Hyperion and Teradata
Best Practices

Scope

Revisions

Date	Name	Comments
11/19/2004	Rupal	Initial Draft
02/28/2005	Allen	Revision
05/26/2005	Rupal	Accepted release 1.0

Table of Contents

Introduction

Terminology and Definitions

Due to diversity in the terminology, when it comes to Essbase, OLAP and Hybrid Analysis, we offer a short list of definitions and terminology for this document:

- *Essbase* can be and is sometimes referred as Essbase database, multidimensional database engine, OLAP server and Essbase XTD Analytic Services platform. All which stores the multidimensional part of the cube (outside of the source) and executes the Hybrid Analysis features to access relational data.
- *Hybrid OLAP/Cube* refers to the OLAP application, with both a relational and a multidimensional part exists.
- *OLAP cube* refers to a pure multidimensional cube, without the relational storage part (this means, without the active usage of Hybrid Analysis).
- *Hybrid Analysis* is the relational "analysis" of the Essbase application.
- *Hybrid Analysis cube* refers to the multidimensional part of the hybrid analysis.
- *Hybrid Analysis levels* refer to the levels defined for relational access in the hybrid cube.
- *Analytic Framework* is a conceptual term to refer to the analytical environment designed from the OLAP model and metaoutline, whether it is an OLAP or Hybrid cube or both.
- *Drill-Down* refers to the ability to drill down a dimension(s) whether it is an OLAP or Hybrid part of the cube.
- *Drill-Through Report* refers to the ability to access additional data (outside of the current defined analysis) for a cell in an OLAP or Hybrid part of the cube. A Drill - Through Report is a predefined report which will filter on the cross-section of cell.

Note: Hybrid Analysis Feature or any of the terms used above (i.e. Hybrid cube or Hybrid Analysis level) does not create any data in any RDBMS to support these types of analysis. Relational queries are generated to retrieve appropriate data for display when accessing a Hybrid Analysis level.

Overview of Essbase and Essbase Integration Services

Essbase Analytic Services (OLAP Server) and Essbase Integration Services (EIS) are two components within Hyperion Essbase XTD – Business Intelligent Platform. Essbase Analytic Services and Essbase Integration Services can be installed on the same or separate platforms based on a customers needs. For a complete Essbase XTD Platform component offering please see www.hyperion.com.

It should be noted,

- Hyperion and Teradata can be utilized "out of the box" without any special configurations to either system.
- Essbase Integration Services is not an ad-hoc or managed reporting tool, but rather a framework for building Essbase Analytical OLAP applications.
- Essbase Analytic Services is an Essbase multidimensional database/data store for OLAP type "speed of thought" (slice and dice) analysis. Built from data that can be extracted from a variety of sources including a RDBMS with drill-down and drill-through capabilities back to originating source.

Essbase Integration Services product provides the critical link between your relational data and Hyperion Essbase OLAP Server. The Essbase Integration Services product provides a way to transfer quickly the relevant data from your relational database to an Essbase multidimensional database whether your Analytic Environment requirements dictates MOLAP, ROLAP or HOLAP type of analysis.

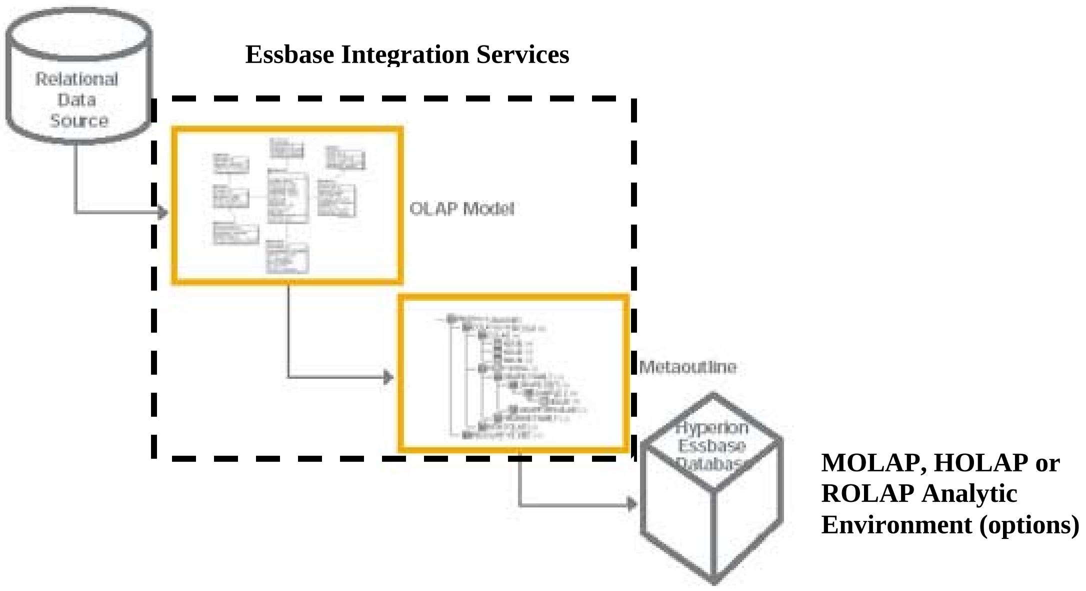

Relational Data Source
Essbase Integration Services
OLAP Model
Metaoutline
Hyperion Essbase Database
MOLAP, HOLAP or ROLAP Analytic Environment (options)

Figure 3-1

Connectivity to the Teradata Database is achieved through the Teradata ODBC driver (Windows and UNIX platforms are currently supported). Connectivity to other Essbase components is via Essbase APIs.

