
MDX Cube Reporting Guide

Version 10

To ensure that you are using the documentation that corresponds to the software you are licensed to use, compare this version number with the software version shown in "About MicroStrategy..." in the Help menu of your software.

Document number: 09481000

Copyright © 2015 by MicroStrategy Incorporated. All rights reserved.

If you have not executed a written or electronic agreement with MicroStrategy or any authorized MicroStrategy distributor (any such agreement, a "Separate Agreement"), the following terms apply:

This software and documentation are the proprietary and confidential information of MicroStrategy Incorporated and may not be provided to any other person. Copyright © 2001-2015 by MicroStrategy Incorporated. All rights reserved.

THIS SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" AND WITHOUT EXPRESS OR LIMITED WARRANTY OF ANY KIND BY EITHER MICROSTRATEGY INCORPORATED OR ANYONE WHO HAS BEEN INVOLVED IN THE CREATION, PRODUCTION, OR DISTRIBUTION OF THE SOFTWARE OR DOCUMENTATION, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, GOOD TITLE AND NON-INFRINGEMENT, QUALITY OR ACCURACY. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE SOFTWARE AND DOCUMENTATION IS WITH YOU. SHOULD THE SOFTWARE OR DOCUMENTATION PROVE DEFECTIVE, YOU (AND NOT MICROSTRATEGY, INC. OR ANYONE ELSE WHO HAS BEEN INVOLVED WITH THE CREATION, PRODUCTION, OR DISTRIBUTION OF THE SOFTWARE OR DOCUMENTATION) ASSUME THE ENTIRE COST OF ALL NECESSARY SERVICING, REPAIR, OR CORRECTION. SOME STATES DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO THE ABOVE EXCLUSION MAY NOT APPLY TO YOU.

In no event will MicroStrategy, Incorporated, or any other person involved with the creation, production, or distribution of the Software be liable to you on account of any claim for damage, including any lost profits, lost savings, or other special, incidental, consequential, or exemplary damages, including but not limited to any damages assessed against or paid by you to any third party, arising from the use, inability to use, quality, or performance of such Software and Documentation, even if MicroStrategy, Inc. or any such other person or entity has been advised of the possibility of such damages, or for the claim by any other party. In addition, MicroStrategy, Inc. or any other person involved in the creation, production, or distribution of the Software shall not be liable for any claim by you or any other party for damages arising from the use, inability to use, quality, or performance of such Software and Documentation, based upon principles of contract warranty, negligence, strict liability for the negligence of indemnity or contribution, the failure of any remedy to achieve its essential purpose, or otherwise. The entire liability of MicroStrategy, Inc. and your exclusive remedy, shall not exceed, at the option of MicroStrategy, Inc., either a full refund of the price paid, or replacement of the Software. No oral or written information given out expands the liability of MicroStrategy, Inc. beyond that specified in the above limitation of liability. Some states do not allow the limitation or exclusion of liability for incidental or consequential damages, so the above limitation may not apply to you.

The information contained in this manual (the Documentation) and the Software are copyrighted and all rights are reserved by MicroStrategy, Inc. MicroStrategy, Inc. reserves the right to make periodic modifications to the Software or the Documentation without obligation to notify any person or entity of such revision. Copying, duplicating, selling, or otherwise distributing any part of the Software or Documentation without prior written consent of an authorized representative of MicroStrategy, Inc. are prohibited. U.S. Government Restricted Rights. It is acknowledged that the Software and Documentation were developed at private expense, that no part is public domain, and that the Software and Documentation are Commercial Computer Software provided with RESTRICTED RIGHTS under Federal Acquisition Regulations and agency supplements to them. Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFAR 252.227-7013 et. seq. or subparagraphs (c)(1) and (2) of the Commercial Computer Software-Restricted Rights at FAR 52.227-19, as applicable. Contractor is MicroStrategy, Incorporated., 1850 Towers Crescent Plaza, Tysons Corner, VA 22182. Rights are reserved under copyright laws of the United States with respect to unpublished portions of the Software.

The following terms and notices apply regardless of whether you have executed a Separate Agreement:

Trademark Information

MicroStrategy, MicroStrategy 9, MicroStrategy 9s, MicroStrategy Analytics Platform, MicroStrategy Desktop, MicroStrategy Analytics Express, MicroStrategy Analytics Enterprise, MicroStrategy Evaluation Edition, MicroStrategy Suite, MicroStrategy Web, MicroStrategy Mobile, MicroStrategy Server, MicroStrategy Parallel Relational In-Memory Engine (MicroStrategy PRIME), MicroStrategy MultiSource, MicroStrategy OLAP Services, MicroStrategy Intelligence Server, MicroStrategy Intelligence Server Universal, MicroStrategy Distribution Services, MicroStrategy Report Services, MicroStrategy Transaction Services, MicroStrategy Visual Insight, MicroStrategy Web Reporter, MicroStrategy Web Analyst, MicroStrategy Web Universal, MicroStrategy Office, MicroStrategy Data Mining Services, MicroStrategy Narrowcast Server, MicroStrategy Health Center, MicroStrategy Power User, MicroStrategy Analyst, MicroStrategy Developer, MicroStrategy Web Professional, MicroStrategy Architect, MicroStrategy SDK, MicroStrategy Command Manager, MicroStrategy Enterprise Manager, MicroStrategy Object Manager, MicroStrategy Integrity Manager, MicroStrategy System Manager, MicroStrategy Analytics App, MicroStrategy Mobile App, MicroStrategy Analytics for iPad® App, MicroStrategy Analytics Express App, MicroStrategy Tech Support App, MicroStrategy Mobile App Platform, MicroStrategy Mobile App Developer Academy, MicroStrategy Cloud, MicroStrategy Cloud Platform Services, MicroStrategy Cloud Data Hosting Services, MicroStrategy Cloud Data Warehouse Services, MicroStrategy Cloud Data Integration Services, MicroStrategy Virtual Business Intelligence (VBI) Appliance, MicroStrategy Cloud Paid Pilot, MicroStrategy R Integration, MicroStrategy Usher, Usher Badge, Usher Security, Usher Security Server, Usher Mobile, Usher Analytics, Usher Network Manager, MicroStrategy Trela for Retail (Alert backwards / used for demonstrations, MicroStrategy Services, MicroStrategy Professional Services, MicroStrategy Consulting, MicroStrategy Customer Services, MicroStrategy Education, MicroStrategy University, MicroStrategy Managed Services, MicroStrategy Business Intelligence QuickStrike, BI QuickStrike, Mobile QuickStrike, Transaction Services QuickStrike, Retail Vendor Portal, Perennial Education Pass, MicroStrategy Web Based Training (WBT), MicroStrategy World, Office Intelligence, Best in Business Intelligence, Pixel Perfect, Global Delivery Center, MicroStrategy Identity Platform, MicroStrategy Loyalty Platform, Direct Connect, Enterprise Grade Security For Every Business, Build Your Own Business Apps, Code-Free, Welcome to Ideal, The World's Most Comprehensive Analytics Platform, The World's Most Comprehensive Analytics Platform. Period.

Other product and company names mentioned herein may be the trademarks of their respective owners.

Specifications subject to change without notice. MicroStrategy is not responsible for errors or omissions. MicroStrategy makes no warranties or commitments concerning the availability of future products or versions that may be planned or under development.

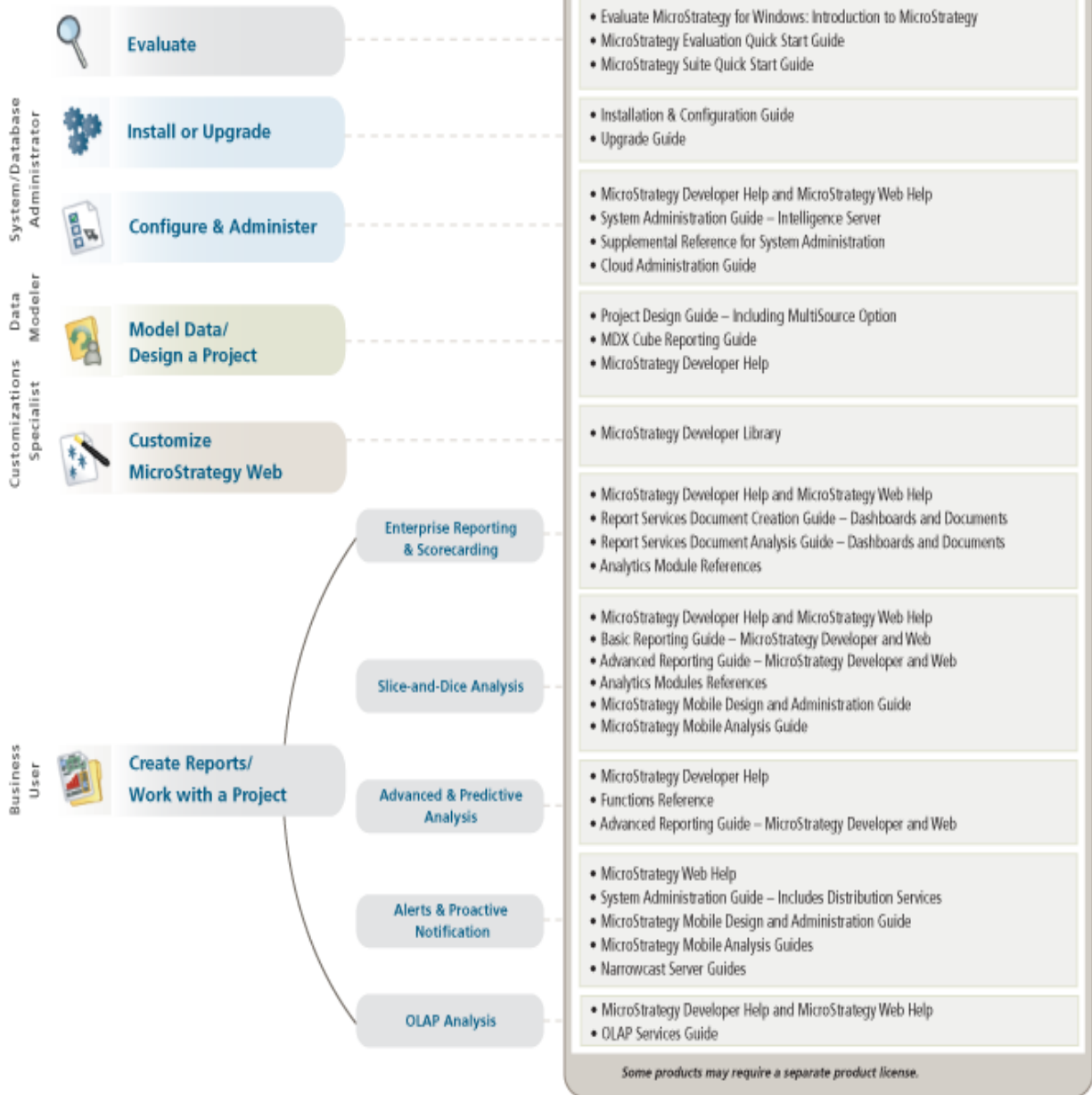
Patent Information

This product is patented. One or more of the following patents may apply to the product sold herein: U.S. Patent Nos. 5,321,520, 5,416,602, 5,748,560, 6,154,766, 6,173,310, 6,260,050, 6,263,051, 6,269,393, 6,279,033, 6,501,832, 6,567,796, 6,587,547, 6,606,596, 6,658,093, 6,658,432, 6,662,195, 6,671,715, 6,691,100, 6,694,316, 6,697,808, 6,704,723, 6,707,889, 6,741,980, 6,765,997, 6,768,788, 6,772,137, 6,788,768, 6,792,086, 6,798,867, 6,801,910, 6,820,073, 6,829,334, 6,836,537, 6,850,603, 6,859,798, 6,873,693, 6,885,734, 6,888,929, 6,895,084, 6,940,953, 6,964,012, 6,977,992, 6,996,568, 6,996,569, 7,003,512, 7,010,518, 7,016,480, 7,020,251, 7,039,165, 7,082,422, 7,113,474, 7,113,993, 7,127,403, 7,174,349, 7,181,417, 7,194,457, 7,197,461, 7,228,303, 7,260,577, 7,266,181, 7,272,212, 7,302,639, 7,324,942, 7,330,847, 7,340,040, 7,356,758, 7,356,840, 7,415,438, 7,428,302, 7,430,562, 7,440,898, 7,457,397, 7,486,780, 7,509,671, 7,516,181, 7,559,048, 7,574,376, 7,617,201, 7,725,811, 7,801,967, 7,836,178, 7,861,161, 7,861,253, 7,881,443, 7,925,616, 7,945,584, 7,970,782, 8,005,870, 8,035,382, 8,051,168, 8,051,369, 8,094,788, 8,130,918, 8,296,287, 8,321,411, 8,452,755, 8,521,733, 8,522,192, 8,577,902, 8,606,813, 8,607,138, 8,645,313, 8,761,659, 8,775,807, 8,782,083, 8,812,490, 8,832,588, 8,943,044, and 8,943,187. Other patent applications are pending.

Third Party Software

Various MicroStrategy products contain the copyrighted technology or software of third parties ("Third Party Software"). A list of Third Party Software, as well as links to any terms and conditions associated with such Third Party Software ("Third Party Terms"), can be found at www.microstrategy.com/third-party-notice. Your use of MicroStrategy products is subject to all applicable Third Party Terms.

WHAT DO YOU WANT TO DO WITH MICROSTRATEGY?



CONTENTS

Book Overview and Additional Resources	Description of this guide..... ix
	About this bookx
	How to find business scenarios and examplesx
	What's new in this guidex
	Prerequisitesxi
	Who should use this guide.....xii
	Resources.....xii
	Documentation.....xii
	Educationxx
	Consultingxxi
	Technical Support.....xxi
	Feedbackxxi
 1. About MDX Cube Sources in MicroStrategy	 Introduction..... 1
	Understanding MicroStrategy architecture 3
	The MicroStrategy object model 5
	Relating objects from MDX cube sources to MicroStrategy 6
	Understanding SAP BW terminology 7
	Relating objects from SAP BW to MicroStrategy 9
	Relating objects from Oracle Essbase to MicroStrategy..... 18
	Relating objects from Analysis Services to MicroStrategy..... 24
	Relating objects from TM1 to MicroStrategy 29

2. Connecting to MDX Cube Sources

Introduction	35
Connecting to SAP BW servers	36
Connecting to SAP BW servers on Windows	38
Connecting to SAP BW servers on UNIX and Linux	42
Connecting to Oracle Essbase servers	47
Configuring the MDX Cube Provider	48
Creating a database instance	49
Connecting to Analysis Services servers	51
Configuring Analysis Services and the MDX Cube provider	52
Creating a database instance	54
Connecting to TM1 servers	56
Creating a database instance	57
Configuring MDX cube database instances in projects	59
Removing an MDX cube database instance from a project	61
MDX cube schema loading	61
Exchanging the database instance for an MDX cube schema	63
Supporting large result sets for MDX cube reports	64
Inheriting MDX cube source formats for metric values	65
Authentication	67
Single sign-on to Microsoft Analysis Services	68
Single sign-on to SAP BW	69
Authenticating SAP BW users in MicroStrategy projects	70

3. Integrating MDX Cubes into MicroStrategy

Introduction	79
Importing MDX cubes	80
Importing MDX cubes before report creation	81
Importing levels and suffixes for characteristics	85
Importing additional measure structure	88
Importing MDX cubes during report creation	91
Updating MDX cube structure	92
Maintaining MDX cubes between multiple projects	94
About managed objects	95
Mapping MDX cubes	96
Shared MDX cube objects	101
Best practices for mapping MDX cubes	103
Mapping MDX cube data to project attributes	103
Defining column data types for MDX cube data	109
Preserving attribute element orders from MDX cube sources	113
Supporting MDX cube source date data in MicroStrategy	117

Defining unbalanced and ragged hierarchies	120
Displaying hierarchies on MDX cube reports	122
Mapping SAP BW variables to MicroStrategy prompts	124
Creating metrics from MDX cube data	129
How to build analysis into metrics with custom MDX	132
Using prompts within MDX cube metrics	138
Deleting compound metrics from MDX cubes	140
Creating data marts of MDX cube data	141
 4. Reporting on MDX Cubes	
Creating MDX Cube Reports and Analyzing Data	
Introduction	143
Creating MDX cube reports	144
Troubleshooting MDX cube report execution	147
Creating Intelligent Cubes based on MDX cubes	148
Including MDX cube data in standard reports	150
Switching the MDX cube for an MDX cube report	153
Analyzing data with MDX cube reports	154
Hierarchies on MDX cube reports	155
Filters on MDX cube reports	160
Prompts on MDX cube reports	169
Drilling on MDX cube reports	173
Sorting structure elements and preserving order	177
Sorting on attribute element orders from MDX cube sources	179
Inheriting MDX cube source formats for metric values	181
Using MDX cube reports to filter other reports	183
 Glossary	185
 Index	193

BOOK OVERVIEW AND ADDITIONAL RESOURCES

Description of this guide

The *MicroStrategy MDX Cube Reporting Guide* provides comprehensive information on integrating MicroStrategy with multidimensional expression (MDX) cube sources.

You can integrate data from MDX cube sources such as SAP® BW, Microsoft® Analysis Services, Oracle® Essbase®, or IBM Cognos® TM1® into your MicroStrategy projects and applications. This integration enables you to use the rich set of MicroStrategy reporting and analysis functionality on the data in your MDX cube source. The complete process of integrating your MDX cube sources into MicroStrategy for reporting and analysis is covered in the following chapters:

- [*Chapter 1, About MDX Cube Sources in MicroStrategy*](#), describes how data from an MDX cube source is converted into a MicroStrategy BI solution. This chapter builds on basic knowledge of MicroStrategy and MDX cube source terminology and architecture, and describes how MDX cube sources are represented in MicroStrategy.
- [*Chapter 2, Connecting to MDX Cube Sources*](#), provides information about connecting to an MDX cube source for use within MicroStrategy.

- [Chapter 3, Integrating MDX Cubes into MicroStrategy](#), describes how you can integrate data from an MDX cube source into standard MicroStrategy objects for reporting and analysis.
- [Chapter 4, Reporting on MDX Cubes](#), provides information and guidelines on how to create MDX cube reports and use MicroStrategy features to analyze data from your MDX cube source.

About this book

The sections below provide the location of examples, list prerequisites for using this book, and describe the user roles the information in this book was designed for.



The sample documents and images in this guide, as well as some example steps, were created with dates that may no longer be available in the MicroStrategy Tutorial project. If you are re-creating an example, replace the year(s) shown in this guide with the most recent year(s) available in the software.

How to find business scenarios and examples

Within this guide, many of the concepts discussed are accompanied by business scenarios or other descriptive examples. For examples of reporting functionality, see the MicroStrategy Tutorial, which is MicroStrategy's sample warehouse and project. Information about the MicroStrategy Tutorial can be found in the *MicroStrategy Basic Reporting Guide*.

Detailed examples of advanced reporting functionality can be found in the *MicroStrategy Advanced Reporting Guide*.

What's new in this guide

MicroStrategy 10

- MicroStrategy supports single sign-on authentication of a Windows user to MicroStrategy Developer, MicroStrategy Web, and SAP BW serving as

an MDX cube source in MicroStrategy, as described in [Single sign-on to SAP BW, page 69](#).

- Updates have been made to the steps to connect to MDX cube sources to reflect the current support of MDX cube sources in MicroStrategy. For steps to connect to your MDX cube sources, see [Chapter 2, Connecting to MDX Cube Sources](#).

MicroStrategy Analytics Enterprise

The name of MicroStrategy Desktop has been changed to MicroStrategy Developer.

MicroStrategy 9.4

Any updates to this guide were minor and not directly related to MicroStrategy 9.4. For a list of new features in MicroStrategy 9.4, see the MicroStrategy Readme for that release.

Prerequisites

[Chapter 1, About MDX Cube Sources in MicroStrategy](#) assumes a basic understanding of the terminology and architecture for MicroStrategy and your MDX cube source.

[Chapter 2, Connecting to MDX Cube Sources](#) assumes that you are familiar with creating database instances in MicroStrategy to establish connections to data sources.

[Chapter 3, Integrating MDX Cubes into MicroStrategy](#) assumes that you are familiar with designing a MicroStrategy project and the structure and design of your data within the MDX cube source. For information on designing a MicroStrategy project, see the *MicroStrategy Project Design Guide*.

[Chapter 4, Reporting on MDX Cubes](#) assumes that you are familiar with the basics of report creation and analysis in MicroStrategy. For information on basic report creation and analysis, see the *MicroStrategy Basic Reporting Guide*.

Who should use this guide

The following business intelligence application users should read this guide:

- Project designers who integrate MDX cube sources into MicroStrategy.
- Report designers who create MDX cube reports.

Resources

This section provides details on how to access books, online help, MicroStrategy Education and Consulting resources, and how to contact MicroStrategy Technical Support.

Documentation

MicroStrategy provides both manuals and online help; these two information sources provide different types of information, as described below:

- **Manuals:** MicroStrategy manuals provide:
 - Introductory information and concepts
 - Examples and images
 - Checklists and high-level procedures to get started

The steps to access the manuals are described in [Accessing manuals and other documentation sources, page xviii](#).

Most of these manuals are also available printed in a bound, soft cover format. To purchase printed manuals, contact your MicroStrategy Account Executive with a purchase order number.

- **Help:** MicroStrategy online help provides:
 - Detailed steps to perform procedures
 - Descriptions of each option on every software screen

Additional formats

MicroStrategy manuals are available as electronic publications, downloadable on the Apple iBooks Store or Google Play, and can be read on your iOS or Android device respectively. To download a book, search for the book's title in the iBookstore or Google Play. To view a list of manuals that are currently available, scan the following QR codes using your device's camera:

- For iOS devices, scan the following QR code:



- For Android devices, scan the following QR code:



For new MicroStrategy releases, it may take several days for the latest manuals to be available on the iBookstore or Google Play.

Translations

For the most up-to-date translations of MicroStrategy documentation, refer to the MicroStrategy Knowledge Base. Due to translation time, manuals in languages other than English may contain information that is one or more releases behind. You can see the version number on the title page of each manual.

Finding information

You can search all MicroStrategy books and Help for a word or phrase, with a simple Google™ search at <http://www.google.com>. For example, type “MicroStrategy derived metric” or “MicroStrategy logical table” into a Google search. As described above, books typically describe general concepts and examples; Help typically provides detailed steps and screen options. To limit your search to MicroStrategy books, on Google’s main page you can click **More**, then select **Books**.

Manuals for MicroStrategy overview and evaluation

- *Introduction to MicroStrategy: Evaluation Guide*

Instructions for installing, configuring, and using the MicroStrategy Evaluation Edition of the software. This guide includes a walkthrough of MicroStrategy features so you can perform reporting with the MicroStrategy Tutorial project and its sample business data.

- *MicroStrategy Evaluation Edition Quick Start Guide*

Overview of the installation and evaluation process, and additional resources.

Resources for security

- *Usher Help*

Steps to perform mobile identity validation using the Usher mobile security network to issue electronic badges for identifying users.

Manuals for query, reporting, and analysis

- *MicroStrategy Installation and Configuration Guide*

Information to install and configure MicroStrategy products on Windows, UNIX, Linux, and HP platforms, and basic maintenance guidelines.

- *MicroStrategy Upgrade Guide*

Steps to upgrade existing MicroStrategy products.

- *MicroStrategy Project Design Guide*

Information to create and modify MicroStrategy projects, and create the objects that present your organization's data, such as facts, attributes, hierarchies, transformations, advanced schemas, and project optimization.

- *MicroStrategy Basic Reporting Guide*

Steps to get started with MicroStrategy Web, and how to analyze and format data in a report. Includes the basics for creating reports, metrics, filters, and prompts.

- *MicroStrategy Advanced Reporting Guide: Enhancing Your Business Intelligence Application*

Steps to create Freeform SQL reports, Query Builder reports, complex filters and metrics, use Data Mining Services, and create custom groups, consolidations, and complex prompts.

- *Document and Dashboard Analysis Guide*

Steps to execute, analyze, and format a dashboard in MicroStrategy Web.

- *MicroStrategy Report Services Document Creation Guide: Creating Boardroom Quality Documents*

Steps to create Report Services documents, add objects, and format the document and its objects.

- *MicroStrategy Dashboards and Widgets Creation Guide: Creating Interactive Dashboards for Your Data*

Steps to create MicroStrategy Report Services dashboards and add interactive visualizations.

- *MicroStrategy In-memory Analytics Guide*

Information to use MicroStrategy OLAP Services features, including Intelligent Cubes, derived metrics, derived elements, dynamic aggregation, view filters, and dynamic sourcing.

- *MicroStrategy Office User Guide*

Instructions to use MicroStrategy Office to work with MicroStrategy reports and documents in Microsoft® Excel, PowerPoint, and Word, to analyze, format, and distribute business data.

- *MicroStrategy Mobile Analysis Guide: Analyzing Data with MicroStrategy Mobile*

Steps to use MicroStrategy Mobile to view and analyze data, and perform other business tasks with MicroStrategy reports and documents on a mobile device.

- *MicroStrategy Mobile Design and Administration Guide: A Platform for Mobile Intelligence*

Information and instructions to install and configure MicroStrategy Mobile, as well as steps for a designer working in MicroStrategy Developer or MicroStrategy Web to create effective reports and documents for use with MicroStrategy Mobile.

- *MicroStrategy System Administration Guide: Tuning, Monitoring, and Troubleshooting Your MicroStrategy Business Intelligence System*

Steps to implement, deploy, maintain, tune, and troubleshoot a MicroStrategy business intelligence system.

- *MicroStrategy Supplemental Reference for System Administration: VLDB Properties, Internationalization, User Privileges, and other Supplemental Information for Administrators*

Steps for administrative tasks such as configuring VLDB properties and defining data and metadata internationalization, and reference material for other administrative tasks.

- *MicroStrategy Functions Reference*

Function syntax and formula components; instructions to use functions in metrics, filters, attribute forms; examples of functions in business scenarios.

- *MicroStrategy MDX Cube Reporting Guide*

Information to integrate MicroStrategy with MDX cube sources. You can integrate data from MDX cube sources into your MicroStrategy projects and applications.

- *MicroStrategy Operations Manager Guide*

Instructions for managing, monitoring, and setting alerts for all of your MicroStrategy systems from one console. This guide also includes instructions for setting up and using Enterprise Manager to analyze your MicroStrategy system usage.

Manual for the Human Resources Analytics Module

- *Human Resources Analytics Module Reference*

Software Development Kits

- *MicroStrategy Developer Library (MSDL)*

Information to understand the MicroStrategy SDK, including details about architecture, object models, customization scenarios, code samples, and so on.

- *MicroStrategy Web SDK*



The Web SDK is available in the MicroStrategy Developer Library, which is part of the MicroStrategy SDK.

Documentation for MicroStrategy Portlets

- *Enterprise Portal Integration Help*

Information to help you implement and deploy MicroStrategy BI within your enterprise portal, including instructions for installing and configuring out-of-the-box MicroStrategy Portlets for several major enterprise portal servers.

This resource is available from <http://www.microstrategy.com/producthelp>.

Documentation for MicroStrategy GIS Connectors

- *GIS Integration Help*

Information to help you integrate MicroStrategy with Geospatial Information Systems (GIS), including specific examples for integrating with various third-party mapping services.

This resource is available from <http://www.microstrategy.com/producthelp>.

Help

Each MicroStrategy product includes an integrated help system to complement the various interfaces of the product as well as the tasks that can be accomplished using the product.

Some of the MicroStrategy help systems require a web browser to be viewed. For supported web browsers, see the MicroStrategy Readme.

MicroStrategy provides several ways to access help:

- **Help button:** Use the Help button or ? (question mark) icon on most software windows to see help for that window.
- **Help menu:** From the Help menu or link at the top of any screen, select MicroStrategy Help to see the table of contents, the Search field, and the index for the help system.
- **F1 key:** Press F1 to see context-sensitive help that describes each option in the software window you are currently viewing.



For MicroStrategy Web, MicroStrategy Web Administrator, and MicroStrategy Mobile Server, pressing the F1 key opens the context-sensitive help for the web browser you are using to access these MicroStrategy interfaces. Use the Help menu or ? (question mark) icon to access help for these MicroStrategy interfaces.

Accessing manuals and other documentation sources

The manuals are available from <http://www.microstrategy.com/producthelp>, as well as from your MicroStrategy disk or the machine where MicroStrategy was installed.



Adobe Reader is required to view these manuals. If you do not have Adobe Reader installed on your computer, you can download it from <http://get.adobe.com/reader/>.

The best place for all users to begin is with the *MicroStrategy Basic Reporting Guide*.

To access the installed manuals and other documentation sources, see the following procedures:

- [To access documentation resources from any location, page xix](#)
- [To access documentation resources on Windows, page xix](#)

- *To access documentation resources on UNIX and Linux, page xix*

To access documentation resources from any location

- 1 Visit <http://www.microstrategy.com/producthelp>.

To access documentation resources on Windows

- 1 From the Windows **Start** menu, choose **Programs** (or **All Programs**), **MicroStrategy Documentation**, then **Product Manuals**. A page opens in your browser showing a list of available manuals in PDF format and other documentation sources.
- 2 Click the link for the desired manual or other documentation source.



If bookmarks are not visible on the left side of a product manual, from the **View** menu click **Bookmarks and Page**. This step varies slightly depending on your version of Adobe Reader.

To access documentation resources on UNIX and Linux

- 1 Within your UNIX or Linux machine, navigate to the directory where you installed MicroStrategy. The default location is `/opt/MicroStrategy`, or `$HOME/MicroStrategy/install` if you do not have write access to `/opt/MicroStrategy`.
- 2 From the MicroStrategy installation directory, open the `Help` folder.
- 3 Open the `Product_Manuals.htm` file in a web browser. A page opens in your browser showing a list of available manuals in PDF format and other documentation sources.
- 4 Click the link for the desired manual or other documentation source.





If bookmarks are not visible on the left side of a product manual, from the **View** menu click **Bookmarks and Page**. This step varies slightly depending on your version of Adobe Reader.

Documentation standards

MicroStrategy online help and PDF manuals (available both online and in printed format) use standards to help you identify certain types of content. The following table lists these standards.



These standards may differ depending on the language of this manual; some languages have rules that supersede the table below.

Type	Indicates
bold	<ul style="list-style-type: none"> Button names, check boxes, options, lists, and menus that are the focus of actions or part of a list of such GUI elements and their definitions <p>Example: Click Select Warehouse.</p>
<i>italic</i>	<ul style="list-style-type: none"> Names of other product manuals and documentation resources When part of a command syntax, indicates variable information to be replaced by the user <p>Example: Type <i>copy c:\filename d:\foldername\filename</i></p>
Courier font	<ul style="list-style-type: none"> Calculations Code samples Registry keys Path and file names URLs Messages displayed in the screen Text to be entered by the user <p>Example: <code>Sum(revenue)/number of months</code>.</p> <p>Example: Type <code>cmdmgr -f scriptfile.scp</code> and press Enter.</p>
+	A keyboard command that calls for the use of more than one key (for example, SHIFT+F1).
	A note icon indicates helpful information for specific situations.
	A warning icon alerts you to important information such as potential security risks; these should be read before continuing.

Education

MicroStrategy Education Services provides a comprehensive curriculum and highly skilled education consultants. Many customers and partners from over 800 different organizations have benefited from MicroStrategy instruction. For a detailed description of education offerings and course curriculums, visit <http://www.microstrategy.com/Education>.

Consulting

MicroStrategy Consulting Services provides proven methods for delivering leading-edge technology solutions. Offerings include complex security architecture designs, performance and tuning, project and testing strategies and recommendations, strategic planning, and more. For a detailed description of consulting offerings, visit <http://www.microstrategy.com/services-support/consulting>.

Technical Support

If you have questions about a specific MicroStrategy product, you should:

- 1 Consult the product guides, Help, and readme files. Locations to access each are described above.
- 2 Consult the MicroStrategy Knowledge Base online at <https://resource.microstrategy.com/support>.



A technical administrator in your organization may be able to help you resolve your issues immediately.

- 3 MicroStrategy Technical Support can be contacted by your company's Support Liaison. Contact information and the Technical Support policy information is available at <http://www.microstrategy.com/services-support/support/contact>.

Feedback

Please send any comments or suggestions about user documentation for MicroStrategy products to:

`documentationfeedback@microstrategy.com`

Send suggestions for product enhancements to:

`support@microstrategy.com`

When you provide feedback to us, please include the name and version of the products you are currently using. Your feedback is important to us as we prepare for future releases.

ABOUT MDX CUBE SOURCES IN MICROSTRATEGY

Introduction

Many companies have stored data that is specifically structured for query, reporting, and analysis, known as a *data warehouse*, and they also have data stored in an *MDX cube source* such as SAP Business Intelligence Warehouse (SAP BW), Microsoft Analysis Services (Analysis Services), Oracle Essbase, or IBM Cognos TM1 (TM1). This system setup requires an integrated *business intelligence system* (BI), such as MicroStrategy, that can concurrently access both the MDX cube source and the data warehouse effectively.

MicroStrategy provides a rich set of functionality ranging from OLAP Services and Report Services to Narrowcast capabilities, all of which can be exposed in a unified Web interface. Using the MicroStrategy standard interface, Intelligence Server can join data from different MDX cube sources, in addition to relational databases, and bring the data into one cohesive user community known as a MicroStrategy *project*. These additional MDX cube sources include the following:

- SAP BW 3.1, 3.5, 7.0, 7.3, and 7.4
- Microsoft Analysis Services 2005, 2008, 2012, and 2014

You can also use Microsoft Analysis Services 2012 in Tabular Mode through the use of InMemory as the query mode.

For purposes of integrating with MicroStrategy, these versions of Microsoft Analysis Services provide the same features and functionality.

- Oracle Essbase Server 11
- Oracle Essbase Server 9.3
- IBM Cognos TM1 10.x or 9.5.x

It is important to understand that the MicroStrategy integration with MDX cube sources does not change the overall structure of MicroStrategy software. Rather, integration allows MicroStrategy to gain additional data sources for analysis. In other words, each of the products can be thought of as simply another *data source*, which is any file, system, or storage location that stores data to be used in MicroStrategy for query, reporting, and analysis.

This chapter builds on basic knowledge of MicroStrategy and MDX cube source terminology and architecture. It describes how MicroStrategy Intelligence Server converts MDX cube source data and objects into MicroStrategy using MultiDimensional Expressions (MDX). SAP BW obtains data from R/3, CRM, SEM, or another SAP data source system. This data is stored in cubes or other SAP objects. Likewise, Analysis Services, Oracle Essbase, and TM1 store data in cubes obtained from various sources. To access the data and convert it into MicroStrategy standard objects and conventions, Intelligence Server generates MDX. Defined by Microsoft, MDX is similar to SQL but is used to query cubes. An MDX expression returns a multidimensional result set (dataset) that consists of axis data, cell data, and properties data. For more information on MDX syntax, refer to <http://msdn.microsoft.com/> and search for MDX.

If you use MDX cube sources and MicroStrategy as your combined BI solution, you can get the best out of both, including the following:

- Access to MDX cube sources and a regular data warehouse
- Five styles of BI
- Custom development of reports and applications
- Transaction-level analysis
- Integration with other systems via Web Services

If you upgrade a pre-9.0 MicroStrategy project that includes MDX cubes, see the *Upgrade Guide* for information on updating MDX objects to enhance the performance of using these objects in MicroStrategy.

For troubleshooting and diagnostics logging routines related to MDX cube sources, see the *Troubleshooting* chapter of the *MicroStrategy System Administration Guide*.

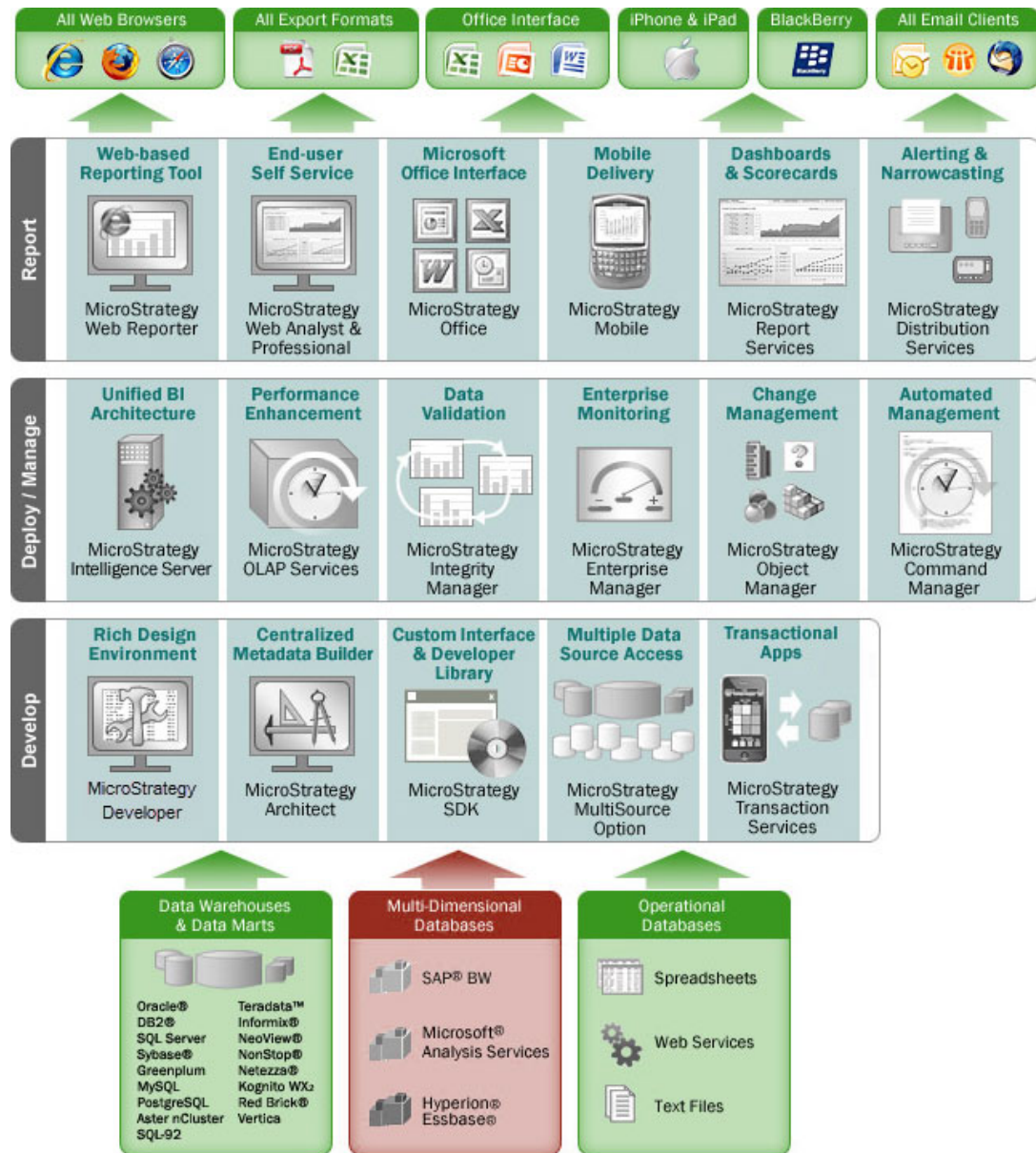
Understanding MicroStrategy architecture

The MicroStrategy platform offers OLAP Services, Report Services, and Narrowcast Server functionality, all of which can be accessed through MicroStrategy Web. Support for SAP BW, Analysis Services, Oracle Essbase, TM1, MicroStrategy Freeform SQL, and MicroStrategy Query Builder provides additional mechanisms for pulling data into the MicroStrategy platform for analysis, as illustrated in the diagram below.

For information on Freeform SQL and Query Builder reporting, refer to the *MicroStrategy Advanced Reporting Guide*.

Data is pulled from multiple MDX cube sources using MDX, and from operational systems using Freeform SQL or Query Builder. Once the data is retrieved, it is treated in the same manner as data pulled from a relational

data warehouse. This means that core MicroStrategy reporting capabilities are available no matter what the original data source is.



MicroStrategy uses a repository known as a *metadata*, whose data associates the tables and columns of a data source with user-defined attributes and facts to enable the mapping of the business view, terms, and needs to the underlying database structure. You can have multiple MicroStrategy projects, each pointing to a data source. A *database instance* represents the connection to a data source, and one database instance could be referenced by multiple projects in a configuration. Each project contains a project *schema*, a set of tables and relationships that defined the logical model for that project. To learn more about how MDX cube sources can be used in addition to a project schema, see [The MicroStrategy object model](#) below.

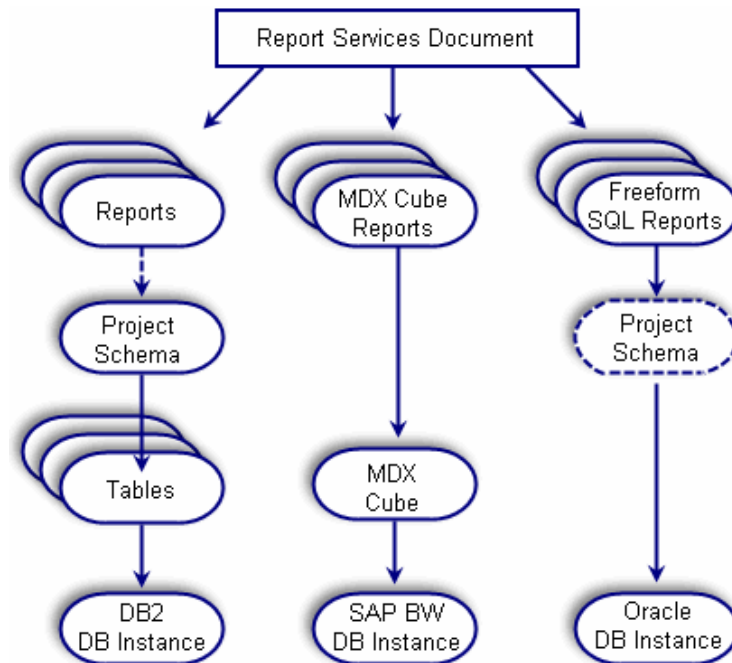
The MicroStrategy object model

The MicroStrategy model shown below highlights how a project can be extended to access MDX cube sources through a separate database instance. However, instead of pointing to the project schema, each MDX cube report points directly to one *MDX cube* in MicroStrategy, which is a logical placeholder for a physical cube that exists in an MDX cube source. Each report can only reference one specific MDX cube, due to the structure in MDX cube sources where queries can only be run against one physical cube at a time. You can create multiple reports to run against one MDX cube, and a single MicroStrategy project can reference multiple database instances, each of which can represent a distinct MDX cube source. For more information on how MicroStrategy can connect to MDX cube database instances along with database instances connected to your relational data warehouse, see [Configuring MDX cube database instances in projects, page 59](#).

The MicroStrategy object model also shows how you can include any number of standard reports, Freeform SQL reports, Query Builder reports, and MDX cube reports in one Report Services document. By bringing these different types of reports together inside a document, report designers can create rich reports and analytics that take advantage of data from both data warehouses and MDX cube sources. You must import and map the data from MDX cubes into MicroStrategy in a manner that fits your data type conventions to support creating relationships between various MDX cube reports as well as between MDX cube reports and standard MicroStrategy reports (see [Defining column data types for MDX cube data, page 109](#)).

For information on Report Services documents, refer to the *MicroStrategy Document Creation Guide* and the *MicroStrategy Document Analysis*

Guide. For information on Freeform SQL and Query Builder reports, refer to the *MicroStrategy Advanced Reporting Guide*.



Relating objects from MDX cube sources to MicroStrategy

This section describes how objects from the various MDX cube sources relate to and are converted into MicroStrategy objects. It is important to understand these relationship before you connect to your MDX cube source and import MDX cubes so that you are fully aware of how your MDX cube source data will be represented in MicroStrategy.

The following MDX cube sources and topics are covered:

- [Understanding SAP BW terminology, page 7](#)
- [Relating objects from SAP BW to MicroStrategy, page 9](#)
- [Relating objects from Oracle Essbase to MicroStrategy, page 18](#)
- [Relating objects from Analysis Services to MicroStrategy, page 24](#)
- [Relating objects from TM1 to MicroStrategy, page 29](#)

Understanding SAP BW terminology

Before you review how SAP BW is converted into a MicroStrategy environment ([Relating objects from SAP BW to MicroStrategy, page 9](#)), you should be familiar with some basic SAP BW terminology. The following list provides information on SAP BW objects and how they relate to and function in MicroStrategy:



For a comprehensive explanation of SAP BW objects, refer to your SAP documentation.

