macromedia®
COLDFUSION®
MX

Working with Verity Tools

Part Number ZCF60M900

Acknowledgments
Project Management: Stephen M. Gilson

Writing: Michael Stillman

First Edition: May 2002

Macromedia, Inc.
600 Townsend St.
San Francisco, CA 94103

CONTENTS

ABOUT THIS BOOK

Working with Verity Tools is intended for ColdFusion developers who want to use the Verity® advanced features, including Verity utilities and the K2 Server.

Contents

Developer resources

Macromedia, Inc. is committed to setting the standard for customer support in developer education, technical support, and professional services. The Macromedia website is designed to give you quick access to the entire range of online resources. The following table shows the locations of these resources.

Resource	Description	URL
Macromedia website	General information about Macromedia products and services	http://www.macromedia.com
Information on ColdFusion	Detailed product information on ColdFusion and related topics	http://www.macromedia.com/coldfusion
Macromedia ColdFusion Support Center	Professional support programs that Macromedia offers	http://www.macromedia.com/support/coldfusion
ColdFusion Online Forums	Access to experienced ColdFusion developers through participation in the Online Forums, where you can post messages and read replies on many subjects relating to ColdFusion	http://webforums.macromedia.com/coldfusion/
Installation Support	Support for installation-related issues for all Macromedia products	http://www.macromedia.com/support/email/isupport
Training	Information about classes, on-site training, and online courses offered by Macromedia	http://www.macromedia.com/support/training
Developer Resources	All the resources that you need to stay on the cutting edge of ColdFusion development, including online discussion groups, Knowledge Base, technical papers, and more	http://www.macromedia.com/desdev/developer/
Reference Desk	Development tips, articles, documentation, and white papers	http://www.macromedia.com/v1/developer/TechnologyReference/index.cfm
Macromedia Alliance	Connection with the growing network of solution providers, application developers, resellers, and hosting services creating solutions with ColdFusion	http://www.macromedia.com/partners/

About Macromedia ColdFusion MX documentation

The ColdFusion documentation is designed to provide support for the complete spectrum of participants. The print and online versions are organized to let you quickly locate the information that you need. The ColdFusion online documentation is provided in HTML and Adobe Acrobat formats.

Printed and online documentation set

The ColdFusion documentation set consists of the following titles:

Book	Description
Installing ColdFusion MX	Describes system installation and basic configuration for Windows NT, Windows 2000, Solaris, Linux, and HP-UX.
Administering ColdFusion MX	Describes how to use the ColdFusion Administrator to manage the ColdFusion environment, including connecting to your data sources and configuring security for your applications,
Developing ColdFusion MX Applications with CFML	Describes how to develop your dynamic web applications, including retrieving and updating your data, using structures, and forms.
Getting Started Building ColdFusion MX Applications	Contains an overview of ColdFusion features and application development procedures. Includes a tutorial that guides you through the process of developing an example ColdFusion application.
Using Server-Side ActionScript in ColdFusion MX	Describes how Macromedia Flash movies executing on a client browser can call ActionScript code running on the ColdFusion server. Includes examples of server-side ActionScript and a syntax guide for developing ActionScript pages on the server.
Migrating ColdFusion 5 Applications	Describes how to migrate a ColdFusion 5 application to ColdFusion MX. This book describes the code compatibility analyzer that evaluates your ColdFusion 5 code to determine any incompatibilities within it.
CFML Reference	Provides descriptions, syntax, usage, and code examples for all ColdFusion tags, functions, and variables.
CFML Quick Reference	A brief guide that shows the syntax of ColdFusion tags, functions, and variables.
Working with Verity Tools	Describes Verity search tools and utilities that you can use for configuring the Verity K2 Server search engine, as well as creating, managing, and troubleshooting Verity collections.
Using ClusterCATS	Describes how to use Macromedia ClusterCATS, the clustering technology that provides load-balancing and failover services to assure high availability for your web servers.

Viewing online documentation

All ColdFusion documentation is available online in HTML and Adobe Acrobat Portable Document Format (PDF) files. To view the HTML documentation, open the following URL on the web server running ColdFusion: http://*web_root*/cfdocs/dochome.htm.

ColdFusion documentation in Acrobat format is available on the ColdFusion product CD-ROM.

Getting answers

One of the best ways to solve particular programming problems is to tap into the vast expertise of the ColdFusion developer communities on the ColdFusion Forums. Other developers on the forum can help you figure out how to do just about anything with ColdFusion. The search facility can also help you search messages from the previous 12 months, allowing you to learn how others have solved a problem that you might be facing. The Forums is a great resource for learning ColdFusion, but it is also a great place to see the ColdFusion developer community in action.

Contacting Macromedia

Corporate headquarters	Macromedia, Inc. 600 Townsend Street San Francisco, CA 94103 Tel: 415.252.2000 Fax: 415.626.0554 Web: http:// www.macromedia.com
Technical support	Macromedia offers a range of telephone and web-based support options. Go to http://www.macromedia.com/support/coldfusionfor a complete description of technical support services. You can make postings to the ColdFusion Support Forum (http://webforums.macromedia.com/coldfusion) at any time.
Sales	Toll Free: 888.939.2545 Tel: 617.219.2100 Fax: 617.219.2101 E-mail: sales@macromedia.com Web: http://www.macromedia.com/store

CHAPTER 1
Introducing Verity Tools

This chapter provides an overview about the advanced Verity® features included in ColdFusion. These include several utilities that you can use to configure, manage, and troubleshoot search functionality in your ColdFusion applications. This chapter also introduces the Verity K2 Server, which lets you provide high-performance search capabilities for your ColdFusion applications.

Contents

About the Verity utilities

ColdFusion includes several Verity utilities to diagnose and manage your collections. These tools include the mkvdk, rcvdk, rck2, and vspider utilities.

The following table describes the relationship between the major Verity utilities and the corresponding `cfcollection`, `cfsearch`, and `cfindex` ColdFusion tags (the `cfcollection` tag operates on the entire collection, whereas the `cfindex` tag operates on records within a collection):

	cfcollection				cfindex				cfsearch
utility	**create**	**repair**	**delete**	**optimize**	**update**	**delete**	**purge**	**refresh**	**search**
mkvdk	X	X		X	X	X	X	X	
rcvdk									VDK mode search
rck2									K2 mode search
vspider	X	X		X	X		X	X	

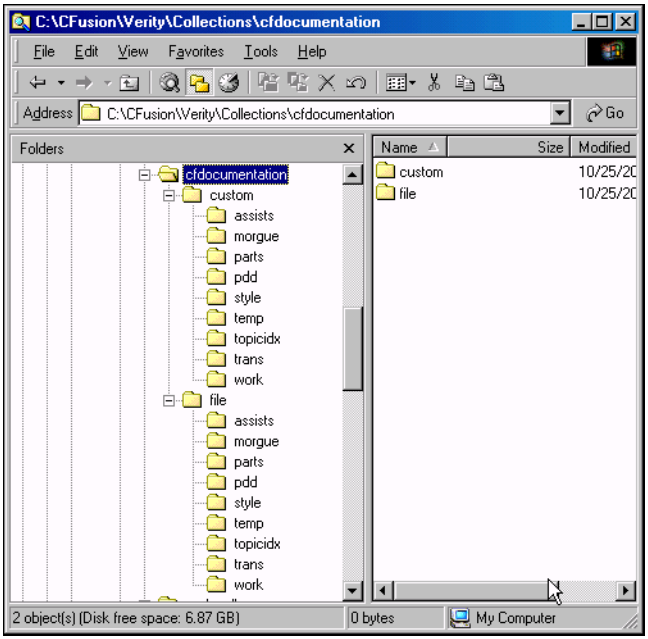
Note: Collections created with ColdFusion and those created externally using native Verity tools differ in structure. When performing operations on Verity collections created with ColdFusion, you may be required to include the full path to the collection. For more information, see "Collection structure and ColdFusion" on page 3.

For more information, see Chapter 6, "Troubleshooting Collections with Verity Utilities" on page 77.

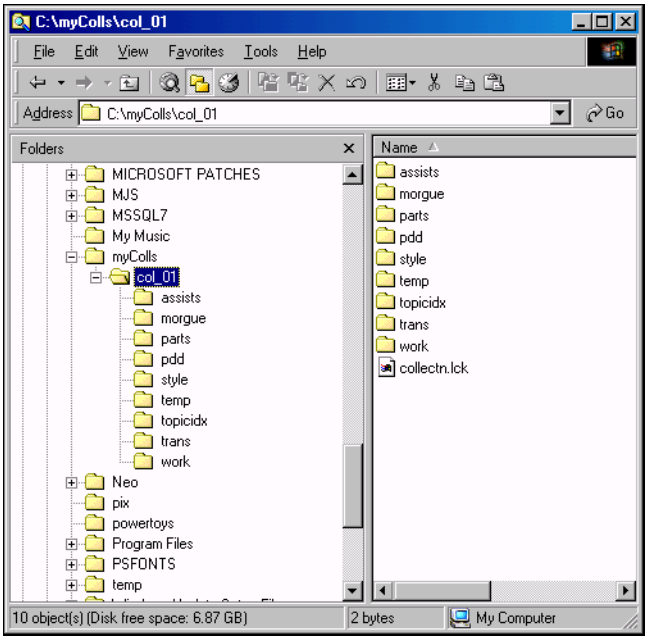
Collection structure and ColdFusion

Collections created in ColdFusion, either through the ColdFusion Administrator or by using the `cfcollection` tag, have different directory structures than external collections. An **external** collection is one created by a tool other than ColdFusion, such as the native Verity indexing tool mkvdk. For more information on mkvdk, see Chapter 2, "Managing Collections with the mkvdk Utility" on page 9.

The following figure shows the directory structure of a collection that was created with ColdFusion. This collection consists of two subdirectories—custom and file—that are not present in external collections:



The following figure shows the directory structure of an external collection, named col_01. Note the absence of custom and file directories:



The type of index used dictates which of these folders is populated with index data. Based on the `type` attribute of the `cfindex` tag, the file folder is used for `type="File"` and for `type="Path"`; the custom folder is used for `type="Custom"`. For more information on indexing, see *Developing ColdFusion MX Applications with CFML*.

The `type` information is important when you configure the `collPath` attribute of a collection in your k2server.ini file. The name of the external collection (col_01) above is C:\myColls\col_01. In contrast, the collection created by ColdFusion (cfdocumentation) actually contains two collections—C:\CFusion\Verity\Collections\cfdocumentation\file and C:\CFusion\Verity\Collections\cfdocumentation\custom. Using CFML tags, you only need to refer to "cfdocumentation" to access both the file and custom collections. However, since Verity tools, such as K2 Server, do not understand the ColdFusion collection structure, you must explicitly specify both the file collection and the custom collection in order for K2 Server to search collections created with ColdFusion.

For more information about configuring the `collPath` attribute, see "Editing the k2server.ini file" on page 66.

Verity search modes in ColdFusion

Your ColdFusion applications can search Verity collections using two modes:

- **VDK mode** The default ColdFusion search mode. You register a collection with ColdFusion by using the `cfcollection` tag or by using the Verity Collections page in the ColdFusion Administrator (which also uses the `cfcollection` tag).
- **K2 mode** The high-performance K2 Server mode. Use the ColdFusion Administrator Verity Server page to configure ColdFusion to also search using K2 Server. Once you add the existing collections to k2server.ini and start K2 Server, the ColdFusion Administrator Verity Collections page indicates these K2 Server-registered collections. For more information, see "Using K2 Server" on page 66.

By default, unless you configure ColdFusion to use K2 Server, ColdFusion uses VDK mode to search collections. The `cfsearch` tag is functionally identical between the two modes.

For more information about the benefits and restrictions of K2 Server, see "About K2 Server" on page 7.

For more information on using VDK mode (the default Verity search mode), see *Developing ColdFusion MX Applications with CFML*.

How ColdFusion determines which mode to use

ColdFusion determines which search mode to use by examining which server (ColdFusion or K2) has registered the collection name(s) that you specified in your `cfsearch` tag.

Note: You cannot combine collections registered with ColdFusion and with K2 Server within a single `cfsearch` tag. To search both types of collections from the same ColdFusion page, use two `cfsearch` tags.

Your server may contain several Verity collections. You can register a collection with the ColdFusion Server (for VDK mode searches) and with the K2 Server (for K2 mode searches). To register a collection for VDK mode searches, you use a `cfcollection` tag, either directly in CFML or indirectly with the ColdFusion Administrator. To register a collection for K2 mode searches, edit the k2server.ini file. For more information, see "Editing the k2server.ini file" on page 66.

In the following example, the plants collection has been registered with ColdFusion and is not listed in the k2server.ini file. ColdFusion uses the VDK mode to search this collection:

```
<cfsearch
   collection="plants"
   name="getData"
criteria="#form.criteria#">
```

In the following example, plants_al has been listed in k2server.ini and is a unique alias. That is, the collection name, plants_al, is different than any Verity collections that are configured for use by ColdFusion. ColdFusion uses K2 mode to search this collection:

```
<cfsearch
   collection="plants_al"
   name="getData"
criteria="#form.criteria#">
```

Tip: Check the Verity Collections page in the ColdFusion Administrator for possible naming conflicts between collection and collection alias names. If you have a collection named plants that is registered with ColdFusion, you must have a unique alias in the k2server.ini file to run a K2 mode search.

Verity information storage

All Verity configuration data and collection name registration information are stored in an XML file (neo-verity.xml), which is used solely by the ColdFusion Server. This XML file, which is located in *cf_root*/lib, contains two collection lists. One list contains collections that are registered with ColdFusion; ColdFusion uses the VDK mode to search these collections. The second list contains collections that are registered with K2 Server; ColdFusion uses the K2 mode to search these collections. You do not need to edit this XML file.

ColdFusion updates neo-verity.xml whenever one of the following occurs:

- ColdFusion starts.
- You change Verity or K2 information in the ColdFusion Administrator.
- You change the list of registered collections in the `cfcollection` tag.
- ColdFusion stops.

Before ColdFusion updates neo-verity.xml, it copies the file, using the BAK extension.

Tip: If the neo-verity.xml and neo-verity.bak files become damaged, use the neo-verity.org file. This file is a valid neo-verity.xml file that has not been modified since you installed ColdFusion.

About K2 Server

The Verity K2 Server is a high-performance search engine designed to process searches quickly in a high-performance, distributed system. The K2 search system has a client/server model. K2 client applications, such as ColdFusion Server, provide users access to document indexes stored in Verity collections. K2 Server supports simultaneous indexing of distributed enterprise repositories and handles hundreds of concurrent queries and users. You will see considerable performance improvements when using K2 Server to search Verity collections.

The K2 search system takes advantage of the latest advances in hardware and software technology, and provides the following features:

- Multithreaded architecture
- Support for Verity knowledge retrieval features, including topics
- Continuous operation support
- High scalability

ColdFusion installs K2 Server by default. You must make minor changes to configure K2 Server to work with ColdFusion.

ColdFusion and K2 Server

ColdFusion includes an OEM-restricted version of the Verity K2 Server. The version of K2 Server that is part of ColdFusion is restricted in the following areas:

- ColdFusion can only interact with one K2 Server at a time.
- K2 Server has the following document search limits (limits are for all collections registered to K2 Server):
  - 25,000 documents for ColdFusion Standard
  - 125,000 documents for ColdFusion Professional
  - 250,000 documents for ColdFusion Enterprise
  - 750,000 documents for Macromedia Spectra sites

  Note: Each row in a database table is considered a document.

If you install a fully licensed version of K2 Server and you configure ColdFusion to use the K2 Broker, ColdFusion will not restrict document searches.

Note: A K2 Broker receives search requests from users. The broker manages communications between clients and the K2 search process, and aggregates all search results and presents them to the user. The K2 Broker is not packaged with ColdFusion; for more information, visit http://www.verity.com.

CHAPTER 2
Managing Collections with the mkvdk Utility

The mkvdk utility is a command-line utility installed with ColdFusion. You can use it to perform maintenance operations on Verity collections.

Contents

About the Verity mkvdk utility

The mkvdk utility is an indexing application, provided with other Verity utilities, that you can use to create and maintain collections. It is a command-line utility that you can use within other applications or shell scripts to provide more sophisticated scheduling and other capabilities.

The mkvdk.exe file, which starts the mkvdk utility, is located in the *cf_root*\lib_nti40\bin directory in Windows, and in the *cf_root*/lib/*platform*/bin directory in UNIX.

In these pathnames, *cf_root* refers to the ColdFusion root directory. In Windows, this is typically C:\CFusionMX; in UNIX, this is typically /opt/coldfusionmx. In UNIX, *platform* refers to the UNIX version of the server that runs ColdFusion: _solaris, _hpux11, or _ilnx21.

The mkvdk utility syntax

The following is the basic syntax of the mkvdk command:

```
mkvdk -collection path [option] [dockey]
```

Multiple options and dockeys can be included, as needed. If dockey is a list of files, it should consist of an at sign (@) followed by the filename that contains a simple list of files (for example, @filelist). For more information about the options for the mkvdk utility, see "Getting started with the Verity mkvdk utility" on page 12.

