Oracle Commerce Guided Search

Developer Studio Help

Version 11.2 • October 2015

ORACLE®

COMMERCE

Contents

Copyright and disclaimer

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

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

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

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

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible

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

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

Preface

Oracle Commerce Guided Search is the most effective way for your customers to dynamically explore your storefront and find relevant and desired items quickly. An industry-leading faceted search and Guided Navigation solution, Guided Search enables businesses to influence customers in each step of their search experience. At the core of Guided Search is the MDEX Engine™, a hybrid search-analytical database specifically designed for high-performance exploration and discovery. The Oracle Commerce Content Acquisition System provides a set of extensible mechanisms to bring both structured data and unstructured content into the MDEX Engine from a variety of source systems. The Oracle Commerce Assembler dynamically assembles content from any resource and seamlessly combines it into results that can be rendered for display.

Oracle Commerce Experience Manager enables non-technical users to create, manage, and deliver targeted, relevant content to customers. With Experience Manager, you can combine unlimited variations of virtual product and customer data into personalized assortments of relevant products, promotions, and other content and display it to buyers in response to any search or facet refinement. Out-of-the-box templates and experience cartridges are provided for the most common use cases; technical teams can also use a software developer's kit to create custom cartridges.

Chapter 1

Welcome to Oracle Endeca Developer Studio

Oracle Endeca Developer Studio is a Windows application that you use to define all aspects of your instance configuration. Use Developer Studio to assemble and configure pipeline components for processing your data.

Overview of Developer Studio functions

Endeca Developer Studio is a Windows application that you use to define all aspects of your instance configuration.

With Developer Studio, you can define:

- Pipeline components for tasks such as loading, standardizing, joining, mapping, and exporting data.
- Endeca properties and property attributes such as sort and rollup.
- Dimensions and dimension values, including dimension hierarchy.
- Precedence rules among dimensions that provide better control over your implementation's navigation flow.
- Search configurations, including which properties and dimensions are available for search.
- Dynamic business rules that allow you to promote certain records on your Web site using data-driven business logic. Dynamic business rules are used to implement merchandising and content spotlighting.
- User profiles that tailor the content returned to an end-user based upon pre-configured rules.

Developer Studio uses a project file, with an .esp extension, that contains pointers to the XML files that support an instance configuration. Editing a project in Developer Studio edits these underlying files.

Note: A few features still require manual editing of the XML files. Those cases are noted in the documentation.

Deprecated Features

The Microsoft Exchange Forge Adapter is deprecated in the Platform Services 11.0 release.

Existing Microsoft Exchange Forge Adapter implementations will continue to be supported. Customers who are interested in indexing information from the Microsoft Exchange systems are encouraged to develop a custom adapter. Custom Adapters are described in detail in the *Content Adapter Developers Guide* on the Oracle Technology Network.

Parts of the Developer Studio window

The Developer Studio window contains the menu bar, toolbar, Project Explorer, work area, and Messages pane.

The Developer Studio window contains the following sections:

- The menu bar and toolbar
- The Project Explorer
- The work area
- The Messages pane

Important Endeca concepts

In order to work with the Endeca Developer Studio, you need to understand fundamental Endeca concepts of an Endeca implementation, including those described in this section.

In order to familiarize yourself with the following concepts, we highly recommend that you read the *Oracle Endeca Guided Search Concepts Guide* and the introductory sections of the *Endeca Basic Development Guide* before beginning to work with Developer Studio.

- Source properties and record
- Endeca properties and record
- Dimensions, dimension values, and dimension hierarchy
- Guided Navigation
- Record search and dimension search

Note: You must have Adobe Reader, available from the Adobe Web site, installed to view them.

About your Endeca Developer Studio Project

Endeca Developer Studio is a project-based tool that creates and modifies an Endeca instance configuration.

The instance configuration consists of the following components and files:

- Your Developer Studio project has a project file that is typically the name of your project followed by an Endeca Studio Project (.esp) extension. This file tracks the other files used in the project.
- The Pipeline.epx file is the Developer Studio pipeline file. The Pipeline.epx file is composed of a collection of components. Each component performs a specific function during the transformation of your raw data into Endeca records. Components are linked together, by means of cross-references, giving the Pipeline.epx file a sequential flow and a "pipeline" feel. At a minimum, a Pipeline.epx file must contain a component to load data and a component to save data. Additional components are added as required to perform other tasks.
- The Dimension.xml file(s) that store the dimension hierarchy. The dimension hierarchy is a collection of dimensions and their dimension values. A data pipeline can have more than one Dimension.xml file.
- Multiple XML files for storing the index configuration. Developer Studio allows you to specify the index configuration in its various editors. This includes specifying the display and handling of dimensions, precedence rules, record sorting, record search, and so forth. The index configuration resides in multiple, smaller XML files.

Creating a new project automatically creates new files for the project. Saving a project can potentially update all of these files.

About Oracle Endeca Workbench

In addition to Developer Studio, the Endeca distribution includes the Oracle Endeca Workbench tool. Workbench is an optional, Web-based application that contains a complementary set of functionality to that found in Developer Studio.

Unlike Developer Studio, which provides a rich development environment for configuring all aspects of an Endeca implementation, Oracle Endeca Workbench focuses on a smaller set of common, every day provisioning, configuration and maintenance tasks. This reduced focus gives Oracle Endeca Workbench a smaller footprint that can exist within the bounds of a Web-based application.

Oracle Endeca Workbench is a Web-based tool intended for business users and system administrators. For business user information, see the *Oracle Endeca Workbench User's Guide*. Oracle Endeca Workbench fully interacts with the EAC Central Server for provisioning and system operations, it lets you configure some aspects of your instance configuration and allows provisioning and system operations.

With Oracle Endeca Workbench, system administrators can perform any of the following tasks:

- Provision the hosts available to an Endeca implementation.
- Provision the applications available to an Endeca implementation.
- Provision the scripts, such as the report generator script and the baseline update script, to an Endeca implementation.
- Configure SSL settings, report generation, and set up a preview application for dynamic business rule testing.
- Perform system operations such as running baseline updates or starting and stopping the MDEX Engine or Log Server.
- Monitor the status of system components such as Forge, Dgidx , MDEX Engine, Log Server, and Report Generator.

Oracle Endeca Workbench and Developer Studio require the Endeca Application Controller (EAC) to control and communicate with other components and hosts in an Endeca implementation.

The two primary audiences for Oracle Endeca Workbench are:

- Business users who define business logic such as merchandising/content-spotlighting rules and thesaurus entries.
- System administrators who provision, maintain, monitor and manage an Endeca implementation.

Oracle Endeca Workbench allows business users to make changes to parts of an Endeca implementation after the pipeline and instance configuration features has been developed for the application in Developer Studio. For example, a developer uses Developer Studio to specify which Endeca properties and dimensions are available for search, then a business user uses Oracle Endeca Workbench to specify thesaurus entries that support search functionality, and a developer uses Oracle Endeca Workbench to provision an application and Oracle Endeca Guided Search components to the EAC Central Server.

Oracle Endeca Workbench provides administrators with features that provision all parts of the Endeca implementation, start, edit and stop components and scripts, monitor the status of an Endeca implementation, and download an implementation's instance configuration for debugging and troubleshooting purposes.

For both audiences, Oracle Endeca Workbench provides access to reports that describe how end-users are using an Endeca implementation, for example, the most popular search terms, the most popular navigation locations, search terms that are most often misspelled, and so forth

Note: For the most part, you can use Developer Studio to make the same changes that a non-technical user might make in Endeca Workbench. One exception is the changing the state of dynamic business rules, such as from Inactive to Active.

You can share the work on a project with others in your organization. For example, you might use Developer Studio to create dynamic business rule zones, styles, and rule groups, and then pass the project to a colleague on your marketing team, who could use Oracle Endeca Workbench to create and test the dynamic business rules themselves.

In Oracle Endeca Workbench, it is possible for multiple users to log in and make non-conflicting changes at the same time. However, between Oracle Endeca Workbench and Developer Studio, there is no built-in allowance for concurrent users. Therefore, to prevent changes from being overwritten or otherwise lost, a project should be active in only one of these applications at a time. For example, in the situation outlined above, after you send the project to the Workbench user, you should wait until the Oracle Endeca Workbench user has finished making modifications before pulling the application back into Developer Studio for continued development.

Developer Studio workflows

Developer Studio is useful by itself for creating instance configuration XML files you need for your Endeca application. You may also split application maintenance tasks between a Developer Studio user and a Workbench user.

You can use Developer Studio on its own to generate the instance configuration XML files needed to run your Endeca application. After Developer Studio generates the XML files, you can incorporate them into EAC scripts or applications that run and manage the Endeca application.

Chapter 2

Getting Started

Creating and Validating Projects

This section provides information on using Developer Studio to create and validate template-based projects.

Creating a new project based on a template

The Endeca Developer Studio offers project templates that can be used as a foundational base for new projects.

To create a project based on a template:

1. From the File menu, select **New Project**. The New Project editor appears.
2. Select a project type from the "Select a project type" list.
3. In the **Project name** text box, type a name for the project. This name appears in the title bar of the project.
4. In the **Save project as** text box, browse to the location where you want to save the project and then type a name for the project file.
5. Click **OK**.

Creating a new project automatically creates rudimentary pipeline components. After you have created a new project, you can use Developer Studio to extend and populate it.

Opening a project

Pre-existing projects can be re-opened in Developer Studio.

To open an existing Developer Studio project:

1. From the File menu, select Open Project.
2. In the Open dialog box, browse to the project you want to open.
3. Click the project file to open it. Typically the project file is the name of your project, followed by an .esp (Endeca Studio Project) extension.

   Note: You can have only one project open at a time.

Saving a project

Be sure to save projects before closing.

To save a project after making changes to it:

From the **File** menu, select **Save Project**.

Note: If there are errors in your project when you save it, you will receive a message on the **Validation** tab of the **Messages** pane.

Backing up a project

Save a project to multiple locations to protect against data loss.

To create a backup by saving a project to a different location:

1. From the File menu, select **Save project as**.
   The Select Project Folder dialog box appears.
2. Browse to the location where you want to save the project.
3. Click **OK**.

Displaying project components

The View menu allows you to change the screen elements that are displayed in Endeca Developer Studio.

To choose which elements are displayed in Developer Studio:

Click the **View** menu, and select the item you want to display.

Menu item	**Description**
Standard Objects	Allows you to drill down to display Properties, Dimensions, Dimension Groups, Precedence Rules, Search Interfaces, and User Profiles.
Search Configuration	Allows you to drill down to display the Thesaurus and Stop Words.
Dynamic Business Rules	Allows you to drill down to display Rules, Zones, and Styles.
Pipeline Diagram	Lets you drill down to the main (Baseline) Pipeline and the Partial Pipeline.
Project Explorer	Displays the Project Explorer to the left of the work area.
Messages	Displays the Messages pane below the work area.
Toolbars	Displays the standard toolbar.
Status Bar	Displays the status bar at the bottom of the screen.

The View Menu

Access Standard Objects, Search Configuration, Dynamic Business Rules, Pipeline Diagram, Project Explorer, Messages, Toolbars, and the Status Bar.

Menu Item	Description
Standard Objects	Allows you to drill down to display Properties, Dimensions, Dimension Groups, Precedence Rules, Search Interfaces, and User Profiles.
Search Configuration	Allows you to drill down to display the Thesaurus and Stop Words.
Dynamic Business Rules	Allows you to drill down to display Rules, Zones, and Styles.
Pipeline Diagram	Lets you drill down to the main (Baseline) Pipeline and the Partial Pipeline.
Project Explorer	Displays the Project Explorer to the left of the work area.
Messages	Displays the Messages pane below the work area.
Toolbars	Displays the standard toolbar.
Status Bar	Displays the status bar at the bottom of the screen.

Resizing columns in a view

There are two ways to resize a column in a view.

To resize a column in a view:

- To resize a column in a view, click the boundary separating the column header from the one that follows it and drag the column boundary to a new width.
- To automatically resize a column according to its contents, double-click on the boundary separating the column header from the one that follows it.
  The column will automatically resize to the size of its data (not to the size of the text in its header).

Closing a project

When you are finished working on a project, you should close it before exiting Developer Studio.

To close a Developer Studio project:

- Choose **File** > **Close Project**.

Note: Only one project can be open at a time.

Exiting Endeca Developer Studio

When you are done working with a Developer Studio project, you should save your work and exit.

To exit Endeca Developer Studio:

From the **File** menu, select **Exit**.
If you have any unsaved work, Endeca Developer Studio will prompt you to save it.

Validating your project

Developer Studio performs semantic validation on your project.

If there are no errors, the Messages pane give you a confirmation message. If there are errors, they are listed on the appropriate tab.

Developer Studio automatically attempts to validate your project when you open and save it. In addition, you can validate your project at any time by choosing **Tools** > **Validate Project**.

About the Messages pane

The Messages pane opens automatically when you validate your project. It lists any errors or warnings that it encounters during validation.

There are four different sections of the Messages pane:

Tab	Description
All	Lists all messages that appear on the other tabs in the Messages pane.
Syntax	Displays syntax errors.
Validation	Displays validation errors.

Clearing messages

You can clear all the messages in the Messages pane with a single command.

To clear the messages that appear in the Messages pane:

On the **Edit** menu, select **Clear Messages**.

Closing an editor

When you are finished working with an editor, you can close it.

To close an editor:

- Click Cancel.

Closing an active view in a project

When you are finished with an active view, you can close it.

To close the active view in a project:

- Click the X in the top right corner.

Closing all open views in a project

When you are finished with all open views, you can close the group with a single command.

To close all open views in a project:

- From the Window menu, choose **Close All**.

Copying messages to the clipboard

You can copy messages from Developer Studio to external text files using the clipboard.

To copy Developer Studio messages:

1. Choose **Edit** > **Copy Messages** .
   All messages in the current tab in the Messages pane are copied to the clipboard.
2. In the external application (for example, Notepad), choose **Paste**.

The menu bar and toolbar

The menu bar allows you to open, close, and save your project, adjust the project view, and access help.

Depending upon which editor is active, you may have additional options.

The toolbar provides graphical access to many of the menu items listed above.

The Project Explorer

You control your project through the Project Explorer. The Project Explorer has two tabs: Project and Pipeline.

The Project tab contains the following selections:

- Properties launches the Property view, where you can view, add, or modify properties.
- Dimensions launches the Dimension view, where you can view, add, or modify dimensions and dimension values.
- Dimension Groups launches the Dimension Group view, where you can view, add, or modify dimension groups.
- Precedence Rules launches the Precedence Rules editor, where you can view, add, or modify precedence rules.
- Search Interfaces launches the Search Interface editor, where you can view, add, or modify search interfaces and establish ranking strategies.
- User Profiles launches the User Profiles view, where you can view, add, or modify user profiles.
- Search Configuration allows you to establish and modify settings for the thesaurus, stop words and stemming.
- Keyword Redirects allows you to create or remove keyword redirect groups.
- Dynamic Business Rules allows you establish and modify rules, zones, styles, and rule groups used by dynamic business rules.
- Pipeline Diagram launches the Pipeline Diagram editor, which allows you to interact with your pipeline visually.

The Pipeline tab allows you to access the following pipeline components if they exist in your project:

- Dimension Adapters
- Dimension Servers
- Record Adapters
- Record Manipulators
- Spiders
- Record Assemblers

- Record Caches
- Indexer Adapters
- Update Adapters
- Property Mappers
- Perl Manipulators

The Messages window

The tabs on the Messages window provide warning and error information about different aspects of your application.

If there are problems in your project, the Messages window opens automatically.

The status bar

View information including recent changes to your project.

The status bar displays information about the currently-selected menu item, the project modified indicator, and the state of the NumLock, CapsLock, and Insert keys.

The work area

The work area contains all views, editors, and other elements that you can open in Developer Studio.

Although multiple views can be open at once, only one can be active.

Chapter 3

Preparing Data, Properties, and Dimensions

Preparing Your Source Data

This section provides instructions for importing and joining source data, managing source properties, and preparing source data for indexing.

Importing Source Data

Adding a record adapter to load data

Record adapters read and write record data. A record adapter describes where the data is located (or will be saved to), the format, and various aspects of processing.

Forge can read source data from a variety of file formats and source systems. Each data source needs a corresponding input record adapter describing the particulars of that source. Based on this information, Forge parses the data and turns it into Endeca records. Input record adapters automatically decompress source data that is compressed in the gzip format.

Note: Output record adapters are generally used for diagnostic purposes. Hence, this section focuses on input record adapters. See Writing out record data for more information on output record adapters.

To add an input record adapter to your pipeline:

1. In the Pipeline Diagram editor, choose **New** > **Record** > **Adapter**.
   The Record Adapter editor appears.
2. In the **Name** text box, type a unique name for this record adapter.
3. In the General tab, do the following:
   a) In the Direction frame, choose **Input**.
   b) In the Format list, choose one of the following: XML, binary, fixed-width, delimited, vertical, document, JDBC adapter, Exchange, ODBC (Windows only), or custom adapter (available only by request from Oracle).
   c) In the **URL** text box, type the location of the source data.
   d) In the Delimiters frame, if the format is delimited, add row and column delimiters. If the format is vertical, add row, column, and record delimiters.
   e) (Optional) In the **Encoding** text box, define the encoding of the input data. If Encoding is not set, it is assumed to be Latin-1.

Note: This setting is ignored by the XML format, because the encoding is specified in the XML header. It is also ignored for binary format because Forge detects the binary format's encoding automatically. The Document format also ignores the Encoding setting.

f) If any of the text boxes in the Java properties frame are made available by your format selection, type in the required information.
g) (Optional) Check **Require Data** if you want Forge to exit with an error if the URL does not exist or is empty.
h) (Optional) Check **Filter empty properties**. Keep in mind that this attribute applies only to input record adapters and is valid only for the Vertical, Delimited, Fixed-width, and ODBC input formats.

Note: If it is not checked, by default the adapter assigns the property a value of " " (an empty string) if a record has no value for a given property.

i) (Optional) Check **Multi File** if Forge can read data from more than one input file.

Note: The URL will be treated as a wildcard and all matching files will be read in alphabetical order.

j) Check **Maintain State** if you are using the Endeca Application Controller (EAC) environment.

Note: This setting specifies that the records are output in the directory structure the EAC requires.

k) (Optional) Check **Custom Compression Level** if your input file is compressed to indicate to Forge that it must decompress data from this source.

Note: The compression level setting is ignored for an input record adapter. Compression of input files is detected automatically.

4. Ignore the Sources tab. Its settings are not used by an input record adapter.
5. (Optional) In the Record Index tab, do the following:
   a) Specify which properties or dimensions you want to use as the record index for this component.
   b) Indicate whether you want to discard records with duplicate keys.

   Note: Developer Studio performs a case-insensitive search for duplicate keys.

6. If you are using XSLT to transform your XML into Endeca-compatible XML, in the Transformer tab, specify the type (XSLT) and the location of the stylesheet.
7. If your format is ODBC, fixed-width, delimited, JDBC, custom, or Exchange, in the Pass Through tab, enter the necessary information.
8. (Optional) In the Comment tab, add a comment for the component.
9. Click **OK**.

Record Adapter editor

The Record Adapter editor contains a unique name for this record adapter.

The Record Adapter editor contains the following tabs:

General

The General tab contains the following options:

Option	Description
Direction	**Input Adapter** Required. Set to input. **Output Adapter** Required. Set to Output.
Format	**Input Adapter** Required. The format type of the raw data to be loaded. One of the following: delimited, XML, binary, fixed width, document, ODBC (Windows only), vertical, JDBC Adapter, Exchange, or Custom Adapter. Your record format affects what delimiter options, if any, are necessary. **Note:** The custom adapter option is only available by request from Oracle. **Output Adapter** Required. Can be set to delimited, XML, binary, fixed width, or vertical.
URL	**Input Adapter** Required for delimited, XML, binary, fixed-width, and vertical input adapters. Location of the file being loaded. The path can be either an absolute path, or a path relative to the Pipeline.epx file. With an absolute path, the protocol can be specified in RFC 2396 syntax. Usually this means the file:/// prefix precedes the path to the data file. Relative paths should not specify the protocol. Any paths that are part of this URL will be overridden if the Forge --inputDir option is specified. **Note:** Exchange input adapters also require a URL but the URL is specified in a pass through element using the Pass Throughs tab. **Output Adapter** Required. Location to which the data will be saved, using the same path caveats as input adapters.
Row, column, and record delimiters	**Input Adapter** Optional. Used by input adapters only if the data is in delimited or vertical format. Row and Column are used

Option	Description
	for Delimited and Vertical formats. Record is used for Vertical formats. **Output Adapter** Not used.
Java properties	**Input Adapter** Required as follows: • Java home Used by JDBC, Exchange, and Custom adapters. Specifies the location of the Java runtime engine (JRE). • Class Used by Custom adapters. Specifies the name of the adapter class to load within the .jar file indicated in **Classpath**. • Class path Used by JDBC and Custom adapters. For Custom adapters, this setting specifies a path to a .jar file (or a set of .jar files, separated by colons ":" on UNIX and semicolons ";" on Windows) containing the classes required by the Custom adapter. For JDBC, this attribute specifies the location of the .jar file containing the JDBC driver. **Note:** When running your pipeline through Forge, you can override the Java home and Class path settings using command-line options. See Overriding Java home and class path settings. **Output Adapter** Not used.
Encoding	**Input Adapter** Optional. Defines the encoding of the input data. Several hundred encodings are supported; the following are typical examples. • ISO8859-1 (Latin-1) • ISO8859-15 (Latin-9) • CP1252 (WINDOWS-1252) • ASCII • UTF-8

Option	Description
	If Encoding is not set, it is assumed to be Latin-1. If an incorrect encoding is specified, then Forge will generate warnings about any characters that do not make sense in the specified encoding. For example, in the ASCII encoding, any character with a number above 127 is considered invalid. **Note:** This setting is ignored by the XML format, because the encoding is specified in the XML header, and by Output record adapters. It is also ignored for binary format encoding only applies to text files. **Output Adapter** Required. Set to UTF-8.
Require data	**Input Adapter** Optional. If checked, Forge exits immediately with an error if the URL does not exist or is empty. The error is sent to wherever logging is configured to send errors, typically to the console or stderr. **Output Adapter** Not used.
Filter empty properties	**Input Adapter** Optional. Determines whether source properties with empty property values are assigned to Endeca Records: • If unchecked, the record adapter does not filter empty properties. In this case, the record adapter assigns an empty string (" ") to the current record for any empty properties. • If checked, properties with empty property values are filtered, or ignored. This attribute is is valid only for the Vertical, Delimited, Fixed-width, and ODBC input formats. For a filtering example, see Filtering empty properties. **Output Adapter** Not used.
Multi file	**Input Adapter** Optional. Specifies whether Forge can read data from more than one input file. If checked, the input URL is interpreted as a pattern, and Forge reads each file

Option	Description
	matching the pattern in alphabetical order. For example, the record adapter may specify a URL pattern of "*.update.txt", in which case Forge reads any file in the given directory that has the .update.txt suffix **Output Adapter** Not used.
Maintain state	**Input Adapter** Not used. **Output Adapter** Optional. If checked, indicates that the value of URL is relative to the Forge flag --stateDir. (This allows you to change your state directory using the --stateDir flag and yet not require you to modify your record adapter configuration.
Compression level	**Input Adapter** Not used. Compression of input files is detected automatically. **Output Adapter** Sets the level of compression to be performed on the record data when its written to disk. To save on the amount of disk space used, check **Custom compression level** and slide the bar to the recommended value of 7. **Note:** Compressed data consumes less disk space but takes longer to read and write.

Sources

The Sources tab contains the following options:

Option	Description
Record source	A choice of the record servers in the project. Used for output record adapters only.
Dimension source	A choice of the dimension adapters and dimension servers in the project. Generally used for output record adapters only. Input record adapters only require a dimension source if they implement a record index that includes dimensions.

Record Index

Optional. The Record Index tab allows you to add or remove dimensions or properties used in a component's record index, and to change their order. Record indexes support join functionality. See Join sources must have matching join keys and record indexes for more details.

The Record Index tab contains the following fields:

Field	Description
Discard records with duplicate keys	When checked, Forge discards any records with duplicate keys and logs a warning that specifies the number of records discarded. **Note:** Developer Studio performs a case-insensitive search for duplicate keys.

Transformer

The Transformer tab is for the XML format. XML adapters assume that data is in the Endeca Record XML format and, without transformation, other XML formats cannot be read by the Data Foundry. To support these situations, an XSLT transformation can be applied to the source data to convert it into Endeca Records XML, which the Data Foundry can read.

The Transformer tab has the following options:

Option	Description
Type	Must be XSLT.
URL	Location of the stylesheet to use.

Pass Throughs

The Pass Throughs tab is used with certain formats to pass additional information to Forge. It contains text boxes where you can add, modify, or delete key/value pairs. Pass throughs are required for ODBC, fixed-width, delimited, JDBC, custom, or Exchange adapters.

Comment

Optional. Provides a way to associate comments with a pipeline component.

Giving Forge permission to access source data

The account under which Forge runs must have appropriate permissions to access source data.

If Forge does not have the appropriate permissions, one of two things will happen when a record adapter requests data from a source:

- If **Require Data**, which is located on the record adapter's General tab, is checked, then Forge will exit immediately with an error. The error is sent to wherever logging is configured to send errors, typically to the console or stderr.
- If **Require Data** is not checked, Forge will continue with the next task in the pipeline without reading any data from the offending source. Forge will not exit with a failure code.

Input formats

Forge can read source data from a variety of file formats.

Delimited

The Delimited format reads source records that are organized into rows and columns.

Each row is separated from other rows by a row delimiter character, such as the new-line character, and each column is separated from other columns by a column delimiter character, such as a comma or the tab character. The row and column delimiters must not be present within the data itself. For example, if the column delimiter is a comma, no data in a column can contain a comma.

When the source records are read into the Data Foundry, two mappings occur:

- The column names are mapped to Endeca properties. The record adapter assumes that the first row of the file is a header row, which specifies column names separated by the column delimiter. If the first row of the data is not a header row, the column names must be specified on the Pass Throughs tab of the Record Adapter editor.
- The rest of the rows are assumed to contain data and are mapped to Endeca records. These additional rows must have the same number of column delimiters.

Properties are trimmed as they are read in. White space on the ends of properties (including the space, tab, new-line, and other characters) is removed. However, white space within a property is preserved.

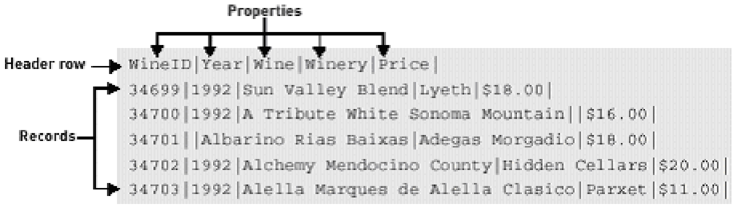
The records in a delimited file must have identical properties, in terms of number and type, although it is possible for a record to have a null value for a property. You can use the "Filter empty properties" checkbox, on the Record Adapter editor's General tab, to tell the record adapter to ignore properties that have null values.

Note:

- When picking delimiters, you should be aware of the content of the data. Using a delimiter that appears in the content of the data itself can cause problems.
- For the Delimited format, Forge has a built-in hard limit of 64K per record. While this setting cannot be configured, there are workarounds.

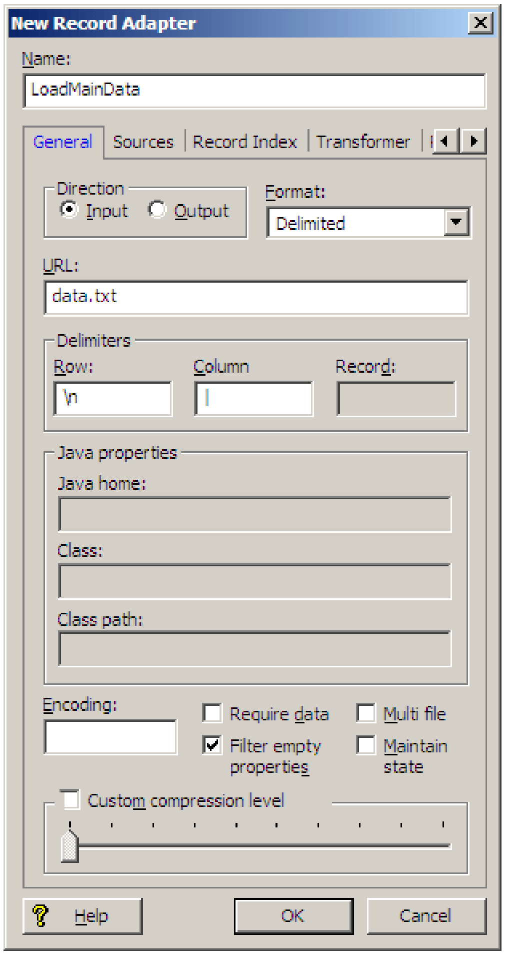
Delimited example

This Delimited format illustration uses the pipe ('|') character as the column delimiter. The column names in the header row are mapped to Endeca properties.



```
WineID|Year|Wine|Winery|Price|
34699|1992|Sun Valley Blend|Lyeth|$18.00|
34700|1992|A Tribute White Sonoma Mountain||$16.00|
34701||Albarino Rias Baixas|Adegas Morgadio|$18.00|
34702|1992|Alchemy Mendocino County|Hidden Cellars|$20.00|
34703|1992|Alella Marques de Alella Clasico|Parxet|$11.00|
```

The record adapter for this delimited file looks like this:



Name: LoadMainData

Direction: ● Input ○ Output

Format: Delimited

URL: data.txt

Delimiters

Row: \n

Column: |

Record:

Java home:

Class:

Class path:

Encoding:

□ Require data

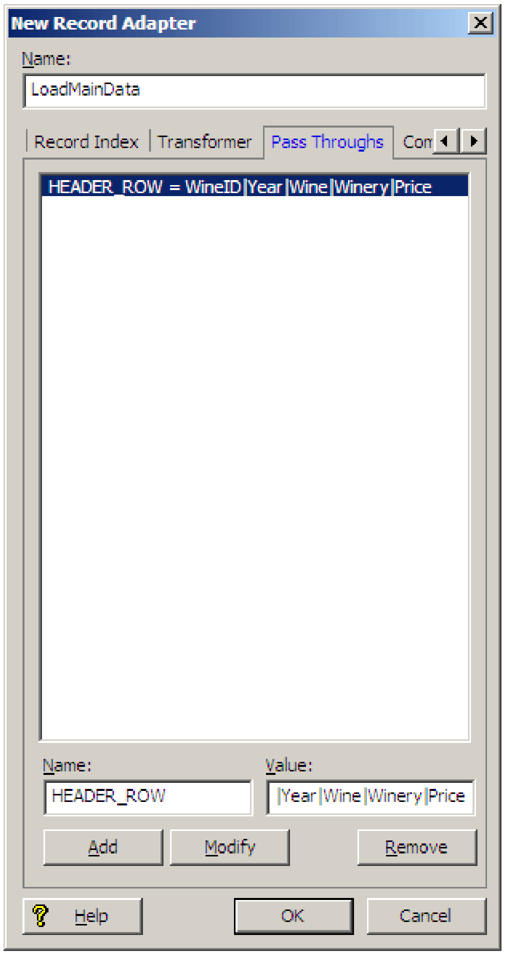
□ Multi file

☑ Filter empty properties

□ Maintain state

□ Custom compression level

If the first row of the data is not the header row, you must use the Record Adapter editor's Pass Throughs tab to specify the column names, as in this example:



The Name field must be HEADER_ROW and the Value field must contain the column names separated by the column delimiter.

Tips When Using Delimited Format

- If a delimited file contains new-line delimiters, the delimiter must be specified exactly or unpredictable results may occur. For example, on UNIX systems, the new line is generally a "\n", while on Windows systems, the new line is a "\r\n" combination.
- If a delimited file contains non-standard delimiters, the file may look different than expected. For example, if a delimited format file has "~" as a row delimiter, the data will be entirely on one line (unless new lines appear within the data). Because delimited files typically have one record per row, the file will appear to have a single record or to be corrupt. In fact, files in this format are valid.

Fixed width

The fixed width format reads source data records from columns of fixed widths.

Each column represents a property and has a width that is a specific number of characters. For example, the first three characters could represent an ID, characters 4 through 10 could represent a name, and so forth. Each row represents a record.

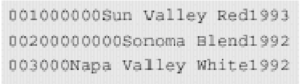
Note: The new-line character at the end of each row is treated identically to the other characters in the row. For example, if you have a row of 100 characters with a new-line character at the end, the total number of characters for the row is 101. Also, the character count is zero-based. For example, the first four characters in a row are characters 0 to 3.

The fixed width record adapter requires the following six attributes, which are specified on the Pass Throughs tab. The names of the attributes must be entered as shown:

- PAD_CHARACTER – The character used to pad columns to the appropriate width (typically, the value is the digit '0').
- PAD_SIDE – The side on which each column is padded. LEFT or RIGHT are the only valid values.
- COLUMN_NAMES – The names of each of the columns, separated by commas.
- COLUMN_TYPES – The data type used to encode each column, separated by commas. CHAR or INT are the only valid values. The INT type should only be used for data that is encoded as native-endian 4-byte integers. All other data must be in the specified character encoding and should have a column type of CHAR (CHAR refers to bytes, not unicode characters).
- COLUMN_STARTS – The byte numbers that start each column of data. The index starts at zero.
- COLUMN_ENDS – The byte numbers that end each column of data. The index starts at zero.

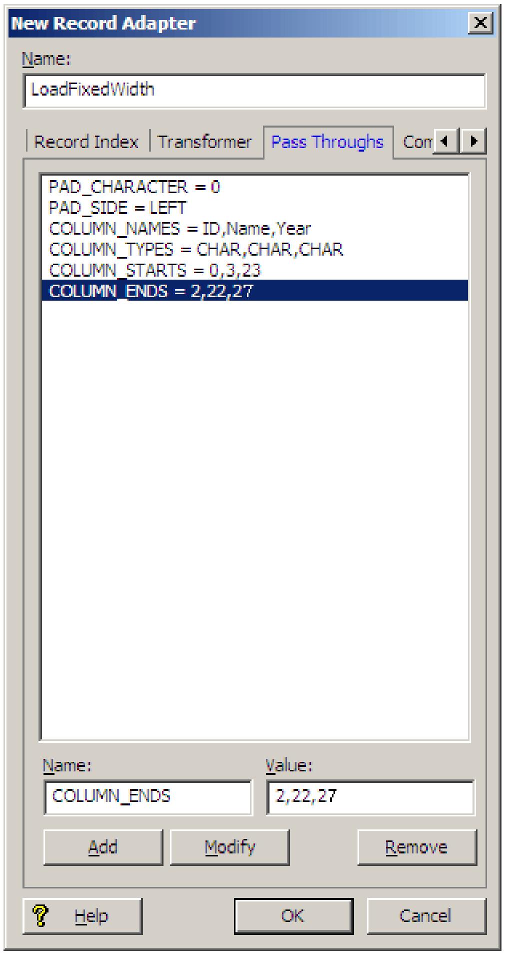
Fixed Width example

In the following fixed width example, the first three characters represent an ID, the next twenty represent a name, and the final four represent a year:



```
001000000Sun Valley Red1993
00200000000Sonoma Blend1992
003000Napa Valley White1992
```

You use the Record Adapter editor's Pass Throughs tab to specify the required fixed width attributes, as in this example:



Name: LoadFixedWidth

```
PAD_CHARACTER = 0
PAD_SIDE = LEFT
COLUMN_NAMES = ID,Name,Year
COLUMN_TYPES = CHAR,CHAR,CHAR
COLUMN_STARTS = 0,3,23
COLUMN_ENDS = 2,22,27
```

Name: COLUMN_ENDS

Value: 2,22,27

Vertical

The vertical format reads source records stored as property name/value pairs.

Vertical format requires delimiters specifying how to identify each property name, property value, and record. These delimiters are defined in the General tab of the Record Adapter editor:

- Column: The delimiter between the name and value, typically an equal sign, comma, or tab.
- Row: The delimiter between adjacent name/value pairs. Typically the row delimiter is a new-line character (causing the format to appear vertical). The row delimiter defaults to a new-line character if omitted.
- Record: The delimiter between adjacent records. Typically it is the text REC or EOR. The record delimiter must be surrounded by Row delimiters in the source data (it identifies a special name/value pair that does not have the Column delimiter).

All name/value pairs leading up to a record delimiter are considered part of a single record. The properties for the records in a vertical file format can be of a variable number and type.

Properties are trimmed as they are read in. White space (such as the space, tab, and new-line characters) is removed from both ends of properties, but white space within a property is preserved.

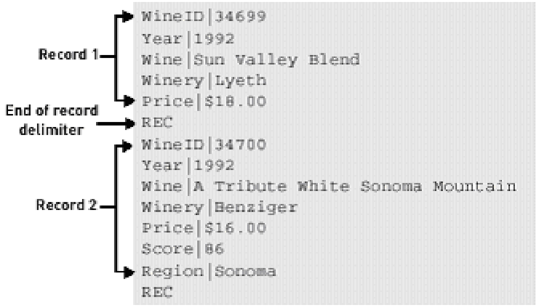
Note:

- When picking delimiters, you should be aware of the content of the data. Using a delimiter that appears in the content of the data itself can cause problems.

- For the Vertical format, Forge has a built-in hard limit of 64K per record. While this setting cannot be configured, there are workarounds.

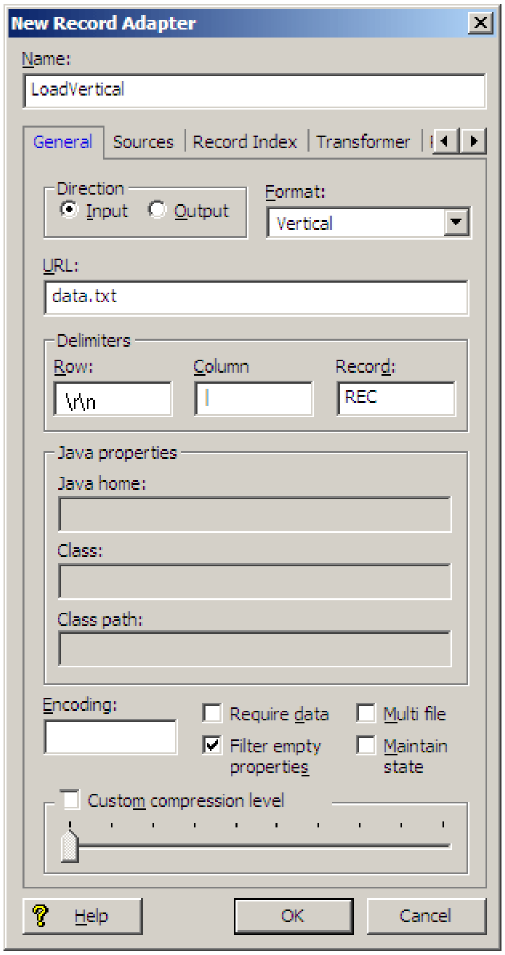
Vertical example

This Vertical format illustration uses the pipe character as the Column delimiter and 'REC' as the Record delimiter. The Property names (such as "WineID") are mapped to Endeca Properties. Note that the second record (WineID 347000) has seven properties while the first only has five:



```
WineID|34699
Year|1992
Wine|Sun Valley Blend
Winery|Lyeth
Price|$18.00
REC
WineID|34700
Year|1992
Wine|A Tribute White Sonoma Mountain
Winery|Benziger
Price|$16.00
Score|86
Region|Sonoma
REC
```

You would define the record adapter for this Vertical format in Developer Studio as follows:



Name: LoadVertical

Direction: ● Input ○ Output

Format: Vertical

URL: data.txt

Delimiters
Row: \r\n
Column: |
Record: REC

Java home:

Class:

Class path:

Encoding:

□ Require data
□ Multi file
☑ Filter empty properties
□ Maintain state

□ Custom compression level

Tips For Using Vertical Format

- If a vertical file contains new-line delimiters, the delimiter must be specified exactly or unpredictable results may occur. For example, on UNIX systems, the new line is generally a "\n", while on Windows systems, the new line is a "\r\n" combination.
- In vertical format, the record delimiter must be surrounded by row delimiters (which are usually new lines).
- If a vertical file contains non-standard delimiters, the file may look different than expected. For example, if a vertical file has "~" as a row delimiter, instead of a new-line delimiter, the data for each record will be entirely on one line (unless new lines appear within the data itself). Because vertical files typically have one property per row, the file will appear to be corrupt. In fact, files in this format are valid.

ODBC
The ODBC format enables the Endeca Information Transformation Layer (ITL) to connect directly to and read records from any database that supports ODBC connections.

Pass Throughs tab

The ODBC format is supported only on the Windows version of Oracle Commerce Guided Search. In addition to name, direction, and format, an ODBC record adapter requires settings on the Pass Throughs and General tabs. An ODBC record adapter requires that the following two attributes be specified on the Pass Throughs tab.

If clear text credentials are being used to pass Database credentials:

- Name = `DSN`

  Value = The data source name, including connection parameters such as username and password.

- Name = `SQL`

  Value = The SQL query to execute on the ODBC data source. Note that stored procedures can be executed; however, the stored procedures must return tables (i.e., stored procedures that return cursors are not supported).

If Oracle Credentials Store is being used to pass Database credentials:

- Name = DSN

  Value = The data source name, without any username and password.

- Name = SQL

  Value = The SQLquery toexecuteonthe ODBC datasource. Note that stored procedures can be executed; however, the stored procedures must return tables (that is, stored procedures that return cursors are not supported).

- Name = CREDENTIALS_KEY

  Value = The key name required to access the credentials information from Oracle Credentials Store

Note: For more information about the Oracle Credentials Store, refer to *Oracle Commerce Guide Search Security Guide*.

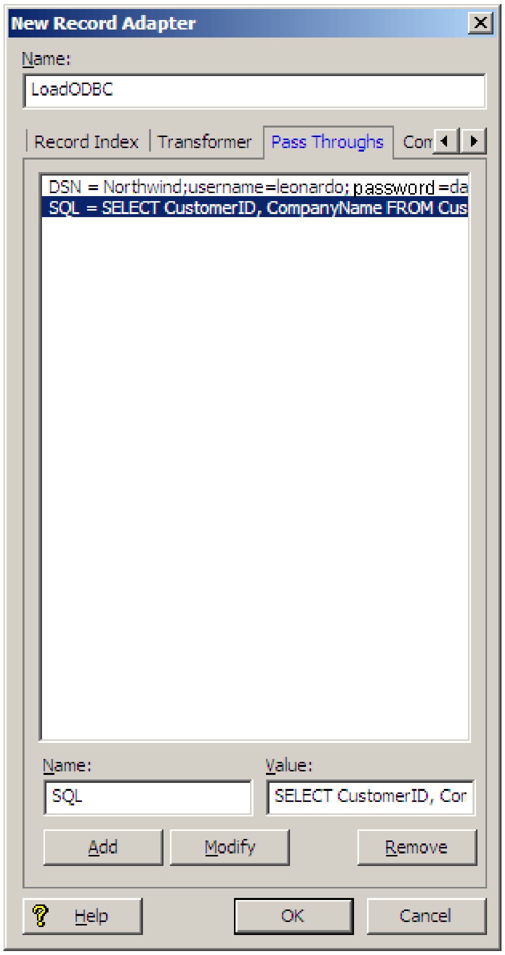
The names and types of properties are determined automatically from the results of the ODBC query.

General tab

Optionally, you may also specify the encoding of character data being returned from the ODBC data source, using the Encoding field of the General tab.

ODBC example

The following illustration shows the Pass Throughs tab of the Record Adapter editor for an ODBC data source:



New Record Adapter

Name: LoadODBC

DSN = Northwind;username=leonardo; password=da
SQL = SELECT CustomerID, CompanyName FROM Cus

Name: SQL

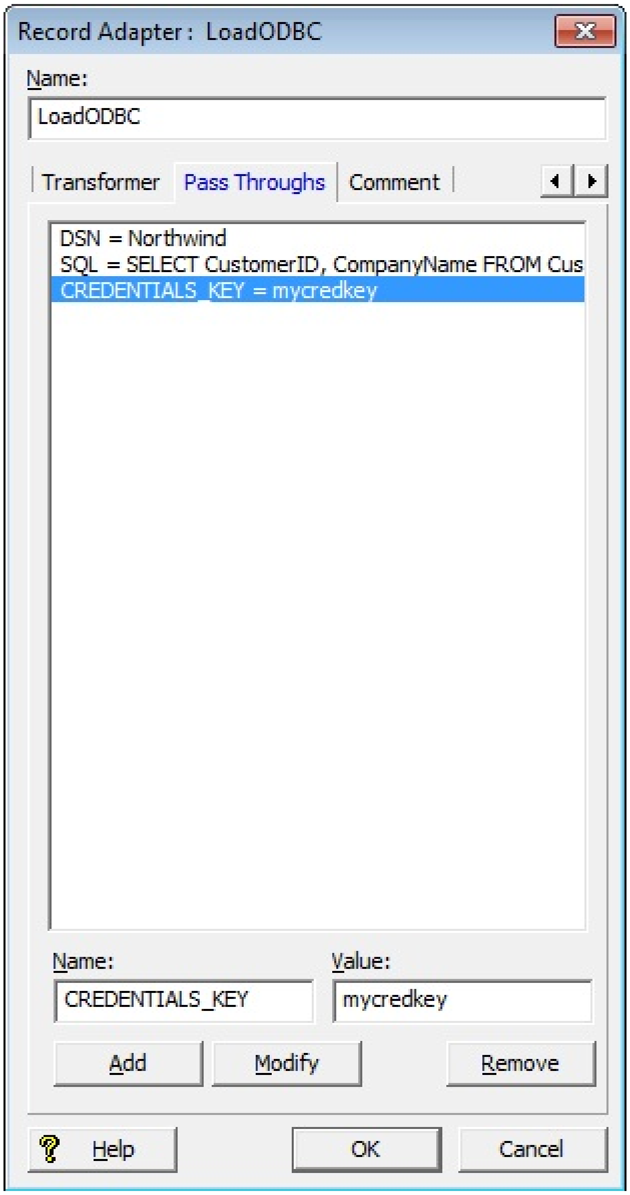
Value: SELECT CustomerID, Cor

Pass Through elements named "DSN" and "SQL" have been defined.

The DSN element is:

`Northwind;username=Leonardo;password=davinci`

For using Oracle Credentials Store:



Record Adapter : LoadODBC

Name: LoadODBC

DSN = Northwind
SQL = SELECT CustomerID, CompanyName FROM Cus
CREDENTIALS_KEY = mycredkey

Name: CREDENTIALS_KEY

Value: mycredkey

Note: Different databases use different syntax for specifying the username and password. For example, some databases use "UID" rather than "username," and "PWD" rather than "password." If "username" and "password" don't work for your data source, see the documentation for your data source to determine the correct syntax.

The SQL query is:

```
SELECT CustomerID, CompanyName FROM Customers
```

Supported data types

The following table provides information on supported data types for the ODBC record adapter:

Data type	Supported
CHAR	Up to 1M bytes. Note that if a field in the result set is larger than 1MB (1,048,576 bytes), then the result is truncated at that length.
VARCHAR	Up to 1M bytes
LONGVARCHAR (CLOB)	No
WCHAR	Up to 1M bytes
VARWCHAR	Up to 1M bytes
LONGWVARCHAR	No
DECIMAL	Yes
NUMERIC	Yes
SMALLINT	Yes
INTEGER	Yes
REAL	Yes
FLOAT	Yes
DOUBLE	Yes
BIT	Yes
TINYINT	Yes
BIGINT	No
BINARY	No
VARBINARY	No
LONGVARBINARY (BLOB)	No
DATE	Yes
TIME	Yes
TIMESTAMP	Yes
UTCDATETIME	No
UTCTIME	No
INTERVAL MONTH	No
INTERVAL YEAR	No
INTERVAL YEAR TO MONTH	No
INTERVAL DAY	No
INTERVAL HOUR	No
INTERVAL MINUTE	No
INTERVAL SECOND	No
INTERVAL DAY TO HOUR	No

Data type	Supported
INTERVAL DAY TO MINUTE	No
INTERVAL DAY TO SECOND	No
INTERVAL HOUR TO MINUTE	No
INTERVAL HOUR TO SECOND	No
INTERVAL MINUTE TO SECOND	No
GUID	Yes

JDBC

The JDBC format enables the Endeca Information Transformation Layer (ITL) to connect to and read records from any JDBC data source.

In addition to name, direction, and format, a JDBC record adapter requires settings on the General and Pass Throughs tabs.

General tab

- Java home - The location of Java 2 SDK.
- Class path - The location of the .jar file containing the JDBC driver. If the JDBC driver you are using is distributed with the JVM, you can omit this setting (for example, the jdbc-odbc bridge does not require a classpath).

  Note: When running your pipeline through Forge, you can override the Java home and Class path settings using command-line options. See Overriding Java home and class path settings.

Pass Throughs tab

- Name = `DB_DRIVER_CLASS`

  Value = The name of the JDBC driver class to use.
- Name = `DB_URL`

  Value = The URL of the data source, in standard JDBC format.
- Name = `SQL`

  Value = The SQL query to run to extract the data. Note that stored procedures can be executed; however, the stored procedures must return tables (i.e., stored procedures that return cursors are not supported).

In addition, if the connection requires properties (such as a password or username), then the following attribute can also be specified, as many times as necessary, on the Pass Throughs tab:

- Name = `DB_CONNECT_PROP`

  Value = The name and value, separated by an equal sign (=), of a property to pass to the JDBC connection.

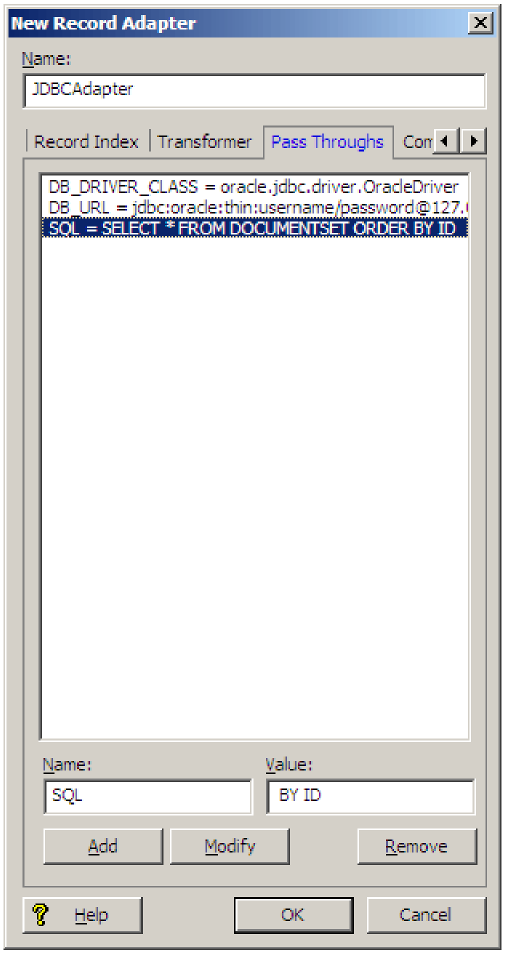
Note that configuring user name and password parameters varies according to your JDBC driver. For example, with Oracle JDBC Thin drivers, the user name and password parameters are included as part of the DB_URL string rather than as separate DB_CONNECT_PROP values. You may have to refer to the documentation for your JDBC driver type to determine exact configuration requirements.

Instead of specifying clear text credentials, you can use Oracle Credentials Store to specify the database credentials information. In which case instead of specifying the username and password information along with DB_URL or DB_CONNECT_PROP, you need to use the passthrough CREDENTIALS_KEY and provide the key name that should be used to retrieve the credentials from Oracle Credentials Store.

• Name = CREDENTIALS_KEY Value = The key name required to access the credentials information from Oracle Credentials Store

Note: For more information on Oracle Credentials Store, please refer to *Oracle Commerce Guided Search Security Guide*.

The following illustration shows the Pass Throughs tab for a record adapter that is configured to access a JDBC data source through an Oracle JDBC Thin driver using clear text credentials:



Name: JDBCAdapter

```
DB_DRIVER_CLASS = oracle.jdbc.driver.OracleDriver
DB_URL = jdbc:oracle:thin:username/password@127.
SQL = SELECT * FROM DOCUMENTSET ORDER BY ID
```

Name: SQL

Value: BY ID

The following illustration shows the Pass Throughs tab for a record adapter that is configured to access a JDBC data source via an Oracle JDBC Thin driver using Oracle Credentials Store:

Supported data types

The following table provides information on supported data types for the JDBC record adapter:

Data type	Supported
CHAR	Up to 1M bytes. Note that if a field in the result set is larger than 1MB (1,048,576 bytes), then the result is truncated at that length.
VARCHAR	Up to 1M bytes
LONGVARCHAR (CLOB)	No
WCHAR	Up to 1M bytes
VARWCHAR	Up to 1M bytes
LONGWVARCHAR	No
DECIMAL	Yes
NUMERIC	Yes
SMALLINT	Yes
INTEGER	Yes
REAL	Yes
FLOAT	Yes
DOUBLE	Yes
BIT	Yes
TINYINT	Yes
BIGINT	No
BINARY	No
VARBINARY	No
LONGVARBINARY (BLOB)	No
DATE	Yes
TIME	Yes
TIMESTAMP	Yes
UTCDATETIME	No
UTCTIME	No
INTERVAL MONTH	No
INTERVAL YEAR	No
INTERVAL YEAR TO MONTH	No
INTERVAL DAY	No
INTERVAL HOUR	No
INTERVAL MINUTE	No
INTERVAL SECOND	No
INTERVAL DAY TO HOUR	No

Data type	Supported
INTERVAL DAY TO MINUTE	No
INTERVAL DAY TO SECOND	No
INTERVAL HOUR TO MINUTE	No
INTERVAL HOUR TO SECOND	No
INTERVAL MINUTE TO SECOND	No
GUID	Yes

Exchange

The Exchange format allows the Endeca Information Transformation Layer (ITL) to connect to one or more Microsoft Exchange Servers (versions 2000 and beyond) and extract information from specified public folders.

The Exchange format produces one record for each document and each sub-folder contained in the specified public folders. This includes mail messages, calendar items, and generic documents of any format.

In addition to name, direction, and format, the Exchange adapter requires the following attributes on the General and Pass Throughs tabs:

General tab

- Java home - The location of Java 2 SDK.

  Note: When running your pipeline through Forge, you can override the Java home setting using a command-line option. See Overriding Java home and class path settings.

Pass Throughs tab

- Name = URL

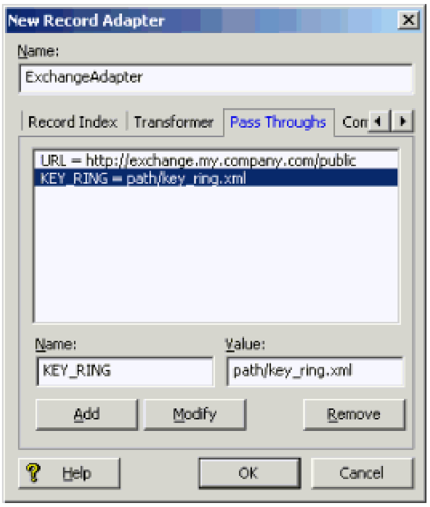
  Value = A URL to a folder on an Exchange server (for example, http://exchange.mycompany.com/public). You can create multiple URL attributes, each with its own URL value, but at least one is required.