- **InfoObjects:** are the building blocks for individual cubes. They include objects such as characteristics and key figures, which are roughly equivalent to attributes and facts in a MicroStrategy project.
- **InfoProviders:** are all SAP BW data structures available for reporting and analysis purposes, such as the following:
 - **InfoCubes:** are multi-dimensional cubes, which are described in more detail below.
 - **ODS objects:** are operational data store objects. ODS objects are flat relational tables and are similar to MicroStrategy fact tables.
 - **MultiProviders:** are logical unions of two or more InfoProviders that are used to combine data from two different subject areas, for example, three InfoCubes or two ODS objects.
- **InfoCubes:** are the primary objects that SAP BW uses to store data for analysis. InfoCubes define a specific domain of analysis in special areas, for example, finance or sales. Data is organized by dimension and stored physically in a star schema. The fact table at the center of an InfoCube contains the data available for analysis.
- **Query cubes (or query):** defines a subset of data from an InfoCube or another InfoProvider. A query cube includes characteristics (dimensions/attributes) and key figures (metrics) from its source provider. The relationship between the query cube and the InfoCube is similar to the relationship between a MicroStrategy report and your data warehouse, in that a MicroStrategy report includes a subset of modeled attributes and metrics that are available in the data warehouse.

Query cubes generally offer better performance than InfoCubes because they are smaller and can be scheduled and cached within SAP BW. Query cubes also provide MicroStrategy users access to additional InfoProviders including ODS objects, InfoSets, and MultiProviders.

Any existing SAP BW query can be released for analysis within MicroStrategy. To release a query for analysis in MicroStrategy, select the **Allow External Access to This Query** check box under the Extended tab in the SAP Query Properties dialog box in the Query Analyzer interface. With this option enabled, report designers can quickly access existing query cubes and business content when working in MicroStrategy.

- **Characteristics:** provide classification possibilities for a dataset, such as sales region, product, customer group, fiscal year, and period. For example, a Sales Region characteristic can have North, Central, and South specifications.

SAP BW characteristics are similar to MicroStrategy attributes. However, when each characteristic is translated into a cube, it is treated as a separate dimension for analysis. In addition, SAP BW hierarchies can be associated with a specific characteristic within SAP BW. These hierarchies are also available when you work with an MDX cube in MicroStrategy.



SAP BW also has an object called an attribute, which is equivalent to an attribute form in MicroStrategy, not a MicroStrategy attribute.

- **Key figures:** describe numeric data, such as revenue, profit, and number of call centers. There are five types of key figures: amounts, quantities, numbers, dates, and times, all of which can be used in InfoCubes, ODS objects, and master data attributes. You can also create calculated key figures and restricted key figures in the query definition in the Business Explorer. This is similar to creating derived metrics and conditional metrics in MicroStrategy.
- **Hierarchies:** are objects that define the relationships among elements within a characteristic. For example, the Item characteristic might have a hierarchy that includes Category, Subcategory, and finally Item. This is a different paradigm from MicroStrategy's model where each attribute defines its own level. However, when the levels of a hierarchy are viewed in MicroStrategy, they are presented with the traditional attribute-based parent-child relationships.
- **Variables:** are used as parameters of a query in SAP BW. Defined in the Query Designer, variables can be of such types as characteristic values, hierarchies, hierarchy nodes, texts, and formulas. When a query is executed, these variables include values supplied by the system or by the user.

When an MDX cube is imported into a MicroStrategy project, all the variables in the MDX cube are represented as MicroStrategy prompts.

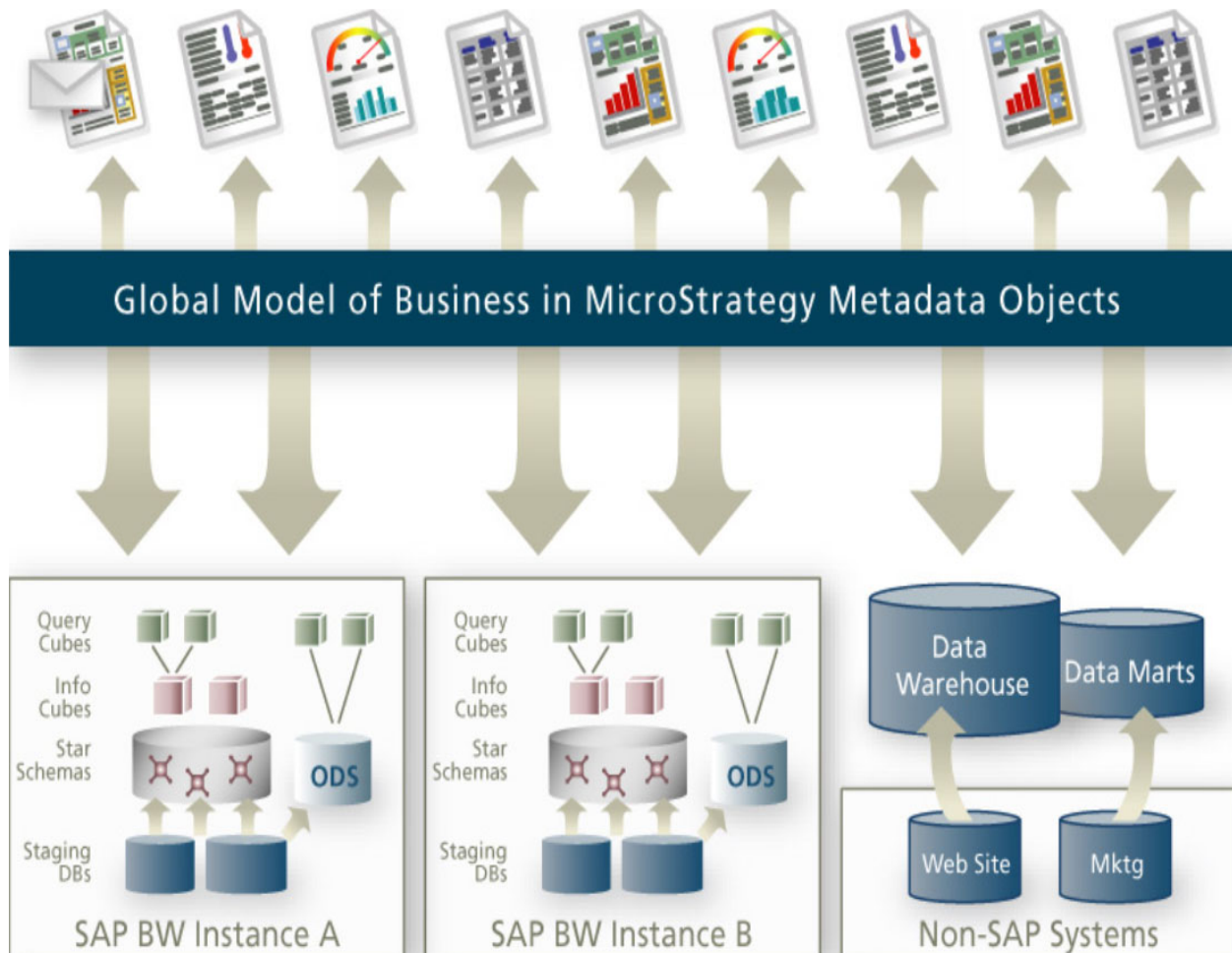
When the MDX cube is used to create a MicroStrategy MDX cube report, the MDX cube report inherits all of the prompts. For more information on variables and how they are converted to prompts in MicroStrategy, see [Converting SAP BW variables into MicroStrategy prompts, page 16](#).

MicroStrategy lists all available MDX cubes for reporting and analysis in the MicroStrategy MDX Cube Catalog, including all of the published query cubes, InfoCubes, and MultiProviders. For information on using the MDX Cube Catalog to import MDX cubes into MicroStrategy, see [Importing MDX cubes, page 80](#).

Relating objects from SAP BW to MicroStrategy

As a MicroStrategy Web or Developer analyst, you can treat MDX cube reports based off SAP BW MDX cubes as if they were standard MicroStrategy reports. If you are a report, document, or dashboard designer, it is helpful to

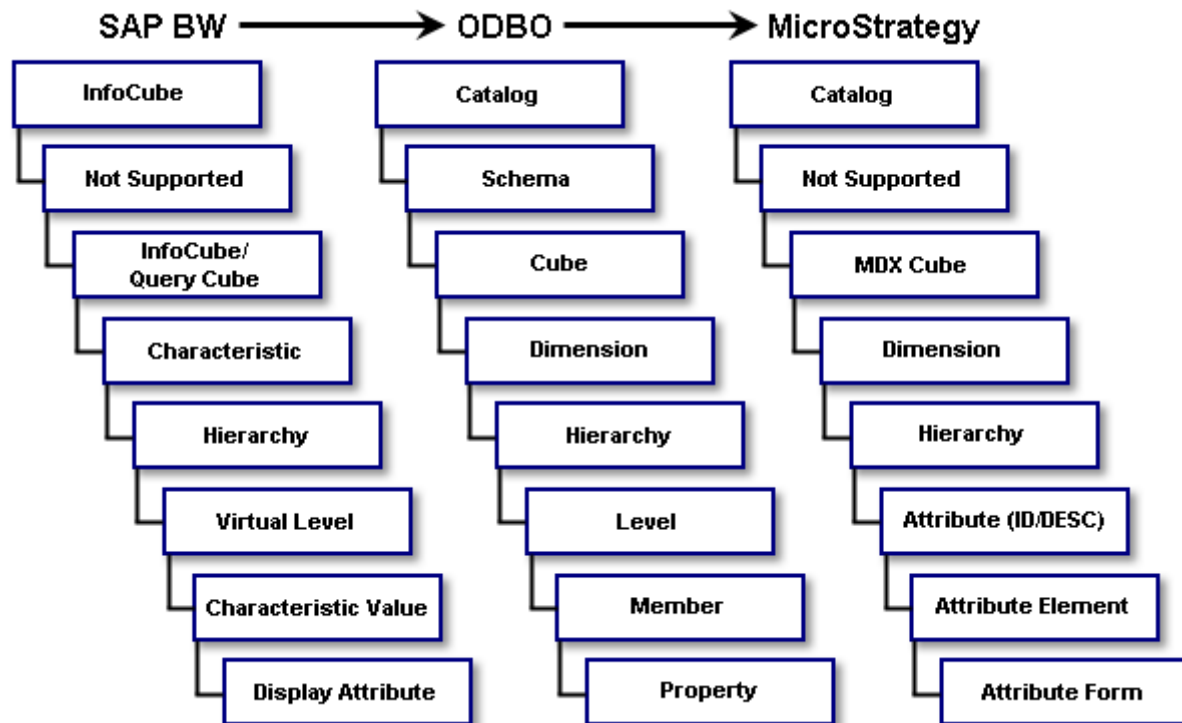
understand how SAP's metadata model is translated into MicroStrategy's metadata model.



The translation process involves the following steps:

- 1 **From SAP BW to ODBO:** SAP exposes its query cubes and InfoCubes to Intelligence Server through the ODBO model. ODBO (OLE database for OLAP) is a protocol defined by Microsoft. ODBO defines an object model that is used in conjunction with MDX to query cubes. The ODBO model is similar to SAP's standard model, but not identical. Thus, when thinking about SAP objects, keep in mind how those objects appear in ODBO.
- 2 **From ODBO to MicroStrategy:** After SAP objects are translated into the ODBO model, they are translated into the MicroStrategy metadata model. You can then interact with SAP content while working within the paradigm that is consistent with the rest of MicroStrategy's products.

The following diagram shows how SAP BW objects are exposed in ODBO and then how they are related to objects in the MicroStrategy environment.



Each layer in the diagram above is described separately in the following sections. The information below provides high-level descriptions of the SAP BW, ODBO, and MicroStrategy objects that are involved in each conversion and how they relate to each other. For more detailed information on SAP BW objects, see [Understanding SAP BW terminology, page 7](#).

InfoCube

SAP BW --->	ODBO --->	MicroStrategy
InfoCube	Catalog	(Catalog)

- SAP BW:** InfoCube

Each InfoCube that has queries associated with it is exposed as a catalog in ODBO. Query cubes are accessed through their respective InfoCube catalogs.

- **ODBO:** Catalog

Catalogs are used to group cubes. Therefore, ODBO catalogs are exposed in a few editors when selecting and managing cubes.

- **MicroStrategy:** (Catalog)

Each catalog includes one InfoCube and associated query cubes, if any. Catalogs in MicroStrategy are represented in a folder.

ODBO schema

SAP BW --->	ODBO --->	MicroStrategy
Not supported	Schema	Not supported

- **SAP BW:** Not supported

- **ODBO:** Schema

A schema in ODBO provides a grouping mechanism not supported by SAP BW or MicroStrategy.

- **MicroStrategy:** Not supported

InfoCube/query cube

SAP BW --->	ODBO --->	MicroStrategy
InfoCube/Query cube	Cube	MDX cube

- **SAP BW:** InfoCube/Query cube

- **ODBO:** Cube

- **MicroStrategy:** MDX cube

A MicroStrategy MDX cube is an *object* that is used to map the levels of an SAP BW cube into the MicroStrategy environment. MDX cubes can be thought of as similar to tables in the MicroStrategy metadata. In the same way that a metadata table maps the physical columns of a relational table to attributes and metrics, a MicroStrategy MDX cube maps the physical columns of an SAP BW cube to attributes and metrics. In MicroStrategy, *attributes* define the business context of data and *metrics* supply the data and calculations which can be combined on reports and documents to

present your data. The MDX cube can be used to represent InfoCubes, MultiProviders, and query cubes.


Characteristic

SAP BW --->	ODBO --->	MicroStrategy
Characteristic	Dimension	Dimension

- **SAP BW:** Characteristic


Characteristics in SAP BW are similar to attributes in MicroStrategy. For example, an InfoCube might include the Month characteristic, which represents months just like it does in a MicroStrategy attribute called Month.

A characteristic appears as a dimension for MicroStrategy users. Each characteristic (dimension) has at least one hierarchy with two levels: the first level is an aggregate of all the data related to the characteristic, and the second level is the detailed data.

 When you import an MDX cube into MicroStrategy, you can control whether a characteristic's first level of aggregate data is imported and whether the second level is imported with a Level 01 suffix. For more information on setting import options for characteristics, see [Importing levels and suffixes for characteristics, page 85](#).

A characteristic can have any number of additional hierarchies, each with an arbitrary number of levels. The SAP BW characteristic hierarchies appear as MicroStrategy hierarchies to MicroStrategy users.

For example, in SAP BW you can build a Time hierarchy that is attached to the Month characteristic. The Time hierarchy defines a number of levels including Year, Quarter, and Month. These same levels could either be specifically defined as part of the hierarchy, or they could be other characteristics that are used to define the levels of this one hierarchy.

 Shared dimensions allow a designer to use only one definition for a dimension across multiple cubes. Each characteristic in SAP is modeled as a dimension in ODBO and is shared across cubes. Therefore, all dimensions in cubes coming from SAP BW are shared.

- **ODBO:** Dimension

A dimension in ODBO defines a logical category of analysis. For example, Time and Geography are dimensions along which you can slice data.

Measures (metrics) are stored in a special measure dimension. In this way, measures are simply one more dimension of a cube.

Measures in ODBO are called key figures in SAP BW, which are similar to metrics in MicroStrategy, and they are represented as physical columns.

If your SAP BW MDX cube source includes variables, there are times when these variables specify a name other than [Measures] for the measure dimension name. You can specify the measure dimension name in your SAP BW MDX cube source using the VLDB properties Name of Measure Dimension and MDX Remember Measure Dimension Name. For information on how to use these VLDB properties to specify the measure dimension name, refer to the *Supplemental Reference for System Administration*.

- **MicroStrategy:** Dimension

A dimension object in MicroStrategy is similar to an ODBO dimension. It is used to group attributes and relate them to each other by defining parent-child relationships.

Hierarchy

SAP BW --->	ODBO --->	MicroStrategy
Hierarchy	Hierarchy	Hierarchy

- **SAP BW:** Hierarchy
- **ODBO:** Hierarchy
- **MicroStrategy:** Hierarchy

A *hierarchy* is used to group attributes (levels) together and define the relationships between these attributes. MicroStrategy uses hierarchy objects to represent both dimensions and hierarchies from ODBO.

Virtual level

SAP BW --->	ODBO --->	MicroStrategy
Virtual level	Level	Attribute

- **SAP BW:** Virtual level

Levels are generated automatically based on either the definition of the characteristic or the hierarchies associated with a characteristic.



SAP BW levels have names such as Region Level 01, Region Level 02, and so on. The inclusion of the term “Level” is an SAP BW convention. In MicroStrategy, architects have the option to rename the levels of a cube with a more user-friendly convention.

- **ODBO:** Level
- **MicroStrategy:** Attribute (ID/DESC)

MicroStrategy attributes map to ODBO levels. Each ODBO level generates two physical columns and forms in MicroStrategy: ID and DESC.

Characteristic value

SAP BW --->	ODBO --->	MicroStrategy
Characteristic value	Member	Attribute element

- **SAP BW:** Characteristic value
- **ODBO:** Member
- **MicroStrategy:** Attribute element

Attribute elements are unique sets of attribute information which can come from either a database or an MDX cube imported from an MDX cube source. For example, 2003 and 2004 are attribute elements of the Year attribute.

Characteristic attribute

SAP BW --->	ODBO --->	MicroStrategy
Characteristic attribute	Property	Attribute form

- **SAP BW:** Characteristic attribute



In SAP BW, forms are sometimes referred to directly as attributes. SAP BW also supports navigational attributes. These attributes are presented as distinct dimensions when working in MicroStrategy.

- **ODBO:** Property
- **MicroStrategy:** Attribute form

Attribute forms provide additional information about a given attribute. For example, the Customer attribute may have the forms First Name and Last Name. This concept also applies to ODBO and SAP BW.

Converting SAP BW variables into MicroStrategy prompts

Variables in SAP BW allow users to enter values as parameters for the queries on a cube. Types of variables include characteristic values, hierarchies, hierarchy nodes, texts, and formula elements. For detailed information on variables, refer to your SAP documentation. The rest of this section addresses how variables are handled in MicroStrategy.

Variable types with the Customer Exit/SAP Exit and Authorization processing types are automatically resolved by the SAP BW system. Only variables with the Manual Entry/Default processing type are presented to users for resolution.



SAP BW variables of the type Replacement Path cannot be imported into MicroStrategy. If your SAP BW cube includes variables of the type Replacement Path, you must remove them before importing the cube into MicroStrategy. Otherwise, an error occurs when you attempt to import the SAP BW cube.

Originally created in an SAP query cube, variables are represented as prompts in the MicroStrategy environment. MicroStrategy users answer

prompts to complete the definition of a report at report execution time. The conversion process involves the following general steps:

- 1** When an SAP query cube is imported into a MicroStrategy project, variables are automatically turned into prompts in the MicroStrategy MDX cube.
- 2** When a MicroStrategy report is created using a MicroStrategy MDX cube, the report inherits the prompts included in the MDX cube.



In addition to the “inherited” variable prompts, standard MicroStrategy prompts can also be created for an MDX cube report. For more information, see [Prompts on MDX cube reports, page 169](#).

The following table contains information on how the different types of SAP BW variables are mapped to MicroStrategy prompts.

SAP Variable Type	MicroStrategy Prompt	Notes
Characteristic Value variable	Element list prompt or attribute qualification prompt	Characteristic value variables offer an “Including/Excluding” option. Qualifications in the Including section cause the data to be included in the query, while those in the Excluding section restrict the data from being displayed in the query. To be consistent with the SAP functionality, the MicroStrategy interface qualifies on the key value of each element by default.
Hierarchy variable	N/A	Not supported.
Hierarchy node variable	Hierarchy element list prompt	Both single and multiple selections are supported.
Text variable	Text value prompt	SAP text variables that provide dynamic names to SAP measures are converted to text value prompts in MicroStrategy. These prompts are then used to provide a dynamic metric alias for the MicroStrategy metric mapped to the SAP measure. When these MicroStrategy metrics are displayed on reports, it's metric name reflects the text value provided from the SAP text variable.
Formula variable	All types of value prompts	No major differences.

If you use any SAP BW key date variables in your query, you need to manually define the variable as a key date variable. You can define key date variables using the MicroStrategy MDX Cube Catalog while mapping your MDX cubes (see [Supporting SAP BW key date variables, page 126](#)).

If you use any SAP BW variables with complex expression qualifications in your query, you can define which form is used to evaluate the variable's qualification, as described in [Supporting SAP BW variable qualifications, page 127](#).

Supporting SAP BW structures

Structures in an SAP BW query cube define the two axes of a query (rows and columns). The two types of SAP BW structures are:

- Key figure structures, which are mapped to unique metrics in the MicroStrategy environment.
- Characteristic structures, which are represented as a single, flat dimension with one level mapped to a unique attribute in the MicroStrategy environment. This representation is consistent with how characteristic variables are represented in SAP BW through the OLAP Business Application Programming Interface (BAPI).



In a MicroStrategy MDX cube report, you cannot drill down into the elements of characteristic structures.


When characteristic structures are imported into MicroStrategy as attributes, the structure element orders are preserved. This maintains a consistent view of your data across your SAP BW MDX cube source and MicroStrategy. Once imported into MicroStrategy, you can sort reports based on the structure element order. For information on sorting reports on structure element orders, see [Sorting structure elements and preserving order, page 177](#).

Characteristic structures imported as attributes in MicroStrategy can also affect the value formats of metrics on MDX cube reports. For information on how you can specify that the metric values in MicroStrategy MDX cube reports inherit value formats from an MDX cube source, see [Inheriting MDX cube source formats for metric values, page 181](#).

Relating objects from Oracle Essbase to MicroStrategy

As a MicroStrategy Web or Developer analyst, you can treat MDX cube reports from an Oracle Essbase 9.3.1 or Oracle Essbase 11 MDX cube like standard MicroStrategy reports. If you are a report designer, it is helpful to

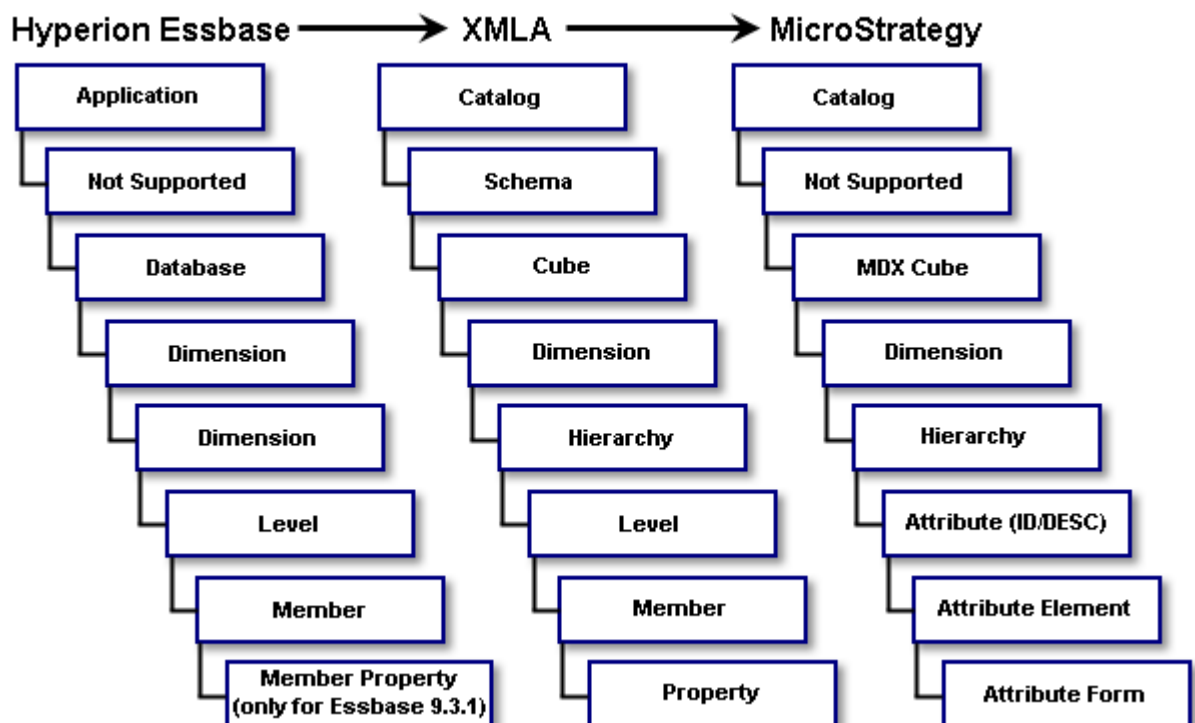
understand how Oracle Essbase's metadata model is translated into MicroStrategy's metadata model.

 In this section, the term Oracle Essbase is used to refer to Hyperion Essbase 9.3.1, as well as Oracle Essbase 11.

The translation process involves the following steps:

- 1 From Oracle Essbase to XMLA:** Oracle Essbase exposes its databases through the XMLA model which is derived from the ODBO model. XMLA defines an object model that is used in conjunction with MDX to query cubes. The Oracle Essbase model pre-dates XMLA so there are some differences. When thinking about Oracle Essbase objects, keep in mind how those objects appear in XMLA.
- 2 From XMLA to MicroStrategy:** After Oracle Essbase objects are translated into the XMLA model, they are translated into the MicroStrategy metadata model. You can then interact with Oracle Essbase content while working within the paradigm that is consistent with the rest of MicroStrategy's products.

The following diagram shows how Oracle Essbase objects are exposed in XMLA and then how they are related to objects in the MicroStrategy environment.



Each layer in the diagram above is described separately in the following sections. The information below provides high-level descriptions of the Oracle Essbase, XMLA, and MicroStrategy objects that are involved in each conversion and how they relate to each other.

Application

Oracle Essbase --->	XMLA --->	MicroStrategy
Application	Catalog	(Catalog)

- **Oracle Essbase:** Application

Each application is exposed as a catalog in XMLA. Databases are accessed through their respective catalogs.

- **XMLA:** Catalog

Catalogs are used to group cubes. Therefore, XMLA catalogs are exposed in editors when selecting and managing cubes.

- **MicroStrategy:** Catalog

Each catalog includes one application and associated databases, if any. Catalogs in MicroStrategy are represented as a folder.

XMLA schema

Oracle Essbase --->	XMLA --->	MicroStrategy
Not supported	Schema	Not supported

- **Oracle Essbase:** Not supported

- **XMLA:** Schema

A schema in XMLA provides a grouping mechanism not supported by Oracle Essbase or MicroStrategy.

- **MicroStrategy:** Not supported

Database

Oracle Essbase --->	XMLA --->	MicroStrategy
Database	Cube	MDX cube

- **Oracle Essbase:** Database
- **XMLA:** Cube
- **MicroStrategy:** MDX cube

A MicroStrategy MDX cube is an object that is used to map the levels of a Oracle Essbase cube into the MicroStrategy environment. MDX cubes are treated similarly to tables in the MicroStrategy metadata. A MicroStrategy MDX cube maps the physical columns of a Oracle Essbase cube to attributes and metrics in the same way that a metadata table maps the physical columns of a relational table to attributes and metrics. The MDX cube represents a Oracle Essbase database.

Dimension

Oracle Essbase --->	XMLA --->	MicroStrategy
Dimension	Dimension	Dimension

- **Oracle Essbase:** Dimension

In Oracle Essbase, a dimension represents the highest consolidation level in the database outline. The dimension is therefore both the highest level member in the dimension and the dimension itself. Each dimension has a single root node or member and is a child of the outline root node which is the database.

A Oracle Essbase dimension appears as a MicroStrategy dimension for MicroStrategy users. Each dimension has a single hierarchy with the number of levels determined by the greatest depth in the outline.

- **XMLA:** Dimension

A dimension in XMLA defines a logical category of analysis. For example, Time and Geography are dimensions along which you can slice data.

Measures (metrics) are stored in a special measure dimension. In this way, measures are simply one more dimension of a cube.

Measures in XMLA are the members of the dimension of type=Accounts in Oracle Essbase. These can be raw data or formulas with associated calculation or aggregation rules.

When importing MDX cubes from Oracle Essbase, you can import the measures as a regular dimension. This can retain the structure of the measures as they exist in the Oracle Essbase MDX cube source. For information on how to import measures as a regular dimension, see [Importing additional measure structure, page 88](#).

If the measures dimension in your Essbase MDX cube source uses a different name other than [Measures] you can specify the name in your Essbase MDX cube source using the VLDB properties Name of Measure Dimension and MDX Remember Measure Dimension Name. For information on how to use these VLDB properties to specify the measure dimension name, refer to the *Supplemental Reference for System Administration*.

- **MicroStrategy:** Dimension

A dimension object in MicroStrategy is similar to an XMLA dimension. It is used to group attributes and relate them to each other by defining parent-child relationships.

Dimension

Oracle Essbase --->	XMLA --->	MicroStrategy
Dimension	Hierarchy	Hierarchy

- **Oracle Essbase:** Dimension

A Oracle Essbase dimension is defined as part of the database outline. The outline is a hierarchical structure of database members with a parent containing its children. As a result, the outline defines a single hierarchy. Therefore the dimension is the same as a MicroStrategy hierarchy.

- **XMLA:** Hierarchy

- **MicroStrategy:** Hierarchy

Hierarchies are used to group attributes (levels) together and define the relationships between these attributes. MicroStrategy uses hierarchy objects to represent both dimensions and hierarchies from XMLA.

Level

Oracle Essbase --->	XMLA --->	MicroStrategy
Level	Level	Attribute

- **Oracle Essbase:** Level

Levels group together members in a Oracle Essbase database outline.



Oracle Essbase levels may have default names such as Time.Levels(o). In MicroStrategy, architects have the option to rename the levels of a cube with a more user-friendly convention.

- **XMLA:** Level
- **MicroStrategy:** Attribute (ID/DESC)

MicroStrategy attributes map to XMLA levels. Each XMLA level also generates two physical columns and the forms ID and DESC in MicroStrategy.

Member

Oracle Essbase --->	XMLA --->	MicroStrategy
Member	Member	Attribute element

- **Oracle Essbase:** Member
- **XMLA:** Member
- **MicroStrategy:** Attribute element

Element values come from either the database or a cube. For example, 2003 and 2004 are elements of the Year attribute.

Member property

Oracle Essbase 9.3.1 and Oracle Essbase 11 --->	XMLA --->	MicroStrategy
Member property	Property	Attribute form

- **Oracle Essbase 9.3.1 and Oracle Essbase 11:** Member property

In Oracle Essbase 9.3.1 and Oracle Essbase 11, a member property is a user-defined property that acts as a descriptive piece of information associated with an outline member.

- **XMLA:** Property
- **MicroStrategy:** Attribute form

Attribute forms provide additional information about a given attribute. For example, the Customer attribute may have the forms First Name and Last Name.

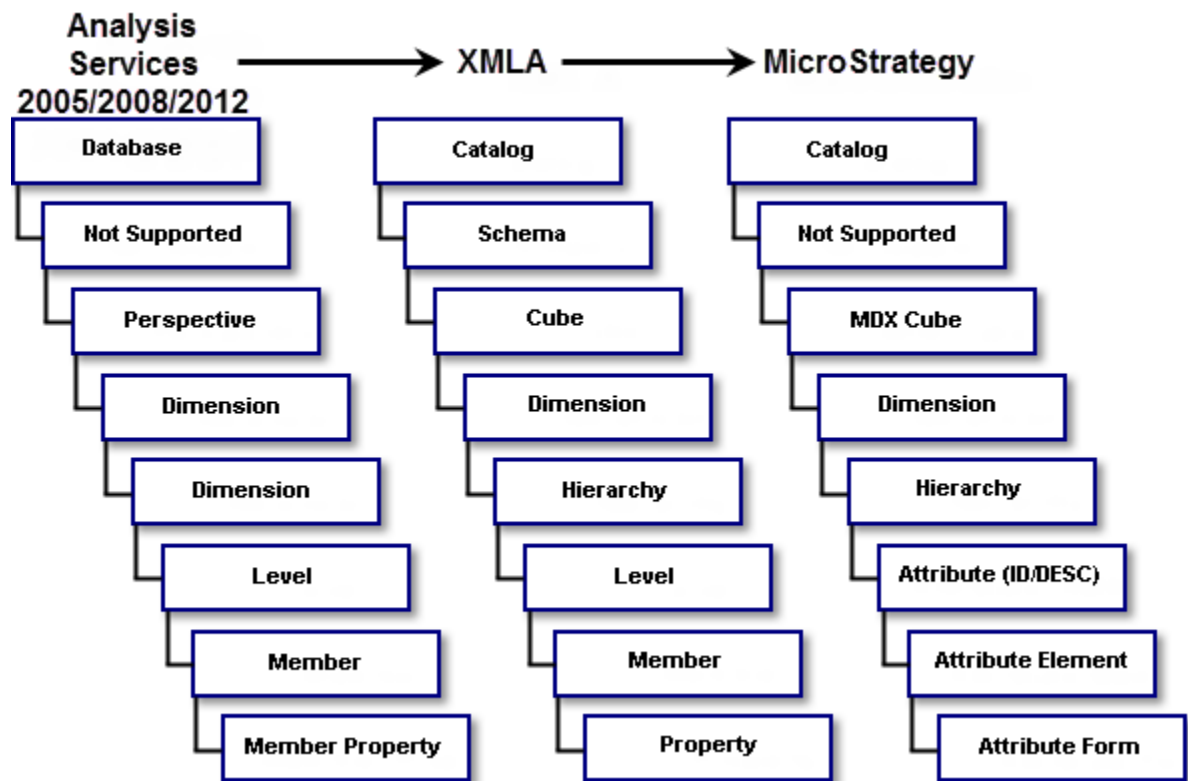
Relating objects from Analysis Services to MicroStrategy

Microsoft Analysis Services 2005, 2008, and 2012 (Analysis Services 2005/2008/2012) has a unique modeling approach for building cubes. The discussion provided below is limited to information on the basic cube object and how it relates to the XMLA model, and then to MicroStrategy.

The translation process involves the following steps:

- 1 **From Analysis Services 2005/2008/2012 to XMLA:** Analysis Services 2005/2008/2012 expose cubes through the XMLA model which is derived from the ODBO model. XMLA defines an object model that is used in conjunction with MDX to query cubes.
- 2 **From XMLA to MicroStrategy:** After Analysis Services 2005/2008/2012 objects are translated into the XMLA model, they are translated into the MicroStrategy metadata model. You can then interact with Analysis Services 2005/2008/2012 content while working within the paradigm that is consistent with the rest of MicroStrategy's products.

The following diagrams shows how Analysis Services 2005/2008/2012 objects are exposed in XMLA and then how they are related to objects in the MicroStrategy environment.



Each layer in the diagram above is described separately in the following sections. The information below provides high-level descriptions of the Analysis Services 2005/2008/2012, XMLA, and MicroStrategy objects that are involved in each conversion and how they relate to each other.

Database

Analysis Services 2005/2008/2012 --->	XMLA --->	MicroStrategy
database	Catalog	Catalog

- Analysis Services 2005/2008/2012: Database**

Each database is exposed as a catalog in XMLA. Cubes are accessed through their respective catalogs.

- **XMLA:** Catalog

Catalogs are used to group cubes. Therefore, XMLA catalogs are exposed in editors when selecting and managing cubes.

- **MicroStrategy:** Catalog

Each catalog includes one database and associated cubes, if any. Catalogs in MicroStrategy are represented as a folder.

XMLA schema

Analysis Services 2005/2008/2012 --->	XMLA --->	MicroStrategy
Not supported	Schema	Not supported

- **Analysis Services 2005/2008/2012:** Not supported

- **XMLA:** Schema

A schema in XMLA provides a grouping mechanism not supported by Analysis Services 2005/2008/2012 or MicroStrategy.

- **MicroStrategy:** Not supported

Perspective

Analysis Services 2005/2008/2012 --->	XMLA --->	MicroStrategy
Perspective	Cube	MDX cube

- **Analysis Services 2005/2008/2012:** Perspective

A perspective in Analysis Services 2005/2008/2012 is a view of the defined cube. The list of perspectives includes the original cube.

- **XMLA:** Cube

- **MicroStrategy:** MDX cube

A MicroStrategy MDX cube is an object that is used to map the levels of an Analysis Services 2005/2008/2012 cube into the MicroStrategy environment. MDX cubes are treated similarly to tables in the MicroStrategy metadata. In the same way that a metadata table maps the physical columns of a relational table to attributes and metrics, a

MicroStrategy MDX cube maps the physical columns of an Analysis Services 2005/2008/2012 cube to attributes and metrics. The MDX cube represents an Analysis Services 2005/2008/2012 cube.

Dimension

Analysis Services 2005/2008/2012 --->	XMLA --->	MicroStrategy
Dimension	Dimension	Dimension

- **Analysis Services 2005/2008/2012: Dimension**

In Analysis Services 2005/2008/2012, a dimension is defined from one or more data source tables. A data source does not always map directly to the tables in a relational database. All columns in the tables are eligible to become attributes of the dimension. Each attribute is used to define a hierarchy within the dimension and multi-level hierarchies can be defined as well.

An Analysis Services 2005/2008/2012 dimension appears as a MicroStrategy dimension for MicroStrategy users. Each dimension represented in MicroStrategy can have one or more MicroStrategy hierarchies.

- **XMLA: Dimension**

A dimension in XMLA defines a logical category of analysis. For example, Time and Geography are dimensions along which you can slice data.

Measures (metrics) are stored in a special measure dimension. In this way, measures are simply one more dimension of a cube.

Measures in XMLA are the members of the Measures dimension in Analysis Services 2005/2008/2012. These can be columns in the data source table or calculated members represented by formulas with associated aggregation rules.

- **MicroStrategy: Dimension**

A dimension object in MicroStrategy is similar to an XMLA dimension. It is used to group attributes and relate them to each other by defining parent-child relationships.

Hierarchy

Analysis Services 2005/2008/2012 --->	XMLA --->	MicroStrategy
Hierarchy	Hierarchy	Hierarchy

- **Analysis Services 2005/2008/2012:** Hierarchy

Analysis Services 2005/2008/2012 allows the definition of one or more hierarchies within a dimension as collections of attributes. The attributes become levels of the hierarchy.

- **XMLA:** Hierarchy
- **MicroStrategy:** Hierarchy

Hierarchies are used to group attributes (levels) together and define the relationships between these attributes. MicroStrategy uses hierarchy objects to represent both dimensions and hierarchies from XMLA.

Level

Analysis Services 2005/2008/2012 --->	XMLA --->	MicroStrategy
Level	Level	Attribute

- **Analysis Services 2005/2008/2012:** Level

Each attribute in a hierarchy becomes a level.

- **XMLA:** Level
- **MicroStrategy:** Attribute (ID/DESC)

MicroStrategy attributes map to XMLA levels. Each XMLA level generates two physical columns and forms in MicroStrategy: ID and DESC.

Member

Analysis Services 2005/2008/2012 --->	XMLA --->	MicroStrategy
Member	Member	Attribute element

- **Analysis Services 2005/2008/2012:** Member
- **XMLA:** Member
- **MicroStrategy:** Attribute element

Element values come from a cube. For example, 2003 and 2004 are elements of the Year attribute.

Member property

Analysis Services 2005/2008/2012 --->	XMLA --->	MicroStrategy
Member property	Property	Attribute form

- **Analysis Services 2005/2008/2012:** Member property

Attributes can be related as member properties when defining the levels of a dimension. Member properties are returned in the XMLA property schema rowset.

- **XMLA:** Property
- **MicroStrategy:** Attribute form



Attribute forms provide additional information about a given attribute. For example, the Customer attribute may have the forms First Name and Last Name.

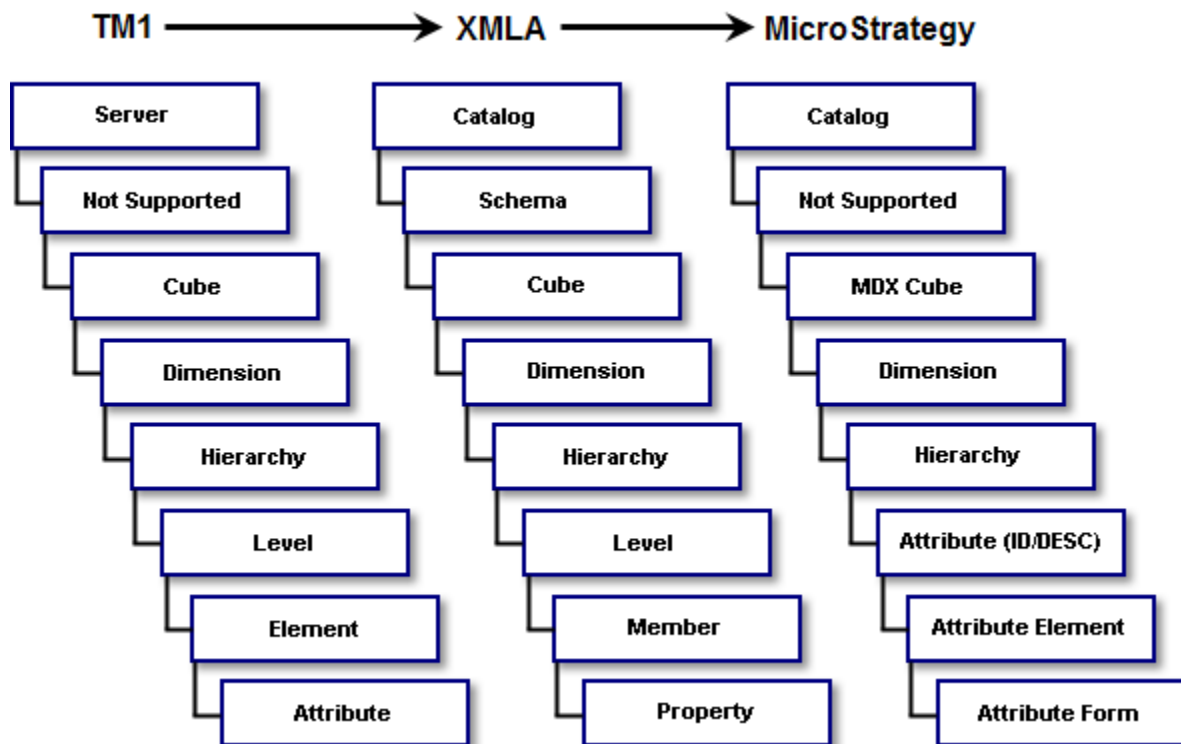
Relating objects from TM1 to MicroStrategy

TM1 has a unique modeling approach for building cubes. The discussion below is limited to information on the basic cube object and how it relates to the XMLA model, and then to MicroStrategy.

The translation process involves the following steps:

- 1 **From TM1 to XMLA:** TM1 exposes cubes through the XMLA model which is derived from the ODBO model. XMLA defines an object model that is used in conjunction with MDX to query cubes.
- 2 **From XMLA to MicroStrategy:** After TM1 objects are translated into the XMLA model, they are translated into the MicroStrategy metadata model. You can then interact with TM1 content while working within the paradigm that is consistent with the rest of MicroStrategy's products.

The following diagram shows how TM1 objects are exposed in XMLA and then how they are related to objects in the MicroStrategy environment.



Each layer in the diagram above is described separately in the following sections. The sections provide high-level descriptions of the TM1, XMLA, and MicroStrategy objects that are involved in each conversion and how they relate to each other.

Server

TM1 --->	XMLA --->	MicroStrategy
Server	Catalog	Catalog

- **TM1:** Server

Each TM1 server is exposed as a catalog in XMLA. Cubes are accessed through their respective catalogs.

- **XMLA:** Catalog

Catalogs are used to group cubes. Therefore, XMLA catalogs are exposed in editors when selecting and managing cubes.

- **MicroStrategy:** Catalog

Each catalog includes one database and any associated cubes. Catalogs in MicroStrategy are represented as a folder.

XMLA schema

TM1 --->	XMLA --->	MicroStrategy
Not supported	Schema	Not supported

- **TM1:** Not supported

- **XMLA:** Schema

A schema in XMLA provides a grouping mechanism not supported by TM1 or MicroStrategy.

- **MicroStrategy:** Not supported

Cube

TM1 --->	XMLA --->	MicroStrategy
Cube	Cube	MDX cube

- **TM1:** Cube
- **XMLA:** Cube
- **MicroStrategy:** MDX cube

A MicroStrategy MDX cube is an object that is used to map the levels of a TM1 cube into the MicroStrategy environment. MDX cubes are treated similarly to tables in the MicroStrategy metadata. In the same way that a metadata table maps the physical columns of a relational table to attributes and metrics, a MicroStrategy MDX cube maps the physical columns of a TM1 cube to attributes and metrics. The MDX cube represents a TM1 cube.

Dimension

TM1 --->	XMLA --->	MicroStrategy
Dimension	Dimension	Dimension

- **TM1:** Dimension

In TM1, a dimension is defined from one or more data source tables. All columns in the tables are eligible to become attributes of the dimension. Each attribute is used to define a hierarchy within the dimension and multi-level hierarchies can be defined as well.

A TM1 dimension appears as a MicroStrategy dimension for MicroStrategy users. Each dimension represented in MicroStrategy can have one or more MicroStrategy hierarchies.

- **XMLA:** Dimension

A dimension in XMLA defines a logical category of analysis. For example, Time and Geography are dimensions along which you can slice data.

Measures (metrics) are stored in a special measure dimension. In this way, measures are simply one more dimension of a cube.

Measures in XMLA are the members of the Measures dimension in TM1. These can be columns in the data source table or calculated members represented by formulas with associated aggregation rules.

