SYBASE®

Administration and User's Guide

Sybase® Search

3.2

[Microsoft Windows, UNIX, and Linux]

DOCUMENT ID: DC35131-01-0320-01

LAST REVISED: March 2007

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Contents

About This Book

Audience

This guide is for Sybase® Search administrators and other professionals who are familiar with their system's environment, networks, disk resources, and media devices.

How to use this book

This book contains the following chapters:

- Chapter 1, "Retrieving Information Intelligently," introduces Sybase Search and describes the features and architecture of Sybase Search.
- Chapter 2, "Administering Sybase Search," describes how to administer Sybase Search. It includes information on setting up Sybase Search, accessing administration pages, and performing various administration tasks.
- Chapter 3, "Configuring Sybase Search," describes the key configuration parameters for containers, modules, and the hub. It includes tips for using the configuration files and changing parameters
- Chapter 4, "Configuring Web Administration," describes the key configuration parameters for the Web application provided with Sybase Search.
- Chapter 5, "Customizing Sybase Search," provides information about developing custom filters, parsers, and text splitters.
- Chapter 6, "Using Sybase Search," provides information about using Sybase Search to search across documents.
- Appendix A, "Generated Files," gives the names and locations of generated module files.
- Appendix B, "Sybase Search Content Adapter," provides information about installing, configuring, and uninstalling Sybase Search Content Adapter.

Related documents

The Sybase Search documentation set consists of the following:

- *Sybase Search Release Bulletin for Microsoft Windows, Linux, and UNIX* – contains last-minute information that was too late to be included in the books.

- *Sybase Search New Features* – describes the new features in Sybase Search 3.2.

Other sources of information

Use the Sybase Getting Started CD, the SyBooks™ CD, and the Sybase Product Manuals Web site to learn more about your product:

- The Getting Started CD contains release bulletins and installation guides in PDF format, and may also contain other documents or updated information not included on the SyBooks CD. It is included with your software. To read or print documents on the Getting Started CD, you need Adobe Acrobat Reader, which you can download at no charge from the Adobe Web site using a link provided on the CD.
- The SyBooks CD contains product manuals and is included with your software. The Eclipse-based SyBooks browser allows you to access the manuals in an easy-to-use, HTML-based format.

  Some documentation may be provided in PDF format, which you can access through the PDF directory on the SyBooks CD. To read or print the PDF files, you need Adobe Acrobat Reader.

  Refer to the *SyBooks Installation Guide* on the Getting Started CD, or the *README.txt* file on the SyBooks CD for instructions on installing and starting SyBooks.
- The Sybase Product Manuals Web site is an online version of the SyBooks CD that you can access using a standard Web browser. In addition to product manuals, you will find links to EBFs/Maintenance, Technical Documents, Case Management, Solved Cases, newsgroups, and the Sybase Developer Network.

  To access the Sybase Product Manuals Web site, go to Product Manuals at http://www.sybase.com/support/manuals/.

Sybase certifications on the Web

Technical documentation at the Sybase Web site is updated frequently.

❖ **Finding the latest information on product certifications**

1. Point your Web browser to Technical Documents at http://www.sybase.com/support/techdocs/.
2. Click Certification Report.
3. In the Certification Report filter select a product, platform, and timeframe and then click Go.
4. Click a Certification Report title to display the report.

❖ **Finding the latest information on component certifications**

1 Point your Web browser to Availability and Certification Reports at http://certification.sybase.com/.

2 Either select the product family and product under Search by Base Product; or select the platform and product under Search by Platform.

3 Select Search to display the availability and certification report for the selection.

❖ **Creating a personalized view of the Sybase Web site (including support pages)**

Set up a MySybase profile. MySybase is a free service that allows you to create a personalized view of Sybase Web pages.

1 Point your Web browser to Technical Documents at http://www.sybase.com/support/techdocs/.

2 Click MySybase and create a MySybase profile.

Sybase EBFs and software maintenance

❖ **Finding the latest information on EBFs and software maintenance**

1 Point your Web browser to the Sybase Support Page at http://www.sybase.com/support.

2 Select EBFs/Maintenance. If prompted, enter your MySybase user name and password.

3 Select a product.

4 Specify a time frame and click Go. A list of EBF/Maintenance releases is displayed.

Padlock icons indicate that you do not have download authorization for certain EBF/Maintenance releases because you are not registered as a Technical Support Contact. If you have not registered, but have valid information provided by your Sybase representative or through your support contract, click Edit Roles to add the “Technical Support Contact” role to your MySybase profile.

5 Click the Info icon to display the EBF/Maintenance report, or click the product description to download the software.

Conventions

The syntax conventions used in this manual are:

Key	Definition
commands and methods	Command names, command option names, utility names, utility flags, Java methods/classes/packages, and other keywords are in lowercase Arial font.
variable	Italic font indicates: • Program variables, such as *myServer* • Parts of input text that must be substituted; for example: *Server*.log • File names
File | Save	Menu names and menu items are displayed in plain text. The vertical bar shows you how to navigate menu selections. For example, File | Save indicates "select Save from the File menu."
`package 1`	Courier font indicates: • Information that you enter in a GUI interface, a command line, or as program text • Sample program fragments • Sample output fragments
install_location	Refers to the Sybase Search installation directory \Sybase\Search-3_2.
sybase\bin	A backward slash ("\") indicates cross-platform directory information. A forward slash ("/") applies to information specific only to UNIX. Directory names appearing in text display in lowercase unless the system is case sensitive.

Accessibility features

This document is available in an HTML version that is specialized for accessibility. You can navigate the HTML with an adaptive technology such as a screen reader, or view it with a screen enlarger.

Sybase Search documentation has been tested for compliance with U.S. government Section 508 Accessibility requirements. Documents that comply with Section 508 generally also meet non-U.S. accessibility guidelines, such as the World Wide Web Consortium (W3C) guidelines for Web sites.

Note You might need to configure your accessibility tool for optimal use. Some screen readers pronounce text based on its case; for example, they pronounce ALL UPPERCASE TEXT as initials, and MixedCase Text as words. You might find it helpful to configure your tool to announce syntax conventions. Consult the documentation for your tool.

For information about how Sybase supports accessibility, see Sybase Accessibility at http://www.sybase.com/accessibility. The Sybase Accessibility site includes links to information on Section 508 and W3C standards.

If you need help

Each Sybase installation that has purchased a support contract has one or more designated people who are authorized to contact Sybase Technical Support. If you cannot resolve a problem using the manuals or online help, please have the designated person contact Sybase Technical Support or the Sybase subsidiary in your area.

