A Hyperion White Paper

Hyperion System 9 BI+ Essbase Analytics Bulk Data Extraction Methodologies White Paper

Hyperion® System™ 9 BI+™ Analytic Services™ (formerly *Essbase®*) is a multidimensional database server that allows users to perform complex analysis on large data sets.

1. Introduction

Analytic Services includes two specific modules for dealing with multidimensional databases.

- ***Hyperion System 9 BI+ Essbase Analytics™*** (Essbase Analytics) stores data as blocks and is widely used for highly interactive read/write implementations, such as *Hyperion System 9 Planning™*. It additionally includes the ability to run complex procedural logic that allows storing of results and key metrics into the database. While it does provide for dynamic calculation of members at query time, the ability to perform batch-oriented calculations helps ensure that deployments can balance the need for speed-of-thought query response with delivery of complex allocations, metrics, and key performance indicators.
- ***Hyperion System 9 BI+ Enterprise Analytics™*** (Enterprise Analytics) stores data in a cell-based metaphor and is widely used for large-scale analytic applications that have high dimensionality, large numbers of members, and/or highly sparse data. It also provides for complex analytics and derived data. The model is largely based on dynamic (query time) calculation of metrics and formulas, as well as control over batch versus real-time aggregation of hierarchies. Procedural-like calculations are provided for through a "solve order" option that can be attached to formulas that must execute in a specific order. While incremental data loads are available in Enterprise Analytics, large-scale read/write is not provided by this module.

Although many tools exist for extracting data from an Analytic Services database, Analytic Services also provides several options to extract data in bulk from a multidimensional database. In Analytic Services 9.3, data export functionality is further enhanced with a calculation script-based interface.

Table of Contents

1 Introduction 1

2 Existing Data Extraction Modules 2

3 Calculation Script-Based Data Export 2

3.1 Design Goal 2

3.2 Usage of the DATAEXPORT Calculation Command 3

3.3 Optimization and Fine-Tuning Techniques 4

4 Comparison of Data Extraction Methods 5

4.1 Bulk Export Capabilities in MaxL 5

4.2 Report Writer Script-Based Export 5

4.3 Export Using CDF @JExport 5

4.4 Calculation Script-Based DataExport 6

5 Summary 6

Appendix A Table of Common Use Cases 7

Appendix B Sample Scripts and Output 9

Appendix C Performance Test Matrix 11

This white paper explains the design goal and details about the new data export functionality available in the Essbase Analytics 9.3 module. Additionally, this paper will also carry out a comparative study of the new data export feature versus other existing options for exporting data from Essbase Analytics.

2. Existing Data Extraction Modules

Analytic Services provides several methods that can be used to extract data from a multidimensional database server.

- The bulk export capabilities from *Hyperion System 9 BI+ Analytic Administration Services™* (Analytic Administration Services) and the MaxL linguistic command interface allow data from Analytic Services to be exported to a flat file located on the database server. These interfaces provide a fast method to export all the physical data stored in the database server. The export file may later be imported back into Analytic Services as a free-form data input file without using a dataload rules file.

  This method is best suited to export the stored data from the entire database. It can utilize multiple export threads to write out the data in parallel. However, the scoping options are limited to level-0, input-level, or all data. The database is set in read-only mode during the export that disallows any updates until the export process has completed. The native export command does not export dynamically calculated data, and cannot be limited to a single slice of data.

- Analytic Services provides a report script interface via Report Writer, which many customers use to export data from a database. A Report Writer-based data export allows formatting of data from Analytic Services, in addition to filtering and other functionality. Therefore, this tool provides excellent formatting capabilities to produce customized reports. The output is sent to the client machine that initializes the query request.

  This method can scope the export to any slice of data from the database. Report Writer scripts provide many formatting commands for easier import to other tools, such as a relational database for archiving. However, writing report scripts with optimal performance requires good understanding of the database outline. Extracting large amounts of data using this method may encounter performance issues due to retrieval buffer size limit and probing data blocks that do not exist in the database.

