

# Oracle® Endeca Server

Administrator's Guide

Version 7.4.0 • June 2014 • Revision A

## Copyright and disclaimer

Copyright © 2003, 2014, Oracle and/or its affiliates. All rights reserved.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners. UNIX is a registered trademark of The Open Group.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

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

If this 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. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

# Table of Contents

<b>Copyright and disclaimer</b> .....	<b>2</b>
<b>Preface</b> .....	<b>7</b>
About this guide .....	7
Who should use this guide .....	7
Contacting Oracle Customer Support .....	7
<b>Chapter 1: Introduction</b> .....	<b>8</b>
Taking ownership of your implementation .....	8
Overview of administrator tasks .....	8
<b>Chapter 2: Oracle Endeca Server</b> .....	<b>10</b>
Overview of Oracle Endeca Server .....	10
Starting the Oracle Endeca Server .....	11
Endeca Server configuration .....	11
Increasing the maximum idle time .....	13
<b>Chapter 3: Oracle Endeca Server Commands</b> .....	<b>14</b>
Endeca Server operational model .....	14
Command global options .....	15
Global options for --host and --port .....	16
Global options for SSL .....	16
create-ds command .....	18
attach-ds command .....	19
detach-ds command .....	21
start-ds command .....	22
stop-ds command .....	22
status-ds command .....	23
list-ds command .....	24
list-jobs command .....	25
version command .....	25
<b>Chapter 4: Using the Administration Web Service</b> .....	<b>26</b>
About the Administration Web Service .....	26
Accessing the Administration Web Service .....	26
Using the Administration Web Service .....	27
<b>Chapter 5: Job Monitoring and Canceling</b> .....	<b>30</b>
About job monitoring .....	30
About jobs .....	30
Obtaining a list of running or pending jobs .....	31
Canceling pending jobs .....	32

<b>Chapter 6: Capturing Snapshots</b> .....	<b>34</b>
About snapshots .....	34
Restrictions for taking a snapshot .....	34
Creating a snapshot .....	35
Restoring data files from a snapshot .....	35
cpmdex syntax .....	36
<b>Chapter 7: Dgraph Administrative Tasks</b> .....	<b>38</b>
Checking the status of the Dgraph .....	38
About connecting Web browsers to the data store .....	38
Managing Dgraph core dump files .....	39
Managing Dgraph crash dump files on Windows .....	39
Managing Dgraph core dump files on Linux .....	39
Collecting debugging information .....	40
Logs created by the Dgraph .....	40
Running multiple Dgraphs on the same Windows machine .....	41
Identifying connection errors .....	41
<b>Chapter 8: Administrative Operations</b> .....	<b>42</b>
About administrative and configuration operations .....	42
List of administrative operations .....	43
help .....	44
flush .....	44
logroll .....	44
merge .....	44
ping .....	44
reload-services .....	44
stats .....	45
statsreset .....	45
updateaspell .....	45
About logging variables for the Dgraph .....	46
Logging variable operation syntax .....	46
List of configuration operations .....	47
List of supported logging variables .....	47
log-enable .....	48
log-disable .....	48
log-status .....	48
help .....	49
<b>Chapter 9: Configuring Endeca Server as a Service</b> .....	<b>50</b>
Running the Endeca Server as a Windows service .....	50
SC Create command syntax .....	50
Creating the Endeca Server Windows service .....	53
Setting a service description .....	53
Modifying the service configuration .....	54
Deleting the Endeca Server Windows service .....	55
Using the Windows Services utility .....	55

Logging in service mode . . . . .	57
Starting the Endeca Server from inittab . . . . .	57
<b>Chapter 10: Managing the Merge Policy . . . . .</b>	<b>58</b>
Using a merge policy for incremental updates . . . . .	58
Types of merge policies . . . . .	58
Setting or changing the merge policy . . . . .	59
Setting the merge policy with the Configuration Web Service . . . . .	59
Retrieving the merge policy with the Configuration Web Service . . . . .	59
Setting the merge policy with the Configuration Web Service . . . . .	60
Changing the merge policy of a running data store . . . . .	60
Forcing a merge . . . . .	61
<b>Chapter 11: Deploying a Cluster of Oracle Endeca Servers . . . . .</b>	<b>62</b>
Cluster overview . . . . .	62
Oracle Endeca Server cluster architecture . . . . .	64
Important cluster concepts . . . . .	66
Before you begin . . . . .	67
System and hardware requirements . . . . .	67
Operating system requirements . . . . .	67
Shared file system requirements . . . . .	67
Load balancer requirements . . . . .	68
About the Cluster Coordinator . . . . .	69
Starting and stopping the Cluster Coordinator service . . . . .	70
Configuration file for the Cluster Coordinator . . . . .	70
Planning cluster nodes . . . . .	72
Cluster behavior . . . . .	72
Building a cluster . . . . .	74
Starting the Dgraph process as the leader node . . . . .	74
Adding a follower node . . . . .	75
Summary of operations handled by the leader node and any node . . . . .	76
Connecting the leader node with Integrator . . . . .	77
Maintaining a cluster . . . . .	78
Removing a follower node . . . . .	78
Changing the name of the leader node . . . . .	78
<b>Chapter 12: Configuring SSL . . . . .</b>	<b>79</b>
About configuring SSL in Endeca Server . . . . .	79
Configuring SSL on Jetty . . . . .	80
Creating a keystore . . . . .	80
Obfuscating passwords . . . . .	82
Configuring Endeca Server Jetty files . . . . .	82
Enabling SSL for the endeca-cmd interface . . . . .	84
Configuring SSL on the Dgraph . . . . .	84
Certificate files used by components connecting with the Dgraph . . . . .	85
Using enecerts to generate SSL certificates . . . . .	85
Generating standard SSL certificates on Linux . . . . .	86

---

Generating standard SSL certificates on Windows .....	86
Generating custom certificates .....	86
Copying the SSL certificates to other machines .....	87
Configuring the Dgraph for SSL mutual authentication .....	87
Converting PEM-format keys to JKS format .....	89
<b>Appendix A: Dgraph Flag Reference .....</b>	<b>91</b>
Dgraph flags .....	91

## Preface

Oracle® Endeca Server is a hybrid search-analytical engine that organizes complex and varied data from disparate sources. At the core of Endeca Information Discovery, the unique NoSQL-like data model and in-memory architecture of the Endeca Server create an extremely agile framework for handling complex data combinations, eliminating the need for complex up-front modeling and offering extreme performance at scale. Endeca Server also supports 35 distinct languages.

## About this guide

This guide describes the administrative tasks for the Oracle Endeca Server.

## Who should use this guide

This guide is intended for system administrators who administer and maintain an Oracle Endeca Server implementation.

This guide assumes that the Oracle Endeca Server software is already installed on a development server. It may be already installed in a production environment. It also assumes that you, or your Oracle Services representatives, have already configured the application on the development server.

You can choose to read specific topics from this guide individually as needed while maintaining your Oracle Endeca Server implementation after it has been initially deployed.

## Contacting Oracle Customer Support

Oracle Endeca Customer Support provides registered users with important information regarding Oracle Endeca software, implementation questions, product and solution help, as well as overall news and updates.

You can contact Oracle Endeca Customer Support through Oracle's Support portal, My Oracle Support at <https://support.oracle.com>.



## Chapter 1

---

# Introduction

This section describes the stage at which you take control of the operation and maintenance of your application powered by the Oracle Endeca Server.

*Taking ownership of your implementation*

*Overview of administrator tasks*

## Taking ownership of your implementation

As a system administrator, you take ownership of the Oracle Endeca Server implementation at a certain stage. This topic describes the context in which you will perform administrative tasks to maintain the stable operation of a properly functioning implementation.

This guide assumes that by this point in using the Oracle Endeca Server, you or your team have done the following:

- Planned and provisioned the hardware needed for the staging and production environments.
- Installed the Oracle Endeca Server.
- Planned the user-facing details of your application, such as the attributes that will be displayed, the search interfaces, and so on.

In addition, the guide assumes that you have performed the following application-building tasks:

- You have completed the process of extracting source information from your incoming data sources.
- You have completed the process of loading your configuration schema and your source the data into the Oracle Endeca Server, thus creating the Oracle Endeca Server data files.
- You have created a working prototype of your front-end application for your end users. This front-end application can be used to issue requests to the running Oracle Endeca Server in a production environment.
- You have deployed your application powered by the Oracle Endeca Server in a staging environment, and are either preparing to deploy it in production, or have already deployed it in production.

## Overview of administrator tasks

This topic provides a brief overview of the administrator tasks described in this guide.

This guide assumes that you are performing administrator tasks on the Oracle Endeca Server. The types of task that are described in this guide are the following (as grouped by their section):

Chapter	Tasks
Using the Administration Web Service	Use the Administration Web Service for administrative tasks.
Job Monitoring	Obtain information about the jobs that are being currently processed by the Oracle Endeca Server.
Capturing Snapshots	Create snapshots of a running Oracle Endeca Server and use them as a part of your backup and archiving strategy.
Dgraph Administrative Tasks	<ul style="list-style-type: none"> <li>• Check the Dgraph process.</li> <li>• Manage Dgraph process core dump files.</li> <li>• Collect debugging information to help solve problems.</li> <li>• Troubleshoot Dgraph process socket and port errors.</li> <li>• Identify Dgraph process connection errors.</li> </ul>
Administrative Operations and Logging Variables	<ul style="list-style-type: none"> <li>• Flush the dynamic cache.</li> <li>• Force a query log roll.</li> <li>• Merge update generations and sets the system's merge policy.</li> <li>• Check the Dgraph process Statistics page.</li> <li>• Rebuild the aspell dictionary for spelling correction.</li> <li>• Modify the logging configuration for the Dgraph process.</li> </ul>
Managing the Merge Policy	<ul style="list-style-type: none"> <li>• Merge update generations.</li> <li>• Set and manage the merge policy for an Endeca data store.</li> </ul>
Deploying Oracle Endeca Server in a Clustered Environment	Set up and manage a cluster of Oracle Endeca Server nodes.
Using SSL Certificate Utilities	<ul style="list-style-type: none"> <li>• Generate standard and custom SSL certificate files to be used for SSL connections to the Oracle Endeca Server.</li> <li>• Convert PEM-format certificates to the standard Java KeyStore (JKS) format.</li> <li>• Configure the Oracle Endeca Server for SSL mutual authentication.</li> </ul>



## Chapter 2

# Oracle Endeca Server

---

This section provides an overview of the Oracle Endeca Server and describes some administration and configuration tasks.

[Overview of Oracle Endeca Server](#)

[Starting the Oracle Endeca Server](#)

[Endeca Server configuration](#)

[Increasing the maximum idle time](#)

## Overview of Oracle Endeca Server

Endeca Server is the control center for Endeca data stores.

Endeca Server uses a database-like operational model to manage the Endeca data stores running on the machine. You install one (and only one) Endeca Server on a Linux or Windows machine that will host your Endeca data stores.

When you first install Endeca Server, it will have no Endeca data stores. You then create named data stores (potentially more than one per Endeca Server). In the simplest case, no information is needed to create and start a data store, other than a name for the data store. In this simple case, Endeca Server will figure out which port to start the data store and which Dgraph configuration flags to use. In the more complex case, you can specify the ports and configuration flags to be used (such as how many threads) and the location of the data files for the newly-created data store.

Once an Endeca data store is created, you only need to use the name of the data store to manage it. You don't even need to know which port the data store is running on, as Endeca Server keeps track of that information (which it stores in its own database). This name-only reference makes it much easier to stop and start data stores, as well as perform other management operations.

Endeca Server has a set of commands with which you create and control the data stores. These commands are documented in the chapter [Oracle Endeca Server Commands on page 13](#).

### Data store monitoring

Endeca Server is responsible for keeping each Endeca data store up and running, and therefore constantly monitors each data store.

If an Endeca data store crashes, Endeca Server attempts to re-start it. If the re-start attempt is unsuccessful, Endeca Server does not try to re-start it again.

As an administrator, you should periodically check the data store logs to make sure the data stores are running properly. These logs are:

- `<dataStore>.out` (for example, `bikes.out`) is the stdout/stderr log for a specific Endeca data store.

- `<dataStore>.reqlog` is the request log for a specific Endeca data store.

By default, these logs are stored in the `endeca-server\logs` directory.

## About the Endeca data store and the Dgraph process

Each Endeca data store is serviced by a Dgraph process. The Dgraph uses proprietary data structures and algorithms that allow it to provide real-time responses to client requests. It stores the data files that were created from loading the data into it. After the data files are stored, the Dgraph receives client requests via the application tier, queries the data files, and returns the results.

The Dgraph is designed to be stateless. This design requires that a complete query be sent to it for each request. The stateless design facilitates the addition of Dgraph processes for load balancing and redundancy — any replica of a Dgraph can reply to queries independently of other replicas.

## Starting the Oracle Endeca Server

This topic describes how to start the Oracle Endeca Server.

The Endeca Server runs on a Jetty application server, which is provided as part of the installation package.

To start the Endeca Server on Windows, use the **Oracle Endeca Server 7.4.0>Start Oracle Endeca Server** option in the Start Menu.

To start the Endeca Server on Linux:

1. From a command prompt, navigate to the Endeca Server install root and then into the `endeca-server` directory.
2. Run the `start.sh` script.

In both cases, a command window is opened that is used for standard out by the Jetty process. By default, the Endeca Server will be running on port 7770 on the localhost machine.

To shut down the Endeca Server on Windows, use the **Oracle Endeca Server 7.4.0>Stop Oracle Endeca Server** option in the Start Menu.

To shut down the Endeca Server on Linux, hit Control-C in the standard out window and then enter `Y` at the **Terminate batch job** prompt.

## Endeca Server configuration

You can set the locations of the Endeca data store components and other settings in the configuration script.

The Endeca Server configuration script (`endeca-server.windows.conf.bat` on Windows or `endeca-server.linux.conf` on Linux) has settings that the Endeca Server will use for its start-up behavior, as well as the locations where certain Endeca data store artifacts are stored. These locations are used by the Endeca Server when Endeca data stores are created or modified, and when the data store's Endeca process is started.

The configuration script is located in the `$ROOT/endeca-server` directory. You can edit the script with any text-based editor.

The meanings of the configuration settings are as follows:

Configuration Setting	Meaning
ENDECA_SERVER_PORT	The port on which the Endeca Server will be listening for requests. The default is 7770.
ENDECA_SERVER_STOP_PORT	The port on which the Endeca Server will be listening for stop commands. The default is 9420.
ENDECA_DATASTORE_PORT_MIN and ENDECA_DATASTORE_PORT_MAX	The range of port numbers from which the Endeca Server selects the HTTP and bulk-load ports for the Dgraph processes. Once a port is assigned to a Dgraph process, that port is not used for subsequent Dgraph port assignments (unless the data store is detached). This port assignment strategy prevents port collisions among the Dgraph processes. Note that when you create an Endeca data store, you can use the <code>create-ds</code> command's <code>--ws-port&gt;</code> and <code>--bulk-load-port</code> flags to set a specific port number.
DATA_DIR	The location where the data files for the Endeca data stores are created. The default is the <code>\$ROOT/endecca-server/data</code> directory.
STATE_DIR	The location of the Endeca Server's state directory. The default is the <code>\$ROOT/endecca-server/state</code> directory.
LOGS_DIR	The location of the Dgraph's standard out/err log and request log, as well as the PID file. The default is the <code>\$ROOT/endecca-server/logs</code> directory.
DGRAPH_INSTALL	The location of the Dgraph application. Do not change this setting.
JETTY_INSTALL	The location of the Jetty installed files. Do not change this setting.

The following is an example of a Windows `endecca-server.windows.conf.bat` script with modified settings:

```
@echo off

set ROOT=%-dp0

set ENDECA_SERVER_PORT=6850
set ENDECA_SERVER_STOP_PORT=9255

REM port range where datastores can be spun up
set ENDECA_DATASTORE_PORT_MIN=8000
set ENDECA_DATASTORE_PORT_MAX=65535

set DATA_DIR=c:\endecca\apps\indexes
set STATE_DIR=%ROOT%\state
set LOGS_DIR=c:\endecca\apps\logs
set DGRAPH_INSTALL=%ROOT%\dgraph
set JETTY_INSTALL=%ROOT%\jetty-distribution-8.0.1.v20110908

REM Use our locally installed JRE, if it's present.
SET JAVA_CMD=java
IF EXIST %ROOT%\..\shared\jre SET JAVA_CMD=%ROOT%\..\shared\jre\bin\java
```

In this example, the Endeca Server now starts on port 6850 and stops on port 9255. The Dgraph port range begins at 8000. The data files for the Endeca data stores and the Dgraph logs are now stored in `C:\Endeca\Apps` sub-directories.

## Increasing the maximum idle time

The `maxIdleTime` configuration property sets the maximum Idle time (in milliseconds) for a connection.

The Oracle Endeca Server comprises two main parts: The Oracle Endeca Server (a servlet deployed in Jetty), and the Dgraph process (a separate binary). Most queries to the Oracle Endeca Server are passed on to the Dgraph process for evaluation.

The Jetty default setting for the `maxIdleTime` property is 5 minutes (300000 milliseconds). This time-out can be triggered if evaluation time in the Dgraph process exceeds 5 minutes. If any queries sent to the Endeca Server are expected to take more than 5 minutes to process, this time-out should be increased.

The new value for `maxIdleTime` should be greater than your maximum expected query latency. It is desirable to never time out at this level, falling back to configurable time-outs in the Dgraph process or the client. We recommend a setting of 86400000 milliseconds (1 day).

Also note that this mechanism times out idle connections in Jetty both when reading from or writing to a connection to the servlet - it is simply the maximum amount of time that Jetty will wait for any information to travel in either direction.

To increase the maximum idle time:

1. Stop the Oracle Endeca Server if it is running.
2. Open the `jetty.xml` configuration file.

The file is in the `%EndecaServerInstall%/endeca-server/jetty-distribution-8.0.1.v20110908/etc` directory.

3. In the `addConnector` configuration section, increase the value for the `maxIdleTime` property, as in this example:

```
<Set name="maxIdleTime">86400000</Set>
```

4. Save the file.
5. Restart Oracle Endeca Server.



## Chapter 3

# Oracle Endeca Server Commands

---

This section describes the commands of the Oracle Endeca Server's `endeca-cmd` command-line interface.

[Endeca Server operational model](#)

[Command global options](#)

[create-ds command](#)

[attach-ds command](#)

[detach-ds command](#)

[start-ds command](#)

[stop-ds command](#)

[status-ds command](#)

[list-ds command](#)

[list-jobs command](#)

[version command](#)

## Endeca Server operational model

The Endeca Server has a command-line interface that lets you control Endeca data stores.

With the Endeca Server commands, you can perform the following tasks:

Endeca Server command	Purpose
<code>create-ds</code>	Creates a named Endeca data store. The command also attaches and starts the data store.
<code>attach-ds</code>	Attaches an existing but detached Endeca data store. This operation in effect registers the Endeca data store with the Endeca Server. The command also starts the data store.
<code>detach-ds</code>	Detaches an attached Endeca data store. The data store must be stopped before it can be detached. Although the Endeca data store's index files are not deleted from the file system, the data store is de-registered from the Endeca Server, so that subsequent commands (except for <code>attach-ds</code> ) will have no effect on the data store.

Endeca Server command	Purpose
<code>start-ds</code>	Starts an attached but stopped Endeca data store.
<code>stop-ds</code>	Stops a started (running) Endeca data store.
<code>status-ds</code>	Shows the status of an attached Endeca data store. The status information includes its assigned ports, as well as the Dgraph arguments (if any) that were specified when the data store was created or attached.
<code>list-ds</code>	Lists all the Endeca data stores that are attached to this Endeca Server.
<code>list-jobs</code>	Lists the running or queued jobs on a specific Endeca data store.
<code>version</code>	Lists the version of the Oracle Endeca Server and the versions of the Dgraph processes powering each of the data stores (if the Dgraph processes for these data stores are currently running).  You do not need to specify the name of the data stores when using this command.

All of the operations that control data stores are also programmatically available in the Endeca Server's Control Web Service. The `list-jobs` and the `version` operations are available as `listJobsOperation` and `GetVersionOperation` in the Administration Web Service.

## endeca-cmd directory

The `endeca-cmd` directory contains the `endeca-cmd` script for running the Endeca Server commands. To run the commands, you use a command-line window (for example, open a Command Prompt in Windows) and navigate to the `endeca-cmd` directory, or run `/<path>/endeca-cmd`.

## Command global options

The Endeca Server command interface has several global options that let you specify a host and port of the Oracle Endeca Server, and enable the interface for SSL.

The global options are:

- `--host`
- `--port`
- `--help`
- `--keystore`
- `--truststore`
- `--ssl`

## Getting online help

The `--help` option provides usage help for the Endeca Server commands.

The syntax for obtaining general help is:

```
endeca-cmd --help
```

The syntax for obtaining help on a specific command is:

```
endeca-cmd <command> --help
```

This example will display usage help for the `create-ds` command:

```
endeca-cmd create-ds --help
```

## Global options for `--host` and `--port`

Two global options let you specify host name and port information for the Endeca Server.

### `--host` option

You use the `--host` option when you want to run a command on an Endeca Server that is running on a remote machine. The `--host` argument can be either the full name of the remote machine or its IP address.

The following example illustrates the `--host` global option:

```
endeca-cmd create-ds bikes --host web7.example.com --args --threads 6
```

The command tells the Endeca Server running on the **web7.example.com** remote machine (and listening on its default port 7770) to create an Endeca data store named **bikes** in the default location on that remote machine. All default configuration values (such as the Dgraph's base port and bulk load ports) are assigned by the Endeca Server on the **web7.example.com** remote machine. The `--args` flag specifies that the data store's Dgraph process be started with 6 threads.

### `--port` option

The `--port` option is used whenever the Endeca Server is not running on its default port 7770, regardless of whether the Endeca Server is running locally or on a remote machine. If you do not specify `--port`, the default port of 7770 is used for the command.