If the measures dimension in your TM1 MDX cube source uses a different name other than [Measures] you can specify the name in your TM1 MDX cube source using the VLDB properties Name of Measure Dimension and MDX Remember Measure Dimension Name. For

information on how to use these VLDB properties to specify the measure dimension name, refer to the *Supplemental Reference for System Administration*.

- **MicroStrategy:** Dimension

A dimension object in MicroStrategy is similar to an XMLA dimension. It is used to group attributes and to relate them to each other by defining parent-child relationships.

Hierarchy

TM1 --->	XMLA --->	MicroStrategy
Hierarchy	Hierarchy	Hierarchy

- **TM1:** Hierarchy

TM1 allows the definition of one or more hierarchies within a dimension as collections of attributes. The attributes become levels of the hierarchy.

- **XMLA:** Hierarchy

- **MicroStrategy:** Hierarchy

Hierarchies are used to group attributes (levels) together and define the relationships between these attributes. MicroStrategy uses hierarchy objects to represent both dimensions and hierarchies from XMLA.

Level

TM1 --->	XMLA --->	MicroStrategy
Level	Level	Attribute

- **TM1:** Level

Each attribute in a hierarchy becomes a level.

- **XMLA:** Level

- **MicroStrategy:** Attribute (ID/DESC)

MicroStrategy attributes map to XMLA levels. Each XMLA level generates two physical columns and forms in MicroStrategy: ID and DESC.

Element

TM1 --->	XMLA --->	MicroStrategy
Element	Member	Attribute element

- **TM1:** Element
- **XMLA:** Member
- **MicroStrategy:** Attribute element

Element values come from a cube. For example, 2009 and 2010 are elements of the Year attribute.

Attribute

TM1 --->	XMLA --->	MicroStrategy
Attribute	Property	Attribute form

- **TM1:** Attribute
- **XMLA:** Property
- **MicroStrategy:** Attribute form

Attribute forms provide additional information about a given attribute. For example, the Customer attribute may have the forms First Name and Last Name.

CONNECTING TO MDX CUBE SOURCES

Introduction

While MicroStrategy handles most of the conversion processes necessary to integrate with MDX cube source data, you must establish the connection between MicroStrategy and your MDX cube source.

This chapter provides the steps and requirements to connect MicroStrategy to MDX cube sources. This chapter assumes that you are familiar with creating database instances in MicroStrategy to establish connections to data sources. MicroStrategy's connection to SAP BW uses SAP's OLAP Business Application Programming Interface (BAPI). MicroStrategy's connection to Analysis Services 2005/2008/2012/2014, Oracle Essbase, and TM1 uses XML for Analysis (XMLA).

These configurations require modifications to projects, *project sources*, database instances, and other objects in MicroStrategy. A project source defines a connection to the metadata database and is used by various MicroStrategy components to access projects.

This chapter covers the following information related to connecting to MDX cube sources:

- [Connecting to SAP BW servers, page 36](#)
- [Connecting to Oracle Essbase servers, page 47](#)
- [Connecting to Analysis Services servers, page 51](#)
- [Connecting to TM1 servers, page 56](#)
- [Configuring MDX cube database instances in projects, page 59](#)—After you have connected to an MDX cube source by creating an MDX cube database instance, you can configure your MDX cube database instance in a project. For example, you can choose to load the schema for imported MDX cubes either when Intelligence Server starts or during MDX cube report execution. For information on setting MDX cube schema loading options for an MDX cube source database instance, see [MDX cube schema loading, page 61](#).
- [Authentication, page 67](#)

After you have connected to your MDX cube sources, you can begin integrating MDX cube data into MicroStrategy as covered in [Chapter 3, Integrating MDX Cubes into MicroStrategy](#).

For troubleshooting and diagnostics logging routines related to MDX cube sources, see the *Troubleshooting* chapter of the *MicroStrategy System Administration Guide*.

Connecting to SAP BW servers

Before creating any reports using the SAP BW data, you need to establish a connection between MicroStrategy and the SAP BW system.

MicroStrategy certifies connecting to SAP BW 3.1, 3.5, 7.0, 7.3, and 7.4.

With the SAP BW Certification on MicroStrategy, MicroStrategy Intelligence Server is certified to connect to and execute reports against SAP BW cubes. As SAP's proprietary API for accessing SAP BW data and functionality, the OLAP BAPI provides an open interface through which Intelligence Server can access the SAP BW data.

MicroStrategy Web is certified to run on SAP Web Application Server, and MicroStrategy Web and SDK are certified to run with SAP Enterprise Portal through iView Packages.

Prerequisites and checklist

Connecting MicroStrategy to SAP BW servers on Windows, UNIX, and Linux follows the workflow listed below, which also contains important prerequisite information:

- Install the SAP Java Connector files. You need a valid login for the SAP Service Marketplace to download the SAP Java Connector files. If you are using MicroStrategy Intelligence Server on UNIX or Linux, you also need Write privileges to the `/install/jar` installation directory on your UNIX or Linux machine.

MicroStrategy supports the use of version 3.x. With your Intelligence Server hosted on a 64-bit machine, you must download and install the 64-bit version of the 3.x Java Connector.

For information on what SAP Java Connector you need to support your SAP BW system, refer to your third-party SAP BW documentation.

- Create a database instance in MicroStrategy that connects to your SAP BW server. The procedure below employs MicroStrategy Developer (available only on Windows) to create a database instance. Creating a database instance and its components described below requires a MicroStrategy login with administrative privileges:
 - Create a database connection with all valid SAP connection information, as covered in the procedures below.
 - Create a database login used to connect to your SAP BW server. MicroStrategy requires that the SAP BW user login used to connect to your SAP BW server has remote function call (RFC) authorization to the groups `RFC1`, `RSOB`, `SDIFRUNTIME`, and `SYST`. If necessary, contact your SAP administrator to grant your SAP BW user login RFC authorization to these groups.

Additionally, the following query and InfoCube access is required:

- For each query to allow access to, you must have `EXECUTE` access for the `S_RS_COMP` authorization object.
- For query object types, you must have `EXECUTE` access for the `S_RS_ICUBE` authorization object.

- For each InfoCube to allow access to, you must have activity `DISPLAY` access and InfoCube subobject `DATA` access for the `S_RS_ICUBE` authorization object.

The following sections provide detailed steps for the workflow listed above to connect MicroStrategy to SAP BW servers in the following environments:

- [Connecting to SAP BW servers on Windows](#)
- [Connecting to SAP BW servers on UNIX and Linux](#)

Connecting to SAP BW servers on Windows

The procedure below provides the necessary steps to connect to SAP BW servers on Windows.



Important note from SAP:

As part of the SAP Java Connector installation, you are required to install the new Visual Studio .NET C/C++ run-time libraries on Windows platforms. See SAP Note 684106 for details on how to install them.

Prerequisites

Before attempting to connect to SAP BW servers, review the prerequisites listed in [Prerequisites and checklist, page 37](#).

To connect to SAP BW servers on Windows

- 1 Open the **SAP Service Marketplace** and download the **SAP Java Connector**. You can use the following URL to download the Java Connector:

https://service.sap.com/~form/sapnet?_SHORTKEY=01100035870000463649

- 2 Install the SAP Java Connector.
- 3 Place the following SAP Java Connector files in the MicroStrategy Common Files folder:
 - `Sapjco3.jar`

▪ Sapjco3.dll

The default location of this folder is `C:\Program files\Common files\MicroStrategy`. This folder must be referenced in the system's Path environment variable.



To verify that the directory is included in the value for the Path variable, view the Path environment by right-clicking on **My Computer** and selecting **Properties**. On the **Advanced** tab, select **Environment Variables**. In the list of **System Variables**, locate Path. Check to see whether the directory is listed in the environment variable.

To create a database instance for your SAP BW connection

- 4 In MicroStrategy Developer from the Folder List, open your project source and expand **Administration**.
- 5 Expand **Configuration Managers**, and select **Database Instance**.
- 6 From the **File** menu, select **New**, and then **Database Instance**. The Database Instance Editor opens.
- 7 In the **Database instance name** text field, type a descriptive name for the database instance.
- 8 From the **Database connection type** drop-down list, select **SAP BW 3.x** or **SAP BW 7.x**, depending on your version of SAP BW.

If you connect to SAP BW 7.01 SP02, you can create MDX cube reports that include more than one million cells. MicroStrategy also recommends using the 3.x version of the SAP Java Connector to fully support this scalability.

- 9 From the **Advanced** tab, select the version of the SAP Java Connector you downloaded at the beginning of these steps. MicroStrategy supports the use of the 3.x version of the SAP Java Connector.

The 2.x version of the SAP Java Connector, was supported in pre-10 versions of MicroStrategy. You can update your database instances that previously used version 2.x to use version 3.x.

To create a database connection

10 From the **General tab**, click **New** to open the Database Connections dialog box and create a database connection. Provide the following information as required, which can be found from your SAP Logon:

- **Database connection name:** This is the name to distinguish the database connection from database connections for other database instances.
- **Logon method:** Determines the type of SAP BW system you are connecting to. You have the following options:
 - **Application server:** This option lets you connect to a specific SAP BW application server. You must provide the following information to connect to an SAP BW application server:
 - **Application Server:** This is the name or IP address of the SAP BW Server.
 - **SAP Router String:** This is the SAP router string, which is only required if you use an SAP Router.
 - **System Number:** This is the system number from the SAP BW system.
 - **Client Number:** This is the client number from the SAP BW system.
 - **Language:** This is the language code provided by your SAP administrator. For example, EN is the language code for English.
 - **Message server:** This option lets you connect to an SAP BW message server, which acts as a load balancing mechanism for a cluster of SAP BW application servers. Once connected to the SAP BW message server, the SAP BW system can its load balancing capabilities to connect to an SAP BW application server. You must provide the following information to connect to an SAP BW message server:
 - **Message Server:** This is the name or IP address of the SAP BW Server.
 - **SAP Router String:** This is the SAP router string, which is only required if you use an SAP Router.
 - **System ID:** This is the system ID from the SAP BW system.

- **Logon group:** This is the SAP BW logon group that determines what SAP BW application servers in the cluster can be accessed.
- **Client Number:** This is the client number from the SAP BW system.
- **Language:** This is the language code provided by your SAP administrator. For example, EN is the language code for English.

To create a database login



Be aware of the following:

- MicroStrategy requires that the SAP BW user login used to connect to your SAP BW server has remote function call (RFC) authorization to the `SDIFRUNITIME` group. If necessary, contact your SAP administrator to grant your SAP BW user login RFC authorization to the `SDIFRUNITIME` group.
- You can configure your SAP BW environment to allow users single-sign on access to MicroStrategy through the use of integrated authentication. If you configure this type of authentication, as described in the *MicroStrategy System Administration Guide*, the database login listed below is required but is not used for authentication.

11 Click **New** to create a database login with the user and password credentials used to connect to SAP BW. The Database Logins dialog box opens.

12 Enter a name for the database login object, login ID, and password.

To save your changes and create the database instance

13 Click **OK** to save your changes and return to the Database Connections dialog box.

14 From the **Default database login name** area, select the database login you created and click **OK**. You are returned to the Database Instances dialog box.

15 From the **Database connection** area, select the database connection you created and click **OK** to create the database instance.

After you have connected a database instance to your SAP BW server, you can begin accessing your SAP BW data from a MicroStrategy project. For

information on configuring the MDX cube database instance in a project, see [Configuring MDX cube database instances in projects, page 59](#).

Connecting to SAP BW servers on UNIX and Linux

Take the following steps to connect MicroStrategy to SAP BW servers on UNIX and Linux.

Prerequisites

Before attempting to connect to SAP BW servers, review the prerequisites listed in [Prerequisites and checklist, page 37](#).

To connect to SAP BW servers on UNIX/Linux

- 1 Open the **SAP Service Marketplace** and download the **SAP Java Connector**. You can use the following URL to download the Java Connector:

https://service.sap.com/~form/sapnet?_SHORTKEY=01100035870000463649
- 2 Select the zip file compatible with your environment and unzip it. For example, use the command `gunzip [file name]` or `gzip [file name]`.
- 3 Search for the files listed in the following table, copy them onto your machine, and create a new directory for them. For example, `/var/opt/MicroStrategy/SAP`.

AIX	SUN	HP-UX	Linux
libsapjco3.so	libsapjco3.so	libsapjco3.so	libsapjco3.so
sapjco3.jar	sapjco3.jar	sapjco3.jar	sapjco3.jar

- 4 Add `sapjco.jar` or `sapjco3.jar` to the MicroStrategy installation directory `[HOME_PATH]/install/jar`.

5 Edit the `SAP.sh` file in the MicroStrategy installation directory
[`INSTALL_PATH`]/`env/SAP.sh` by performing the following steps:

- a Add the Write and Execute privileges to this file. The default is Read Only. You can type the command "`chmod+wx SAP.sh`" in the UNIX/Linux console.
- b Open the `SAP.sh` file and enter the information for `MSTR_SAP_LIBRARY_PATH=' '`. This information indicates where the server needs to point to use the downloaded files.

In the examples below, you must replace `DIRECTORY_PATH` with the path to the directory where you installed the SAP Java Connector files. The default directory path is `/var/opt/MicroStrategy/SAP`.

For example, for **AIX**:

```
#
# set up the environment for SAP
#
MSTR_SAP_LIBRARY_PATH='DIRECTORY_PATH'

if [ "${MSTR_SAP_LIBRARY_PATH}"
!= 'DIRECTORY_PATH' ]; then
mstr_append_path LIBPATH
"${MSTR_SAP_LIBRARY_PATH:?}"
export LIBPATH
fi
```

For example, for **Solaris**:

```
#
# set up the environment for SAP
#
MSTR_SAP_LIBRARY_PATH='DIRECTORY_PATH'

if [ "${MSTR_SAP_LIBRARY_PATH}" != 'DIRECTORY_PATH'
]; then
mstr_append_path LD_LIBRARY_PATH
"${MSTR_SAP_LIBRARY_PATH:?}"
export LD_LIBRARY_PATH
fi
```

For example, for **HP-UX**:

```
#
# set up the environment for SAP
#
MSTR_SAP_LIBRARY_PATH='DIRECTORY_PATH'
```

```

if [ "${MSTR_SAP_LIBRARY_PATH}" != 'DIRECTORY_PATH'
]; then
mstr_append_path LD_LIBRARY_PATH
"${MSTR_SAP_LIBRARY_PATH:?}"
export LD_LIBRARY_PATH
fi

```

For example, for **Linux**:

```

#
# set up the environment for SAP
#
MSTR_SAP_LIBRARY_PATH='DIRECTORY_PATH'

if [ "${MSTR_SAP_LIBRARY_PATH}" != 'DIRECTORY_PATH'
]; then
mstr_append_path LD_LIBRARY_PATH
"${MSTR_SAP_LIBRARY_PATH:?}"
export LD_LIBRARY_PATH
fi

```

c Save the file.

6 Restart the server to apply the latest updates.

To create a database instance for your SAP BW connection

- 7 On a Windows machine, in MicroStrategy Developer from the Folder List, open your project source and expand **Administration**.
- 8 Expand **Configuration Managers**, and select **Database Instance**.
- 9 From the **File** menu, select **New**, and then **Database Instance**. The Database Instance Editor opens.
- 10 In the **Database instance name** field, type a descriptive name for the database instance.
- 11 From the **Database connection type** drop-down list, select **SAP BW 3.x** or **SAP BW 7.x**, depending on your version of SAP BW.

If you connect to SAP BW 7.01 SP02, you can create MDX cube reports that include more than one million cells. MicroStrategy also recommends using the 3.x version of the SAP Java Connector to fully support this scalability.

- 12** From the **Advanced** tab, select the version of the SAP Java Connector you downloaded at the beginning of these steps. MicroStrategy supports the use of the 3.x version of the SAP Java Connector.

The 2.x version of the SAP Java Connector, was supported in pre-10 versions of MicroStrategy. You can update your database instances that previously used version 2.x to use version 3.x.

To create a database connection

- 13** From the **General** tab, click **New** to open the Database Connections dialog box and create a database connection. Provide the following information as required, which can be found from your SAP Logon:
- **Database connection name:** This is the name to distinguish the database connection from database connections for other database instances.
 - **Logon method:** Determines the type of SAP BW system you are connecting to. You have the following options:
 - **Application server:** This option lets you connect to a specific SAP BW application server. You must provide the following information to connect to an SAP BW application server:
 - **Application Server:** This is the name or IP address of the SAP BW Server.
 - **SAP Router String:** This is the SAP router string, which is only required if you use an SAP Router.
 - **System Number:** This is the system number from the SAP BW system.
 - **Client Number:** This is the client number from the SAP BW system.
 - **Language:** This is the language code provided by your SAP administrator. For example, EN is the language code for English.
 - **Message server:** This option lets you connect to an SAP BW message server, which acts as a load balancing mechanism for a cluster of SAP BW application servers. Once connected to the SAP BW message server, the SAP BW system can its load balancing capabilities to connect to an SAP BW application server. You must provide the following information to connect to an SAP BW message server:

- **Message Server:** This is the name or IP address of the SAP BW Server.
- **SAP Router String:** This is the SAP router string, which is only required if you use an SAP Router.
- **System ID:** This is the system ID from the SAP BW system.
- **Logon group:** This is the SAP BW logon group that determines what SAP BW application servers in the cluster can be accessed.
- **Client Number:** This is the client number from the SAP BW system.
- **Language:** This is the language code provided by your SAP administrator. For example, EN is the language code for English.

To create a database login



Be aware of the following:

- MicroStrategy requires that the SAP BW user login used to connect to your SAP BW system has remote function call (RFC) authorization to the `SDIFRUNTIME` group. If necessary, contact your SAP administrator to grant your SAP BW user login RFC authorization to the `SDIFRUNTIME` group.
- You can configure your SAP BW environment to allow users single-sign on access to MicroStrategy through the use of integrated authentication. If you configure this type of authentication, as described in the *MicroStrategy System Administration Guide*, the database login listed below is required but is not used for authentication.

14 Click **New** to create a database login with the user and password credentials used to connect to SAP BW. The Database Logins dialog box opens.

15 Enter a name for the database login object, login ID, and password.

To save your changes and create the database instance

16 Click **OK** to save your changes and return to the Database Connections dialog box.

17 From the **Default database login name** area, select the database login you created and click **OK**. You are returned to the Database Instances dialog box.

18 From the **Database connection** area, select the database connection you created and click **OK** to create the database instance.

After you have connected a database instance to your SAP BW server, you can begin accessing your SAP BW data from a MicroStrategy project. For information on configuring the MDX cube database instance in a project, see [Configuring MDX cube database instances in projects, page 59](#).

Connecting to Oracle Essbase servers

Before creating any reports using the Oracle Essbase data, you need to establish a connection in MicroStrategy to the Oracle Essbase 9.3.1 or 11 servers. Connecting MicroStrategy to Oracle Essbase servers on Windows, UNIX, and Linux follows the workflow listed below, which also contains important prerequisite information:

- Configure the MDX Cube Provider so that MicroStrategy can successfully connect to your Oracle Essbase servers.
- Create a database instance in MicroStrategy that connects to your Oracle Essbase server. The procedure below employs MicroStrategy Developer (available only on Windows) to create a database instance. Creating a database instance and its components described below requires a MicroStrategy login with administrative privileges:
 - Create a database connection with all valid Oracle Essbase connection information.
 - Create a database login used to connect to your Oracle Essbase server.

The following sections provide the detailed steps for the workflow listed above to connect MicroStrategy to Oracle Essbase servers in Windows, UNIX, and Linux environments:

- [Configuring the MDX Cube Provider, page 48](#)
- [Creating a database instance, page 49](#)

Configuring the MDX Cube Provider

Make sure the MDX Cube Provider is correctly deployed and security settings are configured correctly, according to the following guidelines:

- Configure the MicroStrategy MDX Cube Provider, which is included with MicroStrategy.

To support the use of the MicroStrategy MDX Cube Provider you must do the following:

- Install the MicroStrategy MDX Cube Provider. For information on how to install this component, which is provided with MicroStrategy, see the *Installation and Configuration Guide*.
- Install the Essbase client, including the Essbase C API, on the machine where you installed MicroStrategy and the MicroStrategy MDX Cube Provider. The version that you install depends on your operating system's architecture:
 - If you install on a 32-bit operating system, install the 32-bit Essbase client.
 - If you install on a 64-bit operating system, install the 64-bit Essbase client.

For information on how to install the Essbase client libraries, refer to your third-party Oracle documentation.



Using Essbase version 11.1.2.1 can cause XML corruption issues in query results. To avoid this issue, you must upgrade Essbase Analytical Provider Services to version 11.1.2.2 to support integration with the MicroStrategy MDX Cube Provider.

- After installing the Essbase client and Essbase C API, you must create various environment variables. MicroStrategy provides scripts to create the required environment variables. Using a command line interface, navigate to the MicroStrategy Common Files folder and execute one of the following scripts:

- For 64-bit Essbase clients:

```
EssbaseConnector_SetEnv_64_64.cmd
```

- For 32-bit Essbase clients:

```
EssbaseConnector_SetEnv_32_32.cmd
```

- In the installation location for the MicroStrategy MDX Cube Provider, you must modify the `Datasources.xml` file. It is recommended that you make a backup copy before modifying this `.xml` file.
 - In the `Datasources.xml` file, locate the `<AuthenticationMode></AuthenticationMode>` section. Within this section, type the value `EssbaseBasic`.
 - In the `Datasources.xml` file, locate the `<ProgramID></ProgramID>` section. Within this section, type the value `ESSBASE`.



The `Datasources.xml` file includes an example definition for the `<DataSources></DataSources>` section with default values for an Essbase connection. You can use this section as a template for the `<DataSources>` definition of your Oracle Essbase connection.

- Install Oracle Essbase Provider Services on the application server machine to verify that access is available to the Oracle Essbase server via the service console. For information on installation procedures, see your third-party documentation (<http://www.oracle.com/technology/products/bi/essbase/index.html>).
- By default, access to the application uses standard authentication. By setting up a MicroStrategy connection using your Oracle Essbase login and password, you can use your Oracle Essbase login and password through Provider Services.

You can verify whether the configuration is working by connecting to the provider URL from your browser. You should receive a confirmation from the provider that includes a display of currently configured properties.



You can perform a test of the connection to your MDX cube servers, separate from any MicroStrategy dependencies, using the XMLA Connectivity Test Tool provided with your MicroStrategy installation. In Windows environments, this tool can be used by running the `XMLATest.exe` file included in the MicroStrategy common files. In UNIX environments, this tool can be used by running the command `mstrxmlatest` from the `/bin` directory within your MicroStrategy home path.

Creating a database instance

Perform the following steps to connect MicroStrategy to Oracle Essbase 9.3.1 or 11 servers.

To create a database instance for Oracle Essbase 9.3.1 or 11

- 1 From the MicroStrategy Developer Folder List, connect to your project source.
- 2 From the Folder List, expand **Administration**, expand **Configuration Managers**, and select **Database Instance**.
- 3 From the **File** menu, select **New**, and then **Database Instance**. The Database Instance Editor opens.
- 4 In the **Database instance name** field, type a descriptive name for the database instance.
- 5 From the **Database connection type** drop-down list, select one of the following depending on the version of Essbase you are connecting to:
 - For Essbase 9.3.1, select **Oracle Essbase 9.3/11.x MDX Cube Provider**.
 - For Essbase 11, select **Oracle Essbase 9.3/11.x MDX Cube Provider**.

To create database connection parameters

- 6 Click **New** to open the Database Connections dialog box and create a database connection. Provide the following information as required:
 - **Database connection name:** This is the name to distinguish the database connection from database connections for other database instances.
 - **URL:** This is the URL of the provider that was configured for HTTP access. For example: `http://fully-qualified-machinename/MicroStrategyMDX/MicroStrategyMDX.asmx`



The *fully-qualified-machinename* is usually of the form `machine.domain.company.com`. You can also use the IP address as the *fully-qualified-machinename*. For Oracle Essbase the URL is usually case sensitive.

- **DSI:** The DataSourceInfo (DSI) value is of the form:
`Provider=Essbase;Data Source=<machine name>`
- **Catalog:** The Essbase Catalog value is the Oracle Essbase application containing the database to connect to with MicroStrategy. Use the

Essbase Administration Console to view the applications and databases available on the server.

To create a database login

- 7 Click **New** to create a database login with the user and password credentials used to connect to Oracle Essbase. The Database Logins dialog box opens.
- 8 Enter a name for the database login object, login ID, and password. The login ID and password must be a valid Oracle Essbase user and password.

To save your changes and create the database instance

- 9 Click **OK** to save your changes and return to the Database Connections dialog box.
- 10 From the **Default database login name** area, select the database login you created and click **OK**. You are returned to the Database Instances dialog box.
- 11 From the **Database connection** area, select the database connection you created and click **OK** to create the database instance.

After you have connected a database instance to your Oracle Essbase server, you can begin accessing your Oracle Essbase data from a MicroStrategy project. For information on configuring the MDX cube database instance in a project, see [Configuring MDX cube database instances in projects, page 59](#).

Connecting to Analysis Services servers

Before creating any reports using the Analysis Services 2005/2008/2012/2014 data, you need to establish a connection to the Analysis Services 2005/2008/2012/2014 servers. Connecting MicroStrategy to Analysis Services 2005/2008/2012/2014 servers on Windows follows the workflow listed below, which also contains important prerequisite information:

- Configure Analysis Services 2005/2008/2012/2014 and the XMLA provider so that MicroStrategy can successfully connect to your Analysis Services 2005/2008/2012/2014 servers.
- Create a database instance in MicroStrategy that connects to your Analysis Services 2005/2008/2012/2014 server. The procedure below employs MicroStrategy Developer (available only on Windows) to create

a database instance. Creating a database instance and its components requires a MicroStrategy login with administrative privileges:


- Create a database connection with all valid Analysis Services 2005/2008/2012/2014 connection information.
- Create a database login used to connect to your Analysis Services 2005/2008/2012/2014 server.

The following sections provide the detailed steps for the workflow listed above to connect MicroStrategy to Analysis Services 2005/2008/2012 servers in a Windows environment:

- [Configuring Analysis Services and the MDX Cube provider, page 52](#)
- [Creating a database instance, page 54](#)

Configuring Analysis Services and the MDX Cube provider

Make sure Analysis Services 2005/2008/2012/2014 is correctly deployed and security settings are configured correctly, according to the following guidelines:

-  Information for correctly installing Analysis Services 2005/2008/2012/2014 and any other Microsoft products can be found in your Microsoft documentation. The links to related Microsoft documentation provided below are valid as of the release of this guide.
- Install Microsoft Analysis Services 2005/2008/2012. For steps to install Analysis Services 2005/2008/2012, see <http://msdn2.microsoft.com/en-us/library/ms144288.aspx>.
- Configure the MicroStrategy MDX Cube Provider, which is included with MicroStrategy. To support the use of the MicroStrategy MDX Cube Provider you must do the following:
 - Install the MicroStrategy MDX Cube Provider. For information on how to install this component, which is provided with MicroStrategy, see the *Installation and Configuration Guide*.
 - Install the Microsoft Analysis Services OLE DB Provider on the machine where you installed MicroStrategy and the MicroStrategy MDX Cube Provider. The version that you install depends on your operating system's architecture:
 - If you install on a 32-bit operating system, install the 32-bit Microsoft Analysis Services OLE DB Provider. MicroStrategy

supports the Microsoft Analysis Services OLE DB Provider 2008 and newer versions.

- If you install on a 64-bit operating system, install the 64-bit Microsoft Analysis Services OLE DB Provider.



If you are connecting to Microsoft Analysis Services 2014, you must install the 2012 SP2 version of the Microsoft Analysis Services OLE DB Provider.

For information on how to install the Microsoft Analysis Services OLE DB Provider, refer to your third-party Microsoft Analysis Services documentation.

- In the installation location for the MicroStrategy MDX Cube Provider, you must modify the `Datasources.xml` file. It is recommended that you make a backup copy before modifying this `.xml` file.
 - In the `Datasources.xml` file, locate the `<AuthenticationMode></AuthenticationMode>` section. Within this section, type the authentication mode used for the virtual directory running on Microsoft Internet Information Services. You can use either Basic (`MSOLAPBasic`) or Integrated (`MSOLAPIntegrated`) authentication. For information on these authentication options, refer to your third-party Microsoft documentation.
 - In the `Datasources.xml` file, locate the `<ProgramID></ProgramID>` section. Within this section, type the value `MSOLAP`.



The `Datasources.xml` file includes an example definition for the `<DataSources></DataSources>` section with default values for a Microsoft Analysis Services connection. You can use this section as a template for the `<DataSources>` definition of your Microsoft Analysis Services connection.

- Create one or more Microsoft Analysis Services role definitions along with mapped NT user logins. If anonymous access is enabled in the XMLA virtual directory, the Windows user associated with anonymous access must have membership to one of the security roles defined in Analysis Services.
- If IIS and the Analysis Services host are not on the same machine, you must either configure Kerberos authentication protocol for credential delegation or define an anonymous user on the IIS XMLA provider virtual directory. The Windows user associated with anonymous access

needs to have membership in one of the security roles defined in Analysis Services.



You can perform a test of the connection to your MDX cube servers, separate from any MicroStrategy dependencies, using the XMLA Connectivity Test Tool provided with your MicroStrategy installation. In Windows environments, this tool can be used by running the `XMLATest.exe` file included in the MicroStrategy common files. In UNIX environments, this tool can be used by running the command `mstrxmlatest` from the `/bin` directory within your MicroStrategy home path.

Creating a database instance

Perform the following steps to connect to Microsoft Analysis Services 2005/2008/2012/2014 servers.

To create a database instance for Analysis Services 2005/2008/2012

- 1 In MicroStrategy Developer from the Folder List, open your project source and expand **Administration**.
- 2 Expand **Configuration Managers**, and select **Database Instance**.
- 3 From the **File** menu, select **New**, and then **Database Instance**. The Database Instance Editor opens.
- 4 In the **Database instance name** text field, type a descriptive name for the database instance.
- 5 From the **Database connection type** drop-down list, select one of the following:
 - **Microsoft Analysis Services 2005/2008 (MDX Cube Provider)** if you are connecting to Analysis Services 2005 or 2008 with the MicroStrategy MDX Cube Provider.
 - **Microsoft Analysis Services 2012 and 2014 (MDX Cube Provider)** if you are connecting to Analysis Services 2012, Analysis Services 2012 in Tabular Mode, or Analysis Services 2014 with the MicroStrategy MDX Cube Provider.

To create a database connection

- 6 Click **New** to open the Database Connections dialog box and create a database connection. Provide the following information as required:
 - **Database connection name:** This is the name to distinguish the database connection from database connections for other database instances.
 - **URL:** This is the URL of the XMLA Provider that was configured for HTTP access. For example: `http://fully-qualified-machinename/MicroStrategyMDX/MicroStrategyMDX.asmx`

The *fully-qualified-machinename* is usually of the form `machine.domain.company.com`. You can also use the IP address as the *fully-qualified-machinename*. The URL is not case sensitive.
 - **DSI:** Type the fully qualified machine name or the IP address of the machine where the Microsoft Analysis Services server is running.
 - **Catalog:** Use Microsoft's SQL Server Management Studio to view the Analysis Server which contains the cubes to work with in MicroStrategy. The database that contains the cube becomes the catalog.

To create a database login

- 7 Click **New** to create a database login with the user and password credentials used to connect MicroStrategy to Analysis Services. The Database Logins dialog box opens.
- 8 Enter a name for the database login object, a login ID, and a password. For information on allowing Windows users' login credentials to be used to execute against MDX cubes imported from Analysis Services into MicroStrategy, see [Single sign-on to Microsoft Analysis Services, page 68](#).

To save your changes

- 9 Click **OK** to save your changes and return to the Database Connections dialog box.
- 10 From the **Default database login name** area, select the database login you created and click **OK**. You are returned to the Database Instances dialog box.

- 11 From the **Database connection** area, select the database connection you created and click **OK** to create the database instance.

After you have connected a database instance to your Analysis Services 2005/2008/2012 server, you can begin accessing your Analysis Services 2005/2008/2012 data from a MicroStrategy project. For information on configuring the MDX cube database instance in a project, see [Configuring MDX cube database instances in projects, page 59](#).

Connecting to TM1 servers

Before creating any reports using the TM1 data, you need to establish a connection to the TM1 servers. Connecting MicroStrategy to TM1 servers on Windows, UNIX, and Linux follows the workflow listed below, which also contains important prerequisite information:

- Install the MicroStrategy MDX Cube Provider. For information on how to install this component, which is provided with MicroStrategy, see the *Installation and Configuration Guide*.
- Install the TM1 OLEDB Provider on the machine where you installed MicroStrategy and the MicroStrategy MDX Cube Provider. The version that you install depends on your operating system's architecture:
 - If you install on a 32-bit operating system, install the 32-bit TM1 OLEDB Provider.
 - If you install on a 64-bit operating system, install the 64-bit TM1 OLEDB Provider.

For information on how to install the TM1 OLEDB Provider, see your third-party TM1 documentation.

- In the installation location for the MicroStrategy MDX Cube Provider, you must modify the `Datasources.xml` file. It is recommended that you make a backup copy before modifying this `.xml` file.
 - In the `Datasources.xml` file, locate the `<AuthenticationMode></AuthenticationMode>` section. Within this section, type the authentication mode used for the virtual directory running on Microsoft Internet Information Services. You can use either Basic or Integrated authentication. For information on these authentication options, refer to your third-party Microsoft documentation.

- In the `Datasources.xml` file, locate the `<ProgramID></ProgramID>` section. Within this section, type the value `TM1OLAP`.



The `Datasources.xml` file includes an example definition for the `<DataSources></DataSources>` section with default values for a TM1 connection. You can use this section as a template for the `<DataSources>` definition of your TM1 connection.

- Create a database instance in MicroStrategy that connects to your TM1 server. The procedure below employs MicroStrategy Developer to create a database instance. Creating a database instance and its components requires a MicroStrategy login with administrative privileges:
 - Create a database connection with all valid TM1 connection information, including the URL of the MicroStrategy MDX Cube Provider and the TM1 server configuration, as described in the procedure below.
 - Create a database login used to connect to your TM1 server.

Creating a database instance

Perform the following steps to connect to TM1 servers.

To create a database instance for TM1

- 1 In MicroStrategy Developer from the Folder List, open your project source and expand **Administration**.
- 2 Expand **Configuration Managers**, and select **Database Instance**
- 3 From the **File** menu, select **New**, and then **Database Instance**. The Database Instance Editor opens.
- 4 In the **Database instance name** field, type a descriptive name for the database instance.
- 5 From the **Database connection type** drop-down list, select **IBM Cognos TM1**.

To create a database connection

- 6 Click **New** to create a database connection. The Database Connections dialog box opens. Provide the following information as required:
 - **Database connection name:** This name distinguishes the database connection from database connections for other database instances.
 - **URL:** The URL of the MicroStrategy MDX Cube Provider that was configured for HTTP Access. This connector is installed as part of a MicroStrategy installation to support connections to TM1. The default URL is in the following format:

```
http://fully-qualified-machinename/  
MicroStrategyMDX/MicroStrategyMDX.asmx
```



The *fully-qualified-machinename* is usually of the form *machine.domain.company.com*. You can also use the IP address as the *fully-qualified-machinename*.

- **DSI:** The fully qualified machine name of the machine where the TM1 Admin Server is running.
- **Catalog:** The catalog represents the name of the TM1 server configuration, which is defined in the *TM1S.cfg* file. For more information, see your third-party TM1 documentation.

To create a database login

- 7 Click **New** to create a database login with the user and password credentials used to connect MicroStrategy to TM1. The Database Logins dialog box opens.
- 8 Enter a name for the database login object, login ID, and password.

To save your changes

- 9 Click **OK** to save your changes and return to the Database Connections dialog box.
- 10 From the **Default database login name** area, select the database login that you created and click **OK**. You are returned to the Database Instances dialog box.
- 11 From the **Database connection** area, select the database connection that you created and click **OK** to create the database instance.

After you have connected a database instance to your TM1 server, you can begin accessing your TM1 data from a MicroStrategy project. For information on configuring the MDX cube database instance in a project, see [Configuring MDX cube database instances in projects](#) below.

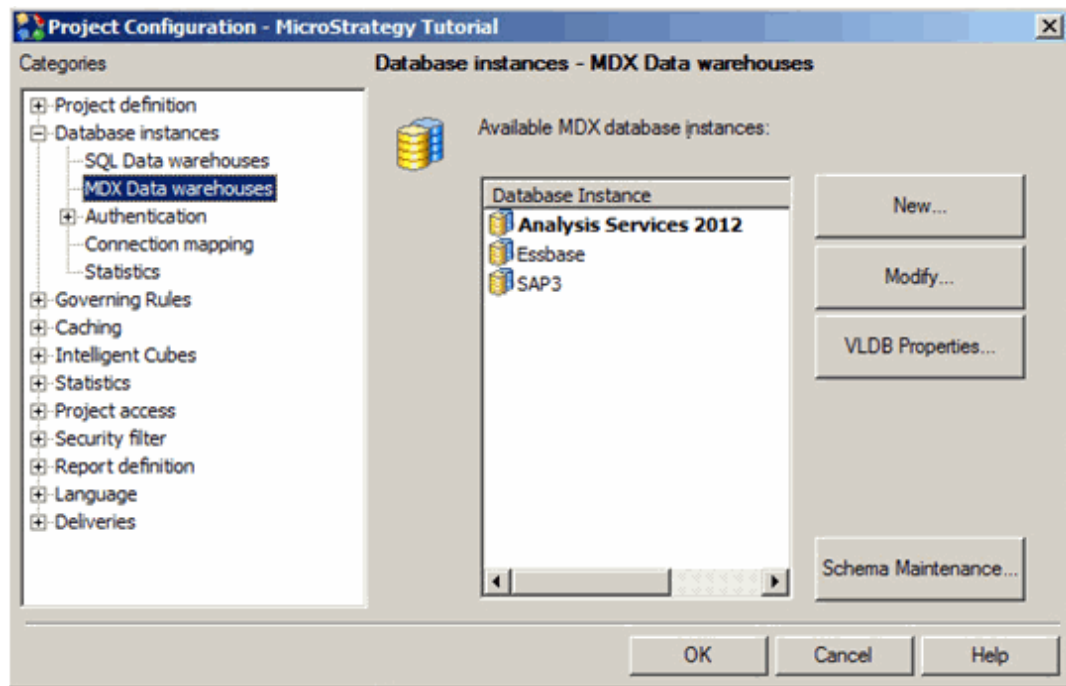
Configuring MDX cube database instances in projects

When a project designer creates a project in MicroStrategy, a database instance is assigned to that project. A project can only have one warehouse database instance, which is the database instance that the MicroStrategy Warehouse Catalog uses to access the warehouse tables available for the project. The warehouse tables for a project determine the set of relational data available to be analyzed in the project.

Additional database instances connected to MDX cube sources can be included in a project along with the warehouse database instance. This enables you to connect to and report on data from your relational data warehouse as well as your MDX cube sources from within the same project.

You can perform database instance configuration tasks for a project from the Project Configuration Editor, which can be accessed by right-clicking a

project and selecting **Project Configuration**. The tasks described in this section require MicroStrategy Administrator privileges.



As shown in the image above, an MDX cube database instance that is connected to and has objects defined in the project is represented in the Project Configuration Editor with bold formatting. The shading and color of a database instance name in the list of MDX cube database instances reflects how the database instance is being used in the project, as described below:

- Normal font: The database instance is not being used in the project.
- **Bolded font:** The project contains objects that are dependent on the database instance. You cannot remove a database instance that has dependent objects for the project.

You can choose when an MDX cube schema associated with an MDX cube database instance is loaded for a project. The appearance of the database instance icon as transparent or opaque indicates whether the schema for the MDX cube database instance is loaded when an MDX cube report is executed or Intelligence Server starts. For more information on schema loading options for MDX cube database instances and how they affect performance, see [MDX cube schema loading](#) below.



Note the following:

- While warehouse database instances are accessed using the Warehouse Catalog, MDX cube database instances are accessed using the MDX Cube Catalog. For information on importing MDX

cubes with the MDX Cube Catalog, see [Importing MDX cubes, page 80](#).

- For more information on the Warehouse Catalog which defines the tables from a relational database included in your project, see the *MicroStrategy Project Design Guide*.
- For more information on connecting a project to a warehouse database instance, see the *MicroStrategy Installation and Configuration Guide*.

Removing an MDX cube database instance from a project

You can remove a database instance from a project only if there are no dependent objects in the project for the database instance. This includes removing all MDX cubes for the MDX cube source database instance that have been integrated into MicroStrategy. For more information on removing a database instance and related MDX cube managed objects from a project, refer to the *MicroStrategy System Administration Guide*.

Once all dependent objects are removed, you can click **Remove** from the Schema Maintenance dialog box to remove an MDX cube database instance from a project.

MDX cube schema loading

You can choose when an MDX cube schema associated with an MDX cube database instance is loaded for a project. This affects when the load time required for MDX cube schemas occurs. By default, MDX cube schemas are loaded as needed when MDX cube reports are executed. You can choose to load MDX cube schemas when Intelligence Server starts.

The considerations for whether to load MDX cube schemas at Intelligence Server startup or at MDX cube execution time are described in the table below:

Method	Pros	Cons
Loading MDX cube schemas when Intelligence Server starts	<ul style="list-style-type: none"> MDX cube report runtime performance is optimized since the schema for the report has already been loaded. This is a good option if MDX cube reports are commonly used in a project. 	<ul style="list-style-type: none"> The overhead experienced during Intelligence Server startup is increased due to the processing of loading MDX cube schemas. If you import a new MDX cube or refresh an MDX cube, you must update the schema of your project to load the MDX cube before report execution and to use the updated MDX cube schema.
Loading MDX schemas when an MDX cube report is executed	<ul style="list-style-type: none"> The overhead experienced during Intelligence Server startup is decreased as compared to including loading MDX cube schemas as part of the startup tasks. If MDX cube schemas are not required, they do not need to be loaded and no overhead is experienced. This is a good option if MDX cube reports are supported for a project, but are rarely used in the project. 	<ul style="list-style-type: none"> MDX cube report runtime performance can be negatively affected due to the processing of loading MDX cube schemas.

Prerequisites

This procedure assumes you have already created an MDX cube database instance that connects to your MDX cube source.

To define MDX cube schema loading options

- 1 In MicroStrategy Developer, log in to your project that is connected to an MDX cube source. You must log in as a user with administrator privileges.
- 2 Right-click the project and select **Project Configuration**. The Project Configuration Editor opens.
- 3 In the **Categories** list, expand **Database instances**, and then select **MDX Data warehouses**. The Database instance - MDX Data warehouses options are displayed on the right.

- 4 Click **Schema Maintenance**. The Schema Maintenance dialog box opens.
- 5 To change the schema loading options for a database instance, from the **Preload** column, select one of the following options:
 - **Yes**: MDX cubes are loaded when Intelligence Server starts.
 - **No**: MDX cubes are loaded when MDX cube reports are executed.
- 6 Click **OK** to close the Schema Maintenance dialog box.
- 7 Click **OK** to accept any changes and close the Project Configuration Editor.
- 8 Whenever you modify the schema loading option for an MDX cube source database instance, you must update the schema for your project to reflect these changes. In **Developer**, from the **Schema** menu, select **Update Schema**.

Exchanging the database instance for an MDX cube schema

When you integrate MDX cube sources into MicroStrategy, the data is integrated as an MDX cube schema. Once you integrate an MDX cube source into MicroStrategy, you can exchange the database instance used to connect to the MDX cube schema for a different database instance. This allows you to use different database instances with different login and connection information to access an MDX cube schema.

The steps below show you how to exchange the database instance used to connect to an MDX cube schema for a different database instance.

Prerequisites

- An MDX cube source has been integrated into MicroStrategy (see [Chapter 3, Integrating MDX Cubes into MicroStrategy](#)).
- You have created the database instance to move the MDX cube schema to. This database instance must meet the following requirements:
 - The database instance does not have any MDX cube schema defined for it.
 - The database instance uses the same database connection type as the database instance that currently stores the MDX cube schema. For example, you cannot move an MDX cube schema for SAP data to a

database instance that has connection information for a Microsoft Analysis Services MDX cube source.

To exchange the database instance for an MDX cube schema

- 1 In MicroStrategy Developer, log in to your project that is connected to an MDX cube source. You must log in as a user with administrator privileges.
- 2 Right-click the project and select **Project Configuration**. The Project Configuration Editor opens.
- 3 In the **Categories** list, expand **Database instances**, and then select **MDX Data warehouses**. The Database instance - MDX Data warehouses options are displayed on the right.
- 4 Click **Schema Maintenance**. The Schema Maintenance dialog box opens, which lists all available MDX cube schemas for the project.
- 5 From the **Database instance** column for an MDX cube schema, select the database instance to move the MDX cube schema to.
- 6 Review the warning message and click **Yes** to move the MDX cube schema to the database instance that you selected.
- 7 Click **OK** to close the Schema Maintenance dialog box.
- 8 Click **OK** to accept any changes and close the Project Configuration Editor.

Supporting large result sets for MDX cube reports

The maximum number of result rows allowed for an MDX cube report is enforced by the same governing setting which is set for reports that have their results returned from a relational database. However, queries on MDX cube sources during MDX cube report execution can return more result rows than may be expected. This can cause some MDX cube reports to fail because the maximum number of result rows allowed for reports is exceeded.

