Ad Hoc OLAP : Expression and Evaluation

Damianos Chatziantoniou
Department of Computer Science , Stevens Institute of Technology
damianos@cs.stevens-tech.edu

Abstract

Users frequently formulate complex data analysis queries in order to identify interesting trends, make unusual patterns stand out, or verify hypotheses. Being able to express these data mining queries concisely is of major importance not only from the user's, but also from the system's point of view. Recent research in OLAP has focused on datacubes and their applications; however, expression and processing of ad hoc decision support queries has been given very little attention. In this paper we present an appropriate framework for these queries and introduce a syntactic construct to support it. This SQL extension allows most OLAP queries, such as pivoting, complex intra- and inter-group comparisons, trends and hierarchical comparisons, to be expressed in a compact, intuitive and simple manner. This succinct representation of a complex OLAP query translates immediately *to a novel, simple and efficient evaluation algorithm. We show how to optimize, analyze and parallelize this algorithm and discuss issues such as multiple query analysis and scaling. We present several experimental results of real-life queries that show orders of magnitude of performance improvement. We argue that this tight coupling between representation and algorithm is essential to efficient processing of ad hoc OLAP queries.*

1. Brief Description

Although significant research has been conducted on datacubes, both in terms of modeling and evaluation, little has been done on query optimization of complex *ad hoc* decision support queries, despite of their importance. To express such queries in SQL, a high degree of redundancy is required: multiple self-joins, correlated subqueries and repeated group-bys. Consider a representative OLAP query over the relation *Sales(cust, prod, day, month, year, sale)*:

"For each product and for each month of 1997, show the product's average sale *before* this month, *during* this month, and *after* this month."

With this query, one can identify those months of 1997 that were "significant" for the sales of a product. The main idea of this query is the following: *for each* value (p, m) of `(prod, month)` attributes we want to define two subsets of `Sales`, $X_{(p,m)}$ and $Y_{(p,m)}$, where the first subset $X_{(p,m)}$ contains the sales of product p prior to month m, and the second subset $Y_{(p,m)}$ contains the sales of product p following month m. Then we want to compute the average quantity of $X_{(p,m)}$ and $Y_{(p,m)}$. We believe that the ability to iterate over the values of one or more attributes domain, coupled with the ability to define for each such value one or more "interesting" subsets of the relation constitutes the gist of complex data analysis. The challenge is to provide the user with this "looping" ability without sacrificing the declarativeness of SQL. To achieve that we extend the syntax presented in [1]. The user can define *for each* group several grouping variables (tuple variables that range over the *entire* table). These are declared in the `group by` clause, separated by the grouping attributes with a *semicolon*. Their range is defined in the `such that` clause. These queries are called extended multi-feature (EMF) queries. The previous query can be expressed in the EMF syntax as:

select prod,month,avg(X.sale),avg(sale),avg(Y.sale)
from Sales
where year="1997"
group by prod, month; X , Y
such that X.prod=prod and X.month<month,
Y.prod=prod and Y.month>month

Our implementation takes advantage of the particular structure of our extended SQL queries. The algorithm scans one or more times the base relation and for each scanned tuple t, the rows of the resulting table affected by t are identified and updated appropriately. Our optimization techniques try to (i) reduce the number of scans, (ii) represent the resulting table optimally by an appropriate data structure, and (iii) parallelize the evaluation.

References

[1] D. Chatziantoniou and K. Ross. Querying Multiple Features of Groups in Relational Databases. In *22nd VLDB Conference*, pages 295–306, 1996.