



Analytics and Mobility

Installation and Porting Guide for the Analytics Module



Version 2020

MicroStrategy 2020

October 2020

Copyright © 2020 by MicroStrategy Incorporated. All rights reserved.

Trademark Information

The following are either trademarks or registered trademarks of MicroStrategy Incorporated or its affiliates in the United States and certain other countries:

MicroStrategy, MicroStrategy 2020, MicroStrategy 2019, MicroStrategy 11, MicroStrategy 10, MicroStrategy 10 Secure Enterprise, MicroStrategy 9, MicroStrategy 9s, MicroStrategy Analytics, MicroStrategy Analytics Platform, MicroStrategy Desktop, MicroStrategy Library, MicroStrategy Operations Manager, MicroStrategy Analytics Enterprise, MicroStrategy Evaluation Edition, MicroStrategy Secure Enterprise, MicroStrategy Web, MicroStrategy Mobile, MicroStrategy Server, MicroStrategy Parallel Relational In-Memory Engine (MicroStrategy PRIME), MicroStrategy MultiSource, MicroStrategy OLAP Services, MicroStrategy Intelligence Server, MicroStrategy Distribution Services, MicroStrategy Report Services, MicroStrategy Transaction Services, MicroStrategy Visual Insight, MicroStrategy Web Reporter, MicroStrategy Web Analyst, MicroStrategy Office, MicroStrategy Data Mining Services, MicroStrategy Geospatial Services, MicroStrategy Narrowcast Server, 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 Tech Support App, MicroStrategy Mobile App Platform, MicroStrategy Cloud, MicroStrategy R Integration, Dossier, Usher, MicroStrategy Usher, Usher Badge, Usher Security, Usher Security Server, Usher Mobile, Usher Analytics, Usher Network Manager, Usher Professional, MicroStrategy Identity, MicroStrategy Badge, MicroStrategy Identity Server, MicroStrategy Identity Analytics, MicroStrategy Identity Manager, MicroStrategy Communicator, MicroStrategy Services, MicroStrategy Professional Services, MicroStrategy Consulting, MicroStrategy Customer Services, MicroStrategy Education, MicroStrategy University, MicroStrategy Managed Services, BI QuickStrike, Mobile QuickStrike, Transaction Services QuickStrike Perennial Education Pass, MicroStrategy Web Based Training (WBT), MicroStrategy World, Best in Business Intelligence, Pixel Perfect, Global Delivery Center, Direct Connect, Enterprise Grade Security For Every Business, Build Your Own Business Apps, Code-Free, Intelligent Enterprise, HyperIntelligence, HyperCard, HyperVoice, HyperVision, HyperMobile, HyperScreen, Zero-Click Intelligence, Enterprise Semantic Graph, Information Like Water, 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.

CONTENTS

Overview and Additional Resources	5
About the Analytics Module	6
About this book	7
Resources	8
Feedback	19
1. Introduction to MicroStrategy Analytics Module	21
Documentation	22
MicroStrategy Analytics Module	22
2. Installation	29
Installation prerequisites	30
Installation procedures	30
Uninstalling a MicroStrategy component	39
3. Configuration	41
Configuring your software	42
4. Using the Analytics Module	50
Using the Analytics Module as an initial design	52
Customize and extend the Analytics Module	53
Roll out to production	56
5. Porting the Analytics Module	59
Introduction to portability	61
Portability methodology	62

General porting prerequisites	62
Port the Analysis Module	64
Perform a gap analysis	65
Map the module	80
Customize and extend the module	90
6. Creating Portable Analytical Applications	91
Architecture of portable analytical applications	92
Best practices for building portable applications	94
Portability rules and recommendations	119

OVERVIEW AND ADDITIONAL RESOURCES

This guide is a technical reference for the Human Resources Analytics Module that comes with MicroStrategy Architect. This guide provides installation and configuration steps for setting up the Analytics Module. It also provides a detailed explanation of the MicroStrategy porting methodology to let you implement the Analytics Module with an existing data warehouse; customization and extension details; and procedures to create your own analytical applications based on the best practices represented by the Analytics Module components:

Chapter 1, Introduction to MicroStrategy Analytics Module presents an introduction to the Analytics Module, and provides an analysis of analytical applications as well as a detailed description of the MicroStrategy Analytics Module components.

Chapter 2, Installation presents steps to install and uninstall the Analytics Module.

Chapter 3, Configuration presents steps to configure your Analytics Module software.

Chapter 4, Using the Analytics Module presents methods to use the Analytics Module as your initial data warehouse when you do not have an existing data warehouse in the analysis area. It also provides steps and scenarios to customize and extend your Analytics Module implementation.

Chapter 5, Porting the Analytics Module presents an introduction to portability, a description of the MicroStrategy portability methodology, and detailed procedures to port the Analysis Module to an existing data warehouse.

Chapter 6, Creating Portable Analytical Applications presents a description of the benefits of building portable analytical applications, provides a

discussion of the architecture and best practices for designing and developing analytical applications, and provides steps for using the Analytics Module as a template to build an analytical application.

Consult the [Human Resources Analysis Module Reference](#) for detailed descriptions of the module's analysis area, scorecards and reports, logical data model, physical schema, and data dictionary.

About the Analytics Module

MicroStrategy helps you build analytical applications by offering a rapid application development framework consisting of analytic starter kits, development products, and methodologies. The Human Resources Analytics Module is built to be portable. You can choose to deploy the analytical application against your existing data warehouse, use the packaged schema as the basis of a new data warehouse, or use the module as a template to build analytical applications.

The components are:

Analytics Module

Prepackaged metadata: Best practices reports, scorecards and dashboards, key performance indicators, attributes, business metrics, filters, and custom groups

Default physical and logical data model: Analytics that are designed to work with your physical schemas and data model or with the module's packaged data warehouse schema

Reference guide: Documentation for the analysis module's data model, the analysis areas, metadata object definitions, data dictionary, and individual report use scenarios

Implementation methodology

Documentation that guides you step-by-step through implementing analytic module against existing data warehouses (known as porting)

Design rules and tenets for designing and developing portable analytical applications

MicroStrategy Architect: A development tool that allows you to map the Analytics Module to an existing data warehouse


The *Advanced Reporting Guide* and other documentation for MicroStrategy Developer and MicroStrategy Architect is available on your MicroStrategy disk (see [Resources, page 8](#)). See the Analysis Module's reference guide for a description of the module's analysis area, packaged reports, logical and physical models, and data dictionary.

About this book

This book is divided into chapters that begin with a brief overview of the chapter's content.

The following sections 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

For examples of report features and a basic introduction to the MicroStrategy business intelligence system, use the MicroStrategy Tutorial, which is MicroStrategy's sample warehouse, metadata, and project. Information about the MicroStrategy Tutorial can be found in the [Basic Reporting Guide](#).

For extensive examples of metrics, filters, and other report objects, see the [Advanced Reporting Guide](#).

Prerequisites

How you use this document depends on the type of user you are and your goals for working with the Analytics Module.

If you intend to evaluate the business value of the module, you should have:

Experience with MicroStrategy reports and metrics using MicroStrategy technology

If you intend to implement and customize the module, you should have:

Experience with logical data modeling and creating business intelligence applications using MicroStrategy technology

A basic understanding of RDBMS concepts and data modeling

Who should use this guide

This document is designed for:

Advanced users and administrators evaluating the business value of the Analytics Module

Consultants and developers implementing and customizing the Analytics Module

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 16](#).

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 16

To access documentation resources on Windows, page 16

To access documentation resources on UNIX and Linux , page 17


To access documentation resources from any location

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

To access documentation resources on Windows

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.

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

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`.

From the MicroStrategy installation directory, open the `Help` folder.


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.

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

Type	Indicates
	<p>actions or part of a list of such GUI elements and their definitions</p> <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 <code>copy c:\filename d:\foldername\filename</code></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>
+	<p>A keyboard command that calls for the use of more than one key (for example, SHIFT+F1).</p>
	<p>A note icon indicates helpful information for specific situations.</p>
	<p>A warning icon alerts you to important information such as potential security risks; these should be read before continuing.</p>

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:

Consult the product guides, Help, and readme files. Locations to access each are described above.

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.

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.

INTRODUCTION TO MICROSTRATEGY ANALYTICS MODULE

The Human Resources Analytics Module is an analytical starter kit designed to solve business problems. The module ships with a sample data model and numerous reports and analytics. The module is designed with a portability paradigm, which means it can work against existing data warehouses and allow businesses to make use of data investments without the need for extensive data extraction and loading concerns.

MicroStrategy helps you build analytical applications by offering a rapid application development framework consisting of an analytic starter kit, development products, and design and development methodologies. The development framework includes the analytics module that is built to be portable. You can choose to:

- Deploy the analytical applications against your existing data warehouse
- Use the packaged physical schema as the basis of your data warehouse
- Use the module as a template to build analytical applications

Documentation

Documentation includes a reference guide for the Analysis Module, and this *Installation and Porting Guide*. Documentation for Developer and Architect is available to help you use these development tools.

MicroStrategy Analytics Module

The MicroStrategy Analytics Module is a set of packaged analytical components built using the MicroStrategy platform. The module can be mapped to a different data warehouse or used as a starter kit to develop custom applications.

The Analysis Module consists of a MicroStrategy project in a metadata repository and comes with a default physical data model and documentation covering that module's reports and metadata. The module contains approximately 50-60 reports, a set of key performance indicators (KPIs) and

business metrics, and sample scorecards. You can use these reports and KPIs as building blocks to create and deploy additional reports and performance measurements. MicroStrategy provides report-building wizards that anyone can use to design and build reports. Developers have access to over 150 functions (arithmetic, aggregate, statistical, financial, mathematical, and OLAP) that they can use to build new KPIs and business metrics.

For example, a developer can customize the areas of analysis and navigation paths for end users, and create workflow wizards for end users to build their own reports and analyze information.

You can take advantage of the Analytics Module in different ways:

If you have an existing data warehouse containing data pertinent to the Analysis Module's analytical area, you can "map" the Analysis Module and its reports to work with your data warehouse's existing data structures.

If you do not have a data warehouse containing data pertinent to the Analysis Module's analytical area, you can use the physical schema that comes with the Analytics Module, as well as the logical data model and reports, to serve as a starter design for a new data warehouse.

You can use the components of the Analytics Module as best practices examples. You can use the packaged components, documentation, and best practices scenarios and examples as templates to build analytical applications.

The Analytics Module:

Provide best practices analysis and MicroStrategy expertise in the area of human resources analysis

Can be integrated into an existing data warehouse, taking advantage of MicroStrategy platform flexibility, schema support, and database independence while protecting your data warehouse investments

Use an easily extendable and modifiable architecture

Are developed with MicroStrategy standard tools so an additional application framework is not required; standard MicroStrategy tools are used to implement and extend the application

Is modular by nature, allowing you to implement only parts of the module, while providing a seamless experience for users

The MicroStrategy Analytics Module includes metadata content and documentation, as described below.

Metadata content is the definition of the logical data model and reports in the MicroStrategy metadata repository.

Documentation includes:

Business content, such as business concept (attribute) definitions and report usage scenarios located in the Analysis Module's reference guide

Technical content, such as the chapters in this guide that describe the portability methodology and explain how to implement the module

Analytics Module documentation

The documentation is an essential component of the modular analytical application, providing both business and technical content.

Documentation	Description
Reference Guide	<p>The Analytics Module has a reference guide, containing all the definitions for the module. The reference guide includes:</p> <ul style="list-style-type: none">• An introduction to the module's analysis area and reporting challenges• Business value information and a complete definition, with a report screen shot, for each packaged report• A complete list of all metrics and their descriptions• A logical data model definition with all hierarchies and attributes, including metadata definitions

Documentation	Description
	<ul style="list-style-type: none"> The default physical schema provided with the module and the data dictionary (tables and columns)
Installation and Porting Guide	<p>This guide contains</p> <ul style="list-style-type: none"> Installation information specific to the Analytics Module Configuration steps for the Analytics Module Information for customizing and extending the Analytics Module to suit your business requirements The MicroStrategy porting methodology, along with detailed steps and examples for porting the Analysis Module to your existing data warehouse Best practices for using the Analytics Module as a template for designing and building analytical applications

An analysis of portable analytical applications

Prepackaged analytical applications like the MicroStrategy Analytics Module provide two major benefits:

They deliver best practices reports for the domain area.

They allow reduced time for implementation.

Drawbacks of traditional analytical applications

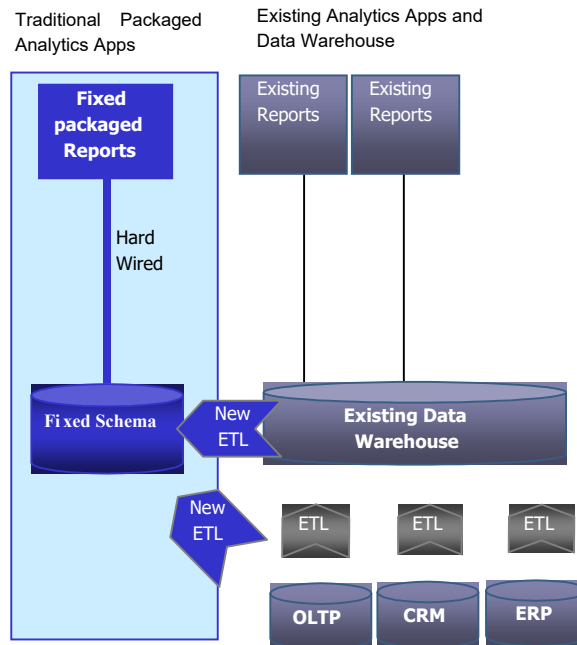
Unfortunately, many prepackaged analytical applications also come with costs, like the following:

A fixed physical schema and reports force you to model your business around packaged components instead of around your own business requirements.

Fixed metadata components are difficult to customize and extend to support your business requirements.

Data must be extracted to a new database schema, requiring an extraction tool and very intensive services.

They have a high cost entry point. Additional costs include a required ETL tool, end-user licenses, and implementation services.



Advantages of MicroStrategy analytical applications

The MicroStrategy Analytics Module lets you benefit from the advantages of using a prepackaged analytical application and provides additional benefits without the costs inherent in traditional packaged analytical applications.

