CHAPTER 24
Building a Search Interface

You can provide a full-text search capability for documents and data sources on a ColdFusion site by enabling the Verity search engine.

This chapter describes how to build a Verity search interface with which users can perform powerful searches on your application. It also describes how to index your documents and data sources so that users can search them.

Contents

About Verity

To efficiently search through paragraphs of text or files of varying types, you need full-text search capabilities. ColdFusion includes the Verity search engine, which provides full-text indexing and searching.

The Verity engine performs searches against collections, not against the actual documents. A **collection** is a special database created by Verity that contains metadata that describes the documents that you have indexed. The **indexing** process examines documents of various types in a collection and creates a metadata description—the **index**—which is specialized for rapid search and retrieval operations.

The ColdFusion implementation of Verity supports collections of the following basic data types:

- Text files such as HTML pages and CFML pages
- Binary documents (see "Supported file types" on page 523)
- Record sets returned from `cfquery`, `cfldap`, and `cfpop` queries

You can build collections from individual documents or from an entire directory tree. Collections can be stored anywhere, so you have much flexibility in accessing indexed data.

In your ColdFusion application, you can search multiple collections, each of which can focus on a specific group of documents or queries, according to subject, document type, location, or any other logical grouping. Because you can perform searches against multiple collections, you have substantial flexibility in designing your search interface.

Using Verity with ColdFusion

Here are some ways to use Verity with ColdFusion:

- Index your website and provide a generalized search mechanism, such as a form interface, for executing searches.
- Index specific directories containing documents for subject-based searching.
- Index `cfquery` record sets, giving users the ability to search against the data. Because collections contain data optimized for retrieval, this method is much faster than performing multiple database queries to return the same data.
- Index `cfldap` and `cfpop` query results.
- Manage and search collections generated outside of ColdFusion using native Verity tools. This additional capability requires only that the full path to the collection be specified in the index and search commands.
- Index e-mail generated by ColdFusion application pages and create a searching mechanism for the indexed messages.
- Build collections of inventory data and make those collections available for searching from your ColdFusion application pages.
- Support international users in a range of languages using the `cfindex`, `cfcollection`, and `cfsearch` tags.

Advantages of using Verity

Verity can index the output from queries so that you or a user can search against the record sets. Searching query results has a clear advantage over using SQL to search a database directly in speed of execution because metadata from the record sets are stored in a Verity index that is optimized for searching.

Performing a Verity search has the following advantages over other search methods:

- You can reduce the programming overhead of query constructs by allowing users to construct their own queries and execute them directly. You need only be concerned with presenting the output to the client web browser.
- Verity can index database text fields, such as notes and product descriptions, that cannot be effectively indexed by native database tools.
- When indexing collections containing documents in formats such as Adobe Acrobat (PDF) and Microsoft Word, Verity scans for the document title (if one was entered), in addition to the document text, and displays the title in the search results list.
- When Verity indexes web pages, it can return the URL for each document. This is a valuable document management feature.

Supported file types

The ColdFusion Verity implementation supports a wide array of file and document types. As a result, you can index web pages, ColdFusion applications, and many binary document types and produce search results that include summaries of these documents.

To support multiple WYSIWYG document types, Verity bundles the KeyView Filter Kit. The KeyView Filter Kit includes document filters that support the indexing and viewing of more than 45 native document formats. Numerous popular document suites and formats are supported, including Microsoft Office 95, 97, and 2000, Corel WordPerfect, Microsoft Word, Microsoft Excel, Lotus AMI Pro, and Lotus 1-2-3.

The Verity KeyView filters support the following formats:

Word processing/text formats

- Applix Words (v4.2, 4.3, 4.4, 4.41)
- ASCII Text (All versions)
- ANSI Text (All versions)
- Folio Flat File (v3.1)
- HTML (Verity Zone Filter)
- Lotus AmiPro (v2.3)
- Lotus Ami Professional Write Plus (All versions)
- Lotus Word Pro (v96, 97, R9)
- Maker Interchange Format (MIF) v5.5
- Microsoft RTF (All versions)
- Microsoft Word (v2, 6, 95, 97, 2000)
- Microsoft Word Mac (v4, 5, 6, 98)
- Microsoft Word PC (v4.,5, 6)
- Microsoft Works (v1.0, 2.0, 3.0, 4.0)

- Microsoft Write (v1.0, 2.0, 3.0)
- PDF (Verity PDF Filter)
- Text files (Verity Text Filter)
- Unicode Text (All versions)
- WordPerfect (v5.x, 6, 7, 8)
- WordPerfect Mac (v2, 3)
- XyWrite (v4.12)

Spreadsheet formats

- Applix Spreadsheets (v4.3, 4.4)
- Corel QuattroPro (v7, 8)
- Lotus 1-2-3 (v2, 3, 4, 5, 96, 97, R9)
- Microsoft Excel (v3, 4, 5, 96, 97, 2000)
- Microsoft Excel Mac (98)
- Microsoft Works spreadsheet (v1.0, 2.0, 3.0, 4.0)

Presentation formats

- Applix Presents (v4.3, 4.4)
- Corel Presentations (v7.0, 8.0)
- Lotus Freelance (v96, 97, R9)
- Microsoft PowerPoint (v4.0, 95, 97, 2000)
- Microsoft PowerPoint Mac (98)

Picture formats

- AMI Draw Graphics (SDW)
- Applix Graphics v4.3, 4.4
- Fax Systems (TIFF CCITT) Groups 3 & 4
- Computer Graphics Metafile (CGM)
- Corel Draw CDR (TIFF Header)
- DCX Fax
- Encapsulated PostScript (EPS)
- Enhanced Metafile (EMF)
- JPEG File Interchange Format
- Lotus Pic (PIC)
- Mac PICT (raster content)
- MacPaint (MAC)
- Microsoft Excel Charts
- Microsoft Windows Animated Cursor
- Microsoft Windows Bitmap (BMP)
- Microsoft Windows Cursor/Icon
- Microsoft Windows Metafile (WMF)
- PC PaintBrush (PCX)
- Portable Network Graphics (PNG)

- Sun Raster SGI RGB
- Truevision Targa
- TIFF
- WordPerfect Graphics (WPG) v1, 2

Multimedia formats

- Audio Interchange File Format (AIFF)
- Microsoft Sound (WAV)
- MIDI (MID)
- MPEG 1 Video (MPG)
- MPEG 2 Audio
- NeXT/Sun Audio (AU)
- QuickTime Movie v2.0
- Video for Windows v2.1

Support for international languages

ColdFusion supports Verity Locales in European and Asian languages. For European languages, ColdFusion uses LinguistX™ technology from Inxight; for Asian languages, ColdFusion uses ICU (IBM® Classes for Unicode) technology. For more information about installing Verity Locales, see *Installing ColdFusion MX*.

