Oracle Business Intelligence Server and Embedded Database Functions

An Oracle White Paper
October 2007

ORACLE

INTRODUCTION

Oracle Business Intelligence Suite Enterprise Edition Plus is a comprehensive and integrated suite of analytic tools designed to bring greater visibility and insight to the broadest audience of users, allowing any user to have Web-based, self-service access to up-to-the-moment, relevant, and actionable business intelligence. The Oracle Business Intelligence Server (OBI Server) is a key component of Oracle Business Intelligence foundation and operational business intelligence applications. OBI Server is a highly scalable, highly efficient query and analysis server that provides services that enable the end user facing components of the Oracle Business Intelligence foundation such as Answers - for ad-hoc query and reporting and Dashboards - for interacting with reports/graphs, navigating, and drilling into hierarchies, and even to operational applications., . The OBI Server has an extensible and open connectivity layer with a set of adapters that are responsible for communicating with source data servers. Individual adapters have been built to communicate to a variety of relational and non-relational data systems.

When communicating with data sources, the OBI Server is aware of the functions and language constructs for the particular data system and generates highly optimized target-specific language. The goal is to optimize the language such that the data system conducts as much of the processing as possible to satisfy an end user request. If the data source does not support a function or feature, the OBI Server will compensate for the missing functionality using its own computation and data processing engine. The ability to customize the query language to a particular data system and to compensate for missing functionality is unique to the Oracle BI Server.

For relational database systems, the OBI Server can perform a superset of the data manipulation and calculation capabilities of the SQL-92 standard. This set of capabilities is encapsulated in a features library that provides coverage for the vast majority of user query and analysis needs. However, there are cases where a user may need to access unique vendor specific capabilities outside of the features library. The goal of the Embedded Database Functions enhancement is to provide a product feature to address these cases.

BUSINESS REQUIREMENT

To satisfy sophisticated reporting and analysis needs, the requirement is to extend the existing standard OBI Server features library by providing a mechanism to support vendor-specific features and custom functions available within the various database systems supported by OBI EE. Administrators should also have the capability to pre-define calculations that leverage external features in the metadata repository for use by end users. Advanced end users should have the ability to access external features from the Oracle BI end user interface to build custom formulas.

Vendor-specific capabilities that the product feature should address can be categorized as follows.

Non-Relational Systems

Examples of non-relational systems include Oracle Analytic Workspace for OLAP, Oracle Data Mining, and Oracle Spatial. An Analytic Workspace is a multi-dimensional technology integrated with the Oracle relational database that utilizes a robust multi-dimensional calculation engine to perform advanced analytics in areas including high-performance multi-dimensional ad-hoc query and reporting as well as a foundation for analytic applications such as planning, budgeting, forecasting, sales, and marketing.

Oracle Spatial allows for complex analysis of data related to location. It helps enterprises answer questions like, "Where are my customers located?" or "What are the best locations for a new store?"

Oracle Data Mining is a predictive intelligence engine used to help business analysts find patterns and insights hidden in their data.

With Oracle Analytic Workspace, Oracle Data Mining, and Oracle Spatial a unique grammar is used to access the capabilities of the underlying technology.

SQL Extensions

Most relational database vendors provide extensions to standard SQL grammar. There are cases in which the extension does not have a counterpart in OBI EE SQL. As an example, the Oracle relational database offers extensive capability to perform complex regression analysis that is not available in the OBI EE feature set.

Non-SQL Functions and Procedures

Relational databases also include a library of non-SQL vendor specific functions. For example, GET_DATETIME is used in Oracle for retrieving the current data/timestamp.

Stored procedures offer the ability to extend the SQL language by providing procedural programming constructs. Stored procedures are often used for

complex processing requirements where several SQL statements need to be executed to arrive at a result. In the Oracle database, store procedures are implemented in a programming language called PL/SQL. PL/SQL routines can be stored and thus made available for general access.

ORACLE BUSINESS INTELLIGENCE SOLUTION

As a point of background, the OBI Server is accessed using logical SQL against a unified semantic layer. The logical SQL is translated into vendor specific physical SQL by a sophisticated navigation and rewrite process driven by the semantic layer.

To enable access to extended physical data source capabilities, a set of new functions are introduced as part of the BI Server SQL API grammar. The new BI Server API functions allow unique source specific features and functions to be embedded into logical SQL and passed through to the physical database. The functions are syntactically similar the C sprintf function. The specific variants of the function include:

EVALUATE

Used to implement scalar functions and are computed post-aggregation.

Syntax:

EVALUATE('*expression*(%1, %2, …)' as *TYPE, parm1*, *parm2*, ...)

The first argument for the EVALUATE function is the external expression in single quotes. The expression may take several arguments referenced by %num in sequential order and can have an optional return TYPE. The expression is followed by the actual parameters to be passed to the external function. Parameters may include a combination of columns, variables, and constants.

Consider the following examples, the first for the Oracle relational database, the second from Oracle Data Mining:

EVALUATE('dense_rank() over(order by %1)', rev) calculates the dense rank for the column rev. With dense rank, items that are equal receive the same ranking number, and the next item(s) receive the immediately following rank.