MicroStrategy advantages include:

Best practices reports for the Human Resources domain area, developed with over a decade of MicroStrategy business intelligence implementation experience and best practices methodologies

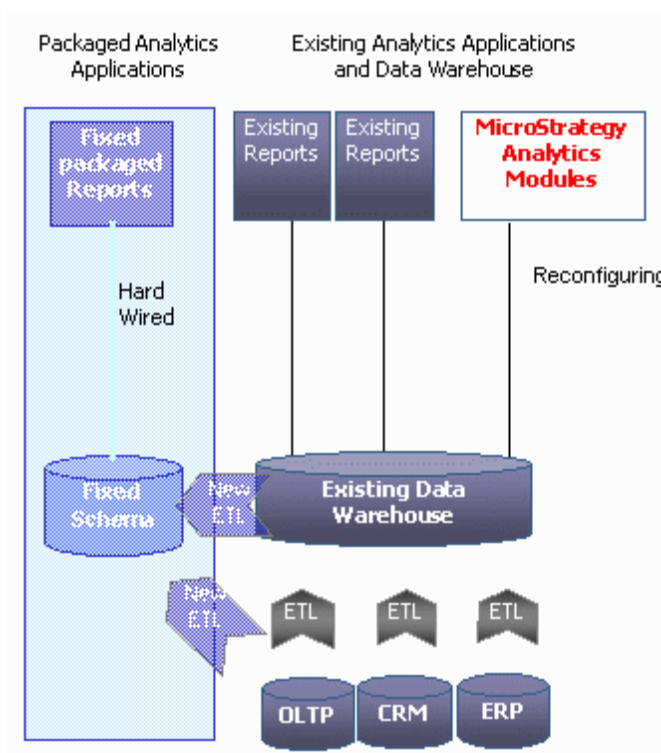
Integration into your existing data warehouse, which is critical to make use of existing data warehouse investments

Rapid deployment, providing packaged components without additional, extensive data extraction

No fixed components, making it easy to extend and customize the analytical applications to address your business and analysis requirements

Modularity, which allows you the flexibility to implement some analysis areas within the Analysis Module

A low cost entry point; the Analytics Module plus standard project development tools have a low per-developer price, while end users only require standard licenses



When using an existing data warehouse, the value obtained from the Analytics Module depends on existing data structures and data elements. If your data warehouse has non-optimal schemas or missing key data elements, it can limit the value obtained from the MicroStrategy Analytics Module.

Analytics Module as best practices examples

The MicroStrategy Analytics Module reflects best practices for analytical applications. Because of this, you can use the Analytics Module and its components as templates on every level.

Examples of best practices templates include

Best practices reports: Use the documentation and metadata definitions to reproduce your favorite Analysis Module reports in an existing MicroStrategy project. You can use the Analytics Module reports as templates, reflecting best practices examples.

Best practices schema design: Use portions of the physical schema design to enhance a data warehouse that has no existing data in the analysis area.

Best practices documentation: Use the documentation as a template to document the processes and tasks required to define a packaged analytical application.

Best practices analytical applications: Use the Analytics Module as a template to design and develop your own portable analytical applications.

INSTALLATION

This chapter describes the prerequisites and procedures for installing the MicroStrategy Analytics Module and its related packaged components.

MicroStrategy Developer and MicroStrategy Architect are the additional features that work with the Analytics Module. Developer and Architect are necessary to make use of the Analytics Module. If you have already installed



Developer and Architect, you can proceed to install the Analytics Module.

However, if you have not yet installed Developer and Architect, see the [Installation and Configuration Guide](#) to install them either before or at the same time as you install the Analytics Module.

Installation prerequisites

Before installing the Analytics Module, review the requirements below.

System software requirements

If you are installing the Analytics Module at a different time than any other MicroStrategy products, make sure MicroStrategy Developer and MicroStrategy Architect have been installed before you install the Analytics Module.

If you have already installed Developer and Architect, there are no additional system software requirements to install the Analytics Module outside of those required for your other MicroStrategy products. See the *MicroStrategy Installation and Configuration Guide* for a complete list of recommended and required hardware, software, and other installation prerequisites before you install any MicroStrategy product.

Installation procedures

For information about installing the Analytics Module in a manner different than that listed in this section, see the [Installation and Configuration Guide](#) for advanced installation functionality.

You can also install the Analytics Module by following the standard installation procedure for the suite of MicroStrategy products in the [Installation and Configuration Guide](#). Select the MicroStrategy Analytics Module along with MicroStrategy Developer, and MicroStrategy Architect during the installation.

The Setup Wizard

The MicroStrategy Setup Wizard provides step-by-step instructions to guide you through the installation process.

To access the Setup Wizard

Log on to the machine where you are installing the Analytics Module. You must log on using a domain account with Windows administrative privileges for the domain or target machine. The domain must include your database servers.

Exit all Windows applications before beginning the installation process.

Run the setup program to begin the installation:

If you are installing the Analytics Module from a CD, insert the CD into the CD-ROM drive and wait a few moments for the MicroStrategy Main Menu window to automatically display; or locate and run the `Setup.exe` file on the CD.


If you are installing the Analytics Module from a set of downloaded files, locate and run the `Setup.exe` file.

Click **Install Software**.

Click **Install MicroStrategy Platform**.

If this is the first time you are running this installation, you are prompted to choose the language for the wizard. Select the appropriate language from the drop-down list and click **OK**.

The Setup Wizard opens and walks you through the rest of the installation process. The following sections of this guide describe the actions you need to take for each page in the wizard.

 Note the following:

The installation screens you see depend on what products you choose to install. These instructions describe only those screens necessary to install the Analytics Module.

At any time during the setup, click **Cancel** to quit the installation.


Welcome

Page Content	Options
Welcome statement	Click Next to proceed.

If any Windows services are running for previously installed MicroStrategy products, you are prompted to stop them. Click **Yes** to proceed. If you click **No**, you cannot install any MicroStrategy products until you stop all MicroStrategy services. If you opened the Setup Wizard through the Microsoft Control Panel Add/Remove Programs option, the wizard opens the Welcome page in maintenance mode.

Page Content	Options
Welcome statement	<ul style="list-style-type: none"> Click Modify to add new program components or to remove currently installed components. The remaining pages are the same as for a first-time installation. Click Repair to reinstall program components if you have problems with

Page Content	Options
	<p>previously installed components. Your program components are returned to their original installation state.</p> <ul style="list-style-type: none"> • Click Remove to quickly remove all installed MicroStrategy components without going through all the Setup Wizard pages. • Click Next to proceed.

 If you installed using a CD, you need your original installation CD to repair the installation by reinstalling.

License Agreement

Page Content	Options
License-related terms and conditions	<ul style="list-style-type: none"> • Read the license agreement. • Select to accept or to decline the agreement. If you choose to decline, you cannot install MicroStrategy products. • Click Next to proceed. • Click Back to return to the previous page.

Customer Information

Page Content	Options
Boxes for name,	<ul style="list-style-type: none"> • Enter your name.

Page Content	Options
company name, and product license key	<ul style="list-style-type: none"> • Enter the name of your company. • Enter the license key. • Click Next to proceed. • Click Back to return to the previous page.

Setup Type

Page Content	Options
Options to select either a typical or advanced setup	<ul style="list-style-type: none"> • Select Typical to place the Analytics Module in the root directory. • Select Advanced to specify a different directory for each MicroStrategy product you install. • Click Next to proceed. • Click Back to return to the previous page.

There is one significant difference between a typical setup and an advanced one. With the advanced setup option, you can select a different location for each product selected in the Select Components window; with the typical option, the Analytics Module is placed by default in `C:\Program Files (x86)\MicroStrategy\Analytics Modules\Analytics_Metadata.mdb`.

Choose Destination Location

Page Content	Options
<p>Location where the MicroStrategy products will be installed</p>	<ul style="list-style-type: none"> • Click Browse to select a location different from the default value (see the following Notes). • Click Next to proceed. • Click Back to return to the previous page.

An MS Access file called `Analytics_Metadata.mdb` will be installed. It contains the production version of the Human Resources Analysis Module (HRAM) project, as well as tutorial projects.

By default the `Analytics_Metadata.mdb` file is installed in the following location: `C:\Program Files (x86)\MicroStrategy\Analytics Modules\Analytics_Metadata.mdb`.

 Note the following:

While this setting determines the default root directory for the installed Analytics Module, you can change the destination of a product later if you chose an advanced setup.

With both typical and advanced setup types, you can choose the directory for a product only if that product is not already installed on the server machine. Otherwise, the product can only be installed in the same directory in which it already exists and you will not see this page.

Select Components

Page Content	Options
<ul style="list-style-type: none"> • A list of MicroStrategy products • Space Required: Space needed for the MicroStrategy products selected; the count changes dynamically as check boxes are selected and cleared • Description: Details about each MicroStrategy product 	<ul style="list-style-type: none"> • Select or clear the appropriate check boxes. • Click Next to proceed. • Click Back to return to the previous page.

If you have not uninstalled previous versions of MicroStrategy products, you are prompted to overwrite them. If you have an old metadata repository and warehouse for MicroStrategy Tutorial for example, perhaps from an evaluation, and you want to keep them, you must rename them or move them to another location; otherwise, they will be overwritten. If you are prompted to overwrite existing files, click **Yes** to ensure that all products and product tutorials will work properly.

Select Program Folder

Page Content	Options
<ul style="list-style-type: none"> • Box to specify the name of the program folder in your Windows Start menu from which MicroStrategy products will be accessed 	<ul style="list-style-type: none"> • If you want, type a folder name different from the default or select an existing folder; otherwise leave as is (recommended).

Page Content	Options
<ul style="list-style-type: none"> List of the existing program folders found under the Windows Start menu 	<ul style="list-style-type: none"> Click Next to proceed. Click Back to return to the previous page.

Start Copying Files


Page content	Options
<p>Current Settings:</p> <ul style="list-style-type: none"> Products that will be installed or updated Locations in which the products will be installed (target directories) Name of the Windows Start menu program folder Virtual directories for Web and Subscription Portal Service account for MicroStrategy Narrowcast Server Location of the installation log file License details 	<ul style="list-style-type: none"> Click Next to proceed. Click Back to return to the previous page.

Click **Next** and the installation process begins, which can take several minutes depending on your computer's hardware configuration.

When the installation process has finished, you are prompted to either view the ReadMe file (click **Yes** or **No**) or go directly to the InstallShield Wizard Complete page.

InstallShield Wizard Complete

Page Content	Options
<ul style="list-style-type: none"> • Message confirming installation completion • Options (yes/no) to restart the machine (if necessary) • Check box to open the ReadMe file (if restarting is not required) • Instructions to empty drives and click Finish 	<ul style="list-style-type: none"> • Click Yes to restart the machine. • Click No to continue without restarting. • Select the check box to open the ReadMe file (if restarting is not required). • Click Finish to complete the setup.

 If the option to restart your machine appears, to ensure that the installation process finishes correctly you should select **Yes I want to restart my computer now**.

After installation is complete, you are ready to configure the MicroStrategy components you have installed. This will ensure that the software can be used immediately. To configure your MicroStrategy software with the Configuration Wizard, see [Configuration, page 41](#).

Installation verification

During the installation process, the Setup Wizard gathers and records information about your system and your installation selections. You can verify installation setup information through the installation log file (`install.log`), located by default in `C:\Program Files (x86)\Common Files\MicroStrategy`.

The installation log file includes the following information:

Installation date

Target directories

Program folder name


Operating system identification

Hardware specifications

Selected installation options

Registry paths

List of registered files

 The installation log file can be particularly helpful if you encounter errors during the installation process. For example, the log can tell you if a registry key or path was not added or if a critical file was not registered successfully.

Uninstalling a MicroStrategy component

You might choose to uninstall one or more MicroStrategy components, perhaps to install those components on a different machine. The uninstall function:

Unregisters and removes selected files, registry entries, and shortcuts logged in the `Uninst.isu` log file

Calls a custom DLL to handle unlogged items such as registry entries and files

Before uninstallation begins, the DLL file

Checks for user privileges. If they are not valid, uninstallation stops.

Checks for running components. If one is found, uninstallation stops.

Stops and deletes the MicroStrategy Intelligence Server service (only when uninstalling Intelligence Server).

Deletes application-created files such as `*.log`, `*.gid`, `*.ldb`, and `*.tb`.

To uninstall from the Control Panel


Close all installed MicroStrategy products.

From the Windows **Start** menu, point to **Settings** and select **Control Panel**.

In the Control Panel, double-click the **Add/Remove Programs** icon. The Add/Remove Programs dialog box opens.

Select **MicroStrategy** and click **Change/Remove** (or **Add/Remove** in Windows NT). The MicroStrategy Setup/Maintenance program opens.

Select **Modify** and click **Next**.

 If you want to remove all MicroStrategy components, select **Remove**. Click **Yes** for any prompts that appear, then click **Finish** when maintenance is complete to close the maintenance program.

Select to accept the license agreement and click **Next**.

Verify your customer information and click **Next**.

Verify your setup type and click **Next**.

Currently installed components appear with a check mark. Clear the check boxes for the components to uninstall and click **Next**.

If you are prompted about stopping your Web server, click **Yes** to stop it and continue with the uninstallation.

Verify the settings and click **Next** to begin removing files.

When the uninstall routine is complete, select **Yes** to restart your computer or **No** to restart it later. Then click **Finish** to close the maintenance program.

 You should restart the computer to achieve a clean uninstall.

CONFIGURATION

Once you have installed your MicroStrategy Analytics Module software, you complete a few configuration tasks before you can begin using it with your own data. This chapter provides steps to configure the installed Analytics Module project.

Configuring your software

Follow the sections below to prepare your system. Then perform the procedures that follow to configure your system to use the Analytics Module.

Configuration preparation

Make the following decisions about product setup and use.

Project sources

The Analysis Module consists of a MicroStrategy project in a metadata repository. A project source contains the information necessary for MicroStrategy to connect to the metadata in which your projects are stored. The project source stores the location of the metadata repository or the MicroStrategy Intelligence Server that is used to run the project. There are two types of project sources:

Direct project sources connect directly to the metadata through ODBC. This is known as a direct (or two-tier) connection.

Server project sources connect to the metadata via a MicroStrategy Intelligence Server. This is known as a server (or three-tier) connection.


The Configuration Wizard guides you through the process of creating both types of project sources. You will use the Configuration Wizard during the configuration procedures described below.

Connection types

You can configure the Analytics Module project to run with a direct connection or a server connection.


A direct connection connects the project source to the metadata repository directly through ODBC.

A server connection connects the project source to the metadata repository through MicroStrategy Intelligence Server. A server connection is the most commonly used connection type. See the [Installation and Configuration Guide](#) for more detailed information on connection types.

 To use the Analytics Module documents (for example, for dashboards and scorecards), you must be connected through a server connection using MicroStrategy Intelligence Server.

With or without a data warehouse

The Analytics Module can be used with or without an existing data warehouse. If you have an existing data warehouse in the analytical area, you can take advantage of best practices reports and metrics, while making the best use of your existing data warehouse investments. If you have no data warehouse storing data for the analysis area, you can use the Analytics Module's default physical schema and packaged metadata components as an initial design for your own data warehouse and analytical application.

 If you do not have an existing data warehouse in the analytical area, follow the procedures below to create a data warehouse. For the new warehouse, you must create the required tables using the scripts generated from the Erwin files that come with the MicroStrategy Analytics Module.

The configuration procedures below include steps important for both uses. These include:

Configuring a production metadata repository

Duplicating the Analytics Module project into the production metadata repository

Configuring the MicroStrategy project to point to the target data warehouse


Configuration prerequisites

Before you begin using the Configuration Wizard you should satisfy the following requirements:

Install the necessary MicroStrategy products. At a minimum, you should have MicroStrategy Architect and MicroStrategy Developer (Architect is a subcomponent of Developer) and the MicroStrategy Analytics Module installed. For steps to install these products, see [Chapter 2, Installation](#).