The default language for Verity collections is English. To index data in another supported language, select it from the drop-down list when you create a collection with the ColdFusion Administrator. In CFML, the `cfcollection`, `cfindex`, and `cfsearch` tags have an optional `language` attribute that you use to specify the language of the collection that you are searching. If you do not specify a language in these tags, ColdFusion checks the neo-verity.xml file for the collection's language. If this is defined, ColdFusion uses that language.

Use the following table to find the correct value for the `language` attribute for your collection; for example, the following code creates a collection for simplified Chinese:

```
<cfcollection action = "create" collection = "lei_01"
  path = "c:\cfusionmx\verity\collections"
  language = "simplified_chinese">
```

The following table lists the languages names and attributes that ColdFusion supports:

Language	Language attribute	Localization technology
Arabic	arabic	ICU
Chinese (simplified)	simplified_chinese	ICU
Chinese (traditional)	traditional_chinese	ICU
Czech	czech	ICU
Danish	danish	LinguistX
Dutch	dutch	LinguistX
English	english	LinguistX
Finnish	finnish	LinguistX
French	french	LinguistX
German	german	LinguistX
Greek	greek	ICU
Hebrew	hebrew	ICU
Hungarian	hungarian	ICU
Italian	italian	LinguistX
Japanese	japanese	ICU
Korean	korean	ICU
Norwegian	norwegian	LinguistX
Norwegian (Bokmal)	bokmal	LinguistX
Norwegian (Nynorsk)	nynorsk	LinguistX

Language	Language attribute	Localization technology
Polish	polish	ICU
Portuguese	portuguese	LinguistX
Russian	russian	ICU
Spanish	spanish	LinguistX
Swedish	swedish	LinguistX
Turkish	turkish	ICU

You can register collections in the Administrator or by creating a collection with the `cfcollection` tag. If you register a given collection with ColdFusion and you specify a `language` attribute, then you do not have to specify the `language` attribute when using `cfindex` and `cfsearch` for that collection. If you do not register a given collection with ColdFusion, the language defaults to English, unless you specify it in the `language` attribute for the `cfindex` and `cfsearch` tags for that collection.

Creating a search tool for ColdFusion applications

There are three main tasks in creating a search tool for your ColdFusion application:

1 Create a collection.

2 Index the collection.

3 Design a search interface.

You can perform each task programmatically—that is, by writing CFML code. Alternatively, you can use the ColdFusion Administrator to create and index the collection. Also, ColdFusion Studio has a Verity Wizard that generates ColdFusion pages that index the collection and design a search interface. The following table summarizes the steps and available methods for creating the search tool:

Step	CFML	ColdFusion MX Administrator	Verity Wizard
Creating a collection	Yes	Yes	No
Indexing a collection	Yes	Yes	Yes
Designing a search interface	Yes	No	Yes

This chapter presents the non-code methods for developing a search tool, followed by code examples that perform the same task. If you have ColdFusion Studio and access to the ColdFusion Administrator, the fastest development method is as follows:

1 Create the collection with the ColdFusion Administrator.

2 Use the Verity Wizard to index the collection and design a search interface.

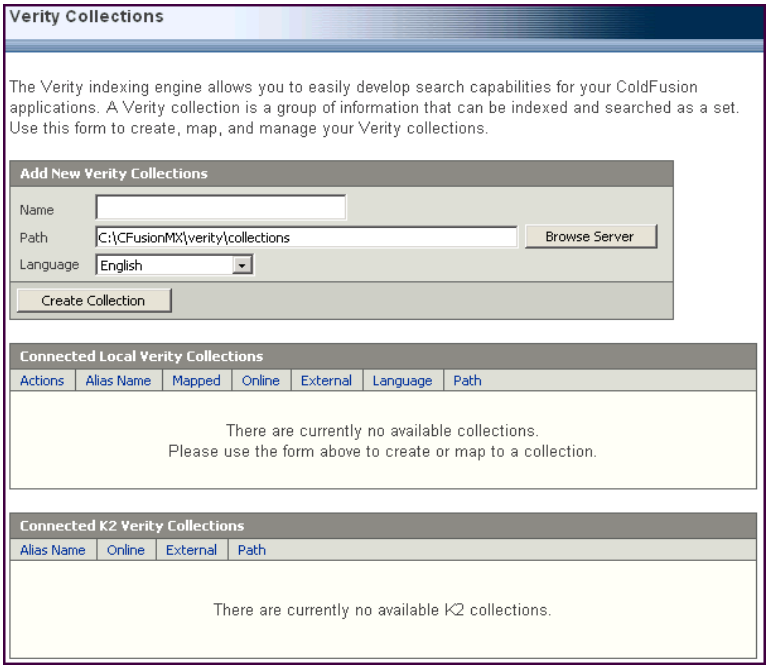
Creating a collection with the ColdFusion MX Administrator

Use the following procedure to quickly create a collection with the ColdFusion Administrator:

To create a collection with the ColdFusion MX Administrator:

1 In the ColdFusion MX Administrator, select **Data & Services > Verity Collections**.

The Verity Collections page appears:



2 Enter a name for the collection; for example, DemoDocs.

3 Enter a path for the directory location of the new collection; for example, C:\cfusionmx\verity\collections\.

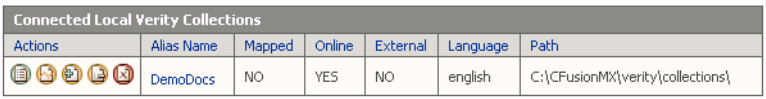
By default, ColdFusion stores collections in *cf_root*\verity\collections\ in Windows and in /*cf_root*/verity/collections in UNIX.

Note: This is the location for the collection, not for the files that you will search.

4 (Optional) Select a language other than English for the collection from the Language drop-down list.

5 Click Create Collection.

The name and full path of the new collection appears in the list of Connected Verity Collections:



Note: You can map a collection currently available on your network or local disk by creating a local reference (an alias) for that collection. In this procedure, enter the collection alias as the collection name, and enter a UNC (Universal Naming Convention) path to the folder for the collection.

You have successfully created a collection, DemoDocs, that currently has no data. A collection becomes populated with data when you index it. For more information, see the next section, "About indexing a collection" on page 530.

About indexing a collection

A new collection is an empty shell that must be indexed before you search it. The indexing procedure also populates the collection with data contained in the collection's files. Similar to creating a collection, you can index a collection either in the ColdFusion Administrator or programmatically.

Note: You can index and search against collections created outside of ColdFusion by using the `external` attribute of `cfindex` and `cfsearch`.