For Essbase Integration Services, Teradata Database and Teradata ODBC Driver version support, see http://www.essbase.com/products/

Essbase Integration Services product provides graphical tools to assist you in the following ways:

- You can create a logical OLAP model from tables, views and columns in your relational database. The OLAP model that you create is a logical schema consisting of a fact table surrounded by related dimension tables.
- You use the OLAP model to create a metaoutline, an outline template that contains the structure and rules required to generate a Hyperion Essbase outline.
- You use the metaoutline to create and populate a Hyperion Essbase multidimensional database.

Architecture

Essbase Integration Services consists of two major components: Essbase Integration Services Desktop and OLAP Integration Server. The desktop is the graphical interface and the server is what communicates to Essbase Server (i.e. multi-dimensional databases), RDBMS and Essbase Services.

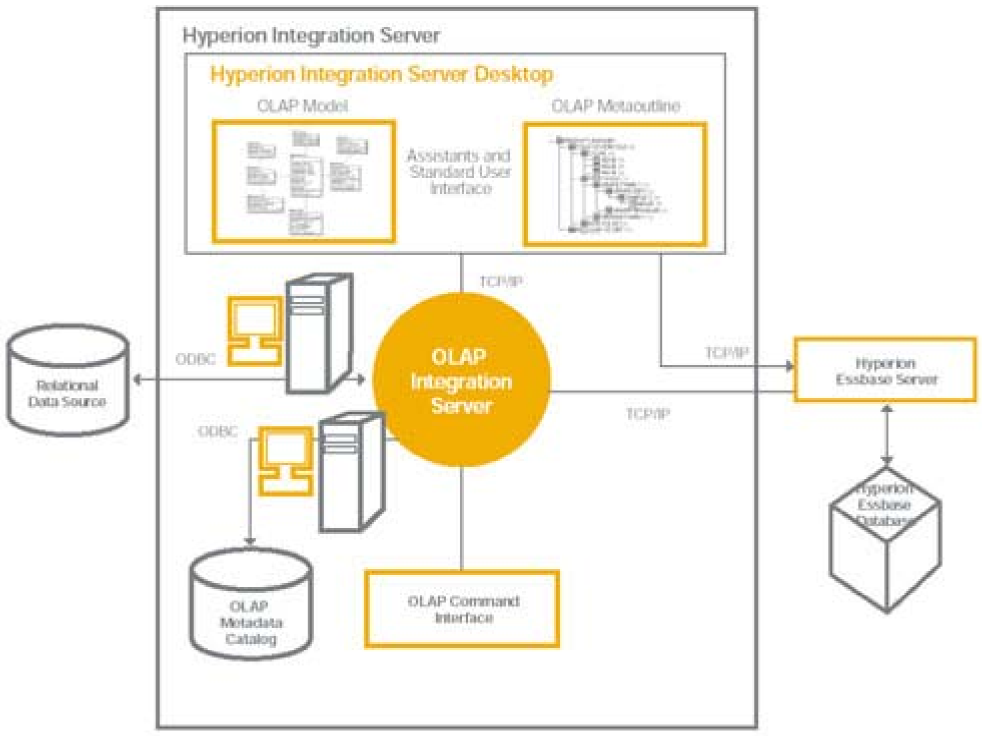


Figure 3-2

Workflow for Using Essbase Integration Services

To create an Essbase multidimensional database from a relational data source:

- Build an OLAP model that is based on the tables in the relational data source. OLAP Integration Server stores the OLAP model and the information necessary to retrieve the relevant tables in OLAP Metadata Catalog.
- Create a metaoutline from the OLAP model. OLAP Integration Server stores the metaoutline in OLAP Metadata Catalog.
- Load members and data into the Hyperion Essbase database.
- Update the Hyperion Essbase database with new members and data.

OLAP Models and Metaoutlines Overview

Essbase Integration Services approach to building an Essbase OLAP Analytic Framework is based on a 'logical' star and snowflake like design. Hence, BI Administrators should work with the DBA to create a view layer to expose the appropriate objects based on the analytic reporting requirements. See next section 'Implementation Considerations' for more information. OLAP models are based on the idea that values in a relational database can be categorized as either facts or dimensions of facts. Facts are the numeric, variable values in the database, such as sales figures and numbers of units sold. Associated with facts are related data values that provide additional information, such as store locations and product IDs of units sold. An OLAP model contains a fact table, one or more dimension tables and one or more dimension branches. An OLAP model may contain time and accounts dimensions.

Essbase Integration Services OLAP Models and Metaoutlines

Essbase Integration Services creates an OLAP model that is a logical model with join and hierarchy information defined. The OLAP model is a logical representation of the data values that you select from 'view layer' in the relational database and that you want to report in Hyperion Essbase.