During the Analytics Module configuration procedures below, you will create a project source, when you can set up the Analytics Module project to run through either a direct connection or a server connection. To run the Analytics Module project through a server connection, you must have Intelligence Server installed. See the *Installation and Configuration Guide* for details to install and configure Intelligence Server.

Be sure you have access to an empty database location certified to house the metadata repository. For a list of certified metadata platforms, see the MicroStrategy `Readme` file. From the Windows **Start** menu, point to **Programs**, then **MicroStrategy Documentation**, and then select **Readme**.

 MicroStrategy products must be configured on the machine on which they are installed. You cannot configure remotely.

Configure the production metadata repository

In this procedure, you create a DSN for the metadata repository that comes with the Analytics Module, connect it to a project source, and then create a space to serve as your production metadata repository.

To configure the production metadata repository


On a machine with a MicroStrategy installation, create a data source name (DSN) using an Access ODBC driver that points to the `Analytics_Metadata.mdb` file. Name the DSN "Analytics_Metadata".

Create a direct connection (a 2-tier project source) pointing to the `Analytics_Metadata` DSN you created in the step above. You can create a project source in the following ways:

If you need guidance, use the MicroStrategy Configuration Wizard. See the *MicroStrategy Installation and Configuration Guide* for steps to open the wizard and create a direct (2 tier) project source.


If you are familiar with creating project sources, use MicroStrategy Developer. From Developer's **Tools** menu, select **Project Source Manager**.

Connect to this metadata repository by logging in as **Administrator** with no password. The metadata repository should display the project for the Analytics Module, plus projects for Tutorial.

 Reports in these projects do not run because the data warehouses are not yet set up. However, you can browse through the metadata objects.

In your database of choice, create a space for storing the project metadata. See the *MicroStrategy Readme* for supported databases.

Create a DSN that points to the new space created in the step above.

 While it is possible to use a Microsoft Access database for the metadata repository, it is not a suitable metadata repository for a production project. Use Access only for a proof-of-concept type of application.

Using the MicroStrategy Configuration Wizard, create a metadata shell for this new space by running appropriate table and trigger creation scripts. See the *MicroStrategy Installation and Configuration Guide* for steps to open and use the Configuration Wizard.

Once the scripts are run, an empty metadata shell is created in this location. This metadata repository is your production metadata repository and will be called the production metadata through the rest of the configuration procedures.

Using either the Configuration Wizard or MicroStrategy Developer, create a direct connection (a 2-tier project source) pointing to the DSN created in step 5 above. You can name this new direct connection anything you want, but it is called PROD_MD (or production metadata) throughout the rest of this chapter.

Connect to the production metadata by logging in as **Administrator** with no password. There will be no projects in this shell.

Duplicate the Analytics Module

The following procedure describes how to duplicate the Analytics Module into the production metadata created in the procedure above.

To duplicate the Analytics Module

In MicroStrategy Developer, from the **Schema** menu, select **Duplicate Project**. Follow each step in the Project Duplication Wizard:

Step 1 - Select source project location: Use `Analytics_Metadata` as the source project location.

Step 2 - Select project to duplicate: Select the Analysis Module.

Step 3 - Select a location for the duplicated project: Use `PROD_MD` as the target location.

Steps 4 - 9: Accept the default settings.

Depending on the configuration, the project duplication process can take several minutes. When the process is complete, examine the logs and verify that the duplication ran error-free.

Configure the data warehouse connection

Use this procedure to configure the production metadata to point to the target data warehouse.

To configure the data warehouse connection

Create a DSN that points to the target data warehouse.

If you intend to use an existing data warehouse with the Analytics Module, the target data warehouse will be your existing data warehouse.

If you do not have an existing data warehouse in the analytical area, the target data warehouse will be a data warehouse created using the physical schema that comes with the Analytics Module. This data warehouse must have the required tables created using the scripts generated from the Erwin file that comes with the MicroStrategy Analytics Module.

Open the Database Instance Manager. (From MicroStrategy Developer, select **Analytics Modules**, select **Administration**, and then select **Database Instance Manager**.)

Edit the database instance for the duplicated project, and point it to the new data warehouse DSN. Select an appropriate database connection type. Changes to the database instance automatically update the project level VLDB settings. Any additional changes to VLDB settings can be made in the project's Project Configuration menu.

Depending on the type of data warehouse to which you port the module, you may need to change certain settings in your system so the module recognizes the new data warehouse. See [Database settings, page 48](#) to locate your database type and required project and other object settings.

After making appropriate changes to the database instance, disconnect and reconnect to the project source so your changes can take effect.

Database settings


Settings for the following certified database types are provided below:

Oracle 8i2 and later versions

SQL Server 2000

IBM DB2 UDB 5.2 and later versions

Teradata v2R4 and later versions

 See the *Supplemental Reference for System Administration* for a complete list of default VLDB settings and an explanation of Join Type settings. See the online help for steps to change join types.

Oracle databases

For porting to an Oracle 8i2 or later database:

Change the project level join type to Join 89.

Change the join type to Join 89 for the Current Headcount and Average Tenure Summary report, located by default in the Human Resources Analysis Module project under

```
Public Objects\Reports\Scorecards\HR Summary Scorecard  
Base Reports\Current Headcount and Average Tenure Summa  
ry.
```

SQL Server database

For porting to a SQL Server 2000 database, to calculate dates: For the Date metrics located by default in the Human Resources Analysis Module project under `Public Objects\Metrics\Dates`, you may need to modify the default metric definition for the metric you want to use, to comply with the SQL Server syntax requirements.

For example, for the metric Current Tenure, you must change the definition to:


```
Applysimple ('Avg (abs (datediff (day, CurrentDate (), #0)))',  
[Hire Date]).
```

DB2 databases

For porting to a DB2 UDB 5.2 or later database, change the formula to: ApplyAgg("Max(DAYS(Current Date) - DAYS(#0))", [Hire Date]) {~+} for the Current Tenure metric, located by default in the Human Resources Analysis Module project under

```
Public Objects\Metrics\Dates\Current Tenure.
```

Teradata databases


For porting to a Teradata V2R4 or later database, no changes are required.

USING THE ANALYTICS MODULE

The MicroStrategy Analytics Module is designed to let you take advantage of best practices analysis while making the best use of your existing data warehouse investments by porting an Analysis Module to your target data warehouse. For more details on porting, see [Chapter 5, Porting the Analytics Module](#).

If you have no data warehouse storing data for the analysis area, you can take advantage of the Analytics Module by using the physical schema and packaged metadata components that come with the module as an initial design for your own data warehouse. You can also use the Analytics Module as a template to design and develop analytical applications.

This chapter explains how to use the Analytics Module when you do not have an existing data warehouse in the Human Resources analysis area. It also provides information on customizing and extending the Analytics Module. For information on using the Analytics Module with an existing data warehouse, see [Chapter 5, Porting the Analytics Module](#). For information on designing and creating your own analytical applications using the Analytics Module as a template, see [Chapter 6, Creating Portable Analytical Applications](#).

 Be aware of the following:

MicroStrategy provides the Analytical Module and its related components, including the default physical schema, as an example. If you decide to populate, customize, enhance, or otherwise change the Analytics Module or any individual components, it becomes the property of your company and you or your company are responsible for any additions, modifications, enhancements, or customizations made.

MicroStrategy does not provide ETL routines or tools to populate the physical schema that comes with the Analytics Module; it is simply provided as an example. If you choose to use the physical schema, you must identify data sources and create the ETL routines necessary to populate the database schema.

Using the Analytics Module as an initial design

The Analytics Module provides the key data elements necessary to define the Human Resources analysis area.

If you do not have an existing data warehouse in the human resources analysis areas, you can use the default physical schema and prepackaged metadata components in the Analytics Module as an initial design for your own new data warehouse.

There are two approaches to using the default physical schema that comes with the Analysis Module as an initial design for your own data warehouse:

Initial Data Warehouse: You can take immediate advantage of the default physical schema for the data warehouse that comes with the module, populating it with your own data. Once you have established your own data warehouse using the module, you can expand and customize the application as you identify additional business and analytical requirements.

Initial Customization: Alternatively, you can enhance the default physical schema initially with your own customizations, to reflect immediate business and analytical requirements you may have. Once the default schema has been customized, you can then populate it with your data.

For customization scenarios and steps to customize and extend the Analytics Module, see [Customize and extend the Analytics Module, page 53](#).

Initial data warehouse

You can use the Analysis Module as an initial design for your analysis solutions in the module's analytical area and then add your company's specific business and analysis requirements by customizing and extending the module.

You begin this approach by populating the physical schema that comes with the module. The Analysis Module's reference guide contains the logical and

physical view of the default data warehouse schema, as well as a complete data dictionary with all tables and columns, and how to populate them.

The Analysis Module's metadata components provide best practices analytics that can be used to define reports addressing any additional requirements of your company. As you continue to add data to the default physical schema, you can decide at any point to enhance the module by customizing or extending the application. The Analysis Module provides its physical schema definition in an Erwin format, allowing you to edit definitions and then generate table creation scripts for any database platform.

See [Customize and extend the Analytics Module, page 53](#) for more information on enhancing the default physical schema.

MicroStrategy provides the Analytics Module and its related components, including the default physical schema, as an example. If you decide to populate, customize, enhance, or otherwise change the Analytical Module or any individual components, it becomes the property of your company and you or your company are responsible for any additions, modifications, enhancements, or customizations made.




Customize and extend the Analytics Module

The Analytics Module can be customized and extended to address your specific business needs. Whether you are using the Analytics Module as your initial data warehouse (see above), porting it to an existing data warehouse (see Chapter 5), or using it as a template to create your own analytical applications (see Chapter 6), you can customize and extend the metadata objects (attributes and facts) to address additional business and reporting requirements.

When you customize or extend the Analytics Module, you are customizing existing data elements to reflect your company's requirements or adding

data elements that are specific to your company. For example, you might want to convert some prompted reports to static selections for end users.

Customization and extension are accomplished using the standard MicroStrategy design and development tools, MicroStrategy Architect and MicroStrategy Developer. Both of these products come with the Analytics Module. For more information, see [About the Analytics Module, page 6](#).

 MicroStrategy does not provide ETL routines or tools to populate the physical schema provided with the Analytics Module; it is provided as an example. If you choose to use it, you must identify data sources and create the ETL routines required to populate the database's physical schema.

Customization scenarios

There are two general types of scenarios when you customize or extend an Analysis Module, depending on what elements you want to change. You can make changes or additions to the application objects (such as reports, metrics, and filters, located in the Public Objects folder). Or you can make changes or additions to the schema objects (such as attributes and facts, located in the Schema Objects folder).

Knowing the different efforts required for each of these approaches can help you decide whether and how to make a modification. The following list presents examples of typical changes based on the two general scenarios and any related updates necessary based on the change.

Application objects

These scenarios detail changes to application objects only. It does not include changes to logical data models.

Examples of changes to the application objects include:

Converting prompts, which enable dynamic selections, of packaged reports to static selections.

Modifying a metrics definition. For example, you might modify Quarter Comparison metrics to compare data at the Month level, using the Previous Month transformation instead of the Previous Quarter.

Updating filter conditions to use values in the target data warehouse. Since packaged filters can include conditions based on values existing in the default database, you may need to do this when you port the Analysis Module to an existing data warehouse.

Modifying report templates to include different attributes or metrics.

Examples of creating new application objects include:

Creating new reports based on existing attributes and metrics. The prepackaged reports can be used as best practices examples to create new reports.

Creating new metrics, such as Percent to all Contributions and Month to Month comparisons, that are not included in the module.

Schema objects

These scenarios include changes to the schema objects. These changes require updates to the physical database schema, and may drive changes to the application objects (reports and metrics) as well.

Examples of changes to the logical data model include:

Changing attribute keys for fact tables.

Modifying a hierarchy to include additional levels. This type of change also requires you to update the physical schema.

Examples of additions to the logical data model include:

Adding additional characteristic attributes to an existing attribute.

Adding facts to existing tables.

Examples of enhancements to the application objects, driven by changes or additions to the schema objects, such as those described above, include:

Adding new Product attributes to existing reports

Creating new metrics and reports to analyze Cost and Profit data

Customization steps

You usually identify necessary additions and modifications for the Analytics Module during a gap analysis, as described in [Port the Analysis Module, page 64](#).

After you complete a gap analysis, you enhance the logical data model to include any additional attributes and facts you require. This may include adding data elements in the data warehouse.

Finally, you modify the module's packaged components or add new ones, such as metrics, prompts, reports, and so on. For details on adding or modifying attributes, facts, metrics, prompts, reports, and so on, see the [Advanced Reporting Guide](#).

The comprehensive documentation and packaged components that come with the Analytics Module can be used as a template for your customization and enhancements. Developers can access and reverse-engineer all packaged components using MicroStrategy developer tools.

Roll out to production

Before the final version of the Analytics Module is made available to end users, you must complete some preparation tasks. Each section listed below describes aspects of your Analytics Module implementation that can cause performance issues for users. Each section also suggests actions to be taken that can lessen or eradicate the potential issue.

You must consider the tasks listed below when identifying resources and determining a schedule for rolling out the Analytics Module to your production environment and making it available to users.

You can find details for many of the MicroStrategy-specific procedures in the *MicroStrategy System Administration Guide*.

Tuning and performance

The Analytics Module does not include any type of tuning or performance optimization. Consider using the following techniques to optimize performance for your users:


aggregate tables (details available in the *MicroStrategy System Administration Guide*)

partitioning (details available in the *MicroStrategy System Administration Guide*)

indexes


MicroStrategy users

Different users and user profiles may be required to allow or restrict access to areas of the Analytics Module. Users and user profiles are defined based on your company's requirements, which include configuring permissions to access MicroStrategy functionality.

 By default, the Analytics Module includes a single user (Administrator) with all privileges.

Security

Your company may require you to limit access to certain objects, and to implement data level security with security filters. See the *MicroStrategy System Administration Guide* for information on implementing these security features.

 By default, the Analytics Module includes a single user (Administrator) with full control for all objects. Additionally, all objects are set up with View access for all users.

Direct and server connections

The Analytics Module is packaged as a direct connection project (also known as a 2-tier project), which means it runs in two architectural layers—the business logic/user interface layer and the database layer. For optimum performance, it should run on a server connection (3-tier), through MicroStrategy Intelligence Server.

The Analysis Module can be easily set up to run on a server connection, through MicroStrategy Intelligence Server, by making the appropriate changes to the project configuration. More details are available in the [System Administration Guide](#), and also in *Configure the production metadata repository, page 44*.

PORTING THE ANALYTICS MODULE

The packaged MicroStrategy Analytics Module application can be used in many ways. The module can be integrated into your existing data warehouse, serve as an initial design for a new data warehouse, and facilitate the deployment of enterprise-class portable analytical applications. For any of these scenarios, you can also customize and extend the Analytics Module to meet changing reporting and business requirements.


Existing data warehouse: If you have a data warehouse already holding data for the analytical area, you can reconfigure the packaged logical data model and analytical reports to work with your existing data warehouse structures. This chapter describes the reconfiguration process, known as porting.