The following example illustrates both global options:

```
endeca-cmd status-ds bikes --host web7.example.com --port 9090
```

The command tells the Endeca Server running on the **web7.example.com** remote machine (and listening on a non-default port 9090) to return the status of the Endeca data store named **bikes**.

## Global options for SSL

The `--ssl`, `--keystore` and `--truststore` options are used to support SSL-enabled communications with an Oracle Endeca Server running over SSL.

You are required to use these options if you have enabled the Oracle Endeca Server to run only over SSL (generally, the server can have two ports, one for SSL and one for non-SSL communication).

Before using these options, you need to create a keystore or truststore file to use. For information, see [Creating a keystore on page 80](#).

The following global options in `endeca-cmd` are provided to enable SSL support:

Option	Description
<code>--keystore</code>	<p>Specifies the location of a keystore file needed for authentication to the Oracle Endeca Server.</p> <p>If you use this option (or the <code>--truststore</code> option), it implies that SSL should be used for communication between <code>endeca-cmd</code> and the Oracle Endeca Server.</p> <p>This means that you don't need to use the <code>--ssl</code> option.</p> <p> <b>Note:</b> If you specify a keystore, this causes <code>endeca-cmd</code> to prompt for a password. Therefore, if you use the <code>--keystore</code> option, you cannot run <code>endeca-cmd</code> as part of a script.</p>
<code>--truststore</code>	<p>Specifies the location of a truststore file needed for verifying the authenticated connection to the Oracle Endeca Server.</p> <p>If you use this option (or the <code>--keystore</code> option), it implies that SSL should be used for communication between <code>endeca-cmd</code> and the Oracle Endeca Server.</p> <p>This means that you don't need to use the <code>--ssl</code> option.</p> <p> <b>Note:</b> If you specify a truststore, this causes <code>endeca-cmd</code> to prompt for a password. Therefore, if you use the <code>--truststore</code> option, you cannot run <code>endeca-cmd</code> as part of a script.</p>
<code>--ssl</code>	<p>Specifies whether to use an authenticated SSL connection to the Oracle Endeca Server.</p> <p>If you use either the <code>--keystore</code> or <code>--truststore</code> option, then you don't need to use the <code>--ssl</code>— the authenticated SSL connection is implied by specifying the keystore or the truststore file.</p> <p>If you use <code>--ssl</code> without either a <code>--keystore</code> or <code>--truststore</code> option, you will be able to use an empty keystore and the <code>cacerts</code> file as your truststore. Alternatively, you can use the standard approaches for creating keystore and truststore files in Java — the <code>keytool</code> or <code>cacerts</code> file, both of which are included in the Oracle Endeca Server installation.</p>

As an alternative to using the SSL options interactively, you can specify the values for `Djavax.net.ssl.keyStore` and `Djavax.net.ssl.trustStore` properties to `endeca-cmd`, by including them in the `ENDECA_CMD_OPTS` environment variable, as follows:

```
export ENDECA_CMD_OPTS="-Djavax.net.ssl.keyStore=path_to_my_keystore_file
-Djavax.net.ssl.trustStore=path_to_my_truststore_file"
```

## create-ds command

The `create-ds` command creates an Endeca data store, attaches it, and starts it.

This command performs three actions:

1. Creates an Endeca data store with the specified name.
2. Attaches the Endeca data store (i.e., executes the `attach-ds` command).
3. Starts the Endeca data store (i.e., executes the `start-ds` command).

If the create operation fails at any point, the newly-created data files are deleted.

The syntax for this command is:

```
endeca-cmd create-ds <datastore-name> [global-options] [create-options]
```

where *datastore-name* is mandatory and is the name of the new Endeca data store. The name must be unique among any other Endeca data stores on this node.

The following additional command options can be used to change the created data store's configuration (defaults will be used otherwise):

Create Option	Meaning
<code>--ws-port &lt;num&gt;</code>	Specifies the Web service port number to be used for the Endeca data store's Dgraph process. The port number cannot be in use by any other process on the machine (including another Endeca data store). Defaults to a port number in the default range specified on the Endeca Server.
<code>--bulk-load-port &lt;num&gt;</code>	Specifies the port number for bulk load ingest operations for the Endeca data store's Dgraph process. This port number must be different from the port specified by the <code>--ws-port</code> flag and cannot be in use by any other process on the machine. If the <code>--bulk-load-port</code> flag is not used, then the bulk load port number defaults to a port number in the default range specified on the Endeca Server.
<code>--data-files &lt;path&gt;</code>	Specifies the absolute path of the Endeca data store's indexes directory to be created. The default is to create the indexes directory in the location configured in the Endeca Server.
<code>--startup-timeout &lt;seconds&gt;</code>	Specifies the maximum length of time (in seconds) that is allowed for the Endeca data store's Dgraph process to start up. Default is 60 seconds.
<code>--shutdown-timeout &lt;seconds&gt;</code>	Specifies the maximum length of time (in seconds) that is allowed for the Endeca data store's Dgraph process to shut down. Default is 60 seconds.
<code>--args --usage</code>	Provides a list of the Dgraph process flags.

Create Option	Meaning
<code>--args &lt;dgraph-flags&gt;</code>	Specifies a list of the Dgraph flags that will be used for the Endeca data store's Dgraph process. The <code>--args</code> flag must be the last flag on the command line as all of its arguments are passed on to the Dgraph process.

## create-ds examples

Example 1:

```
endeca-cmd create-ds adventureworks
```

creates an Endeca data store named **adventureworks** in the default location. The port and bulk load ports for the Endeca data store's Dgraph process are assigned by the Endeca Server, and both the startup and shutdown timeout settings are 60 seconds. This is the simplest use of the command as it uses no configuration flags.

Example 2:

```
endeca-cmd create-ds warranty --ws-port 6060 --bulk-load-port 6061 --shutdown-timeout 90
```

creates a **warranty** data store in the default location. The database port is 6060 and the bulk load port is 6061. The startup timeout is the default 60 seconds while shutdown timeout setting is 90 seconds.

Example 3:

```
endeca-cmd create-ds wine --data-files c:\endeca\apps\wine
```

creates a data store named **wine** in the `C:\Endeca\Apps` directory (the name of the indexes directory is `C:\Endeca\Apps\wine_indexes`). The Endeca data store's Dgraph process uses the configuration defaults.

Example 4:

```
endeca-cmd create-ds books --args --threads 6 --net-timeout 60
```

creates a data store named **books** in the default location. The Dgraph's port and bulk load port numbers are assigned by the Endeca Server, and both the startup and shutdown timeout settings are 60 seconds. The `--args` flag specifies that the Dgraph process be started with 6 threads and a network timeout value of 60 seconds.

## attach-ds command

The `attach-ds` command attaches and starts an Endeca data store.

Running the `attach-ds` command makes the Endeca Server know about an Endeca data store and its parameters. That is, the Endeca data store is registered with the Endeca Server.

The main pre-requisite for this command is that the data files for the specified Endeca data store must exist. This means that:

- You can run this command to re-attach a detached Endeca data store (that is, a data store against which the `detach-ds` command has been run). In this use case, the detached Endeca data store's data files are re-used.

- You can run this command to create a new Endeca data store by pointing (via the `--data-files` option) to the data files of another Endeca data store. In this use case, both Endeca data stores are using the same data files. For example, if you are using the Endeca Clustering feature, you would use this method to create a follower node that is using the same data files as the leader node.

In both use cases, the Endeca data store's Dgraph process is started, thus making the data store open for use.

The syntax for the `attach-ds` command is:

```
endeca-cmd attach-ds <datastore-name> [global-options] [attach-options]
```

where *datastore-name* is the name of an existing, detached Endeca data store.

The *<attach-options>* are optional, and defaults will be used for omitted options.

Attach Option	Meaning
<code>--ws-port &lt;num&gt;</code>	Specifies the Web service port number to be used for the Endeca data store's Dgraph process. The port number cannot be in use by any other process on the machine (including another Endeca data store). Defaults to a port number in the default range specified on the Endeca Server.
<code>--bulk-load-port &lt;num&gt;</code>	Specifies the port number for bulk load ingest operations for the Endeca data store's Dgraph process. This port number must be different from the port specified by the <code>--ws-port</code> flag and cannot be in use by any other process on the machine. If the <code>--bulk-load-port</code> flag is not used, then the bulk load port number defaults to a port number in the default range specified on the Endeca Server.
<code>--data-files &lt;path&gt;</code>	Specifies the absolute path of an existing data files directory to use. Use this option for: <ul style="list-style-type: none"> <li>A detached data store whose data files are not in default location configured in the Endeca Server.</li> <li>A new data store that will use the data files from another data store.</li> </ul>
<code>--startup-timeout &lt;seconds&gt;</code>	Specifies the maximum length of time (in seconds) that is allowed for the Endeca data store's Dgraph process to start up. Default is 60 seconds.
<code>--shutdown-timeout &lt;seconds&gt;</code>	Specifies the maximum length of time (in seconds) that is allowed for the Endeca data store's Dgraph process to shut down. Default is 60 seconds.
<code>--args --usage</code>	Provides a list of the Dgraph process flags.

Attach Option	Meaning
<code>--args &lt;dgraph-flags&gt;</code>	Specifies a list of Dgraph flags that will be used for the Endeca data store's Dgraph process. The <code>--args</code> flag must be the last flag on the command line as all of its arguments are passed on to the Dgraph process.

## attach-ds errors

If *datastore-name* does not exist, the Endeca Server returns an error message similar to this example:

```
Data files don't exist: C:\Endeca\Apps\bikes_indexes
```

This error can also occur if the Endeca data store does exist but is in a non-default location. In this case, use the `--data-files` flag, as in the example below.

## attach-ds examples

Example 1:

```
endeca-cmd attach-ds books
```

attaches an Endeca data store named **books** in the default location and starts it. The Endeca data store's Dgraph port and bulk load port numbers are assigned by the Endeca Server, and both the startup and shutdown timeout settings are 60 seconds. This is the simplest use case because it uses no attach flags.

Example 2:

```
endeca-cmd attach-ds wine --data-files c:\myapps\wine --args --threads 6
```

attaches an Endeca data store named **wine** in the `C:\MyApps` directory. The command also uses the `--args` flag to set the thread count at 6.

## detach-ds command

The `detach-ds` command detaches an attached Endeca data store.

The syntax for the `detach-ds` command is:

```
endeca-cmd detach-ds <datastore-name> [global-options]
```

where *datastore-name* is the name of an existing, attached Endeca data store that is not running.

Note the following about the `detach-ds` command:

- A started data store must be stopped before it can be detached.
- The command removes any configuration settings that were used when the Endeca data store was created (for example, with the `--args` flag). This means that if you subsequently use the `attach-ds` command, you must re-specify those configuration settings or else the data store will be re-attached with the default settings.
- The command does not remove the data store's indexes directory or any of its data. This means that you do not have to use the `create-ds` command to re-use the index data files, but you do have to use the `attach-ds` command to re-attach the data store to the Endeca Server.

## detach-ds errors

If *datastore-name* is started, the Endeca Server returns an error message similar to this example:

```
DataStore 'books' cannot be detached while it is running.
```

If *datastore-name* is already detached or does not exist, the Endeca Server returns an error message similar to this example:

```
Could not find the Endeca data store 'books'
```

## detach-ds example

This command detaches the **books** data store:

```
endeca-cmd detach-ds books
```

## start-ds command

The `start-ds` command starts an Endeca data store.

The Endeca data store to be started must be attached and must be stopped.

The syntax for the `start-ds` command is:

```
endeca-cmd start-ds <datastore-name> [global-options]
```

where *datastore-name* is the name of an attached data store that is stopped.

The data store's Dgraph process is started on the ports set by the `--ws-port` and/or `--bulk-load-port` flags (of the `start-ds` or `attach-ds` command) or by the Endeca Server. The process is also started with the Dgraph flags that were specified with the `--args` flag of the `create-ds` or `attach-ds` command.

## start-ds errors

If *datastore-name* is already started, the Endeca Server returns this error message:

```
Cannot perform this operation while the data store is STARTED
```

If *datastore-name* is detached or does not exist, the Endeca Server returns an error message similar to this example:

```
Could not find the Endeca data store 'bookes'
```

## start-ds example

This command starts the **books** data store:

```
endeca-cmd start-ds books
```

## stop-ds command

The `stop-ds` command stops a started Endeca data store.

The syntax for the `stop-ds` command is:

```
endeca-cmd stop-ds <datastore-name> [global-options]
```

where *datastore-name* is the name of a started Endeca data store.

## stop-ds errors

If *datastore-name* is already stopped, the Endeca Server returns this error message:

```
Cannot perform this operation while the Endeca data store is STOPPED
```

If *datastore-name* is detached or does not exist, the Endeca Server returns an error message similar to this example:

```
Could not find the Endeca data store 'bookes'
```

## stop-ds example

This command stops the **books** data store:

```
endeca-cmd stop-ds books
```

## status-ds command

The `status-ds` command shows the status of an Endeca data store.

The syntax for the `status-ds` command is:

```
endeca-cmd status-ds <datastore-name> [global-options]
```

where *datastore-name* is the name of an existing, attached Endeca data store. The data store may be started or stopped.

The returned status consists of the following information:

Status field	Meaning
Current State	The Endeca data store is <b>Started</b> (and is therefore available for use) or <b>Stopped</b> (and is not available for use).
Data Files	The absolute path of the Endeca data store's indexes directory on disk.
WS Port	The data store's Dgraph port number, as set by the user (via the <code>--ws-port</code> flag of the <code>start-ds</code> or <code>attach-ds</code> command) or by the Endeca Server.
Bulk Load Port	The data store's Dgraph bulk load port number, as set by the user (via the <code>--bulk-load-port</code> flag of the <code>start-ds</code> or <code>attach-ds</code> command) or by the Endeca Server.
Startup Timeout	The maximum length of time (in seconds) that is allowed for the data store's Dgraph process to start up, as set by the user (via the <code>--startup-timeout</code> flag of the <code>start-ds</code> or <code>attach-ds</code> command) or by the Endeca Server.

Status field	Meaning
Shutdown Timeout	The maximum length of time (in seconds) that is allowed for the data store's Dgraph process to shut down, as set by the user (via the <code>--shutdown-timeout</code> flag of the <code>start-ds</code> or <code>attach-ds</code> command) or by the Endeca Server.
Command Line Arguments	The Dgraph flags specified by the <code>--args</code> flag of the <code>start-ds</code> or <code>attach-ds</code> command. This field is not displayed if the <code>--args</code> flag was not used.

## status-ds example

This command:

```
endeca-cmd status-ds books
```

returns this status for the **books** data store:

```
Current State: Started
Data Files: C:\Endeca\Apps\books
WS Port: 5558
Bulk Load Port: 5559
Startup Timeout (s): 60
Shutdown Timeout (s): 90
Command Line Arguments: --threads 6 --net-timeout 60
```

## list-ds command

The `list-ds` command lists all the Endeca data stores that are attached to this Endeca Server.

The syntax for the `list-ds` command is:

```
endeca-cmd list-ds [global-options]
```

An alphabetical list of all attached Endeca data stores is returned.

## list-ds example

This command:

```
endeca-cmd list-ds
```

might return this list:

```
adventureworks
books
warranty
wine
```

The list shows that four Endeca data stores have been created and are attached. Note that the list does not indicate whether a given data store is started or stopped.

## list-jobs command

The `list-jobs` command lists the jobs on a specific Endeca data store.

The jobs, if any, can be queued or being processed.

The syntax for the `list-jobs` command is:

```
endeca-cmd list-jobs <datastore-name> [global-options]
```

where *datastore-name* is the name of a started Endeca data store.

For more information on jobs, see [Job Monitoring and Canceling on page 29](#).

## version command

The `version` command lists the version of the Oracle Endeca Server and the versions of the Dgraph processes powering each of the data stores (if the Dgraph processes for these data stores are currently running).

The syntax for the `version` command is:

```
endeca-cmd version [global-options]
```

If no data stores are currently running on the Oracle Endeca Server, only the version of the Oracle Endeca Server is listed, because in this case versions of the Dgraph processes on the data stores are not available. The following examples illustrate the usage of the `version` command.

### Examples

In the first example, assume that two Endeca data stores, `datastore_A` and `datastore_B`, exist on the Oracle Endeca Server, where `datastore_B` is stopped (thus, its Dgraph process is not running).

The output of the `endeca-cmd version` command will be similar to the following:

```
Oracle Endeca Server 7.4.0.701
Data store information:
  datastore_A: version 7.4.0, build 7.4.0.665981
  datastore_B: -- No version available --
```

In the second example, assume that the Oracle Endeca Server is running but no data stores are running on it (they could either be stopped or not created). In this case, the output of the `endeca-cmd version` command will be similar to the following:

```
Oracle Endeca Server 7.4.0.701
-- No version information available for data stores --
```



## Chapter 4

# Using the Administration Web Service

---

This section describes how to use the Administration Web Service with the Oracle Endeca Server.

[About the Administration Web Service](#)

[Accessing the Administration Web Service](#)

[Using the Administration Web Service](#)

## About the Administration Web Service

The Administration Web Service enables IT engineers to administer and maintain the Endeca data stores.

## Accessing the Administration Web Service

The Administration Web Service is declared in `admin.wsdl`.

You can access the Administration Web Service at the following URL:

```
http://localhost:<port>/ws/admin/DataStore?wsdl
```

where the `localhost` and `port` are the host and port of the running Oracle Endeca Server, and `DataStore` is the name of the Endeca data store.

The namespace for the Administration Web Service is similar to the following example and reflects the version of the service:

```
http://www.endeca.com/MDEX/admin/1/0
```

This namespace is included in the WSDL document for the web service.

In this example, the string `1/0` indicates the version is 1.0, where 1 is the major version, and 0 is the minor version.



**Note:** The version in the service that you have installed may not match this example.

Changes to minor versions are backward-compatible. If any backward-compatible versions exist, additional namespaces are included in the WSDL, listing them. You can use any backward-compatible minor version that is listed. For example, if both 1.0 and 1.1 versions are listed in the WSDL, you can use either of them.

Changes to major versions are not backward-compatible, thus previous major versions are not listed in the WSDL namespaces.



**Important:** After you upgrade the Oracle Endeca Server, verify the versions of the web services you have been using against the installed versions, to avoid version mismatch. It is recommended to use the web service versions that match the ones installed with the Oracle Endeca Server.

In particular:

- If the minor version of the web service on your client does not match the version installed with the Oracle Endeca Server, you can still use this version if it is listed in the WSDL namespaces, although it is recommended to upgrade.
- If the major version of the web service on your client does not match the version of the web service installed with the Oracle Endeca Server, you must upgrade to the most recent major version of the web service (this may include upgrading client code to use client stubs generated from the most recent versions).

For more information on web service versions, see the *Oracle Endeca Server Developer's Guide*.

## Using the Administration Web Service

The Administration Web Service contains administrative operations for creating a snapshot, listing running jobs and canceling jobs that have been scheduled but have not started in a specific data store.

For example, `createSnapshotOperation($name, $path)` creates a snapshot of the Oracle Endeca Server's data store state as a tree of hard links under `$name` in directory `$path`.

### Operation description

The Administration Web Service takes as its input parameters to the functions it contains and performs the requested operations.

### Request

The input to the Administration Web Service depends on the function. For example:

- To create the data store snapshot, specify a name and a directory path to the snapshot file.
- To list or cancel jobs, use the operations for listing or canceling jobs.

### Response

The Administration Web Service returns:

- An `<operation successful>` response element if there are no problems.
- A `<fault>` element if an exception was thrown internally.

## Operations

The Administration Web Service contains the following operations:

Operation	Description
createSnapshotOperation	Create a snapshot representing a consistent view of the state of the data store at a specific point in time. As an argument, specify the name for a snapshot, such as <code>NewSnapshot</code> , and an absolute path to the snapshot directory in the URI format, such as <code>file:///mydirectory/home/snapshots/</code> .
getVersionOperation	List the version of the Oracle Endeca Server and the versions of the Dgraph processes powering each of the data stores (if the Dgraph processes for these data stores are currently running). The data store should not be specified.  In addition to using this operation directly in a request, you can also obtain the version by issuing a <code>version</code> command on the Oracle Endeca Server. For details on this command, see <a href="#">version command on page 25</a> .
listJobsOperation	List the jobs that are currently running in a specific data store, such as queries, updating operations or administrative services.  In addition to using this operation directly in a request, you can also obtain the results of this operation by issuing a <code>list-jobs &lt;datastore-name&gt;</code> command on the Oracle Endeca Server. For details on this command, see <a href="#">list-jobs command on page 25</a> .
cancelJobOperation	Cancel a job that has been scheduled to run but has not been started, by specifying a job ID obtained from the <code>listJobsOperation</code> request.

### Example

The following examples show the Administration Web Service request and response bodies for creating a snapshot.

To access the Administration Web Service, send a SOAP request to the following URL:

```
http://localhost:<port>/ws/admin/datastore?wsdl
```

specifying the host and port of the Oracle Endeca Server and the data store that has been created and is running on it.

This example shows the Post body of the Administration Web Service request that creates a snapshot:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:admin="http://www.endeca.com/MDEX/admin/1/0">
  <soapenv:Header/>
```

```
<soapenv:Body>
  <admin:request>
    <admin:createSnapshotOperation path=
"file:///mydirectory/home/snapshots/" name="NewSnapshot" />
  </admin:request>
</soapenv:Body>
</soapenv:Envelope>
```

This example shows the response body of the Administration Web Service request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <admin:response xmlns:admin="http://www.endeca.com/MDEX/admin/1/0">
      <admin:createSnapshotSuccess/>
    </admin:response>
  </soapenv:Body>
</soapenv:Envelope>
```



**Note:** For more information about the operations used in the Administration Web Service, see the Administration API section of the *Oracle Endeca Server API Reference*.



## Chapter 5

---

# Job Monitoring and Canceling

This section describes how you can monitor running and pending jobs, and cancel pending jobs in the data store, using the Administration Web Service.