If any of the specified Exchange servers require authentication, a reference to a key ring file containing a suitable username/password key combination must be specified. You provide this information using an additional Pass Through element with the name "KEY_RING" and a value that specifies the path to the key ring file. This path may be absolute or relative to the Pipeline.epx file. For each Exchange server that requires authentication, the key ring file should contain an element of the form:

```
<EXCHANGE_SERVER HOST="exchange.myhost.com">
<KEY>B9qtQOON6skNTFTHm9rnn04=</KEY>
</EXCHANGE_SERVER>
```

See "Implementing the Endeca Crawler" in the *Endeca Forge Guide* for more information on how to edit a key ring file and encrypt keys.

The Pass Throughs tab for an Exchange adapter looks similar to this:



Name: ExchangeAdapter

```
URL = http://exchange.my.company.com/public
KEY_RING = path/key_ring.xml
```

Name: KEY_RING

Value: path/key_ring.xml

Retrieving attachments and other documents

Mail messages extracted by the Exchange adapter may contain attachments. The attachments themselves are not retrieved, but a URL pointing to them is included as a property of the associated mail message record. This property has the name "Endeca.Exchange.Attachment" and the URL it contains may be used to retrieve the attachment document using a RETRIEVE_URL expression.

A similar situation occurs when the Exchange adapter encounters a generic document that is not of a format that the Microsoft Exchange Server recognizes internally. In this situation, the Exchange adapter produces a record for the document that contains properties containing meta-data about the document. In addition, the record will have a property named "DAV:a:href" that specifies a URL that may be used to retrieve the document using a `RETRIEVE_URL` expression.

For more information on using a `RETRIEVE_URL` expression, see "Implementing the Endeca Crawler" in the *Endeca Forge Guide*.

Document

The Document format is available only to customers who have purchased the Endeca Crawler.

For details on this format and implementing the Endeca Crawler, see "Implementing the Endeca Crawler" in the *Endeca Forge Guide*.

XML

The XML record format is proprietary to Endeca. XML records can be used to store state between Forge runs, for Advanced Crawler output, for debugging, or to propagate data from one Forge pipeline to another.

An XSL transformation can be applied to source data that is in a non-Endeca XML format, converting it into Endeca Record XML which Forge can read. The XSL transformation is done on the entire input file before Forge starts to read records, so there may be a significant delay before Forge starts processing with this format. Also, because XML is a relatively verbose text format, it is the slowest input format to Forge.

In addition to name and direction, an XML record adapter requires settings on the General and Transformer tabs.

General tab

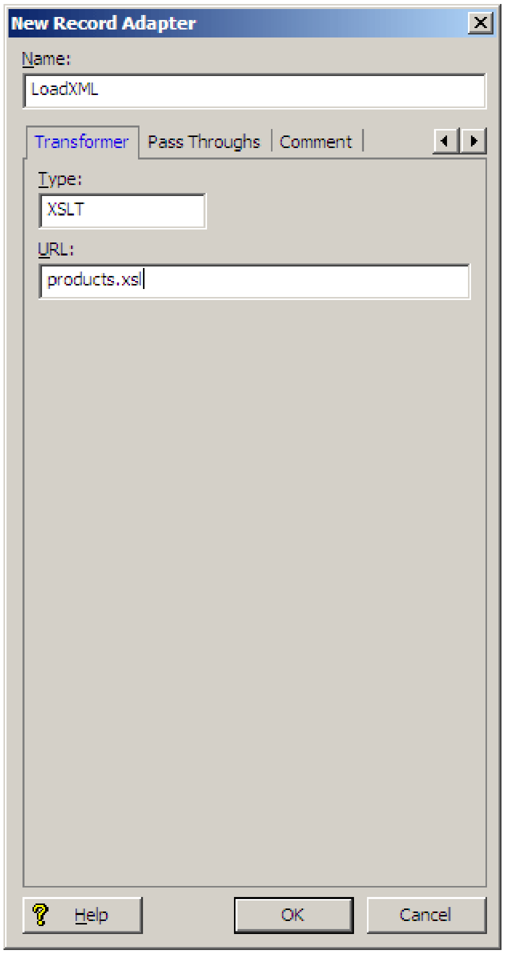
- Format: Set to XML format.
- URL: Set to the location of the input XML document.

Transformer tab

- Type: Must be XSLT.
- URL: The URL of the XSLT transformation style sheet.

XML example

The following illustration shows the Transformer tab of the Record Adapter editor.



Name: LoadXML

Type: XSLT

URL: products.xsl

The sample record adapter reads XML records from the file named products.xml (which is specified in the General tab's URL field), converting the vendor-specific XML to the Endeca Record XML format using the products.xsl stylesheet.

Binary

The binary format is used to store state between runs of the Data Foundry and also as one of the possible output formats of the Endeca Advanced Crawler.

The Binary format is proprietary to Endeca. It is faster and more compact than the XML format.

Custom

The Custom Adapter option is only available by request from Oracle.

Filtering empty properties

When checked, the Filter Empty Properties setting on the General tab of the Record Adapter editor filters out empty property values when the source data is read into the Data Foundry.

For example, assume the following source data contains properties delimited by the pipe ('|') character.

```
WineID|Region|WineType|Price

1||Merlot|8.99

2||Chardonay|14.99

3||Chianti|19.99

4|||10.99

5||Merlot|18.99

6||Spumante

7|Sonoma|Chablis|9.99
```

A record adapter with Filter Empty Properties checked would filter the Region property from records 1 through 6, and the Region and WineType properties from record 4. If Filter Empty Properties is not checked, an empty string (' ') is assigned to the empty properties.

About character limits in text extracts

For text input formats (Delimited and Vertical), Forge has a built-in hard limit of 64K per record.

This built-in limit cannot be configured; however, there are two workarounds:

- Change the format to XML.
- Put large text fields into files, and include the filename in the record. Use the IMPORT_PROP expression to read the file into the record (or leave it on disk, if all you're going to do is run PARSE or CONVERTTOTEXT on it).

About providing pass-through information

The Pass Throughs tab is used to include additional required information for several data formats.

Pass through information is in the form of name/value pairs. There are specific Pass Through details for each of the following formats:

- Delimited
- Fixed-width
- ODBC
- JDBC
- Exchange
- Custom

For detailed information about using Pass Throughs with each of the above data formats, see the section "Input Formats."

Adding pass-through information for a record adapter

The Pass Throughs tab is used to include additional required information for several data formats.

To add pass through information for your record adapter:

1. In the Record Adapter editor, click the Pass Throughs tab.
2. Enter a name and a value for the pass through element, then click Add. The pass through is added to the list at the top of the tab.

Modifying pass-through information

The Pass Throughs tab is used to include additional required information for several data formats. This information can be modified.

To modify a pass through element:

1. Select the pass through element from the list at the top of the tab. The name and value fields are populated with the information for the element you selected.
2. Make any changes necessary, then click Modify.

Removing a pass-through element from a record adapter

The Pass Throughs tab is used to include additional required information for several data formats. This information can be removed if necessary.

To remove a pass-through element from a record adapter:

- Select the pass through element from the list at the top of the tab, then click Remove.

Overriding Java home and class path settings

When running your pipeline, you can override the Java home and Class path settings specified in a record adapter using two Forge command-line flags, --javaHome and --javaClasspath.

The JDBC adapter uses both of these settings. Use a command similar to this one to override them, where j2sdk is an absolute path to your Java 2 SDK and drivers.jar is an absolute path to the drivers you want to use:

```
$ENDECA_ROOT/bin/forge -vvt --javaHome usr/local/j2sdk1.4.1_02 --javaClasspath
usr/local/Oracle/lib/oracle8i.zip pipeline.epx
```

If you specify multiple drivers, delimit them using colons (:) on UNIX and semi-colons (;) on Windows.

The Exchange adapter only uses the Java home setting. Use a command similar to this one to override this setting, where j2sdk is an absolute path to your Java 2 SDK:

```
$ENDECA_ROOT/bin/forge -vvt --javaHome usr/local/j2sdk1.4.1_02 pipeline.epx
```

Joining Source Data

Implementing a join

Generally, applications consist of more than one data source. If those sources exist in separate locations, you would need to implement a join in order to create a single record.

For example, an application used to navigate books would have records that contain both title and author information. If the title and author source data reside in different locations, you would need to join them together to create a single record with both pieces of information.

Implementing a join is typically a three-step process.

To implement a join:

1. In most cases, you add a record cache to your pipeline for each record source that will feed the join
2. Add a record assembler to your pipeline.
3. Use the record adapter that you added to configure the join.

Adding a record cache

With two exceptions, all data sources feeding a join must be record caches, so this section details the procedures from that perspective.

- Switch joins do not do record comparisons and, hence, do not require record caches for their data sources. You can use any type of record server component (record adapter, record cache, record assembler, Perl manipulator, and so on) as a source for a switch join.
- For a left join, for which all of the right sources are record caches, the left source does not require a record cache. This special case is useful for optimizing a left join with a large, unsorted data source.

To add a record cache for each record source that will feed the join:

1. In the Pipeline Diagram editor, click New, and then choose Record > Cache.
   The Record Cache editor appears.
2. In the **Name** text box, type a unique name for this record cache.
3. (Optional) In the General tab, you may do the following:
   - If the cache should load fewer than the total number of records from the record source, type the number of records to load in the **Maximum Records** text box. This feature is provided for testing purposes.
   - If you want to merge records with equivalent record index key values into a single record, check **Combine Records**. For one-to-many or many-to-many joins, leave **Combine Records** unchecked.

     Note: The Combine records option can have unexpected results if you do not understand how it functions. See Tips and troubleshooting for joins for complete details.
4. In the Sources tab, select a record source and, optionally, a dimension source.

   Note: If a component's record index contains dimension values, you must provide a dimension source. Generally, this is only the case if you are caching data that has been previously processed by Forge.
5. In the Record Index tab, do the following:
   - Specify which properties or dimensions you want to use as the record index for this component.

     Note: The record index you specify for a cache must match the join key that you will specify for that cache in the record assembler.
   - Indicate whether you want to discard records with duplicate keys.

     Note: Developer Studio performs a case-insensitive search for duplicate keys.
6. (Optional) In the Comment tab, add a comment for the component.
7. Click **OK**.
8. Repeat these steps for all record sources that will participate in the join.

Record Cache editor

The Record Cache editor contains the unique name for this record cache.

The Record Cache editor contains the following tabs:

General

Option	Description
Maximum records	Optional. Record caches can be set to load a limited number of records. The Max records field specifies that limit. By default it is set to -1, which means the cache will load all records. The **Max records** field only has to be provided if the cache should load fewer than the total number of records.
Combine records	Optional. When checked, Forge merges records with the same key values into a single record. Do not check **Combine records** if you are performing one-to-many or many-to-many joins.

Sources

The Sources tab contains the following fields:

Field	Description
Record sources	Required . A choice of the record server components in the project.
Dimension source	If the record cache's record index uses dimensions, you must provide a dimension source.

Record Index

The Record Index tab allows you to add or remove dimensions or properties used in a component's record index, and to change their order. Record indexes support join functionality. See Join sources must have matching join keys and record indexes for more details.

The Record Index tab contains the following fields:

Field	Description
Discard records with duplicate keys	When checked, Forge discards any records with duplicate keys and logs a warning that specifies the number of records discarded. **Note:** Developer Studio performs a case-insensitive search for duplicate keys.

Comment

Optional. Provides a way to associate comments with a pipeline component.

Adding a record assembler

A record assembler is a pipeline component used to join source records originating from different files.

To add a record assembler to your pipeline:

1. In the Pipeline Diagram editor, click **New**, and then choose **Record** > **Assembler**.
   The Record Assembler editor appears.
2. In the **Name** text box, type a unique name for the new record assembler.
3. In the Sources tab, do the following:
   a) In the Record Sources list, select a record source.
   b) Click **Add**.

   Note: Repeat as necessary to add additional record sources. With two exceptions, record assemblers must use record caches as their source of record data.

   c) In the Dimension Source list, select a dimension source.

   Note: If the key on which a join is performed contains dimension values, you must provide a dimension source. Generally, this is only the case if you are joining data that has already been processed once by Forge.

4. (Optional) In the Record Index tab, do the following:

   Note: An assembler's record index does not affect the join; it only affects the order in which downstream components will retrieve records from the assembler.

   a) Specify which properties or dimensions you want to use as the record index for this component.
   b) Indicate whether you want to discard records with duplicate keys.

   Note: Developer Studio performs a case-insensitive search for duplicate keys.

5. In the Record Join tab, configure the joins.
6. (Optional) In the Comment tab, add a comment for the component.
7. Click **OK**.

See "Configuring the join in the record assembler" for details on this procedure.

Record Assembler editor

The Record Assembler editor contains a unique name for this record assembler.

The Record Assembler contains the following tabs:

Sources

The Sources tab contains the following fields:

Field	Description
Record sources	Required. A choice of the record server components in the project. A record assembler normally contains multiple record sources.
Dimension source	If the key on which a join is performed contains dimension values, you must provide a dimension source. Generally, this is only the case if you are joining data that has already been processed once by Forge.

Record Index

(Optional) The Record Index tab allows you to add or remove dimensions or properties used in a component's record index, and to change their order. Record indexes support join functionality. See Join sources must have matching join keys and record indexes for more details.

The Record Index tab contains the following fields:

Field	Description
Discard records with duplicate keys	When checked, Forge discards any records with duplicate keys and logs a warning that specifies the number of records discarded. **Note:** Developer Studio performs a case-insensitive search for duplicate keys.

Record Join

The Record Join tab contains the following fields:

Field	Description
Join type	A drop-down list of the available join types.
Multi sub-records	If you are performing a left join, check the Multi Sub-records option if the left record can be joined to more than one right record.
Remove duplicate property values	If checked, duplicate property values (same name, value and children) are not added when records are combined. The property values of the new record are unique.
Join entries	List of the record sources to be joined.

Comment

(Optional) Provides a way to associate comments with a pipeline component.

Configuring the join in the record assembler

Applications consisting of more than one data source require a join to combine the separate data sources. These joins must be configured in a record assembler.

To configure the join in the record assembler:

1. In the Record Assembler editor, click the **Record Join** tab.
2. In the Join Type list, select the kind of join you want to perform:
   - Left
   - Inner
   - Combine
   - Outer
   - Switch
   - Sort switch
   - Disjunct
   - First record
3. If you are performing a left join, check **Multi Sub-records** if the left record can be joined to more than one right record.
4. The join entries list represents the record sources that will participate in the join, as specified on the Sources tab. In the Join Entries list, define the order of your join entries by selecting an entry and clicking Up or Down.

   Note: For all joins, properties get processed from join sources in the order in they are in the list. The first entry is the Left entry for a left join. See Join keys that use multiple properties or dimensions for more details.

5. To define the join key for a join entry, select the entry from the Join Entries list and click Edit.
   The Join Entry editor appears.
6. Click **Add**.
   The Key Component editor appears.
7. Using the steps below, create a join key that is identical to the record index key for the join entry you selected.
   a) In the Type frame, choose Custom Property or Dimension.

      If you choose Custom Property, type a name for the property in the Custom Property text box. If you choose Dimension, choose a dimension name from the Dimension list.

   b) Click OK to return to the Join Entry editor.
   c) (Optional) Repeat these steps for each component you want to add to the key.
   d) (Optional) To reorder the components in the key, select a component in the Join Entry editor and click **Up** or **Down**.
   e) Click **OK** to close the Join Entry editor.
8. Repeat steps 5 through 7 for each record source that is participating in the join.
9. When you are done configuring your join, click **OK** to close the Record Assembler editor.

Adding and Editing Source Properties

After source data has been loaded into your implementation, you may need to perform some manipulations.

About adding and editing source properties

The goal of this step is to create source records with the correct set of source properties, before any mapping occurs.

You can use a Perl manipulator component in your pipeline to add, edit, and remove source properties from your records during data processing. Using a Perl manipulator, you can perform a number of tasks, including (but not limited to) the following:

- Remove a source property from each record
- Reformat a source property on each record
- Remove records with a particular source property
- Perform left joins on records from two sources
- Retrieve records matching a key from any number of data sources
- Process records using a Perl subclass

This section of the Developer Studio Help provides basic information on implementing Perl manipulators. Detailed Perl manipulator information can be found in the *Forge API Guide for Perl*, which is available from Help menu in Developer Studio. The *Forge API Guide for Perl* provides descriptions for the classes and methods you can incorporate in a Perl manipulator. It also provides sample code for the most common Perl manipulator tasks.

Note: Because Perl manipulators only deal with source properties, they must always come before property mappers in a pipeline's data flow.

Record Manipulator editor

The Record Manipulator editor contains a unique name for this record manipulator.

The Record Manipulator editor contains the following tabs:

Sources

The Sources tab contains the following options:

Option	Description
Record source	Required. A choice of the record servers in the project.
Dimension source	A choice of the dimension adapters and dimension servers in the project. If any expressions require dimensions, or the record manipulator's record index uses dimensions, then the manipulator must contain a dimension source.

Record Index

Optional. The Record Index tab allows you to add or remove dimensions or properties used in a component's record index, and to change their order. Record indexes support join functionality. See Join sources must have matching join keys and record indexes for more details.

The Record Index tab contains the following fields:

Field	Description
Discard records with duplicate keys	When checked, Forge discards any records with duplicate keys and logs a warning that specifies the number of records discarded. **Note:** Developer Studio performs a case-insensitive search for duplicate keys.

Comment

Optional. Provides a way to associate comments with a pipeline component.

Using Perl Manipulators to change Source Properties

About Perl Manipulators

A Perl manipulator component uses Perl to efficiently manipulate source records as part of Forge's data processing.

You can provide individual Perl methods to the Perl manipulator in either of the following ways:

- Write the code in the Perl Manipulator editor. This approach is useful for simpler data manipulation and cases where you want to keep the Perl code in the Developer Studio project.
- Specify the code in a Perl file (.pl) external to your project, and identify the file's URL in the Perl Manipulator editor. This approach is useful if you want to maintain the Perl code outside the Developer Studio project, reuse the code by calling the file from more than one pipeline, or if you simply prefer to work in an external editor.

Alternatively, if you want to write an entire Perl manipulator, you can specify the code in a Perl class external to your project, and identify the file in the 'Use this Perl class' setting in the Perl Manipulator editor. This approach is useful in cases where the amount of Perl code is large or complex.

Note:

- Perl manipulator names cannot contain spaces.
- Detailed information about the syntax used by the Perl manipulator, including examples, can be found in the *Forge API Guide for Perl* .

Adding a Perl manipulator

You can use a Perl manipulator to add, remove, and reformat properties, join record sources, and so on. If your pipeline contains a property mapper, the Perl manipulator is placed upstream of it.

A Perl manipulator is a pipeline component that uses Perl to efficiently manipulate source records and Endeca records as part of data processing performed in the Endeca Information Transformation Layer. This section describes the procedure for adding a Perl manipulator to your Endeca pipeline.

To add a Perl manipulator to your pipeline:

1. In the Pipeline Diagram editor, click **New**, and then choose **Perl Manipulator**.
   The Perl Manipulator editor appears.
2. In the **Name** text box, type a unique name for this Perl manipulator. Perl manipulator names cannot contain spaces.
3. Click the **Sources** tab and do the following to specify the Perl manipulator's record sources:

To add a record source:

a) Choose its name from the drop-down menu.

b) Click **Add**.

To remove a record source:

a) Select it in the list of sources.

b) Click **Remove**.

4. (Optional) In the Record Index tab, do the following:

   a) Specify which properties or dimensions you want to use as the record index for this component.

   b) Indicate whether you want to discard records with duplicate keys.

   Note: Developer Studio performs a case-insensitive search for duplicate keys.

5. Do one of the following to add or point to the necessary Perl code:

   - Write in-line Perl code.
   - Point to an external Perl file.
   - Point to an external Perl class.

6. In the Perl Manipulator editor, click **OK** to return to the Pipeline Diagram editor.

Perl Manipulator editor
The Perl Manipulator editor contains a unique name for this Perl manipulator.

The Perl Manipulator editor contains the following tabs:

General

The General tab contains the following options:

Option	Description
Override these methods	If you are using your own external Perl file (.pl) or want to write in-line Perl code, select 'Override these methods,' check one of the methods, and then click the method's Edit button to open the Method Override editor.
Use this Perl class	If you are using your own external Perl class , select "Use this Perl class" and then type the module's URL.

Sources

Required. A choice of record servers in the project. You can add more than one.

Record Index

(Optional) The Record Index tab allows you to add or remove dimensions or properties used in a component's record index, and to change their order. Record indexes support join functionality. See Join sources must have matching join keys and record indexes for more details.

The Record Index tab contains the following field:

Field	Description
Discard records with duplicate keys	When checked, Forge discards any records with duplicate keys and logs a warning that specifies the number of records discarded. **Note:** Developer Studio performs a case-insensitive search for duplicate keys.

Comment

(Optional) Provides a way to associate comments with a pipeline component.

Perl manipulator methods

Each Perl manipulator in your pipeline is an instance of the Forge Execution Framework's `EDF::Manipulator` class and can contain up to four methods that Forge executes to perform data retrieval and manipulation:

This topic assumes you understand the basic concepts behind record retrieval and manipulation as implemented by the Forge Execution Framework's four core classes. Oracle strongly recommends that you read Understanding record data flow for a basic discussion of these concepts before attempting to implement a Perl manipulator.

- `EDF::Manipulator::prepare` —The Forge Execution Framework calls this method before individual record processing begins. The `prepare` method performs set up and initialization tasks.
- `EDF::Manipulator::finish` —Similar to `prepare` , the Forge Execution Framework calls this method after all record processing is complete. Typically, a `finish` method performs clean up or logging tasks.
- `EDF::Manipulator::next_record`

  —A Perl manipulator's

  `next_record`

  method accomplishes three tasks:

  1. It calls an upstream component's `next_record`

     method to request the next record for processing.
  2. It processes the record appropriately according to the code in the Perl manipulator.
  3. It passes the processed record to the downstream component, via the Forge Execution Framework.

- `EDF::Manipulator::get_records` —Similar to next_record, a Perl manipulator's get_records accomplishes these tasks:

  1. The `get_records` method calls `next_record` on its upstream component multiple times to retrieve all of the records from the upstream component.
  2. `Get_records` then calculates which records out of the total collection match the specified key, and returns those records to the downstream component, via the Forge Execution Framework.

All record server components (record adapter, record cache, and so on) have native implementations of these four methods. With the exception of the Perl manipulator, however, the methods are internal and not accessible to developers. The Perl manipulator's native implementations of these methods do nothing. You must write your own implementations for a minimum of one of these methods, either `next_record` or `get_records`. The native implementations delegate responsibility for the tasks to your custom implementations. Your custom implementations use methods and classes in the EDF namespace such as EDF::Record, EDF::PVal, EDF::DVal, and so on, to accomplish their tasks. See the *Forge API Guide for Perl* for information about the methods and classes available in the EDF namespace.

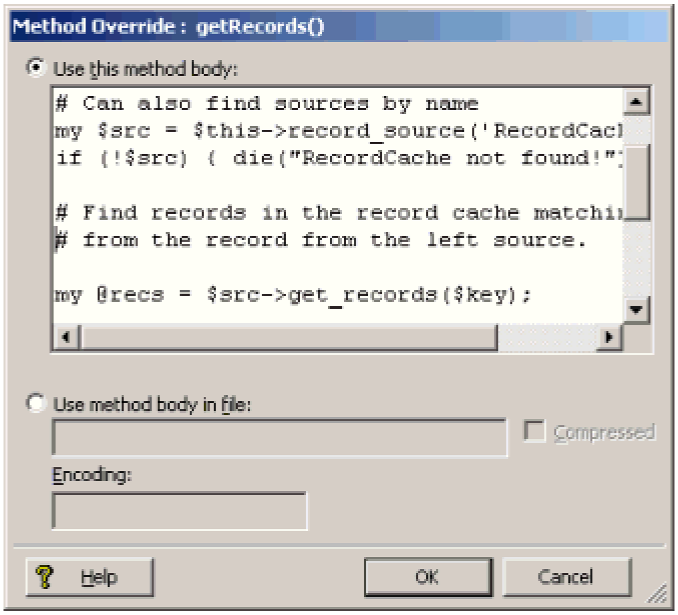
You can provide Perl methods using in-line code, or by providing a Perl file retrievable via URL. Alternatively, you can write your own class that provides these methods and point to it in the Perl Manipulator editor.

Writing code in a Perl manipulator
Include in-line Perl code in your pipeline using the Method Override editor.

To include in-line Perl code in your pipeline:

1. Create a Perl manipulator.
2. In the Perl Manipulator editor's General tab, select **Override these methods**.
3. Check the methods you want to override.
   The Method Override editor appears.
4. Select **Use this Method Body**.
5. Type or paste the method into the text box.

   Consult the *Forge API Guide for Perl* (available on the Developer Studio Help menu) for syntax details and examples.



● Use this method body:

```
# Can also find sources by name
my $src = $this->record_source('RecordCac
if (!$src) { die("RecordCache not found!");

# Find records in the record cache matchi
# from the record from the left source.

my @recs = $src->get_records($key);
```

○ Use method body in file:

□ Compressed

Encoding:

6. Click **OK** to return to the Perl Manipulator editor.

Detailed Perl manipulator information can be found in the *Forge API Guide for Perl*, which is available from Help menu in Developer Studio. The *Forge API Guide for Perl* provides descriptions for the classes and methods you can incorporate in a Perl manipulator. It also provides sample code for the most common Perl manipulator tasks.

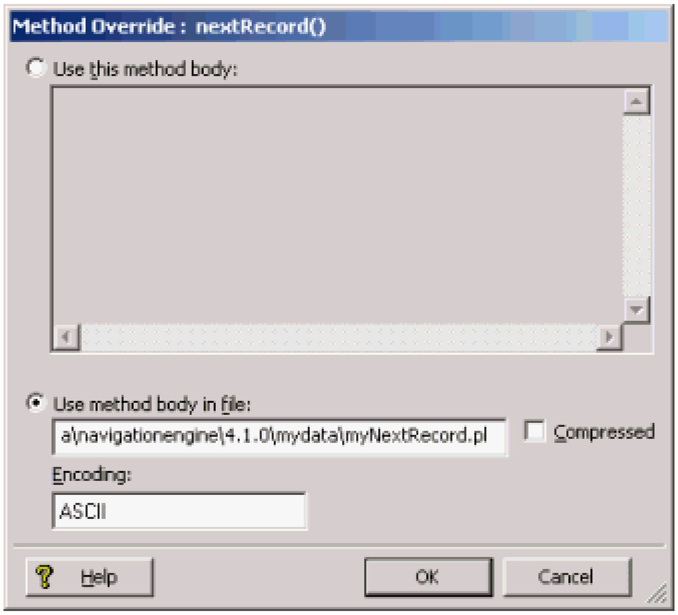
Using an external Perl file
Use the method body of an external Perl file to override methods in a Perl Manipulator.

You must create a Perl manipulator. See "Adding a Perl manipulator" for details on this procedure.

To use an external Perl file to override a method:

1. In the Perl Manipulator editor's General tab, select **Override These Methods**.
2. Check at least one of the methods.
   The Method Override editor appears.
3. Select **Use Method Body in File**.

4. Type the URL to the Perl file.



○ Use this method body:

● Use method body in file: a\navigationengine\4.1.0\mydata\myNextRecord.pl

□ Compressed

Encoding: ASCII

5. In the **Encoding** text box, type the encoding of the input data.
6. (Optional) If the Perl file being accessed is compressed, check **Compressed**.

   Note: This instructs Forge to decompress the file before processing.

7. Click **OK** to return to the Perl Manipulator editor.

Note: Detailed Perl manipulator information can be found in the *Forge API Guide for Perl*, which is available from Help menu in Developer Studio. *The Forge API Guide for Perl* provides descriptions for the classes and methods you can incorporate in a Perl manipulator. It also provides sample code for the most common Perl manipulator tasks.

Using an external Perl class

Use an external Perl class to call a method.

To use an external Perl class:

1. Create a Perl manipulator.
2. In the Perl Manipulator editor's General tab, select **Use this Perl Class**.
3. Type the URL to the Perl class.
   When the Perl manipulator runs, it loads and runs the specified class.

Requirements for a Perl class

This section describes the Perl class requirements for use with the Endeca software.

The Perl class must be located on the machine running Forge. It is convenient to locate the .pm file in the same location as other Perl modules for Endeca (`ENDECA_ROOT\lib\perl`). Placing your .pm file in `ENDECA_ROOT\lib\perl` does not require any additional configuration for Forge to locate it. However, if you upgrade Forge, you will need to copy the file to the new location.

If you place the file in another location, you must modify Perl's library search path to include the path to the .pm file. You can modify the path by either modifying your PERLLIB environment variable or by running Forge with the `--perllib` command line option and providing the path as an argument. In this case, you will not need to copy the file if you upgrade Forge.

Note: Detailed Perl manipulator information can be found in the *Forge API Guide for Perl*, which is available from Help menu in Developer Studio. The *Forge API Guide for Perl* provides descriptions for the classes and methods you can incorporate in a Perl manipulator. It also provides sample code for the most common Perl manipulator tasks.

Method Override editor

You can override methods in two ways: by writing in-line code, or by referencing an external Perl file. Both of these actions are performed in the Method Override editor.

The Method Override editor contains the following fields:

Option	Description
Use this method body	Provides a text box where you can type or paste your custom Perl code. The *Forge API Guide for Perl* , which can be accessed from the Developer Studio Help menu, provides information for the class and method descriptions that can be used in the Perl manipulator component.
Use method body in file	The URL of a Perl file that contains your custom Perl code.
Compressed	If checked, indicates that the file referenced in "Use method body in file" is compressed. In this case, Forge will uncompress it before processing.
Encoding	Optional. Defines the encoding of the input data. Several hundred encodings are supported; the following are typical examples: • ISO8859-1 (Latin-1) • ISO8859-15 (Latin-9) • CP1252 (WINDOWS-1252) • ASCII • UTF-8 If Encoding is not set, it is assumed to be UTF-8.

Using Java Manipulators to change Source Properties

About Java Manipulators

A Java manipulator is your own code in Java that takes records from any number of pipeline components in Forge or, optionally, your source data, and changes it according to your processing requirements.

A Java manipulator can then write any records you choose to its output. For example, a Java manipulator can write the "transformed" records into its output, so that the records can be passed to the next pipeline component in Forge.

Java manipulators are the most generic way of modifying your data and records in the pipeline. In other words, content adapters represent a specific case of Java manipulators. For more information about writing and implementing Java manipulators, refer to the *Endeca Content Adapter Developer's Guide* .

Java Manipulator editor

The Java Manipulator editor contains a unique name for this Java manipulator.

In addition, it contains the following tabs:

General

The General tab contains the following options:

Option	Description
Java home	Optional. Specifies the location of the Java runtime engine (JRE). If you do not specify this value, Forge first uses the value of the --javaHome flag. If the flag is not specified, Forge uses ENDECA_ROOT\j2sdk and lastly uses the JAVA_HOME environment variable.
Class	Required. Specifies the name of the class used by the component.
Class path	Optional. Specifies the path to a .jar file containing the class used by the manipulator. If you do not specify this value, the component checks for the class in the default class path of ENDECA_ROOT/lib/java. **Note:** When running your pipeline, you can override the Java home and Class path settings using command-line options. See Overriding Java home and class path settings.

Sources

A choice of record servers in the project. You can add more than one record server.

Pass Throughs

Optional. The Pass Throughs tab passes additional information to Forge. The tab contains text boxes where you can add, modify, or delete key/value pairs.

Comment

Optional. Provides a way to associate comments with a pipeline component.

Preparing for Indexing

Adding an indexer adapter

Indexer adapters save data that is ready to be indexed by the Dgidx program.

The final component in your pipeline is generally an indexer adapter. These components take all record, dimension hierarchy, and index configuration information from the pipeline and combine it in a format that is ready for the indexer (Agidx or Dgidx). An indexer adapter will generally be the final component in a baseline update pipeline. Attributes of the indexer adapter control where and how it writes its output, as well as where and how it reads its index configuration input.

To add an indexer adapter:

1. In the Pipeline Diagram editor, click **New**.
2. Choose **Indexer Adapter**.
   The Indexer Adapter editor appears.
3. In the **Name** text box, type a unique name for the indexer adapter.
4. In the General tab, do the following:
   a) In the **URL** text box, type the location to which the indexed records are written, relative to the Pipeline.epx file.
   b) In the **Output Prefix** text box, type the prefix that will be attached to the output files.
   c) (Optional) Check **Filter Unknown Properties** if you want the indexer adapter to remove source properties from your records.

      Note: After mapping, source properties still exist as part of the Endeca record. Enabling this option removes those source properties so records consist exclusively of Endeca properties and dimension values.

   d) (Optional) Check **Custom Compression** and slide the bar to the appropriate level.
5. (Optional) In the Sources tab, choose a record source and one or more dimension sources.
6. (Optional) If you are using an Agraph in your implementation, do the following:
   a) Select **Enable Agraph Support**.
   b) In **Number of Agraph Partitions**, specify the number of child Dgraphs that the Agraph coordinates.

      Note: If you want to change the partition property, open the Properties view and modify which properties are enabled for rollup and record spec. For more detailed information, see the *Endeca Advanced Development Guide*.

7. (Optional) In the Comment tab, add a comment for the component.
8. Click **OK**.

   Note: Typically, there is only one indexer adapter per pipeline.

Indexer Adapter editor

The Indexer Adapter editor contains a unique name for this indexer adapter.

The Indexer Adapter editor contains the following tabs:

General

The General tab contains the following options:

Option	Description
URL	Required. Location to which the indexed records are written.
Output prefix	Prefix that will be attached to the output files. For example, if the prefix is wine, the dimensions file will be called wine.Dimension.xml.
Output formats	Read-only information about output formats for the record file and the dimension file.
Filter unknown properties	Optional. Check **Filter unknown properties** if you want the indexer adapter to remove source properties from your records. **Note:** After mapping, source properties still exist as part of the Endeca record. Enabling this option removes those source properties so records consist exclusively of Endeca properties and dimension values.
Custom compression level	Optional. Sets the level of compression to be performed on the data. The values can range from 0 to 10, with higher numbers indicating higher compression (smaller size, slower processing).

Sources

The Sources tab contains the following fields:

Option	Description
Record source	Required. A choice of record server components in the project.
Dimension sources	A choice of dimension adapters and dimension servers in the project. You can add more than one.

Agraph

The Agraph tab contains the following options.

Option	Description
Enable Agraph Support	When checked, enables the Agraph program for use in a baseline update pipeline.

Option	Description
Number of Agraph partitions	Specifies the number of child Dgraphs that the Agraph controls. In an Agraph implementation, this must be a value of 2 or more.
Partition property	The partition property is a read only field that identifies the property by which records are assigned to each partition. If you are using rollup capabilities, the rollup property displays as the partition property. If you do not have a roll up property, but do have a record spec property enabled in your project, the record spec property functions as the partition property. If neither a rollup nor a record spec property exists, the partition property is empty, and Forge assigns records to each partition according to a round-robin strategy.

See the *Endeca Advanced Development Guide* for detailed information on how to configure, provision, and run an Aggregated MDEX Engine implementation.

Comment

Optional. Provides a way to associate comments with a pipeline component.

Preparing Your Endeca Properties

This section provides instructions for using Developer Studio to add and configure Endeca properties.

About Endeca properties

Endeca properties are name/value pairs intended for display in your Endeca implementation, after an end-user has searched for or navigated to a record set or an individual record.

Note: Understanding Endeca properties is critical to building an Endeca implementation. This topic provides a brief overview, however, Oracle strongly recommends that you read Chapter 2, "Understanding Records, Properties, and Dimensions," in the *Oracle Endeca Guided Search Concepts Guide* to make sure you have a solid grounding in this critical concept.

An Endeca implementation has two types of properties:

- Source properties are name/value pairs that contain descriptive information about a source record. Generally, a source record is nothing more than a set of source properties.
- Endeca properties are name/value pairs that are intended for display in your Endeca implementation, after an end-user has searched for or navigated to a record set or an individual record. Endeca properties are also useful for sorting, searching, and filtering. Endeca properties are generally derived from source properties.

While Endeca properties can be displayed and searched, they cannot be used for classification and navigation. You can think of Endeca properties as descriptive information only. If you want a source property to be available for navigation, you must map it to a dimension. See the *Endeca Forge Guide* for details.

Adding an Endeca property to an application

Specify an Endeca property's behaviors and be sure it is mapped to a source property with existing source data.

Deriving an Endeca property from a source property requires creating a mapping between the two. Mapping a source property to an Endeca property sanctions the source property for use within your Endeca implementation, and instructs the Data Foundry to populate the Endeca property with the value from the source property.

Adding an Endeca property to an implementation, therefore, is a three-step process. You must:

1. Make sure the source property you want to map to your Endeca property exists in your source data, or will be manually added to your records during data processing.
2. Add the Endeca property to your Developer Studio project and specify its behaviors, including if it is available for search and if it will appear in a record set and/or on an individual record page.

   See the "Configuring Endeca property behavior" section for details.
3. Create a mapping between the source property and the Endeca property. Without this mapping, the source property is discarded and its associated Endeca property will not be available for display.

   See the *Endeca Forge Guide* for details.

Note: Endeca properties and dimensions share a case-insensitive name space within the MDEX Engine. Therefore, you should avoid using the same name for an Endeca property and a dimension.

Working with Endeca Properties

The following sections describe how to work with Endeca properties.

Viewing all of your Endeca properties

Use the Properties view to see all Endeca properties in a project simultaneously.

The Properties view displays information about all of the Endeca properties in your project, in tabular format. You can add, remove, or modify Endeca properties here, as well as access more detailed information about an individual property.

To open the Properties view:

- In the Project Explorer Project tab, double-click **Properties**. The Properties view appears in the work area.

Properties view columns

The Properties view contains the following columns:

Column	Description
Name	A unique name for the Endeca property.
Type	One of the following: • Alpha • Integer • Floating point

Column	Description
	• File Path (reserved for advanced features) • Geocode (used for geospatial filtering and sorting) • Time • Datetime • Duration
Enable Sort	When Yes, indicates that records can be sorted by this Endeca property.
Enable Record Search	When Yes, indicates that record search should be enabled for this Endeca property. Record search finds all records in an Endeca implementation that have a property whose value matches a term the user provides.
Record Spec Property	When Yes, indicates that this property is enabled as the record specifier for all records.
Show with Record List	When Yes, enables this Endeca property to appear in the record list display. Any records that are tagged with this property will have this value shown as part of their entry in the record list.
Show with Record	When checked, allows this Endeca property to appear on the record page. Any records that are tagged with this property will have this property shown as part of their entry on the record page.

Sorting properties in the Properties view

Alphabetically sort Endeca properties in ascending or descending order.

For ease of use, the Endeca properties in your Properties view are sorted by name. You can choose whether the sort is ascending or descending.

Note: Using this feature just changes the sort in the Properties view. It does not determine how the MDEX Engine sorts results. See Enabling record sort on a property and Enabling record sort on a dimension for more information.

To sort the Endeca properties in your Properties view:

- In the Properties view, click Sort Ascending AZ↓ or Sort Descending ZA↓.

Adding a new Endeca property

Specify the data type and other attributes of the new property.

To add a new Endeca property:

1. On the Project tab, double-click **Properties** to open the Properties view.
2. In the Properties view, click **New**. The New Property editor appears.
3. In the **Name** box, type the name of the new Endeca property.

Endeca properties and dimensions share a namespace in the Endeca MDEX Engine. For this reason, all Endeca property and dimension names must be unique. If the application will use Endeca Query Language requests, all property and dimension names must be in an NCName format. For more information about the Endeca Query Language and the NCName requirement, see the *Endeca Advanced Development Guide*.

4. In the Type list, choose the type of the new Endeca property:
   - Alpha
   - Integer (Integers on all platforms range from -2^31 to 2^31 - 1. Integer property values larger than 2147483647 (2^31 – 1) will wrap around in the MDEX Engine and be displayed as (actual value – 4294967296), where 4294967296 = hex 0x100000000 = 2^32.)
   - Floating point
   - File path (deprecated)
   - Geocode (used for geospatial filtering and sorting)
   - Time
   - Datetime
   - Duration
5. Modify the other settings as appropriate.
6. Click **OK**.

Changing an Endeca property

Use the Property editor in the Properties view to make all changes to an existing property.

To make changes to a particular Endeca property:

1. In the Properties view, double-click the name of the Endeca property you want to modify to open it in the Property editor.

   Note: You can only open one property at a time.

2. Make the necessary changes in the Property editor.
3. Click **OK** to return to the Properties view.

Deleting an Endeca property

Select a property from the Properties view to delete it.

To delete an Endeca property from your project:

1. In the **Properties** view, select the Endeca property you want to remove from your project, and click **Delete**.
2. When the confirmation message appears, click **Yes**.

Deleting an Endeca property clears the property mappings from the full pipeline, but you must manually remove the mappings from the partial update pipeline.

Using time, datetime, and duration property types

The time, datetime, and duration property types let you identify date and time-related values as such so that the MDEX Engine handles those values appropriately.

For example, it is not meaningful to add the contents of two String properties (that is, Alpha properties), and a query that attempts this operation causes an error. Similarly, date and time-related operations, such as extracting the year out of a date property, are limited to date/time types.

To support dates and times, the Presentation API uses the Property class to represent key-value pairs, where the value is a String that represents a date, datetime, or duration. The front-end Web application may display this value without interpreting it, or it can use libraries to convert the value into a pretty-printed string that is appropriate for human consumption. In other words, the application may convert the String into a java.util.Date object (for Java) or DateTime object (for .NET), which can then be formatted using the standard Java and .NET calendar utilities, respectively.

The String value associated with a time, datetime, or duration contains an integer number. The specification for this number is as follows:

- Time: 32-bit unsigned integer representing the time as the number of milliseconds relative to the start of the day (midnight/12:00:00am).

For example, “1:00PM” or “13:00” is represented as “46800000” because:

13 hours *

60 minutes / hour *

60 seconds / minute *

1000 milliseconds / second =

46800000 milliseconds

- Datetime: 64-bit signed integer representing the date and time as the number of milliseconds since January 1, 1970, 12:00:00 GMT (the UNIX epoch).

For example, “August 26, 2004 1:00PM” is represented as “1093525200000” because:

12656 days *

24 hours / day *

60 minutes / hour *

60 seconds / minute *

1000 milliseconds / second +

46800000 milliseconds (13 hrs) =

1093525200000 milliseconds

- Duration: 64-bit signed integer representing the amount of time measured in milliseconds relative to an unspecified point in time.

For example,“100 days” is represented as “8640000000” because:

100days *

24 hours / day *

60 minutes / hour *

60 seconds / minute *

1000 milliseconds / second =

8640000000 milliseconds

Endeca distinguishes time, which represents time, from datetime, which represents a date and time, because it allows us to reduce the memory and storage footprint of applications that include only times. It also allows applications to explicitly detach a time from a specific date.

The duration type holds relative datetime values. This type provides application developers the ability to store values like “2 minutes” or “3 years.” In other words, values that should not be interpreted as absolute dates or times.

Loading date and time data

The MDEX Engine assumes that date/time data that is being loaded has the following characteristics:

- All values are in milliseconds, as described above.
- All values are provided in coordinated universal time, also known as UTC (UTC was
- formerly known as Greenwich mean time, or GMT).

If your date/time data does not conform to these requirements, you must pre-process it before loading, or manipulate it in your pipeline, so that it does.

Note: Endeca follows the POSIX standards.

Using the .NET framework's DateTime structure

The .NET framework uses the DateTime structure to represent an instant in time. In the DateTime structure, time values are measured in 100-nanosecond units called ticks. To load data created with the DateTime structure into the MDEX Engine, you must convert from ticks to nanoseconds, and then from nanoseconds to milliseconds. Conversely, to work with data returned from the MDEX Engine using the DateTime structure, you must convert it from milliseconds to nanoseconds, and then from nanoseconds to ticks.

The conversions work as follows:

1 second = 1,000 milliseconds

1 second = 1,000,000,000 nanoseconds

1,000 milliseconds = 1,000,000,000 nanoseconds

1 millisecond = 1,000,000 nanoseconds

1,000,000 nanoseconds * (1 tick / 100 nanoseconds) = 10,000 ticks

10,000 ticks = 1 millisecond

Working with time zones

Although the MDEX Engine will not interpret time zones directly, it is the query writer’s option to add or subtract another duration, possibly stored in another field on the record, from the record’s datetime value to account for different time zones.

Mapping a source property to an Endeca property

An Endeca property is populated with data from the source property to which it is mapped.

You must map a source property to an Endeca property if you want to retain for use in your Endeca implementation. You use the property mapper component to map source properties to Endeca properties. See the *Endeca Forge Guide* for details.

Property editor

You use the Property editor to create a new Endeca property or modify the attributes of an existing one.

General

Option	Description
Name	The name of the Endeca property. Property names are case sensitive. **Note:** Properties and dimensions share a namespace in the Endeca MDEX Engine. For this reason, you should not use the same name for both a property and a dimension.
Type	A drop-down list where you can select from the following property types: • Alpha • Integer • Floating point • Geocode (used for geospatial filtering and sorting) • Duration • DateTime • Time **Note:** The Reference and File Path property types has been deprecated and should not be used for new implementations.
Prepare sort offline	When checked, record sorting on this Endeca property is optimized.
Rollup	Enables aggregated Endeca record creation by allowing rollups based on this Endeca property.
Enable for record filters	When checked, enables this Endeca property for record filtering, which presents a subset of the data to the end-user. For more information about using record filters, see the *Endeca Advanced Development Guide*.
Use for record spec	When checked, specifies that you want to use this Endeca property as a record specifier (a unique string-based identifier). Record specifiers are required to support the partial updates feature and Term Discovery features. Even though those two features are the only two that require a record spec, Oracle recommends that you assign one property in the project as the record spec.
Show with record list	When checked, enables this Endeca property to appear in the record list display. Any records that are tagged with this property will have this value shown as part of their entry in the record list.
Show with record	When checked, allows this property to appear on the record page. Any records that are tagged with this Endeca property will have this property shown as part of their entry on the record page.

Option	Description
Language	Specifies the language for this property so that the MDEX Engine can perform language-specific operations correctly. If your application tends to have mixed-language records, and the languages are segregated into different properties, setting a per-property language ID might be appropriate. For more information about language settings, see the *Endeca Advanced Development Guide*.

Search Tab

Option	Description
Enable record search	When checked, indicates that record search should be enabled for this Endeca property. Record search finds all records in an Endeca application that have a property whose value matches a term the user provides. Checking "Enable record search" makes the following additional options available.
Enable wildcard search	When checked, indicates that a user query can contain a wildcard character (*) to match against fragments of words in a property value. You must enable each property that you want available for wildcard searching.

Configuring Endeca Property Behavior

This section describes how to configure Endeca property behaviour.

Enabling Display

Enabling Endeca property display in a record list

Enabling record list display means that any records tagged with this Endeca property display the property as part of their entry in the record list.

Note: You must enable a dimension for record list display before you can access its dimension values using the methods in the Endeca Presentation API.

To enable display in a record list for an Endeca property:

1. In the Properties view, double-click the Endeca property you want to change to open it in the Property editor.
2. Check **Show with Record List**
3. Click **OK**.

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Enabling Endeca property display on a record page

Enabling record display means that any records that are tagged with this Endeca property display the property as part of their entry on the record page.

Note: You must enable an Endeca property for record display before you can access it using the methods in the Endeca Presentation API.

To enable display on a record page for an Endeca property:

1. In the Properties view, double-click the Endeca property you want to change to open it in the Property editor.
2. Check **Show with Record**.
3. Click **OK**.

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Configuring Record Sort

About precomputing sort indices on a property

Although users can sort on any record at any time, it is also possible to optimize a property for sort in Developer Studio. This mainly controls the generation of a precomputed sort, and secondarily enables the field to be returned in the API sort keys function.

Sorting functionality allows an Endeca-enabled Web application to define the order of records returned with a query. When making a query request, the application may define a series of sort keys (property/order or dimension/order pairs, where order is either ascending or descending). The MDEX Engine then returns records according to those keys.

All of the records corresponding to a particular navigation state are considered for sorting, not just the records visible in the current request. For example, if a navigation state applies to 100 bottles of wine, all 100 bottles are considered when sorting, even though only the first ten bottles may be returned with the current request. The sort key's data type determines the type of sort that occurs—numeric, alphabetical, geospatial, time, datetime, or duration. If an Endeca record does not include a value for the specified sort key, that record is sorted to the bottom of the list, regardless of whether the sort order is ascending or descending.

Note: An explicit record sort key, specified as part of a MDEX Engine query, takes priority over any other type of record sorting (default sorting and relevance ranking). See Controlling the order of record results for details.

If the Web application does not specify sort order as part of the query, the MDEX Engine returns records in the default sort order, if one has been specified. See Specifying a default record sort order for details.

Record sorting only affects the order of records. It does not affect the ordering of dimensions or dimension values that are returned for query refinement. You use dimension and dimension value ranking to affect the order of dimensions and dimension values.

Configuring an Endeca property for record sort

The data type of an Endeca property determines the sort order of records sorted by that property.

To configure an Endeca property so that it can be used for record sort:

1. In the Properties view, double-click the Endeca property you want to change to open its Property editor.
2. Check **Prepare Sort Offline**.
3. In the Type list, verify that the property’s Type attribute is correct. This attribute affects sorting in the following ways:

If Type is:	Records are sorted:
Alpha	In alphabetical order, when sorted by this property.

If Type is:	Records are sorted:
Integer or Floating Point	In numeric order, when sorted by this property.
Geocode	According to the distance between the specified geocode property and a given reference point.
Time	According to the length of time between the specified time property and the start of the day (midnight or 12am).
Datetime	According to the length of time between the specified datetime property and the start of the epoch (12 am on January 1, 1970). **Note:** A negative datetime indicates time before the start of the epoch.
Duration	According to the length of time represented by the duration property. **Note:** Duration can be a negative number, for example, to express timezones relative to GMT/UTC time.

Note: The file path and reference property types are deprecated.

4. Click **OK**.

Specifying a default record sort order

If an Endeca-enabled Web application does not specify sort order as part of the query, the MDEX Engine returns query results using the default sort order, if one has been specified.

You specify the default sort order and sort direction (ascending or descending) by using the `--sort` flag when running Dgidx. The `--sort` flag has the following syntax:

```
--sort "key|dir"
```

where key is the name of an Endeca property or dimension on which to sort and dir is either asc for an ascending order or desc for descending (if not specified, the order will be ascending).

You can also specify multiple sort keys in the format:

```
--sort "key_1|dir_1||key_2|dir_2||...||key_n|dir_n"
```

If you specify multiple sort keys, the records are sorted by the first sort key, with ties being resolved by the second sort key, whose ties are resolved by the third sort key, and so on.

Note: If you are using the Endeca Application Controller (EAC) to control your environment, you must omit the quotation marks from the --sort flag. Instead, use the following syntax:

If you are using the Endeca Application Controller (EAC) to control your environment, you must omit the quotation marks from the --sort flag. Instead, use the following syntax:

```
--sort key_1|dir_1||key_2|dir_2||...||key_n|dir_n
```

Agraph default sort order and displayed record lists

If a default record sort order is specified in Dgidx based on a property which is not set to show in the record list, an Agraph managing the resulting Dgraphs will not consistently display records in the default sort order.

Each child Dgraph displays its own records in the correct order, but the Agraph does not reliably preserve this order when integrating its child record sets. The resulting record order will be close to—but not the same as—the actual specified default sort order.

To prevent this problem, use Developer Studio's Property editor to enable the 'Show with record list' setting for the sort property. This ensures that the Agraph will determine the correct record display order.

Relevance ranking versus record sort

Relevance ranking is used to control the order of results that are returned in response to a keyword search. Record sorting is used to control the order of records that are returned in response to any type of MDEX Engine query that returns records.

Relevance ranking and record sorting are closely related features but there are some distinct differences.

- Relevance ranking determines which results are more relevant to the user, based on a set of rules you define. For example, you can configure a rule that says "for multi-term searches, rank records that match more of the terms higher than those that match fewer terms." Relevance ranking is configured either as part of a search interface, where each search interface has its own relevance ranking strategy, or is specified in the record search query itself.
- Unlike relevance ranking which is limited to keyword search queries, record sorting can be used with any type of query that returns records. Record sorting is based on a sort key. The sort key can either be defined as a default, or identified by the Web application as part of the query.

Generally, if you have relevance ranking enabled, you would not specify a record sort key within a record search query because record sort keys take priority over all other types of ordering, making the relevance ranking settings useless.

Note: A search interface is a named collection of properties and dimensions, each of which has its Enable Record Search option checked. Search interfaces allow your end-users to search on multiple properties and/or dimensions simultaneously. The search interface's name is used just like a normal property or dimension when performing record searches. A record search query on a search interface returns results that match any of the properties or dimensions in the interface.

Controlling the order of record results

Specify an explicit sort key in the MDEX Engine query, set a default sort order, or use relevance ranking (for records returned in response to record search queries).

There are three ways of controlling the order in which records are returned:

- Specifying an explicit sort key in the MDEX Engine query, either via a URL parameter (Ns) or an ENEQuery setter method (setNavActiveSortKeys())
- Specifying a default sort order (Dgidx option).
- Using relevance ranking, for records returned in response to record search queries only. Relevance ranking settings can either be implicitly defined as part of the search interface, or explicitly defined in the search query itself.

The priority of record sorting/relevance ranking is as follows:

- If none of these three sorting methods is specified, records are returned in an arbitrary, but consistent, order as determined by an internal ID generated by Dgidx during indexing.
- If the MDEX Engine query includes an explicit sort key parameter, that sort key overrides all other sorting and relevance ranking settings.
- If a default sort key is specified, and no other sort parameters are set, records are returned in default sort order. Ties are broken using the arbitrary internal order described above.

- When searching against a search interface that incorporates a relevance ranking strategy, the relevance ranking strategy takes priority but ties are broken using the default sort key, if one has been specified. If there is no default sort key, ties are broken using the internal ID order described above.
- If the MDEX Engine query includes a relevance ranking parameter, that setting overrides any relevance ranking strategies configured in the search interface that is being searched against.

A search interface is a named collection of properties and dimensions, each of which has its Enable Record Search option checked. Search interfaces allow your end-users to search on multiple properties and/or dimensions simultaneously. The search interface's name is used just like a normal property or dimension when performing record searches. A record search query on a search interface returns results that match any of the properties or dimensions in the interface.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

About geospatial sorting

You can add properties to Endeca records that represent latitude and longitude pairs. These properties are known as geocode properties.

Result sets that have geocode properties can be sorted by the distance of the values of the geocode properties to a given reference point. For example, if the records of a particular dataset represent individual books that a large vendor has for sale at a variety of locations, each book could be tagged with a geocode property named Location that holds the store location information for that particular book. Users could then sort those books so that books that are more closely located are displayed first.

Note: Geospatial sorting is closely related to geospatial filtering.

Preparing your geocode source data

Your geocode data must be specially formatted as floating-point values in order to be successfully indexed by Dgidx.

In order for Dgidx to successfully index it, your geocode data must be in the form d,d where each d is a double precision floating-point value:

- The first d is the latitude of the location in whole and fractional degrees. Positive values indicate north latitude and negative values indicate south latitude.
- The second d is the longitude of the location in whole and fractional degrees. Positive values indicate east longitude, and negative values indicate west longitude.

Example

For example, a company’s main office is located at 42.365615 north latitude, 71.075647 west longitude. This geocode should be supplied to Dgidx as:

42.365615,-71.075647

If your source data is not available in this format, it can be assembled from separate properties using a Perl manipulator. See the Forge API Guide for Perl for examples of using a Perl manipulator to concatenate separate properties into one.

Configuring a geocode Endeca property for sort

Before you can map a geocode source property, you must configure the Endeca property that it will map to as geocode.

You map a source property to an Endeca property in order to make the source property available in your Endeca implementation. To configure a geocode Endeca property:

1. Create an Endeca property.

2. On the Project tab, double-click **Properties** to open the Properties view.
3. Double-click the Endeca property you want to configure. The Property editor appears.
4. In the Type list, choose **Geocode**.
5. Check **Prepare sort offline**.
6. Click **OK**.