No existing data warehouse: If you do not have a data warehouse containing data for the analysis area, you can use the default logical data model, physical schema, and analytical reports as a starter design for a new data warehouse. For more information, see [Chapter 4, Using the Analytics Module](#).

Building analytical applications: If you want to build analytical applications, you can use the Analytics Module as a best practices example by treating the Analytics Module components as a template. For more information, see [Chapter 6, Creating Portable Analytical Applications](#).

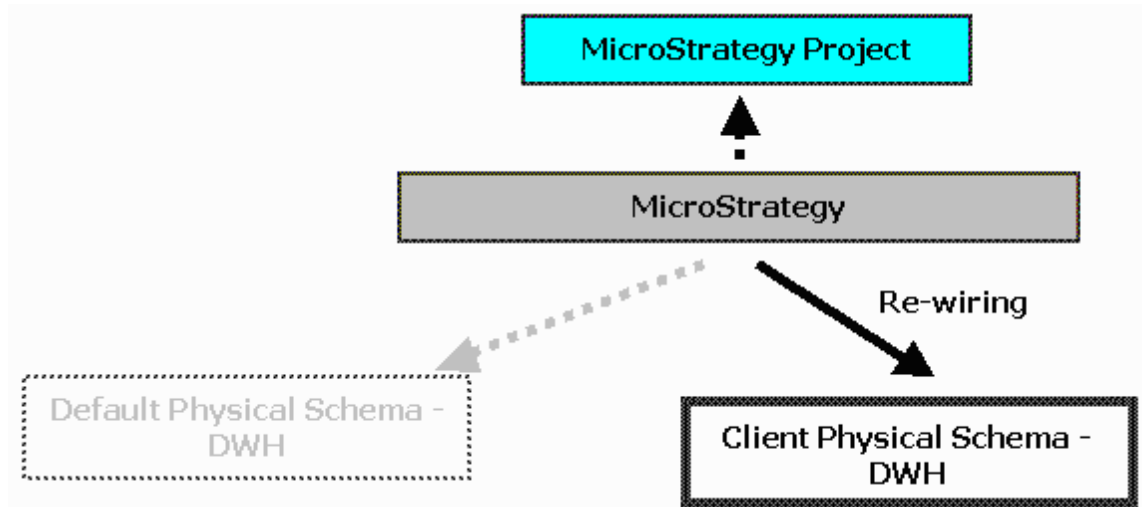
Customize and extend the Analytics Module: You can easily customize and extend the Analytics Module for any of the scenarios above, to meet changing reporting and business requirements. For more information, see [Customize and extend the Analytics Module, page 53](#).

This chapter explains how to port the Analytics Module so it works with an existing data warehouse.

 MicroStrategy does not provide ETL routines or tools to populate data warehouses. You must identify data sources and create the ETL routines necessary to populate your particular database schema.

Introduction to portability

Portability is the ability of an analytical application to be integrated into an existing data warehouse. Porting an analytical application involves mapping the application to an existing physical data warehouse schema while retaining the logical data model definition.



A portable analytical application such as the Analysis Module is a prepackaged project which has been built using portability design rules. These rules are defined to minimize the changes required to map an application's logical data model to a target data warehouse's physical schema. To achieve this, MicroStrategy technology provides an abstraction layer between the logical data model and the physical data warehouse schema, making it possible to map the existing links to a physical schema with a similar data set. By retaining the logical data model definition, all MicroStrategy object definitions such as attributes, facts, metrics, and reports are also retained.

MicroStrategy has developed a detailed methodology to support the mapping of metadata objects from the logical data model to physical database objects in a target data warehouse's physical schema using MicroStrategy tools.

Portability methodology

When you port an `AMInstallPort.AnalysisModule` to an existing data warehouse, you map the facts and attributes in the logical data model to the structure of the physical schema in an existing data warehouse, allowing the analytical reports to reflect the data stored in your data warehouse. Portability centers on the concept of "Metadata Only" analytical applications. By packaging `AMInstallPort.AnalyticsModules` with reports, metrics, facts, and attributes, but no hard-wired data warehouse, MicroStrategy provides you with the tools to perform best practices analysis in the analysis area, based on the data in your existing data warehouse.

General porting prerequisites

This section describes the tools and documentation you need to port the Analytics Module to your data warehouse.

MicroStrategy tools

To port the Analytics Module to your existing data warehouse, you need MicroStrategy Developer and MicroStrategy Architect. Developer and Architect allow you to access metadata objects such as attributes, facts, metrics, and reports.

MicroStrategy Object Manager is an optional tool that can make the porting process more efficient. Object Manager allows you to implement only those portions of the Analytics Module that are relevant to a company's requirements. Object Manager also provides enhanced searching functionality that facilitates the gap analysis. See the [System Administration Guide](#) for information about using Object Manager.

Documentation

In addition to the tools listed above, you must also have all necessary documentation:

Documentation	Description
Analytics Module reference guide	<p>The Analysis Module has its own reference guide which includes:</p> <ul style="list-style-type: none"> • An introduction to the module's analysis area and reporting challenges • A list of packaged reports, and for each report, the business value, report definition, and report screen shots • A complete list of all metrics and descriptions • The logical data model definition, with all hierarchies and attributes, including metadata definitions • The default physical schema provided with the module and a data dictionary
Physical model, in Erwin format	<p>This file contains a definition of the physical schema provided with the module, in an Erwin format:</p> <ul style="list-style-type: none"> • This file can be used to generate database creation scripts for any database platform. • This file can be used as an initial design for your company's data warehouse.
Portability methodology and porting process	<p>This content appears in this Porting chapter, and includes:</p> <ul style="list-style-type: none"> • The Portability methodology section, necessary tools, and overview information for porting a module. • The Port an Analysis Module section with a detailed explanation of how to implement the Analytics Module using an existing data warehouse.
Best practices for portable analytical applications	<p>This content appears in Chapter 6, and includes:</p> <ul style="list-style-type: none"> • A comprehensive explanation of how to design and build applications using the portability paradigm • Best practices used by MicroStrategy to develop the Analytics Module • Development project guidelines (task and plan) • Recommendations for creating MicroStrategy objects

Other prerequisites

Drill maps must be removed from the module before the module can be ported. A drill map is essentially a customized drill path for the attributes and metrics in reports. For some of the reports within the Analytics Module, the drill paths have been customized by assigning a drill map, so that only selected attributes can be drilled on within given reports.

A common example is the quarterly reports that are available in the module. The data in quarterly reports is, logically, only available for the Quarter level. Therefore, users are restricted within these quarterly reports by an associated drill map so that they cannot drill down to Month.

Port the Analysis Module

To port the Analytics Module, you map the module to the physical schema of an existing data warehouse. However, you keep the logical data model and analytics library for the Analytics Module. This allows you to use all the related MicroStrategy objects, such as attributes (analysis-friendly business concepts), hierarchies (relationships between attributes), facts (measures), metrics (business calculations), and reports.

Porting overview

Porting involves the following general steps:

Perform a gap analysis: Identify the overlap between the Analysis Module, your existing data warehouse, and your company's reporting requirements. Once you determine the overlap, you can identify any gaps.

Map the module: Using the overlap information from the step described above, map the Analysis Module to the data warehouse.

Customize and extend the module: Using the gap information from the step described above, customize the module to accommodate additional business and reporting requirements.

Complete the steps below in the order they appear.

Perform a gap analysis

A gap analysis identifies which areas of the Analysis Module's logical data model and reports can be implemented using the structures of your existing data warehouse.

The gap analysis is broken into the following procedures:

Map the target data warehouse to the module's physical schema.

Map the module's logical data model to the target schema.

Identify application objects.

Optionally, identify additional business requirements.

Each procedure is presented below with a description, examples, and many alternative scenarios to support your own gap analysis process.

Gap analysis prerequisites

To complete a successful gap analysis, make sure you have access to the following:

Skills	Good knowledge of your company's target data warehouse schema and your company's reporting requirements, if these exist.
	Experience using MicroStrategy tools and with logical data modeling and physical data modeling.

Tools	Physical schema and data dictionary for the target data warehouse and logical data model, if available.
	The MicroStrategy Analytics Module reference guide, which provides default physical schema, data dictionary, and logical data model.
	Erwin file for the Analysis Module; contains the module's default physical schema and production metadata.
	MicroStrategy Developer and MicroStrategy Architect, to access definitions of attributes, facts, reports, metrics, and so on; and to search for dependencies. For example, if a fact does not exist in the target data warehouse schema, dependencies identify which reports, metrics, and so on will be impacted.
	(Optional) MicroStrategy Object Manager, to search for child-parent and parent-child dependencies. This tool provides a thorough impact analysis.


Map the target data warehouse to the module's physical schema

In this first procedure of the gap analysis, you identify whether each data element from the module's physical schema exists in the target data warehouse. For elements that do not exist, you check whether they can be replaced with data elements that are present in the target data warehouse. The result is a document presenting a complete mapping that focuses on available data elements and any proposed changes needed.

An example follows this procedure, and a list of typical scenarios and actions follows the example.

To map the target schema to the module's schema

Prepare a document to record the information you will identify in this procedure. The document should include space for a list of data elements (tables and columns) and for comments.

 An example document can be found at the end of this procedure; see *(Optional) You have included a list of tables from the target schema*

i *that have no equivalent in the module's schema but will be added to support additional analysis requirements., page 69.*

Compare the target data warehouse's physical schema (target schema) with the module's physical schema (module's schema). Identify the list of tables in the target schema that are relevant to the module's analysis area. Record this information in the document you created above.

In the module's physical schema, for each table, identify whether there is an equivalent table in the target schema. Begin with the lookup tables, then move on to the fact tables. Add this information to your document.

For each matching table identified in the previous step, map the module's schema columns to the corresponding ones in the target schema, and add this information to your document.

In some cases, you can map a data element from the module's schema to a similar but not identical data element in the target schema; the target schema's data element will replace the data element from the module's schema. For example, assume the AGE data element does not exist in the target schema but AGE_RANGE does. You can map the AGE column from the module's schema with the AGE_RANGE column in the target schema.

Identify areas in the target schema where a given hierarchy is stored in a different number of tables than in the module's schema.

For example, the target schema might have several lookup tables for the Time hierarchy:

L_YEAR	L_QUARTER	L_MONTH
year_id	quarter_id	month_id
year_desc	quarter_desc	month_desc
	year_id	quarter_id

But the module's schema might have a single lookup table for the Time hierarchy:

L_TIME
year_id
year_desc
quarter_id
quarter_desc
month_id
month_desc

In the example above, the Time hierarchy in the target schema is stored in more tables than the Time hierarchy in the module's schema.

Identify areas in the target schema where a given hierarchy has a different number of levels than in the module's schema.

For example, referring to the L_TIME table in the step above, the module's schema has three levels in the Time hierarchy: year, quarter, and month. But the target schema might only have two levels in its Time hierarchy (quarter and month), or it might have four (year, quarter, month, and date), and so on.

Identify relevant aggregates and fact tables in the target schema that have information at different levels than that in the module's schema.

Review your document to ensure that:

For all tables and columns in the module's physical schema, you have a list of corresponding tables and columns from the target schema.

You have marked or otherwise indicated any data elements from the module's schema that cannot be used because there are no corresponding data elements in the target schema.

(Optional) You have included a list of tables from the module's schema that do not exist in the target schema but will be added to support packaged reports.

(Optional) You have included a list of tables from the target schema that have no equivalent in the module's schema but will be added to support additional analysis requirements.

Scenarios: Mapping physical schemas

The table below presents some typical scenarios you can encounter while completing the physical schema mapping procedure above.

Object	Scenario	Action
Table	Different name	Map all related schema objects together.
	Not available	Facts and attributes available only in this table should be eliminated after resolving dependencies.
	Missing one or more columns	Update relevant attributes and facts that use the missing column in their expressions.
	Partitioned	Create a partition mapping table and add all partitioned tables to the project.
	Aggregates tables present	Add aggregates tables to project.
	Information distributed in multiple tables	Add tables to the project and map each table to relevant schema objects.

Object	Scenario	Action
Column	Different name	Map all related objects together.
	Not available	Update relevant attributes and facts that use the missing column in their expressions.
	Information distributed in multiple columns <ul style="list-style-type: none"> in the same table in different tables 	<ul style="list-style-type: none"> Update attribute or fact expressions. Create additional attribute forms or add facts.
	Different data types	Update the warehouse catalog to read in new data types.
	Columns with same information have same names.	Use automated mapping for corresponding schema objects and when adding tables to the project.
	Columns with same information have different names	Use manual mapping for corresponding schema objects and when adding tables to the project.


Map the module's logical data model to the target schema

In this second gap analysis procedure, you identify which of the module's logical data model objects can be mapped to the target data warehouse. These objects, also called schema objects, include attributes, facts, hierarchies, and so on. The results are high-level guidelines for mapping the module's logical data model to your target data warehouse schema (target schema).

An example follows this procedure, and a list of typical scenarios and actions follows the example.

To map the logical data model to the target schema

Prepare a document to record the results of your analysis in this procedure. The document will need space for a list of supported schema objects and for guidelines to map those objects to the target schema.

Example documents (one each for attributes, facts, and other schema objects)  can be found at the end of this procedure; see [Schema objects without matching data elements cannot be implemented., page 72](#).

Identify the list of the Analysis Module's logical data model objects that are relevant to the target schema. For attributes, include all tables with relevant information, unless there are good reasons to exclude certain tables.

 Note the following:

You will be working with only a subset of the logical data model objects, based on the fact that missing data elements were identified in the previous procedure.

Refer to the reference guide to find definitions for all objects in the logical data model. Refer to the production metadata for relationships with the module's physical schema. You can also access object definitions and physical structure links for attributes, facts, and so on using MicroStrategy Architect.

Using the document completed in the procedure above ([Map the target data warehouse to the module's physical schema, page 66](#)), determine whether each object is supported by the data elements that exist in the target schema. Start by analyzing attributes, then facts, hierarchies, and so on.

For attributes, identify the following for each form expression:

target schema columns

lookup table

other tables where the data element is present

type of relationship with other attributes

relationship tables in the target schema

For facts, identify the following for each form expression:

target schema fact tables/columns and their entry level attributes

For hierarchies, identify constituent attributes and relationships.

For transformations, identify the following:

attribute

mapping type

member expression

target schema member table

(Optional) Use MicroStrategy Developer or Object Manager to identify dependencies between objects. For example, for an unsupported object, use these tools to identify other objects that depend on it. See the [System Administration Guide](#) for details on using Object Manager.

For each supported object, identify changes required to map it to the target schema.

 Note the following:

In some cases, schema objects might be partially mapped. For example, the target schema might support only one of two attribute forms or a relationship between two attributes might not exist in the target schema.

Schema objects without matching data elements cannot be implemented.

Scenarios: Mapping the logical data model

The table below presents some typical scenarios you can encounter while completing the logical data model mapping procedure above.

