A powerful and easy-to-use querying environment is certainly one of the most important components in a multidimensional database, and its effectiveness is influenced by many other aspects, both logical (data model, integration, policy of view materialization, etc.) and physical (multidimensional or relational storage, indexes, etc.). As is evident, multidimensional querying is often based on the metaphor of the data cube and on the concepts of facts, measures, and dimensions. In contrast to conventional transactional environments, multidimensional querying is often an exploratory process, performed by navigating along the dimensions and measures, increasing/decreasing the level of detail and focusing on specific subparts of the cube that appear to be “promising” for the required information.

In this chapter we focus on the main languages proposed in the literature to express multidimensional queries, particularly those based on: (i) an algebraic approach, (ii) a declarative paradigm (calculus), and (iii) visual constructs and syntax. We analyze the problem of evaluation, i.e., the issues related to the efficient data retrieval and calculation, possibly (often necessarily) using some pre-computed data, a problem known in the literature as the problem of rewriting a query using views. We also illustrate the use of particular index structures to speed up the query evaluation process.
INTRODUCTION

As shown in the previous chapters, multidimensional data modeling is based on the metaphor of the *data cube* and on the concepts of *facts, measures, and dimensions*. Analogously, the techniques to retrieve such data, which have been proposed in the literature and/or implemented in commercial systems, are based on the idea of determining the cube of interest and then navigating along the dimensions, by increasing or decreasing the level of detail (through the well-known operations of *roll-up* and *drill-down*) or selecting specific subparts of the cube (through the operations of *slice and dice*).

The query languages for multidimensional data support both these standard and additional operations for performance of more sophisticated computations. As is common in the literature, we distinguish among:

(i) languages based on an algebra (usually an extension of the relational one), where queries are expressed by using operators that apply to the tables representing the facts, measures, and dimensions;
(ii) languages based on a calculus (again usually an extension of the relational one), where queries are expressed in a more declarative way;
(iii) visual languages, usually relying on an underlying algebra, and based on a more interactive and iconic querying paradigm; this is the approach of most commercial OLAP products.

We analyze the characteristics of the main query languages proposed in the literature, along with the specific advantages and drawbacks, also emphasizing the common features. Particularly, we consider:

(i) query languages based on a relational representation of multidimensional data, hence based on extensions of the relational algebra and calculus;
(ii) query languages based on specifically designed multidimensional models, usually based on an abstraction of cubes or fact tables, on which the operators of the algebra are applied.

In the case of query languages based on specific models, we show that, while typical OLAP operations are expressed in a very straightforward manner, the expression of typical relational operation can become cumbersome, as well as the capability to *symmetrically deal with dimensions and measures*. As a consequence, several studies have been focused on minimizing the “impedance mismatch” between relational and multidimensional models.

We also focus on the problem of query evaluation, i.e., on how the query expressed in the chosen language can be translated into an efficient evaluation plan, which retrieves the necessary information and computes the required results. As already stressed in the previous chapter, the choice of a collection of pre-computed results (materialized views) has dramatic consequences on the overall performances of the query evaluation process.