Use an OLAP model to create one or more metaoutlines. A metaoutline contains the basic structure required to build a Hyperion Essbase outline and to load data into Hyperion Essbase. You can use one OLAP model as the basis for another OLAP model. You save the original OLAP model under a different name and edit it as needed to meet reporting requirements. You can create any number of OLAP models for use in building metaoutlines. Each metaoutline, however, is based on one specific OLAP model. OLAP models have the following features:

- They are reusable. You can use the same OLAP model as the basis for multiple metaoutlines.
- They provide a layer of abstraction that insulates the Hyperion Essbase database outline from changes in the relational database.

- They enable you to create hierarchies to structure and summarize data from a relational database. You can use the hierarchies in multiple metaoutlines.

Connectivity

Essbase Analytic Services and Essbase Integration Services require ODBC for connectivity. Support for both Teradata ODBC for Windows and UNIX is available. See http://www.essbase.com/products/ for latest support matrix for Teradata

Metadata Repository

OLAP Metadata Catlalog is Essbase Integration Services metadata repository, which is a set of tables that exists in a relational database that contains OLAP models, metaoutlines, and the information necessary to source and build your Essbase Integration Services framework and Essbase environment.

Teradata is a supported metadata repository for Integration Services. Though, it is not required that the OLAP Metadata Catalog be on the same database platform as the data you are sourcing for your analytical environment. Hence, based on complexity of your analytical and database warehouse environment, Integration Services also supports for MySQL as a catalog database in the 7.0 release. It is considered a “batteries included” database for Essbase Administration Services.

Hybrid Analysis Feature

Overview

Hybrid Analysis is a feature included in Essbase. It allows part of a multidimensional cube to reside in a relational data source (i.e. does not actually create any structure(s) in the relational database, but rather accesses content via relational queries).

To the end user, the cube employing Hybrid Analysis still looks like a single, seamless multidimensional application. The Hybrid Analysis part of the cube is computed dynamically by SQL queries against the relational source database.

This means that Hybrid Analysis accesses member names and cell values in the relational data source through dynamically generated SQL, instead of being pre-loaded and pre-calculated in OLAP Server.

The hybrid analysis portion of the cube can be thought of as a logical (or virtual) partition of the cube residing in a relational database. The member names and cell values of this portion of the cube are computable at query time by querying the source relational database using information stored in the Integration Server metadata catalog.

The metaoutline specifies which part of the application is situated in relational storage, and which part is situated in multidimensional storage, both when the application is built and when the application is queried, as shown in Figure 3-1 above. The Integration Server does not need to be running for Hybrid Analysis to work: it is only involved in the creation of Hybrid Analysis cubes. After that, clients talk to Essbase, which accesses the metaoutline and the RDBMS.

Figure 3-1 above accessing relational and multidimensional data in a hybrid cube is similar to any drill-through ability, in the sense that it provides a way of getting more detailed data from the relational source. However, it is more convenient, because you don't have to create and maintain several Drill-Through Reports.

Guidelines

You should be familiar with these Hybrid Analysis guidelines:

- A single Analytic Services database can be associated with only one hybrid analysis relational source.
- A hybrid analysis relational source can consist of only one relational database.
- A hybrid analysis enabled member should not be renamed. If a member is renamed, the member may not be retrieved the next time you perform a drill-through operation.
- Hybrid Analysis is not supported on accounts dimensions.
- Hybrid Analysis is not supported on user-defined dimensions.
- Analytic Services does not support aliases for members that are enabled for hybrid analysis.
- Analytic Services requires the OLAP Metadata Catalog created in Integration Services in order to drill down in a hybrid analysis relational source.
- You can perform operations and analyses on dimensions that have attributes attached to one or more levels in an outline that is hybrid analysis enabled. The attribute dimension should be fully loaded into Analytic Services.
- Hybrid Analysis does not support scaling of measures dimension members using any of the operators + (addition), - (subtraction), * (multiplication), and / (division). If you use the scaling operators, drill-through queries into hybrid analysis data may show a mismatch between aggregated level-0 values in the Analytic Services database and the corresponding detail values in your relational data source.
- Analytic Services ignores all member properties, such as formulas, UDAs, and aliases for members that are enabled for hybrid analysis.
- Hybrid Analysis does not support transparent, replicated, or linked partitions.
- Only the first hierarchy of a dimension with alternate hierarchies can have members enabled for hybrid analysis on its lowest levels.
- Hybrid Analysis does not support recursive hierarchies.

- Analytic Services supports drill-through operations defined on members that are enabled for Hybrid Analysis.
- When building a dimension that is enabled for hybrid analysis, you must ensure that the column in the relational table that contributes to the leaf level of the Analytic Services portion of the dimension is non-nullable.
- You can associate an attribute member with a member enabled for hybrid analysis.

Note: Interested in implementing Hybrid Analysis feature should also refer to Hybrid and Teradata Tuning section below for more information. Specifically, Teradata Aggregate Join Index feature to improve Hybrid query response.

Implementation Considerations

At a high-level, the following areas should be considered during your Hyperion and Teradata solution implementation when it comes to integrating and delivering BI content from an Enterprise Data Warehouse environment.