The following operations occur when you use the mkvdk utility to create a new collection:

1. New collection directories are created and the specified style files are copied to the style subdirectory.
2. The style file settings are read and the required information is passed to the Verity search engine.
3. The gateway is used to open the document files, which are parsed according to the settings in various style files.
4. A new partition is created, which includes an index and an attribute table.
5. Assist data is generated, which might include a spanning word list.

When problems occur during an operation, the mkvdk utility writes error messages to the system log file (sysinfo.log). You can direct error and other messages to the console by using the mkvdk command with the -outlevel option. You can direct messages to a file of your choice by using the -loglevel and -logfile options.

The log file contains the following fields:

- Date
- Time
- Level
- Code
- Component
- Description

You can use the log file to view details about what happens during the collection creation process. Use the mkvdk -loglevel command and specify the numeric identifier for the message level you want, as summarized in the following table:

Type	Number
Fatal	1
Error	2
Warning	4
Status	8
Info	16
Verbose	32
Debug	64

To calculate the numeric parameter, add the numbers for the message types you want to include. The default for both -outlevel and -loglevel is 15, which selects fatal, error, warning, and status messages (1+2+4+8).

Getting started with the Verity mkvdk utility

The following is the basic mkvdk syntax:

```
mkvdk -collection path [option] [...] [filespec] [...]
```

Where:

- Square brackets ([ ]) indicate optional items.
- An ellipsis (...) indicates repetition of the previous item. Thus, `[filespec] [...]` indicates an optional series of filespec items.
- `filespec` represents a document filename or a list of document filenames. If `filespec` is a list of files, it should consist of an at sign (@) followed by the filename containing the list (for example, `@filelist`).
- The `-collection path` argument creates or opens a collection. This argument is required.

Numerous optional syntax options are listed below. All syntax options must precede the first `filespec` parameter.

Creating a collection

Creating a collection with the mkvdk utility involves setting up a collection directory structure and inserting documents into this structure. You can create a collection in two steps, using two separate commands.

To create a collection:

1 Set up a collection using the following syntax:

```
mkvdk -create -collection collectionname
```

Where `collectionname` is the path to the collection directory. Running this command creates a collection directory that includes style files with configuration information.

2 Insert documents using the following syntax:

```
mkvdk -collection collectionname -bulk -insert filespec
```

Where `filespec` is the name of a bulk insert file that specifies which documents to index and insert into the collection.

Alternatively, you can set up a collection and insert documents in one command, using the following syntax:

```
mkvdk -create -collection collectionname -bulk -insert filespec
```

Note: You can use the `-create` option only once to create the collection directory structure. After a collection directory structure has been created, do not to use the `-create` option to update the collection.

Accessing online Help for the mkvdk utility

To display a list of mkvdk command-line options, enter the following command:

```
mkvdk -help
```

Collection setup options

The mkvdk utility has a variety of collection setup options, which the following table describes:

Option	Description
-create	Creates a collection in the specified -collection directory. It creates the directory structure, determines the index contents and sets up the document's table schema according to the style files used. If the specified collection already exists, the mkvdk utility exits rather than overwriting the existing collection.
-style dir	Specifies the style directory that contains the style files to use to create a collection. This option can only be used with the -create option. If you do not specify this option when you use the mkvdk utility to create a collection, the mkvdk utility uses the style files in the common/style directory.
-description desc	Sets the collection's description. Enter alphanumeric text, such as "This collection contains electronic mail from ABC Company." Include the quotation marks.
-words	Builds the word list for all partitions in the collection.

Examples: setting up collections

The following examples show the commands for creating a collection and building the word list.

Creating a collection

The following command creates a collection in path_2 using the style files in path_1, and submits and indexes the document(s) in filespec:

```
mkvdk -create -style path_1 -collection path_2 filespec
```

Building the word list

The following command builds the word list in the collection residing in the path directory:

```
mkvdk -words -collection path
```

General processing options

The mkvdk utility provides a variety of general processing options, which the following table describes:

Option	Description
-collection path	Specifies the path of the collection to create or open. This option is required to execute the mkvdk utility.
-nolock	Turns off file locking. Locking is on by default.
-synch	Performs work immediately. If this option is not used, indexing work is done in the background, as time permits.

Option	Description
-about	Shows information about the collection, such as its description and the date when it was last modified.
-datapath path	Specifies the datapath to use to find documents that are added to the specified collection. All relative document paths are relative to this setting. If you do not set this option, the mkvdk utility looks for documents next to the collection directory.
-topicset path	Creates a topic index for the collection, based on the specified topic set, and stores it in the collection directory. This facilitates quick and efficient searches over the collection data when using topics.
-mode mode	Sets the indexing mode. Values are case-insensitive. The following are the valid settings: • Generic • FastSearch • NewsfeedIdx • NewsfeedOpt • BulkLoad • ReadOnly • Any custom mode defined in the style.plc file. The default is Generic mode.
-common	Specifies the path of the Verity common directory. If you do not use this option, the Verity engine looks for the common directory in the directory containing the mkvdk executable, and then along the executable search path. The executable search path is determined by your operating system environment settings. It is the path used by the OS to find the programs you run.
-help	Displays the mkvdk utility syntax options.
-debug	Runs the mkvdk command in debugging mode.
-nooptimize	Prevents optimization by this instance of the mkvdk utility. Using this option turns off the service-level VdkServiceType_Optimize. The service types determine the type of work the Verity engine and its self-administration features will execute on a collection.
-nohousekeep	Prevents housekeeping by this instance of the mkvdk utility. Housekeeping includes deleting files that are no longer needed. Using this option turns off the service-level VdkServiceType_DBA. (Service types are described under nooptimize.)
-noindex	Prevents indexing by this instance of mkvdk. Documents are not inserted or deleted. Using this option turns off the service-level VdkServiceType_Index. (Service types are described under nooptimize.)
-charmap name	Specifies the name of the character set to which to map all strings for your application. Set this to a character set that your system can display properly. Using the search engine with the English locale, the character set that any version of Windows displays is 8859. This is NOT the name of the character set of documents being indexed, it is only the name of the character set that your display can handle properly. (The character set of the document is set in the style.dft file using the /charmap option.) Valid options are 850 and 8859. The default is no mapping.

Option	Description
-locale name	Specifies the name of the Verity locale to be used by the mkvdk utility. The locale name must correspond to the name of an existing locale directory, which must exist in the install_dir/common/locale directory. Valid options are english, deutsch, and francais. The default is english.
-datefmt format	Converts a date field value into Verity's internal data representation.You can use this option in conjunction with the mkvdk options -extract (for the field extraction feature) and -bulk (for the bulk submit feature). The named format string identifies to the date parsing routines in what order dates are written when the date string only consists of a sequence of numbers (for example, 03/03/96). Valid options are described in "Date format options" on page 16. The default is MDY.
-servlev level	Specifies service level. The specifier, level, is a string consisting of keywords separated by hyphens, such as search-index-optimize. Valid keywords are described in "Service-level keywords" on page 16.

Examples: processing documents

The following examples show the commands for processing documents.

Using the default options

By default, the mkvdk command submits and indexes documents specified in the command, and services the specified collection. The following command executes the default options:

```
mkvdk -collection path filespec
```

Servicing only

The following command performs servicing only. Use this command to only index submitted documents and service the collection:

```
mkvdk -collection path
```

Deleting documents from a collection

The following command deletes documents from a collection:

```
mkvdk -delete -collection path filespec
```

Bulk inserting or deleting

The following command specifies bulk insertion of a list of documents:

```
mkvdk -collection coll -bulk -insert filespec
```

Where filespec is the list of files to insert. Since insert is the default, the following command is equivalent to the preceding command:

```
mkvdk -collection coll -bulk filespec
```

The following command specifies bulk deletion of a list of documents:

```
mkvdk -collection coll -bulk -delete filespec
```

Where filespec is the list of files to delete. It can be the same file used to insert documents; the only difference is that -delete is specified instead of -insert (or no specification).

Date format options

The Verity engine supports many import date formats, including many textual date formats, and the numeric date formats listed in the following table:

Format variable	Description
MDY	Dates written as month-day-year (US format, the default)
DMY	Dates written as day-month-year (European format)
YMD	Dates written as year-month-day (ISO international format)
YDM	Dates written as year-day-month (Swedish format)
USA	Dates written in US format (the same as MDY)
EUR	Dates written in European format (the same as DMY)

Service-level keywords

The following table describes the valid keywords for the -servlev keyword:

Keyword	Description
search	Enables search and retrieval
insert	Enables adding and updating documents
optimize	Enables opportunistic collection optimization
assist	Enables building of word list
housekeep	Enables housekeeping of unneeded files
delete	Enables document deletion
backup	Enables backup
purge	Enables background purging
repair	Enables collection repair
dataprep	Same as search-index-optimize-assist-housekeep
index	Same as insert-delete

Message options

The mkvdk utility provides a variety of messaging options, as described in the following table:

Option	Description
-quiet	Displays only fatal and error messages to the console. It overrides the -outlevel setting. For a list of message types, see the table in "The mkvdk utility syntax" on page 10.
-outlevel (num)	Indicates which message types to display to the console. Valid values are determined by adding together the numbers that correspond to the desired message types. The default value is 15. For more information, see the table in "The mkvdk utility syntax" on page 10.
-logfile filename	Saves messages in the specified file.
-loglevel (num)	Indicates which message types to route to the optional log file. Valid values are determined by adding numbers together that correspond to the desired message types. The default value is 15. For more information, see the table in "The mkvdk utility syntax" on page 10.

Document processing options

The mkvdk utility provides a variety of document processing options, as the following table describes:

Option	Description
-extract	Extracts field values from documents, using the field extraction rules specified in the style.tde file.
-insert	Adds documents to the collection. This is the default option for the mkvdk command.
-update	Adds documents to the collection by replacing all previous information about the specified documents.
-delete	Marks the specified documents as deleted, and makes them unavailable for searches. To actually remove deleted documents from the collection's internal documents table and word indexes, use the squeeze keyword (see "About squeezing deleted documents" on page 21).
-nosave	Specifies that a work list, which is generated by the mkvdk utility automatically when you use the -extract option, will not be saved in the collection directory in a file called worklist (in the Verity bulk submit file format). By default, the mkvdk utility saves the worklist in the worklist file.
-nosubmit	Specifies that a work list, which is generated by the mkvdk utility automatically when you use the -extract option, will not be submitted to the indexing engine and will be saved in the collection directory in a file called worklist (in the Verity bulk submit file format). This option allows the mkvdk utility to process field extraction separately from other indexing tasks.

Bulk submit options

The mkvdk utility provides a variety of bulk submit options, as described in the following table:

Option	Description
-bulk	Interprets filespec as a bulk submit file. You can use this option with the -insert, -update, and -delete options.
-offset num	Specifies the offset into a bulk submit file or files. If you specify multiple bulk submit files and use the -offset option, the offset is applied to all of the bulk submit files.
-numdocs num	Specifies the number of documents to insert or delete from the bulk insert file or files. If you specify multiple bulk insert or delete files and use the -numdocs option, the -numdocs setting is applied to all of the bulk insert or delete files.
-autodel	Deletes the bulk submit file or files when the bulk submission work is finished.

Using bulk insert and delete options

The bulk submit feature supports the insertion of documents and related field values into collections.

To use the bulk submit feature to populate fields:

1 Define the fields in the style.sfl and style.ufl file, as appropriate.

2 Create a bulk submit file that specifies the documents to insert and the field values for each document.

3 Run the mkvdk utility using the -bulk option and specifying the bulk submit file or files.

Collection maintenance options

The mkvdk utility provides a variety of collection maintenance options, as described in the following table:

Option	Description
-backup dir	Backs up the collection into the specified directory. The backup does not include the tde subdirectory. The tde subdirectory is created by and for Topic Document Entry if Topic Document Entry is used to create or maintain the collection.
-repair	Repairs the collection, performed by an API call.
-purge	Waits the amount of time specified by the -purgewait option and then deletes all documents in the collection, but not the collection itself. It leaves the collection directory structure intact. To specify a different wait period, use the -purgewait option instead of the -purge option. If you do not use the -purgewait option, the default is 600 seconds.
-purgeback	Used with the -purge option, performs a purge in the background.

Option	Description
-purgewait sec	Specifies to the -purge option how many seconds to wait. If you do not specify sec, the default is 600.
-noservice	Prevents collection servicing, which includes indexing, by this instance of the mkvdk command, performed by an API call.
-persist	Services the collection repeatedly, at default intervals of 30 seconds. Use the -sleeptime option to set a different interval.
-sleeptime sec	Specifies the interval between service calls when the mkvdk utility is run with the -persist option.
-optimize spec	Performs various optimizations on the collection, depending on the value of spec. The specifier, spec, is a string consisting of keywords separated by hyphens, such as maxmerge-squeeze-readonly. For valid keywords, see "Optimization keywords" on page 20.
-noexit	Windows only. Causes the I/O window to remain after the program is finished. By default, the window closes and the program exits, so that scripts calling the mkvdk utility do not hang.

Examples: maintaining collections

The following examples show the commands for maintaining a collection.

Repairing a collection

The following command automatically repairs a collection, or enables it after manual repairs:

```
mkvdk -repair -collection path
```

Backing up a collection

The following command backs up a collection to the specified directory:

```
mkvdk -backup path_1 -collection path_2
```

Deleting a collection

To delete a collection, use the appropriate command for your operating system. For example, to remove the collection directory structure and control files on a UNIX system, use the following command:

```
rm -r -collection_path
```

Purging a collection

The following command deletes all documents from a collection, but does not delete the collection itself:

```
mkvdk -purge -collection path
```

Purging a collection in the background

The following command purges the specified collection in the background:

```
mkvdk -purge -purgeback -collection path
```

Specifying persistent service

The following command runs the mkvdk command as a persistent process, so that servicing is performed repeatedly after num idle seconds:

```
mkvdk -persist -sleeptime num -collection path
```

Deleting a collection

The -purge option deletes all documents in a collection, but does not delete the collection itself. To delete a collection, use operating system commands, such as the rm command on UNIX, to remove the collection directory structure and control files.

Optimization keywords

The following table describes the optimization keywords for the -optimize option:

Keyword	Description
maxclean	Performs the most comprehensive housekeeping possible, and removes out-of-date collection files. Macromedia recommends this optimization only when you are preparing an isolated collection for publication. When using this type, if the collection is being searched, files sometimes get deleted too early, which can affect search results.
maxmerge	Performs maximal merging on the partitions to create partitions that are as large as possible. This creates partitions that can have up to 64000 documents in them.
readonly	Marks the collection as read-only and unchanged after the function call is done. This is appropriate for CD-ROM collections.
spanword	Creates a spanning word list across all the collection's partitions. A collection consists of numerous smaller units, called partitions, each of which includes a word list. Optionally, a spanning word list can be built with an ngram index.
ngramindex	Builds an ngram index for the collection. An ngram index is designed to improve the search performance for queries with the ‹TYPO› and ‹WILDCARD› operators. An ngram index cannot be built without a spanning word list. You can build a spanning word list and ngram index in the same command, for example: `mkvdk -collection collname -optimize spanword -ngramindex`
squeeze	Squeezes deleted documents from the collection. Squeezing deleted documents recovers space in a collection, and improves search performance. (For more information about squeeze, see "About squeezing deleted documents" on page 21.) Using this option invalidates the search results.
vdbopt	Configures the collection's Verity databases (VDBs). Each collection consists of smaller units called VDBs. This keyword has the effect of linearizing the data in a VDB, and making the collection metadata contained in the VDB more streamlined. It also lets the VDB grow to a much larger size.

Keyword	Description
tuneup	Performs the same as combining the maxmerge, vdbopt, and spanword keywords.
publish	Performs the same as all of the optimization types combined. Use this keyword to optimize the collection for the best possible retrieval performance, such as for publication to a network on a server or on a CD-ROM.

About squeezing deleted documents

When a document is deleted from a collection, its space is not recovered. It is merely marked as deleted and not available for subsequent searches. Squeezing actually removes deleted documents from the collection's internal documents table and word indexes, thus creating a smaller collection and reducing the collection's disk space. A smaller collection has a more efficient structure that makes searching slightly faster and uses slightly less memory.

You can safely squeeze deleted documents for a collection at anytime, because the mkvdk utility ensures that the collection is available for searching and servicing through its self-administration features. The application does not need to temporarily disable a collection to squeeze deleted documents, because when a squeeze request is made, the mkvdk utility assigns a new revision code to the collection. After a squeeze has occurred, the next time the application accesses the collection, the Verity engine notifies the application that dramatic changes have been made, and points the application to the new collection data.

Squeezing deleted documents out of a collection is a significant update to the collection. If users are reviewing search results at the time when squeezing occurs, the search results might be invalidated after the squeeze operation.

About optimized Verity databases

The Verity database (VDB) is the fundamental storage mechanism responsible for supporting dynamic access to documents in collections. A VDB consists of simple tables with rows and columns that relate to each other by row position. VDB tables are not relational, and their architecture supports quick and efficient searching over textual data. A VDB consists of segments that are packed into a single file. One of the advantages of having one packed VDB file is optimized search performance. The fewer files that need to be opened during search processing, the faster the search performance.