Mapping, presenting, and sorting by geocode properties

You map, present, and sort geocode properties using similar techniques to those you use for alphanumeric properties.

- See Mapping a source property to an Endeca property for details on mapping.
- See the *Endeca Basic Development Guide* for details on displaying and sorting geocode properties within an Endeca-enabled Web application.

Troubleshooting record sorts

To fix issues with record sorts, check for property type, number of values assigned to each record, and uniqueness of property and dimension names.

If the records returned with a navigation request do not seem to respect the sort key parameter, here are some things to check:

- Was the Endeca property specified as a numeric when it is actually alphanumeric, or vice versa? In this case, the MDEX Engine returns a valid response, but the sorting may not be what you expected. Also, if you are sorting by a dimension, then the sort is always alphabetic.
- In general, properties and dimensions that are enabled for sorting should only have one value assigned per record. If a record has multiple property values or dimension values for a single Endeca property or dimension, the MDEX Engine sorts the records based on the first value associated with the key. If the application is displaying any value other than the first one, then the records may not appear to be sorted correctly.
- If an application has properties and dimensions with the same name and a sort is requested by that name, the MDEX Engine will arbitrarily pick either the property or dimension for sorting. In general, all properties and dimensions should have a unique name.

Enabling Record Search

Enabling a property for record search

You must enable record search for Endeca properties in order for records to appear in search results.

Record search finds all records in an Endeca application that are tagged with a dimension value or Endeca property that matches a term the user provides. In order for an Endeca property to be considered during record searches, you must enable it for record search.

To enable record search for an Endeca property:

1. In the Properties view, double-click the Endeca property you want to change to open it in the Property editor.
2. Select the Search tab.
3. Check **Enable Record Search**.
4. Click **OK.**
5. (Optional) If desired, check **Enable Wildcard Search**.

Note: See "Enabling wildcard search for an Endeca property" for details.

Fully implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Using Range Filtering

Using range filters

A range filter allows an Endeca-enabled Web application to select a subset of the total dataset for display, based on an arbitrary, dynamic range that uses an Endeca property or dimension as the filter key.

Navigation queries that use a range filter return only those records that are included in the selected data subset, along with the refinement dimension values that are appropriate for the filtered records. Range filters are supported for:

- Properties of type integer, floating point, geocode, datetime, time, or duration.
- Dimensions of type Integer or Floating Point.

For values of properties and dimensions of type Floating point, you can specify values using both decimal (0.00...68), and scientific notation (6.8e-10).

It is important to remember that range filters are simply modifiers for a navigation query. The range filter acts in the same manner as a dimension value, even though it is not a specific system-defined dimension value. Consider the following records and examples:

Record ID	Sample Dimension Value (Wine_Type)	Sample Property (Price)	Sample Property (Description)
1	Red (Dim Value 101)	10	Dark ruby in color, with extremely ripe…
2	Red (Dim Value 101)	12	Dense, rich and complex describes this '96 California…
3	White (Dim Value 102)	19	Dense and vegetal, with celery, pear and spice flavors…
4	Other (Dim Value 103)	20	Big, ripe and generous, layered with honey…

Example 1

When making a navigation request with a range filter specifying that the Price property is greater than 15 (with no dimension values specified) the following Navigation object is returned:

- 2 records (records 3 and 4)
- 2 refinement dimension values (White and Other)

Example 2

When making a navigation request with the Red dimension value specified (Dim Value 101) and a range filter specifying a Price less than 11, the following Navigation object is returned:

- 1 record (record 1)
- (No additional refinements)

Valid functions that can be used with range filters include:

- Less than

- Less than or equal to
- Greater than
- Greater than or equal to
- Between

Enabling a property for range filtering

Properties of numeric, geocode, time, datetime, and duration type may be enabled for range filtering.

Note: You may also use a dimension as the key for a range filter. See Enabling a dimension for range filtering for details.

To enable an Endeca property for range filtering:

1. In the Properties view, double-click the Endeca property you want to change to open it in the Property editor.
2. In the Type list, make sure the type is compatible with range filters: Integer, Floating point, Geocode, Time, Datetime, or Duration.

   Note: Be careful of dollar signs or other frequently found characters in property values that would prevent a property from being defined as numeric. For values of properties of type Floating point, you can specify values using both decimal (0.00...68), and scientific notation (6.8e-10).

3. Click **OK**.

About geospatial filtering

Result sets that have geocode properties can be sorted by the distance of the values of the geocode properties to a given reference point.

You can add properties to Endeca records that represent latitude and longitude pairs. These properties are known as geocode properties.

For example, if the records of a particular dataset represent individual books that a large vendor has for sale at a variety of locations, each book could be tagged with a geocode property named Location that holds the store location information for that particular book. Users could then filter the books to show only those books within a certain distance from their current location.

Note: Geospatial filtering is closely related to geospatial sorting.

Configuring a geocode Endeca property for filtering

You map a source property to an Endeca property in order to make the source property available in your Endeca implementation. Before you can map a geocode source property, you must configure the Endeca property that it will map to as geocode. To configure a geocode Endeca property:

1. Create an Endeca property.
2. On the Project tab, double-click Properties to open the Properties view.
3. Double-click the Endeca property you want to configure. The Property editor appears.
4. In the Type list, choose Geocode.
5. Click OK.

Enabling Record Rollups

About aggregated Endeca records

Aggregated records allow you to treat a collection of separate records as one if the rollup key is the same for any number of records.

An aggregated record is a collection of individual Endeca records that have been rolled up based on a rollup key (an Endeca property or dimension name). All records in the current record set that have the same value for the rollup key are collected together into an aggregated record. For example, rolling up on a Name key causes all wines in the current record set that have the value 'My Red Wine' for the Name key to be rolled up into one aggregated record.

Commonly, aggregated records are used to eliminate duplicate display entries. For example, in a music store catalog, an album by the same title may exist in several formats, with multiple prices. Each title is represented in the MDEX Engine as a distinct record. However, from a business perspective, it might be useful to treat these separate records as a single record by creating an aggregate record.

Record aggregation affects the current record set only. In other words, if you have 10,000 Endeca records total but only 3,000 are displayed in the current record set, then the aggregation affects those 3,000 records only.

The aggregated records feature requires that each record should have at most one value from the dimension or Endeca property that has been specified as the rollup key. Also, if an Endeca record has a unique value for the rollup key, it is 'rolled up' into an aggregated record that contains only one sub-record.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

About automatic phrasing

When an application user provides individual search terms in a query, the automatic phrasing feature groups those individual terms into a search phrase and returns query results for the phrase.

Automatic phrasing is similar to placing quotation marks around search terms before submitting them in a query. For example, 'my search terms' is the phrased version of the query my search terms. However, automatic phrasing removes the need for application users to place quotation marks around search phrases to get phrased results.

The result of automatic phrasing is that a Web application can process a more restricted query and therefore return fewer and more focused search results. This feature is available only for record search.

The automatic phrasing feature works by:

1. Comparing individual search terms in a query to a list of application-specific search phrases. The list of search phrases are stored in a project's phrase dictionary.
2. Grouping the search terms into search phrases.
3. Returning query results that are either based on the automatically phrased query, or returning results based on the original unphrased query along with automatically phrased 'Did You Mean?' (DYM) alternatives.

Point three above suggests the two typical implementation scenarios to choose from when using automatic phrasing:

- Process an automatically phrased form of the query and suggest the original unphrased query as a DYM alternative.

  In this scenario, the automatic phrasing feature rewrites the original query's search terms into a phrased query before processing it. If you are also using DYM, you can display the unphrased alternative so the user can opt-out of automatic phrasing and select their original query, if desired.

For example, an application user searches a wine catalog for the terms "low tannin." The MDEX Engine compares the search terms against the phrase dictionary, finds a phrase entry for "low tannin," and processes the phrased query as "low tannin." The MDEX Engine returns 3 records for the phrased query "low tannin" rather than 16 records for the user's original unphrased query "low tannin." However, the Web application also presents a "Did you mean low tannin?" selection so the user may opt-out of automatic phrasing, if desired.

- Process the original query and suggest an automatically-phrased form of the query as a DYM alternative.

  In this scenario, the automatic phrasing feature processes the unphrased query as entered and determines if a phrased form of the query exists. If a phrased form is available, the Web application displays an automatically-phrased alternative as a "Did you mean?" option. The user can opt-in to automatic phrasing, if desired.

  For example, an application user searches a wine catalog for low tannin. The MDEX Engine returns 16 records for the user's unphrased query low tannin. The Web application also presents a "Did you mean "low tannin"?" option so the user may opt-in to automatic phrasing, if desired.

There are two tasks to implement automatic phrasing:

- Add phrases to your project using Developer Studio.
- Add Presentation API code to support either of the two implementation scenarios described above.

Note: Implementing search features requires additional work outside of Developer Studio. Refer to the *Endeca Advanced Development Guide* for details.

Enabling a property for record rollup

In order to use an Endeca property as a rollup key, you must enable its Rollup attribute.

To enable record rollup for an Endeca property:

1. In the Properties view, double-click the Endeca property you want to change to open it in the Property editor.
2. Check **Rollup**.
3. Click **OK**.

Note: This feature requires that each record should have at most one value from the Endeca property or dimension that has been specified as the rollup key.

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Enabling Wildcard Search

Enabling wildcard search for an Endeca property

Wildcard searching allows user queries that contain a wildcard character (*) to match against fragments of words in a property value. You must enable each property that you want available for wildcard searching.

To enable wildcard searching for an Endeca property:

1. In the Properties view, double-click the Endeca property you want to change.
2. In the Property editor, select the **Search** tab.
3. Check **Enable Record Search**.
4. Check **Enable Wildcard Search**.
5. Click **OK**.

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Setting a property to use as the record spec

A "record spec," or record specifier, is a string-based identifier.

These identifiers are used by the Term Discovery and partial updates features, described in detail in the *Endeca Forge Guide* and the *Endeca Partial Updates Guide*, respectively. Featured records also require a record specifier. Oracle recommends that you assign one property in the project as the record spec.

Note: Records can have only one record spec during updates and at startup.

To configure an Endeca property for use as the record specifier:

1. In the Properties view, double-click the Endeca property you want to change to open it in the Property editor.
2. Check **Use as Record Spec**, and click **OK**.

Troubleshooting Endeca properties

Check property mapping and record options if an Endeca property is missing from your Web application.

If an Endeca property is not present in your Endeca-enabled Web application, do the following:

- Make sure there is a mapping between a source property and the Endeca property. (By default, Forge removes source properties that have not been mapped to an Endeca property or dimension. See the *Endeca Forge Guide* for details.
- Check to see if the Show with Record List and Show with Record options are set correctly in the Property editor for the affected Endeca property.

Preparing Your Dimension Hierarchy

This section provides information on working with dimensions, dimension values, and adding pipeline support.

About dimensions and dimension values

Understanding dimensions and dimension values is critical to building an Oracle Commerce Guided Search implementation.

Note: This topic provides a brief overview. However, Oracle strongly recommends that you read "Understanding Records, Properties, and Dimensions," in the *Endeca Concepts Guide.*

Dimensions

Dimensions provide the logical structure for organizing the records in your data set. Your Endeca application can have many dimensions, and dimensions can be hierarchical. For example, Merlot and Chablis dimensions could be children of the Wine dimension.

Dimension values

Dimension values both organize the record within the tree structure of the associated dimension, and identify the record as a valid result when that dimension value is selected via a navigation query.

Dimension values can either be discrete values (as in a year, a flavor, or a price), or include ranges with an upper and lower bound (for example, the years 1990-1999, or all prices under $10).

The top-most dimension value in a dimension tree is known as the dimension root. The bottom-most dimension values in a tree are referred to as leaf dimension values.

Auto-generated dimensions

Dimensions are often, though not always, derived automatically from source data properties.

Dimension value synonyms

Synonyms provide a textual way to describe a dimension value. A dimension value's definition includes one or more synonyms. Synonyms are used for display, search, and/or mapping, depending on how you set their three flags. Each of the synonym flags refers to the text that is contained in the body of the synonym element.

Note: Dimension values have synonyms, but dimensions do not.

Dimension.xml file

Developer Studio stores dimension hierarchy information in an XML file that the documentation and reference implementations generically call Dimension.xml. A data pipeline can have more than one Dimension.xml file, allowing you to provide dimension data from multiple sources.

Adding Pipeline Support for Dimensions

Pipeline components that support dimensions

There are two pipeline components that support dimensions: dimension adapters and dimension servers.

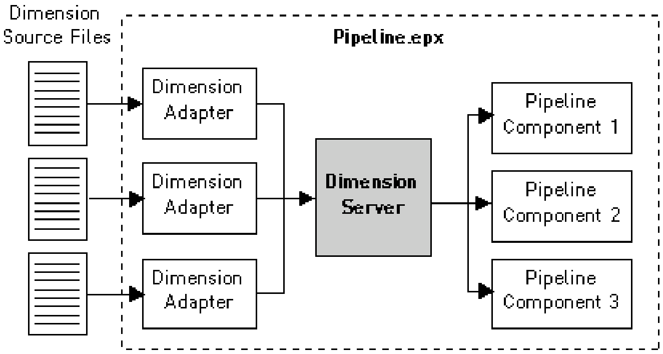
Dimension adapters

Developer Studio stores dimension hierarchy information in an XML file that the documentation and reference implementations generically call Dimension.xml (you can name it anything you like). A dimension adapter reads data from and writes data to this file. The attributes of a dimension adapter describe where the dimension file is located, its format, and various aspects of processing. A data pipeline can have more than one dimension file, allowing you to provide dimension data from multiple sources. Each dimension file requires its own dimension adapter.

Dimension server

Dimension servers work in conjunction with dimension adapters, and serve as a centralized source of dimension information for all other pipeline components:

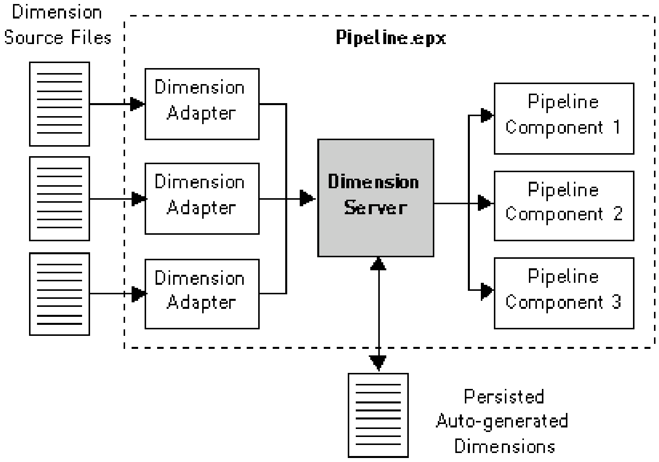
1. Dimension adapters load dimension information from your dimension source files.
2. The dimension server gets its dimension information from the dimension adapters.
3. Other pipeline components get their dimension information from the dimension server.



Setting up your pipeline in this way allows you to change your dimension adapters as needed without having to change the dimension source for all other pipeline components that require dimension information.

Persisting auto-generated dimensions

In addition to functioning as a centralized source for dimension information, dimension servers also coordinate the loading and saving of automatically-generated dimension information. Auto-generated dimensions are persisted in the file location that is specified in the Dimension Server editor

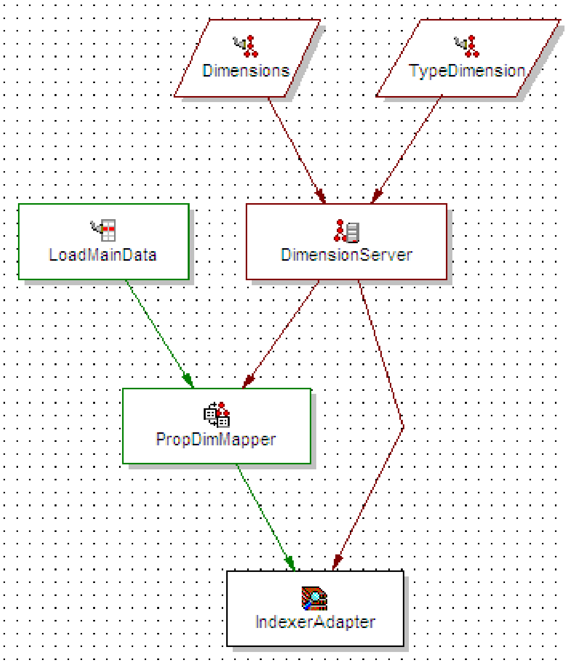


Supplying dimension data to property mappers

Your pipeline's property mapper requires a dimension server as its source of dimension data.

An example

The following pipeline example has two dimension adapters, Dimensions and TypeDimension, that both feed a dimension server, DimensionServer. DimensionServer provides dimension data to the pipeline's property mapper, PropDimMapper.



Adding a dimension adapter

Dimension adapters read and write dimension data. The attributes of a dimension adapter describe where the dimension data is located (or will be saved to), its format, and various aspects of processing.

The adapter must have a unique name within the pipeline. A single input dimension adapter reads all the dimensions in a single dimension file so, if you have multiple dimension files, you must have a separate adapter for each file.

Output dimension adapters are generally only used for diagnostic purposes. Hence, this section focuses on input dimension adapters. See "Writing out dimension data" for more information on output dimension adapters.

To add an input dimension adapter:

1. In the Pipeline Diagram editor, click **New**.
2. Choose **Dimension > Adapter** .
   The Dimension Adapter editor appears.
3. In the General tab, do the following:
   a) In the Direction frame, choose **Input** if the adapter will be loading dimensions.
   b) In the **URL** text box, enter the input file location.
   c) In the Format text box, select the format:

      XML- Internal: Recommended (unless you are importing taxonomies from a third-party tool).

      Binary

      XML - Externally managed: Used when importing dimensions created in a third-party tool. See "Working with an Externally Managed Taxonomy" in the *Endeca Forge Guide* for details.

      Stratify: Used when importing dimensions created with the Stratify taxonomy creation tools. See "Classifying Documents with Stratify" in the *Endeca Forge Guide* for details.

   d) Check **Require Data** if you want an error to be generated if the URL (see step B, above) does not exist or is empty.

e) (Optional) Check **Custom Compression Level** if your input file is compressed to indicate to Forge that it must decompress data from this source.

> **Note:** The compression level slider setting is ignored for an input dimension adapter.

4. (Optional) In the Transformer tab, do the following to transform an external dimension hierarchy to Endeca dimensions:
   a) Enter XSLT in the TYPE field. XSLT is the only valid value.
   b) Enter the full path to the .xsl file.
5. (Optional) In the Comment tab, add a comment for the component.
6. Click **OK**.

> **Note:** You can choose a dimension source on the sources tab. This feature is only used with output dimension adapters.

Dimension Adapter editor

The Dimension Adapter editor contains a unique name for this dimension adapter.

The Dimension Adapter editor contains the following tabs:

General

Option	Description
Direction	**Input Adapter** Required. Must be set to input. **Output Adapter** Required. Must be set to output. **Note:** Output dimension adapters are used for diagnostic purposes.
Format	**Input Adapter** Required. One of the following: • XML - Internal: Recommended (unless you are importing taxonomies from a third-party tool). • Binary • XML - Externally managed: Used when importing dimensions created in a third-party tool. See "Working with an Externally Managed Taxonomy" in the *Endeca Forge Guide* for details. • Stratify: Used when importing dimensions created with the Stratify taxonomy creation tools. See "Classifying Documents with Stratify" in the *Endeca Forge Guide* for details.

Option	Description
	Output Adapter Required. Must be XML.
URL	**Input Adapter** Required. Location of the input file. **Output Adapter** Required. Location to which the dimension data is saved.
Require data	**Input Adapter** Optional. If checked, this field generates an error if the URL does not exist or is empty. **Output Adapter** Not used.
Compression level	**Input Adapter** Optional. Indicates to Forge that it must decompress data from this source. The compression level slider setting is ignored. **Output Adapter** Sets the level of compression to be performed on the dimension data when its written to disk. To save on the amount of disk space used, check **Custom compression level** and slide the bar to Endeca's recommended value of 7. **Note:** Compressed data consumes less disk space but takes longer to read and write.

Sources

Required for output adapters. A choice of the dimension adapters and dimension servers in the project.

Note: Generally, only output adapters use dimension sources. For example, you can have a pipeline where an input dimension adapter reads dimension data from a source, and then passes that data to an output dimension adapter that saves it to disk.

Transformer

The Transformer tab allows a dimension adapter to transform external dimension hierarchies, described in XML format into Endeca records XML format. To support these situations, an XSLT transformation can be applied to the source data to convert it into Endeca Records XML, which the Data Foundry can read.

The Transformer tab has the following options:

Option	Description
Type	Must be XSLT.
URL	Location of the stylesheet to use.

Comment

Optional. Provides a way to associate comments with a pipeline component.

Adding a dimension server

A dimension server collects dimensions from multiple dimension sources, and makes them accessible in a uniform way to other components throughout the pipeline.

The dimension server must have a unique name within the pipeline, and must specify all the input dimension adapters as its dimension sources. If automatic dimension generation is used, a format, compression level, and URL must be provided so the automatically generated data can be persisted. Finally, a property mapper requires a dimension server as its source of dimension data.

Typically, there is only one dimension server per pipeline.

To add a dimension server:

1. In the Pipeline Diagram editor, click **New**, and then choose **Dimension** > **Server**.
   The Dimension Server editor appears.
2. In the **Name** text box, type a unique name for the dimension server.
3. In the General tab, do the following:
   a) In the **URL** text box, type the location where dimension data created by auto-generation will be saved.
   b) In the Format list, choose **Binary** or **XML** (Oracle recommends binary).
   c) To save on the amount of disk space used to persist auto-generated dimensions, check **Custom Compression Level** and slide the bar to Endeca's recommended value of 7.

   Note: Compressed data consumes less disk space but takes longer to read and write.

   d) (Optional) To restrict the range of IDs that Forge assigns to automatically-generated dimension values, check **Dimension Value ID Limits** and enter the minimum and maximum values that you want to use.
4. In the Sources tab, choose a dimension source.
5. Click **Add**. Repeat for all dimension sources.
6. (Optional) In the Comment tab, add a comment for the component.
7. Click **OK**.

Dimension Server editor

The Dimension Server editor contains a unique name for this dimension server.

The Dimension Server editor contains the following tabs:

General

The General tab contains the following options:

Option	Description
URL	This attribute specifies the location of the persistent dimension data created by auto-generation. The path is either an absolute path, or a path relative to the pipeline. With an absolute path the protocol could be specified in RFC 2396 syntax. Usually this means the prefix file:/// precedes the path to the data file.
Format	Required. The dimension server can read and write in two formats: Binary and XML.
Custom compression level	Optional. To save on the amount of disk space used to persist auto-generated dimensions, check **Custom compression level** and slide the bar to Endeca's recommended value of 7. **Note:** Compressed data consumes less disk space but takes longer to read and write.
Dimension value id limits	Optional. To restrict the range of IDs that Forge assigns to automatically-generated dimension values, check **Dimension value id limits** and enter the minimum and maximum values that you want to use.

Sources

Required. A choice of the dimension adapters and dimension servers in the project.

Comment

Optional. Provides a way to associate comments with a pipeline component.

Working with Dimensions and Dimension Values

A dimension is a collection of related dimension values, organized into a tree. Dimension values are tags, or labels, you use to classify the records in your data set.

Basic Dimension Tasks

About the Dimensions view
Access detailed information about a dimension and its dimension values, modify a dimension, or determine the order in which they are returned from the MDEX Engine.

The Dimensions view displays information about all of the dimensions in your project in tabular format. You can add, remove, or modify dimensions here, as well as access more detailed information about an individual dimension and its dimension values. The Dimension view is also where you rank your dimensions to determine the order in which they are returned from the MDEX Engine.

Viewing all of your dimensions
All dimensions are visible in the Dimensions view, opened from the Project Explorer.

To open the Dimensions view:

- In the Project Explorer Project tab, double-click **Dimensions**.
  The Dimensions view appears in the work area.
- Choose from the following options:

Option	Description
Name	A unique name for the dimension. Clicking the Name column head cycles the order of the dimension sort from rank order, to alphabetical ascending, to alphabetically descending. **Note:** In order to rank your dimensions, you must be displaying them in rank order in the Dimensions view.
External type	If the dimension was not manually created in Developer Studio, indicates what type of external dimension it is: • Auto Gen: The dimension's values are being automatically generated because the match mode in the Property Mapper is set to Auto Generate. • External: Indicates that this dimension has been created and is being maintained in a third-party tool, outside of Developer Studio. Externally managed dimensions are read-only. • Prop Mapped: Indicates that this dimension is being created using the "If no mapping is found, map source properties to Endeca: Dimensions" on the Property Mapper's Advanced tab.
Enable record sort	When Yes, indicates that records can be sorted by this dimension.
Show with record list	Enables the dimension values from this dimension to appear in the record list display.
Show with record	Enables the dimension values from this dimension to appear on the record pages of any records that are tagged with these dimension values.
Dynamic ranking	Establishes dynamic dimension value ranking, which orders dimension values according to their frequency of appearance within the current record set (that is, the most popular dimension values are returned).
Multi select	Allows the end user to select more than one dimension value from a dimension within a single query.
Refinements sort order	Defines the default sort order used for dimension values within this dimension. This sort order is used when no other types of dimension value sorting have been specified.

Sorting dimensions in the Dimensions view

To make them easier to find and work with, the dimensions in your Dimensions view can be sorted by name (in alphabetical ascending or descending order) or by rank.

Note: The procedure described below changes the way dimensions are sorted in Dimensions view only. It does not determine how dimensions are ranked when they are returned from the MDEX Engine. See Ranking dimensions manually for more information.

To sort dimensions in the Dimensions view:

- In the Dimensions view, click the **Name** column header to cycle the order of the dimension sort from rank order, to alphabetical ascending order, to alphabetically descending order.

Creating dimensions

Create new dimensions from the Dimensions view, on the Project tab.

To add a new dimension:

1. On the Project tab, double-click **Dimensions** to open the Dimensions view.
2. Click **New**.
   The Dimension editor appears.
3. In the Name box, type the name of the new dimension.

   Note: Endeca properties and dimensions share a namespace in the Endeca MDEX Engine. For this reason, Endeca property and dimension names should always be unique. If the application will use Endeca Query Language requests, all property and dimension names must be in an NCName format. For more information about the Endeca Query Language and the NCName requirement, see the *Endeca Advanced Development Guide*.

4. Modify the other settings as appropriate.
5. Click **OK**.

To add dimension values to the new dimension:

- In the Dimensions view, select the new dimension and then click **Values**. This opens the Dimension Values view, where you can establish the dimension value hierarchy.

  Note: Do not add Dimension Values with ID=0. Dgidx rejects dimension values with ID=0.

Modifying dimensions

You can modify one dimension at a time using the Dimension editor.

To make changes to a particular dimension:

1. In the Dimensions view, double-click the dimension you want to modify.
   This opens in the Dimension editor.

   Note: You can only open one dimension at a time.

2. Make the necessary changes.
3. Click **OK** to return to the Dimensions view.

Deleting dimensions

Delete dimensions from the Dimensions view.

To delete a dimension from your project:

1. In the Dimensions view, select the dimension you want to remove from your project, and click **Delete**.
2. When the confirmation message appears, click **Yes**.

Dimension editor

You use the Dimension editor to create a new dimension or modify the attributes that affect how an existing dimension is evaluated and displayed.

The top of the Dimension editor contains the following information that identifies the dimension:

Option	Description
Name	The name of this dimension. Dimension names are case sensitive.
ID	A unique system-generated identifier.
Member of this dimension group	Allows you to select from existing dimension groups or add a new one.
Refinements sort order	Specifies the sort type for any refinement dimension values that are returned for this dimension: Alpha, Integer, or Floating point.

The lower half of the Dimension editor contains five tabs. See tables below for details.

General

The General tab contains the following settings:

Option	Description
Prepare sort offline	When checked, record sorting on this dimension is optimized.
Hidden	Specifies whether or not this dimension is shown in the navigation controls.
Show with record list	When checked, enables this dimension to appear in the record list display. Any records that are tagged with a value from this dimension will have the value shown as part of their entry in the record list.
Show with record	When checked, allows this dimension to appear on the record page. Any records that are tagged with a value from this dimension will have that value shown as part of their entry on the record page.
Language	Specifies the language for this dimension so that the MDEX Engine can perform language-specific operations correctly. If your application tends to have mixed-language records, and the languages are segregated into different dimensions, setting a per-dimension language ID might be appropriate. For more information about language settings, see the *Endeca Advanced Development Guide*.

Search

The Search tab contains the following settings:

Option	Description
Search hierarchy for dimension search	When checked, allows dimension search to consider ancestor dimension values when matching a dimension search query.
Enable record search	Specifies whether or not record search should be enabled for this dimension. Record search finds all records in an Endeca application that have a dimension whose value matches a term the user provides. Checking **Enable record search** makes the following additional options available.
Search hierarchy for record search	When checked, allows record search to consider ancestor dimension values when matching a record search query. This setting is only enabled when **Enable record search** is checked.
Enable wildcard search	When checked, indicates that a user query can contain a wildcard character (*) to match against fragments of words in a dimension value. You must enable each dimension that you want available for wildcard searching.

Note: The Dimension Search Configuration editor does not specify the same options as the Search tab of the Dimension editor. You use the Dimension Search Configuration editor to configure dimension search options for all dimensions in your project. The Search tab of the Dimension editor affects record search.

Advanced

The Advanced tab contains the following settings:

Option	Description
Primary	Specifies whether this dimension is the project's sole primary dimension. All other dimensions are secondary. (Note that a primary dimension is no longer required and is ignored by the MDEX Engine. The MDEX Engine treats all dimensions as secondary, no matter what you specify in this field).
Multiselect	Allows the end user to select more than one dimension value from a dimension.
Enable for rollup	Enables aggregated Endeca record creation by allowing rollups based on this dimension.
Compute refinement statistics	Enables the computation of refinement statistics.
Collapsible dimension threshold	Allows you to set your application to collapse a deep hierarchy to make it shallower when available data is small.

Dynamic Ranking

The Dynamic Ranking tab contains the following settings:

Option	Description
Enable dynamic ranking	When checked, indicates that the list of refinement dimension values returned for a query should be pruned to those values that occur most frequently in the requested navigation state.
Maximum dimension values to return	Sets the number of most frequently-occurring (popular) dimension values to return.
Sort dimension values	Establishes the sort method used for the most popular dimension values: • "Alphabetically" uses whatever order you've selected for the "Refinements sort order" setting on the main part of the Dimension editor. • "Dynamically" orders the most popular refinement values according to their frequency of appearance within a data set. Dimension values that occur more frequently are returned before those that occur less frequently.
Generate "More..." dimension value	When this option is checked, if the actual number of refinement options exceeds the number set in "Maximum dimension values to return," then an additional option called More is returned for that dimension. If the user selects the More option, then the MDEX Engine will return all of the refinement options for that dimension. If "Generate 'More' dimension value" is not checked, only the number of dimension values defined in "Maximum dimension values to return" is displayed.

Cluster Discovery

The Cluster Discovery tab contains the following settings:

Option	Description
Enable clustering	Specifies whether Cluster Discovery is enabled for this dimension. The checkbox also makes the following controls available. For more information, see the *Endeca Relationship Discovery Guide.*
Sample size	This parameter governs how many documents are sampled. Clustering processing time and memory consumption are both roughly linear with this number; thus, lowering the value results in smaller memory consumption and faster turnaround. However, statistical errors are likely to occur when the sample size is small.

Option	Description
	Setting this value higher will overcome statistical errors for data sets where fewer terms are tagged onto each document. Range: Integer, 50-2000 (default: 500) Recommended value: 500
Maximum clusters	This parameter limits the number of clusters that will be generated by the MDEX Engine. Range: Integer, 2-10 (default: 10) Recommended value: 6
Coherence	This parameter governs the decision of whether a set of terms is coherent enough to form a cluster (that is, each cluster should have only closely related documents). Low values are permissive (i.e., not demanding much coherence) and will result in fewer larger clusters; high values are strict and will result in more smaller clusters. The average value is recommended. Range: Integer, 2-10 (default: 10) Recommended value: 6
Maximum precision	Terms that are extracted from sampled documents are filtered by their precision p (where p = number of sampled documents that this term is tagged onto divided by the number of all sampled documents). Terms that have too high a value of p are likely to be the search term (or be synonymous with it) or be too general to make for a good clustering term. If you use the recommended tuning values of the term extractor, each term is tagged only roughly 1/3 of the documents, which means that the search term, if present, will have p of roughly 0.33 (more or less stringent tuning of the term extractor will change this value). There usually is a gap in the values of p between the search term and the more useful terms, which start at approximately p = 0.25 and less. Range: Float, 0.0 - 1.0 (default: 1.0) Recommended value: 0.25
Maximum cluster size	Sets the maximum number of terms that can be in a cluster. Each cluster will have at least 2 terms. Because of the match-partial cluster selection mechanism, the more terms there are in the cluster, the (potentially) higher its coverage will be. On the other hand, the

Option	Description
	clusters that are too large take up too much space to display and take too long for users to read. Range: Integer, 2 - 10 (default: 10) Recommended value: 8
Maximum cluster overlap	If two clusters overlap (that is, if the document sets that each cluster maps to overlap), then the smaller one (as measured by the estimated size of the document set it maps to) can be removed, depending on how big this overlap is. This parameter dictates the overlap above which the smaller cluster is removed. Clusters which overlap by more than this value will be removed. Thus, the default setting of 10 (out of ten) means that clusters that overlap by more than 10 out of 10 documents will be removed. Since this is impossible, this means that setting of 10 will disable cluster overlap filtering, which is most extreme level of coarseness for this filter. Tuning this parameter down will make the cluster overlap more and more fine-grained. Thus, a value of 9 will remove only the clusters that greatly overlap; setting it to the recommended value of 5 will remove only clusters overlapping half-way or so (remember that the overlap is merely estimated). Setting this parameter to lower values (less than 5) will make overlap filtering quite sensitive and will remove clusters which overlap even by a small amount. Note that clusters that do not overlap at all will never be filtered. Range: Integer, 0-10 (default: 10) Recommended value: 5

Checking the ID of a dimension

A dimension ID is system-generated and cannot be changed, but you can view it.

Each dimension has a unique numeric ID.

To check the ID of a dimension:

1. In the Dimensions view, double-click the dimension you want to examine to open it in the Dimension editor. The Dimension ID appears in the upper right corner of the Dimension editor.
2. Click **OK** to close the Dimension editor.

Adding a dimension to a dimension group

Dimension groups allow you to organize dimensions into groupings for presentation purposes.

Note: A dimension group must exist before you can add a dimension to it.

To add a dimension to an existing dimension group:

1. In the Dimensions view, double-click the dimension you want to change to open it in the Dimension editor.
2. Select the group you want to add the dimension to from the Member of This Dimension Group list.
3. Click **OK**.

Note: A dimension can belong to only one dimension group.

Mapping a property to a dimension

Map a source property to a dimension to enable navigation on that property.

You must map a source property to a dimension in order to enable navigation on that property. You use the property mapper component to map source properties to dimensions. See the *Endeca Forge Guide* for details.

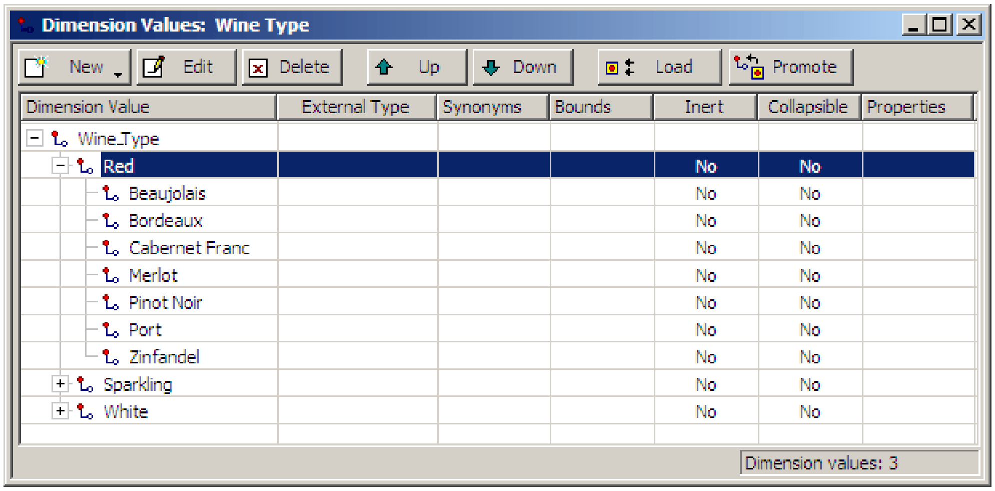
Note: By default, Forge removes source properties that have not been mapped to an Endeca property or dimension; therefore, be sure to create a mapping for a source property if you intend to use it in your Endeca-enabled Web application.

Populating a Dimension with Dimension Values

About viewing dimension values

The Dimension Values view displays hierarchical dimension value information about the selected dimension.

In the example below, the Red dimension value has seven child dimension values: Beaujolais, Bordeaux, Cabernet Franc, and so on. The Sparkling and White dimension values also have children, as indicated by the plus sign (+) next to their names.



Dimension Value	External Type	Synonyms	Bounds	Inert	Collapsible	Properties
Wine_Type						
Red				No	No	
Beaujolais				No	No	
Bordeaux				No	No	
Cabernet Franc				No	No	
Merlot				No	No	
Pinot Noir				No	No	
Port				No	No	
Zinfandel				No	No	
Sparkling				No	No	
White				No	No	

Dimension values: 3

Associating properties with a dimension value

You can provide name/value pairs with descriptive information about a given dimension value.

For example, putting a property that contains a database lookup key on a dimension value will allow your Web application to access an existing database system that stores data related to that dimension value.

Note: Do not confuse these name/value pairs with source properties or Endeca properties. They are purely for descriptive information about a given dimension value.

To associate one or more properties with a dimension value:

1. In the Dimension Value editor, click Properties.

The Properties editor appears.

2. In the Property box, type the new property name.
3. In the Value box, type the property's value.
4. Click Add.
5. (Optional) Repeat steps 2 to 4 to add other name/value pairs.
6. Click OK.

Viewing dimension values for a particular dimension

Open the Dimension Values view from any dimension in the Dimensions view.

To view dimension values for a particular dimension:

1. In the Dimensions view, select a dimension and click Values. The Dimension Values view opens.
2. To expand a node in the dimension hierarchy, click the plus sign (+) next to the node's name. To collapse a node, click the minus sign (-). In addition to buttons that allow you to add, edit, delete, or adjust the rank of dimension values, the Load and Promote buttons support two features: editing of auto-generated dimension values and working with externally-generated taxonomies. This view also contains the following columns:

Column	Description
Dimension Value	Lists dimension values in hierarchical order.
External Type	If the dimension, and its dimension values, were not manually created in Developer Studio, indicates what type of external dimension they are: Auto Gen: The dimension's values are being automatically generated because the match mode in the Property Mapper is set to Auto Generate. External: Indicates that this dimension has been created and is being maintainedin a third-party tool, outside of Developer Studio. Externally managed dimensions and dimension values are read-only. Prop Mapped: Indicates that this dimension, and its dimension values, are being created using the "If no mapping is found, map source properties to Endeca: Dimensions" on the Property Mapper's Advanced tab.
Synonyms	Lists any dimension value synonyms that you have applied.
Bounds	Lists any bounds you have set for a range or sift dimension value.
Inert	Indicates whether a dimension value is non-navigable.
Collapsible	Indicates whether a dimension value is a candidate for collapsing.
Properties	Lists any properties associated with a dimension value.

Note: The Load and Promote buttons support two features: editing of auto-generated dimension values and working with externally-generated taxonomies.

Dimension Value editor

Set dimension value name, type (exact, range, or sift), properties, bounds, synonyms, and specify whether dimension value is inert and/or collapsible.

Option	Description
Name	A unique name for this dimension value. Dimension value names are case sensitive.
Type	Specifies the dimension value's type. A dimension value's type determines how it matches to property values during mapping. • Exact Dimension values of type exact will only match to property values that match one of the synonyms exactly. • Range Dimension values of type range will match to ranges of property values (specified by the upper and lower bounds of the range dimension value). • Sift Auto sifting is an extension to autogeneration which positions newly auto-generated dimension values within an existing "sift" hierarchy. A sift hierarchy is a normal hierarchy that has dimension values of type Sift. Sift dimension values are specified using ranges. Auto-generated dimension values "sift" through the hierarchy according to the ranges they match.
Inert	Check **Inert** if the dimension value is non-navigable.
Collapsible	Check **Collapsible** if the dimension value is a candidate for collapsing.
Bounds	This area is used to describe dimension value bounds for dimension values of type Range or Sift. • Bound type: String, floating point, or integer. • Lower bound: The lower bound of the range. • Upper bound: The upper bound of the range. • Include value in range: Indicates that the bound includes the number that has been entered.
Synonyms	Click to add synonyms for this dimension value.

Option	Description
Properties	Click to associate a descriptive name/value pair with this dimension value. **Note:** Do not confuse these name/value pairs with source properties or Endeca properties. They are purely for descriptive information about a given dimension value.

Working with Manually Created Dimension Values

Adding dimension values

Adding or editing a dimension value affects how records are classified, and which records are available under a given dimension value in the client browser.

To add a dimension value to a dimension:

1. In the Dimensions view, click to select the dimension that you want to add the dimension value to.
2. Click **Values**.
   The Dimension Values view appears.
3. Do one of the following to create the new dimension value:
   - To create a dimension value that is a child of an existing dimension value, select the existing value and click **New** > **Child**.
   - To create a dimension value that is the sibling of an existing dimension value, select the existing value and click **New** > **Sibling**.

   The Dimension Value editor appears.

Configuring dimension values

Set a dimension value's type, navigability, add synonyms, or associate properties to the dimension value.

To configure a dimension value:

1. In the Dimension Value editor, type the dimension value's name in the **Name** text box.
2. In the Type list, choose a dimension value type: **Exact**, **Range**, or **Sift**. A dimension value's type determines how it matches to property values during mapping.

   -

Option	Description
Exact	Dimension values of type exact will only match to property values that match one of the synonyms exactly.
Range	Dimension values of type range will match to ranges of property values (specified by the upper and lower bounds of the range dimension value).
Sift	Auto sifting is an extension to autogeneration which positions newly auto-generated dimension values within an existing 'sift' hierarchy. A sift hierarchy is a normal hierarchy that has dimension values of

Option	Description
	type Sift. Sift dimension values are specified using ranges. Auto-generated dimension values "sift" through the hierarchy according to the ranges they match.

3. (Optional) Check **Inert** if the dimension value is non-navigable.
4. Check **Collapsible** if the dimension value is a candidate for collapsing
5. If you chose **Range** or **Sift** in step 2 above, do the following:
   a) From the Bound Type list, choose one of the following: **String**, **Floating point**, or **Integer**.
   b) In the Lower Bound text box, enter the lower number in the range. If you want the range to include the number you enter, check **Include Value in Range**.
   c) In the **Upper Bound** text box, enter the higher number in the range. If you want the range to include the number you enter, check **Include Value in Range**.
6. Click **Synonyms** to add any dimension value synonyms.
7. (Optional) Click **Properties** to associate any properties to the dimension value.
8. Click **OK**.

Editing dimension values
Make all changes to a dimension value in the Dimension Value editor.

To edit a dimension value:

1. In the Dimension Values view, double-click the dimension value's name to open it in the Dimension Value editor.
2. Make any required changes.
3. Click **OK.**

Deleting dimension values
Remove dimension values and their child dimension values from within the Dimension Values view.

To delete a dimension value:

1. In the Dimension Values view, click the dimension value you want to remove, then click **Delete.**
2. When the confirmation message appears, click **Yes**.

Note: All child dimension values of a deleted dimension value are also removed.

Renaming a dimension value
Rename only a dimension's child dimension values.

You cannot rename a dimension's root dimension value, but you can rename any of its child dimension values.

See Editing dimension values and Deleting dimension values for instructions.

Specifying Dimension Value Synonyms

About synonyms

Synonyms provide a textual way to refer to a dimension value, rather than by ID alone. A dimension value can have multiple synonyms. All synonyms that you assign to dimension values must be unique.

You specify the way each synonym is used by the MDEX Engine with the Search, Classify, and (Display) options:

- Enabling the Search option indicates that this synonym should be considered during record and dimension searches. You can enable search for multiple synonyms, allowing you to create a more robust dimension value for searching.
- Enabling the Classify option indicates that this synonym should be considered when attempting to map a source property value to this dimension value. In order for a source property value to match a dimension value, the dimension value's definition must contain a synonym that:
  - Is an exact text match to the source property value (or, for range and sift dimension values, has a range that includes the property value).
  - Has its Classify option enabled.

  By enabling classification for multiple synonyms, you increase the mapping potential for a dimension value because a source property can map to any of the synonyms that have been marked with Classify. If a synonym does not have its Classify option enabled, it is ignored during mapping, regardless of whether or not it is a text match to a source property value.
- While you can have multiple synonyms for a dimension value, only one synonym can be marked for display. This is the synonym whose text is displayed in your implementation whenever this dimension value is shown. By default, the first synonym you create is set to be displayed, as is indicated by the parentheses around the synonym's name, but you can set any synonym for display in the Synonyms dialog box.

To better understand these three options, consider the following example. This dimension value has an ID of 100 (automatically assigned by Developer Studio) and three synonyms:

```
Dimension Value ID = 100
Synonyms =
2002 SEARCH=enabled CLASSIFY=enabled DISPLAY=yes
'02 SEARCH=enabled CLASSIFY=enabled DISPLAY=no
02 SEARCH=enabled CLASSIFY=enabled DISPLAY=no
```

In this example, dimension searches on any of the following terms would return the dimension value ID 100:

```
2002
'02
02
```

Records that have source property values that match any of the following would be tagged with the dimension value ID 100:

```
2002
'02
02
```

Finally, anytime the dimension value with an ID of 100 is displayed in the implementation, the text used to represent the dimension value is "2002".

Adding synonyms to a dimension value

Configure a synonym to display in the navigation controls, and associate other synonyms with a dimension value.

To add a synonym to a dimension value:

1. In the Dimension Values view, double-click the dimension value's name to open it in the Dimension Value editor.
2. Click **Synonyms**.
   The Synonym editor appears.

3. In the Synonym box, type the synonym that you want to add.
4. If this synonym is to be included in record and dimension searches, check **Search**.
5. If this synonym is to be used during dimension value tagging, check **Classify**.
   This allows you to expand the mapping potential of a dimension value.
6. Click **Add**.
7. If you want this synonym to be the one that is displayed in the navigation controls, click **(Display)**.
   Only this synonym is displayed, regardless of the other synonyms for the dimension value or the original property value for the record.
8. (Optional) Repeat steps 2 through 7 to add additional dimension value synonyms.

   You can only have one synonym marked for display.
9. Click **OK** to return to the Dimension Value editor.

Creating Range Dimension Values

About range dimension values

Configuring a dimension to use range dimension values is useful for data that should be navigated as discrete values.

For example, a price value can be organized into discrete ranges ($0 - $50, $51 - $100, $101 - $200, and so on) with a Price Range dimension.

You can create a hierarchy of range dimension values. Normally the higher levels of a dimension's hierarchy are more general and the lower levels are more specific. For a range dimension, this translates to broader ranges at higher levels and more narrow ranges at lower levels.

This section describes the work necessary to construct range dimensions values and how to configure the pipeline to assign the appropriate range values to records.

Configuring a range dimension value

You must set upper and lower bounds for a range dimension value.

Dimensions that use ranges are very similar in structure to ordinary dimensions. The dimension root and dimension hierarchy are created using the same basic process. The difference is in the configuration of the dimension values, which have a type of Range (as opposed to Exact) and require lower and upper bounds.

To set the bounds for a range dimension value:

1. In the Dimension Value editor, double-click the dimension value you want to change to open it in the Dimension editor.
2. From the Type list, choose **Range**.
3. From the Bound Type list, choose the appropriate data type:
   - Floating point
   - Integer
   - String
4. In the **Lower Bound** text box, enter the lower number in the range. If you want the range to include the value you enter, check **Include Value in Range**.
5. In the **Upper Bound** text box, enter the higher number in the range. If you want the range to include the value you enter, check **Include Value in Range**.
6. If necessary, configure the range dimension value's synonyms.
7. If necessary, add properties to the dimension value.
8. Click **OK** to close the Dimension editor.

Note: Overlap in range values and misaligned ranges in concept hierarchies are not validated by the MDEX Engine. It is your responsibility to correctly configure range dimensions.

Synonym configuration for range dimension values

Synonym configuration for range dimension values is the same as for ordinary dimension values. You can create more than one synonym for any given dimension value (although, you are still restricted to only one displayable synonym per dimension value).

The value of the synonym does not have to correspond to the bounds. For example, a dimension value in a Price range dimension might have a range of [0,10], and the synonyms Under $10 (set for display) and Bargains.

You should configure synonyms for classification and search using the same logic you would for ordinary dimensions. For example, if an end-user searches for Bargains, it would make sense to return the Under $10 dimension value from the Price dimension. In this case, you would configure Under $10 for display while configuring Bargains as searchable.

In most cases, synonyms for range dimension values should have their Classify option disabled so that only the bounds of the range itself are used for record classification and tagging. There are some cases where it is useful to combine range matching on the bounds with exact matching on the synonyms. See "Combining exact and range matching" for details.

Note: For more general information about synonyms, see "Adding synonyms to a dimension value".

Classifying records with ranges

This type of matching is called range matching (as opposed to exact matching where the source property value and the dimension value must be identical).

Matching between a source property value and a range dimension value is a two-step process:

1. The source property value is converted to the type that is used for the bounds values.
2. Forge computes which range the source property value falls into, and then tags the record with that range's dimension value ID.

You configure mapping between a source property and a range dimension as you would with non-range dimensions (see Establishing a dimension mapping). When Forge maps a source property to a dimension that contains range dimension values, it uses range matching for both Normal and Must Match modes. For the AutoGen mode, Forge still uses range matching, but when a property value does not match any ranges, a new exact match dimension value is generated, not a range dimension value.

See Choosing a match mode for more information on Normal, Must Match, and AutoGen match modes.

Combining exact and range matching

A single dimension can contain both exact and range dimension values. During source property mapping, this type of dimension participates in both exact matching, for the exact dimension values, and range matching, for the range dimension values.

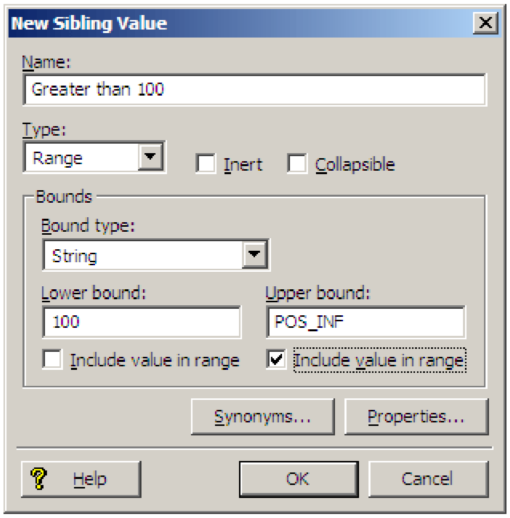
If a range dimension value has synonyms that are enabled for classification, it will participate in both exact matching (for the synonyms) and range matching (for the bounds of the range). The ability to interweave ranges and classifiable synonyms allows you to match heterogeneous property values to a single range dimension.

For example, consider a wine rating property that is mapped to a Wine Rating dimension. Some of the source records have numeric ratings (56, 75, 92, and so on) while others have verbal ratings (poor, good, excellent). Using the technique described above, you can create a range dimension value that has bounds of 90 to 100 and a classifiable synonym for 'excellent' so that all records with numeric ratings above 90 are classified the same as records with 'excellent' ratings.

Using NEG_INF and POS_INF

Specify either of these values as the lower or upper bound for a dimension value, to indicate a value less than or greater than all other values in the range.

You can use two special values, NEG_INF and POS_INF, when creating the bounds for your dimension values. NEG_INF indicates less than all other values while POS_INF indicates greater than all other values. For example, to specify a range of greater than 100, you would use a lower bound of 100 and an upper bound of POS_INF.



New Sibling Value

Name: Greater than 100

Type: Range ☐ Inert ☐ Collapsible

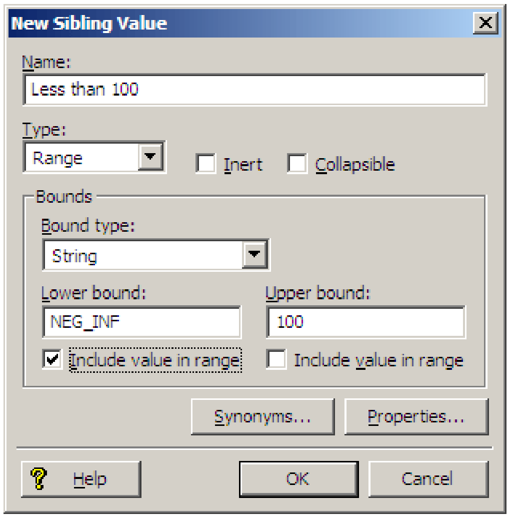
Bounds

Bound type: String

Lower bound: 100 Upper bound: POS_INF

☐ Include value in range ☑ Include value in range

Likewise, less than 100 would use a lower bound of NEG_INF and an upper bound of 100.



New Sibling Value

Name: Less than 100

Type: Range ☐ Inert ☐ Collapsible

Bounds

Bound type: String

Lower bound: NEG_INF Upper bound: 100

☑ Include value in range ☐ Include value in range

You can use POS_INF and NEG_INF with string values as well. For example, setting a lower bound of S, inclusive, and an upper bound of POS_INF, inclusive, would match all strings starting with S and going to the end of the alphabet, including values such as S, Style, Trigger, and Zzzzz.

A note about locale and encoding

The order of symbols depends on your locale setting, which is external to the Endeca software. On UNIX, it is determined by a set of environment variables, typically LOCALE, LANG, or LC_ALL. On Windows, there are separate system and user locales which can be set from the Regional and Language Options control panel. For example, in ASCII, using [NEG_INF, A) as the bounds includes all numerics and many symbols (the '[' symbol indicates the value is inclusive while ')' indicates it is not). Using (Z, POS_INF] includes the rest of the symbols, as well as lower-case letters. This is not the case for other encodings, such as Unicode, which intersperses symbols and numbers with letters much more than ASCII. To use NEG_INF and POS_INF effectively, you must have a good understanding of the order of symbols in your locale's encoding.

Creating an "other" dimension value

When working with range and sift dimension values, it can be useful to have an Other dimension value to capture any source property values that don't fall within the defined ranges.

To catch source property values outside of your defined ranges, you must use a Perl manipulator, placed after the property mapper, to look at each record that has been mapped, and determine if it has any dimension values assigned to it from the range/sift dimension. If not, the Perl manipulator should assign the Other dimension value to the record.

Troubleshooting range dimension values

Some reasons for issues may include the way a dimension value's bounds are configured, or assignment of a dimension value to an incorrect range.