When MDX is executed against an MDX cube source, additional result sets are required because an MDX data set is not simply a flat table as in relational databases. MDX data sets need to account for ragged and

unbalanced hierarchies and other customizable relationships between attribute elements that are not always possible to model in relational databases. The additional information is used to reconstitute the data set in MicroStrategy as it exists in the MDX cube source.

To support the possibility of a large number of result rows returned by MDX cube reports, you must define the governing setting to a higher maximum. Steps to define this governing setting are provided in the procedure below.

To support large result sets for MDX cube reports

- 1 In MicroStrategy Developer, log in to a project that is connected to an MDX cube source.
- 2 Right-click the project, and then select **Project Configuration**. The Project Configuration Editor opens.
- 3 In the **Categories** list, expand **Governing Rules**, and then select **Result sets**.
- 4 Underneath **Final Result Rows**, define the **All other reports** governing setting to a number that is high enough to support the execution of your MDX cube reports.

Be aware that the total number of result rows for an MDX cube report can be at least four times the number of final result rows displayed on an MDX cube report.

- 5 Click **OK** to save your changes and close the Project Configuration Editor.

Inheriting MDX cube source formats for metric values

You can inherit value formats from your MDX cube source and apply them to metric values in MicroStrategy MDX cube reports. This provides more formatting flexibility. If you do not inherit value formats, you can only apply a single format to all metric values on an MDX cube report. For an explanation of all of the reporting benefits of inheriting value formats from your MDX cube source, see [Inheriting MDX cube source formats for metric values, page 181](#).

Defining an MDX cube source database instance to inherit MDX cube source formats means that all of the database instance's MDX cube reports will

inherit MDX cube source formats by default. You can also specify whether to inherit these formats on a report-by-report basis, as described in [To inherit MDX cube source formats for metric values, page 182](#).

To define MDX cube source database instances to inherit MDX cube source formatting

- 1 In MicroStrategy Developer, log in with administrative privileges to a project source.
- 2 In the Folder List, expand **Administration**, expand **Configuration Managers**, and select **Database Instance**. Database instances for the project source are displayed.
- 3 Right-click an MDX cube source database instance, and select **VLDB Properties**. The VLDB Properties Editor opens.
- 4 From the **Tools** menu, select **Show Advanced Settings**.
- 5 In the **VLDB Settings** list, expand **MDX**, and then select **MDX Cell Formatting**.
- 6 Clear the **Use default inherited value** check box.
- 7 You can select one of the following options:
 - **MDX metric values are formatted per column:** If you select this option, MDX cube source formatting is not inherited. You can only apply a single format to all metric values on an MDX cube report.
 - **MDX metric values are formatted per cell:** If you select this option, MDX cube source formatting is inherited. Metric value formats are determined by the formatting that is available in the MDX cube source, and metric values can have different formats.
- 8 Click **Save and Close** to save your changes and close the VLDB Properties Editor.
- 9 Restart Intelligence Server to update the project source with the changes.

Authentication

Most of the standard MicroStrategy platform authentication features also apply to MDX cube sources and MDX cube reports. The authentication methods are described below:

- **NT (Windows) authentication:** Uses your network login ID to authenticate a connection to MicroStrategy Intelligence Server. NT (Windows) authentication can be used to authenticate the user to Intelligence Server, but not to MDX cube sources.
- **Standard authentication and LDAP authentication:** Are supported independently from the data source that is being used, for example, relational databases or MDX cube sources.
- **Connection mapping:** Is supported the same way as for standard MicroStrategy reports. In addition, specific connection mappings may be designated for each database instance and user or group combination.
- **Database authentication:** Is supported in the same way as for relational data providers. If multiple sources are configured for database authentication, the same login information must be applicable to all sources.

If your project connects to SAP BW as an MDX cube source, you can enable users to log in to a MicroStrategy project with their SAP user credentials and use SAP BW roles as a method to grant the users privileges in MicroStrategy. This SAP user creation and security in MicroStrategy requires that you use the MicroStrategy database authentication option. For information on importing SAP users and roles into MicroStrategy, see [Authenticating SAP BW users in MicroStrategy projects, page 70](#).

- **Integrated authentication:** Is supported for single sign-on authentication to Microsoft Analysis Services and SAP BW MDX cube sources. For information on the single sign-on functionalities supported for MDX cube sources and necessary configuration steps, see:
 - [Single sign-on to Microsoft Analysis Services, page 68](#)
 - [Single sign-on to SAP BW, page 69](#)

The authentication methods described above cover authentication within the MicroStrategy platform. For information on authentication and permissions required for components of your SAP BW, Oracle Essbase, and Analysis Services systems, see the sections on connecting to these MDX cube sources below.

For information on MicroStrategy authentication in general, refer to the *MicroStrategy System Administration Guide*.

Single sign-on to Microsoft Analysis Services

MicroStrategy supports single sign-on authentication of a Windows user to MicroStrategy Developer, MicroStrategy Web, and Microsoft Analysis Services serving as an MDX cube source in MicroStrategy. This is supported in MicroStrategy with the integrated authentication option.

Integrated authentication enables a user to log in once to their network account. This can include the user's initial log in to their computer. The user does not need to log in again separately to MicroStrategy Developer or MicroStrategy Web.

With integrated authentication, Windows users' login credentials can also be used to execute against MDX cubes imported from Analysis Services into MicroStrategy. For example, when a Windows user runs an MDX cube report that accesses an Analysis Services MDX cube, their Windows login credentials are used to verify access to the MDX cube. This enables you to enforce Analysis Services security with Windows logins through MicroStrategy.

To support this type of authentication, refer to the *Identifying Users: Authentication* chapter of the *MicroStrategy System Administration Guide*.

After you have enabled integrated authentication you must allow access to Microsoft Analysis Services MDX cubes, which is described in [Allowing access to Microsoft Analysis Services MDX cubes](#) below.

Allowing access to Microsoft Analysis Services MDX cubes

MDX cubes in MicroStrategy can be imported from Microsoft Analysis Services for reporting and analysis purposes. To enable a Windows user access to these MDX cubes in MicroStrategy, you must assign the Windows user membership to one or more Analysis Services security role definitions. You define these security roles in Analysis Services.

For information and steps to connect MicroStrategy to Microsoft Analysis Services, see the following sections:

- [Connecting to Analysis Services servers, page 51](#)

- [Connecting to Analysis Services servers, page 51](#)

Single sign-on to SAP BW

MicroStrategy supports single sign-on authentication of a Windows user to MicroStrategy Developer, MicroStrategy Web, and SAP BW serving as an MDX cube source in MicroStrategy. This is supported in MicroStrategy with the integrated authentication option.

Integrated authentication enables a user to log in once to his network account. This can include the user's initial log in to his computer. This same authentication is used to confirm access to the SAP BW data that has been integrated into MicroStrategy. To support this type of authentication, you must complete the following requirements:

- In addition to your SAP BW server that stores the data that is integrated into MDX cubes in MicroStrategy, you must also configure the SAP NetWeaver Application Server Java to:
 - Use your SAP BW server as the repository for your SAP user accounts.
 - Accept Kerberos tickets as a valid form of login credentials and generate an associated SAP Logon Ticket.

For steps to complete the configurations listed above, see your third-party SAP documentation.

- To configure your MicroStrategy environment to use integrated authentication, refer to the *Identifying Users: Authentication* chapter of the *MicroStrategy System Administration Guide*.
- You must include an additional connection parameter, `EPURL`, as part of the database instance that you created for your SAP BW connection (the process of creating this database instance is covered in [Connecting to SAP BW servers on Windows, page 38](#) and [Connecting to SAP BW servers on UNIX and Linux, page 42](#)). The steps to include this connection parameter are provided below.
 - 1 Within MicroStrategy Developer, right-click the database instance for your SAP BW connection and select **Edit**. The Database Instances Editor opens.
 - 2 In the Database connection area, select the database connection and click **Modify**. The Database Connections dialog box opens.

- 3 On the **Advanced** tab, add the EPURL parameter to the Connection Settings. The EPURL is of the format:

```
EPURL=http://Server:Port/irj/portal;
```

Where:

- *Server* is the host name of the Java version of your SAP server. This must match how the service principal name is defined in your keytab file used for configuring your server with Kerberos. For example, you must include the domain if the domain is provided before the @ symbol for the service principal name.
 - *Port* is the port number used to communication to the Java version of your SAP server.
- 4 Click **OK**, and then click **OK** again to save the changes to the database instance.

Authenticating SAP BW users in MicroStrategy projects

You can enable users to log in to a MicroStrategy project that is connected to an SAP BW MDX cube source with their SAP user credentials, and use SAP BW roles as a method to grant the users privileges in MicroStrategy.

This is achieved by importing SAP BW users and SAP BW roles into MicroStrategy as users and groups, respectively. You can then grant privileges for these imported users and groups in the same way you can for any MicroStrategy user or group. Refer to the *MicroStrategy System Administration Guide* for general information on granting privileges through users and groups.

Imported SAP BW users are granted group privileges at runtime. This means that when a user attempts to perform a task that requires a privilege, it is at this time that the group membership is checked and privileges are granted. This is visible in the User Editor: A MicroStrategy user (associated with an SAP BW user) does not list the privileges it inherits from its MicroStrategy groups (associated with SAP BW roles). Also, users are not shown as members of the group.

SAP BW users and roles are imported into MicroStrategy when a user logs in to a MicroStrategy project with their SAP BW user name and password. How SAP BW users and roles are imported into MicroStrategy is determined by the following options available in the MicroStrategy Intelligence Server Configuration Editor:

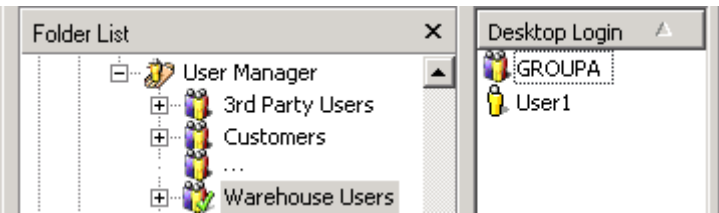
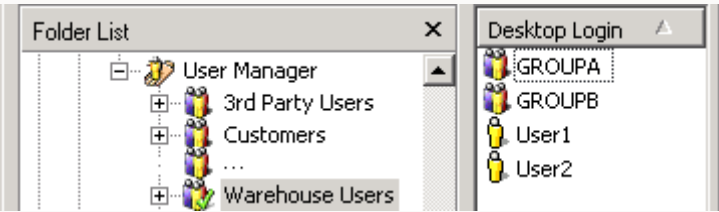
- **Import users:** A user with the same SAP BW user name is created in MicroStrategy. This user is created within the Warehouse Users group.
- **Search for groups:** The MicroStrategy groups assigned to a MicroStrategy user are synchronized with the SAP BW roles assigned to the SAP BW user. This means that if SAP BW roles have been added or removed for an SAP BW user, the associated MicroStrategy user is added or removed from the MicroStrategy groups that represent the SAP BW roles.

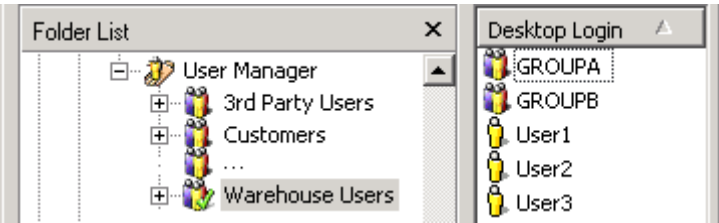
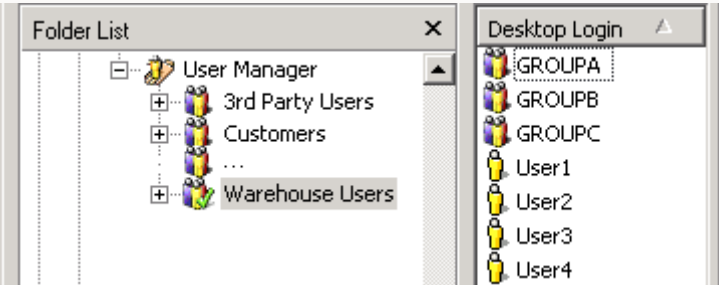
If you select to search for groups, you have the following option to import SAP BW roles as groups in MicroStrategy:

- **Import groups:** All SAP BW roles that the SAP BW user is a member of are imported as groups into MicroStrategy. These groups are created within the Warehouse Users group and only have inherited privileges from the Warehouse Users group. Once these groups are created in MicroStrategy, you can assign privileges to these groups.

For steps to select these options, see [Enabling SAP BW user and group import options, page 75](#).

For example, you select all three options listed above to import SAP BW users and roles as MicroStrategy users and groups. The following four SAP BW users log in to a MicroStrategy project in the order listed below:

SAP BW User	SAP BW Roles	Imported MicroStrategy Users and Groups
User1	GroupA	 <p>The screenshot shows the 'Folder List' on the left with 'User Manager' expanded, showing '3rd Party Users', 'Customers', and 'Warehouse Users'. On the right, the 'Desktop Login' window shows 'GROUPA' and 'User1' listed.</p>
User2	GroupB	 <p>The screenshot shows the 'Folder List' on the left with 'User Manager' expanded, showing '3rd Party Users', 'Customers', and 'Warehouse Users'. On the right, the 'Desktop Login' window shows 'GROUPA', 'GROUPB', 'User1', and 'User2' listed.</p>

SAP BW User	SAP BW Roles	Imported MicroStrategy Users and Groups
User3	GroupA and GroupB	
User4	GroupA and GroupC	

Once the users and groups are created in MicroStrategy, you can grant privileges to the users and groups.

Prerequisites for importing SAP BW users and roles

To import SAP BW users and roles into MicroStrategy as users and groups, you must first perform the following prerequisites in the order they are listed below:

- Establish a connection between MicroStrategy and the SAP BW system using a MicroStrategy database instance. Steps are in [Connecting to SAP BW servers, page 36](#).
- Define your SAP BW database instance as the authentication database instance for the MicroStrategy project that contains your MDX cubes. Steps are in [Defining SAP BW as an authentication database, page 73](#).
- Grant the Use Developer and Web User privileges to the Warehouse Users group for the project that is connected to your MDX cube source. Steps are in [Granting SAP BW users access to MicroStrategy Developer and Web, page 73](#).
- Select the SAP BW user and role import options. Steps are in [Enabling SAP BW user and group import options, page 75](#).

- Define the project source to use database authentication to authenticate users based on SAP BW user credentials. Steps are in [Defining project sources to authenticate using SAP BW user credentials, page 76](#).

Defining SAP BW as an authentication database

To use SAP BW user credentials to log in to a MicroStrategy project, you must define your SAP BW MDX cube source as an authentication database for the project. This means that your SAP BW MDX cube source is the storage location for users and their respective credentials.

To define SAP BW as an authentication database

- 1 In MicroStrategy Developer, log in to a project source with a project connected to an MDX cube source.
- 2 From the **Folder List**, right-click a project and select **Project Configuration**. The Project Configuration Editor opens.
- 3 From the **Categories** list, expand **Database instances**, expand **Authentication**, and then select **Metadata**. The Database Instances Authentication User information is displayed on the right.
- 4 From the drop-down list, select the database instance that connects to your SAP BW MDX cube source.
- 5 Click **OK** to save your changes and close the Project Configuration Editor.

Granting SAP BW users access to MicroStrategy Developer and Web

SAP BW users and roles are imported into MicroStrategy as users and groups when an SAP BW user logs in to a MicroStrategy project. For an SAP BW user to log in to a MicroStrategy project and be imported, you must grant the Use Developer or Web User privileges to the Public/Guest group in MicroStrategy.

It is recommended that you grant a minimum number of privileges to the Public/Guest group. Granting the Use Developer and Web User privileges to the Public/Guest group enables users to attempt to log in to a MicroStrategy project with their SAP BW user credentials. This attempted login imports the SAP BW user and roles into MicroStrategy as determined by the options

described in [Authenticating SAP BW users in MicroStrategy projects, page 70](#).

With only these privileges applied to the Public/Guest group, the user is denied access to the project. This enables the user and any associated groups to be imported before you grant specific, additional privileges to the user for the project. You can then grant the appropriate privileges directly to the imported user and any associated groups, rather than granting such privileges to the Public/Guest group.

To grant SAP BW users access to MicroStrategy Developer and Web

- 1 In MicroStrategy Developer, log in to a project source using an account with administrative privileges.

To create a security role

You can use any security role that has the Use Developer and Web User privileges selected. This procedure demonstrates how to create a security role with only the Use Developer and Web User privileges.

- 2 From the Folder list, expand **Administration** menu, expand **Configuration Managers**, right-click **Security Roles**, point to **New**, and select **Security Roles**. The Security Role Editor opens.
- 3 In the **Name** field, type **Use Developer and Web user**.
- 4 On the **Privileges** tab, from the **Available privileges** pane, expand **Web Reporter**, and then select the **Web user** check box.
- 5 On the **Privileges** tab, from the **Available privileges** pane, expand **Analyst**, and then select the **Use Developer** check box.
- 6 Click **OK** to save your changes and close the Security Roles Editor.

To assign a security role to the Public/Guest group

- 7 From the **Folder List**, expand **Administration**, and then expand **User Manager**.
- 8 Right-click the **Public/Guest** group, and select **Edit**. The Group Editor opens.
- 9 On the **Project Access** tab, in the **Security Role Selection** row at the top, click the drop-down list for the project connected to an SAP BW

MDX cube source, and select **Use Developer and Web user + Inherited Access**.


10 Click **OK** to save your changes and close the Group Editor.

Enabling SAP BW user and group import options

SAP BW users and roles are imported into MicroStrategy when a user logs in to a MicroStrategy project with their SAP BW user name and password. How SAP BW users and roles are imported into MicroStrategy is determined by options available in the MicroStrategy Intelligence Server Configuration Editor.

To enable SAP BW user and group import options

- 1** In MicroStrategy Developer, log in to a project source using an account with administrative privileges.
- 2** From the **Administration** menu, point to **Server**, and then select **Configure MicroStrategy Intelligence Server**. The MicroStrategy Intelligence Server Configuration Editor opens.
- 3** In the **Categories** list, select **SAP User management**.
- 4** Select the check boxes for the following options:
 - **Import users**
 - **Search for groups**
 - **Import groups**

 For information on each option listed above, see [Authenticating SAP BW users in MicroStrategy projects, page 70](#).
- 5** Click **OK** to save your changes and close the MicroStrategy Intelligence Server Configuration Editor.

Defining project sources to authenticate using SAP BW user credentials

You must use database authentication to enable SAP BW users to log in to a MicroStrategy project with their SAP BW user credentials. Defining database authentication for a project source makes database authentication available for the projects within the project source.

Use the procedures below to define project sources to authenticate using SAP BW user credentials, depending on whether users are logging in to MicroStrategy projects through MicroStrategy Developer or Web:

- *To define project sources in MicroStrategy Developer to authenticate using SAP BW user credentials, page 76*
- *To define project sources in MicroStrategy Web to authenticate using SAP BW user credentials, page 76*

To define project sources in MicroStrategy Developer to authenticate using SAP BW user credentials

- 1 In Developer, log in to a project source using an account with administrative privileges.
- 2 Right-click the project source and select **Modify Project Source**. The Project Source Manager opens.
- 3 On the **Advanced** tab, in the Authentication mode area, select **Use login id and password entered by the user for Warehouse (database authentication)**.
- 4 Click **OK** to save your changes.
- 5 A warning message displays that your connection to the project source will be closed. Click **Yes**.

To define project sources in MicroStrategy Web to authenticate using SAP BW user credentials

- 1 From the Windows **Start** menu, point to **Programs**, then **MicroStrategy Tools**, and then select **Web Administrator**. The Administrator page for MicroStrategy Web opens in a browser.

2 On the left, in the **Intelligence Servers** list, click **Default Properties**. The Default Server Properties page opens.

3 In the **Login** area, to the left of the **Database Authentication** login mode, select the **Enabled** check box.



To enable database authentication as the default mode of authentication used to log in to MicroStrategy project, select the **Default** option.

4 Click **Save** to save your changes.

INTEGRATING MDX CUBES INTO MICROSTRATEGY

Introduction

Once you understand the relationships between the objects in an MDX cube source and MicroStrategy, and you connect to your MDX cube source, you can start integrating your MDX cube data into MicroStrategy. This chapter assumes that you are familiar with designing a MicroStrategy project and the structure and design of your data within the MDX cube source.

The best place to start is with the MDX Cube Catalog, where you can perform the following tasks which are covered in this chapter:

- *Importing MDX cubes, page 80*
- *Mapping MDX cubes, page 96*
- *Creating metrics from MDX cube data, page 129*
- *Creating data marts of MDX cube data, page 141*

The importing and mapping tasks integrate your MDX cube source data into your MicroStrategy project, and therefore must be done before you can begin creating any MDX cube reports. The MDX Cube Catalog can be accessed from the **Schema** menu in MicroStrategy Developer.

The MDX Cube Catalog is available only after an MDX cube source database instance has been created. To learn how to create a database instance for an MDX cube source, see [Chapter 2, Connecting to MDX Cube Sources](#).

In the following procedures in this chapter, SAP BW is used as the example MDX cube source but the procedures are similar for Analysis Services, Oracle Essbase, and TM1.

After you have fully integrated your MDX cube data into MicroStrategy, report designers can create MicroStrategy MDX cube reports, and analysts can then view and analyze data from MDX cube sources. MDX cube reports are covered in [Chapter 4, Reporting on MDX Cubes](#).

If you upgrade a pre-9.0 MicroStrategy project that includes MDX cubes, see the *Upgrade Guide* for information on updating MDX objects to enhance the performance of using these objects in MicroStrategy.

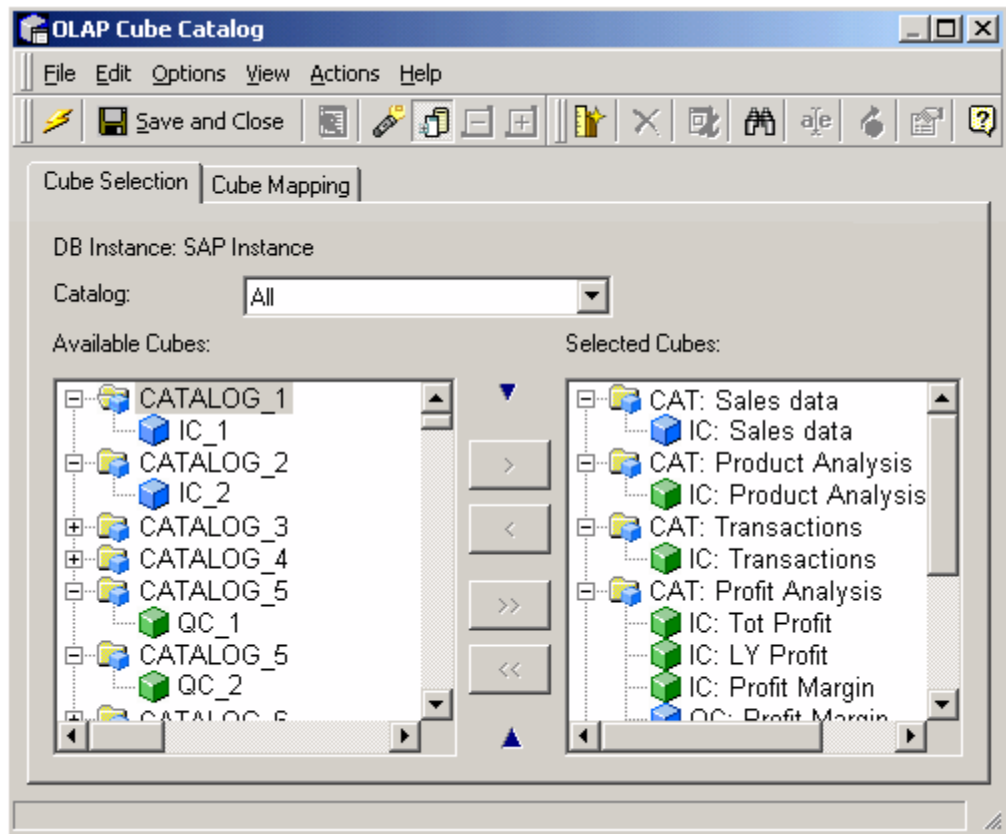
Importing MDX cubes

After you have connected to an MDX cube source, importing MDX cubes is the next step in integrating your MDX cube source data into MicroStrategy. Importing MDX cubes into MicroStrategy is described in the following sections:

- [Importing MDX cubes before report creation, page 81](#)
- [Importing levels and suffixes for characteristics, page 85](#)
- [Importing additional measure structure, page 88](#)
- [Importing MDX cubes during report creation, page 91](#)
- [Updating MDX cube structure, page 92](#)
- [Maintaining MDX cubes between multiple projects, page 94](#)
- [About managed objects, page 95](#)

Importing MDX cubes is performed on the Cube Selection tab of the MDX Cube Catalog, as shown in the image below. When you open the MDX Cube Catalog, all the MDX cubes are displayed under their respective catalog names in the Available Cubes pane. Using the plus (+) or minus (-) sign next to a catalog name, you can expand or hide the cubes contained in this catalog.

A catalog is designated with an icon showing a folder with a small cube super-imposed on it. An InfoCube is designated with a cube icon in blue. A query cube is designated with a cube icon in green.



✪ If you create new cubes in Analysis Services and the cubes are not displayed in the MDX Cube Catalog, you may have to modify some permissions in Analysis Services. For details on how to make Analysis Services cubes available for import in the MDX Cube Catalog, see MicroStrategy Tech Note TN14063.

Importing MDX cubes before report creation

Before you can create MDX cube reports, you need to import MDX cubes from your MDX cube source into MicroStrategy. To import MDX cubes, the following prerequisites must be met:

- MDX cubes can be imported into a MicroStrategy project only by an architect with the Import MDX Cube privilege.
- You can import MDX cubes only after an MDX cube source database instance has been created. For information on creating an MDX cube

source database instance and connecting to an MDX cubes source, see [Chapter 2, Connecting to MDX Cube Sources](#).

- If you are import MDX cubes from SAP BW, any existing SAP BW query can be released for analysis within MicroStrategy. To release a query for analysis in MicroStrategy, select the **Allow External Access to This Query** check box under the Extended tab in the SAP Query Properties dialog box in the Query Analyzer interface.
- If you are importing SAP BW cubes that include variables of the type Replacement Path, you must remove them before importing the cubes into MicroStrategy. Otherwise, an error occurs when you attempt to import the SAP BW cube.

To import MDX cubes

- 1 In MicroStrategy Developer, log in to a project that is connected to an MDX cube source.
- 2 If you are using read only mode for the project, from the **Schema** menu, clear the **Read Only Mode** option to switch to edit mode.

Only one user can be editing a project at a given time. Therefore, if someone else is modifying the project, you cannot use the MDX Cube Catalog.

- 3 From the **Schema** menu, select **MDX Cube Catalog**.
 - If you have a single MDX cube source database instance created for the project, the MDX Cube Catalog opens.
 - If you have multiple MDX cube source database instances created for the project, a Database Instance dialog box opens. Select a valid MDX cube source database instance, click **OK**, and the MDX Cube Catalog opens.
- 4 Click the **Cube Selection** tab.
- 5 From the **Catalog** drop-down list, select the MDX cube to import. The catalog contains all the MDX cubes associated with it. You can also select **All** to display the MDX cubes for all catalogs.



To search for a specific MDX cube to import, from the **Edit** menu, select **Find**, or click the **Find** icon on the toolbar. The Find dialog box opens.

- 6 Click the plus (+) sign to expand the catalog folder and display the MDX cubes in the Available Cubes pane on the left.
- 7 If desired, you can preview the structure of an MDX cube before you import it into MicroStrategy and you can also synchronize with the most recent definition of MDX cube structures in the MDX cube source.

To do this, Right-click the MDX cube and select **Cube Structure**. The Cube Structure *MDX cube name* dialog box opens.

Synchronizing is helpful when a new characteristic or key figure has been added to an InfoCube in SAP BW. You can click **Update Structure** to update the MicroStrategy MDX cube to include these modifications. For information on updating MDX cube structures and its affect on reports and objects in MicroStrategy, see [Updating MDX cube structure, page 92](#).

- 8 In the MDX Cube Catalog, from the **Options** menu, select **Import options** to open the Import options dialog box.
- 9 If desired, you can determine how the two levels of a stand-alone characteristic are imported into MicroStrategy by selecting both of the following import options:
 - **Do not import Level 00 for flat hierarchies:** When selected, the first level aggregate data for a stand-alone characteristic is not imported into MicroStrategy. Only the detail data level for the characteristic is imported and mapped to an attribute in MicroStrategy.
 - **Suppress Level 01 suffix for flat hierarchies:** When selected, the suffix Level 01 is not included in the attribute name mapped to the detail data level of the characteristic.



For more information on stand-alone characteristics and how these options import them into MicroStrategy, see [Importing levels and suffixes for characteristics, page 85](#).

- 10 If desired, you can synchronize the names of schema objects in MicroStrategy with the names of objects in the MDX cube source. To enable this synchronization, select the **Synchronize logical object names with source** check box. This check box is cleared by default.

By default, this check box is cleared. If you select this check box, the attributes, hierarchies, metrics, and other schema objects in MicroStrategy are renamed if the names in the MDX cube source are updated. This name synchronization is applied when you update the MDX cube structure (see [Updating MDX cube structure, page 92](#)).

- 11 If desired, you can import additional measure structure from MDX cube sources. To do this, select the **Import measure as a regular dimension** check box.

By default, this check box is cleared by default. If the check box is greyed out and unavailable for selection, this means the MDX cube source you are connected to cannot support the integration of additional measure structure.

If you select this check box, the additional measure structure is integrated into MicroStrategy, and can support the hierarchal display of measures in an MDX cube source. For examples of this support and how it can be used, see [Importing additional measure structure, page 88](#).

- 12 Click **OK** to save your import option selections and return to the MDX Cube Catalog.
- 13 Select the MDX cubes to import, and click the single arrow (>). To import all the MDX cubes, click the double arrows (>>).
- 14 To remove an MDX cube, right-click any MDX cube in the Selected Cubes pane, and select **Remove [cube name]**.
- 15 Once imported, the imported MDX cubes are displayed in the Selected Cubes pane on the right.
- 16 Click **Save** to save your progress.

MicroStrategy automatically maps attributes, metrics, prompts, and other objects to the MDX cube (see [Mapping MDX cubes, page 96](#)). However, if you plan to create relations between your MDX cube source data to data from a different data source included in the MicroStrategy project, you can map MDX cube data to existing attributes in the project (see [Mapping MDX cube data to project attributes, page 103](#)). Once the data is mapped to MicroStrategy objects, you can build reports that access the imported MDX cubes.

If the MDX cube needs to be updated to retrieve new data, or access the data in a different MDX cube data source, see [Updating MDX cube structure, page 92](#).










Once the first MDX cube for an MDX cube source is imported into MicroStrategy, a **Data Explorer** is added to the MicroStrategy project. The Data Explorer helps you browse through data for its associated MDX cube source. You can find the Data Explorer for the MDX cube source in the Folder List of Developer, under the associated MicroStrategy project.

Importing levels and suffixes for characteristics

When characteristics in MDX cubes are imported into MicroStrategy, they can be imported both as a stand-alone characteristic and as part of a hierarchy defined in the MDX cube source.

Stand alone characteristics are imported into MicroStrategy with two levels: the first level is an aggregate of all the characteristic data, and the second level is the detail data. For example, the Region characteristic shown below is imported into MicroStrategy as a stand-alone characteristic that is mapped to a Region hierarchy with two attributes.

Physical view	Logical view
 Region	
 Region	 Region
 Region Level 00	 Region Level 00
 Region Level 01	 Region Level 01

The Region Level 00 attribute maps to the aggregate data level and the Region Level 01 attribute maps to the detail data level.

However, you may not need to import the aggregate level of a characteristic into MicroStrategy. You can determine how the two levels of a stand-alone characteristic are imported into MicroStrategy with the following import options:






- **Do not import Level 00 for flat hierarchies:** Select this check box to exclude the first level aggregate data for a stand-alone characteristic during the import into MicroStrategy. Only the detail data level for the characteristic is imported and mapped to an attribute in MicroStrategy.
- **Suppress Level 01 suffix for flat hierarchies:** Select this check box to exclude the suffix Level 01 from the attribute name mapped to the detail data level of the characteristic. An attribute named Region (rather than Region Level 01) is mapped to the detail data of the imported Region characteristic.



The behavior listed above assumes that the attributes for the MDX cube are being imported for the first time. Attributes can be shared by multiple MDX cubes imported into a MicroStrategy project. This can affect the import behavior listed above, as described in the section [Shared attribute effects on import behavior, page 86](#).










For steps to select these options as part of the MDX cube import process, see [To import MDX cubes, page 82](#).

If you select both of these import options for a Region characteristic, the characteristic is imported into MicroStrategy as shown below:

Physical view	Logical view
 Region	
 Region	 Region
 Region Level 01	 Region

Notice that only the detail level is imported and mapped to an attribute, and the attribute name does not include the suffix Level 01.

These options do not affect characteristics when they are imported as part of a hierarchy. For example, you have defined a Reg Org hierarchy based off the Region characteristic data. All levels and suffixes for the hierarchy are imported and mapped into MicroStrategy regardless of whether you selected any of the import options, as shown below:

Physical view	Logical view
 Region	
 Reg Org	 Reg Org
 Reg Org Level 00	 Reg Org Level 00
 Reg Org Level 01	 Reg Org Level 01
 Reg Org Level 02	 Reg Org Level 02








If you plan to map your MDX cube data to project attributes that are part of a relational schema, you should map the stand-alone characteristics to the project attributes rather than the levels of a hierarchy. For more information on mapping MDX cube data to project attributes, see [Mapping MDX cube data to project attributes, page 103](#) and [Best practices for mapping MDX cubes to project attributes, page 107](#).

Shared attribute effects on import behavior

MDX cube characteristics mapped to attributes in MicroStrategy can be shared by multiple MDX cubes, which can affect how and when attributes are imported for MDX cubes.

The import options to exclude the Level 00 attributes and Level 01 suffixes are described in the section [Importing levels and suffixes for characteristics, page 85](#). The descriptions assume that the characteristics for the MDX cube are being imported into MicroStrategy for the first time.

By default, Level 00 attributes are not created and Level 01 suffixes are excluded from the attribute names mapped to Level 01 data. For example, a Region characteristic with Region Level 00 and Region Level 01 in your MDX cube source is imported into MicroStrategy. Only one attribute named Region is created, which is mapped to the Region Level 01 data as shown below.

Physical view	Logical view
 Region	
 Region	 Region
 Region Level 01	 Region

After importing this MDX cube and Region characteristic, you can import another MDX cube that shares the region data. If you use the same import options, the MDX cube simply re-uses the Region attribute already created for the characteristic. However, modifying the import options has the following impacts on import behavior in this scenario:

- Level 00 data is imported and a Level 00 attribute is created if you choose to import Level 00 data. A Level 00 attribute is created and shared for all MDX cubes that share imported attributes for the characteristic. This includes MDX cubes that were imported previously and were set to exclude Level 00 data. In the example above, if you choose to import Region Level 00 data, Region Level 00 data is imported and a Region Level 01 attribute is created in all MDX cubes that share the region data.

Once a Level 00 attribute has been imported into MicroStrategy, all MDX cubes that share this data must include this attribute, regardless of whether you choose to include Level 00 data or not.

- The Level 01 suffixes are not included even if you choose to include Level 01 suffixes. In the example above, if you choose to include Region Level 01 suffixes, the suffixes are not included and the existing Region attribute is used. This is to maintain a consistent schema across all MDX cubes that share data.

This is also the case if on your first import you choose to include Level 01 suffixes. The suffixes are included for any imported attributes. If you then choose to exclude the Level 01 suffixes, the suffixes are still included for any attributes that have already been imported and are shared by multiple MDX cubes.

You can rename an attribute from within the MDX Cube Catalog or MDX Cube Editor at any time. These modifications are reflected in all MDX cubes that share the modified attribute.

Importing additional measure structure

Measures in MDX cube sources are integrated into MicroStrategy as metrics by default. However, measures can include additional structure which cannot be supported by MicroStrategy metrics. This additional structure can support the hierarchal display of measures in an MDX cube source, which is shown in the example report below.

Promotions.PromoTypes	Measures.Measures2	Measures.Measures0	Metrics	Amount
Coupon	Original Price			567,531.00
	Price Paid			482,404.07
	Ratios	% of Total		0.01
		Avg Units/Transaction		0.95
	Returns			22,497.90
	No. of Packages			3,346.00
	Items per Package			3,174.00
Newspaper Ad	Original Price			551,840.25
	Price Paid			496,656.99
	Ratios	% of Total		0.01
		Avg Units/Transaction		0.96
	Returns			4,544.57
	No. of Packages			3,181.00
	Items per Package			3,051.00
No Promotion	Original Price			36,328,346.50
	Price Paid			36,328,346.50
	Ratios	% of Total		0.96
		Avg Units/Transaction		0.95
	Returns			893,968.25
	No. of Packages			219,966.00
	Items per Package			208,302.00
Temporary Price Reduction	Original Price			572,821.50
	Price Paid			458,257.20
	Ratios	% of Total		0.02
		Avg Units/Transaction		0.94
	Returns			14,102.00
	No. of Packages			3,466.00
	Items per Package			3,271.00

The Measures attributes shown in the report above display the measures for the MDX cube, including the hierarchical structure of the Ratios measure.

To support this additional structure when importing the data into MicroStrategy, you can import the measures as a regular MDX dimension. This method integrates the measures into MicroStrategy as an attribute,

which can support the additional structure for the measures. To support this additional structure in measures, be aware of the following:

- The additional structure for measures can only be integrated into MicroStrategy for Oracle Essbase MDX cube sources. This option is not available for all other MDX cube sources.
- You must choose to support this additional structure when importing the MDX cube into MicroStrategy. If the MDX cube is already imported without this support, you must remove the MDX cube and import it into MicroStrategy again. Steps on how to import this additional structure are provided in [To import additional measure structure from MDX cube sources, page 90](#).
- If you enable support for the additional structure, you cannot create compound metrics for the MDX cube. Creating compound metrics for MDX cubes is described in [Creating metrics from MDX cube data, page 129](#).
- If you enable support for the additional structure, a single metric called Amount is created for the MDX cube. When using this Amount metric, be aware of the following:
 - You must include the Amount metric, as well as the attributes created for the measures, on an MDX cube report to display the values for all the measure data. The attributes for the measures are created under a hierarchy named Measures by default. The report shown above displays an example of the Amount metric and Measures attributes on a report.
 - Since the measure data is displayed using a single Amount metric, only a single value format such as currency or percentage can be used for all the values. The report shown above uses a fixed value format that displays two decimal places.

To support multiple value formats, you must disable the support for this additional structure, which imports each measure as a separate metric in MicroStrategy.

- Metric qualifications are not supported for the Amount metric. If you create a metric qualification for the Amount metric, it can cause an error when the MDX cube report is executed.

Steps on how to import additional measure structures into MicroStrategy are provided below.

Prerequisites

- MDX cubes can be imported into a MicroStrategy project only by an architect with the Import MDX Cube privilege.
- You can import MDX cubes only after an MDX cube source database instance has been created. For information on creating an MDX cube source database instance and connecting to an MDX cubes source, see [Chapter 2, Connecting to MDX Cube Sources](#).

To import additional measure structure from MDX cube sources

- 1 In MicroStrategy Developer, log in to a project that is connected to a Oracle Essbase MDX cube source.
- 2 If you are using read only mode for the project, from the **Schema** menu, clear the **Read Only Mode** option to switch to edit mode.

Only one user can be editing a project at a given time. Therefore, if someone else is modifying the project, you cannot use the MDX Cube Catalog.


- 3 From the **Schema** menu, select **MDX Cube Catalog**.
 - If you have a single MDX cube source database instance created for the project, the MDX Cube Catalog opens.
 - If you have multiple MDX cube source database instances created for the project, a Database Instance dialog box opens. Select a valid Oracle Essbase MDX cube source database instance, click **OK**, and the MDX Cube Catalog opens.

- 4 In the MDX Cube Catalog, from the **Options** menu, select **Import options** to open the Import options dialog box.

- 5 Select the **Import measure as a regular dimension** check box.

This check box is cleared by default. If the check box is greyed out and unavailable for selection, this means that the MDX cube source you are connected to cannot support the integration of additional measure structure. The additional structure for measures can only be integrated into MicroStrategy for Oracle Essbase MDX cube sources.

- 6 Click **OK** to save your changes and close the Import options dialog box.

- 7 Click the **Cube Selection** tab.
- 8 From the **Catalog** drop-down list, select the MDX cube to import. The catalog contains all the MDX cubes associated with it. You can also select **All** to display the MDX cubes for all catalogs.
 To search for a specific MDX cube to import, from the **Edit** menu, select **Find**, or click the **Find** icon on the toolbar. The Find dialog box opens.
- 9 Click the plus (+) sign to expand the catalog folder and display the MDX cubes in the Available Cubes pane on the left.
- 10 Select the MDX cubes to import, and click the single arrow (>). To import all the MDX cubes, click the double arrows (>>).
- 11 Once imported, the imported MDX cubes are displayed in the Selected Cubes pane on the right.
- 12 Click **Save** to save your progress.
- 13 Click the **Cube Mapping** tab.
- 14 From the **Catalog\Cube** drop-down list, select the MDX cube you just imported. You can view the new objects created to support the additional measure structure:
 - Measures dimension: In the Physical view column, the Measures dimension includes attributes that represent the additional structure of the measures as they exist in the MDX cube source.
 - Measures: In the Physical view column, the Measures object displayed at the bottom of the list includes a single Amount metric.
- 15 You can continue to map the data in the MDX cube to prepare the data for reporting in MicroStrategy (see [Mapping MDX cubes, page 96](#)).

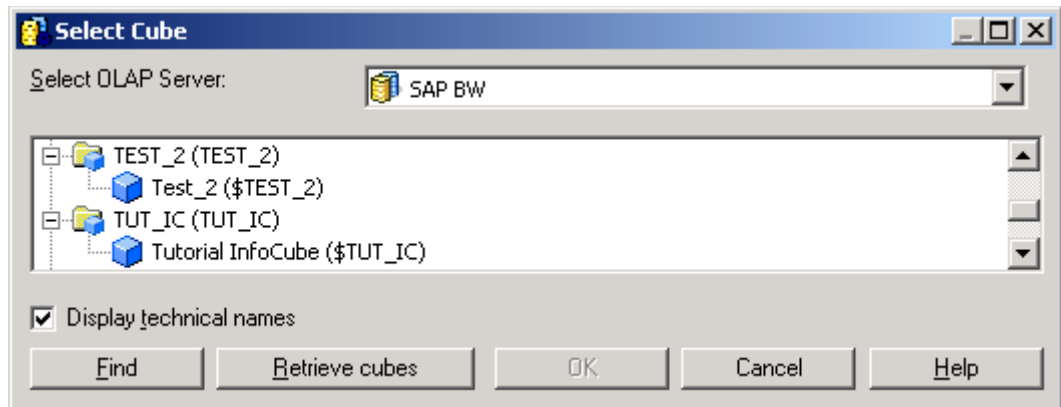
Importing MDX cubes during report creation

When you create an MDX cube report, you choose MDX cubes for your report from the Select Cube dialog box. This dialog box can also be used by an architect with the Import MDX Cubes privilege to import cubes by using the **Retrieve cubes** option. This option is available only after a database instance has been defined. For detailed information on defining a database

instance, see the sections for each MDX cube source covered in [Chapter 2, Connecting to MDX Cube Sources](#).



You can only import and map a single MDX cube when importing an MDX cube for your MDX cube report. This method also does not have all the options to map the MDX cube data to MicroStrategy objects. The MDX cube is imported with all objects mapped to the default managed objects (see [About managed objects](#)).



You can search for an MDX cube for your report by clicking the **Find** button at the bottom of this dialog box. The Find dialog box opens.

Updating MDX cube structure

Updating the structure of an MDX cube synchronizes the MDX cube definition in the MicroStrategy project with the latest MDX cube model in the MDX cube source. As a result, any addition or deletion of levels is reflected in the MDX cube structure that has been imported.



You can update an MDX cube to point to a different MDX cube from the same MDX cube source. For example, your current MDX cube includes data for the year 2014. A new source of data for 2015 is to be included in MicroStrategy as an MDX cube. You can import this data as a new MDX cube in MicroStrategy. Alternatively, you can update the current MDX cube to point to this new data. This update requires that the data for both MDX cubes is similar in structure so that the definition of the MDX cube in MicroStrategy is maintained. You can use Command Manager to apply this update to an MDX cube, and the scripts required are described in the *Command Manager Help*.



Be aware of the following:

- If any MDX cubes have been deleted from the cube source, that information is also updated in MicroStrategy. If any MicroStrategy reports used those MDX cubes, those reports will fail when they are run again.
- If you migrate MDX cube reports and other MDX cube objects between multiple projects, all MDX cube updates should be performed in a single project and migrated over to the other projects. For best practices on maintaining MDX cubes between multiple projects, see [Maintaining MDX cubes between multiple projects, page 94](#).

