# Contents

Documentation Accessibility .......................................................... 11

Chapter 1. Introduction to Smart View ................................................ 13
   Overview .................................................................................. 13
   Smart View Components ......................................................... 13
   Ribbons ................................................................................. 13
   Smart View Panel ................................................................... 14

Chapter 2. Managing Data Source Connections ........................................ 17
   Connections ........................................................................... 17
   Shared Connections and Private Connections ................................. 17
   Connecting to Data Sources ..................................................... 17
   Disconnecting from Data Sources ............................................. 18
   Creating Private Connections ................................................ 19
   Saving Shared Connections as Private Connections ........................ 19

Chapter 3. Dimensions and Members .................................................... 21
   About Dimensions and Members .............................................. 21
   Selecting Members From the Member Selector .............................. 21
   Selecting Members From the POV Toolbar .................................. 23
   Selecting Members From the POV Toolbar in Essbase ..................... 25
   Entering Members in Free-Form Mode ....................................... 26
   Filtering by Attribute ............................................................. 26
   Filtering by Subsets .................................................................. 27
   Selecting Period-to-Date Members ............................................ 27
   Duplicate Member Names ....................................................... 28
   Member Perspective ............................................................... 29
   Aliases and Alias Tables ......................................................... 30
      Selecting Alias Tables ......................................................... 30
      Aliases from Different Alias Tables ....................................... 31
      Displaying Member Names and Their Aliases .......................... 31
   Member Information .............................................................. 31
   The POV Manager ................................................................. 32
Chapter 4. Data and Data Cells ........................................................ 35
Retrieving Data .............................................................................. 35
Submitting Data ............................................................................ 35
Calculating Data ............................................................................ 36
Calculating Data in Financial Management and Hyperion Enterprise .............................................................................. 36
Calculating Data in Essbase .......................................................... 37
Consolidating Data ......................................................................... 37
Working with Currencies ............................................................... 38
Translating Currencies in Financial Management and Hyperion Enterprise .............................................................................. 38
Changing Currency in Planning .................................................... 38
Adjusting Values in Data Cells ...................................................... 39
Data Perspective ............................................................................ 39
Drill-Through Reports .................................................................... 41
Linked Reporting Objects ............................................................... 41
Attaching a Linked Reporting Object to a Data Cell ....................... 42
Launching a Linked Reporting Object from a Data Cell .................. 43
Linked Partitions ........................................................................... 43
Cell Comments ............................................................................. 44
Cell Comments in Planning .......................................................... 44
Cell Comments in Financial Management ......................................... 45
Attachments .................................................................................. 46
Cell History .................................................................................... 47

Chapter 5. Ad Hoc Analysis .......................................................... 49
About Ad Hoc Analysis .................................................................... 49
Starting Ad Hoc Analysis ............................................................... 49
Excel Formulas in Ad Hoc Grids ..................................................... 50
Formatting Ad Hoc Grids ............................................................... 50
Using Smart View Formatting ....................................................... 50
Using Excel Formatting ................................................................... 51
Zooming In and Out ....................................................................... 51
Zooming In .................................................................................... 52
Zooming Out .................................................................................. 53
Setting a Default Zoom Level .......................................................... 53
Selecting Members to Display when Zooming .............................. 53
Enabling Double-Click Zooming .................................................... 54
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zooming and Formatting</td>
<td>54</td>
</tr>
<tr>
<td>Zooming Operations in Cells that Contain Formulas</td>
<td>54</td>
</tr>
<tr>
<td>Pivoting</td>
<td>55</td>
</tr>
<tr>
<td>Removing Selected Members From the Grid</td>
<td>55</td>
</tr>
<tr>
<td>Inserting Rows and Columns</td>
<td>55</td>
</tr>
<tr>
<td>Multiple Grids on a Worksheet</td>
<td>55</td>
</tr>
<tr>
<td>Creating Multiple-Grid Worksheets</td>
<td>56</td>
</tr>
<tr>
<td>Converting Ad Hoc Worksheets to Multiple-Grid Worksheets</td>
<td>56</td>
</tr>
<tr>
<td>Changing Connections in Multiple-Grid Worksheets</td>
<td>57</td>
</tr>
<tr>
<td>Multiple-Grid Example: Butterfly Report</td>
<td>57</td>
</tr>
<tr>
<td>Cascading Reports and Ad Hoc Grids</td>
<td>57</td>
</tr>
<tr>
<td>Substitution Variables</td>
<td>58</td>
</tr>
<tr>
<td>Chapter 6. Data Forms</td>
<td>61</td>
</tr>
<tr>
<td>Working with Forms in Excel</td>
<td>61</td>
</tr>
<tr>
<td>Opening Forms in Excel</td>
<td>61</td>
</tr>
<tr>
<td>Excel Formulas in Forms</td>
<td>62</td>
</tr>
<tr>
<td>Planning Forms</td>
<td>62</td>
</tr>
<tr>
<td>Planning Form Behavior in Smart View</td>
<td>62</td>
</tr>
<tr>
<td>Saving Ad Hoc Grids as Forms</td>
<td>63</td>
</tr>
<tr>
<td>Performing Ad Hoc Analysis in Planning Forms</td>
<td>63</td>
</tr>
<tr>
<td>Financial Management Data Forms</td>
<td>63</td>
</tr>
<tr>
<td>About Financial Management Members</td>
<td>63</td>
</tr>
<tr>
<td>Adding Financial Management Members</td>
<td>64</td>
</tr>
<tr>
<td>Using Financial Management Linked Forms</td>
<td>64</td>
</tr>
<tr>
<td>Chapter 7. Smart View General Operations</td>
<td>65</td>
</tr>
<tr>
<td>Smart View Operations</td>
<td>65</td>
</tr>
<tr>
<td>Using Undo and Redo</td>
<td>65</td>
</tr>
<tr>
<td>Copying and Pasting</td>
<td>66</td>
</tr>
<tr>
<td>Importing Metadata into Copied Worksheets</td>
<td>66</td>
</tr>
<tr>
<td>Copying Data Between Excel, Word, and PowerPoint</td>
<td>67</td>
</tr>
<tr>
<td>Enabling Automatic Column Width Adjustment</td>
<td>68</td>
</tr>
<tr>
<td>Sheet Information</td>
<td>68</td>
</tr>
<tr>
<td>Enabling and Disabling Smart View</td>
<td>69</td>
</tr>
<tr>
<td>Disabling Smart View Within Smart View</td>
<td>69</td>
</tr>
<tr>
<td>Enabling and Disabling Smart View in Microsoft Office</td>
<td>69</td>
</tr>
<tr>
<td>Searching in Smart View</td>
<td>69</td>
</tr>
<tr>
<td>Shared Workbooks</td>
<td>69</td>
</tr>
<tr>
<td>Printing POV Members in Header and Footer</td>
<td>69</td>
</tr>
</tbody>
</table>
## Chapter 8. Smart Query

- About Smart Query ................................................................. 71
- Creating a Smart Query ................................................................. 71
  - Defining Sets ............................................................................... 72
  - Defining Set Filters .................................................................... 73
  - Building the Smart Query ............................................................. 73
  - Completing the Smart Query .......................................................... 75
- Opening a Smart Query ............................................................... 76
- Copying and Pasting ................................................................. 76
  - Copying Smart Query Definitions in Excel ..................................... 76
  - Copying Smart Query Sets and Filters .......................................... 77
  - Copying Smart Query Reports to Word and PowerPoint .................. 77

## Chapter 9. Smart Slices

- About Smart Slices ........................................................................ 79
- Creating Reports with Smart Slices ................................................... 79
  - Deleting Reports ........................................................................ 81
  - Sliders ........................................................................................ 81
- Smart Slices, Ad Hoc Analysis, and Forms .......................................... 82
- Creating Smart Slices ................................................................. 83
  - Setting Smart Slice Data Boundaries ............................................ 83
  - Setting Smart Slice Preferences .................................................... 84

## Chapter 10. The Query Designer and MDX Queries

- The Query Designer ................................................................. 85
- Creating Queries ........................................................................... 85
- Editing Queries and Rerunning Reports .............................................. 86
- Filtering Data ................................................................................ 87
- Analyzing Time-Related Data in Query Designer ................................ 87
- MDX Queries ................................................................................ 88

## Chapter 11. Task Lists

- Task Lists .................................................................................... 89
- Working with Tasks from the Smart View Panel .................................. 89
  - Opening a Task List ..................................................................... 89
  - Viewing the Task List .................................................................. 90
  - Executing a Task .......................................................................... 90
  - Completing a Task ....................................................................... 91
- Creating Task List Reports ............................................................ 91
- Integrating Task Lists with Microsoft Outlook .................................... 91
# Chapter 12. Smart View and Planning

- Planning Approvals ................................................................................. 93
  - Changing Planning Unit Status ......................................................... 93
- Finding Planning Units ............................................................................ 94
- Planning Unit Promotional Path ............................................................ 95
- Planning Unit Annotations ..................................................................... 96
- Out of Office Assistant .......................................................................... 96
- Monitoring Planning Job Status ............................................................ 97
- Searching for a Page in Planning .......................................................... 97
- Copying Versions .................................................................................. 97
- Composite Forms .................................................................................. 98
- Working with Planning Business Rules .................................................. 99
  - Launching Business Rules in Excel ................................................... 99
  - Entering Runtime Prompts ............................................................... 99
  - Executing the Calculate Form and Calculate Currencies Business Rules ............................................................... 100
- Spreading Data for Time Periods .......................................................... 101
  - Spreading Data with Cell Locking ..................................................... 101
  - Spreading Values Using Grid Spread ................................................ 102
  - Spreading Values Using Mass Allocation ........................................ 102
- Member Formula .................................................................................. 103
- Supporting Detail .................................................................................. 103
  - Adding Supporting Detail .................................................................. 103
  - Working with the Supporting Detail Hierarchy .................................. 104
  - Viewing or Changing Supporting Detail .......................................... 105
  - Synchronizing Supporting Detail with Essbase .................................. 105
- Setting Planning Preferences ................................................................. 106
- Working Offline .................................................................................... 106
  - Taking Forms Offline ....................................................................... 106
  - Working Offline with Forms ............................................................... 107
  - Synchronizing Data to the Planning Server ....................................... 108
  - Refreshing the Offline Form Definition and Data ............................ 109

# Chapter 13. Smart View and Reporting and Analysis

- Importing Reporting and Analysis Documents ...................................... 111
- Editing and Refreshing Documents ....................................................... 112
- Refreshing Reporting and Analysis Documents ...................................... 112
- Financial Reporting and Web Analysis Import Formats ......................... 113
- Importing Interactive Reporting Documents ......................................... 114
  - Importing Interactive Reporting Documents into Excel ...................... 115
Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.
Overview

Oracle Hyperion Smart View for Office provides a common Microsoft Office interface for Oracle Essbase, Oracle Hyperion Financial Management, Oracle Hyperion Planning, Oracle Hyperion Enterprise Performance Management Workspace, Oracle Hyperion Reporting and Analysis, Oracle Hyperion Financial Close Management, and Oracle 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.

Smart View Components

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 5), 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
- Other: If the administrator has installed and configured your Smart View system with
  extensions, there may be other ribbons; for example, Smart Query or Oracle Hyperion
  Disclosure Management.

**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 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, forms, and tasks - that you can click to establish
  a connection.

- **Shared Connections**: A drop-down menu of available connections from Oracle 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.

- **Action Panel**: A list of operations available based on the selection in the shared connection,
  private connection, or task list tree list.
Other: If the administrator has installed and configured your Smart View system with extensions, there may be other panels; for example, Smart Query or Crystal Ball EPM workbooks.
Connections

You connect to data sources, manage your connections, and open grids, 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.

Shared Connections and Private Connections

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. 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

Except for Essbase connections, you can connect to one data source per worksheet.
To connect to a data source:

1. From the Smart View ribbon, click **Panel**.

2. From Smart View Home or from the menu displayed when you click the arrow next to 
   - 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 19.

3. In **Connect to Data Source**, enter your user name and password for the data source.

4. In the Smart View Panel tree list, double-click the item - 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 the arrow next to 
   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 the arrow next to 
     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 Panel.
2. From the Smart View panel, click the arrow next to , and then select Private Connections.
3. In the text box, 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/hfmoofficeprovider/hfmoofficeprovider.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
   - Oracle Hyperion Strategic Finance: http://servername:port/StrategicPlanning/SmartView
4. Click .

Deleting the List of Private Connections

To delete the entire list of private connections, click the arrow next to , and then 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.
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.

**Optional:** From **Save as Private Connection**, edit the name and description of the connection.

Click **OK**.

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
Dimensions and Members

About 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.

You can select members for the grid from the Member Selection dialog box available from the data source ribbon, from the POV toolbar, or by entering the member name using free-form mode.

Selecting Members From the Member Selector

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.
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:
   - Select a dimension or member on the grid, and then from an Ad Hoc data source ribbon, click **Member Selection**.
   - From an open dialog box enabled for member selection, click **Member Selection**.
   - On a blank worksheet, from the Smart View Panel, right-click a cube name and select **Member Selection**. Use this method to select members for functions and references (see Chapter 15, “Functions”).

2. From **Member Selection**, to change the dimension, click the **Dimension Selector** button (for example, **Year**) 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 ✽ and select one of these filters (filter options may vary by data source type):
   - **Children** to select only the children of the selected member
   - **Descendants** 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 26, “Filtering by Subsets” on page 27, and “Selecting Period-to-Date Members” on page 27.

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 **Optional:** If this is the first member selection that you make in a blank worksheet, select one of these buttons:
   • ![Icon](image) to display the selected members vertically, in a column
   • ![Icon](image) to display the selected members horizontally, in a row

8 **Click OK**.

   The members selected are displayed in the grid.

9 **From the ribbon**, click **Refresh** to update the data to correspond to the selected members.

---

### Selecting Members From the POV Toolbar

For Essbase connections, see “Selecting Members From the POV Toolbar in Essbase” on page 25.

The POV is the default starting point for dimensions in a data source connection. From the POV toolbar, 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 32.

**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 Toolbar onto the Grid

➢ To select dimensions and members from the POV toolbar:

1. Do one of the following:
   - Enter the name of a member over its corresponding dimension on the POV toolbar, and then click Refresh on the POV toolbar.
   - Click the down arrow next to a dimension on the POV toolbar and select members as described in “Selecting Members From the Member Selector” on page 21.

2. From the POV toolbar, 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 toolbar for editing, right-click its cell and drag it to the POV toolbar.

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 Toolbar

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

Note: On the Essbase ribbon, the POV button operates differently. See “Selecting Members From the POV Toolbar in Essbase” on page 25.

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
Selecting Members From the POV Toolbar in Essbase

Note: The information in this section applies only to Essbase 11.1.2.1.102 and later connections. If you are using earlier Essbase releases, the POV button functions as described in Hiding the POV Toolbar.

In Essbase, you can choose whether to display all members on the grid and hide the POV toolbar or to display the POV toolbar containing the POV members.

By default, all members are displayed on the grid, and the POV toolbar is hidden. In this mode, you can format POV member cells the same as you do other member and data cells, and select members from the ribbon.

If you choose to display the POV toolbar containing the POV members, you can use the POV toolbar to select members and move them to and from the grid as described in “Selecting Members From the POV Toolbar” on page 23.

To display the POV toolbar, from the Essbase ribbon, select **POV**. To hide the POV toolbar and display all members on the grid, toggle **POV** off.

Note: On worksheets that contain multiple grids, the **POV** button is disabled. In these worksheets, the POV toolbar is hidden, and all members are displayed the grid.

In Figure 3, the **POV** button is toggled off, the POV toolbar is hidden, and all members are on the grid.

Figure 4 shows the POV toolbar on the grid. **Measures** and **Year** are displayed on the grid; POV members **Product**, **Market**, and **Scenario** are displayed on the POV toolbar.
Entering Members in Free-Form Mode

If you are familiar with the dimensions and members of your database, you can enter their names directly into cells using free-form mode. You can use aliases from the alias table associated with the current grid in free-form mode. If you enter an alias from a different alias table, it will revert to the alias from the current alias table.

After connecting to a data source, you can enter member names as follows:

- By entering a member name in a blank cell
- By replacing a member name in a cell with a different member from the same dimension

You can still use the POV, member selection, and other ad hoc operations in free-form grids. See Chapter 17, “Free-Form Mode.”

Filtering by Attribute

Data source types: Essbase

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

1. To filter by attribute:
   1. Select an attribute dimension on the grid, and then open Member Selection as described in “Selecting Members From the Member Selector” on page 21.
   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 + Add 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 attribute member on the grid, and then open Member Selection as described in “Selecting Members From the Member Selector” on page 21.

2 Click + Add 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 + Add.

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

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

8 Click + Add.

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, and then open Member Selection as described in “Selecting Members From the Member Selector” on page 21.
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.

**Duplicate Member Names**

**Data source types:** Essbase

Different members may have identical names. For example, a database may have two members named “New York,” one for New York City and one for New York State. Both members can appear as “New York” in the grid, but if you want to distinguish between them, you can display their qualified names instead. Qualified names include the member name and the names of its ancestors to the level that uniquely defines the member; for example, [Market].[New York].

To display the qualified names of duplicate members:

1. From the Smart View ribbon, select Options, and then select Member Options in the left panel.
2. From the Member Name Display drop-down menu, select Distinct Member Name.
3. Click OK.
4. Refresh the grid.

Duplicate members in the grid are displayed as qualified names. In this example, New York City is displayed as [East].[New York]. New York State is displayed as a sibling of East, West, and South: [Market].[New York]:

---

28 Dimensions and Members
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 then 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 then 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 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. A range can be specified only for continuous independent dimensions (“Year” is an example). For discrete independent dimensions, you can make only one selection.

10. Click OK.
Aliases and Alias Tables

**Data source types:** Essbase, Planning, Financial Management

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

Aliases are alternate names for database member names. Database member names are often stock numbers or product codes; their aliases can be more descriptive. For example, in the Sample Basic database, the alias name for the database member 100 is Colas. Aliases are stored in alias tables as part of a database. Dimensions can be associated with multiple alias tables.

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

### Selecting Alias Tables

If more than one alias table has been created in the database, you can select an alias table for the current worksheet or for a private 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.

- 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.

    The new alias table is applied automatically.

#### Selecting an Alias Table for the Connection

You can select an alias table for private connections only. If you want to select an alias table for a shared connection, first save the shared connection as a private connection. See “Saving Shared Connections as Private Connections” on page 19.

An alias table selected for a private connection is permanent until changed and will be used each time you use this connection.

- To select an alias table for the connection:
  1. From a worksheet, connect to a data source.
  2. In the Smart View Panel private connections, right-click a connection name and select **Set Alias Table**.
  3. Select an alias table for the connection.

    The new alias table is applied the next time you open the connection.
Aliases from Different Alias Tables

Data source types: Essbase

If you enter a name from an alias table that is not associated with the current grid, its corresponding alias from the alias table that is associated with the current grid is displayed after you refresh. For example, if you enter Qtr1 into a grid that is associated with the Long Names alias, then after you refresh, Quarter1 is displayed.

Displaying Member Names and Their Aliases

Data source types: Essbase

If you are connected to an Essbase data source, you can display member names and their aliases from the currently selected alias table together in the same row.

Note: This feature applies only to row members and not to column members.

To display both member names and aliases:
1. From the Smart View ribbon, select Options, and then Member Options in the left panel.
2. Under General, for Member Name Display, select Member Name and Alias.

For row members, both member names and their corresponding aliases are displayed. In this example, Product database member names are shown in column A, and their aliases in column B.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>Year</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>Measures</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>Colas</td>
<td>28473</td>
</tr>
<tr>
<td>4</td>
<td>200</td>
<td>Root Beer</td>
<td>27954</td>
</tr>
<tr>
<td>5</td>
<td>300</td>
<td>Cream Soda</td>
<td>25799</td>
</tr>
<tr>
<td>6</td>
<td>400</td>
<td>Fruit Soda</td>
<td>21301</td>
</tr>
<tr>
<td>7</td>
<td>Diet</td>
<td>Diet Drinks</td>
<td>28826</td>
</tr>
<tr>
<td>8</td>
<td>Product</td>
<td>Product</td>
<td>103527</td>
</tr>
</tbody>
</table>

Member Information

Data source types: Essbase

You can view detailed information about any member on the grid.

To view member information:
1. Select a member in the grid.
From the data source ad hoc ribbon, select Member Information. Information is displayed on the following tabs. Only the tabs that are applicable to the member and connection are displayed.

- **Information**: A list of general information about the member such as dimension, level, generation, and so forth
- **Aliases**: A list of alias tables and corresponding aliases associated with the member
- **Attributes**: A table of the dimensions, members, and types of attributes associated with the member
- **Formula**: The formula associated with the member
- **Comments**: A list of comments associated with the member
- **User Defined Attributes**: A list of user defined attributes (attributes of the member defined by the administrator)

Optional: To save the information in an Excel file, select Save.

Click Close.

**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
- Edit a POV
- Delete a POV

**Selecting Members for the Default POV**

In the POV Manager, you can select members as follows:

- To use as a default POV for the ad hoc grids of a given connection
- For the background POV for dimensions when you use functions

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 then select the members that you want to use for the POV. See “Selecting Members From the Member Selector” on page 21.
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 you start working on the ad hoc grid, select or change members as described in “Selecting Members From the Member Selector” on page 21.

### 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. You must paste the copied POV to an unconnected worksheet; otherwise the POV has no effect.

➢ To copy and paste a POV:

1 From the Smart View ribbon, select Function, and then Manage POV.

2 In the left window of POV Manager, expand Active, and then select the application connection that you want to copy.

3 From the POV Manager toolbar, click Member Selector, and then 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) that 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 is saved in a workbook:

1 From the Smart View ribbon, select Function, and 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.
Retrieving Data

Data source types: all

In Excel, you can retrieve and refresh data for the current worksheet or for all worksheets in the workbook. In Essbase worksheets that contain multiple grids, you can also refresh only a selected range of cells (see “Multiple Grids on a Worksheet” on page 55. To refresh the entire current worksheet, from any ribbon, click Refresh. To refresh all worksheets in the current workbook, click the arrow next to Refresh, and then select Refresh All Worksheets.

In Word or Powerpoint, when you click Refresh, all data points copied into the document or presentation are refreshed.

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 forms. You can submit the changes made while you are unconnected after you reconnect.
For data sources other than Essbase, you must refresh the grid before modifying the data when you are in free-form mode. For Essbase connections, you can submit data without first refreshing. See “Retrieving Data” on page 35.

Note: In worksheets that support multiple grids, you can submit data only for one grid at a time. If you try to submit data for more than one grid or for the entire worksheet, no data is submitted.

If you are submitting data from forms:

- In Planning, Financial Management, or Hyperion Enterprise forms, you can lock any cell or range of cells to protect the data until the data is refreshed or submitted. In Financial Management, locking the cell does not lock the actual data cube 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 form definition. This behavior may happen if 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 form definition are saved. This behavior applies to both cells and supporting detail changes, and also applies to both online and offline modes.