The following information will help you troubleshoot range dimension value issues:

- The displayable synonym for a range dimension value provides the text displayed in your Endeca-enabled Web application’s user interface. The bounds of the range are independent of this text. It is the developer's responsibility to correctly align a range dimension value's synonym text with its bounds.
- Particular attention should be paid to correctly configuring the inclusion or exclusion of bounds values. (This is controlled by the Include Value in Range check boxes in the Bounds frame.) Incorrectly setting this field can lead to unwanted bounds overlap or holes in ranges. For example, say you have one range from 0 to 10, and another from 10 to 20. If you included the value of 10 in both ranges, your ranges would overlap (that is, the value 10 would match both ranges). Conversely, if you had the same ranges (0 to 10 and 10 to 20) but did not include the bounds values in either range, you would have a hole (that is, the value 10 would match neither range).
- Ranges, by nature, have an order. However, this order is not preserved by the Presentation API. You can maintain the correct order by manually ranking each dimension value.
- It is valid to create a range dimension containing dimension values with overlapping bounds. In performing the match, all dimension values that bound the property value will match successfully and will be assigned to the record.
- If a dimension value cannot be precisely represented as an IEEE double, the dimension value might get assigned to an incorrect range.

Providing additional descriptive information for a dimension value

You can provide name/value pairs with descriptive information about a given dimension value. For example, associating a key/value pair that contains a database lookup key with a dimension value will allow your Web application to access an existing database system that stores data related to that dimension value.

Note: Do not confuse these name/value pairs with source properties or Endeca properties. They are purely for descriptive information about a given dimension value.

To associate one or more properties with a dimension value:

1. In the Dimension Value editor, click **Properties**.
   The Properties editor appears.
2. In the Property box, type the new property name.
3. In the Value box, type the property's value.
4. Click **Add**.

5. (Optional) Repeat steps 2 to 4 to add other name/value pairs.
6. Click **OK**.

Ranking Dimension Values

Controlling the order of dimension values

If a dimension's hierarchy has been created manually, you should also rank its dimension values manually. Default dimension value ranking is used with dimensions that are auto-generated.

There are two ways to control the order in which dimension values are returned. You can:

- Manually rank dimension values.
- Specify a default rank order.

In an auto-generated dimension, you generally don't have direct access to and, hence, can't manually rank, the dimension values (see the notes below for an exception to this statement).

Note:

- You can load auto-generated dimension values and then rank them manually. See "Editing auto-generated dimension values" for details.
- If dimension values are assigned ranks with values greater than 16,000,000, unpredictable ranking behavior may result.
- A related feature allows you to prune the list of refinement dimension values returned for a query to those values that occur most frequently in the requested navigation state. See "Pruning dimension value refinements by frequency of occurrence" for more information.
- For information on ranking dimensions, see "Ranking dimensions manually".

Ranking dimension values manually

Manual dimension value ranking defines the order in which dimension values appear in your Web application and overrides any default dimension value ranking you have specified.

A dimension value is only ranked relative to its siblings (that is other dimension values at the same level of hierarchy within the same dimension).

To manually rank dimension values:

1. If necessary, in the Dimension Values editor, expand the dimension hierarchy to display the dimension value you want to rank.
2. Click the dimension value to select it.
3. Click **Up** to move the value up in rank, or **Down** to move it down, within its own level of hierarchy.
4. Continue moving dimension values as necessary.

Note: The order in which the dimension values appear in Dimension Values view will be the order in which they appear in your application.

Note:

- You can load auto-generated dimension values and then rank them manually. See "Editing auto-generated dimension values" for details.
- Use of ranked dimensions and dimension values does not affect MDEX Engine performance. However, indexing time is slightly increased by heavy use of this feature.
- Configuring a dimension so that its dimension values are pruned according to their popularity overrides any manual or default dimension value ranking you may have specified.

Setting a default dimension value order

In an auto-generated dimension, you don't have direct access to and, hence, can't manually rank, the dimension values. Instead, you must set a default rank order.

Default dimension value ranking is used with dimensions that are auto-generated.

Note: The paragraph above describes Developer Studio's default behavior with respect to auto-generated dimensions. Developer Studio also offers features that allow you load and/or promote an auto-generated dimension in Developer Studio so that you can edit it, including setting manual ranking for its values.

To set a default dimension value rank order:

1. In Dimension view, double-click the dimension you want to edit to open it in the Dimension editor.
2. From the Refinements sort order menu, choose one of the following:
   - **Alpha**, to rank the dimension in ascending alphabetical order.
   - **Integer**, to rank the dimension in ascending numerical order.
   - **Floating point**, to rank the dimension in ascending numerical order.
3. Click **OK**.

To better understand the difference between the dimension value ordering types, consider an example of a dimension called Score that has the dimension values 1, 5, 5.5, 9, and 10. The following table shows what the ordering would be for each type.

Original dimension	Alpha ordering	Integer ordering	Floating point ordering
-- Score	1	1	1
-- 1	10	5	5
-- 5	5	5	5.5
-- 5.5	5.5	9	9
-- 9	9	10	10
-- 10			

With integer ordering, the value 5.5 has been truncated to 5, and it is unclear which 5 is the original version and which is the truncated version. For some applications this may be acceptable, for others it is not. Additionally, integer ordering is significantly faster than floating point ordering. When choosing a numeric ordering type, you must balance the needs of your application against the extra time it takes to use floating point ordering.

Note:

- The Refinements Sort Order setting has no bearing on how records are sorted. It only controls how refinement dimension values are sorted.
- Configuring a dimension so that its dimension values are pruned according to their popularity overrides any manual or default dimension value ranking you may have specified.

Manual versus auto-generated dimension values

To configure multiple synonyms, collapsibility, or inertness, or to create a hierarchy for dimension values, you must manually populate a dimension. Otherwise, set a dimension to auto-generate its values.

Every dimension must be populated with dimension values. There are two ways to do this:

- Manually create the dimension values, and any dimension value hierarchy, using the Dimension Values editor.
- Configure the dimension to have its dimension values automatically generated from the source data. Unlike manually created dimension values, which are defined during the dimension editing process, auto-generated dimensions must be configured when you create your source property-to-dimension mappings in your property mapper component.

You must populate a dimension manually if any of the following is true:

- You want to create a hierarchy for the dimension values (auto-generated dimensions are always flat).
- You want to configure specific dimension value behavior such as multiple synonyms, collapsibility, or inertness.

If a dimension does not have either of these requirements, you can set it to auto-generate and avoid the labor involved in manually entering dimension values.

The following section, "Working with manually created dimension values," describes how to create, edit, and configure manually created dimension values. See Choosing a match mode for details on automatically generating dimension values.

Creating Non-navigable Dimension Values

Making a dimension value non-navigable (inert)

Marking a dimension value as inert (or non-navigable) indicates that the dimension value should not be included in any navigation state, although it can be displayed in a user interface to help guide the end-user toward a selection.

When a user selects an inert dimension value, the navigation state is not changed, but the children of the dimension value are displayed for selection.

For example, if a Wineries dimension contains 1000 wineries and there is no geographic information from which to create a meaningful hierarchy, you can create a non-navigable alphabetical hierarchy. The first set of refinements returned for the Winery dimension would be the non-navigable refinements (such as A, B, C, and so on). When a user selects 'A,' the resulting query returns the same record set but the winery refinements are limited to those wineries whose name begins with A.

Note: For a common inert dimension use case scenario, see A typical scenario.

To make a dimension value non-navigable:

1. In the Dimension Values view, double-click the name of the dimension value that you want to change to open it in the Dimension Value editor.
2. Click **Inert.**
3. Click **OK.**

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

A typical scenario

This describes a typical scenario for setting dimensions and dimension values.

A typical scenario involves combining three features in one dimension:

- Create a sift dimension with a set of discrete ranges such as A - D, E - H, and so on. Auto sifting is an extension to autogeneration which positions newly generated dimension values within an existing hierarchy, such that Applebrandy Wine appears under A - D, Four Grapes Merlot appears in E - H, and so on.
- Configure the dimension values in the sift dimension as non-navigable so that they can appear in the user interface to assist the end user but do not have any effect on the navigation state.
- Configure the dimension values in the sift dimension so that they are collapsible. A collapsible hierarchy is an ordinary hierarchy, in which some or all of the internal (non-root and non-leaf) dimension values are flagged as potentially collapsible. The MDEX Engine automatically removes, or collapses, these dimension values when there are only a few leaves available for refinement, creating a more streamlined, user-friendly navigation experience for your users.

Pruning dimension value refinements by frequency of occurrence

You can prune the list of refinement dimension values returned for a query to those values that occur most frequently in the requested navigation state.

You can limit the number of frequently-occurring (popular) refinements returned, as well as control the order in which they are returned. Note that configuring a dimension so that its dimension values are pruned according to their popularity overrides any manual or default dimension value ranking you may have specified.

To prune dimension value refinements according to their popularity:

1. In Dimensions view, double-click the dimension you want to edit to open it in the Dimension editor.
2. Click the **Dynamic Ranking** tab.
3. Click **Enable Dynamic Ranking** to specify that this dimension should calculate which refinements are most popular.
4. Type the number of popular refinements to return in the Maximum Dimension Values to Return box. The default value is 10.
5. Choose a method for sorting the popular refinements:
   - Alphabetically uses whatever order you've selected for the Refinements Sort Order setting on the main part of the Dimension editor.
   - Dynamically orders the most popular refinement values according to their frequency of appearance within a data set. Dimension values that occur more frequently are returned before those that occur less frequently.
6. (Optional) click **Generate "More..." Dimension Value**.
   When this option is checked, if the actual number of refinement options exceeds the number set in Maximum Dimension Values to Return, then an additional option called More is returned for that dimension. If the user selects the More option, then the MDEX Engine will return all of the refinement options for that dimension. If Generate "More..." Dimension Value is not checked, only the number of dimension values defined in Maximum Dimension Values to Return is displayed.
7. Click **OK** to close the Dimension editor.

Ranking dimension values manually

Manual dimension value ranking defines the order in which dimension values appear in your Web application and overrides any default dimension value ranking you have specified.

A dimension value is only ranked relative to its siblings (that is other dimension values at the same level of hierarchy within the same dimension).

To manually rank dimension values:

1. If necessary, in the Dimension Values editor, expand the dimension hierarchy to display the dimension value you want to rank.
2. Click the dimension value to select it.
3. Click **Up** to move the value up in rank, or **Down** to move it down, within its own level of hierarchy.
4. Continue moving dimension values as necessary.

   Note: The order in which the dimension values appear in Dimension Values view will be the order in which they appear in your application.

Note:

- You can load auto-generated dimension values and then rank them manually. See "Editing auto-generated dimension values" for details.
- Use of ranked dimensions and dimension values does not affect MDEX Engine performance. However, indexing time is slightly increased by heavy use of this feature.
- Configuring a dimension so that its dimension values are pruned according to their popularity overrides any manual or default dimension value ranking you may have specified.

Editing auto-generated dimension values

Load an auto-generated dimension and then promote its dimension values to make them editable.

Normally, auto-generated dimension values cannot be edited. They are generated by Forge behind the scenes and maintained in state files. With an auto-generated dimension, you can configure the dimension's behavior, but you cannot configure the behavior of individual dimension values within the dimension. Endeca's load and promote functionality, however, allows you to load an auto-generated dimension and then promote its dimension values so that they become editable.

Loading versus promoting

The process of converting an auto-generated dimension has been broken down into two distinct steps, loading and promoting. Loading displays the auto-generated dimension values so that you can inspect them before promoting them. In addition, loaded dimension values can be used in the following ways. You can:

- Use them in dynamic business rules.
- Use them in precedence rules.
- Modify their ranking.
- Enable/disable their inert and Collapsible features.

After loading a dimension, you have the option of promoting its dimension values. Promoting a dimension's values converts them to manual dimension values, with all of the editing capability of a regular manually created dimension value. Promotion is done on a per dimension basis. In other words, when you promote a dimension, all of its dimension values are promoted; you cannot pick individual dimension values to promote and leave others to be auto-generated. It is important to note that, after promotion, you can no longer treat a promoted dimension as auto-generated. All configuration and editing must be performed manually at this point.

Loading and promoting requirements

Loading and promoting auto-generated dimensions has two requirements:

- You must use Endeca Workbench and EAC. Endeca Workbench stores temporary copies of auto-generated dimensions. This is the location that Developer Studio retrieves them from during loading.

- A dimension must be auto-generated before it can be loaded and/or promoted. This means that you must have already run Forge at least once before attempting to load and promote auto-generated dimensions.

Cleaning up after promoting dimensions

After you promote auto-generated dimension values, you must run a baseline update with Forge's

```
--pruneAutoGen
```

flag. The flag cleans out any promoted dimensions from the auto-generated state files. This step is necessary in order to avoid any potential duplicate dimensions in your output records.

Loading and promoting an auto-generated dimension

Load and promote an auto-generated dimension from within the Dimensions view.

- You must use Endeca Workbench and EAC. Endeca Workbench stores temporary copies of auto-generated dimensions. This is the location that Developer Studio retrieves them from during loading.
- A dimension must be auto-generated before it can be loaded and/or promoted. This means that you must have already run Forge at least once before attempting to load and promote auto-generated dimensions.

To load and promote an auto-generated dimension:

1. In Dimensions view, select an auto-generated dimension. Auto-generated dimensions are indicated by this icon:
2. Click **Values** to display the Dimension Values view.
   You see the root of the auto-generated dimension.
3. Click **Load**.
   The dimension is populated with the auto-generated dimension values.
4. Click **Promote**.
5. Click **Yes** to confirm the promotion.
6. Click **OK**.
   The icons next to the promoted dimension values change to indicate that they are now treated as manual dimensions:

Cleaning up after promoting dimensions

Run a baseline update to remove promoted dimensions from auto-generated state files.

After you promote auto-generated dimension values, you must run a baseline update with Forge's

```
--pruneAutoGen
```

flag. The flag cleans out any promoted dimensions from the auto-generated state files. This step is necessary in order to avoid any potential duplicate dimensions in your output records.

To clean up after promoting auto-generated dimensions:

1. Make sure the `--pruneAutoGen` flag is specified for Forge. (This flag is specified by default for new projects)
2. Run a baseline update.

Configuring Dimension Behavior

Enabling and Disabling Display

Enabling dimension display in a record list

You must enable a dimension for record list display before you can access its dimension values using the methods in the Endeca Presentation API.

When record list display is enabled for a dimension, any records that are tagged with dimension values from that dimension display those values as part of their entry in the record list.

To enable record set display:

1. In the Dimensions view, double-click the dimension you want to change to open it in the Dimension editor.
2. On the General tab, check Show with Record List.
3. Click **OK.**

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Enabling dimension display on a record page

When record page display is enabled for a dimension, any records that are tagged with dimension values from that dimension display those values as part of their entry on the record page.

Note: You must enable a dimension for record display before you can access it using the methods in the Endeca Presentation API.

To allow the dimension value for this dimension to appear on record pages:

1. In the Dimensions view, double-click the dimension you want to change to open it in the Dimension editor.
2. On the General tab, check Show with Record.
3. Click **OK.**

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Preventing a dimension from appearing in the navigation controls

Configuring dimensions that are composed of many dimension values as hidden improves Presentation API and MDEX Engine performance to the extent that navigation query results do not have to include these large dimensions, reducing the processing cycles and amount of data the MDEX Engine must return.

You prevent a dimension from appearing in the navigation controls by designating it a hidden dimension. Hidden dimensions, like regular dimensions, are composed of dimension values that allow the user to refine a set of records. The difference between regular dimensions and hidden dimensions is that regular dimensions are returned for both navigation and record queries, while hidden dimensions are only returned for record queries. This means that hidden dimensions cannot be displayed as part of your navigation controls, but can be displayed as part of a record page (assuming the hidden dimension is configured to render on the record page).

Also, although hidden dimensions are not rendered in the navigation UI, records are still indexed with relevant values from these dimensions. Therefore, an end-user can search for records based on values within hidden dimensions.

To configure a hidden dimension:

1. In the Dimensions view, double-click the dimension you want to change to open it in the Dimension editor.
2. On the General tab, check Hidden.
3. Click **OK.**

Hidden dimension example

Marking a dimension as hidden is useful in cases where the dimension is composed of numerous dimension values, and returning these values as navigation options does not add useful navigation information. Consider, for example, an Authors dimension in a bookstore. Scanning thousands of authors for a specific name is less useful than simply using keyword search to find the desired author.

In this case, you can configure the Authors dimension as hidden. The end-user will be able to perform a keyword search on a particular author but will not be able to browse on author names in order to find books by the author. Once the end-user has browsed to the record page for a particular book-either by keyword search or by navigating within other dimensions-he or she may be interested in other books by the same author. Because the hidden dimension is included in the record query results, the user can formulate a new navigation query, including the hidden dimension, that returns a list of books by that author. This process, in effect, creates a store of books by the same author.

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Configuring Record Sort

Specifying a default record sort order

If an Endeca-enabled Web application does not specify sort order as part of the query, the MDEX Engine returns query results using the default sort order, if one has been specified.

You specify the default sort order and sort direction (ascending or descending) by using the `--sort` flag when running Dgidx. The `--sort` flag has the following syntax:

```
--sort "key|dir"
```

where key is the name of an Endeca property or dimension on which to sort and dir is either asc for an ascending order or desc for descending (if not specified, the order will be ascending).

You can also specify multiple sort keys in the format:

```
--sort "key_1|dir_1||key_2|dir_2||...||key_n|dir_n"
```

If you specify multiple sort keys, the records are sorted by the first sort key, with ties being resolved by the second sort key, whose ties are resolved by the third sort key, and so on.

Note: If you are using the Endeca Application Controller (EAC) to control your environment, you must omit the quotation marks from the --sort flag. Instead, use the following syntax:

If you are using the Endeca Application Controller (EAC) to control your environment, you must omit the quotation marks from the --sort flag. Instead, use the following syntax:

```
--sort key_1|dir_1||key_2|dir_2||...||key_n|dir_n
```

Agraph default sort order and displayed record lists

If a default record sort order is specified in Dgidx based on a property which is not set to show in the record list, an Agraph managing the resulting Dgraphs will not consistently display records in the default sort order.

Each child Dgraph displays its own records in the correct order, but the Agraph does not reliably preserve this order when integrating its child record sets. The resulting record order will be close to—but not the same as—the actual specified default sort order.

To prevent this problem, use Developer Studio's Property editor to enable the 'Show with record list' setting for the sort property. This ensures that the Agraph will determine the correct record display order.

Precomputing sort indices on a dimension

Precomputing sort can save query time.

To precompute sort indices on a dimension:

1. In the Dimensions view, double-click the dimension you want to change to open its Dimension editor.
2. On the General tab, check **Prepare Sort Offline**.
3. Click **OK**.

Note:

- An explicit record sort key, specified as part of a MDEX Engine query, takes priority over any other type of record sorting (default sorting and relevance ranking). See "Controlling the order of record results" for details.
- If the Web application does not specify sort order as part of the query, the MDEX Engine returns records in the default sort order, if one has been specified. See "Specifying a default record sort order" for details.
- Record sorting only affects the order of records. It does not affect the ordering of dimensions or dimension values that are returned for query refinement. You use dimension and dimension value ranking to affect the order of dimensions and dimension values.

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Relevance ranking versus record sort
Relevance ranking is used to control the order of results that are returned in response to a keyword search. Record sorting is used to control the order of records that are returned in response to any type of MDEX Engine query that returns records.

Relevance ranking and record sorting are closely related features but there are some distinct differences.

- Relevance ranking determines which results are more relevant to the user, based on a set of rules you define. For example, you can configure a rule that says "for multi-term searches, rank records that match more of the terms higher than those that match fewer terms." Relevance ranking is configured either as part of a search interface, where each search interface has its own relevance ranking strategy, or is specified in the record search query itself.
- Unlike relevance ranking which is limited to keyword search queries, record sorting can be used with any type of query that returns records. Record sorting is based on a sort key. The sort key can either be defined as a default, or identified by the Web application as part of the query.

Generally, if you have relevance ranking enabled, you would not specify a record sort key within a record search query because record sort keys take priority over all other types of ordering, making the relevance ranking settings useless.

Note: A search interface is a named collection of properties and dimensions, each of which has its Enable Record Search option checked. Search interfaces allow your end-users to search on multiple properties and/or dimensions simultaneously. The search interface's name is used just like a normal property or dimension when performing record searches. A record search query on a search interface returns results that match any of the properties or dimensions in the interface.

Controlling the order of record results
Specify an explicit sort key in the MDEX Engine query, set a default sort order, or use relevance ranking (for records returned in response to record search queries).

There are three ways of controlling the order in which records are returned:

- Specifying an explicit sort key in the MDEX Engine query, either via a URL parameter (Ns) or an ENEQuery setter method (setNavActiveSortKeys())
- Specifying a default sort order (Dgidx option).

- Using relevance ranking, for records returned in response to record search queries only. Relevance ranking settings can either be implicitly defined as part of the search interface, or explicitly defined in the search query itself.

The priority of record sorting/relevance ranking is as follows:

- If none of these three sorting methods is specified, records are returned in an arbitrary, but consistent, order as determined by an internal ID generated by Dgidx during indexing.
- If the MDEX Engine query includes an explicit sort key parameter, that sort key overrides all other sorting and relevance ranking settings.
- If a default sort key is specified, and no other sort parameters are set, records are returned in default sort order. Ties are broken using the arbitrary internal order described above.
- When searching against a search interface that incorporates a relevance ranking strategy, the relevance ranking strategy takes priority but ties are broken using the default sort key, if one has been specified. If there is no default sort key, ties are broken using the internal ID order described above.
- If the MDEX Engine query includes a relevance ranking parameter, that setting overrides any relevance ranking strategies configured in the search interface that is being searched against.

A search interface is a named collection of properties and dimensions, each of which has its Enable Record Search option checked. Search interfaces allow your end-users to search on multiple properties and/or dimensions simultaneously. The search interface's name is used just like a normal property or dimension when performing record searches. A record search query on a search interface returns results that match any of the properties or dimensions in the interface.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Troubleshooting record sorts

To fix issues with record sorts, check for property type, number of values assigned to each record, and uniqueness of property and dimension names.

If the records returned with a navigation request do not seem to respect the sort key parameter, here are some things to check:

- Was the Endeca property specified as a numeric when it is actually alphanumeric, or vice versa? In this case, the MDEX Engine returns a valid response, but the sorting may not be what you expected. Also, if you are sorting by a dimension, then the sort is always alphabetic.
- In general, properties and dimensions that are enabled for sorting should only have one value assigned per record. If a record has multiple property values or dimension values for a single Endeca property or dimension, the MDEX Engine sorts the records based on the first value associated with the key. If the application is displaying any value other than the first one, then the records may not appear to be sorted correctly.
- If an application has properties and dimensions with the same name and a sort is requested by that name, the MDEX Engine will arbitrarily pick either the property or dimension for sorting. In general, all properties and dimensions should have a unique name.

Enabling Search

Enabling a dimension for record search

Record search finds all records in an Endeca application that are tagged with a dimension value or Endeca property that matches a term the user provides. In order for a dimension to be considered during record searches, you must enable it for record search.

To enable record search for a dimension:

1. In the Dimensions view, double-click the dimension you want to change to open it in the Dimension editor.
2. On the Search tab, check **Enable Record Search**.

3. Click **OK.**

Fully implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Dimension Search Configuration editor
You use the Dimension Search Configuration editor to configure search options for all dimensions in your project.

The Dimension Search Configuration editor does not specify the same options as the Search tab of the Dimension editor. The Search tab of the Dimension editor affects record searches.

The Dimension Search Configuration editor specifies the following information:

Option	Description
Enable positional indexing	When checked, indicates that Dgidx builds an index that describes word position with respect to other words in a dimension. A word position index is highly recommended when using the relevance ranking Phrase module and also recommended to improve the speed of phrase search.
Enable wildcard search	When checked, indicates that a user query can contain a wildcard character (*) to match against fragments of words in a dimension value.
Return highest ancestor dimension	When checked, the results of a dimension search return only the highest ancestor dimension value. This means that if both red zinfandel and red wine match a search query for 'red' and this checkbox is enabled, only the red wine dimension value is returned. When unchecked, then both dimension values are returned.
Include inert dimension values	When checked, indicates that certain inert (non-navigable) dimension values, such as dimension roots, are also returned as the result of a dimension search query.

Expanding search to include dimension hierarchy
For both record and dimension search, you can expand the search so that a dimension value's search context includes not only its own searchable synonyms but also any searchable synonyms of its ancestors. This type of expanded search is called "hierarchical search."

Record search and dimension search each have their own option to enable this feature.

Hierarchical search during record searches

Consider an example where Record A is tagged with Merlot from the Wine Type dimension below, and the end-user has searched for "red." Without hierarchical search enabled, Record A would not be returned as part of the search results. With hierarchical search enabled, it would.

Wine Type

- Red

- Zinfandel
  - Merlot

Hierarchical search during dimension searches

For example, in the Wine Type dimension below, a search on " red" would return both Zinfandel and Merlot, in addition to Red, with hierarchical dimension search enabled.

Wine Type

- Red
  - Zinfandel
    - Merlot

Using hierarchical search during dimension searches
Normally, dimension search considers only the text in individual dimension value synonyms when searching for dimension values that match a user's search term(s). With hierarchical search enabled, the MDEX Engine considers ancestor dimension values as well.

To enable hierarchical search during dimension searches:

1. In the Dimensions view, double-click the dimension you want to change to open it in the Dimension editor.
2. On the General tab, check **Search Hierarchy for Dimension Search**.
3. Click **OK.**

Fully implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Using hierarchical search during record searches
Normally, record search only returns a record if the specific dimension value tagged to the record matches the search term(s). When hierarchical search is enabled, the MDEX Engine considers ancestor dimension values as well.

To enable hierarchical search during record searches:

1. In the Dimensions view, double-click the dimension you want to change to open it in the Dimension editor.
2. On the Search tab, check **Search Hierarchy for Record Search**.
3. Click **OK.**

Fully implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Using Range Filtering

Using range filters
A range filter allows an Endeca-enabled Web application to select a subset of the total dataset for display, based on an arbitrary, dynamic range that uses an Endeca property or dimension as the filter key.

Navigation queries that use a range filter return only those records that are included in the selected data subset, along with the refinement dimension values that are appropriate for the filtered records. Range filters are supported for:

- Properties of type integer, floating point, geocode, datetime, time, or duration.
- Dimensions of type Integer or Floating Point.

For values of properties and dimensions of type Floating point, you can specify values using both decimal (0.00...68), and scientific notation (6.8e-10).

It is important to remember that range filters are simply modifiers for a navigation query. The range filter acts in the same manner as a dimension value, even though it is not a specific system-defined dimension value. Consider the following records and examples:

Record ID	Sample Dimension Value (Wine_Type)	Sample Property (Price)	Sample Property (Description)
1	Red (Dim Value 101)	10	Dark ruby in color, with extremely ripe…
2	Red (Dim Value 101)	12	Dense, rich and complex describes this '96 California…
3	White (Dim Value 102)	19	Dense and vegetal, with celery, pear and spice flavors…
4	Other (Dim Value 103)	20	Big, ripe and generous, layered with honey…

Example 1

When making a navigation request with a range filter specifying that the Price property is greater than 15 (with no dimension values specified) the following Navigation object is returned:

- 2 records (records 3 and 4)
- 2 refinement dimension values (White and Other)

Example 2

When making a navigation request with the Red dimension value specified (Dim Value 101) and a range filter specifying a Price less than 11, the following Navigation object is returned:

- 1 record (record 1)
- (No additional refinements)

Valid functions that can be used with range filters include:

- Less than
- Less than or equal to
- Greater than
- Greater than or equal to
- Between

Enabling a dimension for range filtering

Make sure your dimension is of Numeric type to enable it for range filtering.

In order to use a dimension as the key for a range filter, you must make sure your dimension is of a compatible type. Range filters are supported for dimensions of Numeric type with Integer or Floating point values.

You may also use an Endeca property as the key for a range filter.

To enable a dimension for range filtering:

1. In the Dimensions view, double-click the dimension you want to change to open it in the Dimension editor.
2. In the Refinements Sort Order list, select a type that is compatible with range filters: Integer or Floating point.

For values of dimensions of type Floating point, you can specify values using both decimal (0.00...68), and scientific notation (6.8e-10). Be careful of dollar signs or other characters in dimension values that would prevent a dimension from being defined as numeric.

3. Click **OK.**

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Enabling Wildcard Search

Enabling wildcard search for a dimension

Wildcard searching allows user queries that contain a wildcard character (*) to match against fragments of words in dimension and its child dimension values.

You can either enable each dimension that you want available for wildcard searching or you can enable all dimensions using the Dimension Search Configuration editor.

To enable wildcard searching for a dimension:

1. In the Dimensions view, double-click the dimension you want to change.
2. In the Dimension editor, select the **Search** tab.
3. Check **Enable Record Search**.
4. Check **Enable Wildcard Search**.
5. Click **OK.**

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Ranking Dimensions

Ranking dimensions manually

Always rank dimensions manually, by placing them in ranked order in the Dimensions view. The MDEX Engine will return dimensions in this order.

Note: The use of manually ranked dimensions does not affect MDEX Engine performance. However, indexing time is slightly increased by heavy use of this feature.

To rank your dimensions:

1. In the Dimensions view, click the **Name** column header to cycle the order of the dimension sort so that the dimensions are displayed in rank order (you'll see Sort by Rank in the lower left corner of the Dimensions view).
2. Select the dimension whose rank you want to change.
3. Click the **Up** or **Down** buttons to move the dimension to the correction position.

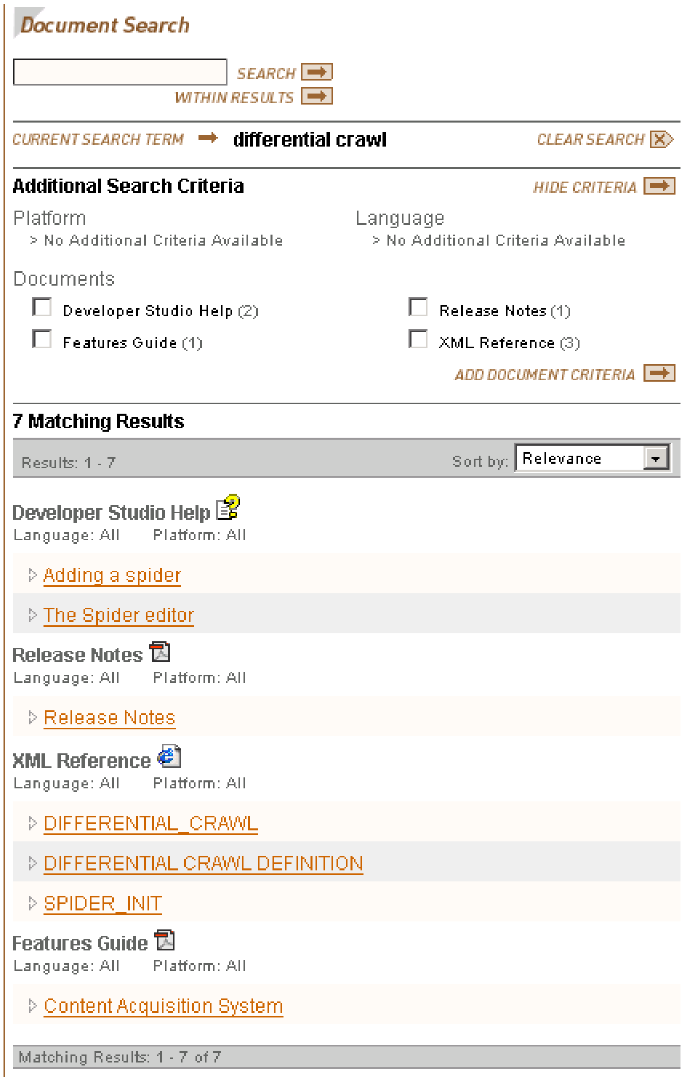
Counting the Number of Records Per Dimension Value

About dimension statistics

Dimension statistics count the number of records, for a given result set, that are tagged with each of a dimension's dimension values.

Giving the end-user an indication of the number of records that will be returned for each refinement enhances an Endeca application's navigation experience by providing more context to the end-user at each point during the navigation. In the example illustration, the query for differential crawl has seven results. Within the Documents dimension, two of the results are tagged with Developer Studio Help, one is tagged with Features Guide, one is tagged with Release Notes, and three are tagged with XML Reference.

Dimension statistics are returned as a property on each dimension value. See the *Endeca Basic Development Guide* for details on accessing and displaying dimension statistics properties.



Document Search

SEARCH

WITHIN RESULTS

CURRENT SEARCH TERM → differential crawl CLEAR SEARCH

Additional Search Criteria HIDE CRITERIA

Platform
> No Additional Criteria Available

Language
> No Additional Criteria Available

Documents

□ Developer Studio Help (2)
□ Release Notes (1)
□ Features Guide (1)
□ XML Reference (3)

ADD DOCUMENT CRITERIA

7 Matching Results

Results: 1 - 7 Sort by: Relevance

Developer Studio Help
Language: All Platform: All

- Adding a spider
- The Spider editor

Release Notes
Language: All Platform: All

- Release Notes

XML Reference
Language: All Platform: All

- DIFFERENTIAL_CRAWL
- DIFFERENTIAL CRAWL DEFINITION
- SPIDER_INIT

Features Guide
Language: All Platform: All

- Content Acquisition System

Matching Results: 1 - 7 of 7

Enabling refinement statistics

You enable refinement statistics on a per dimension basis. The Endeca MDEX Engine dynamically computes dimension statistics at run-time.

To enable statistics for a dimension:

1. In the Dimensions view, double-click the dimension you want to change to open it in the Dimension editor.
2. On the Advanced tab, check **Compute refinement statistics**.
3. Click **OK**.

Fully implementing this feature requires additional work outside of Developer Studio. For details, refer to "Displaying refinement statistics" in the *Endeca Basic Development Guide*.

Enabling Record Rollups

About aggregated Endeca records

Aggregated records allow you to treat a collection of separate records as one if the rollup key is the same for any number of records.

An aggregated record is a collection of individual Endeca records that have been rolled up based on a rollup key (an Endeca property or dimension name). All records in the current record set that have the same value for the rollup key are collected together into an aggregated record. For example, rolling up on a Name key causes all wines in the current record set that have the value 'My Red Wine' for the Name key to be rolled up into one aggregated record.

Commonly, aggregated records are used to eliminate duplicate display entries. For example, in a music store catalog, an album by the same title may exist in several formats, with multiple prices. Each title is represented in the MDEX Engine as a distinct record. However, from a business perspective, it might be useful to treat these separate records as a single record by creating an aggregate record.

Record aggregation affects the current record set only. In other words, if you have 10,000 Endeca records total but only 3,000 are displayed in the current record set, then the aggregation affects those 3,000 records only.

The aggregated records feature requires that each record should have at most one value from the dimension or Endeca property that has been specified as the rollup key. Also, if an Endeca record has a unique value for the rollup key, it is 'rolled up' into an aggregated record that contains only one sub-record.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Enabling a dimension for record rollups
In order to use a dimension as a rollup key, you must enable its Rollup attribute.

To enable record rollup for a dimension:

1. In the Dimensions view, double-click the dimension you want to change to open it in the Dimension editor.
2. On the Advanced tab, check **Enable for Rollup**.
3. Click **OK** .

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Enabling Multi-Select AND or OR Dimensions

About multi-select dimensions
By default, the Endeca MDEX Engine only permits a single dimension value from any given dimension to be added to the navigation state. For some applications, however, it may be useful to allow the user to select more than one dimension value from a dimension.

After a user selects a leaf refinement from any dimension, that dimension is removed from the list of dimensions available for refinement in the query results.

For example, it might be useful to give a user the ability to show wines that have a flavor of Apple AND Apricot, or movies that star Cary Grant OR Clark Gable. This is accomplished by tagging the dimension as multi-select AND or OR , respectively.

Enabling a dimension for multi-select
Enable a dimension for multi-select AND or OR from the Advanced tab of the editor for that dimension.

To enable multi-select AND or OR:

1. In the Dimensions view, double-click the dimension you want to change to open it in the Dimension editor.
2. Click the **Advanced** tab.
3. In the Multiselect frame, choose **None**, **Or**, or**And.**
4. Click **OK.**

5. If you are enabling multi-select OR for a flat dimension, no more configuration is required. If you are enabling multi-select OR for a hierarchical dimension, you must configure all non-leaf dimension values as Inert. See "Multi-select OR and hierarchical dimensions" for details.

Multi-select OR and hierarchical dimensions

In a hierarchical dimension, you must configure all non-leaf dimension values to be inert, or non-navigable, to prevent them from appearing in the navigation query.

Multi-select OR queries are restricted to leaf dimension values. In a flat dimension, all possible refinements are leaf dimension values, so no extra configuration is necessary.

Working with Large Dimension Hierarchies

Enabling Collapsible Dimensions

About collapsible dimensions

Non-root and non-leaf dimension values are collapsible dimensions in a hierarchy.

A collapsible hierarchy is an ordinary hierarchy, in which some or all of the internal (non-root and non-leaf) dimension values are flagged as potentially collapsible. The MDEX Engine automatically removes, or collapses, these dimension values when there are only a few leaves available for refinement, creating a more streamlined, user-friendly navigation experience for your users.

For example, a dimension containing many state names could have a collapsible hierarchy introduced to group the names alphabetically. At query time, the available refinements are determined by the dimension values tagged to the records in the current navigation state. If there are many refinement values, it is easier for a user to select first a letter range, then a letter, and then the state name they want. But if there are only a few values, it is easier for the user to look at a brief list and select the state name directly. In this case, the letter-based dimension values can be collapsed, or removed, so that only the list of state names is displayed.

Dimension values that are configured as collapsible have the potential to be collapsible. Whether or not a dimension value is actually collapsed is controlled by the collapsible dimension threshold.

- Collapsible dimension values participate in matching in the same way as regular dimension values.
- For a common collapsible dimension use case scenario, see A typical scenario.

Note: Collapsible dimension values participate in matching in the same way as regular dimension values.

Creating a collapsible dimension hierarchy

This feature requires that you make all dimension values collapsible, and set a collapsible dimension threshold.

To create a collapsible hierarchy:

1. Create the complete hierarchy as you would for a normal dimension.
2. Make the internal dimension values collapsible.
3. Set a collapsible dimension threshold.

   Note: You will probably want to rank the collapsible dimension values as well, to ensure that they are presented to the user in an order that makes sense.

Making a dimension value collapsible

Making a dimension value collapsible means that it is a candidate for collapsing.

See "About collapsible dimensions" for details.

To make a dimension value collapsible:

1. In the Dimension Values view, double-click the name of the dimension value that you want to change to open it in the Dimension Value editor.

2. Click **Collapsible**.
3. Click **OK.**

Setting a collapsible dimension threshold

The collapsible dimension threshold determines how many refinement values must exist for a set of query results in order for dimension value collapsing to occur.

- If the number of refinement values is less than or equal to the threshold, dimension values that have been configured as potentially collapsible are collapsed (that is, removed from the hierarchy). This creates a shallower hierarchy with fewer conceptual levels of refinement.
- If the number of refinement values is greater than the threshold, dimension values that have been configured as potentially collapsible are not collapsed. They remain in the hierarchy, creating a deeper hierarchy with more conceptual levels of refinement.

To set a collapsible dimension threshold:

1. In the Dimensions view, double-click the dimension you want to change to open it in the Dimension editor.
2. In the Advanced tab, enter the collapsing threshold, in terms of number of refinement values, in the **Collapsible Dimension Threshold** text box.
3. Click **OK.**

A typical scenario

This describes a typical scenario for setting dimensions and dimension values.

A typical scenario involves combining three features in one dimension:

- Create a sift dimension with a set of discrete ranges such as A - D, E - H, and so on. Auto sifting is an extension to autogeneration which positions newly generated dimension values within an existing hierarchy, such that Applebrandy Wine appears under A - D, Four Grapes Merlot appears in E - H, and so on.
- Configure the dimension values in the sift dimension as non-navigable so that they can appear in the user interface to assist the end user but do not have any effect on the navigation state.
- Configure the dimension values in the sift dimension so that they are collapsible. A collapsible hierarchy is an ordinary hierarchy, in which some or all of the internal (non-root and non-leaf) dimension values are flagged as potentially collapsible. The MDEX Engine automatically removes, or collapses, these dimension values when there are only a few leaves available for refinement, creating a more streamlined, user-friendly navigation experience for your users.

Creating Sift Dimensions

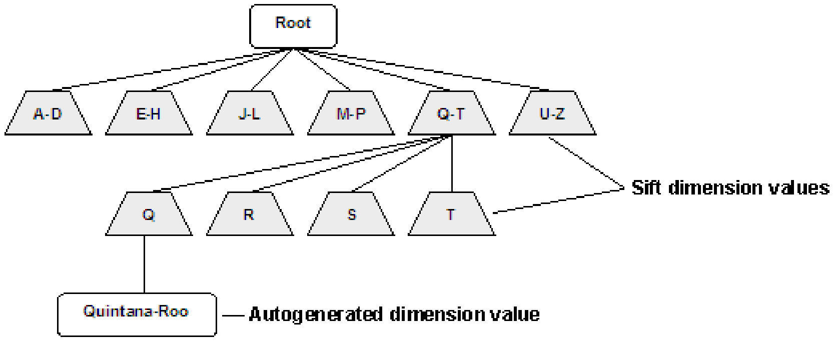
About sift dimensions

Auto sifting is an extension to autogeneration which positions newly generated dimension values within an existing hierarchy.

The hierarchy must be a sift hierarchy, which is a normal hierarchy that contains dimension values of type Sift. As with autogeneration, the dimension mapping in the property mapper must be configured for auto generation.

Configuring sift dimension values

A sift dimension value must be configured as type Sift and specify a range. In the illustration below, all of the dimension values except the root and Quintana-Roo are sift dimension values. The first layer of dimension values after the root specify ranges that span a set of letters, A to D, E to H, and so on. The set of child dimension values below Q-T are also ranges, although they aren't as obvious. The Q dimension value includes all strings that are greater than Q but less than R. Q, therefore, includes strings such as Q, QWERTY, QUEEN, QQQ, QZZZ but not PEZ or RA.



Note: Although the range for a sift dimension value is specified the same way as for a range dimension value, sift and range are separate, and mutually exclusive, features.

Sifting an auto-generated dimension value

When a new dimension value is auto-generated, it sifts down the hierarchy according to the ranges it matches. For example, in the illustration above, the auto-generated Quintana-Roo dimension value would first sift into Q-T, and then into Q. The sifting of auto-generated dimension values is handled according to these rules:

- Dimension values that do not match any sift range in the hierarchy are added as children of the dimension root.
- Dimension values that match the ranges of multiple children of a sift dimension value are placed under the first child they match.
- Dimension values that do not match any child of a sift dimension value are placed under the deepest value they match.

It is a good idea to configure the last child under each sift dimension value as a catch-all, with a range of NEG_INF to POS_INF , inclusive. That way, auto-generated dimension values that do not match any other range can be gathered under an Other dimension value.

Sift dimension values and matching

Sift dimension values do not participate in matching. Even though the range Q-T appears within the dimension above, the property value Quintana-Roo will not match it. Following the match failure, a new Quintana-Roo dimension value is automatically generated and sifted through the hierarchy. Any further properties with the value Quintana-Roo will match this auto-generated dimension value.

Sift hierarchy recommendations

Sift hierarchies are frequently made collapsible. A collapsible hierarchy is a form of conditional hierarchy in which some or all of the internal (non-root and non-leaf) dimension values are flagged as potentially collapsible. The MDEX Engine automatically removes, or collapses, these dimension values when there are only a few leaves available for refinement, creating a more streamlined, user-friendly navigation experience for your users. See About collapsible dimensions for details.

In addition, sift dimension values are typically ranked, to ensure that the ranges appear in order during navigation.

Note: All dimension values in a sift dimension must be specified as type Sift or there may be unexpected behavior.

Saving autosift state information

At the end of each data processing run, the dimension server saves the state of an auto sift dimension, just like any other auto-generated dimension. At the start of the next run, the sift hierarchy may be updated; therefore, all of the previously auto-generated dimension values are re-sifted into the new sift hierarchy. This ensures that they are positioned correctly within the hierarchy, even when the hierarchy has been changed since it was initially generated.

Note: For a common sift dimension use case scenario, see A typical scenario.

Creating a sift dimension hierarchy

Defining a sift dimension is a two-step process. First, you create the sift dimension hierarchy, then you specify that the dimension should use auto-generation during its mapping process.

The procedure below shows you how to create a sift dimension for a hierarchy.

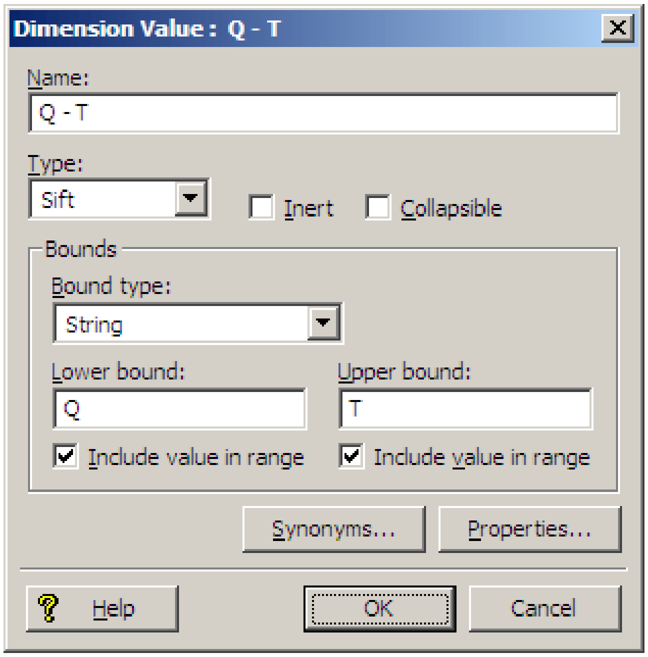
To create a sift dimension:

1. In the Dimensions view, create a new dimension.
2. Add dimension values to the dimension and define them as Sift by doing the following:
   a) In the Type list, choose **Sift**.
   b) From the Bound Type list, choose one of the following: **String**, **Floating point**, or **Integer**.
   c) In the **Lower Bound** text box, enter the lower number in the range.

      Note: If you want the range to include the number you enter, check **Include value in range**.

   d) In the Upper Bound text box, enter the higher number in the range. If you want the range to include the number you enter, check Include Value in Range. This illustration shows how to set the range for the Q - T dimension value.

      Note: By specifying R as the upper bound for the Q dimension value, but not including it in the range, we ensure that all strings that begin with Q are included in the range, such as Queen, Quick, QWERTY, and so on.


Dimension Value : Q - T

Name: Q - T

Type: Sift ☐ Inert ☐ Collapsible

Bounds

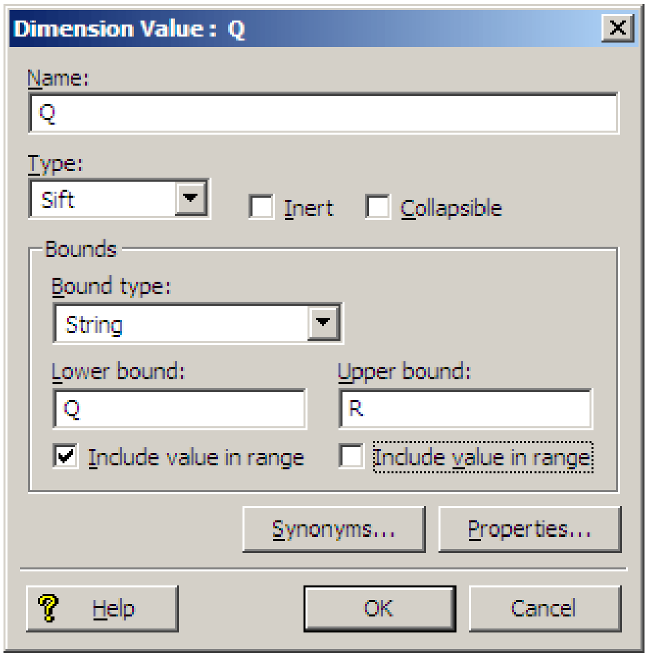
Bound type: String

Lower bound: Q

Upper bound: T

☑ Include value in range ☑ Include value in range

This illustration shows how to set the range for the Q dimension value.


Dimension Value : Q

Name: Q

Type: Sift ☐ Inert ☐ Collapsible

Bounds

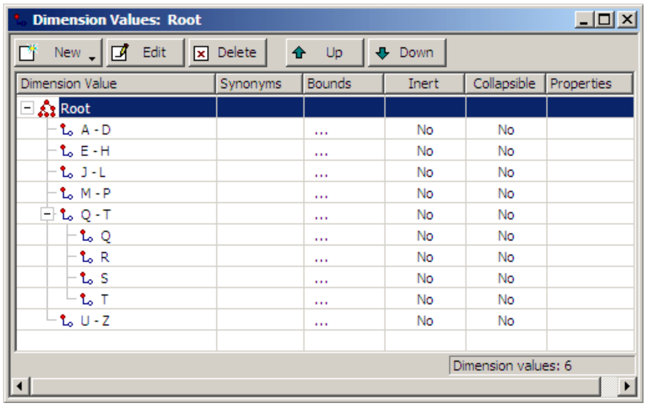
Bound type: String

Lower bound: Q

Upper bound: R

☑ Include value in range ☐ Include value in range

3. Click **OK** to close the Dimension Value editor.
4. Continue to populate the dimension with sift dimension values as needed. This illustration shows what the Dimension Values editor looks like for the example established in this procedure.



Dimension Value	Synonyms	Bounds	Inert	Collapsible	Properties
Root					
A - D		...	No	No	
E - H		...	No	No	
J - L		...	No	No	
M - P		...	No	No	
Q - T		...	No	No	
Q		...	No	No	
R		...	No	No	
S		...	No	No	
T		...	No	No	
U - Z		...	No	No	

Dimension values: 6

Note:

- Although the range for a Sift dimension value is specified the same way as for a Range dimension value, Sift and Range are separate, and mutually exclusive, features.
- As mentioned in step 2 a in the procedure above, all dimension values in a sift dimension must be specified as type Sift or there may be unexpected behavior.
- If a dimension value cannot be precisely represented as an IEEE double, the dimension value might get assigned to an incorrect range.

Defining a sift dimension as autogenerated

Upon creating a sift dimension hierarchy, you may enable Auto Generate as the match mode for a sift dimension.

If you have not already done so, create a sift dimension hierarchy.

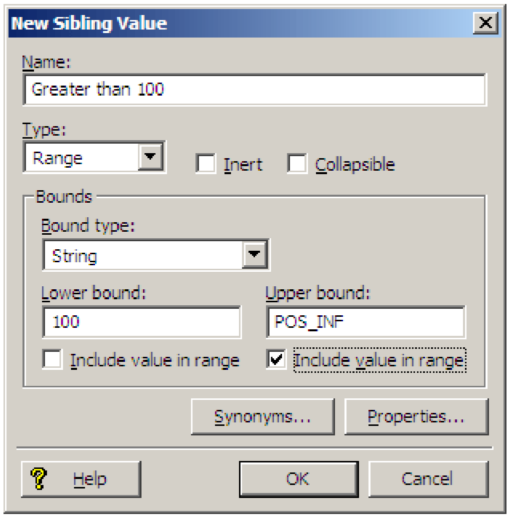
To define the sift dimension as autogenerated:

1. In the pipeline diagram, double-click your property mapper component to open it in the Property Mapper editor.
2. Click **Mappings**.
3. Create a dimension mapping for your sift dimension, being sure to choose **Auto Generate** as the match mode.

Using NEG_INF and POS_INF

Specify either of these values as the lower or upper bound for a dimension value, to indicate a value less than or greater than all other values in the range.

You can use two special values, NEG_INF and POS_INF, when creating the bounds for your dimension values. NEG_INF indicates less than all other values while POS_INF indicates greater than all other values. For example, to specify a range of greater than 100, you would use a lower bound of 100 and an upper bound of POS_INF.



Name: Greater than 100

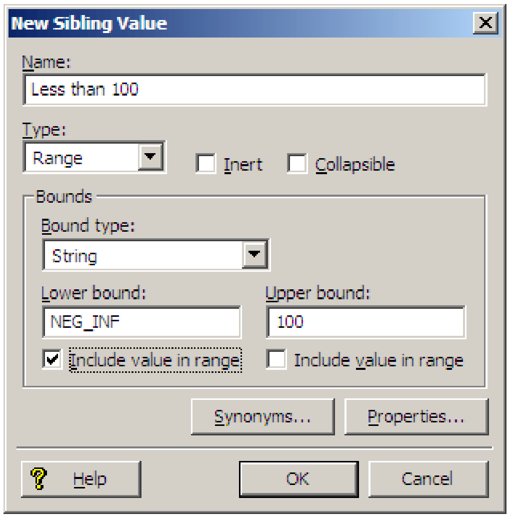
Type: Range ☐ Inert ☐ Collapsible

Bound type: String

Lower bound: 100 ☐ Include value in range

Upper bound: POS_INF ☑ Include value in range

Likewise, less than 100 would use a lower bound of NEG_INF and an upper bound of 100.



Name: Less than 100

Type: Range ☐ Inert ☐ Collapsible

Bound type: String

Lower bound: NEG_INF ☑ Include value in range

Upper bound: 100 ☐ Include value in range

You can use POS_INF and NEG_INF with string values as well. For example, setting a lower bound of S, inclusive, and an upper bound of POS_INF, inclusive, would match all strings starting with S and going to the end of the alphabet, including values such as S, Style, Trigger, and Zzzzz.

A note about locale and encoding

The order of symbols depends on your locale setting, which is external to the Endeca software. On UNIX, it is determined by a set of environment variables, typically LOCALE, LANG, or LC_ALL. On Windows, there are separate system and user locales which can be set from the Regional and Language Options control panel. For example, in ASCII, using [NEG_INF, A) as the bounds includes all numerics and many symbols (the '[' symbol indicates the value is inclusive while ')' indicates it is not). Using (Z, POS_INF] includes the rest of the symbols, as well as lower-case letters. This is not the case for other encodings, such as Unicode, which intersperses symbols and numbers with letters much more than ASCII. To use NEG_INF and POS_INF effectively, you must have a good understanding of the order of symbols in your locale's encoding.

Creating an "other" dimension value

When working with range and sift dimension values, it can be useful to have an Other dimension value to capture any source property values that don't fall within the defined ranges.

To catch source property values outside of your defined ranges, you must use a Perl manipulator, placed after the property mapper, to look at each record that has been mapped, and determine if it has any dimension values assigned to it from the range/sift dimension. If not, the Perl manipulator should assign the Other dimension value to the record.

A typical scenario

This describes a typical scenario for setting dimensions and dimension values.

A typical scenario involves combining three features in one dimension:

- Create a sift dimension with a set of discrete ranges such as A - D, E - H, and so on. Auto sifting is an extension to autogeneration which positions newly generated dimension values within an existing hierarchy, such that Applebrandy Wine appears under A - D, Four Grapes Merlot appears in E - H, and so on.
- Configure the dimension values in the sift dimension as non-navigable so that they can appear in the user interface to assist the end user but do not have any effect on the navigation state.
- Configure the dimension values in the sift dimension so that they are collapsible. A collapsible hierarchy is an ordinary hierarchy, in which some or all of the internal (non-root and non-leaf) dimension values are flagged as potentially collapsible. The MDEX Engine automatically removes, or collapses, these dimension values when there are only a few leaves available for refinement, creating a more streamlined, user-friendly navigation experience for your users.

Setting the primary dimension

Prior to the MDEX Engine version 6.1.0, each project had to have a single primary dimension which functioned as the root for all other secondary dimensions in the project. You needed to set up precedence rules between the primary dimension and each secondary dimension to ensure that the secondary dimensions would appear in the navigation controls.

If a primary dimension was not explicitly defined in Developer Studio, Dgidx created one for you. Dgidx also created the required precedence rules between the primary dimension and the other dimensions in your project, which were, by default, considered secondary. For most projects, this was sufficient and developers did not have to worry about creating a primary dimension.

Partial updates, however, required an explicit primary dimension that had explicit precedence rules to all secondary dimensions. Starting with the MDEX Engine version 6.1.0, you no longer need to specify a primary dimension. If it is specified, it is ignored by the MDEX Engine. Note that partial updates no longer require that a primary dimension is specified explicitly.