[About job monitoring](#)

[About jobs](#)

[Obtaining a list of running or pending jobs](#)

[Canceling pending jobs](#)

## About job monitoring

In many instances, it is useful to have more information about the jobs that are scheduled to be processed for a specific data store but have not yet started to run, as well as obtain a list of jobs that are currently being processed.

When administering a data store, it is useful to know more about which jobs are running in the data store in the following scenarios:

- An administrator of the application sends a query to the Oracle Endeca Server data store and the server becomes unresponsive because it is already running a long-running query. The administrator can make a request to see when the Dgraph process for the data store had started processing the query.
- An administrator of the application would like to send an update and needs to verify whether any other updates are already being processed, or are waiting in the queue, before submitting a new update. Making an Administration Web Service request allows the administrator to understand whether a new update will begin processing immediately.
- An administrator of the application wants to check which updates have been submitted recently.
- An administrator wants to cancel a query that has been sent to the Oracle Endeca Server but has not yet started running (is pending in the queue).

## About jobs

You can monitor or cancel several types of jobs.

In this release, you can:

- Monitor a query, an update, or an administrative operation.
- Cancel a query or an update operation, if they have been scheduled but have not started.

Using the Administrative Web Service, you cannot monitor or cancel the following jobs in the data store:

- An administrative operation that was scheduled to run.
- Bulk ingest operations for adding data. (However, you can use the logging in Integrator to monitor the progress of bulk ingest operations).
- Incremental additions to the data files that occur during updates. These additions occur in the data files whenever an update is being processed; they are internal Dgraph process operations.

## Obtaining a list of running or pending jobs

Using the `listJobsOperation` of the Administration Web Service, you can make a job monitoring request for a list of jobs that are currently running or pending in the queue.

Depending on your environment, you can run a single server, or a cluster. If you are running a cluster, it is important to remember that only the leader node runs updating commands, while all nodes in the cluster process query requests. Thus, in a cluster, the list of jobs returned by the Administrative Web Service depends on which data store (and node) you issue the `listJobsOperation`. For example, the leader node may return a list of jobs that include an update, while follower nodes will only return a list of query processing jobs.

To obtain a list of jobs that are running or pending in the queue in the data store:

1. In a tool such as SOAP UI, access the web service for the specified data store:

```
http://localhost:<port>/ws/admin/dataStore?wsdl
```

where the host and port represent the Oracle Endeca Server on which the specified data store has been created and is running.

2. Specify the `listJobsOperation` to the Administration Web Service, as in the following example:

```
<admin:request xmlns:admin="http://www.endeca.com/MDEX/admin/1/0">
  <admin:listJobsOperation />
</admin:request>
```

The response contains a list of currently running or pending jobs, and includes the following information:

Item	Description
jobId	An internal ID assigned by the Oracle Endeca Server.
jobType	Indicates the type of job that is being monitored. It can be Admin, Query, or Update.
queuedTime	Indicates the time at which the Dgraph process had scheduled to run the job. If the job is already running, this time is in the past. If the job is pending in the queue, this time is in the future.
isRunning	Indicates whether the job is already running. The values are true or false.

Item	Description
isCanceled	<p>Indicates whether the job is canceled.</p> <p>If the job is pending in the queue but has not yet started, you can cancel it. In other words, if the value of <code>isCanceled</code> is <code>false</code>, and the value of <code>isRunning</code> is also <code>false</code>, you can cancel the job.</p>

### listJobsOperation example

In this example of the Administration Web Service response, you can see that a query with the job ID 7 is currently running. You can see its job ID, the ID of the outer transaction within which this job is running, the job type (query), and queued time. You can also see that the job is running (which means you cannot cancel it).

```
<admin:response xmlns:admin="http://www.endeca.com/MDEX/admin/1/0">
  <admin:jobs>
    <admin:job jobId="7">
      <admin:jobType>Query</admin:jobType>
      <admin:OuterTransactionId>123</admin:OuterTransactionId>
      <admin:queuedTime>2012-01-15T01:36:40.694Z</admin:queuedTime>
      <admin:isRunning>true</admin:isRunning>
      <admin:isCanceled>false</admin:isCanceled>
    </admin:job>
  </admin:jobs>
</admin:response>
```

You can use the Endeca Server `list-jobs` command as an alternative to the `listJobsOperation` of the Administration Web Service. For details on this command, see the topic [list-jobs command on page 25](#).

## Canceling pending jobs

To cancel a job that has been scheduled but has not yet run, obtain a list of jobs that are either running or waiting in the queue and then issue the `cancelJobOperation` of the Administration Web Service.

You can only cancel a job that has not yet started.

To cancel a job for a data store:

1. In a tool such as SOAP UI, access the web service for the specified data store:

```
http://localhost:<port>/ws/admin/DataStore?wsdl
```

where the host and port represent the Oracle Endeca Server on which the specified data store has been created and is running.

2. Specify the `listJobsOperation` to the Administration Web Service, as in the following example:

```
<admin:request xmlns:admin="http://www.endeca.com/MDEX/admin/1/0">
  <admin:listJobsOperation/>
</admin:request>
```

The response contains a list of jobs that are running and jobs that are pending in the queue but have not started yet.

Pending jobs that you can cancel are those for which `isRunning` is set to `false` and `isCanceled` is also set to `false`.

3. Identify the job ID of the pending job you want to cancel, and issue the `cancelJobOperation`, as in the following example:

```
<admin:request xmlns:admin="http://www.endeca.com/MDEX/admin/1/0">
  <admin:cancelJobOperation jobId="5"/>
</admin:request>
```

If the job ID that you provided is valid and the job has not yet started, it will be canceled. If the job has already been started, the response indicates that the job is already running.



**Note:** If a job has been canceled in the data store, Studio will receive an HTTP 200 code (request successful), but the end user of the front-end application powered by Studio will see empty content.

### Example

In this example of the Administration Web Service response, you can see that the job with the job ID 5 was canceled successfully:

```
<admin:response xmlns:admin="http://www.endeca.com/MDEX/admin/1/0">
  <admin:cancelJobStatus>
    <admin:jobId>5</admin:jobId>
    <admin:canceledJob>true</admin:canceledJob>
    <admin:canceledJobDetails>Job marked for cancellation</admin:canceledJobDetails>
  </admin:cancelJobStatus>
</admin:response>
```



This section describes how you can create snapshots of a running Dgraph process for a specific data store and use them as a part of your backup and archiving strategy.

[About snapshots](#)

[Restrictions for taking a snapshot](#)

[Creating a snapshot](#)

[Restoring data files from a snapshot](#)

[cpmdex syntax](#)

## About snapshots

A snapshot represents a consistent view of the state of the data files at a specific point in time. By taking a snapshot, you can capture the state of the data files without shutting down the Dgraph process for a particular data store.

When you create a snapshot, the specified Endeca data store identifies the set of files that comprise a version of the data files, and captures the state of the system as it exists at that moment.

A backup operation without taking the snapshot would involve the need to stop the Dgraph process and copy its data files, which can take a long time. In contrast, you can create a snapshot while the Endeca data store is active and the Dgraph process is handling updates and queries, without downtime. After a snapshot is complete, you can plan and create a backup at your convenience.

A snapshot contains all the files needed to restore the data store to a specific state.

To create a snapshot, you issue a request to the Dgraph process for that particular data store through `createSnapshotOperation` in the Administration Web Service.

After you take the snapshot, you can back up the state in a manner compatible with your archiving strategy, whether you have an elaborate backup infrastructure or a simpler solution based on CIFS or NFS protocols.

If the need arises, you can restore data files from a snapshot with the `cpmdex` command.

## Restrictions for taking a snapshot

The following restrictions apply when taking snapshots.

- The `createSnapshotOperation` cannot be combined with other operations in the same Web service request.
- Do not submit snapshot requests when the Dgraph process is running updates for the data store.

- Do not submit snapshot requests when the Dgraph process is running an outer transaction request issued by the Transaction Web Service for the data store.
- The `createSnapshotOperation` requires a URI absolute path indicating where the snapshot should be recorded; it should have the following format: `file:///localdisk/username/dir`. Specifying a relative path causes the operation to fail.
- If you are running a cluster of nodes (as opposed to running a single server that is not part of the cluster), run the `createSnapshotOperation` on the data store whose Dgraph process is configured as the leader node.
- Do not modify snapshot files. Because snapshots represent internal files needed to restore the data structures, they are not human-readable and should be treated as read-only. Modifying a snapshot can corrupt the data files.

## Creating a snapshot

You create a snapshot with the `createSnapshotOperation` interface in the Administration Web Service.

Before taking the snapshot, ensure that you have reviewed the list of restrictions.

To create a snapshot:

1. Run the client application that will invoke the Administration Web Service for the specific data store. For example, in SOAP UI, access the web service for the data store as follows:

```
http://localhost:<port>/ws/admin/DataStore?wsdl
```

2. Specify the `path` and the `name`, as shown in the following excerpt (the entire Web service request is not shown):

Optionally, if you want to send this request within an outer transaction, you can also specify an ID of the outer transaction using the `outerTransactionId` attribute.

```
<admin:createSnapshotOperation path="{snap_URI}" name="{snap_name}"/>
```

- `snap_URI` represents an absolute URI path to the file system location, and is located on the same file system as the data files for a particular data store. It should be of the format `file:///localdisk/username/dir`.
- `snap_name` represents the name of the snapshot.

The Web service returns a confirmation message if the snapshot was successfully captured.

After you capture the snapshot, copy it to a safe location using the archiving method of your choice. Deleting a snapshot does not affect the Dgraph process.

## Restoring data files from a snapshot

You can restore data files from an archived snapshot using the `cpmdex` command. The command copies files from the archived snapshot into the data files of the specific Endeca data store.

The Dgraph process `bin` directory contains `cpmdex.cmd` (Windows) and `cpmdex.sh` (Linux) versions of this command.

The `cpmdex` executable (`cpmdex.cmd` on Windows and `cpmdex.sh` on Linux) is located in:

```
$EndecaServerRoot/endeca-server/dgraph/bin
```

The `cpmdex` command takes as input both the path to the archived snapshot and the path to the data files of the Endeca data store that will be restored.



**Important:** Before running the `cpmdex` command, ensure that the version of the Endeca data files to which you are restoring from the snapshot matches the version of the data files from which the snapshot was captured. You can verify the version of the Endeca data store in the standard out log for the Dgraph process.

To restore the data files from a snapshot:

1. Use the Endeca Server `stop-ds` command to stop the Endeca data store that are you are about to restore.
2. From a command prompt, run the `cpmdex` command.

An example on Windows is:

```
cpmdex -a c:\backup\my_snapshot -m c:\apps\myapp\bikes
```

An example on Linux is:

```
$ cpmdex.sh -a mnt/backup/my_snapshot -m home/apps/myapp/bikes
```

3. Start the Endeca data store with the Endeca Server `start-ds` command.

As a result, the old `bikes_indexes` data files directory is renamed to `bikes_indexes.1` and the snapshot is copied under the `bikes_indexes` name.

## cpmdex syntax

This topic contains syntax for the `cpmdex` command.

The syntax for the `cpmdex` command is as follows:

```
cpmdex -a <archive_path> -m <datastore_path> -t <transfer_path>
```

The `cpmdex` command uses the following parameters:

Options	Description
<code>-a &lt;archive_path&gt;</code>	Required. The absolute file path to the directory containing the archived snapshot.
<code>-m &lt;datastore_path&gt;</code>	Required. The absolute file path to the directory where the snapshot should be restored.  The end of this path should match the value passed to the Dgraph process when it is started.

Options	Description
<code>-t &lt;transfer_path&gt;</code>	The file path to a directory to which the snapshot should be moved. This option uses a move operation, instead of a copy, to restore the files to the data store. You may want to use this option if the backup has already been copied from the archive to the local file system and you want to save considerable I/O bandwidth.
<code>-h</code>	The help for this command.

In this example, the `cpmdex` command copies the snapshot from the `backup\2012-01-20` directory and restores it to the `apps\myapp\my_data_files` directory on Windows:

```
cpmdex -a c:\backup\2012-01-20 -m c:\apps\myapp\my_data_files
```



## Chapter 7

---

# Dgraph Administrative Tasks

This section describes some basic administrative tasks for the Dgraph, contains Dgraph troubleshooting tips, and describes the Dgraph logs.

[Checking the status of the Dgraph](#)

[About connecting Web browsers to the data store](#)

[Managing Dgraph core dump files](#)

[Collecting debugging information](#)

[Running multiple Dgraphs on the same Windows machine](#)

[Identifying connection errors](#)

## Checking the status of the Dgraph

A quick way of checking the health of a Dgraph is to get its status from the Endeca Server.

To check the aliveness of a Dgraph:

1. From a command prompt, navigate to the `endeca-cmd` directory in the Endeca Server root installation.
2. Issue the `status-ds` command, specifying a particular Endeca data store, as in this example:

```
endeca-cmd status-ds books
```

In the command output, the Current State field lists whether the Dgraph is running:

```
Current State: Started
```

## About connecting Web browsers to the data store

For security reasons, you should never allow user Web browsers to connect directly to the machine hosting the Endeca Server and the Endeca data stores.

Browsers started by non-administrators should always connect to your application through an application server.

### IPv4 and IPv6 address support

The Oracle Endeca Server and the Endeca data store Dgraph process support both IPv4 (Internet Protocol Version 4) and IPv6 (Internet Protocol Version 6) addressing schemes for connections. This IPv4 and IPv6

addressing support is configured automatically in the Oracle Endeca Server and the Dgraph, so there is no need for the administrator to do any explicit addressing configuration.

## Managing Dgraph core dump files

In the rare case of a Dgraph crash, the Dgraph writes its core dump files on disk.

When the Dgraph runs on a very large data set, the size of its data files stored in-memory may exceed the size of the physical RAM. If such a Dgraph process fails, it may need to write out potentially very large core dump files on disk. The core files are written to the Endeca Server's `logs` directory.

To troubleshoot the Dgraph, it is often useful to preserve the entire set of core files written out as a result of such failures. When there is not enough disk space, only a portion of the files is written to disk until this process stops. Since the most valuable troubleshooting information is contained in the last portion of core files, to make these files meaningful for troubleshooting purposes, it is important to provision enough disk space to capture the files in their entirety.

Two situations are possible, depending on your goal:

- To troubleshoot a Dgraph crash, provision enough disk space to capture the entire set of core files. In this case, the files will be saved at the expense of potentially filling up the disk.
- To prevent filling up the disk, you can limit the size of these files on the operating system level. In this case, with large Dgraph applications, only a portion of core files is saved on disk. This may limit their usefulness for debugging purposes.

Keep in mind that the Endeca Server will attempt to restart the Dgraph when it crashes. If the start-up retry fails, the Endeca Server will retry the start-up only one more time.

[Managing Dgraph crash dump files on Windows](#)

[Managing Dgraph core dump files on Linux](#)

## Managing Dgraph crash dump files on Windows

On Windows, all Dgraph crash dump files are saved on disk by default.

The Dgraph uses the `MiniDump` function from the Microsoft `DbgHelp` library.

Provision enough disk space to accommodate core files based on this estimate:

- The projected upper limit for the size of these files is equal, at a maximum, to the size of the physical memory used by the data files for all data stores hosted on the Oracle Endeca Server. Often the files take up less space than that.

## Managing Dgraph core dump files on Linux

It is recommended to use the `ulimit -c unlimited` setting for the Dgraph process core dump files. Non-limited core files contain all Dgraph data that is resident in memory (RSS of the Dgraph process).

Since large applications powered by the Oracle Endeca Server may take up the entire amount of available RAM, the core dump files can also grow large and take up the space equal to the size of the physical RAM on disk plus the size of the server data files in memory.

Provision enough disk space to accommodate core files based on this estimate:

- The projected upper limit for the size of these files should be equal, at a maximum, to the size of the physical RAM. Often the files take up less space than that.



**Note:** If you are not setting `ulimit -c unlimited`, you could be seeing the Dgraph process crashes that do not write any core files to disk, since on some Linux installations the default for `ulimit -c` is set to 0.

Alternatively, it is possible to limit the size of core files with the `ulimit -c <size>` command, although this is not recommended. If you set the limit size in this way, the core files cannot be used for debugging, although their presence will confirm that the Dgraph had crashed. To be able to troubleshoot the crash, change this setting to `ulimit -c unlimited`, and reproduce the crash while capturing the entire core file. Similarly, to enable Endeca Support to troubleshoot the crash, you will need to reproduce the crash while capturing the full core file.

## Collecting debugging information

Before attempting to debug an issue with the data store, collect the following information.

- Hardware specifications and configuration.
- Description of the Oracle Endeca Server topology (number and names of servers, number of Dgraphs and their data stores).
- The data from the Dgraph Server Statistics page.
- Dgraph input.
- Description of which Dgraphs are affected.

[Logs created by the Dgraph](#)

### Logs created by the Dgraph

The Dgraph creates several logs, although some of these logs depend on your implementation and the components that you may be using. This topic provides a summary of these logs.

You can use these Dgraph logs to troubleshoot queries, or to track performance of particular queries or updates.

#### Dgraph request log

The Dgraph request log is always created. You can use it to debug both requests and update processing. It contains one entry for each request processed. The requests are sorted by their timestamp.

The name of the Dgraph request log is `dataStoreName.reqlog` (where `dataStoreName` is the name of the Endeca data store that the Dgraph is servicing). By default, the request logs are stored in the Endeca Server's `logs` directory. Note that you can customize the location of the `logs` directory. The `/admin/dataStoreName?op=logroll` command forces a request log roll, with the side effect of remapping it.

By default, the Dgraph truncates the contents of the body for POST requests at 64K. This default setting saves disk space in the log, especially during the process of adding large numbers of records to the data store. If you need to review the log for the full contents of the POST request body, contact Endeca Support.

## Dgraph stdout/stderr log

The name of the Dgraph stdout/stderr log is *dataStoreName.out*. By default, the stdout/stderr logs are also stored in the Endeca Server's `logs` directory.

The Dgraph stdout/stderr log includes startup messages as well as warning and error messages. You can increase the verbosity of the log via the Dgraph `-v` flag.

## Running multiple Dgraphs on the same Windows machine

If you have more than one Endeca data store running on a single Windows machine, the Endeca Server, by default, assigns each Dgraph a port that does not conflict with other Dgraph ports on that machine.

This prevents multiple Dgraphs running on a single machine from presenting inconsistent behavior. If you decide to assign the Dgraph ports yourself (by using the `--ws-port` and `--bulk-load-port` options), make sure that each assigned port is unique on that machine.

## Identifying connection errors

If the Dgraph standard out log contains `connection broken` messages, although it may look like the problem occurred with the Dgraph, the actual cause of the problem is usually a broken connection between the server that hosts the front-end application and the server that hosts the Dgraph.

In the case of connection errors, various parts of the implementation issue the following error and warning messages:

- The Dgraph standard out log contains warnings similar to the following:

```
WARN [DATE TIME] UTC (1239830549803)
DGRAPH {dgraph}: Aborting request: connection broken: client 10.10.21.21
```

- The Dgraph request log contains an abnormal `status 0` message similar to the following:

```
1239830549803 10.6.35.35 - 349 0 19.35 0.00 0 - 0 0 - -
```

Typically, the `connection broken` message means that the Dgraph encountered an unexpected failure in the connection between the client and the Dgraph. This type of error may occur outside the Dgraph, such as in the network, or be caused by the timeout of the client application session.

Investigate the connection between the client and the Dgraph. For example, to prevent timeouts of the client application sessions, you may decide to implement front-end application retries.



## Chapter 8

# Administrative Operations

---

The Oracle Endeca Server supports many Dgraph administrative and configuration operations that you can access through simple URLs. You can use these operations to control the behavior of the Dgraph processes that are running under the Oracle Endeca Server.

[About administrative and configuration operations](#)

[About logging variables for the Dgraph](#)

## About administrative and configuration operations

Administrative and configuration operations make it possible to check Dgraph statistics, and enable or disable diagnostic flags without having to stop a running Dgraph. This section lists the supported administrative and configuration URLs, describes the functions of each URL, and defines the syntax of those URLs.

### Syntax of administrative and configuration operations

In the following listings:

- `<host>` refers to the hostname or IP address of the Oracle Endeca Server.
- `<port>` refers to the port on which the Oracle Endeca Server is listening.
- `<dstore>` refers to the name of the Endeca data store on which command will operate.

Queries to these URLs are handled in the Dgraph's request queue like any other request—that is, they are handled on a first-come, first-served basis. They are also reported in the Dgraph request log like any other request.

For administrative operations, the syntax is:

```
http://<host>:<port>/admin/<dstore>?op=<supported-operation>
```

For configuration operations, the syntax is:

```
http://<host>:<port>/config/<dstore>?op=<supported-operation>
```



**Note:** If you are using HTTPS mode, use `https` in the URL.

## List of administrative operations

Administrative (or admin) operations listed in this topic allow you to control the behavior of an Endeca data store's Dgraph process.

The Dgraph recognizes the following admin operations:

Admin operation	Description
<code>/admin/datastore?op=help</code>	Returns the usage page for all of the admin operations.
<code>/admin/datastore?op=flush</code>	Specifies when the Dgraph should flush its dynamic cache.
<code>/admin/datastore?op=logroll</code>	Forces a query log roll.
<code>/admin/datastore?op=merge</code>	Merges update generations and sets the system's merge policy.
<code>/admin/datastore?op=ping</code>	Checks the aliveness of a Dgraph and returns a lightweight message.
<code>/admin/datastore?op=reload-services</code>	A Web services operation that reloads the application's main and library modules.
<code>/admin/datastore?op=stats</code>	Returns the Dgraph Server Statistics page.
<code>/admin/datastore?op=statsreset</code>	Resets the Dgraph Server Statistics page.
<code>/admin/datastore?op=updateaspell</code>	Rebuilds the aspell dictionary for spelling correction from the data corpus.

[\*help\*](#)

[\*flush\*](#)

[\*logroll\*](#)

[\*merge\*](#)

[\*ping\*](#)

[\*reload-services\*](#)

[\*stats\*](#)

[\*statsreset\*](#)

[\*updateaspell\*](#)

## help

`/admin/datastore?op=help` returns the usage page for all of the administrative operations.

