December 2002

A PARADIGM SHIFT IN DATABASE OPTIMIZATION: FROM INDICES TO AGGREGATES

Ryan LaBrie
Arizona State University

Lin Ye
Arizona State University

A PARADIGM SHIFT IN DATABASE OPTIMIZATION: FROM INDICES TO AGGREGATES

Ryan LaBrie, Robert St. Louis, and Lin Ye
Arizona State University
ryan.labrie@asu.edu st.louis@asu.edu lin.ye@asu.edu

Abstract

Multidimensional databases and online analytical processing (OLAP) tools provide new ways for decision-makers to access data and retrieve information. This paper examines the differences between the optimization techniques that database designers need to consider when developing relational versus multidimensional data warehouses. The multidimensional data storage model allows for large numbers of aggregates to be stored in a very efficient and accessible manner. These aggregates make it possible to not only access information faster, but also at a lower cost in terms of CPU, I/O, and disk space utilization. This research-in-progress demonstrates those speed and cost savings using the TPC-H decision support benchmark. This publicly available dataset is used to contrast the use of indices in its relational implementation with the use of aggregates in its dimensional implementation. These contrasts show why database designers must shift from the index paradigm for relational databases to the aggregate paradigm for dimensional databases.

Keywords: Aggregates, data marts, indices, multidimensional databases, OLAP

Introduction

When database designers begin the process of tuning a relational database, they usually turn to indices and storage structures for help. Optimization, while only playing a minor part in many database management textbooks and course offerings, is a critical concern to every business that depends upon a database for its operations. Expert database designers know where to put indexes to increase speed (e.g. on primary keys, on sorted or group by fields, on fields that are frequently used for selection criteria, etc.). Furthermore advanced techniques for data storage have emerged that include specifying where to store data on hard disks for optimal retrieval. Disks striping and distributed techniques are also available for parallel processing and I/O access. While these strategies and techniques have worked well for relation databases, they are not the most powerful techniques for a multidimensional setting.

The key to optimizing multidimensional databases, including data marts, is identifying and implementing appropriate aggregates. Just as determining the fields on which to place indices is based on usage in a relational setting, determining the correct data to aggregate is based on usage in the dimensional setting. This change in focus from indices to aggregates is so important that we believe it constitutes a paradigm shift. Kuhn (1962, p. 10) defines a paradigm as "some accepted examples of actual scientific practices … from which spring particular coherent traditions of scientific research." From its beginnings with Codd's work on the relational database model (Codd, 1970) and Chen's Entity Relationship Modeling (Chen, 1976), the study of relational database management systems progressed in much the same way as what Kuhn describes as 'normal science'. Kuhn notes that a paradigm shift usually occurs in the midst of anomalies and the emergence of scientific discoveries.

The development of the multidimensional data store was a scientific discovery aimed at solving the problem of data access for very large collections of data. This new way of storing and retrieving information has important implications for database designers, and leads to our current research question, *"What are the most powerful techniques for optimizing multidimensional data marts?"* The remainder of this paper is organized as follows: Section 2 presents a brief literature review. Section 3 discusses the database that is used to contrast the optimization techniques for relational and dimensional databases. Section 4 provides a discussion of the initial and expected results. The paper closes with a discussion of limitations and future research.

Literature Review

Cognitive psychologists have been studying decision making for over 50 years. Edwards (1954, 1961) laid the initial groundwork by putting forth a model for behavioral decision theory. Recently, Edwards and a colleague (Edwards & Fasolo, 2001) adapted this theory to technology. They outline a 19-step process of normative decision-making. Normative decision-making differs from descriptive decision-making in its reliance on information aides (such as information systems). In this 19-step process, steps 6 and 13 both refer to the need for aggregating results.