Use the following guidelines to determine which method to use:

Use the Administrator	Use the cfindex tag
To index document files	To index ColdFusion query results
When the collection does not require frequent updates	When the collection requires frequent updates
To create the collection without writing any CFML code	To dynamically update a collection from a ColdFusion application page
To create a collection once	When the collection requires updating by others

The `cfcollection` tag has the following `action` attribute values that can fix or improve your index:

- **repair** Repairs the internal index files of a collection. This might take a few minutes for large collections.
- **optimize** Optimizes a collection. Use this if you notice that your searches on a collection take longer than previously.

Updating an index

Documents are modified frequently in many user environments. After you index your documents, any changes that you make are not reflected in subsequent Verity searches until you reindex the collection. Depending on your environment, you can create a scheduled task to automatically keep your indexes current. For more information on scheduled tasks, see *Administering ColdFusion MX*.

Indexing and building a search interface with the Verity Wizard

If you have ColdFusion Studio, you can use the Verity Wizard to generate a basic search and index interface. Use the following procedure to quickly create a search application for a collection. This procedure assumes the following:

- There is an empty Verity collection to hold the indexed data. For details on how to use the ColdFusion Administrator to create a collection, see "Creating a collection with the ColdFusion MX Administrator" on page 528.

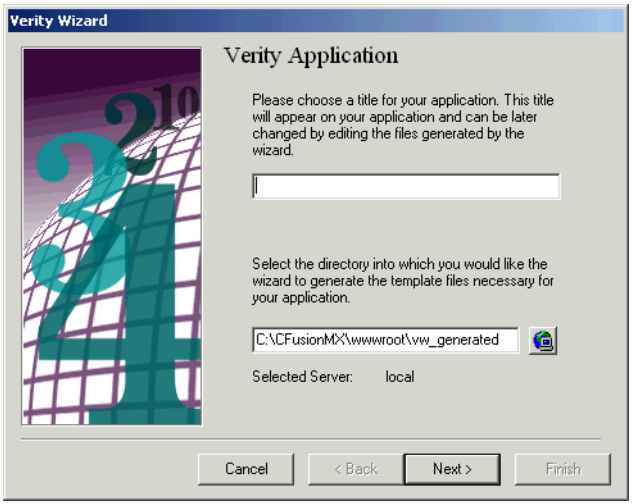
- A directory contains files of several types, such as text, word processing, spreadsheet, and HTML. If this directory is within your *web_root*, then you can view the files from the web browser.
- Some of these files contain a search target word(s).
- There is an available directory to hold the four ColdFusion pages that the wizard generates.

To build a search interface using the Verity Wizard:

1 In ColdFusion Studio, select **File > New**.

2 In the New Document window, click the CFML tab.

3 Double-click the Verity Wizard.
The Verity Application window appears:

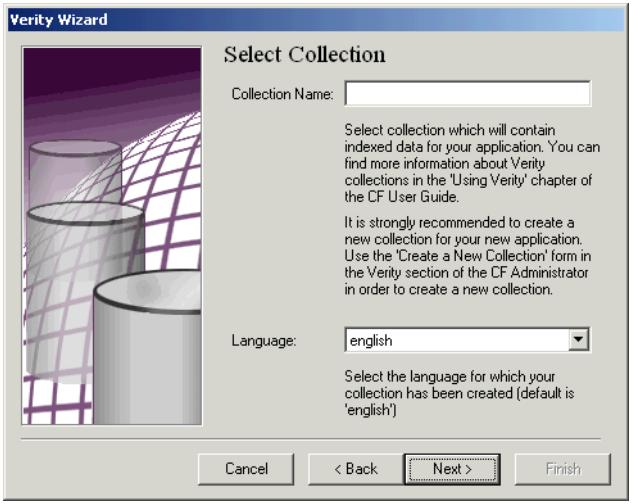


4 Enter the following information:

Field	Description	Example
Title	Appears at the top of each generated ColdFusion page.	Search CF Documentation
Directory	Contains the generated ColdFusion pages. The directory should be under your web_root so that you can view ColdFusion pages in the web browser.	web_root\vw_generated

5 Click Next.

The Select Collection window appears:

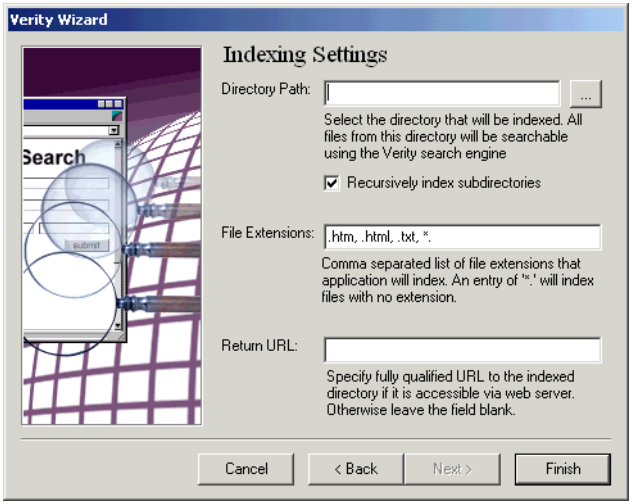


6 Enter the following information:

Field	Description	Example
Collection Name	The name of the collection you created in the ColdFusion Administrator (or by using the `cfcollection` tag).	DemoDocs
Language	The language used to create the collection (English is the default).	english

7 Click Next.

The Indexing Settings window appears:

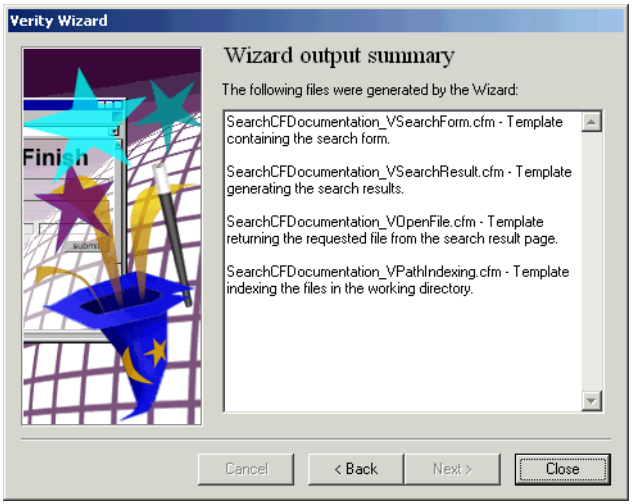


8 Enter the following information:

Field	Description	Example
Directory Path	The directory that contains the documents to be indexed.	C:\CFusionMX\wwwroot\cfdocs
Recursively Index Subdirectories	(Optional) Extends the indexing operation to all directories below the selected path.	enabled (default)
File Extensions	The type(s) of files to index. Use a comma to separate multiple file types.	.htm, .html, .xml
Return URL	(Optional) If your documents are beneath the web_root, enter a URL that corresponds to the Directory Path.	http://127.0.0.1:8500/cfdocs/