CHAPTER 1

Retrieving Information Intelligently

Sybase Search is a knowledge management system that automates the process of locating relevant business information within the masses of unstructured information stored in your organization's network drives, databases, intranets, and the Internet. Sybase Search provides intelligent information retrieval, document index management, and document categorization.

Using the content-based catalog and search tool, Sybase Search automatically analyzes, indexes, and categorizes data and prepares the system for users to visually navigate to the chosen category.

Managing unstructured information

In many organizations, employees keep their research, financial projections, and presentations on local PCs or team-shared space on the company network. Accessing such information—or even finding it—often proves difficult when staff or storage rules and structures change.

Sybase Search technology

Sybase Search extracts and processes the text content from file systems and databases where the content is unstructured. The ability to automatically process this content removes the need to index or describe information manually, and allows organizations to automate such common business operations as data capture, retrieval, and linking. Sybase Search technology offers an efficient and cost-effective solution for searching unstructured information, regardless of the format and the language in which the content is written.

The Internet offers familiarity with keyword search, which is the most common type of search-and-retrieval technology. Most people are familiar with the process of retrieving the information by typing one or two relevant keywords into a search engine.

However, keyword search technology requires a business to identify documents by associating keywords with the document, which are then used for retrieval. This process, known as document "tagging," can be costly and time-consuming.

Sybase Search provides the means to automatically capture and retrieve information based on concepts rather than keywords. Through the use of proprietary algorithms, Sybase Search delivers a language-independent product capable of operating without the costly overhead associated with tagging. This provides an essential tool for managing the proliferation of unstructured information in today's business environment.

Sybase Search features

Sybase Search offers a number of features that allow today's organizations to reduce or eliminate the time and cost required to support the demands of managing an organization's unstructured information. These features include:

- The support of more than 250 different formats of data, including most types of document, presentation, spreadsheet, and Web content formats
- The automatic capture and aggregation all of unstructured data
- The elimination of preprocessing or manual tagging of files, greatly improving the accuracy and efficiency of document retrieval
- The extraction of paragraphs from matching documents
- The ability to find similar documents by automatically providing a set of relevant content that is conceptually related to each document
- The ability to scale to millions of documents using a fully distributed architecture
- The ability to query and process using a natural language
- Language independence
- A well-defined Java and XML API that allows Sybase Search to integrate easily into other applications

Understanding the architecture of Sybase Search

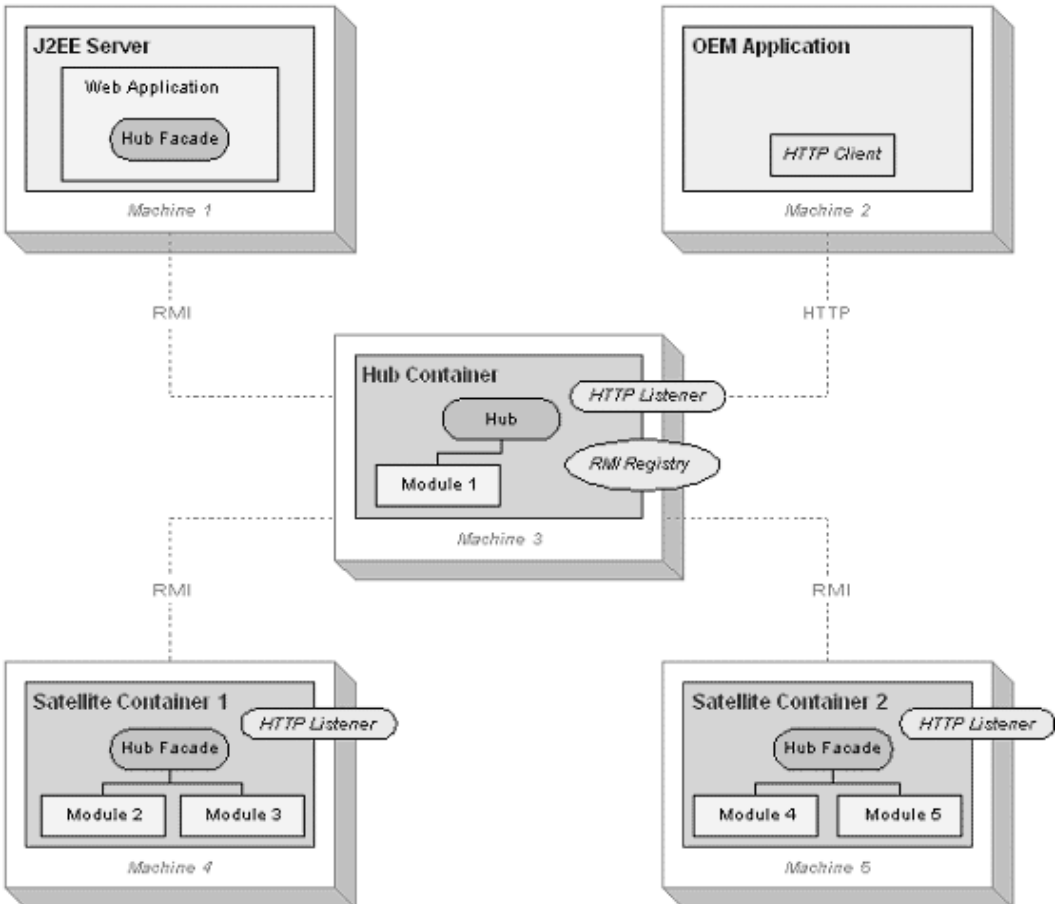
Sybase Search is a fully distributed system, with a central hub server and one or more satellite servers. Each server can contain one or many containers with one or more modules deployed in each container. The exact number of servers, containers, and modules depends on the needs of the Sybase Search installation.

The example architecture in Figure 1-1 contains:

- A central hub
- Two satellite containers
- A J2EE server containing the Web application
- OEM application connecting to Sybase Search

For information about the various modules that comprise Sybase Search, see Chapter 3, “Configuring Sybase Search.”

Figure 1-1: Example of Sybase Search 3.2 architecture



Optimizing search strategies

As a **concept-based search engine**, Sybase Search performs best when you enter queries with search words in context in short phrases rather than as isolated words. In addition, if you know that more than one language is in use, repeating the concepts using different words generally improves results. Searching is often an iterative activity: you expand and refine queries based on the results returned.

Tips for optimizing the search engine

This section provides tips for optimizing a concept-based search engine, which provides greater flexibility than traditional approaches to free-text searching, such as the Boolean combination of keywords.