Object	Scenario	Action
Attribute	Different name	Update the name. All dependents read the change automatically.
	Not available	Remove after resolving dependencies.
	Additional forms	Add forms if relevant to analysis and update the display tab where appropriate.
	Missing forms: ID or DESC	If no substitution is possible, consider eliminating the attribute. Remove the form and update the display as appropriate.
	Split columns for forms	Create an additional form expression.
	Merged columns for forms	Remove the form expression or modify the form expression.
	Changed relationship with parent or child	Update the parent-child relationship, hierarchies, and dimensional metrics.
	Additional parent or child	Add to relationships and update the hierarchy.
	Missing parent or child	Update attribute relationships appropriately and modify any relevant hierarchies.
Change in relate table	Choose a new relate table for the parent or child attribute.	
Fact	Different name	Update the name. All dependents read the change automatically.
	Not available	Remove after resolving dependencies.
	Different column name	Update the fact.
	Expression derived with multiple columns	Update the fact expression and base formulas or metrics.
	Additional expressions: • Information is at	• Update the fact expression. No changes to

Object	Scenario	Action
	<p>same level</p> <ul style="list-style-type: none"> Information is at different level 	<p>the base formulas or metrics are required.</p> <ul style="list-style-type: none"> Update fact expressions to use the ApplySimple function and update base formulas or metrics to use ApplyAgg functions in the platform. Another option is to create separate metrics and update dependent application objects appropriately.
	Missing expression	If there is only one expression, the fact is no longer supported. In the case of multiple expressions, identify metrics/reports that use the missing expression and eliminate those. Checking on the report SQL may be required.
	Change in column data type	Update the warehouse catalog to read in new data types.
	Fact available in partitioned tables	Add a partition mapping table and update the fact appropriately.
	Fact available in aggregates	Add aggregates to the list of other tables.
	Fact available but at different entry levels	Create additional expressions if appropriate. Update related metrics and base formulas.
	Table for dummy fact has different keys	Update the fact and change the base formula/metric appropriately.
Hierarchy	Different name	Update the name. All dependents read the change automatically.
	Not available	Eliminate the hierarchy after resolving dependencies. Metrics using the hierarchy for dimensionality change in meaning and filters/prompts using the hierarchy are removed.
	Additional attribute	Add an attribute if it is a parent or child of other attributes in the hierarchy. The attribute should

Object	Scenario	Action
		also be added if it has relationships (parent and/or child) and meaning similar to those of other attributes in the hierarchy.
	Missing attribute	Update the hierarchy.
Transformation	Different name	Update the name. All dependents read the change automatically.
	Not available	Eliminate the transformation after resolving dependencies. Any metrics using the transformation are no longer supported.
	Same mapping type not available	Remove the transformation after resolving dependent metrics.
	New name for member expression or table	Update the transformation.

Identify application objects

Using the results from the two procedures above, this third gap analysis procedure helps you identify the Analysis Module's components that can be implemented with your target data warehouse. Components include application objects (located by default in the Public Objects folder) such as reports, metrics, filters, and so on. Results of this procedure include a list of supported components, unsupported components, and those that require changes to be supported.

An example follows this procedure, and a list of typical scenarios and actions follows the example.

To identify application objects

Prepare a document to record the results of your analysis in this procedure. The document will need space for a list of supported application objects,

unsupported objects, and objects requiring changes, as well as space for the actions required.

Using the results from the procedure above (*Map the module's logical data model to the target schema, page 70*), identify the dependencies between logical data model objects and application objects.

 Note the following:

See the Analysis Module's reference guide for a list of reports, metrics, and so on. You can use the production metadata to determine dependencies between different objects.


Also, you can use MicroStrategy Developer or Object Manager to complete this task. Object Manager provides enhanced functionality to identify child-parent and parent-child dependencies.

Determine which application objects can be implemented using the target data warehouse schema. Document the results of this analysis and be sure to include:

A list of supported application objects

A list of application objects that require changes and how to implement those changes

A list of application objects not supported

 You can complete the analysis either at the report level or at a more detailed level by taking into account the attributes, metrics, filters, and custom groups that make up a report.

Scenarios: Identifying application objects

The table below presents some typical scenarios you can encounter while completing the application object mapping procedure above. The scenarios are listed by object type.

Object	Scenario	Action
Metrics / Base Formula	Different name	Update the name. All dependents read the change automatically.
	Not available	Delete after resolving dependencies. Filters and custom groups that use such a metric also have to be removed.
	Fact used is not available	Delete the base formula and its dependent metrics after resolving dependencies.
	Table used for counts has different keys	Check the Count Distinct and Fact ID parameters.
	Attribute or hierarchy used for dimensionality is not available	Evaluate whether the metric is meaningful without the dimensionality. If it is, change the name and use it only on appropriate reports.
	Filter used for condition is not available	Evaluate whether the metric is meaningful without the condition. If it is, change the name and use it only on appropriate reports.
	Transformation is not available	Delete the metric after resolving dependencies, because the metric meaning is completely changed.
	Compound metric: Constituent metric is not available	Evaluate whether the remaining constituent metric combination is useful. If it is, change the name and use it only on appropriate reports.
Custom Groups	Different name	Update the name. All dependents read the change automatically.
	One or more embedded filters not supported	Evaluate whether the remaining custom grouping is worth using on the report.
	Embedded filters not supported	Eliminate the custom group after resolving dependencies.
Prompts	Different name	Update the name. All dependents read the change automatically.

Object	Scenario	Action
	Embedded attribute or hierarchy not available	Delete the prompt and dependent filter after resolving dependencies.
	For object prompts, one or more of the constituent list not available	Evaluate whether the remaining list of objects is worth using in a report. If a single object remains, place it on the report and eliminate the object prompt.
Filters	Different name	Update the name. All dependents read the change automatically.
	Attribute Qualification: Attribute not supported	Eliminate the filter after resolving dependencies.
	Metric qualification: Metric not supported	Eliminate the filter after resolving dependencies.
	Relationship filters: Output level, filter qualification, or relate by fact/table not supported	Eliminate the filter after resolving dependencies.

Object	Scenario	Action
Reports	Different name	Update the name.
	Analysis not supported	Eliminate related reports, or move to a separate folder if there are plans for support at a later stage of the deployment.
	Attribute on template not available	Remove the report unless a suitable substitution can be found in the target data model.
	Metric on template not available	Remove the metric and keep the report.
	Filter or parts of it not supported	<p>If the entire filter is not supported:</p> <ul style="list-style-type: none"> • With relationship type set filters, the report may have to be removed. • With other filters, a substitution should be considered for limiting the result set. <p>If part of a filter is not supported, a substitution should be considered but may not be essential.</p>

Identify additional requirements

This procedure is optional.

The three gap analysis procedures performed above can include identifying data elements from the target data warehouse that can be added to the application metadata. This optional procedure is not described in this methodology. For a list of typical scenarios to customize and extend the Analytics Module, see [Customize and extend the Analytics Module, page 53](#).

Map the module

After you perform a gap analysis as described previously, you can map the schema objects (attributes, facts, and so on) to the new target schema and update application objects (reports, metrics, filters, and so on) to ensure they work against the new schema.

You can choose from the following methods for the mapping, depending on whether you want to use MicroStrategy Object Manager:

With Object Manager: Using MicroStrategy Architect and Developer, duplicate the packaged project and map the schema objects. Then move the application objects to the mapped project using MicroStrategy Object Manager. If you decide to use Object Manager, see [Map the module with Object Manager, page 81](#).

Without Object Manager: Using MicroStrategy Architect and Developer, make all changes in the original project. First update all application objects to use only those schema objects supported in the target data warehouse, and then map schema object links to use the new physical schema. If you decide not to use Object Manager, see [Map the module without Object Manager, page 86](#).

Mapping with Object Manager is more efficient because it allows you to implement only those portions of the Analysis Module that are relevant to your company's requirements.

Mapping prerequisites

Before you map the Analysis Module, make sure you have access to the following:

Skills

Experience creating analytical applications with MicroStrategy Developer and MicroStrategy Architect.

Tools	The results from your gap analysis, performed above.
	MicroStrategy Architect, to modify attribute and fact definitions so they use the tables and columns from the target data warehouse.
	MicroStrategy Developer, to modify definitions for the reports, metrics, filters, and so on that may be impacted by the missing attributes and facts in the target data warehouse.
	The Analysis Module's reference guide.
	(Optional) MicroStrategy Object Manager, to move objects (such as reports and metrics) between projects.

Map the module with Object Manager

If you decide to use MicroStrategy Object Manager to assist you in the mapping part of the porting process, you will perform the following procedures:

Use MicroStrategy Architect and Developer to duplicate the packaged project and map schema objects.

Use MicroStrategy Object Manager to move application objects to the mapped project.

If you decide not to use Object Manager, see [Map the module without Object Manager, page 86](#).

Overview: Mapping with Object Manager

This mapping method uses two MicroStrategy projects:

Master project: The original project for the Analytics Module

Destination project: The project that will be mapped to the target schema; created by duplicating the master project

Complete both mapping procedures below in the order they appear.

Map schema objects

This first procedure uses MicroStrategy Architect to map schema objects in the destination project to the target schema, based on results from your gap analysis at the beginning of this chapter. For details on the gap analysis, see [Perform a gap analysis, page 65](#).

An example follows this procedure.

To map schema objects

Using MicroStrategy Architect, duplicate the project for the Analysis Module. Use the project provided in the production metadata, and duplicate it to a new location. Use the option **Duplicate only schema objects**.



Document all actions taken during this mapping process to allow you to backtrack if necessary.

Update the data warehouse catalog to point to the target schema, adding relevant tables from the target schema to the destination project.

Map hierarchies based on the results from your gap analysis:

If a hierarchy exists in the destination project, proceed with mapping. Actions can include changing the name, removing or adding attributes, and so on.

If a hierarchy does not exist, remove it from the project.



Objects can only be removed from the MicroStrategy metadata if no dependents exist in the project. The order for removing objects throughout this procedure is determined based on that principle.

Map transformations based on the results from your gap analysis:

If a transformation exists in the destination project, proceed with mapping. Actions can include changing the name, updating the definition to the target schema tables and columns, and so on.

If a transformation does not exist, remove it from the project.

Map facts based on the results from your gap analysis:

If a fact exists in the destination project, proceed with mapping. Actions can include changing the fact name, updating fact expressions to point to the target schema tables and columns, removing or adding fact expressions, and so on.

If a fact does not exist, remove it from the project.

Map attributes based on the results from your gap analysis:

Start with the lowest attributes in each hierarchy, and proceed sequentially with all parents.

If an attribute exists in the destination project, proceed with mapping. Actions can include changing the attribute name; updating forms to point to the target schema tables and columns; adding or removing forms; updating, removing or adding relationships; and so on.

If an attribute does not exist, remove it from the project.


Update the project schema using the default options to make your changes available. Then follow the final mapping procedure, [Update application objects with Object Manager, page 83](#), to move all application objects to the destination project.

Update application objects with Object Manager

This second procedure uses MicroStrategy Object Manager to move all application objects that can be implemented to the destination project, based on the gap analysis completed earlier in this chapter.

The schema objects represent the building blocks on which application objects are defined. Once schema object definitions are updated to use the target schema, using the procedure above, application objects can be moved to the destination project.

Application objects must be moved in a certain order, based on which application objects are used as building blocks for other application objects. For example, to move a report with its attributes, metrics, and so on, the defining report must be present in the destination project. The procedure below will prompt you to move application objects in the appropriate order, based on specific considerations. The example following this procedure contains additional specifics on the appropriate order for moving objects.

 Be aware of the following:

Before you move any application objects, you must check on the components (schema and application objects) in the master project and determine whether these exist in the destination project. During the procedure below, use MicroStrategy Object Manager child-parent dependencies and its Search functionality to identify object components for a given object in the master project and determine whether all components exist in the destination project.

Document in detail all actions taken during this procedure.

To update application objects with Object Manager


Move prompts and filters, based on the results from the gap analysis, proceeding first with prompts and then filters.

If all components of the object exist in the destination project, move the object to the destination project using the option **Use existing definition in the destination project**. If a name change is required, do that in the destination project.

If all components are not present, do one of the following:

Move the necessary components to update the definition in the master project and then move the object.

Skip the object and update the definition to remove any unavailable components.

 Note the following:

Metrics can use filters in their definition, and filters can use metrics. In this case, filters are considered before metrics because the Analytics Module minimizes the use of metrics qualification within filters, while many metrics include filters in their qualifications. If a filter requires a metric, the metric must be moved first.

Filters qualifying on attribute or metric values must be updated after the filter is moved to the destination project. They also must use suitable new values in the existing data warehouse.

Move base formulas, based on the results from the gap analysis.

If all facts or attributes defining the base formula exist in the destination project, move the object to the destination project using the option **Use existing definition in the destination project**. If a name change is required, do that in the destination project.

If all components are not present, do one of the following:

Move the necessary components to update the definition in the master project and then move the object.

Skip the object and update the definition to remove any unavailable components.

Move metrics, based on the results from the gap analysis. Start with simple metrics, and then proceed with conditional, transformation, dimensional, and compound metrics. This order is important because, for example, compound metrics can be defined based on other metrics.

If all components of the object exist in the destination project, move the object to the destination project using the option **Use existing definition in**

the destination project. If a name change is required, do that in the destination project.

If all components are not present, do one of the following:

Move the necessary components to update the definition in the master project and then move the object.

Skip the object and update the definition to remove any unavailable components.



Changing some components such as dimensionality or conditionality can have some unexpected results in the reports.

Move reports, based on the results from the gap analysis.

If all components (attributes, metrics, prompts, filters, and so on) exist in the destination project, move the object to the destination project using the option **Use existing definition in the destination project.** If a name change is required, do that in the destination project.

If all components are not present, do one of the following:

Move the necessary components to update the definition in the master project and then move the object.

Skip the object and update the definition to remove any unavailable components.

Execute all reports to test the application objects and ensure that the results of your work in this procedure are correct.

Map the module without Object Manager

If you decide not to use MicroStrategy Object Manager to assist you in the mapping part of the porting process, you will:

Use MicroStrategy Architect and Developer to update all application objects to use only those schema objects supported in the target schema


Map schema object links to use the new schema

In this mapping method, all changes are completed in the original project, using Architect and Developer.

If you decide to use Object Manager, see [Map the module with Object Manager, page 81](#).

Update application objects without Object Manager

This first procedure updates the definition of the application objects to use only those schema objects that are supported, based on your gap analysis completed earlier in this chapter.

 Be aware of the following:

You must update application objects in a certain order, based on which application objects are used as building blocks for other application objects. For example, metrics must be removed from all reports before they can be deleted from the project. An object can be removed from the MicroStrategy metadata only if no dependents exist in the project.

Document in detail all actions taken during this procedure.

An example follows this procedure.

To update application objects without Object Manager

Move reports based on the results from the gap analysis:

If all components of the application object are supported, no change is required.

If all underlying objects are not supported, remove underlying objects from the definition.

If an application object is not supported, delete it from the project.

Move metrics based on the results from the gap analysis:

If all components of the application object are supported, no change is required.

If all underlying objects are not supported, remove underlying objects from the definition.

If an application object is not supported, delete it from the project.

Move base formulas based on the results from the gap analysis:

If all components of the application object are supported, no change is required.