9 Click Finish.

The wizard generates four ColdFusion pages to the directory you specified in step 4, and displays an output summary:



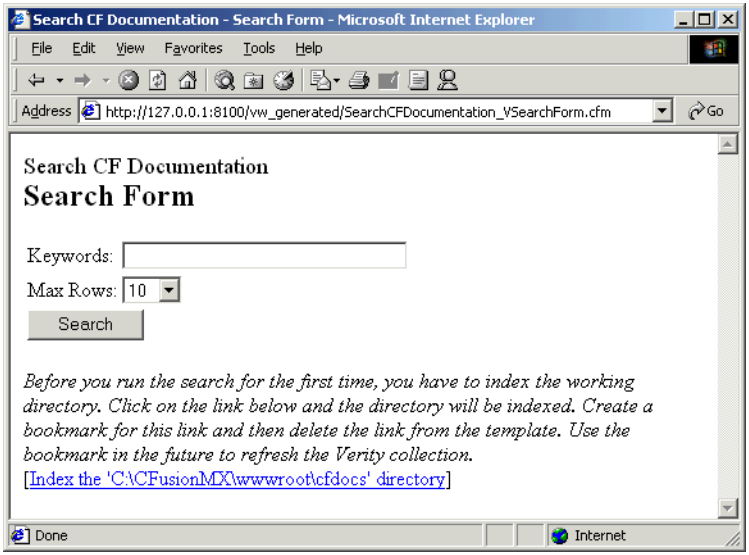
Note: The file names are in the format pagetitle_Vpagename.cfm, where pagetitle is the value you specified in step 4 and pagename is SearchForm, SearchResult, OpenFile, or PathIndexing.

10 Click Close.

The wizard closes and the files open in ColdFusion Studio (you can adjust its size to display all file tabs).

11 Browse the SearchForm page in ColdFusion Studio.

Alternatively, you can use the web browser; if you do so, enter an HTTP URL that corresponds to your SearchForm, such as:
http://127.0.0.1:8500/vw_generated/SearchCFDocumentation_VSearchForm.cfm:



12 Click the Index link at the bottom of the page.

A confirmation message appears when indexing successfully completes.

13 Click the web browser's back button to return to the search form.

14 Enter your search term(s); for example, Verity AND data source.

Tip: For more information on the Verity search syntax, see "Using Verity Search Expressions" on page 553.

15 Click Search.

In ColdFusion Studio 4.*x*, the following compilation error might display:

```
Invalid parser construct found on line 46 at position 49. ColdFusion was
looking at the following text:'
```

To correct this error, do the following:

a In ColdFusion Studio, open the SearchResult page in Edit mode; for example, WizardDocDemo_VSearchResult.cfm.

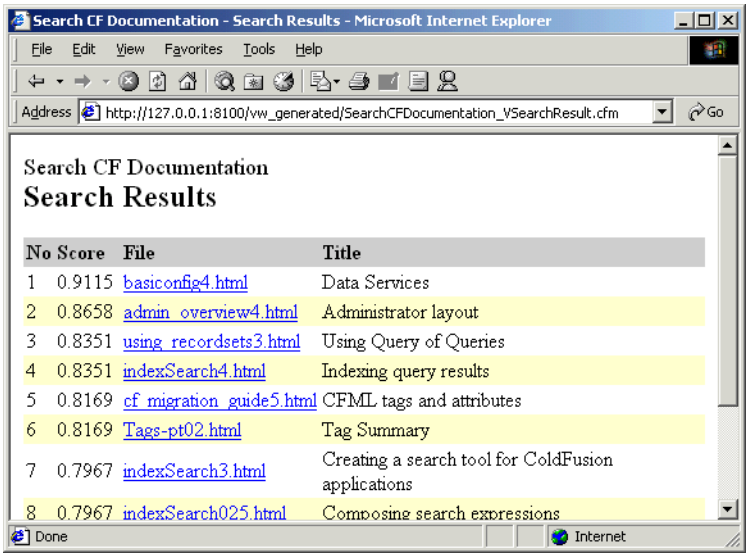
b In line 46, delete the pound signs that precede the hexadecimal color codes. The correct code is:

```
<TR bgcolor="#IIf(CurrentRow Mod 2, DE('FFFFFF'), DE('FFFFCF'))#">
```

c Save the file.

d Browse the SearchForm page and enter the search target.

Your search results appear:



If you entered a Return URL value and your documents are beneath your web_root (as in this procedure), you can click the link to open them.

You now have Verity search capability for your ColdFusion application. You can edit the generated ColdFusion pages or copy the generated code into the current pages to better integrate with your application.

You can create a search interface without using the Verity Wizard. The remainder of this chapter describes how to write CFML code that is functionally identical to the pages generated by the wizard. You can write the code using your text editor and preview it in the web browser.

Creating a ColdFusion search tool programmatically

You can create a Verity search tool for your ColdFusion application in CFML. Although writing CFML code can take more development time than using these tools, there are situations in which writing code is the preferred development method.

Creating a collection with the cfcollection tag

The following are cases in which you might prefer using the `cfcollection` tag rather than the ColdFusion MX Administrator to create a collection:

- You want your ColdFusion application to be able to create, delete, and maintain a collection.
- You do not want to expose the ColdFusion MX Administrator to users.
- You want to create indexes on servers that you cannot access directly; for example, if you use a hosting company.

When using the `cfcollection` tag, you can specify the same attributes as in the ColdFusion MX Administrator:

- **action** (Optional) The action to perform on the collection (create, delete, repair, or optimize). The default value for the `action` attribute is `list`. For more information, see *CFML Reference*.
- **collection** The name of the new collection, or the name of a collection upon which you will perform an action.
- **path** The location for the Verity collection.
- **language** (Optional) The language used to create the collection (English, by default).

You can create a collection by directly assigning a value to the `name` attribute of the `cfcollection` tag, as shown in the following code:

```
<cfcollection action = "create"
   collection = "a_new_collection"
   path = "c:\CFusionMX\verity\collections\">
```

If you want your users to be able to dynamically supply the name and location for a new collection, use the following procedures to create form and action pages.