The VDB optimization option optimizes the packing of a collection's VDBs. When VDBs are built during normal indexing operations, the segments are not stored sequentially in the one-file VDB file system. As a result of VDB optimization, performance can be improved by reserializing the packed segments in the VDBs so that all segments are contiguous, and VDBs can grow in size. Optimized VDBs can grow up to 2 gigabytes in size, as opposed to the maximum 64 megabytes for an unoptimized one.

Using this option might degrade your indexing performance when certain indexing modes are set for the collection.

Performance tuning options

The mkvdk utility provides performance tuning options, as the following table describes:

Option	Description
-maxfiles num	Sets the maximum number of files that the mkvdk utility can have open at once. The default is 50.
-diskcache num	Sets the size of the mkvdk disk cache in kilobytes. The default is 128.

CHAPTER 3
Indexing Collections with Verity Spider

This chapter contains basic Verity Spider information and explains how to index documents on your website.

Contents

About Verity Spider

Verity Spider enables you to index web-based and file system documents throughout your enterprise. Verity Spider works in conjunction with the Verity KeyView document filtering technology, so that you can index more than two hundred of the most popular application document formats, including Microsoft Office2000, WordPerfect, ASCII text, HTML, SGML, XML and PDF (Adobe Acrobat) documents.

Note: The Verity Spider that is included with ColdFusion is licensed for websites that are defined and reside on the same machine on which ColdFusion is installed. Contact Verity Sales for licensing options regarding the use of the Verity Spider for external websites.

Web standard support

Verity Spider supports key web standards used by Internet and intranet sites. Standard HREF links and frames pointers are recognized, so that navigation through them is supported. Redirected pages are followed so that the real underlying document is indexed. Verity Spider adheres to the robots exclusion standard specified in robots.txt files, so that administrators can maintain friendly visits to remote websites. HTTP Basic Authentication mechanism is supported so that password-protected sites can be indexed.

Unlike other web crawlers, Verity Spider does not need to maintain complete local copies of remote documents. When documents are viewed through Verity Information Server, documents are read from their native location with optional highlights.

Restart capability

When an indexing job fails, or for some reason the Verity Spider cannot index a significant number or type of URLs, you can now restart the indexing job to update the collection. Only those URLs that were not successfully indexed previously are processed.

State maintenance through a persistent store

Verity Spider V3.7 stores the state of gathered and indexed URLs in a persistent store, which lets it track progress for the purposes of gracefully and efficiently restarting halted indexing jobs.

Previous versions of Verity Spider only held state information in memory, which meant that any stoppage of spidering resulted in lost work. This also meant that larger target sites required significantly more memory for spidering. The information in the persistent store can help report information, such as the number of indexed pages, visited pages, rejected pages, and broken links.

Performance

Spidering performance is greatly improved over previous versions, because of low memory requirements, flow control, and the help of multithreading and efficient Domain Name System (DNS) lookups.

Flow control

When indexing websites, Verity Spider distributes requests to web servers in a round-robin manner. This means that one URL is fetched from each web server in turn. With flow control, a faster website can finish before a slower one. The Verity Spider optimizes indexing on every web server.

Verity Spider V3.7 adjusts the number of connections per server depending on the download bandwidth. When the download bandwidth from a web server falls below a certain value, Verity Spider automatically scales back the number of connections to that web server. There will always be at least one connection to a web server. When the download bandwidth increases to an acceptable level, Verity Spider reallocates connections (per the value of the -connections option, which is 4 by default). You can turn off flow control with the -noflowctrl option.

Multithreading

Since version 3.1, Verity Spider has separated the gathering and indexing jobs into multiple threads for concurrence. Verity Spider V3.7 can create concurrent connections to web servers for fetching documents, and have concurrent indexing threads for maximum utilization. This translates to an overall improvement in throughput. In previous releases, work was done in a round-robin manner, so that at any given time, only one job was running. Spider attends to the websites within an indexing job in a round-robin manner.

Efficient DNS lookups

Verity Spider V3.7 significantly reduces DNS lookups, which means great improvements to spidering throughput. If spidering is limited by domain or host, then no DNS lookups are made on hosts that fall outside of that range. In earlier versions, DNS lookups were made on all candidate URLs.

Proxy handling efficiency

To allow for greater flexibility when dealing with indexing jobs that involve proxy servers and firewalls, use the following options:

- **-noproxy** To reduce proxy checking for certain hosts
- **-proxyauth** To authenticate on proxy servers

Note: Information Server V3.7does not support retrieving documents for viewing through secure proxy servers. Do not use the -proxyauth option for indexing documents that you will view through Information Server V3.7.

About Verity Spider syntax

Before you create an indexing task for a new collection, make copies of the relevant default style files to ensure that you have a set of template style files in a known, stable state.

Running multiple simultaneous Verity Spider jobs on the Information Server host can cause performance problems for searches. This does not mean that you should never run indexing jobs when users might be searching, because your collections are available for searching even while indexing jobs are running. To optimize performance, try staggering your indexing jobs to avoid overloading your server.

The Verity Spider command

The vspider executable, which starts the vspider application, is located in the *cf_root*\lib_nti40\bin directory in Windows, and in the *cf_root*/lib/*platform*/bin directory in UNIX.

In these pathnames, *cf_root* refers to the ColdFusion root directory. In Windows, this is typically C:\CFusionMX; in UNIX, this is typically /opt/coldfusionmx. In UNIX, *platform* refers to the UNIX version of the server that runs ColdFusion: _solaris, _hpux11, or _ilnx21.

At its most basic level, a Verity Spider command consists of the following:

```
vspider -initialize -collection coll [options]
```

Where -initialize is -start or -refresh (when starting points have changed), and -collection is required to provide a target for the Verity Spider, and [options] can be a near-limitless combination of the options described later in this chapter.

For example:

```
c:\cfusionmx\lib\_nti40\bin\vspider -common c:\cfusionmx\lib\common
-collection c:\new -start http://localhost -indinclude *
```

There are dependencies for other options, depending on the nature of the indexing task. The following are some examples:

- To build a new collection, you must use -style.
- To control how Verity Spider operates, including which documents it indexes, use some Verity Spider options.

If you do not run the Verity Spider executable from its default installation directory, you must include that directory in your path. This is because the Verity Spider executable depends on other files to run properly.

Using a command file

For simpler reuse and archiving of your indexing commands, use the -cmdfile option for abstraction. By using an ASCII text file to store a task's options, you avoid the potential problem of using special characters in an option's parameter value. For example, the -processbif option requires the use of "!*" and therefore any task using that option must also use the -cmdfile option.

Command-line option reference

The following sections describe the Verity Spider V3.7 command-line options. Option names are case-sensitive.

-start

Specifies a starting point for an indexing job. You can specify multiple instances, or use multiple values in a single instance.

When you execute an indexing job from a command line, and you do not use a command file (with the -cmdfile option), you must URL-escape any special characters in the starting point. To URL-escape a special character, use "%hex-ASCII-character-number" in place of the character. For example, use /time%26/ instead of /time&/. This allows the operating system to properly process the command string.

If an indexing task halts, you can rerun the task as-is. The persistent store for the specified collection is read, and only those candidate URLs that are in the queue but not yet processed are parsed. Candidate URLs correspond to URLs of the following status, as reported by vsdb:

```
cand, used, inse, upda, dele, fail
```

Repository type	Starting point
Web	The URL or URLs from which Verity Spider is to begin indexing. Use other options, such as the -jumps option, to control how far from the starting point Verity Spider goes.
File	The starting directory or directories in which Verity Spider will start indexing. All subdirectories beneath the starting point will be indexed, unless you use the -pathlen option or any of the inclusion or exclusion criteria.

Note: By using the -start option with the -refresh option, you provide a starting point for Verity Spider and therefore do not need to use at least one of the following options: -host, -domain, -nofollow, or -unlimited.

-refresh

Used for updating a collection, specifies that Verity Spider process only those documents that qualify, as follows:

- They are new documents in the repository, and they qualify for indexing under the criteria.
- They exist in the collection and are recorded in the Verity Spider persistent store with a status of done. If Verity Spider determines that these indexed documents have been updated in the repository, then they are retrieved again to be reparsed and reindexed. The document VdkVgwKey values do not change.
- They are deleted in the collection. If Verity Spider determines that documents have been deleted from the repository, then they are also deleted from the persistent store and the collection. The exception to this rule is when you use the -nooptimize option with the -refresh option. In this case, any document deleted from the repository is marked for deletion in the collection. It will be removed from the collection and the persistent store when the next indexing task is run for the collection.

When you rerun an existing indexing job, Verity Spider automatically refreshes the collection. If you add or remove any of the starting points, however, you must manually specify the -refresh option to refresh existing documents.

Note: You can also use the -start option to provide a starting point for Verity Spider. If you do not use the -start option, use at least one of the following options: -host, -domain, or -nofollow. For further control, also see the -refreshtime option. If you do not use any constraint criteria, Verity Spider operates without limits and will likely index far more than you intended.

Core options

The following sections describe the Verity Spider core options.

-cmdfile

Syntax: `-cmdfile path_and_filename`

Specifies that Verity Spider reads command-line syntax from a file, in addition to the options passed in the command-line. This option includes the pathname to the file that contains the command-line syntax. The -cmdfile option circumvents command-line length limits.

The syntax for the command-file is:

```
option optional_parameters
```

For better readability, put each option and any parameters on a single line. Verity Spider can properly parse the lines.

Note: Macromedia strongly recommends that you take advantage of the abstraction offered by this option. This can greatly reduce user error in erroneously including or omitting options in subsequent indexing jobs.

-collection

Specifies the full path to the collection to create or update.

Note: You receive an error if you specify a filename with an extension of CLM. Meta collections are not supported.

-help

Displays Verity Spider syntax options.

-jobpath

Syntax: `-jobpath path`

Specifies the location of the Verity Spider databases and the indexing job-related files and directories.

The following are the job-related directories and their contents:

- **log** All Verity Spider log files. For descriptions of the log files, see -loglevel.
- **bif** Bulk insert files.
- **temp** Web pages cached for indexing. You can also specify the temp directory using the -temp option.
- **admin** Files created by the Information Server Admin Tool.

These directories are created for you under the last directory specified in path.

Path values must be unique for all indexing jobs. If you do not use the -jobpath option, Verity Spider creates a /spider/job directory within the collection. For multiple-collection tasks, the first collection specified is used.

Note: You cannot use multiple job paths for multiple simultaneous indexing tasks for the same collection. Only one indexing task at a time can run for a given collection.

-style

Syntax: `-style path`

Specifies the path to the style files to use when creating a new collection.

If the -style option is not specified, Verity Spider uses the default style files in *cf_root*/lib/common/style.

Note: You can safely omit the -style option when resubmitting an indexing job, as the style information will already be part of the collection. If you are using the -cmdfile option, you can leave it there.

Processing options

The following sections describe the Verity Spider processing options.

-abspath

Type: File system only

Generates absolute paths for files. Use this option when the document locations are not going to change, but the collection might be moved around.

When you index a web server's contents through the file system, use the -prefixmap option with the -abspath option to map the absolute file paths to URLs.

See also -prefixmap.

-detectdupfile

Type: File system only

Enables checksum-based detection of duplicates when indexing file systems.

By default, a document checksum is not computed on indexed files. By using the -detectdupfile option, a checksum is computed based on the CRC-32 algorithm. The checksum combined with the document size is used to determine if the document is a duplicate.

-indexers

Syntax: `-indexers num_indexers`

Specifies the maximum number of indexing threads to run on a collection.

The default value is 2. Increasing the value for the -indexers option requires additional CPU and memory resources.

See also -maxindmem.

-license

Syntax: `-license path_and_filename`

Specifies the license file to use.

By default, the ind.lic file is used, from the *cf_root*/lib/*platform*/bin directory; where *platform* represents the platform directory.

-maxindmem

Syntax: `-maxindmem kilobytes`

Specifies the maximum amount of memory, in kilobytes, used by each indexing thread. Specify the number of threads with the -indexers option.

By default, each indexing thread uses as much memory as is available from the system.

-maxnumdoc

Syntax: `-maxnumdoc num_docs`

Specifies the maximum number of documents to download or submit for indexing. The value for num_docs does not necessarily correspond to the number of documents indexed. The following factors affect the actual number:

- Whether the value of `num_docs` falls within a block of documents dictated by the -submitsize option. If it does, the entire block of documents must be processed.
- Whether documents retrieved are actually indexed, because they are invalid or corrupt.

-mimemap

Syntax: `-mimemap path_and_filename`

Specifies a control file (simple ASCII text) that maps file extensions to MIME-types. This lets you make custom associations and override defaults.

The following is the format for the control file:

```
#file_ext_no_dot            mime-type
abc                         application/word
```

-nocache

Type: Web crawling only

Used with the -noindex or -nosubmit options, this option disables the caching of files during website indexing. This has the effect of decreasing the demands on your disk space.

Normally, Verity Spider downloads URLs, then writes them to a bulk insert file and downloads the documents themselves. When indexing occurs, once the -submitsize option has been reached, the cached files are indexed and then deleted. If you use the -noindex option, the bulk insert file is submitted but not processed by Verity Spider, and so the documents are not deleted until indexing occurs. This will usually be `mkvdk` or `collsvc`, or you can use Verity Spider again with the -processbif option.

By using the -nocache option in conjunction with the -noindex or -nosubmit option, you avoid storing files locally. Files are downloaded only when indexing actually occurs.

See also -noindex.

-nodupdetect

Type: Web crawling only

Disables checksum-based detection of duplicates when indexing websites. URL-based duplicate detection is still performed.

By default, a document checksum is computed based on the CRC-32 algorithm. The checksum combined with the document size is used to determine if the document is a duplicate.

See also -followdup.

-noindex

Specifies that Verity Spider gathers document locations without indexing them. The document locations are stored in a bulk insert file (BIF), which is then submitted to the collection. This option is typically used in conjunction with a separate indexing process, such as `mkvdk` or collection servicers (`collsvc`). The BIF will be processed by the next indexing process run for the collection, whether it is Verity Spider, mkvdk, or collection servicers (`collsvc`).

Do not try to start both Verity Spider and another process at the same time. You must allow Verity Spider enough time to generate enough work for the secondary indexing process. If you are using `mkvdk`, you can run it in persistent mode to ensure it will act upon work generated by Verity Spider.

Note: When you execute an indexing job for a collection and you use the -noindex option, the persistent store for the collection is not updated.

See also -nocache and -nosubmit.

For more information on the mkvdk utility, see Chapter 2, "Managing Collections with the mkvdk Utility" on page 9.

-nosubmit

Specifies that Verity Spider gathers document locations without submitting them. The document locations are stored in a bulk insert file (BIF), which is not submitted to the collection. This option is typically used in conjunction with a separate indexing process, such as `mkvdk` or collection servicers (`collsvc`). You can also use Verity Spider again with the -processbif option. With an indexing process other than Verity Spider, you must specify the name and path for the BIF, because the collection has no record of it.

-persist

Syntax: `-persist num_seconds`

Enables the Verity Spider to run in persistent mode, checking for updates every num_seconds seconds until it is stopped.

While Verity Spider is running in persistent mode, there is no optimization. After Verity Spider is taken out of persistent mode, you need to perform optimization on the collection. For more information about using the mkvdk utility, see Chapter 2, "Managing Collections with the mkvdk Utility" on page 9.

Note: Do not run more than one Verity Spider process in persistent mode. As the Verity Spider is a resource-intensive process, only run it in persistent mode with an interval of less than one day. For time intervals greater than twelve hours, use some form of scheduling. Some examples are cron jobs for UNIX, and the AT command for Windows server.

-preferred

Type: Web crawling only

Syntax: `-preferred exp_1 [exp_n] ...`

Specifies a list of hosts or domains that are preferred when retrieving documents for viewing. You can use wildcard expressions, where the asterisk (*) is for text strings and the question mark (?) is for single characters. To use regular expressions, also specify the -regexp option. Use this option when you leave duplicate detection enabled and do not specify the -nodupdetect option.

When indexing, you might encounter a nonpreferred host first. In that case, documents are parsed and followed and stored as candidates. When duplicates are encountered on another server, which is preferred, the duplicate documents from the nonpreferred server are skipped. When documents are requested for viewing, they will be retrieved from the preferred server.

On Windows, include double quotation marks around the argument to protect the special characters, such as the asterisk (*). On UNIX, use single quotation marks. This is only required when you run the indexing job from a command line. Quotation marks are not necessary within a command file (the -cmdfile option).

See also -regexp.

-prefixmap

Type: File system only

Syntax: `-prefixmap path_and_filename`

Specifies a control file (simple ASCII text) that maps file system paths to web aliases.

In conjunction with the -abspath option, this option is typically used to create a URL field that is the web equivalent of a file system path. File system indexing is faster than web crawling over the network. If you use the -prefixmap option to replace the file system path with the web URL, relative hyperlinks in the HTML pages are kept intact when viewed through Information Server.