The procedure below describes how to set a primary dimension. See "About precedence rules" and "Creating, modifying, and deleting precedence rules" for information on creating precedence rules.

Note: Explicitly defined precedence rules override any internally generated rules.

To set the primary dimension:

1. In the Dimensions view, double-click the dimension you want to change to open it in the Dimension editor.
2. On the Advanced tab, check **Primary**.
3. Click **OK**.

Troubleshooting dimensions

Be sure that source data reflects dimension values, match modes are correctly set, and that source properties are mapped to a dimension or an Endeca property.

If a dimension value does not appear as expected in the client browser, double-check the following potential issues:

- Check that the value exists in the source data. If the dimension value 2010 is added to the Year dimension, but there are no records with that year, then the MDEX Engine will never present 2010 as an option in the client browser.
- Matches to dimension values of type Exact must be, as the name implies, exact and are case-sensitive. The property values $20 and Twenty will not match to the dimension values 20 or twenty.
- A source property must be mapped to either a dimension or an Endeca property in order to appear in your Endeca-enabled Web application. If a source property that you want to map to a dimension is not appearing in your Web application, make sure the following is true:

Your source property is mapped correctly to a dimension in your pipeline's property mapper. See "Establishing a dimension mapping" for details.

The dimension is either configured for auto-generation, or the source property values on the records match the dimension values that you explicitly defined for the dimension in the Dimensions editor.

The Show with Record List and Show with Record options are set correctly in the Dimension editor.

Note: By default, Forge removes source properties that have not been mapped to an Endeca property or dimension. See "Removing source properties after mapping" for details.

Mapping Source Properties

This section provides information on working with property mappers in Developer Studio.

Understanding Source Property Mapping

About property mapping

Source property mappings dictate which dimension values are tagged to each record and which property information is available for record search, sort, and display.

Source properties can be mapped in three different ways. They can be:

- Mapped to an Endeca property (for search, sort, and display only).
- Mapped to a (for search, sort, display, and navigation).
- Ignored by specifying a null mapping.

Note the following requirements when you map properties:

- If you are using Oracle, source properties must be uppercase.
- If you are using MSSQL, source properties must be lowercase.

You use a property mapper component to establish source property mappings. Typically, the property mapper is placed in the pipeline after the Perl manipulator that is used to clean and prepare source properties (if one exists). You should use a single property mapper to map all of your source properties to both Endeca properties or dimensions.

Note: Before you can map a source property to an Endeca property or dimension, you must create the Endeca property or dimension.

Understanding property mappers is critical to building an Endeca implementation. Oracle recommends that you read the chapter "Mapping Source Properties" in the *Endeca Forge Guide* to gain a solid grounding in this critical concept.

Using a single property mapper

You should use a single property mapper to map all of your source properties to both Endeca properties or dimensions.

Note: There are rare cases where multiple property mappers may be used. However, Oracle strongly recommends that you use only one property mapper in any given pipeline.

About explicit mapping

When you specify a source property and a target Endeca property or dimension to map to, you are creating an explicit mapping.

In general, this is the type of mapping Oracle recommends you use. However, Endeca also offers some advanced techniques that allow you to automate the mapping process. These techniques are intended to facilitate the process of building prototypes and should not be used for building production-ready implementations.

See the related topics for detailed explicit mapping procedures.

Choosing a match mode

When Forge maps a source property value to a dimension value, the dimension value it uses can either be explicitly defined in the dimension hierarchy or automatically generated by Forge. You set the type of dimension value handling, on a per mapping basis, using the Match Mode in the Dimension Mapping dialog box.

Note: Match modes only apply to dimension values. They are not used when mapping source properties to Endeca properties.

There are three match modes you can choose from:

- Normal
- Must Match
- Auto Generate

Normal

Normal maps only those source property values that have a matching dimension value explicitly defined in the dimension hierarchy. Forge assigns the IDs for any matching dimension values to the Endeca records. Any source property values that do not have matching dimension values in the dimension hierarchy are ignored.

In order for a source property value to match a dimension value, the dimension value's definition must contain a synonym that:

- Is an exact text match, including capitalization and white space, to the source property value.
- Has its Classify option enabled.

Note: There are two types of advanced dimension values, range and sift. These dimension value types have different matching criteria than standard dimension values. See About range dimension values and About sift dimensions, respectively.

Must Match

Must Match behaves identically to Normal, with the exception that Must Match issues a warning for any source property values that do not have matching dimension values.

Auto Generate

Auto Generate specifies that Forge automatically generates a dimension value name and ID for any source property value that does not have a matching dimension value in the dimension hierarchy. Forge uses these automatically-generated names and IDs to tag the Endeca records the same as it would explicitly-defined dimension values.

Auto Generate dramatically reduces the amount of editing you have to do to the dimension hierarchy. However, auto-generated dimensions are always flat. Auto-generated names and IDs are persisted in a file that you specify as part of a dimension server component.

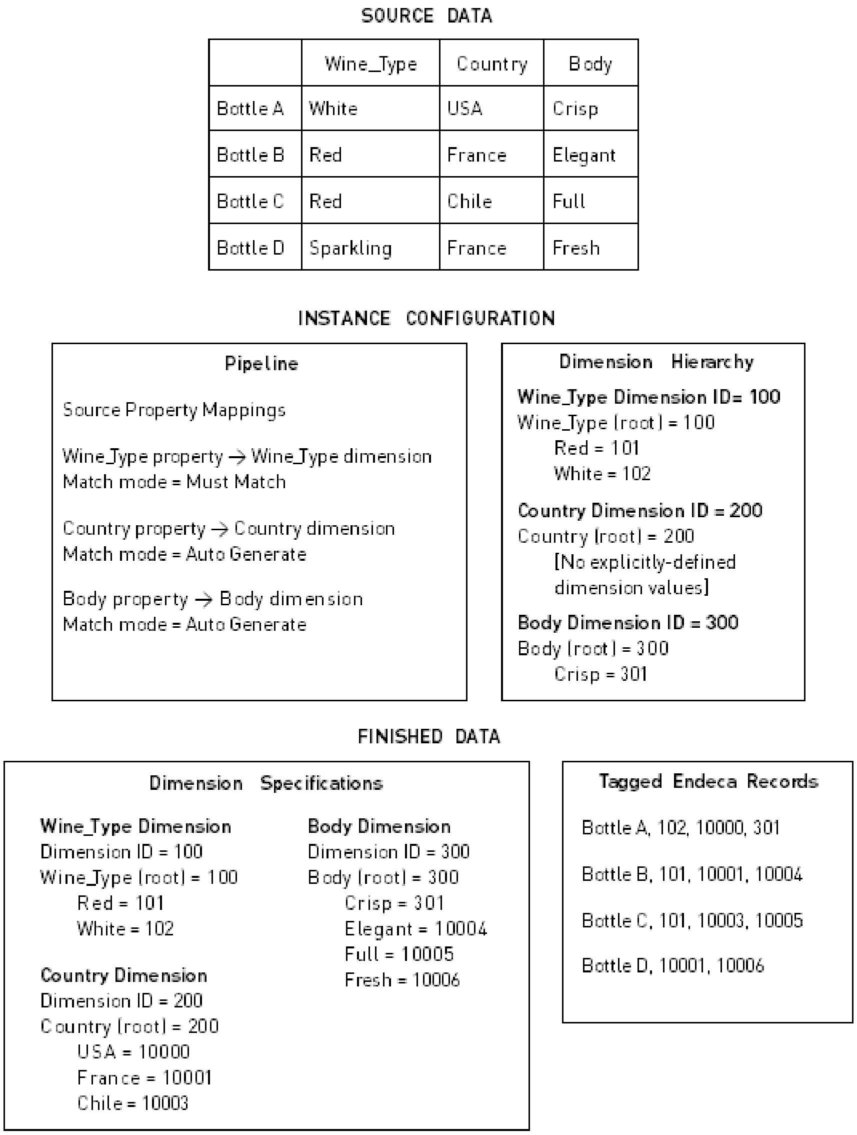
Mapping Rules of Thumb

When you choose which method to use for generating your dimension values, there are two rules of thumb to keep in mind:

- If you manually define dimension values in the dimension hierarchy, the Normal, Must Match, and Auto Generate features behave identically with respect to those dimension values.
- Any source property value that does not have a matching dimension value specified in the dimension hierarchy will not be mapped unless you have set the dimension to Auto Generate in the pipeline.

Mapping Example

The following illustration shows a simple dimension mapping example that uses a combination of generation methods. The sections after the illustration describe the mapping behavior in the example.



SOURCE DATA

	Wine_Type	Country	Body
Bottle A	White	USA	Crisp
Bottle B	Red	France	Elegant
Bottle C	Red	Chile	Full
Bottle D	Sparkling	France	Fresh

INSTANCE CONFIGURATION

Pipeline

Source Property Mappings

Wine_Type property → Wine_Type dimension
Match mode = Must Match

Country property → Country dimension
Match mode = Auto Generate

Body property → Body dimension
Match mode = Auto Generate

Dimension Hierarchy

Wine_Type Dimension ID= 100
Wine_Type (root) = 100
Red = 101
White = 102

Country Dimension ID = 200
Country (root) = 200
[No explicitly-defined dimension values]

Body Dimension ID = 300
Body (root) = 300
Crisp = 301

FINISHED DATA

Dimension Specifications

Wine_Type Dimension
Dimension ID = 100
Wine_Type (root) = 100
Red = 101
White = 102

Country Dimension
Dimension ID = 200
Country (root) = 200
USA = 10000
France = 10001
Chile = 10003

Body Dimension
Dimension ID = 300
Body (root) = 300
Crisp = 301
Elegant = 10004
Full = 10005
Fresh = 10006

Tagged Endeca Records

Bottle A, 102, 10000, 301

Bottle B, 101, 10001, 10004

Bottle C, 101, 10003, 10005

Bottle D, 10001, 10006

Wine Type Dimension

The Red and White property values have matching Red and White dimension values specified in the dimension hierarchy. These property values are mapped to the Red and White dimension values IDs, respectively. Bottles B and C are tagged with the Red dimension value ID, and Bottle A is tagged with the White dimension value ID.

The Sparkling property value does not have a matching dimension value in the dimension hierarchy. The Wine Type dimension is set to Must Match, so this property is ignored and a warning is issued. As a result, Bottle D does not get tagged with a dimension value ID from the Wine Type dimension.

Country Dimension

There are no dimension values explicitly defined in the dimension hierarchy for the Country dimension. However, this dimension is set to Auto Generate, so all three of the Country property values (USA, France, and Chile) are mapped to automatically-generated dimension value IDs.

Bottle A is tagged with the auto-generated ID for the USA dimension value. Bottles B and D are tagged with the auto-generated ID for the France dimension value. Bottle C is tagged with the auto-generated ID for the Chile dimension value.

Body Dimension

The Crisp property value has a matching dimension value specified in the dimension hierarchy, so the Crisp property value is mapped to the Crisp dimension value. Bottle A is tagged with the Crisp dimension value ID.

The other three property values (Elegant, Full, and Fresh) do not have matching dimension values in the dimension hierarchy but, because the Body dimension is set to Auto Generate, these three property values are mapped to automatically-generated dimension values IDs.

Bottle B is tagged with the auto-generated ID for the Elegant dimension value. Bottle C is tagged with the auto-generated ID for the Full dimension value. Bottle D is tagged with the auto-generated ID for the Fresh dimension value.

Regardless of how they were generated, all of the dimension value IDs are included in the finished data that Forge produces for indexing.

Assigning multiple mappings to a single source property

You can assign more than one mapping to a source property-for example, you can map a source property to both a dimension and an Endeca property.

A typical source property that you may want to map to both a dimension and an Endeca property is Price. You can map the Price source property in the following ways:

- To a Price Range dimension that allows the end-user to search for records within a given price range (for example, wines that cost between $10 and $25).
- To an Endeca property that allows you to display the discrete price of each individual record.

Conversely, you can assign more than one source property to a single dimension or Endeca property. For example, if you have multiple source properties that are equivalent, most likely they should all be mapped to the same dimension or Endeca property. Flavor and Color are example properties that might require this behavior

Working with Property Mappers

Adding a property mapper

At a minimum, a property mapper requires both a record source and a dimension source to define the components that will supply it with record and dimension data.

You can leave the other settings at their defaults while developing your initial working pipeline.

To add a property mapper:

1. In the Pipeline Diagram editor, click **New**.
2. Choose **Property Mapper**.
   The Property Mapper editor appears.
3. In the **Name** text box, type a unique name for this property mapper.
4. In the Sources tab, set the record source and dimension source.
5. Click **Mappings** to open the Mappings editor, where you can view and modify existing mappings. For detailed procedures, see the following topics:

- Creating a dimension mapping
- Creating a property mapping
- Creating a null mapping

6. (Optional) In the Record Index tab, do the following:
    - Specify which properties or dimensions you want to use as the record index for this component.
    - Check whether you want to remove records with duplicate keys.

    Note: Developer Studio performs a case-insensitive search for duplicate keys.

7. (Optional) In the Advanced tab, set any advanced mapping options.
8. (Optional) In the Comment tab, add a comment for the component.
9. Click **OK**.

Property Mapper editor

The Property Mapper editor contains a unique name for this property mapper and a button that opens the Mappings editor, where you can view and modify mappings.

The Property Mapper editor contains the following tabs:

Sources

The Sources tab contains the following options:

Option	Description
Record Source	Required. A choice of the record server components in the project.
Dimension Source	Required . A choice of the dimension adapters and dimension servers in the project.

Record Index

Optional. The Record Index tab allows you to add or remove dimensions or properties used in a component's record index, and to change their order. Record indexes support join functionality. See Join sources must have matching join keys and record indexes for more details.

Advanced

The Advanced tab contains the following options:

Option	Description
Map source properties to Endeca dimensions with the same name	Check this field to enable implicit mapping. When implicit mapping is enabled, any source property that has a name that is identical to an existing dimension is automatically mapped to that dimension. (Implicit

Option	Description
	mapping only occurs if an explicit mapping is not specified.
If no mapping is found, map source properties to Endeca	Check this field to enable default mapping . Default mapping defines how Forge will handle source properties that have neither explicit nor implicit mappings.
Default maximum length	(Optional) Specifies the maximum source property value length allowed when creating mappings. Source properties that have values that exceed this length are not mapped, and a warning is issued by the Forge Hierarchical Logging system. If this field is left blank, the default is to not limit Endeca property mappings, and to limit dimension mappings to 255 characters.

Comment

(Optional) Provides a way to associate comments with a pipeline component.

Viewing existing property mappings

All property mappings are visable in the Property Mapper editor of each property mapper. Create new mappings, modify, or remove existing mappings.

To view existing property mappings in your project:

1. In the pipeline diagram, double-click your property mapper to open it in the Property Mapper editor.
2. In the Property Mapper editor, click **Mappings**.
   The Mappings editor appears.
3. (Optional) Make any required modifications, using the **New**, **Modify**, and **Remove** buttons.
4. Click **OK**.

Note: A single source property can be mapped to multiple targets, including a mix of properties and dimensions. A single target can also be mapped from multiple sources.

Creating Explicit Meanings

Establishing a dimension mapping

You must create a dimension before you can map it to a source property.

To map a source property to an Endeca dimension:

1. In the pipeline diagram, double-click your property mapper to open it in the Property Mapper editor.
2. In the Property Mapper editor, click **Mappings**.
   The Mappings editor appears.
3. Click **New**.

4. Select **Dimension Mapping**.
   The Dimension Mapping editor appears.
5. Type the name of the source property you want to map in the **Source Property** text box.
6. Choose the name of the dimension you want to map to from the Target Dimension menu.
7. (Optional) In the Maximum Length text box, type a value that defines the maximum source property value length allowed when creating mappings.

   Note: Source properties that have values that exceed this length are not mapped, and a warning is issued by Forge. This value overrides the default maximum length that you set on the Advanced tab of the Property Mapper editor.

8. Choose a match mode from the Match mode list.
9. Click **OK** to return to the Mappings editor.
10. Continue mapping source properties to dimensions.

    Note: You can map multiple source properties to a single dimension and vice versa .

11. Click **OK** to return to the Property Mapper editor.

Establishing a property mapping

You must create an Endeca property before you can map it to a source property.

To map a source property to an Endeca property:

1. In the pipeline diagram, double-click your property mapper to open it in the Property Mapper editor.
2. In the Property Mapper editor, click **Mappings**.
   The Mappings editor appears.
3. Click **New**.
4. Select **Property Mapping**.
   The Property Mapping editor appears.
5. Type the name of the source property you want to map in the **Source Property** text box.
6. Choose the name of the Endeca property you want to map to from the Target Property menu.
7. (Optional) In the **Maximum Length** text box, type a value that defines the maximum source property value length allowed when creating mappings.

   Note: Source properties that have values that exceed this length are not mapped, and a warning is issued by Forge. This value overrides the default maximum length that you set on the Advanced tab of the Property Mapper editor.

8. Click **OK** to return to the Mappings editor.
9. Continue mapping source properties to Endeca properties.

   Note: You can map multiple source properties to a single Endeca property and vice versa.

10. Click **OK** to return to the Property Mapper editor.

Establishing a null mapping

Null mappings prevent a source property from being mapped to an Endeca property or dimension.

Most commonly, null mappings are used to prevent an automated mapping from being formed for a particular source property. In other words, you can enable automated mapping, using advanced mapping techniques, and then turn off mapping for selected source properties using null mappings.

To establish a null mapping:

1. In the pipeline diagram, double-click your property mapper to open it in the Property Mapper editor.
2. In the Property Mapper editor, click **Mappings**.
   The Mappings editor appears.
3. Click **New**.
4. Select **Null Mappings**.
   The Null Mapping editor appears.
5. In the **Source Property** text box, type the name of the source property that you want to prevent from being mapped.
6. Click **OK** to close the Null Mapping dialog box.
7. Click **OK** to close the Mappings dialog box.
8. Click **OK** to close the Property Mapper editor.

Advanced Property Mapping Techniques

Advanced techniques introduced

Use caution when opting to use advanced techniques instead of the recommended explicit mapping techniques offered in the Property Mapper editor.

Note: Oracle strongly recommends that you use the explicit mapping techniques, described in the "Creating explicit mappings" section, for implementations that are intended to go to production because the advanced mapping techniques can have unexpected results if you are not careful when using them.

There are several advanced mapping techniques, available on the Property Mapper editor's Advanced tab, that you can use when building prototypes. This section describes these techniques and provides information on the priority order that the Data Foundry uses when applying them.

About mapping source properties to like-named dimensions

This type of mapping enables implicit mapping. When implicit mapping is enabled, any source property that has a name that is identical to an existing dimension is automatically mapped to that dimension.

The setting Map Source Properties to Endeca Dimensions with the Same Name (located on the Property Mapper editor's Advanced tab) enables implicit mapping. The like-named dimension, and any of its constituent dimension values, must already exist in your dimension hierarchy (in other words, you've already defined them using the Dimensions and Dimension Values editors).

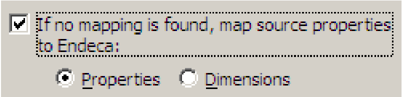
Implicit mapping uses the Normal mapping mode where only those source property values that have a matching dimension value explicitly defined in the dimension hierarchy are mapped. Forge assigns the IDs for any matching dimension values to the Endeca records. Any source property values that do not have matching dimension values in the dimension hierarchy are ignored.

Implicit mapping is limited to mappings between source properties and dimensions. Implicit mapping cannot take place between source properties and Endeca properties.

When no mapping is found

The Property Mapper editor’s Advanced tab lets you map source properties to either Endeca properties or dimensions if no mapping is found.

The Property Mapper editor's Advanced tab contains the following option:



☑ If no mapping is found, map source properties to Endeca:

● Properties ○ Dimensions

This option defines the default that Forge uses to handle source properties that have neither explicit nor implicit mappings. There are three possible settings:

- Uncheck the option altogether to ignore source properties that do not have an explicit or implicit mapping defined.
- Check **Property** to create a mapping between the source property and an Endeca property. Forge does this by creating a new Endeca property that uses the same name and value as the source property and assigning it to the record.
- Check **Dimension** to create a mapping between the source property and a dimension. Forge does this by creating a new dimension, using the source property's name. Forge uses the Auto Generate mode to populate the dimension with dimension values that match the source property's values.

Use the default mapping option with caution because:

- With this option enabled, all source properties will ultimately be mapped and mapped properties use system resources. Ideally, you should only map source properties that you intend to use in your implementation so that you minimize the use of system resources.
- Many production-level implementations automatically pull and process new data when it is available. If this data has new source properties, these properties will be mapped and included in your MDEX Engine indices. Again, this uses system resources unnecessarily but, perhaps more importantly, this situation may also result in the display of dimensions or Endeca properties that you do not want your users to see.

Priority order of source property mapping

Forge looks for explicit mapping and attempts implicit mapping between source property and a like-named dimension if the explicit mapping not exist. Forge defaults to the selected If No Mapping is Found option if neither explicit nor implicit mapping exists.

Forge uses this prioritization when mapping source properties:

1. First, Forge looks for an explicit mapping for the source property.
2. If no explicit mapping exists, and Map Source Properties to Endeca Dimensions with the Same Name is enabled, Forge will try to create an implicit mapping between the source property and a like-named dimension.
3. If no explicit or implicit mapping exists, Forge uses the If No Mapping is Found, Map Source Properties to Endeca: Properties/Dimensions option to determine how to handle the mapping.

About using null mappings to override implicit and default mappings

Explicit null mappings provide a means to prevent an implicit or default mapping from being formed for a particular source property.

In other words, you can enable either implicit or default mapping, and then turn off mapping altogether for selected source properties using explicit null mappings.

About setting a default maximum length for source property values

The Default Maximum Length option, on the Advanced Tab of the Property Mapper, defines the maximum source property value length allowed when creating mappings.

Source properties that have values that exceed this length are not mapped, and a warning is issued by Forge.

If you do not explicitly specify a Default Maximum Length, Forge does the following:

- Source properties that are mapped to Endeca properties can have values of any length.
- Source properties that are mapped to dimensions must have values that are 255 characters or less.

If you do explicitly specify a Default Maximum Length, that length is applied to both Endeca property and dimension mappings.

You can override the Default Maximum Length on a per mapping basis using the Maximum Length field in both the Property Mapping and Dimension Mapping dialog boxes. For example, you can use the Default Maximum Length to limit the length of all your source properties, but set an override for a Description source property that allows the description to be as long as necessary.

Troubleshooting source property mapping

Verify that your mappings are properly configured, and that you have correctly enabled property and dimension display. For projects that use MDEX prior to 6.1.0, verify that precedence rules connect any manually-set primary dimensions to secondary dimensions.

If a source property does not appear correctly in your Endeca-enabled Web application, make sure that:

- Your source property is mapped to either an Endeca property or a dimension in your pipeline's property mapper. See Establishing a property mapping and Establishing a dimension mapping for details.
- If the source property is mapped to a dimension, the dimension is either configured for auto-generation, or the source property values on the records match the dimension values that you explicitly defined for the dimension in the Dimensions editor.
- The Show with Record List and Show with Record options are set correctly in the Property editor or Dimensions editor. See one of the following sections for details:
  - Enabling Endeca property display in a record list
  - Enabling Endeca property display on a record page
  - Enabling dimension display in a record list
  - Enabling dimension display on a record page
- In releases prior to the MDEX Engine v.6.1.0, if you had manually designated a primary dimension, you had to create precedence rules from that dimension to all of your secondary dimensions. If you are using Developer Studio with the MDEX Engine v.6.1.0 and up, you no longer have to create precedence rules from the primary dimension to all your secondary dimensions. Primary dimension is also no longer requiredor used by the MDEX Engine.

Chapter 4

Configuring Application Features

Configuring Precedence Rules

This section provides information on working with the precedence rule editor in Developer Studio.

About precedence rules

Precedence rules provide a way to delay the display of dimensions until they offer a useful refinement of the navigation state.

Precedence rules are defined in terms of a trigger dimension value and a target dimension, where a user's selection of the trigger dimension value reveals the previously unavailable target dimension to the user.

For example, one might not want both the Country and State dimensions to appear simultaneously in a geographical data set. A precedence rule could be defined so that the State dimension would appear only after a dimension value from the Country dimension is selected. This simplifies the user's navigation choices and avoids information overload by hiding the State dimension until it is relevant to the navigation state.

Treatment of target dimensions associated with multiple precedence rules

A target dimension associated with more than one precedence rule is exposed when at least one associated trigger dimension value is selected.

For example, assume we have three dimensions: Author, Region, and Language. We have two precedence rules:

```
Region > Author
Language > Author
```

In this case, the Author dimension is displayed after a dimension value from either the Region or Author dimension is selected.

The Precedence Rules view

The Precedence Rules view contains information about precedence rules you have created for your Endeca implementation.

This view has two columns:

Column	Description
From	Shows the dimension value that serves as a trigger for the target dimension to be displayed. For example, in a wine store catalog, you might set up a precedence rule whereby the Winery dimension is not displayed until the Region dimension value has been selected. In this example, the Region dimension value would be displayed in the From column.
To	Shows the dimension that serves as the target for the trigger dimension value. Extending the example above, the Winery dimension would be the target.

Viewing all of your precedence rules

All precedence rules are visible from the Project tab.

To view all of your precedence rules at once:

On the Project tab, double-click **Precedence Rules**.
The Precedence Rules view opens and displays all of the precedence rules in your project.

Precedence Rule editor

The Precedence Rule editor allows you to create, modify, or delete precedence rules.

The Precedence Rule editor contains the following fields:

Option	Description
Source dimension value	The dimension that serves as a trigger for the To (or target) dimension value to be displayed.
Target dimension	The dimension that serves as a target for the From (or source) dimension value.
Rule type	There are two types of source dimension values: • The standard type (indicated in the table by this icon) means that if the dimension value specified as the trigger or any of its descendents are in the navigation state, then the target is presented (one trigger, one target). • The leaf type (indicated by this icon) means that querying any leaf dimension value from the trigger dimension will cause the target dimension value to be displayed (many triggers, one target).

Creating precedence rules

Create precedence rules from the Precedence Rules view.

To create a precedence rule:

1. On the Project tab, double-click **Precedence Rules** to open the Precedence Rules view.
2. Click **New.**
   The Precedence Rule editor appears.
3. In the Source Dimension value list, select the dimension that should serve as the trigger for the second, or target, value.
4. In the Target Dimension list, select the dimension that the source dimension value triggers.
5. In the Rule Type frame, choose standard or leaf.
6. Click **OK** to return to the Precedence Rule view.

Modifying precedence rules

Make all changes in the Precedence Rule editor, under the Precedence Rules view.

To edit a precedence rule:

1. In the Precedence Rules view, double-click the precedence rule you want to modify to open it in the Precedence Rule editor. (You can only open one precedence rule at a time.
2. Make the necessary changes in the Precedence Rule editor.
3. Click OK to return to the Precedence Rules view.

Deleting precedence rules

Delete precedence rules from the Precedence Rules view.

To delete a precedence rule:

1. In the Precedence Rules view, select the precedence rule you want to remove and click `Delete.`
2. When the confirmation message appears, click `Yes.`

Standard versus leaf precedence rules

Standard precedence rules display the target dimension if the trigger dimension value or its descendants are in the navigation state. Leaf precedence rules display the target dimension only after descendants of the trigger dimension value have been selected.

There are two types of precedence rules, standard and leaf. The types differ in how the trigger dimension value is interpreted.

- For the standard type, if the dimension value specified as the trigger, or any of its descendents, are in the navigation state, then the target dimension is displayed.
- For the leaf type, only leaf dimension values (dimension values with no children) that are descendents of the specified trigger dimension value cause the target dimension to be displayed. The presence of the specified trigger dimension value in the navigation state does not cause the target dimension to appear. Hence, a leaf precedence rule requires that the trigger dimension value has children.

A standard precedence rule is indicated by this icon in the Precedences Rules view. A leaf precedence rule is indicated by this icon

).

Standard Rule Example

In this standard rule example, we have a `Color` dimension with a child dimension value named `blue`. We can construct a standard precedence rule with `blue` as the trigger dimension value and the dimension `Shades of Blue` as the target. Once the user drills into `Color` and selects `blue,` the target dimension `Shades of Blue` is displayed in the user interface.

Leaf Rule Example

For leaf type rules, we'll examine a hierarchical dimension named Country and a second dimension named State/Province. The Country dimension looks like this:

- Country
- North America
- Canada
- Mexico
- United States
- Europe
- England
- Spain
- Italy

Logically, a user should choose a country before choosing a state or province. We can use a leaf precedence rule to suppress the display of the State/Province dimension until a leaf value in the Country dimension (an actual country as opposed to a continent) has been selected. To achieve this, a leaf precedence rule is constructed with the Country dimension root as the trigger and the State/Province dimension as the target. If the user drills into Country and selects an intermediate child dimension value (North America or Europe), the target dimension State/Province is not displayed. However, once the user has selected a leaf value from the Country dimension (United States, Canada, Mexico, England, Spain, or Italy) the State/Province dimension appears.

Precedence rules and the implicit selection of dimension values

When all records in the navigation state are assigned a given dimension value, that dimension value is an implicit selection.

In addition to being selected explicitly by the application, dimension values can be selected implicitly. For example, if all Champagnes are from France, then the explicit selection of Wine Type: Champagne causes the implicit selection of Region: France. Implicit selection is a function of the set of records in the navigation state, regardless of what combination of search, navigation, and record filters was used to obtain them.

Implicitly-selected dimension values trigger precedence rules in exactly the same way as explicitly-selected dimension values. This behavior helps ensure a consistent user experience, by providing the same dimensions for refinement of a given result set, regardless of whether that result set was obtained through search, navigation, or a combination of the two.

For this reason, two navigation paths leading to the same set of records will always have exactly the same set of navigation selections (differing only in whether the selections are implicit or explicit). Because of this equivalence, the set of precedence rules fired in both states will be identical.

If this behavior is undesirable, it can be suppressed or modified at the application layer. In some cases, implicitly-selected dimension values reflect redundancy in the data modeling, such as the presence of duplicate dimensions.

When precedence rules are overridden

Implicit selection of a precedence rule's trigger dimension value fires the rule. Under some circumstances, implicit selection of the rule's target dimension value also fires it.

Specifically, when a precedence rule's target dimension value is implicit in the navigation state, and when refinements are available underneath that target dimension value, the precedence rule fires and the target dimension is displayed. This occurs even when none of the rule's trigger values have been implicitly or explicitly selected. The MDEX Engine treats any precedence rules targeting the parent dimensions of these dimension values as having fired, even though the rules' trigger values have not been selected.

For this reason, precedence rule target dimensions may appear when no precedence rule trigger has been selected.

Tips and troubleshooting for precedence rules

This topic provides information which will help you use precedence rules, and troubleshoot issues you have with them.

The creation of dimensions can be facilitated with precedence rules. Consider the task of creating a Geography dimension as a hierarchy of country, state, and city. The hierarchy would need to be created manually, with Country as the root dimension value. Each country dimension value would have its corresponding states as children and each state its corresponding cities. In this scenario, the onus is on the knowledge worker to create and maintain this potentially enormous hierarchy.

Precedence rules offer a much simpler solution. The knowledge worker can produce the same results by creating three individual dimensions-Country, State, and City-and configuring precedence rules such that the State dimension isn't presented until a country has been chosen and the City dimension isn't presented until a state has been chosen. Each dimension is flat and can be auto-generated, so this solution involves much less initial and maintenance effort. Clearly, creating the hierarchy by hand is a much more difficult task than auto-generating the three dimensions, configuring precedence rules and letting contraction do the work to give the application the desired behavior (that is, to mimic the hierarchy).

Configuring User Profiles

This section provides information about working with the user profiles that trigger rules for merchandising and content spotlighting.

About user profiles

You can create user profiles to trigger dynamic business rules for merchandising and content spotlighting.

You can create user profiles to trigger dynamic business rules for merchandising and content spotlighting. A user profile is a label, such as premium_subscriber, that identifies an application user. If a user who has such a profile makes a query, the query triggers the associated rule and restricts merchandising and content spotlighting records to users who have that identity.

Note: Each business rule is allowed to have at most one user profile trigger.

Viewing all of your user profiles

Open the User Profiles view from the Project tab to see all user profiles in your project.

The User Profiles view contains information about user profiles you have created for your Endeca implementation.

This view has one column for the name of each user profile. You can sort the user profiles in the list by clicking Sort Ascending AZ↓ or Sort Descending ZA↓.

To view all of your user profiles at once:

- On the Project tab, double-click User Profiles. The User Profiles view opens and displays all of the user profiles in your project.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Creating user profiles

Set new user profiles from the User Profile editor, within the User Profiles view.

To create a user profile:

1. On the Project tab, double-click **User Profiles** to open the User Profiles view.
2. Click **New.**
   The User Profile editor appears.
3. Enter a name for the user profile.
4. Click **OK** to return to the User Profiles view.

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Modifying user profiles

Select any profile from the User Profiles view and open it in the editor.

To edit a user profile:

1. In the User Profiles view, double-click the profile you want to modify to open it in the User Profile editor.

   You can only open one user profile at a time.
2. Make the necessary changes in the User Profile editor.
3. Click **OK** to return to the User Profiles view.

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Deleting user profiles

Remove user profiles from the User Profiles view.

To delete a user profile:

1. In the User Profiles view, select the user profile you want to remove and click **Delete.**
2. When the confirmation message appears, click **Yes.**

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Specifying a user profile as a dynamic business rule trigger

Set a user profile as a rule trigger from the Rule editor in the Dynamic Business Rules view.

To specify a user profile as a dynamic business rule trigger:

1. In the Project Explorer, click **Dynamic Business Rules** to expand it.
2. Double-click **Rules** to open the Rules view.
3. Select the rule you want to apply the trigger to.
4. Click **Edit**.
   The Rule editor appears.
5. On the General tab, select a profile from the User Profile list.
6. Click **OK**.
   The user profile information that you added to the rule now appears in the Rules view.

Note: You can also assign a user profile as a business rule trigger in Endeca Workbench. See the Endeca Workbench Help for details.

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Configuring Search Interfaces

This section provides information on configuring the thesaurus, stop words, search characters, stemming, and spelling dictionary to refine the users' search experience.

Configuring Search

Refine user search by configuring thesaurus, stop words, search characters, stemming, and spelling dictionary features.

About search configuration

Developer Studio makes it possible for you to refine your users' search experience by making adjustments to the following features:

- Thesaurus

  You can establish a one-way or a two-way equivalence between words to enrich your users' query results. You access the Thesaurus editor from the Project tab.

- Stop words

  You can add stop words. Stop words are ignored by the Endeca search query engine. You access the Stop Words editor from the Project tab.

- Search characters

  You can add search characters to the search list. This allows non-alphanumeric characters to be indexed along with alphanumeric characters, rather than be treated as whitespace. You access the Search Characters editor from the `Edit > Search Characters` menu.

- Stemming

  You can enable or disable stemming for a variety of languages. Stemming defines sets of words (for example, "shirt" and "shirts") that should be considered strictly equivalent for all search operations. You access the Stemming editor using the `Edit > Stemming` command.

- Spelling dictionary

  You can tune the size of your application's spelling dictionary by instructing Dgidx to exclude small words, large words, and infrequently used words. You access the Spelling Dictionary editor using the Edit > Spelling dictionary command.

Using the Thesaurus

About the thesaurus

The thesaurus allows the system to return matches for related concepts to words or phrases contained in user queries.

It is a powerful tool that allows you to improve control over the vocabulary used in your application.

You can add two kinds of entries to your Endeca thesaurus:

- One-way thesaurus entries establish an equivalence relationship between words or phrases that applies in a single direction only. For example, you could define a one-way mapping so that all queries on Tools would also return matches containing Hammers, but queries on Hammers would not return results for the more general term Tools.
- Two-way thesaurus entries establish a mutual equivalence relationship between words or phrases. For example, an equivalence might specify that the phrase Mark Twain is interchangeable with the phrase Samuel Clemens.

Creating thesaurus entries

Create one-way or two-way thesaurus entries in the Thesaurus view, found under Search Configuration in the Project Explorer.

To create a one-way equivalence relationship between words or phrases:

1. In the Project Explorer, expand **Search Configuration** and double-click **Thesaurus.**
   The Thesaurus view appears.
2. In the Thesaurus view, click **New**, and then choose **One Way**.

The New One-Way Thesaurus Entry editor appears.

3. In the From box, type the word or phrase that, when selected, will also return results for the To entry.
4. In the To box, type the word or phrase the results for which will also be returned when the user's query returns the From entry, and then click **Add**.
5. Click **OK**.

   The new one-way thesaurus entry appears in the Thesaurus view, preceded by a red arrow ☒.

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Modifying thesaurus entries

Select an entry in the Thesaurus view to make changes.

To edit an existing thesaurus entry:

1. In the Thesaurus view, select the thesaurus entry you want to modify and click Edit.
   Depending on the entry you selected, either the One-Way Thesaurus Entry or the Two-Way Thesaurus Entry editor opens.
2. Make the necessary modifications.
3. Click **OK** to return to the Thesaurus view.

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Deleting thesaurus entries

Remove thesaurus entries from the Thesaurus view.

To remove an entry from the thesaurus list:

1. In the Thesaurus view, select the entry that you want to remove and then click **Delete.**
2. When the confirmation message appears, click **Yes.**

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Sorting thesaurus entries

Sort your entries in ascending or descending alphabetical order in the Thesaurus view.

The entries in Thesaurus view are sorted by name to make them easier to work with. You can choose whether the sort is ascending or descending.

To sort your thesaurus entries:

- In the Thesaurus view, click either the Sort Ascending AZ↓ or Sort Descending ZA↓ button

Note: Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Filtering thesaurus entries

If your list of thesaurus entries becomes long, you can filter them by a letter or word.

To filter thesaurus entries:

1. In the Thesaurus view, type a letter or word in the text box next to the **Clear Filter** button.
   The application filters your thesaurus entries dynamically.
2. To remove the filter, so that all of your thesaurus entries once again appear in the view, click **Clear Filter** .

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Thesaurus entry recommendations

Give single-word names to your entries, avoid non-searchable characters and stop words, and avoid creating entries which include substrings of other entries.

To avoid performance problems related to expensive or less than useful thesaurus search query expansions, follow these recommendations:

A thesaurus entry should never include a form that is a substring of another

Such forms rarely lead to useful expansions. For example, consider the following two entries:

- ADAM AND EVE
- EVE

If users type "EVE" they will get results for "EVE OR (ADAM AND EVE)"-the same results they would have gotten for "EVE" without the thesaurus. If the user types "ADAM AND EVE" they get results for (ADAM AND EVE) OR EVE, causing the "ADAM AND" part of the query to basically be ignored.

Stop words should not be used in thesaurus forms

For example, the following entries:

- ANCESTORS AND ANCESTRY
- HERITAGE
- HEREDITY

should be replaced with either:

- ANCESTORS
- ANCESTRY
- HERITAGE
- HEREDITY

or

- ANCESTORS ANCESTRY
- HERITAGE
- HEREDITY

The stop word "AND" should be removed.

Avoid multi-word thesaurus forms where single-word forms are appropriate

In particular, avoid multi-word forms that are not phrases that users are likely to type, or to which phrase expansion is likely to provide relevant additional results. For example, the following thesaurus entries:

- AETHELSTAN, KING OF ENGLAND (D. 939)
- ATHELSTAN, KING OF ENGLAND (D. 939)

should be replaced with the single-word form:

- AETHELSTAN
- ATHELSTAN

Thesaurus forms should not use non-searchable characters.

For example, the form:

- PIKE'S PEAK

should only be used if apostrophe is enabled as a search character.

One-way Thesaurus Entry editor
Specify a single word or phrase that, when searched on, returns results for other words or phrases.

Option	Description
From	The word or phrase that, when searched on, will also return hits on the words or phrases in the To list.
To	The additional words or phrases whose results will also be returned when the user's query returns hits on the From entry.
Add	Adds the word or phrase entered in the field above to the To list.
Modify	Used to modify a word or phrase in the To list.
Remove	Removed the selected word or phrase from the To list.

Two-way Thesaurus Entry editor
Define a list of words or phrases that will return results for each other when searched.

The Two-way Thesaurus Entry editor contains the following fields:

Option	Description
To	The list of words or phrases that will have a mutual equivalence relationship. Searches on any of the words in the list will return hits for the other words in the list as well.
Add	Adds the word or phrase entered in the field above to the To list.
Modify	Used to modify a word or phrase in the To list. To modify an entry, select it in the To list, make your changes in the editing field, and click Modify.
Remove	Removed the selected word or phrase from the To list.

Using Stemming

About stemming

The stemming feature broadens search results to include word roots and word derivations.

Stemming is intended to allow words with a common root form (such as the singular and plural forms of nouns) to be considered interchangeable in search operations. For example, search results for the word "shirt" will include the derivation "shirts," while a search for "shirts" will also include its word root "shirt."

Stemming equivalences are strictly two-way (that is, all-to-all). For example, a search for the singular form of a noun (such as "child") will also return matches for the plural form "children." Likewise, a search for "children" will return matches for "child" and "children." Stemmed words, therefore, are considered equivalent and interchangeable for all search operations.

In contrast, the thesaurus feature supports one-way mappings in addition to two-way mappings.

Note: Stemming files are provided by Endeca for various languages. While you can enable the use of these files, you cannot modify their contents.

Enabling stemming

Open the Stemming editor from the Edit menu to enable stemming.

To enable stemming for one or more languages in your project:

1. On the Edit menu, choose **Stemming.**
   The Stemming editor appears.
2. Check one or more of the language check boxes on the list.
3. Click **OK.**

Note: The Stemming editor allows you to turn the default version of Dutch and German stemming-with the word forms file-on and off. If you want to implement dynamic stemming for Dutch or German, you must edit the `stemming.xml` file directly, as described in the *Endeca Advanced Development Guide*. Subsequent use of the Stemming editor will not overwrite manual changes to the `stemming.xml` file. To disable stemming, use the above procedure, but uncheck the languages for which you do not want stemming.

Related Links

Using Stemming on page 149

Stemming editor

The Stemming editor allows you to enable or disable stemming for your supported languages.

The "Enable stemming for" list contains an entry for every supported language. A language that is checked has stemming enabled for that language.

Using Search Characters

About search characters

You can specify punctuation marks as searchable, in addition to digits and upper- and lower-case letters (automatically set as valid search characters).

Upper- and lower-case letters and the digits 0 to 9 are automatically included as valid search characters in your Endeca-enabled application. However, in the case of other characters, such as certain punctuation characters, you can specify whether the character should be indexed along with alphanumeric characters in a token or instead treated as whitespace.

Search characters are configured globally for all search operations.

Note: Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Adding search characters

Add search characters from the Search Characters editor, under the Edit menu.

To add search characters:

1. On the Edit menu, choose Search Characters.
   The Search Characters editor appears.
2. Click Modify.
   The Additional Search Characters dialog box appears.
3. Type the character(s) you want to add.
4. Click OK.

   If you have more than one, do not separate them with commas or spaces -- simply type them one after another.
5. Click Close.

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Deleting search characters

Delete search characters from the Search Characters editor, under the Edit menu.

To remove an additional character from the list of searchable characters:

1. On the Edit menu, choose **Search Characters**.
   The Search Characters editor appears.
2. Click **Modify**.
   The Additional Search Characters dialog box appears.
3. Highlight the character(s) you want to remove.
4. Press Delete.
5. Click **OK**.
   The confirmation message appears.
6. Click **Yes**.
7. Click **Close**.

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Search Characters editor

The Search Characters editor lists the standard special search characters, as well as any others you have specified.

Upper- and lower-case letters and the digits 0 to 9 are automatically included as valid search characters in your Endeca-enabled application. However, in the case of other characters, such as certain punctuation characters, you can specify whether the character should be indexed along with alphanumeric characters in a token or instead treated as whitespace.

Using Stop Words

About stop words

Stop words are words that are set to be ignored by the Endeca MDEX Engine.

Typically, common words like "the" are included in the stop word list. In addition, you might want to add terms that are prevalent in your data set. For example, if your data consists of lists of books, you might want to add the word "book" itself to the stop word list, since a search on that word would return an impracticably large set of records.

Note:

- Words added to the stop word list are not expanded by other Endeca Developer Studio features like stemming and thesaurus. That means that if you set the word "item" as a stop word, its plural form "items" will not be marked automatically as a stop word. If you want both forms to be on the stop word list, you must add them individually.
- Stop words are counted in any search mode (such as MatchPartial) that calculates results based on number of matching terms. However, the Endeca MDEX Engine reduces the minimum term match and maximum word omit requirement by the number of stop words contained in the query.
- Stop words must be single words only, and cannot contain any non-searchable characters. If more than one word is entered as a stop word, neither the individual words nor the combined phrase will act as a stop word. Non-searchable characters within a stop word will also cause this behavior. Entering "full-bodied" as a stop word acts just as if you had entered "full bodied", and does not have any effect on searches.

Creating stop words

Set stop words from the Stop Words view, under Search Configuration in the Project Explorer.

To add a word to the stop list:

1. In the Project Explorer, expand Search Configuration and double-click `Stop Words.`
   The Stop Words view appears.
2. In the Stop Words view, click `New.`
   The Stop Word editor appears.
3. Type the word that you want to add to the stop word list, and then click `OK.`

Modifying stop words

Modify stop words from the Stop Words view.

To edit a stop word:

1. In the Stop Word view, select the stop word that you want to modify, and then click **Edit.**
   The Stop Word editor appears.
2. Make the necessary changes to the stop word, and then click **OK.**

Deleting stop words

Remove stop words from the Stop Words view.

To remove a word from the stop word list:

1. In the Stop Words view, select the stop word you want to remove and click **Delete.**
2. When the confirmation message appears, click **Yes.**

Sorting stop words

Sort stop words in ascending or descending alphabetical order in the Stop Words view.

The entries in Stop Words view are sorted by name to make them easier to work with. You can choose whether the sort is ascending or descending.

To sort the stop words in your list:

- In the Stop Words view, click either the Sort Ascending AZ↓ or Sort Descending ZA↓ button.

Recommended stop words list

This list contains common words which you may want omitted from search results.

The following is a list of stop words that you may find useful:

- a
- about
- above
- an
- and
- any
- are
- can
- do
- find
- for
- from
- I
- is
- me
- not
- or
- over
- show
- the
- under
- what
- when
- where
- why
- with
- you
- your

Using Automatic Phrasing

About automatic phrasing

When an application user provides individual search terms in a query, the automatic phrasing feature groups those individual terms into a search phrase and returns query results for the phrase.

Automatic phrasing is similar to placing quotation marks around search terms before submitting them in a query. For example, 'my search terms' is the phrased version of the query my search terms. However, automatic phrasing removes the need for application users to place quotation marks around search phrases to get phrased results.

The result of automatic phrasing is that a Web application can process a more restricted query and therefore return fewer and more focused search results. This feature is available only for record search.

The automatic phrasing feature works by:

1. Comparing individual search terms in a query to a list of application-specific search phrases. The list of search phrases are stored in a project's phrase dictionary.
2. Grouping the search terms into search phrases.
3. Returning query results that are either based on the automatically phrased query, or returning results based on the original unphrased query along with automatically phrased 'Did You Mean?' (DYM) alternatives.

Point three above suggests the two typical implementation scenarios to choose from when using automatic phrasing:

- Process an automatically phrased form of the query and suggest the original unphrased query as a DYM alternative.

  In this scenario, the automatic phrasing feature rewrites the original query's search terms into a phrased query before processing it. If you are also using DYM, you can display the unphrased alternative so the user can opt-out of automatic phrasing and select their original query, if desired.

  For example, an application user searches a wine catalog for the terms "low tannin." The MDEX Engine compares the search terms against the phrase dictionary, finds a phrase entry for "low tannin," and processes the phrased query as "low tannin." The MDEX Engine returns 3 records for the phrased query "low tannin" rather than 16 records for the user's original unphrased query "low tannin." However, the Web application also presents a "Did you mean low tannin?" selection so the user may opt-out of automatic phrasing, if desired.
- Process the original query and suggest an automatically-phrased form of the query as a DYM alternative.

  In this scenario, the automatic phrasing feature processes the unphrased query as entered and determines if a phrased form of the query exists. If a phrased form is available, the Web application displays an automatically-phrased alternative as a "Did you mean?" option. The user can opt-in to automatic phrasing, if desired.

  For example, an application user searches a wine catalog for low tannin. The MDEX Engine returns 16 records for the user's unphrased query low tannin. The Web application also presents a "Did you mean "low tannin"?" option so the user may opt-in to automatic phrasing, if desired.

There are two tasks to implement automatic phrasing:

- Add phrases to your project using Developer Studio.
- Add Presentation API code to support either of the two implementation scenarios described above.

Note: Implementing search features requires additional work outside of Developer Studio. Refer to the *Endeca Advanced Development Guide* for details.

Automatic phrasing and query expansion
Grouping of terms as a phrase exempts the phrase from thesaurus expansion and stemming.

Once individual search terms in a query are grouped as a phrase, the phrase is not subject to thesaurus expansion or stemming by the MDEX Engine.

Automatic phrasing, spelling correction, and DYM
Describes the processing order of spelling correction and the DYM function with regard to automatic phrasing.

If you are using automatic phrasing, you should enable the MDEX Engine for both spelling correction and "Did you mean?" If you want spelling-corrected automatic phrases, spelling correction ensures search terms are corrected before the terms are automatically phrased. DYM provides users the choice to opt-in or opt-out of automatic phrasing.

The MDEX Engine applies spelling correction to a query before automatically phrasing the terms. This processing order means, for example, if a user misspells the query Napa Valle, the MDEX Engine first spell corrects to Napa Valley and then automatically phrases to "Napa Valley." Without spelling correction enabled, automatic phrasing would typically not find a matching phrase in the phrase dictionary.

If you implement automatic phrasing to rewrite the query using an automatic phrase, then enabling DYM allows users a way to opt-out of automatic phrasing if they want to. On the other hand, if you implement automatic phrasing to process the original query and suggest automatically-phrased alternatives, then enabling DYM allows users to take advantage of automatically phrased alternatives as follow-up queries.

Note: For details about configuring spelling correction and DYM, see the *Endeca Advanced Development Guide*.

Adding Phrases to a Project

Methods for adding phrases
Import phrases from an XML file, or extract phrases from dimension names.

There are two ways to include phrases in your Developer Studio project:

- Import phrases from an XML file.
- Extract phrases from dimension names.

After you add phrases and update your instance configuration, the MDEX Engine builds the phrase dictionary. You cannot view the phrases in Developer Studio. However, after adding phrases and saving your project, you can examine the phrases contained in a project's phrase dictionary by opening the project file named phrases.xml in a text editor. Directly modifying phrases.xml is not supported.

Importing phrases from an XML file
You import an XML file of phrases using the Import Phrases dialog box in Developer Studio. The Import Phrases dialog box can be accessed from either the File menu or from the Automatic Phrasing dialog box.

Before you import the XML file, it must conform to phrase_import.dtd, in the Endeca Navigation Platform conf/dtd directory.

Here is a simple example of a phrase file that conforms to phrase_import.dtd:

<?xml version='1.0' encoding='UTF-8' standalone='no' ?> <!DOCTYPE PHRASE_IMPORT SYSTEM 'phrase_import.dtd'>

<PHRASE_IMPORT>

<PHRASE>Napa Valley</PHRASE>

<PHRASE>low tannin</PHRASE>

</PHRASE_IMPORT>

To import phrases from an XML file:

1. In the Project Explorer, expand **Search Configuration**.
2. Double-click **Automatic Phrasing**.
   The Automatic Phrasing dialog box displays.
3. Click **Import Phrases**.
   The Import Phrases dialog box displays.

   Note: Alternatively, you can select Import Phrases from the File menu to invoke the Import Phrases dialog box.

4. Either type the path to your phrases file or click the **Browse** button to locate the file.
5. Click **OK** on the Import Phrases dialog box.
6. Click **OK** on the Automatic Phrasing dialog box.
   The Messages pane displays the number of phrases read in from the XML file.
7. Select **Save** from the File menu.

Extracting phrases from dimension names
In addition to importing an XML file of phrases, you can add phrases to your project based on the dimension values of any dimension you choose.

The MDEX Engine adds each multi-term dimension value in a selected dimension to the phrase dictionary. Single-term dimension values are not included. For example, if you import a Winery dimension from a wine catalog, the MDEX Engine creates a phrase entry for multi-term names such as Agostina Pieri but not for single-term names such as Alessi.

In addition, the MDEX Engine adds each multi-term synonym to the phrase dictionary that has "Search" checked on the Synonyms dialog box. In this release, dimension value phrases that have been modified by a partial update pipeline are not reflected in the phrase dictionary.

To extract phrases from dimension name:

1. In the Project Explorer, expand **Search Configuration** .
2. Double-click **Automatic Phrasing** .
   The Automatic Phrasing dialog box displays.
3. From the All dimensions list, select a dimension that you want to extract phrases from and click **Add**. Repeat this step as necessary for additional dimensions.
4. Click **OK**.
5. Select **Save** from the File menu.

Adding search characters that support automatic phrasing
Inclusion of original punctuation marks in search query phrases returns more relevant results.

To add search characters that support automatic phrasing:

If you have phrases that include punctuation, add those punctuation marks as search characters. Adding the punctuation marks ensures that the MDEX Engine includes the punctuation when tokenizing the query, and therefore the MDEX Engine can match search terms with punctuation to phrases with punctuation.

For example, suppose you add phrases based on a Winery dimension, and consequently the Winery name Anderson & Brothers exists in your phrase dictionary. You should create a search character for the ampersand (&).

Note: For details on search characters, see About search characters.

Tips and troubleshooting for automatic phrasing
Depending on how a phrased query is processed it may create dead-end results, for reasons including significance of term order and the fact that the MDEX Engine does not extend user phrases to match those in the phrase dictionary.

The following table provides tips and troubleshooting guidance about using the automatic phrasing feature.

Tip	Description
Examining how a phrased query was processed	
Single word phrases	You can include a single word in your phrases_import.xml file and treat the word as a phrase in your project. This may be useful if you do not want stemming or thesaurus expansion applied to single word query terms. You cannot include single word phrases by extracting them from dimension values using the Phrases dialog box. They have to be imported from your **phrases_import.xml** file.

Tip	Description
Extending user phrases	The MDEX Engine does not extend phrases a user provides to match a phrase in the phrase dictionary. For example, if a user provides the query A "BC" D and "BCD" is in the phrase dictionary, the MDEX Engine does not extend the user's original phrasing of "BC" to "BCD."
Term order is significant in phrases	Phrases are matched only if search terms are provided in the same exact order and with the same exact terms as the phrase in the phrase dictionary. For example, if "weekend bag" is in the phrase dictionary, the MDEX Engine does not automatically phrase the search terms "weekend getaway bag" or "bag, weekend" to match "weekend bag."
Possible dead ends	If an application automatically phrases search terms, it is possible a query may not produce results when it seemingly should have. Specifically, one way in which a dead-end query can occur is when a search phrase is displayed as a DYM link with results and navigation state filtering excludes the results. For example, suppose a car sales application is set up to process a user's original query and display any automatic phrase alternatives as DYM options. Further suppose a user navigates to Cars > Less than $15,000 and then provides the search terms luxury package. The search terms match the phrase 'luxury package' in the phrase dictionary. The user receives query results for Cars > Less than $15,000 and results that matched some occurances of the terms luxury and package. However, if the user clicks the DYM link Did you mean "luxury package"? then no results are available because the navigation state Cars > Less than $15,000 excludes them. **Note:** See the *Endeca Advanced Development Guide*for details about how processing order affects queries.

Using the Spelling Dictionary

About the spelling dictionary

By default, Dgidx creates an application-specific spelling dictionary based on all words contained in searchable dimensions and properties. These words become possible spell correction recommendations.