To create a simple collection form page:

1 Create a ColdFusion page with the following content:

```
<html>
<head>
   <title>Collection Creation Input Form</title>
</head>

<body>
<h2>Specify a collection</h2>
<form action="collection_create_action.cfm" method="POST">

   <p>Collection name:
   <input type="text" name="CollectionName" size="25"></p>

   <p>What do you want to do with the collection?</p>
   <input type="radio"
      name="CollectionAction"
      value="Create" checked>Create<br>
   <input type="radio"
      name="CollectionAction"
      value="Repair">Repair<br>
   <input type="radio"
      name="CollectionAction"
      value="Optimize">Optimize<br>
   <input type="submit"
      name="submit"
      value="Submit">
</form>

</body>
</html>
```

2 Save the file as collection_create_form.cfm in the myapps directory under the web root directory.

Note: The form will not work until you write an action page for it, which is the next procedure.

To create a collection action page:

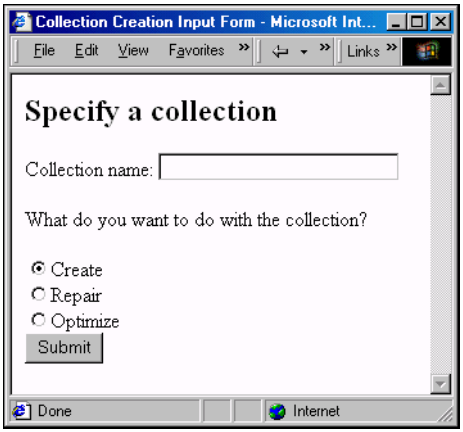
1 Create a ColdFusion page with the following content:

```
<html>
<head>
   <title>cfcollection</title>
</head>

<body>
<h2>Collection creation</h2>

<cfoutput>

   <cfswitch expression=#Form.collectionaction#>
      <cfcase value="Create">
         <cfcollection action="Create"
         collection="#Form.CollectionName#"
         path="c:\cfusionmx\verity\collections\">
         <p>The collection #Form.CollectionName# is created.
      </cfcase>

      <cfcase value="Repair">
         <cfcollection action="Repair"
         collection="#Form.CollectionName#">
         <p>The collection #Form.CollectionName# is repaired.
      </cfcase>

      <cfcase value="Optimize">
         <cfcollection action="Optimize"
         collection="#Form.CollectionName#">
         <p>The collection #Form.CollectionName# is optimized.
      </cfcase>

      <cfcase value="Delete">
         <cfcollection action="Delete"
         collection="#Form.CollectionName#">
         <p>Collection deleted.
      </cfcase>
   </cfswitch>
</cfoutput>
</body>
</html>
```

2 Save the file as collection_create_action.cfm in the myapps directory under the web root directory.

3 In the web browser, enter the following URL to display the form page:
http://127.0.0.1/myapps/collection_create_form.cfm

The following figure shows how the output appears:



4 Enter a collection name; for example, CodeColl.

5 Verify that Create is selected and submit the form.

6 (Optional) In the ColdFusion Administrator, reload the Verity Collections page.

The name and full path of the new collection appears in the list of Connected Verity Collections.

You successfully created a collection, named CodeColl, that currently has no data. For information on indexing your collection using CFML, see "Indexing a collection using the cfindex tag" on page 538.

Indexing a collection using the cfindex tag

You can index a collection in CFML using the `cfindex` tag, which eliminates the need to use the ColdFusion MX Administrator. When using this tag, the following attributes correspond to values entered in the ColdFusion MX Administrator:

- **collection** The name of the collection. If you are indexing an external collection (external = "Yes"), you must also specify the fully qualified path for the collection.
- **action** (Optional) Can be update (the default action), delete, purge, or refresh.
- **extensions** (Optional) The delimited list of file extensions that ColdFusion uses to index files if `type="Path"`.
- **key** (Optional) The path containing the files you are indexing if `type="path"`.
- **URLpath** (Optional) The URL path for files if `type="file"` and `type="path"`. When the collection is searched with `cfsearch`, the pathname is automatically prefixed to filenames and returned as the url attribute.
- **recurse** (Optional) Yes or No. Yes specifies, if `type = "Path"`, that directories below the path specified in the `key` attribute are included in the indexing operation.
- **language** (Optional) The language of the collection. English is the default.

You can use form and action pages similar to the following examples to select and index a collection.

To select which collection to index:

1 Create a ColdFusion page with the following content:

```
<html>
<head>
   <title>Select the Collection to Index</title>
</head>
<body>

<h2>Specify the index you want to build</h2>

<form method="Post" action="collection_index_action.cfm">
   <p>Enter the collection you want to index:
   <input type="text" name="IndexColl" size="25" maxLength="35"></p>
   <p>Enter the location of the files in the collection:
   <input type="text" name="IndexDir" size="50" maxLength="100"></p>

   <input type="submit" name="submit" value="Index">

</form>

</body>
</html>
```

2 Save the file as collection_index_form.cfm in the myapps directory under the *web_root*.

Note: The form will not work until you write an action page for it, which is the next procedure.

To use cfindex to index a collection:

1 Create a ColdFusion page with the following content:

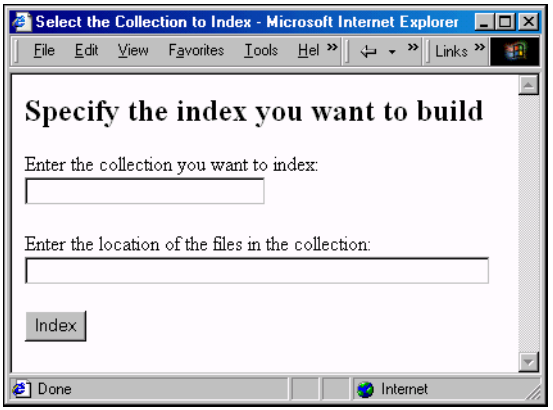
```
<html>
<head>
<title>Creating Index</title>
</head>
<body>
<h2>Indexing Complete</h2>

<cfindex collection="#Form.IndexColl#"
   action="refresh"
   extensions=".htm, .html, .xls, .txt, .mif, .doc"
   key="#Form.IndexDir#"
   type="path"
   urlpath="#Form.IndexDir#"
   recurse="Yes"
   language="English">

<cfoutput>
   The collection #Form.IndexColl# has been indexed.
</cfoutput>
</body>
</html>
```

2 Save the file as collection_index_action.cfm.

3 In the web browser, enter the following URL to display the form page:

http://127.0.0.1/myapps/collection_index_form.cfm

The following figure shows how the output appears:



4 Enter a collection name; for example, CodeColl.

5 Enter a file location; for example, C:\CFusionMX\wwwroot\vw_files.

6 Click Index.

A confirmation message appears upon successful completion.

Note: For information about using the `cfindex` tag with a database to index a collection, see "Using database-directed indexing" on page 551.