- Since Essbase Analytics provides a specific interface for performing procedural calculations, an additional option exists for exporting data from Essbase Analytics databases. The Custom-Defined Function (CDF) framework allows plug-in Java functions to be executed inside the calculation engine. One of the popular custom-defined functions used for data export is @JExport, and a sample is available on the Hyperion Developer Network (http://dev.hyperion.com).

  This method can scope the export to any part of the database, and the calculation engine is highly efficient at accessing and reading blocks of data. Formatting options also can be added to @JExport to produce customized output for importing to other tools. However, getting data from dynamic calc members requires formula manipulation, which adds to the complexity and performance overhead.

3. Calculation Script-Based Data Export

Native calculation script-based data export for Essbase Analytics is a new feature introduced in Analytic Services 9.3. This new feature adds data export capability as a native interface to the Analytic Services calculation engine. A new calculation command, DATAEXPORT, is now available to be used within any calculation script, and extends the functionality of the calculation engine. It is important to note that this feature is available for Essbase Analytics only, and is not available for Enterprise Analytics databases.

3.1 Design Goal

The design goal of the DATAEXPORT command is to leverage the fast data processing capability and the rich member selection syntax offered by the calculation engine, while implementing it into an interface already used by customers. The new feature is designed to match the flexibility offered by the Custom-Defined Function @JExport and the bulk data extraction performance from the native export option using MaxL.

3.2 Usage of the DATAEXPORT Calculation Command

The following section highlights the major usage and feature options for the new DATAEXPORT calculation command.

Exporting to Different Targets

- *Flat File*

  This is the basic export functionality that mimics the export functionality provided by Analytic Administration Services and MaxL shell scripts. Use the "File" option of the DATAEXPORT command to indicate the export target as a flat file in ASCII format.

- *Relational Database*

  The DataExport feature can insert the exported data records directly into a relational database. For security reasons, the DATAEXPORT command does not update the relational schema by creating new tables. The schema of the relational table or view needs to match the data records to be inserted. The target relational database can be specified using data source name (DSN). Use the DSN option of the DATAEXPORT command for this purpose. The data records are inserted in batches to the relational databases if the support is available in the ODBC driver for that particular RDBMS.

- *Binary Output File*

  The DataExport feature can generate a binary output file in a compressed Essbase Analytics internal format for quick back-up and recovery of the database. The generated binary file can only be imported through the calculation script using the DATAIMPORTBIN command. Since the binary output option is based on block number, any outline changes may make the binary output unusable. The binary output contains the outline timestamp at the time of the export. The import operation fails if the outline timestamp at the time of the import does not match the one stored in the binary file. An option is provided to bypass the timestamp check. One valid reason to bypass the timestamp check could be when the same outline is being used to create multiple cubes at different times. Another reason could be when an outline change does not affect block numbers such as modifying a member formula or changing attribute associations. The BINFILE option of the DATAEXPORT command enables the binary export feature.

Scoping and Formatting the Export Data

The following are the various options that can be used with the DATAEXPORT command to generate the desired output:

- *Range of Data Blocks*

  The user has the flexibility to choose the range of blocks for which the data export is requested. The export can be done either for input-level, level-0, or all blocks. Use DATAEXPORTLEVEL option for this purpose. This command provides the same functionality as the bulk export capability from MaxL regarding stored data. The DATAEXPORT command is more flexible because it can include dynamically calculated data. The user can also use the FIX statement in the calculation script to specify the slice of the database to be exported.

- *Precision of Data Values*

  The data value in Essbase Analytics is stored in double precision format. DataExport provides the functionality to either specify the overall precision of the data value required or specify the decimal precision required. Use the options DATAEXPORTPRECISION and DATAEXPORTDECIMAL for this purpose.