For example, a user receives an e-mail message that says:

Following the incident close to Watford railway station in July, we need to assess the damage being done by tree branches tangling in overhead power lines or falling onto the tracks.

The user then wants to locate documents matching the e-mail message. Using a traditional search method, he or she might enter something similar to:

```
branches AND lines AND tracks
```

In this query, the user is using the Boolean operator "AND" to filter the information. This type of query is very precise and is helpful when:

- The user knows exactly what information is required, and it can be expressed in just a few words.
- There is no ambiguity in the words used in the query.
- The vocabulary of the target documents is known precisely.

In practice, this is rarely the case. It is more common that users are unsure of how to formulate their query precisely, thus introducing ambiguity within the query. Differing vocabulary used in documents to describe similar concepts can also result in important documents being missed altogether and too many irrelevant documents being returned.

If the user is searching a large database of documents, a query like the one in the previous example may retrieve a large number of items, many of which are not relevant to the specific query due to the search for a small number of specific, isolated words. Words like "branches" and "lines" are ambiguous and are common in a database of documentation concerning the railway system.

Query a number of concepts

Sybase Search is better suited to a query that contains a number of concepts and is expressed using ambiguous language, thus increasing the likelihood that the user retrieves results that are relevant to the query.

Using the previous e-mail example, isolate the key concepts, which are:

- Damage being done by tree branches
- Tangling of overhead power lines
- Falling trees and tree branches
- Obstruction or damage to tracks

Irrelevant concepts might include:

- Watford Railway Station
- July

Inclusion of irrelevant concepts distorts the search and may introduce some unwanted documents. So, a more effective query is:

```
damage being done by tree branches, tangling of overhead
power lines, falling tree branches, obstruction and
damage to tracks
```

Note You do not need to delimit concepts using a comma.

This is a better query because it contains all of the key concepts in the original query and expresses them using words in context. Results returned by this query are likely to produce significantly better results than the first attempt.

Adding variations

However, it is possible that some relevant documents will still be missed, due to differing vocabulary. Therefore, if you use your knowledge of the environment and expand the original concepts to include variations that you know from experience tend to occur, this may produce a query similar to:

```
damage being done by tree branches, tangling of overhead
power lines, falling tree branches, obstruction and
damage to tracks, forestry, wind damage, storm damage,
damage to rails, lines being pulled down by trees blown
over
```

At first, this may seem more confusing and less precise than the previous examples, but in fact it contains additional ways of defining the original concepts. You may find that no documents achieve a 100% relevance score with this query because no document includes all of these combinations. However, the most relevant documents are at the top of the list.

Often, you can improve search results by feeding back information from documents discovered by the system. For example, if a search produces a document that is relevant but the terminology used in the extracted summary is different from the search text, you may want to expand the original query by appending words or phrases from the document search results. In this way, the search becomes more accurate as you provide additional information.

Improving relevance

Sybase Search automatically determines the documents that are more relevant than others. This decision is based on the information extracted from all the documents that are indexed by Sybase Search. Part of the relevance calculation assigns an internal weighting for each term in the search query. Depending on the search results, you may want to manually adjust the query term weighting in order to bias the search results in favor of a particular query term.

For example, Sybase Search has indexed many documents about trains and railway accidents, and incidents. A typical query to find documents about tree branches causing damage to either trains or track can be:

```
damage being done by tree branches
```

Sybase Search can return relevant documents about damage to branches in the rail tracks that were not caused by trees. This can occur if Sybase Search has indexed documents that are exclusively about "damage to branches in railway tracks," while the documents about "tree branches causing damage" have other sections about other topics. While the second set of documents have relevant matching sections, they are not as relevant overall and is assigned a lower relevance score.

Based on the search results, you can decide to place more emphasis on "tree" damage as opposed to other types of damage. You can use custom term weighting to make your search results more relevant to documents that have references to trees:

```
damage being done by ctw{tree,5} branches
```

Depending on the results from this search, you can further adjust the custom term weighting to get appropriate emphasis on the term "tree."

CHAPTER 2

Administering Sybase Search

This chapter describes how to administer Sybase Search. It includes information on setting up Sybase Search, starting and stopping Sybase Search, accessing administration pages, tracking system details, scheduling tasks, managing documents, and categorizing documents.

Setting up Sybase Search

The internal data structures of Sybase Search rely on certain settings remaining unchanged. Therefore, before you start Sybase Search for the first time and before any indexing is performed, you must perform:

- Language configuration – before starting Sybase Search, you must decide which languages Sybase Search will be required to support.

  If Sybase Search is required to work in a language other than English or across multiple, different languages, consideration should be given to stopwords, word splitting, and word stemming configuration. Depending on the languages involved, Sybase Search can provide better results if the stopwords are revised, and the word splitter and stemmer are disabled or replaced with custom implementations.

  For more information about language-specific configuration of various modules, see:

- "Setting Text Manager parameters" on page 51.
- "Optimizing Sybase Search" on page 59.

For more details on how to write and plug in a new language stemmer into Sybase Search, contact Sybase Technical Support.

- Hub configuration – base this on the level of stress and load the system is expected to cope with. This includes ensuring the various module caches are set to a high enough level, where RAM is available.

  For example, if the Query Module is located on the hub, you must review its settings to make sure that it can handle the required number of concurrent queries.

  For details on configuring various hub-specific modules, see "Configuring modules" on page 50.

- Satellite container configuration – base this on the level of stress and load the system is expected to cope with. This includes ensuring that the various module caches are set to a high enough level, where RAM is available.

  For more details on configuring various satellite container modules, see "Configuring modules" on page 50 and "Configuring modules using system parameters" on page 71.

Installing Sybase Search as a Windows service

You can install and run any Sybase Search container as a Windows service.

1 From a Windows command prompt, navigate to:

```
install_location\OmniQ\bin
```

2 Enter:

```
sysearch32.bat -install ID
```

where *ID* is the container ID of the container you want to install as a Windows service. The container ID of the hub container is always 1.

This installs the "Sybase Search – container ID." Then, you can use the Microsoft Management Console to run the Sybase Search container as a Windows service.

You can also install and run the Web administration server as a Windows service:

1 From a Windows command prompt, navigate to:

```
install_location\Hyena\bin
```

2 Enter:

```
Hyena32.bat -install
```

This installs the service named "Sybase Search – Web Admin Server." Then, you can use the Microsoft Management Console to run Web administration server as a Windows service.