To update an MDX cubes structure

- 1 Access the MDX Cube Catalog.
- 2 In MicroStrategy Developer, log in to a project that is connected to an MDX cube source.
- 3 If you are using read only mode for the project, from the **Schema** menu, clear the **Read Only Mode** option to switch to edit mode.

Only one user can be editing a project at a given time. Therefore, if someone else is modifying the project, you cannot use the MDX Cube Catalog.

- 4 From the **Schema** menu, select **MDX Cube Catalog**.
 - If you have a single MDX cube source database instance created for the project, the MDX Cube Catalog opens.
 - If you have multiple MDX cube source database instances created for the project, a Database Instance dialog box opens. Select a valid MDX cube source database instance, click **OK**, and the MDX Cube Catalog opens.
- 5 Select the **Cube Selection** tab.
- 6 In the **Selected Cubes** pane, right-click an MDX cube and select **Update Structure**. Mappings of the MDX cubes are automatically updated with the latest definition in the MDX cube source.

For example, if a new dimension has been added to an InfoCube in SAP BW, new attributes and metrics are automatically mapped in MicroStrategy and displayed on the Cube Mapping tab to reflect this change.

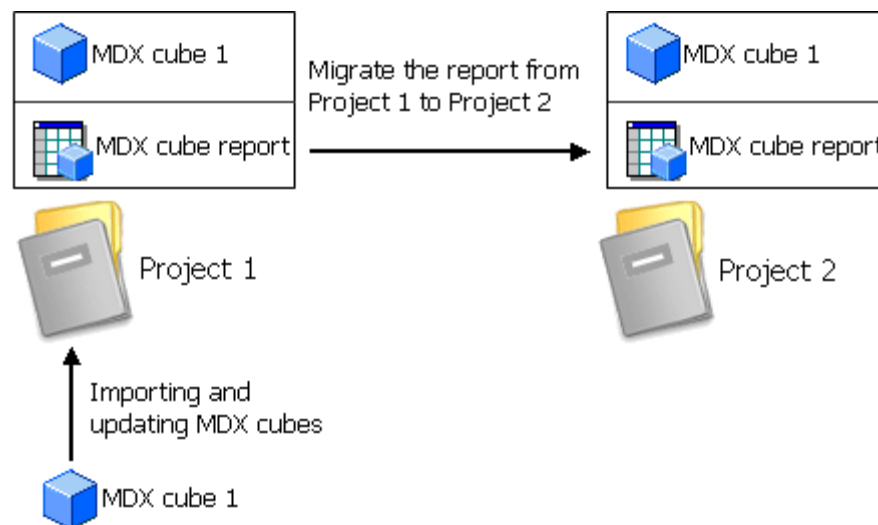
- 7 If desired you can modify any new MDX cube object mappings to MicroStrategy objects in the Cube Mapping tab (see [Mapping MDX cubes, page 96](#)).
- 8 Click **Save and Close** to save your modifications and close the MDX Cube Catalog.

Maintaining MDX cubes between multiple projects

When developing a reporting environment for MDX cube data, a common practice is to use two separate projects. One project is an initial testing project where MDX cubes are imported and MDX cube reports are created to perform the required analysis. This project is referred to as the testing project below. Once the MDX cube reports are fully developed, these reports are migrated into the second project. The second project is a production project, which is available to the users who are reporting and analyzing the MDX cube data. This project is referred to as the production project below.

The following best practices are applicable if you use this multiple project scenario to develop and maintain your reporting environment for MDX cube data:

- To maintain object consistency between the two projects, MDX cubes should be imported and updated only in the testing project, where the MDX cube reports are created. These MDX cubes are then included in the production project as part of the process to migrate the MDX cube reports from the testing project to the production project. This best practice is shown in the image below:



You can use Object Manager to migrate MDX cube reports between projects. Object Manager is described in the *System Administration Guide*.

- If the data for an MDX cube needs to be refreshed in the production project, this MDX cube update should be performed in the testing project. Once the MDX cube data has been updated within the testing project, the MDX cube can then be migrated from the testing project to the production project. This updates the data and ensures that the underlying objects remain consistent across projects.
- To reduce the chance that MDX cubes are imported or refreshed in the production project, users in this project should not be granted the Import MDX cube privilege. Disabling this privilege for users in this project prevents users from inadvertently importing or updating MDX cubes.

About managed objects

In MicroStrategy, standard *schema objects* relate the information in the logical data model and physical warehouse schema to the MicroStrategy environment. Managed objects are a type of schema object that relate MDX cube source data to the MicroStrategy environment. When an MDX cube is imported into a MicroStrategy project, *managed objects* (attributes, metrics, columns, tables, and so on) are created to describe the MDX cube. These schema objects are created automatically by MicroStrategy, which allows an MDX cube to be integrated quickly into your MicroStrategy project.



If your project contains both MDX cube source data and data mapped to a separate relational data source, managed objects do not allow you to create a direct relationship between the two sources of data. To solve this, you must map your MDX cube source data to project attributes that are part of your relational data model. For more information on why you should map your MDX cube source data, see [Mapping MDX cube data to project attributes, page 103](#).

A managed object is just like a standard MicroStrategy object except that it is created automatically by the system and is stored in a special system folder that is hidden from users. There is a way to access managed objects, as described in the procedure below.

To access managed objects

- 1 In MicroStrategy Developer, log in to the project that contains the managed objects you are searching for.
- 2 Right-click the project and select **Search for Objects**. The Search for Objects dialog box opens.
- 3 From the **Tools** menu, select **Options**. The Search Options dialog box opens.
- 4 Select the **Display Managed Objects** check box so that managed objects are displayed in the search result.



You can have the search return only managed objects by selecting **Display Managed Objects Only**.

- 5 Click **OK** to save your changes and return to the Search for Objects dialog box.
- 6 Enter any other search criteria to meet your search requirements, and then click **Find Now**.
- 7 Once the managed objects are listed in the search result, you can rename or edit a managed object by right-clicking its name.

A managed object can be removed once it is no longer referenced by another object in the project. The removal of unused managed objects is usually performed by an administrator. For more information on removing a database instance and its related managed objects, see the *Managing Your Projects* chapter of the *MicroStrategy System Administration Guide*.

Mapping MDX cubes

Mapping MDX cubes to objects in MicroStrategy is described in the following sections:

- [Shared MDX cube objects, page 101](#)
- [Best practices for mapping MDX cubes, page 103](#)
- [Mapping MDX cube data to project attributes, page 103](#)

- [Defining column data types for MDX cube data, page 109](#)
- [Preserving attribute element orders from MDX cube sources, page 113](#)
- [Supporting MDX cube source date data in MicroStrategy, page 117](#)
- [Defining unbalanced and ragged hierarchies, page 120](#)
- [Displaying hierarchies on MDX cube reports, page 122](#)
- [Mapping SAP BW variables to MicroStrategy prompts, page 124](#)

When a MicroStrategy architect defines a project, much of the process centers on identifying logical entities, such as attributes and facts, that exist in physical tables. For example, an architect might identify that the key for the Customer attribute exists in the table LU_CUSTOMER. Once the logical entities are identified, the architect can then define a logical and physical model in the MicroStrategy metadata. This model is referenced by the MicroStrategy SQL Engine to generate SQL when a user executes a report.

In the context of MDX cube sources, an MDX cube, instead of a single table, contains all the metadata information necessary to define a logical model and physical model. When you, as the architect, need to add an MDX cube to a project in MicroStrategy, you can simply select an MDX cube by using the MDX Cube Catalog or Select Cube dialog box, as described in [Importing MDX cubes, page 80](#).

When an MDX cube is imported into MicroStrategy, by default, a MicroStrategy MDX cube is created that maps to the definition of the source cube in the MDX cube source. Intelligence Server automatically creates new attributes, metrics, hierarchies, and other objects that reflect the data and levels of the imported MDX cube. Although these objects, referred to as managed objects, are part of the project, they are not related to the existing project schema and schema objects (see [About managed objects](#) above).

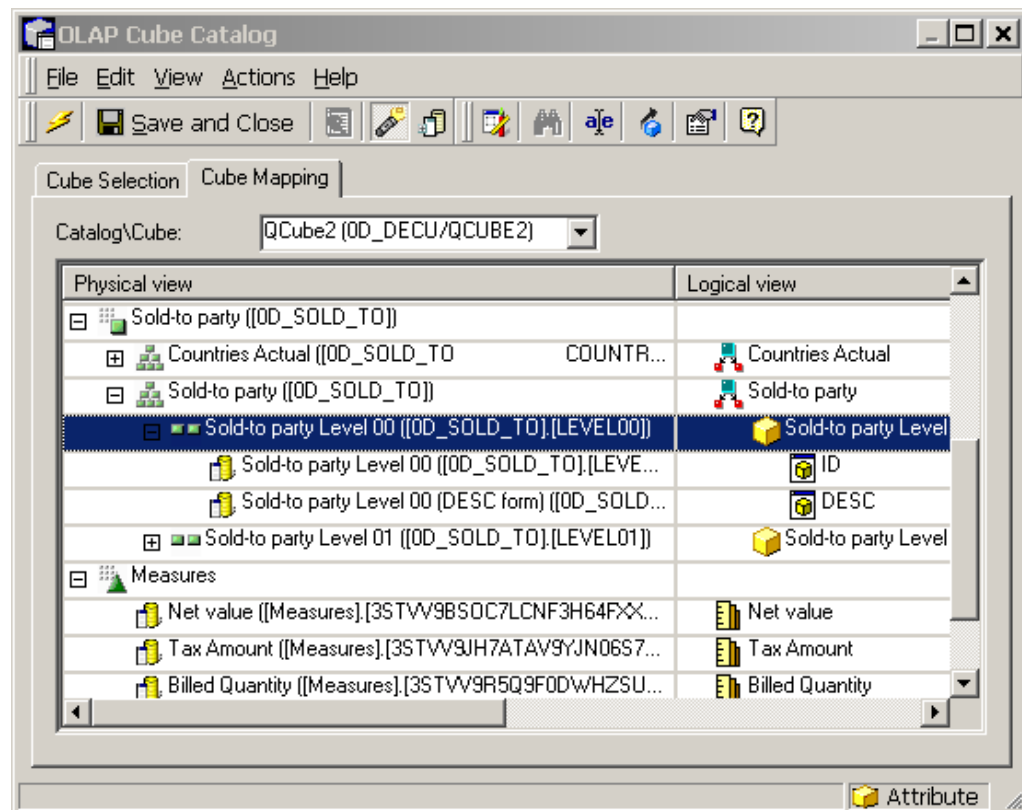
For example, within a given project, a new managed object named Year has no relation to a Year attribute that is mapped to relational data. A new schema is created for each MDX cube source database instance used in a MicroStrategy project. If you plan to use your MDX cube source as its own system of data which does not relate to any other data sources within the MicroStrategy project, managed objects provide a quick way to integrate your data into MicroStrategy.

However, if you plan to create relations between your MDX cube source data to data from a different data source included in the MicroStrategy project, you can map MDX cube data to existing attributes in the project. This allows data to be joined across sources in Report Services documents, which

ensures that a consistent logical model is maintained. Mapping MDX cube data to existing attributes can also facilitate the use of MicroStrategy features such as security filters. For more information on the benefits of mapping MDX cube data to project attributes, see [Mapping MDX cube data to project attributes, page 103](#).


All MDX cube mapping tasks can be completed using the Cube Mapping tab in the MDX Cube Catalog shown below.

After you have imported an MDX cube, you can perform the same mapping tasks available in the Cube Mapping tab of the MDX Cube Catalog by editing the MDX cube with the MDX Cube Editor. To use the MDX Cube Editor, right-click an MDX cube in Developer and select **Edit**.




Mapping MDX cubes in MicroStrategy includes mapping and configuring objects in MicroStrategy as well as supporting various objects and data structures from MDX cube sources within MicroStrategy. Before you can perform any of the MDX cube mapping requirements and techniques listed below, you need to import MDX cubes from your MDX cube source into MicroStrategy. For steps to import MDX cubes, see [Importing MDX cubes, page 80](#). Below is a list of MDX cube mapping requirements as well as various MDX cube mapping techniques you can perform (with references to

sections for further instruction) from the Cube Mapping tab of the MDX Cube Catalog:


- An ID form must be mapped for each attribute. MicroStrategy uses the ID form to map to the primary *ID column* for an attribute, which contains attribute element identification codes.
- By default, only the ID and DESC forms are displayed and automatically mapped for each attribute. MicroStrategy uses the DESC form to map to the primary *description column* for an attribute, which commonly contains descriptive information for the attribute. From the **View** menu, select **Display All Columns** to display or hide the additional attribute forms. Once displayed, you can then map these additional forms as required.
- The MDX cube structure within the MDX cube source is represented in the **Physical View** left-hand column. This column is primarily to display the structure of the MDX cube data within the MDX cube source, as most manipulations can only be made on the related MicroStrategy objects. However, there are MDX cube source objects that can be manipulated in ways that affect how their related MicroStrategy objects are defined:
 -  Dimensions in MDX cube sources are not directly mapped to MicroStrategy objects. However, the levels within a dimension in an MDX cube source are mapped to hierarchies and attributes in MicroStrategy.

The MDX cube source dimension is also where you define whether to preserve the order of MDX cube source data mapped to attribute elements in MicroStrategy. For information on preserving attribute element order for MDX cubes, see [Preserving attribute element orders from MDX cube sources, page 113](#).

-  MDX cube source objects mapped to MicroStrategy hierarchies can be manipulated in the following ways:
 - MicroStrategy's data modeling conventions do not support unbalanced or ragged hierarchies. However, your MDX cube source may support and contain unbalanced or ragged hierarchies. To support these types of hierarchies in MicroStrategy, you must define these hierarchies as unbalanced or ragged (see [Defining unbalanced and ragged hierarchies, page 120](#)).
 - Report designers can include MicroStrategy hierarchies directly on MDX cube reports. The set of attributes that are displayed in place of a hierarchy on an MDX cube report by default is defined for MDX cube source objects mapped to MicroStrategy hierarchies (see [Displaying hierarchies on MDX cube reports, page 122](#)).

-  MDX cube source objects mapped to MicroStrategy attribute forms are integrated into MicroStrategy as a string of characters.

You can modify the data type used to map MDX cube data to attribute forms. This allows the MDX cube data to be correctly represented in MicroStrategy and facilitates the grouping of related attributes as the same attribute in a Report Services document (see [Defining column data types for MDX cube data, page 109](#)).

-  SAP BW variables are mapped to MicroStrategy prompts. If an MDX cube contains key date variables, you must define them as key date variables to distinguish them from characteristic variables on date.

For information on mapping SAP BW variables to MicroStrategy prompts and defining key date variables, see [Mapping SAP BW variables to MicroStrategy prompts, page 124](#).

For information on how SAP BW variables are converted into MicroStrategy prompts, see [Converting SAP BW variables into MicroStrategy prompts, page 16](#).

- An MDX cube's equivalent structure and objects in MicroStrategy are represented in the **Logical View** right-hand column, with the standard MicroStrategy symbols for hierarchies, attributes, metrics, and so on.

For MicroStrategy objects, you can perform the following manipulations by right-clicking the object in the Logical view column and using the various options:

- **Edit** the attribute, metric, or prompt. This option opens the Attribute Editor to edit attributes, the Metric Editor to edit metrics, or the Prompt Generation Wizard to edit prompts.
- **Rename** the attribute, metric, prompt, or hierarchy. If you want to map the object to an existing MicroStrategy object, you should use the Map feature described below rather than this Rename feature.

During the import of MDX cubes, you can choose to have the MicroStrategy objects' names synchronized with their associated objects in the MDX cube source. This synchronization option is described in the steps [To import MDX cubes, page 82](#).

- **Map** the attribute, metric, or prompt to an existing attribute, metric, or prompt in the MicroStrategy project.
 - Attributes mapped to MDX cube data can be mapped to attributes in the MicroStrategy project that are part of the relational schema. For information on how to map MDX cube data to project

attributes and the benefits of this type of setup, see [Mapping MDX cube data to project attributes, page 103](#)).

- Metrics and prompts in MDX cubes can only be mapped to other managed object metrics and prompts that are mapped to MDX cube source data. For information on how one prompt can be mapped to SAP BW variables in multiple MDX cubes, see [Mapping SAP BW variables to MicroStrategy prompts, page 124](#).
- Check the **Properties** of the attribute, metric, prompt, or hierarchy. The properties displayed when accessed from the Logical View column relate to MicroStrategy-specific properties such as the access control list, owner, and long description for the object.
- Once an MDX cube is imported into MicroStrategy, you can use MicroStrategy's Data Mining Services features to perform predictive analysis on your MDX cube data. For steps on how to include predictive analysis with your MDX cube data, refer to the *Advanced Reporting Guide*.

Once an MDX cube is mapped, it can be used to build reports and documents in MicroStrategy. For information on creating MDX cube reports, see [Chapter 4, Reporting on MDX Cubes](#).

Shared MDX cube objects

The data and objects that are shared in your MDX cube source are also shared as attributes in MicroStrategy projects. This maintains a consistent experience across all related MDX cubes imported into MicroStrategy.

For example, you import Cube1 and Cube2 into MicroStrategy. Both MDX cubes share data for year and category. In MicroStrategy, year data is mapped to an attribute named Year, and category data is mapped to an attribute named Category. If you change the name of the Year attribute in Cube1, this change is also reflected in Cube2. If you change the name of the Category attribute in Cube2, this change is also reflected in Cube1. Updating an attribute across all MDX cubes that share the attribute is handled automatically by MicroStrategy.

Attributes that are shared by multiple MDX cubes can have an effect on how and when attributes are imported for an MDX cube. For information on how these shared attributes can affect import behavior, see [Shared attribute effects on import behavior, page 86](#).

Sharing metrics between MDX cubes

By default, each MDX cube creates its own metric objects to support the integration of the data into MicroStrategy. However, you can manually map metrics for multiple MDX cubes to the same metric so that they share a single metric. Sharing metrics between MDX cubes is required if you want to allow MDX cube reports to be able to switch the MDX cube that is used as the source of its data, as described in [Switching the MDX cube for an MDX cube report, page 153](#).

You can use the steps below to share metrics between MDX cubes.

Prerequisites

- At least two MDX cubes must be imported into your MicroStrategy project. These MDX cubes must have data that can be mapped to the same metric. For more information on importing MDX cubes into MicroStrategy, see [Importing MDX cubes, page 80](#).

To share metrics between MDX cubes




- 1 In MicroStrategy Developer, log in to a project that is connected to an MDX cube source.
- 2 If you are using read only mode for the project, from the **Schema** menu, clear the **Read Only Mode** option to switch to edit mode.

Only one user can be editing a project at a given time. Therefore, if someone else is modifying the project, you cannot use the MDX Cube Catalog.
- 3 Right-click an MDX cube and select **Edit**. The MDX Cube Editor opens.
- 4 In the **Physical View** column, expand **Measures** to view the metrics for the MDX cube.
- 5 Right-click a metric and select **Map**. The Select Destination Object dialog box opens.
- 6 Navigate to a different MDX cube that has similar metric data. Select the metric and click **Open**. The metrics are now mapped to the same object and you are returned to the MDX Cube Editor.

- 7 You can map additional metrics to share any other relevant metrics between the MDX cubes. Once all relevant metrics are shared, from the MDX Cube Editor, click **Save and Close** to save your changes.

Best practices for mapping MDX cubes

Provided below is a list of best practices and tips that you can use while mapping MDX cubes to MicroStrategy objects:

- Rather than mapping all additional attribute forms one-by-one, you can have MicroStrategy automatically map all additional attribute forms for you. From the **View** menu, select **Display All Columns** to display all the additional attribute forms. From the **Edit** menu select **Map all attribute forms** to automatically map all additional attribute forms. You can modify these automatic mappings as required.
 -  The Map all attribute forms option maps attribute data to managed object attributes. You must manually map all attribute forms if you are mapping your MDX cube data to project attributes.
- To display the entire MDX cube structure or only the top-level structure, from the **View** menu, you can use the **Expand All**  or **Collapse All**  options.
- To display or hide the SAP BW terms for each object, from the **View** menu, select **Show Technical Names**. The Show Technical Names option applies to SAP BW MDX cubes only.
- After you have imported an MDX cube, you can perform the same mapping tasks available in the Cube Mapping tab of the MDX Cube Catalog by editing the MDX cube with the MDX Cube Editor. To use the MDX Cube Editor, right-click an MDX cube in Developer and select **Edit**.
- For best practices to maintain object consistency between the two projects, see [Maintaining MDX cubes between multiple projects, page 94](#).

Mapping MDX cube data to project attributes

After you have imported MDX cubes, you can use the automatically generated managed object attributes to define the levels of your MDX cube data. Alternatively, you can map MDX cube data to existing attributes in the MicroStrategy project that are part of the relational schema. Mapping MDX

cube data in this way replaces the managed objects that are used to represent MDX cube data with attributes in the MicroStrategy project. Mapping MDX cube data to attributes in a MicroStrategy project that are part of a relational schema has the following benefits:

- Report designers can integrate the logical model of the project with the data in imported MDX cubes, thus creating a relation between the two sets of data. Data can then be joined across sources within a Report Services document. For example, if an MDX cube report and a standard report both use the same Year attribute, then Year can be used to group the data within a document.
- Administrators can search for dependents and manage access control lists (ACLs) for attributes that map both to the data warehouse and an MDX cube source.
- MicroStrategy security filters can be applied to attributes in MDX cube reports. For example, you can map an MDX cube level to the Year attribute in your project. If a user with a security filter on Year runs the MDX cube report that contains Year, the security filter on Year is applied.
- With the MicroStrategy MultiSource Option, MDX cube reports can be used to filter other standard reports in MicroStrategy. For information on filtering standard reports with MDX cube reports, see [Using MDX cube reports to filter other reports, page 183](#).

The following sections provide example scenarios of MDX cubes and how they can be mapped to attributes that are part of a relational schema:

- [Example 1: Unmapped MDX cube, page 105](#)
- [Example 2: Partially mapped MDX cube, page 106](#)

Metrics and prompts mapped to MDX cube data cannot be mapped to objects that are part of a relational schema. These metrics and prompts can only be mapped to managed object metrics and prompts that are mapped to MDX cube source data. For example, three MDX cubes can share the same managed object metric named Revenue. However, none of these metrics can share the same object as a Revenue metric mapped to relational data in the project. Sharing metrics between MDX cubes is required if you want to allow MDX cube reports to be able to switch the MDX cube that is used as the source of its data, as described in [Switching the MDX cube for an MDX cube report, page 153](#).

MDX cubes connected to SAP BW as an MDX cube source can contain variables. These variables are converted into prompts when imported into MicroStrategy (see [Converting SAP BW variables into MicroStrategy prompts, page 16](#) for conversion information). If multiple MDX cubes

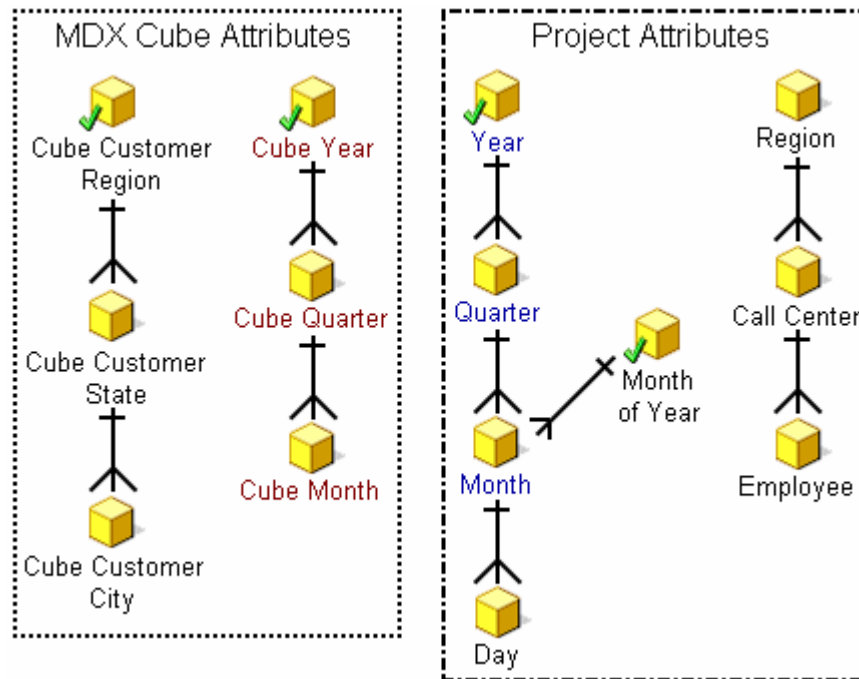
contain the same variable, one MicroStrategy prompt can be mapped to more than one variable across MDX cubes. This enables a prompt to be displayed and answered only once when executing a Report Services document that uses these MDX cubes. For steps to map one prompt to variables in multiple MDX cubes, see [Using one prompt in documents for variables in separate MDX cubes, page 128](#).

Example 1: Unmapped MDX cube

You can map managed object attributes for your MDX cubes instead of using project attributes. This feature allows you to quickly start creating reports for your MDX cube data.

The drawback of an MDX cube mapped only to managed objects is that you cannot create a relation between your MDX cube data and other data in your project connected to a data source other than your MDX cube source. Since this relation is not created, you cannot join data from these different sources in a Report Services document and you cannot support project security filters in MDX cube reports.

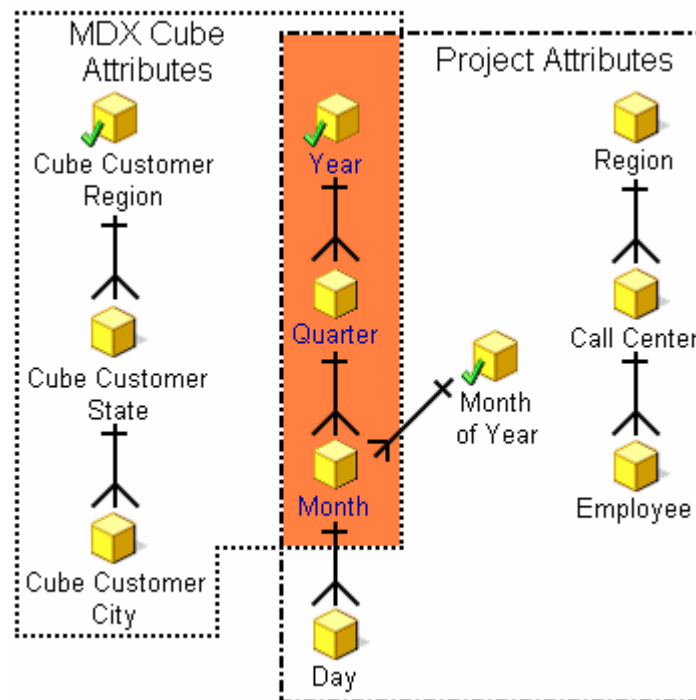
The diagram below shows two logical models. The one on the left exists in a specific MDX cube, and the one on the right exists in a MicroStrategy project. Although both models have a Time hierarchy, none of the individual attributes are shared.



Example 2: Partially mapped MDX cube

After an MDX cube source has been included in MicroStrategy as an MDX cube, you can map the attributes within the MDX cube to existing project attributes.

The example shown in the diagram below also shows two logical models. The difference between this example and the example above is that the MDX cube has been partially mapped so that it shares the attributes Year, Quarter, and Month. With this technique, you can create a Report Services document that contains Year, Quarter, and Month information for both your data warehouse and MDX cube source. In addition, any security filters for Year, Quarter, and Month are applied to MDX cube reports that include these mapped attributes.



The dimensions of MDX cubes are always shared. Therefore, when a level is mapped, that change applies to all the MDX cubes that share that dimension. In this case, changes to the Time dimension apply to MDX cubes in the project that contain this dimension.

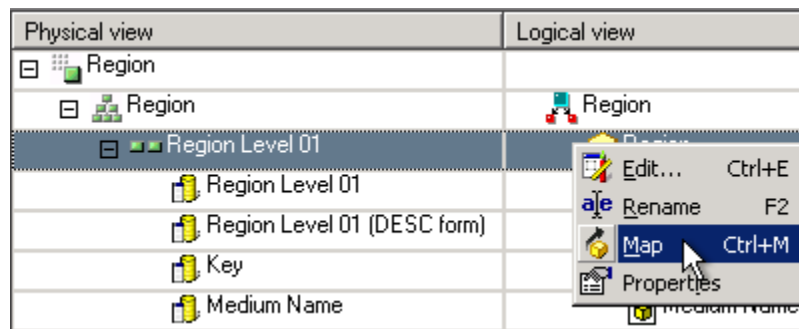
Best practices for mapping MDX cubes to project attributes

The best practices listed below relate to mapping MDX cube data to project attributes. For general best practices of using the MDX Cube Catalog to map MDX cube data, see [Best practices for mapping MDX cubes, page 103](#).

Before you can perform any of the MDX cube mapping techniques listed below, you need to import MDX cubes from your MDX cube source into MicroStrategy. For steps to import MDX cubes, see [Importing MDX cubes, page 80](#).

When mapping MDX cube data to project attributes, you can take advantage of the best practices techniques listed below:

- You can map MDX cube data to a project attribute by right-clicking the attribute in the Logical view column and selecting **Map**, as shown below.




You can then browse to the attribute within your relational project schema to map to the MDX cube data.

- When mapping SAP BW data to project attributes, you should map the SAP BW characteristics to project attributes rather than mapping the levels of SAP BW hierarchies. For example, in the scenario shown below

you must map the Region project attribute to the level in the highlighted Region characteristic rather than to a level within the Reg Org hierarchy.

Physical view	Logical view
Region	
Reg Org	Reg Org
Reg Org Level 00	Reg Org Level 00
Reg Org Level 01	Reg Org Level 01
Reg Org Level 02	Reg Org Level 02
Region	Region
Region Level 01	Region
Region Level 01 (DESC form)	<Unmapped>
Key	DESC
Name	ID
	Name

The concept of SAP BW characteristics versus hierarchies is discussed in [Importing levels and suffixes for characteristics, page 85](#).

- To map MDX cube data to the ID form of a project attribute, you must adhere to the following guidelines:
 - The ID form of the project attribute must be mapped to the column in your MDX cube source you have created to relate the two systems of data. The columns must share the same data type. For example, the Key form in SAP BW can use the same numeric data type standards as is used most commonly for MicroStrategy attribute ID forms.
 -  The Key form is not displayed by default. Within the MDX Cube Catalog, from the **View** menu, select **Display All Columns** to display all available forms for an MDX cube.
 - Once you map the correct column to the ID form of the project attribute, you must define the column data type of your MDX cube data as the same data type used for the attribute's ID form. This is because MDX returns all attribute data for MDX cube sources as strings by default. For information on and steps for defining column data types for MDX cube data, see [Defining column data types for MDX cube data, page 109](#).
- You can map the columns to project attributes either when an MDX cube is first imported or at a later time. It is recommended that you immediately perform this mapping during the initial import to maintain a consistent reporting environment. This also prevents maintenance issues such as having to modify MDX cube reports when MDX cubes are modified after the reports are created.

If you map a column to the incorrect project attribute, do one of the following:

- Once you save your changes, you can unmap the column from the project attribute.
- You can close the MDX Cube Catalog without saving your changes. The mapping is not changed for the column. Be aware that any other changes made after the last time you saved your changes will also be lost.
- If you need to unmap a column that was previously mapped to a project attribute, it is recommended to unmap the column and then save immediately after performing this unmapping. This ensures that the unmapping is processed correctly. You can then open the MDX cube and perform any additional mapping or unmapping of columns as needed.


Defining column data types for MDX cube data


You can define the column data type that is applied to a column of MDX cube data mapped to an attribute. This allows the MDX cube data to be correctly represented in MicroStrategy and facilitates the following:

- Group related attributes, from MDX cubes as well the data warehouse, as the same attribute in a Report Services document. This is discussed in more detail in this section.
- Use value prompts to qualify on your MDX cube data. If you plan to use a value prompt (date, numeric, text, or big decimal) to qualify on an attribute form imported from an MDX cube source, you must define the attribute form with a data type that matches the value prompt. For example, an Integer column data type can be qualified on using a numeric value prompt. Use the procedure [To define column data types for MDX cube data](#) below to define column data types.
- Support date data from your MDX cube source in MicroStrategy. This enables you to filter and qualify on your MDX cube source date data using static and dynamic date qualifications. For background information and steps to support date data from your MDX cube source in MicroStrategy, see [Supporting MDX cube source date data in MicroStrategy, page 117](#).

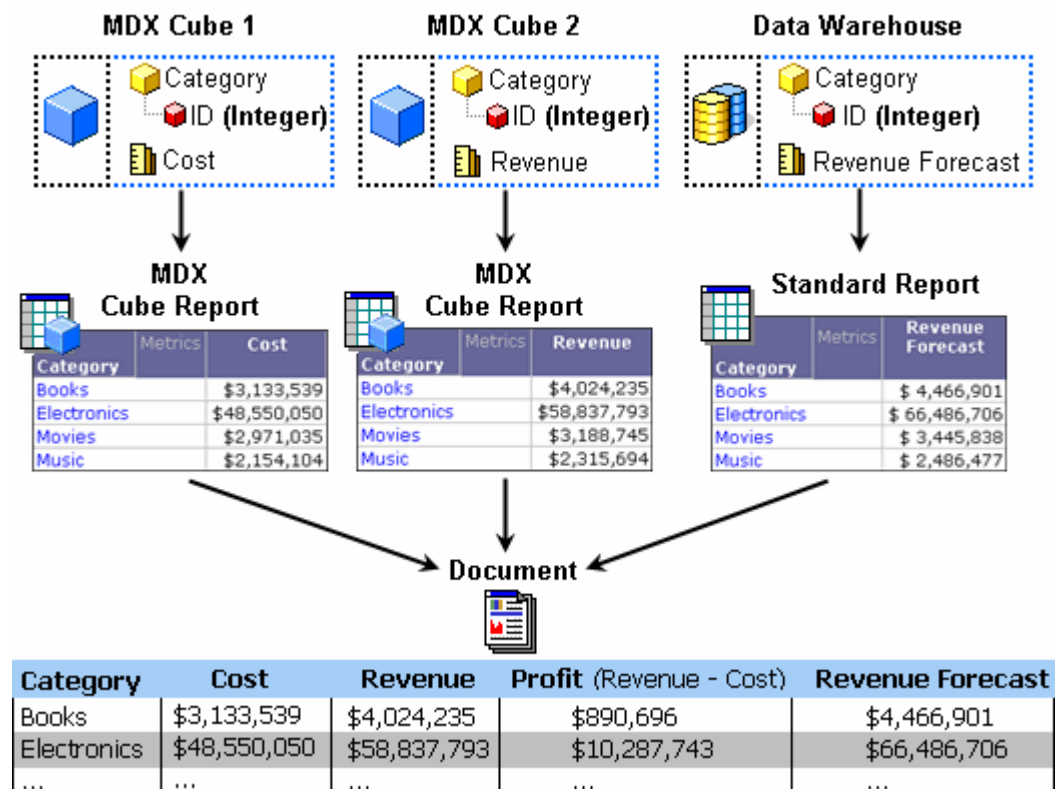
When MDX cube data is mapped to MicroStrategy objects, MicroStrategy retrieves the column data type through MDX. In the case of MDX cube data that is mapped to attributes, the columns are often returned as a string of

characters. This can be the case even with ID columns of data that are commonly of a numeric type such as integer.

 The column of data that is automatically mapped to attribute ID columns in MDX cubes is returned as a character string. The data type of this column cannot be defined to anything other than the default data type because it is not well suited for other data types. If you want to map a column with a numeric or other data type to the ID form of an attribute, you should use a different form such as the key form.

 MDX cube data that is mapped to MicroStrategy metrics is automatically converted to a numeric data type and thus does not need its column data type to be manually set.

For example, you have two MDX cubes that map data to a Category attribute in MicroStrategy. The ID attribute form for Category is returned as a string by default. However, you know that its associated MDX cube column is of type integer and set the data type accordingly in each MDX cube. You can then create MDX cube reports for these MDX cubes. By setting the Category ID attribute form to read the MDX cube data as an integer, you can then include the two MDX cube reports as datasets of a Report Services document and group the Category attribute data. You can also add a standard report, drawing data from a data warehouse, as a dataset of the document to combine its data on the same display, as shown below.



Notice that the Category ID form is defined as the same data type (Integer) in each data source. Without this setup, the Category data from each data source cannot be displayed on a single document.

In addition to displaying data from different data sources on the same document, defining data types lets you perform calculations on metrics from different data sources. In the document shown above, the Profit metric is calculated by creating a calculated expression in the document that subtracts the Cost metric of MDX Cube 1 from the Revenue metric of MDX Cube 2.

The image above shows a scenario of displaying both MDX cube data and data from a data warehouse on one document. If you only include MDX cube data on a document, you can use the default attributes created by MicroStrategy when importing your MDX cubes. However, to include MDX cube data and data from the data warehouse together on a document, you must map your MDX cube data to attributes that are part of the project's relational schema (see [Mapping MDX cube data to project attributes, page 103](#)).

The procedure below describes how to define column data types for MDX cube data. After performing this procedure you can create MDX cube reports that can be displayed on a document with other MDX cube reports and standard reports. For information on how to create a document that displays data from multiple data sources, see the *Advanced Documents* chapter of the *MicroStrategy Document Creation Guide*. Information on calculated expressions and how to create them is covered in *Designing and Creating Documents* chapter of the *MicroStrategy Document Creation Guide*.

Before you can define column data types for MDX cube data, you need to import MDX cubes from your MDX cube source into MicroStrategy. For steps to import MDX cubes, see [Importing MDX cubes, page 80](#).

To define column data types for MDX cube data


You can define column data types for MDX cube data when you are mapping MDX cube data to MicroStrategy objects. The following procedure assumes you are defining column data types as part of the mapping procedure for an MDX cube using the MDX Cube Catalog. However, you can define column data types as a later modification using the MDX Cube Editor.

- 1 In MicroStrategy Developer, log in to a project that is connected to an MDX cube source.

- 2 If you are using read only mode for the project, from the **Schema** menu, clear the **Read Only Mode** option to switch to edit mode.

Only one user can be editing a project at a given time. Therefore, if someone else is modifying the project, you cannot use the MDX Cube Catalog.

- 3 From the **Schema** menu, select **MDX Cube Catalog**.
 - If the project connects to only one MDX cube source, the MDX Cube Catalog opens.
 - If the project connects to more than one MDX cube source, the Database Instance dialog box opens. From the **Select the Database Instance** drop-down list, select an MDX cube source database instance and click **OK**. The MDX Cube Catalog opens.
- 4 Select the **Cube Mapping** tab.
- 5 From the **Catalog\Cube** drop-down list, select an MDX cube to map to MicroStrategy objects. The MDX cube data is displayed.
- 6 In the **Physical view** column, expand the MDX cube data until you find the MDX cube column data for which to manually set the data type.
- 7 If you want to include MDX cube data and data from the data warehouse together on a document, you must map your MDX cube data to attributes that are part of the project's relational schema (see [Best practices for mapping MDX cubes to project attributes, page 107](#)). Using the automatically created managed object attributes enables you to combine data from multiple MDX cubes on the same document, but no data from the data warehouse can be included.
- 8 From the **Logical view** column, right-click the MicroStrategy object mapped to the MDX cube column data and select **Data Type**. The Column Editor — Definition dialog box opens.
- 9 Clear the **Use default from source** check box.
- 10 From the **Data type** drop-down list, select which data type to map the MDX cube data as in MicroStrategy.



The Integer data type is commonly used as an ID form data type.
- 11 Depending on the data type selected, specify the byte length, precision, and scale for the data type.

12 Click **OK** to save your changes and return to the MDX Cube Catalog.

To modify the attribute form format type

13 In the MDX Cube Catalog, right-click the attribute mapped to the column you defined the data type for, and select **Edit**. The Attribute Editor opens.

14 From the **Forms** tab, in the **Attribute forms** pane, select the form you defined the data type for, and click **Modify**. The Modify Attribute Form dialog box opens.

15 In the **Form format** area, click the **Type** drop-down list, and select a form format that matches the data type you selected for the column. For example, if you chose Integer as the data type, select Numeric as the form format. Another example is if you chose Date as the data type, select Date as the form format.

16 Click **OK**.

17 If an inconsistent data type warning message is displayed, click **Yes**.

18 Click **Save and Close** to save your changes to the attribute and return to the MDX Cube Catalog.

19 Click **Save and Close** to save your changes to the attribute and close the MDX Cube Catalog.

Resolving incompatible data type errors

Defining your MDX cube data with an incompatible data type can cause errors to occur when running a MicroStrategy MDX cube report. An MDX cube report that includes MDX cube data mapped to an incompatible data type fails and no data is returned. When an MDX cube report fails for this reason, an error message is displayed that identifies the data that has been mapped to an incompatible data type.

Preserving attribute element orders from MDX cube sources

When data is integrated from your MDX cube source to MicroStrategy, the order of data mapped to attribute elements in MicroStrategy does not always reflect the same order of the data in your MDX cube source. You can ensure that the order of your data as it exists in your MDX cube source is preserved when it is integrated into MicroStrategy.

For example, a hierarchy in an MDX cube source includes data on regions, cities, and employees. This data is integrated into MicroStrategy as Region, City, and Employee attributes. This data is then displayed on an MDX cube report.

The two reports shown below show two different scenarios for how the data can be displayed. The report on the left uses the default order as defined by MicroStrategy. This order is based on the ID columns returned for the MDX cube data. The report on the right uses the order that is present in the MDX cube source. This preserves the same view of data in MicroStrategy as it exists in the MDX cube source, as shown below:

Orders are not preserved

Region	City	Employee	Metrics
Northeast	Boston	De Le Torre	
		Kieferson	
		Sonder	
	New York	Kelly	
		Sawyer	
		Yager	
Mid-Atlantic	Washington, DC	Bernstein	
		Folks	
		Hollywood	
	Charleston	Brown	
		Corcoran	
		Ingles	
		Smith	
		Young	
Southeast	Atlanta	Benner	
	Miami	McClain	
		Lynch	
		Strome	
Central	Milwaukee	Gale	
		Torrison	
		Zemlicka	
	Fargo	Ellerkamp	
South	New Orleans	Connor	
		Nelson	
	Memphis	Pierce	
Northwest	San Francisco	Becker	
	Seattle	Hall	
		Gedot	
Southwest	San Diego	Bates	
		Bell	
		Johnson	
		Schafer	
	Salt Lake City	Hunt	

Orders are preserved

Region	City	Employee	Metrics
Central	Fargo	Ellerkamp	
	Milwaukee	Gale	
		Torrison	
		Zemlicka	
Mid-Atlantic	Charleston	Brown	
		Corcoran	
		Ingles	
		Smith	
	Washington, DC	Young	
		Bernstein	
		Folks	
		Hollywood	
Northeast	Boston	De Le Torre	
		Kieferson	
		Sonder	
	New York	Kelly	
		Sawyer	
Northwest	San Francisco	Becker	
	Seattle	Hall	
		Gedot	
South	Memphis	Pierce	
	New Orleans	Connor	
		Nelson	
Southeast	Atlanta	Benner	
	Miami	McClain	
		Lynch	
Southwest	Salt Lake City	Strome	
		Hunt	
	San Diego	Bates	
		Bell	
		Johnson	
		Schafer	

Preserving the order of data as it exists in your MDX cube source can be helpful for various reasons. For example, this can help support financial balance sheet analysis in which you always want to see your accounts receivable information above your accounts payable information.

To preserve the order of data as it exists in your MDX cube source, you must define dimensions within MDX cubes to return the order of data when integrated into MicroStrategy.



Dimensions in MDX cube sources are not directly mapped to MicroStrategy objects. However, the levels within a dimension in an MDX cube source are mapped to hierarchies and attributes in MicroStrategy.

By default, the order of data is not integrated into MicroStrategy. This is because the need to preserve the order of data is commonly unnecessary and only used on a case-by-case basis, and integrating this order information requires metadata space. Steps to define dimensions in MDX cubes to preserve the order of MDX cube data are provided in the procedure below.

Prerequisites

- Before you can preserve the order of MDX cube source data in MicroStrategy, you need to import MDX cubes from your MDX cube source into MicroStrategy. For steps to import MDX cubes, see [Importing MDX cubes, page 80](#).

To preserve and retrieve attribute element orders for MDX cube sources

- 1 In MicroStrategy Developer, log in to a project that is connected to an MDX cube source.
- 2 If you are using read only mode for the project, from the **Schema** menu, clear the **Read Only Mode** option to switch to edit mode.

Only one user can be editing a project at a given time. Therefore, if someone else is modifying the project, you cannot use the MDX Cube Catalog.

- 3 From the **Schema** menu, select **MDX Cube Catalog**.
 - If the project connects to only one MDX cube source, the MDX Cube Catalog opens.
 - If the project connects to more than one MDX cube source, the Database Instance dialog box opens. From the **Select the Database**

Instance drop-down list, select an MDX cube source database instance and click **OK**. The MDX Cube Catalog opens.