If all underlying objects are not supported, remove underlying objects from the definition.

If an application object is not supported, delete it from the project.

Move filters based on the results from the gap analysis:

If all components of the application object are supported, no change is required.

If all underlying objects are not supported, remove underlying objects from the definition.

If an application object is not supported, delete it from the project.

Move prompts based on the results from the gap analysis:

If all components of the application object are supported, no change is required.

If all underlying objects are not supported, remove underlying objects from the definition.

If an application object is not supported, delete it from the project.

Execute all reports to test the application objects and ensure that the results of your work in this procedure are correct. Then follow the final

mapping procedure, [Update application objects without Object Manager, page 87](#).

Map schema objects

This second procedure maps schema object links to the target schema, based on results from your gap analysis at the beginning of this chapter. For details on the gap analysis, see [Perform a gap analysis, page 65](#).



Document all actions taken during this mapping process to allow you to backtrack if necessary.

An example follows this procedure.

To map schema objects

Map hierarchies based on the results from your gap analysis:

If a hierarchy is supported in the target schema, proceed with mapping. Actions can include changing the name, removing or adding attributes, and so on.

If a hierarchy is not supported, remove it from the project.



Objects can only be removed from the MicroStrategy metadata if no dependents exist in the project. The order for removing objects throughout this procedure is determined based on that principle.

Map transformations based on the results from your gap analysis:

If a transformation is supported in the target schema, proceed with mapping. Actions can include changing the name, updating the definition to the target schema tables and columns, and so on.

If a transformation is not supported, remove it from the project.

Map facts based on the results from your gap analysis:

If a fact is supported in the target schema, proceed with mapping. Actions can include changing the fact name, updating facts expressions to point to the target schema tables and columns, removing or adding fact expressions, and so on.

If a fact is not supported, remove it from the project.

Map attributes based on the results from your gap analysis:

Start with the lowest attributes in each hierarchy and proceed sequentially with all parents.

If an attribute is supported in the target schema, proceed with mapping. Actions can include changing the attribute name; updating forms to point to the target schema tables and columns; adding or removing forms; updating, removing or adding relationships; and so on.

If an attribute is not supported, remove it from the project.

Finally, after all schema objects have been mapped, remove all tables that are not used. The project schema must be updated to reflect all of the changes.

Customize and extend the module

When you customize or extend the Analytics Module, you are adding data elements that are specific to your company.

For typical customization scenarios, ideas and suggestions, and specific information on how to customize and extend the Analysis Module, see [Customize and extend the Analytics Module, page 53](#).

CREATING PORTABLE ANALYTICAL APPLICATIONS

This chapter presents the MicroStrategy best practices for designing and building enterprise-class analytical applications using the portability paradigm. Portability and the MicroStrategy portability methodology are explained fully in *Portability methodology, page 62*.

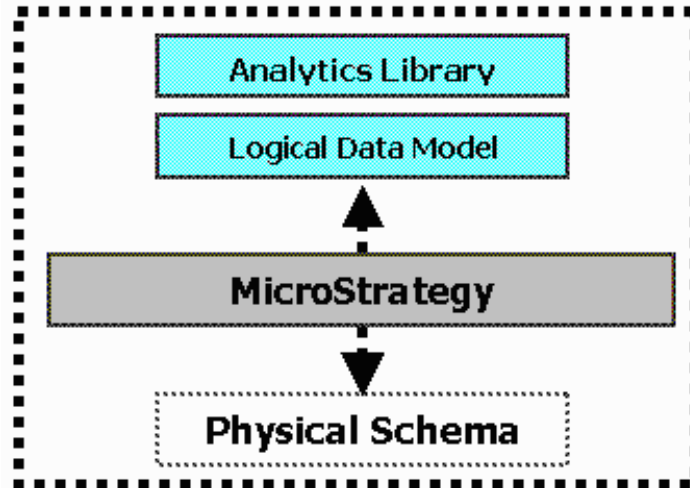
MicroStrategy has developed the content in this chapter based on experience developing the MicroStrategy Analytics Module.

This chapter describes how to build analytical applications using the portability paradigm and helps you understand how the MicroStrategy Analytics Module was designed and developed. For specific examples and more details, see the Analysis Module's reference guide and the related metadata components in your MicroStrategy software.

Architecture of portable analytical applications

The portability paradigm revolves around the concept of metadata-only applications. The MicroStrategy metadata repository allows the logical data model to be defined independently from the physical schema. This means that the business representation of the data is separate from the data's storage structure. For the logical data model, MicroStrategy supports a large number of physical schemas.

The diagram below reflects the independence of the Analytics Module, with the analytics library and logical data model, from the physical schema. Each major element's role in the architecture is also described.



Logical Data Model: This is the cornerstone of the architecture. It is the representation of the analytical domain in the MicroStrategy metadata. The logical data model is defined purely in terms of MicroStrategy schema objects such as facts (measures) and attributes (analysis concepts). The logical data model definition should not depend on any set physical data structures.

Analytics Library: The reports, or analytics, for the analytical domain are built on top of the logical data model. The reports consist of MicroStrategy application objects such as metrics, reports, filters, and so on. Reports are organized into several analysis areas. Definitions of the analytic objects should not depend on any set physical data structures but should be based entirely on the logical data model objects.

Physical Schema: (Optional) The physical schema that comes with the Analysis Module describes the data warehouse that links to the logical data model, and ultimately holds the data. It is used to build and test the logical data model, as well as to store data for evaluation purposes. Although the Analytics Module comes with a default physical schema, you can use the Module's portability feature to substitute an existing physical schema. The physical schema that comes with the Module is an optional part of the final components delivered as an analytical application.

Analytics Module versions

The Analysis Module includes two MicroStrategy projects, an Evaluation project and a Production project, as described in the following table. You can use this concept for your own analytical applications.

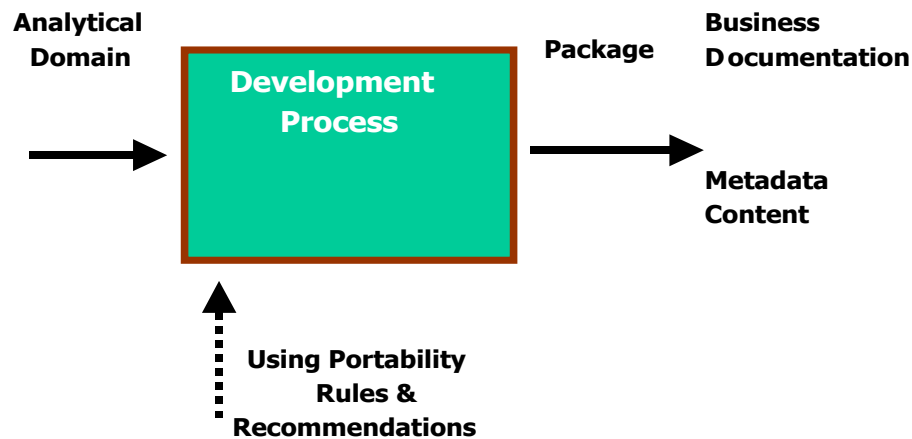
MicroStrategy Project	Description
Evaluation version of the Analysis Module	<ul style="list-style-type: none"> • This project allows you to evaluate the Analytics Module reports without setting up a data warehouse. • The project metadata repository and warehouse are built using Microsoft Access 2000. • The project is included in the evaluation CD of the MicroStrategy products. Installation automatically configures the project in a direct connection (in 2-tier mode). <p>Note: The Access database presents some limitations for supporting count distinct and outer join, because some of the reports in the evaluation version were specifically modified to support the Access data warehouse. Therefore, this project cannot be used directly against any warehouse other than Access 2000.</p>
Production version of the Analysis Module	<ul style="list-style-type: none"> • This project contains a production-ready metadata repository, and logical data model objects are mapped to a default physical schema. • The project metadata repository is built using Access 2000. • The project is configured for an Oracle database, but can be used to access any other major relational database such as DB2 or SQL Server. See the MicroStrategy readme file for details on supported databases.

Best practices for building portable applications

This section outlines the best practices for designing and developing analytical applications using the portability paradigm. It presents

development project guidelines with procedures and a development plan.

The development process starts with selecting an analytical domain (the business analysis area) and defining the basic analysis requirements. Then developers can use the best practices outlined in this section to design and build the analytical application.



Design and development premises

Developing a portable analytical application is not very different from developing a typical business intelligence application. However, the portability paradigm and architecture introduce a number of specific design and development premises:

The application must model a well-known and standard business problem area. The MicroStrategy development process ensures that most data elements in the application have a high probability of existing in the target data warehouse.

The application must be divided into several modular analysis areas, each of them aligned with sub-areas of the application business model. The MicroStrategy development process limits the impact on an analysis area if certain data elements do not exist in the data warehouse.

The logical data model must be defined to support typical analysis types. The MicroStrategy development process minimizes changes required to customize the application to each company's requirements.

The logical data model must be tested against existing physical schemas (data warehouses) in the domain area, using different database platforms. The MicroStrategy development process ensures the application's independence from a specific physical schema and database platform.

The development process must accomplish the dual goal of developing the application using MicroStrategy technology and developing the required business and technical documentation. The MicroStrategy development process builds documentation to support implementation and later customization and extension of the packaged components. Documentation is a key part of the analytical application package.

Development process overview

Based on the premises listed above and the application deliverables, MicroStrategy uses the following general process to build the analytics module:

Define the analytical domain.

Design and develop the logical data model, ensuring that it meets the analytical and portability requirements.

Define the report library.

Develop and test reports, including the packaged application components.

Throughout the development process, each phase includes creating important technical or business documentation. The documentation is used both for specifications for the next step of the development process and as the basis of the final business and technical documentation delivered with the application.

The development process is broken down into four main phases. Each phase is, in turn, broken down into several procedures. Follow each phase and its procedures in the order that they appear below.

Phase I: Define the analytical domain

In this first phase, you follow the procedures as they appear below, to:

Select the analytical domain (business analysis area)

Define the analytical domain by identifying the analysis requirements that specify the application's logical data model


Review the domain specifications that result from your work in the previous two procedures

To complete this phase successfully, make sure you have access to the following:

Skills	Knowledge of the analytical domain area.
	Knowledge of the criteria necessary to build portable applications.
	Experience defining and building analytical applications.
	Experience with overall application architecture and design, including the logical data model, physical schema, and report specifications.
	This role can be filled by experts in the domain, such as consultants with experience implementing this type of application.
	Experience with defining the business requirements.

Select the analytical domain

In this first Phase I procedure, you select an analytical domain and check to ensure it meets the criteria required for portable applications.

 This procedure should be performed by a person responsible for defining the business requirements.

To select the analytical domain

Apply the criteria listed in this step to determine an appropriate analytical domain. Based on the requirements for portability, not all analytical domains can be used to build this type of analytical application. Ensure you select your analytical domain with the following criteria in mind:

Criteria 1: The analytical domain must represent one of the following:

A standard business process, such as sales as defined by Miller-Heiman

A business problem area, such as customer retention

A functional area, such as marketing or service

A standard data source type, such as Web logs or SAP ERP

By restricting the application to a standard domain, you will ensure that the solution applies to a wide number of existing data warehouses and that it takes advantage of standard processes, best practices, and standard data that apply to the domain area.

Criteria 2: The analytical domain's business concepts must be standard.

To identify a data element as standard, review different sources for the analysis area. These sources can include data warehouses, operational systems, logs, and so on. If the data element is always present in most or all data sources, you can consider it as a standard data element. However, if you identify a data element that is not always present in data sources, do not consider it a standard data element and do not include it in your application.

This criteria determines the scope of the application, since standard business concepts define the analytical domain. It also ensures that those data elements will exist in the majority of the data warehouses for that domain.

Taking the criteria above into consideration, select the analytical domain that your application will address.


Evaluate your chosen analytical domain to assess the feasibility of building the application. Consider the following characteristics:

Size of the analysis area, measured as the number of standard data elements, process complexity, potential number of different report types,

and so on. As the analysis domain becomes bigger, development becomes more complicated and the portability paradigm becomes more difficult to accomplish.

In-house domain expertise. This is the knowledge required to define and build the application. This knowledge must exist internally.

Stability of the business rules associated with the domain area, due to industry/regulatory changes.

 If you are developing packaged analytical applications, you should also evaluate the following:

The domain's applicability across multiple verticals/industries. Be sure the application can be reused without major changes in existing data warehouses, independent of the industry.

Applicability across a sizeable geographic market.

Document the information you gather from the steps above. Your documentation should include:

A description of the analysis domain


Key business processes and problem areas

Typical data sources and data elements for the domain

This information will define the general scope of the analytical application and will identify the initial analysis requirements.

Define the analytical domain specifications

In this second Phase I procedure, you define the business analysis specifications to build the portable application.

 This procedure should be performed by a person responsible for defining the business requirements.

To define the analytical domain specifications

Define and document the data elements that make up the analytical domain. Begin by documenting a business process, including:

An explanation of the business process and main entities

Business process diagrams

Analytical needs, including typical usage scenarios and user roles

Document a list of business concepts, which describes all key concepts such as:

Business attributes (business-oriented concepts)

Facts

Metrics (measures)

The relationships between the previous items; for example, how each fact relates to the different business attributes and how they are calculated

Define and document the analytical domain's reporting areas. These are the logical analysis areas into which the domain can be broken. Focus on defining independent analysis areas to limit the impact of non-existent data elements in a future target data warehouse. Be sure this documentation includes:

A description of each reporting area within the analytical domain (the analytical scope)

For each reporting area, an explanation of the key analysis challenges, a list of the relevant attributes and facts, user analysis requirements, and how this reporting area fits with other analysis areas


A list and description of the key reports and types of questions they answer for each reporting area, with a focus on defining which and how concepts are analyzed together

Document a glossary of terms, including detailed descriptions and examples.

Combine all documented content from the first procedure above, with the content from this procedure to create your Analytical Domain Specifications document. See the Analytics Module reference guide for examples of these types of content.

Review the analytical domain specifications

In this third Phase I procedure, you discuss the analytical domain with external experts and the development team. You also identify the necessary quality assurance criteria for the application.


 This procedure should be performed by a person responsible for overall application architecture and design, working with other management, development, and testing team members, as well as domain experts, as appropriate.

To review the analytical domain specifications and define quality assurance criteria

Using the Analytical Domain Specifications document from the previous procedure, review the domain with external domain experts and your development team, or the project team in charge of developing the analytical application. Be sure discussions include:

Identifying potential issues and challenges that can appear when you define the logical data model that supports the application

Identifying the criteria necessary to ensure logical data model quality

 The best strategy to ensure logical data model quality is to build the key reports that identify the typical analysis types and relationships between business concepts. Success reproducing the key reports assures the robustness of the logical data model.

Update your Analytical Domain Specifications document to reflect the reviews above and any revisions necessary due to feedback, corrections, and potential issues.

Create a document of the list of key reports. This document will eventually summarize the criteria that assure the logical data model meets the analytical domain requirements. It will be finalized in a subsequent procedure, when the testing plan for the logical data model is defined.