To submit data:

1. Connect to the data source. If you are using Essbase, skip to step 3. Otherwise, continue with step 2.
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.

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 Form and Calculate Currencies Business Rules” on page 100.

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

Converting currencies is called “translating data” in Financial Management. You can translate data from the entity’s input currency to any other currency 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

In forms enabled for currency conversion, you can enter data in a currency other than the base currency of a cell. 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 form should support multi-currency.

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

1. In a 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 form is saved, and the form is enabled for multiple currencies, the data value is displayed in the currency you selected.

## 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.

1. 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.

## 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.

1. To specify data perspective:
   1. From the Essbase ribbon, select 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 5 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 5 Data Perspective: Reality](image)

Figure 6 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 6 Data Perspective: Last](image)

Figure 7 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.
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 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 134).

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.

   If you want to display a list of available drill-through reports whenever you mouse over a cell, select Display Drill Through Report ToolTips on the Advanced page of the Options dialog box.

2. From the data source ribbon, select Drill-through to display the list of reports associated with the cell.

3. Select a report and click Launch.

Linked Reporting Objects

Data Source types: Essbase
A linked reporting object is a cell note, external file, or URL that is linked to a data cell in an Essbase database, and which can be retrieved by Smart View users in Excel.

You can set a cell style (see “Cell Styles” on page 134) to identify cells that are associated with linked reporting objects.

See also “Linked Partitions” on page 43.

**Attaching a Linked Reporting Object to a Data Cell**

You can attach one or more linked reporting objects to a data cell.

1. Select a data cell.
2. From the Essbase ribbon, select Linked Objects.
3. From Linked Reporting Objects, click ![Attach](attach.png) and select one of the following:
   - **Cell Note** to attach an annotation to the data cell
   - **File** to attach an external file to the data cell
   - **URL** to attach a URL to the data cell

   The dialog box appropriate to your selection is displayed.

4. Enter information as follows:
   - **Cell Note**: Enter text for the note. Then click Close. The first few words of the note are displayed in the Description column of the Linked Reporting Objects list.
   - **File**: Use the Browse button to navigate to the file that you want to attach to the data cell. You can add a brief description for the file. Then click Close.
   - **URL**: In URL, enter a URL for a website, a network or local directory, or a document in a network or local directory. You can add a brief description for the URL. Then click Close.

5. Repeat the procedure to attach other linked reporting objects as needed.

The objects that you created are displayed in the Linked Reporting Objects list as shown here:
To edit or delete a linked reporting object, use the **Edit** button or the **Delete** button. Deleting an object removes it from the database.

### Launching a Linked Reporting Object from a Data Cell

To launch a reporting object from a data cell:

1. Select the data cell that is associated with the linked object that you want to launch.
2. From the Essbase ribbon, select **Linked Objects** to display **Linked Reporting Objects**.
3. From **Linked Reporting Objects**, select the linked object to launch.
4. Click ![link](image). The linked reporting object launches as follows:
   - Cell notes are displayed in the Cell Note dialog box.
   - Files are opened.
   - URL objects are opened in the default web browser.
   - Linked partitions - see [Linked Partitions](#)

### Linked Partitions

A *linked partition* connects two databases by means of a data cell. Using a data cell associated with a linked partition, you can navigate from the database connected to the current grid to a second database. Because the two databases may have different dimensions, you can see the data in different contexts. When you launch a linked partition, a new spreadsheet that displays the dimensions from the linked database opens. From there, you can drill down into the dimensions of the linked database.
To launch a linked partition:

1. Select a data cell associated with a linked partition.
2. From the Essbase ribbon, select **Linked Objects**.
3. From **Linked Reporting Objects**, select the linked partition (displayed as **Linked** in the list).
4. Click 

   The linked partition is launched in a new spreadsheet. From this spreadsheet, you can drill down to data in the linked database.

**Cell Comments**

Data source types: Planning, Financial Management

Comments can be added to data cells in Planning and Financial Management.

**Cell Comments in Planning**

You can add one or more comments per data cell; each data cell can contain comments from multiple users. Depending on the permission level assigned to you by the administrator, you may be able to do any of the following in a data cell:

- Add comments.
- View the comments that you and other users have added.
- Delete comments that you have entered. You cannot delete comments added by other users.

Cells that contain comments can be associated with a cell style (See “**Cell Styles**” on page 134).

To add comments to a data cell:

1. Select one data cell or a range of data cells in an ad hoc grid.
2. From the Planning or Planning Ad Hoc ribbon, select **Cell Actions** and then **Comments**.
3. From **Comments**, click 

   If you selected a range of cells in step 1, you can either enter comments for one cell at a time or apply a comment to all selected cells.
   - To enter a comment for one cell, select the cell from the drop-down menu.
   - To enter a comment for all selected cells, select **Apply to all selected cells**.
4. In the righthand field, enter a comment. If you want to format the comment, use HTML tags.
5. Click 

   The comment is displayed in the list of comments to the left of the comment field. This list contains comments entered by all users.
Cell Comments in Financial Management

Cells in ad hoc grids, forms, Smart Slices, and the Query Designer can contain multiple comments. Comments within a cell are differentiated by their labels, which are defined in Financial Management. You cannot create labels in Smart View.

You select from these defined labels to add and view in Smart View. You cannot edit or delete labels, but you can edit and delete comments.

Viewing and Adding Cell Comments

1. Select a cell in the grid.
2. From the ribbon, select Cell Comments.
   Any comments currently associated with the cell are displayed in the Cell Comments list.
3. To add a comment, from Cell Comments, select a label from the drop-down menu.
4. Click OK.
   The selected cell text label is added to the list of labels.
5. Click in the field under Cell Text and add a comment.
6. Repeat as needed to add other labels.
7. Click OK.
   The labels in the list are now associated with the cell.

Editing and Deleting Cell Comments

1. Select a cell in the grid.
2. From the HFM ribbon, select Cell Comments.
3. From Cell Comments, select a comment in the list and then do one of the following:
   - To edit the comment, select Edit. After editing, click OK.
   - To delete the comment from the cell, select the Delete button. The comment is removed from the list. Deleting a comment removes it only from the selected cell; it remains available for selection from the drop-down menu.
4. Click OK.
In Financial Management, you can use functions HsSetText and HsGetText to submit and retrieve cell text to and from the data source. See Chapter 15, “Functions.”

**Attachments**

**Data source types:** Planning

Documents can be attached to individual data cells by way of URLs. Each data cell can contain multiple documents attached by one or more users. Depending on the permission level assigned to you by the administrator, you may be able to do any of the following in a data cell:

- Attach documents.
- View the documents that you and other users have attached.
- Edit and delete documents that you have attached. You cannot edit or delete documents attached by other users.

Cells that contain attachments can be associated with a cell style (see “Cell Styles” on page 134).

1. To attach documents to a data cell:
   1. Select one data cell or a range of data cells in a Planning ad hoc grid.
   2. From the Planning or Planning Ad Hoc ribbon, select **Cell Actions** and then **Attachment** to display the **Attachments** dialog box.
   3. If you selected a range of cells in step 1, you can either attach documents to one cell at a time or attach the same document to all selected cells.
      - To attach a document to one cell, select the cell from the drop-down menu.
      - To attach one document to all selected cells, select **Apply to all selected cells**.
   4. Select 👇.
   5. Click in the cell under **Description** to add a brief description.
   6. Click in the cell under **Reference** to enter the URL to the document that you want to attach.
   7. Repeat as necessary to add attachments.
   8. To save your attachment selections, click 📋.
   9. **Optional:** To edit a **Reference** entry, click 📝, edit the reference, and then click ✔.
   10. **Optional:** To delete an attachment, select the attachment in the list and click 🗑.
   11. Click Close.
Launching Attachments

To launch an attached document in a new browser:

1. Select the cell that contains the attachment.
2. From the Planning or Planning Ad Hoc ribbon, select Cell Actions and then Attachment.
3. Click ✉️.

Cell History

Data source types: Planning (ad hoc only)

You can view the history of changes made to a data cell or range of data cells. For each change listed, the user who made the change, date, old value, and new value are displayed.

**Note:** Cell history is available only if enabled by the Planning administrator.

To display cell history:

1. Select one data cell or a range of cells in a Planning ad hoc grid.
2. From the Planning Ad Hoc ribbon, select Cell Actions and then Cell History to display the Change History screen.
3. If you selected multiple cells in step 1, select one cell at a time from the drop-down menu in Change History to view its history.
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, 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, 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 form that has been enabled for ad hoc by your administrator. From the data provider ribbon, click **Analyze**. If **Analyze** is disabled, the 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 entering dimension and member names directly into cell to design and create an ad hoc grid. See Chapter 17, “Free-Form Mode.”

**Excel Formulas in Ad Hoc Grids**

You can associate Excel formulas with member cells in ad hoc grids and set a cell style (see “**Cell Styles**” on page 134) to identify such cells.

By default, formulas are preserved when you perform ad hoc operations, except for **Pivot**. To achieve faster execution of queries, you can disable the preservation of formulas and comments. However, if you select this option, formulas are overwritten when you perform ad hoc operations.

To specify preservation of formulas in ad hoc grids:

1 From the Smart View ribbon, click **Options**, and then **Member Options** in the left panel.

2 Do one of the following:

   - To preserve formulas in ad hoc grids, select **Preserve Formulas and Comments in ad hoc operations (except pivot)**.
   - To disable preservation of formulas, clear **Preserve Formulas and Comments in ad hoc operations (except pivot)**. Do this only if you do not need to preserve formulas and you want faster execution of queries.

3 Click **OK**.

**Formatting Ad Hoc Grids**

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

**Using Smart View Formatting**

Smart View formatting consists of formatting selections made in the Cell Styles and Formatting pages of the Options dialog box.

To set Smart View formatting options:

1 From the Smart View ribbon, click **Options**.
2 From Options, to set cell styles, select Cell Styles in the left pane.

3 To set other Smart View formatting options, select Formatting from the left pane (Use Excel Formatting on this page is not a Smart View formatting option).

See Chapter 14, “Smart View Options” for descriptions of options.

4 Click OK.

➤ To apply Smart View formatting selections to data cells created by zooming:

1 From the grid, select a formatted data cell.

2 From the Essbase, Planning Ad Hoc, or Hyperion Enterprise ribbon, select Preserve Format.

Using Excel Formatting

If you use Excel formatting, your formatting selections, including conditional formatting, are applied and retained on the grid when you refresh or perform ad hoc operations.

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.

➤ To use Excel formatting on ad hoc grids:

1 From the Smart View ribbon, select Options.

2 From Options, select Formatting from the left pane.

3 Select Use Excel Formatting.

4 Optional: To copy parent cell formatting to zoomed-in cells, select Move Formatting on Operations. With this option selected, formatting also stays with members when you pivot.

5 Click OK.

Excel Formatting and Merged Cells

To preserve the merged cell formatting during ad hoc operations (except Pivot), you must select both Use Excel Formatting and Preserve Formulas and Comments in ad hoc operations options.

To replicate merged cell formatting during ad hoc operations (except Pivot), you must select all of Use Excel Formatting, Preserve Formulas and Comments in ad hoc operations and Format Fill options.

Zooming In and Out

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise
You can zoom in on members in the grid to display data for their children and descendents. In Essbase connections, you can also zoom to display data for the following:

- Members of the same level, same generation, or sibling level as the selected member
- Members that are defined by the formula of the selected member

**Zooming In**

In Essbase, you can zoom in on a range of cells. In other data sources, you can zoom in on one cell at a time.

**Zooming in to the Default Level**

To zoom in to the default zoom level, which is specified as described in “Setting a Default Zoom Level” on page 53, do one of the following:

- Select a member, and then click **Zoom In** on the data source ribbon. In Essbase, you can select a range of members.
- Double-click a member (double-click zooming must be enabled; see “Enabling Double-Click Zooming” on page 54).

**Zooming in to a Selected Level**

To zoom in to a selected level:

1. Select a member. In Essbase, you can select a range of members.
2. From the data source ribbon, click the down arrow next to **Zoom In**, and then select one of the following options.
   - **Next Level** to retrieve data for the children of the selected members
   - **All Levels** to retrieve data for all descendants of the selected members
   - **Bottom Level** to retrieve data for the lowest level of members in a dimension
   - **Same Level** to retrieve data for all members at the same level as the selected member (Essbase only)
   - **Sibling Level** to retrieve data for the siblings of the selected members (Essbase only)
   - **Same Generation** to retrieve data for all members of the same generation as the selected members (Essbase only)
   - **Formulas** to retrieve data for all members that are defined by the formula of the selected member. The formula can be a member equation or a consolidation to the parent (Essbase only)

**Note:** When you zoom in on a page dimension, the page dimension is pivoted to a row dimension.
Zooming Out

Zooming out collapses the view according to the Zoom In Level option specified as described in “Setting a Default Zoom Level” on page 53.

➤ To zoom out:
1  Select a member. In Essbase, you can select a range of members.
2  From the data source ribbon, click Zoom Out.

Setting a Default Zoom Level

You can specify a default level for zooming operations. This setting applies to the Zoom In button and to double-click zooming if it is enabled (see “Enabling Double-Click Zooming” on page 54).

➤ To set a Zoom In default level:
1  From the Smart View ribbon, select Options, and then select Member Options in the left panel.
2  From the Zoom In Level drop-down menu, select a level described in “Zooming In” on page 52.
3  Click OK.

Selecting Members to Display when Zooming

You can set options to specify which members are retained and displayed as you zoom in and out.

➤ To set member display option for zooming:
1  From the Smart View ribbon, select Options, and then select Member Options in the left panel.
2  Under Member Selection, select:
   ● Include Selection to display both the selected member and the members retrieved as a result of zooming. For example, zooming in on the selected member Qtr1 retrieves data for Jan, Feb, Mar, and Qtr1. If not selected, only the members retrieved as a result of the zoom are displayed: Jan, Feb, and Mar.
   ● Within Selected Group to zoom in only on the selected group of cells, leaving the 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. (This setting also applies to Keep Only and Remove Only.)
   ● Remove Unselected Groups to remove all dimensions and members except the selected member and the members retrieved as a result of zooming.
3  Click OK.
Enabling Double-Click Zooming

If double clicking for ad hoc operations is enabled, you can zoom in to the default zoom level and zoom out by double clicking in a member cell.

1. To enable double clicking for zooming:
   1. From the Smart View ribbon, select Options, and then select Advanced in the left panel.
   2. Under Mode, select Double click for Operations.
      If you do not select Double click for Operations, then double-clicking retains Excel functionality and puts the cell into edit mode.
   3. Click OK.

   Note: In blank worksheets, double-clicking the first time retrieves the default grid and thereafter zooms in or out.

Zooming and Formatting

You can apply the formatting of the cell that you zoom in on to the cells that are created by zooming. For Smart View formatting, this ability applies to data cells. For Excel formatting, this ability applies to member cells.

Excel Formatting

To apply Excel formatting selections to member cells created by zooming, from the Smart View ribbon, select Options. Then select all these options:

- From the Formatting page, select Use Excel Formatting
- From the Member Options page, select Preserve Formulas and Comments in ad hoc operations (except Pivot)
- From the Member Options page, select Formula Fill

Zooming Operations in Cells that Contain Formulas

Data Sources: Essbase

If member cells are associated with formulas, you can propagate these formulas to the members retrieved as a result of zooming in. For example, if member Qtr1 is associated with a formula, then the formula can be propagated to Jan, Feb, and Mar when you zoom in on Qtr1.

1. To propagate formulas:
   1. From the Smart View ribbon, click Options, and then select Member Options in the left panel.
   2. Under Comments and Formulas, ensure that Preserve Formulas and Comments in ad hoc operations (except pivot) is selected.
3 Select Formula Fill.
4 Click OK.

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 that 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.

Removing Selected 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 that 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 that 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.

Inserting Rows and Columns
In ad hoc grids, you can insert calculating and non-calculating columns and rows within or outside the grid. Inserted rows and columns, which may contain formulas, text, or Excel comments, are retained when you refresh or zoom in.

Always refresh the grid before inserting rows or columns.

Multiple Grids on a Worksheet
Data source types: Essbase
In Essbase, you can create multiple grids on one worksheet. These grids can be connected to the same data source or to different Essbase data sources. You can retrieve data in these grids and shift them on the worksheet.

Note the following limitations in worksheets that support multiple grids:

- You can submit data only for one grid at a time. If you try to submit data for more than one grid or for the entire worksheet, no data is submitted.
- You cannot set a cell style for dirty cells.
- You cannot enter comments.
- These buttons on the Essbase ribbon are disabled:
  - Undo
  - Redo
  - Pivot to POV
  - POV

**Creating Multiple-Grid Worksheets**

Data source types: Essbase

1. To create a multiple-grid worksheet:
   1. In Excel, connect to an Essbase data source.
   2. From any location in the worksheet, select a range of cells (You must select a range rather than only one cell).
   3. From the Smart View Panel, right click an application, and then select **Ad Hoc Analysis**.
   4. When prompted to change the worksheet to support multiple grids, select **Yes**.
   5. To create a second grid on the worksheet:
      a. Select a different range of cells.
      b. From the Smart View Panel, right click an application, and then select **Ad Hoc Analysis**.
   6. Repeat step 5 as necessary to add grids to the worksheet.

**Converting Ad Hoc Worksheets to Multiple-Grid Worksheets**

1. To convert an existing ad hoc worksheet to a worksheet that supports multiple grids:
   1. From any location in the worksheet, select a range of cells (You must select a range rather than only one cell).
   2. From the Smart View Panel, right click the application, and then select **Ad Hoc Analysis**.
Changing Connections in Multiple-Grid Worksheets

To change the connection of a grid in a multiple-grid worksheet:

1. In the grid whose connection you want to change, select a range of cells (You must select a range rather than only one cell).
2. Using Excel Name Manager, delete the associated named range.
3. From the Smart View Panel, right click the application to connect to, and then select Ad Hoc Analysis.

Multiple-Grid Example: Butterfly Report

Data source types: Essbase

Typically, Smart View grids consist of member names on rows above and columns on the left of the data grid. Using the range retrieval capabilities of worksheets enabled for multiple grids, you can create grids with different layouts.

For example, you can create “butterfly” reports, with a column of members between two columns of data cells.

Figure 8  Butterfly Report

<table>
<thead>
<tr>
<th>Diet Cola</th>
<th>East</th>
<th>Jan</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Budget</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$190.00</td>
<td>Sales</td>
<td>$200.00</td>
</tr>
<tr>
<td>$80.00</td>
<td>COGS</td>
<td>$84.00</td>
</tr>
<tr>
<td>$110.00</td>
<td>Margin</td>
<td>$116.00</td>
</tr>
<tr>
<td>$20.00</td>
<td>Marketing</td>
<td>$26.00</td>
</tr>
<tr>
<td>$20.00</td>
<td>Payroll</td>
<td>$23.00</td>
</tr>
<tr>
<td></td>
<td>Misc</td>
<td></td>
</tr>
<tr>
<td>$40.00</td>
<td>Total Expenses</td>
<td>$49.00</td>
</tr>
<tr>
<td>$70.00</td>
<td>Profit</td>
<td>$67.00</td>
</tr>
<tr>
<td>$480.00</td>
<td>Opening Inventory</td>
<td>$500.00</td>
</tr>
<tr>
<td>$100.00</td>
<td>Additions</td>
<td>$190.00</td>
</tr>
<tr>
<td>$390.00</td>
<td>Ending Inventory</td>
<td>$490.00</td>
</tr>
<tr>
<td>57.89</td>
<td>Margin %</td>
<td>58</td>
</tr>
<tr>
<td>36.84</td>
<td>Profit %</td>
<td>33.5</td>
</tr>
<tr>
<td>$5.83</td>
<td>Profit per Ounce</td>
<td>$5.58</td>
</tr>
</tbody>
</table>

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, and then one of the following.
   - Same Workbook to use the current workbook
   - New Workbook to use a new workbook
   - Different Workbooks to cascade each report to a different workbook
3. From Member Selection, 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

Substitution variables are global placeholders that represent variable values. For example, “&CurMnth” might be a substitution variable representing the current month. Application designers or administrators define and manage substitution variables and their corresponding values; Smart View users can enter a substitution variable into the grid and retrieve its value by refreshing.

For example, say the value for substitution variable “&CurMnth” is August. When you enter &CurMnth into a grid, Smart View displays August after a refresh. Later, if the value is changed to September, then September is displayed after a refresh when you enter &CurMnth.
For more complete information about substitution variables, see the Essbase and Planning documentation, available in the EPM Documentation Library. To open this library, from the Smart View ribbon, select the arrow next to Help, and then select EPM Documentation.

To retrieve the value for a substitution variable:

1. **Enter a substitution variable into a cell in the grid.**

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

2. **From any ribbon, select Refresh.**

   The current value defined for substitution variable replaces the substitution variable in the cell (and for all cells in the current worksheet that contain &CurMnth).
In This Chapter

Working with Forms in Excel .................................................................61
Opening Forms in Excel ........................................................................61
Excel Formulas in Forms .....................................................................62
Planning Forms .....................................................................................62
Financial Management Data Forms ....................................................63

**Working with Forms in Excel**

Forms are grid displays in which you can enter data into the database from Excel and 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 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.

In forms opened in Smart View:

- You can modify data values but not the form structure in forms.
- Values submitted to the database from Excel must be non-formatted data.
- If a form is currently loaded in Excel and the administrator changes the form definition on the server side, Oracle recommends that you close the form and reload it. This action ensures that the newest form definitions are displayed.

Customizations made to forms are preserved when you save or refresh only if they are made outside the grid or if they are made to thousands and decimal separators.

**Opening Forms in Excel**

*Data source types:* Planning, Financial Management, Hyperion Enterprise
To open a form:

1. Connect to a data source.

2. In the Smart View Panel, do one of the following:
   - To open one form, expand the tree list and select the form you want to open. Then click **Open form** on the Action Panel.
   - To open multiple 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 form, from the Planning ribbon, select **More** and then **Instructions**.

### Excel Formulas in Forms

You can create Excel formulas in 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.

Formulas are preserved in forms when you refresh the 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 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 form offline
   - Select a different form
   - Connect to a different data source

### Planning Forms

### Planning Form Behavior in Smart View

Forms behave differently in Smart View than they do in Planning as follows:

- Attributes in Planning forms are not displayed in Smart View.
- 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 form, this dimension does not display in the row header of the form in Smart View.
Saving Ad Hoc Grids as Forms

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

To save a Planning ad hoc grid as a 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**.
   
   The saved grid is displayed in the Smart View Panel tree list in the location that you selected in step 2.

Performing Ad Hoc Analysis in Planning Forms

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

To perform ad hoc analysis in Planning forms:

1. Open the form.
2. Do one of the following:
   - From the Smart View ribbon, click **Analyze**. This button is enabled only if the current form has been enabled for ad hoc analysis.
   - Select the form in the Smart View Panel and click **Ad hoc analysis** in the Action Panel.
3. See Chapter 5, “Ad Hoc Analysis” for information about performing ad hoc analysis.