The following is the format for the control file:

```
src_field src_prefix dest_field dest_prefix
```

If you use backslashes, you must double them so that they are properly escaped; for example:

```
C:\\test\\docs\\path
```

For example, to map the filepath `/usr/pub/docs` to `http://web/~verity`, use the following:

```
vdkvgwkey /usr/pub URL http://web/~verity
```

See also -abspath.

-processbif

Syntax: `-processbif 'command_string !*'`

Specifies a command string in which you can call a program or script that operates on BIFs generated by Verity Spider.

Due to the use of special characters, which represent the bulk insert file (BIF), you must run Verity Spider with a command file using the -cmdfile option.

For example, if you want to use a script called fix_bif to add customized information to BIF files, use the following command:

```
vspider -cmdfile filename
```

Where `filename` is the text-only command file that contains the following (along with any other necessary options):

```
-processbif 'fix_bif !*'
```

Your command file will include other options as well.

-regexp

Specifies the use of regular expressions rather than the default wildcard expressions for the following options: -exclude, -indexclude, -include, -indinclude, -skip, -indskip, -preferred, and -nofollow.

Wildcard expressions allow the use of the asterisk (*) for text strings, and the question mark (?) for single characters, as the following table shows:

Wildcard expression	Text string
a*t	although, attitude, audit
a?t	ant, art
file?.htm	files.htm, file1.htm, filer.htm
name?.*	names.txt, named.blank, names.ext

Regular expressions allow for more powerful and flexible matching of alphanumeric strings; for example, to match "ab11" or "ab34" but not "abcd" or "ab11cd," you could use the following regular expression:

```
^ab[0-9][0-9]$
```

The full extent to which regular expressions can be employed is beyond the scope of this description. For more information on regular expressions, refer to a book devoted to the subject.

-submitsize

Syntax: `-submitsize num_documents`

Specifies the number of documents submitted for indexing at one time. The default value is 128. The upper limit is 64,000.

Note: Although larger values mean more efficient processing by the indexer, smaller values allow more parallelism on multi-CPU systems. In the event of a halt during indexing, a smaller value means fewer documents will be lost.

If a halt occurs during indexing, the chunk of documents specified by the -submitsize option is lost because there is no transactional rollback for indexing and the documents are no longer in the queue for indexing. When you rerun the indexing task, Verity Spider can only continue with URLs and documents that are enqueued.

-temp

Syntax: `-temp path`

Specifies the directory for temporary files (disk cache). By default, the temp directory is under the job directory (optionally specified with the -jobpath option).

If you do not specify a value for this option, Verity Spider creates a /spider/temp directory within the collection. For multiple-collection tasks, the first collection specified is used.

Note: Make sure the location you specify contains enough disk space to handle the documents that are downloaded and held before indexing. The documents are deleted from the hard disk after they are indexed.

See also -jobpath, for specifying the location of all indexing job directories and files, one of which is the temp directory.

Networking options

The following sections describe the Verity Spider networking options.

-agentname

Type: Web crawling only

Syntax: `-agentname string`

Specifies the value for the agent name field that is part of the HTTP request. Since web servers can be configured to return different versions of the same page depending on the requesting agent, you can use the -agentname option to impersonate a browser client.

Use double quotation marks if the name contains a space. Use the -cmdfile option if the agent name you want to use contains forbidden characters, such as slashes or backslashes.

-connections

Syntax: `-connections num_connections`

Specifies the maximum number of simultaneous socket connections to make to websites for indexing. Each connection implies a separate thread.

The default value is 6.

Note: The Verity Spider dynamic flow control makes the most use of all available connections when indexing websites. If you are indexing multiple sites, you might want to increase this number. Increasing the number of connections does not always help, because of such dependencies as your network connection and the capabilities of the remote hosts.

-delay

Type: Web crawling only

Syntax: `-delay num_milliseconds`

Specifies the minimum time between HTTP requests, in milliseconds. The default value is 0 milliseconds for no delay.

-header

Type: Web crawling only

Syntax: `-header string`

Specifies an HTTP header to add to the spidering request; for example:

```
-header "Referer: http://www.verity.com/"
```

Verity Spider sends some predefined headers, such as Accept and User-Agent, by default. Special headers are sometimes necessary to correctly index a site.

For example, earlier versions of Verity Spider did not support the Host header, which is needed for Virtual Host indexing. Also, a Proxy-authentication header was needed to pass a username and password to a proxy server.

In Verity Spider V3.7, the Host header is supported by default, and the -proxyauth option is available for proxy server authentication. Therefore, the -header option is maintained only for backwards compatibility and possible future enhancements.

Note: Misuse of this option causes spider failure. If this happens, rerun the indexing task with modified -header values.

-hostcache

Syntax: `-hostcache num_hostnames`

Specifies the number of host names to cache to avoid DNS lookups. Without this option, the host cache continues to grow.

The default value is 256.

-noflowctrl

Type: Web crawling only

Disables round-robin indexing of websites with network flow control.

By default, Verity Spider uses round-robin indexing of websites to avoid overwhelming a web server and to improve indexing performance. Verity Spider connects to each web server in a round-robin manner, using up to the value for the -connections option. This means that one URL is fetched from each web server, in turn.

Note: Using the -noflowctrl option can result in a significant drop in performance.

-noproxy

Type: Web crawling only

Syntax: `-noproxy name_1 [name_n] ...`

Used in conjunction with the -proxy option, the -noproxy option specifies that Verity Spider directly access the hosts whose names match those specified. By default, when you specify the -proxy option, Verity Spider first tries to access every host with the proxy information. To improve performance, use the -noproxy option for the hosts you know can be accessed without a proxy host. For the name variable, you can use the asterisk (*) wildcard for text strings; for example:

```
'*.verity.com'
```

You cannot use the question mark (?) wildcard, and the -regexp option does not let you use regular expressions.

On Windows, include double quotation marks around the argument to protect the asterisk special character (*). On UNIX, use single quotation marks. This is only required when you run the indexing job from a command line. Quotation marks are not necessary within a command file (the -cmdfile option).

Note: You must have valid Verity Spider licensing capability to use this option.

-proxy

Type: Web crawling only

Syntax: `-proxy proxyhost:port`

Specifies host and port for proxy server.

Note: You must have valid Verity Spider licensing capability to use this option.

See also -proxyauth for proxy servers that require authentication, and -noproxy for hosts that you know are accessible without having to go through a proxy server.

-proxyauth

Type: Web crawling only

Syntax: `-proxyauth login:password`

Specifies login information for proxy server connections that require authorization to get outside the firewall. Use this option in conjunction with the -proxy option.

Note: You must have valid Verity Spider licensing capability to use this option. Information Server V3.7 does not support retrieving documents for viewing through secure proxy servers. Do not use the -proxyauth option for indexing documents that are viewed through Information Server V3.7

-retry

Type: Web crawling only

Syntax: `-retry num_retries`

Specifies the number of times that Verity Spider should attempt to access a URL. Use the -retry option when it is likely that an unstable network connection will give false rejections.

The default value is 4.

-timeout

Type: Web crawling only

Syntax: `-timeout num_seconds`

Specifies the time period, in seconds, that Verity Spider should wait before timing out on a network connection and on accessing data. The data access value is automatically twice the value you specify for the network connection timeout.

The default value for the network connection time-out is 30 seconds, and therefore the default value for the data access time-out is 60 seconds.

Path and URL options

The following sections describe the Verity Spider path and URL options.

-auth

Syntax: `-auth path_and_filename`

Specifies an authorization file to support authentication for secure paths.

Note: There must be a corresponding "`Authfile=`" entry in the Information Server configuration file, inetsrch.ini, so that documents can be accessed for viewing. Both the -auth option and `Authfile=` must point to the same file.

-cgiok

Type: Web crawling only

Lets you index URLs containing the question mark (?). This typically means that the URL leads to a CGI or other processing program.

The return document produced by the web server is indexed and parsed for document links, which are followed and in turn indexed and parsed. However, if the web server does not return a page, perhaps because the URL is missing parameters that are required for processing in order to produce a page, nothing happens. There is no page to index and parse.

Example

The following is a URL without parameters:

```
http://server.com/cgi-bin/program?
```

If you include parameters in the URL to be indexed, as specified with the -start option, those parameters are processed and any resulting pages are indexed and parsed.

By default, a URL with a question mark (?) is skipped.

-domain

Type: Web crawling only

Syntax: `-domain name_1 [name_n] ...`

Limits indexing to the specified domain(s). You must use only complete text strings for domains. You cannot use wildcard expressions. URLs not in the specified domain(s) are not downloaded or parsed.

You can list multiple domains by separating each one with a single space.

Note: You must have the appropriate Verity Spider licensing capability to use this option. The Verity Spider that is included with ColdFusion is licensed for websites that are defined and reside on the same machine on which ColdFusion is installed. Contact Verity Sales for licensing options regarding the use of Verity Spider for external websites.

-followdup

Specifies that Verity Spider follows links within duplicate documents, although only the first instance of any duplicate documents is indexed.

You might find this option useful if you use the same home page on multiple sites. By default, only the first instance of the document is indexed, while subsequent instances are skipped. If you have different secondary documents on the different sites, using the -followdup option lets you get to them for indexing, while still indexing the common home page only once.

-followsymlink

Type: File system only

Specifies that Verity Spider follows symbolic links when indexing UNIX file systems.

-host

Type: Web crawling only

Syntax: `-host name_1 [name_n] ...`

Limits indexing to the specified host or hosts. You must use only complete text strings for hosts. You cannot use wildcard expressions.

You can list multiple hosts by separating each one with a single space. URLs not on the specified host(s) are not downloaded or parsed.

-https

Type: Web crawling only

Lets you index SSL-enabled websites.

Note: You must have the Verity SSL Option Pack installed to use the -https option. The Verity SSL Option Pack is a Verity Spider add-on available separately from a Verity salesperson.

-jumps

Type: Web crawling only

Syntax: `-jumps num_jumps`

Specifies the maximum number of levels an indexing job can go from the starting URL. Specify a number between 0 and 254.

The default value is unlimited. If you see extremely large numbers of documents in a collection where you do not expect them, consider experimenting with this option, in conjunction with the Content options, to pare down your collection.

-nodocrobo

Specifies to ignoreROBOT META tag directives.

In HTML 3.0 and earlier, robot directives could only be given as the file robots.txt under the root directory of a website. In HTML 4.0, every document can have robot directives embedded in the META field. Use this option to ignore them. Use this option with discretion.

-nofollow

Type: Web crawling only

Syntax: `-nofollow "exp"`

Specifies that Verity Spider cannot follow any URLs that match the exp expression. If you do not specify an exp value for the -nofollow option, Verity Spider assumes a value of "*", where no documents are followed.

You can use wildcard expressions, where the asterisk (*) is for text strings and the question mark (?) is for single characters. Always encapsulate the exp values in double quotation marks to ensure that they are properly interpreted.

If you use backslashes, you must double them so that they are properly escaped; for example:

```
C:\\test\\docs\\path
```

To use regular expressions, also specify the -regexp option.

Earlier versions of Verity Spider did not allow the use of an expression. This meant that for each starting point URL, only the first document would be indexed. With the addition of the expression functionality, you can now selectively skip URLs, even within documents.

See also -regexp

-norobo

Type: Web crawling only

Specifies to ignore any robots.txt files encountered. The robots.txt file is used on many websites to specify what parts of the site indexers should avoid. The default is to honor any robots.txt files.

If you are re-indexing a site and the robots.txt file has changed, Verity Spider deletes documents that have been newly disallowed by the robots.txt file.

Use this option with discretion and extreme care, especially in conjunction with the -cgiok option.

See also -nodocrobo and http://info.webcrawler.com/mak/projects/robots/norobots.html.

-pathlen

Syntax: `-pathlen num_pathsegments`

Limits indexing to the specified number of path segments in the URL or file system path. The path length is determined as follows:

- The host name and drive letter are not included; for example, neither www.spider.com:80/ nor C:\ would be included in determining the path length.
- All elements following the host name are included.
- The actual filename, if present, is included; for example, /world.html would be included in determining the path length.
- Any directory paths between the host and the actual filename are included.

Example

For the following URL, the path length would be four:

```
http://www.spider:80/comics/fun/funny/world.html
       <-1->          <2>  <-3-> <---4--->
```

For the following file system path, the path length would be three:

```
C:\files\docs\datasheets
    <-1-><-2-><---3--->
```

The default value is 100 path segments.

-refreshtime

Syntax: `-refreshtime timeunits`

Specifies not to refresh any documents that have been indexed since the timeunits value began.

The following is the syntax for timeunits:

```
n day n hour n min n sec
```

Where n is a positive integer. You must include spaces, and since the first three letters of each time unit are parsed, you can use the singular or plural form of the word.

If you specify the following:

```
-refreshtime 1 day 6 hours
```

Only those documents that were last indexed at least 30 hours and 1 second ago, are refreshed.

Note: This option is valid only with the -refresh option. When you use vsdb -recreate, the last indexed date is cleared.

-reparse

Type: Web crawling only

Forces parsing of all HTML documents already in the collection. You must specify a starting point with the -start option when you use the -reparse option.

You can use the -reparse option when you want to include paths and documents that were previously skipped due to exclusion or inclusion criteria. Remember to change the criteria, or there will be little for Verity Spider to do. This can be easy to overlook when you are using the -cmdfile option.

-unlimited

Specifies that no limits are placed on Verity Spider if neither the -host nor the -domain option is specified. The default is to limit based on the host of the first starting point listed.

-virtualhost

Syntax: `-virtualhost name_1 [name_n] ...`

Specifies that DNS lookups are avoided for the hosts listed. You must use only complete text strings for hosts. You cannot use wildcard expressions. This lets you index by alias, such as when multiple web servers are running on the same host. You can use regular expressions.

Normally, when Verity Spider resolves host names, it uses DNS lookups to convert the names to canonical names, of which there can be only one per machine. This allows for the detection of duplicate documents, to prevent results from being diluted. In the case of multiple aliased hosts, however, duplication is not a barrier as documents can be referred to by more than one alias and yet remain distinct because of the different alias names.

Example

You can have both marketing.verity.com and sales.verity.com running on the same host. Each alias has a different document root, although document names such as index.htm can occur for both. With the -virtualhost option, both server aliases can be indexed as distinct sites. Without the -virtualhost option, they would both be resolved to the same host name, and only the first document encountered from any duplicate pair would be indexed.

Note: If you are using Netscape Enterprise Server, and you have specified only the host name as a virtual host, Verity Spider will not be able to index the virtual host site. This is because Verity Spider always adds the domain name to the document key.

Content options

The following sections describe the Verity Spider content options.

-casesen

Makes processing case-sensitive by specifying that the spider separately process keys that differ only in case. Use only for indexing UNIX servers.

-exclude

Syntax: `-exclude exp_1 [exp_n] ...`

Specifies that files, paths, and URLs matching the specified expression(s) will not be followed. If you use backslashes, you must double them so that they are properly escaped; for example:

```
C:\\test\\docs\\path
```

You can use wildcard expressions, where the asterisk (*) is for text strings and the question mark (?) is for single characters; for example:

```
'/my_doc*/year199?'
```

On Windows, include double quotation marks around the argument to protect special characters, such as the asterisk (*). On UNIX, use single quotation marks. This is only required when you run the indexing job from a command line. Quotation marks are not necessary within a command file (the -cmdfile option).

To use regular expressions, also specify the -regexp option.

To specify a file, path, or URL that you want followed but not indexed, use the -indexclude option. For document types, use the -mimeexclude option instead; for example, specify `-mimeexclude application/pdf` rather than `-exclude *.pdf`.

Note: When specifying a URL, you must use full, absolute paths using the same format that appears in the HTML hyperlink. If the link is relative, you must change it to absolute to use it with the -exclude option.

See also -regexp.

-include

Specifies that only those files, paths, and URLs that match the specified expression or expressions will be followed. If you use backslashes, you must double them so that they are properly escaped; for example:

```
C:\\test\\docs\\path
```

You can use wildcard expressions, where the asterisk (*) is for text strings and the question mark (?) is for single characters; for example:

```
'/my_doc*/year199?'
```

On Windows, include double quotation marks around the argument to protect the special characters, such as the asterisk (*). On UNIX, use single quotation marks. This is only required when you run the indexing job from a command line. Quotation marks are not necessary within a command file (the -cmdfile option).

To use regular expressions, also specify the -regexp option.

If your starting points do not contain the specified -include expressions, nothing will be indexed. The -include option prevents Verity Spider from even following anything that does not match the specified expressions. You might want to use the -indinclude option instead. Where the -include option prevents Verity Spider from even following anything that does not match the specified expressions, the -indinclude option allows Verity Spider to follow what matches the specified expressions, while not indexing.

For document types, use the -mimeinclude option instead; for example, specify `-mimeinclude text/html` rather than `-include *.htm`.

Note: When specifying a URL, you must use full, absolute paths using the same format that appears in the HTML hyperlink. If the link is relative, you must change it to absolute to use it with the -include option.

See also -regexp.

