

Oracle® Hyperion Smart View for Office, Fusion Edition

User's Guide

RELEASE 11.1.2.1

ORACLE®

**ENTERPRISE PERFORMANCE
MANAGEMENT SYSTEM**

Smart View User's Guide, 11.1.2.1

Copyright © 2004, 2011, Oracle and/or its affiliates. All rights reserved.

Authors: EPM Information Development Team

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited. The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS:

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Documentation Accessibility	11
Chapter 1. Introduction to Smart View	13
Smart View Components	13
Ribbons	13
Smart View Panel	14
Chapter 2. Managing Data Source Connections	15
Connections	15
Connection Types	15
Connecting to Data Sources	16
Disconnecting from Data Sources	16
Creating Private Connections	17
Saving Shared Connections as Private Connections	18
Adding and Deleting Servers	18
Adding Essbase Servers	18
Deleting Servers	19
Disabling Smart View	19
Chapter 3. Smart View General Operations	21
Smart View Operations	21
Using Undo and Redo	21
Copying and Pasting	22
Importing Metadata into Copied Worksheets	22
Copying Data Between Excel, Word, and PowerPoint	23
Sheet Information	24
Searching in Smart View	24
Enabling Advanced Logging	24
Shared Workbooks	25
Printing POV Members in Header and Footer	25
Chapter 4. Ad Hoc Analysis	27
About Ad Hoc Analysis	27

Starting Ad Hoc Analysis	27
Zooming In and Out	28
Formatting Ad Hoc Grids	28
Using Excel Formatting	28
Preserving Formats While Zooming	29
Removing Members From the Grid	29
Other Ad Hoc Options	29
Pivoting	30
Cascading Reports and Ad Hoc Grids	30
Substitution Variables	31
Inserting Rows and Columns	31
Chapter 5. Data Forms	33
Working with Data Forms in Excel	33
Opening Data Forms in Excel	34
Planning Data Forms	34
Financial Management Data Forms	35
Selecting Members	35
Adding Members	35
Using Financial Management Linked Forms	35
Customizing Data Forms	36
Chapter 6. Selecting Members	37
Dimensions and Members	37
Selecting Members	37
Selecting Members from the POV	39
Filtering by Attribute	40
Filtering by Subsets	41
Selecting Period-to-Date Members	41
Member Perspective	42
Aliases and Alias Tables	43
Qualified Member Names	43
The POV Manager	44
Selecting Members for the Default POV	44
Copying and Pasting a POV	45
Deleting a POV	45
Chapter 7. Working with Data	47
Submitting Data	47
Refreshing Data	48

Working with Excel Formulas	48
Excel Formulas in Ad Hoc Grids	48
Excel Formulas in Data Forms	49
Adjusting Values in Data Cells	49
Cell Comments	49
Document Attachment	50
Calculating Data	50
Calculating Data in Financial Management and Hyperion Enterprise	50
Calculating Data in Essbase	51
Consolidating Data	51
Working with Currencies	52
Translating Currencies in Financial Management and Hyperion Enterprise	52
Changing Currency in Planning	52
Drill-Through Reports	53
Data Perspective	54
Chapter 8. Reporting with Smart Slices	57
About Smart Slices	57
Creating Reports with Smart Slices	57
Deleting Reports	59
Sliders	59
Smart Slices, Ad Hoc Analysis, and Data Forms	60
Creating Smart Slices	61
Setting Smart Slice Data Boundaries	61
Setting Smart Slice Preferences	62
Chapter 9. Reporting with the Query Designer and MDX Queries	63
The Query Designer	63
Creating Queries	63
Editing Queries and Rerunning Reports	64
Filtering Data	65
Analyzing Time-Related Data in Query Designer	65
MDX Queries	66
Chapter 10. Task Lists	67
Task Lists	67
Working with Tasks from the Smart View Panel	67
Opening a Task List	67
Viewing the Task List	68
Executing a Task	68

Completing a Task	69
Creating Task List Reports	69
Integrating Task Lists with Microsoft Outlook	69
Chapter 11. Smart View and Planning	71
Planning Approvals	71
Changing Planning Unit Status	71
Finding Planning Units	72
Planning Unit Promotional Path	73
Planning Unit Annotations	74
Out of Office Assistant	74
Monitoring Planning Job Status	74
Searching for a Page in Planning	75
Copying Versions	75
Composite Data Forms	76
Working with Planning Business Rules	76
Launching Business Rules in Excel	77
Entering Runtime Prompts	77
Executing the Calculate Data Form and Calculate Currencies Business Rules	78
Spreading Data for Time Periods	79
Spreading Data with Cell Locking	79
Spreading Values Using Grid Spread	80
Spreading Values Using Mass Allocation	81
Member Formula	81
Supporting Detail	82
Adding Supporting Detail	82
Working with the Supporting Detail Hierarchy	82
Viewing or Changing Supporting Detail	83
Synchronizing Supporting Detail with Essbase	83
Setting Planning Preferences	84
Working Offline	84
Taking Data Forms Offline	84
Working with Data Forms Offline	86
Synchronizing Data to the Planning Server	86
Refreshing the Offline Data Form Definition and Data	87
Chapter 12. Smart View and Reporting and Analysis	89
Importing Reporting and Analysis Documents	89
Editing and Refreshing Documents	90
Refreshing Reporting and Analysis Documents	90

Financial Reporting and Web Analysis Import Formats	91
Adding Security Certificates for SSL-enabled EPM Workspace Servers	92
Importing Interactive Reporting Documents	92
Importing Interactive Reporting Documents into Excel	93
Importing Interactive Reporting Documents into Word and PowerPoint	94
Editing Interactive Reporting Documents	95
Importing Financial Reporting Documents	95
Importing Financial Reporting Documents into Excel	96
Importing Financial Reporting Documents into Word and PowerPoint	97
Editing Financial Reporting Documents	98
Creating Templates in PowerPoint Documents	98
Refreshing PowerPoint Templates	99
Importing Production Reporting Documents	99
Importing Production Reporting Jobs into Excel	99
Importing Production Reporting Jobs into Word and PowerPoint	100
Importing Production Reporting Job Outputs into Word, and PowerPoint	101
Editing Production Reporting Jobs	101
Importing Web Analysis Documents	102
Importing a Web Analysis Document or Document Objects	102
Editing Web Analysis Documents	103
Using Smart Tags to Import Reporting and Analysis Documents	104
Chapter 13. Smart View Global Options	105
Setting Options	105
Member Options	105
Data Options	106
Advanced Options	108
Formatting Options	109
Cell Styles	110
Extensions	110
Provider Preferences	111
Chapter 14. Functions	113
Using Functions	113
Creating Functions	113
Creating Functions in the Function Builder	113
Creating Functions Manually	114
Running Functions	115
Viewing Function Grids as Ad Hoc Grids	116
Function Descriptions	116

HsGetValue	116
HsSetValue	116
HsGetSheetInfo	117
HsCurrency	118
HsDescription	118
HsLabel	118
HsGetText	119
HsSetText	119
Accessing Functions with a Smart Tag	119
Common Function Error Codes	120
Chapter 15. VBA Functions	121
VBA Functions	121
Using VBA Functions	121
Declaring Functions	122
Calling Functions	122
VBA Functions and Dynamic Link Views	122
VBA Parameters	124
VBA Return Values	125
VBA Functions	128
Visual Basic Menu Functions	232
Using Spreadsheet Toolkit VBA Applications in Smart View	253
Chapter 16. Using Free-Form Mode	255
About Free-Form Mode	255
Free-Form Guidelines	256
Entering Comments	257
Entering Dynamic Time Series Members	258
Preserving Comments, Formulas, and Format	258
Valid Simple Grid	259
Valid and Invalid Grids	259
Creating a Free-Form Report	264
Resolving Dimension Names	264
Retrieving Attribute Dimensions in Free-Form Mode	265
Retrieving Data into Asymmetric Reports	266
Appendix A. Accessibility	267
Enabling Accessibility for Smart View	267
Keyboard Equivalents	267
Smart View Ribbon Keyboard Equivalents	268

Data Provider Ribbons	270
Office 2003 Menu Keyboard Equivalents	273
Smart View Panel Navigation	276
Appendix B. Using Other Applications with Smart View	279
Crystal Ball EPM	279
Working with Crystal Ball EPM Workbooks	279
Toolbar Operations	280
Visual Explorer	280
Viewing Visual Explorer Data from Microsoft Office	280
Visual Explorer File Association	281
Smart View and Spreadsheet Add-in	281
Migrating Functions	281
Converting Workbooks	282
Converting One Workbook	282
Converting Multiple Workbooks	283
Migrating Connections for Functions	283
Appendix C. Finding Information	285
Information about Data Sources and Other Products	285
Using Oracle User Productivity Kit	285
Index	287

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at <http://www.oracle.com/accessibility/>.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Access to Oracle Support for Hearing-Impaired Customers

Oracle customers have access to electronic support through My Oracle Support or by calling Oracle Support at 1.800.223.1711. Hearing-impaired customers in the U.S. who wish to speak to an Oracle Support representative may use a telecommunications relay service (TRS). Information about the TRS is available at <http://www.fcc.gov/cgb/consumerfacts/trs.html/>, and a list of telephone numbers is available at <http://www.fcc.gov/cgb/dro/trsphonebk.html>. International hearing-impaired customers should use the TRS at +1.605.224.1837. An Oracle Support engineer will respond to technical issues according to the standard service request process.



Introduction to Smart View

In This Chapter

Smart View Components	13
Ribbons	13
Smart View Panel	14

Smart View Components

Oracle Hyperion Smart View for Office, Fusion Edition provides a common Microsoft Office interface for Oracle Essbase, Oracle Hyperion Financial Management, Fusion Edition, Oracle Hyperion Planning, Fusion Edition, Oracle Enterprise Performance Management Workspace, Fusion Edition, Oracle Business Intelligence Enterprise Edition, Oracle's Hyperion Reporting and Analysis, Oracle Hyperion Financial Close Management, and Oracle's Hyperion® Enterprise® data sources. Using Smart View, you can view, import, manipulate, distribute, and share data from these data sources in Microsoft Excel, Word, Outlook, and PowerPoint.

The basic components of Smart View, from which you connect to your data source and access Smart View functionality, are ribbons and the Smart View panel.

The components displayed depend on the Microsoft Office application that you have open.

Ribbons

Note: Smart View is designed to work optimally with the ribbon structure of Microsoft Office 2007 or later, but you can use Smart View with Office 2003 through the Smart View menu. The organization of items on this menu is analogous to that of the ribbon structure.

You access Smart View functionality in Office applications through ribbon commands. The Smart View ribbon, which contains commands for common Smart View operations and for Reporting and Analysis operations, is always present. When you connect to a data source (other than Reporting and Analysis or Financial Close Management), the corresponding data source ribbon is also displayed. Each of these ribbons displays only the commands supported for that data source and mode.

For Planning, Financial Management, and Hyperion Enterprise, when you enter ad hoc analysis (see [Chapter 4](#)), the data source ribbon is replaced by its ad hoc version. The ribbons are as follows:

- **Smart View**
- **Essbase**
- **Planning**
- **Planning Ad Hoc**
- **HFM (Financial Management)**
- **HFM Ad Hoc**
- **Enterprise (Hyperion Enterprise)**
- **Enterprise Ad Hoc**

Smart View Panel

From the Smart View Panel, you can manage data source connections, access data and task lists, create reports, and open Oracle Crystal Ball Enterprise Performance Management, Fusion Edition workbooks if you are licensed for Crystal Ball EPM or related products.

The Smart View Panel, opened from the Smart View ribbon, is displayed by default on the right side of the Microsoft Office application. You can move, resize, or close the Smart View Panel from the down arrow in the title bar.

The Smart View Panel contains the following panes:

- **Home:** A panel that displays links to Shared Connections and Private Connections as well as a list of recently used items - ad hoc grids, data forms, and tasks - which you can click to establish a connection.
- **Shared Connections:** A drop-down menu of available connections from Oracle's Hyperion® Shared Services and a tree view of the contents of the currently selected connection.
- **Private Connections:** A drop-down menu of available connections saved on the local computer and a tree view of the contents of the currently selected connection. You can also enter a URL to connect directly to a data source here.
- **Task Lists:** A tree list of tasks from which you can manage your tasks. This pane opens only when you select a task list from Shared Connections or Private Connections.
- **Simulation Workbook:** A tree list of available Crystal Ball EPM workbooks that you can open in Smart View.
- **Action Panel:** A list of operations available based on the selection in the shared connection, private connection, or task list tree list.

2

Managing Data Source Connections

In This Chapter

Connections	15
Connection Types	15
Connecting to Data Sources	16
Disconnecting from Data Sources	16
Creating Private Connections	17
Saving Shared Connections as Private Connections	18
Adding and Deleting Servers	18
Disabling Smart View	19

Connections

You connect to data sources, manage your connections, and open grids, data forms, and task lists all from the Smart View Panel.

Depending on how the administrator configured Smart View, you may or may not be required to enter your user name and password as you change data providers and Office applications or enter Oracle Essbase Visual Explorer.

Connection Types

You connect to data sources through shared or private connections.

- **Shared Connections**

Shared connections are stored in a central location and are available to multiple users through the Smart View Panel. They are created and maintained by the administrator. You cannot add, edit, or rename shared connections, but you can save them as private connections, which you can edit and rename.



- **Private Connections**


Private connections are those that you create by saving a shared connection to your local computer or by entering a URL to a provider that is not configured for shared connections. When you create a private connection, it becomes the active connection.

Connecting to Data Sources

You can connect to one data source per worksheet.

► To connect to a data source:

- 1 From the **Smart View** ribbon, click **Open**.
- 2 In **Connect to Data Source**, enter your user name and password for the data source.
- 3 From **Smart View Home** or , do one of the following:
 - Click a connection name under **Recently Used**. You can click  to pin items to this list.
 - Select **Shared Connections** to open the Shared Connections panel, where you select a data source from the drop-down menu. Connections available for the selected data source are displayed in a tree list.
 - Click **Private Connections** to open the Private Connections panel, where you select a connection from the drop-down menu.
 - Click **Private Connections**. Enter a URL in the field and press Enter. For examples of the URL syntax to use, see [“Creating Private Connections” on page 17](#).
- 4 In the **Smart View Panel** tree list, double-click the item - data form, ad hoc grid, Smart Slice, or task list - that you want to open.


After the item is opened on the grid, you can easily locate it in the tree view. Click  and select **Locate Worksheet Connection**.

Note: Essbase only: If external authentication is disabled, for security reasons, you must provide your user name and password each time you connect to a different application on the same server.

Disconnecting from Data Sources


You can disconnect from the current connection or from all connections.

► To disconnect only from the *current* connection:

- 1 From the **Smart View Panel**, select the connection that is currently open in the tree list.
- 2 **Optional:** To find this connection quickly, click  and select **Locate Worksheet Connection**.
- 3 Right-click and select **Disconnect**.

Disconnecting from the current connection does not invalidate single-sign on (SSO).

- To disconnect from *all* connected Shared Connections and Private Connections:



- 1 From the Smart View Panel, click .
- 2 Select **Disconnect All**.

This selection invalidates SSO, and you must log in again the next time you connect.

Creating Private Connections

You can create a private connection from a connection that is not listed in Shared Connections if you know the URL.

- To add a connection using a URL:

- 1 From the Smart View ribbon, click **Open**.
- 2 From the Smart View panel, click  and select **Private Connections**.
- 3 Click  and select **Create a new connection**.
- 4 Under **Location**, enter the URL or the local storage directory for the data source to which you want to connect. The URL syntax for the various data sources is as follows. Contact your system administrator for the URL to use:

Financial Management: `http(s)://servername:port/hfmofficeprovider/hfmofficeprovider.aspx`

Hyperion Enterprise: `http://servername:port/heofficeprovider/heofficeprovider.aspx`

Planning: `http(s)://servername:port/HyperionPlanning/SmartView`

Essbase: `http(s)://servername:port/aps/SmartView`


Reporting and Analysis: `http(s)://servername:port/raframework/browse/listxml`

Financial Close Management: `http://servername:port/fcc/servlets/smartview/fcmsvservlet`

- 5 **Optional:** Select **Set a default connection** (applicable only for functions; see [Chapter 14, "Functions"](#)).
- 6 Click **Next**.
- 7 In **Add Connection - Application/Cube**, a list of servers that are accessible from the URL you selected in [step 4](#) is displayed. Expand the **Servers** node and select a server.
- 8 In **Connect to Data Source**, enter the user name and password, and then click **Connect**.
- 9 Select the application and click **Next**.
- 10 In **Add Connection - Name/Description**, enter a name and description for this data source.

Note: Do not use semicolons (;) in connection names.

11 Click **Finish**.

To delete the entire list of private connections, click  and select **Clear Manually Entered URL Entries**.

Saving Shared Connections as Private Connections

Although you cannot create shared connections without administrative privileges, you can save them as private connections if they are enabled for private connections.

► To create a private connection:

- 1 From the **Shared Connections** tree list, select an item to save as a private connection.
- 2 From the **Action Panel**, select **Add to private connections**. This option is available only if the selected item is enabled for saving as a private connection.
- 3 **Optional:** From **Save as Private Connection**, edit the name and description of the connection.
- 4 Click **OK**.
- 5 The connection name is displayed in the following:
 - The **Shared Connections** tree list, indicated as private by a small arrow
 - The **Private Connections** dropdown menu


Adding and Deleting Servers

Data source types: Essbase


You must have administrative privileges for a data source to add or delete servers in the Smart View Panel.

Adding Essbase Servers

► To add an Essbase Server to the Smart View Panel:

- 1 From the **Smart View Panel**, select **Shared Connections**.
- 2 Click the down arrow and select **Oracle Essbase** to display the list of available Essbase servers.
- 3 Right-click any one of the servers and select **Add new server**.
- 4 In the **Enter new name** field, enter the name of the computer on which the data source server is located.
- 5 Click 

The new server is displayed in the list of servers in the Smart View Panel.

Click  at any time to cancel.

Deleting Servers

- To delete a server from the Smart View Panel:
 - 1 In the **Shared Connections** tree list, select the server to delete.
 - 2 Right-click and select **Remove server**.

Disabling Smart View

Smart View is enabled by default after installation. You can disable Smart View for all Microsoft Office applications on your computer or for Outlook alone.

- To disable Smart View for all Microsoft Office applications (including Outlook):
 - 1 From the Smart View ribbon, select **Help**.
 - 2 Select **About**.
 - 3 Clear **Enable Add-in** to disable Smart View the next time you open an Office application.
- To disable Smart View for Outlook only:
 - 1 From the Smart View ribbon in Excel, select **Options**, then **Advanced** in the left panel.
 - 2 Under **Others**, select **Disable Smart View add-in in Outlook**.

3

Smart View General Operations

In This Chapter

Smart View Operations	21
Using Undo and Redo	21
Copying and Pasting	22
Sheet Information	24
Searching in Smart View	24
Enabling Advanced Logging	24
Shared Workbooks	25
Printing POV Members in Header and Footer	25

Smart View Operations

Smart View provides a set of operations common to all data source types. These include basic operations, functions, and the ability to set preferences.

Using Undo and Redo

Smart View Undo and Redo behave differently depending on the data source to which you are connected.

- In ad hoc analysis with Essbase, Financial Management, Oracle BI EE, or Hyperion Enterprise data sources, Undo undoes Zoom In, Zoom Out, Keep Only, Remove Only, or Refresh and restores the previous database view to the grid. Performing an Undo after modifying member data returns the sheet to its state before the last refresh, not to its state before the data modification.
- In data forms with Financial Management, Hyperion Enterprise, or Planning data sources, Undo undoes the last user action in a cell.

Note: You cannot undo operations that are performed on the server rather than in Smart View, such as calculation status.

➤ To specify the number of permitted undo and redo actions:

- 1 From the Smart View ribbon, select **Options**, then **Advanced** in the left panel.

- 2 In **Number of Undo Actions**, specify the number of permissible **Undo** operations - 0 through 100. This is also the number of **Redo** operations permitted.
- 3 Click **OK**. The setting takes effect after you refresh or perform a drill operation.

Copying and Pasting

Importing Metadata into Copied Worksheets

Data source types: Essbase, Planning, Financial Management, Reporting and Analysis, Hyperion Enterprise, Oracle BI EE

When you copy an Excel worksheet, only the data and not the metadata is copied. (Metadata consists of such things as the POV, alias table, and connection information). However, after the data is copied, you can import this metadata from the original worksheet to the new one.

You can import metadata in the following:

- Ad hoc mode, including Smart Slices
- Data forms
- Functions
 - Query-bound functions in sheets created by Smart View copy and paste
 - Non-query-bound functions created by the Function Builder
- Worksheets that contain reports imported from Reporting and Analysis providers

You cannot import metadata in worksheets that contain Report Designer objects, but such workbooks can be replicated by cascading as described in [“Cascading Reports and Ad Hoc Grids” on page 30](#).

Note: This procedure should be performed only by advanced users.

► To import metadata to a copied worksheet (this operation cannot be undone):

- 1 **Back up your work.**
- 2 From the Smart View menu, select **Options**, then **Advanced**, and ensure that **Improved Metadata Storage** is selected.
- 3 Use Excel to copy a worksheet. This operation copies the visible contents of the source worksheet but not the metadata (connection information, POV selections, alias tables and the like) to the destination worksheet.
- 4 With the destination worksheet active, from the Smart View menu, select **More**, then **Import Metadata** to display a list of all open workbooks and their corresponding open worksheets.
- 5 From the list, select the worksheet that contains the metadata that you want to import to the destination worksheet.

- 6 Click **OK**. You will be asked to confirm your selection.
- 7 Refresh.

Copying Data Between Excel, Word, and PowerPoint

In Smart View, you can copy data from Excel and paste it into Word or PowerPoint. The data you copy and paste is dynamic between Office applications. You can copy and paste data from:

- Excel to Word and PowerPoint
- Word to Word and PowerPoint
- PowerPoint to Word and PowerPoint

The data points retain their original Excel-based query information, enabling you to perform data analysis. Word and PowerPoint can contain data points from multiple data sources, such as Essbase, Financial Management, Oracle BI EE, and Hyperion Enterprise within one document.

Notes:

- Dynamic data points are maintained only in Word and PowerPoint. If you copy and paste data points within Excel, the data points are not linked to the Excel grid.
- When copying and pasting from Word to PowerPoint, or vice versa, data is displayed in a straight line. The tabular format is preserved only when copying data from Excel into Word or PowerPoint.
- Excel formatting is preserved when data is pasted into Word and PowerPoint. Apply the formatting in Excel before copying and pasting data.

Note: If the name of the connection to the data source contains a semicolon (;), you may not be able to paste function data points.

➤ To copy and paste data *from* Excel, Word, or PowerPoint *to* Word or PowerPoint:

- 1 Select a data cell or range (may or may not include members).
- 2 From the Smart View ribbon, select **Copy**.
- 3 Open a Word or PowerPoint document.
- 4 When asked if you want to create a connection, click **Yes**.
- 5 From the Smart View ribbon, select **Paste**.
- 6 Refresh.

Note: If you paste data into a Word document and save it in a different format such as .htm or .mht, you cannot refresh the data in these other formats.

- 7 **Optional:** To change the POV in Word or PowerPoint after you paste the data, click **Manage POV** and follow the procedure in [“Selecting Members for the Default POV” on page 44](#).

- To retrieve the Excel spreadsheets from which data points were copied:
- 1 In a Word or PowerPoint document into which Excel data points were pasted, select the data cells.
 - 2 From the data source ribbon, select **Visualize** and then **Visualize in Excel**.
 - 3 If asked to log on the data source, enter the user name and password.
- Excel displays the spreadsheet associated with the data cells. You can perform ad hoc analysis on the data.

Sheet Information

- To view connection and other details for the current worksheet:
- 1 From the **Smart View** ribbon, click **Sheet Info**.
 - 2 **Optional:** select the following options as needed.
 - **Delete Smart View Info** to delete Smart View connection information to enhance performance.
 - **Copy** to copy the selected item in the list to the clipboard.
 - 3 Click **OK**.

Searching in Smart View

Wild card search patterns for Essbase and Oracle BI EE are different.

- For Essbase, use */? as wild cards.
- For Oracle BI EE, use %/_ as wild cards.

Enabling Advanced Logging

Smart View collects and records events, errors, and other information in a log file, typically `SmartViewLogs.log`. When you experience performance or other issues, you can enable the logging in this file of additional information about profiling and requests and responses between Smart View and the server. This additional information can help Customer Support or IT to troubleshoot your issues.

- To enable additional troubleshooting information:
- 1 Close all Microsoft Office applications.
 - 2 Select **Start** then **Run**.
 - 3 Enter `regedit` and click **OK** to open the registry.
 - 4 Navigate to `HKEY_CURRENT_USER\Software\Hyperion Solutions\HyperionSmartView\Options`.

- 5 Right click **Options**, select **New**, and then **String Value**.
- 6 Name the new string value `Profile`.
- 7 Double-click **Profile** to open **Edit String**.
- 8 From **Edit String**, under **Value Data**, enter `1`. This setting provides the most detailed information.
- 9 Close the registry.
- 10 Restart your computer.
- 11 From the Smart View ribbon, select **Options**.
- 12 Go to the Advanced page and ensure that **Route Messages to File** is selected.

The log file, typically `SmartViewLogs.log`, will now begin recording profiling and request/response information between Smart View and the server in addition to the other information it records.

- To send the log file to Customer Support or IT for troubleshooting:
 - 1 Navigate to the location displayed in the field next to **Route Messages to File**.
 - 2 Locate `SmartViewLogs.log` and add it to a zip file.
 - 3 Distribute as appropriate.

Note: To improve performance, when you no longer need to log profiling and request/response information, delete the Profile entry that you created in the registry.

Shared Workbooks

Smart View does not support Excel shared workbooks.

Printing POV Members in Header and Footer

- To print active POV members in the header or footer of an Excel document:
 - 1 In Excel, insert a header or footer section.
 - 2 In the header or footer, enter a statement that includes `POV: {}`.

When you print the Excel document, the POV members are printed in the header or footer as specified.

4

Ad Hoc Analysis

In This Chapter

About Ad Hoc Analysis	27
Starting Ad Hoc Analysis	27
Zooming In and Out.....	28
Formatting Ad Hoc Grids	28
Removing Members From the Grid	29
Other Ad Hoc Options	29
Pivoting	30
Cascading Reports and Ad Hoc Grids.....	30
Substitution Variables	31
Inserting Rows and Columns.....	31

About Ad Hoc Analysis

In ad hoc analysis, you use Smart View functionality with Excel spreadsheets to retrieve and analyze data by selecting members, using functions, and performing a variety of operations, including formatting, to design your reports.

You can perform ad hoc analysis in Essbase, Planning, Hyperion Enterprise, Oracle BI EE, and Financial Management.

Starting Ad Hoc Analysis

For Essbase, all ad hoc functionality is available from the Essbase ribbon, which is displayed when you connect to Essbase. Planning, Hyperion Enterprise, Oracle BI EE, and Financial Management data providers display **Ad Hoc** ribbons when you enter ad hoc analysis.

► To start ad hoc analysis in data sources other than Essbase:

- 1 **From Recently Used, Shared Connections, or Private Connections on the Smart View Panel, do one of the following:**
 - Open an ad hoc grid or Smart Slice.
 - Open a data form that has been enabled for ad hoc by your administrator. From the data provider ribbon, click **Analyze**. If **Analyze** is disabled, the data form has not been enabled for ad hoc analysis.

The Ad Hoc ribbon for your data source is displayed, replacing the original ribbon.

2 Use the ribbon buttons to perform ad hoc analysis on the current worksheet.

If you are familiar with the dimensions and members of your database, you can use *free-form mode* by typing dimension and member names directly into cell to design and create an ad hoc grid. See [Chapter 16, “Using Free-Form Mode.”](#)

Zooming In and Out

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise, Oracle BIEE

You can zoom in and out from a dimension through its hierarchy of members to the lowest level of members.

► To zoom in or out of a member:

- 1 Select a dimension or member.
- 2 In the data source ribbon, from **Analysis**, select **Zoom In** down arrow, then one of the following options.
 - Next level to display only the next level down in the hierarchy of members.
 - All levels to display all levels in the hierarchy.
 - Bottom level to display only members in the lowest level of the hierarchy.
- 3 To specify how members are displayed when you zoom, see **Member Retention** options in [Table 7, “Member Options”](#).
- 4 To zoom out, select **Zoom Out**.

Formatting Ad Hoc Grids

Using Excel Formatting

You can let either Smart View or Excel control grid formatting.

Smart View formatting consists of such settings as Cell Styles (see [“Cell Styles” on page 110](#)) and Use Thousands Separator (see [“Formatting Options” on page 109](#)).

If you choose to use Excel formatting, then these Smart View settings are overridden by your Excel formatting selections. Your Excel formatting selections, including conditional formatting, are applied and retained on the grid when you refresh or perform ad hoc operations *except* Pivot.

When you use Excel formatting, Smart View does not reformat cells based on your grid operations, and it does not mark cells as dirty when you change data values. Smart View does preserve the formatting on the worksheet between operations.

Using Excel formatting is generally preferable for highly-formatted reports, and you must use Excel formatting for data sources whose application-specific colors are not supported by the Excel color palate.

The following operations cannot be used with Excel formatting:

- **Preserve Format** from the Essbase or Ad Hoc ribbon (see [“Preserving Formats While Zooming”](#) on page 29)
- **Retain Numeric Format** (see [“Formatting Options”](#) on page 109)

➤ To retain Excel formatting on ad hoc grids:

- 1 From the Smart View ribbon, click **Options**.
- 2 From **Options**, select **Formatting** from the left pane.
- 3 Select **Use Excel Formatting**.
- 4 Click **OK**.

Any Excel formatting that you apply is now retained after ad hoc operations, including Zoom In, Zoom Out, Keep Only, Remove Only, Undo, and Redo.

Preserving Formats While Zooming

Data source types: Essbase, Planning, Hyperion Enterprise

If you want the formatting specified for the cell you zoom in on to be applied to the cells that are created by zooming, select **Preserve Format** from the Essbase or Ad Hoc ribbon.

Removing Members From the Grid

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

You can remove members and their associated data from the grid as follows:

- To keep only the currently selected members, select the member cells you want to keep. Then from the data source ribbon, click **Keep Only**. All other members in the dimension are removed.
- To remove all members except the currently selected member cells, select the cells you want to remove. Then from the data source ribbon, click **Remove Only**.

Keep Only and Remove Only operate on all instances of the selected members in the grid.

Other Ad Hoc Options

See [Chapter 13, “Smart View Global Options”](#) for other options for performing ad hoc operations.

Pivoting

You can pivot a dimension between rows and columns if there are two or more dimensions in the row or column that contains the dimension you want to pivot. You can also pivot a member; if you do so, the other members in its group are also pivoted.

► To pivot a dimension or member:

- 1 Select a dimension or member.
- 2 From the data source ribbon, click **Pivot**.

Row dimensions are pivoted to the topmost column dimension.

Column dimensions are pivoted to the leftmost row dimension.

Note: Excel formatting is lost upon pivoting.

Cascading Reports and Ad Hoc Grids

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

You can create separate reports for any or all of the members of one dimension in a report based on an ad hoc grid or Smart Slice query and cascade these reports separately across the worksheets of an Excel workbook. For reports created in the Report Designer, you can also cascade reports across slides in a PowerPoint presentation (worksheets or slides are created as needed to accommodate all reports).

Formulas, comments and other text, Smart Slice function grids, charts, tables, and sliders are included in cascaded reports.

► To cascade an ad hoc grid or Smart Slice report:

- 1 Open an ad hoc grid or Smart Slice report on the worksheet.
- 2 From the Essbase or data source ad hoc ribbon, select **Cascade**, then **Same Workbook** to use the current workbook reports or **New Workbook** to use a new one.

The Member Selection window is displayed.

- 3 Under **Dimension**, select the POV dimension to use as the basis for the report.
- 4 Under **Members**, select all members of the dimension for which you want to create reports. One report will be generated for each member you select.
- 5 Click **OK** to begin cascading.

Depending on your selection in [step 2](#), the resulting reports are created on separate worksheets in the current workbook or in a new one. Each worksheet tab is named for the dimension and member of the report it contains.

Note: To enable worksheet tab naming, do not use more than 31 characters or any the following characters for dimension or member names: () : \ / ? * [] .

Note: Cascading may be very slow for large grids.

Substitution Variables

Data source types: Essbase, Planning

Essbase substitution variables can be used in Smart View spreadsheets. Substitution variables are global placeholders for variable values, and are defined and managed by the administrator. When you enter a substitution variable name in a grid, the current value as defined by the administrator is displayed after the grid is refreshed.

For example, suppose the substitution variable “¤tmonth” represents the current month, and the current month is October. Any cell in the grid that contains “¤tmonth” will display “October” after refresh. When the administrator updates “¤tmonth” to “November”, then November will be displayed after the grid is refreshed.

Note: Substitution variable names must begin with an ampersand (&).

Inserting Rows and Columns

In ad hoc grids, calculating and non-calculating columns and rows may be inserted within or outside the grid. Inserted rows and columns may contain formulas, text, or Excel comments, and are retained when you refresh or zoom in.

Always refresh the grid before inserting rows or columns.

In This Chapter

Working with Data Forms in Excel	33
Opening Data Forms in Excel	34
Planning Data Forms	34
Financial Management Data Forms	35
Customizing Data Forms	36

Working with Data Forms in Excel

Data forms are grid displays that enable you to enter data into the database from Excel and to view and analyze data or related text. Certain dimension member values are fixed, giving you a specific view into the data.

Using Smart View, you can work with Planning, Financial Management, and Hyperion Enterprise data forms in Excel.

Note: Excel worksheets are always protected to prevent entering data for read-only cells. Therefore, some Excel functions, such as AutoSum and F9, are disabled.

When you work with data forms in Smart View:

- You can modify data values but not the form structure in data forms.
- Values submitted to the database from Excel must be non-formatted data.
- If a data form is currently loaded in Excel and the administrator changes the data form definition on the server side, Oracle recommends that you close the data form and reload it. This action ensures that the newest data form definitions are displayed.
- Multiple levels in an outline are displayed differently in Smart View than pages on the Planning Web application. Smart View displays up to four levels, while the Web application displays up to two levels.
- If a Planning administrator hides a dimension in the row axis of a Planning data form, this dimension does not display in the row header of the data form in Smart View.

Opening Data Forms in Excel

► To open a data form:

- 1 **Connect to a data source.**
- 2 **In the Smart View Panel, do one of the following:**
 - To open one data form, expand the tree list and select the data form you want to open. Then click **Open form** on the Action Panel.
 - To open multiple data forms, expand the tree list and select a forms folder. Then click **Open forms** on the Action Panel. In **Select Form**, follow the instructions to open one or more forms.
- 3 **(Planning only) To view any instructions that may be associated with the data form, from the Planning ribbon, select **More** and then **Instructions**.**

Planning Data Forms

If you have been assigned the ad hoc user role by the administrator, you can perform ad hoc analysis on Planning data forms that have been enabled for ad hoc by the administrator.

If you have been assigned the ad hoc grid creator role, you can save Planning ad hoc grids as data forms.

Note: Attributes in Planning data forms are not displayed in Smart View.

► To save a Planning ad hoc grid as a data form:

- 1 **With the Planning ad hoc grid active, from the Planning Ad Hoc ribbon, click **Save Ad Hoc Grid**.**
- 2 **In **Save Grid As**, enter a name, path to the location where you want to save the grid, and description for the grid.**
- 3 **Click **OK**.**

► To perform ad hoc analysis on Planning data forms:

- 1 **Open the data form.**
- 2 **Do one of the following:**
 - From the Smart View ribbon, click **Analyze**. This button is enabled only if the current data form has been enabled for ad hoc analysis.
 - Select the data form in the Smart View Panel and click **Ad hoc analysis** in the Action Panel.
- 3 See [Chapter 4, “Ad Hoc Analysis”](#) for information about performing ad hoc analysis.

Financial Management Data Forms

Selecting Members

In Financial Management, if you use the @CUR functionality in a data form, when the form is imported into Smart View, the @CUR member is taken from the background POV for the selected application.

The Active Member option is available only if the application has been set up for Organization by Period. For information on Organization by Period, see the Financial Management documentation.

Adding Members

If the data form is enabled by the administrator, you can insert and save additional rows of members and data. Totals are updated to reflect the new data.

For example, suppose a data form has been defined for an account with transactions for IC1, IC2, and IC4. You could select members IC3 and IC5 for insertion into the data form. The form is refreshed with the new data and the new rows are displayed in the appropriate hierarchical order.

► To add members to data forms:

- 1 **Open a data form.**
- 2 **From the HFM ribbon, click Add Member.**
A cell style (see “Cell Styles” on page 110) can be designated for Add Member.
- 3 **From the member selector, select the members for which to enter data.**
- 4 **Click OK.**

The new members are listed in the member list.

Using Financial Management Linked Forms

Administrators can define links in data forms from one form to another to enable drill-through to a more specific data entry view. For example, a form that contains summary account balances can link to a corresponding form with the account details. The link from one form to another applies to an entire row. A data form can contain up to 64 linked forms.

► To use linked forms:

- 1 **In a data form, select a row that contains linked forms. Linked forms are indicated by the following icon:**



- 2 **Right-click and select HFM Linked Forms, then select the form name.**

A new form is displayed in a separate browser window.

- 3 When you finish using the linked form, click **Close**.

Customizing Data Forms

Customizations to data forms are preserved when you save or refresh only in the following circumstances:

- When customizations are made outside the grid
- Customizations to thousands and decimal separators

6

Selecting Members

In This Chapter

Dimensions and Members	37
Selecting Members	37
Selecting Members from the POV	39
Filtering by Attribute	40
Filtering by Subsets	41
Selecting Period-to-Date Members	41
Member Perspective	42
Aliases and Alias Tables	43
Qualified Member Names	43
The POV Manager	44

Dimensions and Members

Dimensions are data categories used to organize business data for retrieval and preservation of values. Dimensions usually contain hierarchies of related members grouped within them. For example, a Year dimension often includes members for each time period, such as quarters and months.

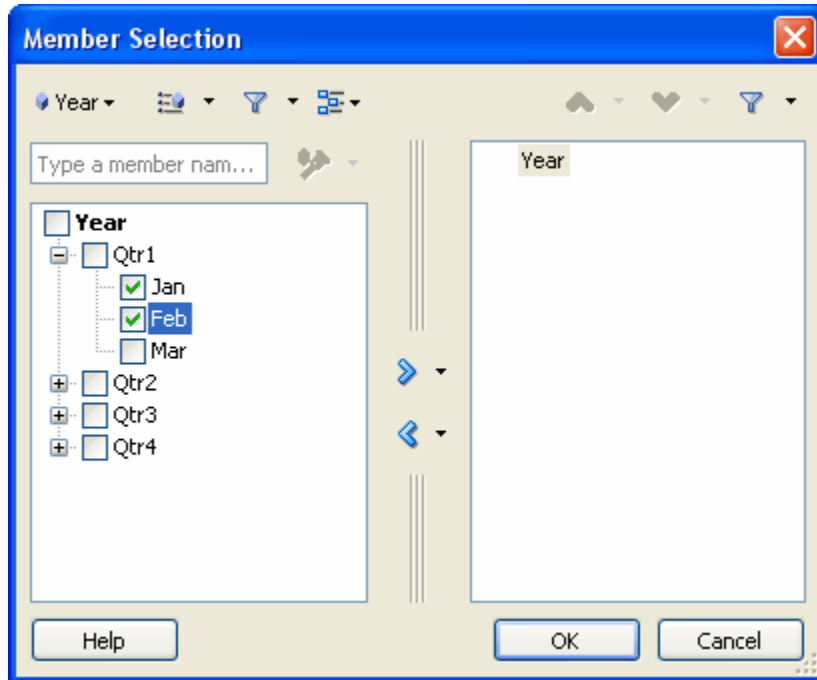
In ad hoc grids, you can select dimensions and members from the Point of View (POV).

Selecting Members




You select members for a variety of purposes within Smart View: ad hoc grids, functions, the POV Manager, and for taking Planning forms offline. The Member Selection dialog boxes in these locations may vary slightly from one another, and not all options are always available. You can select members for one dimension at a time.

Figure 1 shows the Member Selection dialog box with the Year dimension and its members as examples.

Figure 1 The Member Selection Dialog Box



➤ To select members:

- 1 To display the Member Selection dialog box, which contains a tree list of available members for the dimension selected, do one of the following:
 - From an Ad Hoc data source ribbon, select a dimension or member on the grid and click **Member Selection**.
 - From an open dialog box enabled for member selection, click **Member Selection**.
 - To select members for functions and references (see [Chapter 14, “Functions”](#)): From the Smart View Panel, right-click a cube name and select **Member Selection**. The members you select here are placed on a blank worksheet, not an ad hoc grid or data form.
- 2 From **Member Selection**, to change the dimension, click the **Dimension Selector** button (for example, ) and select a dimension.
- 3 **Optional:** To find a specific member in the tree list, enter a member name in the search field and click .
- 4 **Optional:** To find a specific member or group of members in the tree list, click  select one of these filters (filter options may vary by data source type):
 - **Children** to select only the children of the selected member
 - **Descendents** to select all descendents of the selected member
 - **Level** to display **Level**, where you select one level in the hierarchy of members
 - **Generation** to display **Generation**, where you select one generation in the hierarchy of members

- **UDA** to display **UDA**, where you select a user-defined attribute (available only if defined by the administrator)

Note: See also “[Filtering by Attribute](#)” on page 40, “[Filtering by Subsets](#)” on page 41, and “[Selecting Period-to-Date Members](#)” on page 41.

- 5 Under **Members**, select the members that you want to use.

- 6 Click .

The members are transferred from the member tree list to the selection tree list in the pane on the right.

- 7 Click **OK**.

The member selected replaces the dimension in the grid.

- 8 From the ribbon, click **Refresh** to update the data to reflect the selected members.

Selecting Members from the POV

The POV is the default starting point for dimensions in a data source connection. From the POV, you can select members and filters for the dimensions that you want to include in the grid and move members to and from the grid.

Each connection is associated with only one POV. However, the same connection to different worksheets within a workbook may have different POVs.

POVs can be managed as described in “[The POV Manager](#)” on page 44.

Note: Financial Management displays the User Point of View by default. See *the Oracle Hyperion Financial Management User's Guide* for information.

Placing Members and Dimensions from the POV onto the Grid

- To select dimensions and members from the POV:

- 1 Do one of the following:
 - Enter the name of a member over its corresponding dimension on the POV.
 - Click the down arrow next to a dimension on the POV and select one or more members as described in “[Selecting Members](#)” on page 37.
- 2 From the POV, right-click the down arrow next to the member and drag it to the grid.
To move a member or dimension back to the POV for editing, right-click its cell and drag it to the POV.
- 3 Repeat as necessary to place all dimension and members that you want to include on the grid.
- 4 To save these POV selections in the worksheet, you must refresh before you save the worksheet.

Hiding the POV

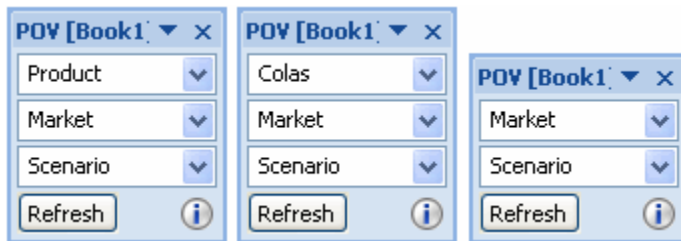
When you finish working with the POV, you can hide it until you need to display it again. To hide the POV, click **POV** on the data source ribbon. The **POV** button toggles to hide and display the POV.

Example

Figure 2 shows, from left to right, a POV in the following conditions:

- **Product**, **Market**, and **Scenario** are the starting dimensions.
- **Colas** is selected as the **Product** member (more than one member at a time can be selected from a dimension).
- **Colas** has been moved to the grid (it can be moved back to edit the dimension).

Figure 2 The POV



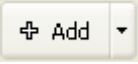


Filtering by Attribute

Data source types: Essbase

You can filter by attributes in dimensions that contain attribute members.

► To filter by attribute:

- 1 Select an attribute dimension on the grid, then open **Member Selection** as described in [“Selecting Members” on page 37](#).
- 2 Click  and select **Attribute**.
- 3 From **Attribute**, click .
- 4 From **Subset**, in **Dimension**, select a dimension; for example, Ounces.
- 5 In **Member**, select an attribute member, for example, Ounces_16.
- 6 Click  to display the attribute.
- 7 Optional: to change the displayed attribute, change the selections in **Dimension** and **Attribute** and click **Set**.
- 8 Click **OK**.

Your selections are displayed in the tree list in **Member Selection**, where you can select from among them for inclusion in the grid.

Filtering by Subsets

Data source types: Essbase

For dimensions that contain attribute members, you can select attributes and set conditions for them to display only those members that meet these conditions.

➤ To filter by condition:

1 Select an on the grid, then open **Member Selection** as described in [“Selecting Members” on page 37](#).

2 Click  and select **Subset**.

3 From **Subset**, in **Dimension**, select an attribute dimension; for example, Ounces.

4 In **Member**, select an attribute member; for example, True.

5 Click .

6 In **Dimension**, select another attribute dimension; for example, Pkg Type.

7 In **Member**, select another attribute member; for example, Bottle.

8 Click .

An AND condition statement is created; for example, [True] AND [Bottle].

9 **Optional:** To change the condition statement, highlight the AND condition statement and select **Operator**, and then **AND** or **OR**.

10 **Optional:** Nest conditions by selecting more attributes, then **Add**, and then **Root**.

11 Click **OK**.

Your selections are displayed in the tree list in **Member Selection**, where you can select from among them for inclusion in the grid.

Selecting Period-to-Date Members

Data source types: Essbase

In time dimensions, you can set up period-to-date members, called Dynamic Time Series members. For example, to see year-to-date data at the end of August, you can set up a Dynamic Time Series member that includes data for January through August.

► To select a Dynamic Time Series member:

1 Select a time dimension on the grid, then open **Member Selection** as described in “[Selecting Members](#)” on page 37.

2 Click  and select **Dynamic Time Series** to display available Time Series Members in the member tree list.

3 Select a time series member from the member tree list and click .

4 From **Select DTS Member**, select the latest period on which to base the to-date calculation; for example, **Aug**.

5 Click **OK**.

6 **Optional:** Repeat [step 3](#) through [step 5](#) as necessary to add other Dynamic Time Series members.

7 click **OK**.

The Time Series Member is displayed on the grid as, in this example, **Y-T-D(Aug)**. After you refresh, the year-to-date data through August is displayed.