- *Changing the Data Focal Point*

  The focal point around which the output data is formatted can be changed using the DATAEXPORTCOLHEADER option. Using this command, the user can specify different dense dimensions to be used. This command enables the user to specify which dense dimension is the primary dimension around which the other data is referenced. The dimension specified here becomes the column header of the output data.

- *Specifying Dimension Names in File Header*

  The user can use an option to indicate whether the exported data includes a header record that has all dimension names in the same order as the data in the file. Use DATAEXPORTDIMHEADER option for this purpose.

- *Exporting Dynamically Calculated Data*

  The user has the option to specify whether the exported data includes dynamically calculated data. Dynamically calculated data is associated with the members of the outline marked as "Dynamic Calc" and "Dynamic Calc and Store." It is important to note that if you are using "Dynamic Calc and Store," this will cause the calculation of these members and the results to be stored in the database, thereby increasing the database disk footprint. Use the DATAEXPORTDYNAMICCALC option for this purpose.

- *Data in Import-Ready Format for Relational Databases*

  Use this method when direct export to a relational database is not possible. This option exports the data in a format that can be imported into the RDBMS as a flat file. This option is useful in offline transfer of data to a relational table. Use the DATAEXPORTRELATIONALFILE option for this purpose.

Performing Conditional Data Export

Conditional data export is supported through the DATAEXPORTCOND command. DATAEXPORTCOND specifies conditions that identify records to be exported based on field values. The "ReplaceAll" option specifies that all records are exported but those records that do not meet the specified conditions are marked as #NoValue. When "ReplaceAll" is not specified, only those records that meet the conditions are exported.

Performing "Dry-Run" of DataExport

This method enables running the calculation script with DataExport in test mode without actually exporting the data. The option validates the set of calculation commands and gives the export statistics, including the elapsed time estimate. This option gives a summary of data export settings, messages reported in the application log, and number of blocks to be exported and estimated export time. All the information is logged into the file selected by the user. Use the DATAEXPORTDRYRUN option to enable this option.

3.3 Optimization and Fine-Tuning Techniques

The following highlights considerations and optimization techniques when using the DATAEXPORT command. This information should be referenced to help ensure optimal implementation of the DATAEXPORT command.

Exporting to Different Targets

a) DataExport exports dynamically calculated data by using a query-based extraction method. Query-based export can be slower as the calculator engine has to get all data blocks specified in the FIX range and perform dynamic calculations on them if Dynamic Calc members are involved. Excluding dynamically calculated data improves the export performance because the calculator engine only needs to fetch existing data blocks from the database. Users need to remember these performance implications when using DATAEXPORTDYNAMICCALC. If the goal of using the DATAEXPORT command is to backup and provide restore capabilities for your database, then exporting dynamic calc members is not necessary, as trying to load dynamic calc data into the database is not accepted.

b) Exporting data to a flat file is generally faster compared to using the custom-defined function @JExport. DataExport exports stored data as-is, while @JExport is invoked from a member formula. This also requires the data to be passed to the Java process in order to export. This requirement adds complexity to the usage of @JExport.

c) Binary export/import is faster than the other forms of export as it does not require textual data transformation. This mode of export/import is preferred for fast back-up and recovery of data. The restriction is that the binary export/import only works against databases with the exact same outline. Any changes to the outline will prevent the binary data import to continue unless the timestamp check is turned off explicitly. The performance advantage of binary export is demonstrated in Appendix C.

d) Exporting data from stored members should have comparable performance with the native export utility in MaxL.

e) Exporting directly to a relational database uses a batch-insert mechanism, which is dependent on RDBMS as well as the corresponding ODBC drivers. By default, the batch size is set at 10 records for each insertion. Users can specify DEXPSQLROWSIZE in Essbase.cfg to modify the number of records for each insertion. If the RDBMS/ODBC combination does not support batch-insert, DataExport defaults to single record-based insertion.

f) The DATAEXPORT command can be a very useful tool to debug batch calculation where a snapshot of the database can be preserved before and after the calculation to trace data changes during the calculation.