Uninstalling the Windows service

If you installed a Sybase Search container as a Windows service, you must invoke *sysearch32.bat* with the uninstall parameter and the container ID to remove this service:

1 From a Windows command prompt, navigate to:

```
install_location\OmniQ\bin
```

2 Enter:

```
sysearch32.bat -uninstall ID
```

where *ID* is the container ID of the container you want to uninstall as a Windows service. The container ID of the hub container is always 1.

If you installed the Sybase Search Web administration server as a Windows service, you must invoke *Hyena32.bat* with the uninstall parameter to remove this service:

1 From a Windows command prompt, navigate to:

```
install_location\Hyena\bin
```

2 Enter:

```
Hyena32.bat -uninstall
```

Starting and stopping Sybase Search

Windows

In Windows, you can start and stop Sybase Search containers and the Web administration server from the Windows Start Menu or from a command prompt.

❖ **To start Sybase Search from the Windows Start menu**

1 Select Programs | Sybase.

2 Select Sybase Search 3.2.

- To start a single server installation, select Start Single Server. The single container and Web administration server start and run in a Windows console.
- To start the hub container, select Start Hub Container. The hub container starts and runs in a Windows console.
- To start a satellite container, select Start Satellite Container *n*, where *n* is the number that identifies the satellite container. The satellite container starts and runs in a Windows console.
- To start the Web administration server, select Start Web Administration Server. The Web administration server starts and runs in a Windows console.

❖ **To stop Sybase Search from the Windows Start menu**

1 Select Programs | Sybase.

2 Select Sybase Search 3.2.

- To stop a single server installation, select Stop Single Server. The single container and Web administration server stop and the Windows console closes.
- To stop the hub container, select Stop Hub Container. The hub container stops and the Windows console closes.
- To stop a satellite container, select Stop Container *n*, where *n* is the number that identifies the satellite container. The satellite container stops and the Windows console closes.
- To stop the Web administration server, select Stop Web Administration Server. The Web administration server stops and the Windows console closes.

❖ **To start Sybase Search from a Windows command prompt**

1 Open a Windows command prompt.

2 To start a container, navigate to the *install_location**OmniQ**bin* directory and enter:

```
sysearch32.bat -start ID
```

where *ID* is the container ID of the container you want to start. The container ID of the hub container is always 1.

3 To start the Web administration server, navigate to the *install_location**Hyena**bin* directory and enter:

```
Hyena32.bat -start
```

❖ **To stop Sybase Search from a Windows command prompt**

1 Open a Windows command prompt.

2 To stop a container, navigate to the *install_location**OmniQ**bin* directory and enter:

```
sysearch32.bat -stop ID
```

where *ID* is the container ID of the container you want to start. The container ID of the hub container is always 1.

3 To stop the Web administration server, navigate to the *install_location**Hyena**bin* directory and enter:

```
Hyena32.bat -stop
```

Solaris

In Solaris, you can start and stop Sybase Search containers and the Web administration server from a command line.

Note Before starting any containers, you must set the library path environment variable LD_LIBRARY_PATH. To set this variable, navigate to the *install_location**OmniQ**bin* directory and enter:

```
. ./env.sh
```

- To start a Sybase Search container, navigate to the *install_location**OmniQ**bin* directory and enter:

  ```
  ./sysearch64.sh start ID
  ```

  where *ID* is the container ID of the container you want to start. The container ID of the hub container is always 1.

- To start the Web administration server, navigate to the *install_location**Hyena**bin* directory and enter:

  ```
  ./Hyena64.sh start
  ```

- To stop a Sybase Search container, navigate to the *install_location**OmniQ**bin* directory and enter:

  ```
  ./sysearch64.sh stop ID
  ```

  where *ID* is the container ID of the container you want to stop. The container ID of the hub container is always 1.

- To stop the Web administration server, navigate to the *install_location**Hyena**bin* directory and enter:

  ```
  ./Hyena64.sh stop
  ```

AIX and Linux

In AIX and Linux, you can start and stop Sybase Search containers and the Web administration server from a command line.

Note Before starting any containers, you must set the library path environment variable for the current profile. The library path environment variable is LIB_PATH and LD_LIBRARY_PATH for AIX and Linux, respectively. To set this variable, navigate to the *install_location**OmniQ**bin* directory and enter:

```
. ./env.sh
```

- To start a Sybase Search container, navigate to the *install_location**OmniQ**bin* directory and enter:

  ```
  ./sysearch32.sh start ID
  ```

  where *ID* is the container ID of the container you want to start. The container ID of the hub container is always 1.

- To start the Web administration server, navigate to the *install_location**Hyena**bin* directory and enter:

  ```
  ./Hyena32.sh start
  ```

- To stop a Sybase Search container, navigate to the *install_location**OmniQ**bin* directory and enter:

  ```
  ./sysearch32.sh stop ID
  ```

  where *ID* is the container ID of the container you want to stop. The container ID of the hub container is always 1.

- To stop the Web administration server, navigate to the *install_location**Hyena**bin* directory and enter:

```
./Hyena32.sh stop
```

Accessing administration pages

Sybase Search is administered through a J2EE Web application; therefore, you can administer Sybase Search from any machine that runs a Web browser. From the Sybase Search administration pages, you can view the distributed Sybase Search installation and administer it.

Note Before accessing the administration pages, make sure that the Sybase Search has been started and is running properly. For more information, see "Starting and stopping Sybase Search" on page 12.

❖ **To access the Sybase Search administration pages**

1 Open a Web browser.

2 Enter the following URL in the address bar of the Web browser:

```
http://hostname:port/omniq
```

where:

- *hostname* is the name or IP address of the machine hosting the Sybase Search Web application.
- *port* is the port number for the J2EE application server hosting the Sybase Search Web application. The default port number is 8111.

3 In the Sybase Search login page, enter the administrator password you provided during the Sybase Search installation and click Login. The Sybase Search Home page appears.

Note If you leave Sybase Search Web administration pages idle for 30 minutes, you will be logged off and prompted to log back. To change the login timeout duration, change the value of the <session-timeout> tag in the *web.xml* file available in the *install_location\webapp\WEB-INF* directory.

The Sybase Search administration pages consist of a home page and the following pages:

- Search – shows the Search page and lets you search across all of the documents that you have indexed in Sybase Search.

- System – lets you view the distributed setup. From the System page, you can view environment details, memory usage, and events for all containers within the Sybase Search installation. You can also schedule tasks.
- Document Management – lets you add, update, and remove documents from Sybase Search indexes. You can also create and manage document stores, organize document stores into groups, and create document categories.