## flush

`/admin/datastore?op=flush` flushes the Dgraph cache.

The `flush` operation clears all entries from the Dgraph cache. It returns the following message:

```
flushing cache...
```

## logroll

`/admin/datastore?op=logroll` forces a query log roll.

The `logroll` command returns a message similar to the following:

```
rolling log... Successfully rolled log file.
```

## merge

`/admin/datastore?op=merge` forces a merge, and (optionally) changes the merge policy of a running Dgraph. In a cluster of Dgraph nodes, this command should be used on the leader node only.

[Managing the Merge Policy](#)

## ping

`/admin/datastore?op=ping` checks the aliveness of a Dgraph and returns a lightweight message.

You can view the Dgraph Statistics page to check whether the Dgraph is running and accepting queries, but that comes with some overhead. A quicker way to check the aliveness of a Dgraph is by running the `ping` command.

Because ping requests are given the highest priority and are processed synchronously (as they are received), a ping response time is independent of the number of outstanding requests in the Dgraph.

The `ping` command returns a lightweight page that lists the Dgraph process, the current date and time, such as the following:

```
dgraph Web07:5556 responding at Wed Mar 28 15:35:27 2012
```

You can use this operation to monitor the health or heartbeat of the Dgraph, and as a health check for load balancers.

## reload-services

`/admin/datastore?op=reload-services` reloads the application's main and library modules for the Web services.

The `reload-services` operation causes the Dgraph to process all existing preceding queries, temporarily stop processing other queries, and begin to process the `reload-services` request. After it finishes

processing this operation, the Dgraph resumes processing queries that queued up temporarily behind this request.

In a cluster of Dgraph nodes, this command should be run on the leader node only.



**Note:** `admin/dstore?op=reload-services` can be a time-consuming operation.

## stats

`/admin/datastore?op=stats` returns the Dgraph Server Statistics page.

The Dgraph Server Statistics page provides a detailed breakdown of what the Dgraph is doing, and is a useful source of information about your Endeca implementation's configuration and performance. It provides information such as startup time, last data files creation time, and path. This lets you focus your tuning and load-balancing efforts. By examining this page, you can see where the Dgraph is spending its time. Begin your tuning efforts by identifying the features on the **Details > Hotspots** section with the highest totals.

## statsreset

`/admin/datastore?op=statsreset` resets the Dgraph Server Statistics page.

The `statsreset` operation returns the following message:

```
resetting server stats...
```

## updateaspell

The `admin/datastore?op=updateaspell` administrative operation lets you rebuild the aspell dictionary for spelling correction from the data corpus while continuing to issue queries and updates to the Dgraph and without stopping and restarting it.

Run this command after you have added data records to the Dgraph, to enable spelling correction in the Dgraph.

During the data ingest process, you can run the `admin/datastore?op=updateaspell` command periodically to update the spelling dictionary used by the Dgraph for Automatic Spelling Correction and Did You Mean (DYM).

In a cluster of Dgraph nodes, this command should be run on the leader node only.

The `admin/datastore?op=updateaspell` operation performs the following actions:

- Crawls the text search portion of the data files in the Oracle Endeca Server for all terms which meet the constraint settings.

The constraint settings include minimum word occurrences and maximum and minimum number of characters, for records and attribute values. The Dgraph uses these constraints to update the spelling dictionary. You can change them in the Global Configuration Record.

- Compiles a temporary text version of the `aspell` word list, `<datastore>.worddat`.
- Converts this word list to the binary format required by `aspell`
- Writes the generated binary file into the current data files representation in the Dgraph.

- Makes the updated `aspell` spelling dictionary available in the Dgraph for processing of all queries arriving after this update to the data files. The Dgraph uses this updated dictionary when processing all future queries.



**Note:** Because of the nature of continuous query, once the Dgraph processes this administrative request, it will start using the updated spelling dictionary after a certain point in its processing, and all newly incoming queries will be answered against the updated spelling dictionary. However, it is not possible to identify after which particular partial update or after which query the Dgraph will start using the newly updated spelling dictionary.

The Dgraph applies the updated settings while continuing to run queries and without needing to restart.

Only one `admin/datastore?op=updateaspell` operation can be processed at a time.



**Note:** If `admin/datastore?op=updateaspell` is started within an outer transaction, it must reference the correct transaction ID, as in the following example:

```
admin/wine?op=updateaspell&outerTransactionId=42
```

If a transaction is running and you don't specify the ID, or the ID is incorrect, the request is rejected.

The `admin/datastore?op=updateaspell` operation returns output similar to the following in the Dgraph error log:

```
...
spellengine aspell ran successfully.
```

If you start the Dgraph with the `-v` flag, the output also contains a line similar to the following:

```
Time taken for updateaspell, including wait time on any
previous updateaspell, was 290.378174 ms.
```

## About logging variables for the Dgraph

You can use logging variables with config operations. This lets you obtain detailed information about Dgraph processing, to help diagnose unexpected application behavior or performance problems, without stopping and restarting the Dgraph or requiring a configuration update.

Although you can also specify general verbose logging at the Dgraph command line with the `-v` flag, it requires a Dgraph restart to take effect.

[Logging variable operation syntax](#)

[List of configuration operations](#)

[List of supported logging variables](#)

### Logging variable operation syntax

Dgraph logging variables are toggled using the `/config/dstore?op=log-enable&name=<variable-name>` and `/config/dstore?op=log-disable&name=<variable-name>` operations.

You can include multiple logging variables in a single request. Unrecognized logging variables generate warnings.

For example, this operation:

```
/config/wine?op=log-enable&name=requestverbose
```

turns on verbose logging for queries, while this operation:

```
config/wine?op=log-enable&name=textsearchrelrankverbose&name=textsearchspellverbose
```

turns on verbose logging for both the text search relevance ranking and spelling features.

However, this operation:

```
config/wine?op=log-enable&name=allmylogs
```

returns an unsupported logging setting message.

In addition, the following operations are supported:

- `/config/dstore?op=log-status` returns a list of all logging variables with their values (true or false).
- The special name `all` can be used with `/config/dstore?op=log-enable` or `/config?/dstoreop=log-disable` to set all logging variables.

## List of configuration operations

Configuration (or config) operations listed in this topic allow you to modify configuration and logging information for the Dgraph from within the system.

The Dgraph recognizes the following config operations:

Config operation	Description
<code>/config/dstore?op=help</code>	Returns the usage page for all of the config operations.
<code>/config/dstore?op=log-enable</code>	Enables verbose logging for one or more specified variables.
<code>/config/dstore?op=log-disable</code>	Disables verbose logging for one or more specified variables.
<code>/config/dstore?op=log-status</code>	Returns verbose logging status.

## List of supported logging variables

The following table describes the supported logging variables that you can use with related config operations to toggle logging verbosity for specified features.

Logging variable names are not case sensitive.

Variable	Description
<code>verbose</code>	Enables verbose mode.
<code>requestverbose</code>	Prints information about each request to <code>stdout</code> .
<code>textsearchrelrankverbose</code>	Enables verbose information about relevance ranking during search query processing.

Variable	Description
textsearchspellverbose	Enables verbose output for spelling correction features.
dgraphperfverbose	Enables verbose performance debugging messages during core Dgraph navigation computations.
dgraphrefinementgroupverbose	Enables refinement verbose/debugging messages.

[log-enable](#)

[log-disable](#)

[log-status](#)

[help](#)

## log-enable

The `log-enable` operation lets you turn on verbose logging.

You can include multiple logging variables in a single request. Unrecognized logging variables generate warnings.

For example, this operation:

```
/config/wine?op=log-enable&name=requestverbose
```

turns on verbose logging for queries, while this operation:

```
config/wine?op=log-enable&name=textsearchrelevanceverbose&name=textsearchspellverbose
```

turns on verbose logging for both the text search relevance ranking and spelling features.

However, this operation:

```
config/wine?op=log-enable&name=allmylogs
```

returns an "Unsupported logging setting" message.

## log-disable

The `log-disable` operation turns off verbose logging.

`/config/dstore?op=log-disable` with no arguments returns the same output as `log-status`.

## log-status

The `log-status` operation returns a list of all logging variables with their values (true or false).

For example, if you have enabled verbose logging on two of the features, you would see a message similar to the following:

```
Logging settings:
verbose - FALSE
requestverbose - TRUE
updateverbose - FALSE
recordfilterperfverbose - FALSE
```

```
textsearchreirankverbose - TRUE
textsearchspellverbose - FALSE
dgraphperfverbose - FALSE
dgraphrefinementgroupverbose - FALSE
```

## help

`/config/dstore?op=help` returns the usage page for all of the config operations.



## Chapter 9

# Configuring Endeca Server as a Service

---

This section describes how to control the Endeca Server from the Windows Services utility or from the Linux `inittab`.

*Running the Endeca Server as a Windows service*

*Starting the Endeca Server from `inittab`*

## Running the Endeca Server as a Windows service

You can create a Windows service for running the Endeca Server in service mode.

The Windows SC tool (`sc.exe`) communicates with the Windows Service Controller and installed Windows services. The SC tool allows you to create a Windows service for the Endeca Server. You can then start and stop the Endeca Server from the Windows Services utility, as well as make configuration changes (such as configuring the service to automatically restart in case of a failure). Keep in mind that the SC tool (`sc.exe`) is case-insensitive.



**Note:** The Endeca Cluster Coordinator service cannot be run as a Windows service. This means that if you are using the Endeca Clustering feature and run the Endeca Server as a Windows service, you should closely monitor the state of the Cluster Coordinator service. The reason is that if the Endeca Server service crashes, the Windows Service Controller will automatically restart it. However, if the Cluster Coordinator service crashes, then it is not automatically restarted. This can lead to a situation where the Dgraph processes are running but the Cluster Coordinator service is not.

For more information, refer to these Web pages on the Microsoft site:

- For more information on creating Windows services: <http://support.microsoft.com/kb/251192>
- For more information on the `sc.exe` command: <http://technet.microsoft.com/en-us/library/bb490995.aspx>

## Windows Service Wrapper file

The `service-wrapper-7.4.exe` file is shipped in the `endeca-server` directory. This is the file that you will use with the SC tool to create the Windows service for the Endeca Server, as described in the following topics.

## SC Create command syntax

This topic describes the various options of the SC command with the `Create` command option.

The SC command communicates with the Windows Service Controller and installed services. When used with its `create` command option, you can use it to create a Windows service under which the Endeca Server will run.

The `SC Create` command uses the following format:

```
sc create serviceName binpath= "path\to\service-wrapper-7.4.exe" optionName= optionValue...
```

where:

- `create` is the command to be run by SC (this command name is mandatory to create a service).
- `serviceName` is the name of the Windows service to be created. This is the name given to the service key in the registry. Note that this name is different from the display name.
- `binpath` is a mandatory parameter that specifies the path to the `service-wrapper-7.4.exe` file. Note that a space must be used between the `binpath` parameter and its argument. You should also use double quotes around the argument.
- `optionName` specifies optional parameters, which are described in the table below.

## SC Create options

You can use these `SC Create` options to further customize the Windows service. Note that the option name includes the equal sign, and a space is required between the equal sign and the option value.

Option Name/Values	Meaning
<code>type= &lt;serviceType&gt;</code>	The type of service to be created. Use the <code>own</code> parameter value, which means the service runs in its own process. It does not share an executable file with other services. This is the default for the <code>sc create</code> command. Note that other service types are available, but you should use the <code>own</code> value.
<code>start= &lt;startType&gt;</code>	The start type for the service: <ul style="list-style-type: none"> <li>• <code>auto</code> – A service that automatically starts each time the computer is restarted.</li> <li>• <code>demand</code> – A service that must be manually started. This is the default value if <code>start=</code> is not specified. <code>demand</code> maps to <code>Manual</code> in the Services Control Manager.</li> <li>• <code>delayed-auto</code> – The SCM supports delayed auto-start services to improve system performance at boot time without affecting the user experience. The SCM makes a list of delayed auto-start services during boot and starts them one at a time after the delay has passed, honoring dependencies. There is no specific time guarantee as to when the service will be started.</li> <li>• <code>disabled</code> – A service that cannot be started. To start a disabled service, change the start type to another start value.</li> </ul>

Option Name/Values	Meaning
error= <errorSeverity>	<p>The severity of error if the service does not start during boot:</p> <ul style="list-style-type: none"> <li>• <b>normal</b> – The error is logged and a message box is displayed informing the user that a service has failed to start. System startup will continue. This is the default setting.</li> <li>• <b>severe</b> – The error is logged (if possible). The computer attempts to restart with the last-known-good configuration. This could result in the computer being able to restart, but the service may still be unable to run.</li> <li>• <b>critical</b> – The error is logged (if possible). The computer attempts to restart with the last-known-good configuration. If the last-known-good configuration fails, system startup also fails, and the boot process halts with a Stop error.</li> <li>• <b>ignore</b> – The error is logged and startup continues. No notification is given to the user beyond recording the error in the Event Log.</li> </ul>
group= <loadOrderGroup>	Name of group of which this service is a member. The list of groups are stored in the registry under the ServiceGroupOrder key. Default is null.
tag= yes   no	Do not use this parameter as tags are used only for device driver service types.
depend= <dependencies>	Names of services or groups that must start before this service. Each name is separated by / (forward slash).
obj= <accountName>	Name of the account under in the service will run. The specified account must exist and must be a valid account. Default is <b>LocalSystem</b> .
password= <password>	Password of the obj account. A password is required if an account other than the <b>LocalSystem</b> account is used.
displayname= <displayName>	A friendly, meaningful name that can be used in user-interface programs to identify the service to users. For example, if the service name is <b>ESService</b> , you can specify <b>Oracle Endeca Server</b> as the display name so that will be more meaningful when shown in the Windows Services Control Manager.

## SC Create example

The following SC Create example creates a Windows service for the Endeca Server (note that the command is on one line, but is indented here for ease of reading):

```
sc create EndecaServer displayname= "Oracle Endeca Server"
  type= own error= severe obj= "CORPDEV\EndecaUser" password= banx912
  binpath= "C:\Oracle\Endeca\Server\7.4.0\endeca-server\service-wrapper-7.4.exe"
```

The sample command does the following:

- Creates a Windows service named **EndecaServer**.
- Uses **Oracle Endeca Server** as the display name for the service.
- Sets the service type as `own` (which means the service runs in its own process).
- Sets `severe` as the severity of error if the service does not start during the boot process.
- Specifies that the service run under the **CORPDEV\EndecaUser** user account, which has **banx912** as its password.
- Sets the binary path of the `service-wrapper-7.4.exe` executable.

For ease of use, you can place the command in a batch script.

## Creating the Endeca Server Windows service

Use the `SC` command's `Create` option to create the Endeca Server Windows service.

Before running this create-service procedure, make sure that you have Administrator rights.

To create a Windows service for the Endeca Server:

1. Click on the **Start** button in the Windows taskbar.
2. Locate the Command Prompt menu item and right-click on the Command Prompt.
3. On the pop-up right click context menu, select **Run as administrator**.
4. In the Command Prompt, enter the `SC Create` command with the appropriate options.

If the command was successful, the SCM will return this message:

```
[SC] CreateService SUCCESS
```

If the command was not successful, the SCM may return this message:

```
[SC] OpenSCManager FAILED 5:
```

```
Access is denied.
```

If you do receive this error, verify that you are a member of the Administrators group on the machine (for example, by using the Microsoft Management Console). If you do have Administrator rights, check that you are opening the Command Prompt with the **Run as administrator** option.

After the service has been created, you can use the `SC Config` command to change any parameter set by the `SC Create` command.

## Setting a service description

Use the `SC` command's `Description` option to set a description for the Endeca Server Windows service.

Before adding a description to the service, make sure that you have Administrator rights.

When you create a service with the `SC Create` command, you cannot set a service description. However, after creating the service, you can use the `SC` command's `Description` option to add a new description or to modify an existing description.

The format of the `SC Description` command is:

```
sc description serviceName descriptionText
```

where *serviceName* is the name of the service to modify and *descriptionText* is the new description within double quotes. There is no limit to the number of characters that can be contained in the service description.

To add or modify the description of the Endeca Server Windows service:

1. Stop the Endeca Server Windows service.
2. Click the **Start** button in the Windows taskbar.
3. In the menu, right-click **Command Prompt**.
4. On the pop-up right click context menu, select **Run as administrator**.
5. At the command prompt, enter the `sc description` command with the service name and new description, as in this example, which sets a description for the EndecaServer service:

```
sc description EndecaServer "Creates and manages Endeca data stores."
```

If the command was successful, the SCM will return this message:

```
[SC] ChangeServiceConfig2 SUCCESS
```

## Modifying the service configuration

Use the `SC` command's `Config` option to modify the configuration of the Endeca Server service.

Before attempting to modify the service configuration, make sure that you have Administrator rights.

After you create the Endeca Server service with the `SC Create` command, you can use the `SC Config` command to modify the service configuration. Because both commands use the exact same set of parameters, any parameter that you set with `SC Create` can be modified with `SC Config`.

The format of the `SC Config` command is:

```
sc config serviceName optionName= optionValue...
```

where *serviceName* is the name of the existing Endeca Server Windows service to be modified.

When using the `SC Config` command, you specify only the parameter settings that will be changed. Any parameter setting that is not specified will remain as-is in the service configuration. Note that to change the service description, you must use the `SC Description` command.

To modify the Endeca Server Windows service:

1. Stop the Endeca Server Windows service.
2. Click the **Start** button in the Windows taskbar.
3. In the menu, right-click **Command Prompt**.
4. On the pop-up right click context menu, select **Run as administrator**.
5. At the command prompt, enter the `SC Config` command with the service name to be modified and the parameters to be changed.

If the command was successful, the SCM will return this message:

```
[SC] ChangeServiceConfig SUCCESS
```

## Deleting the Endeca Server Windows service

Use the `SC` command's `Delete` option to remove the Endeca Server Windows service.

Before deleting the service, make sure that you have Administrator rights.

The format of the `SC Delete` command is:

```
sc delete serviceName
```

where *serviceName* is the name of the service to be deleted.

To delete the Endeca Server Windows service:

1. Stop the Endeca Server Windows service.
2. Click the **Start** button in the Windows taskbar.
3. In the menu, right-click **Command Prompt**.
4. On the pop-up right click context menu, select **Run as administrator**.
5. At the command prompt, enter the `SC Delete` command with the service name to be deleted, as in this example:

```
sc delete EndecaServer
```

If the command was successful, the SCM will return this message:

```
[SC] DeleteService SUCCESS
```

If the command was not successful, the SCM may return this message:

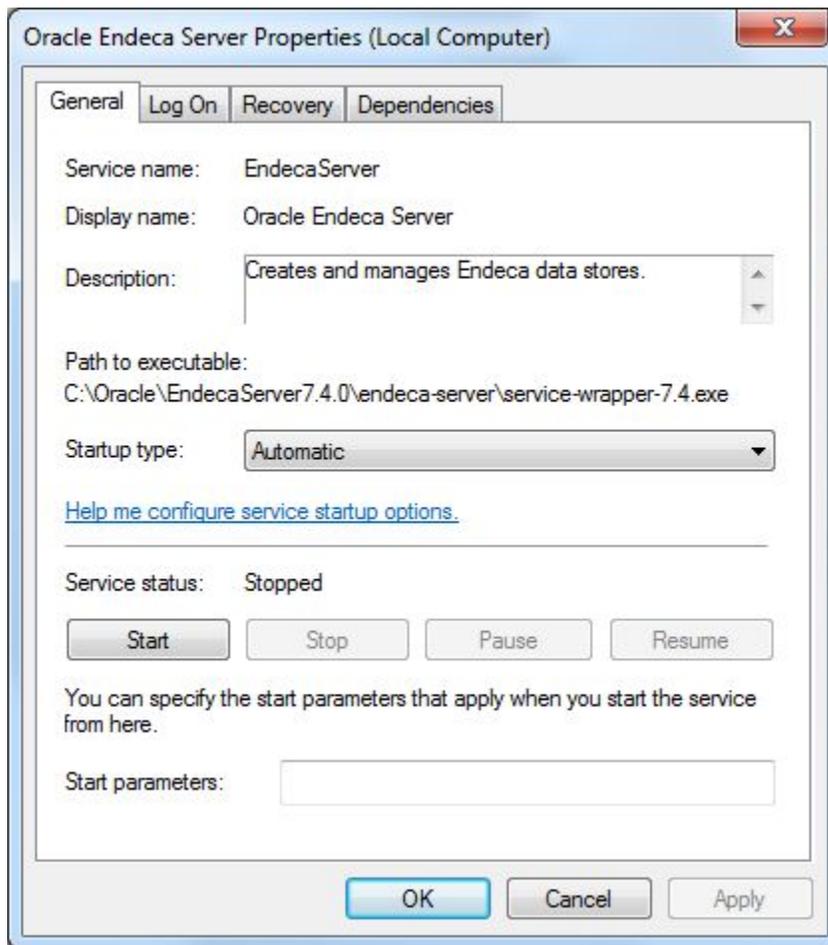
```
[SC] OpenService FAILED 5:  
Access is denied.
```

If you do receive this error, first verify that you are a member of the Administrators group on the machine. If you do have Administrator rights, check that you are opening the **Command Prompt** with the **Run as administrator** option.

## Using the Windows Services utility

The Windows Services utility allows you to control and configure the Endeca Server service.

The Endeca Server service, when selected in the Windows Services utility, looks like this example:



## General tab

The **General** tab allows you to start and stop the Endeca Server service. Either operation will log an appropriate message to the Endeca Server's `endeca-server-service.log` file in the `endeca-server\logs` directory.

Clicking the **Stop** button sends a shutdown request (with time limit of 120 seconds) to the Endeca Server. If any Endeca data stores are running, this 120-second time-out limit will override the shutdown time-outs for the individual Endeca data stores. This means that any background merges or queries that are in progress will be aborted when the time-out limit is reached.