Member Perspective

You can specify member perspective for varying attribute when you are selecting members by using the Varying Attribute filter.

Note: Member perspective may not be enabled in your Smart View system. Your options for member perspective are enabled and configured by the Administrator.

► To specify member perspective:

1 From **Member Selection**, under **Filter**, select **Varying Attribute**.

2 In **Filter Arguments**, click .

3 Specify an attribute to set the perspective and click **OK**.

4 In **Varying Attribute Args** under **Varying Attribute**, click the ellipsis button.

5 In **Subset**, in **Dimension**, enter an attribute dimension.

6 In **Member**, enter an attribute member and click **Set**.

7 Click **OK**.

8 In **Varying Attribute Args**, under **Perspective**, click the ellipsis button.

9 From **Perspective**, select one of the following:

- **Snapshot.** One set of independent dimension members can be selected to identify the members of base dimension associated with the varying attribute. Here the start and end tuple are same.

- **Range.** A finite range of independent dimension members can be selected. A range can be specified only for continuous independent dimensions (“Year” would be an example). For discrete independent dimensions you can make only a single selection.

10 Click **OK**.

Aliases and Alias Tables

Data source types: Essbase, Planning, Financial Management

Note: In Financial Management, aliases are called “languages.”

You can select an alias table for the current worksheet or for a connection.

Selecting an Alias Table for the Current Worksheet

The alias table selected here applies only to the current worksheet and not to future connections. Changing the alias table automatically refreshes the grid with the new alias table.

➤ To select an alias table for the current worksheet:

- 1 From a worksheet, connect to a data source.
- 2 From the Essbase or ad hoc ribbon, select **Change Alias** to display a list of available alias tables.
- 3 Select an alias table for the worksheet.

Selecting an Alias Table for the Connection

An alias table selected for a connection is permanent until changed and will be used each time you use this connection. Setting a connection alias table does not refresh the grid even if you click Refresh.

➤ To select an alias table for the connection:

- 1 From a worksheet, connect to a data source.
- 2 In the Smart View Panel tree list, right-click a connection name and select **Set Alias Table**.
- 3 Select an alias table for the connection. The alias table will be set the next time you connect to that connection.

Qualified Member Names

A qualified name is a member name plus the names of its ancestors to the level in the dimension that uniquely defines the member. The ability to see qualified names is helpful on grids that contains aliases or duplicate member names.

- To view the qualified name of a member:
- 1 Select a member in the grid.
 - 2 From the data source ad hoc ribbon, select **View Qualified Name**.
 - 3 Click **OK**.

The POV Manager

Using the POV Manager, you can perform the following operations:

- Select members for the default POV and edit the default POV
- Save a POV to a workbook
- Copy a POV and paste it to a different workbook
- Delete a POV

Selecting Members for the Default POV

In the POV Manager, you can select members to use as a default POV for the ad hoc grids of a given connection or for the background POV for dimensions when you use functions. You can also edit POVs in the POV Manager.

Oracle recommends a maximum of 1,000 members for the ad hoc POV.

Select members for or edit the POV before starting work on an ad hoc grid.

- To select members for the default POV:
- 1 From the Smart View ribbon, select **Function**, and then **Manage POV**.
 - 2 Expand the POVs list.
 - 3 From the **Active** POV list, select the active connection for which you are changing the POV.
 - 4 Click **Member Selector** and select the members that you want to use for the POV. See [“Selecting Members” on page 37](#).

From the POV Manager, you can select only one member per dimension. If you use aliases, the POV Manager loses the selected members.

- 5 Click **Close**.
- 6 To refresh the worksheet, select **Refresh**.
- 7 To save the POV to the workbook, save the workbook.

Note: After work has been started on the ad hoc grid, select or change members as described in [“Selecting Members” on page 37](#).)

Copying and Pasting a POV

You can use the POV Manager to copy and paste a POV from one workbook to another if the data source is exactly the same for both workbooks. The copied POV must be pasted to an unconnected worksheet; otherwise the POV has no effect.

➤ To copy and paste a POV:

- 1 From the Smart View ribbon, select **Function**, then **Manage POV**.
- 2 In the left window of POV Manager, expand **Active** and select the application connection you want to copy.
- 3 From the POV Manager toolbar, click **Member Selector** and select members for the POV.
- 4 Save the workbook.
- 5 From the POV Manager toolbar, click **Copy**.
- 6 In the left window of the POV Manager, expand **Saved** to select the workbook and worksheet (which must be blank and unconnected) you want to paste the POV into.
- 7 Click **Paste**.
- 8 Refresh the worksheet containing the copied POV.

Deleting a POV

➤ To delete a POV that has been saved in a workbook:

- 1 From the Smart View ribbon, select **Function** then **Manage POV**.
- 2 Expand the POV list.
- 3 From the **POV** drop-down list, select the worksheet that contains the POV that you want to delete.
- 4 Select the POV that you want to delete.
- 5 Click **Delete**.
- 6 Click **Close**.
- 7 To refresh the worksheet, select **Refresh**.

7

Working with Data

In This Chapter

Submitting Data	47
Refreshing Data	48
Working with Excel Formulas	48
Adjusting Values in Data Cells	49
Cell Comments	49
Document Attachment	50
Calculating Data	50
Consolidating Data	51
Working with Currencies	52
Drill-Through Reports	53
Data Perspective	54

Submitting Data

Data source types: all

You can update the data (any type) in the data source by submitting changed data from ad hoc grids and data forms. Changes made while you are unconnected can be submitted after you reconnect.

In free-form mode, you must refresh the grid before modifying the data.

If you are submitting from data forms:

- In Planning, Financial Management or Hyperion Enterprise data forms, you can lock any cell or range of cells to protect the data until the data is refreshed or submitted. Locking the cell does not lock the actual data cube in Financial Management, but only the cell in the form. When the data is refreshed or submitted, the cell is no longer locked.
- Some cells may no longer exist in the data form definition. This may happen if data form definition or access privileges have changed, or if rows or columns are suppressed. In these cases, only writable cells that exist in the new data form definition are saved. This applies to both cells and supporting detail changes, and also applies to both online and offline modes.

Caution! If you intend to modify member names, submit any unsaved data beforehand. If you modify member names, you trigger free-form mode and must refresh before submitting data. Because refreshing replaces data in the worksheet with data from the data source, modifications are lost before you submit them.

► To submit data:

- 1 **Connect to the data source.**
- 2 **If you are working in free-form mode, from any ribbon, select **Refresh**.**
- 3 **Modify data as needed.**
- 4 **From any ribbon, select **Submit Data**.**

Refreshing Data

Data source types: all

You can view the most recent data in your data source by refreshing the data. To refresh data, from any ribbon, click **Refresh**. For further refresh options, click the arrow under **Refresh** and select one of the menu items. Hover the mouse over an option to display information for using the option.

Working with Excel Formulas

You can create Excel formulas in ad hoc and data form cells inside or outside the grid if the cells are not read-only or locked. Cells that contain cell text can contain Excel formulas, but cells containing supporting detail (Planning) or line item detail (Financial Management) cannot.

Excel Formulas in Ad Hoc Grids

Formulas are preserved in ad hoc grids when you add comments or use Refresh, Keep Only, Remove Only, Zoom In, or Zoom Out. When you zoom, referential formulas are updated with their new relative positions.

Note: After zooming in or out, the location of the formula is calculated based on the first occurrence of a member on any row or column. If member names are repeated, zooming in or out may cause the formula location to be wrongly calculated. Oracle recommends that you do not use Zoom In and Zoom Out in grids containing repeated members and formulas that refer to them.

Formulas are not preserved if you use Pivot, clear the grid, retrieve data without saving the grid, or change alias tables.

Excel Formulas in Data Forms

Formulas are preserved in data forms when you refresh the data form even without saving the data, later open the saved worksheet, and when you expand or collapse rows and columns.

If you move a referential formula, its cell references are updated to reflect the new location. Note that formulas cannot reference data within the same grid.

In data forms, you are prompted to save the workbook as an Excel file if you do any of the following (but you temporarily lose access):

- Change the current page
- Take a Planning data form offline
- Select a different data form
- Connect to a different data source

Adjusting Values in Data Cells

Data source types: all

You can adjust the value of one or more data cells by a specified number or percentage if the cells contain numerical data. If you adjust the value of a cell that contains an Excel formula, the adjusted value overwrites the formula.

➤ To adjust data values:

- 1 Click the data cell that contains the value to adjust.
- 2 From the data source ribbon, select **Adjust**.
- 3 From **Adjust Data**, select an option then enter the number or percentage by which you want to adjust the value of the cell.
- 4 Click **Adjust Data**.

Cell Comments

Data source types: Planning, Financial Management

Background notations or comments can be added to any cell in the grid and saved to the database.

Note: In Financial Management, you can use functions `HsSetText` and `HsGetText` to submit and retrieve cell text to and from the data source. See [Chapter 14, “Functions.”](#)

➤ To view, add, or edit cell text:

- 1 Select the cell or range of cells to add cell text.
- 2 From the data provider ribbon, select **Cell Comments**.

- 3 In the **Cell Text** tab, view, add, or edit cell text.
- 4 Click **OK**.

Cells that contain cell text can be indicated by a cell style. See [“Cell Styles” on page 110](#).

Document Attachment

Data source types: Planning

Data cells can be linked to URLs, and you can launch a URL for any data cell from Smart View.

Adding, Editing, and Deleting Cell URLs

- To add, edit, or delete a URL linked to a cell:
 - 1 On a data form, select a data cell.
 - 2 From the Planning or Planning Ad Hoc ribbon, click **Cell Comments**.
 - 3 Click the **Document Links** tab.
 - 4 In the text box, enter, modify, or delete the URL as necessary.
 - 5 Click **Save**.

Launching Cell Documents

- To launch a URL for a data cell in a new browser:
 - 1 Select the cell that contains the URL to launch.
 - 2 From the Planning or Planning Ad Hoc ribbon, click **Document Attachment**.

The document that is linked to the cell opens in a new browser.

Calculating Data

After you submit new or changed data, you need to calculate the data in the database to reflect your changes. Your options for calculating data depend on your data source. To calculate data, you must have security access rights to the data.

For information on calculating business rules on Planning forms, see [“Executing the Calculate Data Form and Calculate Currencies Business Rules” on page 78](#).

Calculating Data in Financial Management and Hyperion Enterprise

Data source types: Financial Management, Hyperion Enterprise

- To calculate data:
 - 1 Select a cell or range of cells for which you want to calculate data.
 - 2 From the data source or data source ad hoc ribbon, select **Calculate** then select one of these options:
 - To calculate the selected cells, select **Calculate**.
 - Force calculation to run for all selected cells regardless of cell status, select **Calculate** then **Force Calculate**.

Calculating Data in Essbase

Data Sources: Essbase

In Essbase, you use a calculation script to calculate the database. Calculation scripts are created by your administrator for your specific system.

- To select a calculation script:
 - 1 From the Essbase ribbon, select **Calculate**.
The Calculation Scripts dialog box is displayed.
 - 2 Under **Cube**, select a database from the list of databases that belong to this application.
 - 3 Under **Calculation Script**, select a script.
 - 4 Click **Launch**.

A status message tells you whether the calculation was successful or not. If the calculation was not successful, contact your Essbase administrator.

Consolidating Data

Data source types: Financial Management, Hyperion Enterprise

Consolidation is the process of gathering data from dependent entities and aggregating the data to parent entities. To consolidate data, you must have security access rights to the data and you must be assigned the Consolidate security role. To Consolidate all data, you must be assigned the Consolidate All security role.

- To consolidate data:
 - 1 Select a cell or range of cells for which you want to run consolidation.
 - 2 From the data source ad hoc ribbon, select **Consolidate**, then select one of the following options:
 - **Consolidate** to consolidate data for the selected entities.
 - **Consolidate All** to consolidate data for all entities, whether or not they contain data
 - **Consolidate All With Data** to consolidate the selected entities only if they contain data.
 - **Calculate Contribution** to calculate contribution values of all dependent entities.

- **Force Calculate Contribution** to force calculation to run for all selected contribution values.

Working with Currencies

Translating Currencies in Financial Management and Hyperion Enterprise

Data source types: Financial Management, Hyperion Enterprise

Translation converts values from one currency to another. You can translate data from the entity's input currency to any other currency that has been defined in the application. Currencies are not associated with a parent-child entity pair, so you can translate data on demand, separately from the consolidation process.

- In ad hoc grids, if you have security access rights to the data, you can convert, or translate, values from one currency to another. To translate data:

- 1 **Select a cell or range of cells.**
- 2 **From the data source ad hoc ribbon, select **Calculation**, then select one of the following:**
 - To translate the selected cells, select **Translate**.
 - To force translation to run for all selected cells, select **Force Translate**.

Changing Currency in Planning

If a Planning administrator has enabled the functionality in a data form, you can enter data in a currency other than a cell's base currency. Currencies in the drop-down list can be designated as the local currency.

Note: To override the base currency for an entity, the cell must be displayed in the local currency, and its version must be bottom-up. The application must be a multi-currency application and the data form should support multi-currency.

- To enter cell data in a local currency other than the base currency for the cell:

- 1 **In a data form, select a local currency member for the cell.**
- 2 **Optional: To look up the currency's code, select **View**, then **Currency**.**

Available Currencies shows the application's currencies. Note the Currency Code for the currency you want to work with, and close the window.

- 3 **In the right column, `HSP_InputCurrency`, type the new Currency Code in the data cell.**

Typing the currency code in the data cell overrides the base currency for the entity.

- 4 Click **Submit** to submit the new currency code to the Planning server.
- 5 Enter the currency value in the left column, `HSP_InputValue`, of the data cell.
- 6 Click **Rules on Form** and select the **Calculate Currencies** rule to calculate and save the new currency value.

If the Calculate Currencies calc script is set to run when the data form is saved, and the data form is enabled for multiple currencies, the data value is displayed in the currency you selected.

Drill-Through Reports

Data source types: Essbase, Planning, Financial Management

You can drill through to the detailed data in a database as follows:

- If you are connected to Planning or Financial Management via Smart View, you can use the drill-through capabilities of Smart View to drill through your Planning or Financial Management application to detailed data in Oracle Hyperion Financial Data Quality Management ERP Integration Adapter for Oracle Applications or Oracle Hyperion Financial Data Quality Management, Fusion Edition data sources.
- For applications created in Oracle Essbase Administration Services, you can drill through to Oracle General Ledger.
- For applications created in Oracle Essbase Studio or Oracle Essbase Integration Services, you can drill through to relational databases. For applications created in Oracle Essbase Studio, you can also drill through to administrator-configured URLs.

Predefined by administrators, drill-through reports are available to users from specified individual member cells and data cells. A cell can be associated with multiple drill-through reports. Cells that contain drill-through reports can be indicated on the grid by a cell style (see [“Cell Styles” on page 110](#)).

The data displayed in a drill-through report is dynamic.

Note: You cannot use alias tables for drill-through; you must use member names.

► To access a drill-through report:

- 1 Select a member or data cell associated with a drill-through report (a tooltip displays a list of available drill-through reports when you mouse over a cell.)
- 2 From the data source ribbon, select **Drill-through Reports** to display the list of reports associated with the cell.
- 3 Select a report and click **Launch**.

Data Perspective

Data source types: Essbase

Note: Data perspective may not be enabled in your Smart View system. Your options for data perspective are enabled and configured by the administrator in Oracle Essbase Administration Services.

Data perspective enables you to specify the perspective to use for viewing data of varying attributes, which are dimension attributes that vary with respect to independent continuous and discrete dimensions. For example, suppose a cola product is sold in both cans and bottles in several different geographical markets over the course of a year. If the packaging (cans or bottles) varies depending on the market or changes from one type to the other during the year, the packaging type is a varying attribute. The data associated with the cola would be different depending on the time of year and the market.

► To specify data perspective:

- 1 **Select Hyperion, then Data Perspective.**
- 2 **From Perspective, under Selection, select an option (see [Data Perspective Illustration](#) for examples of options):**
 - **Reality** to display the data with no perspective.
 - **Last** to display the data for the last level 0 member of each continuous independent dimension. For example, if Year is the continuous dimension and December is the last member of Year, then the data for December is displayed.
 - **Start** to display the data for the first level 0 member of each continuous independent dimension. For example, if Year is the continuous dimension and January is the first member of Year, then the data for January is displayed.
 - **Custom** if you want to specify both continuous and discrete members. For this option, select a **Varying Attribute** from the drop-down list. Then, for the dimensions listed under **Independent Dimension**, select members under **Members**. If you select **Set Dimensions Only**, all independent dimensions across all varying attribute are displayed, enabling you to apply a common perspective to all.
- 3 **Click OK, then refresh the grid.**

Data Perspective Illustration

In our example of cola sold in cans and bottles, suppose the Administrator has specified the following attributes for the cola packaging types to reflect how the cola was sold in Texas and California markets during the year:

- Can: California, January—December year
- Can: Texas, July—December
- Bottle: Texas, January—June

Figure 3 illustrates the Reality perspective. The data shown for California and Texas is data for the entire year. Since bottles were not sold in California, no data is returned (indicated here by #Meaningless).

Figure 3 Data Perspective: Reality

	A	B	C	D
1		California	Texas	Market
2	Bottle	#Meaningless	405	405
3	Can	1587	234	1821
4	Pkg Type	1587	639	2226

Figure 4 illustrates the Last perspective and displays data for cans for California and Texas, but none for bottles, because bottles were sold only January through June in Texas.

Figure 4 Data Perspective: Last

	A	B	C	D
1		California	Texas	Market
2	Bottle	#Meaningless	#Meaningless	#Meaningless
3	Can	1587	234	1821
4	Pkg Type	1587	234	1821

Figure 5 illustrates the Start perspective and displays data for January. Bottles but not cans were sold in Texas in January, so only data for bottles is displayed. Cans but not bottles were sold in California in January, so only data for bottles is displayed.

Figure 5 Data Perspective: Start

	A	B	C	D
1		California	Texas	Market
2	Bottle	#Meaningless	639	639
3	Can	1587	#Meaningless	1587
4	Pkg Type	1587	639	2226



Reporting with Smart Slices

In This Chapter

About Smart Slices	57
Creating Reports with Smart Slices.....	57
Smart Slices, Ad Hoc Analysis, and Data Forms.....	60
Creating Smart Slices.....	61

About Smart Slices

A Smart Slice is a reusable perspective of an Essbase, Financial Management, or Oracle BI EE data source. It can be composed of a single member, a combination of single members, filters, or combination of single members and filters in any order. These components serve as boundaries to the data that users can view and work with in the Smart Slice. Any operation that can be done in Smart View can be done within the confines of a Smart Slice.

An organization can have as many different Smart Slices as it needs to accommodate the specific data requirements of its users. For example, Smart Slices can be created for different sales geographical regions, different product lines, different time frames, or a combination of any of these dimensions.

You can view and work with any data within the boundaries of a Smart Slice, but not with data outside its boundaries. For example, in a Smart Slice that limits sales data to the Western region, you could drill down to data for California or Los Angeles, but could not navigate across to New York.

Creating Reports with Smart Slices

Data source types: Essbase, Financial Management, Oracle BI EE

Smart Slices are created as described in [“Creating Smart Slices” on page 61](#) and managed by administrators or others with privileges specified by the data source, Database Administrator in Essbase, for example. They are stored centrally and are available to users from the Smart View Panel. Other users can create Smart Slices for storage on their local computer.

An entire report is associated with an Excel workbook, a Word document, or a PowerPoint presentation. One report is associated with an Excel worksheet, a Word page, or a PowerPoint slide. For PowerPoint presentations, Oracle recommends one report type per slide.

You can create reports from entire Smart Slices or from subsets of data in a Smart Slice. Reports can then be displayed on an Excel spreadsheet, Word document, or PowerPoint slide. You can display as many reports from as many data sources as space will permit on one sheet.

► To create a report from a Smart Slice:

1 From the Smart View Panel, select a Smart Slice.

2 In the Action Panel, do one of the following.

- To work with the Smart Slice as is, click **Insert Smart Slice into report**. The Smart Slice is displayed in the Report Designer in the lower portion of the Smart View Panel.
- To create a subset of the Smart Slice for local storage, click **Modify Smart Slice and insert into report** and use the Smart Slice Designer as described in [Creating Smart Slices](#).

Note: If you use **Modify Smart Slice** to create a Smart Slice, you must select the newly-created Smart Slice from the Smart View Panel tree list before performing ad hoc analysis.

3 Click .

4 From the drop-down menu, select one of these report types to place on the grid:

- **Function Grid** — a dynamic grid format

Function grids can be used with Word, PowerPoint, and Excel. When you refresh a function grid, data cells are refreshed; members are not. To refresh both data and members, you must reinsert the function grid into the sheet. For this reason, function grids are most useful for reports in which members remain reasonably static. For reports whose members may change more often, tables and charts are better report types. Although you can have multiple reports on a worksheet, you can have only one function grid.

You can use Excel formulas, for example SUM, with function grids. To retain such formulas as part of the function grid, you must leave one empty row between the grid and the cell containing the formula and include the empty row in the range of cells selected for the formula definition. This permits retention of the formula when refreshing the data results in a different number of rows in the grid.

To format a function grid, use Excel formatting capabilities.


- **Table**

Tables can be used with PowerPoint and Excel. Table reports display results in a grid format that floats on the document and can be moved and re-sized. When you refresh a table, both members and data are refreshed. Tables are useful for displaying large grids in a smaller space; their scrollbars enable you quickly to access rows and columns.

You can zoom in and out in a table report, but you cannot perform other ad hoc operations or use free form.

- **Chart**

Charts can be used with PowerPoint and Excel. In PowerPoint, contents of charts and tables are visible only in presentation mode. Chart reports display results in a chart format that floats on the document and can be moved and re-sized. When you refresh a chart, both members and data are refreshed.

5 **Optional:** To move or re-size a table or chart, click  and then move or re-size.

6 To insert a report control, click .

7 From the drop-down menu, select one of these report control types:

- POV (a report can contain only one POV)
- Slider (a report can contain multiple sliders). See [“Sliders” on page 59](#).


A report can contain a POV or sliders, but not both.

8 Refresh.

9 **Optional:** to create a separate report for any or all of the members of one dimension in the report and cascade these reports separately across the worksheets of the workbook, see [“Cascading Reports and Ad Hoc Grids” on page 30](#).

Note: In reports that contain a chart and a table, cascading may cause the chart and table to overlap the next time you open the workbook.

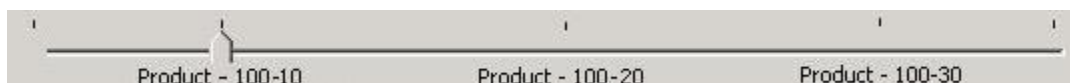
Deleting Reports

To delete reports, click  and then delete. To delete a whole function grid, table, or chart from a document, select the object and press the Delete key. The deletion is reflected in the Report Designer.

Sliders


[Figure 6](#) shows a slider. The slider displays a selected set of dimension members from a query; when you drag the slider marker to a member, its data is displayed in all reports associated with the query on the sheet. Sliders can contain dimensions from more than one query in the Report Designer if the dimensions have the same boundaries.

Figure 6 Slider



Creating a Slider from One Query

► To create a slider:

- 1 Ensure that one or more report type is inserted in the worksheet for the query for which you want to create the slider.
- 2 From the report designer, click **Query View** and select **Query View**.
- 3 In the Report Designer, select the query on which to base the slider.
- 4 Click  and select **Slider** to open **Member Selection**.
- 5 Select a dimension, members, and filters for the slider and click **OK**.


The slider is displayed on the sheet.

- 6 **Optional:** To move or re-size the slider, click  and then move or re-size.

Creating a Slider from Joined Queries

You can create a slider that contains dimensions from multiple queries if, and only if, the dimensions from the different queries have the exact same boundaries.

► To create a slider using a dimension from multiple queries:

- 1 Ensure that one or more report type is inserted in the worksheet for the query for which you want to create the slider.
- 2 Click **Query View** and select **Dimension View**. Notice that the Report Designer tree view is grouped by dimensions rather than by queries. Under each dimension are the queries that contain that dimension. If the dimensions do not contain the same boundaries, multiple sliders will be created to accommodate each of them. For example, if the Market dimension in one query contains a children filter and the Market dimension from another query contains a descendants filter, two Market sliders would be created.
- 3 In the Report Designer, select the dimension on which to base the query.
- 4 Click  and select **Slider** to open **Member Selection**.
- 5 Select dimension members, and filters for the slider and click **OK**.


The slider is displayed on the sheet.

- 6 **Optional:** To move or re-size the slider, click  and then move or re-size.

Smart Slices, Ad Hoc Analysis, and Data Forms

To perform ad hoc analysis on a Smart Slice, (Essbase, Oracle BI EE, Financial Management) in Excel, select the Smart Slice in the Smart View Panel and click **Ad Hoc Analysis** in the Action Panel. Data and POV from the Smart Slice is entered into the worksheet, and you can perform ad hoc analysis.

To use a data form, select the Smart Slice in the Smart View Panel and click **Open Form** in the Action Panel. Only forms enabled by the administrator may be used for ad hoc analysis.

If you want to locate the Smart Slice source of the data in an ad hoc grid, click  and select **Locate Worksheet Connection**. The Smart Slice is highlighted in the Smart View Panel.

Creating Smart Slices

Data sources: Essbase, Financial Management Oracle BI EE

To create a Smart Slice, you must have Administrator, Database Administrator, or other privileges, depending on the data source. Creating Smart Slices involves [Setting Smart Slice Data Boundaries](#) and [Setting Smart Slice Preferences](#).

Setting Smart Slice Data Boundaries

► To create a Smart Slice:

- 1 **Open the Smart View Panel and connect to an Essbase, Oracle BI EE, or Financial Management data source.**
- 2 **Do one of the following:**
 - From the Action Panel, click **Create New Smart Slice**, then select an alias table from the list of alias tables.
 - With an ad hoc grid open, from the data source ribbon, select **Smart Slice**.

The Smart Slice Designer and a **New Smart Slice – Design** worksheet are displayed. You design the Smart Slice from the Smart Slice Designer; results are displayed on the worksheet.

On the Smart Slice Designer are **Rows**, **Columns**, **POV**, and **Attributes** sections for row, column, POV, and attribute dimensions.

- 3 **From the Smart Slice Designer, use any of the following operations to create boundaries for the Smart Slice.**
 - To select members for row or column boundaries, drag members from the **POV** to **Rows** or **Columns** as needed on the Smart Slice Designer. To remove row or column members, drag them to the **POV**. Changes are reflected immediately on the grid.
 - To select members for dimensions under **Rows**, **Columns**, or **Attributes**, section, click the name of the dimension to open the Member Selection dialog box.
 - To select members for the **POV** on the Smart Slice Designer, click the arrow next to the dimension name and select the ellipsis to open the Member Selection dialog box.
- 4 Click **Options** and set preferences as described in [“Setting Smart Slice Preferences” on page 62](#).
- 5 Click **Done**. **Member Selection** is displayed.
- 6 In **Member Selection**, select a dimension member to use as the default POV and click **OK**.

- 7 In the Smart View Panel, in **Enter a new name**, enter a name for the Smart Slice.
- 8 Click **OK**. The Smart Slice is displayed in the tree view of the Smart View Panel under its data source.

Setting Smart Slice Preferences

The preferences that you specify are stored as part of the Smart Slice definition, and they override the global preferences set in the Options dialog box.

- To specify Smart Slice preferences:
 - 1 From the **Smart Slice Designer**, click **Options**.
 - 2 For each option, enter or select the preference from the drop-down menu.

Users can select the options that are enabled here. See [Chapter 13, “Smart View Global Options”](#) for descriptions of the options.

9

Reporting with the Query Designer and MDX Queries

In This Chapter

The Query Designer	63
Creating Queries	63
Editing Queries and Rerunning Reports.....	64
Filtering Data	65
Analyzing Time-Related Data in Query Designer	65
MDX Queries.....	66

The Query Designer

The Query Designer is a Smart View tool from which you can design the layout of a report by selecting dimensions, members, and attributes for rows, columns, and the POV from one interface. You can use the Query Designer to create a query from a blank connected worksheet, which uses the default report as a starting point, or extract a query from a saved report.

Creating Queries

Data source types: Essbase, Oracle BI EE, Financial Management, Hyperion Enterprise

► To create a query report:

- 1 Open a worksheet or an existing report in Excel and connect to a data source.

Note: Workbooks can contain Query Designer worksheets from multiple data sources. However, only one data source can be associated with each worksheet.

- 2 From the data source ribbon, select **Query**, then **Query Designer**.

The Query Designer and a query worksheet named “*Sheetname - Query*” (for example, Sheet1 – Query) are displayed. You design your query on this worksheet.

The following operations are disabled on the query sheet, but are re-enabled after you run the report:

- Formulas
- Asymmetric reports

- Comments
- Blank rows or columns
- Changes to alias tables
- Ad hoc actions such as zoom in and out, keep and remove only, and double-click

The following operations are unavailable in both query sheet and report sheet:

- Filtering of column members
- Changing data sources

3 Use any of the following operations to design your query:

- To select members for the **Rows** and **Columns** dimensions displayed on the Query Designer, click the dimension name to open the Member Selection dialog box.
- To select members for **POV** dimensions displayed on the Query Designer, click the arrow next to the dimension name and select the ellipsis to open the Member Selection dialog box.
- To move a dimension from the **POV** to the grid, drag and drop it from the **POV** section to the **Columns** or **Rows** section in the Query Designer.
- To remove a dimension from the grid, drag and drop the dimension from the Columns or Rows section to the POV section in the Query Designer.
- To add or remove an attribute dimension, select a dimension from the Attributes drop-down menu and drag and drop to the **Rows** or **Columns** section of the Query Designer.
- Enter members directly into the grid.

4 From the Query Designer, click **Apply Query. The resulting report is displayed in a new report sheet called “Sheetname - Report” (for example, Sheet1 - Report). Operations temporarily disabled in step 2 are re-enabled.**

The report sheet replaces the query sheet, but you can retrieve the query sheet by repeating step 2.

5 To save the report, save as an Excel .xls or .xlsx file , which in Essbase or Hyperion Enterprise can be used as a data load data source.

Note: The Query Designer is not designed to work with Smart Slices.

Editing Queries and Rerunning Reports

Rerunning queries regenerates the report; any changes to the original report, such as zooming, comments, and formulas are lost. Formatting is also lost.

You can refresh reports, but this only refreshes the data. It does not rerun the report.



- To edit a query and rerun a report:
 - 1 Open the Query Designer query sheet to edit. If the query sheet is hidden, from the data source ribbon, select **Query** and then **Query Designer**.
 - 2 Edit the query.
 - 3 Select **Query** then **Run the Report**.

The report is updated.

Filtering Data

Data source types: Essbase

Filtering data limits the amount of data returned to a specified top or bottom criterion. Top or bottom ranking enables you to view, for example, the top 10 products in sales for a given region.

- To filter data:
 - 1 In the Query Designer report worksheet, select a dimension.
 - 2 From the Essbase ribbon, select **Query** and then **Data Filter**.
 - 3 From **Data Filter**, under **Count**, select **Top** or **Bottom** and specify a number.
 - 4 Under **Set**, click .
 - 5 From **Member Selection**, select a row member for ranking, and click **OK** to return to **Data Filter**.
 - 6 Under **Value**, click .
 - 7 From **Member Selection**, select a column member to run the ranking against, and click **OK** to return to the **Data Filter** dialog box.
 - 8 Click **OK**.


An MDX query in the form `TopCount({ [Qtr3] }, 10, [Measures].[Profit])`, that represents your data filtering settings is inserted into the grid. The example returns the top 10 most profitable products in quarter 3.
 - 9 Click **Apply Query** to display query results.

Analyzing Time-Related Data in Query Designer

Data source types: Essbase

Using Smart View, you can analyze flash metrics such as sales of cost of goods sold against time-based metrics. This enables you to look for trends, find averages for different time periods, and so forth. To do this, you use linked attributes which enable periodicity of members. Periodicity is a shared pattern among time dimension members that make them meaningful for time-based analysis (January and April share periodicity as first months of quarters, for example). Day by month, day by week, and week by year are examples of linked attributes. You can also set ranges for linked attributes and apply filters.

► To analyze time-related data in Query Designer:

- 1 Create a query.
- 2 From the Query Designer toolbar, select **Date-Time** dimension and drag it to the grid or within the toolbar.
- 3 Click **Date-Time** in the Query Designer toolbar to open **Member Selection**, where you can select members and apply Period, Range, and other filters.
- 4 Under **Attributes** on the Query Designer toolbar, select an attribute or linked attribute in the drop-down menu, then drag it to the grid or within the toolbar. Repeat as necessary for other attributes.
- 5 To select members and apply filters to an attribute, click the attribute name on the Query Designer toolbar to open **Member Selection**.
- 6 Click  on the POV toolbar.

MDX Queries

Data source types: Essbase

MDX users can bypass the Query Designer interface and enter MDX commands in the query sheet or in the Execute MDX dialog box.

► To execute MDX queries:

- 1 In Excel, connect to an Essbase data source.
- 2 From the Essbase ribbon, select **Query**, then **Execute MDX**.
- 3 In **Execute Free Form MDX Query**, enter the MDX query.

For example:

```
SELECT {[Sales], [Cogs]} on columns, Filter ([Product].Levels( 2 ).Members,  
AVG([Year].CHILDREN, 9001.0) > 9000.00) on rows
```

- 4 Click **Execute**.

In This Chapter

Task Lists	67
Working with Tasks from the Smart View Panel	67
Integrating Task Lists with Microsoft Outlook.....	69

Task Lists

Data Source Types: Planning, Financial Management, Financial Close Management

Depending on your data source, you can open and manage tasks from the Smart View panel in Excel or Outlook or integrate task lists from the data source into Outlook and use Outlook functionality to manage your tasks.

- In Planning and Financial Management, you can manage tasks from the Smart View panel in both Excel and Outlook and integrate task lists into Outlook as described in [“Working with Tasks from the Smart View Panel”](#) on page 67.
- In Financial Close Management, you can integrate task lists into Outlook as described in [“Integrating Task Lists with Microsoft Outlook”](#) on page 69.

Working with Tasks from the Smart View Panel

Data Source Types: Planning, Financial Management

Opening a Task List

Data Source Types: Planning, Financial Management

➤ To open a task list from Excel:

- 1 From the Smart View ribbon or menu, click **Open**.
- 2 If prompted, enter your user name and password.
- 3 From the Smart View Panel, do one of the following:
 - From Recently Used on Smart View Home, click the name of a task list.

- From Shared Connections or Private Connections, navigate to the task list that you want to open and click **Open Task List** on the Action Panel.

➤ To open a task list from Outlook:

- 1 Ensure that Outlook displays a Smart View menu. If it does not, do the following:
 - a. Close Outlook.
 - b. In Excel, from the Smart View ribbon, click **Options**, then **Advanced** in the left panel.
 - c. Clear **Disable Smart View add-in in Outlook** and click **OK**.
 - d. Reopen Outlook.
- 2 Ensure that you are connected to a data source as described in [Chapter 2, “Managing Data Source Connections.”](#)
- 3 From the Outlook toolbar, click **Smart View** and select **Open** to display the Smart View Panel.
- 4 From the Smart View Panel, do one of the following:
 - From **Recently Used** on Smart View Home, click the name of a task list.
 - From Shared Connections or Private Connections, navigate to the task list that you want to open and click **Open Task List** on the Action Panel.

Viewing the Task List

Data Source Types: Planning, Financial Management

A task list opened in the Task List pane of the Smart View panel displays the following:

- The individual tasks in the task list. These may contain subordinate tasks. The status of the task – complete, incomplete, or overdue – is indicated by color-coding.
- A drop-down menu from which you can select any of the other task lists associated with the current application
- The Action Panel, which displays the actions that are available for the selected task
- Task Details, which opens when you click the double arrows
- A color-coded status bar for the task list

Executing a Task

Data Source Types: Planning, Financial Management

➤ To execute a task:

- 1 Open the task list that contains the task to execute.
- 2 From the Action Panel, click **Execute Task**.
- 3 Task execution varies with the task and data source.

Completing a Task

Data Source Types: Planning

➤ After completing task requirements, mark the task complete. To complete a task:

- 1 Complete the requirements of the task.
- 2 Open the task list that contains the task to complete.
- 3 Ensure that any dependent tasks are completed.
- 4 Select the task to mark complete.
- 5 From the Action Panel, click **Mark Complete**.

Creating Task List Reports

Data Source Types: Planning, Financial Management

To review the status of your process, you can create a detailed report of one or more task lists in an application in PDF or Excel worksheet format.

➤ To create a task list report:

- 1 From the Smart View Panel, open a task list.
- 2 Right-click a task and select **Create Report**.
- 3 In Report Wizard, use the arrow keys to move all task lists to be included in the report from **Available Task Lists** to **Selected Task Lists**.
- 4 Click **Next**.
- 5 Use the arrow keys to move the users whose status you want to view from **Available Users** to **Selected Users**.
- 6 Click **Next**.
- 7 Select options to create your report.
- 8 Click **Finish**.

The report is created in PDF or Excel, depending on your selection in [step 7](#).

Integrating Task Lists with Microsoft Outlook

Data Source Types: Planning, Financial Management, Oracle Hyperion Financial Close Management

You can import task lists into Microsoft Outlook and use Outlook functionality to manage your tasks. Changes to the status of tasks are sent back to the data source, but you cannot delete tasks in Outlook.

► To import task lists into Microsoft Outlook:

- 1 Ensure that Outlook displays a Smart View menu. If it does not:
 - a. Close Outlook.
 - b. In Excel, from the Smart View ribbon, click **Options**, then **Advanced** in the left panel.
 - c. Clear **Disable Smart View add-in in Outlook** and then click **OK**.
- 2 Open Outlook.
- 3 Click **Smart View** and select **Task List**.
- 4 Select **Shared Connections** or **Private Connections**.
- 5 From the Task List, click **Select application**.
- 6 In **Select Application**, from the drop-down menus, select the server and application associated with the task lists to import and click **OK**.

All task lists associated with the selected application are displayed in Task List.
- 7 Double-click a task list to display its individual tasks in Outlook Task Lists.

From here, you can apply Outlook functionality to your tasks. See the Outlook product documentation for information on working with tasks in Outlook.

In This Chapter

Planning Approvals	71
Monitoring Planning Job Status.....	74
Searching for a Page in Planning.....	75
Copying Versions.....	75
Composite Data Forms	76
Working with Planning Business Rules	76
Spreading Data for Time Periods	79
Member Formula	81
Supporting Detail	82
Setting Planning Preferences	84
Working Offline	84

Planning Approvals

Data Source Types: Planning


Planning Approvals is the submission, review, and approval process of a planning unit. If you are assigned the Approvals role, you can perform the Approvals functions described here. For information about roles, see the administrator.

Changing Planning Unit Status

If you are unfamiliar with the Planning review process, see the *Oracle Hyperion Planning User's Guide*, available on the EPM Documentation Library. To open this library, from the Smart View ribbon, click the arrow next to **Help**, and then **EPM Documentation**.

You can change the status of one or more planning units at a time.


- To view or change the status of a planning unit:
 - 1 Open the appropriate data form.
 - 2 From the Planning ribbon, select **Approvals**.
 - 3 From **Manage Approvals**, select a **Scenario** and **Version**.

- 4 Click  to display the list of planning units to which you have access.
- 5 **Optional:** From the view mode button, select one of the following:
 - **Flat View** to display planning units as a list.
 - **Tree View** to display planning units as a hierarchy (available only to administrators).

From the Tree View, you can select **Start** to start a planning unit and **Exclude** to exclude a planning unit from the process.
 - **My Planning Units** to display only the planning units that you own.
- 6 Select the planning unit or units whose status you want to change. If the list is too long to locate the planning unit easily, you can search or apply filters to the list as described in [“Finding Planning Units” on page 72](#).
- 7 To view details for the selected planning unit, click **Planning Unit Details**.


The **Approval Status** tab displays a history of the process status, owner, actions taken, and the date and times the status changed.

The **Annotations** tab displays any comments that have been entered for the planning unit. See [“Planning Unit Annotations” on page 74](#).
- 8 To change the planning unit status, click **Change Status**.

Note: If you change the status of a parent entity, all of its children change too, unless they were excluded during the First Pass state or were approved.
- 9 From **Approvals - Change Entity's Status**, select an action and next owner for the planning unit.
- 10 **Optional:** Enter comments under **Enter Annotation**.
- 11 Click **Submit**.
- 12 **Optional:** To validate the changed planning unit, click . You can validate only one planning unit at a time.

Finding Planning Units

In **Manage Approval**, you can locate planning units easily by searching or applying a filter to the list of planning units. You can use an auto filter or select members or generations as filter criteria.

- To filter the list of planning units:
- 1 Open **Manage Approval** and select a scenario and version as described in [“Changing Planning Unit Status” on page 71](#).
 - 2 Click  to enable filtering.

The filter bar, which contains filtering tools, is displayed just above the planning unit list.
 - 3 Use one of the following procedures:

Search




To search for a specific planning unit, enter its name in the **Planning Unit** field and click






Auto filter

- a. From the filter bar, click the arrow in the column header for **Approvals Status**, **Sub-Status**, or **Current Owner**
- b. Select the column value to filter by. You can apply auto filters to more than one of these columns.

Filter by member selection

- a. From the filter bar, click , and then select **Member selector**.
- b. Click  and select members for the planning unit list as described in [“Selecting Members” on page 37](#).
- c. Click  to filter the list.

Filter by generation


- a. From the filter bar, click , and then select **Generation**.
- b. Click  and select one or more generations to display in the planning unit list.
- c. Click  to filter the list.
- d. Click OK.

- 4 **Optional:** To undo your filter selections before applying the filter, click .

Planning Unit Promotional Path

A planning unit moves from person to person and department to department based on the owners and reviewers that are assigned to the planning unit and its parents in the planning unit hierarchy.


► To view the promotional path of a planning unit in graphical form:

- 1 From the **Planning** ribbon, select **Approvals**.
- 2 From **Manage Approvals**, select a **Scenario** and **Version**.
- 3 Click **Go** to display the list of planning units to which you have access.
- 4 Select a planning unit.
- 5 Click .

Planning Unit Annotations

You can add or view comments about data in a planning unit that is started. Annotations can vary by combinations of scenario, version, and entity members.

► To add a planning unit annotation:

- 1 From the Planning ribbon, select **Approvals**.
- 2 From **Manage Approvals**, select a **Scenario** and **Version**.
- 3 Click **Go** to display the list of planning units to which you have access.
- 4 Select the planning unit for which you want to add an annotation. To filter the list, see [“Finding Planning Units” on page 72](#).
- 5 **Optional:** to view existing annotations for the selected planning unit, click **Planning Unit Details** and then the **Annotations** tab.
- 6 Click .
- 7 In **Approvals - Add Annotation**, enter a title and annotations (up to 1500 characters). On multibyte systems, Oracle recommends limiting annotations to 750 characters. You can enter URLs and links as well as text.
- 8 Click **Submit**.

Out of Office Assistant

You can set up the Out of Office Assistant to reassign planning units that arrive while you are out of the office.

► To set up the Out of Office Assistant:

- 1 From the Planning ribbon, select **Approvals**.
- 2 From **Manage Approvals**, select a **Out of Office Assistant**.
- 3 From **Out of Office Assistant**, select **I am Currently Out of Office**.
- 4 From **Select Action**, select an action and next owner for planning units that arrive while you are out of the office.
- 5 **Optional:** enter an annotation.
- 6 Click **Submit**.

Monitoring Planning Job Status

► You view the execution status of Planning jobs and delete them if needed on the Job Console. To check the execution status of jobs:

- 1 From the Planning or Planning Ad Hoc ribbon, select **More** and then **Job Console**.

- 2 By default, all jobs are displayed. To filter the list of jobs, from Filter Criteria, use any of the following job criteria:
 - **Type:** From the drop-down menu, select one of these:
 - Business Rule
 - Ruleset (for Calculation Manager)
 - Sequence (for Business Rules)
 - Clear cell detail
 - Copy data
 - Push data
 - **Status:** From the drop-down menu, select Processing, Completed, or Error.
 - **Job Name**
 - **User Name**
 - **Start Date**
 - **End Date**
- 3 Click **Go**. The Job Console displays the jobs matching your selection criteria.
- 4 **Optional:** To view the application name and plan type of a job, select the job and click **Show Details**.
- 5 **Optional:** To delete a job, select the job and click **Delete**.

Searching for a Page in Planning

- If the Planning administrator sets up multiple page dimensions for a form, you select the page with the data you require from the page drop-down menu. To search for a page in Planning:
- 1 Click in the page dimension you want to search to highlight it.
 - 2 From the drop-down menu, select the page name containing the data with which you want to work.

Copying Versions

Data source types: Planning

You can copy data from one bottom-up or target version of a selected scenario to another bottom-up or target version within the same scenario. For example, you can create a Best Case version, and copy some or all the data in that version to a Worst Case version to quickly create a starting point for the new version.

You can copy between bottom-up and target versions.

- When you copy to a bottom-up version, only the selected level 0 members are copied.
- When you copy to a target version, all selected members are copied.

- To protect data in approved planning units, copying a version does not copy to approved planning units.

Note: To successfully copy data, when specifying the copy data criteria, you must select at least one member for the Scenario, Account, Entity, Period, and Version dimensions.

➤ To copy a version:


- 1 From the **Planning** or **Planning Ad Hoc** ribbon, select **Copy Version**.
- 2 From **Scenario**, select the scenario to copy.
- 3 From **Copy From**, select the source version.
- 4 From **Copy To**, select the destination version.
- 5 Click **Go** to display the available entities (planning units) for the selected source version.
- 6 Use the arrow keys to select entities from **Available Entities**. You can copy entities with a **Process Status** of **Not Started** or **First Pass**.
- 7 **Optional:** To copy associated information, select any of these options:
 - Copy Account Annotations. Only annotations for selected entities are copied. If you are copying to a bottom-up version, only level 0 entities (and their annotations) are copied.
 - Copy Cell text and Document links
 - Copy Supporting Details
- 8 Click **Copy Data**.

Note: Wait for the Copy Version completion message before loading another Web page.

Composite Data Forms

Data Source Types: Planning

➤ To open a Planning composite form:

- 1 Connect to a **Planning** data source that contains composite forms.
- 2 From the **Connections** tree list, double-click a composite form (indicated by .

The composite form opens in a new Excel workbook with each subform displayed in a separate worksheet.

Working with Planning Business Rules

In Planning data forms and ad hoc grids you can use business rules to calculate data in Essbase. Some business rules prompt you to enter information, called a *runtime prompt*. After you enter required information and launch a business rule, the data is updated.

