The Cubetree Storage Organization

Nick Roussopoulos
Advanced Communication Technology, Inc.*
nick@act-us.com

Yannis Kotidis
Advanced Communication Technology, Inc.

Abstract

The *Cubetree Storage Organization (CSO)*[1] logically and physically clusters materialized-views data, multi-dimensional indices on them, and computed aggregate values all in one compact and tight storage structure that uses a fraction of the conventional table-based space. This is a breakthrough technology for storing and accessing multi-dimensional data in terms of storage reduction, query performance and incremental bulk update speed.

1 Introduction

The Relational On-Line Analytical Processing (ROLAP) is emerging as the dominant approach in data warehousing. In order to enhance query performance, the ROLAP approach relies on selecting and materializing in summary tables appropriate subsets of aggregate views which are then engaged in speeding up OLAP queries. However, a straight forward relational storage implementation of materialized ROLAP views is immensely wasteful on storage and incredibly inadequate on query performance and incremental update speed.

Cubetrees are collections of packed and compressed R-trees and shown to be the best alternative for storing and indexing at the same time collections of ROLAP aggregate views. The *Cubetree Storage Organization (CSO)* logically and physically clusters materialized-views data, e.g. summary tables, multi-dimensional indices on them, and computed aggregate values all in one compact and tight storage structure that uses a fraction of the conventional table-based space. The physical clustering and the ordering within the CSO permits sequential read and write operations during the creation and bulk incremental update in a *merge-pack* type of operation. It is this sequential I/O that utilizes the transfer rate of the disks and achieves maximum performance for both updates and queries.

CSO offers a breakthrough technology for storing and accessing derived materialized views. Each Cubetree structure is not attached to a specific materialized view or an index but acts like a *place holder* which stores multiple and possibly unrelated views and multi-dimensional indices without having to analyze and identify the dense and sparse area of the underlying multi-dimensional data. In CSO data is clustered regardless of the skewedness underlying data. In an extensive test using the TCP-D benchmark data the CSO implementation achieved at least a 2-1 storage reduction, a 10-1 better OLAP query performance, and a 100-1 faster updates over the conventional (relational) storage organization of materialized OLAP views indexed in the best possible way.

The CSO implementation is offered in two configurations: a) as a Cubetree Datablade developed to run with the Informix Universal Server (IUS) and b) as a stand alone library. CSO is invoked through a Cubetree-SQL or a GUI interface. Both interfaces allow a user to define the set of views to be queried and optimized in the data warehouse.

The CSO optimizer selects the best set of views to materialize and the best placement on one or more cubetrees for achieving maximum clustering. The underlying cost model is tailored to the CSO and uses standard greedy algorithms for selecting the optimal configuration. The DBA can tune the optimization to upper-bound either the duration of the incremental update window or the maximum storage used.

*1209 Goth Lane, Silver Spring, MD 20905, Tel: 301-384-3759, Fax: 301-384-3679

Proceedings of the 24th VLDB Conference
New York, USA, 1998

[1] US Patent Pending