EVALUATE ('PREDICTION (DT_SH_CLAS_SAMPLE COST MODEL USING %1, %2, %3)', cust_marital_status, education, household_size) returns the prediction for the model DT_SH_CLAS_SAMPLE using columns representing martial status, education, and household size as inputs.

Note also that the variants of Evaluate can be nested into other OBI EE supported functions.

EVALUATE_AGGR

Used to implement custom aggregations, not supported by standard aggregation functions

The syntax is similar to EVALUATE.

EVALUATE_AGGR('*expression*(%1, %2, …)' as *TYPE, parm1*, *parm2*, ...)

As an example:

EVALUATE_AGGR('REGR_SLOPE(%1,%2)' as DOUBLE, quantity_sold, list_price) returns the slope component of a linear regression line of quantity_sold over list_price.

EVALUATE_AGGR is computed at the grain of the query.

EVALUATE_PREDICATE

Used to model native Boolean types for relational systems

Syntax:

EVALUATE_PREDICATE ('*expression*(%1, %2, …)', *parm1*, *parm2*, ...)

The syntax is similar to the other two variants of Evaluate. However, the expression for this case is a data source specific Boolean expression. Data sources which utilize such expressions include Oracle Spatial filters and Oracle Analytic Workspaces.

An example for Analytic Workspaces follows:

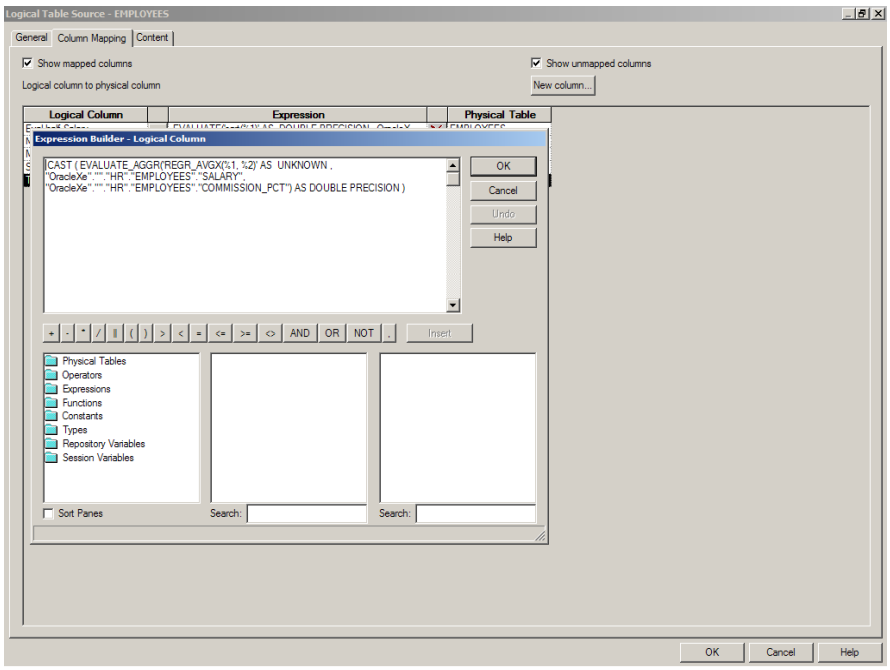
EVALUATE_PREDICATE('OLAP_CONDITION(%1, "LIMIT time KEEP ""1"", ""2"", ""3"", ""4"" ") =1', OLAP_CALC)

Note that EVALUATE_PREDICATE is used in the filter clause of the OBI EE SQL. In the above case, OLAP_CONDITION is used in the filter clause of a query against an OLAP Table.

Deployment

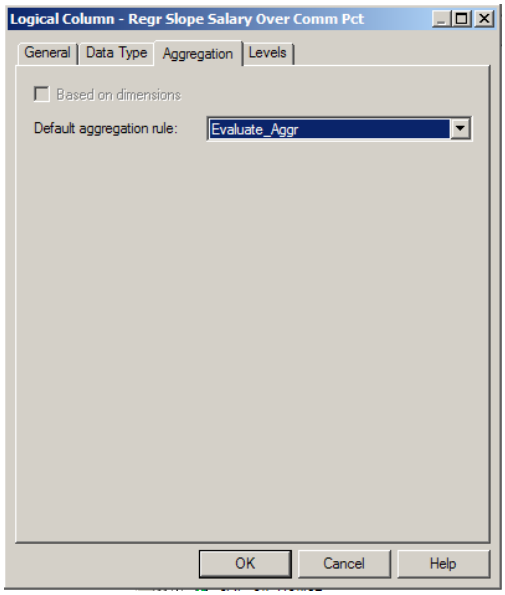
The design goal is to allow access to the Evaluate function either in a pre-defined and registered formula or as an end-user available function to dynamically define a formula. Generally we recommend the former approach. By having administrators define and register a formula, the formula is standardized and available to a broader group of users. In addition, with most enterprise implementation processes the formula will most likely undergo some level of quality testing.