-indexclude

Syntax: `-indexclude exp_1 [exp_n] ...`

Specifies that the files and paths in URLs that match the expressions are not indexed. They are, however, still followed. If you use backslashes, you must double them so that they are properly escaped; for example:

```
C:\\test\\docs\\path
```

You can use wildcard expressions, where the asterisk (*) is for text strings and the question mark (?) is for single characters; for example:

```
'/my_doc*/year199?'
```

On Windows, include double quotation marks around the argument to protect the special characters, such as the asterisk (*). On UNIX, use single quotation marks. This is only required when you run the indexing job from a command line. Quotation marks are not necessary within a command file (the -cmdfile option).

To use regular expressions, also specify the -regexp option.

You would use this option to gather some documents, such as HTML tables of contents, to gain access to other documents for indexing.

Where the -exclude option prevents Verity Spider from even following anything that matches the specified expressions, the -indexclude option allows Verity Spider to follow anything while only skipping that which matches the specified expressions.

For document types, use the -indmimeexclude option instead.

Note: When specifying a URL, you must use full, absolute paths using the same format as appears in the HTML hyperlink. If the link is relative, you must change it to absolute to use it with -indexclude.

See also -regexp.

-indinclude

Syntax: `-indinclude exp_1 [exp_n] ...`

Specifies that only those files and paths in URLs that match the expressions be followed and indexed. If you use backslashes, you must double them so that they are properly escaped; for example:

```
C:\\test\\docs\\path
```

You can use wildcard expressions, where the asterisk (*) is for text strings and the question mark (?) is for single characters; for example:

```
'/my_doc*/year199?'
```

On Windows, include double quotation marks around the argument to protect the special characters, such as the asterisk (*). On UNIX, use single quotation marks. This is only required when you run the indexing job from a command line. Quotation marks are not necessary within a command file (the -cmdfile option).

To use regular expressions, also specify the -regexp option.

Where the -include option prevents Verity Spider from even following anything that does not match the specified expressions, the -indinclude option allows Verity Spider to follow anything while only indexing that which matches the specified expressions.

Example

If you want to index all documents that include "search" in the URL at http://web.verity.com, you cannot use the following:

```
vspider -collection collname -start http://web.verity.com
   -include '*search*'
```

This is because the starting point does not match the the -include option criteria. Instead, use the -indinclude option to follow all documents (unless you have specified any of the exclude options) and index only those documents that match your criteria. Replace the -include option with the -indinclude option in the preceding example.

Note: When specifying a URL, you must use full, absolute paths using the same format that appears in the HTML hyperlink. If the link is relative, you must change it to absolute to use it with the -indinclude option.

See also -regexp.

-indmimeexclude

Syntax: `-indmimeexclude mime_1 [mime_n] ...`

Specifies that only those MIME types that match the expressions be followed but not indexed.

On Windows, include double quotation marks around the argument to protect the special characters, such as the asterisk (*). On UNIX, use single quotation marks. This is only required when you run the indexing job from a command line. Quotation marks are not necessary within a command file (the -cmdfile option).

Use this option to gather some documents, such as HTML tables of contents, to gain access to other documents for indexing. The -mimeexclude option, on the other hand, prevents specified documents from being followed at all. For the mime variable, you can include the asterisk (*) wildcard for text strings; for example:

```
'text/*'
```

You cannot use the question mark (?) wildcard, and the -regexp option does not let you use regular expressions.

-indmimeinclude

Syntax: `-indmimeinclude mime_1 [mime_n] ...`

Specifies that only those MIME types that match the expressions be followed and indexed.

The -mimeinclude option does not let you index desired documents if the starting URL is not followed. For the mime variable, you can include the asterisk (*) wildcard for text strings; for example:

```
'text/*'
```

On Windows, include double quotation marks around the argument to protect the special character (*). On UNIX, use single quotation marks. This is only required when you run the indexing job from a command line. Quotation marks are not necessary within a command file (the -cmdfile option).

You cannot use the question mark (?) wildcard, and the -regexp option does not allow you to use regular expressions.

Example

If you want to index all Word documents at http://web.verity.com, you cannot use:

```
vspider -collection collname -style style_dir -start
   http://web.verity.com -mimeinclude 'application/msword'
```

This is because the starting point does not match the -mimeinclude criteria. You can use the -indmimeinclude option to follow all documents (unless you have specified any of the exclude options) and index only those documents that match your criteria. Replace the -mimeinclude option with the -indmimeinclude option in the preceding example.

-indskip

Syntax: `-indskip HTML_tag "exp"`

Type: Web crawling only

Specifies that Verity Spider follow and parse links, but not index, any HTML document that contains the text of exp within the given HTML_tag. For multiple HTML_tag and exp combinations, use multiple instances of the -skip option.

You can use wildcard expressions, where the asterisk (*) is for text strings and the question mark (?) is for single characters; for example:

```
'/my_doc*/year199?'
```

On Windows, include double quotation marks around the argument to protect the special characters, such as the asterisk (*). On UNIX, use single quotation marks. This is only required when you run the indexing job from a command line. Quotation marks are not necessary within a command file (the -cmdfile option).

If you use backslashes, you must double them so that they are properly escaped; for example:

```
C:\\test\\docs\\path
```

To use regular expressions, also specify the -regexp option.

Example

To skip all HTML documents that contain the word "personnel" in the Title element, while still parsing those documents for links to other documents, use the following:

```
-indskip title "personnel"
```

Example

To avoid indexing directory listing pages, while still parsing the document and path links except for the link to the parent directory, use one of the following, depending on the web server being indexed:

- For Netscape web servers, use the following:

  ```
  -indskip title "*Index of*"
  -nofollow "*parent directory*"
  ```

- For Microsoft Internet Information Server, use the following:

  ```
  -indskip a "*to parent directory*"
  -nofollow "*parent directory*"
  ```

-maxdocsize

Syntax: `-maxdocsize integer`

Specifies the maximum size, in kilobytes, for documents to be indexed. Any documents larger than the value specified by the -maxdocsize option are ignored.

The default is to index documents of any sizes.

-metafile

Type: Web crawling only

Syntax: `-metafile path_and_filename`

Allows you to use a text file to map custom meta tags to valid HTTP header fields. If you use backslashes, you must double them so that they are properly escaped; for example:

```
C:\\test\\docs\\path
```

This means that you can use your own meta tag, in the document, to replace what is returned by the web server, or to insert it if nothing is returned. Currently, the only header fields of real value are "Last-Modified" and "Content-Length." Future enhancements, however, could allow for much greater variety.

The following is the syntax for entries in the text file:

```
name Last-Modified y|n
```

or

```
name Content-Length y|n
```

Where `y|n` is an override flag, which can be yes or no.

Example

A mapping file for the -metafile option might include the following:

```
Doc_Last_Touched Last-Modified n
Doc_Size Content-Length y
```

If you use the y override flag, the value for the custom meta tag overrides the value for the valid field, even if both values are present and differ. This can be useful when the valid field value is always sent, but you want to specify your own value with a custom meta tag.

If you use the n override flag, the value for the custom meta tag is used only if there is no value for the valid field returned by the server. If a value for the valid field exists, then that is given precedence.

Note: If you have several entries mapping to the same valid field, only the last entry takes effect.

-mimeexclude

Syntax: `-mimeexclude mime_1 [mime_n] ...`

Specifies MIME types that are neither followed nor indexed.

On Windows, include double quotation marks around the argument to protect the special characters, such as the asterisk (*). On UNIX, use single quotation marks. This is only required when you run the indexing job from a command line. Quotation marks are not necessary within a command file (the -cmdfile option).

The default is to include all MIME types. For the mime variable, you can include the asterisk (*) wildcard for text strings; for example:

```
'text/*'
```

You cannot use the question mark (?) wildcard, and the -regexp option does not allow you to use regular expressions.

Use the -indmimeexclude option to allow Verity Spider to follow documents, without indexing them, to gain access to other desirable document types.

-mimeinclude

Syntax: `-mimeinclude mime_1 [mime_n] ...`

Specifies MIME types to be included.

On Windows, include double quotation marks around the argument to protect the special characters, such as the asterisk (*). On UNIX, use single quotation marks. This is only required when you run the indexing job from a command line. Quotation marks are not necessary within a command file (the -cmdfile option).

The default is to include all MIME types. For the mime variable, you can include the asterisk (*) wildcard for text strings; for example:

```
'text/*'
```

You cannot use the question mark (?) wildcard, and the -regexp option does not let you use regular expressions.

-mindocsize

Syntax: `-mindocsize integer`

Specifies the minimum size, in kilobytes, for documents to be indexed. Any documents smaller than the value specified by the -mindocsize option are ignored.

The default is to index documents of any sizes.

-skip

Type: Web crawling only

Syntax: `-skip HTML_tag "exp"`

Specifies that Verity Spider not index any HTML document that contains the text of exp within the given HTML_tag. For multiple HTML_tag and exp combinations, use multiple instances of the -skip option.

You can use wildcard expressions, where the asterisk (*) is for text strings and the question mark (?) is for single characters; for example:

```
'/my_doc*/year199?'
```

On Windows, include double quotation marks around the argument to protect the special characters, such as the asterisk (*). On UNIX, use single quotation marks. This is only required when you run the indexing job from a command line. Quotation marks are not necessary within a command file (the -cmdfile option).

If you use backslashes, you must double them so that they are properly escaped; for example:

```
C:\\test\\docs\\path
```

To use regular expressions, also specify the -regexp option.

Example 1

To skip all HTML documents that contain the word "personnel" in the Title element, use the following:

```
-skip title "personnel"
```

Example 2

To skip all HTML documents that contain both the word "private" and the phrase "internal user" in any paragraph element, use the following:

```
-skip title "personnel"
-skip p "*internal use*"
```

See also -regexp.

Locale options

The following sections describe the Verity Spider locale options.

-charmap

Syntax: `-charmap name`

Specifies the character map to use. Valid values are 8859 or 850. The default value is 8859.

-common

Specifies the path to the Verity home directory, *cf_root*/lib/common.

Note: This option is typically not needed, as long as the PATH environment variable is set correctly.

-datefmt

Syntax: `-datefmt format`

Specifies the Verity import date format to use. Valid values are MDY, DMY, YMD, USA and EUR. The default value is MDY. (For descriptions of these values, see "Date format options" on page 16.)

-language

Syntax: `-language name`

Specifies the Verity locale to use in indexing. This option is being replaced by the semantically consistent the -locale option, and is still supported for backwards compatibility.

-locale

Syntax: `-locale name`

Specifies the Verity locale to use in indexing, such as German (deutsch) or French (français). The default is English (english). This option is identical to the -language option.

-msgdb

Syntax: `-msgdb path`

Specifies the path to the ind.msg message database file.

If Verity Spider was installed properly, this option should be unnecessary. By default, the ind.msg message database file is read from the following directory:

cf_root/lib/*platform*/bin

Where *platform* represents the platform directory.

Logging options

The following sections describe the Verity Spider logging options.

-loglevel

Syntax: `-loglevel [nostdout] argument`

Specifies the types of messages to log. By default, messages are written to standard output and to various log files in the subdirectory named /log beneath the Verity Spider job directory. If you add nostdout to the -loglevel option, messages are not written to standard output. Log files, however, are still created.

The following table describes valid message types:

Message type	Description
information	Licensing information written to info.log. Included with all arguments.
warning	Warning messages written to warning.log. Included with all arguments.
error	Error messages written to error.log. Included with all arguments.
badkey	Messages regarding keys that could not be indexed due to invalid documents, written to badkey.log. Included with all arguments.
progress	Current state of a document key written to progress.log. Note that a key with a progress of "inserting" might be a badkey and therefore skipped, rather than an indexed key. Included with all arguments.
summary	Inserted, indexed, and ignored messages written to summary.log. Included with all arguments except skip.
skip	Skipped documents, with explanation, written to skip.log. Included with all arguments, except summary.
debug	Internal Verity Spider processing messages, such as enqueued, written to debug.log. Included with both debug and trace arguments.
trace	Internal Verity Spider processing messages written to debug.log. Included only with the trace argument.

Choose one of the following arguments to determine which message types are logged:

Loglevel arguments	Description
summary	Includes the following message types: information, warning, error, badkey, progress, summary Use this option only if you do not want skip type messages.
skip	Includes the following message types: information, warning, error, badkey, progress, skip Use this option only if you do not want summary type messages.
verbose	Includes the following message types: information, warning, error, badkey, progress, summary, skip

Loglevel arguments	Description
debug	Includes the following message types: information, warning, error, badkey, progress, summary, skip, debug Note: Only use this argument at the direction of Verity technical support or for troubleshooting indexing problems.
trace	Includes the following message types: information, warning, error, badkey, progress, summary, skip, debug, trace Note: Only use this argument at the direction of Verity technical support or for troubleshooting indexing problems.

Maintenance options

The following sections describe the Verity Spider maintenance options.

-nooptimize

Prevents Verity Spider from optimizing the collection, thus reducing processing overhead during indexing. Use this option sparingly, as it leaves the collection in less than optimum shape. The following are some examples of when you might want to use this option:

- You want to manually perform custom optimization of the collection, using the mkvdk utility. By default, the Verity Spider optimization mimics the mkvdk actions of maxmerge and vdbopt. For more information on the mkvdk utility, see V*erity Collection Building Guide* and Chapter 2, "Managing Collections with the mkvdk Utility" on page 9.
- You are running multiple indexing jobs against a collection, and want to wait until they are all finished to optimize.

Generally, you should not leave a collection unoptimized for too long, as search times can slow significantly.

In brief, optimizing a collection means creating a small number of large partitions, which can greatly reduce search times.

-purge

Deletes document tables and index files in the collection, and cleans up the collection's persistent store. The collection is then fresh with its original style files, and is not deleted from the file system.

-repair

Specifies a failure-recovery mode for the collection, where the goal is to determine the causes of any errors, repair the errors (if possible), and restart a collection.

Although the Verity indexing engine always leaves the collection in a consistent, usable state, and no data can be lost or corrupted due to machine failures, it is possible for a process or event external to the Verity engine to corrupt one or more collections.

You can use the -repair option for constant failure-recovery operation, or you can run it selectively on collections that failed.

Setting MIME types

You can use the MIME type criteria options, -mimeinclude, -indmimeinclude, -mimeexclude, and -indmimeexclude, to include or exclude MIME types.

Syntax restrictions

When you specify MIME type criteria, keep in mind the restrictions described in the following sections.

Using the wildcard character (*)

The asterisk (*) wildcard character does not operate as a regular expression for the value of the MIME type criteria. Instead, you can only use it to replace the entire MIME type or MIME sub-type.

For example, the following value is a valid substitute for text/html:

```
text/*
```

The following value is NOT a valid substitute for text/html:

```
text/h*
```

Multiple parameter values

When you specify a series of parameter values for a single instance of one of the MIME type criteria, and you use quotation marks, you must enclose each separate parameter value in single quotation marks. For example:

```
-mimeinclude 'text/plain' 'application/*'
```

If you enclose the entire sequence of parameter values, as follows:

```
-mimeinclude 'text/plain application/*'
```

Verity Spider considers the entire expression a single value.

You can also use multiple instances of the MIME type criteria, each with a single parameter value, where quotation marks are necessary only if you use the wildcard character (*). For example:

```
-mimeinclude text/plain
-mimeinclude 'application/*'.Setting MIME Types
```

MIME types and web crawling

When you index a website, Verity Spider evaluates your MIME type criteria against the "Content-Type" HTTP headers sent by the web server hosting that website. That web server passes along MIME type information based on its own internal tables.

When you encounter MIME types being dropped, make sure that the web server you are indexing has the necessary MIME type information. For information about specifying MIME types, see the documentation for your web server.

You can examine the indexing job's log files for indications that files are being skipped due to MIME types. For example, a typical ASCII file you might want indexed is a log file (filename.log). Unless the web server understands that files with .LOG extensions are ASCII text, of MIME type text/plain, you will see in the indexing job log file that .LOG files are skipped because of MIME type, even if you use the following:

```
-mimeinclude 'text/*'
```

MIME types and file system indexing

When you index a file system, Verity Spider reads filenames and evaluates your MIME type criteria against an internal, compiled list of known MIME types and associated file extensions. You cannot edit this list. However, you can use the -mimemap option to create a custom MIME type mapping.

When you encounter MIME types being dropped, check whether Verity Spider recognizes that particular MIME type. For more information, see the table, "Known MIME types for file system indexing" on page 57.

You can examine the indexing job's log files for indications that files are being skipped due to MIME types. For example, a typical ASCII file you might want indexed is a log file (filename.log). Since Verity Spider does not understand that files with .LOG extensions are ASCII text, of MIME type text/plain, you will see in the indexing job log file that .LOG files are skipped because of MIME type, even if you use the following:

```
-mimeinclude 'text/*'.Setting MIME Types
```

Indexing unknown MIME types

Whenever you find MIME types being dropped, or you know you will be indexing files whose extensions are not known to Verity Spider by default, use the -mimemap option to point to a file that contains your own custom mappings for file extensions and MIME types.