You can use the **Startup type** drop-down menu to change the startup type to Automatic, Automatic (Delayed Start), Manual, or Disabled. Clicking the help link displays usage information for this option.

Note that the **Start parameters** field has no effect on the service.

## Log On tab

The **Log On** tab allows you to change the account under which the Endeca Server service runs. This option is especially useful if you created the service to run under the Local System account and want to change to a user account. The tab has a help link that provides detailed information on configuring the user account log on options.

## Recovery tab

The **Recovery** tab is used to configure recovery actions when a service fails. You can configure the Endeca Server service for automatic restart. That is, the Endeca Server service will restart in the case of a crash or machine reboot.

To obtain information on how to configure the computer's response if the Endeca Server service fails, click the "Help me set up recovery actions" link on the tab.

## Logging in service mode

Endeca Server logging is supported in service mode.

When the Endeca Server is run in service mode, it will log startup and shutdown messages to these logs:

- `endeca-server-service.log` is for messages generated by the Windows Service Manager.
- `endeca-server.log` is the Endeca Server's stdout/stderr log.
- `<dataStore>.out` (for example, `bikes.out`) is the stdout/stderr log for a specific Endeca data store.

By default, these logs are stored in the `endeca-server\logs` directory.

## Starting the Endeca Server from inittab

In a Linux production environment, Endeca Server can be started by `init` from `inittab`.

In a Linux development environment, Endeca Server can be started from the command line. In a Linux production environment, however, Endeca recommends that it be started by `init` from `inittab`. If the service crashes or is terminated, `init` automatically restarts it.

The `inittab` entry should be formatted like this:

```
es:2345:respawn:/bin/su - <endeca_user> -c "/absolute/path/to/start.sh"
```

where:

- **es** is the `inittab` entry identifier.
- **2345** lists the runlevels for which the specified action should be taken.
- **respawn** is the action to be taken, which is that the process will be restarted whenever it terminates.
- **/bin/su** specifies the process to be executed. In this case, a non-root user will run the `start.sh` command. It is a best practice to run the Endeca Server as a user other than root.
- **-c start.sh** specifies that the `start.sh` command will be run, using the absolute path to the command.



This section describes how to set and manage a merge policy for each Endeca data store.

[Using a merge policy for incremental updates](#)

[Types of merge policies](#)

[Setting or changing the merge policy](#)

[Changing the merge policy of a running data store](#)

[Forcing a merge](#)

[merge](#)

## Using a merge policy for incremental updates

A merge policy determines how frequently the Dgraph merges incremental update generations in its data files, for a specific data store.

The data layer stores the data files of an Endeca data store as a series of internal files with versions. As a result:

- Old versions can be accessed while new versions are created.
- Old versions are garbage-collected when no longer needed.

A version is persisted as a sequence of generation files. A new version appends a new generation file to the sequence. Query latency depends, in part, on the number and size of generation files used to store the data files.

Generation files are combined through a process called *merging*. Merging is a background task that does not affect the Endeca data store features, but may affect its performance. Because of this, you can set a *merge policy* that dictates the aggressiveness of the merges. In a clustered environment, merge policy can be set on the leader node only.

## Types of merge policies

You can set the merge policy to one of two settings: balanced or aggressive.

- **Balanced:** This policy strikes a balance between low latency and high throughput. This is the default policy of an Endeca data store.
- **Aggressive:** This policy merges frequently and completely to keep query latency low at the expense of average throughput.

The balanced policy is recommended for the majority of applications. However, aggressive merging may help those applications that meet the following criteria:

- Query latency is the primary concern.
- A large fraction of the records (for example, 20%) are either modified or deleted by incremental updates before re-baselines.
- The time to perform an aggressive merge is less than the time between incremental updates.



**Note:** Under normal conditions, you do not need to change the default balanced policy.

## Setting or changing the merge policy

The `mdex-config_MergePolicy` attribute in the Global Configuration Record (or GCR) sets the merge policy for the Endeca data store.

You can set the merge policy by sending a request to the Configuration Web Service to change the settings for the GCR, utilizing Integrator or any Web services tool, such as soapUI.

Alternatively, you can use the URL `/admin/dstore?op=merge` command to change the merge policy of a running data store, or to force a merge.

If you are running a cluster of nodes, you can change the merge policy on the leader node only.

Both of these methods are discussed in the following topics.

## Setting the merge policy with the Configuration Web Service

You can use the Configuration Web Service to programmatically retrieve and set the merge policy in the GCR.

### Retrieving the merge policy with the Configuration Web Service

You can retrieve the Global Configuration Record to see the current setting for the merge policy.

To retrieve the Global Configuration Record:

1. In a tool such as soapUI, access the Configuration Web Service on the Oracle Endeca Server for the data store, as in this example for the "books" data store:

```
http://localhost:port/ws/config/books?wsdl
```

2. Use the `getGlobalConfigRecord` function to retrieve the Global Configuration Record via the Configuration Web Service, as in this example:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <config:configTransaction
      xmlns:config="http://www.endeca.com/MDEX/config/services/types/1/0"
      xmlns:mdex="http://www.endeca.com/MDEX/config/XQuery/2009/09">
      <config:getGlobalConfigRecord />
    </config:configTransaction>
  </soap:Body>
</soap:Envelope>
```

The results response from the Configuration Web Service should look like this example (the SOAP elements have been removed):

```
<config-service:results
  xmlns:config-service="http://www.endeca.com/MDEX/config/services/types/1/0">
  <mdex:globalConfigRecord xmlns:mdex="http://www.endeca.com/MDEX/config/XQuery/2009/09">
    <mdex:record xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      ...
      <mdex-config_Key type="mdex:string">global</mdex-config_Key>
      <mdex-config_MergePolicy type="mdex:string">balanced</mdex-config_MergePolicy>
      ...
    </mdex:record>
  </mdex:globalConfigRecord>
</config-service:results>
```

In this example, the merge policy is set to `balanced` for the Dgraph process of the Endeca data store.

## Setting the merge policy with the Configuration Web Service

You can programmatically set the merge policy for the data store by updating the Global Configuration Record.

To set the merge policy in the Global Configuration Record:

1. In a tool such as SOAP UI, access the Configuration Web Service on the Oracle Endeca Server for the data store:

```
http://localhost:port/ws/config/datastore?wsdl
```

2. Use the `putGlobalConfigRecord` function to set the value of the `mdex-config_MergePolicy` attribute in the Global Configuration Record, as in this example that changes the merge policy to `aggressive` (note that all attributes must be put, but the example omits most of them for the sake of clarity):

```
<config:configTransaction
  xmlns:config="http://www.endeca.com/MDEX/config/services/types/1/0"
  xmlns:mdex="http://www.endeca.com/MDEX/config/XQuery/2009/09">
  <config:putGlobalConfigRecord>
    <mdex:record xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      ...
      <mdex-config_Key type="mdex:string">global</mdex-config_Key>
      <mdex-config_MergePolicy type="mdex:string">aggressive</mdex-config_MergePolicy>
      ...
    </mdex:record>
  </config:putGlobalConfigRecord>
</config:configTransaction>
```

## Changing the merge policy of a running data store

The URL `/admin/datastore?op=merge` command can be used to change the merge policy of a running Endeca data store.

The sticky version of the `merge` command is intended to change the merge policy of a running data store. The duration of the policy change is for the life of the current Dgraph process (that is, until the Dgraph process is restarted), or until another sticky change is performed during the current Dgraph process.

The format of the sticky version of the command is:

```
/admin/datastore?op=merge&mergepolicy=<policy>&stickymergepolicy
```

where *policy* is either *balanced* or *aggressive*, and *datastore* is the name of the Endeca data store for which you are changing the policy.

The command also performs a merge operation if warranted.

This example:

```
http://localhost:7770/admin/books?op=merge&mergepolicy=aggressive&stickymergepolicy
```

forces a merge operation (if one is needed) and changes the current merge policy to an aggressive policy for the "books" data store. In this example, the Endeca Server is running on the localhost machine on port 7770.

## Forcing a merge

The URL `/admin/datastore?op=merge` command can also be used to force a merge.

Manually forcing a merge is considered a one-time version, because after the merge operation is performed (via a temporary *aggressive* change to the merge policy), the merge policy reverts to its previous setting.

The one-time version of the `merge` command is used to perform a complete merge of all generations without making a change to the default merge policy.

Forcing a merge implies starting a full merge of all generations of data files. When running this command, be aware of the following considerations:

- **Memory requirements.** Forcing a complete merge utilizes the server's memory. If the amount of memory reaches the amount of RAM that is available, the merge operation will continue to work, but could run substantially slower and have a higher impact on query performance.
- **Disk space requirements.** Forcing a merge requires provisioning three times the amount of disk space as the current size of the data files for the particular data store. If not enough disk space is provisioned, it could be disruptive to force a complete merge. This consideration is especially important for running this command on the Oracle Endeca Server in a production environment.

In a cluster of nodes, you can use this command only on the Endeca data store whose Dgraph process is configured as the leader node.

The format of the one-time version of the command is:

```
/admin/datastore?op=merge&mergepolicy=<version>
```

The following example assumes that the Dgraph process of the data store is using a balanced merge policy, and you want to temporarily apply an aggressive policy so that the merging can be performed.

```
http://localhost:7770/admin/books?op=merge&mergepolicy=aggressive
```

When you issue the command, the resulting Web page will look like this example:

```
Dgraph admin, OK.  
Dgraph Manual merge started at Sat March 31 09:52:47 2012
```

After the merging is performed, the merge policy reverts to its previous setting.



## Chapter 11

---

# Deploying a Cluster of Oracle Endeca Servers

This section discusses how to deploy a cluster with multiple Oracle Endeca Servers.

[Cluster overview](#)

[Oracle Endeca Server cluster architecture](#)

[Important cluster concepts](#)

[Before you begin](#)

[Building a cluster](#)

[Maintaining a cluster](#)

## Cluster overview

This topic introduces the cluster of Oracle Endeca Server nodes and describes its capabilities.

### About the cluster

A **cluster** is composed of a set of Oracle Endeca Servers, each of which is hosting a cluster node. All servers can serve query requests. The Dgraph process on one of the servers is identified as the leader node; Dgraph processes on all other servers are follower nodes. All of the nodes share and use one copy of the on-disk representation of the data files for the data store.

A **cluster node** represents one Dgraph process for a specific data store running in the Oracle Endeca Server.



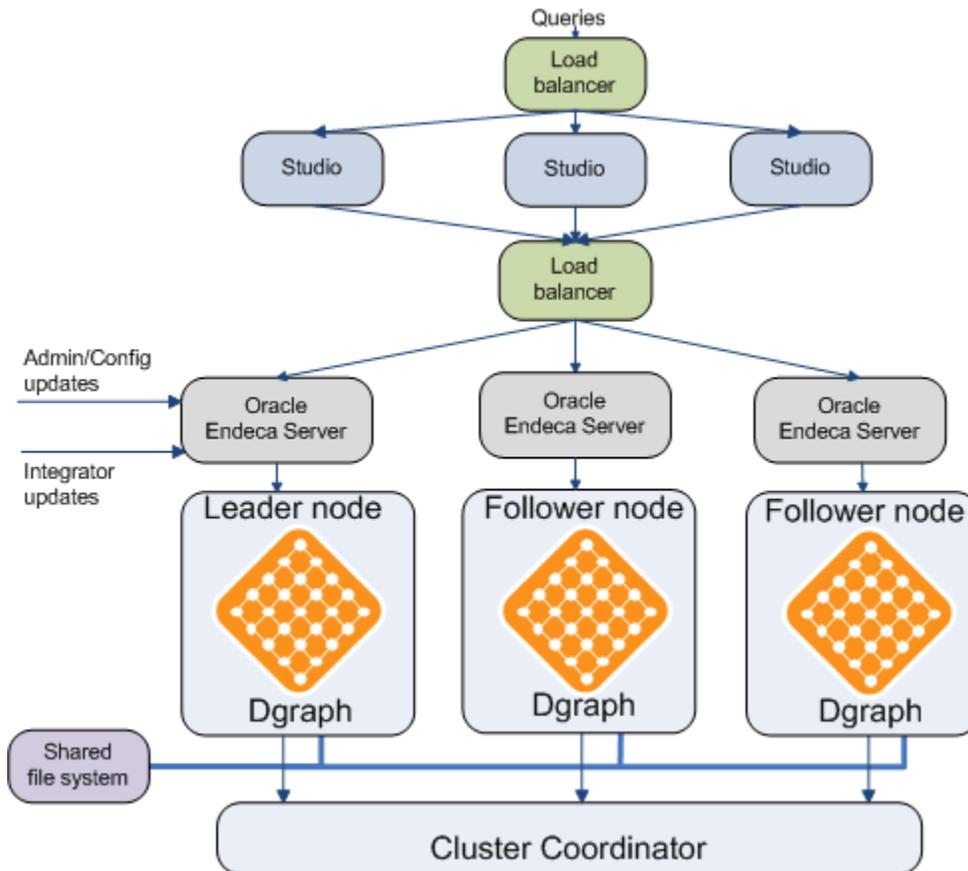
**Important:** In a clustered environment, to ensure adequate performance and ease of cluster configuration, it is recommended to configure a cluster with each Oracle Endeca Server running a single Dgraph process for the same data store as the Dgraph process hosted on each other Oracle Endeca Server.

The Cluster Coordinator provides communication between the nodes in the cluster. The Cluster Coordinator is also used to notify the follower nodes about data updates and updates to the configuration.

In a cluster, it is important to understand the roles played by the Oracle Endeca Server and the Dgraph process:

- The Oracle Endeca Server is installed on each physical server that will constitute a cluster node. Each cluster node is represented by a single Dgraph process and its corresponding data store that are started on the Oracle Endeca Server. The data updates, the queries, and the configuration changes are sent to the Oracle Endeca Server that is hosting the leader node.
- The single Dgraph process on each of the servers is the entity that functions as the cluster node.

- You configure a cluster at the level of Dgraph processes — indicating to the Cluster Coordinator which Dgraph process is going to be the leader node. The Dgraph processes have access to the same file system on which data files for the data store are shared between all the nodes.



## Cluster capabilities

A cluster of nodes provides the following capabilities:

- Enhanced availability of query processing by the Oracle Endeca Server.** In a cluster, if one of the nodes fails, queries continue to be processed by other nodes.
- Increased throughput by the Oracle Endeca Server.** In a cluster, you change throughput capacity by adding or removing physical servers. By adding nodes you can spread the query load across multiple Oracle Endeca Servers without the need to increase storage requirements at the same rate. You can add or remove nodes dynamically, without having to stop the cluster.

In a cluster, you can perform the following administrative tasks:

- Add one or more nodes to a cluster.
- Remove nodes while allowing the cluster to continue running.
- Identify a single Oracle Endeca Server to which you can send data during an initial data load and subsequent updates. (The administration and configuration updates must also be sent to this server.) All types of updates are automatically propagated to all nodes while they continue to process queries.

## Oracle Endeca Server cluster architecture

This topic discusses cluster architecture in the development and production environments.

In the development environment, you can start with a simple single-node cluster configuration and expand it by adding more nodes.

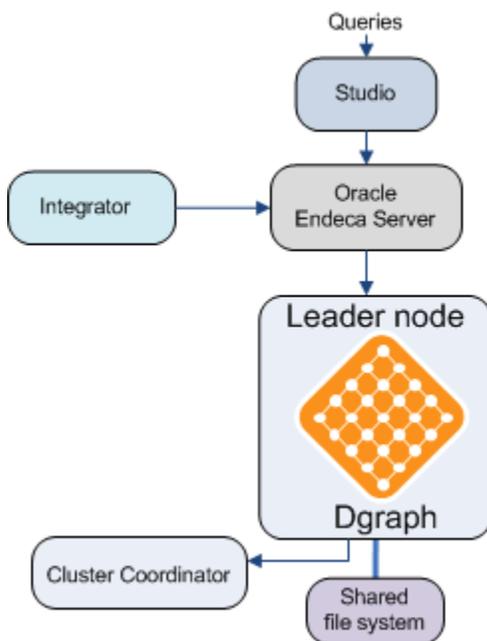
### A single-node cluster in the development environment

In a development environment, the simplest version of a cluster may consist of just one server hosting a Dgraph process for a data store, thus representing a single node. This node is by definition the leader node — in a single-node cluster, the only node is considered the leader node by default.



**Note:** When running a single Oracle Endeca Server, you are not required to have a cluster. Without the cluster services, having a single running Oracle Endeca Server is a valid configuration for starting in the development environment.

This diagram represents a single-node cluster in a development environment:



In this diagram:

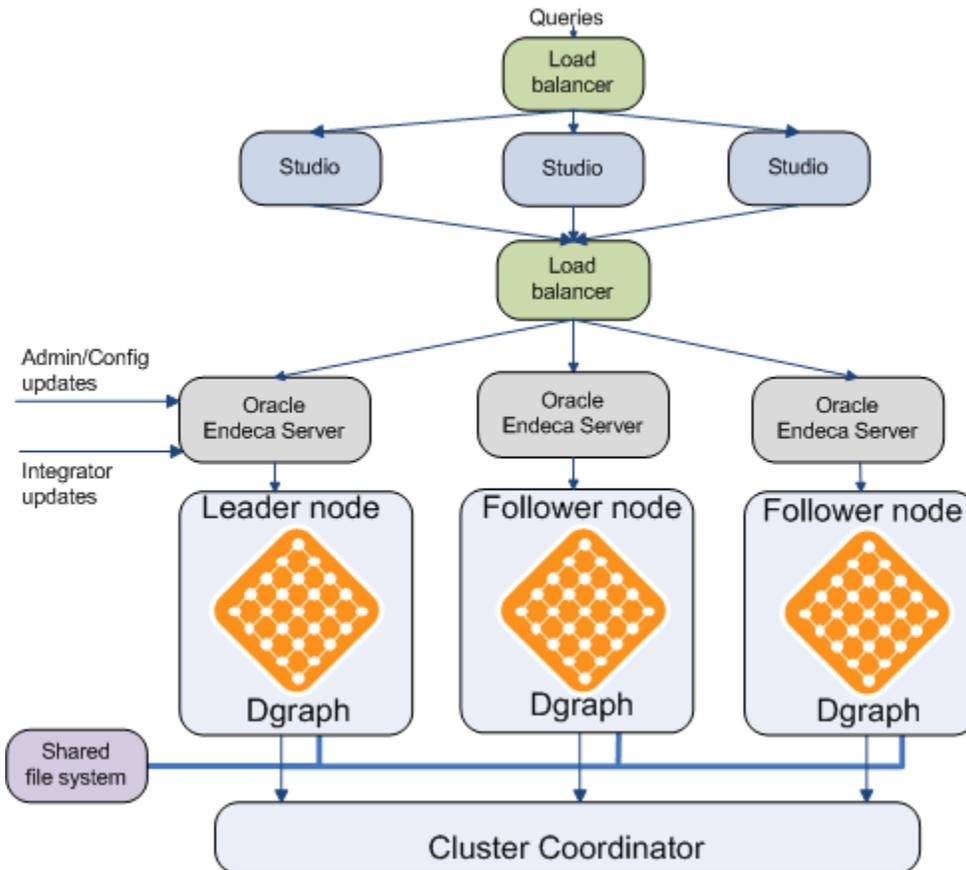
- The Oracle Endeca Server is installed and is hosting a single data store with the Dgraph process running. The Oracle Endeca Server is receiving query requests from Studio (or any other front-end application that is powered by the Oracle Endeca Server).
- The Oracle Endeca Server's host and port are included in the configuration for connectors from Integrator which can send various kinds of data and updates to the Oracle Endeca Server.
- The Dgraph process that is designated as the leader node must have write access to a shared file system on which the data for the specific data store is maintained. All other follower nodes that you add later must have read access to this file system.

- Finally, the cluster is managed by the Cluster Coordinator. In a single-node cluster, the Cluster Coordinator service runs on the same server on which the Dgraph process is running.

## A multiple-node cluster in the development environment

In the development environment, many cluster configurations are possible; they depend on the requirements for your application. For example, while a single leader node is always required, the number of additional follower nodes (represented by the Oracle Endeca Server that is hosting the same data store and running the Dgraph process for it) may vary.

This diagram represents a possible multiple-node cluster in a production environment:



In this diagram, starting from the top, the following actions take place:

- The queries are sent to the load balancer that is configured in front of several servers hosting Studio instances.
- The instances of Studio point to a second load balancer between Studio and the Oracle Endeca Server cluster.
- Integrator is configured to communicate with the host and port of the Oracle Endeca Server hosting the leader node to ensure a point of communication for sending data and updates.
- All nodes in the cluster (Dgraph processes) communicate with each other through the Cluster Coordinator. The Cluster Coordinator service must be running on the leader node.

- All nodes in the cluster have access to a shared file system on which data files for the data store are stored.

## A multiple-node cluster in the production environment

When you move to a production environment, you can duplicate a multi-node development cluster.

## Important cluster concepts

This topic introduces the leader and follower nodes and the Cluster Coordinator.

In a cluster composed of nodes, each of which runs an Oracle Endeca Server that is hosting a specific data store powered by the Dgraph process, the following definitions are used:

- Leader node
- Follower node
- Cluster Coordinator

### Leader node

The **leader node** is a single Dgraph process in the cluster responsible for processing queries and for receiving updates to the data files and to the configuration. This node is responsible for obtaining information about the latest versions of the data files and propagating this information to the follower nodes through the Cluster Coordinator.

When you create a new cluster, you start the leader node first and then add follower nodes. The leader node has the following characteristics:

- Each cluster must have one and only one leader node.
- The leader node must have write access to the same shared file system on which the data files are stored and to which all follower nodes also have access.
- The Cluster Coordinator service must be running on the leader node.
- The entities outside the cluster (such as connectors in Integrator and components of Studio) must be able to access the Oracle Endeca Server that is hosting the Dgraph process for the leader node.

The leader node periodically receives full or incremental data files updates from Integrator. It also receives administration or configuration updates. It is the only node in the cluster that has access with write permissions to the on-disk representation of the index.

Once the leader node acquires access to the new version of the data files, it updates the data files, adding new information and deleting information that has become obsolete. The Cluster Coordinator notifies all follower nodes, alerting them to start using the updated version of the data files. The follower nodes acquire read-only access to an updated version of the data files.

### Follower node

A **follower node** is a node in the cluster responsible for processing queries. The follower node does not update the data files, although it has read-only access to their latest copy.

You can start a follower node after you have started the leader node and the Cluster Coordinator. The follower node has the following characteristics:

- Each cluster can have more than one follower node.
- Each follower node must have a unique name across the cluster. The name also must be a valid directory name (characters such as slashes (/) are not allowed).
- All follower nodes must reference the host name and port of the Cluster Coordinator service.
- All follower nodes must have read-only access to the same shared file system on which the data files are stored. (The leader node must have write access to the file system.)

During the process of acquiring access to the recently updated data files, both the follower and the leader nodes continue to serve queries. Each query is processed against a specific version of the data files that is available to a cluster node at any given time. Query processing performance may slow down as the follower nodes acquire read-only access to the updated data files.

### ***Cluster Coordinator***

An entity that provides a mechanism for the nodes to communicate with each other.

#### *Cluster behavior*

## **Before you begin**

This section discusses requirements for installation related to deploying a cluster, as well as tips for planning your cluster architecture.

### **System and hardware requirements**

This section outlines the operating system and hardware requirements for deploying Oracle Endeca Servers in a clustered environment.

#### **Operating system requirements**

A cluster of nodes can be deployed on either Windows or Linux.

You cannot create a cluster in which some nodes are running on Windows while other nodes are running on Linux.

#### **Shared file system requirements**

This topic describes the requirements for the shared file system in a cluster.

- All nodes in the cluster must have access to a shared file system on which the data files are stored. The leader node must have write access, and the follower nodes must have read access.  
On Windows, it is recommended to utilize a file system that uses the CIFS (also known as SMB) protocol. On Linux, it is recommended to use NFS.
- File system size. You can start a cluster with a single node that serves both as the leader and a follower node. As you add additional follower nodes, file system size requirements (as measured by the high-water

mark parameters for shared storage) increase modestly and do not increase proportionally to the number of follower nodes.

- Performance. Even in a single-node cluster, storing data files on remote disks affects node startup time and performance associated with processing of data files updates. In a multi-node cluster, all nodes are accessing the data files at the same time. This coordinated access may affect performance for the network or shared file system, especially when large updates are accessed for the first time.

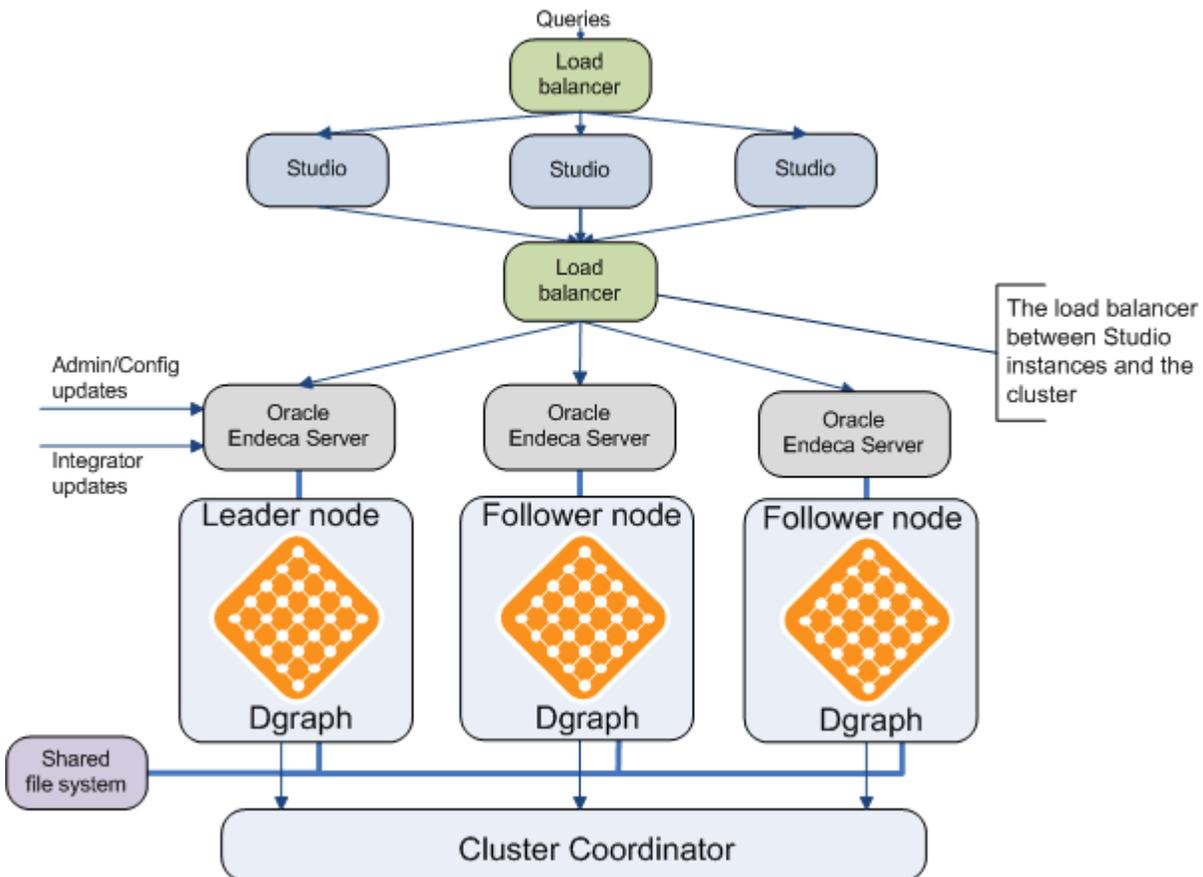
### Load balancer requirements

In most production deployments, it is desirable to configure a load balancer in front of the cluster. This topic discusses the considerations for this load balancer.

The following diagram shows a typical configuration for a multi-node cluster. This configuration contains two load balancers:

- One in front of the servers hosting Studio
- One between the Studio servers and the cluster

This topic discusses the load balancer in front of the cluster (the load balancer with the callout):



For this load balancer, the following considerations apply:

- Include host names and ports of all Oracle Endeca Servers in the load balancer configuration, including the name and port of the leader node. This ensures that regular query-type (non-updating) requests from

Studio are sent to any of the nodes in the cluster. Updating requests from Studio for changing the configuration should be directed to the leader node.

If you add nodes to the cluster, you must update the configuration of the load balancer with the host names and ports of the added Oracle Endeca Servers.

For the duration of an outer transaction (which may include updates sent to the leader node), the load balancer continues sending requests to all nodes in the cluster, including the leader node.

- You may optionally configure the load balancer to use session affinity. In this case, all queries from a given session end up on the same Oracle Endeca Server. This allows the Oracle Endeca Server to use its cache to avoid redundant processing on related queries.

Configuring session affinity also helps minimize consistency problems as updates propagate from the leader to the follower nodes in the cluster (if you are not using outer transactions to run updates).

For information on configuring the data sources in Studio that allow access to any node in the cluster through a load balancer, and access to the leader node for sending updating requests, see the *Oracle Endeca Information Discovery Studio User's Guide*.

## About the Cluster Coordinator

The cluster coordinator provides a mechanism for the nodes (represented by the Dgraph processes running on each of the servers) to communicate with each other while ensuring increased availability of the Oracle Endeca Server.

The Cluster Coordinator has the following characteristics:

- It ensures communication between Dgraphs. For example, in a two-node cluster, the communication is between a Dgraph process on one Oracle Endeca Server and a Dgraph process on another Oracle Endeca Server (each of these Dgraph processes represent cluster nodes).
- It is a shared information repository that provides a set of distributed coordination services.

It ensures that all systems in the cluster coordinate their actions relative to all other systems running in your environment. If one of the nodes communicates any cluster information, all other nodes recognize it and react to it in a manner that ensures synchronization, event notification, and coordination between the nodes.

The communication and coordination mechanisms continue to work in the case when connections or cluster nodes fail.

- The Cluster Coordinator logs messages using the `log4j` file included with its installation in the `/conf` directory. When the Cluster Coordinator service is running on a node, log messages can be logged to the console (default) and/or a log file depending on the `log4j` configuration.

On Linux, when you run the `clusterCoordinator` script, all output is by default sent to the `zookeeper.out` file stored in the current directory. You can override the destination of the log file by setting the environment variable `ZOO_LOG_DIR` before running the script. On Windows, when you run the `clusterCoordinator` script, all output is logged in a separate command window.



**Note:** The Endeca Cluster Coordinator service cannot be configured to run as a Windows service.

## Deployment strategy for the Cluster Coordinator service and nodes

The Cluster Coordinator service must be running on at least one node before starting the Dgraph process and its data store on any of the physical servers.

It is recommended that you start a single Cluster Coordinator service on the node that will be designated as the leader node, and then start the Dgraph processes on all servers by referencing the host and port of the Cluster Coordinator.



**Note:** The port for the Cluster Coordinator is a dedicated port used for cluster node communication.

## Starting and stopping the Cluster Coordinator service

To ensure that nodes can function together in a cluster, the Cluster Coordinator service must be running on the leader node and all the nodes must be started with references to the host name and port of the Cluster Coordinator service.

You start the Cluster Coordinator service on the leader node only.

It is assumed that the Cluster Coordinator package has been downloaded and installed on the server that hosts the Dgraph designated as the leader node.

To start or stop the Cluster Coordinator service:

1. On the server that will serve as the leader node, go to the Oracle\Endeca\ClusterCoordinator\*<version>*\bin directory and locate the clusterCoordinator script.

The script has a different extension for Windows and Linux.

2. From this directory, run the script as follows:

- To start, run:

```
clusterCoordinator start
```

- To stop, run:

```
clusterCoordinator stop
```



**Note:** If you provide a full path to the script on the command line, you can also run it from another directory. In this case, the Cluster Coordinator creates its `dataDir` in the path from which you run the script.

When the Cluster Coordinator is started on a server, it uses the host name of this server, and the port 2181. (You can change the Cluster Coordinator configuration to use another port, using its configuration file.)

After the Cluster Coordinator service has been started on a node, you can start the data store (and its associated Dgraph process) on this node. This will be the leader node.

## Configuration file for the Cluster Coordinator

The Cluster Coordinator service uses a configuration file to specify its settings.



**Note:** This topic provides reference information about this file. For the cluster to work, this file does not require any modifications.

After the installation of the Cluster Coordinator, the configuration file is placed in the `Oracle\Endeca\ClusterCoordinator\<version>\conf` directory. When you run the `clusterCoordinator` script, it detects the configuration file.

## Format

The configuration file should have the following format:

```
# The number of milliseconds of each tick
tickTime=2000
# The number of ticks that the initial
# synchronization phase can take
initLimit=10
# The number of ticks that can pass between
# sending a request and getting an acknowledgment
syncLimit=5
# the directory where the Cluster Coordinator snapshot is stored.
dataDir=.
# the port at which the clients will connect
clientPort=2181
```

Where:

Entry	Description
<code>tickTime</code>	The basic time unit in milliseconds used by the Cluster Coordinator. It is used for heartbeats.  For example, <code>tickTime</code> can be 2000 milliseconds. The minimum session timeout is twice the <code>tickTime</code> .
<code>initLimit</code>	The number of ticks that the initial synchronization phase can take.  This number specifies the length of time the nodes have to connect to the leader node.
<code>syncLimit</code>	The number of ticks that can take place between one node sending a request for an update and receiving an acknowledgment from the leader node.
<code>dataDir</code>	The directory where the in-memory database snapshots for the Cluster Coordinator and the transaction log of updates to its database are stored.  This directory is created in the same file system location from which you run the Cluster Coordinator script.  You can specify to store the log of updates to the database in another directory, using the <code>log4j</code> file stored in the <code>/conf</code> directory of the Cluster Coordinator installation.
<code>clientPort</code>	The port at which clients should connect to the Cluster Coordinator service.  As configured after the initial installation, this port is 2181.

## Planning cluster nodes

To plan cluster nodes, you specify which ones will serve as follower nodes. The one node in a cluster that is not a follower is the leader node.

To plan your cluster nodes:

1. Write down the host and port of each of the Dgraph processes for each of the Oracle Endeca Servers that will host your data store.

You will need this information to identify which of your nodes should serve as a leader node and to designate other nodes as follower nodes.

2. Provision a shared file system on which the data files for the data store will be stored.

When you will install the Oracle Endeca Server and start the data store on each of them, they should each point to the location of the data files (for the same data store) on this file system and have access to these files. (Read-only access is sufficient for follower nodes; write access is required for the leader node.)

3. On the node you would like to designate as the leader node, reserve the port 2181.

This port is used by the Cluster Coordinator service (unless you configure it to use another port). You will also specify this port when starting all nodes in your cluster.

## Cluster behavior

The Cluster Coordinator ensures that the nodes (represented by the Dgraph processes) in the cluster provide query processing that is stable in the face of individual follower node failures. This topic discusses cluster behavior in various scenarios, such as cluster startup, updates to the data files, and response to a node failure.

## Bringing a cluster online

On startup, the following actions take place:

- Any node can be started in either a leader or follower mode. The leader node must be started first. Any number of follower nodes and exactly one leader node can be added to a cluster.
- If you attempt to start two leader nodes in the same cluster, you will receive an error.
- If you attempt to start a follower node before the leader node has been started, the follower node will issue an error and will not start until the leader node is started.
- Once started, each node registers with the Cluster Coordinator that manages the distributed state of the cluster.

One node for which you do not specify that it must be a follower is the leader; you identify all other nodes as follower nodes.

- The leader node determines the current version of the data files and informs the Cluster Coordinator.
- Once the leader node has been started, the follower nodes can be started.
- After all follower nodes have started, each of them acquires read-only access to the current version of the data files.

- Follower nodes do not alter the data files in any way; they continue answering queries based on the version of the data files to which they have read-only access at startup, even if the leader node is in the process of updating, merging, or deleting data files on disk.

Follower nodes refuse all updating web service and HTTP requests (such as `admin?op=updateaspell`) with a 403 HTTP status code (forbidden).

## Processing an update

In a cluster, updates to the records in the data files (or any other updates) are sent only to the Oracle Endeca Server that is hosting the leader node Dgraph process.

The leader node processes the update and commits it to the on-disk data files. The Cluster Coordinator informs all follower nodes that a new version of data files is available. The leader node and all follower nodes can continue to use files from the previous version of the data files to finish query processing that had started against that version.

As each node finishes processing queries on the previous version, it releases references to it. Once the follower nodes are notified of the new version, they acquire read-only access to it and start using it.

It is recommended to wrap updates in the cluster in an outer transaction operation, although this is optional.

## Responding to a node failure

In a cluster, a follower or a leader node may fail:

- Failure of the leader node. When the leader node goes offline, updates do not occur on the leader node, and are not propagated to the follower nodes until the leader comes up again. However, follower nodes continue maintaining a consistent view of the data and answering queries.

For updates to resume, the Dgraph process on the leader node must be restarted. If the Dgraph process is running as a service, you can set it to restart automatically. Once restarted, the leader node joins the cluster and becomes operational, and the updates start to propagate from this node to all other nodes.

- Failure of a follower node. When one of the follower nodes goes offline, it is removed from the cluster. The other nodes do not need to keep track of this event. If the follower node is restarted, it joins the cluster. The follower node should be restarted with the same name.

## Responding to network failures of the Cluster Coordinator service

If a network connection fails between the nodes in the cluster that connect to the Cluster Coordinator service, the Dgraph process on those nodes will shut down.

A node will rejoin the cluster once the Dgraph process on the leader node is restarted (this will happen automatically if it is run as a service) and is able to establish a connection with the Cluster Coordinator service.

## Responding to a Cluster Coordinator failure

The Endeca Cluster Coordinator service cannot be configured to run as a Windows service. This means that if you are using the Endeca Clustering feature and run the Endeca Server as a Windows service, you should closely monitor the state of the Cluster Coordinator service.

The reason is that if the Endeca Server service crashes, the Windows Service Controller will automatically restart it. However, if the Cluster Coordinator service crashes, then it is not automatically restarted. This can lead to a situation where the Dgraph processes are running but the Cluster Coordinator service is not.

## Building a cluster

This section discusses how to build a cluster by starting the leader node and adding follower nodes.

### Starting the Dgraph process as the leader node

When you start the data store without specifying a follower node flag for the Dgraph process, this node serves as the leader node.

The leader node must be started first (before any of the follower nodes have been started).

Before starting the leader node, ensure that the Cluster Coordinator service is running on this node. Note the host name and port of your Cluster Coordinator service, so that you can specify them when starting the Dgraph as the leader node.

You configure one and only one leader node in a cluster. Therefore, if you want to start the Dgraph process as the leader, do not specify the `--follower` flag for it. The Dgraph process instance that is started without the `--follower` flag becomes the leader node. You also do not specify a name for the leader node.

To create the data store and start the Dgraph process as the leader node:

1. Go to the directory `Oracle\Endeca\Server\version\endeca-cmd`.

You can now start the data store from this directory. Alternatively, use Environment Variables to include the `.jar` file located in this directory in your full file path.

2. Create the data store and start the Dgraph process on the node with the `create-ds` command as in the following example:

```
endeca-cmd create-ds MyDataStore
--ws-port 5555
--data-files z:\shared_MyDataStore\MyDataStore
--args
  --coordinator_port 2181
  --coordinator_host My_cluster_coordinator_server.com
  --threads 16
  --log c:\MyDataStore\dgraph1.log
  --out c:\MyDataStore\dgraph1.out
```

In this example:

- `MyDataStore` is the name of this Endeca data store (which will be the leader).
- `5555` is the Dgraph process port. This is the port of the leader node.
- `z:\shared_MyDataStore\MyDataStore` is the location of the data files for this data store, which reside on a shared file system. All nodes in your cluster must point to the same location of the data files on a shared file system and have access to it (with the leader node having write access, and follower nodes having read access).
- The lines after the `--args` flag represent the flags that you are specifying for the Dgraph process hosted on this Oracle Endeca Server for your data store.

- 2181 is the port used by the Cluster Coordinator (if you haven't changed it after installing the Cluster Coordinator).
- `My_cluster_coordinator_server.com` is the host name used by the Cluster Coordinator. This is the host name of the leader node.
- Specifying the location for `--log` and `--out` is optional. If not specified, the default `endeca-server\logs` directory is used to store the logs.

Once the Dgraph process is running on the leader node, this node receives updates to the data files and configuration. The Cluster Coordinator propagates updates to the follower nodes.

3. Note the leader node's port and host name.

You will need to reference this information in the configuration for applications that need to send data and updates to the leader node, such as Integrator and Studio.

Now that you have started the leader node, you can add one or more follower nodes.

## Adding a follower node

You can add a follower node to the cluster after the leader node has been started, by specifying the `--follower` flag for the follower Dgraph process.

Before starting a Dgraph process that will serve as a follower node, ensure that the leader node has been started, and the Cluster Coordinator service is running on the leader node. Note the host name and port of your Cluster Coordinator service, so that you can specify them when starting the Dgraph process as the follower node.

The Dgraph flag `--follower <node_name>` specifies the follower node, where `<node_name>` is the name of the follower node. This name must be unique across the cluster. The name must also be a valid directory name (characters such as slashes (/) are not allowed).



**Important:** If you start a node without this flag, the Cluster Coordinator assumes this is the leader node. Since there can be only one leader node in the cluster, it is important to start just one node without the `--follower` flag. In fact, the Dgraph process will not start if it is asked to be the leader node when a leader node already exists.

Note that for follower nodes, you do not need to use the `create-ds` command to create the follower data store. Instead, you use the `attach-ds` command with a unique data store name, and use the same data files location as the leader node.

To attach the data store and start the Dgraph process as a follower node:

1. Go to the directory `Oracle\Endeca\Server<version>\endeca-cmd`.
2. Attach the data store on the node with the `endeca-cmd` command tool as in the following example, specifying a unique name of the follower node in the `--follower` flag for the Dgraph process:

