**RDBMSs have always turned a fairly blind eye to the kind of relationships important to online analytic processing. That's about to change thanks to new features coming to DB2.**


Photography by GettyOne

# THE OLAP-Aware Database

MICHAEL L. GONZALES AND GARY ROBINSON

Online analytic processing (OLAP) is a core component of business intelligence (BI). OLAP gives users the ability to interrogate data by intuitively navigating from summary to detail data. All OLAP solutions rely on a relational database management system (RDBMS) to source and dynamically query data and to support drill-through reports. But RDBMSs can no longer remain passive data repositories for multidimensional applications. Modern BI requires a more active role: The RDBMS must be optimized to support OLAP, and it must be integrated with the centralized, cohesive, and consistent control of multidimensional data across the enterprise.

The next generation of OLAP support in DB2 Universal Database (UDB) will include features and functions that make the relational database a first-class platform for managing and deploying multidimensional data across the enterprise. Armed with these facilities, architects can provide OLAP solutions that can be deployed faster, are easier to manage, and improve performance across the spectrum of analytical applications regardless of the particular OLAP tools and technologies used.

We'll explain each of the key features that will enhance the OLAP support in DB2, including:

- OLAP awareness, based on multidimensional metadata in the database catalogs
- The ability to optimize for OLAP-style data access
- New interfaces that support OLAP-style Web services against DB2.

## OLAP AWARENESS

RDBMSs play a key role in OLAP solutions; however, their awareness of OLAP is limited. Database catalogs are an example of this limitation: Catalogs provide basic information about the atomic entities the RDBMS manages (for example, tables and columns). But they hold little information about the relationships between tables and no information about how tables and columns are related to the business entities or how dimensions, such as customer and product, are related — critical information from an OLAP standpoint.

To make the database aware of the higher-level data organization OLAP requires, the database catalogs need a set of higher-level objects that relate directly to OLAP and business models. In effect, these objects will take the existing atomic entities and compound them to make dimensional entities, such as attributes, facts, relationships, hierarchies, and dimensions. Figure 1 shows objects that reflect a complete dimensional model based on the underlying star schema.

Once these high-level objects are defined, the new information can be stored and managed as part of the database catalogs. This catalog information then becomes available to the database engine and optimizer; consequently, the RDBMS becomes aware of the higher-level structures and subordinate atomic-level data.

But the value of this metadata goes beyond the RDBMS. The simple DB2 interfaces that tools and applications use to access this information let dimensional intelligence be exchanged from back-end tools through the database to front-end applications (see Figure 2). Dimensions, hierarchies, attributes, and facts are described once and made available to all tools and applications that need the metadata.

This approach makes the metadata valuable to the widest possible range of BI tools and applications. OLAP tools can now learn all they need to know about the data model by querying the database. Reporting tools benefit by understanding attributes, dependencies, and relationships. Even data mining and spatial analysis tools benefit. The result is faster deployment and easier management of BI and OLAP applications based on the end-to-end flow of dimensional metadata.