4. Comparison of Data Extraction Methods

The following is a comparative study in terms of functionality and usage of DataExport with existing bulk data extraction methodologies available for Essbase Analytics.

4.1 Bulk Export Capabilities in MaxL

The DATAEXPORT calculation command and the bulk export utility in MaxL use almost the same back-end functionality. As a result, all options available in the bulk export utility are also available in DataExport. One significant difference is that the bulk export can expose options to multithread the export, whereas DataExport is single threaded. The big advantage offered by DataExport is its capability to export dynamically calculated data, as well as additional options for the target, filtering, and conditioning logic.

Advantages of MaxL-based bulk export

- Support level-0 data export against Aggregate Storage database

Disadvantages of MaxL-based bulk export

- Only supports full database export
- Database in read-only mode during export
- No ability to export dynamically calculated data

4.2 Report Writer Script-Based Export

Because Report Writer script-based export provides much more flexibility in formatting the data, it is ideal for generating professional-looking reports. DataExport provides fewer formatting options. Report Writer scripts also provide extensive selection criteria. The scripts work with all member types (stored member, dynamic calc member, attributes, label only members, shared members, aliases, etc.) while DataExport works only with stored and dynamically calculated members.

DataExport can outperform Report Writer script-based export when extracting data from stored members. As the Report Writer engine always uses query-based data extraction, these scripts are generally slower due to the performance overhead of probing data blocks that are not materialized in the database.

DataExport fetches data blocks in the same order as they are organized in the multidimensional database. Accessing data blocks in ascending order helps ensure the minimal seek time for the export operation. The block accessing order from Report Writer scripts are determined by the order of dimensions and members from a particular script. Unlike DataExport, which obtains the relevant block only once, a Report Writer script may fetch the same data blocks multiple times if sparse dimensions appear on columns and dense dimensions appear on rows.

Advantages of Report Writer script exports

- Rich formatting capability offered by scripting language
- Dimension orders and member orders fully customizable
- Support for attribute members and alias names
- Support for data extraction against both Essbase Analytics, as well as Enterprise Analytics databases

Disadvantages of Report Writer script exports

- Optimal performance requires a good understanding of the database outline structure
- Report output with customized formatting cannot be loaded directly into an Analytic Services databases without removing the formatting

4.3 Export Using CDF @JExport

Users can extend calculation functionality by using the custom-defined function (CDF) framework provided by the Essbase Analytics calculation engine. In this method, the custom-defined functions written in Java can be plugged into the calculation module. The @JExport CDF function is popular in the Analytic Services community because it can export data in a customized format. As the custom-defined functions are extensions to the Essbase Analytics calculator framework, they operate less efficiently than functions that are native to the calculator framework. The DATAEXPORT calculation command performs faster than @JExport in most cases because DataExport does not have the overhead of being an extended calculation function. One important limitation to @JExport is that it has to be attached to a member as part of the member formula. DataExport works as a stand-alone calculation command, and the export scope is defined by the FIX/ENDFIX statement. The CDFs tend to have additional memory requirements as they are written in Java with the overhead of invoking JVM.

Advantages of using @JExport

- Integrates tightly with calculator functions and commands
- Output format fully customizable as the handling can be added using Java language
- Allows the export target to be a relational database via JDBC

Disadvantages of using @JExport

- Requires attachment to a member formula
- Deployment requirement (JVM, CDF registration and configuration)

4.4 Calculation Script-Based DataExport

The following are some of the important points that summarize the overall strengths and weaknesses of DataExport.