Financial Management Data Forms

If you are unfamiliar with Financial Management, see the Financial Management 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**.

About Financial Management 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 Financial Management Members

If 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 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 134) 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 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.
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, 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 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 select **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

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
- 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 57.

**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.
From the list, select the worksheet that contains the metadata that you want to import to the destination worksheet.

Click OK. You will be asked to confirm your selection.

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, 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 Data Point.
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 Data Point.
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.
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 32.

Retrieving Spreadsheets From Which Data Points Were Copied

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.

Enabling Automatic Column Width Adjustment

To enable the automatic adjustment of Excel column width to accommodate the contents of member and data cells:

1. From the Smart View ribbon, select Options.
2. In Options, from the left panel, select Formatting.
3. Select Adjust column width.
4. Click OK.
5. From any ribbon, select Refresh to adjust columns in the current grid.

If Adjust column width is not selected, you can adjust the width of columns manually.

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 to display a list of choices for deleting Smart View metadata.
   - Save to save the Sheet Info content in an Excel spreadsheet.
3. Optional: To copy the selected item in the list to the clipboard, press Ctrl+C.
4. Click OK.
Enabling and 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.

Smart View may also be enabled or disabled through Office applications.

Disabling Smart View Within Smart View

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.

Enabling and Disabling Smart View in Microsoft Office

You can enable or disable Smart View from Add-ins in Excel Options.

Searching in Smart View

For Essbase, use */? as wild cards.

Shared Workbooks

Smart View does not support Excel shared workbooks.

Printing POV Members in Header and Footer

If you are using Excel 2007 or earlier, you can print the active POV members in the header or footer of an Excel document as follows:

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.
About Smart Query

Data source types: Essbase

A Smart Query is a multidimensional analysis and reporting tool constructed from multiple sets of members and filters.

Creating a Smart Query

You create a Smart Query by defining one or more sets of members from the dimensions in an application. To each member set, you can define and apply composite filters to further refine the data to be returned by the Smart Query. Using these sets and their filters, you can create a highly complex query by defining unions, complements, and intersections of data from the different sets.

Once a Smart Query is created, you can use it for ad hoc reporting and analysis. Smart Queries can be saved, reused, and shared. Sets and filters can be saved individually for use in other Smart Queries.

Creating a Smart Query involves performing the following procedures:

1. “Defining Sets” on page 72
2. “Defining Set Filters” on page 73
3. “Building the Smart Query” on page 73
4. “Completing the Smart Query” on page 75
Defining Sets

To define a set:

1. From the Smart View ribbon, select Panel to open the Smart View Panel.
2. From the Smart View Panel, connect to an Essbase cube or application.
3. From the Action Panel, select New Smart Query Sheet to display the Smart Query Panel in place of the Smart View Panel. (You can select the arrow next to to return to the Smart View Panel.)

Default dimensions for the connection are displayed in the Smart Query Panel and on the worksheet. The Smart Query ribbon is displayed.

4. Optional: Pivot dimensions by dragging them from one area of the Smart Query Panel to another.
5. Select the name of a dimension under Row, Column, or Point of View to display it under Sets for..., where you define the set.
6. In Sets for..., click the arrow next to the dimension name and choose Select Base Members.

For POV dimensions, the members you specify in this step are available for selection from drop-down menus in the Point of View section. You can also enter these names directly.

7. From the menu, select a member level to include in the set or select Other to open the Member Selector, where you can select specific members.
8. Click the arrow next to the dimension name and select Add Custom Members (not available to Point of View dimensions).
9. Select from among specified values for the set.

Optional: Select Other to define members with MDX expressions (see the Essbase documentation) in Custom Member Expression. MDX queries must be at least three characters long to accommodate the simplest member expression; for example, two operands and an operator. The OK button here is enabled only after the expression entered is validated.

Queries are given default names, but if you choose to rename an MDX query, do not use the following characters:

- Brackets ([ ])
- Double quotation marks (" ) or their XML-encoded representation ("&quot;", \\&#34;", \\&#0034;", \\&#x22;", \\&#x0022;")
- Single quotation marks (’ ) or their XML-encoded representation (’&apos;",’&#39;",’&#0039;",’&#x27;",’&#x0027;")

10. Optional: To allow duplicate members in the set, click the arrow next to the set name and select Allow Duplicates in Set (not available to Point of View dimensions).

This setting applies only to the set for which it is selected. When there are multiple sets in a Smart Query, members are displayed in the grid for all sets in which they are selected even if Allow Duplicates in Set is not selected. For example, if Jan, Feb, and Mar are selected as members for one set, and Level 0 members of Year are selected in another set in the same
query, then Jan, Feb, and Mar will each appear twice in the grid because they are members of both sets.

11 **Optional:** To rename the set, click the arrow next to the set name and select **Rename**.

12 To save a set, click the arrow next to the set name, then select **Repository** and then **Save Set**. In **Repository**, enter a name and description for the set and click **OK**.

13 **Optional:** To add additional sets for this dimension, click the arrow next to a set name and select **Add New Set** and repeat the procedure.

14 Select other dimensions from the Smart Query Panel and repeat the procedure as necessary to add members for other dimensions.

### Defining Set Filters

- To define filters for a set:

  1. **From Sets for...**, click the arrow next to a set name and select **Add Filter**.

     A new filter is displayed under **Filters for Set**...

  2. Click the arrow next to the filter name and select one of the following to define the filter:

     - Select Top/Bottom
     - Select Based on Value
     - Select Based on Value Specified by Members
     - String Match (This option does not support qualified member names.)

  3. Repeat this procedure as necessary to add more filters to the set.

### Building the Smart Query

To build a Smart Query, you apply composite filters to select unions, complements, and intersections of members from the different sets. Unions are selections of members that belong to either but not both of two sets. Intersections are selections of members that belong to all specified sets. Complements are selections of members of one specified set that are not members of another specified set.

You can use sets and composite filters in any combination to create highly complex Smart Queries.

#### Union of Sets

To select members that belong to either but not both sets, define two or more filters as described in “Defining Set Filters” on page 73 from the dimension in **Sets for**.... For example, in Figure 9, the query will return only Product, Level 0 members that are EITHER packaged in cans (Product = Can) OR are caffeinated (Product = Caffeinated_True).
Intersections include only the members that are common to all specified sets.

To select members common to all specified sets:

1. Define a filter for a set as described in “Defining Set Filters” on page 73.
2. Click the arrow next to the filter name and select Add Filter.
3. Additional Filters and a subset filter are displayed.
4. Define the second filter.

In Figure 10, the query will return only Product, Level 0 members that are caffeinated AND packaged in a can.

Complement of Sets

To select only members of a set that are not members of another specified set.

1. From Filters for..., click the arrow next to a filter name and select Add Filter.
Additional Filters and a subset filter are displayed.

2 Define the second filter.

3 Click the arrow next to Additional Filters. and select Exclude.

In Figure 11, the query will return only Product, Level 0 members that are caffeinated but are not packaged in a can.

![Figure 11 Complement of Sets](image)

**Completing the Smart Query**

To complete the Smart Query:

1 From the Smart Query ribbon, select options for the Smart Query as follows:
   - Indentation (see “Member Options” on page 128)
   - Show Distinct Member Names (see “Member Options” on page 128)
   - Suppress Rows with No Data (see “Data Options” on page 129)

2 **Optional:** Select Change Alias to select an alias table.
   
The full names of duplicate members are shown regardless of the selected alias table (if any). All other member names are shown according to the selected alias table.

3 To save the entire Smart Query definition, from the Smart Query ribbon, select Save.

4 In Repository, enter a name and description for the Smart Query.

5 Click OK.
   
   Smart Queries are also saved within the workbook, so when you save a workbook, Smart Queries within it are also saved.

6 **Optional:** To perform ad hoc analysis, from the Smart Query ribbon, select Analyze.
Opening a Smart Query

Opening from a New Worksheet

To open an existing Smart Query:

1. From the Smart View ribbon, select Panel to open the Smart View Panel.
2. From the Smart View Panel, click the arrow next to , and then select Smart Query.
3. From the Smart Query Panel, select Open Definition to display the Repository list of available Smart Queries.
4. Select a Smart Query from the list.
5. Click OK.

Opening from an Existing Smart Query Worksheet

To open an existing Smart Query:

1. From the Smart Query ribbon, select Open to display the Repository list of available Smart Queries.
2. Select a Smart Query from the list.
3. Click OK.
4. When asked if you want to discard the existing Smart query on the worksheet, click Yes.

The Smart Query is opened on the current worksheet.

Copying and Pasting

In Excel, you can copy an entire Smart Query definition from one worksheet to another, and sets and filters to other Smart Queries.

Copying Smart Query Definitions in Excel

You can copy a Smart Query definition to a different worksheet in the current workbook or in another workbook. If the worksheet into which you want to paste the definition already contains a Smart Query definition, it will be replaced by the pasted definition.

To copy a Smart Query definition from one worksheet to another:

1. Open the Smart Query worksheet that you want to copy.
2. From the Smart Query ribbon, select Copy.
3. Open a new worksheet.
4. Do one of the following:
To copy the entire definition to a blank worksheet, from the Smart Query Panel, select Paste Definition.

To replace a Smart Query definition, from the Smart Query ribbon of the worksheet to be replaced, select Paste.

Copying Smart Query Sets and Filters

To copy a set, under Sets for..., click the arrow next to a set name and select Copy Set. You can then copy the set within the current Smart Query or to a different Smart Query using Paste Set from the same menu.

To copy a filter, under Filters for Set..., click the arrow next to a filter name and select Copy Filter. You can then copy the set within the current Smart Query or to a different Smart Query using Paste Filter from the same menu.

Copying Smart Query Reports to Word and PowerPoint

You cannot copy a Smart Query definition from Excel to Word or PowerPoint directly, but you can copy a grid or partial grid defined by a Smart Query from Excel to Word or PowerPoint. To do so, from the Smart Query ribbon, select Analyze. Then copy data as described in “Copying Data Between Excel, Word, and PowerPoint” on page 67.
About Smart Slices

A Smart Slice is a reusable perspective of an Essbase or Financial Management 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

Smart Slices are stored centrally and are available to users from the Smart View Panel.

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 ribbon, select Panel.
2. From the Smart View Panel, select a Smart Slice.
3. 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.

4. Click **Insert**.

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

7 To insert a report control, click.

8 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 81.

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

9 Refresh.

10 **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 57.

   **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 12 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 12 Slider](image)

**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 descendents 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 Forms**

To perform ad hoc analysis on a Smart Slice, (Essbase, 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 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

Administrators and database administrators can create, modify, and delete Smart Slices. In Essbase, if enabled by the administrator, all users can create, modify and delete Smart Slices. 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. From the Smart view ribbon, select Panel.

2. Open the Smart View Panel and connect to an Essbase or Financial Management data source.

3. 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.

4. 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.

5. Click Options and set preferences as described in “Setting Smart Slice Preferences” on page 84.

6. Click Done. Member Selection is displayed.

7. In Member Selection, select a dimension member to use as the default POV and click OK.

8. In the Smart View Panel, in Enter a new name, enter a name for the Smart Slice.

9. 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 14, “Smart View Options” for descriptions of the options.
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. The Query designer is available only for ad hoc worksheets.

Creating Queries

Data source types: Essbase, 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**.
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 89.
- In Financial Close Management, you can integrate task lists into Outlook as described in “Integrating Task Lists with Microsoft Outlook” on page 91.

Working with Tasks from the Smart View Panel

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 Panel.
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 then 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**.
   d. Click **OK**.
   e. 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 then select **Panel** 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 then 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, and then Advanced in the left panel.
   c. Clear Disable Smart View add-in in Outlook.
   d. 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.

7. Click OK.

   All task lists associated with the selected application are displayed in Task List.

8. 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.
This guide provides only procedural information for using the Planning features that Smart View supports. For detailed information about Planning, 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.

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

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 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 94.

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 were entered for the planning unit. See “Planning Unit Annotations” on page 96.

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 the 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 by 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 93.

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 then select members for the planning unit list as described in “Selecting Members From the Member Selector” on page 21.

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**

Planning units move from person to person and department to department based on the following:

- The owners and reviewers assigned to the planning unit
- The planning unit place in the 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 94.
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 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 that 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 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 ![icon]).

   The composite form opens in a new Excel workbook with each subform displayed in a
   separate worksheet.
Smart View supports Planning master composite forms.

**Working with Planning Business Rules**

In Planning 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*.

**Launching Business Rules in Excel**

1. To launch a business rule in Excel to recalculate data in Essbase:
   
   1. Open a Planning ad hoc grid or 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**, and 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, and then click **Launch**.
      
      If the business rule includes runtime prompts, enter the information described in step 2 of “Entering Runtime Prompts” on page 99.
      
      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.

1. 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:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Expected Input Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Single Selection Icon]</td>
<td>One member selection</td>
</tr>
<tr>
<td>![Multiple Selection Icon]</td>
<td>Multiple member selections</td>
</tr>
<tr>
<td>Icon</td>
<td>Expected Input Type</td>
</tr>
<tr>
<td>------</td>
<td>---------------------</td>
</tr>
<tr>
<td>123</td>
<td>Numeric value (either entered or selected from cell drop-down menu)</td>
</tr>
<tr>
<td>abc</td>
<td>Text value—Use only with enhanced calculation scripts, not with graphical scripts</td>
</tr>
<tr>
<td></td>
<td>Dimension from the database—Use only with enhanced calculation scripts, not with graphical scripts</td>
</tr>
<tr>
<td></td>
<td><strong>For Calculation Manager business rules only:</strong> A member or member combination that includes only one member from each dimension the designer has set for this runtime prompt (for example: Sales -&gt; Actual -&gt; Jan refers to the member intersection of Sales, Actual, and January)</td>
</tr>
<tr>
<td></td>
<td><strong>For Calculation Manager business rules only:</strong> A range of members, selectable from each dimension the designer has set for this runtime prompt (for example: IDescendants(&quot;Marketing&quot;),FY08)</td>
</tr>
</tbody>
</table>

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 Form and Calculate Currencies Business Rules

The Calculate Data Form business rule is created for each form to calculate subtotals. The Calculate Currencies business rule is created for 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 important and may affect the data. If you plan to launch both Calculate Data Form and the Calculate Currencies business rules, always run the conversions before subtotaling the form.

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

1 **Open a 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, and then Rules on Form.**

The business rules associated with the 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

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 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 form.**
2. **In the 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.

Mass allocation is available only for forms, which 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, and then Member Formula.

Details of the formula are displayed.

### Supporting Detail

Supporting detail serves as a built-in calculator for developing data that is not in the member outline. It can include text, values, and operators that define how data aggregates.

### Adding Supporting Detail

Use the Supporting Detail window to set how detail items aggregate to cell values in a form.
To add supporting detail that calculates values in a form or ad hoc grid:

1. **Open a form, and then 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 Cell Actions and then 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 form.

7. **Click Save.**
   
   Values are dynamically calculated and aggregated before the data is saved. Data on the 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 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:**

<table>
<thead>
<tr>
<th>Option</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add Child</td>
<td>Adds an item one level below the selected cell. You can add an unlimited number of children, but consider its potential performance impact.</td>
</tr>
<tr>
<td>Add Sibling</td>
<td>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.</td>
</tr>
<tr>
<td>Delete</td>
<td>Removes the selected item</td>
</tr>
</tbody>
</table>

---

104  Smart View and Planning
<table>
<thead>
<tr>
<th>Option</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete All</td>
<td>Simultaneously removes all supporting detail</td>
</tr>
<tr>
<td>Promote</td>
<td>Moves the selected item to the next-higher level</td>
</tr>
<tr>
<td>Demote</td>
<td>Moves the selected item to the next-lower level</td>
</tr>
<tr>
<td>Move Up</td>
<td>Moves the selected item to before its sibling predecessor</td>
</tr>
<tr>
<td>Move Down</td>
<td>Moves the selected item to after its sibling successor</td>
</tr>
<tr>
<td>Duplicate Row</td>
<td>Adds a row below the selected item, duplicating its structure (text, operator, and values)</td>
</tr>
<tr>
<td>Fill</td>
<td>For rows, copies the data from the current cell to the cells to its right</td>
</tr>
<tr>
<td>Refresh</td>
<td>Gets the latest stored database values, restoring the previously saved values, and possibly overwriting changes you just made.</td>
</tr>
</tbody>
</table>

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 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 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.**

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 form.**

2 **In the 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, and 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, and then select User Preferences.

3 From Preferences, specify options for the following
   - Application Settings: Email, alias and workflow options.
   - Display Settings: Formatting, page, and other options.
   - User Variables: Variables set up by the Planning to help you navigate large forms and grids.

Note: You cannot set preferences in offline mode.

Working Offline

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

Taking Forms Offline

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

Note: Currency conversion is not supported offline.

➤ To take forms offline:

1 In Excel, connect to the Planning data source that contains the forms you want to take offline.

2 From the Planning ribbon, select More, and then Take Offline.

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

3 Expand the Available Forms/Folders and select folders and forms to take offline.

4 Click Next.
Double-click a dimension. You can select only one dimension.
If you selected multiple forms, the dimensions displayed are merged from the dimensions available for the selected forms.

Select members and system variables from the Member Selection page.
About member relationships:

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

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

Click OK.
Repeat steps 5–7 to select members or system variables for each dimension in the list.
Click Next.
Supply a unique name and a description for the offline connection.
Click Finish to download the selected forms and members.
Click OK, and then click Done.

Working Offline with Forms

To work with forms offline:
In Excel, from the Smart View ribbon, select Panel.
2. From the Smart View Panel, 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 form open while you are directly connected to the Planning server, and then take the form offline in the same session, you must reopen the form from the offline connection to work with it offline.

5. In the offline 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 form taken offline since the beginning of the session is saved to the server. You can sync data from all forms at once or from selected forms and members.

- To save changed data to the Planning server for all forms and members taken offline:
  
  1. From the Planning ribbon, select **Forms**, and 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 forms and members taken offline:
  
  1. From the Planning ribbon, select **Forms**, and then **Sync Back To Server**.
  
  2. Logon to the Planning server.
  
  3. Click **Next**.
  
  4. Double-click a dimension.
  
  5. From the Member Selection page, select members and system variables.
  
  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 then **Done**.
Tip: After you reconnect to the server, check that the work that you completed offline is correct in the database. If you lose a row or column of data when you refresh a form, contact the administrator.

**Refreshing the Offline Form Definition and Data**

To refresh an offline form definition:

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

To update offline data and the offline form definition:

1. From the Smart View menu, select Panel.
2. From the Smart View Panel, select the connection associated with the current offline session.
3. From the Planning menu, select More, and 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 forms taken offline with current online values and definitions. Refresh All maintains the current offline form definition. Skip to step 10.
   
   - Click Next to select forms, members, and system variables to update. This selection may change the form definition; only members and forms that you select remain part of the definition. Members and forms not selected are no longer available offline. Continue to step 6.

6. Double-click a dimension.

7. From the Member Selection page, select members and system variables.
   
   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, and then click Done after the refresh is complete.
Importing Reporting and Analysis Documents

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

Table 2  Reporting and Analysis Applications

<table>
<thead>
<tr>
<th>Reporting and Analysis Application</th>
<th>What You Can Import</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Hyperion Financial Reporting</td>
<td>Reports</td>
</tr>
<tr>
<td>Oracle Hyperion Web Analysis</td>
<td>Reports</td>
</tr>
<tr>
<td>Oracle Hyperion Interactive Reporting</td>
<td>Reports, Charts, Dashboards, Using the latest run of BQY jobs, Interactive Reporting supports refresh capabilities</td>
</tr>
<tr>
<td>Oracle Hyperion SQR Production Reporting</td>
<td>Jobs, Job outputs</td>
</tr>
</tbody>
</table>

- “Importing Financial Reporting Documents” on page 117
- “Importing Web Analysis Documents” on page 123
- “Importing Interactive Reporting Documents” on page 114
- “Importing Production Reporting Documents” on page 121
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**

- When you refresh an imported document in which pages have been deleted, only the remaining pages are refreshed. The deleted pages are not reinstated.
- If editing or refreshing results in fewer pages in an imported document, the removed pages display as blanks pages in Office.
- If editing or refreshing results in more pages in an imported document, those pages are appended to the document in Office.
- Formatting headings and comments are retained when you refresh in Word and PowerPoint but not when you refresh in Excel.

**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 116
- “Editing Financial Reporting Documents” on page 119
- “Editing Production Reporting Jobs” on page 123
- “Editing Web Analysis Documents” on page 125

**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 Smart View-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 Smart View formatting definitions and calculated members are retained. Thus, Smart View cannot directly query the data source, but Smart View content can be leveraged by Microsoft Office applications.
Tip: After importing an image in Word or PowerPoint, use the 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.

**Importing Interactive Reporting Documents**

- “Importing Interactive Reporting Documents into Excel” on page 115
- “Importing Interactive Reporting Documents into Word and PowerPoint” on page 116
- “Editing Interactive Reporting Documents” on page 116

Imported Interactive Reporting documents are section-specific.

**Table 3 Interactive Reporting Import Object Types**

<table>
<thead>
<tr>
<th>Section</th>
<th>Excel</th>
<th>Word, PowerPoint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table</td>
<td>Formatted data</td>
<td>N/A</td>
</tr>
<tr>
<td>Results</td>
<td>Formatted data</td>
<td>N/A</td>
</tr>
<tr>
<td>Chart</td>
<td>Formatted data</td>
<td>Image</td>
</tr>
<tr>
<td>Pivot</td>
<td>Formatted data</td>
<td>N/A</td>
</tr>
<tr>
<td>Report</td>
<td>Formatted data</td>
<td>Image</td>
</tr>
<tr>
<td>Dashboard</td>
<td>Image</td>
<td>Image</td>
</tr>
<tr>
<td>Query</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>CubeQuery</td>
<td>Query ready (Internet Explorer only, not supported by Firefox)</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>Formatted data</td>
<td>N/A</td>
</tr>
<tr>
<td>Data model</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

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.
To import Interactive Reporting documents into Excel:

1. From the Smart View ribbon, select Panel.
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 Filters, 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. From the Import Section As drop-down, select one of the following:
   - Data to import content as query-ready HTML. The current page of the current CubeQuery section is converted to HTML and Smart View formatting is removed. This enables you to requery the data source independent of the Web application.
   - Image to import content as formatted HTML. The current page of the CubeQuery section is converted to HTML with Smart View formatting definitions and calculated members. Smart View does not directly query the data source. This option is available only for a CubeView section for Refresh and Preview.
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 **Panel**.
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.oe*).
   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 **Panel**, 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.oce).
   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

- “Importing Financial Reporting Documents into Excel” on page 118
- “Importing Financial Reporting Documents into Word and PowerPoint” on page 119
- “Editing Financial Reporting Documents” on page 119

Table 4 Financial Reporting Import Document Types

<table>
<thead>
<tr>
<th>Document Type</th>
<th>Excel</th>
<th>Word, PowerPoint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report</td>
<td>Fully formatted, query-ready</td>
<td>Image</td>
</tr>
<tr>
<td>Snapshot report</td>
<td>Fully formatted</td>
<td>Image</td>
</tr>
<tr>
<td>Book</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Snapshot Book</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Batch</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Grid Object</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Image Object</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Chart Object</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Text Object</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Row and Column template</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
To import Financial Reporting documents into Excel:

1. From the Smart View ribbon, select Panel.

2. In the Smart View Panel, connect to an EPM Workspace data source.


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 Panel.
2. In the Smart View Panel, connect to a EPM Workspace data source.
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, 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 Panel, 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, in EPM Workspace preferences, select Preview for User Point of View.