Launching Business Rules in Excel

➤ To launch a business rule in Excel to recalculate data in Essbase:

1 Open a Planning ad hoc grid or data form (single or composite).

2 Save any unsaved data.

Unsaved data is lost when you launch a business rule.

3 From the Planning ribbon, select **Calculate**, then **Business Rules**.

4 From **Business Rules**, under **Plan Type**, select the plan type associated with the rule you want to use.

5 Select a rule from the rules listed for that plan type, then click **Launch**.

If the business rule includes runtime prompts, enter the information described in [step 2](#) of “[Entering Runtime Prompts](#)” on page 77.

If the calculation is successful, the values in the Essbase database reflect the results of the calculation.

6 Click **Close**.

7 From the Smart View ribbon, select **Refresh**.

Entering Runtime Prompts

When launched, a business rule can prompt you to enter variable information, called a *runtime prompt*. The business rule designer sets up runtime prompts.








Notes:

- If a business rule has a runtime prompt and Use Members on Forms is selected, the default member on the runtime prompt window matches the current member in the page or POV axes of the open data form.
- The Use Members on Forms option supports runtime prompts having single-member selections only. If the business rule has several runtime prompts for the same dimension, all the prompts are pre-filled with values from the context, even if the runtime prompts were defined as hidden or the Hide Prompt option is selected on the Business Rule Properties tab.
- Members and system variables on the Member Selection page are filtered by your access permissions and limitations set for the runtime prompt (for example, only Descendants of Q1). You cannot select a shared member in a runtime prompt.

➤ To enter a runtime prompt:

1 Launch a business rule having a runtime prompt.

2 Enter or select the input type specified by the runtime prompt, summarized in the following table:

Icon	Expected Input Type
	One member selection
	Multiple member selections
	Numeric value (either entered or selected from cell drop-down menu)
	Text value—use only with enhanced calc scripts, not with graphical scripts
	Dimension from the database—use only with enhanced calc scripts, not with graphical scripts
	For Calculation Manager business rules only: A member or member combination that includes only one member from each dimension the designer has set for this runtime prompt (for example: Sales -> Actual -> Jan refers to the member intersection of Sales, Actual, and January)
	For Calculation Manager business rules only: A range of members, selectable from each dimension the designer has set for this runtime prompt (for example: IDescendants("Marketing"),FY08)

Ensure that the runtime prompts are valid. You cannot launch a business rule until all runtime prompt values are valid.

3 Click **Launch**.

If the calculation is successful, the values in the database reflect the calculation results.

Executing the Calculate Data Form and Calculate Currencies Business Rules

The Calculate Data Form business rule is created for each data form to calculate subtotals. The Calculate Currencies business rule is created for data forms that include multiple currencies in a row, column, or page to enable the conversion of values among available currencies.

The order in which business rules are launched is very important and may affect the data. If you plan to launch both Calculate Data Form and the Calculate Currencies business rules, it is important that you run the conversions first, before subtotaling the data form.

► To launch the Calculate Data Form and Calculate Currencies business rules in Excel:

1 Open a data form.

Any data that is not saved on the spreadsheet is lost when you launch the business rule.

2 From the Planning ribbon, select **Calculate**, then **Rules on Form**.

The business rules associated with the data form are displayed in the Business Rules dialog box.

3 Complete one or both of the following actions:

- To convert currencies, select **Calculate Currencies**.
- To calculate subtotals, select **Calculate Data Forms**.

4 Click **Launch.**

If the calculation is successful, the values in the Essbase database reflect the results of the calculation.

Spreading Data for Time Periods

Data source types: Planning

Note: For detailed information about spreading data, see the *Oracle Hyperion Planning User's Guide*.

In Excel, you can spread, or distribute, values in several ways:

- Spread the value of a summary time period to its base time periods or to the first parent or first child of the parent time period
- Spread values among children and parents proportionally, based on existing distribution
- Spread values based on the weekly distribution of a quarter, which could be 4-4-5, 5-4-4, 4-5-4, or None (as set up by the budget administrator)
- Temporarily lock the values of certain cells while spreading data over time periods

Note: You cannot spread data in a summary time period that includes members with mixed currency types.

Note: Excel formulas in child cells are ignored during spreading.

► To spread data for time periods:

1 Open a data form.

2 Select a cell and enter a new value.

The value is distributed according to the rules described in “Adjusting and Spreading Data” in the *Oracle Hyperion Planning User's Guide*.

3 Click **Save.**

Spreading Data with Cell Locking

When spreading data over time periods, you can temporarily lock the values of one or more cells to preserve their values when other values are recalculated. You can spread data across time periods based on various calculations and visually review the changes before committing them

to the database. For examples of spreading with cell locking, see the *Oracle Hyperion Planning User's Guide*.

► To temporarily lock values:

- 1 Open a data form.
- 2 In the data form, select the cell or group of cells that you want to lock.
- 3 From the Planning ribbon, select **Lock**.

A color change indicates that a cell is locked. You can now spread or manipulate data in the other cells however you want, without affecting the locked cells.

- 4 To unlock a cell, refresh the grid.

Spreading Values Using Grid Spread

If your administrator has enabled Grid Spread, you can specify an amount or percentage to increase or decrease values across multiple dimensions on the grid, based on the existing values in the target cells. When calculating the spread data, read-only and locked cells and cells having supporting detail are ignored. Data integrity is ensured because values can be spread only to cells to which you have access.

► To spread values using Grid Spread:

- 1 Put the cursor in the Subtotal or Total source cell whose value you want to spread to target cells.
- 2 From the Planning or Planning Ad Hoc ribbon, select **Adjust**, and then **Grid Spread**.
- 3 From the drop-down menu, select one of these options:
 - Value to increase or decrease values by a specified amount
 - Percentage to increase or decrease values by a percentage
- 4 Select **Increase By** or **Decrease By** and enter a value or percentage.
- 5 Select a spreading pattern:
 - Proportional Spread to spread the value proportionally, based on the existing values in the target cells (the default)
 - Evenly Split to spread the value evenly among the target cells
 - Fill to replace the value in all target cells

Your administrator can add other spreading patterns.

- 6 Click **Spread**. The specified value or percentage is spread across the target cells, replacing former values with new ones
- 7 To save the new values, click **Save**.

Spreading Values Using Mass Allocation

Using mass allocation, you can spread data to all descendents of a source cell and across all dimensions. Spreading by mass allocation spreads data to cells not displayed on the grid, and does not require that you have access to the target cells.

Data forms must be enabled for mass allocation by the administrator. You must be provisioned with the Mass Allocate role to use mass allocation.

Note: Mass allocation cannot be undone.

➤ To spread values by mass allocation:

- 1 Put the cursor in the **Total** or **Subtotal** cell whose value you want to spread.
- 2 From the **Planning** or **Planning Ad Hoc** ribbon, select **Adjust**, and then **Mass Allocate**.
- 3 Enter a new value in **Spread Value** to replace the current value, or from the drop-down menu, select one of the following options:
 - **Value** to increase or decrease values by a specified amount
 - **Percentage** to increase or decrease values by a percentage
- 4 Select **Increase By** or **Decrease By** and enter a value or percentage.
- 5 Select the **Spread Type** for allocating the specified value or percentage across the target cells:
 - **Proportional Spread** to spread the value proportionally, based on the existing values in the target cells (the default)
 - **Evenly Split** to spread the value evenly among the target cells
 - **Fill** to replace the value in all target cells
 - **Relational Spread** to spread into the selected cells based on values that exist in a different source location. Selecting this option displays the currently selected members for each dimension in the **Selected** column.

Your administrator can add other spreading patterns.

- 6 Click **Spread**. The new values are automatically saved in Essbase.

Member Formula

➤ You can view the underlying formula in cells that contain a formula. Such cells can be indicated on the grid by a cell style specified in the Options window. To view a member formula:

- 1 Select the member whose formula you want to view.
- 2 From the **Planning** or **Planning Ad Hoc** ribbon, select **More**, then **Member Formula**.

Details of the formula are displayed.

Supporting Detail

Please see the *Oracle Hyperion Planning User's Guide* for more information on supporting detail.

Adding Supporting Detail

Use the Supporting Detail window to set how detail items aggregate to cell values in a data form.

► To add supporting detail that calculates values in a data form or ad hoc grid:

1 Open a data form, and select the cells.

You can select one cell or a range of contiguous cells in a row or column. The section cannot include a combination of rows and columns. Select cells that are in the local currency so that you can write to them.

2 From the Planning or Planning Ad Hoc ribbon, select **Supporting Detail.**

The Supporting Detail window reflects your cell selection.

3 Enter a description over the initial “untitled” text.

The text and its associated operator must be unique among children of the same parent. By default, you can enter up to 1,500 characters.

4 Use the buttons to create or change the indented hierarchy to reflect the desired structure and calculations.

For example, click Add Child to add a line item directly below the selected item.

5 Set the mathematical relationships among the line items by selecting an operator for each of them.

Select from these operators: + (add), - (subtract), * (multiply), / (divide), and ~ (ignore).

6 Enter data to set or calculate.

Enter numbers using the same scaling that was set up for the data form.

7 Click **Save.**

Values are dynamically calculated and aggregated before the data is saved. Data on the data form is also saved.

Working with the Supporting Detail Hierarchy

The supporting detail hierarchy should reflect the type of information that supports the cell values and the mathematical operators that create the relationships.

► To create or change the supporting detail hierarchy:

1 In a data form, select the cells with supporting detail.

2 From the Planning or Planning Ad Hoc ribbon, select **Supporting Detail.**

3 Create or change the rows in the hierarchy that provide the detail for the data values by putting the cursor on an item and clicking the options in this table:

Option	Result
Add Child	Adds an item one level below the selected cell. You can add an unlimited number of children, but consider its potential performance impact.
Add Sibling	Adds an item at the same level as the selected cell. You can add an unlimited number of siblings, but consider its potential performance impact.
Delete	Removes the selected item
Delete All	Simultaneously removes all supporting detail
Promote	Moves the selected item to the next-higher level
Demote	Moves the selected item to the next-lower level
Move Up	Moves the selected item before its sibling predecessor
Move Down	Moves the selected item after its sibling successor
Duplicate Row	Adds a row below the selected item, duplicating its structure (text, operator, and values)
Fill	For rows, copies the data from the current cell to the cells to its right
Refresh	Gets the latest stored database values, restoring the previously-saved values, and possibly overwriting changes you just made.

4 Click **Save**.

The save operation stores the detail text, values, and aggregate values.

Viewing or Changing Supporting Detail

Cells that contain supporting detail can be indicated on the grid by a cell style specified in the Options dialog box.

► To view or change calculations or supporting data:

1 Open a data form, and select the cells for which to view or add detail.

You can select one cell or a range of contiguous cells in a row or column. The selection cannot include a combination of rows and columns. Select cells that are in the local currency so that you can write to them.

2 From the Planning or Planning Ad Hoc ribbon, select **Supporting Detail**.

3 View or change the line items or calculations that aggregate the data in the selected cells.

Synchronizing Supporting Detail with Essbase

In Planning applications, when you delete supporting detail for a cell, you affect the associated value in the relational database. You specify how to handle the stored Essbase value. You can set it to #Missing or leave it as it was before the supporting detail was deleted. This feature is useful if you want to use supporting detail as a scratch pad or calculator.

► To synchronize supporting detail with Essbase:

- 1 Open a data form.
- 2 In the data form, click the cell that has the supporting detail you want to remove.
- 3 From the Planning or Planning Ad Hoc ribbon, select **Supporting Detail**.
- 4 In the **Supporting Detail** window, delete the information, then click **OK**.
- 5 Select an option from the displayed message to specify how to handle the aggregate value of the deleted supporting detail stored in Essbase:
 - To delete the value from Essbase, click **Yes**, set the value(s) to **#Missing**.
 - To leave the data value in Essbase as is, click **No**, leave the value(s) as is.

Setting Planning Preferences

► To set user preferences for a Planning application:

- 1 From the tree list in the Smart View Panel, select an application.
- 2 Right-click, then select **User Preferences**.
- 3 From **Preferences**, specify options for the following
 - **Application Settings**: e-mail, alias and workflow options.
 - **Display Settings**: formatting, page, and other options.
 - **User Variables**: variables set up by the Planning to help you navigate large data forms and grids.

Note: You cannot set preferences in offline mode.

Working Offline

If the Planning offline component has been installed and configured for your system, you can take data forms offline and perform essentially the same operations as you do when connected to a Planning server. The changes that you make to offline data forms can be synchronized back to the server.

Taking Data Forms Offline

You can include both online and offline data forms in the same Excel workbook.

Note: Currency conversion is not supported offline.

➤ To take data forms offline:

- 1 In Excel, connect to the Planning data source that contains the data forms you want to take offline.
- 2 From the Planning ribbon, select **More**, then **Take Offline**.

The Take Offline Wizard is displayed; all data forms that you can take offline are listed.

- 3 Expand the **Available Forms/Folders** and select folders and data forms to take offline.
- 4 Click **Next**.
- 5 Double-click a dimension. You can select only one dimension.

If you selected multiple data forms, the dimensions displayed are merged from the dimensions available for the selected data forms.

- 6 Select members and system variables from the **Member Selection** page.

About member relationships:

Table 1 Member Relationships

Relationship	Members Included on the Data Form
Member	The selected member
Descendants	All members below the selected member
Descendants (inc)	The selected member and all its descendants
Ancestors	All members above the selected member
Ancestors (inc)	The selected member and all its ancestors
Siblings	All members from the same level in the hierarchy as the selected member, excluding the selected member
Siblings (inc)	The selected member and all its siblings
Parents	The member in the level above the selected member
Parents (inc)	The selected member and its parent
Children	All members in the level immediately below the selected member
Children (inc)	The selected member and all its children
Level 0 Descendants	All descendants of the selected member that have no children

Note: Different data forms may have children and page-member selections. The Page drop-down list should contain at least one member for each data form from each dimension.

- 7 Click **OK**.
- 8 Repeat steps 5–7 to select members or system variables for each dimension in the list.
- 9 Click **Next**.
- 10 Supply a unique name and a description for the offline connection.

- 11 Click **Finish** to download the selected data forms and members.
- 12 Click **OK**, then click **Done**.

Working with Data Forms Offline

► To work with data forms offline:

- 1 In Excel, from the Smart View ribbon, select **Open**, then **Smart View Panel**.
- 2 Select the offline connection.

Online connections specify *Planning* in the Provider column; offline connections specify *Offline Planning*.

- 3 Right-click and select **Connect**.
- 4 Right-click and select **Open Form**.

Note: If you have a data form open while you are directly connected to the Planning server, then take the data form offline in the same session, you must reopen the data form from the offline connection to work with it offline.

- 5 In the offline data form, add or change data.
- 6 From the Planning menu, select **Submit Data**.

The changed data is saved locally. You can exit Excel without losing the changed data.

Synchronizing Data to the Planning Server

When you synchronize to the server, all data changed within a data forms taken offline since the beginning of the session is saved to the server. You can sync data from all data forms at once or from selected data forms and members.

► To save changed data to the Planning server for all data forms and members taken offline:

- 1 From the Planning ribbon, select **Forms**, then **Sync Back To Server**.
- 2 Log in to the Planning server.
- 3 Click **Sync Back All**.
- 4 Click **OK**.

► To save changed data to the Planning server for selected data forms and members taken offline:

- 1 From the Planning ribbon, select **Forms**, then **Sync Back To Server**.
- 2 Logon to the Planning server.
- 3 Click **Next**.
- 4 Double-click a dimension.

- 5 Select members and system variables from the Member Selection page.
- 6 Click **OK**.
- 7 Repeat steps 4–6 to select members or system variables for each dimension in the list.
- 8 Select **Finish** to save data.
- 9 Click **OK** and **Done**.

Tip: After you reconnect to the server, check that the work you completed offline is correct in the database. If you lose a row or column of data when you refresh a data form, contact the administrator.

Refreshing the Offline Data Form Definition and Data

Refreshing an offline data form definition:

- Updates data on the offline data forms with current values from the online data forms.
- Adds or deletes members or data forms from the ones available during an offline session.

➤ To update offline data and the offline data form definition:

- 1 From the Smart View menu, select **Open**, then **Smart View Panel**.
- 2 Select the connection associated with the current offline session.
- 3 From the Planning menu, select **More**, then **Offline**.

Note: If you are using an offline connection and the Refresh Offline Definition option is not available, contact the Planning administrator. This option is not available if you are using an online connection.

- 4 Enter the user name and password for the online data source.

Because you want to refresh the offline data from the Planning server, you must log on to the server.

- 5 Do one of the following:

- Click **Refresh All** to update all members and data forms taken offline with current online values and definitions. **Refresh All** maintains the current offline data form definition. Skip to [step 10](#).
- Click **Next** to select data forms, members, and system variables to update. This selection may change the data form definition — only members and data forms that you select remain part of the definition. Members and data forms not selected are no longer available offline. Continue to [step 6](#).

- 6 Double-click a dimension.
- 7 Select members and system variables from the Member Selection page.

The list contains members and system variables of the selected dimension.

Use the arrow keys to move members and system variables to or from the **Selected Members** list.

- 8 Click **OK**.
- 9 Repeat steps 6–8 to select members or system variables for each dimension in the list.
- 10 Click **Finish** to start the refresh.
- 11 Click **OK**, then click **Done** when the refresh is complete.

In This Chapter

Importing Reporting and Analysis Documents	89
Editing and Refreshing Documents.....	90
Refreshing Reporting and Analysis Documents	90
Financial Reporting and Web Analysis Import Formats.....	91
Adding Security Certificates for SSL-enabled EPM Workspace Servers.....	92
Importing Interactive Reporting Documents	92
Importing Financial Reporting Documents.....	95
Importing Production Reporting Documents.....	99
Importing Web Analysis Documents.....	102
Using Smart Tags to Import Reporting and Analysis Documents	104

Importing Reporting and Analysis Documents

Using Smart View, you can import Reporting and Analysis documents into Microsoft Excel, Word, or PowerPoint.

- For Oracle Hyperion Financial Reporting, Fusion Edition and Oracle's Hyperion® Web Analysis, you can import reports.
- For Oracle's Hyperion® Interactive Reporting, you can import charts, dashboards, and reports. Using the latest run of BQY jobs, Interactive Reporting supports refresh capabilities.
- For Oracle's Hyperion® SQR® Production Reporting, you can import jobs and job outputs.

Important tasks:

- [“Importing Interactive Reporting Documents” on page 92](#)
- [“Importing Financial Reporting Documents” on page 95](#)
- [“Importing Production Reporting Documents” on page 99](#)
- [“Importing Web Analysis Documents” on page 102](#)
- [“Using Smart Tags to Import Reporting and Analysis Documents ” on page 104](#)

Editing and Refreshing Documents

In Office, you can edit and refresh documents that were previously imported from EPM Workspace. The Smart View ribbon or menu provides the following edit and refresh options:

- Edit—change filters, POVs, or parameters of embedded EPM Workspace documents.
- Refresh—refresh the selected job with the latest EPM Workspace data. Only the job selected in Office is updated; not the entire Office document.
- Refresh All— update all jobs in the Office document.

General edit and refresh behavior

- After the initial import into Office, you can delete any number of pages. Then, when you perform a Refresh, an update is performed on the remaining pages; the deleted pages are not reinstated into the document. For example, if you delete page 2 of 4 pages, only pages 1, 3, and 4 are updated.
- If during Edit or refresh the number of imported pages are less than the original import, the removed pages display as blanks pages in Office.
- If an Edit or Refresh results in additional Workspace pages, those pages are appended to the in Office.
- With refresh, formatted pages such as headings and comments are retained in Word and PowerPoint. In Excel, formatting is not retained on refreshed pages.

Maintaining cell references during document refresh

In Excel, a customized worksheet that references imported document cells or ranges is updated when you execute a Refresh All on the imported documents. For example, imported worksheet A and B are referenced in customized worksheet C. When you execute Refresh All on worksheet A and B, worksheet C is refreshed with updated data from worksheets A and B.

Refreshing Reporting and Analysis Documents

Refreshing updates the report with the latest data from EPM Workspace.

Refresh behavior in Production Reporting and Interactive Reporting:

- In Word, if a report is selected, the entire report is refreshed. If no report is selected, the first report found in the document is refreshed. The first report is not necessarily the report at the beginning of the document.
- In PowerPoint, if no report is selected, the first report found in the slide is updated.

When refreshing job outputs in Production Reporting, new outputs in EPM Workspace are updated.

Refresh behaviors in Financial Reporting and Web Analysis:

- You must select a page in the report to refresh. In Word and PowerPoint, if you do not select any pages when refreshing, a message is displaying stating that no pages are updated.

- If you select Refresh, all pages of the report are refreshed. If you select Refresh All, then all reports in the document are refreshed.

To refresh EPM Workspace documents in Excel, Word, or PowerPoint, perform an action:

- To update the selected Reporting and Analysis document, including all pages associated with that document, select **Refresh** on the Smart View ribbon.
- To update all Reporting and Analysis documents, select **Refresh All** on the Smart View ribbon.

Refreshing Reporting and Analysis documents against Essbase or Financial Management connections

Note: This applies to Financial Reporting and Web Analysis reports imported into query-ready HTML.

Refreshing against Essbase or Financial Management connections updates the report with the latest data from Analytic Services and enables you to perform ad hoc analysis on the Reporting and Analysis document, such as retrieving, zooming, or pivoting data.

Refresh for a report imported in query-ready HTML applies to the current page and not all pages.

Important tasks:

- [“Editing Interactive Reporting Documents” on page 95](#)
- [“Editing Financial Reporting Documents” on page 98](#)
- [“Editing Production Reporting Jobs” on page 101](#)
- [“Editing Web Analysis Documents” on page 103](#)

Financial Reporting and Web Analysis Import Formats

You can import Financial Reporting and Web Analysis documents as fully-formatted HTML, which you can display in Excel, or in query-ready HTML, which enables you to connect to Financial Management or Essbase data sources and run queries.

When you import Reporting and Analysis documents as query-ready HTML, the selected pages of the current data object is converted to HTML, and Hyperion-specific formatting is removed. Thus, Smart View can re-query the data source independent of the Web application.

When you import Reporting and Analysis documents as fully formatted HTML, the selected pages of the current data object is converted to HTML, and Hyperion formatting definitions and calculated members are retained. Thus, Smart View cannot directly query the data source, but Hyperion content can be leveraged by Microsoft Office applications.

Tip: After importing an image in Word or PowerPoint, use the Microsoft Office Format Picture option to format it; for example, to crop and resize. The Format Picture settings are preserved, even after you refresh the image.

Adding Security Certificates for SSL-enabled EPM Workspace Servers

When accessing EPM Workspace servers running on HTTPS protocols, the server's security certificate must be issued by a trusted company. To avoid unnecessary prompts when importing documents, you must add the security certificate into the Internet Explorer certificate path on the Smart View client's machine.

► To add security certificates:

1 In Internet Explorer, connect to EPM Workspace; for example:

`https://<workspace_webserver>:<port>/workspace/index.jsp.`

2 If prompted with the warning message, "The security certificate was issued by a company you have not chosen to trust," select **View Certificate**.

3 Select **Certification Path**.

4 Ensure that all certificates in the certification path have been trusted.

Importing Interactive Reporting Documents

- ["Importing Interactive Reporting Documents into Excel" on page 93](#)
- ["Importing Interactive Reporting Documents into Word and PowerPoint" on page 94](#)
- ["Editing Interactive Reporting Documents" on page 95](#)

Imported Interactive Reporting documents are section-specific.

Table 2 Interactive Reporting Import Object Types

Section	Excel	Word, PowerPoint
Table	Formatted data	NA
Results	Formatted data	NA
Chart	Formatted data	Image
Pivot	Formatted data	NA
Report	Formatted data	Image
Dashboard	Image	Image
Query	NA	NA
CubeQuery	Query ready (Internet Explorer only, not supported by Firefox) Formatted data	N/A
Data model	NA	NA

The following restrictions apply when Interactive Reporting documents are imported into Excel:

- Hidden sections are displayed during import.
- Importing dashboard sections into Excel resizes A1 cells.
- Importing report sections into Excel places chart images before tables
- Importing into Excel may not preserve colors correctly.
- Results sections that contain the euro currency format do not import into Excel.
- Results sections with + (plus sign) in their name do not import.

Importing Interactive Reporting Documents into Excel

➤ To import Interactive Reporting documents into Excel:

- 1 From the Smart View ribbon, select **Open**.
- 2 In the Smart View Panel, connect to a EPM Workspace data source.
- 3 Navigate to the Interactive Reporting document that you want to import.
- 4 From the action panel, click **Open**.

The Import Workspace Document wizard is displayed.

Note: Some wizard screens do not apply to some documents.

- 5 In **Sections**, select the section for importing.
- 6 In **Actions**, select an option:
 - **Refresh and Preview**, to change filters or values prior to previewing the document
 - **Preview**, to preview the document with default settings

If you are importing a CubeQuery section in query ready format, do not select this option.
- 7 Click **Next**.
- 8 If you selected **Preview** in [step 6](#), skip to [step 11](#). If you select **Refresh and Preview**, continue with the next step.
- 9 If user authentication is required to change filters, such as variable, value, or option in the document's settings, in **Specify Database Credentials**, enter the **username** and **password**, and select **Next**.
The connection name is displayed in parentheses (for example, Sample.oce).
- 10 In **Specify Filter**, select a value and click **Next**.
- 11 To import all pages of the document, leave the **All Pages** field check enabled.
- 12 If your document contains multiple pages, select **Split pages across worksheets** to display each page on a separate Excel worksheet.
- 13 In the **Import section As** drop-down, select the method for importing: **Fully Formatted** or **Query Ready**.

- When you export content as query-ready HTML, the current page of the current CubeQuery section is converted to HTML and Hyperion-specific formatting is removed. This enables to requery the data source independent of the Web application.
- When you export content as Formatted HTML, the current page of the CubeQuery section is converted to HTML with the Hyperion formatting definitions and calculated members. This formatting content prevents Smart View from directly querying the data source, but enables Hyperion content to be leveraged by Office applications.

The query ready option is only available for a CubeQuery section for a Refresh and Preview action.

14 Click **Finish**.

The document is displayed in Excel.

Importing Interactive Reporting Documents into Word and PowerPoint

➤ To import Interactive Reporting documents into Word:

- 1 From the Smart View ribbon, select **Open**.
- 2 In the Smart View Panel, connect to a EPM Workspace data source.
- 3 Navigate to the Interactive Reporting document that you want to import.
- 4 From the action panel, click **Open**.

The Import Workspace Document wizard is displayed.

Note: Some wizard screens do not apply to some documents.

5 In **Select an Action**, select an option:

- **Refresh and Preview**, to change filters or values prior to previewing the document
- **Preview**, to preview the document with default settings

6 Click **Next**.

7 If you selected **Refresh and Preview**:

- a. If user authentication is required to change filters, such as variable, value, or option in the document settings, in **Specify Database Credentials**, enter the username and password, and click **Next**.

The connection name is displayed in parentheses (for example, `Sample.occ`).

- b. In **Specify Filters**, select a value.

8 Click **Apply**, and click **Next**.

9 In **Preview**, to import a page, select a page from the drop-down list located in the upper left of the data object.:

10 Optional: To import all pages of the document, select **All Pages**.

11 Click **Finish**.

The document is imported.

Editing Interactive Reporting Documents

➤ To edit Interactive Reporting documents in Excel, Word, and PowerPoint:

- 1 Open the Interactive Reporting document to edit.
- 2 From the Smart View ribbon, select **Open**, then **Reporting and Analysis Document**, and then **Edit**.
- 3 The Import Workspace Document wizard is displayed.

Note: Some wizard screens do not apply to some documents.

4 If you selected **Refresh and Preview**:

- a. If user authentication is required to change filters, such as variable, value, or option in the document settings, in **Specify Database Credentials**, enter the username and password, and click **Next**.

The connection name is displayed in parentheses (for example, `Sample.occ`).

- b. In **Specify Filters**, select a value.

5 Click **Apply**, and click **Next**.

6 In **Preview**, to import a page, select a page from the drop-down list located in the upper left of the data object.

7 Click **Finish**.

Importing Financial Reporting Documents

- [“Financial Reporting and Web Analysis Import Formats” on page 91.](#)
- [“Importing Financial Reporting Documents into Excel” on page 96](#)
- [“Importing Financial Reporting Documents into Word and PowerPoint” on page 97](#)
- [“Editing Financial Reporting Documents” on page 98](#)

Table 3 Financial Reporting Import Document Types

Document Type	Excel	Word, PowerPoint
Report	Fully formatted, query-ready	Image
Snapshot report	Fully formatted	Image
Book	NA	NA
Snapshot Book	NA	NA
Batch	NA	NA

Document Type	Excel	Word, PowerPoint
Grid Object	NA	NA
Image Object	NA	NA
Chart Object	NA	NA
Text Object	NA	NA
Row and Column template	NA	NA

Importing Financial Reporting Documents into Excel

► To import Financial Reporting documents into Excel:

- 1 From the **Smart View** ribbon, select **Open**.
- 2 In the **Smart View Panel**, connect to a **EPM Workspace** data source.
- 3 Navigate to the **Financial Reporting** document that you want to import.
- 4 From the action panel, click **Open**.

The **Import Workspace Document** wizard is displayed.

Note: Some wizard screens do not apply to some documents.

- 5 In **Select a Document**, expand the repository, select a **Financial Reporting** document, and click **OK**.

The document is previewed in the **Import Workspace Document** window.

Note: Some options may not be available for some documents.

- 6 If the **Preview User Point of View** is displayed, preview the current **POV** or change the members of the **POV**.

Note: To display this screen, select **Preview** in **EPM Workspace** preferences, for **User Point of View**.

- 7 Click **Next**.
- 8 Optional: If you want to change the default value, in **Respond to Prompts**, make a selection for prompts, and click **Next**.

Note: This screen is displayed only if the document contains prompts.

- 9 **Optional:** In **Preview from Grid POV**, change the **POV** by selecting a **POV**.
- 10 Change the page dimension by selecting **Page**.
- 11 To import all pages of the document, select **All Pages** to import all pages of the document.
- 12 To display each page on a separate Excel worksheet, select **Split Pages across worksheets**.

13 In Import Document As, select an option:

- **Fully-Formatted** (displays reports in a fully-formatted HTML)
- **Query-Ready** (enables you to run ad hoc analysis on reports when connected to Financial Management and Essbase data sources)
- **Function Grid** (a dynamic grid format)

14 Click Finish.

The document is imported into Excel. If you used the Fully-Formatted option, you can only view the Reporting and Analysis document. If you used the Query-Ready option, then connect to a Financial Management or Essbase data source, you can perform ad hoc analysis, such as retrieving, zooming, or pivoting data.

Importing Financial Reporting Documents into Word and PowerPoint

➤ To import Financial Reporting documents into Word and PowerPoint:

- 1 From the Smart View ribbon, select **Open**.**
- 2 In the Smart View Panel, connect to a EPM Workspace data source.**
- 3 Navigate to the Financial Reporting document that you want to import.**
- 4 From the action panel, click **Open**.**

The Import Workspace Document wizard is displayed.

Note: Some wizard screens do not apply to some documents.

- 5 In **Select a Document**, expand the repository, select a Oracle Hyperion Financial Reporting, Fusion Edition document, then click **OK**.**

The document is previewed in the Import Workspace Document window.

Note: Some screens do not apply to some documents.

- 6 If the **Preview User Point of View** screen is displayed, preview the current POV or change the members of the POV by selecting a member.**

Note: To display this screen, select Preview in EPM Workspace preferences, for User Point of View.

- 7 Optional: If you want to change the default value, in **Respond to Prompts**, make a selection for prompts, and click **Next**.**

Note: This screen is displayed only if the document contains prompts.

- 8 In **Preview** from **Grid POV**, change the POV by selecting a POV.**
- 9 Change the page dimension by selecting **Page**.**

- 10 Select **All Pages** to import all pages of the document.
- 11 In **Import Document As**, select **Image** to import the document as an image.
- 12 Click **Finish**.

The document is imported.

Editing Financial Reporting Documents

- To edit Financial Reporting documents in Excel, Word, and PowerPoint:
- 1 From the Smart View ribbon, select **Open**, then **Reporting and Analysis Document**, and then **Edit**.
 - 2 If the **Preview User Point of View** screen is displayed, preview the current POV or change the members of the POV.

Note: To display this screen, select Preview in EPM Workspace preferences, for User Point of View.
 - 3 Optional: If you want to change the default value, in **Respond to Prompts**, make a selection for prompts, and click **Next**.

Note: This screen is displayed only if the document contains prompts.
 - 4 If you want to change the POV, in **Preview from Grid POV** select a POV.
 - 5 Click **Finish**.

Creating Templates in PowerPoint Documents

You can create PowerPoint template documents by importing one or more Financial Reporting reports to the presentation. Every Create Template action creates a new PowerPoint slide with a report name to show where it will be placed when Refresh Template is used. The template PowerPoint presentation can be saved for future use.

- To create a template:
- 1 Open PowerPoint.
 - 2 Connect to a Reporting and Analysis provider.
 - 3 From the Smart View ribbon, select **Open**, then **Reporting and Analysis Document**, and then **Create Template**.
 - 4 In **Import Workspace Document**, select a Financial Reporting document.
 - Optional: To import all pages of the document, select **All Pages**. A separate slide is created for each page.
 - To import the current screen presentation, clear **All Pages**.
 - 5 Optional: To use the Workspace point of view, select **Refresh Using Workspace Point of View**.

- 6 Click **OK**. The document name is imported into the PowerPoint presentation.

Refreshing PowerPoint Templates

When you refresh a template, the template is replaced with a new presentation and each Financial Reporting report placeholder in the template is replaced with the current Financial Reporting report using the current Workspace Point of View.

► To refresh a template:

- 1 Open the PowerPoint presentation containing the template.
- 2 Connect to a Reporting and Analysis provider.
- 3 From the Smart View ribbon, select **Open**, then **Reporting and Analysis Document**, and then **Refresh Template**.
- 4 Edit and save the PowerPoint presentation as needed.

Importing Production Reporting Documents

Production Reporting documents consist of jobs and job outputs, which you can import into Excel, Word, and PowerPoint.

- [“Importing Production Reporting Jobs into Excel” on page 99](#)
- [“Importing Production Reporting Jobs into Word and PowerPoint” on page 100](#)
- [“Importing Production Reporting Job Outputs into Word, and PowerPoint” on page 101](#)
- [“Editing Production Reporting Jobs ” on page 101](#)

Table 4 Production Reporting Import Object Types

Object Type	Excel	Word, PowerPoint
Job	Formatted data	Image
Job output	Formatted data	Image

Some limitations exist for importing:

- Images and charts are not imported into Excel.
- Secure jobs are supported, but jobs imported as generic jobs are not supported.

Importing Production Reporting Jobs into Excel

► To import Production Reporting jobs into Excel:

- 1 From the Smart View ribbon, select **Open**.
- 2 In the Smart View Panel, connect to a EPM Workspace data source.

- 3 Navigate to the Oracle's Hyperion® Interactive Reporting document that you want to import.
- 4 From the action panel, click **Open**.
The Import Workspace Document wizard is displayed.
- 5 In **Select a Document**, expand the repository, select a Production Reporting job, then click **OK**.
The import wizard screen is displayed.

Note: Depending on the document, some screens may not be applicable.

- 6 If the **Specify Parameters** screen is displayed, define the job parameters, and click **Next**.

Note: This screen is displayed only if the job contains parameters.

- 7 In **Preview**, to import a page, select a page from the drop-down list located in the upper left of the data object.
- 8 To import all pages of the job, select **All Pages**.
- 9 Select **Split Pages across worksheets** to display each page on a separate Excel worksheet.
- 10 Click **Finish**.

The document is displayed in Excel.

Importing Production Reporting Jobs into Word and PowerPoint

The procedures for importing Production Reporting jobs into Word and PowerPoint are similar.

- To import Production Reporting jobs into Word and PowerPoint:

- 1 From the Smart View ribbon, select **Open**.
- 2 In the Smart View Panel, connect to a EPM Workspace data source.
- 3 Navigate to the Production Reporting document that you want to import.
- 4 From the action panel, click **Open**.
The Import Workspace Document wizard is displayed.
- 5 In **Select a Document**, expand the repository, select a Reporting and Analysis document, then click **OK**.
The import wizard is displayed.

Note: Some screens may not apply to some documents.

- 6 If the **Specify Parameters** screen is displayed, define the job parameters, and click **Next**.

Note: This screen is displayed only if the job contains parameters.

7 In **Preview**, to import a page, select a page from the drop-down list located in the upper left of the data object.

8 To import all pages of the job, select **All Pages**.

For Word, **Split pages across pages** is disabled. For PowerPoint, **Split pages across slides** is selected and disabled because by default, the pages from jobs or job outputs always split across pages and slides.

9 Click **Finish**.

The job is imported.

Importing Production Reporting Job Outputs into Word, and PowerPoint

➤ To import Production Reporting job outputs into Excel, Word, and PowerPoint:

1 Connect to a EPM Workspace data source.

2 From the Smart View ribbon, select **Open**, then **Reporting and Analysis Document**, and then **Import**.

The Import Workspace Document dialog box is displayed.

3 In **Select a Document**, expand the repository, select a Production Reporting job output, then click **OK**.

The job output is imported.

Editing Production Reporting Jobs

You can edit imported Production Reporting jobs, but not job outputs. You can edit only job parameters.

➤ To edit Production Reporting jobs:

1 Open an imported Oracle's Hyperion® SQR® Production Reporting document.

2 From the Smart View ribbon, select **Open**, then **Reporting and Analysis Document**, and then **Edit**.

The Import Workspace Document dialog box is displayed.

3 If the **Specify Parameters** screen is displayed, define the job parameters, and click **Next**.

Note: This screen is displayed only if the job contains parameters.

4 In **Preview**, view the job.

Note: If you deleted any imported pages, edit updates only the remaining pages of the job.

5 Click **Finish**.

The job is updated.

Importing Web Analysis Documents

Web Analysis includes five data object display types, but Smart View can import only three (spreadsheet, chart, and pinboard). Smart View cannot import free-form grid and SQL spreadsheets. See “Financial Reporting and Web Analysis Import Formats” on page 91.

- “Importing a Web Analysis Document or Document Objects” on page 102
- “Editing Web Analysis Documents” on page 103

Table 5 Web Analysis Import Document Type

Document Type	Excel	Word, PowerPoint
Report	Fully formatted, query-ready	Image

Table 6 Web Analysis Import Data Object Type

Data Object	Excel	Word, PowerPoint
Spreadsheet	Data + formatting	Image
Chart	Data + formatting	Image
Pinboard	Data + formatting	Image

Importing a Web Analysis Document or Document Objects

Using Smart View in Excel, you can import one or all document pages or multiple data objects with one or more pages from a Web Analysis document residing in the Workspace repository. All Web Analysis data objects (spreadsheet, chart, pinboard) are imported as Excel spreadsheets. Freeform Grid and SQL spreadsheets cannot be imported.

► To import Web Analysis data objects:

- 1 From the Smart View ribbon, select **Open**.
- 2 In the Smart View Panel, connect to a EPM Workspace data source.
- 3 Navigate to the Web Analysis document that you want to import.
- 4 From the action panel, click **Open**.

The Import Workspace Document wizard is displayed.

- 5 In **Select a Document**, expand the repository, select a Web Analysis document, and click **OK**.
- 6 If database credentials are not saved with the Web Analysis document, then the **Specify Database Credentials** page is displayed where you are required to enter valid log on credentials to data sources used in the report. If a report has only one data source and you skip entering credentials, the report is not imported. If you have data objects with different data sources in one report and only want to import one of the data objects, you can enter the credentials for the data objects you want to import and skip credential for the data object you do not wish to import. Enter the user name and password or select **Skip** to skip entering credentials to any of the data sources, and click **Next**.

Tip: Select **Save Credentials** to save credentials with a Web Analysis document. It enables you to refresh an imported document later. Currently, you cannot refresh imported documents without saving credentials.

7 In Preview, when selecting objects to import for Microsoft Excel, Word, and Powerpoint:

- Select individual data objects, by clicking the check box located in the top left corner of each report object OR select all data objects by clicking the **All Objects** check box.
- Select **Split Objects across worksheets** to create a new worksheet for each report object OR deselect **Split Objects across worksheets** to place all report objects in the same worksheet.
- Select a page to import from the drop-down list located in the top of each selected to import data objects OR select **All Pages** to import all pages of all selected to import data objects.
- Select **Split Pages across Worksheets** to create a new worksheet for each import page OR deselect **Split Pages across Worksheets** to place all imported pages of each data object in the same worksheet.

8 In Preview, when selecting object to import for Microsoft Word and PowerPoint, select **Import Screen to import a screen print of the entire report.**

9 For Microsoft Excel, In **Import Document As, select an option:**

- **Fully Formatted** (imports reports in fully-formatted HTML). You can connect to Oracle Hyperion Enterprise Performance Management System at any time and refresh the imported document for current data.
- **Query-Ready** (imports reports in query-ready HTML). You can connect to Financial Management or Essbase data source to get data directly and perform ad hoc analysis, such as retrieving, zooming, and pivoting data.

10 Click **Finish. The document is imported. You can then connect to EPM System at any time and refresh the imported document with current data.**

Editing Web Analysis Documents

➤ To edit Web Analysis documents:

1 Select a page (Excel) or an image (Word or PowerPoint).

Note: When editing a report, select a page (Excel) or an image (Word or PowerPoint) from the report, then Edit. If you do not select a page or an image, a message is displayed stating that no pages are updated.

2 From the Smart View ribbon, select **Open, then **Reporting and Analysis Document**, and then **Edit**.**

3 If database credentials are not saved with the Web Analysis document, then the **Specify Database Credentials page is displayed. In **Specify Database Credentials**, enter the user name and password, or select **Skip**, then click **Next**.**

Tip: You can select Save Credentials to save them with the Oracle's Hyperion® Web Analysis document.

- 4 Select a data object (spreadsheet, chart, or pinboard) to import.
- 5 In **Preview**, to import a page, select a page from the drop-down list located in the upper left of the data object.
- 6 Select **All Pages** to import all pages of the document. Leave the box cleared to import only the current page.
- 7 Select **Split Pages across worksheets** to display each page on a separate worksheet (Excel only).
- 8 In **Import Document As**, select:
 - Fully Formatted (Excel only)
 - Query-Ready (Excel only)
 - Image (Word and PowerPoint)
- 9 Click **Finish**.

Using Smart Tags to Import Reporting and Analysis Documents

Smart tags provide an alternative way of importing Reporting and Analysis documents.

- To import Reporting and Analysis documents using smart tags:
- 1 Open a Microsoft Office document.
 - 2 Connect to an EPM Workspace data source.
 - 3 Ensure that smart tags are enabled in Excel.
 - 4 Type `smartview` anywhere in the document, then move the mouse over the word.
The smart tags action icon is displayed.
 - 5 Click the smart tag icon and select **Reporting and Analysis Content** to display Import Workspace Document, from which you can import documents.

In This Chapter

Setting Options	105
Member Options	105
Data Options	106
Advanced Options	108
Formatting Options.....	109
Cell Styles	110
Extensions	110
Provider Preferences	111

Setting Options

You set Smart View options in the Options dialog box, which can be opened by clicking **Options** on the Smart View ribbon.

Member Options

To set options for the display of member cells as described in [Table 7](#), click **Options** on the Smart View ribbon, and then select **Member Options** in the left panel.

Table 7 Member Options

Option	
General	
Zoom In Level	From the drop-down menu, select one of the following to specify a default zoom level for ad hoc analysis: <ul style="list-style-type: none">● Next level to display only the next level down in the hierarchy of members● All levels to display all levels in the hierarchy● Bottom level to display only members in the lowest level of the hierarchy

Option	
Member Name Display	<p>From the drop-down menu, select one of the following to specify how to display member names in cells:</p> <ul style="list-style-type: none"> ● Member Name Only to display fully qualified names ● Member Name and Description to display fully qualified names and descriptions (aliases) ● Description Only to display aliases. In free-form mode, fully qualified names are displayed until after you manually add, remove, or edit any comments and refresh.
Indentation	<p>From the drop-down menu, select one of the following to specify how hierarchy levels are to be indented:</p> <ul style="list-style-type: none"> ● None ● Subitems to indent descendants. Ancestors are left-justified in the column. ● Totals to indent ancestors. Descendants are left-justified in the column.
Ancestor Position	<p>From the drop-down menu, select one of the following to specify ancestor position in hierarchies:</p> <ul style="list-style-type: none"> ● Top to display hierarchies in order from highest to lowest level ● Bottom to display hierarchies in order from lowest to highest level
Member Retention	
Include Selection	Display the selected member and the members retrieved as a result of the operation.
Within Selected Group	Perform ad hoc operations only on the selected group of cells, leaving unselected cells as is. This setting is meaningful only when there are two or more dimensions down the grid as rows or across the grid as columns. For Zoom , Keep Only , and Remove Only .
Remove Unselected Groups	For Zoom In or Zoom Out , remove all dimensions and members except the selected member and the members retrieved as a result of zooming.
Mode	
Use Double click for Operations	<p>Double-clicking retrieves the default grid in a blank worksheet and thereafter zooms in or out on the cell contents. Otherwise, double-clicking retains standard Excel functionality and puts a cell into edit mode.</p> <p>If Oracle Essbase Spreadsheet Add-in and Smart View are installed on the same computer and you have not completed the steps in “Smart View and Spreadsheet Add-in” on page 281, double-clicking prompts you to log into Spreadsheet Add-in.</p>
Enable Enhanced Comment Handling	Enables you to review and correct comments and member names in ad hoc grids that contain comments
Preserve Formula in POV Change	Preserves formulas in cells when you make changes to the POV. Otherwise, any formulas in the grid are lost.

Data Options

To set options for the display of data cells as described in [Table 8](#), click **Options** on the Smart View ribbon, and then select **Data Options** in the left panel.