- 4 On the **Cube Mapping** tab, from the **Catalog\Cube** drop-down list, select the MDX cube to integrate the order of data for into MicroStrategy. The MDX cube data is displayed.
- 5 You must save the MDX cube before you can modify the properties of a dimension. From the **File** menu, select **Save**.
- 6 In the **Physical view** column, right-click a dimension to integrate the order for all data within the dimension into MicroStrategy, and then select **Properties**. The Properties[*dimension name*] dialog box opens.
- 7 On the **Hierarchies** tab, select the **The order of the hierarchy nodes should be preserved from the source** check box. Selecting this check box marks this dimension to have the order of its data integrated into MicroStrategy.
- 8 To retrieve the order of data from the MDX cube source, click **Retrieve hierarchy structure**. If the order of data in your MDX cube source changes, you can update the order for all dimensions in an MDX cube that are defined to have the order of data integrated into MicroStrategy (see [Updating the order of attribute elements, page 116](#)).
- 9 Click **OK** to save your changes to the dimension and return to the MDX Cube Catalog.
- 10 Click **Save and Close** to save your changes and close the MDX Cube Catalog.

The order of data is now integrated into MicroStrategy. However, MDX cube reports are not sorted using this order by default. A report designer must define an MDX cube report to sort using the order integrated from the MDX cube source, as described in [Sorting on attribute element orders from MDX cube sources, page 179](#).

Updating the order of attribute elements

You can select to preserve the order of data that is mapped to attribute elements in MicroStrategy for various MDX cubes. Once you have done so, you can update these orders for MDX cubes in MicroStrategy when they are changed in your MDX cube source. The steps to update the order of data in MicroStrategy to match any changes in the MDX cube source are provided in the procedure below.

Prerequisites

- The order of data in an MDX cube source is only updated for dimensions that are defined to integrate the order of data into MicroStrategy. For information on defining dimensions to integrate their order of data into MicroStrategy, see [Preserving attribute element orders from MDX cube sources, page 113](#).

To update the order of attribute elements

- 1 In MicroStrategy Developer, log in to a project that is connected to an MDX cube source.
- 2 If you are using read only mode for the project, from the **Schema** menu, clear the **Read Only Mode** option to switch to edit mode.

Only one user can be editing a project at a given time. Therefore, if someone else is modifying the project, you cannot use the MDX Cube Catalog.
- 3 From the **Schema** menu, select **MDX Cube Catalog**.
 - If the project connects to only one MDX cube source, the MDX Cube Catalog opens.
 - If the project connects to more than one MDX cube source, the Database Instance dialog box opens. From the **Select the Database Instance** drop-down list, select an MDX cube source database instance and click **OK**. The MDX Cube Catalog opens.
- 4 On the **Cube Selection** tab, in the **Selected Cubes** area, browse to and right-click an MDX cube to update the order of its data in MicroStrategy, and then select **Update hierarchy structure**. The order of data is updated in MicroStrategy for all dimensions that are defined to integrate the order of data into MicroStrategy.
- 5 Click **Save and Close** to save your changes and close the MDX Cube Catalog.

Supporting MDX cube source date data in MicroStrategy

You can maintain a consistent date format when mapping date data you have stored in your MDX cube source into MicroStrategy. This enables you to view

the data as its intended date data type, as well as filter and qualify on date data using static and dynamic date qualifications.

Date data represents a given day using various formats that include the month, day, and year of a given day. To support MDX cube source date data in MicroStrategy, you must define the column data type for the column of data as well as define a MicroStrategy VLDB property, which is covered in the following procedure.

Before you can support MDX cube source date data in MicroStrategy, you need to import MDX cubes from your MDX cube source into MicroStrategy. For steps to import MDX cubes, see [Importing MDX cubes, page 80](#).

To support MDX cube source date data in MicroStrategy

- 1 In MicroStrategy Developer, log in to a project that is connected to an MDX cube source.
- 2 If you are using read only mode for the project, from the **Schema** menu, clear the **Read Only Mode** option to switch to edit mode.

Only one user can be editing a project at a given time. Therefore, if someone else is modifying the project, you cannot use the MDX Cube Catalog.
- 3 From the **Schema** menu, select **MDX Cube Catalog**.
 - If the project connects to only one MDX cube source, the MDX Cube Catalog opens.
 - If the project connects to more than one MDX cube source, the Database Instance dialog box opens. From the **Select the Database Instance** drop-down list, select an MDX cube source database instance and click **OK**. The MDX Cube Catalog opens.
- 4 Select the **Cube Mapping** tab.
- 5 From the **Catalog\Cube** drop-down list, select the MDX cube to define as a date data type. The MDX cube data is displayed.

- 6 In the **Physical view** column, expand the MDX cube data until you find the MDX cube column data for which to define the data type.



The column of data that is automatically mapped to attribute ID columns in MDX cubes is returned as a character string. The data type of this column cannot be defined to the Date data type.

- 7 From the **Logical view** column, right-click the MicroStrategy object mapped to the MDX cube column data and select **Data Type**. The Column Editor — Definition dialog box opens.
- 8 Clear the **Use default from source** check box.
- 9 From the **Data type** drop-down list, select **Date**.
- 10 Click **OK** to save your changes and return to the MDX Cube Catalog.

To modify the attribute form format type

- 11 In the MDX Cube Catalog, right-click the attribute mapped to the column you defined as a date data type, and select **Edit**. The Attribute Editor opens.
- 12 From the **Forms** tab, in the **Attribute forms** pane, select the form you defined as a date data type, and click **Modify**. The Modify Attribute Form dialog box opens.
- 13 In the **Form format** area, click the **Type** drop-down list, and select **Date**.
- 14 Click **OK**.
- 15 If an inconsistent data type warning message is displayed, click **Yes**.
- 16 Click **Save and Close** to save your changes to the attribute and return to the MDX Cube Catalog.
- 17 Click **Save and Close** to save your changes to the MDX cube and exit the MDX Cube Catalog.

To modify the MicroStrategy VLDB property

- 18 Right-click the project that contains your MDX cubes and select **Project Configuration**. The Project Configuration Editor opens.
- 19 Expand the **Project definition** category, and then select **Advanced**.

- 20 From the **Project-Level VLDB settings** area, click **Configure**. The VLDB Properties Editor for the project opens.
- 21 Expand the **MDX** folder, and then select **Format for date/time values coming from data source**.
- 22 The default date format for MDX cube sources is displayed on the right. If this date format does not match the date formats used in your MDX cube source, clear the Use default inherited value check box. You can then enter the date format used in your MDX cube source.

You can also define the date formats used in an individual MDX cube report. If you define this VLDB property for an MDX cube report, the definition for the MDX cube report takes precedence over the definition for the project. However, the definition of the VLDB property for the project is applied to all MDX cube reports that select to use the default inherited date format.
- 23 Click **Save and Close** to save your changes and return to the Project Configuration Editor.
- 24 Click **OK** to save your changes and close the Project Configuration Editor.

Report designers can now begin to create MDX cube reports that include the date data defined for the attribute form mapped to the MDX cube source date data. For information on how date data can be filtered on using static and dynamic date filters, see [Filtering with static or dynamic date qualifications, page 167](#).

Defining unbalanced and ragged hierarchies

By default, all hierarchies of an MDX cube are treated as balanced hierarchies. However, if you know that the structure of a hierarchy is unbalanced or ragged, you must set the hierarchy's properties to reflect its structure.

The terms balanced, unbalanced, and ragged are used to describe the different characteristics of hierarchical sets of data, as described below:

- **Balanced** hierarchies have an equal number of levels in each branch of the hierarchy. For example, in a Product hierarchy that includes Category, Subcategory, and Item, each branch descends to a particular item, which is at the lowest level.

- **Unbalanced** hierarchies have at least one branch that does not descend to the lowest level. For example, in a Time hierarchy that includes Year, Quarter, and Month, one branch might only have data down to the Quarter level.
- **Ragged** hierarchies have at least one branch that includes a member whose logical parent is not the level immediately above that member. For example, a Product hierarchy may contain the levels Category, Subcategory, and Item, but Item number 22 does not have a Subcategory associated with it. When Category, Subcategory, and Item are displayed on the report, there is an empty cell for the Subcategory of Item number 22.
- **Unbalanced and ragged** hierarchies include at least one branch that does not descend to the lowest level and one branch that includes a skipped level.



The steps below must be performed for any unbalanced or ragged hierarchy, to prevent inaccurate results when applying certain types of filters. Before you can define a hierarchy in an MDX cube as unbalanced or ragged, you need to import MDX cubes from your MDX cube source into MicroStrategy. For steps to import MDX cubes, see [Importing MDX cubes, page 80](#).

To define a hierarchy as unbalanced or ragged

- 1 In MicroStrategy Developer, log in to a project that is connected to an MDX cube source.
- 2 If you are using read only mode for the project, from the **Schema** menu, clear the **Read Only Mode** option to switch to edit mode.

Only one user can be editing a project at a given time. Therefore, if someone else is modifying the project, you cannot use the MDX Cube Catalog.

- 3 From the **Schema** menu, select **MDX Cube Catalog**.
 - If the project connects to only one MDX cube source, the MDX Cube Catalog opens.
 - If the project connects to more than one MDX cube source, the Database Instance dialog box opens. From the **Select the Database Instance** drop-down list, select an MDX cube source database instance and click **OK**. The MDX Cube Catalog opens.
- 4 Select the **Cube Mapping** tab.

- 5 From the **Catalog\Cube** drop-down list, select the MDX cube that contains the unbalanced or ragged hierarchies. The MDX cube data is displayed.
- 6 In the MDX Cube Catalog, right-click the hierarchy name in the Physical View column and select **Properties**. The Properties dialog box is displayed.
 A hierarchy in the Physical View column is represented with a green stacked boxes symbol ().
- 7 Select the **Hierarchies** tab, and then select the check box **This hierarchy is unbalanced or ragged**.
- 8 Click **OK**. The word “**(Unbalanced)**” will be displayed next to the name of the hierarchy in the Logical View column.
- 9 Click **Save and Close** to save your changes and close the MDX Cube Catalog.

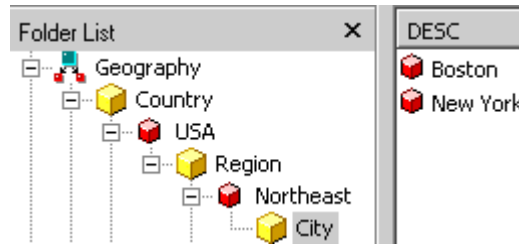
Displaying hierarchies on MDX cube reports

Report designers can include hierarchies within an MDX cube on the templates of MDX cube reports. At report runtime, any hierarchies included on an MDX cube report display attributes that are a part of that hierarchy.

When an MDX cube report includes a hierarchy, the attributes that are displayed for that hierarchy are determined by two factors:

- The default number of attributes to display for a hierarchy. You can define this default for a hierarchy when mapping data from your MDX cube source to an MDX cube in MicroStrategy (see [To define the default number of attributes to display on reports for a hierarchy](#) below). When you define the number of attributes to display for a hierarchy, you define the number of attributes to display from the highest level sequentially down to the lowest level. This is because attributes for a hierarchy are displayed in order from the highest level (first) down to the lowest level (last).

For example, you have a Geography hierarchy with Country, Region, and City attribute levels as shown below.



You define the hierarchy to display only one attribute on reports. When a report with the Geography hierarchy on the template is executed, only Country is displayed because it is the highest level attribute in the hierarchy. Defining Geography to display two attributes displays Country and Region, while defining Geography to display three attributes displays Country, Region, and City.


- The attribute level defined in the report filter of the report. A report designer may include one or more attributes in a report filter that are at a lower level than is set for a hierarchy. For more information on how the report filter affects the attributes that are displayed for a hierarchy on a report, see [Hierarchies on MDX cube reports, page 155](#).

The lower attribute level of the two factors listed above is used as the attribute level displayed on the report. As an architect mapping and configuring MDX cubes, you should set the default number of attributes to display for a hierarchy so that it fits the majority of report requirements. For more information on how this default interacts with report filters to determine the attributes displayed on a report, see [Hierarchies on MDX cube reports, page 155](#).

Only the default report display forms for each attribute are shown when attributes are displayed as part of a hierarchy on a report. To modify the attribute forms that are displayed for each attribute when displayed as part of a hierarchy, you must modify the report display forms for the attributes using the Attribute Editor. For steps to modify an attributes report display forms, see the *MicroStrategy Project Design Guide*.

Before you can define the default number of attributes to display on reports for a hierarchy, you need to import MDX cubes from your MDX cube source into MicroStrategy. For steps to import MDX cubes, see [Importing MDX cubes, page 80](#).

To define the default number of attributes to display on reports for a hierarchy

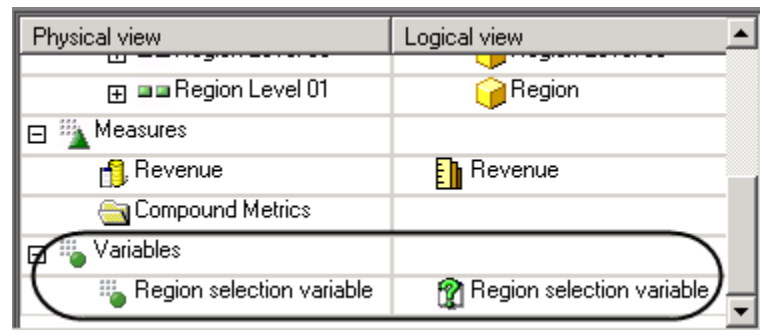
- 1 In MicroStrategy Developer, log in to a project connected to your MDX cube source.
- 2 From the **Data Explorer** for your MDX cube source, browse to an MDX cube.
- 3 Right-click the MDX cube and select **Edit**. The MDX Cube Editor opens.
- 4 In the **Physical view** column, right-click a hierarchy (denoted with the  icon), and select **Properties**. The Properties dialog box opens.
- 5 Select the **Hierarchy** tab.
- 6 Define the **Set default hierarchy display depth** to the depth of attributes to display for the hierarchy on reports. A value of 0 displays only the highest level attribute for the hierarchy, while a value of 1 displays the highest level attribute and the next highest level attribute for the hierarchy, and so on.
- 7 Click **OK** to close the Properties dialog box and return to the MDX Cube Editor.
- 8 Click **Save and Close** to save your modifications and close the MDX Cube Editor.

Mapping SAP BW variables to MicroStrategy prompts

Variables in SAP BW allow users to enter values as parameters for the queries on a cube. SAP BW variables are represented as MicroStrategy prompts when they are imported into the MicroStrategy environment. For information on how SAP BW variables are converted into MicroStrategy prompts, see [Converting SAP BW variables into MicroStrategy prompts, page 16](#).

The mapping between variables and prompts can be viewed in the Cube Mapping tab of the MDX Cube Catalog or in the MDX Cube Editor. Both

interfaces list all the variables that were converted to prompts, as is shown in the MDX Cube Editor below.



To allow attribute element searches on MDX cube reports for SAP BW cubes that include variables, within your SAP BW system, you must either define the variables as optional, or provide a default answer for the required variables. The MicroStrategy prompt that is mapped to the SAP BW variable can be either optional or required.

You can perform the following mapping and configuration tasks for SAP BW variables and their associated MicroStrategy prompts:

- You can view any variable's properties by right-clicking its name in the Physical View column and then selecting **Properties**. Details about the variable as it exists in SAP BW are displayed on the Variable tab.
- If you use any SAP BW key date variables in your query, you need to manually set the variables as key date variables. For instructions on how to set a variable as a key date variable, see [Supporting SAP BW key date variables](#) below.
- After you have saved the definition of an MDX cube, you can edit any prompts created by the import process. To do this, right-click the prompt in the Logical View column, and select **Edit**. The Prompt Generation Wizard opens, which enables you to edit various components of the prompt definition.
- You can rename a prompt by right-clicking the prompt in the **Logical View** column, and selecting **Rename**.
- You can map an SAP BW variable to a specific prompt in the MicroStrategy project by right-clicking the project and selecting **Map**. You can then browse to the prompt to map to the SAP BW variable.

One prompt can be mapped to multiple SAP BW variables in different MDX cubes. This enables a prompt to be displayed and answered only once when a Report Services document using these MDX cubes is executed. If you do not map the same prompt to the variables, the user

must answer the prompt for each variable. For steps to map one prompt to multiple variables, see [Using one prompt in documents for variables in separate MDX cubes, page 128](#).

Supporting SAP BW key date variables

If you use any SAP BW key date variables in your query, you need to manually set the variables as key date variables. This must be done to distinguish them from simple characteristic variables for dates.

Before you can define SAP BW variables in an MDX cube as key date variables, you need to import MDX cubes from your MDX cube source into MicroStrategy. For steps to import MDX cubes, see [Importing MDX cubes, page 80](#).

To define SAP BW variables as key date variables

- 1 In MicroStrategy Developer, log in to a project connected to an MDX cube source.
- 2 Right-click an MDX cube with a key date variable and select **Edit**. If the Read Only dialog box is displayed, select **Edit** and click **OK** to open the MDX Cube Editor in edit mode so that you can make changes to the MDX cube. The MDX Cube Editor opens.



Note the following:

- If you are only given the option of opening the MDX Cube Editor in read only mode, this means another user is modifying the project's schema. You cannot open the MDX Cube Editor in edit mode until the other user is finished with their changes and the schema is unlocked.
 - For information on how you can use read only mode and edit mode for various schema editors, see the *Project Design Guide*.
- 3 In the **Physical View** column, right-click the variable and select **Properties**. The Properties *variable name* dialog box is displayed.
 - 4 On the **Variable** tab, select the **Set Key Date** check box, and then click **OK**.
 - 5 Click **Save and Close** to save your changes to the MDX cube.

Supporting SAP BW variable qualifications

If you use any SAP BW variables with expression qualifications in your query, you can define which form is used to evaluate the variable's qualification. This gives you the flexibility to use either the key form or the ID form from your SAP BW data source to evaluate the variable's qualification.

Before you can define a form to use for an SAP BW variable with an expression qualification in an MDX cube, you need to import MDX cubes from your MDX cube source into MicroStrategy. For steps to import MDX cubes, see [Importing MDX cubes, page 80](#).

To define the form used to evaluate an SAP BW variable's qualification

- 1 In MicroStrategy Developer, log in to a project connected to an MDX cube source.
- 2 Right-click an MDX cube with a variable and select **Edit**. If the Read Only dialog box is displayed, select **Edit** and click **OK** to open the MDX Cube Editor in edit mode so that you can make changes to the MDX cube. The MDX Cube Editor opens.



Note the following:

- If you are only given the option of opening the MDX Cube Editor in read only mode, this means another user is modifying the project's schema. You cannot open the MDX Cube Editor in edit mode until the other user is finished with their changes and the schema is unlocked.
- For information on how you can use read only mode and edit mode for various schema editors, see the *Project Design Guide*.

- 3 In the **Physical view** column, expand the MDX cube data until you find the MDX cube column data for which to define as the form to evaluate the SAP BW variable's qualification.



You can define either the key form or the ID form from your SAP BW data source as the form used to evaluate the variable's qualification.

- 4 From the **Logical view** column, right-click the MicroStrategy attribute form mapped to the MDX cube column data and select **Variable Qualification Form**.
- 5 Click **Save and Close** to save your changes to the MDX cube.

Using one prompt in documents for variables in separate MDX cubes

One MicroStrategy prompt can be mapped to multiple SAP BW variables in different MDX cubes. This enables a prompt to be displayed and answered only once when executing a Report Services document using these MDX cubes. If you do not map the same prompt to the variables, the user must answer the prompt for each variable.

You should be aware of the following when mapping variables from different MDX cubes to the same prompt:

- When you map a variable to a prompt, the prompt that you select replaces the prompt that was previously mapped to the variable.
- Variables mapped to the same prompt do not have to be identical. However, the variables must be similar enough that the prompt can complete any variables it is mapped to with the same prompt answer.
- Prompts are automatically mapped to a variable in an MDX cube. You can then map these automatically created prompts to variables in other MDX cubes. This ensures that the prompt is defined to work correctly with an SAP BW variable.

For example, you import MDX cubes A, B, and C, which all have a variable that you want to map to the same prompt. The prompt that you want to use is mapped to the variable in MDX cube A. You can edit MDX cube B and MDX cube C to map the variables in each of these MDX cubes to the prompt in MDX cube A. All of the variables in the three MDX cubes are then mapped to the same prompt. Use the procedure below for steps to map a variable in one MDX cube to a prompt mapped to a variable in a different MDX cube.

Before you can map multiple variables to one prompt, you need to import MDX cubes from your MDX cube source into MicroStrategy. For steps to import MDX cubes, see [Importing MDX cubes, page 80](#).

To map multiple variables to one prompt

Review the information in [Using one prompt in documents for variables in separate MDX cubes](#) above for tips on how to map the same prompt to variables in different MDX cubes.

- 1 In MicroStrategy Developer, log in to a project connected to an MDX cube source.
- 2 Right-click an MDX cube that contains a variable and select **Edit**. If the Read Only dialog box is displayed, select **Edit** and click **OK** to open the MDX Cube Editor in edit mode so that you can make changes to the MDX cube. The MDX Cube Editor opens.



Note the following:

- If you are only given the option of opening the MDX Cube Editor in read only mode, this means another user is modifying the project's schema. You cannot open the MDX Cube Editor in edit mode until the other user is finished with their changes and the schema is unlocked.
 - For information on how you can use read only mode and edit mode for various schema editors, see the *Project Design Guide*.
- 3 In the **Logical View** column, right-click the prompt mapped to the variable and select **Map**. The Select Destination Object dialog box opens.
 - 4 Browse to and select the prompt to map to the variable, and click **Open**. The prompt that you select replaces the prompt previously mapped to the variable.
 - 5 Click **Save and Close** to save your changes and close the MDX Cube Editor.

Creating metrics from MDX cube data

When you map your MDX cube data into MicroStrategy, you can take advantage of MDX (MultiDimensional eXpressions) to create metrics. Metrics created with MDX combine the robust set of MDX functions and expressions with MicroStrategy analytical tools such as prompts. You can

also use basic arithmetic expressions to create these metrics from MDX cube data.

Once you create metrics using these techniques you can include them in your MicroStrategy reports and report filters in the same ways that you can use any MicroStrategy metric. You can also use prompts in these compound and custom MDX metrics (see [Using prompts within MDX cube metrics, page 138](#)). The metrics created in this way for an MDX cube are stored in a Compound Metrics folder within the Metrics folder for the MDX cube.

Metrics created to map to your MDX cube data are related only to their associated MDX cube. Therefore, these metrics cannot be directly integrated with data from a separate relational data source, except by using calculated expressions in Report Services documents. For information on creating calculated expressions, see the *Designing and Creating Documents* chapter of the *MicroStrategy Report Services Document Creation Guide*.

You can create metrics that map to MDX cube data using either of the following techniques:

- **Compound metrics:** A compound metric is any MicroStrategy metric with an expression that includes a MicroStrategy metric and an arithmetic expression. The expression can be as simple as a metric multiplied by a constant value, such as `Discount * 1.5`, where `Discount` is a metric mapped to data in the MDX cube. These metrics can also reference multiple MicroStrategy metrics within the MDX cube with an expression such as `Revenue - Total Expenses`, where `Revenue` and `Total Expenses` are both metrics, to build a Profit metric.

You can use MicroStrategy analytical and *aggregate functions* such as `SUM`, `COUNT`, and `AVG` with metrics mapped to MDX cube data only if the metric you create is defined as a smart metric. If you do not make the metric a smart metric you can only use basic operators (`+`, `-`, `/`, `*`, and so on). For general information on smart metrics, see the *MicroStrategy Basic Reporting Guide*. For examples of smart metrics, see the *MicroStrategy Advanced Reporting Guide*.

- **MDX customization:** Rather than relying only on MicroStrategy to create MDX to return data from your MDX cube source, you can create your own custom MDX to return data for a metric. This technique allows you to use MDX functions and flexibility to query and report on your MDX cube data. The MDX you create is passed to your MDX cube source to be executed and to return the data. You can reference one or more MicroStrategy metrics mapped to MDX cube data using custom MDX just as you can with a standard arithmetic expression. To use MDX to create your calculated measures you must enclose MDX in double quotes (`"``"`).

For tips and insights on how to build analysis with MDX in MicroStrategy, see [How to build analysis into metrics with custom MDX, page 132](#).

You can create these metrics during the initial importing and mapping procedure of your MDX cube data with the MDX Cube Catalog. These metrics can also be created as a later modification to an MDX cube with the MDX Cube Editor. The following procedure uses the MDX Cube Catalog.

Before you can create a metric from MDX cube data with MDX and compound metric techniques, you need to import MDX cubes from your MDX cube source into MicroStrategy. For steps to import MDX cubes, see [Importing MDX cubes, page 80](#).

To create a metric from MDX cube data with MDX and compound metric techniques

- 1 In MicroStrategy Developer, log in to a project that is connected to an MDX cube source.
- 2 If you are using read only mode for the project, from the **Schema** menu, clear the **Read Only Mode** option to switch to edit mode.

Only one user can be editing a project at a given time. Therefore, if someone else is modifying the project, you cannot use the MDX Cube Catalog.

- 3 From the **Schema** menu, select **MDX Cube Catalog**.
 - If the project connects to only one MDX cube source, the MDX Cube Catalog opens.
 - If the project connects to more than one MDX cube source, the Database Instance dialog box opens. From the **Select the Database Instance** drop-down list, select an MDX cube source database instance and click **OK**. The MDX Cube Catalog opens.
- 4 Select the **Cube Mapping** tab.
- 5 From the **Catalog\Cube** drop-down list, select the MDX cube to create metrics for. The MDX cube data is displayed.
- 6 From the **Edit** menu, select **Add New Compound Metric**. The Metric Editor opens.

7 Create the expression for your metric:

- If you are creating a compound metric, you can simply drag and drop metrics from the MDX cube's Metrics folder. This includes any required constants, arithmetic operators, and MicroStrategy analytical and aggregate functions in your new metric expression. For example, if you have Revenue and Cost metrics in your MDX cube you can create the expression `Revenue - Cost` to create a Profit metric.



You can use MicroStrategy analytical and aggregate functions with metrics mapped to MDX cube data only if the metric you create is defined as a smart metric. See the *MicroStrategy Advanced Reporting Guide* for information on enabling smart metrics.

- If you are creating a metric using custom MDX, enter your custom MDX in the **Definition pane** of the Metric Editor. Make sure to enclose the entire expression in double quotes. For example, you can enter the following:

```
"[Measures].[Discount Amount] * 1.5"
```



You cannot validate MDX in the Metric Editor as you can for a standard expression that is not enclosed by double quotes. Validating MDX verifies that the entire expression is enclosed in double quotes; it does not validate the syntax of the expression.

For an example of creating a metric that includes a prompt, see [Using prompts within MDX cube metrics, page 138](#).

- 8 Click **Save and Close**. The Save As dialog box opens.
- 9 In the **Object name** text field, enter a name for your metric.
- 10 Click **Save** to save your metric.
- 11 Click **Save and Close** to save your changes to the MDX cube.

How to build analysis into metrics with custom MDX

You can build sophisticated analysis into your MDX cube metrics by creating your own custom MDX. This allows you to further combine the analysis capabilities of MDX and MicroStrategy. This section provides some tips and

best practices on how to build analysis into metrics with custom MDX. MDX syntax and functionality is not described in depth in this section; only basic principles of analysis with the use of MDX and MicroStrategy is provided.



Be aware of the following:

- Creating such analysis requires appropriate knowledge of both MDX and MicroStrategy. MicroStrategy does not validate any custom MDX created by users to build metrics for MDX cubes.
- MDX has strict rules about the inclusion or exclusion of dimensions, hierarchies, and attributes on the report template and in custom MDX formulas. Some MDX formulas expect related attributes to exist on the template, and they may return incorrect results (or an error) if the attributes are omitted. Other formulas may return unexpected results if the attributes are included. As a result, certain custom MDX formulas may not be appropriate for ad-hoc reporting and you should be aware of the possible limitations of the custom MDX you create.

You should begin by familiarizing yourself with the prerequisite and basic information described in the [Basics](#) section below. You can then review the following sections for information on specific analysis scenarios achievable with custom MDX:

- [Filtering individual metrics: Conditional metrics, page 135](#)
- [Creating transformation-style analysis, page 136](#)
- [Using prompts and ApplySimple statements, page 137](#)

Basics

Creating your own custom MDX allows you to draw further analysis from your MDX cube source into MicroStrategy. Any expressions that are valid in a `WITH MEMBER` clause may be used, allowing metrics built in MicroStrategy to employ the data manipulation capabilities of the MDX cube source. The custom MDX is placed into a `WITH MEMBER` clause defining a member of the Measures collection for the scope of the query.

To use MDX to create your metrics, you must enclose MDX in double quotes (""). For example, "[Measures].[Total Sales]" is valid syntax for a metric defined with MDX. The expression shown above is a simple expression that returns the Total Sales data from an MDX cube.

Since MDX is passed to and run against your MDX cube source, you must use the names and identifiers used in the MDX cube source to identify the data to be retrieved. For SAP BW, the technical name should be used.

Recall the example syntax above "[Measures].[Total Sales]". In your MDX cube source, your metric data is identified as Total Sales, and this data is then mapped to a metric named Revenue in MicroStrategy. When you are creating custom MDX to retrieve this data from your MDX cube source, you must use the identifier for the data within the MDX cube source. In this example the correct identifier to use is Total Sales.

You can also perform basic arithmetic in your MDX. For example, the following expression applies a multiplier to the Total Sales data:

```
"[Measures].[Total Sales] * .06"
```

Along with these simple expressions, you can also use MDX functions to create more advanced analysis. When you include an MDX function in your custom MDX, the function is passed to the MDX cube source and processed as a pass-through function. For example, you can use the MDX year-to-date (YTD) function to create transformation-style analysis on your MDX cube data, as shown below:

```
"sum(YTD([Quarter].CurrentMember), [Measures].[Profit])"
```

This expression returns year-to-date values by quarter for profit data, as shown in the report below.

Category	Quarter Metrics	Q1 2000		Q2 2000		Q3 2000	
		Profit	Profit YTD	Profit	Profit YTD	Profit	Profit YTD
Books		\$2,000	\$2,000	\$2,434	\$4,434	\$1,550	\$5,984
Electronics		\$340,532	\$340,532	\$475,507	\$816,039	\$285,311	\$1,101,350
Movies		\$77,286	\$77,286	\$103,313	\$180,599	\$60,263	\$240,862
Music		\$35,089	\$35,089	\$46,396	\$81,485	\$27,938	\$109,423

For more background information on and examples of transformation-style analysis with custom MDX, see [Creating transformation-style analysis, page 136](#).

Filtering individual metrics: Conditional metrics

Using MDX, you can create conditional metrics in MicroStrategy from your MDX cube data. Metric *conditionality* enables you to apply a filter to data from only one metric on a report while not affecting the other metric data on a report. Report designers can include these metrics on reports to view multiple perspectives of data on the same report. For example, along with viewing your total revenue on a report, you can also display revenue for a certain category such as electronics.

In general, attribute elements are used to filter metric data using the following MDX syntax:

```
"([Measures].[metric_data_identifier_in_MDX_cube_source], attribute_element_identifier_in_MDX_cube_source)"
```

In the example expression below, bold highlights the part of the expression (including the comma) that applies the condition to the revenue data:

```
"([Measures].[Revenue], [Category].[2])"
```



In the example above, **[Category].[2]** identifies the electronics category. The values that identify data depend on how you have defined data in your MDX cube source.

The report shown below uses this metric to compare total revenue with electronics revenue.

Region	Call Center	Metrics	Revenue	Electronics Revenue
Northeast	Boston		\$1,325,448	\$883,587
	New York		\$1,009,416	\$667,197
Mid-Atlantic	Washington DC		\$1,413,865	\$952,322
	Charleston		\$1,999,475	\$1,329,525
Southeast	Atlanta		\$1,083,016	\$723,567
	Miami		\$933,170	\$633,035
Central	Milwaukee		\$1,340,391	\$890,802
	Fargo		\$432,879	\$293,134

You can group multiple attribute elements of the same attribute in a metric condition using the general syntax shown below:

```
"Aggregate_Function({attribute_element_identifier,..., attribute_element_identifier}, [Measures].[metric_data_identifier])"
```

For example, you can create a Books & Electronics Revenue metric that returns the combined revenue for the electronics and books categories. Syntax for this type of metric is shown below:

```
"Sum ({ [Category] . [1] , [Category] . [2] } ,
[Measures] . [Revenue] ) "
```

The report shown below includes both metrics from the examples above for side-by-side comparison:

Region	Call Center	Metrics	Revenue	Electronics Revenue	Books & Electronics Revenue
Northeast	Boston		\$1,325,448	\$883,587	\$888,562
	New York		\$1,009,416	\$667,197	\$671,315
Mid-Atlantic	Washington, DC		\$1,413,865	\$952,322	\$957,826
	Charleston		\$1,999,475	\$1,329,525	\$1,337,599
Southeast	Atlanta		\$1,083,016	\$723,567	\$727,816
	Miami		\$933,170	\$633,035	\$636,360
Central	Milwaukee		\$1,340,391	\$890,802	\$895,512
	Fargo		\$432,879	\$293,134	\$294,556

You can also combine conditions on different attributes for each metric. For example, you can create the same metric to return electronics revenue for only the year 2006. In the example expression below, a second condition on the year is included by adding another comma and conditional expression:

```
" ( [Measures] . [Revenue] , [Category] . [2] , [Year] . [2006] ) "
```

Creating transformation-style analysis

You can create transformation-style analysis with custom MDX by using two types of MDX functions. MDX provides functions:

- `PrevMember` and `NextMember` to move to the previous or next element in the same attribute.
- Such as `MTD`, `WTD`, or `YTD` (month, week, and year to date) to aggregate data based on time hierarchy models.

The following custom MDX examples include a one-to-one transformation to display the previous quarter's revenue, and a many-to-one transformation calculating year-to-date revenue:

- `"([Measures].[REVENUE],
[Quarter].CurrentMember.Prevmember)"`

Displays the revenue data for the previous quarter. The Last Quarter Revenue metric shown in the report below is mapped to this custom MDX formula.

- `"sum(YTD([Quarter].CurrentMember),
[Measures].[REVENUE])"`

Displays the total revenue data for all quarters of a given year up to and including the current quarter. The Year To Date Revenue metric shown in the report below is mapped to this custom MDX formula.

The metrics created from these two expressions are included on the report shown below:

Quarter	Revenue	Last Quarter Revenue	Year To Date Revenue
Q1 2000	\$1,884,444		\$1,884,444
Q2 2000	\$2,585,002	\$1,884,444	\$4,469,446
Q3 2000	\$1,545,399	\$2,585,002	\$6,014,845
Q4 2000	\$2,861,785	\$1,545,399	\$8,876,630
Q1 2001	\$1,222,374	\$2,861,785	\$1,222,374
Q2 2001	\$2,382,652	\$1,222,374	\$3,605,026
Q3 2001	\$1,560,244	\$2,382,652	\$5,165,270
Q4 2001	\$2,894,534	\$1,560,244	\$8,059,804

Using prompts and ApplySimple statements

All of the MDX examples in the sections above are static expressions, meaning they will produce the same MDX every time. Using the ApplySimple function, you can include prompts in your MDX to provide dynamic analysis on your MDX cube data. For basic information and examples of the ApplySimple function, see the *MicroStrategy Functions Reference*.

The example below shows the basic structure of an ApplySimple statement to create metrics with custom MDX.

```
ApplySimple("MDX expression with placeholders for  
objects", object0, object1, ..., objectN)
```

A simple application of this technique is to use a constant value prompt in your project as a multiplier of metric data, as shown below.

```
ApplySimple("(" & [Measures].[Total Sales] / #0) "  
, ?valueprompt)
```

In the example syntax above, #0 is a placeholder in the MDX expression for the value prompt. The syntax for including a prompt as an object to replace a placeholder is *?promptname*.

You can also use this technique with the conditional metrics techniques described in [Filtering individual metrics: Conditional metrics](#) above. For example, rather than always returning the revenue data for electronics, you can allow users to choose what category to view revenue for. To provide this analysis to users, you can include an element list prompt on the Category attribute of the MDX cube, as shown below.

```
ApplySimple("(" & [Measures].[Revenue], #0) "  
, ?elementlistprompt)
```

For more information on and a procedure for creating metrics in MDX cubes with prompts, see the [Using prompts within MDX cube metrics](#) section below.

Using prompts within MDX cube metrics

If you are creating new metrics in your MDX cube, you can also include MicroStrategy prompts with the metrics. When the metrics are included on a report and the report is run, the prompts are displayed to the user for completion. This adds flexibility to your queries, allowing users to determine the data to see on the report.



The [Using prompts and ApplySimple statements](#) section above describes how to include prompts within custom MDX of MDX cube metrics. This section covers the general requirements and process to include prompts in MDX cube metrics.

For metrics created with compound metric techniques without any custom MDX, you can simply include a prompt in the metric definition. For metrics created using MDX expressions, you must use the ApplySimple function to include prompts in the metric definition.

The following types of prompts can be included with MDX cube metrics:

- Element list prompts defined on an attribute of the associated MDX cube
- Value prompts
- Object prompts defined on objects of the associated MDX cube

For information on how prompts can be included in an MDX cube report and the supported prompt types, see [Prompts on MDX cube reports, page 169](#). For a review of all prompt types, see the *MicroStrategy Basic Reporting Guide*.

Before you can use prompts in MDX cube metrics, you need to import MDX cubes from your MDX cube source into MicroStrategy. For steps to import MDX cubes, see [Importing MDX cubes, page 80](#).

To use prompts in MDX cube metrics

- 1 In MicroStrategy Developer, log in to a project that is connected to an MDX cube source.
- 2 If you are using read only mode for the project, from the **Schema** menu, clear the **Read Only Mode** option to switch to edit mode.

Only one user can be editing a project at a given time. Therefore, if someone else is modifying the project, you cannot use the MDX Cube Catalog.

- 3 From the **Schema** menu, select **MDX Cube Catalog**.
 - If the project connects to only one MDX cube source, the MDX Cube Catalog opens.
 - If the project connects to more than one MDX cube source, the Database Instance dialog box opens. From the **Select the Database Instance** drop-down list, select an MDX cube source database instance and click **OK**. The MDX Cube Catalog opens.
- 4 Select the **Cube Mapping** tab.
- 5 From the **Catalog\Cube** drop-down list, select the MDX cube to create metrics for. The MDX cube data is displayed.
- 6 From the **Edit** menu, select **Add New Compound Metric**. The Metric Editor opens.

- 7 Enter your expression in the **Definition pane** of the Metric Editor. For example, you can enter expressions similar to the following:

- `([Discount Amount] * ?constantprompt)`

This expression applies a special discount amount, entered by the user running the report. In this example, it is assumed that `constantprompt` is the name of a value prompt in the project and `Discount Amount` is a metric within the MDX cube.

- `ApplySimple("([Measures].[Revenue], #0)",
?CategoryElementPrompt)`

This expression creates a Revenue metric, which is conditional on an element list prompt answered by the user running the report. Users can then choose to view revenue data for different categories such as Books or Music. In this example, it is assumed that `CategoryElementPrompt` is the name of an element list prompt in the project that references a Category attribute within the MDX cube.

- 8 Click **Save and Close**. The Save As dialog box opens.
- 9 In the **Object name** text field, enter a name for your metric.
- 10 Click **Save** to save your metric and return to the MDX Cube Catalog.
- 11 Click **Save and Close** to save your changes to the MDX cube and exit the MDX Cube Catalog.

Deleting compound metrics from MDX cubes

When you delete metrics based on multiple metrics of an MDX cube, dependencies may need to be resolved before you can delete the metric. When you try to delete a metric with dependent metrics, a list of metrics that are dependent on the metric you are deleting is returned. If a compound metric of an MDX cube has been added to any reports, a list of reports that depend on the metric is also returned. You must delete all of the metrics and reports which depend on the metric you are trying to delete. Then you can delete the metric.



You can remove the compound metric from the report rather than deleting the report. This removes the dependency between the metric and the report, and you can then remove the metric from the MDX cube.

For example, you import an MDX cube, and its data is automatically mapped to MicroStrategy metrics. You then create a new compound metric named Profit within the MDX cube by subtracting the MDX cube's cost data from its revenue data. Once this metric is created in your MDX cube, you create a Profit Margin metric that uses the Profit metric you just created. This makes the Profit Margin metric dependent on the Profit metric.

If you try to delete the Profit metric, a search for dependent objects is automatically triggered, and the Profit Margin metric is returned. To delete the Profit metric, you must first delete the Profit Margin metric. You also need to delete any reports that include the Profit metric or remove the Profit metric from the reports before you can delete the Profit metric from the MDX cube.

Creating data marts of MDX cube data

A MicroStrategy data mart is a data repository where you store the results of a report as a relational table in a data warehouse. After creating a data mart, you can use it as a source table in your projects, and execute reports against it.

Data marts can also be created based on MDX cube reports. Once this MDX cube data is available as data mart tables in your relational data warehouse, you can then map the data to the schema objects of your project, and then create standard MicroStrategy reports based on the data. This lets you take advantage of MicroStrategy features that cannot be used directly on MDX cube reports. These features include, but are not limited to, custom groups, consolidations, stand-alone filters, and metrics with complex definitions.

For example, the report below is based on MDX cube data from an SAP BW MDX cube source, which has been included in a project as a data mart. The report also includes the Top 10 Tenured Employees custom group, which could not be used directly on an MDX cube report. However, by including the MDX cube data in the project as a data mart and mapping the data to standard attributes and metrics, you can create a standard report based on

this data that can take advantage of MicroStrategy features such as custom groups.

		Metrics	Days Employed	Employee Age	Salary	Timeliness
Top 10 Tenured Employees			Employee Level 01			
Top 10 Tenured Employees	1	McClain	3,841	54	\$42,000	0.908
	2	Bell	3,718	52	\$35,000	0.912
	3	Strome	3,716	49	\$31,000	0.900
	4	Zemlicka	3,573	35	\$31,000	0.916
	5	Becker	3,470	28	\$22,000	0.911
	6	Kieferson	3,435	44	\$22,000	0.920
	7	Nelson	3,394	46	\$27,000	0.910
	8	Ingles	3,336	54	\$22,000	0.910
	9	Hall	3,258	57	\$31,000	0.920
	10	Gale	3,216	37	\$35,000	0.909

Once you create an MDX cube report (see [Creating MDX cube reports, page 144](#)), you can use it to create a data mart. With the MDX cube report open in the Report Editor, from the **Data** menu, select **Configure Data Mart**. The Report Data Mart Setup dialog box opens, which lets you create the data mart. For information on data marts, including prerequisites, examples, and steps to create a data mart, see the *Advanced Reporting Guide*.

REPORTING ON MDX CUBES

Creating MDX Cube Reports and Analyzing Data

Introduction

This chapter provides information on how to create MicroStrategy MDX cube reports and use MicroStrategy features to analyze data from your MDX cube source. This chapter assumes that you are familiar with the basics of report creation and analysis in MicroStrategy. For information on basic report creation and analysis, see the *MicroStrategy Basic Reporting Guide*.

A *report* in MicroStrategy is the central focus for MicroStrategy users to query, analyze, and visually present data in a manner that answers and evaluates their business questions. *MDX cube reports* provide the same data display and analysis functionality, but rather than reporting on data from a relational data warehouse, MDX cube reports report on data from MDX cube sources.

This chapter discusses the following topics related to MDX cube reports:

- *Creating MDX cube reports, page 144*: This section provides the basic procedure to create an MDX cube report.
- *Analyzing data with MDX cube reports, page 154*: This section provides information and procedures to analyze your MDX cube data with MDX cube reports.

Before you can create an MDX cube report, you must import your MDX cubes and integrate them into a MicroStrategy project. For information on integrating MDX cube sources with MicroStrategy, see [Chapter 3, Integrating MDX Cubes into MicroStrategy](#).

Creating MDX cube reports

You can create an MDX cube report using an imported MDX cube as the data source. MDX cube reports can be created in either MicroStrategy Developer or Web.

The steps below show how to create an MDX cube report that reports directly on an MDX cube. You also have the following alternatives for creating reports on MDX cube data:

- With OLAP Services features, you can report on MDX cubes by creating Intelligent Cube reports that connect to Intelligent Cubes containing MDX cube data. For steps to create an Intelligent Cube based on an MDX cube, see [Creating Intelligent Cubes based on MDX cubes, page 148](#). For steps to create an Intelligent Cube report, refer to the *In-memory Analytics Guide*.
- With MultiSource Option features, you can include MDX cube data along with data from your relational project. For steps to create a report with these analysis capabilities, see [Including MDX cube data in standard reports, page 150](#).

Prerequisites

- To create an MDX cube report in Developer, you need to have Developer privileges, including the Define MDX Cube Report privilege.
- To create an MDX cube report in Web, you need to have Web Professional privileges, including the Web Define MDX Cube Report privilege.
- At least one MDX cube must be imported into your MicroStrategy project. Importing MDX cubes is often handled by a MicroStrategy architect. For more information on importing MDX cubes into MicroStrategy, see [Importing MDX cubes, page 80](#).
- You should have some experience and knowledge of designing MicroStrategy reports, as described in the *MicroStrategy Basic Reporting Guide* and the *MicroStrategy Advanced Reporting Guide*.

