This white paper will detail the requirements for operational business intelligence, and will show how HyperRoll Relational can dramatically streamline and enhance the performance of existing Business Objects, Cognos, MicroStrategy and other relational data warehouse BI implementations.

THE CHALLENGE

IT organizations are struggling to meet the demand from end users for near-instantaneous query response. BI applications based on relational data warehouse architectures are not able to consistently satisfy these demands and struggle to handle concurrent user loads at peak periods.

Despite advantages over other approaches, relational data warehouse platforms suffer from several well-known challenges: long-running queries, exploding data volumes, back-end maintenance costs and lengthening batch windows all contribute to the sleepless nights of data warehouse professionals everywhere. The shift towards operational BI applications deployed on the front line of the business has put even more pressure on existing relational data warehouse implementations to deliver performance. These applications include operational dashboards, BAM (business activity monitoring) projects, and web-based ad-hoc reporting applications deployed to large numbers of users. An even newer breed of management by exception applications involve rapid root-cause analysis by potentially thousands of users who need on-demand access to the latest data.

APPLYING REAL-TIME ANALYTICS

Real-time analytic solutions can be hindered by latency of data warehouse or operational data store

Meta Group 2003
THE RISE OF OPERATIONAL BI

As BI makes the move from the analyst’s cube to the branch office, the call center, and even a customer’s own web browser, users refuse to tolerate queries that don’t return immediate results. Unlike 90’s era executive information systems servicing a few knowledge workers back at corporate, the new breed of business intelligence applications are targeted at far larger groups of line employees using thin-client web interfaces. Dashboards are often the interface of choice, featuring gauges, dials and indicators that reflect up-to-date information. Drilling into these objects should return detailed results without delay. Once presented with results, users may need to add information to reports by “drilling across” to other dimensions. More advanced users typically desire to add new metrics or on-the-fly calculations.

Transactional applications also require dynamic, real-time access via web services to historical data stored in relational data warehouses and marts, for example to approve an on-line credit application or to analyze the past purchasing behavior of an on-line customer. These type of “in-process” requests require that relational data warehouse environments provide lightning-fast query response.

“Our research indicates that enterprises increasingly are recognizing the limitations of standalone query and reporting solutions. This is particularly true for businesses attempting to sub-optimize business process performance and respond to the imminent real-time monitoring requirements of Sarbanes-Oxley. As organizations introduce lower-latency analytic solutions that are embedded into core business applications and accessing data where it lives, it’s no surprise that they find the infrastructure performance demands to be overwhelming.”

Doug Laney
Vice President
Enterprise Analytic Strategies
META Group

THE NEED FOR PERFORMANCE INFRASTRUCTURE

While relational data warehouse architectures have many advantages – excellent data breadth and depth, leverage of existing database skills, and a broad range of access methods, to name just a few – the fact remains that RDBMSs were never originally designed to support complex business intelligence applications. The star schema typical in many relational data warehouse scenarios was conceived as a physical representation of a logical multidimensional structure, one that was originally implemented in non-relational BI systems. The addition of snowflake dimensions and dozens of aggregate tables at each level of the dimensional hierarchy requires the database to execute hundreds of costly SQL table joins when processing analytical queries.

Data warehouse managers and DBAs are also forced to make difficult choices about how much data to pre-compute and pre-aggregate in the relational data model, decisions that have critical implications on data storage, maintenance and extending the batch window. IT managers in charge of relational data warehouses are caught in a difficult catch 22: they cannot possibly predict all conceivable query permutations that might be executed by end-users, thus system performance suffers; on the other hand, building aggregate tables and pre-calculated metrics for even just the known query combinations would quickly result in a storage and maintenance nightmare, not to mention extend the batch window.
QUERY PERFORMANCE BAND-AIDS

Faced with these and other challenges, BI managers have resorted to all manner of stop-gap measures to coax performance out of relational data warehouse systems, in many cases entirely shielding the environment from user behaviors that could cause query performance problems and degrade the integrity of the database.

New Server Hardware
A "quick fix" approach to relational data warehouse performance issues has been to throw more hardware at the problem. Rare is the CIO who hasn’t signed off on purchase reqs for server upgrades, clusters, more memory, more CPU’s and more storage arrays to bolster BI infrastructure. Unfortunately, RDBMS platforms only provide near-linear scalability improvements in transactional application environments. Business intelligence applications relying on RDBMS infrastructure perform only incrementally better when more or newer hardware is introduced (given the inherent limitations of SQL joins as a method for complex analytical operations). Clearly the added costs of capital equipment acquisition don’t yield the exponential performance improvements required by today’s operational BI applications.