3 Optional: In documents that contain prompts, to change the default value, in Respond to Prompts, make a selection for prompts, and click Next.

Note: Respond to Prompts is displayed only if the document contains prompts.

4 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 that can be saved 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.

1 To create a template:

1 Open PowerPoint.

2 Connect to a Reporting and Analysis provider.

3 From the Smart View ribbon, select Panel, then Reporting and Analysis Document, and then Create Template.


   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

1 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 Panel, then Reporting and Analysis Document, and then Refresh Template.

4 Edit and save the PowerPoint presentation as needed.
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 121
- “Importing Production Reporting Jobs into Word and PowerPoint” on page 122
- “Importing Production Reporting Job Outputs into Word, and PowerPoint” on page 122
- “Editing Production Reporting Jobs” on page 123

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Excel</th>
<th>Word, PowerPoint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job</td>
<td>Formatted data</td>
<td>Image</td>
</tr>
<tr>
<td>Job output</td>
<td>Formatted data</td>
<td>Image</td>
</tr>
</tbody>
</table>

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

1. From the Smart View ribbon, select Panel.
2. In the Smart View Panel, connect to an EPM Workspace data source.
3. Navigate to the Oracle 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.
To import Production Reporting jobs into Word and PowerPoint:

1. From the Smart View ribbon, select Panel.
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.

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 Panel, 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:
  2. From the Smart View ribbon, select Panel, 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 113.

- “Importing a Web Analysis Document or Document Objects” on page 124
- “Editing Web Analysis Documents” on page 125

<table>
<thead>
<tr>
<th>Table 6</th>
<th>Web Analysis Import Document Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document Type</td>
<td>Excel</td>
</tr>
<tr>
<td>Report</td>
<td>Fully formatted, query-ready</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 7</th>
<th>Web Analysis Import Data Object Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Object</td>
<td>Excel</td>
</tr>
<tr>
<td>Spreadsheet</td>
<td>Data + formatting</td>
</tr>
</tbody>
</table>
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 Panel.
2. In the Smart View Panel, connect to an 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 placed 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 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).
  2. From the Smart View ribbon, select **Panel**, 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**, and then click **Next**.

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

4. Select a 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

You can use Microsoft smart tags to import 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.
Setting Smart View Options

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

Global Options and Sheet Level Options

Smart View provides two types of options, global options and sheet options.

- **Global options** are options that apply to the entire current workbook and to any workbooks and worksheets that are created henceforth. The following are global options:
  - Advanced Options
  - Extensions
  - Cell Styles
- **Sheet level options** are options that are specific to the worksheet for which they are set. They are also the default option settings for new worksheets in the current workbook and for any new workbook. Changes to sheet level option settings do not affect existing worksheets or workbooks. The following are sheet level options:
  - Member Options
  - Data Options
  - Formatting Options
**Member Options**

Member options are sheet level options, which are specific to the worksheet for which they are set. They are also the default member option settings for new worksheets in the current workbook or any new workbook. Changes to member option settings do not affect existing worksheets or workbooks.

To set options for the display of member cells as described in Table 8, click **Options** on the Smart View ribbon, and then select **Member Options** in the left panel. When you are finished, click **OK**.

To set your selections on this page as default settings, click the arrow in the **OK** button, and then select **Set as Default Options**.

**Note:** Not all data providers support all the options listed in the table.

---

**Table 8  Member Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General</strong></td>
<td>General</td>
</tr>
<tr>
<td>Zoom In Level</td>
<td>From the drop-down menu, select one of the following to specify a default zoom level for ad hoc analysis:</td>
</tr>
<tr>
<td></td>
<td>● <strong>Next Level</strong> to retrieve data for the children of the selected members</td>
</tr>
<tr>
<td></td>
<td>● <strong>All Levels</strong> to retrieve data for all descendants of the selected members</td>
</tr>
<tr>
<td></td>
<td>● <strong>Bottom Level</strong> to retrieve data for the lowest level of members in a dimension</td>
</tr>
<tr>
<td></td>
<td>● <strong>Sibling Level</strong> to retrieve data for all members at the same level as the selected member</td>
</tr>
<tr>
<td></td>
<td>● <strong>Same Level</strong> to retrieve data for the siblings of the selected members</td>
</tr>
<tr>
<td></td>
<td>● <strong>Same Generation</strong> to retrieve data for all members of the same generation as the selected members</td>
</tr>
<tr>
<td></td>
<td>● <strong>Formulas</strong> to retrieve data for all members that are defined by the formula of the selected member. The formula can be a member equation or a consolidation to the parent.</td>
</tr>
<tr>
<td>Member Name Display</td>
<td>From the drop-down menu, select one of the following to specify how to display member names in cells:</td>
</tr>
<tr>
<td></td>
<td>● <strong>Member Name Only</strong> to display member names</td>
</tr>
<tr>
<td></td>
<td>● <strong>Distinct Member Name</strong> to display fully qualified names</td>
</tr>
<tr>
<td></td>
<td>● <strong>Member Name and Alias</strong> to display member names and their aliases</td>
</tr>
<tr>
<td></td>
<td>● <strong>Description Only</strong> to display aliases</td>
</tr>
<tr>
<td>Indentation</td>
<td>From the drop-down menu, select one of the following to specify how hierarchy levels are to be indented:</td>
</tr>
<tr>
<td></td>
<td>● <strong>None</strong></td>
</tr>
<tr>
<td></td>
<td>● <strong>Subitems</strong> to indent descendants. Ancestors are left-justified in the column.</td>
</tr>
<tr>
<td></td>
<td>● <strong>Totals</strong> to indent ancestors. Descendants are left-justified in the column.</td>
</tr>
<tr>
<td>Ancestor Position</td>
<td>From the drop-down menu, select one of the following to specify ancestor position in hierarchies:</td>
</tr>
<tr>
<td></td>
<td>● <strong>Top</strong> to display hierarchies in order from highest to lowest level</td>
</tr>
<tr>
<td></td>
<td>● <strong>Bottom</strong> to display hierarchies in order from lowest to highest level</td>
</tr>
<tr>
<td>Member Retention</td>
<td>Member Retention</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Include Selection</td>
<td>Display the selected member and the members retrieved as a result of the operation.</td>
</tr>
<tr>
<td>Within Selected Group</td>
<td>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.</td>
</tr>
<tr>
<td>Remove Unselected Groups</td>
<td>For Zoom In or Zoom Out, remove all dimensions and members except the selected member and the members retrieved as a result of zooming.</td>
</tr>
<tr>
<td>Comments and Formulas</td>
<td>Comments and Formulas</td>
</tr>
<tr>
<td>Preserve Formulas and Comments in ad hoc operations except pivot</td>
<td>Preserves formulas and comments on the grid during queries. You can clear this option to make queries run faster, but if you do, formulas and comments are removed or ignored. This option must be selected if you select Formula Fill or Enable Enhanced Comment Handling.</td>
</tr>
<tr>
<td>Formula Fill</td>
<td>Propagates formulas associated with member cells to the members retrieved as a result of zooming in.</td>
</tr>
<tr>
<td>Enable Enhanced Comment Handling</td>
<td>Enables you to review and correct comments and member names in ad hoc grids that contain comments</td>
</tr>
<tr>
<td>Preserve Formula in POV Change</td>
<td>Preserves formulas in cells when you refresh or make changes to the POV. Otherwise, any formulas in the grid are lost.</td>
</tr>
</tbody>
</table>

**Data Options**

Data options are sheet level options, which are specific to the worksheet for which they are set. They are also the default data option settings for new worksheets in the current workbook or any new workbook. Changes to data option settings do not affect existing worksheets or workbooks.

To set options for the display of data cells as described in Table 9, click Options on the Smart View ribbon, and then select Data Options in the left panel. When you are finished, click OK.

To set your selections on this page as default settings, click the arrow in the OK button, and then select Set as Default Options.

**Note:** Not all data providers support all the options listed in the table.
### Table 9  Data Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
</table>
| **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         | 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**    | Replacement                                                                   |
| #NoData/Missing Label| 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:  
                     • 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. |
| #NoAccess Label    | Select if you entered #NumericZero above and want to be able to submit zeroes to the database. |
| #Invalid/ Meaningless| 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. |
| Submit Zero        | If the administrator has created specific formatting for the display of numerical data, view data in this formatting. |
| Mode               | Mode                                                                         |
Option | Description
--- | ---
Cell Display | As an alternative to displaying actual data, you can display the calculation or process status of the cells:
  - **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**

Advanced options are global options, which apply to the entire current workbook and to any workbooks and worksheets that are created henceforth.

To set options for the administrative and other advanced tasks as described in Table 10, click **Options** on the Smart View ribbon, and then select **Advanced** in the left panel. When you are finished, click **OK**.

**Note:** Not all data providers support all the options listed in the table.

**Table 10  Advanced Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General</strong></td>
<td><strong>General</strong></td>
</tr>
<tr>
<td>Shared Connections URL</td>
<td>Specify a default URL for all connections. Use the following syntax: http://&lt;server&gt;:19000/workspace/SmartViewProviders</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>This field must contain an EPM Workspace URL for Smart View online help to be available.</td>
</tr>
<tr>
<td>Number of Undo Actions</td>
<td>The number of Undo and Redo actions permitted on an operation (0 through 100). See &quot;Using Undo and Redo &quot; on page 65.</td>
</tr>
<tr>
<td>Number of Most Recently Used Items</td>
<td>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.</td>
</tr>
<tr>
<td>Delete All MRU Items</td>
<td>Delete all items in your most recently used list, including those that are pinned to the list.</td>
</tr>
<tr>
<td>Logging</td>
<td>Logging</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Log Message Display                         | 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:  
  - **Information**: All messages, including warnings and errors — recommended to diagnose problems. May adversely impact performance.  
  - **Warnings**: Warnings and error level messages. May adversely impact performance.  
  - **Errors**: Error messages only — recommended for general use. Has minimal impact on performance.  
  - **None**: Suppress all messages.  
  - **Extended Info**: Information-level messages plus all server responses and requests. Adversely impacts performance.  
  - **Profile**: Extended Info log entries and most function calls. Creates XML files for each Office application with active Smart View. Intended for debugging. Severely impacts performance. |
| 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                                     | Display                                                                                                                                                                                                                                                                                                                                 |
| 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.                                                                                                                                                                                                                                               |
| Display Drill Through Report ToolTips      | Display by default lists of available drill-through reports for cells whenever you mouse over them.                                                                                                                                                                                                                                      |
| Compatibility                               | Compatibility                                                                                                                                                                                                                                                                                                                                 |
| Reduce Excel File Size                     | Should always be selected except in the following cases, when it should be cleared:  
  - You send an Excel workbook to users on Smart View releases earlier than 9.3.1.6 or to users on Microsoft Office regardless of Smart View release. In these workbooks:  
    - Grids that contain functions must be refreshed before data can be displayed.  
    - In ad hoc mode, POV settings are lost; the behavior is similar to that of a fresh ad hoc grid.  
  - You open a workbook sent from users on Smart View release earlier than 9.3.1.6 or on Microsoft Office regardless of Smart View release.                                                                                                                                 |

---

Smart View Options

132
### Option Description

**Improve Metadata storage**
Should always be selected except in the following cases, when it should be cleared:
- You send an Excel workbook to users on Smart View releases earlier than 9.3.1.6 or to users on Microsoft Office regardless of Smart View release
- You open a workbook sent from users on Smart View release earlier than 9.3.1.6 or on Microsoft Office regardless of Smart View release

**Refresh Selected Functions and their dependents**
Execute dependent functions on the same sheet before executing the selected functions.

### Mode

**Use Double click for Operations**
Double-clicking retrieves the default grid in a blank worksheet and thereafter zooms in or out on the cell contents. If not selected, double-clicking retains standard Excel functionality and puts a cell into edit mode.

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 318, double-clicking prompts you to log into Spreadsheet Add-in.

---

### Formatting Options

Formatting options are sheet level options, which are specific to the worksheet for which they are set. They are also the default formatting option settings for new worksheets in the current workbook or any new workbook. Changes to formatting option settings do not affect existing worksheets or workbooks.

To set options for formatting numbers as described in Table 11, click Options on the Smart View ribbon, and then select Advanced in the left panel. When you are finished, click OK.

To set your selections on this page as default settings, click the arrow in the OK button, and then select Set as Default Options.

**Note:** Not all data providers support all the options listed in the table.

#### Table 11  Number Formatting Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Thousands Separator</td>
<td>Use a comma or other thousands separator in numerical data. Do not use # or $ as the thousands separator in Excel International Options.</td>
</tr>
<tr>
<td>Use Cell Styles</td>
<td>Use formatting that is defined in Cell Styles (see “Cell Styles” on page 134) or by the data provider. Overrides any user formatting.</td>
</tr>
<tr>
<td>Use Excel Formatting</td>
<td>Use Excel rather than Smart View formatting and retain Excel formatting for ad hoc operations.</td>
</tr>
<tr>
<td>Move Formatting on Operations</td>
<td>Copy parent cell formatting to zoomed in cells and retain this formatting even if the cell location changes after an operation.</td>
</tr>
</tbody>
</table>
### Cell Styles

Cell style options are global options, which apply to the entire current workbook and to any workbooks and worksheets that are created henceforth.

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.
  7. Optional: To set your selections on this page as default settings, click the arrow in the OK button, and then select Set as Default Options.

### Extensions

Extension options are global options, which apply to the entire current workbook and to any workbooks and worksheets that are created henceforth.

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.

Smart View supports extensions for the following.
- Oracle Hyperion Disclosure Management
- Oracle Hyperion Financial Reporting
- Oracle Hyperion Strategic Finance
- The Predictive Planning feature of Planning
- Crystal Ball EPM
- Smart Query

To set your selections on this page as default settings, click the arrow in the OK button, and then select **Set as Default Options**.
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.
- **HsGetVariable**: Retrieves the associated value for a substitution variable.
- **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. Ensure that your connection is a private connection. If it is a shared connection, save it as a private connection before connecting.
2. Click the cell in which you want to enter the function.
3. From the Smart View ribbon, select Functions, and 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 HsGetText and HsSetText only:
   - Select Comments or Cell Reference, and then enter the cell text to submit
   - Select Cell Text Label, and then select a label from the drop-down menu.

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. It is enabled only if the syntax is invalid or if the syntax has not already been validated.

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, follow the procedure in “Running Functions” on page 140.

---

Creating Functions Manually

In Excel 2003, functions can contain a maximum of 255 characters. See Microsoft documentation and support site for information about character and other Excel limitations.

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 141.

   Connection parameters can have these values:
   - Empty: the default connection
   - HsActive: the active associated connection
   - The user-defined name for a private 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 141 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. Do one of the following:
   - For HsSetValue, from the Smart View ribbon, select Submit Data.
   - For other functions, select one:
     - 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.
# 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 9, “Data Options,” on page 130.)

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 12.

### Table 12  HsGetSheetInfo Details

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

**Syntax**

HsGetSheetInfo("<string equivalent>")

HsGetSheetInfo("<numerical equivalent>")

**Example**

In this example, HsGetSheetInfo tells you whether the worksheet contains an ad hoc grid or a 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 alias 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 for dimension members, cell references, the default POV, or a combination of all three.
Syntax

HsGetText ("Connection","POV","CellTextLabel")

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")

HsGetVariable

Data sources: Essbase

HsGetVariable retrieves the associated value for a substitution variable.

You cannot use HsGetVariable with Smart Slices.

Syntax

HsGetVariable can use the default connection name, a private connection name, or an Excel
named range on a multiple-range grid, as follows:

- **Default connection**: HsGetVariable("substitution variable name")
- **Private connection**: HsGetVariable("private connection
  name","substitution variable name")
- **Named range on a multiple-range grid**: HsGetVariable("range
  name","substitution variable name")
**Examples**

- **Default connection:** HsGetVariable("CurMonth")
- **Private connection:** HsGetVariable("stm10026_Sample_Basic","CurMonth")
- **Named range:** HsGetVariable("stm10026_Sample_Basic","CurMonth")

**Note:** An ampersand (&) is generally used to refer to a substitution variable, but is optional in this function.

---

**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 From the Member Selector” on page 21.

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.
About VBA Functions in Smart View

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

- “Assumed Knowledge” on page 147
- “VBA Parameters” on page 148
- “VBA Return Values” on page 149
- “VBA Function Types” on page 151
- “Using Spreadsheet Toolkit VBA Applications in Smart View” on page 306

Assumed Knowledge

To use the information in this chapter to develop VBA applications for Smart View, you need the following:

- Familiarity with Smart View and how it is used in your organization
- Working knowledge of Visual Basic or VBA programming language
- Understanding of Excel Visual Basic Editor as an environment for VBA development

Location of Smart View VBA Functions File

All Smart View VBA functions are contained in the file smartview.bas, located by default in EPM_ORACLE_HOME/smartview/bin. Import smartview.bas into a Visual Basic Editor module, and use this module to copy and paste VBA functions into your program.
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 from `smartview.bas`, using descriptions from “VBA Function Types” on page 151

VBA Parameters

Most VBA functions require you to supply one or more parameters. Table 13 lists the parameter types and the valid values for each type:

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text</td>
<td>A word or phrase or name in quotes. For example:</td>
</tr>
<tr>
<td></td>
<td>Smart View</td>
</tr>
<tr>
<td></td>
<td>[Book2.xls]Sheet1</td>
</tr>
<tr>
<td>Boolean</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>False</td>
</tr>
<tr>
<td>Range Object</td>
<td>A cell, row or column, one or more selections of cells, or a three-dimensional range address, surrounded by quotes. For example:</td>
</tr>
<tr>
<td></td>
<td>RANGE(“A1”)</td>
</tr>
<tr>
<td></td>
<td>RANGE(“A1:B2”)</td>
</tr>
<tr>
<td></td>
<td>RANGE(“G:G,K:K”)</td>
</tr>
<tr>
<td></td>
<td>RANGE(“A1:B5,C1:C10,D5:L8”)</td>
</tr>
<tr>
<td></td>
<td>RANGE(“Sheet1!C3:R20,Sheet2!C3:R20”)</td>
</tr>
<tr>
<td>Number</td>
<td>A number without quotes and without commas. For example:</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2.5</td>
</tr>
<tr>
<td></td>
<td>50000</td>
</tr>
<tr>
<td>List of Strings</td>
<td>A list of text values, separated by commas. For example:</td>
</tr>
<tr>
<td></td>
<td>“Qtr1”, “Actual”, “Oregon”</td>
</tr>
<tr>
<td>Constant</td>
<td>A predefined constant from <code>smartview.bas</code>.</td>
</tr>
<tr>
<td>Parameters</td>
<td>Values</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Default Value</td>
<td>Null</td>
</tr>
<tr>
<td></td>
<td>Empty</td>
</tr>
</tbody>
</table>

**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.

---

**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 14** lists the return values.

<table>
<thead>
<tr>
<th>Return Value</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Function ran successfully</td>
</tr>
<tr>
<td>1</td>
<td>Typically, the user pressed Escape or clicked Cancel from a dialog box.</td>
</tr>
<tr>
<td>Large positive number</td>
<td>Failure because of server problem; for example, server not running or an invalid user name.</td>
</tr>
<tr>
<td>-1</td>
<td>Valid return value, True</td>
</tr>
<tr>
<td>-2</td>
<td>Termination error</td>
</tr>
<tr>
<td>-3</td>
<td>Initialization error</td>
</tr>
<tr>
<td>-4</td>
<td>Spreadsheet is not yet connected to the server</td>
</tr>
<tr>
<td>-6</td>
<td>Not used</td>
</tr>
<tr>
<td>-7</td>
<td>Spreadsheet has become unstable</td>
</tr>
<tr>
<td>-8</td>
<td>No Undo information exists</td>
</tr>
<tr>
<td>-9</td>
<td>Operation has been canceled</td>
</tr>
<tr>
<td>-12</td>
<td>Undo is not enabled</td>
</tr>
<tr>
<td>-13</td>
<td>Not enough memory resources are available</td>
</tr>
<tr>
<td>-14</td>
<td>Appropriate dialog box could not be displayed</td>
</tr>
<tr>
<td>-15</td>
<td>Function contains an invalid parameter</td>
</tr>
<tr>
<td>-16</td>
<td>Calculation is in progress</td>
</tr>
<tr>
<td>-17</td>
<td>Obsolete setting</td>
</tr>
<tr>
<td>-18</td>
<td>Operation is not allowed because the spreadsheet is in formula preservation mode</td>
</tr>
<tr>
<td>Return Value</td>
<td>Definition</td>
</tr>
<tr>
<td>--------------</td>
<td>------------</td>
</tr>
<tr>
<td>-19</td>
<td>Operation cannot take place on the specified sheet</td>
</tr>
<tr>
<td>-20</td>
<td>Current sheet cannot be determined</td>
</tr>
<tr>
<td>-21</td>
<td>Spreadsheet name was not specified and no active sheet is selected</td>
</tr>
<tr>
<td>-22</td>
<td>Calculation cannot be canceled because no calculation is running</td>
</tr>
<tr>
<td>-23</td>
<td>Selection parameter is invalid</td>
</tr>
<tr>
<td>-25</td>
<td>Cascade list file cannot be created, or you are attempting to cascade while the spreadsheet is embedded in another document</td>
</tr>
<tr>
<td>-26</td>
<td>Spreadsheet macros cannot be run due to a licensing agreement</td>
</tr>
<tr>
<td>-27</td>
<td>Spreadsheet macros which update the database cannot be run due to a licensing constraint</td>
</tr>
<tr>
<td>-28</td>
<td>Database cannot be updated because you have a read-only license for the database</td>
</tr>
<tr>
<td>-29</td>
<td>Obsolete setting</td>
</tr>
<tr>
<td>-30</td>
<td>Menu is removed already</td>
</tr>
<tr>
<td>-31</td>
<td>Menu is added already</td>
</tr>
<tr>
<td>-39</td>
<td>The specified worksheet is protected. Unprotect the worksheet and try the operation again.</td>
</tr>
<tr>
<td>-40</td>
<td>Calc script not found.</td>
</tr>
<tr>
<td>-41</td>
<td>Provider not supported</td>
</tr>
<tr>
<td>-42</td>
<td>Invalid alias</td>
</tr>
<tr>
<td>-43</td>
<td>Connection not found</td>
</tr>
<tr>
<td>-44</td>
<td>Provider Services connection not found</td>
</tr>
<tr>
<td>-45</td>
<td>Provider Services not connected</td>
</tr>
<tr>
<td>-46</td>
<td>Provider Services cannot connect</td>
</tr>
<tr>
<td>-47</td>
<td>Connection already exists</td>
</tr>
<tr>
<td>-48</td>
<td>Provider Services url not saved</td>
</tr>
<tr>
<td>-49</td>
<td>Migration of Connection Not Allowed</td>
</tr>
<tr>
<td>-50</td>
<td>Connection manager not initialized</td>
</tr>
<tr>
<td>-51</td>
<td>Failed to Get Provider Services Override Property</td>
</tr>
<tr>
<td>-52</td>
<td>Failed to Set Provider Services Override Property</td>
</tr>
<tr>
<td>-53</td>
<td>Failed to Get Provider Services URL</td>
</tr>
<tr>
<td>-54</td>
<td>Provider Services disconnect failed</td>
</tr>
<tr>
<td>Return Value</td>
<td>Definition</td>
</tr>
<tr>
<td>--------------</td>
<td>------------</td>
</tr>
<tr>
<td>-55</td>
<td>Operation failed</td>
</tr>
<tr>
<td>-56</td>
<td>Cannot associate sheet with connection</td>
</tr>
<tr>
<td>-57</td>
<td>Refresh sheet needed</td>
</tr>
<tr>
<td>-58</td>
<td>No grid object on sheet</td>
</tr>
<tr>
<td>-59</td>
<td>No connection associated</td>
</tr>
<tr>
<td>-60</td>
<td>Non data cell passed</td>
</tr>
<tr>
<td>-61</td>
<td>Data cell is not writeable</td>
</tr>
<tr>
<td>-62</td>
<td>No Smart View content on sheet</td>
</tr>
<tr>
<td>-63</td>
<td>Failed to get Office object</td>
</tr>
<tr>
<td>-64</td>
<td>Operation failed as chart is selected</td>
</tr>
<tr>
<td>-65</td>
<td>Excel in edit mode</td>
</tr>
<tr>
<td>-66</td>
<td>Sheet not compatible with Smart View</td>
</tr>
<tr>
<td>-67</td>
<td>Application not stand alone</td>
</tr>
<tr>
<td>-68</td>
<td>Smart View is disabled.</td>
</tr>
<tr>
<td>-69</td>
<td>The function has been deprecated.</td>
</tr>
<tr>
<td>-70</td>
<td>The operation is not supported in worksheets that are in multiple grid mode.</td>
</tr>
</tbody>
</table>