To create an MDX cube report

- 1 Create a new report and select an MDX cube to report on. The steps to do this are different for MicroStrategy Developer and Web:

- [In Developer](#)
- [In MicroStrategy Web](#)

In Developer

- a Log in to a project connected to an MDX cube source.
- b From the **File** menu, select **New**, and then select **Report**. The New Grid dialog box opens.
- c On the **MDX Sources** tab, select a database instance connected to an MDX cube source and click **OK**. The Select Cube dialog box opens.
- d Select an MDX cube to report on:



To show the technical names for MDX cubes, select the **Display technical names** check box.

- If MDX cubes have already been imported by an architect, they are displayed in their respective catalog structure. Click the plus sign (+) next to the catalog name to display its contents, and then select the imported cube to use for your report.

You can use the Find dialog box to search for a specific cube to use for your report.

- If the MDX cube that you want to report on is not imported yet, you can click **Retrieve cubes**. Select the MDX cube from the list of MDX cubes from the selected database instance. If you select an MDX cube that has not been imported yet, it is imported into MicroStrategy before the report is created. You must have the Import MDX Cube privilege to perform this action.

You can also use the MDX Cube Catalog to import cubes, as described in [Importing MDX cubes, page 80](#).

- You can later switch the MDX cube for your report if necessary, as described in [Switching the MDX cube for an MDX cube report, page 153](#).

In MicroStrategy Web

- a Log in to a project connected to an MDX cube source.
- b Click the **MicroStrategy** icon, and then select **Create Report**. A page with several report creation choices is displayed.
- c Underneath **Choose data source**, click **MDX Cube Report**. The Select MDX Cube dialog box opens.
- d Click an MDX cube source. Catalogs available for the MDX cube source are displayed.
- e Click a catalog. MDX cubes available in the catalog are displayed.
- f Select an MDX cube.



You can later switch the MDX cube for your report if necessary, as described in [Switching the MDX cube for an MDX cube report, page 153](#).

- 2 Click **OK** to open the Report Editor.
- 3 Select attributes, metrics, prompts, hierarchies, and other objects as required from the Object Browser and put them on the report template. Some MicroStrategy objects on MDX cube reports work differently than on standard MicroStrategy reports, as described in the sections listed below:
 - [Hierarchies on MDX cube reports, page 155](#)
 - [Prompts on MDX cube reports, page 169](#)
- 4 Create a filter, if needed. Filters on MDX cube reports work differently than on standard MicroStrategy reports, as described in the section [Filters on MDX cube reports, page 160](#).
- 5 From the **Data** menu, select **Run Report**. The report is executed and displayed in the view selected.



From the **View** menu, you can change the view to Grid View, Graph View, Grid Graph View, or MDX View.

- 6 Format the report, as needed.

You can use OLAP Services features such as view filters, derived metrics, and dynamic aggregation. If you are using MicroStrategy Web to create and view the MDX cube report, you can also use the OLAP Services

feature derived elements to analyze the data. For details on how to use OLAP Services features, see the *In-memory Analytics Guide*.

- 7 From the toolbar, select **Save and close** to save the report and close the Report Editor.

Troubleshooting MDX cube report execution

Review the list below for common MDX cube report troubleshooting considerations:

- If you experience long wait times when running MDX cube reports, this may be caused by the processing time required to load MDX cube schemas. MDX cube schemas that are loaded at report runtime can negatively affect the performance of MDX cube report execution. A project administrator can select to load MDX cube schemas when Intelligence Server starts, thus removing the overhead of MDX cube schema loading from the report execution process. Steps to change the MDX cube schema load time are provided in [MDX cube schema loading, page 61](#).
- An MDX cube report fails with an error message that identifies data that has been mapped to an incompatible data type. A project designer can resolve the incompatible data type mapping to MDX cube data. Steps to resolve this type of error are provided in [Resolving incompatible data type errors, page 113](#).
- An MDX cube report fails with an error message that indicates the maximum row limit has been exceeded. A project administrator can resolve this error by increasing a project governing setting. Steps to increase the maximum row limit for reports are provided in [Supporting large result sets for MDX cube reports, page 64](#).
- A report that includes MDX cube data and relational project data fails with an error message that indicates data for an MDX metric is not available at the level for the report. This can occur if you are using MultiSource Option to include both MDX cube data and relational project data on a standard report. You can resolve this issue by removing any attributes from the report that are not mapped to data in the MDX cube used in the report. Creating these types of reports is described in [Including MDX cube data in standard reports, page 150](#).
- An MDX cube report does not provide options to search the attribute elements for the report, or these options are greyed out. For example, the ability to search for attribute elements can be provided when answering a

prompt while executing a report or while selecting attributes as part of designing a report. These search options are unavailable if the MDX cube source the MDX cube report is associated with includes required variables. To support searching the attribute elements, within your SAP BW system, you must either define the variables as optional, or provide a default answer for the required variables. The MicroStrategy prompt that is mapped to the SAP BW variable can be either optional or required.

Creating Intelligent Cubes based on MDX cubes

Once MDX cubes are integrated into MicroStrategy, you can begin to create MDX cube reports directly on these MDX cubes, as described in [Creating MDX cube reports, page 144](#).

Alternatively, once an MDX cube is created, you can create an Intelligent Cube based on the MDX cube. This stores the MDX cube data as an Intelligent Cube, which allows you to take advantage of various Intelligent Cube features, including the improved response time of reporting against Intelligent Cubes.

Once an Intelligent Cube is created for an MDX cube, reports can be created based on the Intelligent Cube. These reports can analyze the MDX cube data, while also taking advantage of OLAP Services analysis features such as derived elements.


The steps below show you how to create an Intelligent Cube based on an MDX cube.

Prerequisites

- At least one MDX cube must be imported into your MicroStrategy project. Importing MDX cubes is often handled by a MicroStrategy architect. For more information on importing MDX cubes into MicroStrategy, see [Importing MDX cubes, page 80](#).
- You need the Use Intelligent Cube Editor privilege to create Intelligent Cubes, which is part of OLAP Services privileges.

To create an Intelligent Cube based on an MDX cube

- 1 Using MicroStrategy Developer, log in to a project connected to an MDX cube source.

- 2 From the **File** menu, select **New**, and then select **Intelligent Cube**. The New Intelligent Cube dialog box opens.
- 3 On the **MDX Sources** tab, select a database instance connected to an MDX cube source and click **OK**. The Select Cube dialog box opens.
- 4 Select an MDX cube to report on:
 -  To show the technical names for MDX cubes, select the **Display technical names** check box.
 - If MDX cubes have already been imported by an architect, they are displayed in their respective catalog structure. Click the plus sign (+) next to the catalog name to display its contents, and then select the imported cube to use for your Intelligent Cube.
 - If the MDX cube that you want to report on has not been imported yet, click **Retrieve cubes**. Select the MDX cube from the list of MDX cubes from the selected database instance. If you select an MDX cube that has not been imported yet, it is imported into MicroStrategy before the report is created. You must have the Import MDX Cube privilege to perform this action.

You can also use the MDX Cube Catalog to import cubes, as described in [Importing MDX cubes, page 80](#).
- 5 Click **OK** to open the Report Editor.
- 6 From the **Object Browser** pane, select attributes, metrics, and other objects from the MDX cube to create the data available for the Intelligent Cube.
- 7 From the **File** menu, select **Save**. The Save Intelligent Cube As dialog box opens.
- 8 In the **Object name** field, type a descriptive name for the Intelligent Cube.
- 9 Click **OK** to save the Intelligent Cube and return to the Report Editor.
- 10 To make the Intelligent Cube available for creating reports, you must publish the Intelligent Cube. This can be done by selecting **Run Report** in the toolbar. For important details on publishing Intelligent Cubes, including prerequisites to consider before publishing an Intelligent Cube, see the *In-memory Analytics Guide*.

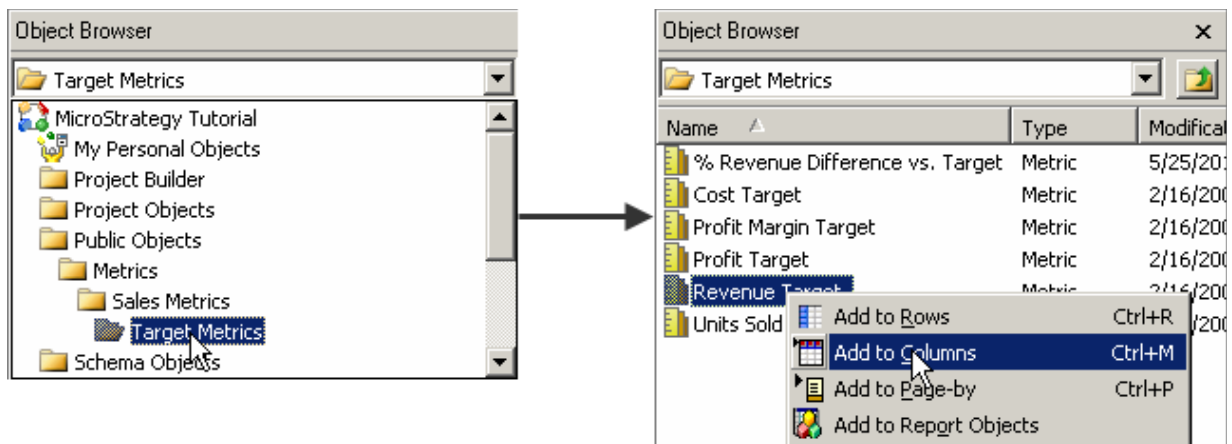
- 11 Once the Intelligent Cube is published, reports can be created on the Intelligent Cube. For steps on how to report on Intelligent Cubes, see the *In-memory Analytics Guide*.

Including MDX cube data in standard reports

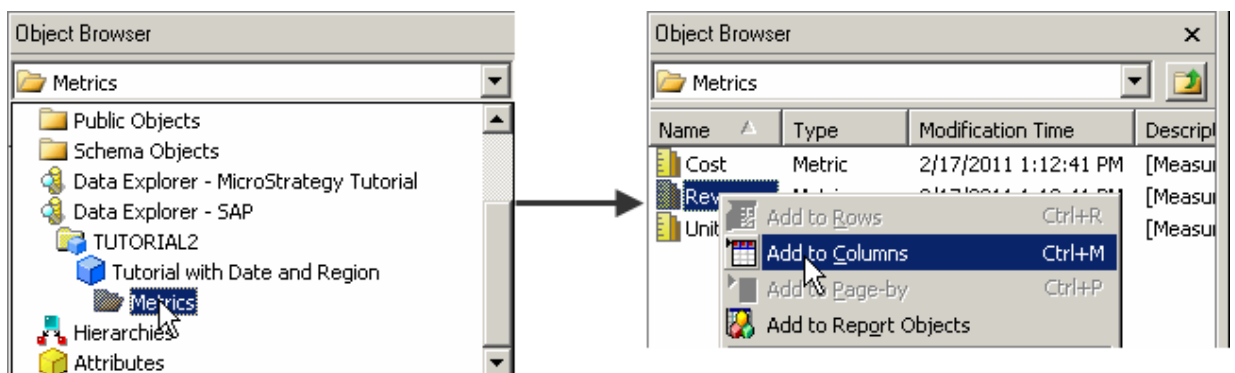
In addition to creating MDX cube reports that report on single MDX cubes, you can include MDX cube data in standard reports. This lets you include MDX cube data in a report along with data from your relational project.

For example, the images below demonstrate using project attributes along with metrics from the relational project, as well as metrics from an MDX cube.

After attributes are added to the report, a metric from the project is added to the report.



Then, a metric from an MDX cube is added to the report.



The resulting report displays both the relational project metrics and the MDX cube metrics on the same standard report.

Region	Call Center	Metrics	Revenue Target	Revenue
Central	Milwaukee		\$3,973,003.48	\$ 1,340,391.00
	Fargo		\$799,971.94	\$ 432,879.00
Mid-Atlantic	Washington, DC		\$2,955,064.89	\$ 1,413,865.00
	Charleston		\$1,253,854.52	\$ 1,999,475.00
Northeast	Boston		\$1,410,959.66	\$ 1,325,448.00
	New York		\$6,749,413.80	\$ 1,009,416.00
Northwest	San Francisco		\$974,118.26	\$ 1,050,983.00
	Seattle		\$705,431.37	\$ 434,199.00
South	New Orleans		\$3,141,629.17	\$ 867,240.00
	Memphis		\$1,976,536.69	\$ 513,751.00
Southeast	Atlanta		\$1,002,525.17	\$ 1,083,016.00
	Miami		\$1,119,542.54	\$ 933,170.00
Southwest	San Diego		\$2,821,301.97	\$ 2,397,919.00
	Salt Lake City		\$701,024.14	\$ 418,415.00
Web	Web		\$3,702,625.89	\$ 1,716,267.00

In this example, the report is able to display revenue data from the MDX cube along with revenue targets from metrics that are part of the relational project.

The steps below show you how to create a report that includes data from MDX cubes and a relational project.

Prerequisite

- To create an MDX cube report in Developer, you need to have Developer privileges, including the Define MDX Cube Report privilege.
- To create an MDX cube report in Web, you need to have Web Professional privileges, including the Web Define MDX Cube Report privilege.
- At least one MDX cube must be imported into your MicroStrategy project. Importing MDX cubes is often handled by a MicroStrategy architect. For more information on importing MDX cubes into MicroStrategy, see [Importing MDX cubes, page 80](#).
- To include MDX cube data in standard reports, you must map MDX cube columns for each project attribute you plan to include in the reports. For example, if a report includes the project attributes Year, Region, and Category, you must map MDX cube columns to these three attributes. For steps to map MDX cube columns to project attributes, see [Mapping MDX cube data to project attributes, page 103](#).

- You must have MultiSource Option privileges. For information on the capabilities available in MicroStrategy with the MultiSource Option, see the *Project Design Guide*. For additional best practices information on how you can use MultiSource Option to report on MDX cube data, see the MicroStrategy Tech Note TN36763.

To include MDX cube data in standard reports

- 1 Create a new report. The steps to do this are different for MicroStrategy Developer and Web:

- [In Developer](#)
- [In MicroStrategy Web](#)

In Developer

- a Log in to a project connected to an MDX cube source.
- b From the **File** menu, point to **New**, and then select **Report**. The New Grid dialog box opens.
- c On the **General** tab, select **Blank Report**, and click **OK**. The Report Editor opens.

In MicroStrategy Web

- a Log in to a project connected to an MDX cube source.
 - b Click the **MicroStrategy** icon, and then select **Create Report**. A page with several report creation choices is displayed.
 - c Underneath **Choose data source**, click **Blank Report**. The Report Editor opens.
- 2 Select the attributes to include on the report. Attributes from the relational project can be added by navigating the folders of the project. You can include attributes that are also mapped to data for the MDX cube that you plan to report on.



If you include attributes that are not on the MDX cube, an error is shown when trying to run the report that explains the data is not available at the specified level.

3 Select the metrics to include on the report:

- Metrics from the relational project can be added by navigating the folders of the project.
- To include metrics from MDX cubes:
 - In Developer: From the **Object Browser** drop-down list, select the Data Explorer for an MDX cube source. The MDX cubes for the MDX cube source are displayed. Navigate to an MDX cube. Within an MDX cube, click the **Metrics** folder to view the metrics for an MDX cube. You can then drag and drop the MDX cube metrics onto the report.
 - In Web: In the pane on the left, click **MDX Objects**. MDX cube data sources are displayed. Click the links for the MDX cube sources to navigate to an MDX cube. Within an MDX cube, click the **Metrics** folder to view the metrics for an MDX cube. You can then drag and drop the MDX cube metrics onto the report.

4 Include any other additional objects, such as prompts or filters, as required.

5 Run the report to view the project metrics and MDX cube metrics on the same report.

Switching the MDX cube for an MDX cube report

After creating an MDX cube report, you can switch the MDX cube that is used as the source of data for the report. This can be helpful if another MDX cube has other additional or updated attribute information that could be of importance to the report.

Prerequisites

- To be able to switch to a different MDX cube, the following conditions need to be met:
 - The attributes that are included on the template of the MDX cube report must also be available in the MDX cube you are switching to. If the MDX cubes are from the same MDX source, the attributes are shared between the MDX cubes by default, as described in [Shared MDX cube objects, page 101](#).

However, if you have manually mapped the attributes in either of the MDX cubes to other attributes, either in the MDX cube source or in

the relational project, the attributes in the two MDX cubes need to be mapped to the same objects to be able to switch between MDX cubes for the report.

- The metrics that are included on the template of the MDX cube report must also be available in the MDX cube you are switching to. In addition, you must ensure that the metrics are shared between the two MDX cubes, as described in [Sharing metrics between MDX cubes, page 102](#).

To switch the MDX cube for an MDX cube report

- 1 In MicroStrategy Developer, log in to a project.
- 2 Browse to an MDX cube report, right-click the report, and select **Edit**. The MDX cube report is opened in the Report Editor.
- 3 From the **Data** menu, point to **MDX Cube Options**, and select **Switch MDX Cube**. The Select Destination Object dialog box opens.
- 4 Navigate to the MDX cube that is to act as the new source of data for the report.



Only MDX cubes that meet the requirements described in the prerequisites of this procedure are available for selection. If no MDX cubes are available, you cannot switch the MDX cube for the report until you modify the MDX cubes as necessary.

- 5 Select the MDX cube and click **Open**. The process of switching the report to the new MDX cube is started, and once complete, you are returned to the Report Editor.

You can continue to report on the data, and include objects on the report from the new MDX cube that is connected to the report.

Analyzing data with MDX cube reports

MDX cube reports benefit from much of the same reporting and analysis functionality available for standard MicroStrategy reports. This section discusses the following reporting and analysis features, which have some unique functionality for MDX cube reports:

- [Hierarchies on MDX cube reports, page 155](#)
- [Filters on MDX cube reports, page 160](#)
- [Prompts on MDX cube reports, page 169](#)
- [Drilling on MDX cube reports, page 173](#)
- [Sorting structure elements and preserving order, page 177](#)
- [Sorting on attribute element orders from MDX cube sources, page 179](#)
- [Inheriting MDX cube source formats for metric values, page 181](#)



Once an MDX cube is imported into MicroStrategy, you can use MicroStrategy's Data Mining Services features to perform predictive analysis on your MDX cube data. For steps on how to include predictive analysis with your MDX cube data, refer to the *Advanced Reporting Guide*.

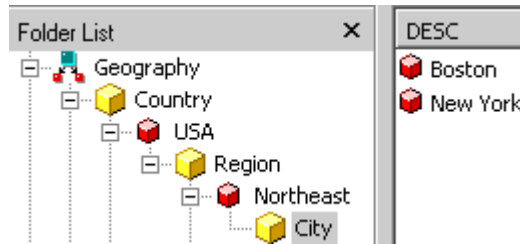
Hierarchies on MDX cube reports

You can include hierarchies of an MDX cube on the templates of MDX cube reports. *Templates* define the columns of data and data formatting displayed on reports and MDX cube reports. Hierarchies can be added to report templates using the same standard techniques to add attributes to a report template. At report run-time, any hierarchies included on an MDX cube report display the attributes that are part of that hierarchy.

When an MDX cube report includes a hierarchy, the attributes that are displayed for that hierarchy are determined by two factors:

- The default number of attributes to display for a hierarchy. This is defined for a hierarchy when a project designer maps data from MDX cube sources to an MDX cube. For a detailed explanation of how to define the default number of attributes to display for a hierarchy, see [Displaying hierarchies on MDX cube reports, page 122](#).
- The attribute level defined in the report filter of the report. You can include an attribute in a report filter that is at a lower level than is set for

its hierarchy. For example, you have a Geography hierarchy with Country, Region, and City attribute levels as shown below.



The lower attribute level of the hierarchy default and the report filter is used as the attribute level displayed on the report. Consider the Geography hierarchy example above. The hierarchy is defined to display one attribute on reports. If the Geography hierarchy is included on a report with no report filter qualifications on attributes of the Geography hierarchy, only Country is displayed for the hierarchy on the report. However, a report that includes a report filter qualification on City requires all attributes down to the level of the report filter qualification to be displayed for the hierarchy. (A *qualification* is the condition in a filter that limits the data to be included on a report.) These two cases are shown in the image below.

Report Description: Sample Cube		Report Description: Sample Cube	
Report Filter (Local Filter): Filter is empty.		Report Filter (Local Filter): ((City = Atlanta, Washington, DC or New Orleans))	
Cache Used: No		Cache Used: No	
Country	Metrics	Profit	
USA		3,683,983	
Web		415,125	

Country	Region	City	Metrics	Profit
	Mid-Atlantic	Washington, DC		342,746
USA	Southeast	Atlanta		262,582
	South	New Orleans		211,257

The two example reports shown below illustrate the other scenario in which the Geography hierarchy has been defined to display all of its attributes on a report with a report filter at a higher attribute level. Even though the second report has a report filter on the Region attribute, all of the Geography

attributes are displayed because the hierarchy is defined to display all attributes down to City.

Report Description: Sample Cube				
Report Filter (Local Filter): Filter is empty.				
Cache Used: No				
Country	Region	City	Metrics	Profit
USA	Northeast	Boston		\$320,601
		New York		\$243,978
	Mid-Atlantic	Washington, DC		\$342,746
	Southeast			
	Central			
	South			
	Northwest			
Web	Web			

Report Description: Sample Cube				
Report Filter (Local Filter): ((Region = Mid-Atlantic,Southeast or South))				
Cache Used: No				
Country	Region	City	Metrics	Profit
USA	Mid-Atlantic	Washington, DC		\$342,746
		Charleston		\$483,623
	Southeast	Atlanta		\$262,582
		Miami		\$226,672
	South	New Orleans		\$211,257
		Memphis		\$124,076

Be aware of the following when using hierarchies on MDX cube reports:

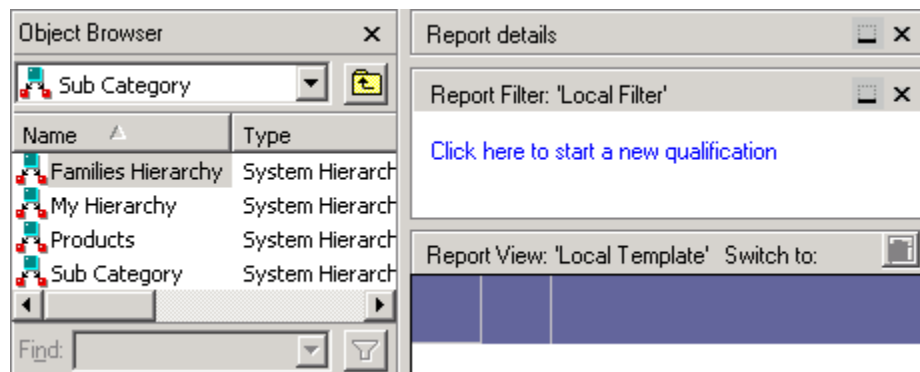
- Only the default report display forms for each attribute are shown when attributes are displayed as part of a hierarchy on a report. To modify the attribute forms that are displayed for each attribute when displayed as part of a hierarchy, you must modify the report display forms for the attributes using the Attribute Editor. For details on defining report display forms, see the *MicroStrategy Project Design Guide*.
- Attributes displayed as part of a hierarchy on a report act as one unit on the grid or graph. This means that all attributes displayed as a hierarchy are pivoted or moved together. However, you can sort each attribute of a hierarchy individually by using the advanced sorting options for a report.
- Hierarchies must display attributes sequentially from the highest level attribute down to the lowest level defined for the hierarchy. This means that to see the lowest level attribute for a hierarchy displayed on the report, the hierarchy must display every attribute for the hierarchy. To see lower level attributes in the hierarchy without displaying the higher

level attributes, add the attributes to the report one-by-one rather than including the hierarchy on the report.

- In SAP BW systems, dimensions can include multiple hierarchies for data display purposes. In MicroStrategy, only one hierarchy per dimension can be represented on an MDX cube report (see [SAP BW dimensions with multiple hierarchies on reports](#) below).
- You can drill on hierarchies included on your MDX cube reports using most of the standard techniques available for drilling on attributes on reports. For more information and best practices on drilling on hierarchies, see [Drilling on hierarchies on MDX cube reports, page 176](#).

SAP BW dimensions with multiple hierarchies on reports

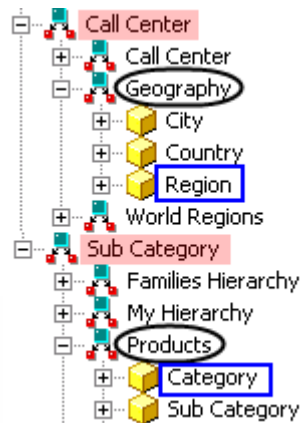
When data from an SAP BW MDX cube source is mapped to MicroStrategy objects, dimensions in SAP BW are mapped to a hierarchy object in MicroStrategy. However, some SAP BW dimensions are mapped to multiple hierarchy objects in MicroStrategy. This occurs because an SAP BW dimension can be broken up into separate MicroStrategy hierarchies for the purposes of formatting and structuring the display of data. For example, in the image shown below, the Sub Category dimension is broken up into four distinct MicroStrategy hierarchies; each hierarchy structures the same data in different ways.



When including hierarchies or attributes from these hierarchies on MicroStrategy reports, you are restricted to including only one hierarchy per dimension. This means that once you add a hierarchy itself or attributes from a hierarchy on a report, all other hierarchies and their related attributes within the same dimension cannot be added to the report or report filter. This restriction also applies to prompts that contain objects from a different hierarchy within a dimension already included on the report. However, you can include other attributes, hierarchies, and prompts on a report or report

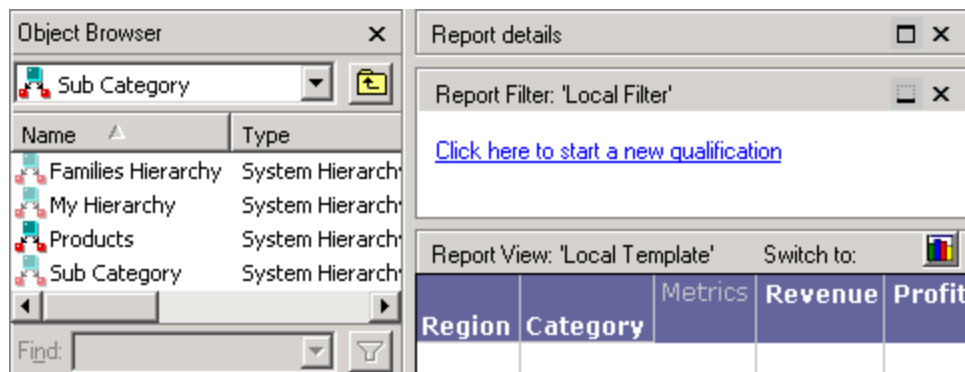
filter as long as the hierarchies associated with these objects are not from the same dimension of the MDX cube.

For example, you have an SAP BW MDX cube with a Geography hierarchy within the Call Center dimension and a Products hierarchy within the Sub Category dimension as shown below.



You include the Region attribute from the Geography hierarchy and the Category attribute from the Products hierarchy on the report.

Notice in the report below that once you include the Category attribute from the Products hierarchy, the rest of the hierarchies within the same dimension are made unavailable. This is because for each dimension you can only include a single hierarchy on a MicroStrategy report's template and filter. The report below also shows that attributes (in this example, Region and Category) from different hierarchies can be included on the same report as long as the hierarchies are in different dimensions.



You could also include the Geography and Products hierarchy objects on the same report together rather than selecting individual attributes. This is possible because the hierarchies are from different dimensions.

Filters on MDX cube reports

In MicroStrategy you can use a *filter* to specify the conditions that report data must meet to be included in report results. For MDX Cube reports, most of the filtering features remain the same as those for standard MicroStrategy reports. For information on filters in general, refer to the Filters chapter in the *MicroStrategy Advanced Reporting Guide*.

In a standard report, the filter is evaluated independently of the report template in most cases. However, in an MDX cube report, due to the nature of MDX, a close relationship exists between the objects in the filter and the objects in the report template. Because of this relationship, qualifications on MDX cube reports are performed at different points during report execution, as described below:



- Qualifications on dimensions that are not included in the template are evaluated as a filter before metric aggregation. For example, the Year attribute is qualified on in the report filter and an attribute in a different dimension such as Store is on the template. In this scenario, each metric value is restricted to consider only those years determined by the filter before aggregation.
- Qualifications on dimensions that are included in the template are applied as a limit after metric aggregation. For example, the Year attribute is on the template and Year is qualified on in the report filter. In this scenario, the report results are restricted as a limit after metric aggregation.

Metric qualifications that have a specific output level are evaluated along with the output level attribute, either before or after aggregation.

The logical relationships between qualifications are set automatically depending on the dimension or dimensions to which the filtered objects belong. The following rules govern the logical operators between qualifications:

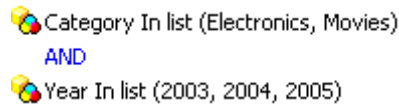
- Qualifications that define attributes in the same dimension are joined by the OR operator.

For example, Customer and Customer Region both belong to the Customer dimension. Therefore, you could have two qualifications joined as follows:

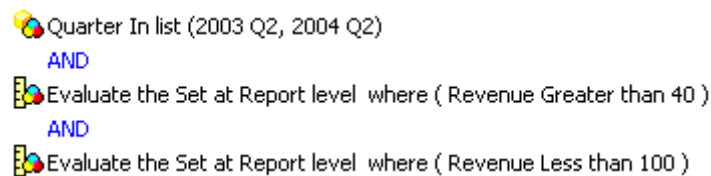
 Customer In list (Addison Don, Adess Merrell, Adler Keith)
OR
 Customer Region In list (France, Germany)

- Qualifications that define attributes in different dimensions are joined by the AND operator.


For example, Category belongs to the Product dimension, and Year belongs to the Time dimension. Therefore, you could have two qualifications joined as follows:



- Metric limits are always joined by the AND operator with all the other qualifications as follows:



You create all filters using the Report Filter pane in Design view in Developer. For information on how to create a filter for an MDX cube report, see [Creating report filter qualifications, page 161](#).

 If you plan to perform data qualifications on MDX property data, you can support mapping MicroStrategy date forms to MDX property data of the date data type. For information on how to support these date forms and qualifications, see [Supporting MDX cube source date data in MicroStrategy, page 117](#).

Creating report filter qualifications

The workflow to create a report filter in an MDX cube report is slightly different than creating a report filter for a standard MicroStrategy report. Rather than creating a report filter qualification by selecting the type of qualification, in MDX cube reports you create report filter qualifications by selecting the fields, operators, and values for the qualification. Each of these components is described below:

- Field:** The field of a qualification is the object that you are qualifying on. You can only choose attributes and metrics which are included in the MDX cube that the MDX cube report is connected to. The type of object you choose determines the type of qualification you create, as follows:
 - Attribute:** Selecting an attribute available in the MDX cube enables you to create an attribute qualification, or a metric set qualification

with the selected attribute as the set qualification's output level (see [Filtering metric data at a specific attribute level, page 165](#)). For attribute qualifications, you can create an element list qualification by selecting the In list or Not in list operator, or you can qualify on any available attribute forms by selecting the Where operator.



Due to MDX restrictions, the MDX cube data that is automatically mapped to the DESC attribute form in MicroStrategy cannot be used in attribute qualifications. To support attribute qualifications on the MicroStrategy DESC attribute form, you must map the DESC form to one of the extended properties of the level in the MDX cube. This mapping can be done in the MDX Cube Catalog (see [Mapping MDX cubes, page 96](#)).

- **Metric:** Selecting a metric available in the MDX cube enables you to create a metric set qualification. To create a metric set qualification with a specific attribute level, you must select an attribute as the first field (see [Filtering metric data at a specific attribute level, page 165](#)). Metric set qualifications filter report data based on metric data restrictions.
- **Operator:** The operator of the qualification determines how to qualify the object of the qualification (field) on the values you have selected. The operators available depend on the object of the qualification. For example, if you are qualifying on an attribute, you can choose from the operators In list, Not in list, and Where. Qualifying on a metric enables you to select from various comparison operators.
 - SAP BW does not support string operators. Therefore, operators such as LIKE, CONTAINS, and so on cannot be used for SAP BW MDX cubes.
- **Value:** The values combined with the operator comprise the qualification criterion used to filter the report. The value of the qualification can be a list of attribute elements (for attribute element list qualifications), a list of values, a single value, or a prompt that returns a value.

The table below lists the types of filter qualifications available for MDX cube reports and the general format required to create them:

Filter Qualification Type	Format	Example
Attribute element list qualifications	Field = <i>attribute</i> Operator = <ul style="list-style-type: none"> • In list • Not in list Value = <i>list of attribute elements</i>	Call Center In list {Atlanta, San Diego}
Attribute form qualifications	Field = <i>attribute</i> Operator = Where Field = <i>attribute form</i> Operator = <i>any available operator</i> Value = <i>any valid value or value prompt</i>	Category Where ID Between 2 AND 4
Metric set qualifications without an output level	Field = <i>metric</i> Operator = <i>any available operator</i> Value = <i>any valid value or value prompt</i>	Revenue Greater than or equal to 100000
Metric set qualifications with an output level	Field = <i>attribute</i> Operator = Where Field = <i>metric</i> Operator = <i>any available operator</i> Value = <i>any valid value or value prompt</i>	Call Center Where Profit Less than 200000

You can only use objects on an MDX cube report that are available within the associated MDX cube. Since report filter qualifications in MDX cube reports depend on the associated MDX cube, you cannot save the qualifications as filters for use with other reports.

Once you create a report filter qualification, you can convert it into a prompt to enable MDX cube report users to select their own filtering criteria. For information on converting report filter qualifications into prompts, see [Prompts on MDX cube reports, page 169](#).


To create a report filter qualification

- 1 Open an MDX cube report for editing or create a new MDX cube report, as described below:
 - To edit an MDX cube report, from Developer, right-click an MDX cube report and select **Edit**. The Report Editor opens.
 - To create a new MDX cube report, see [To create an MDX cube report, page 145](#).
- 2 Add attributes and metrics to the MDX cube report.
- 3 If the Report Filter pane is not displayed, from the **View** menu, select **Report Filter Definition**. The Report Filter pane is displayed.
- 4 In the **Report Filter** pane, click **Click here to start a new qualification**.


Define a report filter qualification

For information on how field, operator, and value components define a report filter qualification, see [Creating report filter qualifications, page 161](#). Be aware that the options you see to create a report filter qualification change depending on your choices for each component.

- 5 Click **Field**, and then select the attribute or metric to qualify on.



If the MDX cube report is connected to an SAP BW MDX cube with dimensions that contain multiple hierarchies, you may only be able to select from a subset of the attributes available in the MDX cube. This is because, within a dimension, only one hierarchy can be represented on the report and report filter. For more information on this restriction, see [SAP BW dimensions with multiple hierarchies on reports, page 158](#).
- 6 Click **Operator**, and then select the operator for the qualification.
- 7 Click **Value**, and then either type a value, select attribute elements, or select a prompt that returns a value.



If you need to import a text file for an attribute element qualification, note that the same rules apply to MDX cube reports as for standard reports. Namely, data in the file needs to be tab-delimited, return-delimited, or list-delimited as specified in the Regional Settings.

The report filter qualification is created.

Filtering metric data at a specific attribute level

The attribute level of a metric set qualification specifies the output level at which the metric is calculated for the qualification. For example, if a metric set qualification is Sales > 100000, Sales could mean sales per day, month, or year. You must specify a time attribute as an output level to clarify the qualification.

By default, metric set qualifications are evaluated using the level of the metric itself. Using the example mentioned above, this means that if the Sales metric is defined to be calculated at the year level, the metric set qualification restricts report results to data that has sales over \$100,000 for a given year. However, metrics commonly do not have a defined level. In this case, metric set qualifications are evaluated at the report level, which is the level defined by the lowest-level attributes on the report.

For example, you create an MDX cube report that displays data for call center, category, and profit. You define a metric set qualification of Profit less than 200,000, which displays all profit data for product categories that have less than \$200,000 of profit for a given call center. A subset of the report results is shown below.

Call Center	Category	Metrics	Profit
Atlanta	Books		\$1,129
	Electronics		\$198,084
	Movies		\$43,866
	Music		\$19,503
San Diego	Books		\$2,502
	Movies		\$98,082
	Music		\$43,780
Salt Lake City	Books		\$390
	Electronics		\$76,741
	Movies		\$16,826
	Music		\$7,646

While this is a valid result based on your filter and the level of the report, you require a different view of data. Rather than returning results for all call center and category combinations that have profits of less than \$200,000, you need to display data only for call centers that have profits less than \$200,000 total across all product categories. In the report example shown above, it is easy to see that while all categories within the Atlanta call center

had profits less than \$200,000, the total profit for all categories in the Atlanta call center is well above \$200,000.

To return your required results, you create a metric set qualification on Profit less than 200,000 with an output level of Call Center (see [To create a metric set qualification with an output level](#) below). A subset of the report results is shown below, with subtotals to verify that the qualification has been calculated at the Call Center level.

Call Center	Category	Metrics	Profit
Salt Lake City	Books		\$390
	Electronics		\$76,741
	Movies		\$16,826
	Music		\$7,646
	Total		\$101,603
Seattle	Books		\$484
	Electronics		\$78,655
	Movies		\$17,447
	Music		\$8,208
	Total		\$104,794

To create a metric set qualification with an output level

This procedure follows the example scenario described above. It also shows the general workflow for creating a metric set qualification with an output level, for MDX cube reports. You can substitute the attributes and metrics described in this procedure with attributes and metrics that are available in your MDX cubes.

- 1 Create a new MDX cube report connected to an MDX cube that contains the attributes Call Center and Category and the metric Profit. For steps to create an MDX cube report, see [To create an MDX cube report, page 145](#).
- 2 Add attributes and metrics to the report. For this example, add Call Center, Category, and Profit to the report.
- 3 If the Report Filter pane is not displayed, from the **View** menu, select **Report Filter Definition**. The Report Filter pane is displayed.
- 4 In the **Report Filter** pane, click **Click here to start a new qualification**.

Define the metric set qualification with an output level

- 5 Click **Field** and select the attribute to use as the output level for the metric set qualification. For this example, browse to and select the Call Center attribute.
- 6 Click **Operator** and select **Where**. Field, Operator, and Value components are displayed.
- 7 Click the new **Field** option and select the metric to qualify on. For this example, select the Profit metric.
- 8 Click the new **Operator** option and select the operator for the qualification. For this example, select **Less than**.
- 9 Click **Value** and then type a value or select a prompt that returns a value. For this example, type 200000.

Your qualification should match the syntax of the qualification shown below:



View the report results

- 10 Run the report.

The report is displayed with the metric set qualification filtering data at the attribute level you selected.

Filtering with static or dynamic date qualifications

You can create static and dynamic date qualifications in MicroStrategy to filter the data in your MDX cube reports. Static date qualifications use a specific date to qualify on data while dynamic date qualifications enable you to use conditions based on the current date to qualify on data.

For example, a dynamic date can be used in a report that examines revenue amounts from the previous two months. This is represented as “today” with an offset of two months. For background information on and examples of dynamic date qualifications, see the *MicroStrategy Advanced Reporting Guide*.

Date data represents a given day using various formats that include the month, day, and year of a given day. In MicroStrategy, this type of data is commonly represented as an attribute form of an attribute that contains data describing individual days.

Before you can filter data using date qualifications as described in the procedure below, the data of your MDX cube report must have some data defined as the date data type. For steps to define data in an MDX cube as the date data type, see [Supporting MDX cube source date data in MicroStrategy, page 117](#).

To create a date qualification on MDX cube data

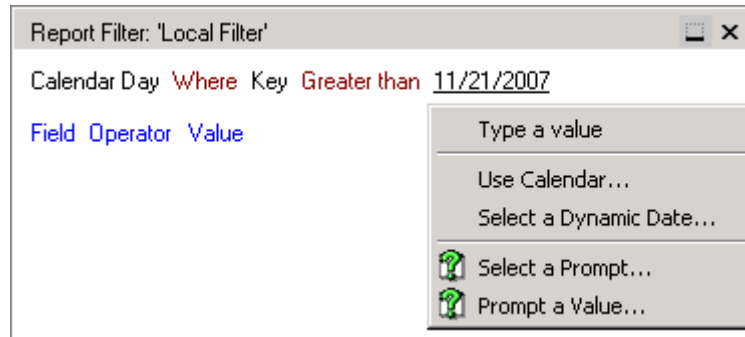
You can substitute the attributes and attribute forms described in this procedure with attributes and attribute forms that are available in your MDX cubes and are defined as the date data type.

- 1 Create a new MDX cube report connected to an MDX cube that contains the attribute Calendar Day. For steps to create an MDX cube report, see [To create an MDX cube report, page 145](#).
- 2 If the Report Filter pane is not displayed, from the **View** menu, select **Report Filter Definition**. The Report Filter pane is displayed.
- 3 Drag-and-drop the attribute to create a date qualification for into the **Report Filter** pane. The attribute is included as the Field of a report filter qualification. For this example, drag-and-drop Calendar Day into the Report Filter pane.
- 4 Click **Operator** and select **Where**. Field, Operator, and Value components are displayed.
- 5 Click the new **Field** option and select the attribute form that is defined with a date data type. For this example, select the Key form.
- 6 Click the new **Operator** option and select the operator for the qualification. You can select any operator except for In list and Not in list, which are operators used to qualify on a list of elements. For this example, select **Greater than**.
- 7 Click **Value**. Options appear to select a dynamic date or use a calendar to select a static date, as shown and described below.



If you do not see these date options, this means the attribute form has not been defined correctly as a date data type. For steps to

define MDX cube data as a date data type, see [Supporting MDX cube source date data in MicroStrategy, page 117](#).



- **Use Calendar:** This option enables you to select a static date from a calendar. You cannot create a dynamic date qualification with this option.
 - **Select a Dynamic Date:** This option opens up the Date Editor, with which you can select a static date from a calendar or create a dynamic date qualification. Dynamic date qualifications are created as an offset of the current date. The Date Editor provides a preview of what your dynamic date qualification would return using the current date as the starting point. For background information on and examples of dynamic date qualifications, see the *MicroStrategy Advanced Reporting Guide*.
 - **Select a Prompt and Prompt a Value:** These options enable you to select a date prompt or create a new date prompt for a user to answer at report runtime to complete the filter qualification. Prompts can use static or dynamic dates to return a list of answers for users to choose from.
- 8 Build the rest of the report by adding attributes, metrics, and any other valid objects to meet your report requirements. Click **Save and Close** to save your changes to the report and close the Report Editor.

You can run the report to verify that the report is returning the required results.

Prompts on MDX cube reports

MicroStrategy automatically converts SAP BW variables into prompts when SAP BW cubes are imported into a project. In addition to these inherited prompts, you can create standard prompts for MDX cube sources in the same way as you can in MicroStrategy reports that access a data warehouse.

For information on how SAP BW variables:

- Are converted into MicroStrategy prompts, see [Converting SAP BW variables into MicroStrategy prompts, page 16](#).
- Are mapped to MicroStrategy prompts using the MDX Cube Catalog, see [Mapping SAP BW variables to MicroStrategy prompts, page 124](#).



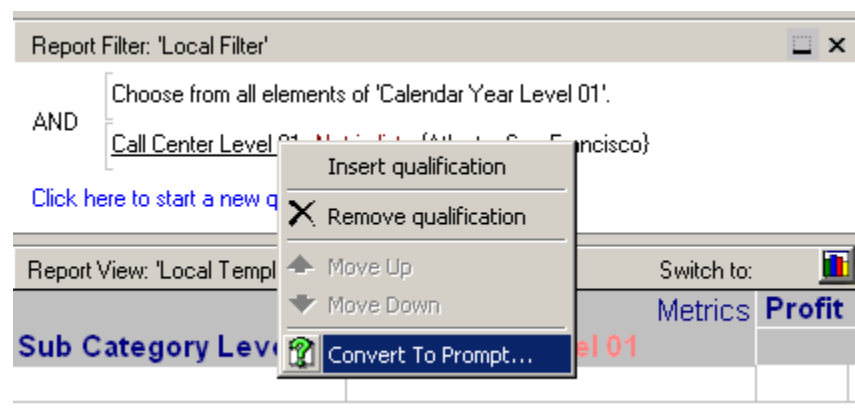
To allow attribute element searches on MDX cube reports for SAP BW cubes that include variables, within your SAP BW system, you must either define the variables as optional, or provide a default answer for the required variables. The MicroStrategy prompt that is mapped to the SAP BW variable can be either optional or required.



You can display prompt details for MDX cube reports just as you can for standard reports by opening the Developer Preferences dialog box, selecting **Reports, Show report details**, and then **Include prompt details**. This feature is especially useful if you want to display a summary of the variable elements that are used to answer the variable prompts.