Table 8 Data Options

Option	
Suppress Rows	To streamline the grid, you can suppress rows that contain types of data that you do not need to view. Note: In suppressed rows, cell references to Excel formulas are not updated.
No Data/Missing	Suppress rows that contain only cells for which no data exists in the database (no data is not the same as zero. Zero is a data value.) If you later clear No Data/Missing , suppressed values are returned only from that point on. You must zoom out and then zoom in on a member to retrieve values that were suppressed while this option was selected.
Zero	Suppress rows that contain only zeroes.
No Access	Suppress rows that contain data that you do not have the security access to view.
Invalid	Suppress rows that contain only invalid values.
Underscore Characters	Suppress rows that contain underscore characters in member names (not available in Smart Slice operations).
Repeated Members	Suppress rows that contain repeated member names, regardless of grid orientation.
Suppress Columns	To streamline the grid, you can suppress columns that contain types of data that you do not need to view. Note: In suppressed columns, cell references to Excel formulas are not updated.
No Data/Missing	Suppress columns that contain cells for which no data exists in the database (no data is not the same as zero. Zero is a data value.) If you later clear No Data/Missing , suppressed values are returned only from that point on. You must zoom out and then zoom in on a member to retrieve values that were suppressed while this option was selected.
Zero	Suppress columns that contain only zeroes.
No Access	Suppress columns that contain data that you do not have the security access to view.
Replacement	
#NoData/Missing Label #NoAccess Label #Invalid/ Meaningless	Data cells may contain missing or invalid data, or data that you do not have permission to view. In such cells, Smart View by default displays #Missing, #Invalid, or #No Access, respectively, but you can change these labels. To do so, in any of these fields, enter one of the following: <ul style="list-style-type: none"> Text of your choice (or leave the default). Text labels have the advantage of being descriptive, but they cause Excel functions to fail. #NumericZero to specify numeric zero (0) replacement labels. With #NumericZero, you can use functions, but you cannot submit zeroes to the database (even if the zeroes are actual zeroes and not replacement labels) unless you select Submit Zero. Calculations that are dependent on a cell with a numeric zero label compute correctly and take the value of the cell as zero.
Submit Zero	Select if you entered #NumericZero above and want to be able to submit zeroes to the database.
Display Invalid Data	Display actual data even if it is invalid, rather than #Invalid/Meaningless or other replacement text. If no data exists, the cell is left blank.
Enable Essbase Format String	If the administrator has created specific formatting for the display of numerical data, view data in this formatting.
Mode	

Option	
Cell Display	<p>As an alternative to displaying actual data, you can display the calculation or process status of the cells:</p> <ul style="list-style-type: none"> ● Data to show actual data ● Calculation Status to show whether data needs to be calculated, translated, or consolidated ● Process Management to show the entities level (Financial Management) or Approvals level for combinations of data called process units (Planning)
Navigate Without Data	Speeds up operations such as Pivot, Zoom, Keep Only, and Remove Only by preventing the calculation of source data while you are navigating. When you are ready to retrieve data, clear Navigate without Data .
Suppress Missing blocks	Suppress blocks of cells for which no data exists in the database.

Advanced Options

To set options for the administrative and other advanced tasks as described in [Table 9](#), click **Options** on the Smart View ribbon, and then select **Advanced** in the left panel.

Table 9 Advanced Options

Option	
General	
Shared Connections URL	<p>Specify a default URL for all connections. Use the following syntax: <code>http://<server>:19000/workspace/SmartViewProviders</code></p> <p>Note: This field must contain an EPM Workspace URL for Smart View online help to be available.</p>
Number of Undo Actions	The number of Undo and Redo actions permitted on an operation (0 through 100). See “Using Undo and Redo ” on page 21 .
Number of Most Recently Used Items	The number, 15 or fewer, of your most recently used connections to be displayed on Smart View Home and the Open menu on the Smart View ribbon.
Delete All MRU Items	Delete all items in your most recently used list, including those that are pinned to the list.
Logging	
Log Message Display	<p>All error, warnings, and informational messages from the connected data source are displayed when they occur, but you can choose which of these message levels to record in a log file. Select a message level to display and record:</p> <ul style="list-style-type: none"> ● Information: All messages, including warnings and errors – recommended to diagnose problems. Adversely impacts performance. ● Warnings: Warnings and error level messages. Adversely impacts performance. ● Errors: Error messages only – recommended for general use; has minimal impact on performance. ● None: Suppress all messages. <p>For information about enhanced logging capabilities, see “Enabling Advanced Logging ” on page 24</p>

Option	
Route message to files	Save log messages in a file. Click the ellipsis button to change the location of the log file.
Clear Log File on Next Launch	Clear the log file starting with the next log message generation, which will be seen after Excel is closed.
Display	
Adjust Column width	Adjust column widths to fit cell contents automatically.
Language	Select a language in which to display Smart View. You must restart the Office application when you change languages. Default is the language specified when Smart View was installed.
Display Smart View Short Cut Menus Only	Display only Smart View menu items on shortcut menus. Otherwise, shortcut menus display both Excel and Smart View items.
Disable Smart View in Outlook	Disable Smart View in Outlook if you do not want to use Smart View task lists in Outlook
Enable Ribbon Context Changing	Display the active data provider ribbon automatically after you use a button on the Smart View ribbon.
Disable options that are not valid for the active connection	Disable options in the Options dialog box that are not valid for the active connection.
Compatibility	
Reduce Excel File Size	Compress the metadata maintained in Excel files that contain Smart View workbooks
Improve Metadata storage	Enable workbooks saved in Smart View 9.3.1.0.x or earlier or saved in Office 2000 to be opened. See “Importing Metadata into Copied Worksheets” on page 22 .
Refresh Selected Functions and their dependents	Execute dependent functions on the same sheet before executing the selected functions.
Performance	
Enhanced Query Performance	Enable faster performance during queries by directing Smart View not to maintain comments and formulas when you perform such operations as zooming. When you select this option, comments and formulas are not deleted from the original worksheet, but they are not carried through operations on the sheet.

Formatting Options

To set options for formatting numbers as described in [Table 10](#), click **Options** on the Smart View ribbon, and then select **Advanced** in the left panel.

Table 10 Number Formatting Options

Option	Purpose
Use Thousands Separator	Use a comma or other thousands separator in numerical data. Do not use # or \$ as the thousands separator in Excel International Options.
Use Excel Formatting	Use Excel rather than Smart View formatting and retain Excel formatting for all ad hoc operations except for pivoting.
Retain Numeric Formatting	When you drill down in dimensions, use the scale selected from Scale and/or number of decimal places from Decimal Places under Retain Numeric Formatting for data.

Cell Styles

On the Cell Styles page, you can specify formatting to indicate certain types of member and data cells.

You can specify a style to indicate the type of member and data cells. Because cells may belong to more than one type — a member cell can be both parent and child, for example — you can also set the order of precedence for how cell styles are applied.

► To specify a style:

- 1 Expand the list of available cell types.
- 2 Select a cell type.
- 3 Select **Properties** and specify a font, background color, or border.
- 4 To re-order precedence of cell styles, use the **Move Up** and **Move Down** buttons or drag and drop the cell styles.
- 5 Click **OK**. The setting takes effect after you refresh or perform a drill operation.
- 6 **Optional:** To revert cell styles or precedence to the default styles of the connected Smart View provider, click **Default Styles**.

Extensions

The Extensions page contains a list of the extensions that are installed to leverage Smart View functionality for other Oracle products. From this page you can do the following:

- Enable and Disable extensions.
- Check for updates to extensions.
- Enable logging for extension installations.

Provider Preferences

Data source types: Essbase, Financial Management, Oracle BI EE (See [“Setting Planning Preferences” on page 84](#) for Planning preferences.)

You must have administrative privileges to set these preferences.

➤ To set preferences for the data source provider:

1 From Smart View Home, select an application.

2 Right-click and enter or select any of the following options:

- **Force Client Upgrade:** to force an upgrade of Smart View when a newer version is available
- **Warn Client Upgrade:** to a message to upgrade to a newer version of Smart View when the newer version is available
- **Session Timeout:** the period of inactivity (in minutes) before a user session times out
- **Number of Rows:** the maximum number of rows permitted in a Smart View grid
- **Number of Columns:** the maximum number of columns permitted in a Smart View grid

3 Click **OK**.

In This Chapter

Using Functions	113
Creating Functions	113
Running Functions	115
Viewing Function Grids as Ad Hoc Grids	116
Function Descriptions	116
Accessing Functions with a Smart Tag	119
Common Function Error Codes	120

Using Functions

If you are familiar with the contents of your database, you can use the Smart View functions described below to perform operations on specific data in Excel cells.

- **HsGetValue**: Retrieves data from a data source.
- **HsSetValue**: Sends values to the data source.
- **HsCurrency**: Retrieves the entity currency for the selected members.
- **HsDescription**: Displays the description for the default member.
- **HsLabel**: Displays the label for the default member.
- **HsGetText**: Retrieves cell text from the data source.
- **HsSetText**: Sends cell text to the data source.
- **HsGetSheetInfo**: Retrieves detailed information about the current worksheet.

Creating Functions

You can create functions manually or by using the Function Builder.

Creating Functions in the Function Builder

In the Function Builder, you select a function and specify the connection and members that you want the function to use. The Function Builder then creates the function using the proper syntax and enters it into the selected cell. You can edit these functions.

The selections available to you in a given Function Builder field are limited by your selections in other fields of the Function Builder. For example, only the functions supported by the data source you select are displayed, and only the dimensions supported by the function you select are displayed.

► To create functions using the Function Builder:

- 1 Connect to all data sources that your functions will access.
- 2 Click the cell in which you want to enter the function.
- 3 From the Smart View ribbon, select **Functions**, then **Function Builder**.
- 4 From **Select Connection**, select a data source.
- 5 From **Select Function**, select a function.
- 6 From **Select Member**, select a dimension.
- 7 In the **Member** column, click **Select Member** to select a member. (If you want to use the POV default members for the dimension, do not select a member.)

You can also double-click in the column and type in a member name or cell reference.

- 8 Double-click in the **Member Type** column. From the drop-down list, select **Member** or **Cell Reference** (cell references cannot be used with HsLabel).

When you select Cell Reference, the value in the referenced cell is used in the function.

- 9 **HsSetValue** only: Select **Data** or **Cell Reference** and enter the value to submit.
- 10 **HsSetText** only: Select **Comment** or **Cell Reference** and enter the text to submit.
- 11 Repeat [step 6](#) through [step 10](#) as needed.
- 12 Click **Add to Function**.
- 13 **Optional:** To add another function, click **Add to Function**.
- 14 **Optional:** If you have edited the function in the **Function** field, to validate the function syntax before inserting it into the worksheet, click **Validate Syntax**.

Note: **Validate Syntax** validates only the syntax you are using for the function. It does not validate the members you have selected.

- 15 Click **OK** to insert the function in the selected cell.

The OK button is enabled only if you have first selected **Add to Function**.

- 16 To execute the function, select **Refresh**.

Creating Functions Manually

► Functions can contain a maximum of 256 characters. To create a function manually:

- 1 In Excel, click the cell in which you want to enter the function.
- 2 Enter = (equal sign).

- 3 Enter the function name, `HsSetValue`, for example.
- 4 Enter parameters for the function, using the information specific to each function in [“Function Descriptions” on page 116](#).

Connection parameters can have these values:

- **Empty:** the default connection
- The user-defined name for the connection

- 5 To refresh the worksheet, from the Smart View menu, select **Refresh**.

Functions are validated only when you refresh them.

Syntax Guidelines

See [“Function Descriptions” on page 116](#) for the syntax of individual functions.

- The POV is composed of *dimension#member* pairs, for example, `Entity#Connecticut`.
- If you specify a connection name, it must precede the POV.
- Parent-child relationships are designated by a period, for example, `Entity#UnitedStates.Maine`.
- The connection and POV can be grouped as one parameter, for example `“My_connection;Entity#UnitedStates”`.

Alternatively, they can be split up into multiple function parameters, for example, `“My_connection”`, `“Entity#UnitedStates”`, `“Account#Sales”`.

- If the connection and POV are in the same parameter, the connection and each *dimension#member* pair are separated by a semi-colon (;), for example, `“My_connection;Entity#UnitedStates;Account#Sales”`.

Running Functions

When a worksheet that contains saved functions is opened on a different computer from the one on which it was created, the functions include the full path of the original computer. Smart View automatically updates these function paths when you open the worksheet if all three of the following conditions are met. Otherwise, you must manually update functions using the Excel Links option.

- The worksheet is unprotected.
- The Excel option **Ask to update automatic links** is cleared.
- When you open a workbook, if prompted to update link automatically, select **Continue** or **Cancel**. Do not select **Edit Links**.

➤ To run functions and retrieve values:

- 1 Open the worksheet that contains the functions you want to run.
- 2 Select one of the following:

- To run functions and update all worksheets in the workbook, from the Smart View ribbon, select **Refresh all Worksheets**.
- To run functions and update only the active worksheet, select **Refresh**.

Viewing Function Grids as Ad Hoc Grids

To view grids that contain functions as ad hoc grids, from the Smart View ribbon, select Functions, then Visualize, and then Visualize in Excel.

Function Descriptions

HsGetValue

Data sources: Financial Management, Hyperion Enterprise, Essbase

HsGetValue retrieves data from the data source for selected members of a dimension.

When HsGetValue retrieves no data, the value specified for the #NoData/Missing Label replacement option is used (see [Table 8, “Data Options,” on page 107.](#))

When users select Refresh or Refresh All, only HsGetValue is called. When users select Submit, HsSetValue is called first, HsGetValue is then called only if HsSetValue returns successfully.

Syntax

```
HsGetValue("Connection", "POV")
```

Example

In this example, HsGetValue returns the value from the HFM01 application for the default POV.

```
HsGetValue("HFM01"; "Scenario#Actual;Year#2004;Period#July;View#YTD;  
Entity#UnitedStates.Connecticut;Value#USD;Account#Sales;ICP#[ICP  
None];Custom1#GolfBalls;Custom2#Customer2;Custom3#[None];Custom4#  
Increases")
```

HsSetValue

Data sources: Financial Management, Hyperion Enterprise, Essbase

HsSetValue sends a data value from a worksheet to a data source selected members of a dimension. To send data to a data source, you must have the appropriate load rule and write access for the data source.

Syntax

```
HsSetValue (dollar amount, "Connection", "POV")
```

Example

In this example, HsSetValue sends the value from cell H4 to the HFM01 application.

```
HsSetValue(H4, "HFM01", "Scenario#Actual;Year#2004;Period#"&B$2&" ;View#<Scenario View>;Entity#UnitedStates.Connecticut;Value#<Entity Currency>;Account#"&$A4&" ;ICP#[ICP None];Custom1#GolfBalls;Custom2#Customer2;Custom3#[None];Custom4#Increases")
```

HsGetSheetInfo

Data sources: all

HsGetSheetInfo retrieves detailed information about the current worksheet, as described in [Table 11](#).

Table 11 HsGetSheetInfo Details

Numerical Equivalent	String Equivalent	Sheet Information
1	Connected	Connection status
2	Sheet Type	Ad hoc or data form
3	Server	The server to which the sheet is connected
4	Application	The application to which the sheet is connected
5	Cube	The cube to which the sheet is connected
6	URL	The URL to which the sheet is connected
7	Provider	The data source type to which the sheet is connected
8	Provider URL	The provider to which the sheet is connected; applicable for Oracle Hyperion Provider Services connections
9	Friendly Name	The data source connection name
10	Alias Table	The current alias table
11	User	The user name
12	Description	The connection description

Syntax

```
HsGetSheetInfo("<string equivalent>")
```

```
HsGetSheetInfo("<numerical equivalent>")
```

Example

In this example, HsGetSheetInfo tells you whether the worksheet contains an ad hoc grid or a data form.

```
HsGetSheetInfo("Sheet Type")
```

HsCurrency

Data sources: Financial Management, Hyperion Enterprise

HsCurrency retrieves the currency value of the specified dimension member. Entity and Value are the only valid members for the HsCurrency function.

Syntax

```
HsCurrency ("Connection,Entity;Value")
```

Note: Hyperion Enterprise does not use the Value dimension

Example

In this example, HsCurrency retrieves the entity currency where the currency for the East Sales entity is USD, and the currency for the UKSales entity is GBR. The EastSales entity displays USD, and UKSales displays GBR.

```
HsCurrency("Comma","Entity#EastRegion.EastSales;Value#<Entity Currency>.")  
HsCurrency("Comma","Entity#EastRegion.UKSales;Value#<Entity Currency>.")
```

HsDescription

Data sources: Essbase, Financial Management, Hyperion Enterprise

HsDescription displays the description of the specified dimension member.

Syntax

```
HsDescription ("Connection","Dimension#Member")
```

Example

In this example, HsDescription displays the description for Custom 4.

```
HsDescription("HFM01","Custom4#Increases")
```

HsLabel

Data sources: Financial Management, Hyperion Enterprise

HsLabel displays the default member label for the specified dimension member.

Syntax

```
HsLabel ("Connection,Dimension#")
```

Example

In this example, HsLabel function retrieves the label for the Scenario dimension in the Comma application:

```
HsLabel ("Comma","Scenario#")
```

HsGetText

Data sources: Financial Management

HsGetText retrieves cell text from the data source. You can use all dimension members, cell references, the default POV, or a combination of all three.

Syntax

```
HsGetText ("Connection","POV")
```

Example

In this example, HsGetText returns the cell text from the HFM01 data source for the default POV.

```
HsGetText ("HFM01","Scenario#Actual;Year#2004;Period#&B$2&";View#<Scenario View>;Entity#UnitedStates.Connecticut;Value#<Entity Currency>;Account#&$A3&";ICP#[ICP None];Custom1#GolfBalls;Custom2#Customer2;Custom3#[None];Custom4#Increases")
```

HsSetText

Data sources: Financial Management

HsSetText sends cell text to a data source. You can use all dimension members, cell references, the default POV, or a combination of all three.

Syntax

```
HsSetText ("Cell Text Comments","Connection;POV")
```

Example



In this example, HsSetText sends the text from cell H3 to the HFM01 application.

```
HsSetText ("H3","HFM01;Scenario#Actual;Year#2004;Period#&B$2&";View#<Scenario View>;Entity#UnitedStates.Connecticut;Value#<Entity Currency>;Account#&$A3&";ICP#[ICP None];Custom1#GolfBalls;Custom2#Customer2;Custom3#[None];Custom4#Increases")
```

Accessing Functions with a Smart Tag

You can access HsGetValue, HsGetText, HsCurrency, and HsDescription functions with a Microsoft Office smart tag (see Microsoft documentation for information on smart tags). Smart View's smart tag is smartview.

► To access functions using the smart tag:

- 1 Ensure that smart tags are enabled in Excel.
- 2 Ensure that you are connected to a data source.
- 3 Enter `smartview` anywhere in the document, then mouse over it to display the Smart Tags Action icon .
- 4 Click  to display the **Smart View** menu.
- 5 Select **Functions**, then **connection name**, and then a function name.
- 6 From Member Selection, select members as described in [“Selecting Members” on page 37](#)

The results of your selected function are displayed.

Common Function Error Codes

Some common error codes displayed in functions:

#NO CONNECTION - You are not connected or logged on to a data source.

#INVALID - Invalid metadata. Invalid cells that contain a value display the value as zero.

#LOCKED - The cell is locked.

#NO ACCESS - You do not have access to this cell.

#NO DATA - The cell contains NoData. You can select to display zeros instead of NoData. Cells use the Replacement text that you specify in the Options dialog box.

#INVALID INPUT - The HsSetValue data value is not valid, for example, a text string.

#READ ONLY - This is for the HsSetValue function only when the cell is Read-only.

#NO ROLE ACCESS - You do not have the Financial Management LoadExcelData security role.

#NEEDS REFRESH - Data needs to be refreshed.

#INVALID DIMENSION - An invalid dimension is specified in the function.

#INVALID MEMBER - An invalid dimension member name is specified in the function.

#NAME - Excel does not recognize text in a formula. When you forward a worksheet that contains functions to a user who does not have Smart View, they can view the same data as the functions on the worksheet. When the user edits or refreshes the function, it changes to #Name.

In This Chapter

VBA Functions	121
Using Spreadsheet Toolkit VBA Applications in Smart View	253

VBA Functions

You can customize and automate common tasks using Visual Basic for Applications (VBA) functions in Smart View using Excel's Visual Basic Editor.

- “Using VBA Functions” on page 121
- “VBA Functions and Dynamic Link Views” on page 122
- “VBA Parameters” on page 124
- “VBA Return Values” on page 125
- “VBA Functions” on page 128
- “Visual Basic Menu Functions” on page 232
- “Using Spreadsheet Toolkit VBA Applications in Smart View” on page 253

Using VBA Functions

This procedure is an example of how to use VBA functions in Smart View, using [HypConnect](#) as an example.

Note: For information on using Visual Basic Editor, see Excel documentation.

➤ To set up HypConnect:

- 1 **Ensure that the worksheet is active.**
- 2 **From Excel, open Visual Basic Editor and import `smartview.bas`, located by default in `EPM_ORACLE_HOME/smartview/bin`.**

Tip: For convenience, maintain the entire contents of `smartview.bas` in a separate module and copy from it as needed.

3 Open a module and enter VBA code for the function you want to use, in this example:

```
Sub Conn()  
    X=HypConnect(vtSheetName, User, Password, vtFriendlyName)  
End Sub
```

Substitute your user name, password, and connection name for the data source provider.

4 Create a command button.

5 Assign a macro to the command button using the name of the subroutine; in this example, Conn.

6 Optional: Rename the button.

7 To run this function, click the button that you just created.

Declaring Functions

To declare all Smart View VBA functions, in Excel Visual Basic Editor import `smartview.bas`, located by default in `EPM_ORACLE_HOME/smartview/bin` into a module. Then you can use any Smart View Excel VBA function in your program. You can delete any declarations that you do not need.

To declare individual Smart View VBA functions, enter the appropriate declarations `smartview.bas`, using descriptions from [“VBA Functions” on page 128](#)

Calling Functions

When you call a function, observe the following guidelines:

- Substitute the appropriate value for each parameter.
- Type a value for every parameter. If you want to use the default value, enter `Null` or `Empty`.
- Assign the function to a variable. After the function runs, the variable stores the return value, which indicates the success or failure of the function.

VBA Functions and Dynamic Link Views

A link view is used to display the details about a data point in an adjacent window without disturbing the contents in the main window. Link views can be either static or dynamic.

In a static link view, the link action is predefined and details about a data point being queried are displayed in the adjacent window. Static link view behavior is already built in to Smart View.

With a dynamic link view, VBA programmers have the option to change the link behavior as required. Using the set options, you can change the row, column, POV, column information, and the connection information.

For instructions on working with dynamic link views, see:

- [“Setting Up Dynamic Link Views” on page 123](#)

- [“Automating Macro Execution” on page 124](#)

The VBA functions related to dynamic link view:

- [HypUseLinkMacro](#)
- [HypSetLinkMacro](#)
- [HypGetLinkMacro](#)
- [HypGetSourceGrid](#)

Note: For all of the following dynamic link view VBA functions, it is assumed that a call has already been made to [HypGetSourceGrid](#) to initialize the dynamic link query, which contains the information about the active data source and the grid on the sheet.

- [HypGetConnectionInfo](#)
- [HypSetConnectionInfo](#)
- [HypGetRowCount](#)
- [HypGetColCount](#)
- [HypGetPOVCount](#)
- [HypGetRowItems](#)
- [HypGetColItems](#)
- [HypGetPOVItems](#)
- [HypSetRowItems](#)
- [HypSetColItems](#)
- [HypSetPOVItems](#)

When the dynamic link query has been initialized, all the subsequent `setinfo`, `getinfo`, `displaytolinkview` calls are performed on that saved dynamic link query. If the user changes the grid on the sheet and wants to perform the dynamic link action as per the new grid, the user must again initialize the query, using the various `setinfo` calls available.

Prerequisite to [HypGetSourceGrid](#) is that a connected grid must exist on the active sheet and a valid data point should be selected.

Setting Up Dynamic Link Views

Use dynamic link views to customize the link behavior according to your requirements. With dynamic link view, you can change the row, column, POV, and column information as well as the connection information.

➤ To set up a dynamic link view:

- 1 Set the [HypUseLinkMacro](#) flag to true.

2 Set the macro name to run.

The macro name you set should contain all the function calls to initialize the grid and to set the connection, row, POV, and column items as needed.

3 Connect the sheet and retrieve the appropriate grid onto the sheet.

4 Select a data point on the sheet.

5 From the Essbase ribbon, select **Visualize**, then **Visualize in Excel**.

The macro name set in [step 2](#) is executed and the link action is performed.

Note: When the `HypUseLinkMacro` flag is set to false, the predefined link query is performed.

Automating Macro Execution

You can automate execution of a macro through the Smart View menu.

► To set up a macro to execute manually through the Smart View menu:

1 Set the `HypUseLinkMacro` flag to false.

2 Connect the sheet and retrieve a grid.

3 Select a data point on the sheet.

4 Run the macro that contains all the function calls to initialize the grid and set the connection, row, column, and POV items.

VBA Parameters

Most Visual Basic functions require you to supply one or more parameters. [Table 12](#) lists the parameter types and the valid values for each type:

Table 12 VBA Parameters

Parameters	Valid Values
Text	A word or phrase or name in quotes. For example: Smart View [Book2.xls]Sheet1
Boolean	True False

Parameters	Valid Values
Range Object	<p>A cell, row or column, one or more selections of cells, or a three-dimensional range address, surrounded by quotes. For example:</p> <p>RANGE("A1")</p> <p>RANGE("A1:B2")</p> <p>RANGE("A1:B2")</p> <p>RANGE("G:G,I:I,K:K")</p> <p>RANGE("A1:B5,C1:C10,D5:L8")</p> <p>RANGE("Sheet1!C3:R20,Sheet2!C3:R20")</p>
Number	<p>A number without quotes and without commas. For example:</p> <p>1</p> <p>2.5</p> <p>50000</p>
List of Strings	<p>A list of text values, separated by commas. For example:</p> <p>"Qtr1", "Actual", "Oregon"</p>
Constant	A predefined constant from <code>smartview.bas</code> .
Default Value	<p>Null</p> <p>Empty</p> <p>Note: Many parameters have default values or behavior that the function uses if you specify Null or Empty. If you do not specify a value for such parameters, use Null or Empty. See the description of each function for default values of such parameters.</p>

VBA Return Values

Smart View VBA functions return any of the following values to indicate success or failure of the function. Negative numbers represent client issues; positive numbers represent server issues.

[Table 13](#) lists the return values.

Table 13 VBA Return Values

Return Value	Definition
0	Function ran successfully
1	Typically, the user pressed Escape or clicked Cancel from a dialog box.
Large positive number	Failure because of server problem; for example, server not running or an invalid user name.
-1	Valid return value, True
-2	Termination error
-3	Initialization error

Return Value	Definition
-4	Spreadsheet is not yet connected to the server
-6	Not used
-7	Spreadsheet has become unstable
-8	No Undo information exists
-9	Operation has been canceled
-12	Undo is not enabled
-13	Not enough memory resources are available
-14	Appropriate dialog box could not be displayed
-15	Function contains an invalid parameter
-16	Calculation is in progress
-17	Obsolete setting
-18	Operation is not allowed because the spreadsheet is in formula preservation mode
-19	Operation cannot take place on the specified sheet
-20	Current sheet cannot be determined
-21	Spreadsheet name was not specified and no active sheet is selected
-22	Calculation cannot be canceled because no calculation is running
-23	Selection parameter is invalid
-25	Cascade list file cannot be created, or you are attempting to cascade while the spreadsheet is embedded in another document
-26	Spreadsheet macros cannot be run due to a licensing agreement
-27	Spreadsheet macros which update the database cannot be run due to a licensing constraint
-28	Database cannot be updated because you have a read-only license for the database
-29	Obsolete setting
-30	Menu is removed already
-31	Menu is added already
-39	The specified worksheet is protected. Unprotect the worksheet and try the operation again.
-40	Calc script not found.
-41	Provider not supported
-42	Invalid alias

Return Value	Definition
-43	Connection not found
-44	Provider Services connection not found
-45	Provider Services not connected
-46	Provider Services cannot connect
-47	Connection already exists
-48	Provider Services url not saved
-49	Migration of Connection Not Allowed
-50	Connection manager not initialized
-51	Failed to Get Provider Services Override Property
-52	Failed to Set Provider Services Override Property
-53	Failed to Get Provider Services URL
-54	Provider Services disconnect failed
-55	Operation failed
-56	Cannot associate sheet with connection
-57	Refresh sheet needed
-58	No grid object on sheet
-59	No connection associated
-60	Non data cell passed
-61	Data cell is not writeable
-62	No Smart View content on sheet
-63	Failed to get Office object
-64	Operation failed as chart is selected
-65	Excel in edit mode
-66	Sheet not compatible with Smart View
-67	Application not stand alone
-68	Smart View is disabled.
-69	Function has been deprecated.

VBA Functions

Table 14 lists the Smart View VBA functions alphabetically. A detailed description for each function, including the syntax, parameter, return value, and sample code, follows the table.

Table 14 VBA Functions and Supported Providers

VBA Functions
HypCalculate
HypCalculateContribution
HypCell
HypConnect
HypConnected
HypConnectionExists
HypConsolidate
HypConsolidateAll
HypConsolidateAllWithData
HypCopyMetaData
HypCreateConnection
HypCreateConnectionEx
HypDeleteCalc
HypDeleteMetadata
HypDisconnect
HypDisconnectAll
HypDisconnectEx
HypDisplayToLinkView
HypExecuteCalcScript
HypExecuteCalcScriptEx
HypExecuteMDXEx
HypExecuteQuery
HypFindMember
HypFindMemberEx
HypForceCalculate

VBA Functions

[HypForceCalculateContribution](#)

[HypForceTranslate](#)

[HypFreeDataPoint](#)

[HypGetAncestor](#)

[HypGetChildren](#)

[HypGetCellRangeForMbrCombination](#)

[HypGetColCount](#)

[HypGetColItems](#)

[HypGetConnectionInfo](#)

[HypGetDataPoint](#)

[HypGetDimMbrsForDataCell](#)

[HypGetGlobalOption](#)

[HypGetLinkMacro](#)

[HypGetPagePOVChoices](#)

[HypGetParent](#)

[HypGetPOVCount](#)

[HypGetPOVItems](#)

[HypGetRowCount](#)

[HypGetRowItems](#)

[HypGetSharedConnectionsURL](#)

[HypGetSheetOption](#)

[HypGetSourceGrid](#)

[HypGetSubstitutionVariable](#)

[HypInvalidateSSO](#)

[HypIsAncestor](#)

[HypIsAttribute](#)

[HypIsCellWritable](#)

[HypIsChild](#)

VBA Functions

[HypIsConnectedToSharedConnections](#)

[HypIsDataModified](#)

[HypIsDescendant](#)

[HypIsExpense](#)

[HypIsFreeForm](#)

[HypIsParent](#)

[HypIsSmartViewContentPresent](#)

[HypIsUDA](#)

[HypKeepOnly](#)

[HypListCalcScripts](#)

[HypListCalcScriptsEx](#)

[HypOpenForm](#)

[HypOtlGetMemberInfo](#)

[HypPivot](#)

[HypPivotToGrid](#)

[HypPivotToPOV](#)

[HypPreserveFormatting](#)

[HypQueryMembers](#)

[HypRedo](#)

[HypRemoveConnection](#)

[HypRemoveOnly](#)

[HypRemovePreservedFormats](#)

[HypResetFriendlyName](#)

[HypRetrieve](#)

[HypRetrieveAllWorkbooks](#)

[HypRetrieveRange](#)

[HypSetActiveConnection](#)

[HypSetAliasTable](#)

VBA Functions

[HypSetAsDefault](#)

[HypSetBackgroundPOV](#)

[HypSetCellsDirty](#)

[HypSetCollItems](#)

[HypSetConnAliasTable](#)

[HypSetConnectionInfo](#)

[HypSetGlobalOption](#)

[HypSetLinkMacro](#)

[HypSetMenu](#)

[HypSetPages](#)

[HypSetPOV](#)

[HypSetPOVItems](#)

[HypSetRowItems](#)

[HypSetSharedConnectionsURL](#)

[HypSetSheetOption](#)

[HypSetSubstitutionVariable](#)

[HypSubmitData](#)

[HypTranslate](#)

[HypUndo](#)

[HypUseLinkMacro](#)

[HypZoomIn](#)

[HypZoomOut](#)

HypCalculate

Data source types: Financial Management, Hyperion Enterprise

Description

HypCalculate() calls the Calculate method for Financial Management and Hyperion Enterprise data sources.

Syntax

HypCalculate (vtSheetName, vtRange)

ByVal vtSheetName As Variant

By Val vtRange As Variant

Parameters

vtSheetName: For future use; currently the active sheet is used.

vtRange: Range object which refers to the data to be used. Passing an empty or null parameter uses the current selection from the sheet.

Return Value

Returns 0 if successful; otherwise, returns the corresponding error code.

Example

```
Declare Function HypCalculate Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtRange As Variant) As Long
```

```
sts = HypCalculate (Empty, Empty)
```

HypCalculateContribution

Data source types: Financial Management (ad hoc only)

Description

HypCalculateContribution() calls the Calculate Contribution method for Financial Management and Hyperion Enterprise data sources.

Syntax

HypCalculateContribution (vtSheetName, vtRange)

ByVal vtSheetName As Variant

By Val vtRange As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtRange: Range object which refers to the data to be used. Passing an empty or null parameter uses the current selection from the sheet.

Return Value

Returns 0 if successful; otherwise, returns the corresponding error code.

Example

```
Declare Function HypCalculateContribution Lib "HsAddin" (ByVal vtSheetName As Variant,  
ByVal vtRange As Variant) As Long
```

```
sts = HypCalculateContribution (Empty, Empty)
```

HypCell

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypCell() retrieves a cell value for a single member combination.

Syntax

HypCell(vtSheetName, ParamArray MemberList())

ByVal vtSheetName As Variant

ByVal ParamArray MemberList() As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

MemberList: A list of strings that describe the member combination for which a data value will be retrieved. If MemberList is Null or Empty, the top level value is used. Represent members as "Dimension#Member"; for example, "Year#Jan" or "Market#East".

Return Value

Returns the value of the data point if successful. Returns #No Connection if the sheet cannot be determined or is not connected to a data source. Returns "Invalid Member *MemberName* or dimension *DimensionName*" if a member is incorrect.

Example

```
Declare Function HypCell Lib "HsAddin" (ByVal vtSheetName As Variant, ParamArray  
MemberList() As Variant) As Variant
```

```
Sub InCell()  
Dim X As String  
X=HypCell(Empty, "Year#Qtr1", "Scenario#Actual", "Market#Oregon")  
If X = "#No Connection" Then  
    MsgBox("Not logged in, or sheet not active.")
```

```

Else
    If Left(X, 15) = "#Invalid member" then
        MsgBox("Member name incorrect.")
    Else
        MsgBox(X + " Value retrieved successfully.")
    End If
End If
End Sub

```

Note: The value of the data point returned is not placed in a cell in the spreadsheet automatically. To place the value in a cell, use the Visual Basic select method and the ActiveCell property. See your Visual Basic documentation for more information.

HypConnect

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypConnect() logs into a data source provider and associates the worksheet with that connection. HypConnect() must be called for each sheet in order to associate this connection with that sheet.

Syntax

HypConnect(vtSheetName, vtUserName, vtPassword, vtFriendlyName)

ByVal vtSheetName As Variant

ByVal vtUserName As Variant

ByVal vtPassword As Variant

ByVal vtFriendlyName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtUserName: Text name of a valid user for the data source provider.

vtPassword: Text name of the password for this user.

vtFriendlyName: The friendly connection name for the data source provider. This is the connection name created by [HypCreateConnection](#).

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypConnect Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal  
vtUserName As Variant, ByVal vtPassword As Variant, ByVal vtFriendlyName As Variant) As  
Long
```

```
Sub Conn()  
    X=HypConnect(Empty, username, password, "My Sample Basic")  
End Sub
```

HypConnected

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypConnected() provides the connection status of the sheet. A true value indicates that the sheet is connected to a provider; a false value indicates that the sheet is not connected.

Syntax

```
HypConnected (vtSheetName)  
ByVal vtSheetName As Variant
```

Parameters

vtSheetName: For future use. Currently the active sheet is used.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypConnected Lib "HsAddin" (ByVal vtSheetName As Variant) As Variant  
  
Sub Sample_HypConnected  
    Dim X as Variant  
    X = HypConnected(Empty)  
End sub
```

If the sheet is connected, a variant with a value of -1 is returned, which is interpreted as True by VBA. In order to get -1 as the return value, you must declare the variable (which takes a return value) as a number type (Long, Integer, Double, etc.). The script given below demonstrates this:

```
Dim X as Integer 'Can also be Long or Double  
X = HypConnected (Empty) 'Value of X will become -1  
                        'if Sheet1 is connected
```

If variable X is not defined, VBA interprets it (and any other variable which is not defined) as being of the type, Variant. Then, if Sheet1 is connected, X will be equal to True.

If variable X is defined as Variant, the return value is correctly displayed as True.

HypConnectionExists

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypConnectionExists() is used to check if a particular connection name exists in the list of all connections as viewed in the Smart View Panel. The particular connection may or may not be active (i.e., connected).

Syntax

HypConnectionExists(vtConnectionName)

ByVal vtConnectionName as Variant

Parameters

vtConnectionName: Name of the connection to search for in the list of all connections. It is not case-sensitive.

Return Value

Boolean. If successful, return value is TRUE; otherwise, return value is FALSE.

Example

```
Declare Function HypConnectionExists Lib "HsAddin" (ByVal vtConnectionName As Variant)
As Variant

Sub Sample_SetActiveConnection
    Dim bIsConnection as Boolean
    bIsConnection = HypConnectionExists ("Demo_Basic")
End sub
```

HypConsolidate

Data source types: Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypConsolidate calls the Consolidate method for data sources.

Syntax

HypConsolidate (vtSheetName, vtRange)

ByVal vtSheetName As Variant

By Val vtRange As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtRange: Range object which refers to the data to be used. Passing an empty or null parameter uses the current selection from the sheet.

Return Value

Returns 0 if successful; otherwise, returns the corresponding error code.

Example

```
Declare Function HypConsolidate Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtRange As Variant) As Long  
sts = HypConsolidate (Empty, Empty)
```

HypConsolidateAll

Data source types: Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypConsolidateAll() calls the Consolidate All method.

Syntax

HypConsolidateAll (vtSheetName, vtRange)

ByVal vtSheetName As Variant

By Val vtRange As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtRange: Range object which refers to the data to be used. Passing an empty or null parameter uses the current selection from the sheet.

Return Value

Returns 0 if successful; otherwise, returns the corresponding error code.

Example

```
Declare Function HypConsolidateAll Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtRange As Variant) As Long
```

```
sts = HypConsolidateAll (Empty, Empty)
```

HypConsolidateAllWithData

Data source types: Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypConsolidateAllWithData calls the Consolidate All With Data method.

Syntax

HypConsolidateAllWithData (vtSheetName, vtRange)

ByVal vtSheetName As Variant

By Val vtRange As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtRange: Range object which refers to the data to be used. Passing an empty or null parameter uses the current selection from the sheet.

Return Value

Returns 0 if successful; otherwise, returns the corresponding error code.

Example

```
Declare Function HypConsolidateAllWithData Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtRange As Variant) As Long
```

```
sts = HypConsolidateAllWithData (Empty, Empty)
```

HypCopyMetaData

Data Source Types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypCopyMetaData() performs Export Metadata.

Syntax

HypCopyMetaData (vtSourceSheetName, vtDestinationSheetName)

ByVal vtSourceSheetName As Variant

ByVal vtDestinationSheetName As Variant

Parameters

vtSourceSheetName: Name of the source sheet where the Custom Properties should be copied from. (Required)

vtDestinationSheetName: Name of the destination sheet where the Custom Properties should be copied to. (Required)

Return Value

Returns SS_OK if successful; otherwise, the appropriate error code.

Example

```
Public Declare Function HypCopyMetaData Lib "HsAddin" (ByVal vtSourceSheetName As Variant, ByVal vtDestinationSheetName As Variant) As Long
```

```
Sub Sample_HypCopyMetaData()  
    Dim LRet As Long  
    LRet = HypCopyMetaData (Empty, "Sheet1(2)")  
End Sub
```

HypCreateConnection

Data source types: Essbase, Financial Management, Hyperion Enterprise

Description

HypCreateConnection() creates a connection to the data source provider from the specified information. See also HypCreateConnectionEX.

Note: Planning users who want to add data sources in the Smart View Panel must use HypCreateConnectionEX.

Note: Use [HypConnect](#) to establish the connection.

Syntax

HypCreateConnection(vtUserName, vtPassword, vtProvider, vtProviderURL, vtServerName, vtApplicationName, vtDatabaseName, vtFriendlyName, vtDescription)

ByVal vtSheetName As Variant — not used

ByVal vtUserName As Variant

ByVal vtPassword As Variant

ByVal vtProvider As Variant

ByVal vtProviderURL As Variant

ByVal vtServerName As Variant

ByVal vtApplicationName As Variant

ByVal vtDatabaseName As Variant

ByVal vtFriendlyName As Variant

ByVal vtDescription As Variant

Parameters

vtUserName: Text name of a valid user on the server.

vtPassword: Text name of the password for this user.

vtProvider: Description for the data source provider. Supported vtProvider types:

- **New:** Global Const HYP_ESSBASE = "Essbase"
- **New:** Global Const HYP_PLANNING = "Planning"
- **New:** Global Const HYP_OBIEE = "OBIEE"
- **Deprecated:** Global Const HYP_ANALYTIC_SERVICES = "Analytic Provider Services"
- Global Const HYP_FINANCIAL_MANAGEMENT = "Hyperion Financial Management"

vtProviderURL: Data source provider URL which to connect.

vtServerName: Name of the server on which the application resides.

vtApplication: Name of the application.

vtDatabase: Name of the database.

vtFriendlyName: Connection name for the data source provider.

vtDescription: Description for the data source provider.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypCreateConnection Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal  
vtUserName As Variant, ByVal vtPassword As Variant, ByVal vtProvider As Variant, ByVal  
vtProviderURL As Variant, ByVal vtServerName As Variant, ByVal vtApplicationName As  
Variant, ByVal vtDatabase As Variant, ByVal vtFriendlyName As Variant, ByVal  
vtDescription As Variant) As Long  
  
Sub Conn()  
    X=HypCreateConnection(Empty, username, password, HYP_ANALYTIC_SERVICES,  
        "http://localhost:13080/smartview/SmartView", "localhost", "Sample",  
        "Basic", "My Connection", "Analytic Provider Services")  
End Sub
```

HypCreateConnectionEx

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypCreateConnectionEX is a superset of HypCreateConnection; it has additional parameters that enable use of the Smart View Panel. Planning users who want to add data sources in the Smart View Panel must use HypCreateConnectionEX.

Syntax

HypCreateConnectionEx(vtProviderType, vtServerName, vtApplicationName,
vtDatabaseName, vtFormName, vtProviderURL, vtFriendlyName, vtUserName, vtPassword,
vtDescription, vtReserved1, vtReserved2)

ByVal vtProviderType As Variant

ByVal vtServerName As Variant

ByVal vtApplicationName As Variant

ByVal vtDatabaseName As Variant

ByVal vtFormName As Variant

ByVal vtProviderURL As Variant

ByVal vtFriendlyName As Variant

ByVal vtUserName As Variant

ByVal vtPassword As Variant

ByVal vtDescription As Variant

ByVal vtReserved1 As Variant (reserved for future use)

ByVal vtReserved2 As Variant (reserved for future use)

Parameters

vtProviderType: Description for the data source provider. Supported vtProvider types:

- Global Const HYP_ESSBASE = "Essbase"
- Global Const HYP_PLANNING = "Planning"
- Global Const HYP_OBIEE = "OBIEE"
- Global Const HYP_FINANCIAL_MANAGEMENT = "Hyperion Financial Management"

vtServerName: Name of the server on which the application resides.

vtApplication: Name of the application.

vtDatabase: Name of the database.

vtFormName: Name of the data form. Required to create Planning connection in Smart View Panel under Favorites

vtUserName:Text name of a valid user on the server.

vtPassword: Text name of the password for this user.

vtProviderURL: Data source provider URL which to connect. Required to create Planning connection in Smart View Panel.

vtFriendlyName: Connection name for the data source provider.

vtDescription: Description for the data source provider.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypCreateConnectionEx Lib "HsAddin" (ByVal vtProviderType As Variant, ByVal vtServerName As Variant, ByVal vtApplicationName As Variant, ByVal vtDatabaseName As Variant, ByVal vtFormName As Variant, ByVal vtProviderURL As Variant, ByVal vtFriendlyName As Variant, ByVal vtUserName As Variant, ByVal vtPassword As Variant, ByVal vtDescription As Variant, ByVal vtReserved1 As Variant, ByVal vtReserved2 As Variant) As Long
```

```
Sub CreateConnExTest()
```

```
Dim lRet As Long
```

```
lRet = HypCreateConnectionEx("Essbase", "server12", "Demo", "Basic", "", "", "My Demo", "system", "password", "", "", "")
```

```
lRet = HypCreateConnectionEx("Planning", "plange14", "TotPlan", "", "/Forms/Smart View Forms/01 Product Revenue", "http://plange14:8300/HyperionPlanning/SmartView", "My Planning VBA Conn", "admin", "password", "", "", "")
```

```
End Sub
```

HypDeleteCalc

Data source types: Essbase

Description

HypDeleteCalc() allows the user to delete a calculation script object from an Analytic Server.

Syntax

HypDeleteCalc (vtSheetName, vtApplicationName, vtDatabaseName, vtCalcScript)

ByVal vtSheetName As Variant

ByVal vtApplicationName As Variant

ByVal vtDatabaseName As Variant

ByVal vtCalcScript As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtApplicationName: Specify the application name containing the calculation script.

vtDatabaseName: Specify the database name containing the calculation script.

vtCalcScript: Specify the calculation script name to be deleted.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypDeleteCalc Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtApplicationName As Variant, ByVal vtDatabaseName As Variant, ByVal vtCalcScript As Variant) As Long
```

```
Sub Sample_HypDeleteCalc
Dim X as Long
    X = HypDeleteCalc (Empty, "Sample", "Basic", "CalcYear")
End Sub
```

HypDeleteMetadata

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise, Oracle's Hyperion Reporting and Analysis

Description

HypDeleteMetadata() deletes Smart View metadata from the workbook in any of three modes:

- Mode 1 - Delete all Smart View metadata only for the provided worksheet storage
- Mode 2 - Delete all Smart View metadata only from the provided workbook storage
- Mode 3 - Delete all Smart View metadata from the provided workbook storage and from all the worksheets' storage

Syntax

HypDeleteMetadata(vtDispObject, vtbWorkbook, vtbClearMetadataOnAllSheets)

vtDispObject As Variant

vtbWorkbook As Variant

vtbClearMetadataOnAllSheetsWithinWorkbook AsVariant

Parameters

vtDispObject: Dispatch object of worksheet or workbook indicating where to delete metadata. If NULL is passed, the vtbWorkbook flag will determine the active worksheet or active workbook and will be operated upon.

vtbWorkbook: Boolean flag indicating that you passed worksheet dispatch or workbook dispatch. If NULL is passed in vtDispObject, then this flag will determine that the user wants to delete metadata from active worksheet or active workbook.

vtbClearMetadataOnAllSheetsWithinWorkbook: Boolean flag indicating that if Smart View metadata should be deleted from all sheets within the workbook. This flag is used only if vtbWorkbook flag is True.

Return Value

Returns SS_OK if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypDeleteMetaData Lib "HsAddin" (ByVal vtDispObject As Variant, _  
_  
ByVal vtbWorkbook As Variant, _  
ByVal vtbClearMetadataOnAllSheetsWithinWorkbook As Variant) As Long  
  
Sub TestDeleteMetadata()  
    Dim oRet As Long  
    Dim oWorkbook As Workbook  
    Dim oSheet As Worksheet  
  
    Set oWorkbook = ActiveWorkbook  
    Set oSheet = ActiveSheet
```



```

oRet = HypDeleteMetaData(oSheet, False, True)      ` Mode 1
'oRet = HypDeleteMetaData(oWorkbook, True, False) ` Mode 2
'oRet = HypDeleteMetaData(oWorkbook, True, True)  ` Mode 3

MsgBox      (oRet)

End Sub

```

HypDisconnect

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypDisconnect() logs out from the data source provider.