Indexing a collection with the ColdFusion Administrator

As an alternative to programmatically indexing a collection and to using the Verity Wizard, use the following procedure to quickly index a collection with the ColdFusion Administrator.

To use ColdFusion Administrator to index a collection:

1 In the list of Connected Verity Collections, select a collection name; for example, CodeColl.

2 Click Index to open the index page.

3 For File Extensions, enter the type(s) of files to index. Use a comma to separate multiple file types; for example, .htm, .html, .xls, .txt, .mif, .doc.

4 Enter (or Browse to) the directory path that contains the files to be indexed; for example, C:\Inetpub\wwwroot\vw_files.

5 (Optional) To extend the indexing operation to all directories below the selected path, select the Recursively index subdirectories check box.

6 (Optional) Enter a Return URL to prepend to all indexed files.

This step lets you create a link to any of the files in the index; for example, http://127.0.0.1/vw_files/.

7 (Optional) Select a language other than English.

For more information, see "Support for international languages" on page 526.

8 Click Submit Changes.

The indexing process. On completion, the Verity Collections page appears.

Note: The time required to generate the index depends on the number and size of the selected files in the path.

This interface lets you easily build a very specific index based on the file extension and path information you enter. In most cases, you do not need to change your server file structures to accommodate the generation of indices.

Using the cfsearch tag

You use the cfsearch tag to search an indexed collection. Searching a Verity collection is similar to a standard ColdFusion query: both use a dedicated ColdFusion tag that requires a name attribute for their searches. The following table compares the two tags:

cfquery	cfsearch
Searches a data source	Searches a collection
Requires name attribute	Requires name attribute
Uses SQL statements to specify search criteria	Uses a criteria attribute to specify search criteria
Returns variables keyed to database table field names	Returns a unique set of variables
Uses cfoutput to display query results	Uses cfoutput to display search results

Note: You receive an error if you attempt to search a collection that has not been indexed.

The following are important attributes for the cfsearch tag:

- name The name of the search query.
- collection The name of the collection(s) being searched. Use a fully qualified path for an external collection. Separate multiple collections with a comma; for example, collection = "sprocket_docs,CodeColl".
- criteria The search target (can be dynamic).

Each cfsearch returns variables that provide the following information about the search:

- RecordCount The total number of records returned by the search.
- CurrentRow The current row of the record set being processed by cfoutput.
- RecordsSearched The total number of records in the index that were searched. If no records were returned in the search, this property returns a null value.

Note: To use cfsearch to search a Verity K2 Server collection, the collection attribute must be the collection's unique alias name as defined in the k2server.ini and the external attribute must be "No" (the default). For more detail, see *Administering ColdFusion MX*.

You can use search form and results pages similar to the following examples to search a collection.

To create a search form:

1 Create a ColdFusion page with the following content:

```
<html>
<head>
   <title>Searching a collection</title>
</head>
<body>
<h2>Searching a collection</h2>

<form method="post" action="collection_search_action.cfm">
   <p>Enter search term(s) in the box below. You can use AND, OR, NOT, and
   parentheses. Surround an exact phrase with quotation marks.</p>
   <p><input type="text" name="criteria" size="50" maxLength="50">
```

```
    </p>
    <input type="submit" value="Search">
</form>
</body>
</html>
```

2 Save the file as collection_search_form.cfm.

Enter a search target word(s) in this form, which passes this as the variable `criteria` to the action page, which displays the search results.

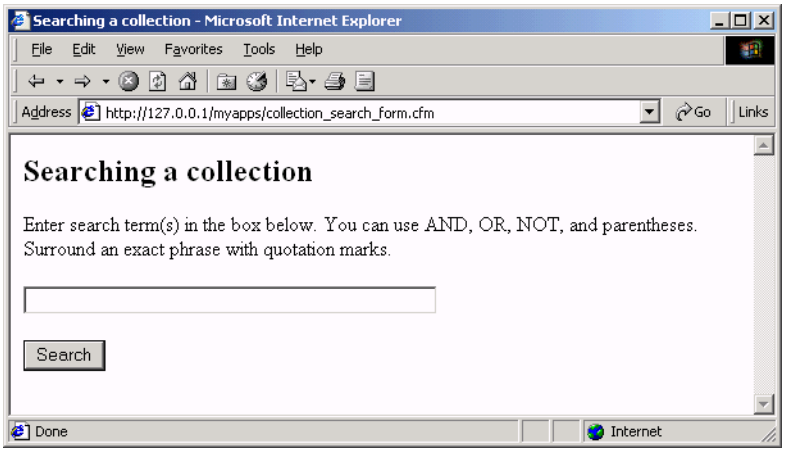
To create the results page:

1 Create a ColdFusion page with the following content:

```
<html>
<head>
   <title>Search Results</title>
</head>
<body>
<cfsearch
   name = "codecoll_results"
   collection = "CodeColl"
   criteria = "#Form.Criteria#">

<h2>Search Results</h2>
<cfoutput>
Your search returned #codecoll_results.RecordCount# file(s).
</cfoutput>

<cfoutput query="codecoll_results">
   <p>
   File: <a href="#URL#">#Key#</a><br>
   Document Title (if any): #Title#<br>
   Score: #Score#<br>
   Summary: #Summary#</p>
</cfoutput>
</body>
</html>
```

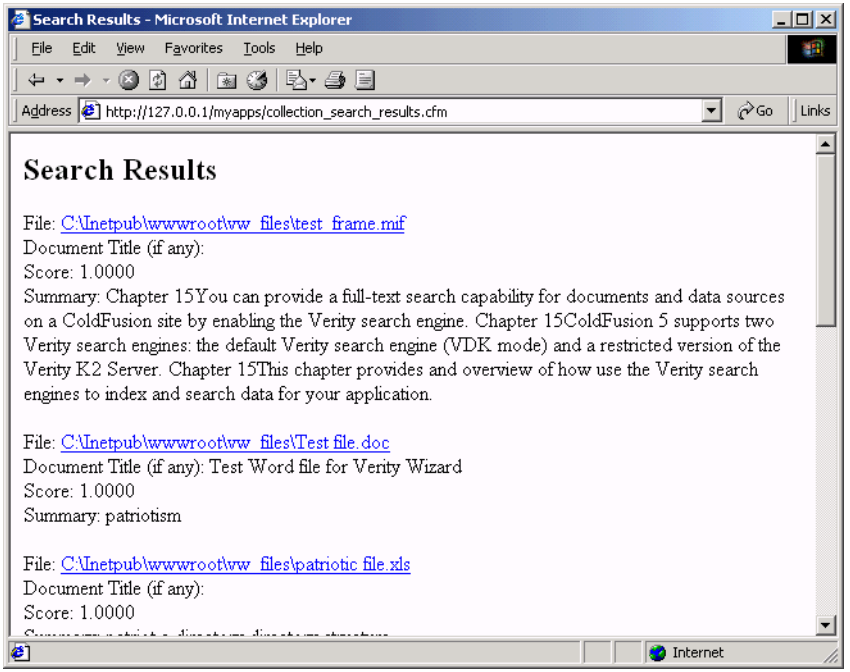
2 Save the file as collection_search_action.cfm.