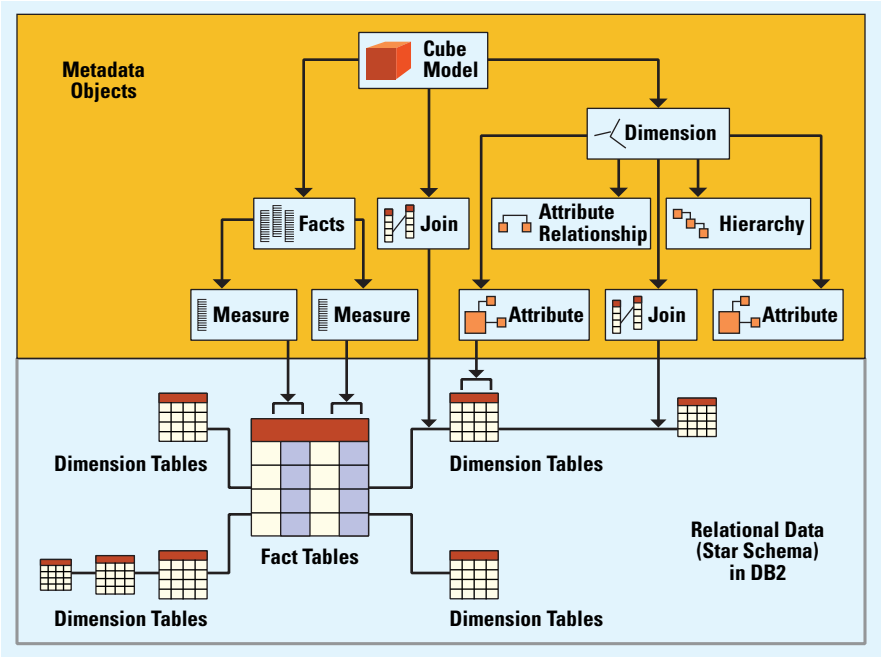


FIGURE 1: *OLAP objects built on the DB2 star schema.*

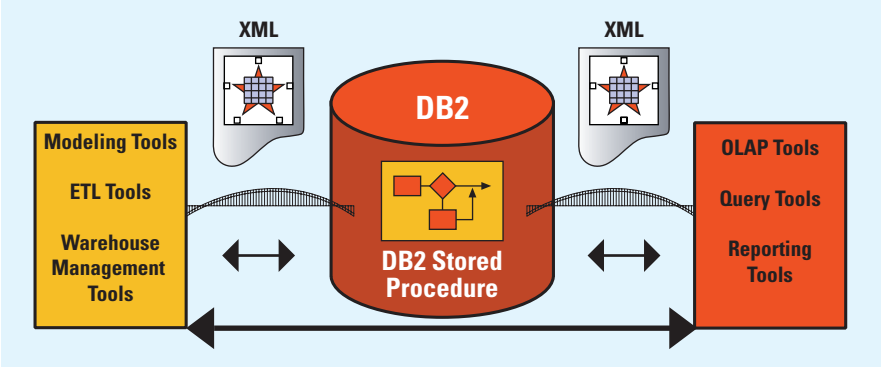


FIGURE 2: *DB2 interfaces enabling an end-to-end metadata flow.*

## OPTIMIZING FOR OLAP

Managing multidimensional data efficiently requires the ability to create and manage aggregates and index data across multiple dimensions.

Aggregate management is particularly challenging because storage requirements explode when you combine measures across multiple dimensions. Let's assume you want to model sales by product by customer by channel over time for plan and actual results. To aggregate all combinations across all dimensions to provide weekly data for two years across 1,000 products sold to

500,000 customers through four channels, you'd create a multidimensional space of 104×1,000×500,000×4=208,000,000,000 sales figures. OLAP models are rarely this simple; they usually include more dimensions and require additional storage for aggregates such as months, quarters, years, product groups, and sales regions. The space and time required to build such aggregates makes aggregation strategies impractical.

The alternative to aggregating everything is to choose where you build aggregates within the multidimensional space. Although the complete multidimensional space still contains the same number of total values, only some of the values are pre-aggregated and stored; the rest are aggregated on demand. Deciding which values to pre-aggregate isn't easy. You want to make the best use of the available time and resources by building cost-effective aggregates. To do this, you must understand the dimensional model, the nature of the data, and the access patterns. Cost/benefit calculations that consider the cost to build, the space the aggregates will consume, and the benefit they'll yield may help. The cost/benefit analysis will help determine which slices of the multidimensional model will be pre-aggregated and stored and which will be computed on demand. Some incoming queries will directly correspond to pre-aggregated stored values; others can be quickly derived from existing partial aggregates. In both cases, faster queries result.

DB2 already provides materialized query tables, analytic functions, and OLAP operators (including GROUPING SETS, CUBE, and ROLLUP) that can provide cost-effective aggregates to dramatically improve OLAP performance. To use these features in designing efficient aggregates, you'll need tools that carry out the cost/benefit analysis to make sure you get the right aggregates in the right place. The dimensional metadata we've described forms a starting point for such analysis. An understanding of the high-level structure of the model enables additional analysis based on statistics and samples of the data. Using this information, tools can recommend aggregates that best serve the expected access patterns while staying within an application's time and space constraints.

**OLAP tools can learn all they need to know about the data model by querying the database.**

The DB2 optimizer transparently rewrites queries to take advantage of available aggregates. Tools and applications don't need to be aware of the aggregates or make any adjustments to benefit from the aggregates. The DB2 optimizer will make the best possible use of the aggregates regardless of the source of the query. When considering how best to resolve a query, the optimizer considers both exact and partial matches before it resorts to building the result from the ground up.

Here's an example of how the optimizer helps OLAP efforts. Consider a database that stores sales data by product at the daily level. Aggregates exist at the calendar month and quarter levels. The optimizer would rewrite a query that requests sales by product summed across January, February, and March to simply select the existing Quarter 1 aggregate using a direct match. The optimizer would rewrite a query that requests sales by product summed across January and February to sum the existing monthly aggregates for January and February, making use of existing partial aggregates. In both cases, query response will be much faster than fetching and aggregating the corresponding daily data.

DB2 UDB v.8 introduced a new indexing capability known as multidimensional clustering. This feature lets you index data across multiple dimensions. When data is indexed in this manner, rows with the same values across dimensions are physically stored together; therefore, only one index entry is required for each set of rows. The result is very efficient indexing across multiple dimensions, which in turn improves query times.

## WEB SERVICES FOR OLAP

Web services are gaining acceptance as the basis of the next generation of applications and the new model for application development. Much of Web services' promise lies in open, service-based interfaces that applications can dynamically bind to. An OLAP ser-

## THE ABCs OF OLAP

Depending on your requirements, there are several different approaches to providing multidimensional data and the associated OLAP technology. For example, desktop OLAP (DOLAP) enables users to quickly pull together small cubes that run on their desktops or laptops. Multidimensional OLAP (MOLAP) provides managed, server-side cubes stored in special structures that are optimized for multidimensional analysis. Relational OLAP (ROLAP) provides highly scalable and flexible server-based OLAP by generating SQL directly against an RDBMS.