To achieve the best possible spelling correction behavior and performance, it is typically necessary to specify constraints on the list of words that Dgidx can include for spelling correction.

Specifying constraints on spelling dictionary content

Constraining spelling dictionary entries improves the performance of spelling corrected queries.

If you want to fine tune the size of the spelling dictionary and consequently tune the performance of spelling corrected queries, you can specify constraints to control what words Dgidx adds to the spelling dictionary. You can separately configure entries in the dictionary based for dimension search and record search.

To constrain spelling dictionary entries:

1. In the Project Explorer, expand **Search Configuration.**
2. Double-click **Spelling.**
3. Select the **Dimension Search** tab.
4. In the It Occurs at Least ... Times field, provide a number that indicates the minimum number of times the word must appear in your source data before the word should be included in the spelling dictionary.
5. In the And Is Between ... and ... Characters Long fields, provide values that represent the minimum and maximum length of a word that should be included in the spelling dictionary.
6. Select the **Record Search** tab.
7. Repeat steps 4 and 5.
8. Click **OK.**

Configuring Search Interface Options

Search interfaces allow your end-users to search on multiple properties and/or dimensions simultaneously.

About search interfaces

A search interface is a named collection of properties and dimensions, each of which has its Enable Record Search option checked. A search interface may also contain a number of attributes such as name, cross-field matching configuration, and an ordered collection of one or more ranking strategies that determine the order of search results.

The search interface's name is used just like a normal property or dimension when performing record searches. A record search query on a search interface returns results that match any of the properties or dimensions in the interface.

For example, if a data set contains both an Actor property and Director dimension, it would be useful to provide the end-user with the ability to search for a person's name in both. In this case you might create a search interface called People that contained both the Actor property and the Director dimension.

Viewing all of your search interfaces

All search interfaces are shown in the Search Interfaces view on the Project tab.

To view all of your search interfaces at once:

On the Project tab, double-click **Search Interfaces**.
The Search Interfaces view opens and displays all of the search interfaces in your project.

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Search Interfaces view

The Search Interfaces view displays all of the search interfaces established in your project, their member dimensions and properties, as well as any associated relevance ranking strategies.

The Search Interfaces view contains the following columns.

Field	Description
Name	A unique name for this search interface. **Note:** A search interface cannot share a name with a dimension value or a property.
Members	The dimensions and Endeca properties that make up this search interface.
Ranking Strategy	The ranking modules associated with this search interface.

Creating search interfaces

Create new search interfaces within the Search Interfaces editor of the Search Interfaces view.

To create a new search interface:

1. On the Project tab, double-click **Search Interfaces** to open the Search Interface view.
2. In the Search Interface view, click **New.**
   The Search Interfaces editor appears.
3. In the Name box, type the name of the new search interface.
4. From the Allow Cross-field Matches list, choose **Always, Never,** or **On Failure.** The Allow Cross-field Matches option specifies when the MDEX Engine should try to match search queries across dimension or property boundaries, but within the members of the search interface. There are three possible values:
   - Always—the MDEX Engine always looks for matches across dimension or property boundaries, in addition to matches within a dimension or property. This is the default value.
   - Never—the MDEX Engine does not look across dimension or property boundaries for matches.
   - On Failure—the MDEX Engine only tries to match queries across dimension or property boundaries if it fails to find any match within a single dimension or property.
5. In the All (Searchable) Members list, select a member and click **Add** to add it to the Selected Members list. Repeat as many times as necessary to add additional members to the search interface.

   Only Endeca properties and dimensions that have their **Enable record search** option checked appear in this list.
6. (Optional) If you want to associate a relevance ranking strategy to the search interface, click Relevance Ranking Modules.
7. (Optional) If you want to make more detailed adjustments to the search interface, click **Options** and configure the Customize partial match settings, which specify if partial matches for search terms should be supported for this search interface.

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Modifying search interfaces

Select a search interface from the Search Interfaces view to change it in the editor.

To edit a search interface:

1. In the Search Interfaces view, double-click the name of the search interface you want to modify to open it in the Search Interface editor. (You can only open one search interface at a time.
2. Make the necessary changes to the search interface.
3. Click OK to return to the Search Interfaces view.

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Deleting search interfaces

Remove search interfaces from the Search Interfaces view.

To delete a search interface:

1. In the Search Interfaces view, select the search interface you want to remove and click **Delete.**
2. When the confirmation message appears, click **Yes.**

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Search Interface editor

You use the Search Interface editor to create a new search interface or modify the attributes of an existing one.

The Search Interface editor contains the following fields:

Option	Description
Name	A unique name for this search interface. Search interface names are case sensitive.
Allow cross-field matches	Specifies when the MDEX Engine should try to match search queries across dimension or property boundaries, but within the members of the search interface. There are three possible values: • Always—the MDEX Engine always looks for matches across dimension or property boundaries, in addition to matches within a dimension or property. This is the default value. • Never—the MDEX Engine does not look across boundaries for matches. • On Failure—the MDEX Engine only tries to match queries across dimension or property boundaries

Option	Description
	if it fails to find any match within a single dimension or property.
All (searchable) members	A list of all dimensions and Endeca properties in the project that have the "Enable record search" field checked.
Selected members	A list of the searchable dimensions and properties that have been added to this search interface.
Relevance Ranking Modules	Clicking this button opens the Relevance Ranking Modules editor, where you can add, remove, and order the relevance ranking modules that compose the ranking strategy associated with this search interface.
Options	Clicking this button opens the Search Interface Options editor, where you can make more detailed adjustments to this search interface.

Allowing cross-field matches

The Allow Cross-field Matches option, in the Search Interface editor, specifies when the MDEX Engine should try to match search queries across dimension or property boundaries, but within the members of the search interface.

There are three possible values:

- Always (default)

  The MDEX Engine always looks for matches across dimension or property boundaries, in addition to matches within a dimension or property. This is the default.

- Never

  The MDEX Engine does not look across dimension or property boundaries for matches.

- On Failure

  The MDEX Engine only tries to match queries across dimension or property boundaries if it fails to find any matches within a single dimension or property.

Note: Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Customizing partial matching

This section describes customizations available in the Customize Partial Match Settings feature of Developer Studio.

The Customize Partial Match Settings feature specifies if partial matches for search terms should be supported for this search interface.

- Match at Least sets the minimum number of words that must be matched. The default is 2.
- Omit at Most sets the maximum number of words that can be omitted. The default is 2.

To customize partial matching in a search interface:

1. In the Search Interface Options editor, check **Customize Partial Match Settings.**
   The Match at Least ... Words and Omit at Most ... Words text boxes are each populated with 2 (the suggested value).
2. In the Match at Least ... Words text box, modify the minimum number of words that must match in order to consider a match.

   You cannot enter 0 for this value.
3. In the Omit at Most ... Words text box, modify the maximum number of words that can be omitted in order to consider a match.

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Using Snippeting

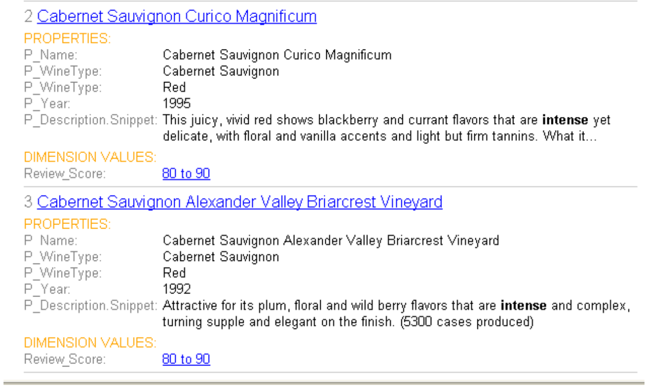
About snippeting

A snippet contains the search terms that the user provides along with a portion of the term's surrounding content to provide context.

The snippeting feature (also referred to as keyword in context) provides the ability to return an excerpt from a record—a snippet—to an application user who performs a record search query. A snippet contains the search terms that the user provides along with a portion of the term's surrounding content to provide context. A Web application displays these snippets on the record list page of a query's results. With the added context, a user can more quickly choose the individual records they are interested in.

You enable snippeting on individual members (fields) in a search interface that typically have many lines of content. For example, fields such as Description, Abstract, DocumentBody, and so on are good candidates to provide snippeting results.

For example, if a user searches for intense in a wine catalog, the record list for this query has many records that match intense. A snippet for each matching record displays on a record list page:



2 Cabernet Sauvignon Curico Magnificum

PROPERTIES:

P_Name:	Cabernet Sauvignon Curico Magnificum
P_WineType:	Cabernet Sauvignon
P_WineType:	Red
P_Year:	1995
P_Description.Snippet:	This juicy, vivid red shows blackberry and currant flavors that are **intense** yet delicate, with floral and vanilla accents and light but firm tannins. What it...

DIMENSION VALUES:

Review_Score:	80 to 90

3 Cabernet Sauvignon Alexander Valley Briarcrest Vineyard

PROPERTIES:

P_Name:	Cabernet Sauvignon Alexander Valley Briarcrest Vineyard
P_WineType:	Cabernet Sauvignon
P_WineType:	Red
P_Year:	1992
P_Description.Snippet:	Attractive for its plum, floral and wild berry flavors that are **intense** and complex, turning supple and elegant on the finish. (5300 cases produced)

DIMENSION VALUES:

Review_Score:	80 to 90

Snippet format and size

A snippet consists of search terms, surrounding context words, and ellipses.

A snippet can contain any number of search terms bracketed by `<endeca_term></endeca_term>` tags. The tags call out search terms and allow you to more easily reformat the terms for display in your Web application.

The snippet size is the total number of search terms and surrounding context words. You can configure the total number of words in a snippet as described in Enabling snippeting. In order to adhere to the size setting for a snippet, it is possible that the MDEX Engine may omit some search terms and context words from a snippet. This situation becomes more likely if an application user provides a large number of search terms and the maximum snippet size is comparatively small.

A snippet consists of one or more segments. The segments are delimited by ellipses in between them. Ellipses (...) indicate that there is text omitted from the snippet occurring before or after the ellipses.

For example, here is a snippet made up of two segments with a maximum size set at 20 words. The snippet resulted from a search for the search terms Scotland and British which are enclosed within `<endeca_term>` tags.

...in Edinburgh `<endeca_term>`Scotland`</endeca_term>,` and has been employed by Ford for 25 years ... He first joined Ford's `<endeca_term>`British`</endeca_term>` operation. Mazda motor ...

Note: Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Snippet property names

The MDEX Engine dynamically creates new snippet properties by appending `.Snippet` to the original name of the search interface members (fields) that you enabled for snippeting.

For example, if you enable snippeting for properties named `Description` and `Reviews,` the MDEX Engine creates new properties named `Description.Snippet and Reviews.Snippet` and returns these properties with the result set for a user's record search.

Note: Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Snippets are dynamically generated properties

The snippet property appears with a record only on a record list page.

It is important to emphasize that the MDEX Engine dynamically generates snippet properties. This means the snippet properties, unlike other Endeca properties, are not created, configured, or mapped using Developer Studio. A dynamically generated snippet property is not tagged to an Endeca record.

Note: Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Enabling snippeting

You enable the snippeting feature in the Member Options dialog box, which is accessed from the Search Interface editor.

You enable the snippeting feature in the Member Options dialog box, which is accessed from the Search Interface editor. Each member of a search interface is enabled and configured separately. In other words, snippeting results are enabled and configured for each member of a search interface and not for all members of a single search interface.

A search interface member is a dimension or property that has been enabled for search and that has been added to the Selected members pane of the Search Interface editor. You can enable and configure any number of individual search interface members. Each member that you enable produces its own snippet.

Enabling a member in one search interface does not affect that member if it appears in other search interfaces. For example, enabling the Description property for Search Interface A does not affect the Description property in Search Interface B.

To configure a search interface member for snippeting results:

1. Open your project file in Developer Studio and double-click **Search Interfaces.**

2. Either create a new search interface or select an existing one from the Search Interfaces view and click **Edit.**
3. From the Selected Member area of the Search Interface editor, click a member that you want to configure for snippeting.
4. Click **Edit.**
   The Member Options dialog box displays.
5. From the Member Options dialog box, check Enable snippeting.
6. Specify the maximum snippet size (number of words) a snippet can contain.
7. Click **OK.**
8. Repeat steps 3-7 if you want to configure additional search interface members.

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Troubleshooting snippets

You can increase the maximum size of snippets to include more context words.

If you are not seeing enough context words in your snippet, open the Member Options editor and increase the value for Maximum Snippet Size. The default value is 25 words.

Note: Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Basic Development Guide* for details.

Using Relevance Ranking Modules

About relevance ranking

You use relevance ranking to control the order in which record search results are displayed to the end-user.

You use relevance ranking to control the order in which record search results are displayed to the end-user. Typically, relevance ranking is used to ensure that the most important search results are displayed earliest to the user, since users are generally unlikely to page or scan through large result sets.

The importance of a particular record search result is generally an application-specific concept. Thus, the relevance ranking feature provides a flexible, configurable set of ranking modules. These modules are then grouped into strategies that can be used in combination to produce a wide range of relevance ranking effects. Each search interface has its own ranking strategy.

Relevance ranking contains a rich set of features that should be used advisedly. Misuse of relevance ranking strategies can cause unexpected results and degraded performance.

Creating ranking modules

Ranking modules are selected from a stock list of modules. The Static and Phrase modules both take parameters.

You may have multiple instances of the Static module, however, you should only have one instance of the Phrase module for each search interface.

To assign one or more ranking modules to a search interface:

1. In the Search Interface editor, click **Relevance Ranking Modules**.
   The Relevance Ranking Modules editor appears.
2. In the All Modules list, select a relevance ranking module and click **Add**.
   The module is moved to the Selected Modules list.

   Note: Selecting a module causes a brief description to appear in the frame in the lower left corner of the editor.

3. If you selected the Static module, edit the Static module parameters.
4. If you selected the Phrase module, edit the Phrase module parameters.
5. (Optional) Repeat step 2 to add additional modules to the search interface.
6. (Optional) Use the up ↑ and down ↓ arrows to adjust the relative rank of the modules in your ranking strategy.
7. Click **OK** to return to the Search Interface editor.

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Modifying ranking modules

Edit ranking modules from the Relevance Ranking Modules editor, in the Search Interface editor.

To edit a ranking module in a search interface:

1. In the Search Interfaces view, select the search interface you want to modify and click **Edit** to open it in the Search Interface editor.
2. In the Search Interface editor, click **Relevance Ranking Modules.**
   The Relevance Ranking Modules editor appears.
3. Make the necessary changes.
4. Click **OK** to return to the Search Interface editor.

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Deleting ranking modules

Delete ranking modules from the Relevance Ranking Modules editor.

To remove a ranking module from a search interface:

1. In the Search Interfaces view, select the search interface you want to modify and click **Edit** to open it in the Search Interface editor.
2. In the Search Interface editor, click **Relevance Ranking Modules.**
   The Relevance Ranking Modules editor appears.
3. In the Selected Modules list, select the module you want to delete and click **Remove.**
4. Click **OK** to return to the Search Interface editor.

Implementing search features requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* for details.

Changing the order of ranking modules

Select and move modules as needed from the Relevance Ranking Modules editor.

By default, ranking modules are evaluated in the order in which you created them.

To change the order of ranking modules:

1. In the Search Interfaces view, select the search interface you want to modify and click Edit to open it in the Search Interface editor.
2. In the Search Interface editor, click Relevance Ranking Modules.
   The Relevance Ranking Modules editor appears.
3. In the Selected Modules list, select a module that you want to move and click either the up arrow or the down arrow until the module is in the correct order.
4. (Optional) Repeat step 3 as needed.
5. Click OK to return to the Search Interface editor.

Editing Static module parameters

Edit static module parameters from the Relevance Ranking Modules editor.

The Static relevance ranking module, which indicates that a constant score be applied to a given result, is one of two modules that take parameters. You can apply it to a specific searchable dimension or property, and specify whether the records will be sorted in ascending or descending order. You can have multiple Static modules, as long as they have different configurations.

To rank the members of a dimension or property statically:

1. Open the Relevance Ranking Modules editor.
2. In the All Modules list, select Static and click Add. The Edit Static Relevance Rank Module editor appears.
3. In the New Property or Dimension list, select the property or searchable dimension to which you want to apply the static ranking module.
4. Check Sort Records in Descending Order if you want the resulting records sorted in that order. If you want the records sorted in ascending order (the default), make sure the checkbox is cleared.
5. Click OK to return to the Relevance Ranking Modules editor.

Editing Phrase module parameters

Edit phrase modules parameters from the Relevance Ranking Modules editor.

You can use only one Phrase module in any given search interface, but you can set all of your options in it.

The Phrase relevance ranking module states that results containing the user's query as an exact phrase, or a subset of the exact phrase, should be considered more relevant than matches simply containing the user's search terms scattered throughout the text. Phrase is one of two modules that take parameters.

To edit Phrase module parameters:

1. Open the Relevance Ranking Modules editor.
2. In the All Modules list, select Phrase and click **Add**. The Edit Phrase Relevance Rank Module editor appears.
3. Set the following options. See "Phrase" for detailed descriptions, interaction information, and examples of how to use these options.
   a) Rank based on length of subphrases: Ranks results based on the length of their subphrase matches. In other words, results that match three terms are considered more relevant than results that match two terms, and so on.
   b) Use approximate subphrase/phrase matching: When enabled, the Phrase module looks at a limited number of positions in each result that a phrase match could possibly exist, rather than all the positions.
   c) Apply spell correction, thesaurus, and stemming: When enabled, the Phrase module ranks results that match a phrase's expanded forms in the same stratum as results that match the original phrase.
4. Click **OK** to return to the Relevance Ranking Modules editor.

Recommended relevance ranking strategies

Relevance ranking contains a rich set of features that should be used advisedly.

Misuse of relevance ranking strategies can cause unexpected results and degraded performance. See the *Endeca Advanced Development Guide* for detailed information on relevance ranking and recommended strategies.

Ranking Modules in Detail

Exact

The Exact module provides a finer grained (but more computationally expensive) alternative to the Phrase module.

The Exact module groups results into three strata based on how well they match the query string:

- The highest stratum contains results whose complete text matches the user’s query exactly.

- The middle stratum contains results that contain the user's query as an exact substring.
- The lowest stratum contains other hits (such as normal conjunctive matches) .

The Exact module is computationally expensive, especially on large text fields. It is intended for use only on small text fields (such as dimension values or small property values like part IDs). This module should not be used with large or offline documents (such as FILE or ENCODED_FILE properties). Use of this module in these cases will result in very poor performance and/or application failures due to request timeouts. The Phrase module does similar but less sophisticated ranking and can be used as a higher performance substitute.

Field
The Field module ranks documents based on the search interface field with the highest priority in which it matched.

Only the best field in which a match occurs is considered. The Field module is often used in relevance ranking strategies for catalog applications, because the category or product name is typically a good match. Field assigns a score to each result based on the static rank of the dimension or property member or members of the search interface that caused the document to match the query .

In Developer Studio, static field ranks are assigned based on the order in which members of a search interface are listed in the Search Interfaces view. The first (left-most) member has the highest rank.

By default, matches caused by cross-field matching are assigned a score of zero. The score for cross-field matches can be set explicitly in Developer Studio by moving the <<CROSS_FIELD>> indicator up or down in the Selected Members list of the Search Interface editor. The <<CROSS_FIELD>> indicator is available only for search interfaces that have the Field module and are configured to support cross-field matches.

All non-zero ranks must be non-equal and only their order matters. For example, a search interface might contain both Title and DocumentContent properties, where hits on Title are considered more important than hits on DocumentContent (which in turn are considered more important than <<CROSS_FIELD>> matches). Such a ranking is implemented by assigning the highest rank Title, the next highest rank to DocumentContent, and setting the <<CROSS_FIELD>> indicator at the bottom of the Selected Members list in the Search Interface editor.

If a document matches on multiple fields, it is ranked based on the best field that it matches.

Note: The Field module is only valid for record search operations. This module assigns a score of zero to all results for other types of search requests.

First
Designed primarily for use with unstructured data, the First module ranks documents by how close the query terms are to the beginning of the document.

First groups its results into variably-sized strata. The strata are not the same size, because while the first word is probably more relevant than the tenth word, the 301st is probably not so much more relevant than the 310th word. This module takes advantage of the fact that the closer something is to the beginning of a document, the more likely it is to be relevant.

The First module works as follows:

- When the query has a single term, First's behavior is straight-forward: it retrieves the first absolute position of the word in the document, then calculates which stratum contains that position. The score for this document is based upon that stratum; earlier strata are better than later strata .
- When the query has multiple terms, First behaves as follows:

  1. The first absolute position for each of the query terms is determined.
  2. The median position of these positions is calculated. This median is treated as the position of this query in the document and can be used with stratification as described in the single word case.

- With query expansion (using stemming, spelling correction, or the thesaurus), the First module treats expanded terms as if they occurred in the source query. For example, the phrase glucose intolerence would be corrected to glucose intolerance (with intolerence spell-corrected to intolerance). First then continues as it does in the non-expansion case. The first position of each term is computed and the median of these is taken.
- In a partially matched query, where only some of the query terms cause a document to match, First behaves as if the intersection of terms that occur in the document and terms that occur in the original query were the entire query. For example, if the query *cat bird dog* is partially matched to a document on the terms *cat* and *bird*, then the document is scored as if the query were *cat bird*.

First's interaction with other features

First works for partial match modes, such as MatchPartial, as well as for MatchAll. For partial matches, First ranks documents based on the median position of the matching terms.

First does not work with Boolean searches, cross-field matching, or wildcard search. It assigns all such matches a score of zero.

Frequency
The Frequency (freq) module provides result scoring based on the frequency (number of occurrences) of the user's query terms in the result text.

Results with more occurrences of the user search terms are considered more relevant.

Frequency values are capped at 1024.

Glom
The Glom module ranks single-field matches ahead of cross-field matches.

This module serves as a useful tie-breaker function in combination with the Maximum Field module. It is only useful in conjunction with record search operations.

Interpreted
The Interpreted (interp) ranking module is a general-purpose module that assigns a score to each result based on the query processing techniques used to obtain the match.

Matching techniques considered include partial matching, cross-field matching, spelling correction, thesaurus, and stemming matching (discussed in detail in the *Endeca Advanced Development Guide*).

Specifically, the interpreted ranking module ranks results as follows:

- All non-partial matches are ranked ahead of all partial matches.
- Within the above layer, all single-field matches are ranked ahead of all cross-field matches.
- Within the above layer, all non-spelling-corrected matches are ranked above all spelling-corrected matches.
- Within the above layer, all non-thesaurus matches are ranked above all thesaurus matches.
- Within the above layer, all non-stemming matches are ranked above all stemming (word form) matches.

Maximum Field
Unlike Field, which assigns a static score to cross-field matches, Maximum Field selects the score of the highest-ranked field that contributed to the match.

The Maximum Field (maxfield) module behaves identically to the Field module, except in how it scores cross-field matches.

Because Maximum Field defines the score for cross-field matches dynamically, it does not make use of the <<CROSS_FIELD>> indicator set in the Search Interface editor.

Nterms
The Nterms module ranks matches according to how many query terms they match.

For example, in a three-word query, results that match all three words will be ranked above results that match only two, which will be ranked above results that match only one.

The Nterms module is only applicable to search modes where results can vary in how many query terms they match. These include MatchAny, MatchPartial, MatchAllAny, and MatchAllPartial. For details on specifying a search mode for a query, see the *Endeca Advanced Development Guide*.

Number of Fields
The Number of Fields (numfields) module ranks results based on the number of fields in the associated search interface in which a match occurs.

Note that we are counting whole-field rather than cross-field matches, for example, a result that matches two fields matches each field completely, while a cross-field match typically does not match any field completely.

Phrase
The Phrase module states that results containing the user's query as an exact phrase, or a subset of the exact phrase, should be considered more relevant than matches simply containing the user's search terms scattered throughout the text.

Note the following points about the Phrase module:

- If a query contains only one word, then that word constitutes the entire phrase and all of the matching results will be put into one stratum (score = 1).
- Because of the way hyphenated words are positionally indexed, Oracle recommends that you enable subphrase if your results contain hyphenated words.

Configuring the Phrase module

When you add the Phrase module in the Relevance Ranking Modules editor, you are presented with an editor that allows you to set these options.

The Phrase module has a variety of options that you use to customize its behavior:

- Rank based on length of subphrases
- Use approximate subphrase/phrase matching
- Apply spell correction, thesaurus, and stemming

Ranking based on sub-phrases

Subphrasing ranks results based on the length of their subphrase matches. In other words, results that match three terms are considered more relevant than results that match two terms, and so on.

When you configure the Phrase module, you have the option of enabling subphrasing.

A subphrase is defined as a contiguous subset of the query terms the user entered, in the order that he or she entered them. For example, the query "fax cover sheets" contains the subphrases "fax," "cover," "sheets," "fax cover," "cover sheets," and "fax cover sheets," but not "fax sheets."

Content contained inside nested quotation marks in a phrase is treated as one term. For example, consider the following phrase: "the question is 'to be or not to be.' " The quoted text, "to be or not to be," is treated as one query term, so this example consists of four query terms even though it has a total of nine words.

When subphrasing is not enabled, results are ranked into two strata: those that matched the entire phrase and those that didn't.

About approximate matching

The approximate setting is appropriate in cases where the runtime performance of the standard Phrase module is inadequate because of large result contents and/or high site load.

Approximate matching provides higher-performance matching, as compared to the standard Phrase module, with somewhat less exact results. With approximate matching enabled, the Phrase module looks at a limited number of positions in each result that a phrase match could possibly exist, rather than all the positions. Only this limited number of possible occurrences is considered, regardless of whether there are later occurrences that are better, more relevant matches.

Enabling positional indexing increases the number of occurrences that the Phrase module looks at, thereby increasing the accuracy of the approximate phrase matching results. See Using positional indexing with the Phrase module for more information.

Applying spelling correction, thesaurus, and stemming with the Phrase module

Describes available functions with query expansion enabled.

Applying spelling correction, thesaurus, and stemming adjustments to the original phrase is generically known as query expansion. With query expansion enabled, the Phrase module ranks results that match a phrase's expanded forms in the same stratum as results that match the original phrase. Consider the following example:

- A thesaurus entry exists that expands "US" to "United States."
- The user queries for "US government."

The query, "US government," is expanded to "United States government" for matching purposes, but the Phrase module gives a score of two to any results matching "United States government" because the original, unexpanded version of the query, "US government," only had two terms.

Editing Phrase module parameters

Edit phrase modules parameters from the Relevance Ranking Modules editor.

You can use only one Phrase module in any given search interface, but you can set all of your options in it.

The Phrase relevance ranking module states that results containing the user's query as an exact phrase, or a subset of the exact phrase, should be considered more relevant than matches simply containing the user's search terms scattered throughout the text. Phrase is one of two modules that take parameters.

To edit Phrase module parameters:

1. Open the Relevance Ranking Modules editor.
2. In the All Modules list, select Phrase and click **Add**. The Edit Phrase Relevance Rank Module editor appears.
3. Set the following options. See "Phrase" for detailed descriptions, interaction information, and examples of how to use these options.
   a) Rank based on length of subphrases: Ranks results based on the length of their subphrase matches. In other words, results that match three terms are considered more relevant than results that match two terms, and so on.
   b) Use approximate subphrase/phrase matching: When enabled, the Phrase module looks at a limited number of positions in each result that a phrase match could possibly exist, rather than all the positions.
   c) Apply spell correction, thesaurus, and stemming: When enabled, the Phrase module ranks results that match a phrase's expanded forms in the same stratum as results that match the original phrase.
4. Click **OK** to return to the Relevance Ranking Modules editor.

Summary of Phrase module option interactions

You should only use one Phrase module in any given search interface and set all of your options in it.

The three configuration settings for the Phrase module can be used in a variety of combinations for different effects. The following matrix describes the behavior of each combination. You should only use one Phrase module in any given search interface and set all of your options in it.

Subphrase	Approximate	Expansion	Behavior
Off	Off	Off	Default. Ranks results into two strata: those that match the user's query as a whole phrase, and those that do not.

Subphrase	Approximate	Expansion	Behavior
Off	Off	On	Ranks results into two strata: those that match the original, or an extended version, of the query as a whole phrase, and those that do not.
Off	On	Off	Ranks results into two strata: those that match the original query as a whole phrase, and those that do not. Look only at the first possible phrase match within each record.
Off	On	On	Ranks results into two strata: those that match the original, or an extended version, of the query as a whole phrase, and those that do not. Look only at the first possible phrase match within each record.
On	Off	Off	Ranks results into N strata where N equals the length of the query and each result's score equals the length of its matched subphrase.
On	Off	On	Ranks results into N strata where N equals the length of the query and each result's score equals the length of its matched subphrase. Extend subphrases to facilitate matching but rank based on the length of the original subphrase (before extension).
On	On	Off	Ranks results into N strata where N equals the length of the query and each result's score equals the length of its matched subphrase. Look only at the first possible phrase match within each record.
On	On	On	Ranks results into N strata where N equals the length of the query and each result's score equals the length of its matched subphrase. Expand the query to facilitate matching but rank based on the length of the original subphrase (before extension). Look only at the first possible phrase match within each record.

Effects of match modes on Phrase behavior

Phrase, like the other relevance ranking modules, is never applied to the results of MatchBoolean queries.

Endeca provides a variety of match modes to facilitate matching during search (MatchAny, MatchAll, MatchPartial and so on). These modes only determine which results match a user's query; they have no effect on how the results are ranked after the matches have been found. Therefore, the Phrase module works as described in this section, regardless of match mode. The one exception to this rule is MatchBoolean.

Stop words and Phrase module behavior

When using the Phrase module, stop words are always treated like non-stop word terms and stratified accordingly.

For example, the query 'raining cats and dogs' will result in a rank of two for a result containing 'fat cats and hungry dogs' and a rank of three for a result containing 'fat cats and dogs' (this example assumes subphrase is enabled).

Phrase module behavior with cross-field matches and multiple matches

If a single result has multiple subphrase matches, either within the same field or in several different fields, the result is slotted into a stratum based on the length of the longest subphrase match.

An entire phrase, or subphrase, must appear in a single field in order for it to be considered a match. (In other words, matches created by glomming fields together are not considered by the Phrase module.)

Treatment of wildcards by the Phrase module

The Phrase module translates each wildcard in a query into a generic placeholder for a single term.

For example, the query "sparkling w* wine" becomes "sparkling * wine" during phrase relevance ranking, where "*" indicates a single term. This generic wildcard replacement causes slightly different behavior when subphrasing is and isn't enabled.

When subphrasing is not enabled, all results that match the generic version of the wildcard phrase exactly are still placed into the first stratum. It is important, however, to understand what constitutes a matching result from the Phrase module's point of view.

Consider the search query "sparkling w* wine" with the MatchAny mode enabled. In MatchAny mode, search results only need to contain one of the requested terms to be valid, so a list of search results for this query could contain phrases that look like this:

- sparkling white wine
- sparkling refreshing wine
- sparkling wet wine
- sparkling soda
- wine cooler

When phrase relevance ranking is applied to these search results, the Phrase module looks for matches to "sparkling * wine" not "sparkling w* wine." Therefore, there are three results-"sparkling white wine," "sparkling refreshing wine," and "sparkling wet wine"-that are considered phrase matches for the purposes of ranking. These results are placed in the first stratum. The other two results are placed in the second stratum.

When subphrasing is enabled, the behavior becomes a bit more complex. Again, we have to remember that wildcards become generic placeholders and match any single term in a result. This means that any subphrase that is adjacent to a wildcard will, by definition, match at least one additional term (the wildcard). Because of this behavior, subphrases break down differently. The subphrases for "cold sparkling w* wine" break down into the following (note that w* changes to *):

- cold
- sparkling *
- * wine
- cold sparkling *
- sparkling * wine
- cold sparkling * wine

Notice that the subphrases "sparkling," "wine," and "cold sparkling" are not included in this list. Because these subphrases are adjacent to the wildcard, we know that the subphrases will match at least one additional term. Therefore, these subphrases are subsumed by the "sparkling *", "* wine", and "cold sparkling *" subphrases.

Like regular subphrase, stratification is based on the number of terms in the subphrase, and the wildcard placeholders are counted toward the length of the subphrase. To continue the example above, results that contain "cold" get a score of one, results that contain "sparkling *" get a score of two, and so on. Again, this is the case even if the matching result phrases are different, for example, "sparkling white" and "sparkling soda."

Finally, it is important to note that, while the wildcard can be replaced by any term, a term must still exist. In other words, search results that contain the phrase "sparkling wine" are not acceptable matches for the phrase "sparkling * wine" because there is no term to substitute for the wildcard. Conversely, the phrase "sparkling cold white wine" is also not a match because each wildcard can be replaced by one, and only one, term. Even when wildcards are present, results must contain the correct number of terms, in the correct order, for them to be considered phrase matches by the Phrase module.

Proximity

Designed primarily for use with unstructured data, the Proximity module ranks how close the query terms are to each other in a document by counting the number of intervening words.

Like First, this module groups its results into variable sized strata, because the difference in significance of an interval of one word and one of two words is usually greater than the difference in significance of an interval of 21 words and 22.

Single words and phrases get assigned to the best stratum because there are no intervening words. When the query has multiple terms, Proximity behaves as follows:

1. All of the absolute positions for each of the query terms are computed.
2. The smallest range that includes at least one instance of each of the query terms is calculated. This range's length is given in number of words. The score for each document is the strata that contains the difference of the range's length and the number of terms in the query; smaller differences are better than larger differences.

Under query expansion (that is, stemming, spelling correction, and the thesaurus), the expanded terms are treated as if they were in the query, so the proximity metric is computed using the locations of the expanded terms in the matching document.

For example, if a user searches for big cats and a document contains the sentence, "Big Bird likes his cat" (stemming takes cats to cat), then the proximity metric is computed just as if the sentence were, "Big Bird likes his cats."

Proximity scores partially matched queries as if the query only contained the matching terms. For example, if a user searches for cat dog fish and a document is partially matched that contains only cat and fish, then the document is scored as if the query cat fish had been entered.

Proximity interacts with other features as follows:

- Proximity works for partial match modes, such as Match Partial, as well as for MatchAll. For partial matches, Proximity ranks documents based on the median position of the matching terms.
- Proximity does not work with Boolean searches, cross-field matching, or wildcard search. It assigns all such matches a score of zero.

Spell

The Spell module ranks true matches ahead of spelling-corrected matches.

Static

The Static module assigns a static or constant data-specific value to each search result, depending on the type of search operation performed and depending on optional parameters that can be passed to the module.

For record search operations, the first parameter to the module specifies a property, which will define the sort order assigned by the module. The second parameter can be specified as ascending or descending to indicate the sort order to use for the specified property.

For example, using the module *Static(Availability,descending)* would sort result records in descending order with respect to their assignments from the Availability property. Using the module *Static(Title,ascending)* would sort result records in ascending order by their Title property assignments. In a catalog application, setting the static module by *Price, descending* leads to more expensive products being displayed first.

For dimension search, the first parameter can be specified as nbins, depth, or rank:

- Specifying nbins causes the static module to sort result dimension values by the number of associated records in the full data set.
- Depth causes the static module to sort result dimension values by their depth in the dimension hierarchy.
- Rank causes dimension values to be sorted by the ranks assigned to them for the application.

Note: the Static module is not compatible with the Agraph.

Weighted Frequency
The Weighted Frequency module scores results based on frequency of user terms, and weights those frequencies for each result by the content of each term.

Like the Frequency module, the Weighted Frequency (wfreq) module scores results based on the frequency of user query terms in the result. Additionally, the Weighted Frequency module weights the individual query term frequencies for each result by the information content (overall frequency in the complete data set) of each query term. Less frequent query terms (that is, terms that would result in fewer search results) are weighted more heavily than more frequently occurring terms.

Weighted Frequency values are capped at 1024.

Configuring Keyword Redirects

This section provides information on using Developer Studio to configure keyword redirects that redirect a user's search to a specific page.

About keyword redirects

Keyword redirects, like dynamic business rules, have trigger and target values. The trigger of a keyword redirect is one or more search terms; the target of a keyword redirect is a URL.

Keyword redirects are used to redirect a user's search to a Web page (that is, a URL). Conceptually, keyword redirects are similar to dynamic business rules in that both have trigger and target values. The trigger of a keyword redirect is one or more search terms; the target of a keyword redirect is a URL. If users search with the particular keyword, the redirect URL displays in the application.

For example, you can set up a keyword redirect with a single keyword trigger of "delivery" and a redirect URL of http://shipping.acme.com. Or you might create a keyword redirect with a trigger of "stores" and a redirect URL of http://www.acme.com/store_finder.htm.

Multiple keyword redirect triggers

A keyword redirect may have multiple triggers. In this case, each trigger is separated by an OR. A user's query that exactly matches any trigger is sufficient to display the redirect URL. For example 'store' OR 'location' are two triggers in a keyword redirect where each is a single term. A user's query that exactly matches either term displays the redirect URL.

A trigger may have more than one search term. In this case, the trigger is treated as a search phrase. For example 'store finder' or "location" are two triggers in a keyword redirect where the first trigger is a phrase and the second trigger is a single term. A user's query that exactly matches either the phase 'store finder' or the term 'location' displays the redirect URL.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

Creating keyword redirects

Create new keyword redirects from the Keyword Redirects view in the Project Explorer.

To create a keyword redirect in Developer Studio:

1. In the Project Explorer, double-click **Keyword Redirects.**
   This opens the Keyword Redirects view and also activates the Redirect menu on the menu bar.
2. From the Redirect menu, select **New.**
   The Keyword Redirects editor displays.
3. In the Keyword text box, enter a keyword.
4. Click **Add.** If necessary, add additional keywords.
5. In the Redirect Link text box, enter the URL that loads in a user's browser if a user searches the associated keywords.
6. Click **OK.**

   You can add the keyword redirect to the default redirect group, or if you created additional redirect groups, select a redirect group to which this redirect belongs. For more information, see

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

Modifying keyword redirects

Open redirects in the Keyword Redirects view to modify them in the Keyword Redirects editor.

To modify a keyword redirect:

1. In the Keyword Redirects view, double-click the redirect you want to modify to open it in the Keyword Redirects editor.
2. Select a keyword trigger.
3. Revise.
4. Click **Modify.**
5. Click **OK** to return to the Redirects view.

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

Deleting keyword redirects

Remove keyword redirects from the Keyword Redirects view.

To delete a keyword redirect:

1. In the Keyword Redirects view, select the redirect you want to remove from your project, and click **Delete.**
2. When the confirmation message appears, click **Yes.**

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

Keyword Redirect editor

Edit keyword triggers and redirect link.

The Keyword Redirect editor contains the elements of a keyword redirect - one or more keyword triggers and the redirect link.

Grouping Keyword Redirects

About keyword redirect groups

Keyword redirect groups let you categorize your keyword redirects, as well as let multiple users access keyword redirects at the same time.

Keyword redirect groups serve two functions:

- They provide a means to logically organize keyword redirects into categories. This provides a means to organize a large number of keyword redirects into smaller logical categories.
- They allow multiple users to access keyword redirects simultaneously. Each Endeca Workbench user can access a single group at a time. Once a user selects a keyword redirect group, Workbench prevents other users from editing that group until the user returns to the group selection list or closes the browser window.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

Creating a keyword redirect group

Open the Redirect Group menu from the Keyword Redirects view to create a new keyword redirect group.

To create a keyword redirect group:

1. In the Project Explorer window, double-click Keyword Redirects.
   This opens the Redirects view and also activates the Redirect Group menu on the menu bar.
2. From the `Redirect Group` menu, select `New`.
3. Type a unique name in the Group name field. Use only alphanumeric, dash, or underscore characters. The name must be unique across both keyword redirect groups and rule groups.
4. To select a keyword redirect for this group, highlight a keyword direct in the `All Redirects` list and click `Add`.
   The keyword redirect appears in the `Redirects` in `Group` list.
5. Click `OK` .
   The new rule group appears in the `Rules` view.
6. Repeat steps 2 to 5 if you want multiple rule groups in your project.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

Deleting a keyword redirect group

Delete keyword redirect groups from the Keyword Redirects view.

To delete a keyword redirect group:

1. In the Project Explorer, double click **Keyword Redirects** to open the `Redirects` view.
2. In the Name column, select a keyword redirect group and then click **Delete.**
3. When the confirmation message appears, click **Yes.**
4. If your project contains more than one other group, the `Select Redirect Group` dialog box appears. In the drop-down list, select the redirect group you want to move the keyword redirects in this group to, and click **OK.**

If the group you are deleting is the second to last group, the redirects are moved to the one remaining redirect group.

Redirect Group editor

The Redirect group editor contains the following fields:

Option	Description
Group name	A unique name for this keyword redirect group.
All redirects	Displays all of the keyword redirects that have been defined in your project.
Redirects in group	Displays the keyword redirects that are included in this keyword redirect group.
Add	Adds the selected keyword redirect from the "All redirects" list to the keyword redirect group.
Remove	Removes the selected keyword redirect from the "Redirects in group" list.

Configuring Dynamic Business Rules

This section provides information on using Developer Studio to create rules that promote records, to define which records to promote, and to indicate how to display the records in an application.

Using dynamic business rules to promote records

You implement merchandising and content spotlighting using dynamic business rules. The rules and their supporting constructs define when to promote records, which records may be promoted, and also indicate how to display the records to application users.

Endeca provides the functionality to promote contextually relevant records to application users as they search and navigate within a data set. In catalog applications, this activity is called merchandising because the Endeca records you promote often represent product data. In document repositories, this activity is called content spotlighting™ because the Endeca records you promote often represent some type of document (HTML, DOC, TXT, XLS, and so on). For the sake of simplicity, this help system uses 'promoting records' to generically describe both merchandising and content spotlighting.

Here is a simple merchandising example using a wine data set. An application user enters a query with the search term Bordeaux. This search term triggers a rule that is set up to promote wines tagged as Best Buys. In addition to returning standard query results for term Bordeaux, the rule instructs the MDEX Engine to dynamically generate a subset of records that are tagged with both the Best Buy and also Bordeaux properties. The Web application displays the standard query results that match Bordeaux and also displays some number of the rule results in an area of the screen set aside for 'Best Buy' records. These are the promoted records.

Effect of rules on query results

Once you implement dynamic business rules in your application, each query a user makes is compared to each rule to determine if the query triggers a rule. If a user's query triggers a rule, the MDEX Engine returns several types of results.

- Standard record results for the query.
- Promoted records specified by the triggered rule's target.
- Any rule properties specified for the rule.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

Comparing dynamic business rules to content management publishing

In rule-based merchandising, a dynamic business rule specifies how to query for records to promote, and not necessarily what the specific records are.

Endeca's record promotion works differently from traditional content management systems (CMS), where you select an individual record for promotion, place it on a template or page, and then publish it to a Web site. Endeca's merchandising is dynamic, or rule based.

This means that, as your users navigate or search, they continue to see relevant results, because appropriate rules are in place. Also, as records in your data set change, new and relevant records are returned by the same dynamic business rule. The rule remains the same, even though the promoted records may change.

In a traditional CMS scenario, if Wine A is Recommended, it is identified as such and published onto a static page. If you need to update the list of recommended wines to remove Wine A and add Wine B to the static page, you must manually remove Wine A, add Wine B, and publish the changes.

With Endeca's dynamic record promotion, the effect is much broader and requires much less maintenance. A rule is created to promote wines tagged as Recommended, and the search results page is designed to render promoted wines. In this scenario, a rule promotes recommended Wine A on any number of pages in the result set. In addition, removing Wine A and adding Wine B is simply a matter of updating the source data to reflect that Wine B is now included and tagged as Recommended. After making this change, the same rule can promote Wine B on any number of pages in the result set, without adjusting or modifying the rule or the pages.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* and the *Oracle Endeca Workbench Help* for details.

Dynamic business rule constructs

A trigger and a target are required constructs for a business rule; a zone, style, and rule group are additional constructs that support rules.

Two constructs make up a dynamic business rule:

- Trigger--a set of navigation locations that must exist in a query for a rule to file. A trigger may include dimension values and keywords. When a user's query contains a navigation location that triggers a rule, the MDEX Engine fires the rule and returns a set of records that are candidates for promotion to application users.

- Target--specifies which records are eligible for promotion to application users. A target may include dimension values, custom properties, and featured records. For example, dimension values in a trigger are used to identify a set of records that are candidates for promotion to application users.

Three additional constructs support rules:

- Zone—specifies a collection of rules to ensure that rule results are produced in case a single rule does not provide a result.
- Style—specifies the minimum and maximum number of records a rule can return. A style also specifies any property templates associated with a rule. Rule properties are key/value pairs that are typically used to return supplementary information with promoted record pages. For example, a property key might be set to SpecialOffer and its value set to BannerAd.gif.

  Note: A rule's style is passed back along with the rule's results, to the Web application. The Web application uses the style as an indicator for how to render the rule's results. The code to render the rule's results is part of the Web application, not the style itself.

- Rule Group—provides a means to logically organize large numbers of rules into categories. This organization facilitates editing by multiple business users.

The core of a dynamic business rule is its trigger values and target values. The target identifies a set of records that are candidates for promotion to application users. The zone and style settings associated with a rule work together to restrict the candidates to a smaller subset of records that the Web application then promotes.

Record promotion examples

Read these examples for a better understanding of how to use dynamic business rules to promote Endeca records.

The following sections describe two examples of using dynamic business rules to promote Endeca records. The first example shows how a single rule provides merchandising results when an application user navigates to a dimension value in a data set. The scope of the merchandising coverage is somewhat limited by using just one rule.

The second example builds on the first by providing broader merchandising coverage. In this example, an application user triggers two additional dynamic business rules by navigating to the root dimension value for the application. These two additional rules ensure that merchandising results are always presented to application users.

An example with one rule promoting records

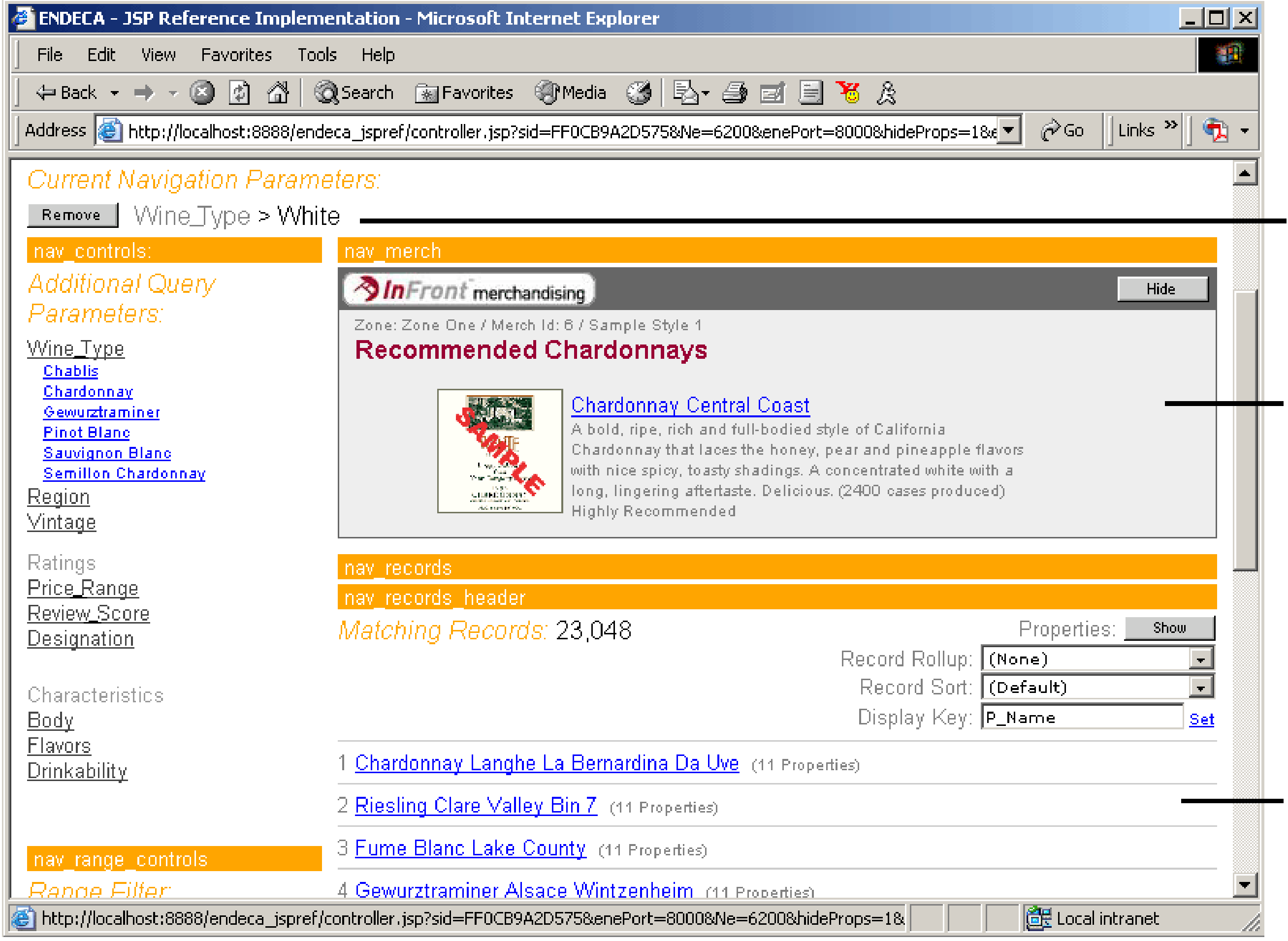
This simple example demonstrates a basic merchandising scenario where an application user navigates to Wine Type > White, and a dynamic business rule called "Recommended Chardonnays" promotes chardonnays that have been tagged as Highly Recommended. From a merchandising perspective, the marketing assumption is that users who are interested in white wines are also likely to be interested in highly recommended chardonnays.

The Recommended Chardonnays rule is set up as follows:

- The rule's trigger, which specifies when to promote records, is the dimension value Wine Type > White.
- The rule's target, which specifies which records to promote, is a combination of two dimension values, Wine Type > White > Chardonnay and Designation > Highly Recommended.
- The style associated with this rule is configured to provide a minimum of at least one promoted record and a maximum of exactly one record.
- The zone associated with this rule is configured to allow only one rule to produce rule results.

When an application user navigates to Wine Type > White in the application, the rule is triggered. The MDEX Engine evaluates the rule and returns promoted records from the combination of the Chardonnay and Highly Recommended dimension values. There may be a number of records that match these two dimension values, so zone and style settings restrict the number of records actually promoted to one.

The promoted record along with the user's query and standard query results are called out in the following graphic:


ENDECA - JSP Reference Implementation - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost:8888/endeca_jspref/controller.jsp?sid=FF0CB9A2D575&Ne=6200&enePort=8000&hideProps=1&e
Go
Links

Current Navigation Parameters:

Remove Wine_Type > White

nav_controls:

Additional Query Parameters:

Wine_Type
- Chablis
- Chardonnay
- Gewurztraminer
- Pinot Blanc
- Sauvignon Blanc
- Semillon Chardonnay

Region
Vintage

Ratings
Price_Range
Review_Score
Designation

Characteristics
Body
Flavors
Drinkability

nav_range_controls

Range Filter

nav_merch

InFront merchandising Hide

Zone: Zone One / Merch Id: 6 / Sample Style 1

Recommended Chardonnays

Chardonnay Central Coast

A bold, ripe, rich and full-bodied style of California Chardonnay that laces the honey, pear and pineapple flavors with nice spicy, toasty shadings. A concentrated white with a long, lingering aftertaste. Delicious. (2400 cases produced)
Highly Recommended

nav_records

nav_records_header

Matching Records: 23,048

Properties: Show
Record Rollup: (None)
Record Sort: (Default)
Display Key: P_Name Set

1 Chardonnay Langhe La Bernardina Da Uve (11 Properties)

2 Riesling Clare Valley Bin 7 (11 Properties)

3 Fume Blanc Lake County (11 Properties)

4 Gewurztraminer Alsace Wintzenheim (11 Properties)

http://localhost:8888/endeca_jspref/controller.jsp?sid=FF0CB9A2D575&enePort=8000&Ne=6200&hideProps=1&
Local intranet

User's query

Rule results
Recommend

Standard res
query Wine T

An expanded example with three rules

The previous example used just one rule to merchandise highly recommended chardonnays. The following example expands on the previous one by adding two more rules called Best Buys and Highly Recommended. These rules merchandise wines tagged with a Best Buy property and a Highly Recommended property, respectively. Together, the three rules merchandise records to expose a broader set of potential wine purchases.

The Best Buys rule is set up as follows:

- The rule's trigger is set to the Web application's root dimension value. In other words, the trigger always applies.
- The rule's target is the dimension value named Best Buy.
- The style associated with this rule is configured to provide a minimum of four promoted records and a maximum of eight records.

- The zone associated with this rule is configured to allow only one rule to produce rule results.

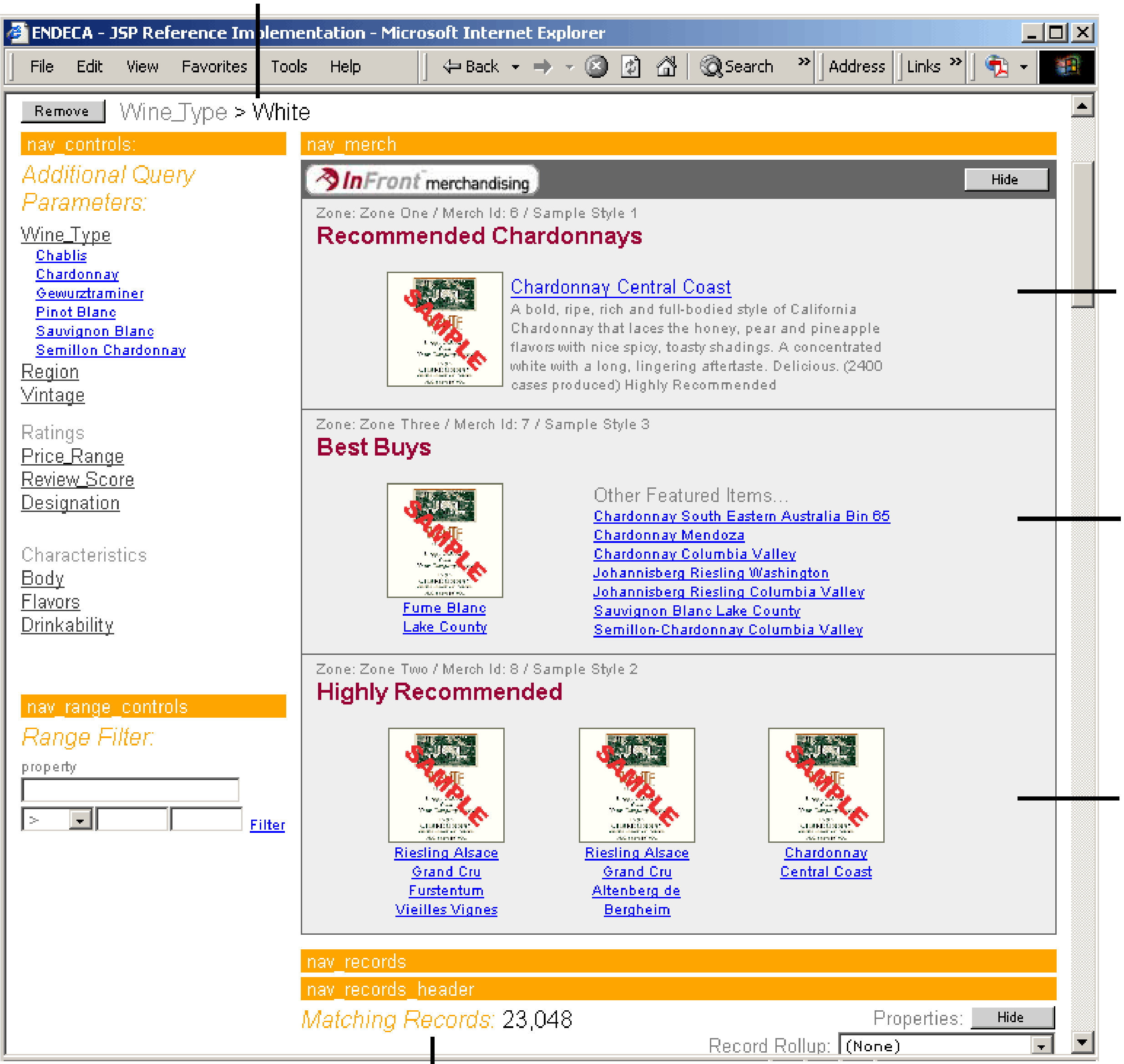
The Highly Recommended rule is set up as follows:

- The rule's trigger is set to the Web application's root dimension value. In other words, the trigger always applies.
- The rule's target is the dimension value named Highly Recommended.
- The style associated with this rule is configured to provide a minimum of at least one promoted record and a maximum of three records.
- There is the only rule associated with the zone, so no other rules are available to produce results. For details on how zones can be used when more rules are available, see See Ensuring Promoted Records are Always Produced.