3 View collection_search_form.cfm in the web browser:



4 Enter a target word(s) and click Search.

The following figure shows how the output appears:



Note: As part of the indexing process, Verity automatically produces a summary of every document file or every query record set that gets indexed. The default summary selects the best sentences, based on internal rules, up to a maximum of 500 characters. Every `cfsearch` operation returns summary information by default. For more information on this topic, see "Using Verity Search Expressions" on page 553.

Working with record sets

The cfquery, cfldap, and cfpop tags return the results of a database query in a record set. In some cases, you might want to search the record set. This section describes the reasons and procedures for indexing the results of database, LDAP, and pop queries. It also describes how a database can direct the indexing process, using different values for the type attribute of the cfindex tag.

Indexing database record sets

The following are the steps to perform a Verity search on record sets:

1 Write a query to generate a record set.

2 Index the record set.

3 Search the record set.

Performing searches against a Verity collection rather than using cfquery provides faster access, because the Verity collection indexes the database. Use this technique instead of cfquery in the following cases:

- You want to index textual data. You can search Verity collections containing textual data much more efficiently with a Verity search than with a SQL query.
- You want to give your users access to data without interacting directly with the data source itself.
- You want to improve the speed of queries.
- You want your users to run queries but not update database tables.

Indexing the record set from a ColdFusion query involves an extra step not required when you index documents. You must code the query and output parameters, and then use the cfindex tag to index the record set from a cfquery, cfldap, or cfpop query.

You write a cfquery that retrieves the data to index, then you pass this information to a cfindex tag, which populates the collection. The cfindex tag contains the following attributes that correspond to the data source:

The cfindex attribute	Description
key	Primary key of the data source table
title	Specifies a query column name
body	Column(s) that you want to search for the index

Using the cfindex tag on large custom query data can cause a “Java out of memory error” or lead to excessive disk use on your computer. Because ColdFusion reads custom queries into memory, if the query size is larger than your physical memory, then paging of physical memory to disk occurs. The size of physical memory used is the smaller of the actual physical memory on your computer and the Java Virtual Machine (JVM) masimum memory parameter. You can specify the JVM parameter in the Administrator or in the configuration file cfsuionmx/runtime/bin/jvm.config by the argument [-Xmx512m].

The following procedure assumes that you have a Verity collection named CodeColl. For more information, see "Creating a collection with the cfcollection tag" on page 535. The following procedure uses the CompanyInfo data source that is installed with ColdFusion.

To index a ColdFusion query:

1 Create a ColdFusion page with the following content:

```
<html>
<head>
   <title>Adding Query Data to an Index</title>
</head>
<body>

<!--- retrieve data from the table --->
<cfquery name="getEmps" datasource="CompanyInfo">
   SELECT * FROM EMPLOYEE
</cfquery>

<!--- update the collection with the above query results --->
<cfindex
   query="getEmps"
   collection="CodeColl"
   action="Update"
   type="Custom"
   key="Emp_ID"
   title="Emp_ID"
   body="Emp_ID,FirstName,LastName,Salary">

<h2>Indexing Complete</h2>

<!--- output the record set --->
<p>Your collection now includes the following items:</p>
<cfoutput query="getEmps">
   <p>#Emp_ID# #FirstName# #LastName# #Salary#</p>
</cfoutput>
</body>
</html>
```

2 Save the file as collection_db_index.cfm in the myapps directory under the web root directory.

3 Open the file in the web browser to index the collection.
The resulting record set appears:

Using the `cfindex` tag for indexing tabular data is similar to indexing documents, with the following exceptions:

- You set the `type` attribute to `custom` when indexing tabular data.
- You refer to column names from the `cfquery` in the `body` attribute.

To search and display database records:

1 Create a ColdFusion page with the following content:

```
<html>
<head>
   <title>Searching a collection</title>
</head>
<body>

<h2>Searching a collection</h2>

<form method="post" action="collection_db_results.cfm">
   <p>Collection name: <input type="text" name="collname" size="30"
      maxLength="30"></p>

   <p>Enter search term(s) in the box below. You can use AND, OR, NOT,
   and parentheses. Surround an exact phrase with quotation marks.</p>
   <p><input type="text" name="criteria" size="50" maxLength="50">
   </p>
   <p><input type="submit" value="Search"></p>
</form>

</body>
</html>
```

2 Save the file as collection_db_search_form.cfm in the myapps directory under the *web_root*.

This file is similar to collection_search_form.cfm, except the form uses collection_db_results.cfm, which you create in the next step, as its action page.

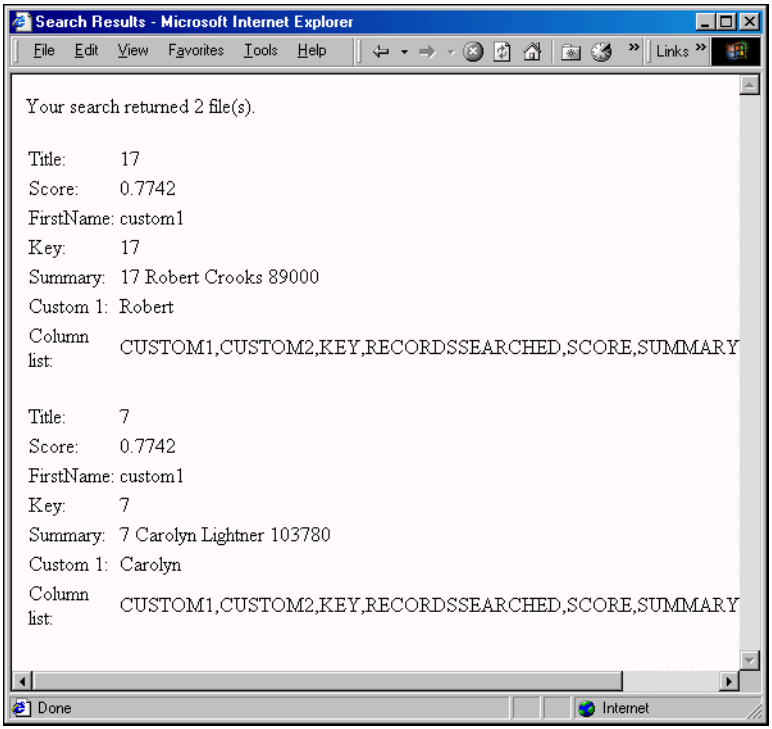
3 Create another ColdFusion page with the following content:

```
<html>
<head>
<title>Search Results</title>
</head>

<body>

<cfsearch
   collection="#Form.collname#"
   name="getEmps"
   criteria="#Form.Criteria#">


<!--- output the record set --->
<cfoutput>
Your search returned #getEmps.RecordCount# file(s).
</cfoutput>

<cfoutput query="getEmps">
   <p><table>
   <tr><td>Title: </td><td>#Title#</td></tr>
   <tr><td>Score: </td><td>#Score#</td></tr>
   <tr><td>Key: </td><td>#Key#</td></tr>
   <tr><td>Summary: </td><td>#Summary#</td></tr>
   <tr><td>Custom 1:</td><td>#Custom1#</td></tr>
   <tr><td>Column list: </td><td>#ColumnList#</td></tr>
</table></p>

</cfoutput>

</body>
</html>
```