And, finally, there's hybrid OLAP (HOLAP). Driven by customer demand for a single solution that provides scalability and speed, OLAP solutions are converging on a HOLAP approach. Hybrid solutions are server based, use cube structures optimized for speed, and can generate SQL to retrieve data directly from an RDBMS when required.

vice that delivers cubes, slices, or cells from a multidimensional model would be a valuable foundation service.

Of course, Web services are unlikely to become the new slice, dice, and drill interface for dedicated OLAP tools. These tools require the high-speed service they get from existing native interfaces. But Web services-based analytic applications will need access to multidimensional information. These new applications will cross organizational and business boundaries, assembling information from a variety of sources and using it to inform and drive business processes.

Consider an inventory management application that calls on internal Web services and Web services from suppliers and partners to balance stock against demand. An application of this sort will require access to multidimensional data such as plan and actual inventory by product and location over time. Application developers need to spend their time and expertise on the algorithms and business models that provide the best service at the lowest cost. These developers are familiar with Web services development tools and techniques (such as XML and XPath) and with the business they work in. But they're not OLAP specialists. Simple Web services that provide fast access to analytic information can make the developers' jobs easier.

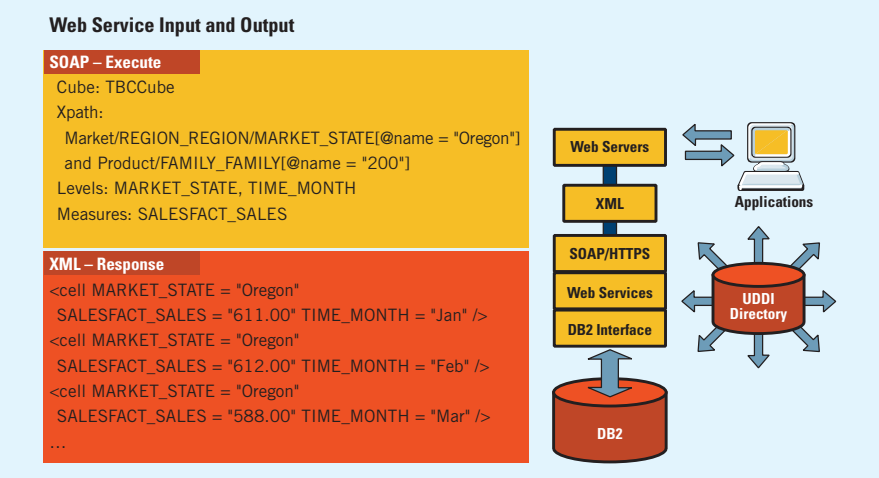


FIGURE 3: *A Web service for OLAP.*

Figure 3 shows a simple, high-level Web service for OLAP. The service allows developers to quickly find sources of dimensional information, determine the slices they need, and retrieve the data using an XPath-based execute method. Without learning OLAP interfaces and query languages, Web services developers will be able to call on their existing knowledge of XML and XPath to quickly add analytic information to their applications.

## ELIMINATING REDISCOVERY OF OLAP METADATA

When you use a modeling tool to design a dimensional schema, you're designing fact and dimension tables with all relevant attributes and facts. The designer knows how these attributes relate to each other and how they form the levels of hierarchies that reflect the way you do business. Unfortunately, most of this information remains locked up in the modeling tool, the extract, transform, load tool, or in the designer's head. The RDBMS has no knowledge of the higher-level organization of the data.

In order to make sense of the atomic elements, OLAP technologies rediscover these dimensions, hierarchies, attributes, and facts. So you must redescribe the same information for each tool every time you introduce a new schema or a schema change.

The process of rediscovery is error-prone, inefficient, and expensive. If you're lucky, some of your tools are able to talk to each other to exchange dimensional information. But even with integrated tools, you often don't have smooth, end-to-end flow of information from design to information delivery. This problem can be addressed by providing a simple database interface through which tools and applications can communicate and discover dimensional information. Incoming information can be stored and managed in the database catalog tables and made available to all tools that need to understand the higher-level organization of the data.

## AWARENESS AND FLOW

By building OLAP-centric features into recent RDBMS releases, database software vendors are acknowledging the increasing importance of OLAP integration.

The technology described in this article, however, establishes the future direction of BI architectures. An RDBMS that's OLAP-aware can play a key role in the end-to-end flow of dimensional information. This flow results in faster deployment, easier data management, and improved performance for OLAP and any BI applications that can benefit from dimensional information. ■

**MICHAEL L. GONZALES** has been a database developer for more than a decade. He manages a consulting firm, The Focus Group Ltd., and teaches a series of data warehousing courses across the United States. You can reach him at mlg@starfocus.com.

**GARY ROBINSON** is a senior software engineer in OLAP development at IBM's Santa Teresa Laboratory in San Jose, Calif., with more than 10 years' experience in business intelligence and decision support. You can reach him at robinsg@us.ibm.com.

## RESOURCES

**"Relational Extensions for OLAP,"** by N. Colossi, W. Malloy, and B. Reinwald, *IBM Systems Journal*, Vol. 41, Num. 4, 2002
researchweb.watson.ibm.com/journal/sj/414/colossi.pdf