Phase II: Develop the logical data model

In this second phase, you follow the procedures as they appear below, to:

Create the logical data model

Create the physical schema


Define an assurance/testing plan

Create a MicroStrategy project and execute testing

To complete this phase successfully, make sure you have access to the following:


Skills	Knowledge of the analytical domain area, typical logical data models used to model the domain, and typical physical schemas used.
	Knowledge of the criteria necessary to build portable applications.
	Extensive experience building analytical applications, as well as solid logical data and physical modeling skills.
	Working experience creating analytical solutions using MicroStrategy technology, including creating reports, metrics, filters, and other MicroStrategy objects.
	Experience with overall application architecture and design, including the logical data model, physical schema, and report specifications.
	This role can be filled by experts in the domain, such as consultants with experience implementing this type of application, or expert users in the analysis domain.


	Experience with defining business requirements.
	Experience with application development using MicroStrategy technology.
	Experience performing testing tasks.

The rules that appear in each of the Phase II procedures below are a collection of general design strategies. Additional rules and recommendations apply  when defining and building logical data model objects and reports for portability using MicroStrategy tools and technology. For more details see [Portability rules and recommendations, page 119](#).

Create the logical data model

In this first Phase II procedure, you define the logical data model on which the application will be built, by following the logical data model design rules listed within the procedure. This is a key procedure in the design and development process, since all further work depends on a robust logical data model that supports all the requirements of the analytical domain.

The design rules listed in this procedure are general design strategies. When you design a logical data model for portability and are using MicroStrategy  technology, additional design rules apply. For detailed rules for each object, see [Portability rules and recommendations, page 119](#) at the end of this chapter.

 This procedure should be performed by a person responsible for overall application architecture and design, working with developers and domain experts.

To create the logical data model

Using the Analytical Domain Specifications document you created in [Phase I: Define the analytical domain, page 96](#), identify the data elements that

define the core of the analytical domain and that are required to provide the core analysis. This step helps you adhere to the following rule:

Rule 1: Use only objects (data elements) that are standard across the domain.

Logical data model elements must be based on viable source system data.

The logical data model must incorporate only the basic facts captured by a standard organization.

Avoid elements that are industry- or country-specific, as well as those that may require customization.

Using the Analytical Domain Specifications document, identify and document the key analysis types for the analytical domain. This step helps you adhere to the following rule:

Rule 2: The logical data model must be designed to support generic types of analysis, not fixed reports.

A logical data model based on fixed reports is more rigid and harder to customize further.

Using the examples and tips listed in this step, identify and document 30-40 data elements (attributes and facts) to be included in the model. This step helps you adhere to the following rule:

Rule 3: Limit the logical data model to a manageable number of data elements and relationships.

Limiting the number of data elements and relationships helps keep the model simple.


By limiting the number to 30-40, you maximize the probability that the majority of the data elements are present in the target data warehouse.

Use simple (one-to-many) parent-child relationships. This ensures the model is less dependent on a specific physical schema.

Identify and document how different areas of the logical data model relate to each other. This includes identifying fact-attribute relationships and whether different facts exist at the same level of analysis. This step helps you adhere to the following rule:

Rule 4: Use a modular approach.

A modular approach ensures the different analysis areas depend on limited portions of the logical data model. If some data elements are missing from the target data warehouse, modularity limits the impact to only the areas using those data elements.

 If you are developing packaged analytical applications, you should also identify areas that can be customized and extended by customers. This allows you to provide guidance to clients when they customize and extend the application.

Identify and document any elements that depend on specific physical structures. This step helps you adhere to the following rule:


Rule 5: The logical data model should not depend on a fixed physical data structure.

For example, avoid many-to-many relationships because they require a specific physical schema.

This rule implies that the logical data model is supported by different types of physical schemas. This will be discussed further in the next procedure, [Create the physical schema, page 106](#).

Create a graphical view of your logical data model.

Combine all your documentation from this procedure. Make sure it includes technical and business content explaining each of the model objects with a description, relationships, data examples, and loading strategies.

 See the Analytics Module reference guide for examples of this documentation.


Review your work products from this procedure with appropriate team members, and incorporate any feedback.


If applicable, incorporate any changes or additions to the Analytical Domain Specifications document you created in [Phase I: Define the analytical domain, page 96](#) that may be required due to additions or corrections that occurred during this procedure. Be sure to track the changes in a revision table or similar method.

Create the physical schema

In this second Phase II procedure, you define a physical schema to support the analytical application development. This physical schema will be used to develop all MicroStrategy objects, such as attributes, facts, metrics, reports, and so on. The physical schema will be populated with test data for development, and with demonstration (demo) data to create a demo or evaluation version of the analytical application.

Most of the physical schema rules that appear in this procedure exist to facilitate a gap analysis between the default physical schema (the one you are creating with this procedure) and any future target physical schema (a data warehouse to which you will port the logical data model later). These rules also facilitate the mapping process that occurs during the porting process. For more details on porting, see [Portability methodology, page 62](#).

 The design rules listed in this procedure are general design strategies. When you design a physical schema for portability and are using MicroStrategy technology, additional design rules apply. For detailed rules on designing a physical schema, see [Portability rules and recommendations, page 119](#).

 This procedure should be performed by a person responsible for overall application architecture and design, working with developers and testers.

To create the physical schema

Using the logical data model and related documentation you created in the procedure above, design the physical schema using a star or snowflake pattern. This step helps you adhere to the following rule:

Rule 1: The physical schema must be designed to be simple and efficient.

Star and snowflake schemas are more common database schemas for analysis.

While different physical schemas can support the same logical data model, using the simplest one requires the fewest tables, joins, and columns.

Minimize the number of tables, especially lookup and relationship tables.

Avoid consolidated models and recursive hierarchies. They require a specific physical schema.

Avoid using any views in the schema design. Views can be dependent on the database engine. If they are required, use them when the target schema is mapped to the application during the porting process; see [Map the module, page 80](#) for more details.

Establish a naming convention to facilitate a gap analysis. For more information on a gap analysis and porting, see [Port the Analysis Module, page 64](#). This step helps you adhere to the following rule:

Rule 2: Define a naming convention for all schema objects.

Use prefixes to indicate specific table types. For example, use F_ to denote fact tables, L_ for lookup tables, A_ for aggregate tables, and so on.

Schema object names should be self-explanatory.

Limit object names to 18 characters since some DB2 databases have a limit of 18 characters.

Use suffixes to identify column types. For example, use `_ID` and `_DESC` for identifier and descriptor fields, respectively.

Ensure that a data element has the same column name across all tables.

To support portability across database engines, check to ensure your physical model avoids specific database parameters. This step helps you adhere to the following rule:

Rule 3: The physical model should be defined to avoid any specific database platform parameters.

Use standard data types across database platforms. For example, use `VARCHAR2(30)`.

Use the same data type for all columns representing the same data element.

Use the same data type for all data elements containing similar data. For example, define all description columns as `VARCHAR2(50)`.

Document the following items to facilitate mapping the application to a target physical schema:

Naming conventions, standard data types, and so on

Any deviation from the rules listed in this procedure, including an explanation of why the deviation occurred

Loading procedures for each of the physical schema tables

Make sure your physical schema adheres to the final two rules:


Rule 4: The physical schema will not include any query performance optimization.

Optimization is addressed during implementation, to account for specific client requirements and needs of the physical schema.

Rule 5: Document primary keys based on the logical data model relationships, although they are not required during the definition of the physical schema.

The logical data model, including attributes, relationships, and facts, determines which keys define the primary key for each physical schema table.

Review the physical schema with the appropriate individuals to ensure quality. Incorporate any feedback into the model where appropriate.

 You can also modify and update the logical data model to fit with these design rules or to address issues not previously considered. Be sure to track any changes made.

Using the information you have documented in this procedure, create:


A physical schema in Erwin format

A data dictionary and loading requirements to populate data structures

A mapping between the logical data model and the physical schema

Define an assurance/testing plan

In this third Phase II procedure, you determine the quality criteria that will ensure the logical data model supports the requirements of the analytical domain, and that the application is portable.

 This procedure should be performed by a person responsible for overall application architecture and design, working with developers and testers.

To define an assurance/testing plan

Using the Analytical Domain Specifications document created above, as well as the logical data model and physical schema also created above, identify the key reports that define the analytical domain.

Based on the key reports, create data scenarios for testing purposes. Be sure to include expected results.

Create a Test Plan document to ensure the logical data model can support different physical schema variations. This can include developing a few variations of your default physical schema.

Once your Test Plan is created, review it with appropriate individuals. Incorporate any feedback into the test plan where appropriate. In your review, be sure the Test Plan includes:

Report specifications and expected results based on your test data set

Data scenarios and a plan to generate the data to populate the default physical schema

A plan to map the MicroStrategy project to several physical schemas


Create a project and execute testing


In this last Phase II procedure, you develop and test a MicroStrategy project to ensure the logical data model meets requirements.

 This procedure should be performed by developers and testers.

To create a project and execute testing

Using the logical data model and physical schema documentation you created, as well as the Test Plan you created in the procedure above with its list of key reports and test data, create the MicroStrategy schema objects that represent the logical data model.

 See [Portability rules and recommendations, page 119](#), for important rules on building MicroStrategy objects for portability.


 Refer to your MicroStrategy documentation for basic information on creating a MicroStrategy project and objects.

Link the newly created schema objects to your default physical schema.

Create the reports you have identified and test them to ensure the data results are correct, the generated SQL is optimal and uses the appropriate tables, and so on.

Map the MicroStrategy project to different physical schemas to ensure that it supports other physical schema variations. See [Porting the Analytics Module, page 59](#), for details on mapping a logical data model to a different physical schema.

Review the results of your testing from the steps above, and identify and complete any fixes necessary to the MicroStrategy project.

 Be sure to update the Analytical Domain Specifications document, the logical data model and physical schema documentation, the Test Plan, and any other related documentation. Track any changes made to each individual document.

Once you have completed your review and made any updates, you have confirmed the quality of your MicroStrategy project, its logical data model, and its related set of reports. You will reuse this project in the next two phases to build the final report library.

Phase III: Define the report library

In this third phase, you follow the procedures as they appear below to define the reports library. The procedures are:

Define reports specifications

Review reports specifications and defining testing criteria

To complete this phase successfully, make sure you have access to the following:


Skills


Knowledge of the analytical domain area and the best analytical practices in that area.

	Working experience creating reports, metrics, filters, and other MicroStrategy objects.
	Experience with overall application architecture and design, including the logical data model, physical schema, and report specifications. This role can be filled by experts in the domain, such as consultants with experience implementing this type of application, or expert users in the analysis domain.
	Experience with defining business requirements.
	Experience with application development using MicroStrategy technology.
	Experience performing testing tasks.

Define the reports library

In this first Phase III procedure, you define a library with 30 to 50 reports that address the key analytical requirements for your analytical domain.


 The design rules listed in this procedure are general design strategies. When you design a report library for portability using MicroStrategy technology, additional design rules apply. See [Portability rules and recommendations, page 119](#) for detailed rules for designing reports.

 This procedure should be completed by a person responsible for overall application architecture and design and a person responsible for gathering business requirements. It can include input from developers.

To define the reports library

Research the analytical domain and identify the reports that need to be developed. This step should adhere to the following rule:

Rule 1: The report library should be defined within the boundaries of the logical data model.

Use only objects and relationships that exist in the model. Do not modify a tested model to support new report requirements. Changes can impact the  integrity and robustness of the model. If new reports require changes, move back to *Phase II: Develop the logical data model, page 102* and proceed again from that point.


Define a set of 30 to 50 reports that focus on the best analytical practices for your analytical domain. This step helps you adhere to the following rule:

Rule 2: The application should provide 30 to 50 reports.

This set of reports will include:

Key reports defined in the procedures completed above

Additional reports that use MicroStrategy advanced analytical capabilities such as ranks, contribution, transformations, and so on; see the *MicroStrategy Advanced Reporting Guide* for details on advanced reporting features

Make sure the reports do not rely on any physical schema structure. The design rules listed in this procedure are general design strategies. When you  design reports for portability and are using MicroStrategy technology, additional design rules apply. See *Portability rules and recommendations, page 119* for detailed rules for designing a report.

This step provides a core group of reports to address key business questions in the analytical domain. They can also be used as templates to create additional reports later.

Arrange your reports into groups that reflect business problem areas. This step helps you adhere to the following rule:

Rule 3: The analytical library should be organized using analysis areas.

Make sure the analysis areas are aligned with specific problem areas, so that each group of reports consists of a unique report collection.

Make sure each analysis area uses a different part of the logical data model, so reports do not overlap. This limits the impact on the porting process later, if data elements are missing from the target data warehouse.

For each report identified in the steps above, write a definition that includes:

Report display (grid/graph)

Report formatting

Report filter conditions

Any other report definitions that are appropriate



See the Analytics Module reference guide for examples of the types of report definitions to be documented.

Identify all the objects required to build each report, such as metrics and filters.

Using the information gathered in this procedure, as well as the Analytical Domain Specifications document, the logical data model and physical schema documentation, and the MicroStrategy project created in the procedures above, create a Report Specifications document. The Report Specifications document should contain the following information for each report:


A description of each report's business value, key performance indicators (KPIs), and usage scenarios

The layout and formatting for each report

The filter conditions applied to each report


Typical drilling scenarios for each report, where applicable

The definition of all objects required to build the reports, such as metrics, custom groups, filters, attributes, and so on

 See the Analytics Module reference guide for examples of this documentation.

Review report specifications and define testing criteria


In this last Phase III procedure, you review the report specifications you created in the procedure above, and you define the report testing criteria.

This procedure should be completed by developers and testers. It can include  input from management, a person responsible for overall application architecture and design, and/or other experts in the analytical domain area.

To review reports specifications and define testing criteria

Using the Reports Specifications you created in the previous procedure, review each report with experts in the analytical domain.

Define criteria to test each report. Document your criteria in the Reports Specifications document, and be sure to track the update to the document.

 See [Portability rules and recommendations, page 119](#) for special object design rules for portability. These rules can also inform your report testing scenarios.

Define and document a real scenario for data generation to address testing and demonstration (demo) needs. Your documentation should include:

A data generation plan

Your data scenarios for testing and demo needs

Phase IV: Develop and test the reports

In this fourth (and final) phase, you follow the procedures as they appear below to create and test the reports for your analytical application. The

procedures are:

Develop the reports

Test the reports

Package the application

To complete this phase successfully, make sure you have access to the following:


Skills	Knowledge of the analytical domain, logical data model, physical schema, reports library, and report specifications.
	Working experience creating reports, metrics, filters, and other MicroStrategy objects.
	Experience building and documenting analytical applications.
	Working experience developing technical documentation.
	Experience with overall application architecture and design, including the logical data model, physical schema, and report specifications.
	This role can be filled by experts in the domain, such as consultants with experience implementing this type of application, or expert users in the analysis domain.
	Experience with application development using MicroStrategy technology.
	Experience performing testing tasks.

Develop the reports

In this first Phase IV procedure, you create the report library for your analytical application, including any supporting objects such as metrics and filters.