Tracking system details

From the System page, you can track the system details of each Sybase Search container from the following pages:

- Environment – lets you view the following details about each container:
  - Host name and port on which each container runs
  - Loaded modules
  - Data, home, library, and configuration directories

  You can also view the Java system properties of each container's Java Runtime Environment (JRE) and Sybase Search license information.
- Memory Usage – lets you track the memory consumption of the Sybase Search containers, including the Java Virtual Machine (JVM) allocation and consumption. You can also track the resources loaded within the loaded modules, such as data caches.
- Events – lets you view pages of recorded events that have occurred within the distributed Sybase Search installation. Sybase Search records information, warning, and error events through a Hub Manager. A Hub Manager is always present within a participating Sybase Search container. Events can be selected by Hub Manager, filtered by severity, date, and sorted in chronological or reverse order.
- Scheduler – lets you set up specific tasks to run at configured intervals, thus automating repeatable duties, such as index updates and unifications. The Scheduler is implemented as a module and resides in the container of the Sybase Search Administrator's choice. A single Scheduler manages the scheduled tasks for an entire Sybase Search deployment. For more information, see "Scheduling tasks" on page 17.

Scheduling tasks

From the System page, you can select Scheduler to configure tasks that run at scheduled time intervals. Tasks can be added and edited, as necessary. Sybase Search displays all scheduled tasks in the Scheduled Tasks list. The Scheduled Tasks list shows when the task is scheduled to run, when it last ran, and how many times it has run.

You can set up the following Sybase Search task types:

- Log Janitor – examines the current logs and deletes old log files. Optionally, it can compress inactive log files.

  Note Deleted logs cannot be recovered. Log Janitor does not decompress logs it has already compressed.

- Document Store Runner – runs a full update on the document store at the configured interval.
- Index Unifier – unifies the document store's indexes at the configured interval. See "Unifying an index stripe" on page 34.
- Web Robot Runner – re-crawls the Web sites specified in the Web robot at specified intervals. The Web robot runner does not use the Web robot force refresh option.

❖ **To schedule tasks**

1 From the System page, select Scheduler. The Scheduled Tasks page appears.

2 Select New Scheduled Task. The New Task page appears.

3 If you want to enter a label for the task you want to schedule, clear the Auto Label check box and enter a phrase or short description in the Label field.

Note By default, the Auto Label check box is selected and a label is created depending on the task you choose to schedule.

4 From the Type list, select a task type. Sybase Search displays fields relevant to the selected task type. Table 2-1 lists the fields associated with each task type.

Table 2-1: Scheduled task types and associated fields

Task type	Fields
Log Janitor	Container – select a container ID. Keep daily logs for – select the amount of time that you want the system to keep daily logs. Compress non-active logs – specify whether you want Sybase Search to compress non-active logs.
Document Store Runner Index Unifier	Document Store – select a document store.
Web Robot Runner	Web Robot – select a Web robot.

5 In the Every field, enter how often you want the task to run and select the one of the following options:

- Never
- Minutes
- Hours
- Days
- Weeks
- Months

6 Click Create. You return to the Scheduled Tasks list. The new task is added to the list and runs at its scheduled interval.

Managing documents

You can create document stores and organize document stores into groups. You can also create document categories.

Creating, editing, and removing document stores

A **document store** is a collection of documents in Sybase Search related by physical location. You can organize documents into the following types of document stores:

- File system document stores

- Database document stores
- Web document stores

Creating document stores

The file system and database document stores acquire and maintain documents through internal processes. The Web document stores are passive, and are managed by a Web robot.

File system document stores

A **file system document store** represents one or more collections of documents imported into Sybase Search from a local file system, including mapped network drives or mounted remote file systems. The file system document store accepts one or more directory roots (for example, *D:\documents\office*), the contents of which Sybase Search indexes.

Although documents from different file systems (for example, *C:\docs* and *\\network-share\docs*) can coexist in the same document store, internally, all documents found in all root directories of a file system document store are indexed together. This means they share the same data structures, and they are updated and removed together. Sybase Search analyzes directories and subdirectories. Files with valid MIME types are then indexed. You can customize the list of valid MIME types.

❖ **To create a file system document store**

1 Click Document Management. The Document Stores Summary page appears.

2 Click File System. The File System Document Stores page appears.

3 Click Import from file system. The Create Document Store page appears.

4 Complete the following fields:

Field	Description
Name	Indicates the name of the document store.
Manager	Indicates the document store manager for which the document store should exist. A document store manager manages zero or more document stores. Typically, there is one document store manager for each server where document indexing occurs. The document store manager for each document store that you create lets you set up document indexing on the different servers in the system. See “Managing document stores” on page 35.

Field	Description
Member of	Indicates the document groups in which the document store is a member. See “Grouping document stores” on page 36.
Not Member of	Indicates the document groups of which the document store is not a member.
Index now	Indicates whether to proceed with indexing immediately or to save the configuration without indexing at this time. See “Indexing document stores” on page 31.
Directories	Indicates one or more root directory whose contents will be indexed and available for searching.
Include subdirectories	Indicates whether all subdirectories under the root directory will be indexed.
File Type Filter	Includes or excludes documents by file extension or MIME type, for example: • Include – indexes only documents of the specified file type. • Exclude – indexes all documents except those of the specified file type.

5 Click Create.

The file system document store is created. Now, the Document Stores Summary page shows the details of the document store, such as type of document store, name of the document store, and number of searchable documents.

An indexing summary is also listed, and, if the store is being indexed, the current indexing session information is displayed. For more information, see “Indexing document stores” on page 31.

Database document stores

A **database document store** represents a collection of documents imported into Sybase Search from one or more database tables. You use a SQL query to import documents from database tables into Sybase Search. See “Constructing an import SQL statement” on page 23.

All data conversions are handled internally, including files stored in binary format and links to files elsewhere on a system. Sybase Search can import data from any database for which JDBC drivers can be obtained.

Note The database document stores use Java Database Connectivity (JDBC) drivers to import data. Before creating a database document store, make sure that the appropriate JDBC driver is available in the *install_location/OmniQ/lib* directory. If it is not available, copy an appropriate JDBC driver to the *install_location/OmniQ/lib* directory and restart the container that manages the database import function. For more information about the JDBC driver's location, see your database vendor's documentation.