### VBA Function Types

- **General** functions perform actions, set options, or retrieve information typically performed from the Smart View ribbon or Options dialog box. See “General Functions” on page 152.
- **Connection** functions perform actions related to connections to data sources. See “Connection Functions” on page 165.
- **Ad hoc** functions perform ad hoc operations such as zooming, retrieving and submitting data, and pivoting. See “Ad Hoc Functions” on page 179.
- The **Form** function opens a data form. See “Form Functions” on page 193.
- **Cell** functions perform operations and retrieve information for data cells and their contents. See “Cell Functions” on page 194.
- **POV** functions specify or retrieve settings for the POV. See “POV Functions” on page 209.
- **Calculation script and business rule** functions retrieve lists of or execute calculation scripts and business rules. See “Calculation Script and Business Rule Functions” on page 220.
• Calculation, consolidation, and translation functions executes these operations on data for Financial Management and Hyperion Enterprise applications. See “Calculation, Consolidation, and Translation Functions” on page 229

• Member query functions retrieve generation, level, attribute, and other information about members. See “Member Query Functions” on page 235

• Options functions set and retrieve information for global and/or sheet options, and enable deletion of MRU items. See “Options Functions” on page 254

• The MDX query function executes an MDX query whose results are not displayed in a worksheet. See “MDX Query Functions” on page 265

• Menu functions are identical to the equivalent commands on the Smart View menu and ribbon. See “Menu Functions” on page 267.

• Dynamic link functions set or retrieve data point details that are displayed in separate windows via dynamic links. See “Dynamic Link Functions” on page 289

**Note:** For an alphabetical list of VBA functions, see the index.

**General Functions**

• “HypShowPOV” on page 152
• “HypSetMenu” on page 153
• “HypCopyMetaData” on page 154
• “HypDeleteMetadata” on page 155
• “HypIsDataModified” on page 156
• “HypIsSmartViewContentPresent” on page 157
• “HypIsFreeForm” on page 158
• “HypUndo” on page 158
• “HypRedo” on page 159
• “HypPreserveFormatting” on page 160
• “HypRemovePreservedFormats” on page 161
• “HypSetAliasTable” on page 162
• “HypGetSubstitutionVariable” on page 162
• “HypSetSubstitutionVariable” on page 164

**HypShowPOV**

Data source types: All
**Description**

HypShowPOV() determines whether the POV toolbar is to be displayed or hidden.

**Syntax**

HypShowPov(bShowPov)

ByVal bShowPov As Boolean

**Parameters**

bShowPov: Set to TRUE to show the POV toolbar.

**Return Value**

Boolean. If the POV is shown, the return value is TRUE; otherwise, the return value is FALSE.

**Example**

```vba
Public Declare Function HypShowPov Lib "HsAddin" (ByVal bShowPov As Boolean) As Long

Sub DisConn()
    X=HypShowPov(True)
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**

```vba
Declare Function HypSetMenu Lib "HsAddin" (ByVal bSetMenu As Boolean) As Long

Sub SetMyMenu()
    X = HypSetMenu(TRUE)
End Sub
```

**HypCopyMetaData**

**Data Source Types:** Essbase, Planning, Financial Management, Hyperion Enterprise

**Description**

HypCopyMetaData() performs Export Metadata.

**Syntax**

```vba
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**

```vba
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
```
HypDeleteMetadata

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise, 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)
vDispObject As Variant
vtbWorkbook As Variant
vtbClearMetadataOnAllSheetsWithinWorkbook As Variant

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

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
**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.

```vbnet
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 TestIsSVCContentOnSheet()
    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

**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()
    Dim oRet As Boolean

    oRet = HypIsFreeForm(Empty)
    MsgBox (oRet)
End Sub

**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

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**

```vba
Declare Function HypRedo Lib "HsAddin" (ByVal vtSheetName As Variant) As Long

Sub Redo()
    X = HypRedo(Empty)
End Sub
```

**HypPreserveFormatting**

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

**Description**

HypPreserveFormatting() applies grid formatting to cells created by zooming in.

**Syntax**

```vba
HypPreserveFormatting (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**

```vba
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

**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], vtbRemoveAllPreservedFormats [in], vtSelectionRange [in])

ByVal vtSheetName As Variant
ByVal vtbRemoveAllPreservedFormats As Variant
ByVal vtSelectionRange As Variant

**Parameters**

**vtSheetName:** For future use. Currently the active sheet is used.

**vtbRemoveAllPreservedFormats:** 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

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

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

**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

**Connection Functions**

Connection functions perform actions related to connections to data sources.

- “HypConnect” on page 166
- “HypConnected” on page 166
- “HypConnectionExists” on page 167
- “HypCreateConnection” on page 168
- “HypCreateConnectionEx” on page 170
- “HypDisconnect” on page 172
- “HypDisconnectAll” on page 172
- “HypDisconnectEx” on page 173
- “HypGetSharedConnectionsURL” on page 173
- “HypSetSharedConnectionsURL” on page 174
- “HypIsConnectedToSharedConnections” on page 175
- “HypRemoveConnection” on page 175
- “HypinvalidateSSO” on page 176
- “HypResetFriendlyName” on page 176
- “HypSetActiveConnection” on page 177
- “HypSetAsDefault” on page 178
- “HypSetConnAliasTable” on page 179
**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**

```vba
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

**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_ENTERPRISE = "Hyperion Enterprise"
  ● Global Const HYP_FINANCIAL_MANAGEMENT = "Hyperion Financial Management"
  ● New: Global Const HYP_PLANNING = "Planning"
  ● Deprecated: Global Const HYP_ANALYTIC_SERVICES = "Analytic Provider Services"
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", "Sample", "Sample")
End Sub

About VBA Functions in Smart View  169
HypCreateConnectionEx

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

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.

For Essbase, Planning, and Financial Management, HypCreateConnectionEx can be used to create private connections using a Workspace URL.

Syntax

HypCreateConnectionEx(vtUserName, vtPassword, vtProvider, vtProviderURL, vtServerName, vtApplicationName, vtDatabaseName, vtFriendlyName, vtDescription)

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_FINANCIAL_MANAGEMENT = "Hyperion Financial Management"
Global Const HYP_RA = "Hyperion Smart View Provider for Hyperion Reporting and Analysis"

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.

Note: For Oracle Hyperion Reporting and Analysis, only the provider URL, provider type, and connection name are required.

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", ",", ",", ",")
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:** 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
```

**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 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

**HypSetSharedConnectionsURL**

**Data Source Types:** Essbase, Planning, Financial Management, Hyperion Enterprise

**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

**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

**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

**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

**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 HypResetFriendlyNameTest()
    Dim lRet As Long
    lRet = HypResetFriendlyName("server2_Sample_Basic", "My Sample Basic")
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

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

**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

**Ad Hoc Functions**

Ad hoc functions perform ad hoc operations such as zooming, retrieving and submitting data, and pivoting.
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

**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
    sts = HypRetrieveRange("Empty, range("E11:F28"), "Samp1")
        'retrieve by regular range
    sts = HypRetrieveRange(Empty, range("MyRange"), "Samp1")
        'retrieve by named range
End sub

HypRetrieveNameRange

Data source types: Essbase

Description

HypRetrieveNameRange refreshes the grid created by HypRetrieveRange.

Syntax

HypRetrieveNameRange (vtSheetName, vtGridName)

ByVal vtSheetName As Variant
ByVal vtGridName As Variant

Parameters

vtSheetName: For future use. Currently the active sheet is used.
vtGridName: Input variable, contains the name of the named range or grid to be refreshed.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Examples

Example 1

Public Declare Function HypRetrieveNameRange Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtGridName As Variant) As Long

Sub RetrieveAllRange()
    'connect all required connections
    sts = HypConnect("Sheet1", "admin", "password", "stm10026_Sample_Basic")
    'get list of named grids available
sts = HypGetNameRangeList("Sheet1", "", vtList)
' refresh each range one by one
For i = 0 To 2
sts = HypRetrieveNameRange("Sheet1", vtList(i))
Next i
End Sub

Example 2
Public Declare Function HypRetrieveNameRange Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtGridName As Variant) As Long
Sub RefreshNameRange()
sts = HypRetrieveNameRange("Sheet1", "DMDemo_Basic_2")
' if you know the name of the grid, you can pass it directly
End Sub

HypGetNameRangeList
Data source types: Essbase

Description
HypGetNameRangeList returns a list of named grids for a given connection.

Syntax
HypGetNameRangeList (vtSheetName, vtConnName, vtNameList)
ByVal vtSheetName As Variant
ByVal vtConnName As Variant
ByRef vtNameList As Variant

Parameters
vtSheetName: For future use. Currently the active sheet is used.
vtConnName: Input variable, contains the connection name whose specific list of name ranges must be retrieved. If this is empty then all the name range list in the sheet will be retrieved
vtNameList: Contains the list output.

Return Value
Returns 0 if successful; otherwise, returns the appropriate error code.
**Example**

```
Sub HypGetNameRangeList()
sts = HypGetNameRangeList("Sheet1", "stml0026_Sample_Basic", vtList)
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
```

**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**

```vba
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"
   stst = HypConnect (Empty, "system", "password", "Sample_Basic")
   stst = HypExecuteQuery (Empty, vtQuery)
   stst = HypDisconnect (Empty, True)
End sub
```

**HypSubmitData**

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

**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)

**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
**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
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
' 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
**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.")
End Sub
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**

```vba
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
```
Form Functions

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
Cell Functions

Cell functions perform operations and retrieve information for data cells and their contents.

- “HypGetDimMbrsForDataCell” on page 194
- “HypCell” on page 196
- “HypFreeDataPoint” on page 197
- “HypGetCellRangeForMbrCombination” on page 198
- “HypGetDataPoint” on page 199
- “HypIsCellWritable” on page 200
- “HypSetCellsDirty” on page 201
- “HypDeleteAllLROs” on page 202
- “HypDeleteLROs” on page 202
- “HypAddLRO” on page 203
- “HypUpdateLRO” on page 204
- “HypListLROs” on page 205
- “HypRetrieveLRO” on page 206
- “HypExecuteDrillThroughReport” on page 207
- “HypGetDrillThroughReports” on page 208

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.

vtServerName: Name of the server the associated connection on the sheet is connected to

vtApplicationName: Name of the application the associated connection on the sheet is connected to

vtCubeName: Name of the cube/database (Plan Type in Planning) the associated connection on the sheet is connected to

vtFormName: Name of the form the associated connection on the sheet is connected to (in ad hoc grids, this is returned as empty string)

vtDimensionNames: Array of dimension names

vtMemberNames: 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, _ ByVal vtServerName As Variant, ByVal vtAppName As Variant, _ ByVal vtCubeName As Variant, ByVal vtFormName As Variant, _ ByVal vtDimensionNames As Variant, ByVal 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$)
Sub HypGetDimMbrsForDataCell()
    Dim oRet As Long
    Dim oSheetDisp As Object
    Dim vtServerName As String
    Dim vtAppName As String
    Dim vtCubeName As String
    Dim vtFormName As String
    Dim vtDimNames() As Variant
    Dim vtMbrNames() As Variant

    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

---

**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())

**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.

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 199 for an example of HypFreeDataPoint.
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"
'tvtDimNames(4) = ""

vtMbrNames(0) = "Sales"
vtMbrNames(1) = "New York"
vtMbrNames(2) = "Year"
vtMbrNames(3) = "Product"
'tvtMbrNames(4) = ""

oRet = HypGetCellRangeForMbrCombination ("", vtDimNames, vtMbrNames, vtReturnCellRange)
If (oRet = 0) Then
    Set oRange = vtReturnCellRange
End If

**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

**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

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
**HypDeleteAllLROs**

Data source types: Essbase

**Description**

HypDeleteAllLROs() deletes all linked reporting objects from the cells specified by the vtSelectionRange parameter.

**Syntax**

HypDeleteAllLROs (vtSheetName, vtSelectionRange)

ByVal vtSheetName As Variant
ByVal vtSelectionRange As Variant

**Parameters**

- **vtSheetName**: For future use. Currently the active sheet is used.
- **vtSelectionRange**: The range of cells from which to delete all linked reporting objects

**Return Value**

Returns 0 if successful; otherwise, returns the appropriate error code.

**Example**

Public Declare Function HypDeleteAllLROs Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtSelectionRange As Variant) As Long

Sub HypDeleteAllLROs
sts = HypDeleteAllLROs("Sheet1", Range("B3"))
End Sub

**HypDeleteLROs**

**Description**

HypDeleteLROs() deletes one or more linked reporting objects from the cells specified by the vtSelectionRange parameter.

**Syntax**

HypDeleteLROs (vtSheetName, vtSelectionRange, vtLROIDs())

ByVal vtSheetName As Variant
ByVal vtSelectionRange As Variant
ByRef vtLROIDs() As Variant

**Parameters**

**vtSheetName:** For future use. Currently the active sheet is used.

**vtSelectionRange:** Input variable; the range of cells from which to delete all linked reporting objects

**vtLROIDs():** Input variable; the array of LRO Ids to be deleted.

**Return Value**

Returns 0 if successful; otherwise, returns the appropriate error code.

**Example**

Public Declare Function HypDeleteLROs Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtSelectionRange As Variant) As Long

Sub HypDeleteLROs()
sts = HypDeleteLROs("Sheet1", Range("B3"), LROIDs)
End Sub

**HypAddLRO**

**Data source types:** Essbase

**Description**

HypAddLRO() adds linked reporting objects to the cells specified by the vtSelectionRange parameter.

**Syntax**

HypAddLRO(vtSheetName, vtSelectionRange, vtIType, vtName, vtDescription)

ByVal vtSheetName As Variant
ByVal vtSelectionRange As Variant
ByVal vtIType As Variant
ByVal vtName As Variant
ByVal vtDescription As Variant

**Parameters**

**vtSheetName:** For future use; currently the active sheet is used.
vtSelectionRange: Input parameter; the range of cells to associate with the linked reporting object.

vtlType: Input parameter; the linked reporting object type expressed as a constant
- 1 - Cell note
- 2 - File
- 3 - URL

tvName: Input parameter; the location of the file with filename and URL information. Not used for cell note.

tvDescription: Input parameter; the description of the cell note, file, or URL.

Return Value
Returns 0 if successful; otherwise, returns the corresponding error code.

Example
Public Declare Function HypAddLRO Lib "HsAddin" (ByVal vtSheetName As Variant, _ ByVal vtSelectionRange As Variant, _ ByVal vtlType As Variant, _ByVal vtName As Variant, _ByVal vtDescription As Variant) As Long

Sub HypAddLRO()
    sts = HypAddLRO("Sheet1", Range("B3"), 1, ",", "Hello World")
End Sub

HypUpdateLRO

Description
HypUpdateLRO() updates linked reporting objects associated with the cells specified by the vtSelectionRange parameter.

Syntax
HypUpdateLRO(vtSheetName, vtSelectionRange, vtID,vtlType, vtName, vtDescription)
ByVal vtSheetName As Variant
ByVal vtSelectionRange As Variant
ByVal vtID As Variant
ByVal vtlType As Variant
ByVal vtName As Variant
ByVal vtDescription As Variant
Parameters

vtSheetName: For future use; currently the active sheet is used.

vtSelectionRange: Input parameter; the range of cells to associate with the linked reporting object.

vtID: Input parameter; the ID of the linked reporting object to be updated

vt1Type: Input parameter; the linked reporting object type expressed as a constant

- 1 - Cell note
- 2 - File
- 3 - URL

vtName: Input parameter; the location of the file with filename and URL information. Not used for cell note.

vtDescription: Input parameter; the description of the cell note, file, or URL.

Return Value

Returns 0 if successful; otherwise, returns the corresponding error code.

Example

Public Declare Function HypUpdateLRO Lib "HsAddin" (ByVal vtSheetName As Variant ByVal vtSelectionRange As Variant, ByVal vtID As Variant, ByVal vt1Type As Variant, ByVal vtName As Variant, ByVal vtDescription As Variant) As Long

Sub HypUpdateLRO
sts = HypUpdateLRO("Sheet1", Range("B3"), "2", 2, "d:\test2.txt", "linked object")
End Sub

HypListLROs

Data source types: Essbase

Description

HypListLROs() lists all linked reporting objects associated with the cells specified by the vtSelectionRange parameter.

Syntax

HypListLROs (vtSheetName, vtSelectionRange, vtLRO As LRO_Info)

ByVal vtSheetName As Variant

ByVal vtSelectionRange As Variant

ByRef vtLRO As LRO_Info
Parameters

vtSheetName: For future use. Currently the active sheet is used.

vtSelectionRange: Input parameter; the range of cells from which to list all linked reporting objects

vtLRO: Output parameter; 2-dimensional array of linked reporting objects

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

Public Declare Function HypListLROs Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtSelectionRange As Variant, ByRef vtLRO As LRO_Info) As Long

Dim ObjectList As LRO_Info
Sub HypListLROs()
sts = HypListLROs("Sheet1", Range("B3"), ObjectList)
End Sub

HypRetrieveLRO

Data source types: Essbase

Description

HypRetrieveLRO() retrieves linked reporting objects associated with the cells specified by the vtSelectionRange parameter.

Syntax

HypRetrieveLRO(vtSheetName, vtSelectionRange, vtID, vtType, vtName, vtDescription)

ByVal vtSheetName As Variant
ByVal vtSelectionRange As Variant
ByVal vtID As Variant
ByVal vtType As Variant
ByVal vtName As Variant
ByVal vtDescription As Variant

Parameters

vtSheetName: For future use; currently the active sheet is used.

vtSelectionRange: Input parameter; the range of cells to associate with the linked reporting object.
vtID: Input parameter; the ID of the linked reporting object to be retrieved. This is provided when you execute HypListLROs.

vtName: Output parameter; the name of the linked reporting object

vtDescription: Output parameter; the description of the retrieved linked reporting object.

**Return Value**

Returns 0 if successful; otherwise, returns the corresponding error code.

**Example**

Public Declare Function HypRetrieveLRO Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtSelectionRange As Variant, ByVal vtID, ByRef vtName, ByRef vtDescription) As Long

Sub HypRetrieveLRO
sts = HypRetrieveLRO("Sheet1", Range("B3"), "1", vtName, vtDescription)
End Sub

**HypExecuteDrillThroughReport**

**Data source types:** Essbase

**Description**

HypExecuteDrillThroughReport() executes the specified drill-through report. See also “HypGetDrillThroughReports” on page 208.

**Syntax**

HypExecuteDrillThroughReport(vtSheetName, vtSelectionRange, vtID, vtName, vtURL, vtURLTemplate, vtType)

- ByVal vtSheetName As Variant
- ByVal vtSelectionRange As Variant
- ByVal vtID As Variant
- ByVal vtName As Variant
- ByVal vtURL As Variant
- ByVal vtURLTemplate As Variant
- ByVal vtType As Variant

**Parameters**

vtSheetName: For future use; currently the active sheet is used.
vtSelectionRange: Input parameter; the range of cells in which to execute the drill-through report

vtID: Input parameter; the ID for the execution of the drill-through report. This is returned from the server when you run HypGetDrillThroughReport.

vtName: Input parameter; the name of the drill-through report. This is returned from the server when you run HypGetDrillThroughReport.

vtURL: Input parameter; the URL of the drill-through report. This is returned from the server when you run HypGetDrillThroughReport.

vtURLTemplate: Input parameter; the URL template of the drill-through report. This is returned from the server when you run HypGetDrillThroughReport.

vtType: Input parameter; the type of the drill-through report. This is returned from the server when you run HypGetDrillThroughReport.

**Return Value**

Returns 0 if successful; otherwise, returns the appropriate error code.

**Example**

```vba
Public Declare Function HypExecuteDrillThroughReport Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtSelectionRange As Variant, ByVal vtID As Variant, ByVal vtName As Variant, ByVal vtURL As Variant, ByVal vtURLTemplate As Variant, ByVal vtType As Variant) As Long

Sub HypExecuteDrillThroughReport()
    sts = HypExecuteDrillThroughReport("Sheet3", Range("B3"), ids(0), names(0), ",", ",", ")
End Sub
```

**HypGetDrillThroughReports**

Data source types: Essbase

**Description**

HypGetDrillThroughReports() retrieves a list of drill-through reports. See also "HypExecuteDrillThroughReport" on page 207.

**Syntax**

HypGetDrillThroughReports(vtSheetName, vtSelectionRange, vtIDs, vtNames, vtURLs, vtURLTemplates, vtTypes)

ByVal vtSheetName As Variant
ByVal vtSelectionRange As Variant
ByVal vtIDs As Variant
ByVal vtNames As Variant
ByVal vtURLs As Variant
ByVal vtURLTemplates As Variant
ByVal vtTypes As Variant

**Parameters**

**vtSheetName**: For future use; currently the active sheet is used

**vtSelectionRange**: The range of cells in which to execute the drill-through report

**vtIDs**: Output parameter; the array of the IDs returned from the server

**vtNames**: Output parameter; the array of the names returned from the server

**vtURLs**: Output parameter; the array of the URLs returned from the server

**vtURLTemplates**: Output parameter; the array of the URL templates returned from the server

**vtTypes**: Output parameter; the array of the types returned from the server

**Return Value**

Returns 0 if successful; otherwise, returns the appropriate error code.

**Example**

Public Declare Function HypGetDrillThroughReports Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtSelectionRange As Variant, ByRef vtIDs As Variant, ByRef vtNames As Variant, ByRef vtURLs As Variant, ByRef vtURLTemplates As Variant, ByRef vtTypes As Variant) As Long

Sub HypGetDrillThroughReports()
    sts = HypGetDrillThroughReports("Sheet3", Range("B3"), ids, names, urls, urltemplates, types)
End Sub

**POV Functions**

POV functions specify or retrieve settings for the POV.

- “HypSetPOV” on page 210
- “HypGetBackgroundPOV” on page 210
- “HypSetBackgroundPOV” on page 211
- “HypGetPagePOVChoices” on page 212
- “HypSetPages” on page 213
- “HypGetMembers” on page 214
- “HypSetMembers” on page 215
**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 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.

**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

**HypGetBackgroundPOV**

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

HypGetBackgroundPOV() returns the list of background POV members as two string arrays. One string array contains the POV dimension names; the other contains the member names.

**Syntax**

HypGetBackgroundPOV (vtFriendlyName, vtDimensionNames, vtMemberNames)

ByVal vtFriendlyName As Variant

ByRef vtDimensionNames As Variant

ByRef vtMemberNames As Variant

**Parameters**

vtFriendlyName: Input variable; connection name for the data source provider.

vtDimensionNames: Output variable; contains the dimension names array.

vtMemberNames: Output variable; contains the member names array (one member per POV dimension)

**Return Value**

Returns 0 if successful; otherwise, returns the appropriate error code.

**Example**

This example assumes that the worksheet is connected and has a grid.