When you develop reports for portability and are using MicroStrategy technology, additional design rules apply. See [Portability rules and recommendations, page 119](#) at the end of this chapter for detailed rules for reports.




-  This procedure should be completed by developers, working with a person responsible for overall application architecture and design.
-


To develop the reports

Make sure you have the Reports Specifications document as well as the logical data model and physical schema documentation available. Make sure you also have access to the MicroStrategy project as created in [Phase II: Develop the logical data model, page 102](#).

Create all objects required for the reports. Objects to be created include such supporting objects as metrics and filters.

-  Be sure to add a detailed description to all objects created with MicroStrategy tools.

Once the objects are created, design the reports to include the final formatting. Your completed reports form the report library for your MicroStrategy project.

-  Be sure to update the Reports Specifications document if any modifications were required during report development.

Test the reports

In this second Phase IV procedure, you test all reports to ensure they meet your documented Reports Specifications.

-  This procedure should be performed by developers and testers.
-

To test the reports

Make sure you have access to your MicroStrategy project, including your default data warehouse with test/demo data loaded. Also be sure you have your Report Specifications document and your Test Plan.

Execute your Test Plan.

Review report results, such as the following:

Report data set

Report display

Report formatting

SQL


Identify any issues and resolve them. Be sure to update any related documentation that may be affected.

Take final screen shots of each report, and include the screen shots in your Report Specifications document.

Once this procedure is complete, you have produced the final version of your MicroStrategy project. You have also produced updated versions of the Report Specifications document to include screen shots of the final reports.

Package the application

In this last Phase IV procedure, you create the final package for your analytical application, including the metadata components and documentation.

 This procedure should be performed by developers, testers, and writers.


To package the application

Gather all documentation created during the development process (Phases I-III plus the completed procedures in Phase IV). Be sure to include all business and technical specifications.

Be sure you have access to the MicroStrategy project, including metadata, default physical schema, and demonstration (demo) database.

Transform your existing documentation to a format appropriate for your target audience (such as developers, system administrators, end users, or another group). Depending on the audience, your documentation's level of detail can differ significantly. For example, a software company developing a packaged application intended for an outside audience might create more detailed documentation. A company producing an application for internal use might need less detailed documentation in certain areas.

Package the software components, and create installation routines if required.

 See [MicroStrategy Analytics Module, page 22](#) for details about packaged components included with the Analytics Module. The Analytics Module can be used as a template for documentation and packaging.

Your final package includes documentation, metadata-ware (MicroStrategy metadata, a demo data warehouse, and so on), and a demo/evaluation version of your analytical application.

Portability rules and recommendations

This section presents development rules and recommendations for creating enterprise-class portable analytical applications within the context of the MicroStrategy platform, based on best practices used by MicroStrategy to develop the Analytics Module. It includes recommendations for creating MicroStrategy objects.

The rules and recommendations are important to ensure that the analytical application is portable. See [Chapter 5, Porting the Analytics Module](#) for a full explanation of portability. These rules help minimize the impact a missing data element in the target data warehouse has on the logical data model (the packaged components). Some of the rules and recommendations are best practices guidelines and can be applied when you are developing any analytical application. The rules also facilitate application customization and extension.

This section is organized into two subsections:

Schema Objects: This subsection has rules for developing the schema objects (attributes, facts, and hierarchies) in the logical data model.

Application Objects: This subsection has rules for developing the application objects (reports, metrics, filters, and so on).

Rules and recommendations include examples from the MicroStrategy Analytics Module. See the module's reference guide and metadata components for details and definitions.

When defining MicroStrategy objects, it is recommended that you always add a detailed description. This facilitates future changes and additions, particularly if an object does not comply with the rules and recommendations below. A detailed description can include such information as an object's type, a specific design rule and the reason why it was not used, a general object definition, guidelines for customization, and so on.

Schema objects

MicroStrategy schema objects are the representation of the logical data model in MicroStrategy technology. Schema objects include such items as attributes, facts, and hierarchies. They are defined with MicroStrategy Architect.

Attribute rules and recommendations

Attributes are the entities that represent the business dimensions (business concepts), such as Customer or Product.

Attribute forms

Attributes are represented by at least one attribute form.

Designing the logical data model:

Assign two forms to each attribute, an identifier and descriptor.

Defining the physical schema:

Each form requires a column in the database. For example, an attribute with two forms requires two columns.

Defining the attribute in MicroStrategy:

Define two forms, ID and DESC, using the standard form of numeric for ID and text for DESC.

For date attributes, use one form, defined as ID with a form type of Date.

Set the ID form as the key for the attribute. Set the DESC form as the display default.

Add descriptions to all forms. Descriptions should indicate the business meaning and the source.

Avoid compound keys and group forms. Only specific physical schemas support compound keys.

Attribute form expression

Each attribute form is linked to the physical schema, defining a form expression. A form expression can be column names from the physical schema such as CUST_ID or a combination of names using a database operator, such as [FIRST_NAME]+[LAST_NAME].

Defining the physical model:

Map each attribute form to a single column in the physical schema. Avoid combining columns using database operators.

Use a homogeneous naming convention by applying the same column name in all tables where it appears. This convention requires only one form expression to be set up. If the target warehouse has different column names, additional form expressions can be added.

Defining the form expression in MicroStrategy:

Use the guidelines in the bullet above for a single column and one form expression per form, with the same column name in all tables.

If a more complex form is required, avoid database-specific operators or functions.

Avoid constant values (also called derived attributes) in the form expression. These are mapped to a specific table and require a similar table to exist in the target data warehouse.

Attribute relationships

Attribute relationships define how different attributes are associated.

Designing the logical data model:

Use one-to-many or one-to-one relationships whenever possible, since they are less dependent on specific table structures.

Avoid many-to-many or many-to-one relationships, because such relationships assume a certain physical schema and require a relationship table.

Defining the physical schema:

Use lookup or relationship tables to establish attribute relationships. Avoid using fact tables.

A many-to-many relationship cannot be defined in lookup tables and requires an additional relationship table.

Defining the attribute relationship in MicroStrategy:

Use lookup tables as the relationship table, based on the guidelines in the bullet above.

Fact rules and recommendations

Facts are the entities that define the business measures, such as revenue, cost, and price.

Fact expressions

Facts are represented by at least one fact expression in MicroStrategy. Fact expressions link facts with the physical schema. They can be a single column such as `REVENUE`, several columns combined using operations such as `profit=REVENUE-COST`, or a constant value.

Designing the logical data model:

Represent each fact by a single concept in the logical data model.

Defining the physical schema:

Define facts based on a single column.

Use a homogeneous naming convention by using the same column name in all the tables where the fact exists.

Defining the fact expression in MicroStrategy:

Use one fact expression per fact, mapped to a single column. Irregularities in the target data warehouse, such as different column names or multiple columns, can be addressed during the implementation.

If several columns and operators are required, avoid database-specific functions.

Defining fact expressions with a constant value (derived facts):

Fact expressions with a constant value (derived facts) are used to define count metrics in a specific fact table, and so they assume a specific table name and table structure. Document any derived facts, including table structure. This facilitates mapping to a target data warehouse, since derived facts require a fact table with a similar structure.

Fact levels

The fact level determines the attribute level at which the fact can be analyzed.

Designing the logical data model:

Define facts at the lowest attribute level that can be used for analysis.

Defining the physical schema:

Ensure all required attribute keys are in the fact table. These are usually the lowest attribute in the hierarchy.

Defining the fact level in MicroStrategy:

Avoid fact level extensions whenever possible. All required relations must exist at the fact table level.

Hierarchy rules and recommendations

Hierarchies are groups of attributes, organized in a way which defines their relationships to each other. For example, the Time hierarchy consists of the group of all time-related attributes such as Year, Quarter, Month, and Date.

MicroStrategy comes with one hierarchy, called System, which represents the relationships defined at the attribute level. You can also define user hierarchies to create custom groupings.

Defining hierarchies in MicroStrategy:

Use the System hierarchy.

Avoid user hierarchies. However, you can use them to provide prompt selections for any level in a hierarchy.

Rules for other schema objects

You must consider some additional schema objects when building analytical applications. They include transformations, tables, and mapping

methods.

Transformations

Defining transformations in MicroStrategy:

If you need transformations, use simple base table transformations.

Do not use expression-based transformations, especially if they require database-specific functions.

Tables

Defining the physical schema tables in MicroStrategy:

Use the table size calculated by MicroStrategy Architect.

Use the default values generated by MicroStrategy Architect for the keys.

Mapping methods

This parameter, which applies to attributes and facts, indicates how to update physical mappings when the data warehouse catalog is updated.

Defining a mapping method:

Use the Automatic method because it automatically updates all mappings when new tables are added.

If required, change the mappings manually in each attribute or fact.

Alternatively, disable automatic checking for all objects.

Application objects

MicroStrategy application objects are the representation of the reports and business measures in MicroStrategy, such as metrics, filters, prompts, reports, and so on. These are defined based on schema objects (attributes and facts) and are created using MicroStrategy Developer.

Consolidations and custom groups rules and recommendations

These application objects are used to define special groupings of attributes.

Consolidations

Avoid using consolidations because their definition is based on filter conditions. Consolidations assume that certain attribute values exist in the target data warehouse. For more information, see [Filters rules and recommendations, page 129](#).

Custom groups

Custom groups can be based on filters (qualifying attributes or metrics) or on banding metric values.

Avoid using custom groups based on filters. They assume that certain attribute values exist in the target data warehouse.

Use custom groups based on metric bands.

Document the definition, including the steps required to customize them for the target data warehouse.

Base formulas and metrics rules and recommendations

Base formulas are used to define a metric formula without including other parameters such as level, conditionality, transformation, VLDB properties, and so on. Base formulas create an additional level of abstraction between the physical schema and the metrics, that is, the actual calculations.

Base formulas

Use base formulas if several metrics share the same formula.

If changes are required when implementing the analytical application against a target data warehouse, applying the changes to the base formula automatically updates all metrics using that base formula.

The Analytics Module makes extensive use of base formulas. The majority of the metrics are defined based on a collection of key formulas, which are located in one folder for easy access.

Do not use database-specific functions in the formula.

Minimize the number of facts or attributes (for counts) used.

Use simple functions or operators.

For some count metrics, special parameters such as `FACT ID` may be required. The associated fact must be updated to point to the new physical schema. If a table with a similar structure does not exist, then the base formula cannot be implemented.

Metrics

Metrics define how facts are calculated when used in reports. The definition can include a formula based on facts, attribute counts, or other metrics.

Simple metrics

Simple metrics include a calculation applied to facts or attributes.

Creating the metric formula:

Use base formulas whenever possible. See [Base formulas and metrics rules and recommendations, page 126](#), for more details.

Use simple functions or operators. Do not use database-specific functions.

Minimize the number of facts or attributes (for counts) used.

Avoid special operator parameters, although `FACT ID` and `count distinct` may be required for some count metrics.

Adding conditionality to metrics:

Avoid conditions because a qualification implies the existence of specific values in the target data warehouse. For more information, see [Filters rules and recommendations, page 129](#).

If conditions are required, document the qualification and how to customize it to the target data warehouse.

Adding metric levels:

Use the default value of the report level.

Avoid defining a specific level, which implies the existence of the attribute in the target data warehouse. For non-aggregatable metrics, levels require a specific physical schema.

If a level is required, document the definition and how to customize it to the target data warehouse.

For example, the Analytics Module includes a number of dimensional metrics as examples of contribution analysis (percent to all calculations). The module also includes a number of non-aggregatable metrics.

Creating transformation metrics:

Add examples based on typical time transformations, such as Month vs. Previous Month. For example, the Analytics Module includes some transformation metrics as examples of time-based analysis.

Compound metrics

Compound metrics are defined based on other metrics and arithmetical operators. These metrics do not include level, condition, or transformation.

Creating compound metrics:

Limit the number of metrics used in the definition. This minimizes the impact if some metrics are missing from the target data warehouse.

Other metric parameters

Other metric parameters include formatting, subtotals, VLDB properties, join type, and so on.

Use outer join as the default for all metrics in both the formula and metric join type. The join required could differ depending on the report in which the metric is used. However, this setting can be changed at the report level.

Most of the metrics in the Analytics Module use outer joins by default. This ensures that data for an attribute is always returned, even if one metric does not return any values.

Use the default value for all VLDB properties. To address irregularities, change the property at the report level and document the change.

The metric format is based on the business purpose of the metric.

Use defaults for the subtotals. If changes are required for a particular report, make the changes at the report level and document them as part of the definition.

Filters rules and recommendations

A filter specifies the conditions that apply to the data included in the report. A qualification assumes the existence of certain values or a particular data structure in the target data warehouse.

Minimize filter usage. If a filter is needed, document it and how to change the definition.

Create a collection of reusable filters. Creating standard filters and reusing them in metrics and reports minimizes the number of filters that are needed. When a filter must be modified, the changes are automatically applied to all objects using it. If filters are defined in each report, changes must be made in each embedded filter. A collection of filters has been defined for the Analytics Module.

Use prompts to define a qualification when the report is executed, rather than hard-coding it.

Attribute qualification

Avoid qualifying on attributes, because this assumes the values exist in the target data warehouse. For example, a filter is defined as `month=Oct-2001`. However, the value may not exist or the client may use a different format for the month ID, such as `month=102001`.

Do not use database-specific functions when defining the qualification. If data functions are required for date type filters, use the MicroStrategy date functions such as Current Date. These are independent of the database engine.

Set qualification: metrics

Avoid metric qualifications, because they assume the values exist in the target data warehouse.

Do not use database-specific functions when defining the qualification.

Set qualification: relationship filters

Avoid using output level, including relate output level and filter qualification, because it assumes a physical schema structure or table name.

Prompts rules and recommendations

Prompts can be used to define a run-time qualification or to select attributes and metrics to be displayed in the report.

Use prompts instead of static filters, as discussed in [Filters rules and recommendations, page 129](#).

Create a collection of reusable prompts.

Use object prompts to define the report template dynamically or select it from a list of filters. Document the attributes or metric used, including an explanation of how to update the prompt if the objects are missing from the target data warehouse.

Avoid value prompts. If a value prompt is required, do not define the default values.

Avoid level prompts, because they assume that the metric can be calculated at a certain attribute level in the target data warehouse.

Reports rules and recommendations

The report object defines how information is calculated and displayed for the analytic.

Filter definition

Use the rules from [Filters rules and recommendations, page 129](#).

Use embedded filters and prompts instead of static filters. All reports include prompts to allow dynamic selection.

Template definition: report layout

Limit the number of metrics and attributes in the report. This minimizes changes if some data elements do not exist in the target data warehouse.

Use object prompts to allow dynamic selection of metrics or attributes when they could be different in the target data warehouse.

Template definition: formatting

Use the defaults for sorting, attribute display, subtotal, and so on.

Avoid advanced sorting and subtotals. Users can set them up based on their needs.

Minimize using graphs, because they are very dependent on the report template and values. If the report changes, the graph may be invalidated.

Template definition: other parameters

Use the default VLDB properties.

Use the default values for the report data options.

Avoid report limits because they assume certain values exist in the target data warehouse.

Avoid metric aliases.