There are two ways to create a standard prompt for an MDX cube report. You can convert report filters to prompts or create new prompts using the Prompt Generation Wizard. Both of these techniques are described below:

- **Converting report filters to prompts:** After you create a report filter qualification for an MDX cube report, you can convert the filter qualification into a prompt in the Report Editor by right-clicking the filter qualification and selecting **Convert to Prompt**. This process is shown in the image below.



When a report filter qualification is converted into a filter definition prompt, the type of filter definition prompt that is created depends on the type of filter qualification you convert. The table below lists the prompt type created for different types of filter qualifications. For information on

how to create different types of filter qualifications and the formats for each type, see [Creating report filter qualifications, page 161](#).

Filter Qualification Type	Filter Definition Prompt Type Created
Attribute element list qualifications	An attribute element list prompt, based on the attribute of the attribute element list qualification. Any attribute elements selected as values for the filter qualification are converted into default answers for the attribute element list prompt.
Attribute form qualifications	An attribute prompt, based on the attribute of the attribute form qualification. The attribute form qualification is converted into the default answer for the attribute prompt.
Metric set qualifications	A metric prompt, based on the metric of the metric set qualification. The metric set qualification is converted into the default answer for the metric prompt.

- Prompt Generation Wizard:** This procedure is the same as for a standard MicroStrategy prompt. Prompts in MDX cube reports can only access objects that are included in an MDX cube report's associated MDX cube. Therefore, when using the Prompt Generation Wizard to create prompts for an MDX cube report, it is important that the prompt only contain objects from the associated MDX cube. For example, an MDX cube report is associated with MDX cube 1 that has metrics Revenue, Cost, and Profit. Including an object prompt with the Revenue, Cost, and Profit metrics from MDX cube 1 enables users of the report to select which of these metrics they want to view on the report. However, if the object prompt included a metric from an MDX cube that is not associated with the MDX cube report, the prompt cannot be included in the MDX cube report. The same object prompt could be used in other MDX cube reports that are associated with MDX cube 1.

You can create different types of prompts depending on your needs (see [Supported prompt types](#) below).

You can select attributes and metrics for your prompt definitions by browsing to the **Data Explorer** for your MDX cube source.

When you save a prompted report, whether it has inherited prompts converted from SAP BW variables or standard prompts, you can save it as a static or prompted report. See the *MicroStrategy Basic Reporting Guide* for details on saving a prompted report.

Supported prompt types

When you create a prompt for an MDX cube report using the Prompt Generation Wizard, you must select the type of prompt you want to create. While most of the standard prompt types are supported for MDX cube reports, there are some guidelines you should be aware of when creating prompts for your MDX cubes. The table below lists the prompt types supported for MDX cube reports and guidelines on how to create them and include them in MDX cube reports:

Prompt Type	Guidelines For Creation
Filter definition prompt: Choose from all attributes in a hierarchy	<p>Creating the prompt:</p> <ul style="list-style-type: none"> You must choose a specific hierarchy rather than using the results of a search or listing all hierarchies with no restrictions. This ensures that objects included in the prompt all come from the same MDX cube. <p>Including the prompt in an MDX cube report:</p> <ul style="list-style-type: none"> When you include the prompt in the report filter of a report, it must meet the required logical relationships between filter qualifications described in Filters on MDX cube reports, page 160. MDX cube reports associated with an SAP BW MDX cube can only have one hierarchy per dimension on a report. Therefore, you can only add this type of prompt on an MDX cube report associated with an SAP BW MDX cube if it meets the one hierarchy per dimension requirement, which is described in SAP BW dimensions with multiple hierarchies on reports, page 158.
Filter definition prompt: Qualify on an attribute	<p>Creating the prompt:</p> <ul style="list-style-type: none"> You must choose a specific attribute rather than using the results of a search. This ensures that attributes included in the prompt all come from the same MDX cube. <p>Including the prompt in an MDX cube report:</p> <ul style="list-style-type: none"> When you include the prompt in the report filter of a report, it must meet the required logical relationships between filter qualifications described in Filters on MDX cube reports, page 160.
Filter definition prompt: Choose from an attribute element list	<p>Creating the prompt:</p> <ul style="list-style-type: none"> To determine the attribute elements available as prompt answers, you must list all attribute elements or select a list of elements rather than use a filter. This ensures that the attribute elements included in the prompt all come from the same MDX cube. <p>Including the prompt in an MDX cube report:</p> <ul style="list-style-type: none"> When you include the prompt in the report filter of a report, it must meet the required logical relationships between filter qualifications described in Filters on MDX cube reports, page 160.

Prompt Type	Guidelines For Creation
Filter definition prompt: Qualify on a metric	<p>Creating the prompt:</p> <ul style="list-style-type: none"> You must choose a specific metric rather than using the results of a search. This ensures that metrics included in the prompt all come from the same MDX cube. <p>Including the prompt in an MDX cube report:</p> <ul style="list-style-type: none"> When you include the prompt in the report filter of a report, it must meet the required logical relationships between filter qualifications described in Filters on MDX cube reports, page 160.
Object prompts	<p>Creating the prompt:</p> <ul style="list-style-type: none"> You must choose specific objects rather than using the results of a search. This ensures that objects included in the prompt all come from the same MDX cube.
Value prompts	<p>Creating the prompt:</p> <ul style="list-style-type: none"> You can create standard MicroStrategy value prompts to use in MDX cube reports. <p>Including the prompt in an MDX cube report:</p> <ul style="list-style-type: none"> To use a value prompts with MDX cube data, you must define MDX cube data as a data type that matches a value prompt. For instructions on how to define column data types for MDX cube data, see Defining column data types for MDX cube data, page 109. For instructions on how to define data in an MDX cube as the date data type, see Supporting MDX cube source date data in MicroStrategy, page 117.

Drilling on MDX cube reports

You can *drill* in reports to obtain additional information after a report has been executed. For MDX cube reports, drilling is supported within the MDX cube. This means that you can drill up or down within the dimension that the attribute belongs to, as well as in other directions to dimensions included in the same MDX cube.

Drill paths are automatically generated based on the definition of the MDX cube. For information and best practices for drilling on hierarchies on MDX cube reports, see [Drilling on hierarchies on MDX cube reports](#) below.

You cannot drill down into the elements of SAP BW structures. For more information on structures, see [Supporting SAP BW structures, page 18](#).

When you run an MDX cube report in MicroStrategy Web, it is recommended that you use the sub-menu display option to view your drilling options in MDX cube reports. This option allows you to drill in other

directions to all available hierarchies within the MDX cube rather than only allowing you to drill up and down within the current hierarchy.

To display all available hierarchies as drilling options in MicroStrategy Web

- 1 In MicroStrategy Web, log in to a project.
- 2 Click the **MicroStrategy** icon, and then select **Preferences**.
- 3 On the left, beneath **Preferences Level**, select **Project Defaults**.
- 4 On the left, beneath **Preferences**, select **Drill Mode**. The Drill mode page opens.
- 5 Beneath **Display advanced drill options as**, select **Sub menus on the context menu**.
- 6 Click **Apply**. An update confirmation is displayed.

All hierarchies from dimensions not already included on the MDX cube are displayed when drilling on attributes and hierarchies on reports.

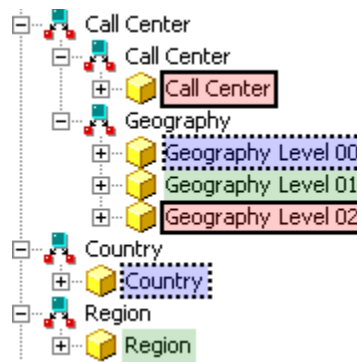
Drill paths available with MDX cube reports

In standard MicroStrategy reports, when you drill on attributes that are part of the relational project schema, drill paths enable you to drill up and down to other attributes that are part of the same hierarchy. The up and down drill paths are based on the system hierarchy.

In MDX cube reports, you can drill across from any attribute to any other attribute available in the MDX cube, but up and down drill paths may not be available. The drill paths available in MDX cube reports are determined by whether attributes are mapped directly to characteristics, or whether attributes are mapped to attributes that are part of a hierarchy.

Up and down drill paths for an attribute are only available for attributes mapped to levels of an SAP BW hierarchy. In SAP BW, hierarchies can be created to organize and logically relate a group of characteristic data (represented as attributes in MicroStrategy). When these hierarchies are imported into MicroStrategy, up and down drill paths are created for the hierarchy's attributes.

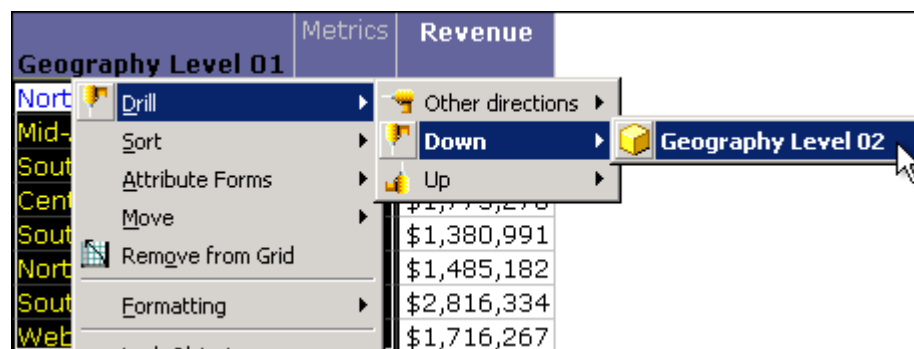
For example, you create an MDX cube report on an SAP BW MDX cube. The MDX cube includes a Geography hierarchy which organizes the characteristics Country, Region, and Call Center (Geography Level 00, Level 01, and Level 02 respectively). Attributes in MicroStrategy are mapped to the characteristics Country, Region, and Call Center of the MDX cube, as shown below.



In this example, the attributes of the Geography hierarchy are intentionally not renamed so that the difference between the attributes that are part of the hierarchy and the attributes mapped directly to the characteristics can be observed. You can rename the hierarchy attributes to match the attributes mapped to the characteristics. For example, you can rename the attribute Geography Level 00 as Country.

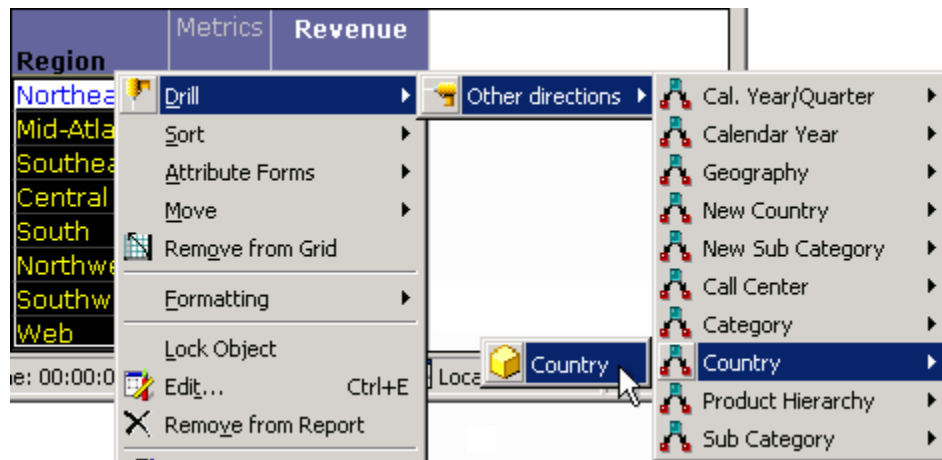
You then create an MDX cube report that includes the Geography Level 01 attribute of the Geography hierarchy that contains regional data. To drill down to call center data, you can drill down on the Geography Level 01 attribute as shown below.

You can also drill up to country data or you can use the other drill path directions to drill to any other attribute available in the MDX cube.



You can also create an MDX cube report that includes the Region attribute mapped directly to the Region characteristic in SAP BW. When you drill on

this attribute, you can still drill to country data, but this is only available as an **Other directions** drill path.



To enable up and down drill paths on your MDX cube reports, you must include the attributes that are part of a hierarchy during the mapping process. Including attributes that are mapped directly to characteristics enables drilling to the same set of attributes available in the MDX cube, but all drilling is available only through the **Other directions** drill path.

Drilling on hierarchies on MDX cube reports

You can drill on hierarchies included on your MDX cube reports using most of the standard techniques available for drilling on attributes in MicroStrategy reports.

MicroStrategy recommends the following best practices for drilling on hierarchies in MDX cube reports:

- Drilling up to a higher-level attribute in a hierarchy is disabled. This is because the highest level attribute is always included on the report since hierarchies must display attributes sequentially from the highest level attribute down to the lowest level defined for the hierarchy.
- You can drill down from a hierarchy to any attribute within the hierarchy not already included on the report.
- You can drill from a hierarchy to attributes within different hierarchies as long as the hierarchy is in a different dimension. This is a requirement of SAP BW's definition of hierarchies and dimensions, which affects what hierarchies can be included on the same report. This is explained in more detail in [SAP BW dimensions with multiple hierarchies on reports](#), page 158.

- If you choose to keep the hierarchy on the report you drill to, the hierarchy object is replaced with the attribute objects that were displayed for the hierarchy in the original report. For example, consider an MDX cube report that includes a Geography hierarchy with the attributes Country, Region, and City. In this example your original report includes Country and Region as part of the Geography hierarchy.

Report objects				Metrics	Profit
Name	Type				
Profit	Metric				
Geography	System Hierarchy				
		Country	Region		
		USA		Northeast	\$564,579
				Mid-Atlantic	\$826,369
				Southeast	\$489,254
				Central	\$428,729
				South	\$335,333
				Northwest	\$358,453
				Southwest	\$681,266
		Web		Web	\$415,125

When you drill down to City and select to keep the hierarchy on the report you drill to, the Geography hierarchy is replaced with the Country and Region attributes that were displayed on the original report, as shown below.

Report objects					Metrics	Profit
Name	Type	Country	Region	City		
City	Attribute					
Country	Attribute					
Region	Attribute					
Profit	Metric					
		USA			Northeast	\$320,601
					New York	\$243,978
					Mid-Atlantic	\$342,746
					Washington, DC	\$483,623
					Charleston	\$262,582
					Southeast	\$226,672
					Atlanta	\$323,583
					Miami	
					Central	
					Milwaukee	

Since the attributes are included in the drilled to report, you can then use them individually as you can use any attributes on reports. For example you can pivot them on the grid, or move them to the page-by area.

Sorting structure elements and preserving order

In SAP BW, when you create a characteristic structure and its structure elements, you can re-order the structure elements as required. When characteristic structures are imported into MicroStrategy as attributes, the order of the structure elements in your SAP BW MDX cube source is preserved as the default order of the attribute elements. This maintains a

consistent view of your data across your SAP BW MDX cube source and MicroStrategy.

For example, the report shown below includes a Regions attribute mapped to a characteristic structure.

Regions	Metrics	Profit	Revenue
East Region		\$ 78,276.00	\$ 323,439.00
West Region		\$ 76,904.00	\$ 320,748.00
East + West		\$ 155,180.00	\$ 644,187.00
East/All		50.44%	50.21%
West/All		49.56%	49.79%

The attribute elements East Region, West Region, East + West, East/All, and West/All are displayed in the same order that is available in the SAP BW MDX cube source. If the order of structure elements has changed in your SAP BW MDX cube source, you can update the MDX cube structure to apply these changes in MicroStrategy (see [Updating MDX cube structure, page 92](#)).


You can also use this structure element order to sort the information on the report. For example, you can perform a descending sort on this structure element order to display the structure elements in the opposite order, as shown below.

Regions	Metrics	Profit	Revenue
West/All		49.56%	49.79%
East/All		50.44%	50.21%
East + West		\$ 155,180.00	\$ 644,187.00
West Region		\$ 76,904.00	\$ 320,748.00
East Region		\$ 78,276.00	\$ 323,439.00

Steps to perform a sort on the default structure element order are described in the procedure below.

To sort on structure element orders

- 1 In MicroStrategy Developer, log in to a project.
- 2 Browse to an MDX cube report, right-click the report, and select **Run**. The MDX cube report is executed.

- 3 From the **Data** menu, select **Advanced Sorting**. The Sort dialog box opens.
- 4 Select the **Rows** tab, and then click **Add**.
- 5 From the drop-down list in the **Sort By** column, select the attribute mapped to the structure.
- 6 From the drop-down list in the **Criteria** column, select **Source Order**.
 Source Order is an attribute form automatically created for the attribute mapped to an SAP BW structure to preserve the structure element order.
- 7 From the drop-down list in the **Order** column, select **Ascending** or **Descending**.
- 8 Click **OK** to save your changes and return to the report.
- 9 Re-execute the report to apply the new sort.

Sorting on attribute element orders from MDX cube sources

The order of MDX cube data mapped to attribute elements in MicroStrategy can be defined to include the same order of the data in your MDX cube source. If the order is defined in this way, as described in [Preserving attribute element orders from MDX cube sources, page 113](#), you can sort MDX cube reports so that the data is in the same order as it exists in the MDX cube source.

Sorting MDX cube reports to represent data as it exists in your MDX cube source can be helpful in various scenarios. For example, this can help support financial balance sheet analysis in which you always want to see your accounts receivable information above your accounts payable information.

Steps to perform a sort on the attribute element order as it exists in your MDX cube source are described in the procedure below.

Prerequisites

- You can only sort an MDX cube report based on the order in your MDX cube source if the order has been integrated into MicroStrategy. For information on defining MDX cubes to integrate their order of data into

MicroStrategy, see [Preserving attribute element orders from MDX cube sources, page 113](#).

To sort on attribute element orders from MDX cube sources

- 1 In MicroStrategy Developer, log in to a project.
- 2 Browse to an MDX cube report, right-click the report, and then select **Run**. The MDX cube report is executed.
- 3 From the **Data** menu, select **Advanced Sorting**. The Sort dialog box opens.
- 4 Select the **Rows** tab, and click **Add**.
- 5 From the drop-down list in the **Sort By** column, select an attribute on the MDX cube report.

If you are sorting multiple attributes in a hierarchy and want to replicate the order that exists in your MDX cube source, you should sort on higher level attributes first. For example, if you have attributes for Year, Quarter, and Day, you should first sort on Year, then on Quarter, and then on Day.

- 6 From the drop-down list in the **Criteria** column, select **Source Order**.

Source Order is an attribute form automatically created for any attributes within dimensions that are defined to integrate the order of their data into MicroStrategy. If there is no Source Order form, the order information has not been integrated. For information on defining MDX cubes to integrate their order of data into MicroStrategy, see [Preserving attribute element orders from MDX cube sources, page 113](#).

- 7 From the drop-down list in the **Order** column, select **Ascending**. An ascending sort on the Source Order form displays the data in the same order that exists in the MDX cube source.
- 8 Click **OK** to save your changes and return to the report.
- 9 Re-execute the report to apply the new sort.

Inheriting MDX cube source formats for metric values

You can specify for the metric values in MicroStrategy MDX cube reports to inherit their value formatting from an MDX cube source. This enables MicroStrategy MDX cube reports to use the same data formatting available in your MDX cube source. It also maintains a consistent view of your MDX cube source data in MicroStrategy.

Inheriting value formats from your MDX cube source also enables you to apply multiple value formats to a single MicroStrategy metric. Normally, a MicroStrategy metric can only have one value format for all of its values. For example, all values for the Profit metric are displayed with a currency format.



Custom groups and consolidations can apply multiple value formats for a metric. However, custom groups and consolidations are not available for MDX cube reports. You can however use custom groups and consolidations on MDX cube data if you create a data mart from an MDX cube report. For information on creating data marts from MDX cube reports, see [Creating data marts of MDX cube data, page 141](#).

For information on custom groups and consolidations, see the *MicroStrategy Advanced Reporting Guide*.

However, some MDX cube sources allow data that can be mapped to MicroStrategy metrics to have more than one value format. For example, profit data in your SAP BW MDX cube source has some values displayed with a currency format, while other values are displayed with a percentage format.

This scenario is common when reports include characteristic structures mapped to MicroStrategy attributes. These structures can perform calculations that return various value formats. This is shown in the report below, which includes a Regions attribute mapped to a characteristic structure:


Regions	Metrics	Profit	Revenue
East Region		\$ 78,276.00	\$ 323,439.00
West Region		\$ 76,904.00	\$ 320,748.00
East + West		\$ 155,180.00	\$ 644,187.00
East/All		50.44%	50.21%
West/All		49.56%	49.79%

Another common scenario in SAP BW MDX cube sources is when formats are defined to reflect the locale of the data. This also requires multiple value formats for MicroStrategy metric data.

For example, in SAP BW you can define your profit data to return currency formats based on the locale of the data. You can then map this data to a Profit metric and a Country attribute in MicroStrategy. When you include Country and Profit on a report, you can select whether to inherit MDX cube source formatting. If you choose to inherit formatting, currency formats are applied based on the locales listed for the Country attribute.

To inherit MDX cube source formats for metric values

- 1 In MicroStrategy Developer, log in to a project.

 VLDB properties cannot be modified from MicroStrategy Web.
- 2 Browse to an MDX cube report, right-click the report, and then select **Run**. The MDX cube report is executed.
- 3 From the **Data** menu, select **VLDB Properties**. The VLDB Properties Editor opens.
- 4 From the **Tools** menu, select **Show Advanced Settings**.
- 5 In the **VLDB Settings** list, expand **MDX**, and then select **MDX Cell Formatting**.
- 6 Clear the **Use default inherited value** check box.
- 7 Select one of the following options:
 - **MDX metric values are formatted per column:** If you select this option, MDX cube source formatting is not inherited. An MDX cube report that is defined to format metric values individually for each column is shown below:

Regions	Metrics	Profit	Revenue
East Region		78276	323439
West Region		76904	320748
East + West		155180	644187
East/All		0.5044	0.5021
West/All		0.4956	0.4979

You can then format each metric on the MDX cube report to use only one value format for all the metric values.

- **MDX metric values are formatted per cell:** If you select this option, MDX cube source formatting is inherited. Metric value formats are

determined by the formatting that is available in the MDX cube source, and a metric can have various value formats. The report shown below is defined to inherit formatting from the MDX cube source, and applies multiple formats to the values of metrics.

Regions	Metrics	Profit	Revenue
East Region		\$ 78,276.00	\$ 323,439.00
West Region		\$ 76,904.00	\$ 320,748.00
East + West		\$ 155,180.00	\$ 644,187.00
East/All		50.44%	50.21%
West/All		49.56%	49.79%



You can use standard MicroStrategy techniques to further format the inherited formatting for metric values. However, you must use default Number formatting. Modifying the Number formatting for the metric values to anything other than default removes any inherited formatting from the MDX cube source. Number formats include general, fixed, currency, percent, and so on.

- 8 Click **Save and Close** to save your changes and close the VLDB Properties Editor.

The MDX Cell Formatting VLDB property can also be defined for database instances connected to MDX cube sources. Defining this VLDB property for a database instance applies the definition as the default for all MDX cube reports associated with the database instance. For steps to define this VLDB property for a database instance, see [Inheriting MDX cube source formats for metric values, page 65](#).

Using MDX cube reports to filter other reports

MDX cube reports can be used to filter other standard reports in MicroStrategy. This can be supported by using a MDX cube report as shortcut-to-a-report qualification on a standard report. For information on shortcut-to-a-report qualification, see the *Advanced Reporting Guide*.



In Developer, with a standard report open, you select **Add a Shortcut to a Report** to access the report as filter functionality.

However, there are additional configurations and requirements to allow MDX cube reports to be used as a shortcut-to-a-report qualification, as described below:

- The MDX cube report must map data to project attributes rather than managed object attributes. This is required for the standard report to recognize the data that is available on the MDX cube report. For information on mapping data in MDX cube reports to project attributes, see [Mapping MDX cube data to project attributes, page 103](#).
- You must have the Execute Multiple Source Report privilege, which is part of the MultiSource Option group of privileges. For information on MultiSource Option, see the *Project Design Guide*.

To add an MDX cube report as a shortcut-to-a-report qualification in a standard report, open the standard report in Developer. In the **Report Filter** area, double-click the arrow to add a new qualification. In the **Filtering Options**, select **Add a Shortcut to a Report**. Click **OK**. Click the ... (browse button) to select an MDX cube report, and click **OK**. Click **OK** again to create the filter qualification.

GLOSSARY

aggregate function A numeric function that acts on a column of data and produces a single result. Examples include SUM, COUNT, MAX, MIN, and AVG.

attribute A data level defined by the system architect and associated with one or more columns in a data warehouse lookup table. Attributes include data classifications like Region, Order, Customer, Age, Item, City, and Year. They provide a means for aggregating and filtering at a given level.

See also:

- **attribute element**
- **attribute form**

attribute element A unique set of information for an attribute, defined by the attribute forms. For example, New York and Dallas are elements of the attribute City; January, February, and March are elements of the attribute Month.

attribute form One of several columns associated with an attribute that are different aspects of the same thing. ID, Name, Last Name, Long Description, and Abbreviation could be forms of the attribute Customer. Every attribute supports its own collection of forms.

business intelligence (BI) system A system that facilitates the analysis of volumes of complex data by providing the ability to view data from multiple perspectives.

conditionality Conditionality of a metric enables you to associate an existing filter object with the metric so that only data that meets the filter conditions is included in the calculation.

Data Explorer A portion of the interface used to browse through data contained in the warehouse. Users can navigate through hierarchies of attributes that are defined by the administrator to find the data they need.

data source A data source is any file, system, or storage location which stores data that is to be used in MicroStrategy for query, reporting, and analysis.

A data warehouse can be thought of as one type of data source, which refers more specifically to using a database as your data source. Other data sources include text files, Excel files, and MDX cube sources such as SAP BW, Microsoft Analysis Services, Oracle Essbase, and IBM Cognos TM1.

See also:

- **data warehouse**
- **MDX cube source**

data warehouse 1) A database, typically very large, containing the historical data of an enterprise. Used for decision support or business intelligence, it organizes data and allows coordinated updates and loads.

2) A copy of transaction data specifically structured for query, reporting, and analysis.

See also **data source**.

database instance 1. A MicroStrategy object created in MicroStrategy Developer that represents a connection to the warehouse. A database instance specifies warehouse connection information, such as the data warehouse DSN, Login ID and password, and other data warehouse specific information.

2. Database server software running on a particular machine. Although it is technically possible to have more than one instance running on a machine, there is usually only one instance per machine.

description column Optional columns that contain text descriptions of attribute elements.

drill A method of obtaining supplementary information after a report has been executed. The new data is retrieved by re-querying the Intelligent Cube or database at a different attribute or fact level.

filter A MicroStrategy object that specifies the conditions that the data must meet to be included in the report results. Using a filter on a report narrows the data to consider only the information that is relevant to answer your business question, since a report queries the database against all the data stored in the data warehouse.

A filter is composed of at least one qualification, which is the actual condition that must be met for the data to be included on a report. Multiple qualifications in a single filter are combined using logical operators. Examples include "Region = Northeast" or "Revenue > \$1 million".

A filter is normally implemented in the SQL WHERE clause.

hierarchy A set of attributes defining a meaningful path for element browsing or drilling. The order of the attributes is typically—though not always—defined such that a higher attribute has a one-to-many relationship with its child attributes.

ID column A column that contains attribute element identification codes. All attributes must have an ID column.

managed object A schema object unrelated to the project schema, which is created by the system and stored in a separate system folder. Managed objects are used to map data to attributes, metrics, hierarchies and other schema objects for Freeform SQL, Query Builder, and MDX cube reports.

See also **schema object**

MDX cube An MDX cube is a collection or set of data retrieved from an MDX cube source, which is imported into MicroStrategy and mapped to various objects to allow query, reporting, and analysis on the data.

See also **MDX cube source**.

MDX cube report The central focus for MicroStrategy users to query, analyze, and visually present data from MDX cube sources in a manner that answers and evaluates their business questions. MDX cube reports provide the same data display and analysis functionality as standard MicroStrategy reports, but rather than reporting on data from a relational data warehouse, MDX cube reports report on data from MDX cube sources.

See also **report**

MDX cube source When integrated with MicroStrategy, the third-party tools SAP BW, Microsoft Analysis Services, Oracle Essbase, and IBM Cognos TM1 are referred to as MDX cube sources. You can import and map data from these different MDX cube sources in MicroStrategy to query, report on, and analyze data with MicroStrategy.

MicroStrategy can integrate with MDX cube source data as well as access data from a relational database concurrently.

See also:

- **MDX cube**
- **data source**

metadata A repository whose data associates the tables and columns of a data warehouse with user-defined attributes and facts to enable the mapping of the business view, terms, and needs to the underlying database structure. Metadata can reside on the same server as the data warehouse or on a different database server. It can even be held in a different RDBMS.

metric 1) A business calculation defined by an expression built with functions, facts, attributes, or other metrics. For example:
`sum(dollar_sales)` or `[Sales] - [Cost]`

2) The MicroStrategy object that contains the metric definition.

object Conceptually, an object is the highest grouping level of information about one concept, used by the user to achieve the goal of specified data analysis. More concretely, an object is any item that can be selected and manipulated, including folders, reports, facts, metrics, and so on.

project 1) The highest-level intersection of a data warehouse, metadata repository, and user community, containing reports, filters, metrics, and functions.

2) An object containing the definition of a project, as defined in (1). The project object is specified when requesting the establishment of a session.

project source Defines a connection to the metadata database and is used by various MicroStrategy components to access projects. A direct project source is a two-tier connection directly to a metadata repository. A server project source is a three-tier connection to a MicroStrategy Intelligence Server. One project source can contain many projects and the administration tools found at the project source level are used to monitor and administer all projects in the project source.

prompt 1) MicroStrategy object in the report definition that is incomplete by design. The user is asked during the resolution phase of report execution to provide an answer that completes the information. A typical example with a filter is choosing a specific attribute on which to qualify.

2) In general, a window requesting user input, as in “type login ID and password at the prompt.”

qualification The actual condition that must be met for data to be included on a report. Examples include “Region = Northeast” or “Revenue > \$1 million”. Qualifications are used in filters and custom groups. You can create multiple qualifications for a single filter or custom group, and then set how to combine the qualifications using the logical operators AND, AND NOT, OR, and OR NOT.

report The central focus of any decision support investigation, a report allows users to query for data, analyze that data, and then present it in a visually pleasing manner.

See also **MDX cube report**

schema 1) The set of tables in a data warehouse associated with a logical data model. The attribute and fact columns in those tables are considered part of the schema itself.

2) The layout or structure of a database system. In relational databases, the schema defines the tables, the fields in each table, and the relationships between fields and tables.

See also **schema object**

schema object A MicroStrategy object created, usually by a project designer, that relates the information in the logical data model and physical warehouse schema to the MicroStrategy environment. These objects are developed in MicroStrategy Architect, which can be accessed from MicroStrategy Developer. Schema objects directly reflect the warehouse

structure and include attributes, facts, functions, hierarchies, operators, partition mappings, tables, and transformations.

See also **managed object**

template A MicroStrategy object that serves as a base on which you can build other objects of the same type. You can create a template for almost any kind of MicroStrategy object, such as filters or reports.

- Object templates allow you to start with a predefined structure when creating a new object. You can use object templates for many MicroStrategy objects, including metrics, documents, reports, and report templates.
- Report templates define the layout of general categories of information in a report. In a report template, you specify the information that you want to retrieve from your data source, and the way that you want the data to be displayed in Grid view. A report template does not include filter information. Report templates are often referred to as just as templates.

INDEX

A

- aggregate function [defined on 130](#)
 - using with a metric in an MDX cube [132](#)
- Analysis Services 2005/2008 conversion to MicroStrategy
 - cube [24](#)
- Analysis Services 2005/2008/2012
 - catalog [55, 58](#)
 - connecting to [51](#)
 - DSI [55, 58](#)
 - hierarchy [28](#)
 - relating objects to MicroStrategy [24](#)
 - single sign-on [68](#)
 - URL [55, 58](#)
- Analysis Services 2005/2008/2012 conversion to MicroStrategy [24, 29](#)
 - database [25](#)
 - database instance [54, 57](#)
 - dimension [27](#)
 - level [28](#)
 - member [29](#)
 - member property [29](#)
 - perspective [26](#)
- application for Oracle Essbase [20](#)
- architecture, MicroStrategy [3](#)
- attribute [defined on 12](#)
 - effects on import behavior [86](#)
 - for Oracle Essbase [23](#)
 - for SAP BW [15](#)
 - for TM1 [34](#)
 - mapping to an MDX cube [103](#)
 - MicroStrategy to Analysis Services 2005/2008/2012 conversion [28](#)
 - MicroStrategy to TM1 [34](#)
 - sorting element [113, 179](#)
- attribute element [defined on 15](#)
 - for Analysis Services 2005/2008/2012 [29](#)
 - for Oracle Essbase [23](#)
 - for SAP BW [15](#)
 - for TM1 [34](#)
 - sorting [113, 179](#)
 - updating order from an MDX cube [116](#)
- attribute form [defined on 16](#)
 - for Analysis Services 2005/2008/2012 [29](#)
 - for Oracle Essbase [24](#)
 - for SAP BW [16](#)
 - for TM1 [34](#)

authenticating
 imported SAP BW users [70](#)
 single sign-on to Analysis Services [68](#)
 single sign-on to SAP BW [69](#)
authenticating an MDX cube report [67](#)

B

best practices
 mapping MDX cubes [103](#)
 mapping MDX cubes to project attributes [107](#)
business intelligence [defined on 1](#)

C

catalog
 for Analysis Services
 2005/2008/2012 [26, 31](#)
 for Oracle Essbase [20](#)
 for SAP BW [12](#)
characteristic
 importing [85](#)
 in SAP BW [8](#)
characteristic attribute vs. attribute form [16](#)
characteristic value [15](#)
column data type definition for an MDX cube [109](#)
compound metric, creating for an MDX cube [129](#)
conditionality [defined on 135](#)
connecting
 to Analysis Services
 2005/2008/2012 [51](#)
 to Hyperion Essbase 9.3.1 [47](#)
 to TM1 [56](#)
cube
 See also MDX cube.
cube for TM1 [32](#)

D

Data Explorer [defined on 84](#)
data source [defined on 2](#)
data warehouse [defined on 1](#)
database
 for Analysis Services
 2005/2008/2012 [25](#)
 for Oracle Essbase [21](#)
 for TM1 [31](#)
 instance. See database instance.
database instance [defined on 4](#)
 creating for Analysis Services
 2005/2008/2012 [54, 57](#)
 creating for Hyperion Essbase 9.3.1 [49](#)
 creating for SAP BW [39](#)
 creating for SAP BW (UNIX/Linux) [44](#)
 removing for an MDX cube source [61](#)
description column [defined on 99](#)
dimension
 for Analysis Services
 2005/2008/2012 [27](#)
 for Oracle Essbase [21, 22](#)
 for SAP BW [14](#)
 for TM1 [32, 33](#)
drill [defined on 173](#)
drilling
 hierarchy on an MDX cube report [176](#)
 MDX cube report [173](#)

E

element for TM1 [34](#)
error, incompatible data type [113](#)
Essbase. See Oracle Essbase.

F

filter [defined on 160](#)
 date data [167](#)

MDX cube report [160](#)

H

hierarchy [defined on 14](#)
 drilling on [176](#)
 for Analysis Services
 2005/2008/2012 [28](#)
 for Oracle Essbase [22](#)
 for SAP BW [14](#)
 for TM1 [33](#)
 in SAP BW [8](#)
 on an MDX cube report [122](#)
 on an MDX cube reports [155](#)
 unbalanced and ragged [120](#)

Hyperion Essbase
 catalog [50](#)
 connecting to 9.3.1 [47](#)
 database instance for 9.3.1 [49](#)
 DSI (DataSourceInfo) [50](#)
 importing measure structure [88](#)
 URL [50](#)

I

ID column [defined on 99](#)
importing
 characteristic [85](#)
 MDX cube [79, 80](#)
 MDX cube before report creation [81](#)
 MDX cube during report creation [91](#)
InfoCube [11, 12](#)
InfoObject in SAP BW [7](#)
InfoProvider in SAP BW [7](#)
Intelligent Cube
 creating based on MDX cube [148](#)

J

Java Connector [38, 42](#)

K

Kerberos single sign-on
 authentication [68, 69](#)
key date variable support [126](#)
key figure in SAP BW [8](#)

L

level
 for Analysis Services
 2005/2008/2012 [28](#)
 for Oracle Essbase [23](#)
 for TM1 [33](#)
 importing [85](#)
 virtual [15](#)
locating an MDX cube [92](#)

M

managed object [defined on 95](#)
 in an MDX cube [95](#)
mapping [96](#)
 MDX cube [79](#)
MDX cube [defined on 5](#)
 best practices for mapping to project
 attributes [107](#)
 creating a compound metric [129](#)
 creating a metric with custom
 MDX [129](#)
 data order preservation [113, 179](#)
 defining column data types [109](#)
 deleting a compound metric [140](#)
 for Analysis Services
 2005/2008/2012 [26](#)
 for Oracle Essbase [21](#)
 for SAP BW [12](#)

- for TM1 [32](#)
- importing [80](#)
- importing before report creation [81](#)
- importing during report creation [91](#)
- importing measure structure [88](#)
- Intelligent Cube based on [148](#)
- managed object [95](#)
- mapping [96](#)
- mapping best practices [103](#)
- mapping to project attributes [103](#)
- MicroStrategy Data Mining Services [101](#)
- MultiSource Option [150](#)
- partially mapped [106](#)
- predictive analysis [101](#)
- removing [84](#)
- renaming [92](#)
- searching for [92](#)
- shared objects [101](#)
- sharing metrics [102](#)
- switching for report [153](#)
- synchronizing object names [83](#)
- unbalanced and ragged hierarchy [120](#)
- unmapped [105](#)
- updating [92](#)
- updating attribute element order [116](#)
- using a prompt within a custom MDX metric [138](#)
- MDX Cube Catalog [79](#)
 - SAP BW variable [124](#)
- MDX Cube Editor [98](#), [103](#)
- MDX cube report [defined on 143](#)
 - authentication [67](#)
 - creating [144](#)
 - drill path [174](#)
 - drilling [173](#)
 - drilling on a hierarchy [176](#)
 - filter [160](#)
 - hierarchy on [122](#), [155](#)
 - incompatible data type error [113](#)
 - Intelligent Cube report [148](#)
 - managed object [95](#)
 - MultiSource Option [150](#)
 - prompt on [169](#)
 - shortcut-to-a-report qualification [183](#)
 - supported prompt types [172](#)
 - switching MDX cube [153](#)
 - troubleshooting [147](#)
- MDX cube report, MultiSource Option [183](#)
- MDX cube source [defined on 1](#)
 - conversion [1](#)
 - data order preservation [113](#), [179](#)
 - metric inheriting format from [181](#)
 - removing database instance [61](#)
 - supporting date data [117](#)
 - synchronizing object names [83](#)
- MDX metric customization [132](#)
- member
 - for Analysis Services
 - 2005/2008/2012 [29](#)
 - for Oracle Essbase [23](#)
- member property
 - for Analysis Services
 - 2005/2008/2012 [29](#)
- metadata [defined on 4](#)
- metadata model
 - Oracle Essbase [19](#)
 - SAP BW [10](#)
- metric [defined on 12](#)
 - creating a compound metric for an MDX cube [129](#)
 - creating with custom MDX [129](#)
 - deleting a compound metric from an MDX cube [140](#)
 - filtering at an attribute level [165](#)
 - importing measure structure [88](#)

- inheriting MDX cube source
 - format [181](#)
 - using a prompt within custom MDX [138](#)
 - metrics
 - sharing between MDX cubes [102](#)
 - Microsoft Analysis Services 2005. See Analysis Services 2005/2008/2012.
 - Microsoft Analysis Services 2008. See Analysis Services 2005/2008/2012.
 - MicroStrategy
 - architecture [3](#)
 - object model [5](#)
 - MicroStrategy Data Mining Services
 - creating in an MDX cube [101](#)
 - MicroStrategy MDX Cube Provider [48, 52](#)
 - connecting to TM1 [56](#)
 - MicroStrategy to Analysis Services 2005/2008/2012 relation [24](#)
 - attribute [28](#)
 - attribute element [29](#)
 - attribute form [29](#)
 - catalog [26, 31](#)
 - dimension [27](#)
 - hierarchy [28](#)
 - MDX cube [26](#)
 - MicroStrategy to Oracle Essbase relation [18](#)
 - attribute [23](#)
 - attribute element [23](#)
 - attribute form [24](#)
 - catalog [20](#)
 - dimension [22](#)
 - hierarchy [22](#)
 - MDX cube [21](#)
 - MicroStrategy to SAP BW relation [9](#)
 - attribute [15](#)
 - attribute element [15](#)
 - attribute form [16](#)
 - catalog [12](#)
 - dimension [14](#)
 - hierarchy [14](#)
 - MDX cube [12](#)
 - MicroStrategy to TM1 relation [29](#)
 - attribute [34](#)
 - attribute element [34](#)
 - attribute form [34](#)
 - dimension [33](#)
 - MDX cube [32](#)
 - MultiDimensional Expression [2](#)
 - MultiProvider in SAP BW [7](#)
 - MultiSource Option
 - MDX cube reports [183](#)
 - reporting on MDX cubes [150](#)
- ## O
- object [defined on 12](#)
 - object model
 - in MicroStrategy 8 [5](#)
 - using SAP direct access [5](#)
 - ODS object [7](#)
 - OLAP cube source. See MDX cube source.
 - OLAP cube. See MDX cube.
 - OLAP, BAPI certification [36](#)
 - Operational Data Store object. See ODS object.
 - Oracle Essbase
 - metadata model [19](#)
 - relating objects to MicroStrategy [18](#)
 - Oracle Essbase conversion to MicroStrategy [18](#)
 - application [20](#)
 - database [21](#)
 - dimension [21, 22](#)
 - level [23](#)
 - member [23](#)
 - Oracle Hyperion Essbase. See Oracle Ess-

base.

P

perspective for Analysis Services

2005/2008/2012 [26](#)

project [defined on](#) [1](#)

project source [defined on](#) [35](#)

prompt [defined on](#) [17](#)

converted from a report filter [170](#)

on an MDX cube report [169](#)

re-using in documents [128](#)

supported types [172](#)

using in a metric with custom
MDX [138](#)

Q

qualification [defined on](#) [156](#)

query cube in SAP BW [7](#)

R

ragged hierarchy [120](#)

relating an object to MicroStrategy

from Analysis Services

2005/2008/2012 [24](#)

from Oracle Essbase [18](#)

from SAP BW [9](#)

from TM1 [29](#)

relational schema, mapping to an MDX
cube [103](#)

removing an MDX cube [84](#)

report [defined on](#) [143](#)

report filter converted to a prompt [170](#)

S

SAP BI. See SAP BW.

SAP BW

authentication [70](#)

characteristic [8](#)

configuring SAP.sh [43](#)

connecting to [38](#)

connecting to, on UNIX/Linux [42](#)

connecting to, on Windows [38](#)

creating users and roles [70](#)

database instance [39, 44](#)

drilling [173](#)

hierarchy [8](#)

importing users and roles [70](#)

InfoObject [7](#)

InfoProvider [7](#)

Java Connector [38, 42](#)

key date variable support [126](#)

key figure [8](#)

MDX Cube Catalog [79](#)

metadata model [10](#)

MultiProvider [7](#)

ODS object [7](#)

query cube [7](#)

relating an object to MicroStrategy [9](#)

single sign-on [69](#)

structure [18](#)

structure sorting [177](#)

terminology [7](#)

variable [8, 16, 124](#)

variable properties [125](#)

variable with a qualification [127](#)

SAP BW conversion to MicroStrategy [9](#)

characteristic [13](#)

characteristic attribute [16](#)

characteristic value [15](#)

hierarchy [14](#)

InfoCube [11, 12](#)

virtual level [15](#)

SAP.sh, configuring [43](#)

schema [defined on](#) [4](#)

- schema object [95](#)
- searching for an MDX cube [92](#)
- shortcut-to-a-report qualification for an MDX cube report [183](#)
- single sign-on to Analysis Services [68](#)
- single sign-on to SAP BW [69](#)
- structure
 - in SAP BW query cube [18](#)
 - sorting and preserving element order [177](#)
- support. See technical support.
- synchronizing MDX cube object names [83](#)

T

- technical support [xxi](#)
- template [defined on 155](#)
- TM1
 - connecting to [56](#)
 - hierarchy [33](#)
 - MicroStrategy MDX Cube Provider [56](#)
 - relating objects to MicroStrategy [29](#)
- TM1 conversion to MicroStrategy
 - cube [29, 32](#)
 - database [31](#)
 - dimension [32](#)
 - element [34](#)
 - level [33](#)
 - member property [34](#)
- troubleshooting an MDX cube report [147](#)

U

- unbalanced hierarchy [120](#)
- UNIX/Linux, connecting to SAP BW [42](#)
- URL
 - for Analysis Services 2005/2008/2012 [55, 58](#)
 - for Hyperion Essbase [50](#)

V

- variable
 - including/excluding [17](#)
 - key date support [126](#)
 - overview [8](#)
 - qualification support [127](#)
 - supporting [16, 124](#)
- virtual level in SAP BW [15](#)

X

- XMLA
 - Analysis Services 2005/2008/2012 [24](#)
 - Oracle Essbase [19](#)
 - TM1 [30](#)