Advantages of calculation script-based DataExport

- Supports multiple targets: flat file, relational database, and binary file
- Can be used to create a snapshot for database backup or to restore or roll back the database because the exported data can be imported using free-form dataload feature of Analytic Services
- As part of the calculation command, can be easily deployed in a batch process to provide lights-out operations
- Can export dynamic calculation members, which has been a limitation with the native export utility in MaxL
- If the dynamic calculation export option is not used, then DataExport exports the data by directly accessing kernel storage blocks; this is extremely fast compared to any other way of accessing Essbase Analytics data
- Using batch-insert technology, DataExport provides a faster option for exporting data to a relational table
- The binary export/import option provides a faster way to back-up and restore data from/to an Essbase Analytics database. The compressed format provided by binary export requires less storage for the export files
- Can be used as a good tool to debug and trace batch calculation. The command can be used before and after other calculation commands to track the exact data changes during the batch calculation

Disadvantages of calculation script-based DataExport

- Contains limited data formatting options compared with Report Writer
- Exporting dynamic calculation data can have significant performance impact
- No streaming support
- No support for Enterprise Analytics databases
- Cannot export data directly to the client
- Work only with stored members and dynamically calculated members in the outline. No support on attribute members and alias names

5. Summary

The new calculation script-based DataExport command is an important new feature available for Essbase Analytics deployments, and provides an optimized way to backup, restore, and integrate data with other systems in your environment. It improves the flexibility and capability of bulk data extraction, while using the performance and semantics of the calculator engine.

Appendix A provides a common use case table that describes advantages and disadvantages for using each tool in common situations. Appendix B contains sample scripts and output to show the basic syntax for the extraction methods. Appendix C showcases the performance advantage of the binary export option using DATAEXPORT calculation command.

APPENDIX A: Table of Common Use Cases

The following table summarizes the common use cases related to extracting data out of *Hyperion System 9 BI+ Analytic Services* and the pros and cons of various export methodologies.

Table 1: Data Extraction Tools

Use Case	Tool	Advantage/ Disadvantage	Suggestions
Backing up and restoring data from the entire database	Export command in MaxL	Export only stored members in the outline. The exported file can be imported easily with the free-form data load.	Use the native export utility in MaxL if the database outline may change. The export file is editable to resolve potential outline change issues. Use DataExport binary export/import feature if no outline change is expected and the speed of the operation is a priority.
	@JExport custom-defined function	Not a good choice. This function has higher execution overhead because it works on only one data cell at a time.	
	Report Writer script	Can be used if the data is required in a particular custom format. All member types including stored member, dynamic calc member, label only, shared member, etc., can be exported.	
	DATAEXPORT calculation command	Can be used for this purpose. Dynamic calc members can be exported along with stored members.	
Highly formatted reports	Export command in MaxL	Does not have much formatting capability.	Use report-script-based export to generate formatted industry report. Use DataExport when minimal formatting is required and bulk data reports are to be generated.
	@JExport custom-defined function	Not designed to generate highly formatted data. Being a CDF, it is flexible and can be extended to add more formatting options.	
	Report Writer script	The powerful report commands can be used to create custom formatted report.	
	DATAEXPORT calculation command	Not designed to generate formatted reports. It can be used to generate reports with minimal formatting.	
Ad-hoc analysis	Export command in MaxL	Cannot be used effectively for ad-hoc analysis.	Use Essbase spreadsheet add-in or *Hyperion System 9 Smart View for Office™*. They are best suited for ad-hoc analysis. The bulk data extraction tools discussed in this white paper can be used with limited functionalities.
	@JExport custom-defined function	Cannot be used effectively for ad-hoc analysis.	
	Report Writer script	Can be used to generate reports based on ad-hoc analysis data.	
	DATAEXPORT calculation command	Can be used to generate simple reports for ad-hoc analysis data. A comparative report can be generated before and after inserting the ad-hoc analysis data.	