Physical Database Design

Physical database design of any Enterprise Data Warehouse by definition should reflect the customers business independent of any tools requirement or approach to delivering BI content. The Teradata EDW should be designed to adhere to the practices and methodologies to best support a enterprise data warehousing environment. Hence, it will be assumed in this document that this Teradata EDW foundation exists. Meaning, the physical design, whatever it may be, should be agnostic to any tool, and should be ultimately driven by to the customer’s business requirements.

Semantic Layer

Independent of the physical database design mentioned above, the recommended method to access relational content is to create a semantic layer (i.e. view “database/user” layer). This should contain appropriate objects (i.e. views) pointing to base/production tables that are required to support the customers reporting/analytic requirements and any tool dependencies to deliver and access BI content. Hence, it is appropriate to have many databases/users representing a semantic layer for accessing base tables for different analytical needs. This approach fits in well with Essbase Integration Services approach to designing OLAP models, which is based on a ‘logical’ star and/or snowflake approach. Not to mention, creating a view layer for users/BI administrators, is generally considered ‘good practice’ in a Teradata environment. A DBA should work closely with the BI Administrator in determining the appropriate views required to meet the analytic reporting needs. The semantic layer can also contain specific objects required to meet performance needs for the analytic environment. Aggregate Join Indexes and/or Summary Tables are two such examples.

Approaching Performance

Once you have built your EDW and BI semantic layer, and created and built your Essbase Analytic Environment, naturally, the next question your end users will ask you is whether or not you can make it *faster* or improve performance. In general, the word 'performance' itself has no meaning unless it is tied to a specific aspect of the analytical environment you are interested in improving, like; query time, data transfer time and/or cube calculation/building time. Or, it could be about hybrid drill-down and/or drill-through report response times. Provided you have adhere to the first two items above and exhausted basic database and Essbase tasks, there is no hard and fast "rule of thumb" when it comes to performance without identifying a specific 'pain point'. See *Best Practices for Performance Tuning* section below for more information. Sometimes all the tuning in the world may not improve your OLAP environment to the desired level (i.e. deeper analytics) unless you change your approach in how you deliver your analytics (i.e. moving from a MOLAP to HOLAP/ROLAP solution, see *Improve your OLAP environment with Hyperion and Teradata* white paper for more information). Regardless, your approach to improving your OLAP environment should be based on empirical evidence first and compared to the customer's performance requirements. Hence, if the dataload 'query' time is not being met, based on the customer performance requirements, then the BI and DBA administrators will need to look into 'specific' areas (i.e. physical database design, Essbase tuning parameters and/or revisit customer requirements) to improve this requirement and understanding any impacts if changes are made. For example, de-normalizing tables in a normalized environment may improve query time, but may limit what you can ask against the now 'new' physical database (PDM) and change your update strategies to those tables.

Creating an ODBC datasource

Accessing the relational database from Essbase Integration Services and Essbase Analytic Services will require an ODBC datasource DSN definition (a point of entry). Using the 'semantic layer' methodology above, the ODBC datasource should reference this database/user created in this the view layer. Your ODBC default database and logon information should reflect this 'point of entry' for accessing your OLAP environment. Refer to the Teradata ODBC documentation for more information on how to set your default database and logon information.

Use of Teradata XViews

To control the selection of objects/views/tables presented to the Essbase Integration Services administrator, ensure your Teradata ODBC datasource definition uses "Use XViews" option to display only those objects the ODBC datasource logon user has access too. This may result in better behavioral performance during OLAP modeling activities with the database. Refer to the Teradata ODBC documentation for more information on how to set this option.

MaxResponseBufferSize (8K to 65K)

To maximize the Teradata ODBC Driver response buffers used for SQL requests and data retrieval. It is recommended to increase the ODBC parameter "Maximum Response Buffer Size" from the default setting of 8192 to 65477. This will help in large answer set retrieval. The difference in performance may be considerable if the result set is large. If the Row Size of the result set is large and the Number of Rows in the result set is also large, then a better performance can be expected on increasing the 'Maximum Response Buffer'. With small result sets the difference in performance is negligible.

Refer to the Teradata ODBC documentation for more information on how to set this option.

Metadata Repository Considerations

Essbase Integration Service – Though, you can install the OLAP Metadata Catalog in Teradata and follow the above sections recommendations on 'Creating an ODBC Datasource' and the 'Use of XViews'. If you still experience unacceptable behavioral performance, depending on the amount of relational objects and OLAP model design complexity of your analytical environment, consider using MySQL as an alternative.

User Management (Authentication) – Essbase and Teradata

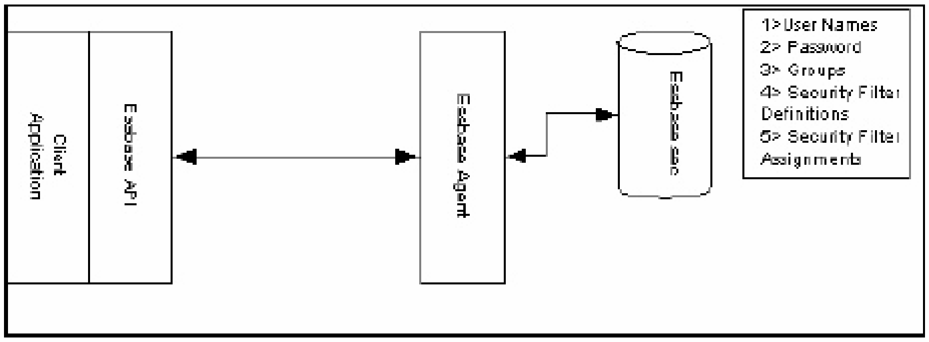
This section is intended to explain the various authentication and synchronization options available within and between each Hyperion product and with other environments. Basically, there are 3 areas of authentication available depending on your level of need.