```vba
Public Declare Function HypGetBackgroundPOV Lib "HsAddin" (ByVal vtFriendlyName As Variant, ByRef vtDimensionNames As Variant, ByRef vtMemberNames As Variant) As Long
Sub GetBackgroundPOV()
  sts = con = HypGetBackgroundPOV("stm10026_Sample_Basic", vtDim, vtMem)
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 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.

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

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 vtMbrDescChoices 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

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
HypGetMembers

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

Description

HypGetMembers() gets the list of selected or used members for a given dimension present in the grid.

For Essbase and Planning, member names are based on the selected alias table.

For Financial Management, the second array returns the descriptions.

For POV (forms), Page (in ad hoc) and user variables, a single member is returned.

To uniquely identify the user variable, provide the user variable name rather than the dimension name.

Syntax

HypGetMembers (vtSheetName, vtDimensionName, vtMbrNameChoices, vtMbrDescChoices)

ByVal vtSheetName As Variant

ByVal vtDimensionName As Variant

ByRef vtMbrNameChoices As Variant

ByRef vtMbrDescChoices As Variant

Parameters

vtSheetName: Input variable; contains the sheet name from which the list must be retrieved.

vtDimensionName: Input variable; contains the dimension name for which the used or selected member list must be returned

vtMbrNameChoices: Output variable; array of member names used.

vtMbrDescChoices: Output variable; array of member name descriptions. For Essbase and Planning, this is the same as member names. This list will be empty if the dimension is row/column type.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.
Example

This example assumes that the worksheet is connected and has a grid.

Public Declare Function HypGetMembers Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtDimensionName As Variant, ByRef vtMbrNameChoices As Variant, ByRef vtMbrDescChoices As Variant) As Long
Sub GetDimensionsMemberList()
sts = HypGetMembers("Sheet1", "Year", vtMbr, vtDes)
End Sub

HypSetMembers

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

Description

HypSetMembers() sets the list of POV dimension choices in ad hoc grids and the Page list in Financial Management forms.

This function cannot be used to set the Page list in Planning forms, nor can it be used to set row or column members.

The member list submitted by the user is validated before it is set.

Syntax

HypSetMembers (vtSheetName, vtDimensionName, ParamArray MemberList())
ByVal vtSheetName
ByVal vtDimensionName As Variant
ParamArray MemberList() As Variant

Parameters

vtSheetName: Input variable; contains the sheet name from which the list must be retrieved
vtDimensionName: Input variable; contains the dimension name for which the used or selected member list must be set
MemberList: Input variable; array of member names to be set as choices

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.
Example
This example assumes that the worksheet is connected and has a grid. Note: “InvalidMember”
does not belong to the Entity dimension and therefore will not be included in the list of
dimension choices.

Public Declare Function HypSetMembers Lib "HsAddin" (ByVal vtSheetName, ByVal
vtDimensionName As Variant, ParamArray MemberList() As Variant) As Long
Sub SetPOVChoices()
sts = HypSetMembers("Sheet1", "Entity", "Regional", "InvalidMember", "None")
End Sub

HypGetActiveMember

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

Description
HypGetActiveMember () returns the active member name for the given dimension. The active
member for page dimensions, POV dimensions, and user variables can be retrieved on ad hoc
or form worksheets. Row and column dimensions are not returned.

Syntax
HypGetActiveMember (vtDimName, vtMember)
ByVal vtDimName As Variant
ByRef vtMember As Variant

Parameters
vtDimName: Input variable; the dimension name whose active member must be retrieved.
vtMember: Output variable; the active member name returned.

Return Value
Returns 0 if successful; otherwise, returns the appropriate error code.

Example
This example assumes that the worksheet is connected and has a grid.

Public Declare Function HypGetActiveMember Lib "HsAddin" (ByVal vtDimName As Variant,
ByRef vtMember As Variant) As Long
Sub GetActiveMemberName()
sts = HypGetActiveMember("Market", vtMem)
End Sub
**HypSetActiveMember**

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

**Description**

HypSetActiveMember() sets the active member for a given dimension: page, POV, and user variables. Does not apply to row and column dimensions.

**Syntax**

HypSetActiveMember (vtDimName, vtMember)

ByVal vtDimName As Variant
ByVal vtMember As Variant

**Parameters**

vtDimName: Input variable; the dimension name whose active member must be changed or set.

vtMember: Input variable; the active member to be set.

**Return Value**

Returns 0 if successful; otherwise, returns the appropriate error code.

**Example**

This example assumes that the worksheet is connected and has a grid.

```vba
Public Declare Function HypSetActiveMember Lib "HsAddin" (ByVal vtDimName As Variant, ByVal vtMember As Variant) As Long
Sub SetActiveMemberName()
    sts = HypSetActiveMember("Market", "Washington")
End Sub
```

**HypGetDimensions**

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

HypGetBackgroundPOV() returns the list of background POV members as two string arrays. One string array contains the POV dimension names; the other contains the member names.

**Description**

HypGetDimensions() returns an array containing the dimension names in the grid and an array containing their corresponding types.

Type array has five possible types (row, column, page, POV, user variable), which can be identified using the following enumeration:
Enum DIMENSION_TYPE
    ROW_DIM = 0
    COL = 1
    POV = 2
    PAGE = 3
    USERVAR = 5
End Enum

To uniquely identify the user variable, use the user variable name rather than the dimension name.

Syntax

HypGetDimensions (vtSheetName, vtDimNames, vtType)
ByVal vtSheetName
ByRef vtDimNames As Variant
ByRef vtType As Variant

Parameters

vtSheetName: Input variable; contains the sheet name from which the list must be retrieved.
vtDimNames: Output variable; contains the dimension name array present in the grid.
vtType: Output variable; contains type information for the respective dimension.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

This example assumes that the worksheet is connected and has a grid.

Public Declare Function HypGetDimensions Lib "HsAddin" (ByVal vtSheetName, ByRef vtDimNames As Variant, ByRef vtType As Variant) As Long
Sub GetDimensionsOnGrid()
    sts = HypGetDimensions("Sheet1", vtDim, vtType)
End Sub

HypSetDimensions

Data source types: Essbase, Planning (ad hoc only), Financial Management (ad hoc only), Hyperion Enterprise (ad hoc only)

Description

HypSetDimensions() enables users to specify an array containing the dimension names in the grid and an array containing their corresponding types.
This function rearranges the metadata of the grid. It is used to set the grid layout when you want to start ad hoc with a specific layout, not with the default grid. Using this on an already used or established ad hoc report has the following effects:

- The entire grid layout is rearranged
- Comments, formulas, and formatting will be lost

Type array has five possible types (row, column, page, POV, user variable), which users can specify using the following enumeration:

```
Enum DIMENSION_TYPE
    ROW_DIM = 0
    COL = 1
    POV = 2
    PAGE = 3
    USERVERVAR = 5
End Enum
```

**Syntax**

```
HypSetDimensions(vtSheetName, vtDimNames(), vtType())
```

- **ByVal vtSheetName**: Input variable, contains the sheet name from which the list must be retrieved.
- **ByRef vtDimNames() As Variant**: Input variable, contains the dimension name array present in the grid.
- **ByRef vtType() As Variant**: Input variable, contains type information for the respective dimension.

**Parameters**

- **vtSheetName**: Input variable, contains the sheet name from which the list must be retrieved.
- **vtDimNames()**: Input variable, contains the dimension name array present in the grid.
- **vtType()**: Input variable, contains type information for the respective dimension.

**Return Value**

Returns 0 if successful; otherwise, returns the appropriate error code.

**Example**

This example assumes that the worksheet is connected.

```
Public Declare Function HypSetDimensions Lib "HsAddin" (ByVal vtSheetName, ByRef vtDimNames() As Variant, ByRef vtType() As Variant) As Long
Sub test()
    'assumption: sheet already connected and has a grid
    Dim dims(3) As Variant
    Dim types(3) As Variant
    dims(0) = "Product"
    dims(1) = "Market"
    dims(2) = "Scenario"
    dims(3) = "Measures"
```

Calculation Script and Business Rule Functions

Calculation script and business rule functions retrieve or execute calculation scripts and business rules.

- “HypListCalcScripts” on page 220
- “HypListCalcScriptsEx” on page 221
- “HypExecuteCalcScript” on page 222
- “HypExecuteCalcScriptEx” on page 223
- “HypDeleteCalc” on page 228

HypListCalcScripts

Data source types: Essbase

Description

HypListCalcScripts() lists all calculation scripts present on an Essbase 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.

**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], vtBRTyp [in], vtBRRHasPrompts [in], vtBBRNeedPageInfo [in], vtRTPNames() [in], vtRTPValues() [in], vtRTPShowRTPDlg [in], vtRTRuleOnForm [in], vtBRRanSuccessfully [out], vtCubeName [out], vtBRName [out], vtBRTyp [out], vtBRRHasPrompts [out], vtBBRNeedPageInfo [out], vtBRRHidePrompts [out], vtRTPNamesUsed [out], vtRTPValuesUsed [out])

ByVal vtSheetName As Variant
ByVal vtCubeName As Variant
ByVal vtBRName As Variant
ByVal vtBRTyp As Variant
ByVal vtBRRHasPrompts As Variant
ByVal vtBBRNeedPageInfo As Variant
ByRef vtRTPNames() As Variant
ByRef vtRTPValues() As Variant
ByVal vtRTPShowRTPDlg As Variant
ByVal vtRTRuleOnForm As Variant
ByRef vtBRRanSuccessfully As Variant
ByRef vtCubeName As Variant
ByRef vtBRName As Variant
ByRef vtBRTyp 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

vtBRTyp: Business Rule Type for the BR to be run

vtbBRHasPrompts: 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 parameters 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

pvbBRRanSuccessfully: 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

pvtBRTypUsed: 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 vtbrRuleOnForm 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 vtBTType As Variant, ByVal vtbrBRHasPrompts As Variant, ByVal vtBRNeedPageInfo As Variant, ByRef vtRTPNames() As Variant, ByRef vtRTPValues() As Variant, ByVal vtbrShowRTPDlg As Variant, ByVal vtbrRuleOnForm As Variant, ByRef vtbrBRRanSuccessfully As Variant, ByRef vtCubeName As Variant, ByRef vtBRName As Variant, ByRef vtBRType As Variant, ByRef vtbrBRHasPrompts As Variant, ByRef vtbrBRNeedPageInfo As Variant, ByRef vtbrBRHidePrompts 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 vtBRranSuccessfully As Variant
Dim vtBRranSuccessfully2 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
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.

**HypDeleteCalc**

Data source types: Essbase

**Description**

HypDeleteCalc() deletes a calculation script from an Essbase 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
Calculation, Consolidation, and Translation Functions

Calculation, consolidation, and translation functions executes these operations on data for Financial Management and Hyperion Enterprise applications. See “Calculation, Consolidation, and Translation Functions” on page 229

- “HypCalculate” on page 229
- “HypCalculateContribution” on page 230
- “HypConsolidate” on page 230
- “HypConsolidateAll” on page 231
- “HypConsolidateAllWithData” on page 232
- “HypForceCalculate” on page 232
- “HypForceCalculateContribution” on page 233
- “HypForceTranslate” on page 234
- “HypTranslate” on page 235

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)

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)

**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

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)
**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)

**Member Query Functions**

Member query functions retrieve generation, level, attribute, and other information about members.

The member query functions:

- “HypFindMember” on page 236
- “HypFindMemberEx” on page 237
- “HypGetAncestor” on page 238
- “HypGetChildren” on page 239
- “HypGetParent” on page 240
- “HypIsAttribute” on page 241
- “HypIsDescendant” on page 242
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)

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**

```vba
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
```

**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
```

---

238  VBA Functions
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

**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
```

**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

**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

**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

**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
HypIsAncestor (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
**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 vtParentName As Variant) As Boolean

Sub Sample_HypIsParent
    Dim b as Boolean
    b = HypIsParent (Empty, "East", "Market")
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
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 vtChildName As Variant) As Variant

Sub Sample_HypIsChild
    Dim b as Boolean
    b = HypIsChild ("Sheet1", "Year", "Qtr1")
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

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

HypQueryMembers

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 HYPSAMEGENERATION
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
16 HYP_DIMUSERATTRIBUTES

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 search string)
- 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)

250 VBA Functions
' 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)

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

**HypGetMemberInformation**

Data source types: Essbase

**Description**

HypGetMemberInformation returns the properties for a selected member.

**Syntax**

HypGetMemberInformation (vtSheetName, vtMemberName, vtPropertyName, vtPropertyValue, vtPropertyValuesString)

ByVal vtMemberName As Variant
ByVal vtPropertyName As Variant
ByVal vtPropertyValue As Variant
ByRef vtPropertyValuesString As Variant

**Parameters**

**vtSheetName**: For future use. Currently the active sheet is used.
vtMemberName: Input parameter; the name of the member whose properties are to be returned.

vtPropertyName: Output parameter; the name of the property for which information is required. See Table 15.

vtPropertyValue: Input parameter; the property array for the member, returned as numerical value from the server.

vtPropertyValuesString: Output parameter; the property array for the member, returned as string equivalent of numerical value for properties for which numerical values do not make sense.

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

HypGetMemberInformation Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtMemberName As Variant, ByVal vtPropertyName As Variant, ByRef vtPropertyValue As Variant, ByRef vtPropertyValuesString As Variant) As Long

Sub HypGetMemberInformation
sts = HypGetMemberInformation("Sheet1", "Jan", HYP_MI_NAME, vtValues, vtPropertyValuesString)
End Sub

Table 15  Constants for Member Information

<table>
<thead>
<tr>
<th>Constants for Member Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global Const HYP_MI_NAME = &quot;Name&quot;</td>
</tr>
<tr>
<td>Global Const HYP_MI_DIM = &quot;Dim&quot;</td>
</tr>
<tr>
<td>Global Const HYP_MI_LEVEL = &quot;Level&quot;</td>
</tr>
<tr>
<td>Global Const HYP_MI_GENERATION = &quot;Generation&quot;</td>
</tr>
<tr>
<td>Global Const HYP_MI_PARENT_MEMBER_NAME = &quot;ParentMbrName&quot;</td>
</tr>
<tr>
<td>Global Const HYP_MI_CHILD_MEMBER_NAME = &quot;ChildMbrName&quot;</td>
</tr>
<tr>
<td>Global Const HYP_MI_PREVIOUS_MEMBER_NAME = &quot;PrevMbrName&quot;</td>
</tr>
<tr>
<td>Global Const HYP_MI_NEXT_MEMBER_NAME = &quot;NextMbrName&quot;</td>
</tr>
<tr>
<td>Global Const HYP_MI_CONsolidation = &quot;Consolidation&quot;</td>
</tr>
<tr>
<td>Global Const HYP_MI_IS_TWO_PASS_CAL_MEMBER = &quot;IsTwoPassCalcMbr&quot;</td>
</tr>
<tr>
<td>Global Const HYP_MI_IS_EXPENSE_MEMBER = &quot;IsExpenseMbr&quot;</td>
</tr>
<tr>
<td>Global Const HYP_MI_CURRENCY_CONVERSION_TYPE = &quot;CurrencyConversionType&quot;</td>
</tr>
<tr>
<td>Global Const HYP_MI_CURRENCY_CATEGORY = &quot;CurrencyCategory&quot;</td>
</tr>
</tbody>
</table>
**Constants for Member Information**

```vba
Global Const HYP_MI_TIME_BALANCE_OPTION = "TimeBalanceOption"
Global Const HYP_MI_TIME_BALANCE_SKIP_OPTION = "TimeBalanceSkipOption"
Global Const HYP_MI_SHARE_OPTION = "ShareOption"
Global Const HYP_MI_STORAGE_CATEGORY = "StorageCategory"
Global Const HYP_MI_CHILD_COUNT = "ChildCount"
Global Const HYP_MI_ATRIBUTED = "Attributed"
Global Const HYP_MI_RELATIONAL_DESCENDENT_PRESENT = "RelDescendantPresent"
Global Const HYP_MI_RELATIONAL_PARTITION_ENABLED = "RelPartitionEnabled"
Global Const HYP_MI_DEFAULT_ALIAS = "DefaultAlias"
Global Const HYP_MI_HIERARCHY_TYPE = "HierarchyType"
Global Const HYP_MI_DIM_SOLVE_ORDER = "DimSolveOrder"
Global Const HYP_MI_IS_DUPLICATE_NAME = "IsDuplicateName"
Global Const HYP_MI_UNIQUE_NAME = "UniqueName"
Global Const HYP_MI_ORIGINAL_MEMBER = "OrigMember"
Global Const HYP_MI_IS_FLOW_TYPE = "IsFlowType"
Global Const HYP_MI_AGGREGATE_LEVEL = "AggLevel"
Global Const HYP_MI_FORMAT_STRING = "FormatString"
Global Const HYP_MI_ATTRIBUTE_DIMENSIONS = "AttributeDims"
Global Const HYP_MI_ATTRIBUTE_MEMBERS = "AttributeMbrs"
Global Const HYP_MI_ATTRIBUTE_TYPES = "AttributeTypes"
Global Const HYP_MI_ALIAS_NAMES = "AliasNames"
Global Const HYP_MI_ALIAS_TABLES = "AliasTables"
Global Const HYP_MI_FORMULA = "Formula"
Global Const HYP_MI_COMMENT = "Comment"
Global Const HYP_MI_LAST_FORMULA = "LastFormula"
Global Const HYP_MI_UDAS = "Udas"
```

**HypGetMemberInformationEx**

Data source types: Essbase
Description
HypGetMemberInformationEx returns all information about a member in an array.

Syntax
HypGetMemberInformationEx (vtSheetName, vtMemberName, vtPropertyNames, vtPropertyValues, vtPropertyValueStrings)
ByVal vtSheetName As Variant
ByVal vtMemberName As Variant
ByRef vtPropertyNames As Variant
ByRef vtPropertyValues As Variant
vtPropertyValueStrings As Variant

Parameters
vtSheetName: For future use currently the active sheet is used.
vtMemberName: The name of the members whose properties are to be returned.
vtPropertyNames: The property name array
vtPropertyValues: The property value array
vtPropertyValueStrings: The property string value array

Return Value
Returns 0 if successful; otherwise, returns the appropriate error code.

Example
Public Declare Function HypGetMemberInformationEx Lib "HsAddin" (ByVal vtSheetName As Variant, ByVal vtMemberName As Variant, ByRef vtPropertyNames As Variant, ByRef vtPropertyValues As Variant, ByRef vtPropertyValueStrings As Variant) As Long
sub HypGetMemberInformationEx()
    stst = HypGetMemberInformationEx(Empty, "100-10", proper tynames, propertyvalues, propertyvaluestrings)
End Sub

Options Functions
Options functions set and retrieve information for global and/or sheet options, and enable deletion of MRU items.
The options functions:
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 14, “Smart View Options.”

Syntax

HypGetGlobalOption(vtItem)

ByVal vtItem As Long

Parameters

vtItem: Number that indicates which option is to be retrieved.

Table 16 lists the numbers of options and their return data types.

Table 16  HypGetGlobalOption Parameter Numbers and Options

<table>
<thead>
<tr>
<th>vtItem</th>
<th>Option</th>
<th>Return Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Enable Excel formatting</td>
<td>Boolean</td>
</tr>
<tr>
<td>2</td>
<td>Enable double-click for ad hoc operations</td>
<td>Boolean</td>
</tr>
<tr>
<td>3</td>
<td>Enable undo</td>
<td>Boolean</td>
</tr>
<tr>
<td>4</td>
<td>Not used</td>
<td>Boolean</td>
</tr>
<tr>
<td>vtItem</td>
<td>Option</td>
<td>Return Data Type</td>
</tr>
<tr>
<td>--------</td>
<td>----------------------------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>5</td>
<td>Specify message level setting:</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td>0 Information messages</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 Warning messages</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 Error messages</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 No messages</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 Debug</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Use thousands separator</td>
<td>Boolean</td>
</tr>
<tr>
<td>7</td>
<td>Enable route messages to log file</td>
<td>Boolean</td>
</tr>
<tr>
<td>8</td>
<td>Clear log file on next launch</td>
<td>Boolean</td>
</tr>
<tr>
<td>9</td>
<td>Enable navigate without data</td>
<td>Boolean</td>
</tr>
<tr>
<td>10</td>
<td>Display Member Selection save</td>
<td>Boolean</td>
</tr>
<tr>
<td>11</td>
<td>Enable double-clicking to browse LRO</td>
<td>Boolean</td>
</tr>
<tr>
<td>12</td>
<td>Specify Meaningless label</td>
<td>Text</td>
</tr>
<tr>
<td>13</td>
<td>Reduce Excel file size</td>
<td>Boolean</td>
</tr>
<tr>
<td>14</td>
<td>Enable formatted strings</td>
<td>Boolean</td>
</tr>
<tr>
<td>15</td>
<td>Retain numeric formatting</td>
<td>Boolean</td>
</tr>
<tr>
<td>16</td>
<td>Enable enhanced comment handling</td>
<td>Boolean</td>
</tr>
<tr>
<td>17</td>
<td>Enable retain ribbon context</td>
<td>Boolean</td>
</tr>
<tr>
<td>18</td>
<td>Display Smart View Panel on start-up</td>
<td>Boolean</td>
</tr>
</tbody>
</table>

**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.

```vba
Declare Function HypGetGlobalOption Lib "HsAddin" (ByVal vtItem As Long) As Variant

Sub GetGlobal()
    sts = HypGetGlobalOption(5)
    If sts = -15 Then
        MsgBox("Invalid Parameter")
    Else
        ' Further code
    End If
End Sub
```
MsgBox ("Message level is set to" & sts)
End Sub

**HypSetGlobalOption**

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

**Description**

HypSetGlobalOption() sets individual workspace options. For option descriptions, see Chapter 14, “Smart View 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 16, “HypGetGlobalOption Parameter Numbers and Options,” on page 255 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
**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 17 for a list of values.