```
endeca-cmd attach-ds MyDataStore2
--ws-port 5557
--data-files z:\shared_MyDataStore\MyDataStore
--args
--follower FollowerNode1
--coordinator_port 2181
--coordinator_host My_cluster_coordinator_server.com
--log c:\MyDataStore\dgraph2.log
--out c:\MyDataStore\dgraph2.out
```

In this example:

- `MyDataStore2` is the name of this Endeca data store (which will be a follower).
- `5557` is the Dgraph process port. This is the port of the follower node.
- `z:\shared_MyDataStore\MyDataStore` is the location of the existing data files for the leader data store, which resides on a shared file system. All follower nodes in your cluster must point to the same location of the data files on a shared file system and have read access to them.
- The lines after the `--args` flag represent the flags that you are specifying for the Dgraph process hosted on this Oracle Endeca Server for your data store.
- `FollowerNode1` is the name of the follower node.
- The `--coordinator_host` and `--coordinator_port` reference the host name and port of the Cluster Coordinator service. This service uses the host name of the leader node, and the port `2181` (unless you configure the Cluster Coordinator to use another port).
- Specifying the locations of the log and output files is optional.

The node `FollowerNode1` is now known to the Cluster Coordinator as a follower node — when changes occur to the data files stored on a shared disk, the Cluster Coordinator notifies this node to start using the new version of the data files.

3. Proceed by adding additional follower nodes if needed.

For each follower node:

- Specify (in the `attach-ds` command) a different name for the both the Endeca data store and the argument to the `--follower` flag.
- Reference the host name and port of the Cluster Coordinator service.
- Ensure that the location of the leader node's data files is referenced in the same way on a shared file system as for other nodes in the cluster.



**Note:** Since there is one Cluster Coordinator service running on the leader node, you can add more than one follower node, all referencing the same Cluster Coordinator service host name and port.

If a follower node fails, the cluster continues to run. Once you identify a follower node failure, restart the follower node with the same name and the node will join the cluster.

## Summary of operations handled by the leader node and any node

This topic summarizes which specific requests to the Oracle Endeca Server should be directed to the leader node and which can be handled by any node.

### Operations on the leader node only

These operations should be directed to the leader node only:

- Updates to data records. If you are adding more records to the application, they should be sent to the Oracle Endeca Server that is hosting the leader node.

This means that operations from the Data Ingest Web Service and the Bulk Load interface should be directed to the leader node only.

- Snapshot operations from the Administrative Web Service. Operations for taking and applying a snapshot should be directed to the leader node only.
- Updating operations from the Configuration Web Service. All requests to the Oracle Endeca Server that require changing schema for the records or the XML configuration documents should be directed to the leader node.
- The `rollbackOuterTransaction` operation from the Transaction Web Service should be run on the leader node.
- Administrative operations for the data store of the Oracle Endeca Server. The following administrative operations should be directed to the server that is hosting the Dgraph designated as the leader node:
  - `/admin/datastore?op=merge`
  - `/admin/datastore?op=reload-services`
  - `/admin/datastore?op=updateaspell`

## Operations on any node

The following operations can be directed to any node in the cluster (including any of the follower nodes):

- Any request from the Conversation Web Service (this means any request from Studio asking for read-only queries against the data).
- Any request from the Administrative Web Service other than snapshot-related operations.
- Any read-only request from the Configuration Web Service.
- Some administrative operations for the Oracle Endeca Server. The following administrative operations can be directed to any node in the cluster:
  - `/admin/datastore?op=flush`
  - `/admin/datastore?op=logroll`
  - `/admin/datastore?op=stats`
  - `/admin/datastore?op=statsreset`

## Connecting the leader node with Integrator

The connectors in Integrator that send data to the Oracle Endeca Server must be configured to reference the host name and port of the leader node.

The Dgraph process instance running on the leader node is capable of receiving updates, since it has write permissions to the shared file system hosting the data files for the data store.

It is assumed that by this point, you have configured a leader node in the cluster.

To reference the leader node in Integrator connector's configuration:

1. Specify the host name and port of the leader node.

## Maintaining a cluster

This section contains tasks you need to perform for cluster maintenance.

### Removing a follower node

To remove a follower node, stop the data store and its associated Dgraph process on this server with the `endeca-cmd stop-ds <datastore>` command. This command gracefully shuts down a running Dgraph process.

When you stop the Dgraph process and the data store on one of the follower nodes, this node is removed from the cluster.

### Changing the name of the leader node

To change the leader node, stop the existing cluster by stopping all Dgraph processes on each of the servers, start the Cluster Coordinator on another node, and recreate the cluster with the new leader node.

To change which node is the leader node:

1. Stop the existing cluster by stopping the nodes and the Cluster Coordinator service.
2. Start the Cluster Coordinator service on another node.
3. Recreate the cluster with another node as the leader, by referencing the new location of the Cluster Coordinator to all nodes.



## Chapter 12

# Configuring SSL

---

This section describes how configure Oracle Endeca Server to use SSL.

[About configuring SSL in Endeca Server](#)

[Configuring SSL on Jetty](#)

[Enabling SSL for the endeca-cmd interface](#)

[Configuring SSL on the Dgraph](#)

## About configuring SSL in Endeca Server

Configuring SSL in Endeca Server enables SSL communication among all the components.

Configuring SSL in Endeca Server comprises these high-level steps:

1. Enable SSL for the Endeca Server's Jetty application server.
2. Enable SSL for the Endeca Server's `endeca-cmd` command interface.
3. Enable SSL for the Endeca data store.

The steps are described in detail in later topics.

If you are running Studio, you can also enable SSL for Studio. For details, see the *Oracle Endeca Information Discovery Studio User's Guide*.



**Note:** Enabling SSL on the Endeca Cluster Coordinator service is not supported.

## Certificate-generating tools

There are a variety of tools that you can use to generate keys and certificates. One of the most common is the `keytool` utility.

Endeca Server ships with a copy of `keytool` in this default location:

- Windows: `C:\Oracle\Endeca\Server\<version>\shared\jre\bin`
- Linux: `Oracle/Endeca/Server/<version>/shared/jre/bin`

Full documentation for the Java 6 version of `keytool` (for Windows) is available here:

<http://docs.oracle.com/javase/6/docs/technotes/tools/windows/keytool.html>

Alternatively, you can use the `enecerts` utility described later in this chapter, in [Using enecerts to generate SSL certificates on page 85](#).

## Default cacerts certificates file

A certificates file named `cacerts` is shipped in the `shared/jre/lib/security` directory. The `cacerts` file represents a system-wide keystore with CA certificates. The default password of the `cacerts` file is `changeit`. You should change the default password if you intend to use the file. You can configure and manage the file with the `keytool` utility, specifying `jks` as the keystore type.

The `cacerts` keystore file ships with several root CA certificates. The Web page accessed by the link above lists the aliases and X.500 owner distinguished names in the file.

## Default Jetty keystore

Endeca Server ships with a default `keystore` file in the `Jetty etc` directory. For the keystore:

- `jetty` is the alias name.
- `storepwd` is the password.

The default `jetty-ssl.xml` configuration file comes pre-configured with this default keystore. You can use this keystore only for preliminary SSL testing. However, you should create your own keystore when you are ready to set up your SSL environment.

# Configuring SSL on Jetty

This section describes the tasks necessary to enable SSL on the Jetty application server.

## Creating a keystore

This topic describes how to create a JKS keystore.

The procedure uses the `keytool` utility to create a simple JKS keystore suitable for use with JSSE. Details of the keystore creation will vary depending on such factors as the tool you are using and the Certificate Authority who will sign the certificate.

This procedure assumes that you have added the `keytool` path to your `PATH` environment variable. This allows you to run the utility from anywhere on your machine. The procedure also assumes that you have created a directory (`C:\MyKeys` in this example) to create and store the keystore. Afterwards, you can copy it to the `Jetty etc` directory. (Alternatively, you can create the keystore in the `Jetty etc` directory.)

While creating the certificate, you will be asked to enter the Common Name for the certificate with this prompt:

```
What is your first and last name?
```

The Common Name is typically composed of Host + Domain Name. The Common Name must be the same as the Web address you will be accessing when connecting to a secure site. For the Endeca Server certificate, you can use the name of the server, including its full domain name. This procedure will use `app23.example.com` as the Common Name. After enabling SSL, you can specify the same Common Name with the `--host` option of the `endeca-cmd` commands.

To create a keystore:

1. From a command prompt, navigate to the `C:\MyKeys` directory.
2. Generate a private key and a self-signed public key, as in this sample command:

```
keytool -genkey -alias server -validity 365 -keyalg RSA -keystore keystore
```

The example uses the RSA algorithm and the keys will be stored in the `keystore` file. Note that you can use another name for your keystore file other than our `keystore` example).

3. Answer the `keytool` DN prompts:

```
Enter keystore password: strongKeystorePassword
Re-enter new password: strongKeystorePassword
Or
What is your first and last name?
[Unknown]: app23.example.com
What is the name of your organizational unit?
[Unknown]: Apps Department
What is the name of your organization?
[Unknown]: example.com
What is the name of your City or Locality?
[Unknown]: Cambridge
What is the name of your State or Province?
[Unknown]: Massachusetts
What is the two-letter country code for this unit?
[Unknown]: US
Is CN=app23.example.com, OU=Apps Department, O=example.com, L=Cambridge,
ST=Massachusetts, C=US correct?
[no]: yes

Enter key password for <server>
(RETURN if same as keystore password): <RETURN>
```

When you answer the last prompt, `keytool` writes the `keystore` file in the current directory. The keystore contains a private key and a self-signed public key.

Note that although you have the minimal requirements to run an SSL connection, the certificate you have generated will not be trusted by most clients. Therefore, the next step is to obtain a trusted certificate by having the public key signed by a known Certificate Authority (CA).

4. Generate a Certificate Signing Request (CSR) with the `-certreq` option:

```
keytool -certreq -alias server -keyalg RSA -file endeca.csr -keystore keystore
```

When asked for the keystore password, enter the password you specified at Step 3.

5. Send the CSR to a Certificate Authority (CA) for signing.

The CA will send you a certificate file that bears the CA's signature.

6. Import the signed certificate into the keystore:

```
keytool -import -file rootCA.pem -keystore keystore -trustcacerts
```

As part of the `keytool` dialog, you will be asked for the password you specified at Step 3 and also if you trust the certificate:

```
Enter keystore password: strongKeystorePassword
...
Trust this certificate? [no]: yes
Certificate was added to keystore
```

7. If you created the keystore in a separate working directory, you can copy the keystore to the Jetty etc directory.

The next two steps are to obfuscate the keystore password and activate the SSL connector in the `jetty-ssl.xml` file.

## Obfuscating passwords

You should obfuscate the SSL connector keystore password for greater security.

When adding the `SslSelectChannelConnector`, you must specify the keystore passwords in the configuration. Specifying the passwords as clear text is not recommended because they are not secure. Instead, you should obfuscate the passwords so that they are not easily read.

The Jetty package includes a password utility that can generate secure passwords. The utility comprises two JAR files:

- `jetty-http-8.0.1.v20110908.jar`
- `jetty-util-8.0.1.v20110908.jar`

Note that the version number (8.0.1.v20110908) will change if the Jetty package is upgraded to a later revision.

To produce an obfuscated password:

1. From a command prompt, navigate to the Jetty `lib` directory.
2. On Windows, Issue the following command on a single line:

```
java -cp jetty-http-8.0.1.v20110908.jar;jetty-util-8.0.1.v20110908.jar  
org.eclipse.jetty.http.security.Password endeca
```

Note there is a space between the second ".jar" and the "org." that starts the package name. Use your own password instead of the "endeca" example.

On Linux, use a colon (:) instead of a semi-colon (;) to separate the two JAR names.

The output of the utility will look like this for the "endeca" password example:

```
endeca  
OBF:lsarl1uh61svy1sw0lugk1saj  
MD5:2a1ec2aefbd80e6043b394cb2314e9c6
```

The OBF output is the obfuscated version of the password, while the MD5 output is the checksummed version. You will be using the OBF version in the Jetty configuration file.

## Configuring Endeca Server Jetty files

You must modify Jetty files for SSL, as well as changing the start-up port for Endeca Server.

The pre-requisites for this task are that you must have created a keystore and generated secure passwords.

During the procedure, you will be modifying these files:

- `start.ini` (in Jetty root directory)
- `jetty-ssl.xml` (in Jetty `etc` directory)
- `jetty.xml` (in Jetty `etc` directory)
- `endeca-server.windows.conf.bat` on Windows or `endeca-server.linux.conf` on Linux (in `endeca-server` directory)

To modify Endeca Server's Jetty files for SSL:

1. Back up the three configuration files, in case you need to revert your changes.

- In the Jetty root directory, open the `start.ini` file and uncomment this line:

Before:

```
#etc/jetty-ssl.xml
```

After:

```
etc/jetty-ssl.xml
```

Uncommenting this line enables the `jetty-ssl.xml` configuration file to be used as part of the Jetty start-up process.

- In the Jetty `etc` directory, open the `jetty-ssl.xml` file and make these changes to the `configure` section for the `sslContextFactory` object:

sslContextFactory Property	Value
KeyStore	The name and path of the keystore file.
KeyStorePassword	The obfuscated password for the keystore.
KeyManagerPassword	The obfuscated password for the key manager.
TrustStore	The name and path of the truststore file.
TrustStorePassword	The obfuscated password for the truststore.
needClientAuth	Set to <code>true</code> if you are using mutually-authenticated SSL.

You can also set other configuration properties that are documented on this page:

[http://wiki.eclipse.org/Jetty/Reference/SSL\\_Connectors](http://wiki.eclipse.org/Jetty/Reference/SSL_Connectors)

At this point, the `sslContextFactory` configuration should look similar to this example:

```
<New id="sslContextFactory" class="org.eclipse.jetty.http.ssl.SslContextFactory">
  <Set name="KeyStore"><Property name="jetty.home" default="." />/etc/keystore</Set>
  <Set name="KeyStorePassword">OBF:1sarluh61svylsw0lugk1saj</Set>
  <Set name="KeyManagerPassword">OBF:1sarluh61svylsw0lugk1saj</Set>
  <Set name="TrustStore"><Property name="jetty.home" default="." />/etc/keystore</Set>
  <Set name="TrustStorePassword">OBF:1sarluh61svylsw0lugk1saj</Set>
  <Set name="needClientAuth">true</Set>
</New>
```

In the `addConnector` configuration section, make sure that you leave 8443 as the port setting.

- With the above changes, you would have an SSL connector on port 8443 and a non-SSL connector on the default 8080 port. If you want to have only the SSL connector active, you can disable the non-SSL connector by commenting out the `addConnector` section in the `jetty.xml` configuration file.
- In the `endeca-server` directory, open the Endeca Server start-up configuration file (`endeca-server.windows.conf.bat` on Windows or `endeca-server.linux.conf` on Linux) and set the `ENDECA_SERVER_PORT` variable to 8443:

For Windows:

```
set ENDECA_SERVER_PORT=8443
```

For Linux:

```
ENDECA_SERVER_PORT=8443
```

## 6. Re-start Endeca Server.

The next step is to configure SSL for the `endeca-cmd` interface.

## Enabling SSL for the `endeca-cmd` interface

You must use `endeca-cmd` interface global options to specify the location of the keystore and truststore files.

To enable SSL for the `endeca-cmd` interface:

1. Generate a keystore and truststore.

If you have already generated them for the Jetty application server, you can use them for the `endeca-cmd` interface.

2. When issuing an `endeca-cmd` interface command, use the `--keystore` and `--truststore` global options.

For information on these options, see [Global options for SSL on page 16](#).

You will also use the `--port` global option to specify the Endeca Server's SSL port and the `--host` global option to specify the machine name.

The following example (on a Windows machine) shows the use of the SSL global options and the SSL dialog:

```
C:\Oracle\Endeca\Server\7.4.0\endeca-cmd>endeca-cmd status-ds books --port 8443
--host app23.example.com --keystore c:\mykeys\keystore --truststore c:\mykeys\truststore
key store password:
trust store password:
Current State: Stopped

Data Files: C:\Oracle\Endeca\Server\7.4.0\endeca-server\data\books
WS Port: 7773
Bulk Load Port: 7774
Startup Timeout (s): 60
Shutdown Timeout (s): 60
```

After the command is issued, the user is asked for the keystore and truststore passwords. If an incorrect password is entered, the command fails with this error message:

```
Keystore was tampered with, or password was incorrect
caused by:
Password verification failed
```

## Configuring SSL on the Dgraph

This section describes how to configure SSL on the Dgraph.

The section also describes how use the `enecerts` utility of the Dgraph to generate standard and custom SSL certificate files. That is, you can use the `enecerts` utility as an alternative to the `keytool` utility described in the topic [Creating a keystore on page 80](#).

## Communications between Endeca Server and the Dgraph

When configuring SSL on the Dgraph, keep in mind how the Oracle Endeca Server communicates with the Dgraph processes it controls. The Endeca Server has hostname validation explicitly hard-coded to be off when

using HTTPS between the Endeca Server and the Dgraph processes it is controlling. This is because all the URLs reference `localhost` since the Endeca Server and its Dgraph processes are always on the same machine. Therefore, the Endeca Server is hard-coded not to check that the hostname in the Dgraph's certificate matches the URL that it is talking to.

However, if you enable SSL on the Endeca Server, you should also enable SSL on the Dgraph. The main reason is that if the Dgraph is not secure, it can be an entry point for someone attempting to gain access to your application environment. Another use case is if you are accessing the Dgraph's Bulk Load Interface to load data from an SSL-enabled client (such as Integrator).

## Certificate files used by components connecting with the Dgraph

You configure SSL among the components by using a set of certificate files.

The certificate files are listed in the following table:

Certificate file	Description
<code>eneCert.pem</code>	Certificate file used by all clients and servers to specify their identity when using SSL to connect to the Oracle Endeca Server. This certificate should be thought of as the identity of the system powered by the Dgraph, or as the identity of all components of the system.
<code>eneCA.pem</code>	Certificate authority file used by all clients and servers to authenticate the other endpoint of a communication channel.
<code>eneCA.key</code>	Private key that is used by the <code>enecerts</code> certificate authority program to sign the <code>eneCert.pem</code> certificate.
<code>eneCA.cer</code>	Certificate authority file.
<code>eneCert.p12</code>	Personal Information Exchange (PKCS12-format) key file.

Because these certificate files are not provided in the Dgraph package, you must use the `enecerts` utility available with the installation package to generate them.

## Using `enecerts` to generate SSL certificates

You can use the `enecerts` utility program to generate new SSL certificate files.

The two typical scenarios for generating SSL certificates are:

- You are setting up SSL for the first time and need to generate the set of standard certificates.
- You want to generate custom certificates, such as those with a private key size greater than the default 1024 bits.

The `enecerts` utility resides in the `bin` directory (located in the Dgraph root directory) under the name `enecerts` (`enecerts.exe` on Windows).

The `dgraph` directory is located in the `endeca-server` directory.

## Generating standard SSL certificates on Linux

This procedure shows how to generate the set of standard certificates with a 1024-bit private key size on Linux platforms.

To generate the SSL certificates on a Linux machine:

1. Make sure that the `bin` directory (located in the Dgraph root directory) is in your `$PATH` environment variable.
2. Change to the directory in which the certificate files should reside.
3. Run the `enecerts` utility that creates the certificates.
4. Enter an export password of your choice.

If the program finishes successfully, it displays the list of certificates that it generated.

## Generating standard SSL certificates on Windows

This procedure shows how to generate the set of standard certificates with a 1024-bit private key size on Windows platforms.

To generate the SSL certificates on a Windows machine:

1. Open a command prompt.



**Note:** Make sure you are using a new command prompt window, not one that is left over from earlier tasks.

2. Change to the directory in which the certificate files should reside.
3. Run the `enecerts` utility that creates the certificates:

```
enecerts
```

4. Enter an export password of your choice.

If the program finishes successfully, it displays the list of certificates that it generated.

## Generating custom certificates

You can use the `enecerts` utility to generate customized certificates.

You can generate two types of customized certificates by:

- Specifying a private key size larger or smaller than the default 1024-bit size.
- Using your own CA file and private key to generate the `eneCert.pem` certificate.

The next two sections describe these operations.

## Specifying a different certificate key size

The `--keysize` flag of the `enecerts` utility lets users specify the size of the generated private key. The flag syntax is:

```
--keysize bits
```

where *bits* is the private key size in bits (default value is 1024).

For example, the following Windows command creates certificates with a private key size of 2048 bits:

```
enecerts --keysize 2048
```

## Using your CA file to generate certificates

By default, the `enecerts` utility produces the `eneCert.pem` certificate (used by all clients and servers to specify their identity when using SSL) and the `eneCA.pem` CA certificate (used by all clients and servers that wish to authenticate the other endpoint of a communication channel).

If you have your own CA certificate and private-key files, you can use the `--CAkey` and `--CAcert` flags to generate the `eneCert.pem` certificate. The private-key file (.key extension) is used to digitally sign the public key that is generated by the `enecerts` utility. Both flags must be used for this operation.

The syntax for the `--CAkey` flag is:

```
--CAkey private-key
```

where *private-key* is your own .key file with the private key for the CA that should be used to sign the generated certificate.

The syntax for the `--CAcert` flag is:

```
--CAcert cert-pem
```

where *cert-pem* is your CA certificate (.pem extension). This file is the same type of file as the default `eneCA.pem` CA certificate.

For example, the following Windows command creates a signed certificate file using your own CA certificate and private-key files:

```
enecerts --CAkey myCA.key --CAcert myCA.pem
```

You would then use the resulting `eneCert.pem` certificate and your CA file (`myCA.pem` in the example) to configure SSL for your Endeca components. If you have multiple machines in your deployment, you must also copy these files to the other machines.

## Copying the SSL certificates to other machines

All machines that are running your deployment must use the same SSL certificates.

If you have multiple machines in your deployment, the standard or custom SSL certificates should be created only once, on one machine. You must then copy them to the directories (on all other machines) from which the Dgraph is started. All of the machines must use the same SSL certificates.

## Configuring the Dgraph for SSL mutual authentication

This topic describes high level steps required to configure an SSL mutual authentication between the Dgraph and an external machine. The authentication uses certificates signed by a certificate authority (CA). This setup may apply if your Dgraph and external machines are hosted outside the firewall, or if a two-way authentication is required between them.

An SSL-enabled client (such as the Integrator component of Oracle Endeca Information Discovery) may need to access the Dgraph securely. In such cases, a secure connection must be established between these servers by configuring the Dgraph for authentication with SSL certificates.

This procedure is an example of how you can establish a mutual (two-way) authentication. Treat this procedure as a high-level recommendation rather than the only way to establish a secure connection. Other steps may be required depending on your specific security requirements.

In this procedure, you create two signed certificates. First, you create a private key and send a Certificate Signing Request (CSR) to a CA from the external server. Next, you create a private key and send a CSR from the Dgraph. You can then start the Dgraph referencing the `sslcertfile`, which contains the Dgraph private key and the signed certificate.

To configure an SSL mutual authentication between the Dgraph and an external client:

1. Create a private key and send a Certificate Signing Request (CSR) from the external server. You can create a private key and issue a CSR by using one of these methods:
  - Use the `keytool` utility shipped with Endeca Server. For details, see [Creating a keystore on page 80](#).
  - Use `openssl` commands.
  - Consult your security and server administrator for assistance.



**Note:** Some CA vendors require that the CSR be generated from 2046-bit length private keys and not from 1024-bit length keys. Please confirm with your CA vendor before issuing the CSR.

2. Send the CSR to a CA for signing.

A CA provides a bundled key file (including intermediate keys) along with a signed certificate.



**Note:** You will need the bundled key file for the Dgraph `--sslcafile` startup flag later on in this procedure.

3. Add the signed certificate to the keystore of the external client.  
For information, refer to the client's documentation or your security administrator.

4. Create a private key and certificate for the Dgraph using the `openssl` utility.

For example, the following command creates a 2048-bit RSA key that is valid for a year:

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout MyCert.pem -out MyCert.pem
```

The resulting `MyCert.pem` file stores both the private key and the certificate.