You can also use the regular expression '*/*' for your MIME type criteria; for example:

```
-mimeinclude '*/*'
```

On either platform, you must include single quotation marks for values that include wildcard characters.

Also use inclusion and exclusion criteria to finely control what is indexed, as follows:

- If your list of file types to index is rather long, use exclusion criteria (-exclude, -indexclude, -mimeexclude, or -indmimeexclude) to exclude extensions you know you do not want to index; for example:

  ```
  -exclude '*.exe' '*.com'
  ```

- If the list of file types you want to index is relatively small, use inclusion criteria (-include, -indinclude, -mimeinclude, or -indmimeinclude) to specify them; for example:

  ```
  -include '*.txt' '*.1st' '*.log'.Setting MIME Types
  ```

Known MIME types for file system indexing

The following table lists the MIME types that Verity Spider recognizes when indexing file systems:

Format	MIME type	Extension
HTML	text/html	htm, html
ASCII	text/plain	txt, text
ASCII, source files	text/plain	c, h, cpp, cxx
PDF	application/pdf	pdf
MS Word	application/msword	doc
MS Excel	application/excel	xls
MS PowerPoint	application/vnd.ms-powerpoint	ppt
WordPerfect 5.1	application/wordperfect5.1	wpd
RTF	application/rtf	rtf
FrameMaker MIF	application/vnd.mif	mif

CHAPTER 4
Searching Collections with the rcvdk Utility

This chapter provides information about using the rcvdk utility to search Verity collections.

Contents

Using the Verity rcvdk utility

Using the Verity rcvdk utility, you can check the contents of a collection from the command line. The rcvdk utility lets you write a variety of queries, using words and phrases separated by commas and Verity query language. A viewing option lets you see document contents and highlights in a simple text display.

The rcvdk executable is located in the *cf_root*\lib_nti40\bin directory in Windows, and in the *cf_root*/lib/*platform*/bin directory in UNIX.

In these pathnames, *cf_root* refers to the ColdFusion root directory. In Windows, this is typically C:\CFusionMX; in UNIX, this is typically /opt/coldfusionmx. In UNIX, *platform* refers to the UNIX version of the server that runs ColdFusion: _solaris, _hpux11, or _ilnx21.

To start the rcvdk utility on most systems, type the path and executable name at a command prompt. The following examples assume you have set your PATH variable, so you just have to enter `rcvdk` at a command prompt to run it.

For example:

```
c:\cfusionmx\lib\platform\bin\rcvdk /common = c:\cfusionmx\lib\common
```

When you start the rcvdk utility with no arguments, you get the following message, followed by the rcvdk prompt:

```
Type 'help' for a list of commands.
RC>
```

The `help` command produces the following list of available commands:

```
RC> help
Available commands:
search     s Search documents.
results    r Display search results.
clusters   c Display clustered search results.
view       v View document.
summarize  z Summarize documents.
attach     a Attach to one or more collections.
detach     d Detach from one or more collections.
quit       q Leave application.
about      Display VDK 'About' info
help       ? Display help text; 'help help' for details.
expert     x Toggle expert mode on/off.
RC>
```

You can enter the letter q at the RC prompt at any time to quit the application.

Attaching to a collection using the rcvdk utility

To search a collection, you first must attach to it using the `attach (a)` command. This command must include the pathname to a collection directory as an argument. After you press Return, the rcvdk utility reports whether the `attach` command was successful; for example:

```
RC>a /z/doc1/c/public/Collection/file_walking/collbldg/html
Attaching to collection:
/z/doc1/c/public/Collection/file_walking/collbldg/html
Successfully attached to 1 collection.
RC>
```

The rcvdk utility lets you attach to one or more collections. The specified collections remain attached until you detach from one or more collections using the `detach (d)` command.

Basic searching

To retrieve all documents, use the `search (s)` command without arguments. After you press Return, a search update message is produced, as follows:

```
RC>s
Search update: finished (100%). Retrieved: 85(85)/85.
RC>
```

The search results indicate that 85 of the total 85 documents in the collection were retrieved. If you specify a query argument, such as "universal filter," a subset of the total documents in the collection that contain the specified string is retrieved; for example:

```
RC>s universal filter
Search update: finished (100%). Retrieved: 18(18)/85.
RC>
```

In the message returned for the preceding search, the rcvdk utility indicates that 18 documents matched the query. You can perform more elaborate queries using the Verity query language, as shown in the following example:

```
RC>s universal filter <OR> filter.Troubleshooting and Maintenance Tools
```

Viewing results of the rcvdk utility

After you have attached to a collection and issued a `search` command successfully, you can view the results list and look at the retrieved documents. You can use the options in the following table:

Option	Description
r	Displays the results list, starting with the first document. A maximum of 24 documents are displayed.
r n	Displays the results list, starting with the nth document. A maximum of 24 documents are displayed.
v	Displays the first or next document in the results list. Highlights are indicated using reverse video, if possible. If not, double angle brackets are used, as in: `>>universal<< >>filter<<` To exit the document display, enter the letter q.
v n	Displays the nth document in the results list. To exit the document display, enter the letter q.

The following is the results list for the "universal filter" search. For each document, these fields are displayed by default: Number, Score, and VdkVgwKey.

```
RC> r
Retrieved: 18(18)/85
Number SCORE VdkVgwKey
1: 1.00 d:\search97\s97is\locale\english\doc\collbldg\08_cbg3.htm
2: 0.97 d:\search97\s97is\locale\english\doc\collbldg\11_cbg2.htm
3: 0.97 d:\search97\s97is\locale\english\doc\collbldg\08_cbg7.htm
4: 0.97 d:\search97\s97is\locale\english\doc\collbldg\08_cbg1.htm
5: 0.95 d:\search97\s97is\locale\english\doc\collbldg\cbgtoc.htm
6: 0.95 d:\search97\s97is\locale\english\doc\collbldg\08_cbg4.htm
7: 0.93 d:\search97\s97is\locale\english\doc\collbldg\cbgix.htm
8: 0.92 d:\search97\s97is\locale\english\doc\collbldg\08_cbg6.htm
9: 0.90 d:\search97\s97is\locale\english\doc\collbldg\08_cbg.htm
10: 0.90 d:\search97\s97is\locale\english\doc\collbldg\04_cbg1.htm
11: 0.90 d:\search97\s97is\locale\english\doc\collbldg\01_cbg1.htm
12: 0.87 d:\search97\s97is\locale\english\doc\collbldg\f_cbg.htm
13: 0.87 d:\search97\s97is\locale\english\doc\collbldg\08_cbg2.htm
14: 0.84 d:\search97\s97is\locale\english\doc\collbldg\06_cbg1.htm
15: 0.80 d:\search97\s97is\locale\english\doc\collbldg\part4.htm
16: 0.80 d:\search97\s97is\locale\english\doc\collbldg\f_cbg1.htm
17: 0.80 d:\search97\s97is\locale\english\doc\collbldg\11_cbg5.htm
18: 0.80 d:\search97\s97is\locale\english\doc\collbldg\08_cbg5.htm
RC>
```

The following table describes each of the default fields:

Field name	Description
Number	The rank of the document in the results list. The document with the highest score is ranked number 1.
Score	The score assigned to each retrieved document, based on its relevance to the query. For a NULL query, no scores are assigned, so the Score column in the results list is blank.
VdkVgwKey	The document key used by the Verity engine to manage the document. If the document is accessed through the file system, the primary key is a pathname. If the document is accessed through a web server, using HTTP, the primary key is a URL.

Displaying more fields

You can tell the rcvdk utility to display certain fields in the results list using the `fields` command, which is available in the expert mode. To go to the expert mode, enter `x` or `expert` at the RC prompt, then press Return.

All fields in a column are blank if the field is not defined for the collection's schema in the documents table (in style.ddd, style.sfl, or style.ufl). A field in a document's row is blank if the field was not populated by a gateway, bulk submit action, or filter.

Displaying a field

The `fields` command includes the field name and length to be displayed. When used, the `fields` command overrides the default Score and VdkVgwKey fields for the results list.

The search engine returns fields for the results list, so if you do a search, then go to expert mode to use the `fields` command, you must run the search again in order to see the results list with the fields you requested. For example:

```
RC> expert
Expert mode enabled
RC> fields title 20
RC> s universal filter
Search update: finished (100%). Retrieved: 18(18)/85.
RC> r
Retrieved: 18(18)/85
Number title
1: Using the Universal Filter
2: Using the Zone Filter
3: The Zone Filter
4: Overview
5: Table of Contents
6: Universal Filter Configuration Using the
7: Index
8: The PDF Filter
9: Document Filters and Formatting
10: Collection Style Summary
11: Collection Basics
12: Universal Filter Document Types
```

```
13: Using the style.dft File
14: Supported Field Types
15:
16: Recognized Document Types
17: Custom Zone Definitions
18: The KeyView Filter Kit
RC>
```

Displaying multiple fields

You can specify multiple fields with the `fields` command, as shown in the following example. The field order corresponds to the order of the columns, with the first field specified appearing in the second column. The first column is reserved for the rank order.

Rerun the search before you display the results list with the fields specified.

For example:

```
RC> fields score 5 title 40
RC> s universal filter
Search update: finished (100%). Retrieved: 18(18)/85.
RC>
```

CHAPTER 5
Searching Collections with K2 Server

This chapter provides information about how to configure the Verity K2 Server, which is installed with ColdFusion Server.

Contents

Using K2 Server

You configure K2 Server to work with ColdFusion with the following steps:

1 Edit the k2server.ini file to specify the alias collection names you want to expose to K2 Server. (See "Editing the k2server.ini file" on page 66.)

2 Start K2 Server by running the `k2server` executable. (See "Starting K2 Server" on page 68.)

3 Specify hostname and port information for K2 Server. (See "Specifying K2 Server parameters in the ColdFusion Administrator" on page 68.)

Editing the k2server.ini file

To enable a collection for searching using K2 Server, you must first configure the k2server.ini file. This file is located in the *cf_root*\lib\ directory in Windows, and in the *cf_root*/lib/ directory in UNIX.

In these pathnames, *cf_root* refers to the ColdFusion root directory. In Windows, this is typically C:\CFusionMX; in UNIX, this is typically /opt/coldfusionmx.

The k2server.ini file consists of several parameters that typically remain unchanged. You must verify or make minor edits to settings in the portNo, vdkHome, and Coll-n sections.

To edit the k2server.ini file:

1 Open the k2server.ini file in your text editor.

Tip: Use your text editor's search function to locate the appropriate code. For example, to locate the settings for the port number, as described in the next step of this procedure, search for portNo=.

Note: If you did not install ColdFusion into the default directory, edit the paths in this procedure to reflect the appropriate directories.

2 In the code for portNo=, verify that the value matches the value for the K2 Server port. The default value is 9901:

```
##portNo: TCP port number for client connections.
portNo=9901
```

3 (Required only if you run K2 Server as a Windows service) In the code for vdkHome=, verify that the value matches the location of the Verity common directory. This is the *cf_root*\lib\common directory in Windows, and is the *cf_root*/lib/common directory in UNIX.

```
## vdkHome: directory containing Verity resources (common directory)
   - need it running as an NT service
## vdkHome=c:/cfusionmx/lib/common
```

Note: If you run K2 Server as a Windows NT service, you must remove the pound signs (##) in the highlighted line above to uncomment the code. If the line remains as a comment, K2 Server will not execute correctly.

4 In the code for [Coll-0], specify in the `collPath` parameter the directory of a collection that K2 Server will search:

```
[Coll-0]
collPath=c:\cfusionmx\verity\collections\test_01\file
collAlias=test_01_file
```

The `collPath` value must point to an existing Verity collection; the k2server executable cannot be used to create a collection.

Note: The final subdirectory for your collPath might differ, based on whether it is an external collection (that is, a native Verity tool created it) or ColdFusion created it. If ColdFusion created the collection, there are file and custom subdirectories; these subdirectories are not present in external collections. For more information, see "Collection structure and ColdFusion" on page 3.

5 In the next line, specify a collection alias in the `collAlias` parameter:

```
[Coll-0]
collPath=c:\cfusionmx\verity\collections\test_01\file
collAlias=test_01_file
```

You use this value to reference the collection in CFML.

Note: Collection alias values must be unique. They must be different from any collection names managed by ColdFusion.

The following CFML code performs a K2 mode search on the test_01_file collection:

```
<cfsearch
   collection="test_01_file"
   name="getData"
   criteria="#form.criteria#">
```

Note: To search multiple collections, use a comma-delimited list. For example, use `collection="test_01_file,test_02_file"` in your `cfsearch` tag. Within a single `cfsearch` tag, the collections must be either all K2 Server-registered or all ColdFusion-registered; you cannot use one `cfsearch` tag to search a K2 Server-registered collection and a ColdFusion-registered collection.

In the following example, the `collPath` value points to a collection for the ColdFusion online documentation:

```
[Coll-1]
collPath=c:\cfusionmx\verity\collections\cfdocumentation\custom
collAlias=cfdoc_custom
topicSet=
knowledgeBase=
onLine=2
```

6 (Optional) Create a `Coll-n` section for other collections that you want to search with K2 Server. For each entry, increment the value *n* by one. The first collection is number 0, not number 1, as in the following example:

```
[Coll-2]
collPath=C:\cfusionmx\Verity\Collections\bbb\file
collAlias=bbb_file
```

7 Stop and restart K2 Server for changes in the k2server.ini file to take effect. For more information, see "Stopping K2 Server" on page 69.

For more information about k2server.ini parameters, see "The k2server.ini parameter reference" on page 70.

Starting K2 Server

You start K2 Server from the command line in UNIX or Windows. In UNIX, you run the startk2server script; in Windows, you run the startk2server.bat file. These files are located in the *cf_root*\lib\ directory in Windows, and in the *cf_root*/lib/ directory in UNIX.

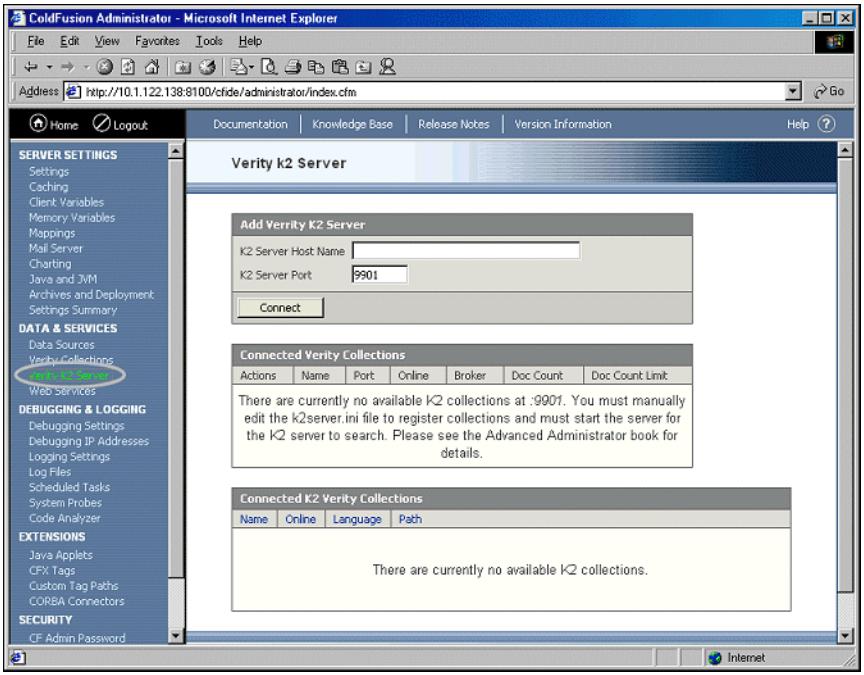
In Windows, you can start K2 Server as a service by entering the following command in the *cf_root*\lib\ directory:

```
k2server -ntservice 1 -inifile k2server.ini
```

Note: Macromedia does not recommend running K2 Server as a Windows service. You must stop the service before you modify or delete collections registered with K2 Server. You must then remember to restart the service. You must also verify that the vdkHome information in your k2server.ini file is uncommented—that is, it has no leading pound (#) signs—and points to the correct location of the common directory.

Specifying K2 Server parameters in the ColdFusion Administrator

You use the Verity K2 Server page in the ColdFusion Administrator to specify the hostname and port number for the K2 Server you want to use, as the following figure shows:



Make sure that you started K2 Server on the host you specify in the Verity Server hostname field. Also, the port number you enter must match the port number you specify in the k2server.ini file.

Stopping K2 Server

You can run K2 Server as a Windows service or in a command window, as an ordinary application. Unless you use the `-ntService 1` option when starting K2 Server, K2 Server runs in the command window. There are several ways to stop K2 Server, depending on how it runs.

Stopping K2 Server when run as a service

To halt K2 Server when it is running as a Windows service, do either of the following:

- Open the Services Control Panel and stop the K2 Server service.
- Open a command window and enter the following command:

```
k2server -ntService 0
```

Stopping K2 Server when run as an application

When K2 Server is running as an application in a command window, you stop it by pressing Ctrl+C to kill the process in the window where it is running.