<table>
<thead>
<tr>
<th>vtItem</th>
<th>Option</th>
<th>Data Type and Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Set zoom in level:</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td>0 Next level</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 All levels</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 Bottom level</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 Sibling level</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 Same level</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 Same generation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6 Formulas</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Enable Include Selection setting</td>
<td>Boolean</td>
</tr>
<tr>
<td>3</td>
<td>Enable Within Selection Group setting</td>
<td>Boolean</td>
</tr>
<tr>
<td>4</td>
<td>Enable Remove Unselected Groups setting</td>
<td>Boolean</td>
</tr>
<tr>
<td>5</td>
<td>Specify Indent setting:</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td>0 No indentation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 Indent sub items</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 Indent totals</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Enable suppress missing setting</td>
<td>Boolean</td>
</tr>
<tr>
<td>7</td>
<td>Enable suppress zeros setting</td>
<td>Boolean</td>
</tr>
<tr>
<td>8</td>
<td>Enable suppress underscores setting</td>
<td>Boolean</td>
</tr>
<tr>
<td>vtItem</td>
<td>Option</td>
<td>Data Type and Values</td>
</tr>
<tr>
<td>--------</td>
<td>---------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>9</td>
<td>Enable No Access setting</td>
<td>Boolean</td>
</tr>
<tr>
<td>10</td>
<td>Enable Repeated Member setting</td>
<td>Boolean</td>
</tr>
<tr>
<td>11</td>
<td>Enable Invalid setting</td>
<td>Boolean</td>
</tr>
<tr>
<td>12</td>
<td>Ancestor Position:</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td>0 Top</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 Bottom</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Specify Missing Text label</td>
<td>Text</td>
</tr>
<tr>
<td>14</td>
<td>Specify No Access label</td>
<td>Text</td>
</tr>
<tr>
<td>15</td>
<td>Cell Status:</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td>0 Data</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 Calculation Status</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 Process Management</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Member Name Display options:</td>
<td>Number</td>
</tr>
<tr>
<td></td>
<td>0 Name Only</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 Name and Description</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 Description only</td>
<td></td>
</tr>
</tbody>
</table>

**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

**HypSetSheetOption**

Data source types: Essbase, Planning, Financial Management, Hyperion Enterprise
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 17 on page 258 for a list of values.
vtOption: A Boolean value denoting the new value of item.
Table 17 on page 258 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

HypGetOption

Data Source types: Essbase
Description
HypGetOption() enables you to get Smart View options that are both default and sheet specific so you do not need separate VBA commands for the two types of options.

Syntax
HypGetOption (vtItem, vtRet, vtSheetName)
ByVal vtItem As Variant
ByRef vtRet As Variant
ByVal vtSheetName As Variant

Parameters
vtItem: The index/constant to refer to a particular option. See Table 18.
vtRet: The output variable
vtSheetName: The sheet name of a sheet level option. If not provided, then the default option is used.

Return Value
Returns 0 if successful; otherwise, returns the appropriate error code.

Example
Public Declare Function HypGetOption Lib "HsAddin" (ByVal vtItem As Variant, ByRef vtRet As Variant, ByVal vtSheetName As Variant) As Long
Sub getOption()
    sts = HypGetOption(HSV_ZOOMIN, Var, "Sheet2") 'get zoom in option for sheet2
    sts = HypGetOption(1, Var, "") 'get default zoom in option
End Sub

Table 18  Options Constants for HypGetOption and HypSetOption

<table>
<thead>
<tr>
<th>Options Constants for HypGetOption and HypSetOption</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSV_ZOOMIN = 1</td>
</tr>
<tr>
<td>HSV_INCLUDE_SELECTION</td>
</tr>
<tr>
<td>HSV_WITHIN_SELECTEDGROUP</td>
</tr>
<tr>
<td>HSV_REMOVE_UNSELECTEDGROUP</td>
</tr>
<tr>
<td>HSV_INDENTATION</td>
</tr>
<tr>
<td>Options Constants for HypGetOption and HypSetOption</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
</tr>
<tr>
<td>HSV_SUPPRESSROWS_MISSING</td>
</tr>
<tr>
<td>HSV_SUPPRESSROWS_ZEROS</td>
</tr>
<tr>
<td>HSV_SUPPRESSROWS_UNDERSCORE</td>
</tr>
<tr>
<td>HSV_SUPPRESSROWS_NOACCESS</td>
</tr>
<tr>
<td>HSV_SUPPRESSROWS_REPEATEDMEMBERS</td>
</tr>
<tr>
<td>HSV_SUPPRESSROWS_INVALID</td>
</tr>
<tr>
<td>HSV_ANCESTOR_POSITION</td>
</tr>
<tr>
<td>HSV_MISSING_LABEL</td>
</tr>
<tr>
<td>HSV_NOACCESS_LABEL</td>
</tr>
<tr>
<td>HSV_CELL_STATUS</td>
</tr>
<tr>
<td>HSV_MEMBER_DISPLAY</td>
</tr>
<tr>
<td>HSV_INVALID_LABEL</td>
</tr>
<tr>
<td>HSV_SUBMITZERO</td>
</tr>
<tr>
<td>HSV_19 Reserved for future use</td>
</tr>
<tr>
<td>HSV_20 Reserved for future use</td>
</tr>
<tr>
<td>HSV_PRESERVE_FORMULA_COMMENT</td>
</tr>
<tr>
<td>HSV_22 Reserved for future use</td>
</tr>
<tr>
<td>HSV_FORMULA_FILL</td>
</tr>
<tr>
<td>HSV_EXCEL_FORMATTING = 30</td>
</tr>
<tr>
<td>HSV_RETAIN_NUMERIC_FORMATTING</td>
</tr>
<tr>
<td>HSV_THOUSAND_SEPARATOR</td>
</tr>
<tr>
<td>HSV_NAVIGATEWITHOUTDATA</td>
</tr>
<tr>
<td>HSV_ENABLE_FORMATSTRING</td>
</tr>
<tr>
<td>HSV_ENHANCED_COMMENT_HANDLING</td>
</tr>
<tr>
<td>HSV_ADJUSTCOLUMNWIDTH</td>
</tr>
<tr>
<td>HSV_DOUBLECLICK_FOR_ADHOC = 101</td>
</tr>
<tr>
<td>HSV_UNDO_ENABLE</td>
</tr>
<tr>
<td>HSV_103 Reserved for future use</td>
</tr>
</tbody>
</table>
Options Constants for HypGetOption and HypSetOption

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSV_LOGMESSAGE_DISPLAY</td>
</tr>
<tr>
<td>HSV_ROUTE_LOGMESSAGE_TO_FILE</td>
</tr>
<tr>
<td>HSV_CLEAR_LOG_ON_NEXTLAUNCH</td>
</tr>
<tr>
<td>HSV_REDUCE_EXCEL_FILESIZE</td>
</tr>
<tr>
<td>HSV_ENABLE_RIBBON_CONTEXT</td>
</tr>
<tr>
<td>HSV_DISPLAY_HOMEPAGEAL_ONSTARTUP</td>
</tr>
<tr>
<td>HSV_SHOW_COMMENTDIALOG_ON_REFRESH</td>
</tr>
<tr>
<td>HSV_NUMBER_OF_UNDO_ACTION</td>
</tr>
<tr>
<td>HSV_NUMBER_OF_MRU_ITEMS</td>
</tr>
<tr>
<td>HSV_ROUTE_LOGMESSAGE_FILE_LOCATION</td>
</tr>
<tr>
<td>HSV_DISABLE_SMARTVIEW_IN_OUTLOOK</td>
</tr>
<tr>
<td>HSV_DISPLAY_SMARTVIEW_SHORTCUT_MENU_ONLY</td>
</tr>
<tr>
<td>HSV_DISPLAY_DRILL_THROUGH_REPORT_TOOLTIP</td>
</tr>
</tbody>
</table>

HypSetOption

**Data Source types:** Essbase

**Description**

HypSetOption() enables you to set Smart View options as both default and sheet specific so you do not need separate VBA commands for the two types of options.

**Syntax**

HypSetOption (vtItem,vtOption,vtSheetName)

ByVal vtItem As Variant

ByVal vtOption As Variant

ByVal vtSheetName As Variant

**Parameters**

- **vtItem:** The index/constant to refer to a particular option See Table 18 on page 261 for a list of constants.

- **vtOption:** The input value to set for an option
vtSheetName: The sheet name to set a sheet level option. If not provided, then the default option is used.

Return Value
Returns 0 if successful; otherwise, returns the appropriate error code.

Example
Public Declare Function HypSetOption Lib "HsAddin" (ByVal vtItem As Variant, ByVal vtOption As Variant, ByVal vtSheetName As Variant) As Long
Sub setOption()
    sts = HypSetOption(HSV_ZOOMIN, 2, "Sheet2") ' set zoom in option for sheet2
    sts = HypSetOption(HSV_ZOOMIN, 1, "") ' set default zoom in
    sts = HypSetOption(HSV_INVALID_LABEL, "#InvalidTest", "Sheet2") ' set invalid label for sheet2
    sts = HypSetOption(17, "#globalinvalid", "") ' set default invalid label, numbers can be used instead of declared constants
End Sub

HypDeleteAllMRUItems

Data Source types: All

Description
HypDeleteAllMRUItems () deletes all items in the most recently used list, including those that are pinned to the list.

Syntax
HypDeleteAllMRUItems Lib "HsAddin" () As Long

Return Value
Returns 0 if successful; otherwise, returns the appropriate error code.

Example
Public Declare Function HypDeleteAllMRUItems Lib "HsAddin" () As Long
Sub HypDeleteAllMRUItems ()
    sts = HypDeleteAllMRUItems()
End Sub
MDX Query Functions

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**

```vba
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"
vAliasTable = "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)
```

266 VBA Functions
Menu Functions

VBA menu functions are identical to the equivalent commands on the Smart View menu and ribbon. The requirements for the functions are the same as those for the menu commands. For example, if you must be logged in to an Essbase server to use a menu command, then you must also be logged in to an Essbase server to use the equivalent VBA.

The menu equivalent functions:

- HypMenuVAbout
- HypMenuVAdjust
- HypMenuVBusinessRules
- HypMenuVCalculation
- HypMenuVCascadeNewWorkbook
- HypMenuVCascadeSameWorkbook
- HypMenuVCellText
- HypMenuVCollapse
- HypMenuVConnect
- HypMenuVCopyDataPoints
- HypMenuVExpand
- HypMenuVFunctionBuilder
- HypMenuVInstruction
- HypMenuVKeepOnly
- HypMenuVLRO
- HypMenuVMemberInformation
- HypMenuVMemberSelection
- HypMenuVMigrate
- HypMenuVOptions
- HypMenuVPasteDataPoints
- HypMenuVPivot
- HypMenuVPOVManager
- HypMenuVQueryDesigner
- HypMenuVRedo
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**

```vba
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**

```vba
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

HypMenuVCascadeNewWorkbook

Data source types: Essbase, Planning, Hyperion Enterprise

Description
HypMenuVCascadeNewWorkbook() cascades separate reports for members of one dimension across worksheets of a newly-opened Excel workbook.

Syntax
HypMenuVCascadeNewWorkbook()

Return Value
Returns 0 if successful; otherwise, returns the appropriate error code.

Example
Public Declare Function HypMenuVCascadeNewWorkbook Lib "HsAddin" () As Long
Sub MCascadeNewWorkbook()
HypMenuVCascadeSameWorkbook

Data source types: Essbase, Planning, Hyperion Enterprise

Description

HypMenuVCascadeSameWorkbook() cascades separate reports for members of one dimension across worksheets of the current Excel workbook.

Syntax

HypMenuVCascadeSameWorkbook()

Return Value

Returns 0 if successful; otherwise, returns the appropriate error code.

Example

Public Declare Function HypMenuVCascadeSameWorkbook Lib "HsAddin" () As Long

Sub MCascadeSameWorkbook()
    X=HypMenuVCascadeSameWorkbook()
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.
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**

```vba
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**

```vba
Declare Function HypMenuVKeepOnly Lib "HsAddin"() As Long

Sub MKeepOnly()
    X=HypMenuVKeepOnly()
End Sub
```

**HypMenuVLRO**

**Data source types:** Essbase

**Description**

HypMenuVLRO() opens the Linked Objects dialog box when you select cells to edit linked objects.
**Syntax**
HypMenuVLRO()

**Return Value**
Returns 0 if successful; otherwise, returns the appropriate error code.

**Example**
Public Declare Function HypMenuVLRO Lib "HsAddin" () As Long
Sub MLRO()
    X=HypMenuVLRO()
End Sub

**HypMenuVMemberInformation**

**Description**
HypMenuVMemberInformation() opens the Member Information dialog box where you view detailed information about members.

**Syntax**
HypMenuVMemberInformation()

**Return Value**
Returns 0 if successful; otherwise, returns the appropriate error code.

**Example**
Public Declare Function HypMenuVMemberInformation Lib "HsAddin" () As Long
Sub MMemberInformation()
    X=HypMenuVMemberInformation()
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**
```vba
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**
```vba
HypMenuVPivot()
```

**Return Value**
Returns 0 if successful; otherwise, returns the appropriate error code.

**Example**
```vba
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**

```vba
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**

```vba
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**

```vba
HypMenuVRedo()
```

**Return Value**

Returns 0 if successful; otherwise, returns the appropriate error code.

**Example**

```vba
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**

```vba
HypMenuVRefresh()
```

**Return Value**

Returns 0 if successful; otherwise, returns the appropriate error code.

**Example**

```vba
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, 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, 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

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**

```vba
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**

```vba
HypMenuVZoomOut()
```

**Return Value**

Returns 0 if successful; otherwise, returns the appropriate error code.

**Example**

```vba
Declare Function HypMenuVZoomOut Lib "HsAddin"() As Long
Sub MZoomOut()
    X = HypMenuVZoomOut()
End Sub
```

**Dynamic Link Functions**

You can use static or dynamic link views to display details about a data point in an adjacent window without disturbing the contents in the main window. Static link views are predefined and are built into Smart View. With dynamic link views, you can use the VBA functions in this section to change row, column, POV, and connection information.

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.
The dynamic link functions:

- “HypUseLinkMacro” on page 291
- “HypSetLinkMacro” on page 292
- “HypGetLinkMacro” on page 292
- “HypGetSourceGrid” on page 293
- “HypDisplayToLinkView” on page 294
- “HypGetConnectionInfo” on page 295
- “HypSetConnectionInfo” on page 297
- “HypGetRowCount” on page 298
- “HypGetColCount” on page 299
- “HypGetPOVCount” on page 300
- “HypGetRowItems” on page 300
- “HypSetRowItems” on page 301
- “HypGetColItems” on page 302
- “HypSetColItems” on page 303
- “HypGetPOVIItems” on page 304
- “HypSetPOVIItems” on page 305

**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. (When the `HypUseLinkMacro` flag is set to false, the predefined link query is performed.)
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.

**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.

**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.

**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
**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.

**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

**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.

**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

**HypGetSourceGrid**

**Data source types:** Essbase, Planning (adhoc only), Financial Management (adhoc only), Hyperion Enterprise (adhoc 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.

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

**Note:** A cell in the grid must be selected before this making this call.

**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

HypDisplayToLinkView

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

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: 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 worksheet.

Note: After a call, the password is not actually returned, but for security reasons, is returned as empty.
Syntax

HypGetConnectionInfo(vtServerName, vtUserName, vtApplicationName, vtDatabaseName, vtFriendlyName, vtURL, vtProviderType)
ByRef vtServerName As Variant
ByRef vtUserName 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.
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 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 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, app, db, conn, url, provider)
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:** 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 worksheet.

**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)
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:** 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 worksheet.

**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

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: 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 worksheet.

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
**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:** 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 worksheet.

**Syntax**

HypGetPOVCount()

**Return Value**

Returns the number of column dimensions if successful; otherwise, returns the negative error code.

**Example**

```vba
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
```

**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: 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 worksheet.

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", "DemoBasic_Connection")
    Sts = HypRetrieve(Empty)
    Range ("B2").Select
    Sts = HypGetSourceGrid (Empty, vtGrid)
    Sts = HypGetRowItems(1, vtDimName, vtMembers)
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:** 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 worksheet.

**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 ppMemberList() 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

**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: 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 worksheet.

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

HypSetColItems

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: 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 worksheet.

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

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:** 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 worksheet.

**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

**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:** 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 worksheet.
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

Using Spreadsheet Toolkit VBA Applications in Smart View

VBA applications created in Oracle 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 Hyperion Essbase Spreadsheet Toolkit menu functions with HypMenuV; for example, change EssMenuVZoomIn to HypMenuVZoomIn.
- Replace the declarations in essxlvba.txt with the declarations in smartview.bas.
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, member selection, and other ad hoc operations in free-form grids.

The components of Smart View grids are described in Table 19.

<table>
<thead>
<tr>
<th>Grid Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row Dimension</td>
<td>A dimension or member placed down one column across one or more rows in a worksheet</td>
</tr>
<tr>
<td>Column Dimension</td>
<td>A dimension or member placed on a row across one or more columns in a worksheet</td>
</tr>
<tr>
<td>Page Dimension</td>
<td>A dimension that applies to the entire page (Essbase only)</td>
</tr>
<tr>
<td>Comments</td>
<td>Text added by the user</td>
</tr>
<tr>
<td>Data Region</td>
<td>Areas of the grid that contain data for dimensions or members</td>
</tr>
<tr>
<td>Blank Region</td>
<td>Areas of the worksheet that contain no entries</td>
</tr>
</tbody>
</table>
Free-Form Guidelines

- Grids do not need to start in cell A1.
- 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.
- 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, and as comments outside the data region. 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 this syntax for qualified member names:
  
  `[Income] . [Other]`
  
  `[Expenses] . [Other]`
- Aliases from the current alias table are permitted in free-form grids, but aliases from other alias tables are treated as comments.
- For Hyperion Enterprise data sources, you cannot type dimension names in the free-form grids. You can type only member names.
- Dynamic Time Series members (Essbase) must use one of the following formats:
  - Q-T-D(Jan)
  - Y-T-D(Mar)
  - M-T-D(Jun)
Free-Form Grid Examples

Simple Grids

Figure 13 shows a valid simple grid, where Year is the row dimension, Measures is the column dimension and Product is the page dimension.

Figure 13 Simple Grid

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>Product</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>Profit</td>
<td>Inventory</td>
<td>Ratios</td>
<td>Measures</td>
</tr>
<tr>
<td>3</td>
<td>Qtr1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Qtr2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Qtr3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Qtr4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Year</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 14 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 14 Two Columns by Two Rows Layout

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>Product</td>
<td>Product</td>
<td>Product</td>
<td>Product</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>Profit</td>
<td>Inventory</td>
<td>Ratios</td>
<td>Measures</td>
</tr>
<tr>
<td>3</td>
<td>Market</td>
<td>Qtr1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Market</td>
<td>Qtr2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Market</td>
<td>Qtr3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Market</td>
<td>Qtr4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Market</td>
<td>Year</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Column Dimensions

Column Dimensions Interpreted as Page Dimensions

When there is one row dimension and multiple members, all of different dimensions, in the same top row, the leftmost dimension in the row is treated as column dimension and the others as page dimensions. Figure 15 shows a valid grid in which Year is the row dimension, Measure is the column dimension, and Product and Market are page dimensions.

Figure 15 Column and Page Dimensions in the First Row

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Measures</td>
<td>Product</td>
<td>Market</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Year</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The first members of each column dimension must occur on the same column, and the first members of every row dimension must occur on the same row. Figure 16 is invalid because cell B2 is on the first column of the column dimensions and must be a member of the Measures dimension, whereas it is a comment.

![Figure 16 Invalid Placement in Column](image)

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>100-10</td>
<td>100-30</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td>Comment</td>
<td></td>
<td>Measures</td>
<td>Measure</td>
</tr>
<tr>
<td>3</td>
<td>Year</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Stacked Dimensions**

The first row that contains multiple members of the same dimension is a column dimension. All dimensions placed above this row are candidates for page dimension if they comply with the rules for page dimension. However, dimensions above this column dimension that are in the same column (“stacked”) and have no other members are column dimensions and not page dimensions. Such grids are not valid.

In Figure 17, Product is a column dimension that is stacked on Profit; Market is a page dimension, because it is not stacked on Profit. Scenario is a page dimension, even though it is stacked on Profit, because its row is above a page dimension.

![Figure 17 Stacked Dimensions as Page Dimensions](image)

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>Scenario</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>Market</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>Product</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>Profit</td>
<td>Ratio</td>
<td>Measures</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Year</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In Figure 18, Product and Market are stacked above a column dimension and contain no other members. Therefore, this grid is not valid.

![Figure 18 Invalid Stacked Column](image)

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>Market</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>Product</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>Profit</td>
<td>Inventory</td>
<td>Ratio</td>
<td>Measures</td>
</tr>
<tr>
<td>4</td>
<td>Year</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Comments in Free-Form Grids**

Data source types: Essbase
Note: You cannot add comments on worksheets that are enabled for multiple grids.

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
- Interleaved with members of row, column and page dimensions
- To the left, right, top, bottom of the grid.

Comment rows and comment columns can be interleaved with row and columns dimensions. Comments cannot be placed in data cells or in cells that intersect row and column dimensions in the upper right corner.

For information about unexpected behavior that may occur, see “Actions That May Cause Unexpected Behavior” on page 315.

Comments in Blank Rows and Columns

Figure 19 shows a grid with comments in cells A5, A6, C1, C2, C10, D1, D2, D10, H5, and H6. These comments are retained in retrieval and zoom operations.