- *Native* – Authentication within a Hyperion application or product.
- *External* – Authentication using an external repositories such as LDAP and Active Directory
- *External (custom)* – Authentication using an external (custom) module against a relational database

Native Authentication

The native authentication is contained within the security layer in Essbase, which requires user passwords to be stored in an internal encrypted format in the ESSBASE.SEC security file.

When users log into the Essbase server, their identity is validated against the corresponding entry in the security file. This process is illustrated below.



If the user is correctly validated (correct user id and password are entered) the user is then presented with the applications/databases that he or she has access to.

Native authentication presents many security issues to customers that have deployed many users and implemented many Essbase servers:

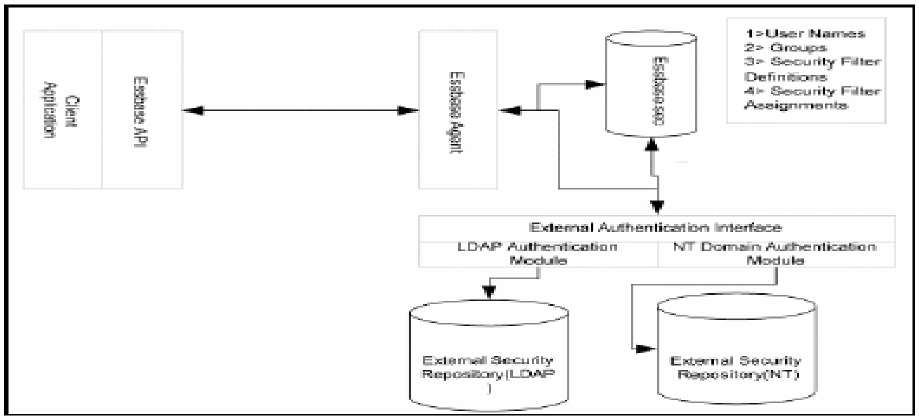
- User passwords are not kept in synch between servers
- No complex password rules are enforced, such as non-dictionary words, non-recycling of passwords, etc.
- Employee terminations may leave access vulnerable if the user is not removed from each server.

Note: This similar 'native' authentication process exists across other Hyperion products with their own internal encrypted format and would adhere to the same security and authentication challenges mentioned above.

External Authentication

External Authentication is a feature that allows Essbase to authenticate a password against an external security repository. Therefore, users are defined in Essbase, but no passwords need to be entered for those users.

Whenever a user flagged as externally authenticated logs in, Essbase calls out to the external security repository to authenticate the user. This process is illustrated below.


Client Application
Essbase API
Essbase Agent
Essbase.sec
1> User Names
2> Groups
3> Security Filter Definitions
4> Security Filter Assignments
External Authentication Interface
LDAP Authentication Module
NT Domain Authentication Module
External Security Repository(LDAP)
External Security Repository(NT)

As the above figure indicates, the authentication module is a separate library that can easily be adapted to support other repositories as needed. As of Essbase XTD Analytic Services Release 6.5.1, the following security repositories are supported:

- NTLM
- LDAP 3.0
- Microsoft Active Directory Services

Other repositories will be formally supported over time. Additionally, the authentication module (i.e. Hyperion APIs) allows customization to support custom repositories such as relational databases (see External (custom) Authentication).

Note: This similar 'external' authentication process can exists across other Hyperion products with the above repositories mentioned. Hence, this would address authentication and synchronization between Hyperion products and users, but not against relational database user accounts.

External (custom) Authentication

The third option is providing external authentication to an existing relational database user accounts. Here there are two options available depending on your level of need and resources.

1. Write a custom Teradata module to interact directly to the relational database (i.e. bypass any 3rd party external authentication such as LDAP and Active Directory)
2. Export/Import database user account information to any 3rd party external authentication repository (i.e. LDAP and Active Directory)

Both have their advantages and disadvantages. A custom module would eliminate any synchronization maintenance between Hyperion applications and the database. However, would limit non-Hyperion or those applications that does not have the external (custom) authentication option incorporated in their application. Whereas, using a 3rd party external repository would be application independent, but would require synchronization and export/import capabilities from the relational database vendor.

For more information regarding guidelines to writing an external 'custom' module see document "Common Security Services" at http://gsssystem.sandiegoca.ncr.com/tadc/tpcoe/default.htm under Hyperion section

Best Practices for Performance Tuning

It is assumed in this section both Hyperion and Teradata environments and associated components are installed and configured appropriately. Use this section as a check list, since not all considerations may apply depending on your specific performance requirements.