❖ **To create a database document store**

1 Click Document Management. The Document Stores Summary page appears.

2 Click Database. The Database Document Stores page appears.

3 Click Import from database. The Create Document Store Page appears.

4 Complete the following fields:

Field	Description
Name	Indicates the name of the document store.
Manager	Indicates the document store manager for which the document store should exist. A document store manager manages zero or more document stores. Typically, there is one document store manager for each server where document indexing occurs. The document store manager for each document store that you create lets you set up document indexing on the different servers in the system. See "Managing document stores" on page 35.
Member of	Indicates the document groups in which the document store is a member. See "Grouping document stores" on page 36.
Not Member of	Indicates the document groups of which the document store is not a member.
JDBC Connection Details	
Host	Indicates the network name or IP address of the database server.
DB Name	Indicates the name of the database.
Username	Indicates the name of the user and authenticates access to the database.

Field	Description
Password	Indicates the password used to authenticate access to the database.
Preset	Indicates the type of database and the configuration of the JDBC options. When you select a database from the Presets list, Sybase Search automatically displays the port, driver, and URL with common values for the type of database selected. The Presets list is configurable. To use a preset: 1 Complete the Name, Manager, and Member of fields for the database document store. 2 Complete the Host, DB Name, Username, Password, and Port fields for the JDBC connection details. 3 Select a preset from the Presets list. The port, driver, and URL fields display the corresponding default values. 4 Click the Translate URL placeholders link to replace the URL template placeholders with the correct values. If you do not select any database from the Presets list, you must enter appropriate values for the Driver and URL fields. **Note** Inclusion of a database driver in the Presets list does not mean the driver is available to the system. Make sure that the correct driver is available to the selected document store manager.
Port	Indicates the database server listener port. If you select a database from the Presets list, this field is populated automatically.
Driver	Indicates the full class name of the JDBC driver. If you select a database from the Presets list, this field is populated automatically.
URL	Indicates the JDBC URL to use to contact the database. If you select a database from the Presets list, this field is populated automatically.
SQL Query	Indicates the SQL statement designed to import documents from a database. See “Constructing an import SQL statement” on page 23.

Document Reference

Field	Description
Class	Identifies the document reference class, signifies the Java class type that should be used by Sybase Search internally to store the DOC_REF SQL datatype. The document reference class is automatically determined the first time data is extracted from the database, and it cannot be changed.
Length	Identifies the document reference length. The document reference length is only used for java.lang.String document reference types (the lengths of other types are implicit). In most cases, it should be the same as the VARCHAR column width from which the document references are being extracted. If the document reference is not a string, this value is ignored.
Index now	Indicates whether to proceed with indexing immediately or to save the configuration without indexing at this time. See "Indexing document stores" on page 31.

5 Click Create.

The database document store is created. Now, the Document Stores Summary page shows the details of the document store such as type of document store, name of the document store, and number of searchable documents. An indexing summary is also listed, and, if the store is being indexed, the current indexing session information is displayed. For more information about Sybase Search indexing concepts, see "Indexing document stores" on page 31.

Passive document stores

A **passive document store** represents a collection of documents imported into Sybase Search by an external process such as Web robot. The Web robot manages the download of Web content from the Internet and intranets. The Web content is sent to a passive document store, which indexes it and makes it available for searching.

For more information about creating Web robots, see "Creating, editing, and removing Web robots" on page 27.

Constructing an import SQL statement

You can construct a SQL query to retrieve content and metadata from columns in a database. Each row of data represents a document. Each document requires a unique identifier (a document reference) and content (body text). Optionally, it can have a title and other metadata.

Each database document store can have only one SQL query. A single SQL query can import one or all of your database documents into a document store, provided that the documents are all in a single database and that no authentication or authorization constraints require you to make multiple queries. For example, if you must specify more than one user name and password or more than one host, then you must construct more than one SQL query. You then require a database document store for each SQL query. Documents in separate databases require their own database document stores.

When constructing an import SQL statement, the following column names (or column aliases) have specific meaning. All are not case sensitive:

- DOC_REF – a unique token by which the document can be referred to for updates and deletions. A primary key column is most suitable for this.

  Sybase Search supports the following SQL types:

  - TINYINT
  - SMALLINT
  - INTEGER
  - BIGINT
  - REAL
  - FLOAT
  - DOUBLE
  - CHAR
  - VARCHAR – you can define the maximum length of VARCHAR.
- DOC_CONTENT – the text used as the body of the document. The text can be the content from TEXT or VARCHAR fields, or if used in conjunction with a content type value, it can be any of the supported MIME types.
- DOC_CONTENT_TYPE – the content type (or MIME type) of the document. When content is contained in the database in a binary format, DOC_CONTENT_TYPE provides the additional information required to decode it. For example, if the document content was UTF-8 encoded, plain text, then the content type is “text/plain; charset=UTF-8”. Similarly, if the content field contains PDF bytes, the content type is “application/pdf”. When the document content is binary and no content type is specified, Sybase Search attempts to decode it as plain text using the JRE default character set.

- DOC_LINK – a link to an external document on a file system. The link must be an absolute path, visible to Sybase Search. The document properties (where present) are extracted as metadata and Sybase Search uses the text as the document's body text.

Sybase Search treats all other column names and aliases as metadata and saves the information with the document as its metadata. If the metadata is to be indexed, its name and type must be known by the metadata manager. You are not required to supply metadata; however, as a best practice, you should supply a document TITLE, which is shown on the document search results page.

Example of SQL query

The following example of a SQL query shows how a recruitment agency might import their current candidate CV resumés:

```
SELECT ID
      AS DOC_REF,                    /*INT*/
      PROFILE AS DOC_CONTENT,        /*VARCHAR*/
      CV AS DOC_CONTENT_2            /*BLOB*/
      CV_MIME AS DOC_CONTENT_TYPE_2,
      FIRST_NAME + ' ' + LAST_NAME AS TITLE,
      PREF_SALARY                    /*FLOAT*/
FROM
      CANDIDATES
WHERE
      LIVE=1
```

The example shows how the primary key column ID is used as a document reference, and how the document content (body text) is composed from both VARCHAR text and document bytes in a BLOB column. In this example, the MIME type of each document is stored in the database. Also, a title is constructed from the first and last name of the candidate, and the preferred salary is saved as metadata.

Editing the attributes of document stores

You can edit most of the document store attributes. For example, you can rename a document store; add or remove document roots; add or remove File Type Filters; and move the document store in and out of document groups.

❖ **To edit a file system document store**

1 From the File System Document Stores page, select the file system document store. The Document Store Information page appears, displaying the details of the selected file system document store.

2 Click Edit. The Edit Document Store page appears. You can change the information in all the fields except Type, ID, and Manager. For more information, see “To create a file system document store” on page 19.

3 Make the changes and click Save Changes. Sybase Search saves the changes and returns you to the Document Store Information page.

❖ **To edit a database document store**

1 From the Database Document Stores page, select the database document store. The Document Store Information page appears, displaying the details of the selected database document store.