Exporting data directly to relational databases	Export command in MaxL	Cannot directly export data to relational databases.	Use DataExport for a large-volume transaction. Use @JExport for transactions pertaining to few data members.
	@JExport custom defined function	Supports data insertion to relational databases via JDBC.	
	Report-script-based export	Cannot directly export data to relational databases.	
	DATAEXPORT calculation command	Supports data insertion to relational databases via ODBC.	
Exporting a subset of the database	Export command in MaxL	Allows selection range only at block level. No individual selection can be made at the member level.	Use DataExport as this is the tool designed for this purpose. Use report-script-based export if more formatted report is required.
	@JExport custom-defined function	Can be used with FIX command to export a subset of the database.	
	Report Writer script	Can indicate the member ranges to export data. Need to pay attention to member and dimension order to get optimal performance.	
	DATAEXPORT calculation command	The FIX command can be used to limit the scope of the data export.The exported data can also be imported without any change through free-form data load.	
Collecting calculation statistics without performing the real calculation	Export command in MaxL	Not supported.	Use the dry-run feature of DataExport.
	@JExport custom-defined function	Not supported.	
	Report Writer script-based export	Not supported.	
	DATAEXPORT calculation command	Provides a dry-run option that outputs statistics including the estimated export time.	
Exporting dynamic calculation data	Export command in MaxL	Not supported.	Use DataExport or @JExport if only dynamically calculated members are needed. Use Report Writer script-based export to extract data from all member types including label-only, shared, and attribute members.
	@JExport custom-defined function	Can export dynamically calculated members.	
	Report Writer script	Can export dynamically calculated members.	
	DATAEXPORT calculation command	Can export dynamically calculated members.	

APPENDIX B: Sample Scripts and Output

The following samples are based on the Essbase Analytics Sample:Basic database distributed as part of the sample applications from the Analytic Services installation.

1. DataExport from Calculation Script

Calculation script with DataExport:

```
//ESS_LOCALE English_UnitedStates.Latin1@Binary

SET DATAEXPORTOPTIONS
{
DATAEXPORTLEVEL "ALL";
DATAEXPORTCOLHEADER "Measures";
DATAEXPORTDYNAMICCALC OFF;
DATAEXPORTCOLFORMAT OFF;
};

FIX ("100-10", "New York")
DATAEXPORT "File" "," "E:\temp\NY_100_10.txt" ;
ENDFIX;
```

Output:

```
"Sales","COGS","Marketing","Payroll","Misc","Opening Inventory","Additions","Ending Inventory"
"100-10","New York"
"Jan","Actual",678,271,94,51,0,2101,644,2067
"Feb","Actual",645,258,90,51,1,2067,619,2041
"Mar","Actual",675,270,94,51,1,2041,742,2108
"Apr","Actual",712,284,99,53,0,2108,854,2250
….
```

2. Bulk MaxL-Based Export of 'ALL' Data

MaxL script:

```
export database sample.basic all data to data_file 'd:\\temp\\temp.txt';
```

Output:

```
"Sales" "COGS" "Marketing" "Payroll" "Misc" "Opening Inventory" "Additions" "Ending Inventory"
"100-10" "New York"
"Jan" "Actual" 678 271 94 51 0 2101 644 2067
"Feb" "Actual" 645 258 90 51 1 2067 619 2041
"Mar" "Actual" 675 270 94 51 1 2041 742 2108
"Apr" "Actual" 712 284 99 53 0 2108 854 2250
….
```

3. Report Writer Script-Based Export

Report script:

```
//ESS_LOCALE English_UnitedStates.Latin1@Binary
<COLUMN (Measures)
<ROW (Year, Scenario)

"Sales" "COGS" "Marketing" "Payroll" "Misc" "Opening Inventory" "Additions" "Ending Inventory"
"New York"
"100-10"
"Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov" "Dec"
"Actual" "Budget"
!
```

Output:

```
                               New York 100-10
          Sales      COGS    Marketing   Payroll    Misc     Opening   Addition    Ending      I
=====================================================================================================
Jan   Actual      678       271        94         51         0         2,101      644       2,067
      Budget      640       260        80         40      #Missing     2,030      600       1,990
Feb   Actual      645       258        90         51         1         2,067      619       2,041
      Budget      610       240        80         40      #Missing     1,990      600       1,980
Mar   Actual      675       270        94         51         1         2,041      742       2,108
      Budget      640       250        80         40      #Missing     1,980      700       2,040
Apr   Actual      712       284        99         53         0         2,108      854       2,250
      Budget      670       270        80         40      #Missing     2,040      800       2,170
….
```