Hybrid Analysis Tuning

The following parameters are important in regard to Hybrid Analysis, and are set in the ESSBASE.CFG file located in the Essbase/bin directory:

- **HAENABLE** – Set to TRUE (i.e. HAENABLE   TRUE) to enable Hybrid queries to retrieve relational members and data. This is the most important setting, since it enables or disables Hybrid Analysis on the Essbase OLAP Server.
- **HAMEMORYCACHESIZE** – Size the amount of memory for member caching. This reduces the amount of member lookup SQL, while data lookup still takes the same duration.
- **HAMAXNUMCONNECTION** – Determine the number of maximum connections per Essbase OLAP multidimensional database.
- **HAMAXSQLQUERY** – Set the maximum numbers of SQL queries.
- **HAMAXQUERYROWS** – Set the maximum number of rows returned from your SQL query.
- **HAMAXQUERYTIME** – Set the maximum query time before terminated.
- **HARETRIEVENUMROW** – Determine the number of rows, resulting from an SQL query, to process at one time.

These parameters will not be effective until you have restarted the Essbase OLAP Server. For these and additional parameter settings information see “Essbase Analytic Services System Administrator’s Guide”

To improve Hybrid OLAP environment using Teradata Database features, see Teradata Database Tuning section (i.e. Aggregate Join Index feature).

Essbase Integration Services Tuning

EIS.CFG File

Whether you start Essbase Integration Services from a command prompt (i.e. olapisvr) or from a Windows Desktop Start menu, here are a few settings that might be of help. The following parameters can be set in the EIS.CFG file located in the EssIntegration\bin directory.

-N To specify the number of threads Integration Server uses when sending data to Analytic Services during a data load, type -N*number_of_threads* when you start Integration Server. The default setting is 1 thread.

The -N switch may be used concurrently with the -C switch.

The number of threads allocated to the Integration Server data load optimization process is controlled by the -N switch.

For example, to set the number of threads to 3, type

olapisvr -N3

In the eis.cfg file, this parameter is specified in the following format:

[N]=3

-V Disables the validation and speeds up the saving process of model and metaoutline

[V] = 0

These parameters will not be effective until you have restarted the Essbase Integration Services. For these and additional parameter settings information see “Essbase XTD Integration Services System Administrator’s Guide” section Integration Server Startup Switches.

Hierarchies

Using Hierarchies Dimensions are usually structured to contain a hierarchy of related members. For example, for relational database users, the Time dimension might include members such as Year, Quarter, and Month. This hierarchy creates an Analytic Services outline with members such as 2004, Quarter1, and January. Hierarchies also make use of attributes to classify members logically within a dimension. (For example, a Product dimension can have attributes such as Size and Flavor.) Hierarchies keep track of how the attributes relate to one another and how they are structured within a dimension.

Using Hierarchies

You can create hierarchies for a metaoutline as you create an OLAP model. Refer to Integration Services Console online help for information on hierarchies. The sample application provided with Integration Services includes a sample metaoutline, TBC

Metaoutline. You can see the dimensions named Year, Market, Measures, Product, Scenario, and Supplier. The Year dimension contains a hierarchy—Quarter and Month.

Types of Hierarchies

There are several types of hierarchies used in data retrieval and analysis. Balanced Hierarchy

A balanced hierarchy has two or more branches in a hierarchical tree, and each member is consistent with other members at the same level in each branch. For example, in a relational database, a dimension might have a branch for the year 2003 and a branch for 2004. In each of these two branches, Q1 would be at the same level, as would the months Jan, Feb, and Mar. Time dimensions typically have balanced hierarchies. In a data warehouse, a characteristic would have analogous branches.

Unbalanced Hierarchy

An unbalanced hierarchy contains branches with unequal numbers of member levels although the parent-child relationships are usually consistent from branch to branch. For example, a Sales Personnel dimension might have a branch for Sales Manager East and a branch for Sales Manager West. Each of these two branches then might have States. The Sales Manager East, however, might have four states and the Sales Manager West might have two states. In this case, the parent-child relationships on both branches are the same, but the member levels are not equivalent. Human resource dimensions sometimes have unbalanced hierarchies.

Ragged Hierarchy

A ragged hierarchy occurs when a dimension has branches in which there are different numbers of levels. For example, a Sales Regions dimension might have one branch for North America and a branch for Europe. Both branches have member level attributes for Country, State, and City. In this example, the North America branch might have United States, Massachusetts, and Boston. The Europe branch might have Greece, Athens because the country of Greece does not have individual states in the sense that the United States does. Thus the State level for Greece is empty, causing a ragged hierarcy.