5. Create a Certificate Signing Request (CSR) from the `MyCert.pem` file, as follows:

```
openssl req -new -key MyCert.pem -out MyCertCSR.pem
```

6. Send the `MyCertCSR.pem` file to the CA for signing.
7. Obtain from the CA a bundled key file (including intermediate keys) along with a signed certificate.
8. Create an empty file, such as `MySSLCert.pem`, to store the combination of the Dgraph private key and the signed certificate.

You can do this by copying the entry for the private key (step 4) into the empty file, and appending the contents of the signed certificate (Step 7) underneath the private key entry in the new file.

9. Reference the file `MyCert.pem` created in the previous step in the `--sslcertfile` startup flag for the Dgraph.
10. Add both the `sslcafile` and `sslcertfile` flags to your Dgraph startup options, as follows:

```
--sslcafile <full_path_to_location_of_bundled_key_file_in_step2>
--sslcertfile <full_path_to_location_of_MySSLCert.pem>
```

11. (Optional) If the external client requires the bundled keys for the Dgraph that you obtained in step 7, add them accordingly to its keystore.

## Converting PEM-format keys to JKS format

This topic describes how to convert PEM-format certificates to the standard Java KeyStore (JKS) format.

The Java KeyStores can be used for communication between components that are configured for SSL (for example, between Studio and the Oracle Endeca Server, if both are SSL-enabled).

Two utilities (located in Endeca Server directories) are referenced in the instructions below:

- `openssl` (located in the `endeca-server/dgraph/bin` directory).
- `keytool` (located in the `shared/jre/bin` directory).

This procedure assumes the following:

- You have set your path environment variable to add Dgraph utilities directory and the Dgraph binaries to the search path, to allow you to run the `openssl` utility from the directory of your choice.
- Your path will allow you to use the `keytool` utility from the directory of your choice.
- You have already generated the set of standard SSL certificates with the `enecerts` command, as documented earlier in this section.
- All of the input files are located in the local directory.

To convert the PEM-format keys to Java KeyStores:

1. Convert the certificate from PEM to PKCS12, using the following command:

```
openssl pkcs12 -export -out eneCert.pkcs12 -in eneCert.pem
```

You may ignore the warning message this command issues.

2. Enter and repeat the export password.
3. Create and then delete an empty truststore using the following commands:

```
keytool -genkey -keyalg RSA -alias endeca -keystore truststore.ks
keytool -delete -alias endeca -keystore truststore.ks
```

The `-genkey` command creates the default certificate shown below. (This is a temporary certificate that is subsequently deleted by the `-delete` command, so it does not matter what information you enter here.)

```
Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]:
What is the name of your organizational unit?
  [Unknown]:
What is the name of your organization?
  [Unknown]:
What is the name of your City or Locality?
  [Unknown]:
What is the name of your State or Province?
  [Unknown]:
What is the two-letter country code for this unit?
```

```
[Unknown]:
Is CN=Unknown, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown correct?
[no]: yes

Enter key password for <endeca>
(RETURN if same as keystore password):
Re-enter new password:
```

4. Import the CA into the truststore, using the following command:

```
keytool -import -v -trustcacerts -alias endeca-ca -file eneCA.pem -keystore truststore.ks
```

5. Enter the keystore password).
6. At the prompt, "Trust this certificate?" type *yes*.
7. Create an empty Java KeyStore, using the following commands:

```
keytool -genkey -keyalg RSA -alias endeca -keystore keystore.ks
keytool -delete -alias endeca -keystore keystore.ks
```

The `-genkey` command creates the default certificate shown below. (This is a temporary certificate that is subsequently deleted by the `-delete` command, so it does not matter what information you enter here.)

```
Enter keystore password:
Re-enter new password:
What is your first and last name?
[Unknown]:
What is the name of your organizational unit?
[Unknown]:
What is the name of your organization?
[Unknown]:
What is the name of your City or Locality?
[Unknown]:
What is the name of your State or Province?
[Unknown]:
What is the two-letter country code for this unit?
[Unknown]:
Is CN="Unknown", OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown correct?
[no]: yes
```

8. Import your private key into the empty JKS, using the following command:

```
keytool -v -importkeystore -srckeystore eneCert.pkcs12 -srcstoretype PKCS12 -destkeystore
keystore.ks -deststoretype JKS
```



# Dgraph Flag Reference

This section provides a description of the flags (options) used by the Dgraph process.

## Dgraph flags

### Dgraph flags

The Oracle Endeca Server starts the Dgraph process for each Endeca data store.

When you create or attach an Endeca data store, you can optionally specify that its Dgraph process start with any of the flags in the following table. These Dgraph flags allow you to adjust its configuration.

To obtain a listing of the Dgraph process arguments at the Oracle Endeca Server host and port, issue the `create-ds` or the `attach-ds` command with the `--args --usage` flag, as in this example for the `create-ds` command:

```
endeca-cmd create-ds --args --usage
```

Flag	Description
?	Print the help message and exit.
-v	Verbose mode. Print information about each request to <code>stdout</code> .
--ancestor_counts	Compute counts for root managed attribute values and any intermediate managed attribute value selections.  By default, the Dgraph only computes refinement counts for proper refinements (in other words, for actual managed attribute values).  It does not compute counts for root managed attribute values or for any intermediate managed attribute value selections.
--backlog-timeout <seconds>	Specify the wait limit (in seconds) for a query that has been read and queued for processing.  This is the maximum number of seconds that a query is allowed to spend waiting in the processing queue before the Dgraph responds with a timeout message.  The default value is 0 seconds.

Flag	Description
<code>--cmem &lt;MB&gt;</code>	<p>Specify an absolute value in MB for the Dgraph cache.</p> <p>When an absolute value is not specified with the <code>--cmem</code> flag, the default Dgraph cache size is computed as 10% of the amount of RAM available in the system.</p>
<code>--coordinator_host &lt;host name&gt;</code>	<p>Specify the host name of the server on which the Cluster Coordinator service is running.</p> <p>You specify this flag along with the <code>--coordinator_port</code> flag when you start the Dgraph as one of the nodes in the cluster.</p>
<code>--coordinator_port &lt;num&gt;</code>	<p>Specify the port of the server on which the Cluster Coordinator service is running.</p> <p>The Cluster Coordinator expects that you specify the port 2181 (if you specify another port, changes to the Cluster Coordinator configuration file are required).</p> <p>You specify this flag along with the <code>--coordinator_host</code> flag when you start the Dgraph as one of the nodes in the cluster.</p>
<code>--disable_fast_aspell</code>	<p>Disable fast mode for the aspell spelling module. If you disable fast mode, it decreases the performance of the spelling correction, but may allow additional queries to be corrected.</p> <p>When the fast mode is enabled, it can significantly speed up applications that use spelling correction features with the aspell module. The fast mode is used by default.</p>
<code>--esampmin &lt;num&gt;</code>	<p>Specify the minimum number of records to sample during refinement computation. The default is 0.</p> <p>Tuning recommendations:</p> <ul style="list-style-type: none"> <li>• For most applications, larger values reduce performance without improving dynamic refinement ranking quality.</li> <li>• For some applications with extremely large, non-hierarchical managed attributes (if they cannot be avoided), larger values can meaningfully improve dynamic refinement ranking quality with minor performance cost.</li> </ul>

Flag	Description
<pre>--follower &lt;name&gt;</pre>	<p>Specify the name of the node that should serve as one of the follower nodes in the cluster. This name must be unique across the cluster, and must also be a valid directory name (characters such as slashes (/) are not allowed).</p> <p>You can start more than one node in the cluster with this command, thus designating more than one follower node.</p> <p>Before starting the Dgraph with this command flag, ensure that the Cluster Coordinator service is running on the server that serves as the leader node.</p> <p>All nodes must be able to connect to the Cluster Coordinator. Therefore, when you specify a follower node with the <code>--follower</code> flag, also specify for the follower node the host name and port of the Cluster Coordinator service using the <code>--coordinator_host</code> and <code>--coordinator_port</code> commands.</p> <p>For a list of operations allowed in the Dgraph when you use the <code>--follower</code> flag, see <a href="#">Deploying a Cluster of Oracle Endeca Servers on page 61</a>.</p> <p> <b>Note:</b> If you start a node without the <code>--follower</code> flag, the Cluster Coordinator assumes this is the leader node. Since there can be one and only one leader node in the cluster, the Dgraph will not start if it is asked to be the leader node when a leader node already exists.</p>
<pre>--help</pre>	<p>Print the help message and exit.</p>
<pre>--implicit_exact</pre>	<p>Disable approximate computation of implicit refinements.</p> <p>Use of this option is not recommended.</p> <p>If this option is not enabled, managed attribute values without full coverage of the current result record set may sometimes be returned as implicit refinements, although the probability of such "false" implicit refinements is minuscule.</p>
<pre>--implicit_sample &lt;num&gt;</pre>	<p>Set the maximum number of records to sample when computing implicit refinements (which are a performance tuning parameter). The default value is 1024.</p>
<pre>--latin1</pre>	<p>Ignore character accents when handling search requests, and use ISO Latin 1 character mappings when processing search requests.</p>
<pre>--log &lt;path&gt;</pre>	<p>Specify the path for the Dgraph request log file. The default log file is named <code>datastore.reqlog</code> and is located in the Endeca Server's <code>logs</code> directory.</p>

Flag	Description
<code>--net-timeout &lt;num&gt;</code>	Specify the maximum number of seconds the Dgraph waits for the client to download data from queries across the network. The default network timeout value is 30 seconds.
<code>--out &lt;stdout/stderr file&gt;</code>	Specify file path to which stdout/stderr should be remapped. The default stdout/stderr file is named <code>datastore.out</code> and is located in the Endeca Server's <code>logs</code> directory.
<code>--pidfile &lt;pidfile-path&gt;</code>	Specify the file to which to write the process ID (pid). The default PID file is named <code>datastore.pid</code> and is located in the Endeca Server's <code>logs</code> directory.
<code>--read-only</code>	<p>Provide read-only access to the data files for the data store. If you start the Dgraph with this flag, it performs only read-only operations. Any operations that attempt to write to the data files are rejected by the Oracle Endeca Server, and return an HTTP status code 403.</p> <p>This flag can be useful in implementations with multiple staging and production environments. You can also use this flag for hosting demos on public-facing web sites.</p> <p>If you use the <code>--read-only</code> flag, you cannot use the <code>--follower</code> flag at the same time.</p>
<code>--search_max &lt;num&gt;</code>	Specify the maximum number of terms for text search. Default is 10.
<code>--snip_cutoff &lt;num&gt;</code>	<p>Limit the number of words in an attribute that the Dgraph evaluates to identify the snippet.</p> <p>If a match is not found within <code>&lt;num&gt;</code> words, the Dgraph does not return a snippet, even if a match occurs later in the attribute value.</p> <p>If the flag is not specified, or <code>&lt;num&gt;</code> is not specified, the default is 500.</p>
<code>--snip_disable</code>	Globally disable snippeting.
<code>--sslcafile &lt;CA-certfile-path&gt;</code>	<p>Specify the path of the <code>eneCA.pem</code> Certificate Authority file that the Dgraph will use to authenticate SSL communications with other components that must communicate with the Dgraph.</p> <p>If not given, SSL mutual authentication is not performed.</p>
<code>--sslcertfile &lt;certfile-path&gt;</code>	Specify the path of the <code>eneCert.pem</code> certificate file that will be used by the Dgraph to present to any client for SSL communications. If not given, SSL is not enabled for Dgraph communications.

Flag	Description
<code>--sslcipher &lt;cipher-list&gt;</code>	Set one or more cipher names (such as RC4-SHA) that specify the minimum cryptographic algorithm that the Dgraph will use during the SSL negotiation. If multiple ciphers are specified, the names must be separated by colons.
<code>--stat-all</code>	Enable all available dynamic attribute value characteristics. Note that this option has performance implications and is not intended for production use.
<code>--stat-brel</code>	Create dynamic record attributes indicating the relevance rank assigned to full-text search result records.
<code>--syslog</code>	Direct all output to syslog.
<code>--thesaurus_cutoff &lt;limit&gt;</code>	<p>Set a limit on the number of words in a user's search query that are subject to thesaurus replacement. If more terms than this number match thesaurus entries, none of the terms are thesaurus expanded.</p> <p>The default value of <i>&lt;limit&gt;</i> is 3. This means that up to 3 words in a user's search query can be replaced with thesaurus entries.</p> <p>This option is intended as a performance guard against very expensive thesaurus queries. Lower values improve thesaurus engine performance.</p>
<code>--thesaurus_multiword_nostem</code>	<p>Specify that words in a multiple-word thesaurus form should be treated like phrases and should not be stemmed, which increases performance for some query loads.</p> <p>Single-word terms are subject to stemming regardless of whether this flag is specified.</p> <p>This flag prevents the Dgraph from expanding multi-word thesaurus forms by stemming. Thesaurus entries continue to match any stemmed form in the query, but multi-word expansions only include explicitly listed forms. To get the multi-word stemmed thesaurus expansions, the various forms must be listed explicitly in the thesaurus.</p>
<code>--threads &lt;num&gt;</code>	<p>Specify the number of threads in the Dgraph threading pool.</p> <p>The value of <i>&lt;num&gt;</i> must be a positive integer (that is, 1 or greater).</p> <p>The default for <i>num</i> is 2.</p> <p>The recommended number of threads for the Dgraph is typically equal to the number of cores on the host machine.</p>

Flag	Description
--unctrct	<p>Specify to the Dgraph not to compute implicit managed attributes, and to only compute and present explicitly specified managed attributes, when displaying refinements in navigation results.</p> <p>Specifying this flag does not reduce the size of the resulting record set that is being displayed; however, it improves run-time performance of the Dgraph process.</p> <p>Be aware that if you use this flag, in order to receive meaningful navigation refinements, you need to make top-level precedence rules work for ALL outbound queries.</p>
--validate_data	Validate that all processed data loads and then exit.
--version	Print version information and then exit. This includes both the Oracle Endeca Server version and the internal Dgraph identifier.
--wildcard_max <count>	Specify the maximum number of terms that can match a wildcard term in a wildcard query that contains punctuation, such as <code>ab*c.def*</code> . The default is 100.
--whymatch	<p>Enable computation of Why Did It Match dynamic record attributes returned as results of full-text search queries.</p> <p>These dynamic attributes contain a copy of the attribute key and value that caused the match, along with query interpretation notes (spelling, thesaurus, and so on).</p>
--whymatchConcise	<p>Similar to <code>--whymatch</code>, but produces more concise dynamic attribute values containing only the attribute key and query interpretation notes.</p> <p>This is useful when the attribute value might include large amounts of text, such as document contents.</p>
--wordinterp	Enable computation of word interpretation dynamic supplement (or see-also) objects, which report on alternate forms of user query terms considered by the text search engine while processing full-text (record) search requests.

Flag	Description
<code>--xquery_fndoc &lt;mode&gt;</code>	<p>Specifies the handling of the <code>fn:doc()</code> function within XQuery. The following values are supported:</p> <ul style="list-style-type: none"><li>• <code>none</code> causes all calls to <code>fn:doc()</code> to fail.</li><li>• <code>sandbox</code> allows <code>fn:doc()</code>, but interprets its argument as a relative path within the XML subdirectory of the XQuery service directory.</li><li>• <code>open</code> allows <code>fn:doc()</code> and interprets its argument as a URL. Note that <code>open</code> is not supported for use in deployed applications.</li></ul> <p>If not specified, defaults to <code>sandbox</code>.</p>
<code>--xquery_path &lt;path&gt;</code>	<p>Specify the directory in which XQuery Web service resources are located. (This Web service is internal to the Dgraph and should not be used directly). XQuery main modules and WSDL files are loaded from this directory. If not specified, a user XQuery path is not used.</p>

# Index

## A

- Administration Web Service
  - about 26
  - accessing 26
  - using 27
- administrative tasks
  - admin and config operations 42
  - overview 9
- admin operations
  - flush 44
  - help 44
  - list of 43
  - logroll 44
  - merge 44
  - ping 44
  - reload-services 45
  - stats 45
  - statsreset 45
  - updateaspell 45
- aggressive merge policy 58
- architecture, cluster 64
- attach-ds command, Endeca Server 19

## B

- balanced merge policy 58

## C

- cacerts certificates file 80
- canceling a job 32
- certificates
  - copying to other machines 87
  - eneCA.cer 85
  - eneCA.key 85
  - eneCA.pem 85
  - eneCert.p12 85
  - eneCert.pem 85
  - generating from own private key 87
- changing the merge policy of a data store 60
- cluster
  - about 62
  - and Integrator configuration 77
  - architecture 64
  - behavior 72
  - bringing it online 72
  - Cluster Coordinator failure 73
  - file system requirements 67
  - leader node 66
  - load balancer requirements 68
  - node failure 73
  - operating systems requirements 67
  - planning nodes 72

- sending data updates 77
- Cluster Coordinator
  - and Endeca Server as a Windows service 73
  - configuration file 70
- config operations
  - about 42
  - for logging verbosity 46
  - help 49
  - list of 47
  - log-disable 48
  - log-enable 48
  - logging variables 47
  - log-status 49
- connecting a Web browser to the Oracle Endeca Server 38
- connection errors, Dgraph and client 41
- core dump files
  - in the Dgraph 39
  - managing 39
- create-ds command, Endeca Server 18

## D

- detach-ds command, Endeca Server 21
- Dgraph 46
  - admin operations 43
  - checking aliveness of 38
  - config operations 47
  - crash dump files on Linux 39
  - crash dump files on Windows 39
  - flags 91
  - identifying connection errors 41
  - logging variables for 46
  - logs 40
  - running multiple instances on a single machine 41
  - what to collect for debugging 40

## E

- endeca-cmd
  - global options 15
  - options for SSL 17, 84
- Endeca data store
  - about 11
  - attaching to Endeca Server 19
  - creating 18
  - detaching 21
  - job monitoring 25
  - listing available 24
  - logs 11
  - starting 22
  - status of 23
  - stopping 22

**Endeca Server**

- attach-ds command 19
- command interface, global options 15
- configuration 11
- configuring automatic restart 57
- create-ds command 18
- creating as a Windows service 53
- data store logs 11
- deleting Windows service 55
- detach-ds command 21
- host option 16
- increasing the maximum idle time 13
- list-ds command 24
- list-jobs command 25
- logging as a Windows service 57
- modifying Windows service 54
- overview 10
- port option 16
- setting description for Windows service 53
- start-ds command 22
- started from inittab on Linux 57
- starting and stopping 11
- starting or stopping from Services utility 56
- status-ds command 23
- stop-ds command 22
- version command 25

eneCA.cer, description of 85

eneCA.key, description of 85

eneCA.pem, description of 85

eneCert.p12, description of 85

eneCert.pem

- description 85

- generating with own private key 87

enecerts utility

- changing key size 86

- generating certificates with own private key 87

- overview 85

**F**

flags, Dgraph 91

flush admin operation 44

follower node

- about 67

- adding 75

forcing a merge 61

**G**

Global Configuration Record

- retrieving with API 59

- setting merge policy 60

**H**

help admin operation 44

help config operation 49

**I**

incremental updates, merge policy for 58

inittab, starting Endeca Server from 57

IPv4 and IPv6 address support in Endeca Server 38

**J**

Java KeyStores, converting 89

job monitoring

- canceling a job 32

- listing running and pending jobs 24, 31

- types 30

- when to use 30

job start time 31

**K**

key size, changing private 86

keystore

- creating for endeca-cmd 80

- default 80

keytool

- creating keystore 80

- location of 79

**L**

leader node

- about 66

- adding, to a cluster 74

- changing name 78

- list of updating operations to send to it 76

list-ds command, Endeca Server 24

listing

- ID of outer transaction 31

- jobs 31

list-jobs command, Endeca Server 25

log-disable config operation 48

log-enable config operation 48

logging variables 46

- operation syntax 47, 48

- supported variables for 47

logroll admin operation 44

log-status config operation 49

**M**

maximum idle time, increasing 13

merge admin operation 44

merge policy

- changing 59

- changing in a running data store 60

- forcing a merge 61

- for incremental updates 58

- getting programmatically 59

- setting programmatically 60

- types of 58
  - multiple-node cluster development environment 65
  - mutual authentication for Dgraph 88
- O**
- obfuscating passwords for keystore 82
  - operation syntax for Dgraph logging variables 47
  - Oracle Endeca Server
    - connecting Web browsers to 38
    - IPv4 and IPv6 address support 38
  - outer transaction ID, listing 31
- P**
- passwords for keystore, obfuscating 82
  - ping admin operation 44
  - private key for certificates
    - changing size of 86
    - description 85
- R**
- reload-services admin operation 45
- S**
- single-node cluster development environment 64
  - snapshot
    - about 34
    - cpmdex command 36
    - creating 35
    - deleting 35
    - restoring data files 36
    - restrictions 34
  - spelling, enabling 45
  - SSL
    - converting PEM-format keys to JKS format 89
    - creating keystore and truststore 80
    - endeca-cmd interface 84
    - enecerts utility 85
    - global options 17
    - mutual authentication 88
    - obfuscating keystore passwords 82
  - start-ds command, Endeca Server 22
  - stats admin operation 45
  - statsreset admin operation 45
  - status-ds command, Endeca Server 23
  - stop-ds command, Endeca Server 22
- T**
- truststore
    - conversion from PEM 89
    - creating for endeca-cmd 80
- U**
- updateaspell admin operation 45
  - URL operations, about 42
- V**
- variables for Dgraph logging 47
  - version command, Endeca Server 25
- W**
- Who should use this guide 7
  - Windows service
    - creating for Endeca Server 53
    - deleting Endeca Server 55
    - Endeca Server logs 57
    - modifying Endeca Server configuration 54
    - setting description 53