Additionally, several database researchers and practioners have stressed the importance of aggregates. Kimball devotes an entire chapter to the subject and makes the assertion that "The use of prestored summaries (aggregates) is the single most effective tool the data warehouse designer has to control performance" (1996, p.190). Inmon (1996) refers to aggregates as "profile records" and suggests that their benefits include organizing data in a compact and convenient form for the end user to access and analyze. Gray & Watson (1998) discuss the importance of "summary data" in terms of lightly summarized and heavily summarized data. They too suggest that storing highly summarized data improves response times.

TPC-H Database

To illustrate the differences between optimizing for relational decision support systems and optimizing for multidimensional data marts, we are in the initial stages of developing a multidimensional database benchmark based on the Transaction Processing Councils TPC-H relational decision support benchmark. We chose this relational benchmark to start with for a number of reasons. First it provides a vendor neutral (both hardware and software) benchmark. Second, the TPC-H benchmark provides standardized summary metrics with which to compare systems. Third, the data, the queries used in the benchmarking system, and their results, are freely available for validation and replication of results. Benchmarking has been used in prior research in the information systems domain to gage the performance of databases, development methods, and networks (Dey & Seidmann, 1994; Chao & Hsin, 1989; Johnson & Gray, 1993).

The TPC-H benchmark is a set of 8 related tables that can be used to generate datasets from 1GB to 3TBs in size. Figure 1 below shows the TPC-H database schema.

The TPC-H benchmark was developed as an ad-hoc decision support benchmark; that is, its data and the accompanying queries were developed to closely mimic real world business scenarios. The two performance metrics reported by the TPC-H are the Composite Query-per-Hour Performance Metric, expressed as QphH@Size, and the Price/Performance Metric, expressed as $/QphH@Size.

On our benchmarking system we created both a 1GB and a 10GB TPC-H data set. We have run the 22 standardized decision support queries and obtain base metrics for our system. In our system the QphH@1 returned was 82.9, while our $/QphH@1 equaled 12.1. While these numbers are not going to rival the top 10 systems reported by the TPC, they do give us a base comparison metric to compare with once the multidimensional benchmark is complete. The system configuration used in this research included an Intel-based PC using a 1.8GHz Pentium 4, 768MB RAM, and an 80GB 7200 rpm hard disk. The software included Microsoft Windows 2000 Advanced Server, Microsoft SQL Server 2000 Enterprise Edition, and Microsoft SQL Server 2000 Analysis Services.

After the TPC-H relational database was loaded onto our system, benchmarked, and the query results verified to be accurate, we began the task of developing an equivalent dimensional data warehouse and accompanying MDX queries. To date, we have shown using TPC-H query number four (see Table 1 below) that we can build a multidimensional data mart, which can be asked the same question, albeit in a different (MDX) format, and produce the same exact results. Figure 2 below shows the dimensional model of our multidimensional data mart created to accommodate this query. Table 1 below shows both the relational SQL for TPC-H query 4 and the equivalent MDX code for returning the same information from our multidimensional data mart.