Stopping K2 Server on UNIX

The ColdFusion installation includes a script that you run to halt K2 Server. By default, the stopk2server script is located in the /*cf_root*/lib directory.

The k2server.ini parameter reference

The K2 Server configuration file, k2server.ini, contains many sections. The first section, [server], provides parameters that control the behavior of the entire server. Each subsequent collection section (in the form `[Coll-1]`, `[Coll-2]`, and so on) controls each collection and search service configured for the server.

Server section

The following table describes the parameters that you can use in the server section of the server configuration file. The K2 Server executable includes a sample configuration file (k2server.ini).

Parameter	Description
serverAlias	An arbitrary name used to identify the server.
numThreads	The default number of search threads to be started in the server process. If too many threads exist, the system can run out of memory; if too few threads exist, searches will be blocked and forced to wait for a Verity engine thread to become available. The value of numThreads is based on hardware resources and system needs.

maxFiles	The maximum number of file handles that can be opened by a specific search thread. The default value for maxFiles is dependent on the limits of the OS used. The maxFiles value affects how file handles are shared between the operating system and the search engine. The maxFiles and numThreads values together can be used to tune system performance.
	The following values can be set for a server:
	`[server]`
	`numThreads=4`
	`maxFiles=100`
	These entries for a K2 Server cause the system to support a maximum of 4 concurrent searches, with 100 file handles allocated for each search thread. The search engine determines default values per operating system. For large or fragmented collections, Macromedia recommends that you explicitly set a value for maxFiles.

portNo	The TCP port number for client connections. The value of portNo is the same value assigned to portNo in the k2broker.ini file that identifies the broker referring to this server.
numListeners	The maximum number of clients that can connect to the server at one time. The numListeners value must be equal to or greater than the sum of all numThreads values specified by all K2 Brokers in the K2 search system. The numThreads value is set for a K2 Broker in the k2broker.ini file.

broker(n)	The brokers to ping on startup. Multiple brokers can be specified. For example:
	`broker(1)=machinea:9900`
	`broker(2)=machineb:9901`

maxColSize	The maximum width of the fields to return to the results list, in bytes. The default is 2048 bytes.

Search thread keywords

The following table describes keywords that you can use in your search threads:

Keyword	Description
vdkHome	The directory containing Verity resources.
vdkSortingFlag	A flag indicating whether the Verity engine sorts at the collection level. Valid values include: • **NO, False, or 0** Do not perform sorting at the collection level (default) • **YES, True, or 1** Perform sorting at the collection level. To implement sorting at the collection level, you must set vdkSortingFlag to YES in the k2server.ini file (in the server section) and the k2broker.ini file (in the broker section).
sortTruncDocs	The maximum number of documents to consider when sorting.
accessProfile	The Security Access Profile specified in the form of a query expression. The security access profile represents the access question that a document must pass in order for users to have access to it.
topicSet	The default pathname to a directory for the default topic set, which is an indexed set of topics. The value of the topicSet parameter identifies the default topic set to make available by every search service to clients at startup .
knowledgeBase	The default pathname to a knowledgeBase map file, which identifies numerous topic sets (indexed topics). The value of the knowledgeBase parameter identifies the topic sets (multiple) to make available to clients for every search service at startup).
charMap	A string that names the character set to use for strings that are sent to the server and enerated by the server. This string must match the name of a .cs file in the root of the common directory that configures a character set and its mappings. For example, if your application uses character set 8859 for all of its interactions with the server, then set this charMap parameter to the string 8859. Valid values include, but are not limited to, the character sets supplied by Verity: 850 (default) for code page 850; 8859 for code page 8859.
locale	The name of the locale (combination of language, dialect, and character set) to use for all internal Verity engine operations. This name must correspond to a subdirectory in the common directory where the configuration file for the locale is found and where the message database and other locale-specific files are located. Leaving this parameter null means that the server uses the default internal locale, which is “english” written in the “850” character set.
resultCacheTimeout	The timeout in milliseconds for the result cache. The timeout occurs after 60 seconds or when the cache overflows based on resultCacheQuota.

Keyword	Description
resultCacheQuota	The number of slots per segment for the result cache. The result cache is composed of 16 segments, each of which has a number of slots for caching items : K2SearchNew, K2SearchRecv, K2DocReadBatch. The timeout occurs after `resultCacheQuota value * 16`. If `resultCacheQuota=10`, each of the segments has 10 slots. Since a search operation involves a call to K2SearchNew and a call to K2SearchRecv, an additional slot is used.
resultCacheEnabled	A flag indicating whether the result cache is enabled. Valid values include: • **Yes, True, or 1** Enables the result cache. • **No, False, or 0** Disables the result cache (default). By default, the cache is not enabled.
resultCacheMaxInBytes	Amount of memory, in bytes, to use for the cache.

Collection sections

The K2 Server initializes a separate search service for each collection that you identify in the server configuration file. To add one or more collections to the configuration file, enter a separate block of keywords for each collection, in the following format:

```
[Coll-n]
collPath=<pathname>
collAlias=<value>
topicSet=<topicset>
knowledgeBase=<knowledgeBase>
numThreads=<value>
maxFiles=<value>
onLine=<value>
maxColSize=<value>
locale=<language>
charMap=<charmap>
inputDateFormat=<format>
```

Increment the block label for each collection that you configure, starting with Coll-0. The following table describes the keywords used to configure each collection and search service:

Keyword	Description
collPath	The pathname identifying the collection home directory.
collAlias	An arbitrary name used to identify the collection.
topicSet	The pathname to a directory for the default topic set, which is an indexed set of topics. The value of the topicSet parameter identifies the default topic set to make available to clients by every search service at startup. If not specified, the value of topicSet from the server section is used.

Keyword	Description
knowledgeBase	The pathname to a knowledgeBase map file, which identifies numerous topic sets (indexed topics). The value of the knowledgeBase parameter identifies the topic sets (multiple) to make available to clients for every search service at startup. If not specified, the value of the knowledgeBase parameter from the [server] section is used.
numThreads	The number of concurrent searches for the collection. If not specified, the value of numThreads from the [server] section is used.

maxFiles	The maximum number of files that can be opened by a specific search thread for a collection. If not specified, the value of the maxFiles parameter from the server section is used. The maxfiles and numThreads values together can be used to tune system performance. The following values can be set for a collection:
	`[Coll-0]`
	`numThreads=4`
	`maxFiles=100`
	These entries for collection 0 cause K2 Server to support a maximum of 4 concurrent searches, with 100 file handles allocated for each search thread.

onLine	A flag indicating whether the server starts up with the collection on-line. Valid values include: • **0** Start the server with the collection offline • **1** Start the server with the collection in a hidden state • **2** Start the server with the collection online (default). In the hidden state, collections can be primed and tested, but are not yet available for searching by users. When collections are set offline, any queries currently running complete using these resources; subsequent queries do not see the resource.
maxColSize	The maximum width of the fields to return to the results list, in bytes. If not specified, the value of maxColSize from the server section is used.
locale	The name of the locale (combination of language, dialect, and character set) to use for all internal Verity engine operations. This name must correspond to a subdirectory in the common directory where the configuration file for the locale is found and where the message database and other locale-specific files are located. If not specified, the value of the locale parameter from the server section is used.

Keyword	Description
charMap	A string that names the character set to use for strings that are sent into the server and generated by the server. This string must match the name of a .cs file in the root of the common directory that configures a character set and its mappings. If not specified, the value of the charMap parameter from the server section is used. For example, if your application uses character set 8859 for all of its interactions with the server, set this charMap parameter to the string 8859. Valid values include, but are not limited to, the character sets supplied by Verity: 850 (default) for code page 850; 8859 for code page 8859
inputDateFormat	The input date format to be used. If there is no specified value for the inputDateFormat parameter, the default is MDY (Month-Day-Year), a numeric format.

Using the rck2 utility to search K2 Server documents

The rck2 command-line utility lets you search collections associated with a K2 Server in a K2 Search System. The rck2.exe file, which starts the rck2 utility, is located in the *cf_root*\lib_nti40\bin directory in Windows, and in the *cf_root*/lib/*platform*/bin directory in UNIX.

In these pathnames, *cf_root* refers to the ColdFusion root directory. In Windows, this is typically C:\CFusionMX; in UNIX, this is typically /opt/coldfusionmx. In UNIX, *platform* refers to the UNIX version of the server that runs ColdFusion: _solaris, _hpux11, or _ilnx21.

rck2 syntax

Use the following syntax to start rck2 from the command line:

```
rck2 -server <servername> -port <portno>
```

For example: `c:\cfusionmx\lib\_nti40\bin\rck2 -server localhost -port 9901.`

The following table describes rck2 syntax elements:

Syntax element	Description
-server <servername>	The server name for K2 Server to which to attach. The server name is defined in the k2server.ini file. The rck2 utility searches the collections attached to this server.
-port <portno>	The port number where K2 Server (specified by -server) is running.

rck2 command options

The following table describes rck2 command options:

rck2 command	Description
p <sortspec>	The sort specification for the search results. By default, results are sorted by Score. Multiple fields must be specified in a space-separated list using asc or desc to indicate ascending or descending order. For example: `p score desc title asc`
m <maxdocs>	The maximum number of documents to return in the results list.
c <collections>	The list of collections to search. Multiple collections must be specified in a space-separated list. For example: `c coll1 coll2 coll3`
f <fields>	The list of fields to retrieve. For example: `f k2dockey title date`
s <query text>	The query (or question) to be used to process the search. The query can be expressed as words and phrases separated by commas. Additionally, the query can include Verity query language, operators and modifiers.
g <collection>	Display collection information.
d <k2dockey>	Display fields for the K2 document key specified.
v <k2dockey>	Stream the document and display it with highlights.

rck2 command	Description
r ‹docstart›	Display results starting with the first result in the results list. Fields specified using the f command are displayed. Docstart indicates the first result to be displayed. For example, `r 10` displays results starting with the 10th document in the results list.
b ‹docstart›	Display results based on the last field selection.
i	Display information about K2 Server, including nodes and collections.
x ‹score precision›	Set score precision to 8- or 16-bit. By default, 16-bit precision is used.
h or ?	Display online Help for the rck2 command options.

CHAPTER 6
Troubleshooting Collections with Verity Utilities

This chapter provides information about using Verity utilities to configure, maintain, and troubleshoot Verity collections.

Contents

Overview of Verity utilities

The following command-line utilities are included with ColdFusion for performing a variety of operations on Verity collections:

Verity utility	Description	For more information
rcvdk	Search collections and display documents.	See Chapter 4, "Searching Collections with the rcvdk Utility" on page 59.
rck2	Search K2 Server collections.	See "Using the rck2 utility to search K2 Server documents" on page 75.
mkvdk	Create and maintain collections.	See Chapter 2, "Managing Collections with the mkvdk Utility" on page 9.
didump	View collection word lists.	See "Using the Verity didump utility" on page 79.
browse	Browse documents table and search results.	See "Using the Verity browse utility" on page 82.
merge	Combine collections.	See "Using the Verity merge utility" on page 84.

Using the Verity didump utility

Using the didump utility, you can view key components of the word index per partition. The word list consists of a list of all words indexed by the Verity engine. The zone list is a list of all zones found by the Verity engine. The zone attribute list is a list of the zone attributes found by the Verity engine.

The didump executable, which starts the didump application, is located in the *cf_root*\lib_nti40\bin directory in Windows, and in the *cf_root*/lib/*platform*/bin directory in UNIX.

In these pathnames, *cf_root* refers to the ColdFusion root directory. In Windows, this is typically C:\CFusionMX; in UNIX, this is typically /opt/coldfusionmx. In UNIX, *platform* refers to the UNIX version of the server that runs ColdFusion: _solaris, _hpux11, or _ilnx21.

For example:

```
c:\cfusionmx\lib\platform\bin\didump /common = c:\cfusion\verity\common
         -pattern llama
c:\new\parts\00000001.did
```

Viewing the word list with the didump utility

You can view the contents of the word list for a partition by using the didump utility with the -words flag. The command-line syntax must include the -words flag and a pathname to a partition file, like the following:

```
didump -words /z/collbldg/html/parts/00000003.did
```

An alphabetical listing of the words in the word index displays, as follows:

```
didump - Verity, Inc. Version 2.5.0 (_nti31, Jul 7 1999)

Text            Size    Doc    Word
A               10      3      4
a               34      5      24
abbreviations   4       1      1
about           4       1      1
acronym         5       1      2
acronyms        4       1      1
actual          4       1      1
administrator   3       1      1
advance         3       1      1
all             8       2      3
also            9       2      4
Always          4       1      1
always          9       2      3
ampersand       4       1      1
```

The columns in the display indicate the following:

- **Size** The number of bytes used by the Verity engine to store information about the word
- **Doc** The number of unique documents in which the word appears
- **Word** The total number of occurrences of a word for the partition

To view the occurrences of a specific word or pattern, enter a command using the -pattern option, as in the following example:

```
didump -pattern acronym 00000003.did
```

In this example, the didump utility displays information about the number of occurrences of the word **acronym**. You can display the individual occurrences of a word using the -verbose option.

Viewing the zone list with the didump utility

The zone list contains a list of the zones identified by the zone filter. You can search the zones listed using the Verity IN operator in a query. To view the contents of the zone list, use the didump utility with the -zones flag plus the pathname to a partition, like the following:

```
didump -zones /z/collbldg/html/parts/00000003.did
```

This partition is for a collection containing the Verity Collection Building Guide in HTML format. The Verity universal filter invoked the HTML filter by default, and indexed the documents using these zones.

```
didump - Verity, Inc. Version 2.5.0 (_solaris, Jul 07 1999)

ZoneName    Fmt    Size   Doc   Regions
A           Wct    10239  85    5016
ADDRESS     Array  34     1     1
BODY        Array  197    85    85
CAPTION     Wct    298    31    85
CODE        Wct    3868   66    1829
H1          Array  80     83    83
H2          Wct    646    53    212
H3          Wct    517    49    171
H4          Wct    128    8     47
HEAD        Array  70     85    85
HTML        Array  165    85    85
TITLE       Array  70     85    85
```

The columns in the display indicate the following:

- **Fmt** The internal data format used to store the zone information.
- **Size** The number of bytes used by the Verity engine to store information about the zone.
- **Doc** The number of unique documents in which the zone appears
- **Region** The total number of instances of a zone for the partition

Viewing the zone attribute list with the didump utility

The zone attribute list contains a list of the HTML attributes for the zones identified by the HTML zone filter. You can search the zone attributes listed using the Verity IN operator together with the WHEN operator in a query. To view the contents of the zone attributes list, use the didump utility with the -attributes flag plus the pathname to a partition, like the following:

```
didump -attributes /z/collbldg/html/parts/00000003.did
```

This partition is for a collection containing the *Verity Collection Building Guide* in HTML format.

```
didump - Verity, Inc. Version 2.5.0 (_solaris, Jul 9 1999)

Text                        Size   Doc   Word
href 01_cbg.htm             10     2     4
href 01_cbg.htm#282870      3      1     1
href 01_cbg.htm#282872      6      2     2
href 01_cbg1.htm            8      2     3
href 01_cbg1.htm#286513     7      2     2
href 01_cbg1.htm#286520     3      1     1
...
```

The columns in the display indicate the following:

- **Size** The number of bytes used by the Verity engine to store information about the zone attribute
- **Doc** The number of unique documents in which the zone attribute appears
- **Word** The total number of occurrences of a zone attribute for the partition

Using the Verity browse utility

A documents table is built for each partition in a collection. The documents table is used for field searching and for sorting search results. The fields within the documents table are defined by the following collection style files:

- **style.ddd** Defines fields used internally by the Verity engine, identified by an initial underscore character (_).
- **style.sfl** Defines standard fields (many of which are commented out to limit the size of the documents table).
- **style.ufl** Defines custom fields that are not included in the style.sfl file.

The value of each field can be filled in from source documents or can be provided explicitly. If a field is blank, it has not been populated.

The browse utility executable, which starts the browse utility application, is located in the *cf_root*\lib_nti40\bin directory in Windows, and in the *cf_root*/lib/*platform*/bin directory in UNIX.

In these pathnames, *cf_root* refers to the ColdFusion root directory. In Windows, this is typically C:\CFusionMX; in UNIX, this is typically /opt/coldfusionmx. In UNIX, *platform* refers to the UNIX version of the server that runs ColdFusion: _solaris, _hpux11, or _ilnx21.

For example:

```
c:\cfusionmx\lib\_nti40\bin\browse /common = c:\cfusionmx\lib\common
         c:\my_collection\parts\0000001.ddd
```

Using menu options with the browse utility

Use the following browse command to start the utility and display a set of menu options:

```
browse 00000003.ddd
```

The system displays the following menu of options available for the browse utility:

```
D:\VERITY\colltest\parts>browse 00000003.ddd
BROWSE OPTIONS
?) help
q) quit
c) Number of entries in field
_) Toggle viewing fields beginning with '_'
v) Toggle viewing selected fields
##) Display all fields in specified record number
Dispatch/Compound field options:
n) No dispatch
d) Dispatch
s) Dispatch as stream
Action (? for help):
```

Displaying fields

You can use several options to control the display of field information.