2 Click Edit. The Edit Document Store page appears.You can change the information in all the fields except Type, ID, Manager, and Length. For more information, see “To create a database document store” on page 21.

3 Make the changes and click Save Changes. Sybase Search saves the changes and returns you to the Document Store Information page.

❖ **To edit a passive document store**

1 From the Passive Document Stores page, select the passive document store. The Document Store Information page appears, displaying the details of the selected passive document store.

2 Click Edit. The Edit Document Store page appears. You can change the document groups in which the selected passive document store is a member.

3 Make the required changes and click Save Changes. Sybase Search saves the changes and returns you to the Document Store Information page.

Removing document stores

When you remove a document store, all settings and indexes are permanently removed from the disk. All documents indexed under the removed document store are no longer returned in searches.

❖ **To remove a file system document store**

1 From the File System Document Stores page, select the file system document store. The Document Store Information page appears, displaying the details of the selected file system document store.

2 Click Remove. You are prompted to confirm whether you want to remove the selected file system document store.

3 Click OK. Sybase Search removes the file system document store and returns you to the File System Document Stores page.

❖ **To remove a database document store**

1 From the Database Document Stores page, select the database document store. The Document Store Information page appears, displaying the details of the selected database document store.

2 Click Remove. You are prompted to confirm whether you want to remove the selected database document store.

3 Click OK. Sybase Search removes the database document store and returns you to the Document Store Information page.

Creating, editing, and removing Web robots

Web robots create and manage their own passive document store. The Web robot crawls the specified Web sites and saves textual content of each page locally. However, it does not save the Web content such as images, JavaScript, and style sheets.

Note Depending on the Web robot configuration and size of the Web site to index, Web robots may take a considerable time to crawl the target Web sites. Sybase recommends that you configure only one Web robot per Web site.

❖ **To create a Web robot**

1 Click Document Management. The Document Stores Summary page appears.

2 Click Web Robots. The Web Robots page appears.

3 Click Import from the Web. The Create Web Robot page appears.

4 Complete the following fields:

Field	Description
Main	
Name	Name of the Web robot.
Crawl Now	Indicates whether the Web robot should begin crawling immediately or wait until it is scheduled, or manually started later.

Field	Description
Force Refresh	Indicates whether the Web robot should discard the previously collected URL data and start a fresh crawling. When a Web robot crawls a Web site, it stores some of the HTTP response headers of each page it downloads, such as, the *status code*, *Expires*, *Last-Modified*, and *ETag* headers. This information helps to determine whether the page needs re-downloading. So, the re-crawl process becomes more efficient. The Force Refresh check box is enabled when you edit the Web robot.
Web Robot Manager	Indicates the Web Robot Manager that hosts the Web robot.
Passive Document Store Manager	Indicates the Document Store Manager to which the Web robot should send its crawled documents for indexing.
URLs	
Start URLs	Indicates the URLs the Web robot will visit first.
Link extractor patterns	Indicates that the links of the pages downloaded from URLs that match one of these patterns are extracted and put into the URL (work) queue.
Regular expressions	Indicates whether the patterns should be treated as Java 1.5 regular expressions. A regular expression pattern follows a set of syntax rules to describe or match a set of strings. For more information go to the Java API Web site at http://java.sun.com/j2se/1.5.0/docs/api/java/util/regex/package-summary.html. If this check box is not selected, patterns are treated as non-regular expressions. Non-regular expression patterns, which begin with http:// or https:// are considered as “starts with” patterns. All other non-regular expression patterns are considered as “contains string” patterns. For example: • http://www.mysite.net – extracts links from all pages. • http://www.mysite.net/public/ – only extracts links from pages in the /public directory. • /public/ – extracts all links that include “/public/” as part of their URL.
Link extractor pattern exceptions	Indicates the exceptions to the general rules specified in Link extractor patterns.
Index patterns	Indicates that the pages downloaded from URLs that match one of these patterns are indexed.

Field	Description
Index pattern exceptions	Indicates the exceptions to the general rule(s) specified in Index patterns.
User Agent	
User-Agent	Corresponds to the HTTP User-Agent request header. This value is sent with all HTTP requests.
Maximum pages to download	Indicates the maximum number of pages the Web robot will download before auto-terminating and saving what it has crawled so far.
Maximum crawl duration	Indicates the maximum length of time the Web robot will spend downloading before auto-terminating and saving what it has crawled so far. Since this amount of time may extend into days, it must be specified as an ISO 8601 Duration string.
Maximum consecutive failures	Indicates the maximum number of consecutive failures the Web robot will handle before auto-terminating and saving what it has crawled so far.
Courtesy timeout	Indicates the length of time, in seconds, the Web robot will wait between successful HTTP requests.
Error timeout	Indicates the length of time, in seconds, the Web robot will wait between unsuccessful HTTP requests. This is typically slightly longer than the courtesy timeout to allow the network and target Web server time to recover before the next attempt.
Maximum page tries	Indicates the maximum number of times the Web robot will attempt to download any Web page. Setting to a higher value enables Web robots to overcome temporary network or Web server failures.
Connect timeout	Indicates the maximum length of time, in seconds, the Web robot will wait to connect to the target Web server.
Read timeout	Indicates the maximum length of time, in seconds, the Web robot will wait on a connection to receive a response.
Authentication	
HTTP Authentication	
URL (prefix)	Indicates the prefix to the URLs that require authentication, for example, http://example.net/protected/
Realm	Indicates the name of the realm, if applicable.
Username	Indicates the username required for authentication.
Password	Indicates the password required for authentication.
Confirm password	Re-enter the password for confirmation.

Field	Description
Form Authentication	
Action	Indicates the URL, which performs the authentication. This is the URL where the HTML form is submitted.
Method	Indicates the request method, either GET or POST.
Username Form Field	
Field name	Indicates the form input field, which represents the username, for example, username, uname, or usr.
Field value	Indicates the username value, for example, jsmith.
Password Form Field	
Field name	Indicates the form input field, which represents the password for example, password, passwd, or pwd.
Field value	Indicates the password value.
Confirm password	Re-enter the password for confirmation.
Misc.	
Default page names	Indicates the pages names, which the Web robot expects will match the target Web server's welcome file list, for example, index.html, index.jsp. This enables the Web robot to guess that the following URLs are equivalent and only one version should be indexed: • http://example.net/ • http://example.net/index.html

5 Click Create to create the Web robot. The Web robot is created and the Web Robot Information page appears.

❖ **To edit a Web robot**

1 Click Document Management. The Document Stores Summary page appears.

2 Click Web Robots. The Web Robots page appears.

3 Select the Web robot you want to edit. The Web Robot Information page appears, displaying the details of the selected Web robot.