Syntax

HypDisconnect(vtSheetName, bLogoutUser)

ByVal vtSheetName As Variant

ByVal bLogoutUser As Boolean

Parameters

vtSheetName: For future use. Currently the active sheet is used.

bLogoutUser: Optional. Set to True to disconnect and log out from the provider session. Default value is False.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```

Declare Function HypDisconnect Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal
bLogoutUser As Boolean) As Long

Sub DisConn()
    X=HypDisconnect(Empty, True)
End Sub

```

HypDisconnectAll

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypDisconnectAll is a security measure that disconnects all connected users and invalidates the SSO. Equivalent of the **Disconnect All** shortcut menu item.

Syntax

HypDisconnectAll()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypDisconnectAll Lib "HsAddin" () As Long
    Sub SubDisconnectExTest()
End Sub
```

HypDisconnectEx

Data source types: Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypDisconnectEx disconnects the connection with the the connection (friendly) name passed in the argument. The connection to be disconnected need not be associated as in HypDisconnect.

Syntax

HypDisconnectEx (vtConnFriendlyName)
ByVal vtConnFriendlyName as Variant

Parameters

vtConnFriendlyName: The friendly connection name

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypDisconnectEx Lib "HsAddin" (ByVal vtConnFriendlyName As Variant) As Long
```

```
Sub SubDisconnectExTest()  
    Dim lRet As Long  
    lRet = HypDisconnectEx("My Sample")  
End Sub
```

HypDisplayToLinkView

Data source types: Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypDisplayToLinkView() displays Office documents to Word or PowerPoint; or displays a grid to Excel.

Note: The link action is performed as per the latest content of the dynamic link query.

Note: This function is used specifically with dynamic link views, as described in Dynamic Link Views

Syntax

```
HypDisplayToLinkView (vtDocumentType, vtDocumentPath)  
ByVal vtDocumentType As Variant  
vtDocumentPath As Variant
```

Parameters

vtDocumentType: Indicates the destination for the link view. Valid values:

- EXCEL_APP
- WORD_APP
- PPOINT_APP

vtDocumentPath: The path to the document. Required only in case of WORD_APP or PPOINT_APP.

Return Value

Returns 0 if successful; otherwise, returns the negative error code.

Examples

```
Declare Function HypDisplayToLinkView Lib "HsAddin" (ByVal vtDocumentType As Variant,  
ByVal vtDocumentPath As Variant) As Long
```

```

Sub Macro()
    Dim vtGrid as Variant
    Sts = HypConnect(Empty, "system", "password", "MyDemoBasic")
    Sts = HypRetrieve(Empty)
    Range ("B2").Select
    Sts = HypGetSourceGrid (Empty, vtGrid)
    Sts = HypSetColItems (1, "Market", "East", "West", "South", "Central",
        "Market")
    Sts = HypDisplayToLinkView ("EXCEL_APP", "")
End sub

```

HypExecuteCalcScript

Data source types: Essbase

Description

HypExecuteCalcScript() uses a calculation script (business rule script) to initiate a calculation on the server.

Syntax

HypExecuteCalcScript (vtSheetName, vtCalcScript, bSynchronous)

ByVal vtSheetName As Variant

ByVal vtCalcScript As Variant

ByVal bSynchronous As Boolean

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtCalcScript: Text name of the calculation script on the Analytic Server in the database directory to run. To run the default calculation script, use "Default".

bSynchronous: Boolean value indicating whether the calculation script should be run synchronously. If synchronous is Null or Empty, True is used. Currently this flag is unused.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```

Declare Function HypExecuteCalcScript Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal
vtCalcScript As Variant, ByVal bSynchronous As Variant) As Long

```

```

Sub RunCalculate()
    X = HypExecuteCalcScript (Empty, "Default", False)

```

```

    If X = 0 Then
        MsgBox("Calculation complete.")
    Else
        MsgBox("Calculation failed.")
    End If
End Sub

```

HypExecuteCalcScriptEx

Data source types: Essbase, Planning

Description

HypExecuteCalcScriptEx() executes the selected business rule.

Syntax

HypExecuteCalcScriptEx(vtSheetName [in], vtCubeName [in], vtBRName [in], vtBRType [in],
 vtbBRHasPrompts [in], vtbBRNeedPageInfo [in], vtRTPNames() [in], vtRTPValues() [in],
 vtbShowRTPDlg [in], vtbRuleOnForm [in], vtbBRRanSuccessfully [out], vtCubeName [out],
 vtBRName [out], vtBRType [out], vtbBRHasPrompts [out], vtbBRNeedPageInfo [out],
 vtbBRHidePrompts [out], vtRTPNamesUsed [out], vtRTPValuesUsed [out])

ByVal vtSheetName As Variant

ByVal vtCubeName As Variant

ByVal vtBRName As Variant

ByVal vtBRType As Variant

ByVal vtbBRHasPrompts As Variant

ByVal vtbBRNeedPageInfo As Variant

ByRef vtRTPNames() As Variant

ByRef vtRTPValues() As Variant

ByVal vtbShowRTPDlg As Variant

ByVal vtbRuleOnForm As Variant

ByRef vtbBRRanSuccessfully As Variant

ByRef vtCubeName As Variant

ByRef vtBRName As Variant

ByRef vtBRType As Variant

ByRef vtbBRHasPrompts As Variant

ByRef vtbBRNeedPageInfo As Variant

ByRef vtbBRHidePrompts As Variant

ByRef vtRTPNamesUsed As Variant

ByRef vtRTPValuesUsed As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtCubeName: Cube Name (Plan type incase of Planning) Business Rule is associated with

vtBRName: Business Rule Name of the BR to be run

vtBRType: Business Rule Type for the BR to be run

vtbBRHasPromps: Boolean indicating if the Business Rule has RTPs

vtbNeedPageInfo: Boolean indicating if the Business Rule needs Page Info to be run (Get this info either from HypListCalcScriptsEx or from prior run of HypExecuteCalcScriptEx)

ppRTPNames: Array of RTP Names associated with the Business Rule

ppRTPValues: Array of RTP Values corresponding to the RTP Names

vtbShowBRDlg: Boolean indicating whether to show the Business Rule dialog box and let the user select the Business Rule to run or of automating execution of BR. If this flag is true, all the input parameters related to the BR are ignored. Recommendation: This flag should be true when running the BR for the first time and then using the output paramters to automate the execution of the same BR from second time onwards. In this case, this flag should be false second time

vtbRuleOnForm: Boolean indicating if the Business Rule is associated to the form opened on active sheet

pvtbBRRanSuccessfully: Return boolean value indicating if the last Business Rule ran successfully

pvtCubeNameUsed: Cube name (Plan Types incase of Planning) associated with the last run Business Rule

pvtBRNameUsed: Business Rule Name of the last run Business Rule

pvtBRTypeUsed: Business Rule type of the last run Business Rule

pvtbBRHasPrompts: Boolean indicating if the last run Business Rule has RTPs

pvtbBRNeedPageInfo: Boolean indicating if the last run Business Rule requires Page information

pvtbBRHidePrompts: Boolean indicating if the last run Business Rule has hidden RTPs

pvtRTPNamesUsed: Array of RTP Names used to run last run Business Rule

pvtRTPValuesUsed: Array of RTP Values associated with RTP names used to run last run Business Rule

Return Value

Returns SS_OK if successful; otherwise, the appropriate error code.

Example

```
Declare Function HypListCalcScriptsEx Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal  
vtbRuleOnForm As Variant, ByRef vtCubeNames As Variant, ByRef vtBRNames As Variant,  
ByRef vtBRTypes As Variant, ByRef vtBRHasPrompts As Variant, ByRef vtBRNeedsPageInfo As  
Variant, ByRef vtBRHidePrompts As Variant) As Long
```

```
Declare Function HypExecuteCalcScriptEx Lib "HsAddin" (ByVal vtSheetName As Variant,  
ByVal vtCubeName As Variant, ByVal vtBRName As Variant, ByVal vtBRType As Variant, ByVal  
vtbBRHasPrompts As Variant , ByVal vtbBRNeedPageInfo As Variant, ByRef vtRTPNames() As  
Variant, ByRef vtRTPValues() As Variant, ByVal vtbShowRTPDlg As Variant, ByVal  
vtbRuleOnForm As Variant, ByRef vtbBRRanSuccessfully As Variant, ByRef vtCubeName As  
Variant, ByRef vtBRName As Variant, ByRef vtBRType As Variant, ByRef vtbBRHasPrompts As  
Variant, ByRef vtbBRNeedPageInfo As Variant, ByRef vtbBRHidePrompts As Variant, ByRef  
vtRTPNamesUsed As Variant, ByRef vtRTPValuesUsed As Variant) As Long
```

```
Sub TestListAndExecuteCalcScriptsEx()
```

```
Dim oRet As Long  
Dim oSheetName As String  
Dim oSheet As Worksheet  
Dim vtCubeNames As Variant  
Dim vtBRNames As Variant  
Dim vtBRTypes As Variant  
Dim vtBRHasPrompts As Variant  
Dim vtBRNeedsPageInfo As Variant  
Dim vtBRHidePrompts As Variant  
Dim sAllCalcs As String  
Dim sCalcName As String  
Dim bNeedPageInfo As Variant  
Dim vtInRTPNames() As Variant  
Dim vtInRTPValues() As Variant  
Dim vtOutRTPNames As Variant  
Dim vtOutRTPValues As Variant  
Dim vtbBRRanSuccessfully As Variant  
Dim vtbBRRanSuccessfully2 As Variant  
Dim vtOutCubeName As Variant  
Dim vtOutBRName As Variant  
Dim vtOutBRType As Variant  
Dim bBRHasPrompts As Variant  
Dim bBRNeedPageInfo As Variant  
Dim bBRHidePrompts As Variant  
Dim bShowDlg As Variant  
Dim bRuleOnForm As Variant
```

```
'Set oSheet = ActiveSheet  
'oSheetName = oSheet.Name  
oSheetName = "Sheet3"
```

```
oRet = HypListCalcScriptsEx (oSheetName, False, vtCubeNames, vtBRNames, vtBRTypes,  
vtBRHasPrompts, vtBRNeedsPageInfo, vtBRHidePrompts)  
If (oRet = 0) Then  
    If IsArray(vtBRNames) Then  
        lNumMbrs = (UBound(vtBRNames) - LBound(vtBRNames) + 1)  
    End If  
  
    sPrintMsg = "Number of Calc Scripts = " & lNumMbrs
```

```

MsgBox (sPrintMsg)

'Start Executing the Calc Script

bShowDlg = True
bRuleOnForm = False
iScript = 1

oRet = HypExecuteCalcScriptEx (oSheetName, vtCubeNames(iScript), vtBRNames(iScript),
    vtBRTypes(iScript), vtBRHasPrompts(iScript), vtBRNeedsPageInfo(iScript), vtInRTPNames,
    vtInRTPValues, bShowDlg, bRuleOnForm, vtbBRRanSuccessfully, vtOutCubeName, vtOutBRName,
    vtOutBRType, bBRHasPrompts, bBRNeedPageInfo, bBRHidePrompts, vtOutRTPNames,
    vtOutRTPValues)
If (oRet = 0) Then
    MsgBox ("Last BR ran successfully - " & vtbBRRanSuccessfully)

    If (vtbBRRanSuccessfully = True) Then
        bShowDlg = False
        bRuleOnForm = False

        If IsArray(vtOutRTPNames) And IsArray(vtOutRTPValues) Then
            lNumRTPNames = (UBound(vtOutRTPNames) - LBound(vtOutRTPNames) + 1)
            lNumRTPVals = (UBound(vtOutRTPValues) - LBound(vtOutRTPValues) + 1)
        End If

        If (lNumRTPNames > 0) Then
            ReDim vtInRTPNames(lNumRTPNames - 1) As Variant
            ReDim vtInRTPValues(lNumRTPNames - 1) As Variant

            For iRTPs = 0 To lNumRTPNames - 1
                sBRName = vtOutRTPNames(iRTPs)
                sBRVal = vtOutRTPValues(iRTPs)

                vtInRTPNames(iRTPs) = sBRName
                vtInRTPValues(iRTPs) = sBRVal
            Next iRTPs
        End If

        oRet = HypExecuteCalcScriptEx (oSheetName, vtOutCubeName, vtOutBRName,
            vtOutBRType, bBRHasPrompts, bBRNeedPageInfo, vtInRTPNames, vtInRTPValues, bShowDlg,
            bRuleOnForm, vtbBRRanSuccessfully2, vtOutCubeName, vtOutBRName, vtOutBRType,
            bBRHasPrompts, bBRNeedPageInfo, bBRHidePrompts, vtOutRTPNames, vtOutRTPValues)
        MsgBox ("Automated BR ran successfully - " & vtbBRRanSuccessfully2)
    End If
Else
    sPrintMsg = "Error - " & oRet
    MsgBox (sPrintMsg)
End If
Else
    sPrintMsg = "Error - " & oRet
    MsgBox (sPrintMsg)
End If

End Sub

```


Usage

You can use HypExecuteCalcScriptEx in four modes, depending on whether HypListCalcScriptsEx is called before HypExecuteCalcScriptEx.

If you do NOT call HypListCalcScriptsEx before HypExecuteCalcScriptEx, then the first time you call HypListCalcScriptsEx you should set the vtbShowBRDlg argument to true for the first usage and to false thereafter.

- When vtbShowBRDlg argument is true (mode 1):
 - In arguments: vtSheetName, vtCubeName, vtbRuleOnForm are used. vtBRName, vtBRType, vtbBRHasPrompts, vtbNeedPageInfo, ppRTPNames, ppRTPValues are ignored.
 - Behavior: The Business Rule dialog box displays all possible rules depending upon the vtbRuleOnForm value. When the user selects, runs and exits the Business Rule dialog box, the details of that Business Rule are filled in the out arguments and returned to the caller.
 - Out arguments: All out arguments are filled and returned to the caller so that they can be used in subsequent calls.
- When vtbShowBRDlg argument is false (mode 2):
 - In arguments: All in arguments are used.
 - Behavior: The business rule is run without displaying the Business Rule dialog box, and the appropriate status is returned to the caller.
 - Out arguments: All out arguments are left unmodified as nothing needs to be passed on to the caller, who already has all the information to run this particular business rule.

If you DO call HypListCalcScriptsEx before HypExecuteCalcScriptEx, then when HypListCalcScriptsEx is called, users get information about all business rules and runtime prompts (RTP), if any.

If a user runs a business rule that has no RTP, HypExecuteCalcScriptEx can be called with vtbShowBRDlg argument as false and provides all other information as the in arguments.

If a user runs a business rule that has an RTP, HypExecuteCalcScriptEx must be called with vtbShowBRDlg as true so that the business rule and its RTPs can be displayed and the user can select the RTP values to run the business rule. (Note: inPlanning, the RTP flag may be true for a business rule when there are no RTPs to be displayed.)

- If the cube name, business rule name and business rule type are passed as empty in HypExecuteCalcScriptEx (mode 3), the Business Rule dialog box is displayed and all business rules are shown depending upon vtbRuleOnForm argument. All else is the same as mode 1.
- If the cube name, business rule name and business rule type are passed with filled values in HypExecuteCalcScriptEx (mode 4), the Business Rule dialog box is displayed and only the passed business rule (business rule name for the provided cube name) is displayed along with its RTPs. All else is the same as mode 1.

HypExecuteMDXEx

Data source types: Essbase

Description

HypExecuteMDXEx() executes an MDX query whose results are output in a data structure but are not displayed on the worksheet. (If you want to display the query results on a worksheet, use HypExecuteQuery instead.)

Syntax

```
HypExecuteMDXEx  
(  
ByVal vtSheetName As Variant,  
ByVal vtQuery As Variant,  
ByVal vtBoolHideData As Variant,  
ByVal vtBoolDataLess As Variant,  
ByVal vtBoolNeedStatus As Variant,  
ByVal vtMbrIDType As Variant,  
ByVal vtAliasTable As Variant,  
ByRef outResult As MDX_AXES_NATIVE  
) As Long
```

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtQuery: The MDX query to be executed

vtBoolHideData: The Boolean flag hide or unhide data in the result

vtBoolDataless: The Boolean flag to get or avoid data in the result

vtBoolNeedStatus: The Boolean flag to get or avoid status info in the result

vtMbrIDType: The member type identifier for the result (name or alias)

vtAliasTable: The alias table to be used

outResult: Pointer to a structure of type MDX_AXES. It contains the query output. (See [Data Types Specific to HypExecuteMDXEx](#) for data types and support functions for this API)

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Data Types Specific to HypExecuteMDXEx

The following data types apply exclusively to HypExecuteMDXEx:

MDX_CELL: The data type corresponding to a cell

MDX_PROPERTY: The data type containing properties info for members and dimensions

MDX_MEMBER: The data type for members information

MDX_DIMENSION: The data type for dimensions information

MDX_CLUSTER: The data type for cluster information

MDX_AXIS: The data type representing an axis

MDX_AXES: The root level structure containing a collection of axes and cells

MDX_AXES_NATIVE: The data type used as an out parameter for HypExecuteMDXEx. This structure should be converted to MDX_AXES using procedure GetVBCompatibleMDXStructure.

Example

```
Sub GetVBCompatibleMDXStructure (ByRef inStruct As MDX_AXES_NATIVE, ByRef outStruct As MDX_AXES)
```

```
Public Declare Function HypExecuteMDXEx Lib "HsAddin"
    (ByVal vtSheetName As Variant, ByVal vtQuery As Variant, ByVal vtBoolHideData As Variant,
    ByVal vtBoolDataLess As Variant, ByVal vtBoolNeedStatus As Variant, ByVal vtMbrIDType As Variant,
    ByVal vtAliasTable As Variant, ByRef outResult As MDX_AXES_NATIVE) As Long
```

```
Sub Example_HypExecuteMDXEx ()
```

```
Dim Query As Variant
Dim vtBoolHideData As Variant
Dim vtBoolDataLess As Variant
Dim vtBoolNeedStatus As Variant
Dim vtMbrIDType As Variant
Dim vtAliasTable As Variant
Dim result_Native As MDX_AXES_NATIVE
Dim result_VBCompatible As MDX_AXES
```

```
Query = "select {Jan} on COLUMNS, {Profit} on ROWS from Sample.Basic"
vtBoolHideData = True
vtBoolDataLess = True
vtBoolNeedStatus = True
vtMbrIDType = "alias"
vtAliasTable = "none"
```

```
sts = HypConnect(Empty, "system", "password", "SB")
```

```
If sts = 0 Then
```

```
sts = HypExecuteMDXEx (Empty, Query, vtBoolHideData, vtBoolDataLess, vtBoolNeedStatus,
vtMbrIDType, vtAliasTable, result_Native)
```

```
sts = GetVBCompatibleMDXStructure (result_Native, result_VBCompatible) --- New support
function ... More Info under Notes section
```

```
sts = HypDisconnect(Empty, True)
```

```
Else
```

End If

End Sub

HypExecuteQuery

Data source types: Essbase

Description

HypExecuteQuery() executes an MDX query and displays the results on a worksheet. (If you do not want to display the query results on a worksheet, use HypExecuteMDXEx instead.)

Syntax

HypExecuteQuery (ByVal vtSheetName As Variant, ByVal vtMDXQuery As Variant) As Long

ByVal vtSheetName As Variant

ByVal vtMDXQuery

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtMDXQuery: The MDX query statement to be executed on the worksheet.

Return Value

Long. If successful, return value is 0; otherwise, returns the appropriate error code.

Example

```
Declare Function HypExecuteQuery Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal  
vtMDXQuery As Variant) As Long
```

```
Sub Sample_HypExecuteQuery ()  
    Dim vtQuery As Variant  
    vtQuery = "SELECT {[Jan]} on COLUMNS, {[East]} on ROWS from  
                Sample.Basic"  
    sts = HypConnect (Empty, "system", "password", "Sample_Basic")  
    sts = HypExecuteQuery (Empty, vtQuery)  
    sts = HypDisconnect (Empty, True)  
End sub
```

HypFindMember

Data source types: Essbase

Description

HypFindMember() retrieves member information like dimension, alias, generation and level numbers.

Syntax

HypFindMember (vtSheetName, vtMemberName, vtAliasTable, vtDimensionName, vtAliasName, vtGenerationName, vtLevelName)

ByVal vtSheetName As Variant

ByVal vtMemberName As Variant

ByVal vtAliasTable As Variant

ByRef vtDimensionName As Variant

ByRef vtAliasName As Variant

ByRef vtGenerationName As Variant

ByRef vtLevelName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtMemberName: The name of the member. This parameter is required because there is no default value.

vtAliasTable: The name of the alias table to search for the alias name. If Null, the default alias table is searched.

vtDimensionName: The output parameter that contains the dimension, if successful.

vtAliasName: The output parameter that contains the alias name of the member, if successful.

vtGenerationName: The output parameter that contains the generation name of the member, if successful.

vtLevelName: The output parameter that contains the level name of the member, if successful.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypFindMember Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtMemberName As Variant, ByVal vtAliasTable As Variant, ByRef vtDimensionName As Variant, ByRef vtAliasName As Variant, ByRef vtGenerationName As Variant, ByRef vtLevelName As Variant) As Long
```

```
Sub FindMember()  
    X = HypFindMember(Empty, "100", "Default", dimName, aliasName,
```

```

        genName, levelName)
    MsgBox (dimName)
    MsgBox (aliasName)
    MsgBox (genName)
    MsgBox (levelName)
End Sub

```

HypFindMemberEx

Data source types: Essbase

Description

HypFindMemberEx() retrieves member information like dimension, alias, generation and level names.

Syntax

HypFindMember (vtSheetName, vtMemberName, vtAliasTable, vtDimensionName, vtAliasName, vtGenerationName, vtLevelName)

ByVal vtSheetName As Variant

ByVal vtMemberName As Variant

ByVal vtAliasTable As Variant

ByRef vtDimensionName As Variant

ByRef vtAliasName As Variant

ByRef vtGenerationName As Variant

ByRef vtLevelName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtMemberName: The name of the member. This parameter is required because there is no default value.

vtAliasTable: The name of the alias table to search for the alias name. If Null, the default alias table is searched.

vtDimensionName: The output parameter that contains the dimension, if successful.

vtAliasName: The output parameter that contains the alias name of the member, if successful.

vtGenerationName: The output parameter that contains the generation name of the member, if successful.

vtLevelName: The output parameter that contains the level name of the member, if successful.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypFindMember Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtMemberName As Variant, ByVal vtAliasTable As Variant, ByRef vtDimensionName As Variant, ByRef vtAliasName As Variant, ByRef vtGenerationName As Variant, ByRef vtLevelName As Variant) As Long
```

```
Sub FindMember()  
    X = HypFindMember(Empty, "100", "Default", dimName, aliasName, genName, levelName)  
    MsgBox (dimName)  
    MsgBox (aliasName)  
    MsgBox (genName)  
    MsgBox (levelName)  
End Sub
```

HypForceCalculate

Data source types: Financial Management

Description

HypForceCalculate() calls the Force Calculate method for Financial Management data sources.

Syntax

HypForceCalculate(vtSheetName, vtRange)

ByVal vtSheetName As Variant

By Val vtRange As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtRange: Range object which refers to the data to be used. Passing an empty or null parameter uses the current selection from the sheet.

Return Value

Returns 0 if successful; otherwise, returns the corresponding error code.

Example

```
Declare Function HypForceCalculate Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtRange As Variant) As Long
```

```
sts = HypForceCalculate (Empty, Empty)
```

HypForceCalculateContribution

Data source types: Financial Management (ad hoc only)

Description

HypForceCalculateContribution calls the Force Calculate Contribution method for Financial Management data sources.

Syntax

```
HypForceCalculateContribution (vtSheetName, vtRange)
```

ByVal vtSheetName As Variant

By Val vtRange As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtRange: Range object which refers to the data to be used. Passing an empty or null parameter uses the current selection from the sheet.

Return Value

Returns 0 if successful; otherwise, returns the corresponding error code.

Example

```
Declare Function HypForceCalculateContribution Lib "HsAddin" (ByVal vtSheetName As  
Variant, ByVal vtRange As Variant) As Long
```

```
sts = HypForceCalculateContribution (Empty, Empty)
```

HypForceTranslate

Data source types: Financial Management (ad hoc only)

Description

HypForceTranslate calls the Force Translate method for Financial Management data sources.

Syntax

HypForceTranslate (vtSheetName, vtRange)

ByVal vtSheetName As Variant

By Val vtRange As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtRange: Range object which refers to the data to be used. Passing an empty or null parameter uses the current selection from the sheet.

Return Value

Returns 0 if successful; otherwise, returns the corresponding error code.

Example

```
Declare Function HypForceTranslate Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtRange As Variant) As Long
```

```
sts = HypForceTranslate (Empty, Empty)
```

HypFreeDataPoint

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypFreeDataPoint() frees any memory allocated by HypGetDataPoint.

Syntax

HypFreeDataPoint()

ByRef vtInfo As Variant

Parameters

vtInfo: Variant array returned by HypGetDataPoint.

Return Value

Returns 0 if successful; returns -15 if not successful.

Example

See [“HypGetDataPoint” on page 168](#) for an example of HypFreeDataPoint.

HypGetAncestor

Data source types: Essbase

Description

HypGetAncestor() returns the ancestor at any specific generation/level for the specified member.

Syntax

HypGetAncestor (vtSheetName, vtMemberName, vtLayerType, intLayerNum, vtAncestor)

ByVal vtSheetName As Variant

ByVal vtMemberName As Variant

ByVal vtLayerType As Variant

ByVal intLayerNum As Integer

ByRef vtAncestor As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtLayerType: Specify either “Gen” or “Level”. If vtLayerType is Null or Empty, Gen is taken as default.

vtMemberName: Specify a member name. Required field.

intLayerNum: Specify the Level/Generation number. Required Field.

vtAncestor: Output. Contains the ancestor name on successful execution of the macro.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypGetAncestor Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal  
vtMemberName As Variant, ByVal vtLayerType As Variant, ByVal intLayerNumber As Integer,  
ByRef vtAncestor As Variant) As Long
```

```
Sub Sample_HypGetAncestor  
    Dim X as Long  
    Dim vtAncestor as Variant
```

```
    X = HypGetAncestor (Empty, "100-20", "Level", 1, vtAncestor)
End Sub
```

HypGetCellRangeForMbrCombination

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypGetCellRangeForMbrCombination() retrieves the cell range for the selected combination of members.

Syntax

HypGetCellRangeForMbrCombination (vtSheetName [in], ppvtDimNames [in],
ppvtMbrNames [in], pvtCellIntersectionRange [out])

By Val vtSheetName As Variant

ByRef vtDimNames As Variant

ByRef vtMbrNames As Variant

ByRef vtCellIntersectionRange As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

ppvtDimNames: Array of dimension names

ppvtMbrNames: Array of member names corresponding to the dimensions (in the same order)

pvtCellIntersectionRange: Range of the cell(s) on the grid

Return Value

Returns SS_OK if successful; otherwise, the appropriate error code.

Example

```
Public Declare Function HypGetCellRangeForMbrCombination Lib "HsAddin" (ByVal  
vtSheetName As Variant, ByRef vtDimNames() As Variant, ByRef vtMbrNames() As Variant,  
ByRef vtCellIntersectionRange As Variant) As Long
```

```
Sub GetCellRangeForMbrCombination()  
  
    Dim oRet As Long  
    Dim oSheetName As String  
    Dim oSheetDisp As Worksheet  
    Dim vtDimNames(3) As Variant  
    Dim vtMbrNames(3) As Variant
```

```

Dim vtReturnCellRange As Variant
Dim oRange As Range

'oSheetName = Empty
'Set oSheetDisp = Worksheets(oSheetName$)

vtDimNames(0) = "Measures"
vtDimNames(1) = "Market"
vtDimNames(2) = "Year"
vtDimNames(3) = "Product"
'vtDimNames(4) = " "

vtMbrNames(0) = "Sales"
vtMbrNames(1) = "New York"
vtMbrNames(2) = "Year"
vtMbrNames(3) = " Product"
'vtMbrNames(4) = " "

oRet = HypGetCellRangeForMbrCombination ("", vtDimNames, vtMbrNames, vtReturnCellRange)

If (oRet = 0) Then
    Set oRange = vtReturnCellRange
End If

```

HypGetChildren

Data source types: Essbase

Description

HypGetChildren() returns the children for the specified member.

Syntax

HypGetChildren (vtSheetName, vtMemberName, intChildCount, vtChildArray)

ByVal vtSheetName As Variant

ByVal vtMemberName As Variant

ByVal intChildCount As Integer

ByRef vtChildArray As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtMemberName: Specify a member name. Required Field.

intChildCount: To restrict the number of children returned.

- ChildCount <=0. All children are returned.

- ChildCount >0. The result set is limited to the number specified as the argument. If the result set is less than the specified argument, all result are returned.

vtChildArray: Output Result Vector that contains the list of the children. Its contents are unknown if the macro fails.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypGetChildren Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtMemberName As Variant, ByVal intChildCount As Integer, ByRef vtChildNameArray As Variant) As Long
```

```
Sub Sample_HypGetChildren
    Dim vtChildren as Variant
    Dim vtChild as Variant
    Dim X as Long
    X = HypGetChildren (Empty, "Market", 0, vtChildren)
    If IsArray (vtChildren) Then
        For i = LBound (vtChildren) To UBound (vtChildren)
            VtChild = vtChildren (i)
        Next
    End If
End Sub
```

HypGetColCount

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypGetColCount() returns the number of column dimensions.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#)

Syntax

HypGetColCount()

Return Value

Returns the number of column dimensions if successful; otherwise, returns the negative error code.

Example

```
Declare Function HypGetColCount Lib "HsAddin" () As Long
```

```
Sub Macro()  
    Dim vtGrid as Variant  
    Sts = HypConnect(Empty, "system", "password", "MyDemoBasic")  
    Sts = HypRetrieve(Empty)  
    Range ("B2").Select  
    Sts = HypGetColCount ()  
End sub
```

HypGetColItems

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypGetColItems() returns the members present in the dynamic link query for the nth column dimensions.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#)

Syntax

HypGetColItems(vtColumnID, vtDimensionName, vtMembers)

ByVal vtColumnID As Variant

ByRef vtDimensionName As Variant

ByRef vtMembers As Variant

Parameters

vtColumnID: The column number n.

vtDimensionName: Returns the nth column dimension name.

vtMembers: Returns members for the nth column dimensions.

Return Value

Returns 0 if successful; otherwise, returns the negative error code.

Example

```
Declare Function HypGetColItems Lib "HsAddin" (ByVal vtColID As Variant, ByRef  
vtDimensionName As Variant, ByRef vtMembernames As Variant) As Long
```

```

Sub Macro()
    Dim vtGrid as Variant
    Dim vtDimensionName as Variant
    Dim vtMembers as Variant
    Sts = HypConnect(Empty, "system", "password", "AnamikaDemoBasic")
    Sts = HypRetrieve(Empty)
    Range ("B2").Select
    Sts = HypGetSourceGrid (Empty, vtGrid)
    Sts = HypGetColItems (1, vtDimensionName, vtMembers)
End sub

```

HypGetConnectionInfo

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypGetConnectionInfo() returns the connection information for the dynamic link query.

HypGetConnectionInfo assumes that a call has already been made to HypGetSourceGrid to initialize the dynamic link query, which contains the information about the active data source and the grid on the sheet.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#)

Note: After a call, the password is not actually returned, but for security reasons, is returned as empty.

Syntax

HypGetConnectionInfo(vtServerName, vtUserName, vtPassword, vtApplicationName, vtDatabaseName, vtFriendlyName, vtURL, vtProviderType)

ByRef vtServerName As Variant

ByRef vtUserName As Variant

ByRef vtPassword As Variant

ByRef vtApplicationName As Variant

ByRef vtDatabaseName As Variant

ByRef vtFriendlyName As Variant

ByRef vtURL As Variant

ByRef vtProviderType As Variant

Parameters

vtServerName: Output. Contains the server name for the dynamic link query.

vtUserName: Output. Contains the user name for the dynamic link query.

vtPassword: The user password.

vtApplicationName: Output. Contains the application name for the dynamic link query.

vtDatabaseName: Output. Contains the database name for the dynamic link query.

vtFriendlyName: Output. Contains the friendly connection name for the dynamic link query.

vtURL: Output. Contains the URL for the dynamic link query.

vtProviderType: Output. Contains provider type for the dynamic link query.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypGetConnectionInfo Lib "HsAddin" (ByRef vtServerName As Variant,  
ByRef vtUserName As Variant, ByRef vtPassword As Variant, ByRef vtApplicationName As  
Variant, ByRef vtDatabaseName As Variant, ByRef vtFriendlyName As Variant, ByRef vtURL  
As Variant, ByRef vtProviderType As Variant) As Long
```

```
Sub Macro()  
    Dim vtGrid as Variant  
    Dim server As Variant  
    Dim user As Variant  
    Dim app As Variant  
    Dim pass As Variant  
    Dim db As Variant  
    Dim provider As Variant  
    Dim conn As Variant  
    Dim url As Variant  
    Sts = HypConnect(Empty, "system", "MyDemoBasic")  
    Sts = HypRetrieve(Empty)  
    Range ("B2").Select  
    Sts = HypGetSourceGrid (Empty, vtGrid)  
    Sts = HypGetConnectionInfo(server, user, pass, app, db, conn, url, provider)  
End sub
```

HypGetDataPoint

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypGetDataPoint() retrieves member information for a single data cell. For example, to find out the members that consist of the data intersection at cell B6, HypGetDataPoint may return the members January, California, Actual, Root Beer, Profit.

Syntax

HypGetDataPoint (vtSheetName, vtCell)

By Val vtSheetName As Variant

By Val vtCell As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtCell: Cell (range) that describes the reference cell for which to retrieve the member combination information.

Return Value

Returns an array of member names.

Example

```
Declare Function HypGetDataPoint Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal cell As Variant) As Variant
```

```
Sub DataPointsSub()  
Dim vt As Variant  
Dim cbItems As Variant  
Dim i As Integer  
Dim pMember As String  
vt = HypGetDataPoint(Empty, range ("B3")).  
If IsArray(vt) Then  
    cbItems = UBound(vt) - LBound(vt) + 1  
    MsgBox ("Number of elements = " + Str(cbItems))  
  
    For i = LBound(vt) To UBound(vt)  
        MsgBox ("Member = " + vt(i))  
    Next  
    X = HypFreeDataPoint(vt)  
Else  
MsgBox ("Return Value = " + Str(vt))  
End If  
End Sub
```

HypGetDimMbrsForDataCell

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypGetDimMbrsForDataCell() retrieves the entire set of dimension members for a data cell.

Syntax

HypGetDimMbrsForDataCell (vtSheetName [in], vtCellRange [in], vtServerName [out], vtAppName [out], vtCubeName [out], vtFormName [out], vtDimensionNames [out], vtMemberNames [out])

ByVal vtSheetName As Variant

ByVal vtCellRange As Variant

ByRef vtServerName As Variant

ByRef vtAppName As Variant

ByRef vtCubeName As Variant

ByRef vtFormName As Variant

ByRef vtDimensionNames As Variant

ByRef vtMemberNames As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtCellRange: Range of the cell (one cell only) whose writability must be checked.

pvtServerName: Name of the server the associated connection on the sheet is connected to

pvtApplicationName: Name of the application the associated connection on the sheet is connected to

pvtCubeName: Name of the cube /database (Plan Type in Planning) the associated connection on the sheet is connected to

pvtFormName: Name of the form the associated connection on the sheet is connected to (in ad hoc grids, this is returned as empty string)

pvtDimensionNames: Array of dimension names

pvtMemberNames: Array of member names

Return Value

Returns SS_OK if successful; otherwise, the appropriate error code.

Example

```
Public Declare Function HypGetDimMbrsForDataCell Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtCellRange As Variant, _ ByRef vtServerName As Variant, ByRef vtAppName As Variant, _ ByRef vtCubeName As Variant, ByRef vtFormName As Variant, _ ByRef vtDimensionNames As Variant, ByRef vtMemberNames As Variant) As Long
```

```
Sub TestGetDimMbrsForDataCell()
```

```

Dim oRet As Long
Dim oSheetName As String
Dim oSheetDisp As Worksheet
Dim vtDimNames As Variant
Dim vtMbrNames As Variant
Dim vtServerName As Variant
Dim vtAppName As Variant
Dim vtCubeName As Variant
Dim vtFormName As Variant
Dim lNumDims As Long
Dim lNumMbrs As Long
Dim sPrintMsg As String

oSheetName = Empty
Set oSheetDisp = Worksheets(oSheetName$)
oRet = HypGetDimMbrsForDataCell("", oSheetDisp.Range("B2"), vtServerName, vtAppName,
vtCubeName, vtFormName, vtDimNames, vtMbrNames)

If (oRet = SS_OK) Then
    If IsArray(vtDimNames) Then
        lNumDims = UBound(vtDimNames) - LBound(vtDimNames) + 1
    End If

    If IsArray(vtMbrNames) Then
        lNumMbrs = UBound(vtMbrNames) - LBound(vtMbrNames) + 1
    End If

    sPrintMsg = "Number of Dimensions = " & lNumDims & " Number of Members = " &
lNumMbrs & " Cube Name - " & vtCubeName
    MsgBox (sPrintMsg)
End If

End Sub

```

HypGetGlobalOption

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypGetGlobalOption() returns information about individual Smart View workspace options.

Note: For option descriptions, see [Chapter 13, “Smart View Global Options.”](#)

Syntax

HypGetGlobalOption(vtItem)

ByVal vtItem As Long

Parameters

vtItem: Number that indicates which option is to be retrieved.

Table 15 lists the numbers of options and their return data types.

Table 15 HypGetGlobalOption Parameter Numbers and Options

vtItem	Option	Return Data Type
1	Enable Excel formatting	Boolean
2	Enable double-click for ad hoc operations	Boolean
3	Enable undo	Boolean
4	Not used	Boolean
5	Specify message level setting: 0 Information messages 1 Warning messages 2 Error messages 3 No messages	Number
6	Use thousands separator	Boolean
7	Enable route messages to log file	Boolean
8	Clear log file on next launch	Boolean
9	Enable navigate without data	Boolean
10	Display Member Selection save	Boolean
11	Enable double-clicking to browse LRO	Boolean
12	Specify Meaningless label	Text
13	Reduce Excel file size	Boolean
14	Enable formatted strings	Boolean
15	Retain numeric formatting	Boolean
16	Enable enhanced comment handling	Boolean
17	Enable retain ribbon context	Boolean
18	Display Smart View Panel on start-up	Boolean

Return Value

Returns a number or Boolean value indicating the state of the requested option. Returns an error code if parameter item is out of range.

Example

The following example sets the option for specifying a message level and checks whether the value set is valid.

```
Declare Function HypGetGlobalOption Lib "HsAddin" (ByVal vtItem As Long) As Variant

Sub GetGlobal()
    sts = HypGetGlobalOption(5)
    If sts = -15 then
        MsgBox ("Invalid Parameter")
    Else
        MsgBox ("Message level is set to" & sts)
    End Sub
```

HypGetLinkMacro

Data source types: Essbase, Planning (ad hoc only)Financial Management, (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypGetLinkMacro() returns the macro name currently set to be run to perform the dynamic link query.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#)

Syntax

```
HypGetLinkMacro (vtMacroName)
ByRef vtMacroName As Variant
```

Parameters

vtMacroName: Output. Returns the currently set macro name.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypGetLinkMacro Lib "HsAddin" (ByRef vtMacroName As Variant) As Long

Sub Auto_Open()
    Dim Macroname as Variant
    Sts = HypUseLinkMacro(True)
    Sts = HypSetLinkMacro("Sheet1.Macro8")
```

```

    Sts = HypGetLinkMacro(MacroName)
    If (StrComp(MacroName, "Sheet1.Macro8")) Then
        MsgBox ("Error Occurred")
    End If
End Sub

```

HypGetPagePOVChoices

Data Source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypGetPagePOVChoices returns the available member names and member description for a given dimension.

Syntax

HypGetPagePOVChoices(vtSheetName, vtDimensionName, vtMbrNameChoices, vtMbrDescChoices)

ByVal vtSheetName As Variant

ByVal vtDimensionName As Variant

ByRef vtMbrNameChoices As Variant

ByRef vtMbrNameChoices As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtDimensionName: The dimension names in the POV

vtMbrNameChoices: Output parameter containing array of member names, if successful

vtMbrDescChoices: Output parameter containing array of member descriptions, if successful

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```

Public Declare Function HypGetPagePOVChoices Lib "HsAddin As Long

Sub getpagepovchoices_test()
    Dim mbrName As Variant
    Dim mbrDesc As Variant
    sts = HypGetPagePOVChoices("Sheet7", "Entity", mbrName, mbrDesc)
    MsgBox (sts)
End Sub

```

HypGetParent

Data source types: Essbase

Description

HypGetParent() returns the parent name for the specified member.

Syntax

HypGetParent(vtSheetName, vtMemberName, vtParentName)

ByVal vtSheetName As Variant

ByVal vtMemberName As Variant

ByRef vtParentName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtMemberName: Specify a member name. Required Field.

vtParentName: Output. Contains the parent name on successful execution of the macro.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypGetParent Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtMemberName As Variant, ByRef vtParentName As Variant) As Long
```

```
Sub Sample_HypGetParent
    Dim vtParent as Variant
    X = HypGetParent (Empty, "East", vtParent)
End sub
```

HypGetPOVCount

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypGetPOVCount() returns the number of dimensions in the POV from the dynamic link query.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#)

Syntax

HypGetPOVCount()

Return Value

Returns the number of column dimensions if successful; otherwise, returns the negative error code.

Example

```
Declare Function HypGetPOVCount Lib "HsAddin" () As Long
```

```
Sub Macro()  
    Dim vtGrid as Variant  
    Sts = HypConnect(Empty, "system", "password", "MyDemoBasic")  
    Sts = HypRetrieve(Empty)  
    Range ("B2").Select  
    Sts = HypGetSourceGrid (Empty, vtGrid)  
    Sts = HypGetPOVCount ()  
End sub
```

HypGetPOVItems

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypGetPOVItems() returns the dimensions in the POV and the currently selected member for each dimension.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#)

Syntax

HypGetPOVItems(vtDimensionNames, vtPOVNames)

ByRef vtDimensionNames As Variant

ByRef vtPOVNames As Variant

Parameters

vtDimensionNames: The dimension names in the POV

vtPOVNames: The currently selected member for each dimension in the POV.

Return Value

Returns 0 if successful; otherwise, returns the negative error code.

Example

```
Declare Function HypGetPOVItems Lib "HsAddin" (ByRef vtDimensionNames As Variant, ByRef vtPOVNames As Variant) As Long
```

```
Sub Macro()  
    Dim vtGrid as Variant  
    Dim vtDimNames As Variant  
    Dim vtPOVNames As Variant  
    Sts = HypConnect(Empty, "system", "password", "MyDemoBasic")  
    Sts = HypRetrieve(Empty)  
    Range ("B2").Select  
    Sts = HypGetSourceGrid (Empty, vtGrid)  
    Sts = HypGetPOVItems (vtDimNames, vtPOVNames)  
End sub
```

HypGetRowCount

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypGetRowCount() returns the number of row dimensions.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#)

Syntax

HypGetRowCount()

Return Value

Returns number of row dimensions if successful; otherwise, returns the negative error code.

Example

```
Declare Function HypGetRowCount Lib "HsAddin" () As Long
```

```
Sub Macro()  
    Dim vtGrid as Variant  
    Sts = HypConnect(Empty, "system", "password", "MyDemoBasic")
```

```

    Sts = HypRetrieve(Empty)
    Range ("B2").Select
    Sts = HypGetSourceGrid (Empty, vtGrid)
    Sts = HypGetRowCount ()
End sub

```

HypGetRowItems

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypGetRowItems() returns the members present for the nth row dimension in the dynamic link query.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#)

Syntax

HypGetRowItems(vtRowID, vtDimensionName, vtMemberNames)

ByVal vtRowID As Variant

ByRef vtDimensionName As Variant

ByRef vtMemberNames As Variant

Parameters

vtRowID: The row number n.

vtDimensionName: Returns the nth row dimension name.

vtMemberNames: Returns the members for the nth row dimensions.

Return Value

Returns 0 if successful; otherwise, returns the negative error code.

Example

```

Declare Function HypGetRowItems Lib "HsAddin" (ByVal rowID As Variant, ByRef
vtDimensionName As Variant, ByRef vtMemberNames As Variant) As Long

```

```

Sub Macro()
    Dim vtGrid as Variant
    Dim vtDimName as Variant
    Dim vtMembers as Variant
    Sts = HypConnect(Empty, "system", "password", "AnamikaDemoBasic")

```

```

    Sts = HypRetrieve(Empty)
    Range ("B2").Select
    Sts = HypGetSourceGrid (Empty, vtGrid)
    Sts = HypGetRowItems(1, vtDimName, vtMembers)
End sub

```

HypGetSharedConnectionsURL

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypGetSharedConnectionsURL() returns the Shared Connections URL to be used. (also shown in the Options dialog box).

