

# ORACLE DATABASE 10G OLAP OPTION

## NEW FEATURES AND ENHANCEMENTS



### OLAP OPTION

- Partitioned variables
- Enhanced storage model
- Multi-Writer mode
- Parallel update
- Aggregation from formulas
- Optimizations to composite dimension indexing
- Certified with Real Application Clusters and Oracle Grid Computing
- Wider relational filters to multidimensional data types
- Support of SQL model clause
- Query rewrite to views over multidimensional data types
- Automatic runtime generation of abstract data types

*The OLAP option to Oracle Database 10g represents a truly unique offering to the OLAP market. It offers an industrial-strength calculation engine and performance unmatched by any stand-alone multidimensional database, yet it does so in the context of the reliable and secure platform of the Oracle Database. If simply competing in the OLAP database market were the goal, the OLAP option with all its calculation power and its OLAP API would be more than sufficient. The goal of Oracle, however, is higher. The goal is to present OLAP as a central component to the data warehouse rather than as an add-on to the data warehouse.*

### Partitioned Variables

The multidimensional engine provides direct support for partitioned variables. This support for partitioning presents many opportunities for both enhancing manageability and supporting large multidimensional data sets.

Three partitioning methods are supported:

- Range partitioning allows data to be partitioned based on a range of dimension members. For example, one partition might contain time dimension members that are less than '13', another that are less than '25', and so on.
- List partitioning allows data to be partitioned based on a list of specific dimension members. For example, a partition might contain dimension members <'JAN02','FEB02','MAR02'> and other partition might contain members <'JAN03','FEB03','MAR03'>.
- CONCAT partitioning partitions data according to the dimension members that belong to a CONCAT dimension.

With each partitioning method, the multidimensional engine creates separate variables to store data. To the application, it appears that all data is stored in a single variable.

Scalability is enhanced in a number of different ways:

- Data can be partitioned across time, thus providing the ability to store more historical data in the analytic workspace without affecting performance or manageability.
- Calculations can be easily limited to a subset of dimension members, or they can be parallelized. For example, aggregations, allocations and other calculations can be performed on time periods within a particular partition.

- Data loading can be parallelized.
- When partitioned by the logical model, for example, by level of summarization, the definition of the variable can be adjusted to account for changes in sparsity between detail data and summary data.
- Disaster recovery tasks can be performed on subsets of data and can be parallelized.
- Partitioned variables can be partitioned across different data files and disks to minimize I/O bottlenecks.

### Enhanced Storage Model

The storage model is enhanced to support the placement of objects in the analytic workspaces into specific rows of the AW\$ table. Objects can be further partitioned by segment size to allow for large objects. The AW\$ table can then be partitioned across multiple data files.

The obvious benefit of the enhanced storage model is that database administrators have complete control over how data is distributed across data files and can therefore optimize I/O for data and data access patterns.

### Multi-Writer Mode

The multidimensional engine supports a multi-writer attachment mode, which allows an analytic workspace to be modified simultaneously by several sessions. In multi-writer mode, users can simultaneously modify the same analytic workspace in a controlled manner by specifying the attachment mode (read-only or read-write) for individual variables, relations, valuesets and dimensions.

The `MULTI` attach mode provides the opportunity to parallelize any number of activities in the analytic workspace. Some examples follow:

- Using separate simultaneous sessions to load data into different variables can parallelize data loading tasks. For example, different sessions could be used to load data into `SALES` and `COST` variables. When combined with partitioned variables, different sessions could load into each partition in parallel.
- Separate sessions can be used to aggregate separate variables or partitions of a variable.
- Separate sessions can be used to solve models, allocations and virtually any other calculation within the analytic workspace as long as the calculation is directed to different variables or partitions of a variable.

### Parallel Update

The OLAP DML `UPDATE` command runs automatically in parallel on partitioned variables, thus optimizing performance of this command on servers with multiple processors. Significant improvements will be seen in cases where large volumes of data are updated (such as a data load or aggregation) and partitioned variables are used.