4. Custom-Defined Function @JExport

Calculation script with @JExport:

```
//ESS_LOCALE English_UnitedStates.Latin1@Binary
Set updatecalc off;
Fix ("actual")
 Sales (
   IF ("variance"< 0)
@JExportTo("file","E:/temp/jexp1.txt",",","",@List(@NAME(@currmbr(Market)),@NAME(@currmbr(Product)),@NAME(@
currmbr(measures)),@NAME(@currmbr(year))), @LIST(actual,budget,Variance) ) ;

  ENDIF;
        )
 EndFix;

 RUNJAVA com.hyperion.essbase.cdf.export.CloseTarget "file" "E:/temp/jexp1.txt";
```

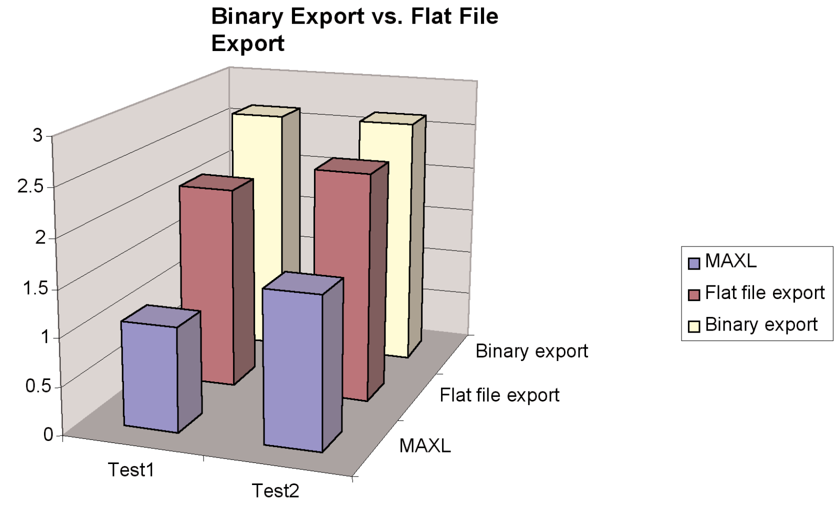
Output:

```
"New York","400-10","Sales","Jan",234.00,250.00,-16.00
"New York","400-10","Sales","Feb",232.00,250.00,-18.00
"New York","400-10","Sales","Mar",234.00,250.00,-16.00
"New York","400-10","Sales","Qtr1",700.00,750.00,-50.00
….
```

APPENDIX C: Performance Test Matrix

The following test compares the data export performance using binary export versus regular flat file export in DataExport. Binary export stores data in a compressed binary format and can be used to backup and restore a subset of database image quickly.

	Test 1			Test 2		
	Full database export after loading a 500 MB data file			Full database export after loading a 500 MB data file and running the default calc		
	Export file size (MB)	Time(sec)	Speed (MB/sec)	Export file size (MB)	Time(sec)	Speed (MB/sec)
Binary export using DataExport	540	204	2.65	5300	1995	2.66
Flat file export using DataExport	776	363	2.14	9792	4023	2.43
Export command from MaxL	776	713	1.09	9792	6214	1.58



Hyperion Solutions Corporation Worldwide Headquarters
5450 Great America Parkway, Santa Clara, CA 95054
voice 1.408.588.8000 / fax 1.408.588.8500 / www.hyperion.com

product information voice 1.800.286.8000 (U.S. only)
consulting services e-mail *northamerican_consulting@Hyperion.com* / voice 1.203.703.3000
education services e-mail *education@Hyperion.com* / voice 1.203.703.3535
worldwide support e-mail *worldwide_support@Hyperion.com*

Please contact us at *www.Hyperion.com/contactus* for more information.

Hyperion™
The future in sight