Syntax

HypGetSharedConnectionsURL (vtSharedConnURL As Variant)

ByRef vtSharedConnURL As Variant

Parameters

vtSharedConnURL: the output parameter that contains the Oracle's Hyperion® Shared Services URL, if successful.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```

Declare Function HypGetSharedConnectionsURL Lib "HsAddin" (ByRef vtSharedConnURL As
Variant) As Long
Sub SubHypGetURL()
Dim lRet As Long
Dim conn As Variant
lRet = HypGetSharedConnectionsURL(conn)
MsgBox (lRet)
MsgBox (conn)
End Sub

```

HypGetSheetOption

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypGetSheetOption() returns information about individual spreadsheet options.

Syntax

HypGetSheetOption(vtSheetName, vtItem)

ByVal vtSheetName As Variant ByVal vtItem As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtItem: Number indicating which option is to be retrieved. See [Table 16](#) for a list of values.

Table 16 Options for vtItem

vtItem	Option	Data Type and Values
1	Set zoom in level: 0 Next level 1 All levels 2 Bottom level	Number
2	Enable Include Selection setting	Boolean
3	Enable Within Selection Group setting	Boolean
4	Enable Remove Unselected Groups setting	Boolean
5	Specify Indent setting: 0 No indentation 1 Indent sub items 2 Indent totals	Number
6	Enable suppress missing setting	Boolean
7	Enable suppress zeros setting	Boolean
8	Enable suppress underscores setting	Boolean
9	Enable No Access setting	Boolean
10	Enable Repeated Member setting	Boolean
11	Enable Invalid setting	Boolean
12	Ancestor Position: 0 Top 1 Bottom	Number
13	Specify Missing Text label	Text
14	Specify No Access label	Text

vtItem	Option	Data Type and Values
15	Cell Status: 0 Data 1 Calculation Status 2 Process Management	Number
16	Member Name Display options: 0 Name Only 1 Name and Description 2 Description only	Number

Return Value

Returns the value of the current setting as a string, number, or Boolean. Returns an error code if parameter item is out of range.

Example

```
Declare Function HypGetSheetOption Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtItem As Variant) As Variant
```

```
Sub GetSheet()
sts = HypGetSheetOption("Sheet", 5)
If sts = -15 then
Msgbox ("Invalid Parameter")
Else
Msgbox ("Indentation is set to" & sts)
End Sub
```

HypGetSourceGrid

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypGetSourceGrid() creates a query from the source grid for the dynamic link query.

This function applies to both static and dynamic link views.

Note: A cell in the grid must be selected before this making this call.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#)

Syntax

HypGetSourceGrid(vtSheetName, vtGrid)

ByVal vtSheetName As Variant

ByRef vtGrid As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtGrid: The grid XML is returned on successful execution.

Return Value

Returns 0 if successful or the appropriate error code otherwise.

Example

```
Declare Function HypGetSourceGrid Lib "HsAddin" (ByVal vtSheetName As Variant, ByRef  
vtGrid As Variant) As Long
```

```
Sub Macro()  
    Dim vtGrid as Variant  
    Range ("B2").Select  
    Sts = HypGetSourceGrid (Empty, vtGrid)  
End sub
```

HypGetSubstitutionVariable

Data source types: Essbase

Description

HypGetSubstitutionVariable() retrieves substitution variables and their current value from Essbase.

Syntax

HypGetSubstitutionVariable (vtSheetName, vtApplicationName, vtDatabaseName,
vtVariableNameList, vtVariableValueList)

ByVal vtSheetName As Variant

ByVal vtApplicationName As Variant

ByVal vtDatabaseName As Variant

ByVal vtVariableName As Variant

ByRef vtVariableNameList As Variant

ByRef vtVariableValueList As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtApplicationName: The application name to return variables scoped for the specified application. If vtApplicationName is Null or Empty all the applications are considered.

vtDatabaseName: The database name to return variables scoped for the specified database. If vtDatabaseName is Null or Empty all the databases are considered.

vtVariableName: The variable name to be retrieved. If vtVariableName is Null or Empty the entire list of variables is returned.

vtVariableNameList: Output Result Vector that contains the list of the variable names. Its contents are unknown if the macro fails.

vtVariableValueList: Output Result Vector that contains the list of the variable values corresponding to each variable returned. Its contents are unknown if the macro fails.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypGetSubstitutionVariable Lib "HsAddin" (ByVal vtSheetName As Variant,  
ByVal vtApplicationName As Variant, ByVal vtDatabaseName As Variant, ByVal  
vtVariableName As Variant, ByRef vtVariableNames As Variant, ByRef vtVariableValues As  
Variant) As Long
```

```
Sub Sample_HypGetSubstitutionVariable  
    Dim vtVarNameList as Variant  
    Dim vtVarValueList as Variant  
    Dim vtVarVal as Variant  
    Dim vtVar as Variant  
    X = HypGetSubstitutionVariable (Empty, "Sample", "Basic",  
    Empty, vtVarNameList, vtVarValueList)  
    If IsArray (vtVarNameList) Then  
        For i = LBound (vtVarNameList) To UBound (vtVarNameList)  
            vtVar = vtVarNameList (i)  
            Next  
        End If  
    If IsArray (vtVarValueList) Then  
        For i = LBound (vtVarValueList) To UBound (vtVarValueList)  
            vtVarVal = vtVarValueList (i)  
            Next  
        End If  
    End Sub
```

HypInvalidateSSO

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypInvalidateSSO() discards the existing SSO token.

Example

```
Declare Function HypInvalidateSSO Lib "HsAddin" () As Long
    Sub HypInvalidateSSO()
End Sub
```

HypIsAncestor

Data source types: Essbase

Description

HypIsAncestor() checks whether the specified member is the ancestor of another specified member.

Syntax

HypIsAncestor(vtSheetName, vtMemberName, vtAncestorName)

ByVal vtSheetName As Variant

ByVal vtMemberName As Variant

ByVal vtAncestorName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtMemberName: A member name. Required.

vtAncestorName: The member name of the ancestor. Required.

Return Value

Returns a variant in which -1 is true, 0 is false; otherwise, returns the appropriate error code.

Example

```
Declare Function HypIsAncestor Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtMemberName As Variant, ByVal vtAncestorName As Variant) As Variant
```



```

Sub Sample_HypIsAncestor
    Dim b as Variant
    b = HypIsAncestor (Empty, "Year", "Jan")
End sub

```

HypIsAttribute

Data source types: Essbase

Description

HypIsAttribute() checks to see if the specified member has a specific attribute.

Syntax

HypIsAttribute(vtSheetName, vtDimensionName, vtMemberName, vtAttributeName)

ByVal vtSheetName As Variant

ByVal vtDimensionName As Variant

ByVal vtMemberName As Variant

ByVal vtAttributeName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtDimensionName: The name of the dimension where the member belongs.

vtMemberName: The name of the member for which we must test the condition.

vtAttributeName: Input string that is compared against the attributes of the member.

Return Value

Returns a variant in which -1 is true, 0 is false; otherwise, returns the appropriate error code.

Example

```

Declare Function HypIsAttribute Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal
vtDimensionName As Variant, ByVal vtMemberName As Variant, ByVal vtAttribute As Variant)
As Variant

```

```

Sub CheckAttribute()
    vtret = HypIsAttribute(Empty, "Market", "Connecticut", "MyAttribute")
    If vtret = -1 Then
        MsgBox ("Found MyAttribute")
    ElseIf vtret = 0 Then
        MsgBox ("MyAttribute not available for Connecticut")
    Else
        MsgBox ("Error value returned is" & vtret)
    End If
End Sub

```

```
End If
End Sub
```

HypIsCellWritable

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypIsCellWritable() checks to see whether a cell is writable.

Syntax

HypIsCellWritable (vtSheetName [in], vtCellRange [out])

ByVal vtSheetName As Variant

ByVal vtCellRange As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtCellRange: Range of the cell (one cell only) whose writability must be checked.

Return Value

VARIANT_TRUE if the cell is writable, otherwise VARIANT_FALSE.

Example

```
Public Declare Function HypIsCellWritable Lib "HsAddin" (ByVal vtSheetName As Variant,
ByVal vtCellRange As Variant) As Boolean
```

```
Sub TestIsCellWritable()
```

```
    Dim oRet As Boolean
```

```
    Dim oSheetName As String
```

```
    Dim oSheetDisp As Worksheet
```

```
    oSheetName = Empty
```

```
    Set oSheetDisp = Worksheets(oSheetName$)
```

```
    oRet = HypIsCellWritable (Empty, oSheetDisp.Range("G2"))
```

```
End Sub
```

HypIsChild

Data source types: Essbase

HypIsChild() checks whether the specified child member is the child of a specified parent member. HypIsChild checks only for children, not for all descendants.

Syntax

HypIsChild(vtSheetName, vtParentName, vtChildName)

ByVal vtSheetName As Variant

ByVal vtParentName As Variant

ByVal vtChildName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtParentName: The member name of the parent. Required.

vtChildName: The member name of the child. Required.

Return Value

Returns a variant in which -1 is true, 0 is false; otherwise, returns the appropriate error code.

Example

```
Declare Function HypIsChild Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal  
vtParentName As Variant, ByVal ParentName As Variant) As Variant
```

```
Sub Sample_HypIsChild  
    Dim b as Boolean  
    b = HypIsChild ("Sheet1", "Year", "Qtr1")  
End Sub
```

HypIsConnectedToSharedConnections

Data Source Types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypIsConnectedToSharedConnections() checks whether SmartView is connected to Shared Connections.

Syntax

HypIsConnectedToSharedConnections ()

Return Value

Return: true if Smart View is connected to Shared Connections, otherwise false.

Example

```
Declare Function HypIsConnectedToSharedConnections Lib "HsAddin" () As Variant
Sub SubIsSharedConnectedTest()
Dim vtRet As Variant
vtRet = HypIsConnectedToSharedConnections ()
MsgBox(vtRet)
End Sub
```

HypIsDataModified

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypIsDataModified() checks to see whether any data cells have been modified but not yet submitted.

Syntax

HypIsDataModified (vtSheetName [in])

By Val vtSheetName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

Return Value

VARIANT_TRUE if the sheet contains any data cells that have been updated and not yet submitted, otherwise VARIANT_FALSE.

Example

```
Public Declare Function HypIsDataModified Lib "HsAddin" (ByVal vtSheetName As Variant)
As Boolean

Sub TestIsSheetDirty()

    Dim oRet As Boolean

    oRet = HypIsDataModified(Empty)
    MsgBox (oRet)
```

End Sub

HypIsDescendant

Data source types: Essbase

Description

HypIsDescendant() checks if the specified member is the descendant of another specified member.

Syntax

HypIsDescendant(vtSheetName, vtMemberName, vtAncestorName)

ByVal vtSheetName As Variant

ByVal vtMemberName As Variant

ByVal vtAncestorName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtMemberName: A member name. Required.

vtAncestorName: The member name of the ancestor. Required.

Return Value

Returns a variant in which -1 is true, 0 is false; otherwise, returns the appropriate error code.

Example

```
Declare Function HypIsDescendant Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal  
vtMemberName As Variant, ByVal vtDescendantName As Variant) As Boolean
```

```
Sub Sample_HypIsDescendant  
    Dim b as Boolean  
    b = HypIsDescendant (Empty, "Year", "Jan")  
End sub
```

HypIsExpense

Data source types: Essbase

Description

HypIsExpense() verifies that the member specified has an Expense tag.

Syntax

HypIsExpense(vtSheetName, vtDimensionName, vtMemberName)

ByVal vtSheetName As Variant

ByVal vtDimensionName As Variant

ByVal vtMemberName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtDimensionName: The name of the dimension where the member belongs. If vtDimensionName is Null or Empty, the active dimension is used.

vtMemberName: The name of the specified member.

Return Value

Returns a variant in which -1 is true, 0 is false; otherwise, returns the appropriate error code.

Example

```
Declare Function HypIsExpense Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtDimensionName As Variant, ByVal vtMemberName As Variant) As Variant
```

```
Sub CheckExpense()  
vtret = HypIsExpense(Empty, "Measures", "Opening Inventory")  
    If vtret = -1 Then  
        MsgBox ("Opening Inventory has expense flag set")  
    ElseIf vtret = 0 Then  
        MsgBox ("Expense flag has not been set")  
    Else  
        MsgBox ("Error value returned is" & vtret)  
    End If  
End Sub
```

HypIsFreeForm

Data Sources: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypIsFreeForm() checks to see whether the worksheet is in free-form mode.

Syntax

HypIsFreeForm (vtSheetName [in])

By Val vtSheetName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

Return Value

VARIANT_TRUE if the cell is in free-form state, i.e., either member cells or comment cells have been modified and the sheet has not been refreshed, otherwise VARIANT_FALSE.

Example

```
Public Declare Function HypIsFreeForm Lib "HsAddin" (ByVal vtSheetName As Variant) As Boolean
Sub TestIsSheetFreeForm()

Sub HypIsFreeForm()
    Dim oRet As Boolean

    oRet = HypIsFreeForm(Empty)
    MsgBox (oRet)

End Sub
```

HypIsParent

Data source types: Essbase

HypIsParent() checks whether the specified member is the parent of another specified member.

Syntax

HypIsParent(vtSheetName, vtMemberName, vtParentName)

ByVal vtSheetName As Variant

ByVal vtMemberName As Variant

ByVal vtParentName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtMemberName: A member name. Required.

vtParentName: The member name of the parent. Required.

Return Value

Returns a variant in which -1 is true, 0 is false; otherwise, returns the appropriate error code.

Example

```
Declare Function HypIsParent Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtMemberName As Variant, ByVal ParentName As Variant) As Boolean
```

```
Sub Sample_HypIsParent
    Dim b as Boolean
    b = HypIsParent (Empty, "East", "Market")
End Sub
```

HypIsSmartViewContentPresent

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypIsSmartViewContentPresent() checks to see whether the sheet contains Smart View content.

Syntax

HypIsSmartViewContentPresent(vtSheetName [in], pContentType [out])

ByVal vtSheetName As Variant

ByRef vtTypeOfContentsInSheet

Parameters

vtSheetName: For future use. Currently the active sheet is used.

pContentType: Function returns appropriate type of content on the sheet. Possible values are in the enum as defined below.

```
Enum TYPE_OF_CONTENTS_IN_SHEET
    EMPTY_SHEET
    ADHOC_SHEET
    FORM_SHEET
    INTERACTIVE_REPORT_SHEET
End Enum
```

Return Value

VARIANT_TRUE if the worksheet contains Smart View content, otherwise VARIANT_FALSE.

Example

```
Public Declare Function HypIsSmartViewContentPresent Lib "HsAddin" (ByVal vtSheetName As
```



```

Variant, _
ByRef vtTypeOfContentsInSheet As TYPE_OF_CONTENTS_IN_SHEET) As Boolean

Sub TestIsSVCCContentOnSheet()

    Dim oRet As Boolean
    Dim oContentType As TYPE_OF_CONTENTS_IN_SHEET
    Dim oSheetName As String
    Dim oSheetDisp As Worksheet

    oSheetName = Empty
    Set oSheetDisp = Worksheets(Empty)
    oRet = HypIsSmartViewContentPresent (Empty, oContentType)

End Sub

```

HypIsUDA

Data source types: Essbase

Description

HypIsUDA() checks to verify if the member specified has a specific UDA.

Syntax

HypIsUDA (vtSheetName, vtDimensionName, vtMemberName, vtUDAString)

ByVal vtSheetName As Variant

ByVal vtDimensionName As Variant

ByVal vtMemberName As Variant

ByVal vtUDAString As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtDimensionName: The name of the dimension where the member belongs.

vtMemberName: The name of the member for which we must test the condition.

vtUDAString: Input string that is compared against the attributes of the member.

Return Value

Returns a variant in which -1 is true, 0 is false; otherwise, returns the appropriate error code.

Example

```
Declare Function HypIsUDA Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal  
vtDimensionName As Variant, ByVal vtMemberName As Variant, ByVal vtUDAString As Variant)  
As Variant
```

```
Sub CheckUDA()  
vtret = HypIsUDA(Empty, "Market", "Connecticut", "MyUDA")  
    If vtret = -1 Then  
        MsgBox ("Found MyUDA")  
    ElseIf vtret = 0 Then  
        MsgBox ("Did not find MyUDA")  
    Else  
        MsgBox ("Error value returned is" & vtret)  
    End If  
End Sub
```

HypKeepOnly

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypKeepOnly() retains only the selected member(s) in the sheet and removes unselected members.

The selection must be limited to member cells only, not data cells.

Syntax

HypKeepOnly(vtSheetName, vtSelection)

ByVal vtSheetName As Variant

ByVal vtSelection As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtSelection: Range object which refers to the member(s) that will be kept. If selection is Null or Empty, the active cell is used.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypKeepOnly Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal  
vtSelection As Variant) As Long
```

```

Sub KOnly()
' Keep Only on one member name
  X=HypKeepOnly(Empty, RANGE("D2"))
  If X = 0 Then
    MsgBox("Keep Only successful.")
  Else
    MsgBox("Keep Only failed." + X)
  End If
' Keep Only on two member names
  X=HypKeepOnly(Empty, RANGE("D2:A5"))
  If X = 0 Then
    MsgBox("Keep Only successful.")
  Else
    MsgBox("Keep Only failed." + X)
  End If
End Sub

```

HypListCalcScripts

Data source types: Essbase

Description

HypListCalcScripts() lists all calculation scripts present on Analytic Server.

Syntax

HypListCalcScripts (vtSheetName, vtScriptArray)

ByVal vtSheetName As Variant

ByRef vtScriptArray As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtScriptArray: The business rule scripts are returned in this array.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```

Declare Function HypListCalcScripts Lib "HsAddin" (ByVal sheetName As Variant,ByRef
scriptArray) As Long

```

```

Dim sts As Long
Dim paramList As Variant
sts=HypListCalcScripts (Empty,paramList)

```

```

If IsArray(paramList) Then
    cbItems = UBound(paramList) - LBound(paramList) + 1
    MsgBox ("Number of elements = " + Str(cbItems))
    For i = LBound(paramList) To UBound(paramList)
        MsgBox ("Member = " + paramList(i))
    Next
Else
    MsgBox ("Return Value = " + sts)
End If

```

HypListCalcScriptsEx

Data Sources: Essbase, Planning

Description

HypListCalcScriptsEx() lists all business rules.

Note: See **Usage** under **HypExecuteCalcScriptsEx** for more information and example.

Syntax

HypListCalcScriptsEx (vtSheetName [in], vtbRuleOnForm [in], pvtArCubeNames [out],
pvtArBRNames [out], pvtArBRTypes [out], pvtArBRHasPrompts [out],
pvtArBRNeedPageInfo [out], pvtArBRHidePrompts [out])

ByVal vtSheetName As Variant

ByVal vtbRuleOnForm As Variant

ByRef vtCubeNames As Variant

ByRef vtBRNames As Variant

ByRef vtBRTypes As Variant

ByRef vtBRHasPrompts As Variant

ByRef vtBRNeedsPageInfo As Variant

ByRef vtBRHidePrompts As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtbRuleOnForm: Boolean to indicate whether the user wants to list business rules associated only with the form opened on the sheet. If this argument is false, all the business rules associated with the application will be returned.

pvtArCubeNames: Array of cube names (Plan Types in Planning) associated with the Business rules

pvtArBRNames: Array of Business Rule Names

pvtArBRTypes: Array of Business Rule Types

pvtArBRHasPrompts: – Array of booleans indicating whether the Business Rule has Run Time Prompts

pvtArBRNeedPageInfo: Array of booleans indicating whether the Business Rule needs Page Information on the sheet to be run

pvtArBRHidePrompts: Array of booleans indicating whether the RTPs for this Business Rule are hidden

Return Value

Returns SS_OK if successful; otherwise, the appropriate error code.

HypOpenForm

Data source types: Planning, Financial Management, Hyperion Enterprise

Description

HypOpenForm opens the form.

Syntax

HypOpenForm (vtSheetName, vtFolderPath, vtFormName, vtDimensionList(),
vtMemberList())

ByVal vtSheetName As Variant

ByVal vtFolderPath As Variant

ByVal vtFormName As Variant

ByRef vtDimensionList() As Variant

ByRef vtMemberList() As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtFolderPath: Folder path name

vtFormName: Name of the data form

vtDimensionList(): not in use

vtMemberList(): not in use

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypOpenForm Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtFolderPath As Variant, ByVal vtFormName As Variant, ByRef vtDimensionList() As Variant, ByRef vtMemberList() As Variant) As Long

Sub getpagepovchoices_test()
    Dim DimList As Variant
    Dim MemList As Variant
    sts = HypOpenForm(Empty, "/Forms/data1", "data1", DimList, MemList)
    MsgBox (sts)
End Sub
```

HypOtlGetMemberInfo

Data source types: Essbase

Description

HypOtlGetMemberInfo() returns the following information related to a member selection: member comment, formula, UDA, attribute, etcetera.

Syntax

HypOtlGetMemberInfo (vtSheetName, vtDimensionName, vtMemberName, vtPredicate, vtMemberArray)

ByVal vtSheetName As Variant

ByVal vtDimensionName As Variant

ByVal vtMemberName As Variant

ByVal vtPredicate As Variant

ByRef vtMemberArray As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtDimensionName: The name of the dimension. Can be Null; if Null, search for the predicate in the whole outline. Dimension to limit the scope of the query.

vtMemberName: Member name for which information is being queried on.

vtPredicate: Member selection criteria:

1 HYP_COMMENT

2 HYP_FORMULA

3 HYP_UDA

4 HYP_ATTRIBUTE

vtMemberArray: Output that contains the result of the query. Its contents are unknown if the macro fails.

Return Value

Returns 0 if successful; otherwise returns the appropriate error code.

Example

```
Declare Function HypOtlGetMemberInfo Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtMemberName As Variant, ByVal vtPredicate As Variant, ByRef vtMemberArray As Variant) As Long
```

```
Sub HypOtlGetMemberInfo()  
    vtRet = HypOtlGetMemberInfo (Empty, "Year", "Jan",  
        HYP_COMMENT, vt)  
If IsArray(vt) Then cbItems = UBound(vt) + 1  
    MsgBox ("Number of elements = " + Str(cbItems))  
For i = 0 To UBound(vt)  
    MsgBox ("Member = " + vt(i))  
Next  
Else  
MsgBox ("Return Value = " + vtRet)  
End If  
End Sub
```

HypPivot

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypPivot() transposes spreadsheet rows and columns, based on the selected dimension.

Syntax

HypPivot(vtSheetName, vtStart, vtEnd)

ByVal vtSheetName As Variant

ByVal vtStart As Variant

ByVal vtEnd As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtStart: Range object which refers to the single cell starting point of the pivot.

vtEnd: Range object which refers to the single cell ending point of the pivot

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

Declare Function HypPivot Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtStart As Variant, ByVal vtEnd As Variant) As Long

```
Sub DoPivot()  
X=HypPivot(Empty, RANGE("B2"), RANGE("D1"))  
If X = 0 Then  
    MsgBox("Pivot successful.")  
Else  
    MsgBox("Pivot failed.")  
End If  
End Sub
```

HypPivotToGrid

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypPivotToGrid() moves the selected dimension and members from the POV to the spreadsheet grid.

Syntax

HypPivotToGrid (vtSheetName, vtDimensionName, vtSelection)

ByVal vtSheetName as Variant

ByVal vtDimensionName as Variant

ByVal vtSelection as Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtDimensionName: Currently selected dimension from the toolbar.

vtSelection: Range object which refers to the single cell starting point of the pivot. Orientation is calculated based on the selection.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypPivotToGrid Lib "HsAddin" (By Val vtSheetName As Variant, ByVal vtDimensionName as Variant, ByVal vtSelection as Variant) As Long
```

```
Sub DoPivotGrid()  
X=HypPivotToGrid(Empty, "Product", RANGE("E6"))  
If X = 0 Then  
    MsgBox("Pivot to grid successful.")  
Else  
    MsgBox("Pivot to grid failed.")  
End If  
End Sub
```

HypPivotToPOV

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypPivotToPOV() pivots from the grid to the POV.

Syntax

HypPivotToPOV (vtSheetName, vtSelection)

ByVal vtSheetName as Variant

ByVal vtSelection as Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtSelection: Range object which refers to the single cell starting point of the pivot. Orientation is calculated based on the selection.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypPivotToPOV Lib "HsAddin" (By Val vtSheetName As Variant, ByVal vtSelection as Variant) As Long
```

```
Sub DoPivotPOV()  
X=HypPivotToPOV(Empty, RANGE("E6"))  
If X = 0 Then  
    MsgBox("Pivot to POV successful.")  
Else
```

```
MsgBox("Pivot to POV failed.")
End If
End Sub
```

HypPreserveFormatting

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypPreserveFormatting() applies grid formatting to cells created by zooming in.

Syntax

HypHypPreserveFormatting (vtSheetName [in], vtRange [in])

ByVal vtSheetName As Variant

ByVal vtSelectionRange As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtRange: Range of the cell(s) for which formatting needs to be preserved. (Multiple ranges are supported)

Return Value

Returns SS_OK if successful; otherwise, the appropriate error code.

Example

```
Public Declare HypPreserveFormatting Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtSelectionRange As Variant) As Long
```

```
Sub TestPreserveFormatting()

    Dim oRet As Long
    Dim oSheetName As String
    Dim oSheetDisp As Worksheet

    oSheetName = Empty
    Set oSheetDisp = Worksheets(oSheetName$)
    oRet = HypPreserveFormatting (" ", oSheetDisp.Range("B2"))

    MsgBox (oRet)

End Sub
```

HypQueryMembers

Data source types: Essbase

Description

HypQueryMembers() executes the member selection query.

Syntax

HypQueryMembers (vtSheetName, vtMemberName, vtPredicate, vtOption, vtDimensionName, vtInput1, vtInput2, vtMemberArray)

ByVal vtSheetName As Variant

ByVal vtMemberName As Variant

ByVal vtPredicate As Variant

ByVal vtOption As Variant

ByVal vtDimensionName As Variant

ByVal vtInput1 As Variant

ByVal vtInput2 As Variant

ByRef vtMemberArray As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtMemberName: (string) The member name on which to perform the query.

vtPredicate: (integer) Member selection criteria:

1 HYP_CHILDREN

2 HYP_DESCENDANTS

3 HYP_BOTTOMLEVEL

4 HYP_SIBLINGS

5 HYP_SAMELEVEL

6 HYP_SAMEGENERATION

7 HYP_PARENT

8 HYP_DIMENSION

9 HYP_NAMEDGENERATION

10 HYP_NAMEDLEVEL

11 HYP_SEARCH

12 HYP_WILDSEARCH

13 HYP_USERATTRIBUTE

14 HYP_ANCESTORS

15 HYP_DTSMEMBER

vtOption: (integer) Options are dependent on the predicate:

For the predicate values, HYP_SEARCH and HYP_WILDSEARCH, specify query options:

HYP_MEMBERSONLY

HYP_ALIASESONLY

HYP_MEMBERSANDALIASES

vtDimensionName: (string) Dimension to limit the scope of the query. It is used with the following query options and ignored otherwise: HYP_NAMEDGENERATION, HYP_NAMEDLEVEL, HYP_USERATTRIBUTE HYP_SEARCH (set to Null to search through all dimensions), HYP_WILDSEARCH (set to Null to search through all dimensions).

vtInput1: (string) Input string that is determined by the option. It is used with the following query options and ignored otherwise:

- HYP_NAMEDGENERATION (The name of the generation)
- HYP_NAMEDLEVEL (The name of the level)
- HYP_SEARCH (The string to search for. The string is defined as an exact)
- HYP_WILDSEARCH (The string to search for. The string is defined as an exact search string with an optional '*' at the end to mean any set of characters)
- HYP_USERATTRIBUTE (The user-defined attribute)

vtInput2: (string) Input string that is determined by the option. It is used with the following query options and ignored otherwise:

- HYP_USERATTRIBUTE (The user-defined attribute)
- HYP_SEARCH, HYP_WILDSEARCH (If the options are set to search in the alias tables, this string specifies which alias table to search. If the string is Null, all alias tables will be searched).

vtMemberArray: Output that contains the result of the query. If unsuccessful, its contents are unknown.

Return Value

Returns a zero if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypQueryMembers Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal  
vtMemberName As Variant, ByVal vtPredicate As Variant, ByVal vtOption As Variant, ByVal  
vtDimensionName As Variant, ByVal vtInput1 As Variant, ByVal vtInput2 As Variant, ByRef  
vtMemberArray As Variant) As Long
```

```
Sub QueryMembersEmptyValues()
```

```

' sts = HypQueryMembers(Empty, "Profit", HYP_CHILDREN, Empty, Empty, Empty, Empty,
vArray)
' sts = HypQueryMembers(Empty, "Profit", HYP_DESCENDANTS, Empty, Empty, Empty, Empty,
vArray)
' sts = HypQueryMembers(Empty, "Profit", HYP_BOTTOMLEVEL, Empty, Empty, Empty, Empty,
vArray)
' sts = HypQueryMembers(Empty, "Sales", HYP_SIBLINGS, Empty, Empty, Empty, Empty,
vArray)
' sts = HypQueryMembers(Empty, "Sales", HYP_SAMELEVEL, Empty, Empty, Empty, Empty,
vArray)
' sts = HypQueryMembers(Empty, "Sales", HYP_SAMEGENERATION, Empty, Empty, Empty, Empty,
vArray)
' sts = HypQueryMembers(Empty, "Sales", HYP_PARENT, Empty, Empty, Empty, Empty, vArray)
' sts = HypQueryMembers(Empty, "Sales", HYP_DIMENSION, Empty, Empty, Empty, Empty,
vArray)
' sts = HypQueryMembers(Empty, "Year", HYP_NAMEDGENERATION, Empty, "Year", "Quarter",
Empty, vArray)
' sts = HypQueryMembers(Empty, "Product", HYP_NAMEDLEVEL, Empty, "Product", "SKU",
Empty, vArray)
' sts = HypQueryMembers(Empty, "Product", HYP_SEARCH, HYP_ALIASESONLY, "Product",
"Cola", Empty, vArray)
' sts = HypQueryMembers(Empty, "Year", HYP_WILDSEARCH, HYP_MEMBERSONLY, "Year", "J*",
Empty, vArray)
' sts = HypQueryMembers(Empty, "Market", HYP_USERATTRIBUTE, Empty, "Market", "Major
Market", Empty, vArray)
' sts = HypQueryMembers(Empty, "Sales", HYP_ANCESTORS, Empty, Empty, Empty, Empty,
vArray)
' sts = HypQueryMembers(Empty, "Jan", HYP_DTSMEMBER, Empty, Empty, Empty, Empty, vArray)
' sts = HypQueryMembers(Empty, "Product", Empty, Empty, Empty, Empty, vArray)

If IsArray(vt) Then
    cbItems = UBound(vt) + 1
    MsgBox ("Number of elements = " + Str(cbItems))
    For i = 0 To UBound(vt)
        MsgBox ("Member = " + vt(i))
    Next
Else
    MsgBox ("Return Value = " + Str(vt))
End If
End Sub

```

HypRedo

Data source types: Essbase, Planning (ad hoc only) Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypRedo() restores the database view as it was before an Undo was performed.

Syntax

HypRedo (vtSheetName)

ByVal vtSheetName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code..

Example

```
Declare Function HypRedo Lib "HsAddin" (ByVal vtSheetName As Variant) As Long

Sub Redo()
    X=HypRedo(Empty)
End Sub
```

HypRemovePreservedFormats

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypRemovePreservedFormats() removes preserved formats.

Note: Users must refresh before the original formatting is applied.

Syntax

HypRemovePreservedFormats (vtSheetName [in], vtRemoveAllPreservedFormats
[in],vtSelectionRange [in])

ByVal vtSheetName As Variant

ByVal vtRemoveAllPreservedFormats As Variant

ByVal vtSelectionRange As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtRemoveAllPreservedFormats: Boolean to indicate whether all preserved formats on the grid should be deleted. (If this parameter is true, the next parameter value is not used, so users can pass NULL for vtSelectionRange.)

vtSelectionRange: Range of the cell(s) for which formatting needs to be preserved. (Multiple ranges are supported)

Return Value

Returns SS_OK if successful; otherwise, the appropriate error code.

Example

```
Public Declare Function HypRemovePreservedFormats Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtbRemoveAllPreservedFormats As Variant, ByVal vtSelectionRange As Variant) As Long
```

```
Sub TestRemovePreservedFormatting()
```

```
    Dim oRet As Long
    Dim oSheetName As String
    Dim oSheetDisp As Worksheet
```

```
    oSheetName = "Sheet1"
```

```
    Set oSheetDisp = Worksheets(oSheetName)
    'oRet = HypRemovePreservedFormats(Empty, False, oSheetDisp.Range("B2"))
    oRet = HypRemovePreservedFormats(Empty, True, Null)
    MsgBox (oRet)
```

```
End Sub
```

HypRemoveConnection

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypRemoveConnection() removes the specified connection from the list of all available Smart View connections in the Smart View Panel.

Syntax

HypRemoveConnection(vtFriendlyName)

ByVal vtFriendlyName As Variant

Parameters

vtFriendlyName: The friendly connection name for the data source provider

Return Value

Returns 0 if successful, otherwise, returns the appropriate error code.

Example

```
Declare Function HypRemoveConnection Lib "HsAddin" (ByVal vtFriendlyName As Variant) As Long

Sub RConn()
    X=HypRemoveConnection("My Connection")
End Sub
```

HypRemoveOnly

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypRemoveOnly() removes only the selected member(s) in the sheet and retains unselected members in the selected dimension.

Selection should include only member cells, not data cells.

Syntax

HypRemoveOnly(vtSheetName, vtSelection)

ByVal vtSheetName As Variant

ByVal vtSelection As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtSelection: Range object which refers to the member(s) that will be removed. If selection is Null or Empty, the active cell is used.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypRemoveOnly Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtSelection As Variant) As Long

Sub ROnly()
    ' Remove Only on one member name
    X=HypRemoveOnly(Empty, RANGE("D2"))
    If X = 0 Then
        MsgBox("Remove Only successful.")
    Else
        MsgBox("Remove Only failed." + X)
    End If
End Sub
```



```

End If
' Remove Only on two member names
X=HypRemoveOnly(Empty, RANGE("D2, A5"))
If X = 0 Then
    MsgBox("Remove Only successful.")
Else
    MsgBox("Remove Only failed." + X)
End If
End Sub

```

HypResetFriendlyName

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypResetFriendlyName resets the friendly name to the new friendly name if the new name does not exist. To modify friendly name of a connection in the Smart View Panel, Smart View must be connected to Oracle Hyperion Provider Services.

Syntax

HypResetFriendlyName (vtOldFriendlyName, vtNewFriendlyName)

By Val vtOldFriendlyName as Variant

By Val vtNewFriendlyName as Variant

Parameters

vtOldFriendlyName: The old friendly connection name.

vtNewFriendlyName: The new friendly connection name.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```

Declare Function HypResetFriendlyName Lib "HsAddin" (ByVal vtOldFriendlyName As Variant,
ByVal vtNewFriendlyName As Variant) As Long

```

```

Sub SubHypResetFriendlyNameTest()
    Dim lRet As Long
    lRet = HypResetFriendlyName("server2_Sample_Basic", "My Sample Basic")
End Sub

```

HypRetrieve

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypRetrieve() retrieves data from the database.

Syntax

HypRetrieve(vtSheetName)

ByVal vtSheetName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypRetrieve Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtRange As Variant, ByVal vtLock As Variant) As Long
```

```
Sub RetData()  
X=HypRetrieve(Empty)  
If X = 0 Then  
    MsgBox("Retrieve successful.")  
Else  
    MsgBox("Retrieve failed.")  
End If  
End Sub
```

HypRetrieveAllWorkbooks

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypRetrieveAllWorkbooks() refreshes all open workbooks from the same instance of Excel.

Syntax

HypRetrieveAllWorkbooks()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypRetrieveAllWorkbooks Lib () As Long

Sub Sample_HypRetrieveAllWorkbooks()
    X=HypRetrieveAllWorkbooks()
End Sub
```

HypRetrieveRange

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypRetrieveRange() gives users the ability to refresh a selected or named range of cells in a grid or worksheet. If the range provided to this function contains more rows or columns than the actual grid has, the additional rows and columns are treated as comments and are thus part of the grid.

Range retrieval clears the Undo buffer, therefore the Undo operation cannot be used afterward.

Syntax

HypRetrieveRange(vtSheetName,vtRange,vtConnName)

ByVal vtSheetName As Variant

ByVal vtRange As Variant

ByVal vtConnectionName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtRange: Single continuous range to be refreshed. If vtRange is Null, the entire worksheet is refreshed, and GetUsedRange is used on the worksheet specified to get the range to be refreshed.

vtConnectionName: Friendly name of the connection to be used to refresh the range. If vtConn is Null, the active connection associated with the worksheet is used to refresh the range on that worksheet. If no connection is associated, an error is returned.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypRetrieveRange Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtRange As Variant, ByVal vtConnName As Variant) As Long
Worksheets("Sheet2").Names.Add name:="MyRange", RefersTo:="=$E$11:$F$28"

Sub Sample_RetrieveRange
    Worksheets(Empty).Names.Add name:="MyRange", RefersTo:="=$E$11:$F$28"
    sts = HypRetrieveRange("Empty", range("E11:F28"), "Sample")
    'retrieve by regular range
    sts = HypRetrieveRange(Empty, range("MyRange"), "Sample")
    'retrieve by named range
End sub
```

HypSetActiveConnection

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypSetActiveConnection() is used to associate the current active worksheet with one of the active connections.

Note: HypSetActiveConnection does not work with worksheets that contain Report Designer objects

Syntax

HypSetActiveConnection (vtConnectionName)

ByVal vtConnectionName as Variant

Parameters

vtConnectionName: Name of the active connection which is to be associated with the current active worksheet. It is not case-sensitive.

Return Value

Long. If successful, return value is 0; otherwise, the appropriate error code is returned.

Example

```
Declare Function HypSetActiveConnection Lib "HsAddin" (ByVal vtConnectionName As Variant) As Long

Sub Sample_SetActiveConnection
    sts = HypSetActiveConnection ("Demo_Basic")
End sub
```

HypSetAliasTable

Data source types: Essbase, Planning

Description

HypSetAliasTable() enables users to set the alias table

Syntax

HypSetAliasTable (ByVal vtSheetName As Variant, ByVal vtAliasTableName As Variant)

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtAliasTableName: Text name of the alias table. vtAliasTableName is of the form "Default", "Long Names" and so forth.

Return Value

0 if successful, else negative value

Example

```
Public Declare Function HypSetAliasTable Lib "HsAddin" (ByVal vtSheetName As Variant,  
ByVal vtAliasTableName As Variant) As Long
```

```
Sub Sample_SetActiveConnection  
    sts = HypSetAliasTable(Empty, "Long Name")  
End sub
```

HypSetAsDefault

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypSetAsDefault() is used to create a connection default.

Syntax

HypSetAsDefault (vtConnectionName)

ByVal vtConnectionName as Variant

Parameters

vtConnectionName: Name of the private active connection which needs to be made default. Parameter to be passed should be a private connection name whose value can be found in the Registry :- HKCU\Software\Hyperion Solutions\HyperionSmartView\Connections

Return Value

Long. If successful, return value is 0; otherwise, the appropriate error code is returned.

Example

```
Declare Function HypSetAsDefault Lib "HsAddin" (ByVal  
vtConnectionName As Variant) As Variant  
  
Sub Sample_SetAsDefault  
sts = HypSetAsDefault("buildtie7_w32Simple_w32Simple")  
MsgBox (sts)  
End sub
```

HypSetBackgroundPOV

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypSetBackgroundPOV() sets the POV for the connection object in the POV Manager.

Syntax

HypSetBackgroundPOV(vtFriendlyName, ParamArray MemberList())

ByVal vtFriendlyName As Variant

ParamArray MemberList() As Variant

Parameters

vtFriendlyName: Connection name for the data source provider.

MemberList: A list of strings which describe the member combination for which a data value will be retrieved. If MemberList is Null or Empty, the top level value is used.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypSetBackgroundPOV Lib "HsAddin" (ByVal vtFriendlyName, ParamArray  
MemberList() As Variant) As Long
```

```
Sub SetBGPOV()  
    X=HypSetBackgroundPOV ("My Connection","Year#Qtr1", "Market#East")  
End Sub
```

HypSetCellsDirty

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypSetCellsDirty() marks selected data range dirty for submit data.

Syntax

HypSetCellsDirty (vtSheetName, vtRange)

ByVal vtSheetName As Variant

ByVal vtRange As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtRange: Variant data range to be marked as dirty.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypSetCellsDirty Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal  
vtRange As Variant) As Long
```

```
Sub SetDirtyCells()  
    X=HypSetCellsDirty (Empty, Range ("A3:B3"))  
End Sub
```

HypSetCollItems

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only),
Hyperion Enterprise (ad hoc only)

Description

HypSetColItems() sets the members for the nth column dimension for the dynamic link query. If the nth column does not exist, a new column is appended.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#)

Syntax

HypSetColItems (vtColumnID, vtDimensionName, ppMemberList())

ByVal vtColumnID As Variant

ByVal vtDimensionName As Variant

ParamArray ppMemberList() As Variant

Parameters

vtColumnID: The column number n.

vtDimensionName: The dimension name.

ppMemberList: The list of member names.

Return Value

Long. Returns 0 if successful, otherwise, returns the appropriate error code.

Example

```
Declare Function HypSetColItems Lib "HsAddin" (ByVal vtColID As Variant, ByVal vtDimensionName As Variant, ParamArray MemberList() As Variant) As Long
```

```
Sub Macro()  
    Dim vtGrid as Variant  
    Sts = HypConnect(Empty, "system", "password", "SalesDemoBasic")  
    Sts = HypRetrieve(Empty)  
    Range ("B2").Select  
    Sts = HypGetSourceGrid (Empty, vtGrid)  
    Sts = HypSetColItems (1, "Market", "East", "West", "South",  
        "Central", "Market")  
End sub
```

HypSetConnAliasTable

Data source types: Essbase, Planning

Description

HypSetConnAliasTable() enables users to set the alias table for a connection.

Syntax

HypSetConnAliasTable (ByVal vtConnName As Variant, ByVal vtAliasTableName As Variant)

Parameters

vtConnName: Text name of the connection. vtConnName is of the form "SampleBasic". If vtConnName is Null or Empty, it will return an error . The basic requirement for this function is that it should have an active connection. For an active connection only the Alias table can be changed.

vtAliasTableName: Text name of the Alias table. vtAliasTableName can be of the form "Default", "Long Names", "None" and so forth. This parameter cannot be Null or Empty. If no Alias has to be applied then you can use the parameter "None" for that purpose.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypSetConnAliasTable Lib "HsAddin" (ByVal vtConnName As Variant,  
ByVal vtAliasTableName As Variant) As Long
```

```
Sub Sample_SetAliasTableForConnection
```

```
    sts = HypSetConnAliasTable("SampleBasic", "Long Name")
```

```
End sub
```

HypSetConnectionInfo

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypSetConnectionInfo() is used to modify the connection information in the query.

The parameters passed for HypSetConnectionInfo() should be match the connection information stored with that connection name.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#)

Syntax

HypSetConnectionInfo (vtServerName, vtUserName, vtPassword, vtApplicationName, vtDatabaseName, vtFriendlyName, vtURL, vtProviderType)

ByVal vtServerName As Variant

ByVal vtUserName As Variant

ByVal vtPassword As Variant

ByVal vtApplicationName As Variant

ByVal vtDatabaseName As Variant

ByVal vtFriendlyName As Variant

ByVal vtURL As Variant

ByVal vtProviderType As Variant

Parameters

vtServerName: The server name in the query.

vtUserName: The user name in the query.

vtPassword: The user password in the query.

vtApplicationName: The application name in the query.

vtDatabaseName: The database name in the query.

vtFriendlyName: The friendly connection name in the query.

vtURL: The provider URL in the query.

vtProviderType: The provider type in the query.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypSetConnectionInfo Lib "HsAddin" (ByVal vtServerName As Variant,  
ByVal vtUserName As Variant, ByVal vtPassword As Variant, ByVal vtApplicationName As  
Variant, ByVal vtDatabaseName As Variant, ByVal vtFriendlyName As Variant, ByVal vtURL  
As Variant, ByVal vtProviderType As Variant) As Long
```

```
Sub Macro()  
    Dim vtGrid as Variant  
    Sts = HypConnect(Empty, "system", "password", "DemoBasic")  
    Sts = HypRetrieve(Empty)  
    Range ("B2").Select  
    Sts = HypGetSourceGrid (Empty, vtGrid)  
    Sts = HypSetConnectionInfo("localhost", "system",  
        "password", "Sample", "Basic", "SampleBasic",
```

```
        "http://localhost:13080/aps/SmartView", provider)
End sub
```

HypSetGlobalOption

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypSetGlobalOption() sets individual workspace options. For option descriptions, see [Chapter 13, "Smart View Global Options."](#)

Note: You can set only one option at a time.

Syntax

HypSetGlobalOption(vtItem, vtGlobalOption)

ByVal vtItem As Long

ByVal vtGlobalOption As Variant

Parameters

vtItem: Number indicating which option is to be retrieved. See [Table 15, "HypGetGlobalOption Parameter Numbers and Options,"](#) on page 172 for values.

vtGlobalOption: A Boolean or Number value denoting the option being set for vtItem. If vtGlobalOption is Null or Empty, no action is performed.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

The following example sets the option to display error messages only.

```
Declare Function HypSetGlobalOption Lib "HsAddin" (ByVal vtItem As Long, ByVal vtGlobalOption As Variant) As Long
```

```
Sub SetGlobal()
    X=HypSetGlobalOption(5, 3)
    If X=0 Then
        MsgBox("Message level is set to 3 - No messages")
    Else
        MsgBox("Error. Message level not set.")
    End If
End Sub
```

HypSetLinkMacro

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypSetLinkMacro() sets the macro name to be run to perform the dynamic link query action.

Note: Once the link action is triggered from the **Visualize in Excel** menu item, the macro name set by this function name will be run.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#).

Syntax

HypSetLinkMacro (vtMacroName)

ByVal vtMacroName As Variant

Parameters

vtMacroName: The name of the macro to be run.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypSetLinkMacro Lib "HsAddin" (ByVal vtMacroName As Variant) As Long

Sub Auto_Open()
    Sts = HypUseLinkMacro(True)
    Sts = HypSetLinkMacro("Sheet1.Macro8")
End Sub
```

HypSetMenu

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypSetMenu() removes or restores the Smart View menu from Excel.