For administrators, the EVALUATE and EVALUATE_AGGR functions can be implemented as a column expression in the Logical Table Source of the OBI EE Metadata.



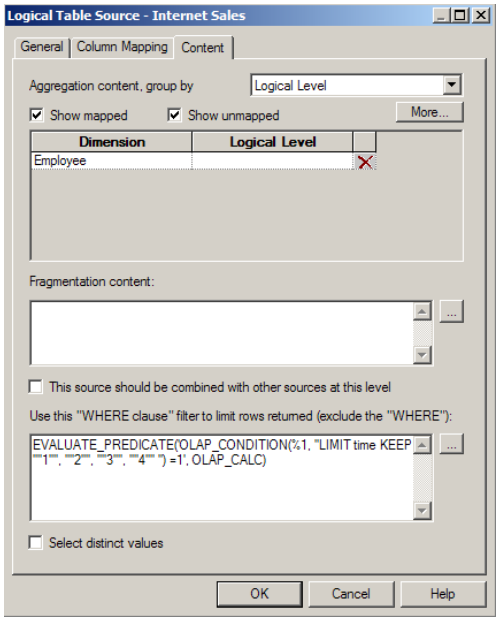
Since, Evaluate references a physical data source expression, the Logical Table Source is most appropriate location for the logical column expression.

For EVALUATE_AGGR, Administrators will need to set the default Aggregation rule for the logical column to Evaluate_Aggr.



Note that Evaluate_Aggr is not allowed for level-based measures.

EVALUATE_PREDICATE, as a conditional expression will most often be deployed as a Filter for a Logical Table Source.



On certain occasions an end-user may need access to a physical feature that may not be satisfied by a column registered in the OBI EE metadata. In this case, the user can use EVALUATE (note that EVALUATE_AGGR and EVALUATE_PREDICATE are not supported at this time) in an Answers column formula. There are certain precautions and limitations to this technique that are discussed in the next section.

Guidelines and Limitations

Following are a set of implementation guidelines for deploying Embedded Database Functions:

- Because of issues that may arise when the AW level pinning predicates do not exactly match the grain of the logical SQL request, we recommend that customers use the EVALUATE_AGGR function to ship OLAP_EXPRESSION calculations to the AW. However, since EVALUATE_AGGR expects to receive an aggregate SQL expression, the OLAP_EXPRESSION function must be wrapped with a SQL aggregate function such as SUM or MAX. For example: EVALUATE_AGGR('SUM(OLAP_EXPRESSION (%1,"MOVINGAVERAGE(UNITS_CUBE_SALES, -2, 0, 1, TIME)"))' , Facts.OLAP_CALC)
- Pass-through features are supported only against logical items that have a single, common physical source. End-users that decide to deploy a pass-

through feature in Answers must have an understanding of the physical sourcing of the logical column affected.

- Administrators must take measures to ensure that risks for malicious use of pass-through features are mitigated. Most often, this can be accomplished by applying appropriate security at the physical data source level.
- Validation of logical expressions is only syntactical, i.e. the engine does not check for the validity of the pass-through feature for the underlying physical data source.
- Cache hits are limited to "exact match" cache entries.
- When the output data type is not specified, the software will attempt to predict the output type from input arguments. If there are of various input types, an incorrect output type may be selected potentially causing an error.
- MDX-based sources are not supported for OBI EE 10.1.3.3. They will be supported in a future release

BUSINESS BENEFITS

Enterprises make significant investments in database systems to support on-going business analytics requirements. OBI EE is designed to take advantage of the strengths of underlying database systems through optimized SQL generation and function shipping. The Embedded Database Function feature further enhances the ability to fully leverage the unique capabilities of the database system. The benefits to the business include:

- **Faster ROI.** OBI EE's ability to take advantage of all the unique capabilities of a database system allows the enterprise to leverage the investment in a database system to the fullest extent.
- **Better Business Decisions.** Providing end-users access to advanced analytical capabilities available within relational databases and specialized data systems in an easy to use interface allows users to gain better insight into business problems. Better insight results into better business decisions.
- **Lower TCO.** Often specialized interfaces or coding is required to access advanced capabilities. By providing this capability directly to end users through the easy to use Oracle BI presentation interfaces, enterprises are allowed to further standardize on a single tool for all analytical needs. Lower TCO is achieved by lower training, support, and maintenance costs.

CONCLUSION

The Embedded Database Functions feature further enhances the OBI Server's ability to fully leverage the capabilities of a source database system. The feature

provides a new set of functions that allows both transparent and direct access to unique and vendor specific capabilities. The feature enhances the business user's access to insight in areas such as multi-dimensional, financial, and statistical analysis. The results for the enterprise are faster ROI on their database investment, better business decisions, and, lower TCO on their Business Intelligence investment.

White Paper Title
[Month] 2007
Author: [OPTIONAL]
Contributing Authors: [OPTIONAL]

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com