When an application user navigates to Wine Type > White, the Recommended Chardonnays rule fires and provides rule results as described in 'An example with one rule promoting records' above. In addition, the Highly Recommended and Best Buys rules also fire and provide results because their triggers always apply to any navigation query.

The promoted records for each of the three rules, along with the user's query, and standard query results are called out in the following graphic:

User's query and trigger value



A bold, ripe, rich and full-bodied style of California Chardonnay that laces the honey, pear and pineapple flavors with nice spicy, toasty shadings. A concentrated white with a long, lingering aftertaste. Delicious. (2400 cases produced) Highly Recommended

Rule results for Recommended Chardonnays

Rule results for Best Buys

Rule results for Highly Recommen

Standard results for the query Wine Type > White

Suggested workflow using Endeca tools to promote records

Place supporting constructs for rules before creating the Web application. The pipeline developer should then give business users access to the project through Endeca Workbench to create, edit, and test rules.

You can build dynamic business rules and their constructs in Developer Studio. In addition, business users can use Endeca Workbench to perform any of the following rule-related tasks:

- Create a new dynamic business rule.
- Modify an existing rule.
- Deploy a rule to a preview application and test or preview its results.

Because either tool can modify tools, the tasks involved in promoting records require coordination between the pipeline developer and the business user. The recommended workflow is as follows:

1. A pipeline developer uses Developer Studio in a development environment to create the supporting constructs (zones, styles, rule groups, and so on) for rule and perhaps small number of dynamic business rules as placeholders or test rules.
2. An application developer creates the Web application including rendering code for each style.

For general information about using Endeca tools and sharing projects, see the *Oracle Endeca Workbench Administrator's Guide.* Endeca Workbench tasks are described in the *Oracle Endeca Workbench Help.*

Note: Any changes to the constructs that support rules such as changes to zones, styles, rule groups, and property templates have to be performed in Endeca Developer Studio.

Incremental Implementation

Merchandising and content spotlighting are complex features to implement, and the best approach for developing your dynamic business rules is to adopt an incremental approach as you and business users of Endeca Workbench coordinate tasks. It is also helpful to define the purpose of each dynamic business rule in the abstract (before implementing it in Developer Studio or Endeca Workbench) so that everyone knows what to expect when the rule is implemented. If rules are only loosely defined when implemented, they may have unexpected side effects.

Begin with a single, simple business rule to become familiar with the core functionality. Later, you can add more advanced elements, along with additional rules, rule groups, zones, and styles. As you build the complexity of how you promote records, you will have to coordinate the tasks you do in Developer Studio (for example, zone and style definitions) with the work that is done in Endeca Workbench.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

Building Rules and Their Supporting Constructs

This section describes how to build rules and their supporting constructs.

Building the Supporting Constructs

Building the supporting constructs for rules

Zone and style settings are used together to restrict rule results promoted to users.

As discussed in Dynamic business rule constructs, the records identified by a rule's target are candidates for promotion and may or may not all be promoted in a Web application. It is a combination of zone and style settings that work together to effectively restrict which rule results are actually promoted to application users.

- A zone identifies a collection of rules to ensure at least one rule always produces records to promote. For information on zones, see Ensuring promoted records are always produced.
- A style controls the minimum and maximum number of results to display, defines any property templates, and indicates how to display the rule results to the Web application. For more information on styles, see Creating a style.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

Ensuring promoted records are always produced

You ensure promoted records are always produced by creating a zone in Developer Studio to associate with a number of dynamic business rules.

A zone is a logical collection of rules that allows you to have multiple rules available, in case a single rule does not produce a result. The rules in a zone ensure that the screen space dedicated to displaying promoted records is always populated.

A zone has a rule limit that dictates how many rules may successfully return rule results. For example, if three rules are assigned to a certain zone but the rule limit is set to one, only the first rule to successfully provide rule results is evaluated. Any remaining rules in the zone are ignored.

1. In the Project Explorer, expand **Dynamic Business Rules** .
2. Double-click **Zones** to open the Zones view.
3. Click **New** to open the Zone editor.
4. In the **Name** field, provide a unique name for the zone.
5. (Optional) If you want to limit the number of rules that can provide rule results within a zone, type a number in the **Rule Limit** text box.
6. (Optional) If you want to randomly order the rules in the zone, select the **Shuffle rules** check box. When checked, this indicates that the MDEX Engine randomly shuffles the order of the rules within this zone before evaluating them.
7. (Optional) Select **Valid for Search** to indicate whether a zone (and all of the rules associated with that zone) is valid for navigation queries that include a record search parameter.

   Rules that include a keyword trigger require the Valid for Search setting.
8. (Optional) If you want to prevent the same record from appearing multiple times for multiple rules, check **Unique by This Dimension/Property** and specify a unique record criterion.

   Selecting a dimension or property allows the MDEX Engine to identify individual records and prevent the same record from appearing multiple times. If you check Unique by This Dimension/Property and do not select a dimension or property, the same record may appear multiple times for multiple rules. Note that this setting applies across all zones.
9. Click **OK**.

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

The Zones View

Fields in the **Zones** view include name of the zone, rule limit, shuffle, valid for search, and unique record spec.

Option	Description
Name	The name of a zone.
Rule Limit	(Optional) When set, this entry must be an integer that limits the number of rules that can be returned for this specific zone. For example, if three rules are assigned to a certain zone but the zone threshold is set to one, only the first rule that is successfully evaluated is returned. (This does not mean the first rule is always

Option	Description
	returned—rather, it is the first rule that successfully returns results.) Any remaining rules are skipped. If no threshold is set, all matching rules will be returned within a zone. If no rules are valid, no rules will be returned for a zone, regardless of what the threshold is set at.
Shuffle	Set to Yes or No, this entry indicates whether rules within a zone are shuffled before evaluating them. In the above example, if shuffle was set to true, the "Best Buy" and "Recommended" rules would be randomly returned (assuming both were successfully evaluated and the zone threshold was set to one).
Valid for search	Set to Yes or No, this entry indicates whether a zone (and all of the rules associated with that zone) is valid for navigation queries that include a record search parameter.
Unique record spec	(Optional) Specifies a unique record criteria across zones. When selected, this ensures that every record returned within any zone (including all rules in a zone) is classified with a unique dimension or property value for that dimension or property. If no dimension or property is specified, the same record may appear multiple times for multiple rules within a zone.

Zone editor

Configure options such as rule limit and setting zone as valid for search.

The **Zone** editor contains the following fields:

Option	Description
Name	A unique name for this zone.
Rule limit	(Optional) A limit to the number of rules that can be returned.
Shuffle rules	When checked, indicates that rules within this zone are shuffled before being evaluated.
Valid for search	When checked, indicates that a zone (and all of the rules associated with that zone) is valid for navigation queries that include a record search parameter.
Unique by this dimension/property	(Optional) Specifies a unique record criteria across all zones. When checked, this ensures that every record

Option	Description
	returned across any zone (including all rules in that zone) is classified with a unique dimension or property value for that dimension or property.

Working with Styles

About styles

A style controls the number of records a rule may promote; it also defines property templates, and tells the Web application how to display rule results.

You create a style in the Styles view of Endeca Developer Studio. A style serves three functions:

- It controls the minimum and maximum number of records that may be promoted by a rule.
- It defines property templates, which facilitate consistent property usage between pipeline developers and business users of Endeca Workbench.
- It indicates to a Web application which rendering code should be used to display a rule's results.

Controlling the number of promoted records

Styles can be used to affect the number of promoted records in two scenarios:

- A rule produces less than the minimum number of records. For example, if the Best Buys rule produces only two records to promote and that rule is assigned a style that has Minimum Records set to three, the rule does not return any results.
- A rule produces more than the maximum. For example, if the Best Buys rule produces 20 records, and the Maximum Records value for that rule's style is five, only the first five records are returned.

If a rule produces a set of records that fall between the minimum and maximum settings, the style has no affect on the rule's results.

Performance and the Maximum Records setting

The Maximum Records setting for a style prevents dynamic business rules from returning a large set of matching records, potentially overloading the network, memory, and page size limits for a query. For example, if Maximum Records is set to 1000, then 1000 records could potentially be returned with each query, causing significant performance degradation.

Ensuring consistent property usage with property templates

As discussed in Dynamic business rule constructs, rule properties are key/value pairs typically used to return supplementary information with promoted record pages. For example, a property key might be set to SpecialOffer and its value set to BannerAd.gif.

As Endeca Workbench users and Developer Studio users share a project with rule properties, it is easy for a key to be mis-typed. If this happens, then the supplementary information represented by a property does not get promoted correctly in a Web application. To address this, you can optionally create property templates for a style. Property templates ensure that property keys are used consistently when pipeline developers and Endeca Workbench users share project development tasks.

If you add a property template to a style in Developer Studio, that template is visible in Endeca Workbench in the form of a pre-defined property key with an empty value. Endeca Workbench users are allowed to add a value for the key when editing any rule that uses the template's associated style. Endeca Workbench users are not allowed to edit the key itself.

Furthermore, pipeline developers can restrict Endeca Workbench users to creating new properties based only on property templates, thereby minimizing potential mistakes or conflicts with property keys.

For example, a pipeline developer can add a property template called WeeklyBannerAd and then make the project available to Endeca Workbench users. Once the project is loaded in Endeca Workbench, a property template is available with a populated key called WeeklyBannerAd and an empty value. The Endeca Workbench user provides the property value. In this way, property templates reduce simple project-sharing mistakes such as creating a similar, but not identical property called weeklybannerad.

Note: Property templates are associated with styles in Developer Studio, not rules. Therefore, they are not available for use on the Properties tab of the Rule editor.

Indicating how to display promoted records

by creating application-level rendering code for the style. You create a style in Developer Studio. You create rendering code in your Web application. Creating a style describes how to create styles. Information about rendering code is described in Presenting rule results in a Web application.

A style has a name and an optional title. Either the name or title can be displayed in the Web application. When the MDEX Engine returns rule results to your application, the engine also passes the name and title values to your application. The name uniquely identifies the style. The title does not need to be unique, so it is often more flexible to display the title if you use the same title for many dimension value targets, for example, the title On Sale may be commonly used.

Note: Without application-level rendering code that uses the specific style or title values, the style and title are meaningless. Both require application-level rendering code in an application.

Creating a style
Create a style from the Style editor in the Styles view. Here you can set minimum and maximum records returned for a rule, add or remove property templates, and allow Endeca Workbench users to set new rule properties.

To create a style in Developer Studio:

1. In the Project Explorer, double-click **Styles** to open the Styles view.
2. Click **New** to open the Style editor.
3. In the Name field, provide a unique name for the style.
4. If desired, specify a title in the Title field. The title does not need to be unique.
5. In the Minimum Records field, specify the minimum number of records that must be returned by a rule's target in order for that rule to return results. (The default Minimum Records value, if not specified, is zero.)
6. In the Maximum Records field, specify the maximum number of records that can be returned for a rule. (The default Maximum Records value, if not specified, is ten. If Maximum Records is set to zero, the rule returns zero records.)
7. If you want to create a property template, click **Add** in the Property Templates frame.
   The Property Template dialog box displays.
8. Provide the key for the property template.
9. Click **OK.**
10. Repeat for as many new property templates as necessary.
11. If you need to remove a property template, select it in the Property Templates frame and click **Remove.**
12. If you want to allow Endeca Workbench users the ability to create new properties for a rule, check "Allow additional, custom properties." Unchecking this option prevents Endeca Workbench users from creating new properties to associate with a rule. In other words, Endeca Workbench users will be restricted to creating properties based only on the property templates you have created in Developer Studio.
13. Click **OK.**

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

Style editor
Describes the options in the Style editor.

The Style editor contains the following fields:

Option	Description
Name	A unique name for a style.
Title	(Optional) A title that may be used for display in your Endeca application.
Records minimum	The minimum number of records that must be returned by a rule's target request in order for that rule to return business rule results.
Records maximum	The maximum number of records that will be returned for a rule.
Property templates	Indicates the property templates available to rules associated with the style. You can add and remove property templates. Property templates ensure that property keys are used consistently when pipeline developers and Endeca Workbench users share project development tasks. If you add a property template to a style, that template becomes available in Endeca Workbench in the form of a pre-defined property key with an empty value. Endeca Workbench users can add a value for the key when editing any rule that uses the template's associated style.
Add	Displays the New Property Template dialog box in which you enter the key value of a template's key/value pair.
Remove	Deletes a selected property template.
Allow additional, custom properties	To allow Endeca Workbench users the ability to create new properties for a rule, check "Allow additional, custom properties." Unchecking this option prevents Endeca Workbench users from creating new properties to associate with a rule. In other words, Endeca Workbench users will be restricted to creating properties based only on the property templates you have created in Developer Studio .

Creating Rules

About creating dynamic business rules

You must have at least one zone and one style before you can create rules.

After you have created zones and styles, you can start creating the dynamic business rules themselves. It is not necessary to create additional rule groups unless your application requires it. If you want to create additional rule groups before you create a rule, see About grouping rules.

As mentioned in Suggested workflow using Endeca tools to promote records, a developer usually creates the preliminary rules and the other constructs in Developer Studio, and then hands off the project to a business user to fine tune the rules and create additional rules in Endeca Workbench. However, the business user can use Endeca Workbench to perform any of the tasks described in the following sections that are related to creating a rule. For details, see Endeca Workbench Help.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

Add Trigger editor

The Add Trigger editor allows you to build a one or more triggers for a dynamic business rule.

The Add Trigger editor contains the following fields:

Option	Description
Dimension values	A list of dimension values that may be added to a trigger. If checked, a query that matches the trigger can fire the rule at any location (navigation state) in the data set. Any dimension value below a trigger dimension value (i.e. a child dimension value) can also trigger the rule. If unchecked, a query can fire the rule only if the trigger matches the location in the data set.
Any location containing this criteria	A word or phrase that must be present in the end-user's search query in order for the rule to fire. **Note:** A phrase represents terms surrounded in quotes.
Match mode	Indicates how a keyword trigger must be matched from a user's record search query in order to fire the rule. • In Phrase mode (the default), all of the words of the keyword trigger must match in the same order in the user's query for the rule to fire. • In All mode, all of the words of the keyword trigger must match (without regard for order in the user's query) for the rule to fire. • In Exact mode, all the words of the keyword trigger must exactly match a user's query for the rule to fire. Unlike the other two modes, a user's query must exactly match the keyword trigger in the number of words and cannot be a super set of the keyword triggers. If no trigger keywords are specified, the rule does not need any specific keywords to qualify the rule for

Option	Description
	evaluation, but is still limited by other parameters of the rule. **Note:** All modes allow the rule to fire if the spelling auto-correction and auto-phrasing, and/or stemming corrections of a user's query match the keywords or the phrase (terms surrounded in quotes).
Enable stemming	If checked, the rule fires if stemming corrections of a user's query match the keyword trigger. If you select this option, make sure stemming is enabled in your project. For more information, see Enabling stemming.
Enable spelling	If checked, the rule fires if spelling corrections of a user's query match the keyword trigger. If you select this option, make sure spelling is enabled in your project. For more information, see Automatic phrasing, spelling correction, and DYM.

Creating a rule and ordering its results

Associate a zone and style with a rule and define result sort order on the Rule editor of the Rules view, found under Dynamic Business Rules in the Project Explorer.

To create a dynamic business rule in Developer Studio:

1. In the Project Explorer, expand `Dynamic Business Rules.`
2. Make sure you have already created at least one zone and one style.
3. Double-click `Rules,` which opens the Rules view and also activates the Rule menu on the menu bar.
4. From the Rule menu, select `New.`
   The Rule editor displays.
5. In the Name text box, enter a unique name for the new rule.
6. From the Zone list, choose a zone to associate with the rule.
7. From the Style list, choose a style to associate with the rule.
8. You can add the rule to the default rule group, or if you created additional rule groups, select a rule group to which this rule belongs. For more information, see About grouping rules.
9. If you want to sort the rule's promoted records by a property or dimension value, select the property or dimension value from the Sort key list. Select `[None]` to accept the default sort order specified for the project.
10. If you chose a Sort key, choose `Ascending` or `Descending` to define sort order.
11. If you want to randomly order the promoted records for this rule, select Shuffle.
    Selecting `Shuffle` overrides any Sort key and Order options you specified above.
12. If you want to allow a rule to fire even if the current navigation state overlaps a rule's trigger, select `Self-pivot.` Unchecking `Self-pivot` prevents a rule from firing if any dimension values in the current navigation state overlap with dimension values in the rule's trigger.

13. If you want to restrict who sees the results of this rule, select a pre-defined user profile from the drop-down list.
14. Specify when to promote records.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

Specifying when to promote records

You indicate when to promote records by specifying one or more triggers on the Triggers tab of the Rule editor.

A trigger may have any combination of dimension values and keywords. If a user's query matches a trigger, the MDEX Engine fires the rule and promotes records. In other words, if a user's query contains the dimension value or values you specify in a trigger, the MDEX Engine fires that rule, and if a user's query contains a keyword or words that match a trigger, then the MDEX Engine fires the associated rule.

Keywords in a trigger require that the zone associated with the rule have "Valid for search" enabled on the Zone editor in Developer Studio. Keywords in a trigger also require a match mode that specifies how the keyword should match in order to trigger the rule. There are three match modes:: Phrase, All, and Exact.

Triggers can also be empty (no specified dimension values or keywords) on the Triggers tab. In this case, there are two options to determine when an empty trigger fires a rule:

- Applies everywhere-Any navigation query and any keyword search in the application triggers the rule.
- Applies only at root-Any navigation query and any keyword search from the root dimension value only (N=0) triggers the rule.

Multiple triggers in a rule

A dynamic business rule can have one or more triggers that include both dimension values and keywords. Adding more than one trigger to a rule is very useful if you want to promote the same records from multiple locations in your application. Each trigger can describe a different location where a user's query can trigger a rule; however, the rule promotes records from a single target location.

Specifying a time trigger to promote records

A time trigger is useful if you want to promote records for a particular period of time.

You specify a time trigger on the Time Trigger tab of the Rule editor. A time specified on this tab is a date/time value that indicates the time at which the rule may fire and the time after which the rule may not fire. Any matching query that occurs between these two values fires the rule.

For example, you might create a rule called This Weekend Only Sale whose time trigger starts Friday at midnight and expires on Sunday at 6 p.m. Only a start time is required for a time trigger. If you do not specify an expiration time, the rule can be triggered indefinitely.

To specify a time trigger:

1. In the Rule editor for the rule you want to configure, click the **Time Trigger** tab.
2. Select **Give This Rule a Time Trigger** to enable the Start time options.
3. From the Start Time drop-down list, select the start time for the time trigger.
4. (Optional) Select **Give This Rule an Expiration Date** and from the Expiration Time drop-down list, choose the end time for the time trigger.

   Note: If you do not specify an expiration time, the trigger does not expire.

5. Identify which records to promote by specifying the rule's target.

Synchronizing time zone settings

The start time and expiration time values do not specify time zones. The server clock that runs your Web application identifies the time zone for the start and expiration times. If your application is distributed on multiple severs, you must synchronize the server clocks to ensure the time triggers are coordinated.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the *Oracle Endeca Workbench Help* for details.

Specifying which records to promote

You indicate which records to promote by specifying a target on the Target tab of the Rule editor.

A target is a collection of one or more dimension values. These dimension values identify a set of records that are all candidates for promotion. Zone and style settings further control the specific records that are promoted to a user.

To specify which records to promote:

1. In the Rule editor for the rule you want to configure, click the **Targets** tab.
2. Click **Add.**
   The Select Dimension Value dialog box displays.
3. Select a dimension value from the list and click **OK.**

   Note: You cannot add auto-generated dimension values until you first load and promote the dimension values.

4. If necessary, repeat the above steps to add multiple dimension values to the target.
5. Check Augment Navigation State to add target dimension values to the existing navigation state when evaluating the rule. If not checked, the MDEX Engine ignores all current navigation state filters when evaluating the rule. Navigation state filters include dimension value selections, record search, range filters, and so on. The one exception is custom catalog filters, which always apply regardless of this setting.
   For example, if checked, and a user navigates to Wine Type > Red and that triggers a rule that promotes wines under $10, then the rule results will include only red wines that are under $10. The rule results are always a subset of the standard query results. Conversely, if Augment Navigation State is not checked, and a user navigates to Wine Type > Red which triggers a rule that promotes wines under $10, then the rules results will include any wine type (red, white, sparkling) with wines under $10. The rule results are not a subset of the standard query results.
6. At this point, you can either:
   - Promote custom properties or featured records, or
   - Click **OK** if you are finished configuring the rule.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the Endeca Workbench Help for details.

Adding custom properties to a rule

You can promote custom properties by creating key/value pairs on the Properties tab of the Rule editor.

You can promote custom properties by creating key/value pairs on the Properties tab of the Rule editor. Rule properties are typically used to return supplementary information with promoted record pages. Properties could specify editorial copy, point to rule-specific images, and so on. For example, a property name might be set to SpecialOffer and its value set to BannerAd.gif.

You can add multiple properties to a dynamic business rule. These properties are accessed with the same method calls used to access system-defined properties that are included in a rule's results, such as a rule's zone and style.

To add a custom rule property:

1. In the Rule editor for the rule you want to change, click the **Properties** tab.
2. Type the property name in the **Property** field and its corresponding value in the **Value** field.
3. Click **Add**.
4. Repeat steps 2 and 3 if you want to add additional properties.
5. Click **OK**.

   You can also create templates to facilitate the creation of rule properties in Endeca Workbench. See About styles for details.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the Endeca Workbench Help for details.

Adding static records in rule results

In addition to defining a rule's dimension value targets and custom properties, you can optionally specify any number of static records to promote. These static records are called featured records, and you specify them on the Featured Records tab of the Rule editor.

You access featured records in your Web application using the same methods you use to access dynamically generated records.

To add featured records to a rule:

1. In the Rule editor for the rule you want to change, click the **Featured Records** tab.
2. From the Record spec list, choose an Endeca property to uniquely identify featured records.

   Note: Oracle strongly recommends that you specify the record specifier properties for the featured records in Developer Studio, and do not edit the XML configuration files directly.

3. In the **Value** text box, type a value for the selected Endeca property.

   This value identifies a featured record you want to promote.
4. Click **Add**.
5. (Optional) Repeat steps 3 and 4 to add additional featured records to the list.
6. To change the order in which the featured records appear:
   a) Select a record.
   b) Click **Up** or **Down**.
7. To change a record spec value:
   a) Select it from the Record spec values list.
   b) Modify its value in the **Value** text box.
   c) Click **Update**.
8. Click **OK** when you are done adding featured records to a rule.

   The MDEX Engine treats featured records differently than dynamically generated records. In particular, featured records are not subject to any of the following:

   - Record order sorting by sort key
   - Uniqueness constraints

- Maximum record limits

Order of featured records

The General tab of the Rule editor allows you to specify a sort order for dynamically generated records that the MDEX Engine returns. This sort order does not apply to featured records. Featured records are returned in a Supplement object in the same order that you specified them on the Featured Records tab. The featured records occur at the beginning of the record list for the rule's results and are followed by any dynamically generated records. The dynamically generated records are sorted according to your specified sort options.

No uniqueness constraints

The zone associated with a rule allows you to indicate whether rule results are unique by a specified property or dimension value. This uniqueness constraint does not apply to featured records even if uniqueness is enabled for dynamically generated rule results. For example, if you enabled Color to be the unique property for record results and you have two dynamically generated records with Blue as property value, then the MDEX Engine excludes the second record as a duplicate. On the other hand, if you have the same scenario but the two records are featured results not dynamically generated results, the MDEX Engine returns both records .

Note: Developer Studio performs a case-insensitive search for duplicate keys.

No maximum record limits

The style associated with a rule allows you to set a maximum number of records that the MDEX Engine may return as rule results. This Maximum Records value does not apply to featured records. For example, if the Maximum Records value is set to three and you specify five featured records, the MDEX Engine returns all five records. Also, the MDEX Engine returns featured records before dynamically generated records, and the featured records count toward the maximum limit. Consequently, the number of featured records could restrict the number of dynamically generated rule results.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the Endeca Workbench Help for details.

Sorting the rules in the Rules view

To make rules easier to find and work with, they can be sorted by name (in alphabetical ascending or descending order) or by priority.

The dynamic business rules you create in Developer Studio appear in the Rules view.

The procedure described below changes the way rules are sorted in Rules view only. Sorting does not affect the priority used when processing the rules. See Prioritizing rule evaluation order for information on controlling the order in which rules are evaluated.

To sort the rules in the Rules view:

- In the Rules view, click the Name column header to cycle the sort order from Sorted by name (ascending), Sorted by name (decending), and Sort by priority order.

Prioritizing rule evaluation order

You modify the relative priority of a rule by moving it up or down in the Rules view.

In addition to sorting rules by name or priority, you can also modify a rule's priority in the Rules view of Developer Studio. Priority is indicated by a rule's position in the Rules view, relative to the position of other rules when you have sorted the rules by priority.

A rule's priority affects the order in which the MDEX Engine evaluates the rule. The MDEX Engine evaluates rules that are higher in the Rules view before those that are positioned lower. By increasing the priority of a rule, you increase the likelihood that the rule is triggered before another, and in turn, increase the likelihood that the rule promotes records before others.

It is important to consider rule priority in conjunction with the settings you specify in the Zone editor. For example, suppose a zone has Rule Limit set to three. If you have ten rules available for the zone, the MDEX Engine evaluates the rules, in the order they appear in the Rules view, and returns results from only the first three that have valid results. In addition, the Shuffle Rules check box on the Zone editor overrides the priority order you specify in the Rules view. When you check Shuffle Rules, the MDEX Engine randomly evaluates the rules associated with a zone.

If you set up rule groups, you can modify the priority of a rule within a group and modify the priority of a group with respect to other groups. For details, see Prioritizing rule groups.

To prioritize rules:

1. In the Rules view, click the Name column header to cycle the order of the rule sort so that the rules are displayed in priority order (you'll see Sorted by Priority in the lower left corner of the Rules view).
2. Select the rule whose priority you want to change and click the Up or Down buttons to move the rule to the desired position.

Rule editor
The Rule editor lets you configure various rule parameters including zones, styles, and rule groups.

The Rule editor contains a unique name for this rule. In addition, it contains the following tabs:

General

The General tab contains the following fields:

Option	Description
Zone	A list of all available zones.
Style	A list of all available styles.
Member of this rule group	If this project uses rule groups, a list of all available rule groups. (If this project does not contain rule groups, this field is disabled.
Sort key	Optional. A list of sortable dimensions and Endeca properties. [None] indicates that the rule has no record order.
Shuffle	If checked, shuffles the promoted records for the rule.
Self-Pivot	The Self-Pivot feature prevents business rules from displaying duplicate records in merchandising and search results lists when the trigger and target of the business rule contain the same dimension values. With Self-Pivot unchecked, the MDEX Engine looks at each dimension value in the current navigation state. If any of those dimension values exists in the rule's target, the rule will not fire.

Option	Description
	When checked, Self-Pivot allows business rules to fire even if the user navigates to a location which explicitly contains a dimension value already in the rule target. In version 5.1.4 and later, self-pivot is enabled by default for each new rule created in Endeca Workbench, and the option is not displayed in Endeca Workbench. However, you can change the default and set the check box to display on the Triggers tab in Endeca Workbench. Please refer to the *Endeca Advanced Development Guide* for more information.
Order	If a sort key is selected, specifies whether the sort order will be ascending or descending.
User profile	Optional. A user profile that must be present in the query in order for the rule to fire. There can only be one user profile per rule.

Triggers

The Triggers tab contains the following field:

Option	Description
Rule is triggered by any of these conditions	A list of triggers that can be made up of any combination of keywords and dimension values.

Time Triggers

The Time Triggers tab contains the following fields:

Option	Description
Give this rule a time trigger	Enables a time trigger for the rule. A trigger specified on this tab is a date/time value that indicates the time at which to start the rule's trigger and the time at which the trigger ends. Any matching query that occurs between these two values triggers the rule.
Start time	Specifies the time after when the rule can be triggered by user queries. Only a start time value is required for a time trigger.
Give this rule an expiration date	Enables the Expiration time field. If you do not specify an expiration time, the rule can be triggered indefinitely.

Option	Description
Expiration time	Specifies the time after which the rule can no longer be triggered.

Targets

The Targets tab contains the following fields:

Option	Description
Targets	A list of target dimension values.
Augment navigation state	If checked, target dimension values are added to the existing navigation state when evaluating the rule. If not checked, evaluation of that rule ignores all current navigation state filters (such as dimension value selections, record search, range filters, and so on) except custom catalog filters.

Properties

The Properties tab contains the following field:

Option	Description
Property/value table	Specifies any rule properties that you want to return as supplementary information with the promoted record pages. You can add multiple properties to a rule. Such properties can be used to specify editorial copy, point to rule-specific images, and so on. These properties can then be accessed with the same method calls used to access system-defined properties that appear in your Endeca application. To add a property: 1. Type the key name in the **Key** field and its corresponding value in the Value field. 2. Click Add.

Featured Records

The Featured Records tab contains the following fields:

Option	Description
Record spec	A list of dimensions and Endeca properties that can be used for the record specifier. [None] indicates no dimension or Endeca property has been selected.

Option	Description
Record spec values	A list of the record identifiers that you type into the Value text box.

Presenting rules results in a Web application

The MDEX Engine returns rule results to a Web application in a Supplement object.

To display rule results to Web application users, an application developer writes code that extracts the rule results from the Supplement object and displays the results in the application. See the *Endeca Advanced Development Guide* for details.

Grouping Rules

About grouping rules

Rule groups complement zones and styles in supporting dynamic business rules.

Rule groups serve two functions:

- They provide a means to logically organize rules into categories to facilitate creating and editing rules.
- They allow multiple business users to access Endeca Workbench simultaneously

A rule group provides a means to organize a large number of rules into smaller logical categories, which usually affect distinct (non-overlapping) parts of a Web site. For example, a retail application might organize rules that affect the electronics and jewelry portions of a Web site into a group for Electronics Rules and another group for Jewelry Rules.

A rule group also enables multiple business users to access Endeca Workbench simultaneously. Each Endeca Workbench user can access a single rule group at a time. Once a user selects a rule group, Endeca Workbench prevents other users from editing that group until the user returns to the group selection list or closes the browser window.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the Endeca Workbench Help for details.

Creating a rule group

A rule's order within the group affects its evaluation order. The MDEX Engine evaluates rules at the top over a group before rules that are placed lower in the group.

To create a rule group:

1. In the Project Explorer window, expand `Dynamic Business Rules.`
2. Double-click `Rules.`
   This opens the Rules view and also activates the Rule Group menu on the menu bar.
3. From the Rule Group menu, select `New.`
4. Type a unique name in the Group name field. Use only alphanumeric, dash, or underscore characters. The name must be unique across both keyword redirect groups and rule groups.
5. To select a rule for this group, highlight a rule in the All Rules list and click `Add.`
   The rule appears in the Rules in Group list. (If this is the first group you created, all the rules are moved to the Rules in Group list and the Remove button is inactive.)

   Note: A rule can belong to only one rule group. Adding a rule to a group removes it from any group to which it previously belonged.

6. To change the order of a rule in a group, select the rule and click either the Up or Down arrow buttons.
7. Click `OK`.
   The new rule group appears in the Rules view.
8. Repeat steps 3 to 7 if you want multiple rule groups in your project.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the Endeca Workbench Help for details.

Deleting a rule group

Delete rule groups from the Rules view.

To delete a rule group:

1. In the Rules view's `Rule Groups Name` column, select a rule group and then click `Delete`.
2. When the confirmation message appears, click `Yes`.
3. If your project contains more than one other rule group, the `Select Rule Group` dialog box appears. In the drop-down list, select the rule group you want to move the rules in this group to, and click `OK`.

   If the rule group you are deleting is the second to last rule group, the rules are moved to the one remaining rule group.

Prioritizing rule groups

In the Rules view, move rule groups up or down to change their priority in relation to other rule groups.

In the same way that you can modify the priority of a rule within a group, you can also modify the priority of a rule group with respect to other rule groups.

The MDEX Engine evaluates rules first by group order, as shown in the Rules view of Developer Studio or Endeca Workbench, and then by their order within a given group. For example, if Group_B is ordered before Group_A, the rules in Group_B will be evaluated first, followed by the rules in Group_A. Rule evaluation proceeds in this way until a zone's Rule Limit value is satisfied.

This relationship is shown in the example below. In it, suppose zone 1 has a Rule Limit setting of 2. Because of the order of group B before group A, rules 1 and 2 satisfy the Rule Limit rather than rules 4 and 5.

Group B

- Rule1, Zone 1
- Rule 2, Zone 1
- Rule 3, Zone 2

Group A

- Rule 4, Zone 1
- Rule 5, Zone 1
- Rule 6, Zone 2

To prioritize rule groups:

1. In the Rules view, select a group whose priority you want to change in the Rule Groups: Name column.
2. Click the **Up** or **Down** buttons to move the group to the desired position.

Note: If you want to further prioritize the rules within a particular rule group, see, Prioritizing rule evaluation order.

Rule Group editor

The Rule Group editor allows you to create a new rule group or modify an existing one.

The Rule Group editor contains the following fields:

Option	Description
Group name	A unique name for this rule group.
All rules list	Displays all of the rules that have been defined in your project.
Rules in group list	Displays the rules that are included in this rule group.
Add	Adds the selected rule from the "All rules" list to the rule group.
Remove	Removes the selected rule from the "Rules in group" list from the rule group.
Up arrow	Moves the selected rule from the "Rules in group" list up.
Down arrow	Moves the selected rule from the "Rules in group" list down.

Notes:

- A rule can only belong to one rule group. If a rule that you select currently belongs to another rule group, adding it to this group will in effect remove it from the other group.
- New rule groups created with Developer Studio get the same default value that you specify in Endeca Workbench. (See the Rule Group Permission setting on the User Management page of Endeca Workbench.) If you happen to be using Developer Studio in stand-alone mode, Developer Studio does not create permissions for rule groups. Also, renaming a rule group preserves its associated permissions. In other words, if Rule Group A has Approve permissions and you rename it to Rule Group B, then Rule Group B has Approve permissions.

Rule Group name editor

After a rule group is created, you are only allowed to edit its name in the group's editor.

After a rule group is created, you are only allowed to edit its name in the group's editor.

To edit an existing group's name:

1. Enter the name in the "Group name" field.
2. Click OK.

To move a rule from one group to another:

1. Drag the rule from the "Rules in group" pane.
2. Drop it on the rule in the "Rule groups" pane.

Interaction between dynamic business rules and rule groups

A rule may only belong to one group but a group does not necessitate rules. In addition, you may change the order of rule groups themselves.

When creating or editing rule groups, keep in mind the following interactions between rules and rule groups:

- Rules may be moved from one rule group to another. However a rule can appear in only one group.
- A rule group may be empty (that is, it does not have to contain rules).
- The order of rule groups with respect to other rule groups may be changed.

Using dynamic business rules with an Agraph

When updating dynamic business rule configurations on Dgraphs, Oracle recommends shutting down the Agraph.

To implement dynamic business rules when you are using the Agraph, keep in mind the following points:

- Using dynamic business rules with the Agraph affects performance if you are using zones configured with Unique by This Dimension/Property combined with a high setting for the maximum number of records or a large numbers of rules. To avoid response time problems, you may need to reduce the number of rules, reduce the maximum records that can be returned, or abandon uniqueness.
- All Dgraphs serving one Agraph must share the same set of dynamic business rules. To ensure this, it is necessary to update their configurations synchronously by running Dgidx with the `--keepcats` flag.
- If you update your Dgraphs with dynamic business rule changes using Developer Studio or Endeca Workbench, and a request comes to the Agraph while the update is in progress, the Agraph issues a fatal error similar to the following:

  ```
  [Fri Mar 24 16:26:29 2006] [Fatal] (merchbinsorter.cpp::276) - Dgraph 1 has
  fewer rules fired.
  ```

As long as the Agraph is running under the Endeca JCD, the JCD automatically restarts the Agraph. No data is lost. However, end-users will not receive a response to requests made during this short time. This problem has little overall impact on the system, because business rule updates are quick and infrequent. Nevertheless, Oracle recommends that you shut down the Agraph during business rule updates. To shut down the Agraph, go to a CMD prompt on Windows or a shell prompt on UNIX and type:

```
GET 'http://HOST:PORT/admin?op=exit'
```

where *HOST* is machine running the Agraph and *PORT* is the port number of the Agraph. *GET* is a Perl utility, so be sure the Perl binaries are in your system path variable.

Note: Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Advanced Development Guide* as well as the Endeca Workbench Help for details.

Applying relevance ranking to rule results

You can order results in a rule by their relevance ranking score for a given query term.

In some cases, it is a good idea to apply relevance ranking to a rule's results.

For example, if a user performs a record search for Mondavi, the results in the Highly Rated rule can be ordered according to their relevance ranking score for the term Mondavi. In order to create this effect, there are three requirements:

- The navigation query that is triggering the rule must contain record search parameters (Ntt and Ntk). Likewise, the zone that the rule is assigned to must be identified as Valid for search (Otherwise, the rule will not be triggered).
- The rule's target must be marked to Augment Navigation State.
- The rule must not have any sort parameters specified. If the rule has an explicit sort parameter, that parameter overrides relevance ranking. Sort parameters for a rule are set on the General tab of the Rule editor.

If these three requirements are met, then the relevance ranking rules specified with MDEX Engine startup options are used to rank specific business rules when triggered with a record search request (a keyword trigger).

Configuring Dimension Groups

This section describes how to use the Project Explorer to organize dimensions into groups for presentation.

About dimension groups

Dimension groups allow you to organize dimensions into groupings for presentation purposes.

Viewing all of your dimensions

All dimensions are visible in the Dimensions view, opened from the Project Explorer.

To open the Dimensions view:

- In the Project Explorer Project tab, double-click **Dimensions**.
  The Dimensions view appears in the work area.
- Choose from the following options:

Option	Description
Name	A unique name for the dimension. Clicking the Name column head cycles the order of the dimension sort from rank order, to alphabetical ascending, to alphabetically descending. **Note:** In order to rank your dimensions, you must be displaying them in rank order in the Dimensions view.
External type	If the dimension was not manually created in Developer Studio, indicates what type of external dimension it is: • Auto Gen: The dimension's values are being automatically generated because the match mode in the Property Mapper is set to Auto Generate. • External: Indicates that this dimension has been created and is being maintained in a third-party tool, outside of Developer Studio. Externally managed dimensions are read-only. • Prop Mapped: Indicates that this dimension is being created using the "If no mapping is found, map source properties to Endeca: Dimensions" on the Property Mapper's Advanced tab.
Enable record sort	When Yes, indicates that records can be sorted by this dimension.
Show with record list	Enables the dimension values from this dimension to appear in the record list display.
Show with record	Enables the dimension values from this dimension to appear on the record pages of any records that are tagged with these dimension values.

Option	Description
Dynamic ranking	Establishes dynamic dimension value ranking, which orders dimension values according to their frequency of appearance within the current record set (that is, the most popular dimension values are returned).
Multi select	Allows the end user to select more than one dimension value from a dimension within a single query.
Refinements sort order	Defines the default sort order used for dimension values within this dimension. This sort order is used when no other types of dimension value sorting have been specified.

Creating dimension groups

Create dimension groups in the New Dimension Group editor, found in the Dimension Groups view.

To create a new dimension group:

1. In the Project Explorer's Project tab, double-click **Dimension Groups** to open the Dimension Groups view.
2. Click **New.**
   The New Dimension Group editor appears.
3. In the Group Name box, type a unique name for the dimension group.
4. Select the dimension(s) you want to add in the All Dimensions list and click **Add.** You can add dimensions one by one or hold down the Ctrl key and add several at a time.

   Note: There is no limit to the number of dimensions in a group, but each dimension can only belong to one group.

5. Click **OK.**
   The new dimension group is added to the list of available dimension groups.

Modifying dimension groups

Make changes to dimension groups in the Dimension Group editor, found in the Dimension Groups view.

To edit a dimension group:

1. In the Dimension Groups view, double-click the dimension group you want to modify to open it in the Dimension Group editor. (You can only open one dimension group at a time.
2. Make the necessary changes in the Dimension Group editor.
3. Click **OK** to return to the Dimension Groups view.

Deleting dimension groups

Delete dimension groups from the Dimension Groups view.

To delete a dimension group:

1. On the Project Explorer's Project tab, double-click **Dimension Groups** to open the Dimension Groups view.
2. Select the dimension group you want to remove from your project, and click **Delete.**
3. When the confirmation message appears, click **Yes.**

For information on displaying dimension groups in an Endeca-enabled Web application, see the *Endeca Basic Development Guide*.

Dimension Group editor

Use the Dimension Group editor to create new dimension groups and modify existing ones.

You use the Dimension Group editor to create new dimension groups and modify existing ones. The Dimension Group editor contains the following fields:

Option	Description
Group name	A unique name for this dimension group.
All dimensions	A list of all the dimensions that exist in your project.
Dimensions in group	A list of dimensions that are part of this group.
Add	Adds the selected dimension, from the "All dimensions list," to the group.
Remove	Removes the selected dimension, from the "Dimensions in group list," from the group.

Adding a dimension to an existing group

From the Dimensions view, select a dimension and a group you will add it to.

To add a dimension to an existing dimension group:

1. In the Dimensions view, select the dimension you want to add to a group and click Edit.
2. In the Member of this Dimension Group list, select the dimension group to which you want to add this dimension.
3. Click OK.

Dimension groups versus dimension hierarchy

Dimension hierarchy refines search by eliminating all but one dimension value from a dimension in the navigation state. Dimension groups allow users to select values from each dimension in the group.

Dimension groups allow the user to select dimension values from each of the dimensions contained in them. If the relationships made by a dimension group were instead created using dimension hierarchy, once a dimension value had been selected from one of the branches, then the remaining dimension values would no longer be available for refinement.

For example, in mutual funds data, a user may want to navigate on a variety of performance criteria. A Performance dimension group that contains YTD Total Returns, 1 Year Total Returns, and Five Year Total Returns dimensions would allow the user to select criteria from all three dimensions. If the same relationship had been created using dimension hierarchy, then once a selection had been made from the YTD Total Returns branch, the 1 Year Total Returns and Five Year Total Returns branches would no longer be available for navigation.

Dimension groups and ranking

The display order of dimension groups is determined by the ranking of the individual dimensions within the groups.

A dimension group inherits the highest rank of its member dimensions. For example, if the highest-ranked dimension in dimension Group A is ranked higher than the highest-ranked dimension in Group B, then group A will be ordered before group B.

Dimension groups are also ranked relative to dimensions that are not included in groups. Continuing the previous example, if Dimension XYZ has a rank that is lower than Group A but higher than Group B, the ranking would look like this:

Group A
Dimension XYZ
Group B

Dimensions with the same rank are ordered alphanumerically by name. It is important to note that dimension name, not dimension group name, determines the display order of dimension groups in this situation. Dimension groups are ordered according to their highest alphanumerically ranked member dimension. So dimension group Z, which contains dimension H, will be ordered before dimension group A, which contains dimension I.

For more information on ranking, see Ranking dimensions manually.

Dimension Groups view

The Dimension Groups view displays all of the dimension groups in your project.

The Dimension Groups view contains two columns:

Field	Description
Name	A unique name for a dimension group.
Dimensions	The dimensions that make up the group.

Chapter 5

Working with Your Pipeline

About pipelines

A pipeline functions as the script for the entire data transformation process that occurs when you run the Forge program.

The pipeline specifies things like the format and location of the source data, any changes to be made to the source data (standardization), and the mapping method to use for each of the source data’s properties. A pipeline is composed of a collection of components. Each component performs a specific function during the transformation of your source data into Endeca records. Components are connected by links, giving the pipeline a sequential flow from inputs to outputs.

You add and edit components in your pipeline using the Pipeline Diagram editor. The pipeline diagram graphically depicts the components in your pipeline and the links between them. It describes the series of transformations that occur in the process of converting raw data to a format that the Endeca MDEX Engine can use, making it easy for you to trace the logic of your data transformation. The pipeline diagram is the best way to maneuver and maintain a high-level view of your pipeline as it grows in size and complexity.

All Endeca projects require a main pipeline to process baseline updates. A baseline update (also called a full update) is a complete re-index of the entire dataset. Baseline updates occur infrequently, usually once per day or once per week. They usually involve the customer generating an extract from their database system and making the files accessible either on an FTP server or on the indexing server. This data is processed by Forge and the Dgidx Indexer, and is then finally made available through the MDEX Engine. You define the steps in a baseline update in your project's main pipeline.

Endeca projects may optionally contain a partial pipeline to process partial updates. A partial update is a much smaller change in the overall dataset. Partial updates affect a small percentage of the total records in the system, and therefore occur much more frequently. They consist of a much smaller extract from the customer’s database and contain volatile information. For example, the price and availability of products on a retail store site are usually volatile.

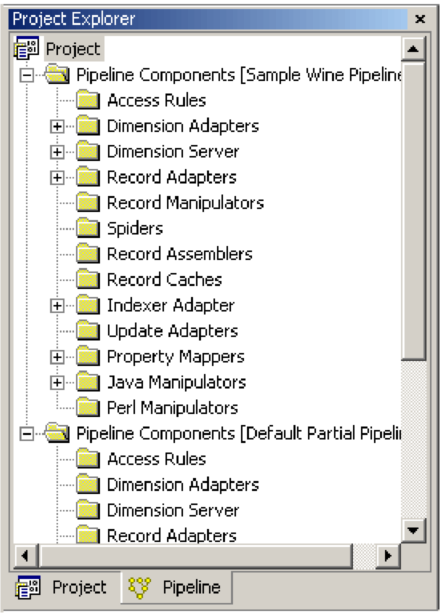
You can create, view, and edit both types of pipelines in Developer Studio. You access your baseline and partial update pipelines via the Pipeline tab, and the Pipeline Diagram and Partial Pipeline Diagram editors.

Note: Refer to Section I in the *Endeca Forge Guide* for an introduction to main pipelines. Refer to the *Endeca Partial Updates Guide* for details on using and implementing partial pipelines.

About the pipeline tab

The Project Explorer's Pipeline tab displays all of the pipeline components in your project, for both your baseline pipeline and your partial update pipeline.

If you double-click any component, Developer Studio opens the appropriate editor (either the Pipeline Diagram editor or Partial Pipeline Diagram editor), with the selected component highlighted, and displays the component's editor.



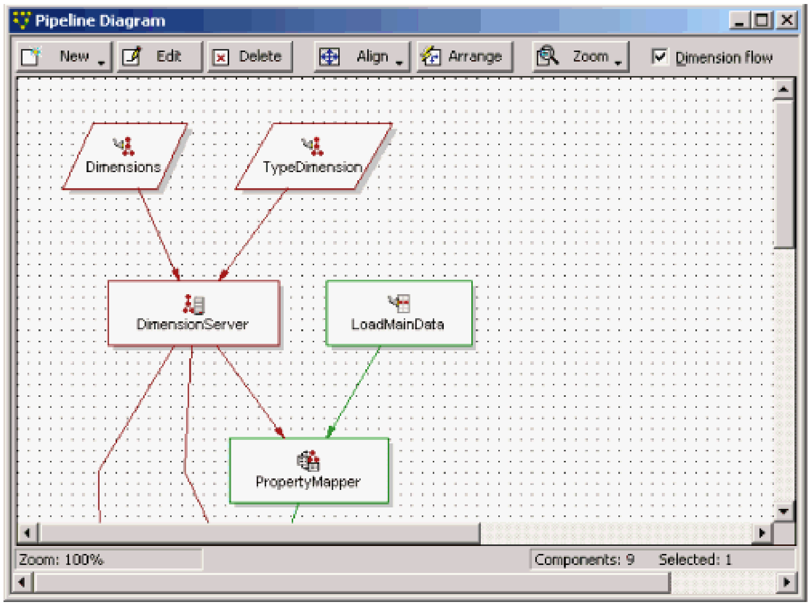
Project Explorer

- Project
  - Pipeline Components [Sample Wine Pipeline
    - Access Rules
    - Dimension Adapters
    - Dimension Server
    - Record Adapters
    - Record Manipulators
    - Spiders
    - Record Assemblers
    - Record Caches
    - Indexer Adapter
    - Update Adapters
    - Property Mappers
    - Java Manipulators
    - Perl Manipulators
  - Pipeline Components [Default Partial Pipeli
    - Access Rules
    - Dimension Adapters
    - Dimension Server
    - Record Adapters

About the Pipeline Diagram editor

The Pipeline Diagram editor depicts all of the components in your main pipeline and the relationship between them.

It describes the flow of events that occur in the process of converting raw data to a format that the Endeca MDEX Engine can use, making it easy for you to trace the logic of your data model. The pipeline diagram is the best way to maneuver and maintain a high-level view of your pipeline as it grows in size and complexity. The background of the main pipeline editor is white.



About pipeline components

A pipeline consists of components that perform specific functions during the transformation of your source data into Endeca records. These components are connected by links, giving the pipeline a sequential flow from inputs to outputs.

Both baseline and partial update pipelines can use any of these components. You can access the components in your project from three places in Developer Studio:

- The Project Explorer's Pipeline tab.
- The Pipeline Diagram editor for the main pipeline.
- The Partial Pipeline Diagram editor for the partial update pipeline.

Note: Refer to Section I of the *Endeca Forge Guide* for an introduction to pipelines. For information on partial updates, see the *Endeca Partial Updates Guide*.

About adapters

Adapter components load and save data.

Component	Description
Dimension Adapters	You use dimension adapters to load dimension information into your pipeline. Dimension adapters can load dimension information in either XML or binary format.
Record Adapters	Record adapters load and save records. Input record adapters can load data in a variety of formats, including delimited, JDBC, ODBC,

Component	Description
	and so on. Output record adapters are generally used for diagnostic purposes only.
Record to Dimension Adapters	Record to dimension adapters dynamically build hierarchical dimensions from a subset of the information in records loaded into the pipeline. The record to dimension adapter offers an alternative to working with externally created dimensions, and saves you from having to manually create hierarchical dimensions in Developer Studio.
Indexer Adapters	Indexer adapters save data that is ready to be indexed by the Dgidx program. They take all record, dimension hierarchy, and index configuration information from the pipeline and combine it in a format that is ready for the indexer (Agidx or Dgidx). An indexer adapter will generally be the final component in a baseline update pipeline.
Update Adapters	Update adapters process partial update information to perform a live update of volatile information on a running Endeca MDEX Engine.

About manipulators

Manipulator components change the data associated with an Endeca record.

Component	Description
Perl Manipulators	Perl manipulator components perform data transformation, including changing properties and property values, deleting records with particular properties, and so on. Perl manipulators use the Forge API for Perl to perform these tasks.
Java Manipulators	Java manipulator components perform data transformation, including changing properties and property values, deleting records with particular properties, and so on. Java manipulators use Java to perform these tasks. You add Java manipulators to Developer Studio similar to how you add Perl manipulators.
Record Manipulators	Record manipulator components provide support, such as URL extraction, for the Endeca Crawler, that allows you to crawl document hierarchies on a file system or over HTTP or HTTPS. **Note:** Previous to the introduction of the Perl manipulator component, record manipulators were used to perform property and property value manipulation. This functionality still exists but Oracle highly recommends that you use Perl manipulators instead.

About property mappers

Property mappers connect source data to properties or dimensions.

Property mapper components map properties on the records in your source data to Endeca properties and/or dimensions to make them navigable, displayable, both, or neither.

About record assemblers

Record assembler components merge data from one or more secondary data sources to the current record.

Record server components

Describes record server components.

"Record server" is a general name for any component that can serve records to a downstream component. Record server components include record adapters, record assemblers, record caches, record manipulators, Perl manipulators, spiders, property mappers, indexer adapters, and update adapters.

About utility components

Utility components help you perform basic tasks such as logging and record caching.

Component	Description
Dimension Servers	Dimension servers work in conjunction with dimension adapters, and serve as a centralized source of dimension information for all other pipeline components.
Record Caches	A record cache component stores a temporary copy of record data that has been read in by a record adapter. With two exceptions, all sources feeding a join (record assembler) must be record caches.
Spiders	The spider component is the core of the Content Acquisition System, which adds the capability to crawl document hierarchies on a file system or over HTTP or HTTPS.

Adding a pipeline component

Add new pipeline components from the Pipeline Diagram editor.

To add a pipeline component:

1. In the Pipeline Diagram editor, click New, and then choose the type of component to add. The component editor appears.

2. In the Name text box, type a unique name for the component.
3. In the Sources tab, set any required data sources.
4. Use the other tabs to specify other component attributes.
5. Click OK.

Adding a spider

A spider component crawls documents rather than loading records from a file, thereby adding document crawling capabilities to your pipeline.

The spider handles URL management and content manipulation. Using spiders, you can configure a pipeline that crawls the network at which it is directed, and facilitates the processing of the documents it finds there using the Endeca Information Transformation Layer (ITL).

Note: The Endeca Crawler, which uses the spider component, has been deprecated and support for it will be removed in a future version. It is recommended that you use the Endeca Web Crawler for Web crawls and the Endeca CAS Server for file system crawls.

To be able to crawl, a spider component has a dependency on at least one other pipeline component. Upstream, there needs to be a record adapter with a Format of type "document." This record adapter needs to contain a reference to the spider as its URL source. The two components work together to crawl a content repository (such as a Web server, file server, or FTP server). Between the record adapter and the spider, other pipeline components can be implemented. For example, you can have a record manipulator between the record adapter and the spider.

To add a spider component:

1. In the Pipeline Diagram editor, click **New**, and then choose **Spider**.
   The Spider editor appears.
2. In the **Name** text box, type a unique name for this spider
3. In the **General** tab, do the following:
   a) (Optional) In the **Maximum hops** text box, type an integer greater than zero to limit the depth of the crawl. If this field is not set, the crawl depth is unlimited.
   b) (Optional) In the **Maximum depth** text box, type an integer greater than zero to limit the depth of the URL path. If this field is not set, the URL depth is unlimited.
   c) If you are using a `robots.txt`file, in the **Agent name** text box, type the name of the spider as it will be referred to in the `User-agent`field of a `robots.txt`file.
   d) If the spider is performing a differential crawl, in the **Differential crawl URL** text box, type the location to store the spider's state (as a state.tmp file) between Forge executions
   e) If you do not want the spider to conform to the `robots.txt`standard, check **Ignore robots**.
   f) If you do not want the spider to accept cookies, check **Disable cookies**.
4. In the **Root URLs** tab, add one or more URLs from which a crawl can be launched.
5. (Optional) In the **URL Configuration** tab, add enqueue URLs and URL filters.
6. In the **Sources** tab, choose a record source from the list.
7. (Optional) In the **Comment** tab, add a comment for the component.
8. Click **OK**.

Implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Forge Guide* for details.

Spider editor

The Spider editor contains a unique name for this spider.

The spider editor contains the following tabs:

General

Field	Description
Maximum hops	Optional. Limits the depth of the crawl to a maximum number of hops from the root URL (see below). The value must be an integer greater than zero. If no value is provided, the maximum number of hops is unlimited.
Maximum depth	Optional. Limits the depth of the crawl to a maximum depth of URL path. The value must be an integer greater than zero. If no value is provided, the maximum path depth is unlimited.
Agent name	Identifies the name of the spider as it will be referred to in the **User-agent** field of a `robots.txt file.` Required if you are following the `robots.txt` standard.
Differential crawl URL	Optional. When configuring a spider to perform a differential crawl, the differential crawl URL specifies a file location to store the spider's state between Forge executions. This file is read in at the beginning of a differential crawl to enqueue URLs from previous crawls and may be updated during the crawl.
Ignore robots	When checked, the spider does not adhere to the `robots.txt` standard, which tells the spider which files it can crawl. By default, the spider follows the standard and looks for `robots.txt` on the server.
Disable cookies	When checked, the spider refuses cookies sent from a host server during a crawl.

Root URLs

The Root URLs tab is where you manage root URLs, which specify the location from which the spider starts crawling. There must be at least one root URL specified for each spider.

URL Configuration

The URL Configuration tab is where you manage enqueue URLs and URL filters.

Field	Description
Enqueue URLs	Optional. Takes each URL link from a specified property and adds them to the queue for further filtering.
URL filters	Optional. Provides pattern matching capabilities to control document filtering.

Timeout

The Timeout tab is where you manage connection and fetch timing. It contains the following fields:

Option	Description
Maximum time spent fetching a URL	When checked, type the time in seconds.
Maximum time to wait for a connection to be made	When checked, type the time in seconds.
Abort fetch if transfer rate falls below	When checked, type the bytes per second and the number of seconds.

Proxy

The Proxy tab is where you establish the use of proxy servers. It contains the following fields:

Option	Description
Proxy mode list	Choose whether to use no proxy servers, a single proxy server, or separate proxy servers for HTTP and HTTPS requests.
HTTP proxy server	The hostname and port number for the HTTP proxy server (if one is being used).
HTTPS proxy server	The hostname and port number for the HTTPS proxy server (if one is being used).
Bypass URLs	When clicked, opens the Bypass URLs editor, where you specify the list of URLs that should be fetched directly, bypassing any proxy servers.

Sources

Required. A choice of record server components in the project.

Comment

Optional. Provides a way to associate comments with a pipeline component.

Adding an update adapter

An update adapter is part of a partial update pipeline. It writes partial update information to a running Endeca MDEX Engine to perform a live update.

The partial update pipeline reads its input as normal records, using a normal record adapter. The input records are transformed into record updates using the update record expression, then written out using an update adapter. An update adapter is typically the last component of a partial update pipeline.

Forge processing for full updates and partial updates is done in separate pipelines. Therefore, you would never add an update adapter to a full update pipeline. See the *Endeca Partial Updates Guide* for detailed information on how to configure a partial update pipeline.

To add an update adapter:

1. In the Pipeline Diagram editor, click **New**, and then choose **Update Adapter**.
   The Update Adapter editor appears.
2. In the **Name** text box, type a unique name for this update adapter.
3. In the General tab, do the following:
   a) In the Output URL text box, type the location to which Forge writes its files and processed records.
   b) (Optional) In the **Output prefix** text box, type a filename prefix to add when Forge writes its output files. If none is specified, Forge uses the prefix `Out`.
   c) (Optional) Check **Multi** if you want Forge to be able to read from multiple files.
   d) If you checked Multi in step C , then in the **Multi property** text box, type a property name that indicates the location to which a record coming through Forge should be written
   e) If you enabled **Filter unknown properties** in your full update pipeline's indexer adapter, enable **Filter unknown properties** in the update adapter.
4. In the Sources tab, do the following:
   a) In the Record Source list, choose a record source.
   b) (Optional) In the Dimension Sources list, choose one or more dimension sources.
5. (Optional) If you are using an Agraph in your implementation, do the following:
   a) Select **Enable Agraph support**.
   b) In **Number of Agraph partitions**, specify the number of child Dgraphs that the Agraph coordinates.

   If you want to change the partition property, open the Properties view and modify which properties are enabled for rollup and record spec. For more detailed information, see the *Endeca Advanced Development Guide*.
6. (Optional) In the Comment tab, add a comment for the component.
7. Click **OK**.

Fully implementing this feature requires additional work outside of Developer Studio. Please refer to the *Endeca Partial Updates Guide* for details.

Update Adapter editor

The Update Adapter editor contains a unique name for this update adapter.

The Update Adapter editor contains the following tabs:

General

The General tab contains the following options:

Option	Description
Output URL	Required. Identifies the directory to which Forge writes its files and processed records. The path is either an absolute path or a path relative to the location of the Pipeline.epx file. With an absolute path, the protocol must be specified in RFC 2396 syntax. Usually this means the prefix file:/// precedes the path to the data file. Relative URLs must not specify the protocol and are relative to the URL used to locate the Pipeline.epx file.
Output prefix	Specifies the filename prefix to add when Forge writes its output files.
Multi	Optional. The **Multi** and **Multi property** fields specify which file each record is written to. If **Multi** is checked, Forge checks the **Multi property** field for the location.
Multi property	Optional . Specifies a property name that indicates the location that a record coming through Forge should be written to. (Forge stores a property on each record that is the source URL for that record.) Setting this property is useful if you want to process multiple input files and also keep the output distinct.
Filter unknown properties	Optional. If you enabled **Filter unknown properties** in your full update pipeline's indexer adapter , enable **Filter unknown properties** in the update adapter.
Custom Compression level	Optional. Sets the level of compression to be performed on the data. The values can range from 0 to 10, with higher numbers indicating higher compression (smaller size, slower processing).

Sources

The Sources tab contains the following options:

Option	Description
Record source	Required. A choice of the record servers in the project.
Dimension source	A choice of the dimension adapters and dimension servers in the project.

Agraph

The Agraph tab contains the following options:

Option	Description
Enable Agraph Support	When checked, enables the Agraph program for use in a partial update pipeline.
Number of Agraph partitions	Specifies the number of child Dgraphs that the Agraph controls. In an Agraph implementation, this must be a value of 2 or more.
Partition property	The partition property is a read only field that identifies the property by which records are assigned to each partition. If you are using rollup capabilities, the rollup property displays as the partition property. If you do not have a roll up property, but do have a record spec property enabled in your project, the record spec property functions as the partition property. If neither a rollup nor a record spec property exists, the partition property is empty, and Forge assigns records to each partition according to a round-robin strategy.

See the *Endeca Advanced Development Guide* for detailed information on how to configure, provision, and run an Aggregated MDEX Engine implementation.

Comment

Optional. Provides a way to associate comments with a pipeline component.

Editing a pipeline component

Select a component from the Pipeline Diagram editor to make modifications.

1. In the Pipeline Diagram editor, double-click the component that you want to modify.
2. In the component editor, make the necessary changes.
3. Click OK.

Deleting a pipeline component

Delete any pipeline components from the Pipeline Diagram editor.

To delete a pipeline component:

1. In the Pipeline Diagram editor, click the component you want to delete. Use Shift-click to select multiple components.
2. Click Delete.

Record Manipulators and Expressions

This section provides information on creating record manipulators and adding expressions to them.

Adding a record manipulator

Add record manipulators from the Pipeline Diagram editor.

Several of Endeca's advanced features (namely, the content acquisition system and partial updates) require record manipulator components. Record manipulators contain expressions, which are functions that are evaluated against each record as it flows through the pipeline. When an expression is evaluated, it may change the current record. The changes take a variety of forms, from adjustments of property values to creation of new data.

To add a record manipulator:

1. In the Pipeline Diagram editor, click **New**.
2. Choose **Record > Manipulator** .
   The Record Manipulator editor appears.
3. In the **Name** text box, type a unique name for this record manipulator.
4. In the Sources tab, use the lists to select a record source and a dimension source.
5. (Optional) In the Record Index tab, do the following:
   a) Specify which properties or dimensions you want to use as the record index for this component.
   b) Indicate whether you want to disregard records with duplicate keys.

   Note: Developer Studio performs a case-insensitive search for duplicate keys.

6. (Optional) In the Comment tab, add a comment for the component.
7. Click **OK**.
   The new record manipulator appears in the pipeline diagram.
8. Add expressions to your record manipulator.

Note: Implementing this feature requires additional work outside of Developer Studio. Please refer to the following sections in the *Endeca Forge Guide* for details: Classifying Documents with Stratify, and Implementing the Endeca Crawler. See the *Endeca Partial Updates Guide* for details on implementing partial updates.

Editing a record manipulator

Select a manipulator from the Pipeline Diagram to edit.

To edit a record manipulator:

1. In the Pipeline Diagram, double-click the icon for the record manipulator you want to edit.
   The Expression editor opens.
2. Click **Properties**.
   The Record Manipulator editor opens.
3. Make any required changes.
4. Click **OK**.

5. In the Expression editor, click **Commit Changes**.
6. Close the Expression editor.

Implementing this feature requires additional work outside of Developer Studio. Please refer to the following sections in the *Endeca Forge Guide* for details: Classifying Documents with Stratify, and Implementing the Endeca Crawler. See the *Endeca Partial Updates Guide* for details on implementing partial updates.

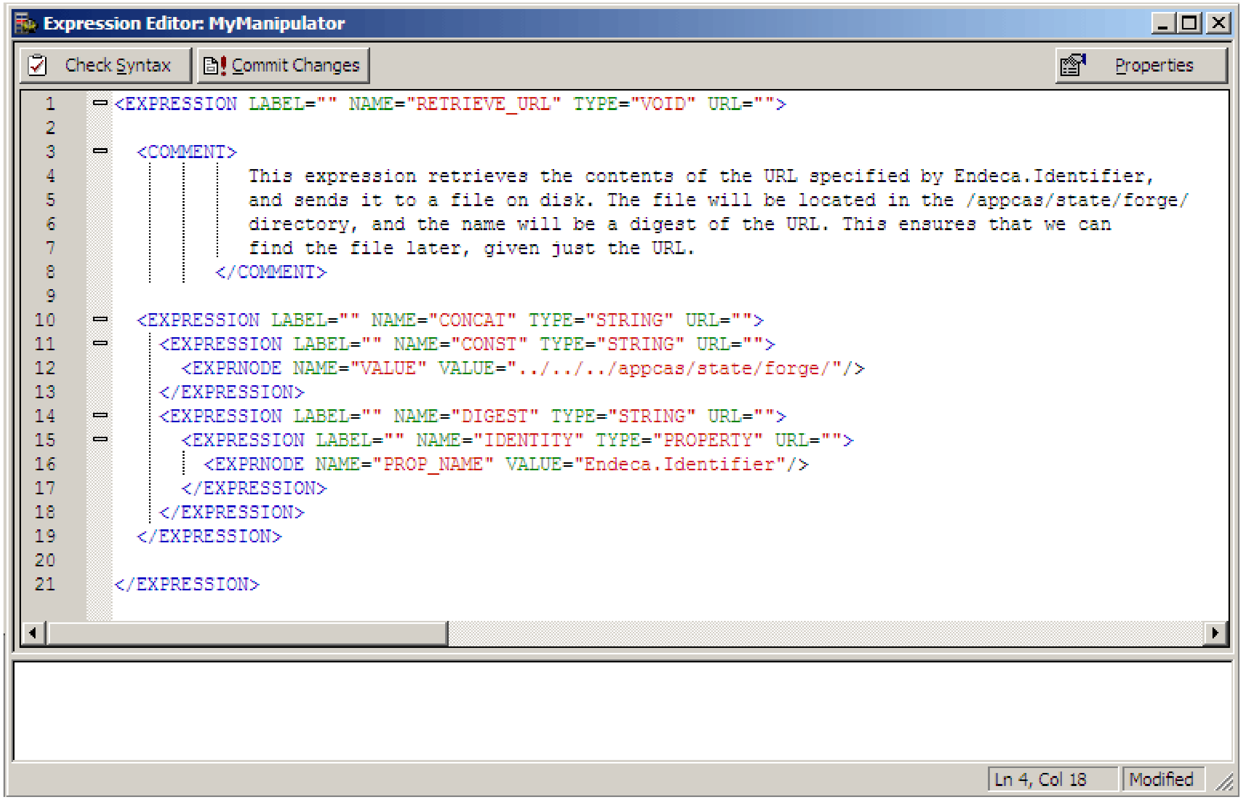
About expressions

Expressions are analogous to functions, and tell Forge which records, properties, or dimensions to affect, and how to affect them.

Record manipulator components do their work on record data by means of expressions. Record manipulator components can contain an arbitrary number of expressions which are evaluated against each record as Forge processes it. When Forge evaluates an expression, it may modify the current record. The changes take a variety of forms depending on the expression, from adjusting property values to creating new data.

Expressions can contain sub-expressions; the sub-expressions may provide values used by the parent expression, or the parent expression may provide control over which of the sub-expressions are evaluated. Also, many expressions contain expression nodes, which are name/value pairs that specify property names, constant values, dimension IDs, and so forth. Expression nodes represent parameters required by the expression. If a parent expression contains both sub-expressions and expression nodes, the nodes must be placed before the sub-expressions.

In Developer Studio, you write and modify expressions in the Record Manipulator component's XML editor. The following is an example of a record manipulator component with expressions, sub-expressions, and expression nodes.



```
<EXPRESSION LABEL="" NAME="RETRIEVE_URL" TYPE="VOID" URL="">

 <COMMENT>
      This expression retrieves the contents of the URL specified by Endeca.Identifier,
      and sends it to a file on disk. The file will be located in the /appcas/state/forge/
      directory, and the name will be a digest of the URL. This ensures that we can
      find the file later, given just the URL.
     </COMMENT>

 <EXPRESSION LABEL="" NAME="CONCAT" TYPE="STRING" URL="">
  <EXPRESSION LABEL="" NAME="CONST" TYPE="STRING" URL="">
   <EXPRNODE NAME="VALUE" VALUE="../../../appcas/state/forge/"/>
  </EXPRESSION>
  <EXPRESSION LABEL="" NAME="DIGEST" TYPE="STRING" URL="">
   <EXPRESSION LABEL="" NAME="IDENTITY" TYPE="PROPERTY" URL="">
    <EXPRNODE NAME="PROP_NAME" VALUE="Endeca.Identifier"/>
   </EXPRESSION>
  </EXPRESSION>
 </EXPRESSION>

</EXPRESSION>
```

See the Endeca Data Foundry Expression Reference for complete details on all expressions and their syntax. In addition, the documentation for the advanced features that use expressions provide detailed examples.

Adding an expression to a record manipulator

You add expressions to a record manipulator using the Expressions editor. You can have more than one Expression editor open at any given time, allowing you to cut or copy and paste between them.

To add an expression to a record manipulator:

1. In the Pipeline Diagram, double-click the icon for the record manipulator you want to edit.
   The Expression editor opens.
2. Enter your XML in the **Expression editor** window.

   Note: See the *Endeca Data Foundry Expression Reference* for complete details on all expressions and their syntax. In addition, the documentation for the advanced features that use expressions provide detailed examples.

3. To check your syntax, click **Check Syntax**.
   Syntax issues are displayed at the bottom of the Expression editor window.
4. To commit your changes, click **Commit Changes**.
5. To view and edit the Record Manipulator component's settings, click **Properties**.
   The Record Manipulator editor opens.
6. To close the Expression editor without committing changes, click the close **X** in the upper right corner of the **Expression editor** window.

Implementing this feature requires additional work outside of Developer Studio. Please refer to the following sections in the *Endeca Data Foundry Guide* for details: Crawler Implementations, Implementing Partial Updates, and Classifying Documents with Stratify.

Writing Out Data for Diagnostic Purposes

This section describes how to use the Pipeline Diagram editor to write record and dimension data output.

Writing out record data

To use a record adapter to save data, you specify Output as the direction of data flow, along with the location the data should be saved to and the file format for the saved data.

Output record adapters are generally used as diagnostic tools or for translation purposes.

To add an output record adapter to your pipeline:

1. In the Pipeline Diagram editor, click **New**.
2. Choose **Record**.
3. Choose **Adapter**.
   The Record Adapter editor appears.
4. In the **Name** text box, type a unique name for this record adapter.

5. In the General tab, do the following:
   a) In the Direction frame, choose **Output**.
   b) In the Format list, choose either **delimited**, **XML**, **binary**, **fixed width**, or **vertical** how you prefer to use the diagnostic data.
   c) In the **URL** text box, type the location to which the data will be written.
   d) In the **Encoding** text box, enter `UTF-8`.
   e) (Optional) To save on the amount of disk space used, check Custom Compression Level and slide the bar to Endeca's recommended value of 7.

   Note: Compressed data consumes less disk space but takes longer to read and write.

6. (Optional) In the Comment tab, add a comment for the component.
7. Click **OK**.

Writing out dimension data

To use an output dimension adapter to save data, you specify Output as the direction of data flow, along with the location the data should be saved to and the file format for the saved data.

Output dimension adapters are generally used as diagnostic tools.

To add an output dimension adapter:

1. In the Pipeline Diagram editor, click **New**.
2. Choose **Dimension > Adapter** .
   The Dimension Adapter editor appears.
3. In the **Name** text box, enter a unique name for the dimension adapter.
4. In the General tab, do the following:
   a) In the Direction frame, choose **Output**.
   b) In the **URL** text box, enter the location to which the dimension data will be saved.
   c) In the Format text box, select **XML**.
   d) (Optional) To save on the amount of disk space used, check Custom Compression Level and slide the bar to Endeca's recommended value of 7.

   Note: Compressed data consumes less disk space but takes longer to read and write.

5. In the Sources tab, choose a dimension source.
6. (Optionally) In the Comment tab, add a comment for the component.
7. Click **OK**.

Changing the Display of the Pipeline Diagram

This section describes how to change the appearance of components in the Pipeline Diagram editor.

Resizing the pipeline diagram

Zoom in or out to view your pipeline in better detail, or see more of it at once.

To zoom in or out on the pipeline diagram:

1. In the Pipeline Diagram editor, click **Zoom**.
2. From the Zoom menu, choose a percentage: **Fit in Window**, or **Always Fit in Window**.

Changing the alignment of selected components in the diagram

To clarify the view of your components in the pipeline, you may realign them.

To change the alignment of selected components in the diagram:

1. In the Pipeline Diagram editor, hold down the Shift key and click each of the components that you want to realign.
2. Click **Align**, and choose the orientation to which you want to align the selected components.
3. (Optional) To undo the last alignment action:
   a) Click **Align**.
   b) Choose **Undo Align**.

Returning to the default layout

After moving elements in the pipeline diagram, you can easily return to the default layout.

To restore the original layout of the pipeline diagram:

In the Pipeline Diagram editor, click **Arrange**.

Displaying components that have dimension sources

The Pipeline Diagram editor gives you the option of looking at your entire pipeline or focusing on a particular data flow. This feature is especially useful when your project becomes very large.

To display components that have dimension sources:

Check **Dimension Flow**.

Dimension flow component outlines and the lines that connect them are red.

Hiding the dimension flow

The Pipeline Diagram editor gives you the option of looking at your entire pipeline or focusing on a particular data flow. This feature is especially useful when your project becomes very large.

To hide the dimension flow:

Clear the **Dimension Flow** check box.

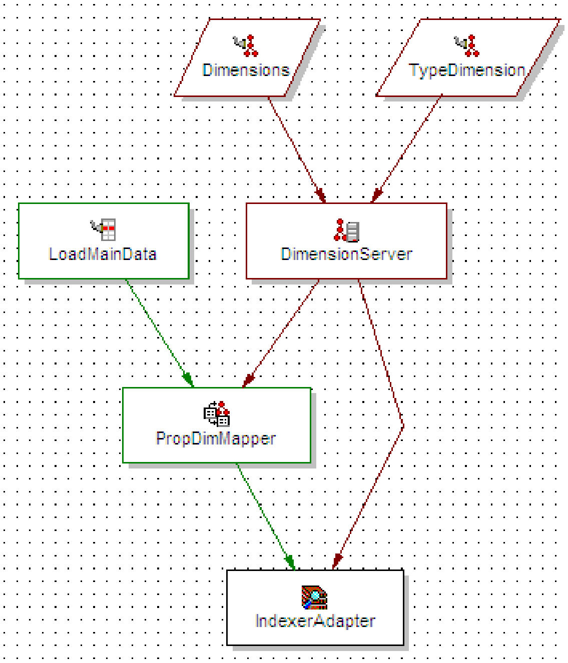
What remains, the Record flow, is outlined in green.

About specifying data flow among components

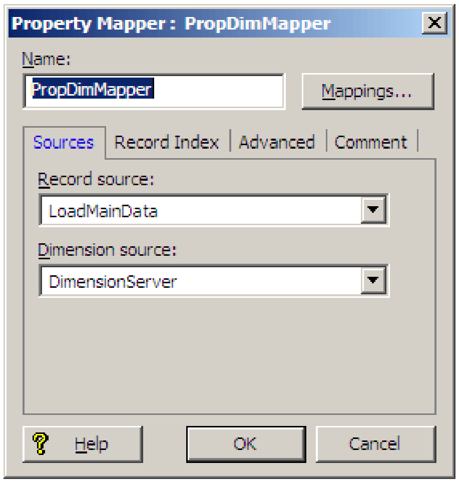
You must give every component in your pipeline a unique name that identifies it to the other components. You use these names to specify cross-references between components, effectively creating a flow of data through the pipeline.

For example, starting from the bottom, in the following illustration:

- IndexerAdapter requests its data from PropDimMapper and DimensionServer.
- PropDimMapper requests its data from LoadMainData and DimensionServer.
- DimensionServer requests its data from Dimensions and TypeDimension.
- LoadMainData, Dimensions, and TypeDimension get their data from source files (this is indicated by the lack of arrows feeding them).



When you specify a data source within a component's editor, you are indicating which of the other components will provide data to that component. Components can have multiple data sources, such as the PropDimMapper component below, which has both a record source, LoadMainData, and a dimension source, DimensionServer.



Property Mapper : PropDimMapper

Name: PropDimMapper

Record source: LoadMainData

Dimension source: DimensionServer

Viewing data flow among components

The Pipeline Diagram in Developer Studio displays a graphical representation of the components in your pipeline, making it easy to see the relationships and data flow among them.

To display the pipeline diagram editor:

- Choose Pipeline Diagram from the View menu.

Connecting pipeline components graphically

You can add and connect pipeline components using their respective editors. Alternatively, you can connect them graphically in the Pipeline Diagram editor.

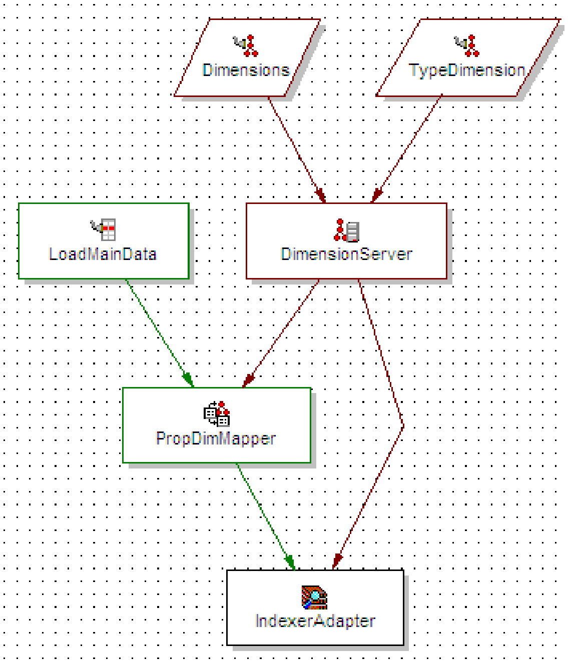
To connect pipeline components graphically:

1. Choose Pipeline Diagram from the View menu to display the Pipeline Diagram.
2. Click the component that you want to serve as the data source to highlight it.
3. Drag the black square in the center of the highlighted component to the component you want to connect it to. A line with an arrow indicating the direction of the data flow appears between the components.

About adding and removing pipeline components

When you add or remove components, you must be careful to make any data source changes required to maintain the correct data flow.

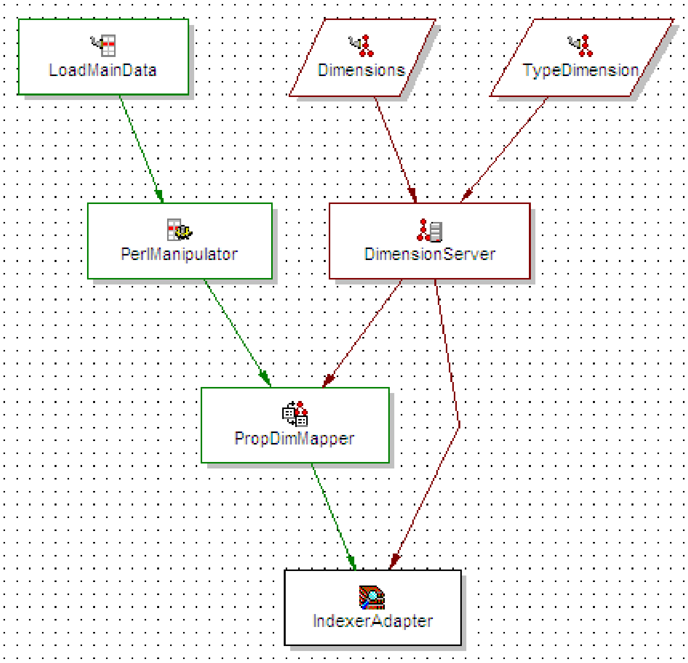
To illustrate this point, examine the following pipeline:



If the example above is modified to include another component, PerlManipulator, that component must come between LoadMainData and PropDimMapper in the data flow of the pipeline. Adding PerlManipulator in this location requires that:

- PerlManipulator’s data source is set to LoadMainData.
- PropDimMapper’s data source is changed to PerlManipulator.

The resulting pipeline diagram would look like this:



Chapter 6

Advanced Features

This section describes the advanced features in Developer Studio.

Restricting Access to Records

This section describes how to use the Pipeline Diagram editor to control which records a user can access.

About access rules

Access rules, in conjunction with Endeca's Access Control System, allow you to control which records a user can access.

Access rules, in conjunction with Endeca's Access Control System, allow you to control which records a user can access. Access rules use an if/then construct of this form:

if `<property>` equals `<value>` then grant read permissions to `<group name>`

For example:

if `language`equals `italian`then grant read permissions to `italians`

Access rules add an Endeca.ACL.Allow.Read property to records to identify which groups may access them. Any record whose `<property>` equals `<value>` has an Endeca.ACL.Allow.Read property with a value of `<group name>` added to it. The MDEX Engine then uses this property to filter records according to a group's access permissions. Continuing the example above, each record that has the value `italian` for the property `language` will have an Endeca.ACL.Allow.Read property added to it with the value `italians.`

You must manually add the Endeca.ACL.Allow.Read property to your project and enable it for record filtering before creating access rules. In other words, the Endeca.ACL.Allow.Read property must exist in order for it to be added to records during processing.

You use the Access Rules component, on the Pipeline tab, to create and manage access rules. Each access rules component must have either a property mapper or another access rules component as its record source. For internal reasons, you also must specify a dimension source for an access rules component, even though, externally, access rules do not appear to need them. Use the same dimension source that you use for your property mapper (this should be a dimension server).

About the Access Control System

Access rules must be used in conjunction with Endeca's Access Control System. You use the Access Control System to authenticate a user's identity against and obtain authorization information from a variety of external systems, such as an LDAP directory. The authorization information is used to control which records are retrieved during a query.

After the Access Control System authenticates the user against the specified directory, the returned group information for the user is automatically transformed into a user entitlement filter. The user entitlement filter defines the user's access rights to the data in the Endeca implementation. This filter is automatically added to every query that the user makes to the MDEX Engine, which uses the filter to return only those records that the user has a right to see. In effect, this filtered query returns only those records whose Endeca.ACL.Allow.Read property values match the values specified in the filter.

- See the *Endeca Security Guide* for detailed information on configuring and using the Access Control System.
- Unlike either of the Endeca crawlers, the Access Control System uses only the Endeca.ACL.Allow.Read permission to determine record access. It ignores all others, including Endeca.ACL.Deny.Read.

Working with multiple access rules

Having two rules in a single Access Rules component is equivalent to having two Access Rules components, each with a single rule.

Each Access Rules component can contain multiple rules. You can also have multiple Access Rules components in your pipeline. The way you organize your rules is arbitrary and you can choose whichever organizational method works best for your situation.

Access rules do not support boolean ANDs. However, you can add multiple rules to simulate this effect. For example, these two rules evaluate the same property but give permissions to two different groups.

if `language` equals `english` then grant read permissions to `canadians`

if `language` equals `english` then grant read permissions to `americans`

If `language=english` for Record A, then both the `canadians` and `americans` are given access to Record A. Records that have permissions granted to multiple users or groups will have multiple instances of the Endeca.ACL.Allow.Read property. In this case, Record A would have both Endeca.ACL.Allow.Read=canadians and `Endeca.ACL.Allow.Read=americans` appended to it.

On the other hand, the following two rules evaluate two different properties but give permissions to the same group. If either of the If statements evaluate to true, the swiss group is granted permission to the record.

if `language` equals `french` then grant read permissions to `swiss`

if `language` equals `german` then grant read permissions to `swiss`

Creating the Endeca.ACL.Allow.Read property

In order to use access rules, you must create an Endeca.ACL.Allow.Read property and enable it for record filtering. You only need one instance of this property per project, even if you have multiple access rules that use it.

Developer Studio automatically creates a mapping for the Endeca.ACL.Allow.Read property in the property mapper so you don't have to do it manually.

To create the Endeca.ACL.Allow.Read property with a property mapper:

1. On the Project tab, double-click **Properties** to open the Properties view.
2. In the Properties view, click **New**.
   The New Property editor appears.

3. In the Name box, enter `Endeca.ACL.Allow.Read` as the property name.
4. In the Type list, choose **Alpha**.
5. Check **Enable for record filters**.
6. Disable "Show with record list" and "Show with record."

   As a general rule, you should disable 'Show with record list' and 'Show with record,' unless you want the Endeca.ACL.Allow.Read property to appear in your Web application.

7. Click **OK**.

Creating an access rule

Allow a group to access records matching a value specified in the access rule.

Follow the procedure below to create an Access Rules component in your pipeline and add rules to it.

To create access rules:

1. If you haven't already, create an Endeca.ACL.Allow.Read property and enable it for record filtering.
2. In the Pipeline Diagram, click **New > Access Rules** .
3. Enter a unique name for the access rule in the **Name** field.

   Note: Access Rule components can only use Property Mappers or other Access Rule components as their record source. The Record Source menu is limited accordingly.

4. Choose a record source from the Record Source menu.
5. Choose a dimension source from the Dimension Source menu. Use the same dimension source that you use for your property mapper. This should be a dimension server.
6. Add access rules:

   a) Click the **Rules** tab.

   b) Click Add.
   The Edit Access Rule editor appears.

   c) In the **If** area, select the property you want to use to identify records for access and enter a value.

   Note: The value you enter must be an exact match with the value in the property on the record. For example, if you enter "merlot" it will not match properties that have a value of "French merlot".

   d) In the **Then** area, enter the name of a group.

   Note: This group is allowed access to any records that have the property and value you specified in the previous step.

   e) Click **OK** to close the Edit Access Rule editor.

7. To add additional access rules, repeat step 6.
8. To remove a rule:

   a) Select it in the list.

   b) Click **Remove**.

9. To modify a rule:

   a) Select it in the list.

   b) Click **Modify**.

10. When you are done specifying access, click **OK** to close the Access Rules editor and save your changes.

Running a baseline update for access rules

After modifying rules, run a baseline update.

Any changes to access rules require you to run a baseline update using Endeca Workbench.

Configuring the Web application to use access rules

You can use access rules with the Endeca Standard Application, or with your own custom application.

With the exception of enabling authentication, all of the configuration required to use access rules in the Standard Application has already been done for you.

If you want to use access rules with your own custom application, you will need to configure the Access Control System from scratch. Refer to the *Endeca Security Guide* for details on how to do this.

Crawling source data

Oracle Endeca Guided Search supports file system and Web crawlers that gather data from source documents and allow you to write out Endeca records based on the source documents.

You can modify the records as necessary in your pipeline by adding dimension and property values, and build an Endeca application to access the records. This allows your application users to search and navigate the document contents contained in the records. The following crawlers are supported:

- The Endeca Web Crawler is appropriate for large-scale crawling of Web documents. Supports crawling HTTP and HTTPS sites. Available as part of the Endeca CAS package.
- The Endeca CAS Server can run both file system and CMS (Content Management System) crawls. Supports crawling Windows and UNIX systems. Available as part of the Endeca CAS package.
- The Endeca Crawler is a lightweight crawler that is configured via a Spider component in Developer Studio. Supports crawling HTTP, HTTPS, and file systems. Note that the Endeca Crawler is deprecated, and therefore it is recommended that you use the Endeca Web Crawler or the Endeca CAS Server for your crawling requirements.

For more information on the CAS crawlers, see the *Endeca Web Crawler Guide* or the *Endeca CAS Server Guide*. For details on the Endeca Crawler, see the *Endeca Forge Guide*.

About implementing partial updates

Pipelines that implement partial updates must incorporate an update adapter.

The Endeca MDEX Engine processes two types of updates:

- Full updates
- Partial updates

A full, or baseline, update fully reprocesses the entire dataset. Full updates occur infrequently—usually once per day or once per week. They usually involve generating an extract from your database system and making the files accessible either on an FTP server or on the indexing server. This data is processed by Forge and Dgidx and then finally made available through the Endeca MDEX Engine.

A partial update reprocesses a smaller subset of the overall dataset. Partial updates affect a small percentage of the total records in the system. They occur much more frequently and consist of a much smaller extract from your database that contains volatile information. For example, the price and availability of products on a retail store site are usually volatile. Pipelines that implement partial updates must incorporate an update adapter.

See the *Endeca Partial Updates Guide* for detailed information on how to configure a partial update pipeline.

Working with external taxonomies

You may build all or part of a logical hierarchy for your data set outside of Developer Studio and then transform that logical hierarchy into Endeca dimensions and dimension values for use in search and Guided Navigation.

Endeca offers functionality that allows you to work with external taxonomies, or logical data hierarchies. This capability allows you to build all or part of a logical hierarchy for your data set outside of Developer Studio and then transform that logical hierarchy into Endeca dimensions and dimension values for use in search and Guided Navigation.

This topic introduces the external taxonomy features and provides cross-references to the *Endeca Forge Guide*, where you can find complete implementation details.

Externally managed

An externally managed taxonomy is a logical hierarchy for a data set that is built and managed using a third-party tool. You can use the Discover and Load buttons in the Dimensions view to include an externally managed taxonomy in your project. At that point, the taxonomy becomes a dimension. However, it is a read-only dimension whose hierarchy is managed by the third-party tool that created it.

In Developer Studio, you can view this type of dimension and delete dimension values from it. You cannot, however, add dimension values to it. If you want to modify an externally managed dimension or add dimension values, you have to edit the taxonomy using the third-party tool and then update the taxonomy in your project.

See "Working with an Externally Managed Dimension" in the *Endeca Forge Guide* for implementation details.

Externally created

An externally created dimension also describes a logical hierarchy for a data set; however, the dimensions are built outside of Developer Studio in an .xml file. The logical hierarchy of the dimension conforms to Endeca's external interface for describing a data hierarchy-external_dimensions.dtd. Once you import an externally created dimension, its ownership is wholly transferred to Developer Studio. As such, you can modify the dimension in any way necessary in Developer Studio, but you can no longer edit it in the .xml file itself (unless you re-import it).

See "Working with an Externally Created Dimension" in the *Endeca Forge Guide* for implementation details.

The difference between externally managed and externally created taxonomies

It is important to clarify the difference between an externally managed taxonomy and an externally created dimension. An externally managed taxonomy is created using a third-party tool, managed by a third party tool, and transformed by Forge via .xslt into a read-only Endeca dimension. By comparison, externally created dimensions are manually created in an .xml file, imported via Developer Studio without an .xslt transformation, and are fully editable in Developer Studio.

Using Stratify taxonomies

Integrating a Stratify taxonomy and Stratify document classification capabilities into your pipeline is a special case of working with external taxonomies that allows you to classify unstructured source documents. For more information, see "Classifying Documents with Stratify" in the *Endeca Forge Guide*.

About the Agraph

The Agraph enables scalable search and navigation by partitioning a very large data set into multiple Dgraphs running in parallel.

Implementing the Agraph program allows your application users to search and navigate very large data sets. You add Agraph support to a pipeline by configuring the Agraph tab of either the Indexer adapter or the Update adapter.

See the *Endeca Advanced Development Guide* for detailed information on how to configure, provision, and run an Aggregated MDEX Engine implementation.

Chapter 7

Context Sensitive Help

The topics in this section pertain to the options available for specific windows or dialog boxes in Developer Studio.

About automatic phrasing

When an application user provides individual search terms in a query, the automatic phrasing feature groups those individual terms into a search phrase and returns query results for the phrase.

Automatic phrasing is similar to placing quotation marks around search terms before submitting them in a query. For example, 'my search terms' is the phrased version of the query my search terms. However, automatic phrasing removes the need for application users to place quotation marks around search phrases to get phrased results.

The result of automatic phrasing is that a Web application can process a more restricted query and therefore return fewer and more focused search results. This feature is available only for record search.

The automatic phrasing feature works by:

1. Comparing individual search terms in a query to a list of application-specific search phrases. The list of search phrases are stored in a project's phrase dictionary.
2. Grouping the search terms into search phrases.
3. Returning query results that are either based on the automatically phrased query, or returning results based on the original unphrased query along with automatically phrased 'Did You Mean?' (DYM) alternatives.

Point three above suggests the two typical implementation scenarios to choose from when using automatic phrasing:

- Process an automatically phrased form of the query and suggest the original unphrased query as a DYM alternative.

  In this scenario, the automatic phrasing feature rewrites the original query's search terms into a phrased query before processing it. If you are also using DYM, you can display the unphrased alternative so the user can opt-out of automatic phrasing and select their original query, if desired.

  For example, an application user searches a wine catalog for the terms "low tannin." The MDEX Engine compares the search terms against the phrase dictionary, finds a phrase entry for "low tannin," and processes the phrased query as "low tannin." The MDEX Engine returns 3 records for the phrased query "low tannin" rather than 16 records for the user's original unphrased query "low tannin." However, the Web application also presents a "Did you mean low tannin?" selection so the user may opt-out of automatic phrasing, if desired.

- Process the original query and suggest an automatically-phrased form of the query as a DYM alternative.

  In this scenario, the automatic phrasing feature processes the unphrased query as entered and determines if a phrased form of the query exists. If a phrased form is available, the Web application displays an automatically-phrased alternative as a "Did you mean?" option. The user can opt-in to automatic phrasing, if desired.

  For example, an application user searches a wine catalog for low tannin. The MDEX Engine returns 16 records for the user's unphrased query low tannin. The Web application also presents a "Did you mean "low tannin"?" option so the user may opt-in to automatic phrasing, if desired.

There are two tasks to implement automatic phrasing:

- Add phrases to your project using Developer Studio.
- Add Presentation API code to support either of the two implementation scenarios described above.

Note: Implementing search features requires additional work outside of Developer Studio. Refer to the *Endeca Advanced Development Guide* for details.

Additional Search Characters dialog box

Search characters are configured globally for all search operations.

To create additional search characters:

Enter search characters in the **Additional Search Characters** field.

Note: If you have more than one, do not separate them with commas or spaces—simply type them one after another.

Bypass URLs dialog box

Add, modify, or remove URLs you want access to without using a proxy server.

Option	Description
Bypass proxy server for these URLs	A list of URLs that you want to access without using a proxy server.
Add	Type the name of the host you want to access without the use of a proxy server in the field above and click Add. You can use wildcards to indicate a number of Web servers within a domain. Repeat this step as necessary for additional URLs.
Modify	Used to modify a URL in the bypassed URLs list. To modify a URL:

Option	Description
	1. Select it in the list. 2. Make your changes in the editing field. 3. Click **Modify**.
Remove	Removes the selected URL from the bypassed URLs list.

Dimension Mapping dialog box

This includes the name of the dimension the source property maps to, and which match mode is used.

Option	Description
Source property	Name of the source property to be mapped.
Target dimension	Name of the dimension the source property will map to.
Maximum length	Optional. Defines the maximum source property value length allowed when creating mappings. Source properties that have values that exceed this length are not mapped, and a warning is issued by the Forge Hierarchical Logging system. This value overrides the default maximum length that you set on the Advanced tab of the Property Mapper editor.
Match mode	The mapping mode to be used. • Normal Normal maps only those source property values that have a matching dimension value explicitly defined in the dimension hierarchy. Forge assigns the IDs for any matching dimension values to the Endeca records. Any source property values that do not have matching dimension values in the dimension hierarchy are ignored. In order for a source property value to match a dimension value, the dimension value's definition must contain a synonym that: • Is an exact text match, including capitalization and white space, to the source property value. • Has its Classify option enabled. **Note:** There are two types of advanced dimension values, range and sift. These

Option	Description
	dimension value types have different matching criteria than standard dimension values. See About range dimension values and About sift dimensions, respectively . • Must Match Must Match behaves identically to Normal, with the exception that Must Match issues a warning for any source property values that do not have matching dimension values. • Auto Generate Auto Generate specifies that Forge automatically generates a dimension value name and ID for any source property value that does not have a matching dimension value in the dimension hierarchy. Forge uses these automatically-generated names and IDs to tag the Endeca records the same as it would explicitly-defined dimension values. Auto Generate dramatically reduces the amount of editing you have to do to the dimension hierarchy. However, auto-generated dimensions are always flat. Auto-generated names and IDs are persisted in a file that you specify as part of a dimension server component.

Edit Static Relevance Rank Module dialog box

This shows the property or dimension by which records are sorted. Specify record sort order.

Option	Description
Current property or dimension	Indicates the current property or dimension used to define the sort order of record results. Records are sorted according to their values for the specified property or dimension.
New property or dimension	Indicates the new property or dimension used to define the sort order of record results.
Sort records in descending order	Indicates the order of the sort. For example, sorting by Price, descending displays the most expensive items first.

Enqueue URL dialog box

This shows the property storing URLs to other documents. You may remove Enqueue URL property from record after value is queued.

Option	Description
Property	The name of the property that stores links (URLs) to other documents to crawl.
Remove	Optional. Removes the Enqueue URL property from the record after its value has been queued.

Importing phrases from an XML file

You import an XML file of phrases using the Import Phrases dialog box in Developer Studio. The Import Phrases dialog box can be accessed from either the File menu or from the Automatic Phrasing dialog box.

Before you import the XML file, it must conform to phrase_import.dtd, in the Endeca Navigation Platform conf/dtd directory.

Here is a simple example of a phrase file that conforms to phrase_import.dtd:

```
<?xml version='1.0' encoding='UTF-8' standalone='no' ?> <!DOCTYPE PHRASE_IMPORT SYSTEM
'phrase_import.dtd'>

<PHRASE_IMPORT>

<PHRASE>Napa Valley</PHRASE>

<PHRASE>low tannin</PHRASE>

</PHRASE_IMPORT>
```

To import phrases from an XML file:

1. In the Project Explorer, expand **Search Configuration**.
2. Double-click **Automatic Phrasing**.
   The Automatic Phrasing dialog box displays.
3. Click **Import Phrases**.
   The Import Phrases dialog box displays.

   Note: Alternatively, you can select Import Phrases from the File menu to invoke the Import Phrases dialog box.

4. Either type the path to your phrases file or click the **Browse** button to locate the file.
5. Click **OK** on the Import Phrases dialog box.
6. Click **OK** on the Automatic Phrasing dialog box.
   The Messages pane displays the number of phrases read in from the XML file.
7. Select **Save** from the File menu.

Join Entry dialog box

This shows the record source for this join entry, and options for adding, modifying, or removing key components of this entry's join key.

Option	Description
Record source	The name of the record source that feeds this join entry.
Keys	A list of the key components that make up this join entry's join key.
Add	Allows you to add a key component to the join key.
Edit	Allows you to edit the selected key component.
Remove	Removes the selected key component.
Up	Moves the selected key component up in the Keys list.
Down	Moves the selected key component down in the Keys list.

Key Component dialog box

Select either a source property or dimension as your key component.

Option	Description
Type	Choose "Custom property" if the key component will be a source property. Choose "Dimension" if the key component will be a dimension.
Custom property field	Appears when Type is set to "Custom property" and allows you to enter the name of a source property as your key component.
Dimension drop-down list	Appears when Type is set to "Dimension" and allows you to select a dimension as your key component.

Mappings dialog box

This dialog contains a Mappings table, and options to create new, modify, or remove mappings.

Option	Description
Mappings table	This table displays all of the source property mappings that have been defined for the project. It includes the following columns: • Source: The source property to be mapped. • Target: The dimension or Endeca property that the source property will be mapped to. • Match mode: For dimension mappings, indicates the type of match mode used. Normal Normal maps only those source property values that have a matching dimension value explicitly defined in the dimension hierarchy. Forge assigns the IDs for any matching dimension values to the Endeca records. Any source property values that do not have matching dimension values in the dimension hierarchy are ignored. In order for a source property value to match a dimension value, the dimension value's definition must contain a synonym that: • Is an exact text match, including capitalization and white space, to the source property value. • Has its Classify option enabled. **Note:** There are two types of advanced dimension values, range and sift. These dimension value types have different matching criteria than standard dimension values. See About range dimension values and About sift dimensions, respectively. Must Match Must Match behaves identically to Normal, with the exception that Must Match issues a warning for any source property values that do not have matching dimension values. Auto Generate Auto Generate specifies that Forge automatically generates a dimension value name and ID for any source property value that does not have a matching dimension value in the dimension hierarchy. Forge uses these automatically-generated names and IDs to tag the

Option	Description
	Endeca records the same as it would explicitly-defined dimension values. Auto Generate dramatically reduces the amount of editing you have to do to the dimension hierarchy. However, auto-generated dimensions are always flat. Auto-generated names and IDs are persisted in a file that you specify as part of a dimension server component.
New	Click to create a new mapping of one of the following types: • Null Mapping: Indicates that the source property should be ignored during mapping. • Property Mapping: Maps the source property to an Endeca property. • Dimension Mapping: Maps the source property to a dimension.
Modify	Used to modify an existing mapping. To modify a mapping: 1. Select it in the mappings table. 2. Click **Modify** . 3. Make your changes in the dialog box that appears. 4. Click **OK** .
Remove	Removes the selected mapping.

Member Options dialog box (snippeting)

You enable the snippeting feature in the Member Options dialog box. Each member of a search interface is enabled and configured separately.

In other words, snippeting results are enabled and configured for each member of a search interface and not for all members of a single search interface.

Note: A search interface member is a dimension or property that has been enabled for search and that has been added to the Selected members pane of the Search Interface editor.

You can enable and configure any number of individual search interface members. Each member that you enable produces its own snippet.

Enabling a member in one search interface does not affect that member if it appears in other search interfaces. For example, enabling the Description property for Search Interface A does not affect the Description property in Search Interface B.

Option	Description
Enable snippeting	Enables the snippeting feature for the selected search interface member only.
Max snippet size	Determines the size of the snippets returned for the search interface member.

New Project dialog box

You may create a project from a template. This dialog is also where you name and save your project.

Option	Description
From a project template	Select this option to create a new project based on an existing project template.
Select a project template	Appears when "From a project template" is selected. If you are creating a project based on an existing template, select the template from this list.
Description	Appears when "From a project template" is selected. Describes the current selected template in the "Select a project template" list.
Project name	The name of your project. This name appears in the Developer Studio title bar to indicate which project you are working on.
Save project as	The location and name for the .esp project file.

Null Mapping dialog box

Specify a source property that will be ignored during mapping.

To use the Null Mapping dialog box:

1. Enter the name of a source property to indicate that it should be ignored during mapping.
2. Click **OK.**

Properties dialog box

This shows properties associated with a dimension value. You may add, modify, or remove properties.

Option	Description
Properties list	A list of properties associated with this dimension value. **Note:** Do not confuse these name/value pairs with source properties or Endeca properties. They are purely for descriptive information about a given dimension value.
Property	Field for entering and editing a property's name.
Value	Field for entering and editing a property's value.
Add	Adds the property entered in the 'Property' and 'Value' fields to the Properties list.
Modify	Used to modify an existing property. To modify a property, select it in the Properties list, make your changes to the property name and/or value, and click Modify.
Remove	Removes the selected property from the Properties list.

Property dialog box

Specify a source property.

1. Enter the name of a source property.
2. Click OK.

Property Mapping dialog box

Specify the source property, target property, and maximum length of property value allowed during mapping.

Option	Description
Source property	Name of the source property to be mapped
Target property	Name of the Endeca property the source property will map to
Maximum length	Optional. Defines the maximum source property value length allowed when creating mappings. Source properties that have values that exceed this length are not mapped, and a warning is issued by the Forge Hierarchical Logging system. This value overrides the default maximum length that you set on the Advanced tab of the Property Mapper editor.

Import External Dimensions dialog box

Specify the external dimsensions file to import, and which dimension adapter will receive these dimensions.

Option	Description
External dimensions file	The name of the external dimensions file to be imported.
Dimension adapter to receive imported dimensions	All imported dimensions must be received by an existing dimension adapter. Choose the dimension adapter from this list.

Relevance Ranking Modules dialog box

The Relevance Ranking Modules editor allows you to associate ranking modules with a search interface.

Ranking modules are selected from a stock list of modules. Only the static module takes parameters; for this reason, it is the only module type for which you can have multiple instances. The Relevance Ranking Module editor contains the following fields:

Option	Description
All modules	A list of all of the ranking modules that are available. Selecting a module causes its full name and a brief description to appear in the frame in the lower left corner of the editor.
Selected modules	A list of all modules currently composing the ranking strategy for this search interface. You can add, remove, and reorder the modules in this list using the buttons on the editor. In addition, for Static modules, you can edit associated parameters.
Add	Adds the selected module, in the "All modules" list, to the "Selected modules" list.
Remove	Removes the selected module from the "Selected modules" list.
Edit	This option is only available when you have a Static module selected in the "Selected modules" list. Click it to edit the parameters associated with the selected module.
Up arrow	Moves the selected module, in the "Selected modules" list, up.

Option	Description
Down arrow	Moves the selected module, in the "Selected modules" list, down.

Search Interface Options dialog box

You have the option to customize partial match settings.

The Search Interface Options editor allows you to adjust various settings at the search interface level. The Search Interface Options editor contains the following fields:

Option	Description
Customize partial match settings	Specifies if partial query matches should be supported for this search interface. • "Match at least" sets the minimum number of words that must be matched. The default is 2. • "Omit at most" sets the maximum number of words that can be omitted. The default is 2.

Select Dimension Adapter dialog box

All promoted dimensions must be affiliated with a dimension adapter.

To use the Select Dimension Adapter dialog box:

1. Select a dimension adapter from the menu.
2. Click **OK**.

Select Dimension editor

To use the Select Dimension editor:

1. Select a dimension from the list.
2. Click **OK**.

Select Dimension or Property dialog box

Select a dimension or property from the list and click OK.

Select Dimension Value editor

1. Select a dimension value from the list.
2. Click OK.
   a) To expand a dimension value and expose its children, click the + sign next to the dimension value.

Note: Selecting a dimension name is equivalent to selecting the dimension's root.

Select Redirect Group dialog box

Describes available option for the Select Redirect Group dialog box.

Option	Description
Move the selected redirects in "redirectGroupName" to this redirect group	A drop-down list with the names of the other keyword redirect groups. When you delete a redirect group, its keyword redirects must be moved to another redirect group. Use this drop-down list to indicate which redirect group will receive the keyword redirects.

Select Rule Group dialog box

Describes available options for the Select Rule Group dialog box.

Option	Description
Move the rules in "ruleGroupName" to this rule group	A drop-down list with the names of the other rule groups. When you delete a rule group, its rules must be moved to another rule group. Use this drop-down list to indicate which rule group will receive the rules.

Specifying constraints on spelling dictionary content

Constraining spelling dictionary entries improves the performance of spelling corrected queries.

If you want to fine tune the size of the spelling dictionary and consequently tune the performance of spelling corrected queries, you can specify constraints to control what words Dgidx adds to the spelling dictionary. You can separately configure entries in the dictionary based for dimension search and record search.

To constrain spelling dictionary entries:

1. In the Project Explorer, expand **Search Configuration.**
2. Double-click **Spelling.**
3. Select the **Dimension Search** tab.
4. In the It Occurs at Least ... Times field, provide a number that indicates the minimum number of times the word must appear in your source data before the word should be included in the spelling dictionary.

5. In the And Is Between ... and ... Characters Long fields, provide values that represent the minimum and maximum length of a word that should be included in the spelling dictionary.
6. Select the **Record Search** tab.
7. Repeat steps 4 and 5.
8. Click **OK.**

Stop word editor

To create a stop word, enter it in the Stop Word field and click OK.

Synonyms dialog box

Define a synonym and its behaviors, or modify an existing synonym.

Option	Description
Synonyms list	A list of synonyms of the dimension value.
Synonym area	Use this area to define a synonym and its behaviors: • Synonym field: Use this field to enter and edit synonyms. • Search: Indicates that the selected synonym will be used during search. • Classify: Indicates that the selected synonym will be during classification.
Add	Adds the synonym entered in the 'Synonym' field to the Synonyms list.
Modify	Used to modify an existing synonym. To modify a synonym, select it in the Synonyms list, make your changes in the Synonym field, and click Modify.
(Display)	Sets the selected synonym in the Synonyms list to be used for display. Only one synonym can be set for display, and it is indicated by parentheses in the Synonyms list.
Remove	Removes the selected synonym from the Synonyms list.

URL Filter dialog box

You use this dialog box to specify the filters by which the spider includes or excludes URLs during a crawl.

Filters are expressed as wildcards or Perl regular expressions. URL filters are mutually exclusive; that is, URL filter A does not influence URL filter B and vice versa. At least one URL filter is required to allow the spider make additional processing loops over the root URL.

Option	Description
URL filter	Enter either a wildcard filter or regular expression filter. Filters can be specified either by using wildcard filters for example, *.endeca.com or Perl regular expressions, for example /.*\.html/i. Generally, you should use "Wildcard" patterns for "Host" filters and use "Regular expression" patterns for "URL" filters. **Note:** There are additional samples in the Crawler Implementations section of the *Data Foundry Expression Reference*.
Type	Select either Host or URL: • Host filters apply only to the host name portion of a URL. • URL filters are more flexible and can filter URLs based on whether the entire URL matches the specified pattern. For example, the spider may crawl a file system in which a directory named "presentations" contains PowerPoint documents that, for some reason, should not be crawled. They can be excluded using a URL exclusion filter with the pattern /.*\/presentations\/.*\.ppt/.
Action	Select either Include or Exclude: • Include indicates that the spider crawls documents that match the URL filter. • Exclude indicates that the spider excludes documents that match the URL filter. **Note:** A URL must pass both inclusion and exclusion filters for the spider to queue it. In other words, a URL must match at least one inclusion filter and a URL also must not match any exclusion filter.
Pattern	Select either Wildcard or Regular expression depending on the syntax of the filter you specified in the "URL filter" field.

User Profile editor

To use the User Profile editor:

1. Enter a name for the user profile.
2. Click **OK**.

Index