Query Governing
The simplest way to protect against long-running queries is to prohibit them entirely before they are launched. Most relational BI vendors offer user and role-based governing mechanisms to limit the time that a query can run as well as how many rows it is allowed to return. More advanced systems feature heuristics-based algorithms that predict query response time using the historical performance statistics of queries run in the past. Other strategies include limitations on a user’s ability to drill down below a certain level in the hierarchy. These tactics may protect database performance integrity, but they raise the frustration-level of end users who can’t get answers to critical business questions.

Report Caching & Broadcasting
A strategy that has become increasingly popular is to run long-running reports at night or whenever users are not accessing the system and then cache them in the file system or on a reporting server. When users access the reports they receive the cached version without delay and without executing a new query against the RDBMS. Business Intelligence vendors have built entire product lines around the notion of broadcasting cached reports to users via email and in a variety of formats. While caching may move processing to off-peak periods, global organizations servicing geographically-dispersed users 24/7 are finding it increasingly difficult to allocate sufficient blocks of time to process these reports without monopolizing system resources. Report caching effectively lengthens the batch window – the time it takes to conduct ETL processes and build aggregates – and therefore cuts down on the "analysis window". Cached reports are also by definition "stale" since they contain hours or days-old data depending on when they were processed. Finally, cached reports are "canned" and have limited benefit in an environment where ad-hoc, on-demand reporting is required.

Summary Tables
Summary or aggregate tables are probably the most widely used strategy to improve query performance in relational data warehouse environments. Also referred to as Materialized Views in Oracle, summary tables generally aggregate data for a single measure and for no more than two dimension attributes. A moderately complex relational schema can require literally hundreds of summary tables to address only the most commonly accessed query paths. Building, loading, incrementally updating and allocating storage space for so many tables can quickly overload well-staffed data warehouse teams. Even with data summarized at a high level, using SQL to execute joins on multiple aggregate tables with three or four dimension tables can cause tangible delays in query execution, depending on the concurrent database load and the availability of temp table space. Anecdotal evidence suggests that organizations only build a limited number of summary tables that cover a very small percentage of all possible user requests; the maintenance burden introduced by even several dozen summary tables quickly outweighs their incremental benefit.
“DBA’s often get themselves into trouble because performance trade-offs appropriate at one level of activity often become extremely inappropriate down the road at greater activity levels. And for many companies, that greater level of activity is now! So lots of them are scrambling around for ways to improve performance without doing a yank and replace.”

- Eric Rogge
VP and Research Director
for Ventana Research

USING HYPERROLL TO IMPROVE PERFORMANCE WHILE REDUCING COSTS

Organizations deploying large-scale relational data warehouses need to address critical performance and maintenance challenges with a holistic and reliable approach that can yield 10s to 100s of times improvements in performance, as well as a significant reduction in maintenance and storage requirements. Data warehouse teams are looking for a way to once and for all eliminate the aforementioned stop-gap measures that are expensive, time-consuming and yield only incremental benefit.

HyperRoll has introduced the first software-based data aggregation layer for relational data warehouse environments. HyperRoll Relational fits seamlessly into existing environments and requires no change to end user applications.
REDUCING MAINTENANCE AND IMPROVING MANAGEABILITY

HyperRoll effectively replaces summary (aggregate) tables in a relational database environment with a single aggregation object that is visible as a view in the RDBMS system catalog. Unlike summary tables, HyperRoll can aggregate:

- Many dimensions and measures
- 10s of billions of rows of detail data
- Data at all levels of each dimension hierarchy
- Incrementally, eliminating the need to rebuild the entire aggregation
- In a fraction of the time it takes to build summary tables
- Using a fraction of the storage space of summary tables

SEAMLESS INTEGRATION IN RELATIONAL ENVIRONMENTS

HyperRoll is non-invasive and requires no changes to existing relational data warehouse architectures or applications. In fact, HyperRoll is completely transparent to end users who continue to access the relational environment using the tools and interfaces they are familiar with.

MAKING OPERATIONAL BI A REALITY

HyperRoll delivers exponential (vs. linear) improvements in query performance, making front-line BI applications highly performant in relational data warehouse environments. Organizations can finally deploy operational dashboards that instantly display requests for detailed data when users click on top-level gauges and indicators.

SHORTENING BATCH WINDOWS

Global 2000 companies use HyperRoll to reduce their data warehouse batch window from hours to minutes, making intra-day refreshes finally a reality. Because of HyperRoll’s advanced data aggregation algorithms, data can be made available to users in a fraction of the time it takes to build and load individual aggregate tables.

TURN THE DATA WAREHOUSE INTO A STRATEGIC, CORPORATE-WIDE ASSET

HyperRoll enables a data warehouse that is servicing one or two applications to be suddenly leveraged by dozens of different end user constituencies, without the addition of costly hardware or additional database licenses. In addition to stunning gains in query performance, HyperRoll allows exponentially larger numbers of concurrent users to access a single RDBMS instance, with absolutely no performance degradation.