Syntax

HypSetMenu(bSetMenu)

ByVal bSetMenu As Boolean

Parameters

bSetMenu: Boolean value indicating whether to remove or restore the Smart View menu for Excel. A True value indicates that the menu should be restored. A False value indicates that the menu should be removed.

Return Value

Returns 0 if successful. If the menu cannot be set, returns an error code.

Example

```
Declare Function HypSetMenu Lib "HsAddin" (ByVal bSetMenu As Boolean) As Long

Sub SetMyMenu()
    X=HypSetMenu(TRUE)
End Sub
```

HypSetPages

Data source types: Planning (forms only), Financial Management (forms only), Hyperion Enterprise (forms only)

Description

HypSetPages() sets the page members for the selected sheet.

Syntax

HypSetPages (ByVal vtSheetName, ParamArray MemberList())

ByVal vtSheetName As Variant

ParamArray MemberList() As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

ParamArray MemberList(): The list of desired page member items in the form `Dimension#Current Member`. If MemberList is Null or Empty, the top level value is used.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypSetPages Lib "HsAddin" (ByVal vtSheetName, ParamArray  
MemberList() As Variant) As Long  
  
Sub SetPages()  
X=HypSetPages (Empty,"Entity#Operations","Scenario#Current")  
End Sub
```

HypSetPOV

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypSetPOV() sets the POV for the selected sheet.

Syntax

```
HypSetPOV(vtSheetName ParamArray MemberList())  
  
ByVal vtSheetName As Variant  
  
ParamArray MemberList() As Variant
```

Parameters

vtSheetName: For future use. Currently the active sheet is used.

MemberList: A list of strings which describe the member combination for which a data value will be retrieved. If MemberList is Null or Empty, the top level value is used.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypSetPOV Lib "HsAddin" (ByVal vtSheetName, ParamArray MemberList() As  
Variant) As Long  
  
Sub SetPOV()  
X=HypSetPOV (Empty,"Year#Qtr1", "Market#East")  
End Sub
```

HypSetPOVItems

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypSetPOVItems() sets the POV dimensions for the dynamic link query.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#)

Syntax

HypSetPOVItems (ppMemberList())

ParamArray ppMemberList() As Variant

Parameters

ppMemberList: The list of desired POV items in the form Dimension#Current Member.

Return Value

Returns 0 if successful; otherwise, returns the negative error code.

Example

```
Declare Function HypSetLinkMacro Lib "HsAddin" (ByVal vtMacroName As Variant) As Long

Sub Macro()
    Dim vtGrid as Variant
    Sts = HypConnect(Empty, "system", "password", "MyDemoBasic")
    Sts = HypRetrieve(Empty)
    Range ("B2").Select
    Sts = HypGetSourceGrid (Empty, vtGrid)
    Sts = HypSetPOVItems ("Scenario#Scenario", "Measures#Measures")
End sub
```

HypSetRowItems

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

Sets the members for the nth row dimension for this dynamic link query. If the nth row does not exist, a new row is appended.

Note: This function is used specifically with dynamic link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#)

Syntax

HypSetRowItems (vtRowID, vtDimensionName, ppMemberList())

ByVal vtRowID As Variant

ByVal vtDimensionName As Variant

ParamArray ppMemberList() As Variant

Parameters

vtRowID:The row number n.

vtDimensionName: The dimension name.

ppMemberList: The list of member names.

Return Value

Long. Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypSetRowItems Lib "HsAddin" (ByVal vtRowID As Variant, ByVal vtDimensionName As Variant, ParamArray MemberList() As Variant) As Long

Sub Macro()
    Dim vtGrid as Variant
    Sts = HypConnect(Empty, "system", "password", "DemoBasic")
    Sts = HypRetrieve(Empty)
    Range ("B2").Select
    Sts = HypGetSourceGrid (Empty, vtGrid)
    Sts = HypSetRowItems(1, "Product", "100", "200", "300", "400", "Diet", "Product")
End sub
```

HypSetSharedConnectionsURL

Data Source Types: Essbase, Planning, Financial Management, Hyperion Enterprise, Oracle BI EE

Description

HypSetSharedConnectionsURL() sets the Shared Connections URL in the config file and Options dialog box.

Syntax

HypSetSharedConnectionsURL (vtDefaultURL As Variant)
ByVal vtAPSURL As Variant)

Parameters

vtDefaultURL: the new Shared Services URL to be set.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypSetSharedConnectionsURL Lib "HsAddin" (ByVal vtAPSURL  
As Variant) As Long  
Sub SubHypSetSharedConnectionsURLTest()  
Dim lRet As Long  
lRet = HypSetSharedConnectionsURL("http://<server>:19000/workspace/SmartViewProviders")  
End Sub
```

HypSetSheetOption

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise, Oracle BI
EE

Description

HypSetSheetOption() sets individual spreadsheet options.

Note: You can set only one option at a time (this function is not plural).

Syntax

HypSetSheetOption(vtSheetName, vtItem, vtOption)
ByVal vtSheetName As Variant
ByVal vtItem As Variant
ByVal vtOption As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtItem: Number indicating which option is to be set. See [Table 16 on page 180](#) for a list of values.

vtOption: A Boolean value denoting the new value of item.

[Table 16 on page 180](#) indicates which options are set for which number and the expected data type.

Return Values

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypSetSheetOption Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtItem As Variant, ByVal vtOption As Variant) As Long
```

```
Sub SetSheet()  
X=HypSetSheetOption(Empty, 6, FALSE)  
If X=0 Then  
    MsgBox("#Missing values will appear. ")  
Else  
    MsgBox("Error. #Missing option not set.")  
End If  
End Sub
```

HypSetSubstitutionVariable

Data source types: Essbase

Description

HypSetSubstitutionVariable() creates substitution variables in Essbase. If the variable already exists, then its value is set to the new specified value.

Syntax

```
HypSetSubstitutionVariable (vtSheetName, vtApplicationName, vtDatabaseName,  
vtVariableName, vtVariableValue)
```

ByVal vtSheetName As Variant

ByVal vtApplicationName As Variant

ByVal vtDatabaseName As Variant

ByVal vtVariableName As Variant

ByVal vtVariableValue As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtApplicationName: The application name to define the scope for the new variable. If vtApplicationName is Null or Empty , the scope of the variable created is global.

vtDatabaseName: The database name to define the scope for the new variable. If vtDatabaseName is Null or Empty, the scope of the variable created is global within the application specified.

vtVariableName: The variable name to be created. Required.

vtVariableValue: The value to be assigned to the variable. Required.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypSetSubstitutionVariable Lib "HsAddin" (ByVal vtSheetName As Variant,  
ByVal vtApplicationName As Variant, ByVal vtDatabaseName As Variant, ByVal  
vtVariableName As Variant, ByVal vtVariableValue As Variant) As Long
```

```
Sub Sample_HypSetSubstitutionVariable  
    Dim X as Long  
    X = HypSetSubstitutionVariable (Empty, "Sample", "Basic",  
        "Account", "100")  
End Sub
```

HypSubmitData

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise, Oracle Business Intelligence Enterprise Edition

Description

HypSubmitData() updates the database with modified data from the specified spreadsheet.

Note: HypSubmitData() is not supported with aggregate storage databases or in a clustered environment.

Note: The ability to update the database depends on the access permissions of the submitter. To update data, you must have at least Write access to the database.

Syntax

HypSubmitData(vtSheetName)

ByVal vtSheetName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

Return Value

For forms: Returns 0 if form is submitted successfully; otherwise, returns the appropriate error code.

For ad hoc: Returns 0 if ad hoc grid is submitted successfully and HsSetVal functions were run, if any. Returns 1 if sheet was not connected but HsSetVal functions were run, if any. Returns 2 if sheet had no adhoc grid but HsSetVal functions were run, if any. Otherwise, returns the appropriate error code.

Example

```
Declare Function HypSubmitData Lib "HsAddin" (ByVal vtSheetName As Variant) As Long

Worksheets(Empty).range("B2").value = 8023
Worksheets(Empty).range("B2").Select
sts = HypSubmitData(Empty)
```

HypTranslate

Data source types: Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypTranslate() calls the Translate method for Financial Management data sources.

Syntax

HypTranslate (vtSheetName, vtRange)

ByVal vtSheetName As Variant

By Val vtRange As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtRange: Range object which refers to the data to be used. Passing an empty or null parameter uses the current selection from the sheet.

Return Value

Returns 0 if successful; otherwise, returns the corresponding error code.

Example

```
Declare Function HypTranslate Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtRange
As Variant) As Long

sts = HypTranslate (Empty, Empty)
```

HypUndo

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypUndo() restores the previous database view. A database view is the view of the spreadsheet after performing Zoom In, Zoom Out, Keep Only, Remove Only, or Refresh commands.

Syntax

HypUndo (vtSheetName)

ByVal vtSheetName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypUndo Lib "HsAddin" (ByVal vtSheetName As Variant) As Long
```

```
Sub Undo()  
    X=HypUndo(Empty)  
End Sub
```

HypUseLinkMacro

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypUseLinkMacro() is used to set the flag to specify the type of link view, static or dynamic.

Note: Static and dynamic link views share the same menu option; therefore, it is necessary to turn the flag on before performing the dynamic link query. Once done with dynamic link views, turn the flag off.

Note: This function is used specifically with link views, as described in [“VBA Functions and Dynamic Link Views” on page 122](#).

Syntax

HypUseLinkMacro (bSetView)

ByVal bSetView as Boolean

Parameters

bSetView: When flag is set to true, dynamic link is performed. When the flag is set to false, static link is performed.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypUseLinkMacro Lib "HsAddin" (ByVal bUse As Boolean) As Long

Sub Macro()
    Sts = HypUseLinkMacro(True)
End sub
```

HypZoomIn

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypZoomIn() retrieves and expands data from Smart View based on the selected members.

Syntax

HypZoomIn(vtSheetName, vtSelection, vtLevel, vtAcross)

ByVal vtSheetName As Variant

ByVal vtSelection As Variant

ByVal vtLevel As Variant

ByVal vtAcross As Variant (not used)

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtSelection: Range object which refers to the members that will be zoomed. If selection is Null or Empty, the active cell is used

vtLevel: Number indicating the granularity of the zoom. The following list describes the valid level numbers and their actions:

0 = Children

1 = Descendants

2 = Bottom level

If Null, Empty or an incorrect value is passed, the currently selected option is used.

vtAcross: Not used.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypZoomIn Lib "HsAddin" (ByVal sheetName As Variant, ByVal vtSelection As Variant, ByVal vtLevel As Variant, ByVal vtAcross As Variant) As Long
```

```
Sub ZoomData()  
X=HypZoomIn(Empty, RANGE("B3"), 1, FALSE)  
If X = 0 Then  
    MsgBox("Zoom successful.")  
Else  
    MsgBox("Zoom failed.")  
End If  
End Sub
```

HypZoomOut

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypZoomOut() collapses the view of data based on the selected members.

Syntax

HypZoomOut(vtSheetName, vtSelection)

ByVal vtSheetName As Variant

ByVal vtSelection As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtSelection: Range object which refers to the members that will be zoomed out. If selection is Null or Empty, the active cell is used.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypZoomOut Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtSelection As Variant) As Long
```

```
Sub UnZoomData()  
X=HypZoomOut(Empty, RANGE("B3"))  
If X = 0 Then  
    MsgBox("Zoom out successful.")  
Else  
    MsgBox("Zoom out failed.")  
End If  
End Sub
```

Visual Basic Menu Functions

VBA menu functions are identical to the equivalent commands on the Smart View menu and ribbon. Use the functions to perform actions as if you selected them from the menu or ribbon. The requirements for the functions are the same as those for the menu commands. For example, if you must be logged in to Essbase to use a menu command, you must also be logged in to an Essbase server to use the equivalent Visual Basic function.

The Smart View VBA menu equivalent functions:

- [HypMenuVAbout](#)
- [HypMenuVAdjust](#)
- [HypMenuVBusinessRules](#)
- [HypMenuVCalculation](#)
- [HypMenuVCellText](#)
- [HypMenuVCollapse](#)
- [HypMenuVConnect](#)
- [HypMenuVCopyDataPoints](#)
- [HypMenuVExpand](#)
- [HypMenuVFunctionBuilder](#)
- [HypMenuVInstruction](#)
- [HypMenuVKeepOnly](#)

- [HypMenuVMemberSelection](#)
- [HypMenuVMigrate](#)
- [HypMenuVOptions](#)
- [HypMenuVPasteDataPoints](#)
- [HypMenuVPivot](#)
- [HypMenuVPOVManager](#)
- [HypMenuVQueryDesigner](#)
- [HypMenuVRedo](#)
- [HypMenuVRefresh](#)
- [HypMenuVRefreshAll](#)
- [HypMenuVRefreshOfflineDefinition](#)
- [HypMenuVRemoveOnly](#)
- [HypMenuVRulesOnForm](#)
- [HypMenuVRunReport](#)
- [HypMenuVSelectForm](#)
- [HypMenuVShowHelpHtml](#)
- [HypMenuVSubmitData](#)
- [HypMenuVSupportingDetails](#)
- [HypMenuVSyncBack](#)
- [HypMenuVTakeOffline](#)
- [HypMenuVUndo](#)
- [HypMenuVVisualizeinExcel](#)
- [HypMenuVVisualizeinHVE](#)
- [HypMenuVZoomIn](#)
- [HypMenuVZoomOut](#)

HypMenuVAbout

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVAbout() opens the About box which displays copyright, version, and system information.

Syntax

HypMenuVAbout()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVAbout Lib "HsAddin" () As Long

Sub MAbout()
    X=HypMenuVAbout()
End Sub
```

HypMenuVAdjust

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVAdjust() enables you to adjust values in data cells.

Syntax

HypMenuVAdjust()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVAdjust Lib "HsAddin" () As Long

Sub MAdjust()
    X=HypMenuVAdjust()
End Sub
```

HypMenuVBusinessRules

Data source types: Planning

Description

HypMenuVBusinessRules() enables you to select Business Rules for Planning data forms.

Syntax

HypMenuVBusinessRules()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVBusinessRules Lib "HsAddin" () As Long

Sub MBusinessRules()
    X=HypMenuVBusinessRules()
End Sub
```

HypMenuVCalculation

Data source types: Essbase, Financial Management (ad hoc only), Hyperion Enterprise

Description

HypMenuVCalculation() can be used to open the Calculation Scripts dialog box and calculate the active database or checks on the status of an active database calculation.

Syntax

HypMenuVCalculation()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypMenuVCalculation Lib "HsAddin"() As Long

Sub MCalc()
    X=HypMenuVCalculation()
End Sub
```

HypMenuVCellText

Data source types: Planning, Financial Management, Hyperion Enterprise (forms only)

Description

HypMenuVCellText() enables you to enter, view, and edit background comments in cells and save them to the database.

Syntax

HypMenuVCellText()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVCellText Lib "HsAddin" () As Long

Sub MCellText()
    X=HypMenuVCellText()
End Sub
```

HypMenuVCollapse

Data source types: Planning (forms only)

Description

HypMenuVCollapse() collapses all levels of detail for the selected cells.

Syntax

HypMenuVCollapse()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVCollapse Lib "HsAddin" () As Long

Sub MHypMenuVCollapse()
    X=HypMenuVCollapse()
End Sub
```

HypMenuVConnect

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVConnect() can be used to connect to a data source instance.

Syntax

HypMenuVConnect()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypMenuVConnect Lib "HsAddin" () As Long

Sub MConn()
    X=HypMenuVConnect()
End Sub
```

HypMenuVCopyDataPoints

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVCopyDataPoints() copies data points from Excel for pasting into Word or PowerPoint.

Syntax

HypMenuVCopyDataPoints()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVHypMenuVCopyDataPoints Lib "HsAddin" () As Long

Sub MCopyDataPoints()
    X=HypMenuVCopyDataPoints()
End Sub
```

HypMenuVExpand

Data source types: Planning (forms only)

Description

HypMenuVExpand() displays all levels of detail for the selected cells.

Syntax

HypMenuVExpand()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVExpand Lib "HsAddin" () As Long

Sub MExpand()
    X=HypMenuVExpand()
End Sub
```

HypMenuVFunctionBuilder

Data source types: Essbase Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVFunctionBuilder() opens the Function Builder, where you create and validate functions.

Syntax

HypMenuVFunctionBuilder()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVFunctionBuilder Lib "HsAddin" () As Long

Sub MFunctionBuilder()
    X=HypMenuVFunctionBuilder()
End Sub
```

HypMenuVInstruction

Data source types:Planning (forms only), Financial Management (forms only), Hyperion Enterprise (forms only)

Description

HypMenuVInstruction() displays instructions that may be associated with a Planning data form.

Syntax

HypMenuVInstruction()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVInstruction Lib "HsAddin" () As Long

Sub MInstruction()
    X=HypMenuVInstruction()
End Sub
```

HypMenuVKeepOnly

Data source types: Essbase (ad hoc only), Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypMenuVKeepOnly() retains only the selected member (the active cell) or member range in the sheet.

Syntax

HypMenuVKeepOnly()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypMenuVKeepOnly Lib "HsAddin" () As Long

Sub MKeepOnly()
    X=HypMenuVKeepOnly()
End Sub
```

HypMenuVMemberSelection

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVMemberSelection() enables you to select and filter members.

Syntax

HypMenuVMemberSelection()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVMemberSelection Lib "HsAddin" () As Long

Sub MMemberSelection()
    X=HypMenuVMemberSelection()
End Sub
```

HypMenuVMigrate

Data source types: Financial Management, Hyperion Enterprise

Description

HypMenuVMigrate() enables users to launch the Financial Management and Hyperion Enterprise migration utility for Active WorkBook Migration and Batch Migration.

Syntax

HypMenuVMigrate (vtOption, vtOutput)

ByVal vtOption As Variant

ByRef vtOutput As Variant

Parameters

vtOption: Number that indicates which migration utility to be launched.

- 1. Financial Management Active Workbook Migration
- 2. Financial Management Batch Migration
- 3. Hyperion Enterprise Active WorkBook Migration

- 4. Hyperion Enterprise Batch Migration

vtOutput: Output parameter. Returns the migration result.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVMigrate Lib "HsAddin" (ByVal vtOption As Variant, ByRef  
vtOutput As Variant) As Long
```

```
Sub MigrateHFM()  
sts = HypMenuVMigrate(1, out)  
MsgBox (out)  
MsgBox (sts)  
End Sub
```

HypMenuVOptions

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVOptions() enables you to select options for the active sheet and customize the behavior of Smart View, using the Options dialog box.

Syntax

HypMenuVOptions()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypMenuVOptions Lib "HsAddin" () As Long
```

```
Sub MOptions()  
X=HypMenuVOptions()  
End Sub
```

HypMenuVPasteDataPoints

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVPasteDataPoints() pastes data points that were copied from Excel into Word or PowerPoint.

Syntax

HypMenuVPasteDataPoints()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVPasteDataPoints Lib "HsAddin" () As Long

Sub MVPasteDataPoints()
    X=HypMenuVPasteDataPoints()
End Sub
```

HypMenuVPivot

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypMenuVPivot() changes the orientation (from row to column or from column to row) of the group of members associated with the active cell.

Syntax

HypMenuVPivot()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypMenuVPivot Lib "HsAddin"() As Long

Sub MPivot()
    X=HypMenuVPivot()
End Sub
```

HypMenuVPOVManager

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVPOVManager() opens the POV Manager where you can perform operations on a POV.

Syntax

HypMenuVPOVManager()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVPOVManager Lib "HsAddin" () As Long

Sub MPOVManager()
    X=HypMenuVPOVManager()
End Sub
```

HypMenuVQueryDesigner

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypMenuVQueryDesigner() opens the Query Designer.

Syntax

HypMenuVQueryDesigner()

Return Value

Returns 0 if successful. A negative number indicates a local failure. A return value greater than zero indicates a failure on the server.

Example

```
Declare Function HypMenuVQueryDesigner Lib "HsAddin" () As Long
Sub MDesigner()
```

```
X=HypMenuVQueryDesigner ()  
End Sub
```

HypMenuVRedo

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypMenuVRedo() reverses an Undo operation.

Syntax

```
HypMenuVRedo()
```

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVRedo Lib "HsAddin" () As Long  
  
Sub MRedo()  
    X=HypMenuVRedo()  
End Sub
```

HypMenuVRefresh

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVRefresh() retrieves data into the active sheet, and places the data at the beginning of the active worksheet.

Syntax

```
HypMenuVRefresh()
```

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypMenuVRefresh Lib "HsAddin" () As Long

Sub MRetrieve()
    X=HypMenuVRefresh()
End Sub
```

HypMenuVRefreshAll

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVRefreshAll() refreshed data in all worksheets in an Excel workbook.

Syntax

HypMenuVRefreshAll()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVRefreshAll Lib "HsAddin" () As Long

Sub MRefreshAll()
    X=HypMenuVRefreshAll()
End Sub
```

HypMenuVRefreshOfflineDefinition

Data source types: Planning

Description

HypMenuVRefreshOfflineDefinition() refreshes the Offline data form definition and data.

Syntax

HypMenuVRefreshOfflineDefinition()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVRefreshOfflineDefinition Lib "HsAddin" () As Long

Sub MRefreshOfflineDefinition()
    X=HypMenuVRefreshOfflineDefinition()
End Sub
```

HypMenuVRemoveOnly

Data source types: Essbase Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypMenuVRemoveOnly() removes only the selected member (the active cell) or member range in the sheet.

Syntax

HypMenuVRemoveOnly()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypMenuVRemoveOnly Lib "HsAddin"() As Long

Sub MRemoveOnly()
    X=HypMenuVRemoveOnly()
End Sub
```

HypMenuVRulesOnForm

Data source types: Planning (forms only)

Description

HypMenuVRulesOnForm() enables you to execute Calculate Form and Calculate Currencies business rules.

Syntax

HypMenuVRulesOnForm()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVRulesOnForm Lib "HsAddin" () As Long

Sub MRulesOnForm()
    X=HypMenuVRulesOnForm()
End Sub
```

HypMenuVRunReport

Data source types: Essbase Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypMenuVRunReport() runs a report designed in the Query Designer.

Syntax

HypMenuVRunReport()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVRunReport Lib "HsAddin" () As Long

Sub MRunReport()
    X=HypMenuVRunReport()
End Sub
```

HypMenuVSelectForm

Data source types: Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVSelectForm() enables you to select Financial Management data forms.

Syntax

HypMenuVSelectForm()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVSelectForm Lib "HsAddin" () As Long

Sub MSelectForm()
    X=HypMenuVSelectForm()
End Sub
```

HypMenuVShowHelpHtml

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVShowHelpHtml() launches the online help.

Syntax

HypMenuVShowHelpHtml()

Parameter

vtHelpPage: The name of the HTML file that launches the help.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVShowHelpHtml Lib "HsAddin" (ByVal vtHelpPage As Variant)
As Long

Sub MDisConn()
    X=HypMenuVShowHelpHtml(launch.html)
End Sub
```

HypMenuVSubmitData

Data source types: Essbase, Essbase, Planning, Financial Management, Hyperion Enterprise

Description

HypMenuVSubmitData() updates the active database on the server with data that has been modified in your sheet or marked as “dirty” using the SetCellsDirty call.

Syntax

HypMenuVSubmitData()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypMenuVSubmitData Lib "HsAddin" () As Long

Sub MSubmit()
    X=HypMenuVSubmitData()
End Sub
```

HypMenuVSupportingDetails

Data source types: Planning

Description

HypMenuVSupportingDetails() For Planning data sources, enables you to provide supplemental calculations for a one-dimensional range of cells.

Syntax

HypMenuVSupportingDetails()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVSupportingDetails Lib "HsAddin" () As Long

Sub MSupportingDetails()
    X=HypMenuVSupportingDetails()
End Sub
```

HypMenuVSyncBack

Data source types: Planning

Description

HypMenuVSyncBack() synchronizes data from an offline Planning data form to the server.

Syntax

HypMenuVSyncBack()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVSyncBack Lib "HsAddin" () As Long

Sub MSyncBack()
    X=HypMenuVSyncBack()
End Sub
```

HypMenuVTakeOffline

Data source types: Planning

Description

HypMenuVTakeOffline() enables you to take Planning data forms offline.

Syntax

HypMenuVTakeOffline()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVTakeOffline Lib "HsAddin" () As Long

Sub MTakeOffline()
    X=HypMenuVTakeOffline()
End Sub
```

HypMenuVUndo

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypMenuVUndo() restores the previous database view.

Syntax

HypMenuVUndo()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVUndo Lib "HsAddin" () As Long

Sub MUndo()
    X=HypMenuVUndo()
End Sub
```

HypMenuVVisualizeinExcel

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypMenuVVisualizeinExcel() disconnects you from any currently connected databases.

Syntax

HypMenuVVisualizeinExcel()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVVisualizeinExcel Lib "HsAddin" () As Long

Sub MVisualizeinExcel()
```

```
X=HypMenuVVisualizeinExcel()  
End Sub
```

HypMenuVVisualizeinHVE

Data source types: Essbase

Description

HypMenuVVisualizeinHVE() opens Visual Explorer.

Syntax

```
HypMenuVVisualizeinHVE()
```

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Public Declare Function HypMenuVVisualizeinHVE Lib "HsAddin" () As Long  
  
Sub MVisualizeinHVE()  
    X=HypMenuVVisualizeinHVE()  
End Sub
```

HypMenuVZoomIn

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypMenuVZoomIn() expands the view of data according to the options specified in the Options dialog box.

Syntax

```
HypMenuVZoomIn()
```

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypMenuVZoomIn Lib "HsAddin" () As Long

Sub MZoomIn()
    X=HypMenuVZoomIn()
End Sub
```

HypMenuVZoomOut

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypMenuVZoomOut() collapses the view of data according to the options specified in the Options dialog box.

Syntax

HypMenuVZoomOut()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

```
Declare Function HypMenuVZoomOut Lib "HsAddin" () As Long

Sub MZoomOut()
    X=HypMenuVZoomOut()
End Sub
```

Using Spreadsheet Toolkit VBA Applications in Smart View

VBA applications created in Oracle's Hyperion® Essbase® Spreadsheet Toolkit can be converted to Smart View by making the following modifications:

- Replace the EssV prefix of Spreadsheet Toolkit functions with Hyp; for example, change EssVRemoveOnly to HypVRemoveOnly.
- Replace the EssMenuV prefix of Oracle's Hyperion® Essbase® Spreadsheet Toolkit menu functions with HypMenuV; for example, change EssMenuVZoomIn to HypMenuVZoomIn.
- Replace the declarations in `essxlvb.txt` with the declarations in `smartview.bas`.

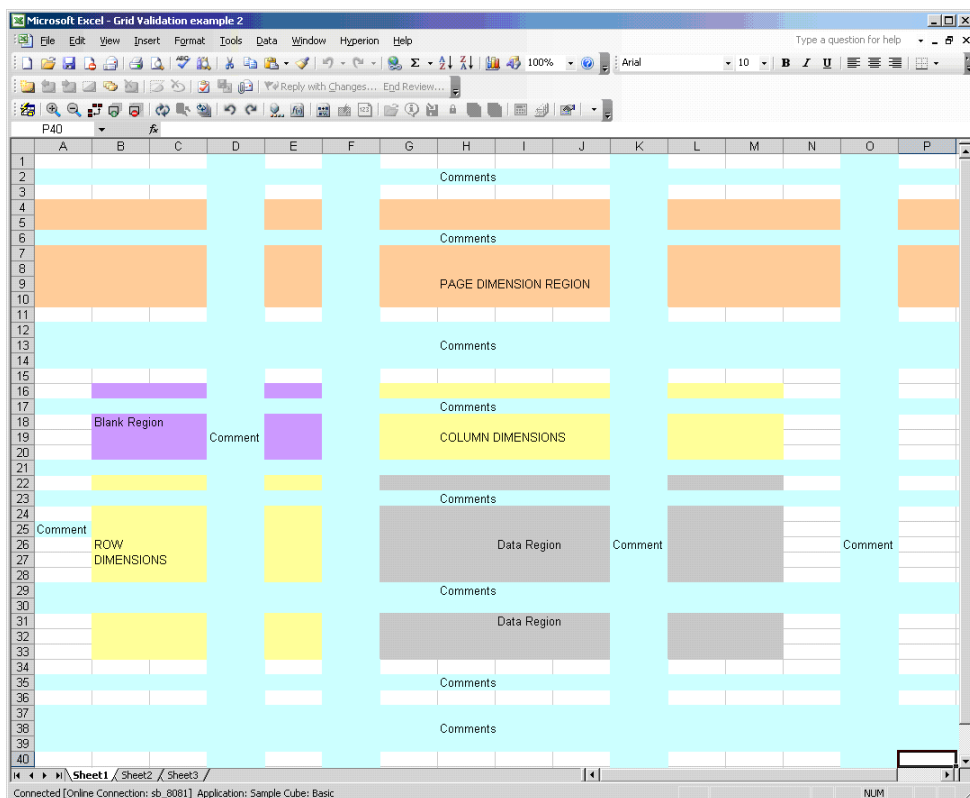
In This Chapter

About Free-Form Mode	255
Free-Form Guidelines	256
Creating a Free-Form Report	264
Resolving Dimension Names	264
Retrieving Attribute Dimensions in Free-Form Mode	265
Retrieving Data into Asymmetric Reports	266

About Free-Form Mode

In ad hoc analysis, if you are familiar with the dimensions and members of your database, you can use *free-form mode* by typing dimension and member names directly into cells. You can still use the POV and member selection and other ad hoc operations in free-form grids.

The graphic in this section shows the Smart View grid components. In the example Smart View grid, the Row dimensions are to the left and to the bottom of the Column dimensions. The Page dimensions can occur anywhere in the rows above the Column dimensions (applies to Essbase only).



The components of the Smart View grid are described in [Table 17](#).

Table 17 Smart View Grid Components

Grid Component	Description
Row Dimension	A dimension or member placed down one column across one or more rows in a worksheet
Column Dimension	A dimension or member placed on a row across one or more columns in a worksheet
Page Dimension	A dimension that applies to the entire page (Essbase only)
Comments	Text added by the user
Data Region	Areas of the grid that contain data for dimensions or members
Blank Region	Areas of the worksheet that contain no entries

Free-Form Guidelines

- A grid must have at least one row dimension and one column dimension.
- Each row dimension can contain members of only one dimension. Each column dimension can contain members of only one dimension.
- Members of one dimension can be entered only in **one** of the following regions:
 - In the same row

- In the same column
- Anywhere in the page dimension region
- The page dimension region can contain members of different dimensions, but no two members in the page dimension region can belong to the same dimension.
- Grids do not need to start in cell A1.
- Dimensions entered into the page dimension region override any corresponding default or existing dimensions in the page dimension region. For example, if the page dimension contains a Year dimension, and you enter Qtr1, then Qtr replaces Year in the page dimension.
- The replacement labels specified in the Data Options page of Smart View Options apply in free-form mode.
- Numerical entries are identified as data in the data region, otherwise as comments. If you want to use a number as a member name, precede it with a single quotation mark, for example '100.
- Precede member names that contain spaces between words with a single quotation mark.
- When connected to a duplicate member Essbase data source, select **Member Name Only** on the Member Options page of the Smart View Options dialog box to display fully qualified member names in the worksheet. To enter duplicate members, use the syntax for qualified member names as shown here:


```
[Income] . [Other]
[Expenses] . [Other]
```
- Oracle recommends a maximum grid size of 65,000 cells.
- Aliases from the current alias table are permitted in free-form grids, but aliases from other alias tables are treated as comments.

Entering Comments

Data source types: Essbase

Comments can be placed as follows:

- Between row dimensions
- Between column dimensions
- Between page dimensions
- Between dimensions and data cells
- Interleaved with members of page dimensions
- Comment rows and comment columns can be interleaved with row and columns dimension member
- Interleaved with members of row, column and page dimensions
- Cells that intersect row and column dimensions are data cells and cannot contain comments.

- Comments can be placed to the left, right, top, bottom of the grid. However, cells that intersect row and column dimensions in the upper right corner (the blank region) cannot contain comments.

Entering Dynamic Time Series Members

Smart View supports Dynamic Time Series members typed directly into free-form grids. This section contains some examples of supported and unsupported formats.

Supported Format Examples

When entering dynamic time series members into a free-form grid, some examples of supported formats are:

- Q-T-D(Jan)
- Y-T-D(Mar)
- M-T-D(Jun)

Unsupported Format Examples

When entering dynamic time series members into a free-form grid, some examples of unsupported formats are:

- Q-T-Date(Jan)
- QTDate(Jan)
- YTD (Jun)

Note: Dynamic time series members are not supported for Hyperion Enterprise.

Preserving Comments, Formulas, and Format

Smart View tries to preserve all comments, formulas, and customized report layouts. Some exceptions that may result in unexpected behavior are when the following actions are performed:

- Zoom in on a page dimension
- Pivot a dimension from the POV to a row or column
- Drag and drop a dimension from the POV to the worksheet
- Pivot a row dimension to a column dimension
- Switch the location of a row dimension to another row
- Switch the location of a column dimension to another column
- Change member aliases using the Change Alias Table command
- In Essbase or Hyperion Enterprise data sources, cutting and pasting from Microsoft Word into an Excel worksheet may cause unexpected behavior because of hidden characters. If this happens, contact your administrator, who can identify the issue through logs.

Valid Simple Grid

Figure 7 shows a valid simple grid, where Year is the Row dimension, Measures is the Column dimension and Product is the Page dimension.

Figure 7 Valid Simple Grid

	A	B	C	D	E
1				Product	
2		Profit	Inventory	Ratios	Measures
3	Qtr1				
4	Qtr2				
5	Qtr3				
6	Qtr4				
7	Year				

Valid and Invalid Grids

The following sections contain samples of valid and invalid grid layouts. Use these samples as guidelines when building your own free-form grids. The sample free-form grids pertaining to page dimensions and dynamic time series members are not applicable for Hyperion Enterprise.

For Hyperion Enterprise data sources, you cannot type dimension names in the free-form grids. You can type only member names.

Column Dimensions Interpreted as Page Dimensions

A valid free-form grid must have at least one row and at least one column dimension. When there is one row dimension and multiple members, all of different dimensions in the same top row, Smart View interprets the leftmost dimension in the row to be a column dimension and the others as page dimensions. Figure 8 shows a valid grid in which Year is the row dimension, Measure is the column dimension, and Product and Market are page dimensions.

Figure 8 Valid Grid

	A	B	C	D
1		Measures	Product	Market
2	Year			

Invalid Grid Where Smart View Interprets Stacked Dimensions

The first row that has multiple members, all of the same dimension, is identified as a column dimension. All the dimensions that are placed above this row are candidates for page dimension, if they comply with the rules for page dimension. There is an exception to this rule, as shown in Figure 9. According to the rules described to this point, you might expect Product and Market to be interpreted as page dimensions. However, Smart View interprets Product and Market as column dimensions, making this grid invalid.

Figure 9

	A	B	C	D	E
1		Market			
2		Product			
3		Profit	Inventory	Ratios	Measures
4	Year				

The Smart View logic is that if you identify the first column dimension on row R, and let C be the first column on this row R that contains a member, for each subsequent row above row R, if there is a member on the same column C and that row does not contain any other members, then that row is identified as a column dimension. Each such row that contains a member that is “stacked” on top of column C is a column dimension. A row that does not obey this condition is a candidate for a page dimension as long as it complies with the rules for page dimensions.

Valid Grid Where Stacked Dimensions Are Interpreted as Page Dimensions

Rows above a page dimension can only be candidates for a page dimension and cannot be a column dimension even if they obey the condition for “stackable” members. In [Figure 10](#), Product is a column dimension since it stacked on top of Profit; Market is a page dimension since it is not stacked on top of Profit. However, Scenario is a page dimension, even though it is stacked on top of Profit, because it is a row that is above a page dimension; hence, it cannot be a column dimension.

Figure 10

	A	B	C	D	E
1		Scenario			
2			Market		
3		Product			
4		Profit	Inventory	Ratios	Measures
5	Year				

Zooming In On a Page Dimension

If you zoom in on a page dimension, the page dimension is moved to a row dimension through a POV to Grid operation.

Figure 11

	A	B	C	D	E
1			Product		
2		Profit	Inventory	Ratios	Measures
3	Year	105522	117405	55.26163	105522

Results of Zoom In

For example, if you zoom in on Market in [Figure 11](#), it will result in [Figure 12](#).

Figure 12

	A	B	C	D	E	F
1			Profit	Inventory	Ratios	Measures
2	Product	Year	105522	117405	55.26163	105522

Invalid Row and Column Dimensions and First Members

The first members of every column dimension must occur on the same column, and the first members of every row dimension must occur on the same row. [Figure 13](#) is invalid because cell B2 is on the first column of the column dimensions and it has to be a member of the Measures dimension, whereas it is a comment.

Figure 13

	A	B	C	D
1		100-10	100-30	100
2		Comment	Measures	Measure
3	Year			

Invalid Grid that Contains Comment Row and Comment Column in Dimension Region

Row and column dimension regions can be interleaved with comment rows and comment columns. [Figure 14](#) below is invalid because the comment in cell C2 does not belong to either a comment row or a comment column. (Both row 2 and column C have dimension members.)

Figure 14

	A	B	C	D
1		100-10	100-30	100
2		Measures	Comment	Measures
3	Year			

Valid Grid Containing Dynamic Time Series Member

Dynamic Time Series (DTS) members can be placed on the free-form grid using Member Selection. Additionally, you can also manually type a DTS member name into a grid.

[Figure 15](#) is a valid free-form grid and the member name Q-T-D(Jan) can be manually typed into the grid.

Figure 15

	A	B
1		Measures
2	Year	
3	Qtr1	
4	Q-T-D(Jan)	

See [“Entering Dynamic Time Series Members” on page 258](#) for examples of supported format types.

Valid Grid Consisting of Two Columns by Two Rows

[Figure 16](#) is a basic two columns by two rows layout showing the Product and Market dimensions in the first row and column, and members of Sales and Year in the second row and column. [Figure 26](#) is specific for Hyperion Enterprise. [Figure 26](#) is a basic two columns by two rows layout showing the Scenario and Entity dimensions in the first row and column, and members of Account and Period dimensions in the second row and column.

Figure 16

	A	B	C	D	E	F
1			Product	Product	Product	Product
2			Profit	Inventory	Ratios	Measures
3	Market	Qtr1				
4	Market	Qtr2				
5	Market	Qtr3				
6	Market	Qtr4				
7	Market	Year				

Valid Grid with Comments Inserted in Blank Row and Column

[Figure 17](#) provides an example of comment usage in Smart View. It is a valid grid with comments bordering the data/metadata region. In this case comments are listed in columns A and H, and in rows 1, 2, and 10. These comments are retained for retrieval and zoom operations. For a list of rules when using comments on a grid, see [“Entering Comments” on page 257](#).

Figure 17

	A	B	C	D	E	F	G	H
1			c1	d1				
2			c2	d2				
3				Product	Product	Product	Product	
4				Profit	Inventory	Ratios	Measures	
5	a5	Market	Qtr1					h5
6	a6	Market	Qtr2					h6
7		Market	Qtr3					
8		Market	Qtr4					
9		Market	Year					
10			c10	d10				

Valid Grid with Formulas Inserted in Blank Row and Column

[Figure 18](#) provides a valid example of Excel-based formula usage in the Smart View environment. Formulas are added to the edges of the grid. These are standard Excel formulas and resolve based on your personal Excel settings. Formulas can be added in any area in which a comment can exist. For a list of formula usage guidelines, see [“Preserving Comments, Formulas, and Format” on page 258](#).

Figure 18

	A	B	C	D	E	F
1			Product	Product	Product	Product
2			Profit	Inventory	Ratios	Measures
3	Market	Qtr1				
4	Market	Qtr2				
5	Market	Qtr3				
6	Market	Qtr4				
7	Market	Year				
8			=SUM(C3:C6)			=SUM(F3:F6)

Valid Grid with POV Region and Attribute

Figure 19 provides a valid example of both the page region and attribute usage. In this example, Pkg Type and Budget are understood (by Smart View) to be page dimensions. Additionally, Pkg Type is an attribute dimension attached to the base member product. By drilling down on Pkg Type you can do attribute based analysis on measures as it relates to specific Product attributes. This can be further used to create a cross-tab analysis of product SKUs by attribute.

Figure 19

	A	B	C	D	E	F
1	Pkg Type					
2	Budget					
3						
4			Product	Product	Product	Product
5			Profit	Inventory	Ratios	Measures
6	Market	Qtr1				
7	Market	Qtr2				
8	Market	Qtr3				
9	Market	Qtr4				
10	Market	Year				

Valid Grid with Complex Comments

Figure 20 provides a valid, combined sample of using the page region, attributes, and comments on a single grid.

Figure 20

	A	B	C	D	E	F	G	H
1	Pkg Type							
2	Budget							
3		B3	C3	D3				
4				Product	Product		Product	Product
5					E5	F5	G5	
6				Profit	Inventory		Ratios	Measures
7	Market		Qtr1					
8	Market		Qtr2					
9		B9			E9	F9	G9	
10	Market		Qtr3					
11	Market		Qtr4					
12	Market		Year					
13								
14				D14	E14	F14		

Creating a Free-Form Report

Data source types: Essbase, Financial Management, Hyperion Enterprise

► To construct a free-form report:

- 1 Open a worksheet and connect to a data source.
- 2 In the worksheet, enter member names according to the rules specified in [“Free-Form Guidelines” on page 256](#).
- 3 Members may have duplicate names (for example, both East and West markets may contain a member named Portland — Maine and Oregon). To enter a duplicate member name:
 - In Essbase, use Member Selection to select members.
 - In Financial Management, the Member Name Resolution window is displayed if the member you enter has a duplicate. From the drop-down list, select the dimension of the member you entered and click OK. Repeat as necessary.
- 4 Refresh the grid.
- 5 Perform ad hoc operations and formatting as needed.

Resolving Dimension Names

When you select Member Selection in a blank grid, the Dimension Name Resolution dialog box is displayed, where you choose a dimension or members to place on the grid. This process is referred to as resolving dimensions because when you choose the Member Selection option on a blank sheet, the data source has no way of predicting which dimension or members, including attribute members, you want to see and choose from in the Member Selection dialog box.

When you choose dimensions and members using this method, be sure that the first cell you select on the sheet is the cell in which you want to begin your free-form layout.

Note: In Hyperion Enterprise, you cannot place dimensions on the grid.

► To resolve dimensions names in a grid:

- 1 Select a cell.
- 2 From the data source ribbon, select **Member Selection**.
- 3 In the **Dimension Name Resolution** dialog box, select the dimension to place on the sheet.
- 4 To orient members vertically in the worksheet starting from the cell you selected in [step 1](#), select the **Vertical Orientation** check box.

This check box is cleared by default; meaning that members will be oriented horizontally across the sheet from the cell you selected in [step 1](#).

- 5 Click **OK** to launch the **Member Selection** dialog box.
- 6 Select the members to place on the worksheet and click **OK**.
- 7 View the layout of the dimension or members you just placed on the sheet.
- 8 Perform one of the following actions:
 - If you placed members vertically on the grid:
 - Repeat [step 1](#) through [step 7](#) to place a new set of members horizontally on the grid
 - Use the POV to pivot a dimension to a row
 - Type a dimension or member name (including attribute dimension or member names) directly on the grid
 - If you placed members horizontally on the grid:
 - Repeat [step 1](#) through [step 7](#) to place a new set of members vertically on the grid
 - Use the POV to pivot a dimension to a column
 - Type a dimension or member name (including attribute dimension or member names) directly on the grid
- 9 Refresh.

Retrieving Attribute Dimensions in Free-Form Mode

In structured grid processing operations, attribute dimensions are not shown. In free-form, you can type an attribute dimension member in the grid and it will be processed and validated. Only the attribute dimension member that you added will be displayed and used during processing and validation; the remaining attribute dimension members will not be included. Structured grid operations that follow this free-form request will retain the attribute dimension member.

If the base dimension exists in the worksheet, you can also retrieve an attribute member by typing the name directly in the worksheet.

The method for retrieving attributes described in this section assumes you are starting with a blank worksheet.

Note: Hyperion Enterprise does not support Attribute dimensions.

► To retrieve an attribute dimension in free-form using Member Selection:

- 1 In a blank worksheet, select a cell.
- 2 From the data source ribbon, select **Member Selection**.
- 3 In the **Dimension Name Resolution** dialog box, select the attribute dimension.
- 4 To orient members vertically in the worksheet starting from the cell you selected in [step 1](#), select the **Vertical Orientation** check box.

This check box is cleared by default; meaning that members will be oriented horizontally across the sheet from the cell you selected in [step 1](#).

- 5 Click **OK** to launch the **Member Selection** dialog box.
- 6 Select the members to place on the worksheet.

Note: You can also add attribute dimensions and members to the sheet.

Retrieving Data into Asymmetric Reports

When you retrieve data into a worksheet, the resulting report can be either *symmetric* or *asymmetric*. Symmetric reports are characterized by repeating identical groups of members. For example, a symmetric report may contain Actual and Budget members nested below Year members (Qtr1, Qtr2, Qtr3, and Qtr4).

An asymmetric report is characterized by groups of nested members that differ by at least one member. There can be a difference in the number of members or in the names of members.

You can create asymmetric reports in one of the following ways:

- Enter member names into the worksheet in free-form retrieval mode.
- Zooming in with **Within Selected Group** selected on the Member Options page of the Options dialog box.
- Suppress rows that contain missing values, zero values, or underscore characters during data retrievals.

Retrieving data into an asymmetric report may take a long time on large reports.



Accessibility

In This Appendix

Enabling Accessibility for Smart View	267
Keyboard Equivalents.....	267
Smart View Panel Navigation	276

This appendix describes the accessibility features of Smart View. For information regarding supported assistive technologies, refer to the *Oracle Hyperion Enterprise Performance Management System Installation Start Here*.

Enabling Accessibility for Smart View

You do not need to enable accessibility specifically for Smart View; it is always in accessible mode. Smart View output is in the form of Excel spreadsheets, Word documents, and PowerPoint slides, which are accessible through Microsoft Office. For information about Excel, Word, or PowerPoint accessibility, refer to Microsoft Office product documentation.

Note: If you are using JAWS® Screen Reading Software, we recommend using the Internet Explorer browser.

Keyboard Equivalents

This section describes the keyboard equivalents for the following:

- Smart View ribbon keys in Excel (Office 2007 and 2010)
- Smart View ribbon keys in Word and PowerPoint (Office 2007 and 2010)
- Data provider ribbon keys in Excel (Office 2007 and 2010)
- Data provider ad hoc keys in Excel (Office 2007 and 2010)
- Smart View menu items (Office 2003)
- Smart View Panel navigation

Note: Keyboard equivalents for languages other than English may be different from those listed in these tables.

Keyboard equivalents for provider ribbons may vary if other Excel add-ins are installed. For example, Y may become Y1 or Y2 if there are ribbons created by add-ins in addition to Smart View.

Smart View Ribbon Keyboard Equivalents

Table 18 shows the keyboard equivalents for the Smart View ribbon in Excel. Table 19 shows the keyboard equivalents for the Smart View ribbon in Word and PowerPoint.

Table 18 Smart View Ribbon Keyboard Equivalents for Excel

Ribbon Item	Key
Display Smart View ribbon	Alt+S
Open	Alt+S+O
Open, Smart View Panel	Alt+S+O+P
Open, Active Connections	Alt+S+O+T
Open, Reset Connections	Alt+S+O+E
Open, Reporting and Analysis Document	Alt+S+O+O
Open, Most Recently Used	Alt+S+O+U
Undo	Alt+S+E+N
Redo	Alt+S+E+D
Copy	Alt+S+E+C
Paste	Alt+S+E+V
Functions	Alt+S+U
Functions, Manage POV	Alt+S+U+M
Functions, Build Function	Alt+S+U+B
Refresh	Alt+S+D+R
Refresh, Refresh	Alt+S+D+R+R
Refresh, Refresh All	Alt+S+D+R+A
Submit Data	Alt+S+D+B
Options	Alt+S+S
Help	Alt+S+H

Ribbon Item	Key
Help, Contents	Alt+S+H+C
Help, Technical Support	Alt+S+H+S
Help, EPM Documentation	Alt+S+H+E
Help, About	Alt+S+H+Z
Sheet Info	Alt+S+F
More	Alt+S+R+M
More, Migrate Active Workbook (Financial Management)	Alt+S+R+M+G
More, Migrate Batch (Financial Management)	Alt+S+R+M+B
More, Migrate Active Workbook (Hyperion Enterprise)	Alt+S+R+M+R
More, Migrate Batch (Hyperion Enterprise)	Alt+S+R+M+E
Insert Connection List	Alt+S+R+M+I
Import Metadata	Alt+S+R+M+P

Table 19 Smart View Ribbon Keyboard Equivalents for Word and PowerPoint

Ribbon Item	Key
Display Smart View ribbon	Alt+S2
Open	Alt+S2+O
Open, Smart View Panel	Alt+S2+O+P
Open, Reporting and Analysis Document	Alt+S2+O+O
Open, Reporting and Analysis Document, Create Template	Alt+S2+O+O+C
Open, Reporting and Analysis Document, Refresh Template	Alt+S2+O+O+R
Open, Most Recently Used	Alt+S2+O+U
Copy	Alt+S2+E+C
Paste	Alt+S2+E+V
Manage POV	Alt+S2+E+M
Refresh	Alt+S2+R
Visualize in HVE	Alt+S2+D+H
Visualize in Excel	Alt+S2+E
Options	Alt+S2+S

Ribbon Item	Key
Help	Alt+S2+H
Help, Contents	Alt+S2+H+C
Help, Technical Support	Alt+S2+H+S
Help, EPM Documentation	Alt+S2+H+E
Help, About	Alt+S2+H+Z

Data Provider Ribbons

Table 20 lists the keyboard equivalents on data provider ribbons. Table 21 lists the keyboard equivalents for items only on the Essbase, Planning Ad Hoc, HFM Ad Hoc, Enterprise Ad Hoc and OBIEE ribbons. Exceptions are noted in parentheses.

Table 20 Data Provider Ribbon Keyboard Equivalents for Excel

Ribbon Item	Key
Analyze (Planning only)	Alt+Y+Y
Refresh	Alt+Y+D+R
Refresh, Refresh	Alt+Y+D+R+R
Refresh, Refresh All	Alt+Y+D+R+A
Drill-through	Alt+Y+D+D
Add Member	Alt+Y+A+M
Submit Data	Alt+Y+D+B
Cell Comments	Alt+Y+D+E
Supporting Detail (Planning only)	Alt+Y+D+P
Document Attachment (Planning only)	Alt+Y+D+T
Lock (Planning only)	Alt+Y+D+L
Calculate	Alt+Y+D+C
Calculate, Business Rules (Planning only)	Alt+Y+D+C+B
Calculate, Rules on Form (Planning only)	Alt+Y+D+C+U
Calculate, Calculate	Alt+Y+D+C+D
Calculate, Force Calculate (HFM and Enterprise only)	Alt+Y+D+C+F
Calculate, Translate (HFM and Enterprise only)	Alt+Y+D+C+T

Ribbon Item	Key
Calculate, Force Translate (HFM and Enterprise only)	Alt+Y+D+C+R
Consolidate (HFM and Enterprise only)	Alt+Y+D+O
Consolidate, Consolidate (HFM and Enterprise only)	Alt+Y+D+O+N
Consolidate, Consolidate All (HFM and Enterprise only)	Alt+Y+D+O+S
Consolidate, Calculate Contribution (HFM and Enterprise only)	Alt+Y+D+O+A
Consolidate, Force Calculate Contribution (HFM only)	Alt+Y+D+O+O
Consolidate, Consolidate All With Data (HFM and Enterprise only)	Alt+Y+D+O+L
View Comments	Alt+S+W
Adjust	Alt+Y+D+U
Adjust, Adjust	Alt+Y+D+U+J
Adjust, Grid Spread (Planning only)	Alt+Y+D+U+G
Adjust, Mass Allocate (Planning only)	Alt+Y+D+U+A
Drill-through	Alt+Y+D+D
More (Planning only)	Alt+Y+D+M
More, Job Console (Planning only)	Alt+Y+D+M+J
More, Member Formula (Planning only)	Alt+Y+D+M+M
More, Instructions (Planning only)	Alt+Y+D+M+I
More, Offline (Planning only)	Alt+Y+D+M+F
More, Offline, Take Offline (Planning only)	Alt+Y+D+M+F+T
More, Offline, Sync Back to Server (Planning only)	Alt+Y+D+M+F+Y
More, Offline, Refresh Offline Definition (Planning only)	Alt+Y+D+M+F+R
Manage Process (Planning only)	Alt+Y+W+P
Copy Version (Planning only)	Alt+Y+W+V

Table 21 Essbase, OBIEE, and Ad Hoc Ribbon Keyboard Equivalents for Excel

Ribbon Item	Key
Zoom In	Alt+Y+A+Z
Zoom In, Next Level	Alt+Y+A+Z+N
Zoom In, All Levels	Alt+Y+A+Z+A

Ribbon Item	Key
Zoom In, Bottom Level	Alt+Y+A+Z+B
Zoom Out	Alt+Y+A+U
Pivot	Alt+Y+A+T
Pivot, Pivot	Alt+Y+A+T+P
Pivot Pivot to POV	Alt+Y+A+T+T
Keep Only	Alt+Y+A+K
Remove Only	Alt+Y+A+O
Member Selection	Alt+Y+A+I
Query	Alt+Y+A+Q
Query, Query Designer	Alt+Y+A+Q+Q
Query, Run Report	Alt+Y+A+Q+R
Query, Execute MDX (Essbase only)	Alt+Y+A+Q+E
Query, Data Filter	Alt+Y+A+Q+D
View Qualified Name	Alt+Y+A+N
Preserve Format (except Enterprise Ad Hoc)	Alt+Y+A+P
Change Alias (Essbase only)	Alt+Y+A+A
Data Perspective (Essbase only)	Alt+Y+A+D
Smart Slice (except Enterprise)	Alt+Y+A+S
Cascade (except Enterprise Ad Hoc)	Alt+Y+A+C
Cascade, Same Workbook (except Enterprise Ad Hoc)	Alt+Y+A+C+S
Cascade, New Workbook (except Enterprise Ad Hoc)	Alt+Y+A+C+N
Refresh	Alt+Y+D+R
Refresh, Refresh	Alt+Y+D+R+R
Refresh, Refresh All	Alt+Y+D+R+A
POV	Alt+Y+D+P
View Comments	Alt+Y+D+W
Calculate	Alt+Y+D+C
Vizualize (Essbase only)	Alt+Y+D+V

Ribbon Item	Key
Vizualize, Vizualize in HVE (Essbase only)	Alt+Y+D+V+H
Vizualize, Visualize in Excel (Essbase and OBI EE only)	Alt+Y+D+V+E
Drill-through	Alt+Y+D+D
Adjust	Alt+Y+D+U
Submit Data	Alt+Y+D+B

Office 2003 Menu Keyboard Equivalents

Microsoft Office 2003 products do not have ribbons. [Table 22](#) displays the keyboard equivalents for items on the Office 2003 Smart View menu.

Table 22 Smart View Menu Keyboard Equivalents

Menu Item	Key
Open	Alt+S+O
Open, Smart View Panel	Alt+S+O+P
Open, Active Connections	Alt+S+O+T
Open, Reset Connection	Alt+S+O+E
Open, Reporting and Analysis Document	Alt+S+O+O
Open, Reporting and Analysis Document, Import	Alt+S+O+O+I
Open, Reporting and Analysis Document, Edit	Alt+S+O+O+E
Open, Recently Used	Alt+S+O+U
Edit	Alt+S+E
Edit, Undo	Alt+S+E+N
Edit, Redo	Alt+S+E+D
Edit, Copy	Alt+S+E+C
Edit, Paste	Alt+S+E+V
Edit, Manage POV	Alt+S+E+M
Edit, Build Function	Alt+S+E+B
Data	Alt+S+D
Data, Refresh	Alt+S+R
Data, Refresh All	Alt+S+D+A

Menu Item	Key
Data, Submit Data	Alt+S+D+B
Data, Cell Comments	Alt+S+D+E
Data, Supporting Details	Alt+S+D+P
Data, Document Attachment	Alt+S+D+T
Data, Drill-through	Alt+S+D+D
Data, Lock	Alt+S+D+L
Data, Calculate	Alt+S+D+C
Data, Calculate, Calculate	Alt+S+D+C+C
Data, Calculate, Force Calculate	Alt+S+D+C+F
Data, Calculate, Translate	Alt+S+D+C+T
Data, Calculate, Force Translate	Alt+S+D+C+R
Data, Calculate, Business Rules	Alt+S+D+C+B
Data, Calculate, Rules on Form	Alt+S+D+C+U
Data, Consolidate	Alt+S+D+O
Data, Consolidate, Consolidate	Alt+S+D+O+N
Data, Consolidate, Consolidate All	Alt+S+D+O+S
Data, Consolidate, Calculate Contribution	Alt+S+D+O+A
Data, Consolidate, Force Calculate Contribution	Alt+S+D+O+O
Data, Consolidate, Consolidate All With Data	Alt+S+D+O+L
Data, Adjust	Alt+S+D+U
Data, Adjust, Adjust	Alt+S+D+U+J
Data, Adjust, Grid Spread	Alt+S+D+U+G
Data, Adjust, Mass Allocate	Alt+S+D+U+A
Data, Visualize	Alt+S+D+V
Data, Visualize, Visualize in HVE	Alt+S+D+V+H
Data, Visualize, Visualize in Excel	Alt+S+D+V+E
Data, View Comments	Alt+S+D+W
Data, Job Console	Alt+S+D+J

Menu Item	Key
Data, Member Formula	Alt+S+D+M
Data, Instructions	Alt+S+D+I
Data, Offline	Alt+S+D+F
Data, Offline, Take Offline	Alt+S+D+F+T
Data, Offline, Sync Back To Server	Alt+S+D+F+Y
Data, Offline, Refresh Offline Definition	Alt+S+D+F+R
Analysis	Alt+S+A
Analysis, Zoom In	Alt+S+A+Z
Analysis, Zoom In, Zoom In	Alt+S+A+Z+Z
Analysis, Zoom In, Next Level	Alt+S+A+Z+N
Analysis, Zoom In, All Level	Alt+S+A+Z+A
Analysis, Zoom In, Bottom Level	Alt+S+A+Z+B
Analysis, Zoom Out	Alt+S+A+U
Analysis, Keep Only	Alt+S+A+K
Analysis, Remove Only	Alt+S+A+O
Analysis, Pivot	Alt+S+A+V
Analysis, Pivot To POV	Alt+S+A+T
Analysis, SmartSlice	Alt+S+A+S
Analysis, Cascade	Alt+S+A+C
Analysis, Cascade, Same Workbook	Alt+S+A+C+S
Analysis, Cascade, New Workbook	Alt+S+A+C+N
Analysis, Query	Alt+S+A+Q
Analysis, Query, Query Designer	Alt+S+A+Q+Q
Analysis, Query, Run Report	Alt+S+A+Q+R
Analysis, Query, Execute MDX	Alt+S+A+Q+E
Analysis, Query, Data Filter	Alt+S+A+Q+D
Analysis, View Qualified Name	Alt+S+A+N
Analysis, Preserve Format	Alt+S+A+P

Menu Item	Key
Analysis, Change Alias	Alt+S+A+A
Analysis, Member Selection	Alt+S+A+I
Analysis, Data Perspective	Alt+S+A+D
Analysis, Add Member	Alt+S+A+M
Workflow	Alt+S+W
Workflow, Manage Process	Alt+S+W+P
Workflow, Copy Version	Alt+S+W+V
Options	Alt+S+S
Sheet Info	Alt+S+F
More	Alt+S+M
More, Migrate Active Workbook (Financial Management)	Alt+S+M+G
More, Migrate Batch (Financial Management)	Alt+S+M+B
More, Migrate Active Workbook (Hyperion Enterprise)	Alt+S+M+R
More, Migrate Batch (Hyperion Enterprise)	Alt+S+M+E
More, Insert Connection List	Alt+S+M+I
More, Import Metadata	Alt+S+M+P
POV	Alt+S+D+P
Help	Alt+S+H
Help, Contents	Alt+S+H+C
Help, Technical Support	Alt+S+H+S
Help, EPM Documentation	Alt+S+H+E
Help, About	Alt+S+H+Z

Smart View Panel Navigation

Table 23 displays the keys for navigating tin the Smart View Panel.

Table 23 Smart View Panel Navigation

Navigation	Keystroke
Switch between Smart View and Office components	F6

Navigation	Keystroke
Scroll through Smart View Panel	Tab
Scroll through items in tree view	Up and down arrow keys
In toolbars - move through buttons In tree views - expand or collapse a node	Left and right arrow keys
Move from tree view to Action Panel	Alt+tab
Open the drop-down menu	Alt+down arrow
Scroll through drop-down list	Up and down arrow keys
Select an item in the drop-down menu	Enter
Cancel selection	Escape
Close the Smart View Panel and move to grid (Office 2007 and 2010)	Alt+S+O+P
Close the Smart View Panel and move to grid (Office 2003)	Alt+F4



Using Other Applications with Smart View

In This Appendix

Crystal Ball EPM.....	279
Visual Explorer.....	280
Smart View and Spreadsheet Add-in.....	281
Migrating Functions.....	281

These are applications that you can use with Smart View if you hold the appropriate licenses for them.

Crystal Ball EPM

Data source types: See the *Oracle Crystal Ball Enterprise Performance Management Integration Guide*.

You use Crystal Ball EPM to analyze data from Smart View data sources in simulation and forecasting workbooks. These are Excel workbooks that contain one or more worksheets with a Crystal Ball EPM model and one or more other worksheets, each of which may be connected to any of the supported data sources. They are stored in a centralized EPM Workspace repository and can be accessed and managed through the Smart View Panel.

For more information, see the Crystal Ball EPM documentation set.

Working with Crystal Ball EPM Workbooks

Permissions set by the EPM Workspace administrator govern simulation and forecasting workbooks operations that you can perform from the Smart View Panel.

► To work with data in a Crystal Ball EPM workbook:

1 From the Smart View ribbon, select **Open**.

2 In the Smart View Panel, click  and select **Simulation Workbook**.

3 Click  and if requested, log in to the Crystal Ball EPM repository. A tree list containing the workbooks for which you have permission is displayed.

- 4 Double-click a workbook to open.
- 5 Perform Crystal Ball EPM operations as described in the Crystal Ball EPM product documentation.
Oracle recommends keeping the Oracle Crystal Ball Enterprise Performance Management, Fusion Edition model on a worksheet separate from data source worksheets.
- 6 Click **Submit Data** if needed.

Toolbar Operations

Use Simulation Workbook toolbar buttons to perform the following operations on workbooks and folders in the tree list.

- Connect to a repository
- Add, save, and delete workbooks
- Add and rename folders
- Refresh the tree list
- Set options to specify where workbook files are to be stored and the EPM Workspace agent with which to communicate (these options apply across all sessions running on the server). To do so, click **Options** and enter this information:
 - **URL:** the Web Services agent URL. Use this syntax: `http://<host>/raframework/services/BiPlus`
 - **Folder:** the name of the repository folder to contain the workbook file

Visual Explorer

Data Sources: Essbase

If your Smart View and Essbase installation includes Visual Explorer (HVE), you can use Visual Explorer's capabilities to view Essbase data in various graphical formats, and pass data back and forth between an Excel worksheet and Visual Explorer. For more information about Visual Explorer, see the Visual Explorer online help.

Viewing Visual Explorer Data from Microsoft Office

- To view data in Excel using Visual Explorer:
- 1 Open a worksheet in Excel.
 - 2 Connect to an Essbase data source.
 - 3 Create a report or use an existing one.
 - 4 Depending on your Office application:
 - Excel: From the Essbase ribbon, select **Visualize**, then **Visualize in HVE**.
 - Word or PowerPoint: From the Smart View ribbon, select **Visualize in HVE**.

The Visual Explorer interface is displayed (you may be prompted to sign in).

- 5 From **Show Me!**, select a graphical format and click **OK**.

Oracle Essbase Visual Explorer displays the spreadsheet associated with the data cells in graphical format.

Visual Explorer File Association

If Smart View and Spreadsheet Add-in are installed on the same computer, HVE Workbook (.twb) files are opened in Windows Explorer by Smart View even if the file was created in Spreadsheet Add-in.

If you uninstall Smart View and retain Spreadsheet Add-in on the computer, the file association for .TWB files is lost. Use Windows Explorer to establish .TWB file association manually with Spreadsheet Add-in.

Smart View and Spreadsheet Add-in

When both Smart View and Spreadsheet Add-in are installed on the same computer, mouse actions are interpreted as Spreadsheet Add-in commands. If you want Smart View to control mouse commands instead, you can instruct Spreadsheet Add-in to respond to commands only in Essbase connections that were established through Spreadsheet Add-in.

► To enable Smart View to control mouse commands:

- 1 Open Excel.
- 2 Select **Essbase**, then **Options**, then **Global**.
- 3 Select **Limit to Connected Sheets**.
- 4 Click **OK**.

Smart View will respond to mouse commands unless the connection to Oracle Essbase is established through Spreadsheet Add-in and not Smart View.

Note: You can connect to data sources from Smart View and Spreadsheet Add-in in the same workbook but not on the same worksheet.

Migrating Functions

Oracle Essbase Spreadsheet Add-in functions in Financial Management and Hyperion Enterprise can be converted to current Smart View syntax with the migration utility.

Converting Workbooks

You can convert workbooks that contain Financial Management Retrieve Data functions or Hyperion Enterprise HP Retrieve and VBA Retrieve functions by using the migration utility. For example, you can convert Financial Management functions such as HFMVal, HFMLnk, HFMLab, HFMDes, and HFMCur and Hyperion Enterprise functions such as HPVal, HPLnk, HPCur, HPHea, HPCde, and HPFul.

The utility might not be able to convert all of your functions. Some functions might require manual adjustment.

For functions that use cell references, the following functions are converted:

- If every parameter in the function is a cell reference. For example: =HFMVal(\$B\$1&\$C\$1&\$B\$2&\$C\$3&\$B\$5&\$C\$5&\$B\$6&\$C\$6).
- If the dimension parameters are specified in the function, the members are cell references, but the period separator is hard coded in the function. For example: =HFMVal("S#"&D2&".Y#"&D3&".VW#"&D5&".')

The following functions that use cell references are **not** converted:

- If the dimension parameters are specified in the function and the members and period separator are cell references. For example: =HFMVal("S#"&E2&".Y#"&E3&".VW#"&E5), where E2=Actual, E3=2004, E5="<Scenario View>."
- If the dimension parameters are specified in the function, the members are cell references, but the period separator is in a separate cell, the function is not converted. For example: =HFMVal("S#"&F2&C1&".Y#"&F3&C1&".VW#"&F5&C1), where C1=. (period separator).
- If the application specified in the function is a cell reference.
- If any cell in a workbook contains more than 1024 characters, the workbook does not convert properly. To reduce the size of data in cell, reference multiple functions, or remove dimensions that can be set in the background POV.

Before you run the migration utility, ensure that the path is correct (the default path is MIDDLEWARE_HOME\EPMSys11R1\common\empstatic\wsspace\). During migration, Excel inserts the original path of the add-in file to functions. This can make the functions too long and cause errors. Excel limits Smart View functions to a maximum of 256 characters.

Converting One Workbook

Data source types: Financial Management, Hyperion Enterprise

► To convert a workbook:

- 1 From the Smart View ribbon, select **More**, then **Migrate Active Workbook (Financial Management)** or **Migrate Active Workbook (Hyperion Enterprise)**.

If your functions contain application references, you must map the application to the corresponding connection.

- 2 Click **Convert**, then **OK**.
- 3 Migration results are displayed, including a list of any functions that failed to convert. You can manually adjust those functions.
- 4 To save the conversion results, click **Save Result**.
- 5 Select a location to store the results file, and click **Save**.
- 6 Click **Close**.

Converting Multiple Workbooks

Data source types: Financial Management, Hyperion Enterprise

➤ To convert multiple workbooks:

- 1 From the Smart View ribbon, select **More**, then **Migrate Batch (Financial Management)** or **Migrate Batch (Hyperion Enterprise)**.
- 2 In the Migration Wizard, click **Add** and select the workbooks that you want to convert.
- 3 Click **Next**. If your functions contain application references, you must map the application to the connection.

Migration results are displayed, including a list of any functions that failed to convert. You can manually adjust those functions.

- 4 In Oracle's Hyperion® Enterprise®, converted workbooks are automatically saved in the location of the original workbooks. In Financial Management, click **Save Result**.
- 5 Select a location for the results file and click **Save**.
- 6 Click **Done**.

Migrating Connections for Functions

In Financial Management, you can select a connection or connection reference for functions that do not contain an application reference when you migrate to Smart View.

➤ To migrate connections for functions:

- 1 From the Smart View ribbon, select **More**, then **Migrate Active Connections (HFM)**.
- 2 From **Function Migration — Application reference**, select an option:
 - **Do not update functions with a connection reference**.
 - **Add connection name to existing functions**, then select a connection name from the Connection Name list. This updates all functions with the specified connection name.
 - **Update functions with reference to connection list within selected worksheet**, then in Cell Reference, enter the cell to reference, for example, A2. This updates all functions with a cell reference in the current worksheet.

- **Update functions with reference to connection list on a new worksheet**, then enter the Worksheet name, and Cell Reference. This updates all functions with a cell reference to a different worksheet in the workbook.

Tip: You can create a drop-down list in any cell to be used as a reference within functions to refer to a connection name. From the Smart View ribbon, select **More**, then **Insert Connection List** to display a list of connections from which to choose in the current cell.

3 Click **OK**.



Finding Information

In This Appendix

Information about Data Sources and Other Products	285
Using Oracle User Productivity Kit	285

Information about Data Sources and Other Products

In general, this guide provides only procedural information for using the data source features that Smart View supports. For detailed information about the data sources and other products, see the product documentation available on the EPM Documentation Library. To open this library, from the Smart View ribbon, click the arrow next to **Help** and then **EPM Documentation**.

Using Oracle User Productivity Kit

If the Oracle User Productivity Kit (UPK) is deployed and Oracle Enterprise Performance Management Workspace, Fusion Edition is configured by an Administrator with a valid URL for the UPK Player package, users can access UPK content for EPM System. For more information on configuring UPK, see the “*Workspace Server Settings*” section in the *Oracle Enterprise Performance Management Workspace Administrator’s Guide* and the “Oracle User Productivity Kit” section in the *Application Support Guide*.

Note: There are pre built UPK content modules available. See the data sheets that include UPK for Oracle Hyperion Enterprise Performance Management System available on Oracle.com, <http://www.oracle.com/us/products/applications/tutor-upk/064788.html>. Financial Management and Planning modules include appropriate content for Smart View and Oracle Hyperion Financial Reporting Studio, Fusion Edition. Oracle Hyperion Financial Management, Fusion Edition and Oracle Hyperion Planning, Fusion Edition support invoking UPK content in a context sensitive manner. UPK content launched from Smart View or Reporting Studio launches the full player package outline unfiltered for context. Reporting Studio and Smart View users can utilize a roles filter to see only the Oracle Hyperion Smart View for Office, Fusion Edition or Oracle Hyperion Financial Reporting Studio, Fusion Edition content.

To open UPK Help, from the Smart View ribbon, click the arrow next to **Help**, and then select **Oracle User Productivity Kit**.

Index

A

- accessibility
 - enabling, [267](#)
 - keyboard equivalents, [267](#)
- ad hoc
 - Dynamic Time Series members, [41](#)
 - free-form, [255](#)
 - selecting members, [37](#)
- ad hoc analysis
 - about, [27](#)
 - starting, [27](#)
- adjusting values, [49](#)
- alias tables, [43](#)
- annotations, [74](#)
 - planning units, [74](#)
- Approvals, [71](#)
- asymmetric reports, [266](#)
 - definition of, [266](#)
 - retrieving data into, [266](#)
- attribute dimensions, in free-form grids, [265](#)

B

- business rules
 - Calculate Currencies, [78](#)
 - Calculate Form, [78](#)
 - launching, [77](#)
 - overview, [76](#)
 - runtime prompts, [77](#)

C

- Calculate Currencies business rule, [78](#)
- Calculate Form business rule, [78](#)
- calculating data, [50](#)
- calculating rows, inserting, [31](#)
- calculation scripts.. *See* business rules
- calculations, [82](#). *See also* supporting detail and business rules

- cascading, [30](#)
- cell comments, [49](#)
- cell styles, [110](#)
- cell text , [49](#)
- cells
 - currency code, location of, [52](#)
- comments, [49](#)
- comments in free-form
 - preserving, [258](#)
- comments, adding to
 - planning units, [74](#)
- components
 - Smart View, [13](#)
- composite data forms, [76](#)
- connecting
 - data sources, [16](#)
- connections
 - locating, [16](#)
 - private connections, [15](#)
 - private, creating, [17](#)
 - saving, [18](#)
 - shared connections, [15](#)
- consolidating data, [51](#)
- copy version, [75](#)
- copying
 - data points, [23](#)
 - worksheets, [22](#)
- Cross Dimension runtime prompt, [78](#)
- Crystal Ball, [279](#)
- currencies, [52](#)
 - changing, [52](#)
 - translating, [52](#)
- currency codes
 - location of, [52](#)
- custom formats, [36](#)

D

data

- adjusting, [49](#)
- calculating, [50](#)
- consolidating, [51](#)
- refreshing, [48](#)
- submitting, [47](#)

data filtering, [65](#)

data forms

- about, [33](#)
- ad hoc analysis, [34](#)
- composite, [76](#)
- customizing format, [36](#)
- Financial Management, [35](#)
- linked forms, [35](#)
- opening, [34](#)
- Planning, [34](#)
- Planning attributes, [34](#)
- refreshing offline, [87](#)
- working with offline, [86](#)

data perspective, [54](#)data source connections, [15](#)

data sources

- connecting, [16](#)
- disconnecting, [16](#)

decimal separator in data forms, [36](#)

dimensions

- about, [37](#)

disabling Smart View, [19](#)document attachment, [50](#)drill-through reports, [53](#)dynamic data points, [23](#)

dynamic link views

- working with, [122](#)

Eextensions, [110](#)**F**filtering data, [65](#)

Financial Reporting

- editing, [98](#)
- formats, [91](#)
- importing documents, [95](#)
- importing into Excel, [96](#)
- importing into Word and PowerPoint, [97](#)

templates, [98](#), [99](#)

formats in free-form

- preserving, [258](#)

formatting

- ad hoc grids, [28](#)
- Excel, [28](#)
- preserving formats, [29](#)

forms, data

- refreshing offline, [87](#)
- using linked forms, [35](#)
- working with offline, [86](#)

formulas

- Excel, [48](#)
 - ad hoc grids, [48](#)
 - data forms, [49](#)

formulas in free-form

- preserving, [258](#)

free-form, [255](#)

- asymmetric reports, [266](#)
- attribute dimensions, working with, [265](#)
- guidelines, [256](#)
- invalid grids, [259](#)
- valid grids, [259](#)

Function Builder, [113](#)

functions

- about, [113](#)
- creating, [113](#)
- creating manually, [114](#)
- error codes, [120](#)
- running, [115](#)

Gguidelines for free-form reporting, [256](#)**H**hierarchy of supporting detail, [82](#)

HsCurrency function

- syntax, [118](#)

HsDescription function

- syntax, [118](#)

HsGetSheetInfo function, [117](#)

HsGetText function

- syntax, [119](#)

HsGetValue function

- syntax, [116](#)

HsLabel function, [118](#)

- HsSetText function, [119](#)
- HsSetValue function, [116](#)
- HypCalculate VBA function, [131](#)
- HypCalculateContribution VBA function, [132](#)
- HypCell VBA function, [133](#)
- HypConnect VBA function, [134](#)
- HypConnected VBA function, [135](#)
- HypConnectionExists VBA function, [136](#)
- HypConsolidate VBA function, [136](#)
- HypConsolidateAll VBA function, [137](#)
- HypConsolidateAllWithData VBA function, [138](#)
- HypCreateConnection VBA function, [139](#)
- HypCreateConnectionEX VBA function, [141](#)
- HypDeleteCalc VBA function, [143](#)
- HypDeleteMetadata VBA function, [143](#)
- HypDisconnect VBA function, [145](#)
- HypDisconnectEx VBA function, [146](#)
- HypDisplayToLinkView, [147](#)
- HypExecuteCalcScript VBA function, [148](#)
- HypExecuteMDXEx VBA function, [154](#)
- HypExecuteQuery VBA function, [156](#)
- HypFindMember VBA function, [156](#)
- HypFindMemberEx VBA function, [158](#)
- HypForceCalculate VBA function, [159](#)
- HypForceCalculateContribution VBA function, [160](#)
- HypForceTranslate VBA function, [160](#)
- HypFreeDataPoint VBA function, [161](#)
- HypGetAncestor VBA function, [162](#)
- HypGetChildren VBA function, [164](#)
- HypGetColCount VBA function, [165](#)
- HypGetCollItems VBA function, [166](#)
- HypGetConnectionInfo VBA function, [167](#)
- HypGetDataPoint VBA function, [168](#)
- HypGetGlobalOption VBA function, [171](#)
- HypGetLinkMacro VBA function, [173](#)
- HypGetPagePOVChoices VBA function, [174](#)
- HypGetParent VBA function, [175](#)
- HypGetPOVCount VBA function, [175](#)
- HypGetPOVItems VBA function, [176](#)
- HypGetRowCount VBA function, [177](#)
- HypGetRowItems VBA function, [178](#)
- HypGetSharedConnectionsURL VBA function, [179](#)
- HypGetSheetOption VBA function, [179](#)
- HypGetSourceGrid VBA function, [181](#)
- HypGetSubstitutionVariable VBA function, [182](#)
- HypInvalidateSSO VBA function, [184](#)
- HypIsAncestor VBA function, [184](#)
- HypIsAttribute VBA function, [185](#)
- HypIsChild VBA function, [186](#)
- HypIsConnectedToSharedConnections VBA function, [187](#)
- HypIsDescendant VBA function, [189](#)
- HypIsExpense VBA function, [189](#)
- HypIsParent VBA function, [191](#)
- HypIsUDA VBA function, [193](#)
- HypKeepOnly VBA function, [194](#)
- HypListCalcScripts VBA function, [195](#)
- HypMenuVAbout VBA function, [233](#)
- HypMenuVAdjust VBA function, [234](#)
- HypMenuVBusinessRules VBA function, [234](#)
- HypMenuVCalculation VBA function, [235](#)
- HypMenuVCellText VBA function, [235](#)
- HypMenuVCollapse VBA function, [236](#)
- HypMenuVConnect VBA function, [236](#)
- HypMenuVCopyDataPoints VBA function, [237](#)
- HypMenuVExpand VBA function, [237](#)
- HypMenuVFunctionBuilder VBA function, [238](#)
- HypMenuVInstruction VBA function, [238](#)
- HypMenuVKeepOnly VBA function, [239](#)
- HypMenuVMemberSelection VBA function, [240](#)
- HypMenuVMigrate VBA function, [240](#)
- HypMenuVOptions VBA function, [241](#)
- HypMenuVPasteDataPoints VBA function, [241](#)
- HypMenuVPivot VBA function, [242](#)
- HypMenuVPOVManager VBA function, [243](#)
- HypMenuVQueryDesigner VBA function, [243](#)
- HypMenuVRedo VBA function, [244](#)
- HypMenuVRefresh VBA function, [244](#)
- HypMenuVRefreshAll VBA function, [245](#)
- HypMenuVRefreshOfflineDefinition VBA function, [245](#)
- HypMenuVRemoveOnly VBA function, [246](#)
- HypMenuVRulesOnForm VBA function, [246](#)
- HypMenuVRunReport VBA function, [247](#)
- HypMenuVSelectForm VBA function, [247](#)
- HypMenuVShowHelpHtml VBA function, [248](#)
- HypMenuVSubmitData VBA function, [248](#)
- HypMenuVSupportingDetails VBA function, [249](#)
- HypMenuVSyncBack VBA function, [250](#)
- HypMenuVTakeOffline VBA function, [250](#)
- HypMenuVUndo VBA function, [251](#)
- HypMenuVVisualizeinExcel VBA function, [251](#)
- HypMenuVVisualizeinHVE VBA function, [252](#)
- HypMenuVZoomIn VBA function, [252](#)

HypMenuVZoomOut VBA function, [253](#)
 HypOpenForm VBA function, [197](#)
 HypOtlGetMemberInfo VBA function, [198](#)
 HypPivot VBA function, [199](#)
 HypPivotToGrid VBA function, [200](#)
 HypPivotToPOV VBA function, [201](#)
 HypQueryMembers VBA function, [203](#)
 HypRedo VBA function, [205](#)
 HypRemoveConnection VBA function, [207](#)
 HypRemoveOnly VBA function, [208](#)
 HypResetFriendlyName VBA function, [209](#)
 HypRetrieve VBA function, [210](#)
 HypRetrieveAllWorkbooks VBA function, [210](#)
 HypRetrieveRange VBA function, [211](#)
 HypSetActiveConnection VBA function, [212](#)
 HypSetAsDefault VBA function, [213](#)
 HypSetBackgroundPOV VBA function, [214](#)
 HypSetCellsDirty VBA function, [215](#)
 HypSetColItems VBA function, [215](#)
 HypSetConnAliasTable VBA function, [216](#)
 HypSetConnectionInfo VBA function, [217](#)
 HypSetGlobalOption VBA function, [219](#)
 HypSetLinkMacro VBA function, [220](#)
 HypSetMenu VBA function, [220](#)
 HypSetPages VBA function, [221](#)
 HypSetPOV VBA function, [222](#)
 HypSetPOVItems VBA function, [223](#)
 HypSetRowItems VBA function, [223](#)
 HypSetSharedConnectionsURL VBA function, [224](#)
 HypSetSheetOption, [225](#)
 HypSetSubstitutionVariable VBA function, [226](#)
 HypSubmitData VBA function, [227](#)
 HypTranslate VBA function, [228](#)
 HypUndo VBA function, [229](#)
 HypUseLinkMacro VBA function, [229](#)
 HypZoomIn VBA function, [230](#)
 HypZoomOut VBA function, [231](#)

I
 Import Metadata, [22](#)
 information, finding, [285](#)
 inserting
 calculating rows, [31](#)
 non-calculating rows, [31](#)
 Interactive Reporting
 editing, [95](#)
 importing documents, [92](#)

 importing into Excel, [93](#)
 importing into Word and PowerPoint, [94](#)

J
 job console, [74](#)

K
 Keep Only, [29](#)
 keyboard equivalents
 Smart View functions, [267](#)
 Smart View ribbon items, [268](#)

L
 line item detail, [82](#)
 link views. *See* dynamic link views
 linked data forms, [35](#)
 logging, [24](#)

M
 MDX queries, [66](#)
 member names, qualified, [43](#)
 member perspective, [42](#)
 Member Range runtime prompt, [78](#)
 member selection
 for runtime prompts, [77](#)
 members
 about, [37](#)
 adding in Financial Management data forms, [35](#)
 filtering by attribute, [40](#)
 filtering by subsets, [41](#)
 period-to-date, [41](#)
 selecting, [37](#)
 selecting from POV, [39](#)
 selecting in Financial Management data forms, [35](#)
 migrating
 connections, [283](#)
 considerations, [282](#)
 converting a workbook, [282](#)
 converting multiple workbooks, [283](#)
 migration utility, [282](#)
 Move Down option, [83](#)
 Move Up option, [83](#)

N
 non-calculating rows, inserting, [31](#)

O

offline. *See* Offline Planning

offline data forms

- refreshing, 87
- working with, 86

Offline Planning

- refreshing data in, 87
- working in, 86

offline, Planning, 84

- data forms, 84

options

- advanced, 108
- cell styles, 110
- data, 106
- formatting, 109
- member, 105
- provider, 111

Oracle User Productivity kit (UPK)

- using, 285

P

Page dimension

- searching for members in, 75

pivoting, 30

Planning

- Approvals, 71
- copying versions, 75
- data forms, 34
- offline, 84
- preferences, 84

planning units

- annotations, 74
- changing the status of, 71
- finding, 72
- out of office assistant, 74
- promotional path, 73

POV

- copying, 45
- deleting, 45
- hiding, 40
- printing in header and footer, 25
- selecting members, 39

POV Manager, 44

- selecting members, 44

preferences

- Planning, 84

preserving in free-form

- comments, 258
- formats, 258
- formulas, 258

private connections, 15

- saving shared connections as, 18

Production Reporting

- editing, 101, 103
- importing documents, 99, 102
- importing into Excel, 99
- importing into Word and PowerPoint, 100, 101

Q

queries

- creating, 63
- editing, 64

Query Designer, 63

- time-related data, 65

R

redo, 21

- options, 21

Refresh, 48

refreshing data

- while working offline, 87

refreshing Reporting and Analysis documents, 90

Remove Only, 29

Reporting and Analysis

- editing and refreshing documents, 90
- importing documents, 89

Reporting and Analysis documents

- refreshing, 90
- updating, 90

reports

- asymmetric, 266
- cascading, 30
- free-form, 255

retrieving

- increasing speed, 266
- into asymmetric reports, 266
- performance impact, 266

ribbons, 13

Rules on Form dialog box, 78

runtime prompts

- types, 77

S

search wild cards, [24](#)
 searching
 for members in the Page dimension, [75](#)
 security certificate for Workspace servers, [92](#)
 servers
 adding and deleting, [18](#)
 shared connections, [15](#)
 shared workbooks, [25](#)
 Sheet Info, [24](#)
 sheet information, [24](#)
 simulation workbooks, [279](#)
 toolbar, [280](#)
 sliders, [59](#)
 creating from joined queries, [60](#)
 creating from query, [60](#)
 Smart Slices
 about, [57](#)
 boundaries, [61](#)
 creating, [61](#)
 creating reports, [57](#)
 deleting reports, [59](#)
 preferences, [62](#)
 sliders, [59](#)
 Smart View
 disabling, [19](#)
 Smart View Panel
 about, [14](#)
 Smart View ribbon
 keyboard equivalents, [268](#)
 spreading data
 over time periods, [79](#)
 Spreadsheet Add-in, [281](#)
 static link views. *See* dynamic link views
 submitting a planning unit, [72](#)
 submitting data, [47](#)
 substitution variables, [31](#)
 supporting detail
 adding, [82](#)
 changing, [83](#)
 hierarchy of, [82](#)
 overview, [82](#)
 viewing, [83](#)
 Supporting Detail window
 using, [82](#)
 symmetric reports, [266](#)
 synchronizing data, [86](#)

T

task lists, [67](#)
 about, [67](#)
 completing tasks, [69](#)
 creating reports, [69](#)
 executing, [68](#)
 integrating with Outlook, [69](#)
 opening, [67](#)
 viewing, [68](#)
 thousands separator in data forms, [36](#)
 time periods, spreading data over, [79](#)
 time-related data
 Query Designer, [65](#)
 troubleshooting, [24](#)

U

undo, [21](#)
 options, [21](#)
 URL
 and data cells, [50](#)
 launching from cell, [50](#)
 linked to planning cells, [50](#)
 Use Members on Forms option
 with runtime prompts, [77](#)
 User Productivity Kit, [285](#)

V

VBA functions
 HypCalculate, [131](#)
 HypCalculateContribution, [132](#)
 HypCell, [133](#)
 HypConnect, [134](#)
 HypConnected, [135](#)
 HypConnectionExists, [136](#)
 HypConsolidate, [136](#)
 HypConsolidateAll, [137](#)
 HypConsolidateAllWithData, [138](#)
 HypCreateConnection, [139](#)
 HypCreateConnectionEX, [141](#)
 HypDeleteCalc, [143](#)
 HypDeleteMetadata, [143](#)
 HypDisconnect, [145](#)
 HypDisconnectEx, [146](#)
 HypDisplayToLinkView, [147](#)
 HypExecuteCalcScript, [148](#)
 HypExecuteMDXEx, [154](#)

[HypExecuteQuery](#), 156
[HypFindMember](#), 156
[HypFindMemberEx](#), 158
[HypForceCalculate](#), 159
[HypForceCalculateContribution](#), 160
[HypForceTranslate](#), 160
[HypFreeDataPoint](#), 161
[HypGetAncestor](#), 162
[HypGetChildren](#), 164
[HypGetColCount](#), 165
[HypGetColItems](#), 166
[HypGetConnectionInfo](#), 167
[HypGetDataPoint](#), 168
[HypGetGlobalOption](#), 171
[HypGetLinkMacro](#), 173
[HypGetPagePOVChoices](#), 174
[HypGetParent](#), 175
[HypGetPOVCount](#), 175
[HypGetPOVItems](#), 176
[HypGetRowCount](#), 177
[HypGetRowItems](#), 178
[HypGetSharedConnectionsURL](#), 179
[HypGetSheetOption](#), 179
[HypGetSourceGrid](#), 181
[HypGetSubstitutionVariable](#), 182
[HypInvalidateSSO](#), 184
[HypIsAncestor](#), 184
[HypIsAttribute](#), 185
[HypIsChild](#), 186
[HypIsConnectedToSharedConnections](#), 187
[HypIsDescendant](#), 189
[HypIsExpense](#), 189
[HypIsParent](#), 191
[HypIsUDA](#), 193
[HypKeepOnly](#), 194
[HypListCalcScripts](#), 195
[HypMenuVAboutt](#), 233
[HypMenuVAdjust](#), 234
[HypMenuVBusinessRules](#), 234
[HypMenuVCalculation](#), 235
[HypMenuVCellText](#), 235
[HypMenuVCollapse](#), 236
[HypMenuVConnect](#), 236
[HypMenuVCopyDataPoints](#), 237
[HypMenuVExpand](#), 237
[HypMenuVFunctionBuilder](#), 238
[HypMenuVInstruction](#), 238
[HypMenuVKeepOnly](#), 239
[HypMenuVMemberSelection](#), 240
[HypMenuVMigrate](#), 240
[HypMenuVOptions](#), 241
[HypMenuVPasteDataPoints](#), 241
[HypMenuVPivot](#), 242
[HypMenuVPOVManager](#), 243
[HypMenuVQueryDesigner](#), 243
[HypMenuVRedo](#), 244
[HypMenuVRefresh](#), 244
[HypMenuVRefreshAll](#), 245
[HypMenuVRefreshOfflineDefinition](#), 245
[HypMenuVRemoveOnly](#), 246
[HypMenuVRulesOnForm](#), 246
[HypMenuVRunReport](#), 247
[HypMenuVSelectForm](#), 247
[HypMenuVShowHelpHtml](#), 248
[HypMenuVSubmitData](#), 248
[HypMenuVSupportingDetails](#), 249
[HypMenuVSyncBack](#), 250
[HypMenuVTakeOffline](#), 250
[HypMenuVUndo](#), 251
[HypMenuVVisualizeinExcel](#), 251
[HypMenuVVisualizeinHVE](#), 252
[HypMenuVZoomIn](#), 252
[HypMenuVZoomOut](#), 253
[HypOpenForm](#), 197
[HypOtlGetMemberInfo](#), 198
[HypPivot](#), 199
[HypPivotToGrid](#), 200
[HypPivotToPOV](#), 201
[HypQueryMembers](#), 203
[HypRedo](#), 205
[HypRemoveConnection](#), 207
[HypRemoveOnly](#), 208
[HypResetFriendlyName](#), 209
[HypRetrieve](#), 210
[HypRetrieveAllWorkbooks](#), 210
[HypRetrieveRange](#), 211
[HypSetActiveConnection POV](#), 212
[HypSetAsDefault](#), 213
[HypSetBackgroundPOV](#), 214
[HypSetCellsDirty](#), 215
[HypSetColItems](#), 215
[HypSetConnAliasTable](#), 216
[HypSetConnectionInfo](#), 217
[HypSetGlobalOption](#), 219

- HypSetLinkMacro, [220](#)
- HypSetMenu, [220](#)
- HypSetPages, [221](#)
- HypSetPOV, [222](#)
- HypSetPOVItems, [223](#)
- HypSetRowItems, [223](#)
- HypSetSharedConnectionsURL, [224](#)
- HypSetSheetOption, [225](#)
- HypSetSubstitutionVariable, [226](#)
- HypSubmitData, [227](#)
- HypTranslate, [228](#)
- HypUndo, [229](#)
- HypUseLinkMacro, [229](#)
- HypZoomIn, [230](#)
- HypZoomOut, [231](#)
- VBA functions, working with
 - calling functions, [122](#)
 - declaring functions, [122](#)
 - dynamic link views, [122](#)
 - menu functions, [232](#)
 - migrating legacy VBA applications, [253](#)
 - parameters, [124](#)
 - return values, [125](#)
 - using functions in Smart View, [121](#)
- viewing
 - supporting detail, [83](#)
- Visual Explorer, [280](#)

W

- Web Analysis
 - formats, [91](#)
 - importing documents, [102](#)
- Within Selected Group option, [266](#)
- workbooks
 - shared, [25](#)

Z

- Zooming in and out, [28](#)