Geographical dimensions and product dimensions often have ragged hierarchies. For example, branches in a Product dimension may have member level attributes for Size, Color, Weight, and Frozen. Some of these member level attributes may be empty on products for which the **attributes are not applicable.**

SQL Override

For editing your Data Load query (i.e. data extraction query for Essbase OLAP storage part of your Analytical Framework) there is a feature called SQL Override which enables you to modify the Essbase Integration Services dynamically-generated

SQL, and allows you to replace it with a custom SQL statements. This feature can be found under Essbase Integration Services console menu bar under *Outline→User Defined SQL.*

Exploiting Teradata Functions

Ordered Analytics

CASE Statement

Derived Tables

Multiple Distinct Aggregates (V2R5+)

Note: For additional, tips and techniques refer to "Data Preparation Guide for Essbase Integration Services" located online at NCR: http://gsssystem.sandiegoca.ncr.com/tadc/tpcoe/default.htm *under Hyperion section. Though, keep in mind the following sections as it applies to Teradata.*

- *Preparing Data Sources – Introduction makes reference to relational data in the form of a star schema. This is not required if Semantic Layer is properly defined for any PDM design.*

  Source of Data / Defining User Needs / Defining Datasource Needs – refer to the following sections within this document

  - *Creating an ODBC datasource*
  - *Use of Teradata XViews*
  - *MaxResponseBufferSize (8K to 65K)*
  - *Metadata Repository Considerations*
- *Consolidation Data into a Single Source – Not, really an issue for Teradata..since that is what we preach.*
- *Deciding whether to use a Staging Area – Again, NA for a Teradata environment. That is the purpose re: Semantic Layer (i.e. view database)*
- *Creating Views, Tables and User-Defined Tables – Again, purpose for Semantic Layer*
- *Denormalizing Source Data Tables – Denormalizing should be the 'very' last resort. And most likely isn't required if Semantic Layer is properly defined. Plus, using AJI should prevent having to denorm to address performance.*

Teradata Database Tuning

Primary Index Considerations for Data Distribution

Teradata is parallel database system where the data is distributed across a set of nodes, called AMPs. Teradata processes queries in parallel, with each AMP doing a portion of the work.

A primary index is required for all Teradata Database tables. If you do not assign a primary index explicitly when you create a table, then the system assigns one automatically. Data accessed using a primary index is always a one-AMP operation because a row and its primary index are stored together in the same structure. This is true whether the primary index is unique or nonunique, and whether it is partitioned or nonpartitioned.

The primary index has four purposes:

- To define the distribution of the rows to the AMPs.

  The Teradata Database distributes table rows across the AMPs on the hash of their primary index value. The determination of which hash bucket, and hence which AMP the row is to be stored on, is made solely on the value of the primary index.

  The choice of columns for the primary index affects how even this distribution is. An even distribution of rows to the AMPs is usually of critical importance in picking a primary index column set.

- To provide access to rows more efficiently than with full table scan.

  If the values for all the primary index columns are specified in a DML statement, single-AMP access can be made to the rows using that primary index value.

  With a partitioned primary index, faster access is also possible when all the values of the partitioning columns are specified or if there is a constraint on partitioning columns.

  Other retrievals might use a secondary index, a hash or join index, a full table scan, or a mix of several different index types.

- To provide for efficient joins.

  If there is an equality join constraint on the primary index of a table, it may be possible to do a direct join to the table (that is, rows of the table might not have to be redistributed, spooled, and sorted prior to the join).

- To provide a means for efficient aggregations.

  If the GROUP BY key is on the primary index of a table, it is often possible to perform a more efficient aggregation.

Collecting Statistics

One of the most important sources of information the Teradata optimizer uses for choosing access plans is the set of statistics Teradata collects about the system and the data in it. Database statistics include "data demographics" such as the number of rows in each table, the number of unique values within each column, the skew and distribution of values within each column, etc. The optimizer uses this information when selecting the best access plan from among all the possible access plans that could satisfy a given query.

It is important for the statistics to be updated as frequently as practical. Whenever new data is loaded or new indexes are built or any other significant change occurs to data, the statistics should be updated to reflect the change. Note that out-of-date statistics can be worse than not collecting statistics at all.

In Teradata, statistics are particularly important because the optimizer does not allow the user to "override" its decisions through constructs like hints or rewritten SQL. The optimizer will select the best plan for a given query, but its decisions are only as good as the information it uses to make them.

Aggregate Join Index

Pre-aggregating data is a common and powerful way to improve end user query response times. Rather than aggregating many, many rows at query time, pre-summarization allows the database to perform row access and aggregation ahead of time, satisfying the query at request time much faster. Data warehouse practitioners have long used physical summary tables to pre-aggregate data in a ROLAP environment. This is one way, however, not all tools and applications are aggregate aware. Hence, requires modification to the application or reporting metadata catalog use the physical tables. But not when using Teradata aggregate join indexes.

Teradata supports aggregate join indexes, a database server feature that provides for the creation, maintenance, and automatic navigation of aggregate tables or pre-joined tables. This makes the use of aggregate tables entirely transparent to the end user and to the application or tool. With aggregate join indexes, Teradata will rewrite the query to use pre-joined tables, provided that the optimizer determines its cost to be less than the original query. Query rewrite in Teradata is based on the SQL text itself or on rewrite conditions explicitly determined by Teradata.

Hence, any application or tool generated SQL when submitted to Teradata will be available for query rewrite. Provided that the proper conditions are set up on Teradata, Applications and tools will transparently take advantage of aggregate join indexes.

For more information refer to the Teradata documentation at www.teradata.com or see document *"Improve your OLAP Environment with Hyperion and Teradata"* at http://gsssystem.sandiegoca.ncr.com/tadc/tpcoe/default.htm under Hyperion section.