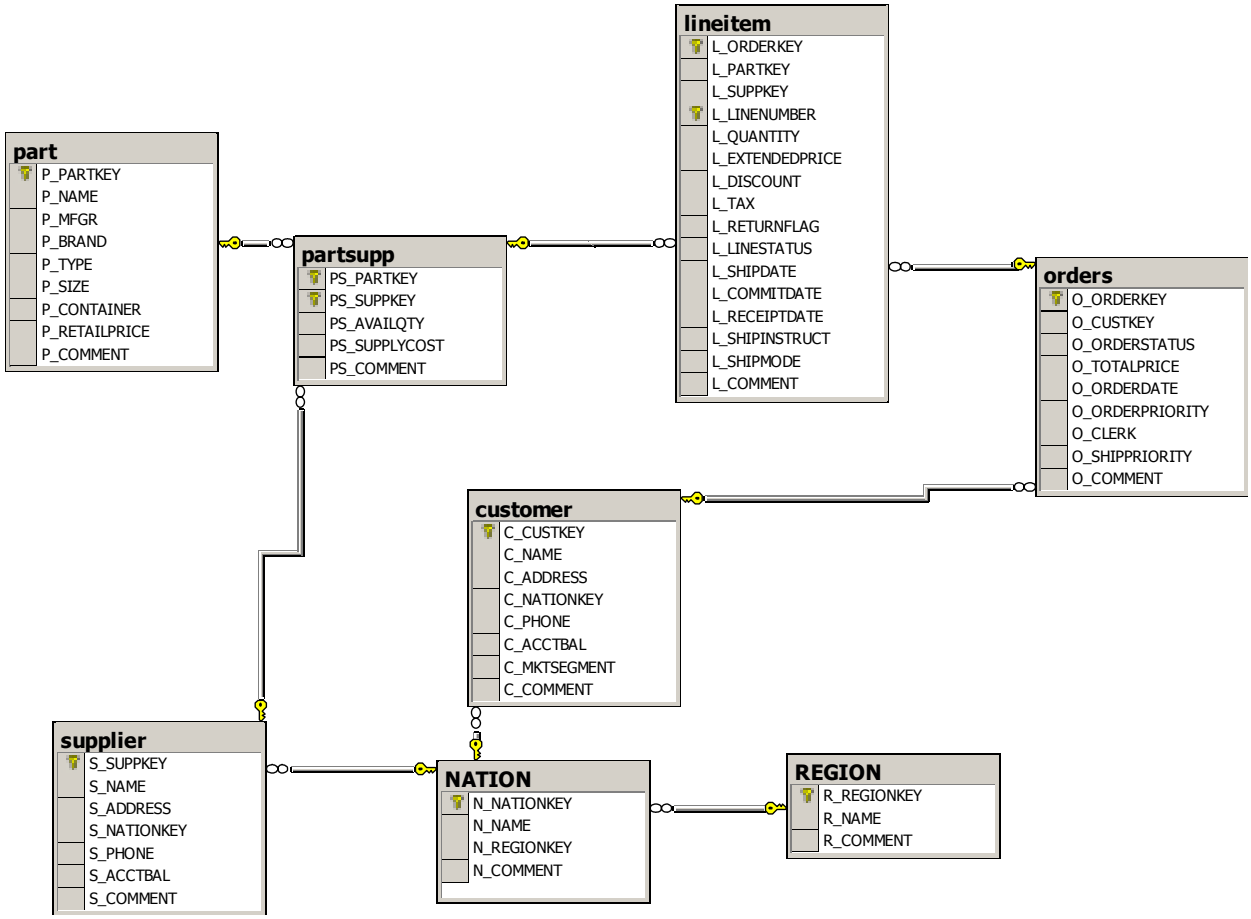

Figure 1. TPC-H Database Schema

Figure 2. Dimensional Model for Handling TPC-H Query 4 Equivalent

Table 1. SQL Query and Equivalent MDX Code

TPC-H Query 4 (relational SQL)	Equivalent Query 4 (multidimensional MDX)
SELECT o_orderpriority, COUNT(*) AS order_count FROM orders WHERE o_orderdate >= '1993-07-01' AND o_orderdate < '1993-10-01' AND EXISTS (SELECT * FROM lineitem WHERE l_orderkey = o_orderkey AND l_commitdate < l_receiptdate) GROUP BY o_orderpriority ORDER BY o_orderpriority	SELECT {[Measures].[FACT Latecount]} ON COLUMNS, {[PriorityDim].[All PriorityDim].[1], [PriorityDim].[All PriorityDim].[2], [PriorityDim].[All PriorityDim].[3], [PriorityDim].[All PriorityDim].[4], [PriorityDim].[All PriorityDim].[5]} ON ROWS FROM Q4Cube WHERE ([TimeDim].[All TimeDim].[1993].[Quarter 3])

Results

Table 2 below shows the results of running TPC-H Query 4 on a the relational data warehouse using SQL, and the results of running the same query on a dimensional data mart using MDX with an aggregate. Note that the MDX query ran 141 times faster than the SQL query. These results are biased in that no indices were used for the SQL query. However, it is clear that no amount of indexing will enable the SQL query to run as quickly as the MDX query.