To display all the document fields:

1 At the Action prompt, enter ##

2 Press Return twice to display the fields for the first document record.

3 Press Return to view the document fields for the next sequential record.

The following partial display of the results of the browse command includes internal fields, used by the Verity search engine. An internal field name starts with an underscore character (_).

```
50 Created              FIX-date ( 4) = 12-Jan-1998 01:52:27 pm
51 Modified             FIX-date ( 4) = 24-Sep-1997 02:40:26 pm
52 Size                 FIX-unsg ( 4) = 5381
53 DOC_OF               FIX-unsg ( 4) = 0
54 DOC_SZ               FIX-unsg ( 4) = 4294967295
55 DOC_FN_OF            FIX-unsg ( 4) = 436
56 DOC_FN_SZ            FIX-unsg ( 2) = 58
57 _CACHE_FN_OF         FIX-unsg ( 4) = 2922
58 _CACHE_FN_SZ         FIX-unsg ( 2) = 0
59 _ParentID_OF         FIX-unsg ( 4) = 354
60 _ParentID_SZ         FIX-unsg ( 2) = 46
61 Title_OF             FIX-unsg ( 4) = 2481
62 Title_SZ             FIX-unsg ( 2) = 15
```

You can eliminate the internal fields. To do this, type the underscore character, then press Return. If you enter an underscore character again, then press return, the internal fields are displayed.

Using the Verity merge utility

The `merge` utility lets you combine multiple collections with identical schemas. This is useful for merging smaller collections built from different sources into one, large collection. Also, you can use the `merge` utility to break up the collection into smaller collections of a roughly uniform size.

Note: The Verity `merge` utility is available only on Windows.

Collections can be merged only if they have identical schemas. Collections can be merged if they have exactly the same set of style files (and style file entries).

Breaking up a large collection helps to optimize search performance, because it allows many applications to perform multiple concurrent search requests over the different collections. After breaking up a large collection, you can also discard older collections to reclaim limited disk storage space.

The merge executable, which starts the merge application, is located in the *cf_root*\lib_nti40\bin directory.

In the above location, *cf_root* refers to the ColdFusion root directory.

For example:

```
c:\cfusionmx\lib\_nti40\bin\merge /common = c:\cfusionmx\lib\common
```

To obtain help for the merge utility, enter the following command:

```
merge -help
```

Note: After running the merge utility, you must optimize the collection, using the mkvdk -optimize option.

Merging collections using the merge utility

The following is the syntax for using the merge utility to merge multiple collections into a single collection:

```
merge <newCollection> <srcCollection1> <srcCollection2> [srcCollectionN]
```

The utility reads srcCollection1, srcCollection2 and so on and merges them into a single collection with the directory name given for newCollection If the directory name given for newCollection does not exist, it is created.

Splitting collections using the merge utility

The following is the syntax for using the merge utility to split a single large collection into smaller collections:

```
merge -split <srcCollection> <newCollection1> <newCollection2> [-number]
```

The merge utility reads srcCollection and splits it into roughly equal pieces, using the filenames given for newCollection1 and so on.

If you want to split a very large collection into a large number of new collections, you can use the following command, instead of explicitly naming each new collection:

```
merge -split -number newCollection srcCollection
```

The merge utility reads the collection identified by srcCollection and splits it into the number of segments specified by the -number option. The name of the first new collection is generated by appending the first two letters in the alphabet (aa) to the directory name given for newCollection. Each subsequent filename is generated by incrementing one of the appended letters (up to zz) for a maximum of 676 partitions. For example, if the value of -number is 3, and the value of newCollection is Collection1, the collections are named, Collection1aa, Collection1ab, and Collection1ac.

Note: The maximum length of the directory name given for `newCollection` is two characters less than the length allowed by the file system.

CHAPTER 7
Verity Error Messages

This chapter provides information about error messages that might occur when using Verity in either VDK mode or K2 mode.

Contents

VDK mode error codes

All Verity Developer's Kit (VDK) API functions return an error code, and VdkSuccess is the successful return value. The following sections list the API error codes. These reflect actions of the `cfcollection`, `cfindex`, or `cfsearch` tags.

Generic error codes

Error code	No.	Description
VdkSuccess	(0)	Operation completed successfully.
VdkFail	(-2)	A general failure not covered by another API error code.
VdkWarn	(1)	A general warning.

Usage error codes

Error code	No.	Description
VdkError_BadArgStruct	(-10)	Invalid argument structure.
VdkError_BadHandleType	(-11)	Improper object type.
VdkError_HandleNotFound	(-12)	Object not found.
VdkError_MissingArgs	(-13)	Missing required arguments.
VdkError_InvalidArgs	(-14)	Invalid arguments.
VdkError_MultipleSesNew	(-16)	VdkSessionNew called twice.
VdkError_NestedService	(-17)	VdkService called reentrantly.
VdkError_NestedFree	(-18)	VdkSessionFree called reentrantly.
VdkError_Unsupported	(-19)	Using an unsupported feature.

Runtime error codes

Error code	No.	Description
VdkError_NoMsgDb	(-20)	Cannot find the message database.
VdkError_FatalError	(-21)	Fatal error.
VdkError_OutOfMemory	(-22)	Out of memory.
VdkError_DiskFull	(-23)	Out of disk space.
VdkError_NoFileHandles	(-24)	Out of file handles.
VdkError_InvalidDoc	(-25)	Bad document ID or key (internal or external).
VdkError_FileNotFound	(-26)	File not found.
VdkError_ArgTooLarge	(-27)	Argument too large.
VdkError_InvalidSortSpec	(-28)	Invalid sort specification.

Error code	No.	Description
VdkError_GatewayNotAvail	(-29)	Gateway driver not available.
VdkError_VersionMismatch	(-30)	Argument or object mismatch.
VdkError_NoInstallDir	(-100)	Cannot find installation directory.

Data error codes

Error code	No.	Description
VdkError_StyleFiles	(-31)	Invalid style files.
VdkError_Permissions	(-32)	Bad file or directory permission.
VdkError_CollNotAvail	(-33)	The collection is not available because it is down or under repair. This error occurs only when the Verity engine is attempting a submit action (for example, insert, update, or delete), to a collection. If this error is returned, the submit action does not occur.
VdkError_CollIII	(-34)	The collection is very sick.
VdkError_CollRepair	(-36)	The collection has been repaired.
VdkError_CollReadOnly	(-37)	This collection is read-only. No submits are allowed.
VdkError_CollPurge	(-38)	Purge failed due to problems deleting from any of the following directories: pdd, work, trans
VdkError_CollPathTooBig	(-39)	Collection path supplied for the path member in VdkCollectionOpenArgRec is too long. For more information, refer to the description of the VdkPath_MaxSize macro in your Verity documentation.
VdkError_V3Legacy	(-35)	Unsupported legacy collection(s).
VdkError_LocaleIncompat	(-101)	Collection and session locales are incompatible.
VdkError_KBNotOpened	(-102)	Knowledgebase is incompatible and cannot be opened.

Query error codes

Error code	No.	Description
VdkError_QueryParse	(-40)	Query has a parsing error.

Licensing error codes

Error code	No.	Description
VdkError_Signature	(-50)	Invalid/missing signature.
VdkError_LicenseFile	(-51)	Invalid license file.

Error code	No.	Description
VdkError_LicenseColl	(-52)	Too many collections open.
VdkError_LicenseVolume	(-53)	Too many documents in collection.
VdkError_LicenseAdvQuery	(-54)	No advanced query capability.
VdkError_LicenseHetero	(-56)	No heterogeneous collections.
VdkError_LicenseDataPrep	(-57)	Not licensed to index documents.
VdkError_LicenseStreams	(-58)	Not licensed for streams.
VdkError_LicenseTopics	(-59)	Not licensed for topics.
VdkError_LicenseThes	(-60)	Not licensed for thesaurus.
VdkError_LicenseAdvFeat	(-64)	Not licensed for advanced features.
VdkError_LicenseSesSpawn	(-65)	No spawning sessions.
VdkError_LicenseWatchers	(-66)	No watchers.
VdkError_LicenseAcrocoll	(-67)	No access to Acrobat.
VdkError_LicenseProfile	(-68)	No profilers.
VdkError_LicenseProfileLatency	(-69)	Low-speed profiler.
VdkError_LicensePrfCount	(-110)	Too many profiles.
VdkError_LicenseClustering	(-111)	No clustering.
VdkError_LicenseSummarization	(-112)	No summarization.
VdkError_LicenseNLQP	(-113)	No natural language queries.
VdkError_LicenseQBE	(-114)	No query-by-example.
VdkError_LicenseAdvSGML	(-115)	No support for advanced SGML search.
VdkError_LicenseZone	(-116)	No support for zone search.
VdkError_LicenseField	(-117)	No support for field search.
VdkError_LicenseAccrue	(-118)	No support for the ACCRUE operator.
VdkError_LicenseProximity	(-119)	No support for the proximity operators.
VdkError_LicenseStem	(-120)	No stemming.
VdkError_LicenseWildcard	(-121)	No support for wildcard queries.
VdkError_LicenseTypo	(-122)	No support for typo assist.
VdkError_LicenseOperator	(-123)	Unlicensed operator.
VdkError_LicenseInso	(-124)	Not licensed for INSO software.
VdkError_LicenseInvalid	(-125)	Invalid license.
VdkError_LicenseVgw	(-126)	No collection gateways.
VdkError_LicenseSoundex	(-127)	No support for Soundex queries.
VdkError_LicenseSentpara	(-128)	No support for SENTENCE or PARAGRAPH operators.

Error code	No.	Description
VdkError_Scoreop	(-129)	No support for Score operators.
VdkError_Opmod	(-130)	No support for query language modifiers.
VdkError_LicenseSession	(-131)	Too many top-level sessions.

Security error codes

Error code	No.	Description
VdkError_InvalidUser	(-80)	Invalid user/password combination.

Remote connection error codes

Error code	No.	Description
VdkError_HostNotAvail	(-90)	Cannot contact remote host.
VdkError_NotReEntrant	(-91)	Not reentrant.
VdkError_CallDenied	(-92)	Call cannot be executed.

Filtering error codes

Error code	No.	Description
VdkError_BadFile	(-140)	Corrupt or unreadable file.
VdkError_EmptyFile	(-141)	Empty file.
VdkError_ProtectedFile	(-142)	Password protected or encrypted file.
VdkError_FilterNotAvail	(-143)	No appropriate filter for a file format.
VdkError_FilterLoadFailed	(-144)	Error occurred during filter initialization.
VdkError_FileOpenFailed	(-145)	File could not be opened.

Dispatch error codes

Error code	No.	Description
VdkError_CouldntLoadDLL	(-200)	Cannot load DLL.
VdkError_NoSuchFunction	(-201)	Function not available.

Warning error codes

Error code	No.	Description
VdkWarning_CollectionDown	(10)	The collection was down when it was opened.
VdkWarning_QueryComplex	(11)	Too many matching words.
VdkWarning_LowMemory	(12)	Memory is low for indexing.

Error code	No.	Description
VdkWarning_CollectionReadOnly	(13)	The collection is read-only.
VdkWarning_DriverNotFound	(14)	Couldn't locate specified driver.
VdkWarning_LargeToken	(15)	Returned a token greater than maxSize.
VdkWarning_ArgTooLarge	(16)	Argument too large.
VdkWarning_DataSrcNotAvail	(17)	Cannot locate collection data.
VdkWarning_SearchRestricted	(18)	Search restricted to a subset of the collection.

K2 mode error codes

All K2 Client API functions return an error code, and K2Success is the successful return value. The following sections list the API error codes. These reflect actions of the `cfsearch` tag.

Generic error codes

Error code	No.	Description
K2Success	(0)	Operation completed successfully.
K2Fail	(-2)	A general failure not covered by another API error code.
K2Warn	(1)	A general warning.

Usage error codes

Error code	No.	Description
K2Error_NoConnectAvail	(-9)	A K2 connection is not available.
K2Error_BadArgStruct	(-10)	Invalid argument structure.
K2Error_BadHandleType	(-11)	Improper object type.
K2Error_HandleNotFound	(-12)	Object not found.
K2Error_MissingArgs	(-13)	Missing required arguments.
K2Error_InvalidArgs	(-14)	Invalid arguments.
K2Error_Unsupported	(-19)	Using an unsupported feature.

Runtime error codes

Error code	No.	Description
K2Error_NoMsgDb	(-20)	Cannot find the message database.
K2Error_FatalError	(-21)	Fatal error.
K2Error_OutOfMemory	(-22)	Out of memory.
K2Error_DiskFull	(-23)	Out of disk space.
K2Error_NoFileHandles	(-24)	Out of file handles.
K2Error_InvalidDoc	(-25)	Bad document ID or key (internal or external).
K2Error_FileNotFound	(-26)	File not found.
K2Error_ArgTooLarge	(-27)	Argument too large.
K2Error_InvalidSortSpec	(-28)	Invalid sort specification.
K2Error_GatewayNotAvail	(-29)	Gateway driver not available.

Error code	No.	Description
K2Error_VersionMismatch	(-30)	arg or Vdk Object mismatch.
K2Error_NoInstallDir	(-100)	Cannot find installation directory.

Data error codes

Error code	No.	Description
K2Error_StyleFiles	(-31)	Invalid style files.
K2Error_Permissions	(-32)	Bad file or directory permission.
K2Error_CollNotAvail	(-33)	The collection is not available because it is down or under repair. This error occurs only when the Verity search engine is attempting a submit action (for example, insert, update, or delete), to a collection. If this error is returned, the submit action does not occur.
K2Error_CollIII	(-34)	The collection is corrupt and needs repair.
K2Error_v3Legacy	(-35)	Unsupported on Legacy V3 database.
K2Error_CollRepair	(-36)	The collection has been repaired.
K2Error_CollReadOnly	(-37)	This collection is read-only. No submits are allowed.
K2Error_CollPurge	(-38)	Purge failed due to problems deleting from any of the following directories: pdd, work, trans
K2Error_CollPathTooBig	(-39)	Collection path supplied for the path member in K2CollectionOpenArgRec is too long.
K2Error_LocaleIncompat	(-101)	Collection and session locales are incompatible.
K2Error_KBNotOpened	(-102)	Knowledgebase cannot be opened.

Query error codes

Error code	No.	Description
K2Error_QueryParse	(-40)	Query has a parsing error.

Security error codes

ErrorCode	No.	Description
K2Error_InvalidUse	(-80)	Invalid user/password combination.

Remote connection error codes

Error code	No.	Description
K2Error_HostNotAvail	(-90)	Cannot contact remote host.
K2Error_NotReEntrant	(-91)	Not reentrant.
K2Error_CallDenied	(-92)	Call cannot be executed.

File handling error codes

Error code	No.	Description
K2Error_BadFile	(-140)	Corrupt or unreadable file.
K2Error_EmptyFile	(-141)	Empty file.
K2Error_ProtectedFile	(-142)	Password protected or encrypted.
K2Error_FilterNotAvail	(-143)	No appropriate filter.
K2Error_FilterLoadFailed	(-144)	Error during filter initialization.
K2Error_FileOpenFailed	(-145)	File could not be opened.

Dispatch error codes

Error code	No.	Description
K2Error_CouldntLoadDLL	(-200)	Cannot load DLL.
K2Error_NoSuchFunction	(-201)	Function not available.

Warning error codes

Error code	No.	Description
K2Warning_CollectionDown	(10)	The collection was down when it was opened.
K2Warning_QueryComplex	(11)	Too many matching words.
K2Warning_LowMemory	(12)	Memory is low for indexing.
K2Warning_CollectionReadOnly	(13)	The collection is read-only.
K2Warning_DriverNotFound	(14)	Couldn't locate specified driver.
K2Warning_LargeToken	(15)	Returned a token greater than maxSize.
K2Warning_ArgTooLarge	(16)	Argument too large.
K2Warning_DataSrcNotAvail	(17)	Cannot locate collection data.
K2Warning_SearchRestricted	(18)	Searching subset of collection.

TCP/IP error codes

Error code	No.	Description
K2TcpError_Memory	c100	Out of memory.
K2TcpError_ConnDrop	c200	Connection closed by remote host.
K2TcpError_WillBlock	c300	Will block on this call.
K2TcpError_Call_DNS	c600	DNS lookup failed (use IP address).
K2TcpError_Call_Send	c700	Send failed (maybe connection damaged).
K2TcpError_Call_Recv	c800	Recv failed (maybe connection damaged).
K2TcpError_Call_Ioctl	c900	Ioctl failed (Internal error).
K2TcpError_Call_Socket	ca00	Socket failed (maybe out of file handles).
K2TcpError_Call_Bind	cb00	Bind failed (local address already in use).
K2TcpError_Call_Listen	cc00	Listen failed (maybe out of resources).
K2TcpError_Call_Accept	cd00	Accept failed (maybe out of resources).
K2TcpError_Call_Select	ce00	Select failed (maybe connection damaged).
K2TcpError_Call_Connect	cf00	Connect failed (connection not accepted).

INDEX

S

V