Capturing an ODBC Trace

Basically, there are two types of ODBC tracing available, Driver Manager and Data Source Name (DSN) tracing. Once set you need to restart your application to capture the trace. How to set these tracings on Windows and UNIX follows:

Windows – On Windows platform the ODBC Administrator manages the Data Sources available to the Window user.

Driver Manager Tracing
To enable Driver Manager tracing on Windows, you need to click on “Start Tracing Now” under Start Menu→Programs→ODBC→32-bit ODBC Administrator→Tracing tab. To stop tracing, click on “Stop Tracing Now”.

DSN Tracing
To enable Data Source Name tracing on Windows, you need to set the ODBCTRACE environment variable under ‘user variables’. Right click on My Computer→ Properties→Advance→Environment Variables→click New under ‘User variables’.

Variable name: ODBCTRACE
Variable value: C:\dsntrace.txt

To stop tracing, remove environment variable.

UNIX – On UNIX platforms the *.odbc.ini* file contains the names and descriptions of the Data Sources available to the UNIX user.

Driver Manager Tracing
To enable Driver Manager tracing on UNIX, you need to edit or include the following two parameters in your *.odbc.ini* file under ODBC Options section [ODBC].

Keyword	Description
Trace=[0] or [1]	**Default = 0** **When Trace is 0**, Driver Manager tracing is disabled. **When Trace is 1**, Driver Manager tracing is enabled.
TraceFile=<*filepath*>	Enter the complete pathname of the file where trace information will be logged.

The following is an example of how to enable driver manager tracing in the ODBC Options section of the *.odbc.ini* file.

```
[ODBC]
InstallDir=/usr/odbc
Trace=1
TraceFile=/usr/odbcusr/joe/odbctrace.log
TraceAutoStop=0
```

DSN Tracing
To enable Data Source Name tracing on UNIX, you need to edit or include the following two parameters in your .odbc.ini file under your Data Source Specification or Default Data Source Specification section [testdsn].

Keyword	Description
DSNTraceEnable=[*Yes* | *No*]	**Default = No** When DSNTraceEnable is **Yes**, this option turns on tracing for the DSN. When DSNTraceEnable is **No**, or if the DSNTraceEnable entry is missing, tracing is not activated for any connection using the DSN.

DSNTraceFilePath=<*path*>	**Default = /tmp/ODBC.Trace.xxxxx** Enter the relative or absolute pathname, ***path***, of the trace file. If this entry is missing, a default pathname of */tmp/ODBC.Trace.xxxxx* is used, where *xxxxx* is the pid of the creating process.

The following is an example of a Data Source Specification Section with DSN Tracing entries included:

```
[testdsn]
Driver=/usr/odbc/drivers/tdata.so
Description=NCR 3600 running Teradata V2R5.1
DBCName=123.45.67.10
DBCName2=123.45.67.11
DBCName3=123.45.67.12
Username=odbcadm
Password=
Database=
DefaultDatabase=sales
DSNTraceEnable=YES
DSNTraceFilePath=/tmp/dsntrace.log
DSNTraceFileMaxSize=1000000
DSNTraceOverwrite=NO
DSNTraceLineNumbers=YES
```

For more information refer to the ODBC Driver for Teradata User Guide.

*Important**:** Using any of the Tracing Options can significantly degrade performance.*

Other Reporting Front-End Considerations

Hyperion Excel Add-in

Is an Excel spreadsheet add-in software program that merges seamlessly with Microsoft Excel and Lotus 1-2-3. After Hyperion Excel Add-in is installed, a special menu is added to Excel. The menu provides enhanced commands such as Connect, Pivot, Drill-down, and Calculate. Users can access and analyze data on the Hyperion Essbase OLAP Server by using simple mouse clicks and drag-and-drop operations. Spreadsheet Add-in enables multiple users to access and update data on the Hyperion Essbase OLAP Server simultaneously and supports not only drill-through access, but relational access as well via Hybrid Analysis feature of Essbase.

Potential Concerns

Alias Convention issue

This issue may not occur in most implementation sites. However, it is worth mentioning nonetheless. This incident has been reported to Hyperion and will be addressed in a future release of Essbase Integration Services product.

Issue: Depending on how you create your FACT object in the OLAP model (ie. w/ multiple database objects, view or UDT), the SQL (ie. data load query) that Integration Services generates will hiccup against any data source if the number of objects in the SELECT and FROM clause exceeds 18 objects. Reason being, the aliases convention used by Integration Services starts with ‘aa’. Hence, it will create ‘as’ which is a reserved word with most database vendors.

Workaround: (Solved) Use views to minimize number of objects, which is the preferred method. Or edit the data load SQL under Outline→User Defined SQL, sometimes referred to SQL Override. (Note that this was fixed in EIS version 7.1.2.)

Reserved Word / Double Quote issue

This issue is different than the above one. This incident has been reported to Hyperion and will be addressed in a future release of Essbase Integration Services product.

Issue: Integration Services does not generate SQL statements with double quotes around <projected list> (ie. column names). Hence, if any DDL objects contains a reserved word (ie. YEAR), Integration Services will hiccup against any data source.

Workaround: (Fixed in next release) Double quote any reserved words used in the DDL object.