### Aggregation from Formulas

Oracle OLAP 10g allows formulas to be used as a source of data to the AGGREGATE command. This eliminates the need to calculate and store data at the detail level, yet still retains the ability to aggregate to summary levels. The benefit is that the multidimensional engine presents large volumes of derived information from relatively little stored data.

### Optimizations to Composite Dimension Indexing

New 64-bit B-Tree+ indexes and optimizations to the process of synchronizing composite dimensions to base dimensions support excellent query response times with very large composite dimensions (for example, composite dimensions in excess of 1 billion members).

### Certified with Real Application Clusters and Grid Computing

Real Application Clusters and Oracle Grid Computing provide a database platform of virtually limitless computing capacity and scalability. The multidimensional engine and data types of the OLAP option, being part of the Oracle Database, have been tested with Real Application Clusters and Oracle Grid Computing. This provides Oracle OLAP the capability to support very large user communities and data sets.

### Wider Relational Filters to Multidimensional Data Types

The OLAP 10g optimizes a wider range of SQL predicates when selecting from multidimensional data types. This is accomplished by applying SQL filters before the data is converted to a row set using OLAP\_TABLE. As a result, the risk of pushing large volumes of data through OLAP\_TABLE is minimized and applications need not be as concerned with optimizing SQL for selecting from OLAP\_TABLE. The net result is that a wider variety of SQL applications can be used with the OLAP option without special considerations.

### Support of SQL Model Clause

Oracle Database 10g introduces OLAP-like calculations that are expressed with a SQL MODEL clause, which is similar to what the OLAP community commonly refers to as *custom dimension members*. A custom dimension member is a virtual member whose value is calculated at runtime.

The SQL MODEL clause provides an additional method for defining certain types of calculations against multidimensional data types, and the SQL interface to multidimensional data types has been optimized for SQL models. Optimization occurs by having the multidimensional engine completely bypass OLAP\_TABLE as data is being returned.

As a result, the processing of SQL with the MODEL clause is highly efficient against multidimensional data types. In many cases, performance of MODEL with multidimensional data types exceeds that of the same SQL against relational tables. This provides SQL based applications with both new analytic features and performance advantages.

**SUMMARY****ORACLE OLAP  
IN THE DATA WAREHOUSE**

The OLAP option to Oracle Database 10g focuses on the primary issues affecting the status of multidimensional data types within the data warehouse: scalability and the ability to support SQL as a query language. The result is the ability to efficiently manage very large multidimensional data sets and to support a wide range of SQL based tools and applications.

**RELATED PRODUCTS**

Oracle OLAP 10g is part of the family of Oracle Business Intelligence Products. The following related products are also available:

- Oracle Warehouse Builder
- Oracle Data Mining
- Oracle Business Intelligence Beans

**RELATED SERVICES**

The following services are available from Oracle Support Services:

- Update Subscription Services
- Product Support Services
- OnlineDBA
- OnlineDBA for Applications

**Query Rewrite to Views over Multidimensional Data Types**

In Oracle Database 10g a new feature, *query equivalence*, allows query rewrite to be used with views. With query equivalence, the DBA indicates to the database what SQL *could have* been used to create the view even if the view was created in some other way. For example, if the application likes to emit SQL with SUM ... GROUP BY but the view was created with entirely different SQL, the DBA could indicate that the view is equivalent to SUM ... GROUP BY.

This feature of the database is extremely useful with the OLAP option since SQL access is always through views. This provides the DBA and application with benefits similar to those of materialized views – simplified maintenance and improved query performance.

**Automatic Runtime Generation of Abstract Data Types**

Abstract data types are used by object technology of the Oracle Database to define the relational columns for data that is returned from a non-relational data source. In the case of the OLAP option, abstract data types describe data being selected from analytic workspaces in terms of relational columns.

Previously it was a requirement that abstract data types be created as part of the administrative process of enabling analytic workspaces for query by SQL. To provide applications and database administrators with additional flexibility in the administration of SQL access to analytic workspaces, Oracle OLAP 10g supports automatic runtime generation of abstract data types as part of the query process.

With the addition of this new feature, it is now possible to query analytic workspaces without requiring the DBA to predefine either abstract data types or views.