![Figure 19 Comments in Blank Rows and Columns](image)

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>c1</td>
<td>d1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>c2</td>
<td>d2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td>Product</td>
<td>Product</td>
<td>Product</td>
<td>Product</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Profit</td>
<td>Inventory</td>
<td>Ratio</td>
<td>Measures</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>a5</td>
<td>Market</td>
<td>Ctrl1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>a6</td>
<td>Market</td>
<td>Ctrl2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>a7</td>
<td>Market</td>
<td>Ctrl3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>a8</td>
<td>Market</td>
<td>Ctrl4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>a9</td>
<td>Market</td>
<td>Year</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>c10</td>
<td>d10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Grid with Complex Comments

Figure 20 shows an example of a combination of the page region, attributes, and comments on a single grid.
Invalid Placement of Comments

Row and column dimension regions can be interleaved with comment rows and comment columns. Figure 21 shows a grid that 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.)

Formulas in Free-Form Grids

You can enter Excel formulas in cells that can contain comments. Figure 22 shows a grid with Excel formulas in cells C8 and F8. For information about unexpected behavior that may occur, see “Actions That May Cause Unexpected Behavior” on page 315.
Attribute Dimensions in Free-Form Grids

Figure 23 shows an example of both page region and attribute usage. In this example, Pkg Type (an attribute dimension attached to the base member Product) and Budget are page dimensions. 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.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Pkg Type</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Budget</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Product</td>
<td>Product</td>
<td>Product</td>
<td>Product</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Profit</td>
<td>Inventory</td>
<td>Ratios</td>
<td>Measures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Market</td>
<td>Ctrl1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Market</td>
<td>Ctrl2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Market</td>
<td>Ctrl3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Market</td>
<td>Ctrl4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Market</td>
<td>Year</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Creating Free-Form Reports

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 308.
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.

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.

**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. (The worksheet must be blank.)
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 3, 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 3.
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.

### Creating Asymmetric Reports

Essbase reports can contain symmetric or asymmetric column groups. Essbase determines the symmetry of column groups automatically, based on the members you select.

Symmetric reports, as shown in Figure 24, are characterized by repeating, identical groups of members.

**Figure 24  Symmetric Report**

<table>
<thead>
<tr>
<th></th>
<th>East</th>
<th></th>
<th>West</th>
</tr>
</thead>
<tbody>
<tr>
<td>Budget</td>
<td>Qtr1</td>
<td>Qtr2</td>
<td>Qtr3</td>
</tr>
<tr>
<td>Actual</td>
<td>Qtr1</td>
<td>Qtr2</td>
<td>Qtr3</td>
</tr>
<tr>
<td>Budget</td>
<td>Qtr1</td>
<td>Qtr2</td>
<td>Qtr3</td>
</tr>
<tr>
<td>Actual</td>
<td>Qtr1</td>
<td>Qtr2</td>
<td>Qtr3</td>
</tr>
</tbody>
</table>

Asymmetric reports, as shown in Figure 25, are 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 as follows:

- Enter member names in free-form mode.
- Zoom 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.

**Note:** Retrieving data into an asymmetric report may take a long time on large reports.

**Actions That May Cause Unexpected Behavior**

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.
Using Other Applications with Smart View

In This Appendix

Crystal Ball EPM .......................................................... 317
Smart View and Spreadsheet Add-in .................................. 318
Migrating Functions .......................................................... 319

These are applications that you can use with Smart View if you hold the appropriate licenses for them.

Crystal Ball EPM

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 Panel.
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 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

**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**, and then **Global**.
3 Select **Limit to Connected Sheets**.
4 Click **OK**.

Smart View will control 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\EPMSistema11R1\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

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 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.

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 ................................................ 323
Using Oracle User Productivity Kit ...................................................................... 323

Information about Data Sources and Other Products

In general, this guide provides only procedural information for using the data provider features that Smart View supports. For detailed information about the data providers 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 Hyperion Enterprise Performance Management Workspace 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 Hyperion 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 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. Oracle Hyperion Financial Management and Oracle Hyperion Planning 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 or Oracle Hyperion Financial Reporting Studio content.

To open UPK Help, from the Smart View ribbon, click the arrow next to Help, and then select Oracle User Productivity Kit.
Glossary

**ad hoc report** An online analytical query that an end user creates dynamically.

**ancestor** A branch member that has members below it. For example, the members Qtr2 and 2006 are ancestors of the member April.

**application** 1) A software program designed to run a specific task or group of tasks such as a spreadsheet program or database management system; 2) A related set of dimensions and dimension members that are used to meet a specific set of analytical requirements, reporting requirements, or both.

**attribute** A characteristic of a dimension member. For example, Employee dimension members may have attributes of Name, Age, or Address. Product dimension members can have several attributes, such as a size and flavor.

**attribute dimension** A type of dimension that enables analysis based on the attributes or qualities of dimension members.

**attribute reporting** A reporting process based on the attributes of the base dimension members. See also base dimension.

**base dimension** A standard dimension that is associated with one or more attribute dimensions. For example, assuming products have flavors, the Product dimension is the base dimension for the Flavors attribute dimension.

**business rules** Logical expressions or formulas that are created within an application to produce a desired set of resulting values.

**cell** 1) The data value at the intersection of dimensions in a multidimensional database; the intersection of a row and a column in a worksheet; 2) A logical group of nodes belonging to one administrative domain.

**cell note** A text annotation for a cell in an Essbase database. Cell notes are a type of LRO.

**cube** A block of data that contains three or more dimensions. An Essbase database is a cube.

**data form** A grid display that enables users to enter data into the database from an interface such as a Web browser, and to view and analyze data or related text. Certain dimension member values are fixed, giving users a specific view into the data.

**descendant** Any member below a parent in the database outline. In a dimension that includes years, quarters, and months, the members Qtr2 and April are descendants of the member Year.

**dimension** A data category used to organize business data for the 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.

**drill-through** The navigation from a value in one data source to corresponding data in another source.

**duplicate member name** Multiple occurrences of a member name in a database, with each occurrence representing a different member. For example, a database has two members named New York. One member represents New York state and the other member represents New York city.

**Dynamic Time Series** A process that performs period-to-date reporting in block storage databases.

**free-form grid** An object for presenting, entering, and integrating data from different sources for dynamic calculations.

**free-form reporting** Creating reports by entering dimension members or report script commands in worksheets.
**generation** A layer in a hierarchical tree structure that defines member relationships in a database. Generations are ordered incrementally from the top member of the dimension (generation 1) down to the child members. Use the unique generation name to identify a layer in the hierarchical tree structure.

**latest** A spreadsheet keyword used to extract data values from the member defined as the latest time period.

**level** A layer in a hierarchical tree structure that defines database member relationships. Levels are ordered from the bottom dimension member (level 0) up to the parent members.

**level 0 member** A member that has no children.

**linked partition** A shared partition that enables you to use a data cell to link two databases. When a user clicks a linked cell in a worksheet, Essbase opens a new sheet displaying the dimensions in the linked database. The user can then drill down those dimensions.

**MDX (multidimensional expression)** A language used for querying and calculation in multidimensional-compliant databases.

**member** A discrete component within a dimension. A member identifies and differentiates the organization of similar units. For example, a time dimension might include members Jan, Feb, and Qtr1.

**metadata** A set of data that defines and describes the properties and attributes of the data stored in a database or used by an application. Examples of metadata are dimension names, member names, properties, time periods, and security.

**missing data (#MISSING)** A marker indicating that data in the labeled location does not exist, contains no value, or was never entered or loaded. For example, missing data exists when an account contains data for a previous or future period but not for the current period.

**nested column headings** A report column heading format that displays data from multiple dimensions. For example, a column heading that contains Year and Scenario members is a nested column. The nested column heading shows Q1 (from the Year dimension) in the top line of the heading, qualified by Actual and Budget (from the Scenario dimension) in the bottom line of the heading.

**page** A display of information in a grid or table often represented by the Z-axis. A page can contain data from one field, derived data from a calculation, or text.

**page heading** A report heading type that lists members represented on the current page of the report. All data values on the page have the members in the page heading as a common attribute.

**page member** A member that determines the page axis.

**pivot** Alter the perspective of retrieved data. When Essbase first retrieves a dimension, it expands data into rows. You can then pivot or rearrange the data to obtain a different viewpoint.

**POV (point of view)** A feature for setting data focus by selecting members that are not already assigned to row, column, or page axes. For example, selectable POVs in FDM could include location, period, category, and target category. In another example, using POV as a filter in Smart View, you could assign the Currency dimension to the POV and select the Euro member. Selecting this POV in data forms displays data in Euro values.

**preserve formulas** User-created formulas kept within a worksheet while retrieving data.

**qualified name** A member name in a qualified format that differentiates duplicate member names in a duplicate member outline. For example, [Market].[East].[State]. [New York] or [Market].[East].[City].[New York].

**report object** In report designs, a basic element with properties defining behavior or appearance, such as text boxes, grids, images, and charts.

**root member** The highest member in a dimension branch.

**runtime prompt** A variable that users enter or select before a business rule is run.

**scenario** A dimension for classifying data; for example, Actuals, Budget, Forecast1, or Forecast2.

**sibling** A child member at the same generation as another child member and having the same immediate parent. For example, the members Florida and New York are children of East and each other’s siblings.

**Smart Slice** In Smart View, a reusable perspective of a data source that contains a restricted set of dimensions or dimension members.
**smart tags**  Keywords in Microsoft Office applications that are associated with predefined actions available from the Smart Tag menu. In Oracle EPM System products, smart tags can also be used to import Reporting and Analysis content and to access Financial Management and Essbase functions.

**supporting detail**  Calculations and assumptions from which the values of cells are derived.

**suppress rows**  A setting that excludes rows containing missing values and underscores characters from spreadsheet reports.

**task list**  A detailed status list of tasks for a particular user.

**time series reporting**  A process for reporting data based on a calendar date (for example, year, quarter, month, or week).

**user variable**  A variable that dynamically renders data forms based on a user’s member selection, displaying only the specified entity. For example, a user variable named Department displays specific departments and employees.
# Index

### A
ad hoc analysis
- about, 49
- Dynamic Time Series members, 27
- free-form, 307
- selecting members, 21
- starting, 49
adjusting values, 39
alias tables, 30, 31
- selecting for connection, 30
- selecting for current worksheet, 30
aliases, 30
- with member names, 31
annotations, 96
- planning units, 96
Approvals, 93
asymmetric reports, 314
attribute dimensions, in free-form grids, 313

### B
business rules
- Calculate Currencies, 100
- Calculate Form, 100
- launching, 99
- overview, 99
- runtime prompts, 99

### C
Calculate Currencies business rule, 100
Calculate Form business rule, 100
calculating data, 36
calculating rows, inserting, 55
calculation scripts. See business rules
calculations, 103. See also supporting detail and business rules
cascading, 57
cell comments, 44
in Financial Management, 45
in Planning, 44
cell history, 44
cell styles, 134
cell text, 44
comments, 44
comments, adding to planning units, 96
components
- Smart View, 13
composite forms, 98
connecting
- data sources, 17
connections
- locating, 18
- private connections, 17
- private, creating, 19
- private, deleting, 19
- saving, 19
- shared connections, 17
consolidating data, 37
copy version, 97
copying
data points, 67
worksheets, 66
Cross Dimension runtime prompt, 100
Crystal Ball, 317
currencies, 38
- changing, 38
- translating, 38
currency codes
- location of, 38

data
- adjusting, 39
- calculating, 36
consolidating, 37
refreshing, 35
submitting, 35
data filtering, 87
data perspective, 39
data source connections, 17
data sources
  connecting, 17
  disconnecting, 18
dimensions, 21
  about, 21
disabling Smart View, 69
document attachment, 46
drill-through reports, 41
dynamic data points, 67

E
enabling Smart View, 69
extensions, 134

F
filtering data, 87
Financial Reporting
  editing, 119
  formats, 113
  importing documents, 117
  importing into Excel, 118
  importing into Word and PowerPoint, 119
  templates, 120
formatting
  ad hoc grids, 50
  Excel, 51
  Smart View, 50
forms
  about, 61
  ad hoc analysis, 62
  composite, 98
  Financial Management, 63
  linked forms, 64
  opening, 61
  Planning, 62
  Planning attributes, 62
  refreshing offline, 109
  working with offline, 107
forms, data
  refreshing offline, 109
using linked forms, 64
working with offline, 107
formulas
  and zooming, 54
  Excel
    ad hoc grids, 50
    forms, 62
  free-form, 307
  asymmetric reports, 314
  attribute dimensions, working with, 313
guidelines, 308
free-form mode
  entering members, 26
Function Builder, 137
functions
  about, 137
  creating, 137
  creating manually, 139
  error codes, 145
  running, 140

G
grids, multiple, 55
  creating, 56
guidelines for free-form reporting, 308

H
hierarchy of supporting detail, 104
history
  cell, 47
  HsCurrency function
    syntax, 142
  HsDescription function
    syntax, 143
  HsGetSheetInfo function, 141
  HsGetText function
    syntax, 143
  HsGetValue function
    syntax, 141
  HsGetVariable function, 144
  HsLabel function, 143
  HsSetText function, 144
  HsSetValue function, 141

I
Import Metadata, 66
information, finding, 323
Interactive Reporting
  editing, 116
  importing documents, 114
  importing into Excel, 115
  importing into Word and PowerPoint, 116

J
  job console, 97

K
  Keep Only, 55

L
  line item detail, 103
  linked forms, 64
  linked partitions, 43
  linked reporting objects, 41
    attaching, 42
    launching, 43

M
  MDX queries, 88
  member information, 31
  member names
    duplicate, displaying, 28
    member perspective, 29
  Member Range runtime prompt, 100
  member selection
    for runtime prompts, 99
  members, 21
    about, 21
    adding in Financial Management forms, 64
    filtering by attribute, 26
    filtering by subsets, 27
    information, 31
    period-to-date, 27
    selecting, 21
    selecting from POV toolbar, 23
    selecting in Financial Management forms, 63
  merged cells
    Excel formatting, 51
  migrating
    connections, 320
    considerations, 319
    converting a workbook, 319
    converting multiple workbooks, 320
    migration utility, 319
    Move Down option, 105
    Move Up option, 105

N
  non-calculating rows, inserting, 55

O
  offline. See Offline Planning
  offline forms
    refreshing, 109
    working with, 107
  Offline Planning
    refreshing data in, 109
    working in, 107
  offline, Planning, 106
  options
    advanced, 131
    cell styles, 134
    data, 129
    formatting, 133
    member, 128
  Oracle User Productivity kit (UPK)
    using, 323

P
  Page dimension
    searching for members in, 97
  pivoting, 55
  Planning
    Approvals, 93
    copying versions, 97
    forms, 62
    offline, 106
    preferences, 106
  planning units
    annotations, 96
    changing the status of, 93
    finding, 94
    out of office assistant, 96
    promotional path, 95
  POV
    copying, 33
deleting, 33
printing in header and footer, 69
POV Manager, 32
selecting members, 32
POV toolbar, 23
hiding, 24
selecting Essbase members, 25
selecting members, 23
preferences
Planning, 106
private connections, 17
saving shared connections as, 19
Production Reporting
editing, 123, 125
importing documents, 121, 124
importing into Excel, 121
importing into Word and PowerPoint, 122
Queries
creating, 85
editing, 86
Query Designer, 85
time-related data, 87
Redo, 65
options, 66
Refresh, 35
refreshing data
while working offline, 109
refreshing Reporting and Analysis documents, 112
Remove Only, 55
Reporting and Analysis
editing and refreshing documents, 112
importing documents, 111
Reporting and Analysis documents
refreshing, 112
updating, 112
Reports
cascading, 57
free-form, 307
retrieving data, 35
ribbons, 13
Rules on Form dialog box, 100
runtime prompts
types, 99
search wild cards, 69
searching
for members in the Page dimension, 97
shared connections, 17
shared workbooks, 69
Sheet Info, 68
sheet information, 68
simulation workbooks, 317
toolbar, 318
sliders, 81
creating from joined queries, 82
creating from query, 81
Smart Query, 71
building, 73
copying and pasting, 76
creating, 71, 73
defining set filters, 73
defining sets, 72
opening, 76
Smart Slices
about, 79
boundaries, 83
creating, 83
deleting reports, 81
preferences, 84
sliders, 81
Smart View
disabling, 69
enabling, 69
Smart View Panel
about, 14
spreading data
over time periods, 101
Spreadsheet Add-in, 318
submitting a planning unit, 94
submitting data, 35
substitution variables, 58
supporting detail
adding, 103
changing, 105
hierarchy of, 104
overview, 103
viewing, 105
Supporting Detail window
  using, 103
  synchronizing data, 108

T
  task lists, 89
    about, 89
    completing tasks, 91
    creating reports, 91
    executing, 90
    integrating with Outlook, 91
    opening, 89
    viewing, 90
  time periods, spreading data over, 101
  time-related data
    Query Designer, 87

U
  undo, 65
    options, 66
  unexpected behavior in free-form, 315
  URL
    and data cells, 46
    launching from cell, 46
    linked to planning cells, 46
  Use Members on Forms option
    with runtime prompts, 99
  User Productivity Kit, 323

V
  VBA functions
    HypAddLRO, 203
    HypCalculate, 229
    HypCalculateContribution, 230
    HypCell, 196
    HypConnect, 166
    HypConnected, 166
    HypConnectionExists, 167
    HypConsolidateAll, 231
    HypConsolidateAllWithData, 232
    HypCopyMetaData, 154
    HypCreateConnection, 168
    HypCreateConnectionEX, 170
    HypDeleteAllLROs, 202
    HypDeleteAllMRUItems, 264
    HypDeleteCalc, 228
    HypDeleteLROs, 202
    HypDeleteMetadata, 155
    HypDisconnect, 172
    HypDisconnectAll, 172
    HypDisconnectEx, 173
    HypDisplayToLinkView, 294
    HypExecuteCalcScript, 222
    HypExecuteCalcScriptEx, 223
    HypExecuteDrillThroughReport, 207
    HypExecuteQuery, 184
    HypFindMember, 236
    HypFindMemberEx, 237
    HypForceCalculate, 232
    HypForceCalculateContribution, 233
    HypForceTranslate, 234
    HypFreeDataPoint, 197
    HypGetActiveMember, 216
    HypGetAncestor, 238
    HypGetBackgroundPOV, 210
    HypGetCellRangeForMbrCombination, 198
    HypGetChildren, 239
    HypGetColCount, 299
    HypGetColItems, 302
    HypGetConnectionInfo, 295
    HypGetDataSource, 199
    HypGetDimensions, 217
    HypGetDimMbrsForDataCell, 194
    HypGetDrillThroughReports, 208
    HypGetGlobalOption, 255
    HypGetLinkMacro, 292
    HypGetMemberInformation, 251
    HypGetMemberInformationEX, 253
    HypGetMembers, 214
    HypGetNameRangeList, 183
    HypGetOption, 260
    HypGetPagePOVChoices, 212
    HypGetParent, 240
    HypGetPOVCount, 300
    HypGetPOVItems, 304
    HypGetRowCount, 298
    HypGetRowItems, 300
    HypGetSharedConnectionsURL, 173
    HypGetSheetOption, 258
    HypGetSourceGrid, 293
    HypGetSubstitutionVariable, 162
    HypIsolateSSO, 176
    HypIsAncestor, 243
HypIsAttribute, 241
HypIsCellWritable, 200
HypIsChild, 245
HypIsConnectedToSharedConnections, 175
HypIsDataModified, 156
HypIsDescendant, 242
HypIsExpense, 244
HypIsFreeForm, 158
HypIsParent, 245
HypIsSmartViewContentPresent, 157
HypIsUDA, 246
HypKeepOnly, 189
HypListCalcScripts, 220
HypListCalcScriptsEx, 221
HypListLROs, 205
HypMenuVAboutt, 268
HypMenuVAdjust, 269
HypMenuVBusinessRules, 269
HypMenuVCalculation, 270
HypMenuVCascadeNewWorkbook, 270
HypMenuVCascadeSameWorkbook, 271
HypMenuVCellText, 271
HypMenuVCollapse, 272
HypMenuVConnect, 272
HypMenuVCopyDataPoints, 273
HypMenuVExpand, 273
HypMenuVFunctionBuilder, 274
HypMenuVInstruction, 274
HypMenuVKeepOnly, 275
HypMenuVLRO, 275
HypMenuVMemberInformation, 276
HypMenuVMemberSelection, 276
HypMenuVMigrate, 277
HypMenuVOptions, 278
HypMenuVPasteDataPoints, 278
HypMenuVPivot, 279
HypMenuVPPOVManager, 279
HypMenuVQueryDesigner, 280
HypMenuVRedo, 280
HypMenuVRefresh, 281
HypMenuVRefreshAll, 281
HypMenuVRefreshOfflineDefinition, 282
HypMenuVRemoveOnly, 282
HypMenuVRulesOnForm, 283
HypMenuVRunReport, 284
HypMenuVSelectForm, 284
HypMenuVShowHelpHtml, 285
HypMenuVSubmitData, 285
HypMenuVSupportingDetails, 286
HypMenuVSyncBack, 286
HypMenuVTakeOffline, 287
HypMenuVUndo, 287
HypMenuVVisualizeinExcel, 288
HypMenuVZoomIn, 288
HypMenuVZoomOut, 289
HypOpenForm, 193
HypOtlGetMemberInfo, 247
HypPivot, 186
HypPivotToGrid, 187
HypPivotToPOV, 188
HypPreserveFormatting, 160
HypQueryMembers, 248
HypRedo, 159
HypRemoveConnection, 175
HypRemoveOnly, 190
HypRemovePreservedFormats, 161
HypResetFriendlyName, 176
HypRetrieve, 180
HypRetrieveAllWorkbooks, 184
HypRetrieveLRO, 206
HypRetrieveNameRange, 182
HypRetrieveRange, 181
HypSetActiveConnection POV, 177
HypSetActiveMember, 217
HypSetAliasTable, 162
HypSetAsDefault, 178
HypSetBackgroundPOV, 211
HypSetCellsDirty, 201
HypSetCollItems, 303
HypSetConnAliasTable, 179
HypSetConnectionInfo, 297
HypSetDimensions, 218
HypSetGlobalOption, 257
HypSetLinkMacro, 292
HypSetMembers, 215
HypSetMenu, 153
HypSetOption, 263
HypSetPages, 213
HypSetPOV, 210
HypSetPOVItems, 305
HypSetRowItems, 301
HypSetSharedConnectionsURL, 174
HypSetSheetOption, 259
HypSetSubstitutionVariable, 164
HypShowPOV, 152
HypSubmitData, 185
HypTranslate, 235
HypUndo, 158
HypUpdateLRO, 204
HypUseLinkMacro, 291
HypZoomIn, 191
HypZoomOut, 192

VBA functions, working with
  declaring functions, 148
  menu functions, 267
  migrating legacy VBA applications, 306
  parameters, 148
  return values, 149
  using functions in Smart View, 147

viewing
  supporting detail, 105

W
Web Analysis
  formats, 113
    importing documents, 123
Within Selected Group option, 315

workbooks
  shared, 69

Z
zoom level
  setting default, 53
zooming
  and formulas, 54
    display settings, 53
    double-click, 54
  zooming in, 52
  Zooming in and out, 51
  zooming out, 53