We currently are building a 10-gigabyte version of the TPC-H data warehouse. When that is completed, we will run TPC-H Query 4 against this expanded data warehouse. We also will fully index the SQL queries. These results should make Table 2 even more interesting, and convince even the most hardened skeptic that aggregates are the most powerful technique for tuning dimensional data marts. Database designers must make the shift from indices to aggregates.

Table 2. Initial Results Relational Versus Multidimensional

Measure	Relational Model (SQL)	Multidimensional Model (MDX)
Query Response Time (Speed)	46.6 seconds	0.33 seconds

Conclusion

Tuning is critical to the performance of all databases. Optimizing techniques for relational database management systems have for years focused on indices and storage structures. While these techniques are powerful for the relational model, they are not the most powerful techniques for the multidimensional model. It has been suggested in the practitioner literature that aggregates are the single most effective tool a data warehouse developer has to increase performance. This research provides preliminary evidence in support of that assertion. These results are directly related to the capabilities of multidimensional databases to store aggregates efficiently and access them effectively.

This research-in-progress, like all research, has several limitations. First we are just in the initial stages of our benchmark development and have only converted a small number of queries. The TPC, in its development of the TPC-H decision support benchmark, attempted to provide a representative sample of the types of queries that might be used in business. Our results need to be replicated for this larger set of queries.

While this research is in the very initial stages of progress, we do have quite an elaborate set of future research plans. First, we hope to extend our testing to additional data sets to match the TPC-H's data sets (100GB, 300GB, 1TB, 3TB). This research will be performed to see if there are any performance gains or degradation at larger sizes. Analysis on multiple platforms such as Oracle Express and IBM's DB2 OLAP Server also will be conducted. Currently there are no benchmarks for multidimensional engines.

And finally, we want to explore whether the use of aggregates can improve decision-making. The mere fact that aggregates have been created may cause end-users to ask questions and explore alternatives that otherwise would not have been explored. If that is the case, speed will be only one of several reasons, and not necessarily the most important one, for using aggregates.

References

Chao, H.C. and Hsin, H.L. "Evaluating Microcomputer DBMSs," *Journal of Information Systems Management* (6:2), Spring 1989, pp. 69-75.

Chen, P.P. "The Entity-Relationship Model – Toward a Unified View of Data," *ACM Transactions on Database Systems* (1:1), March 1976, pp. 9-36.

Codd, E.F. "A Relational Model of Data for Large Shared Data Banks," *Communications of the ACM* (13:6), June 1970, pp. 377-387.

Dey, D. and Seidmann, A. "Benchmarking decision models for database management systems," *Information Systems Research* (5:3), September 1994, pp. 275-293.

Edwards, W. "The Theory of Decision Making," *Psychological Bulletin* (51), 1954, pp. 380-417.

Edwards, W. "Behavioral Decision Theory," *Annual Review of Psychology* (12), 1961, pp. 473-498.

Edwards, W. and Fasolo, B. "Decision Technology," *Annual Review of Psychology* (52), 2001, pp. 581-606.

Gray, P. and Watson, H.J. *Decision Support in the Data Warehouse*, Upper Saddle River, NJ: Prentice Hall, 1998.

Inmon, W.H. *Building the Data Warehouse, 2nd Ed.,* Redding, MA: John Wiley & Sons, 1996.

Johnson, J. and Gray, J. "Benchmarks Help with Server Platform Buys," *Software Magazine* (13:13), September 1993, pp. 93-100.

Kimball, R. *The Data Warehouse Toolkit,* New York, NY: Wiley Computer Publishing, 1996.

Kuhn, T.S. *The Structure of Scientific Revolutions*, Chicago, Ill: University of Chicago Press, 1962.

Transaction Processing Performance Council, http://www.tpc.org.