4 Save the file as collection_db_results.cfm in the myapps directory under the *web_root*.

5 View collection_db_search_form.cfm in the web browser and enter the name of the collection and search terms; for example, search the CodeColl collection for lightner or crooks.

The following figure shows how the output appears:



Indexing cfldap query results

The widespread use of the Lightweight Directory Access Protocol (LDAP) to build searchable directory structures, internally and across the web, gives you opportunities to add value to the sites that you create. You can index contact information or other data from an LDAP-accessible server and allow users to search it.

When creating an index from an LDAP query, remember the following considerations:

- Because LDAP structures vary greatly, you must know the server's directory schema and the exact name of every LDAP attribute that you intend to use in a query.
- The records on an LDAP server can be subject to frequent change, so re-index the collection before processing a search request.

In the following example, the search criterion is records with a telephone number in the 617 area code. Generally, LDAP servers use the Distinguished Name (dn) attribute as the unique identifier for each record so that attribute is used as the `key` value for the index.

```
<!--- Run the LDAP query --->
<cfldap name="OrgList"
   server="myserver"
   action="query"
   attributes="o, telephonenumber, dn, mail"
   scope="onelevel"
   filter="(|(O=a*) (O=b*))"
   sort="o"
```

```
   start="c=US">

<!--- Output query record set --->
<cfoutput query="OrgList">
   DN: #dn# <br>
   O: #o# <br>
   TELEPHONENUMBER: #telephonenumber# <br>
   MAIL: #mail# <br>
==========<br>
</cfoutput>

<!--- Index the record set --->
<cfindex action="update"
   collection="ldap_query"
   key="dn"
   type="custom"
   title="o"
   query="OrgList"
   body="telephonenumber">

<!--- Search the collection --->
<!--- Use the wildcard * to contain the search string --->
<cfsearch collection="ldap_query"
   name="s_ldap"
   criteria="*617*">

<!--- Output returned records --->
<cfoutput query="s_ldap">
   #Key#, #Title#, #Body# <br>
</cfoutput>
```

Indexing cfpop query results

The contents of mail servers are generally volatile; specifically, the message number is reset as messages are added and deleted. To avoid mismatches between the unique message number identifiers on the server and in the Verity collection, you must re-index the collection before processing a search.

As with the other query types, you must provide a unique value for the `key` attribute and enter the data fields to index in the `body` attribute.

The following example updates the pop_query collection with the current mail for user1, and searches and returns the message number and subject line for all messages containing the word **action**:

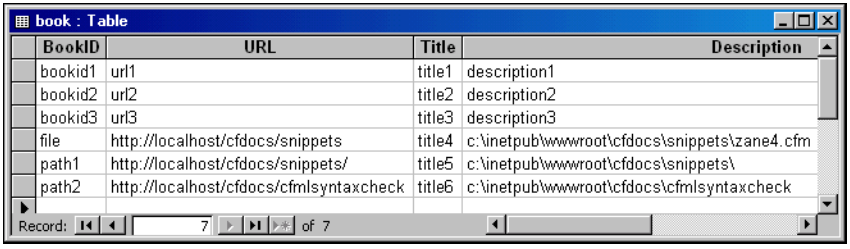
```
<!--- Run POP query --->
<cfpop action="getall"
   name="p_messages"
   server="mail.company.com"
   userName="user1"
   password="user1">

<!--- Output POP query record set --->
<cfoutput query="p_messages">
   #messagenumber# <br>
   #from# <br>
```

```
   #to# <br>
   #subject# <br>
   #body# <br>
<hr>
</cfoutput>

<!--- Index record set --->
<cfindex action="update"
   collection="pop_query"
   key="messagenumber"
   type="custom"
   title="subject"
   query="p_messages"
   body="body">

<!--- Search messages for the word "action" --->
<cfsearch collection="pop_query"
   name="s_messages"
   criteria="action">

<!--- Output search record set --->
<cfoutput query="s_messages">
   #key#, #title# <br>
</cfoutput>
```

Using database-directed indexing

You can use the `cfindex` tag with a database that contains information on how to construct, or populate, the index. The `cfindex` tag has a `type` attribute, which can have `custom`, `file`, or `path` as its value. When `type=custom`, ColdFusion populates a collection with the contents of the record set. When `type=file` or `type=custom`, the record set becomes the input to perform any action—as defined by the `action` attribute—that uses the `key` attribute as input for filenames or filepaths.

The following figure shows a database that you can use to populate a collection:



The following code shows how to populate a collection named snippets with files that are specified in the description column of the database:

```
<html>
<head>
   <title>Database-directed index population</title>
</head>

<body>

<cfquery name="bookquery"
    datasource="book">
    SELECT * FROM book where bookid='file'
        </cfquery>

<cfoutput query="bookquery">
    #url#,#description# <br>

<cfindex collection="snippets" action="update" type="file" query="bookquery"
          key="description" urlpath="url">

</cfoutput>
</body>
</html>
```

Use the following code to search the snippets collection and display the results:

```
<cfsearch name="mySearch" collection="snippets" criteria="*.,.*">
<cfdump var="#mySearch#">
```

The following code shows how to populate the snippets collection with paths that are specified in the description column of the database:

```
<html>
<cfquery name="bookquery"
    datasource="book">
    SELECT * FROM book where bookid='path1' or bookid='path2'
        </cfquery>

<cfoutput query="bookquery">
    #url#,#description# <br>

<cfindex collection="snippets" action="update" type="path" query="bookquery"
          key="description" urlpath="url" >

</cfoutput>
```