4 Click Edit. The Edit Web Robot page appears. You can change the information in all the fields except Web Robot Manager and Passive Document Store Manager.

5 Make the required changes and click Update. Sybase Search saves the changes and returns you to the Web Robot Information page.

❖ **To remove a Web robot**

1 Click Document Management. The Document Stores Summary page appears.

2 Click Web Robots. The Web Robots page appears.

3 Select the Web robot you want to remove. The Web Robot Information page appears, displaying the details of the selected Web robot.

4 Click Remove. You are prompted to confirm whether you want to remove the selected Web robot.

5 Click OK. Sybase Search removes the Web robot and the associated passive document store and returns you to the Web Robots page.

Indexing document stores

Indexing is the process of collecting data about documents contained in a document store and storing its proprietary data structures, generically called indexes. After documents in a document store are indexed, they are available for search.

An indexing session describes all data collected during the pass of a document store's indexer. Data for all documents is collected during the first indexing session; subsequent indexing sessions collect data for new documents, modified documents, and deleted documents. Thus, the amount of data collected during two different indexing sessions can vary dramatically.

When creating a document store, you can specify Sybase Search to immediately index the document store. You can also perform the following types of indexing after creating a document store in the Document Store Information page:

- Incremental Index – click to rerun the indexing process over the saved document store configuration. All new documents are indexed; all updated documents are indexed again; and all deleted documents are removed from the indexes.
- Part Index – click to define specific documents that you want to add to a document store. There are two types of part indexes:

- File System Part Index – when processing a File System Part Index, Sybase Search indexes only those documents that exist in any of the document store's document roots. If a document is already indexed, Sybase Search checks for modification and re-indexes, if necessary. If the document parameter is in the Sybase Search indexes but no longer exists on the file system, it is removed. The Part Index process does not check the directory trees for new, modified, or deleted documents, which can save a significant amount of time for large document stores.

  Note The document must exist within one of the document store's root directories. Documents that are not available in a valid root directory are ignored.

- Database Part Index – during a Database Incremental Index, the original SQL statement is re-run to find new, updated, and deleted rows. For large databases, this can be time consuming. However, this can be avoided. The Database Part Index can run SQL statements tailored to fetch only the new and updated rows or only the document references of the rows that should be removed. It can also accept a delimited list of document references to remove.

The Part Index processes are primarily for use within OEM applications.

All data collected during an indexing session is stored in the indexing session's data buffer. The data buffer is a RAM-oriented data structure, where data is aggregated, ready to be written to an index stripe. This buffer is flushed when the maximum memory threshold has been exceeded (specified in the system property omniq.index.buffer.maxMemory). The buffer shares this memory allocation with the document store's active index stripe. See "Striping index data" on page 33.

Viewing indexed details

Sybase Search displays details of indexing activity for both the previous and any current indexing session with the details of the corresponding document store on the Document Stores page.

Table 2-2 summarizes the details of indexing activity displayed for each indexed document store.

Table 2-2: Indexing activity

Property	Value
Total	The total number of documents found
Indexable	The number of documents eligible for indexing
Selected	The number of documents selected for indexing
Skipped	The number of documents purposefully ignored
Deleted	The number of documents that have been indexed but no longer exist
New	The number of new unindexed documents found
Updated	The number of updated (changed since indexing) documents found
Unchanged	The number of indexed documents that have not changed
Failed	The number of documents that should have been indexed but were not, due to a problem

❖ **To view indexing data**

- From the Document Store Information page, click Index Information. The Document Store Index Information page appears and displays the data on indexing.

Table 2-3 summarizes the data collected during an indexing session.

Table 2-3: Index information data

Property	Value
Documents Indexed	The total number of live and deleted documents in the indexes of all index stripes
Deleted Documents	The total number of documents in the indexes of all index stripes that reference deleted documents
Live Documents	The total number of documents in the indexes of all index stripes that reference live documents
Number of Stripes	The number of index stripes the indexed data is split across
Index Stripes	The details of each index stripe that the indexed data is split across

Striping index data

Index data is transferred from the data buffer and written to active or static stripes. Whether data is written to an active or a static index stripe is decided during the indexing session. The current active stripe stores all the data collected during the indexing session if it can accommodate it; otherwise, the active index stripe is emptied into a new static stripe, and all data collected during the indexing session is stored in the new static index stripe.

Active index stripes

Each document store's collection of index stripes contain exactly zero or one **active index stripe**. An active index stripe is a collection of RAM-oriented data structures—all of its data is stored in RAM while it keeps a copy on disk for persistence. An active index stripe is always writable, thus may contain data collected over numerous indexing sessions.

When an active index stripe is emptied into a static index stripe, the data files from the active index stripe are deleted and it is discarded. A new active stripe is created the next time an indexing session collects sufficiently small amount of data to fit into an active index stripe.

Static index stripes

Each document store's collection of index stripes contain zero or more **static index stripes**. A static index stripe is a collection of read-only, disk-oriented data structures.

Viewing index stripe information

Each index stripe and details of its internal data structures are listed on the Index Information page. The details include the generic term, metadata indexes, and the data structures needed to track indexed documents.

Table 2-4: Index stripe properties

Property	Value
Root	The location where the index stripe stores its data. The root property creates directories and data files in here as necessary.
Term Index Segments	The number of segments into which the term indexes are divided.
Metadata Index Segments	The number of segments into which the metadata indexes are divided.
Deleted Documents	The number of deleted documents for which this stripe still holds data (data that is purged on unification).
Live Documents	The number of live documents for which this stripe holds data.
Document lexicon	
Segments	The number of segments into which the document lexicon is divided.
Documents	The number of documents in the lexicon.
ID Range	The ID range of the document IDs (first to last).
Last Indexed	The name of the last document indexed and the time it was added.

Unifying an index stripe

Too many index stripes can eventually cause a bottleneck; therefore, you periodically should unify the stripes into a single stripe.

❖ **To unify an index stripe**

1 From the Document Stores page, select a document store and click Index Information. The Index Information page appears and displays the index stripe details.

2 Click Unify Indexes. The unification process runs. Sybase Search displays the progress of the unification process. Additionally, the unification process purges data marked for deletion and defragments the indexed data structures.

Managing document stores

Depending on the Sybase Search configuration, each container provides a document store manager for each type of document store. For example, a file system import manager contains file system document stores and lets you create file system document stores on the same container.

A document store manager manages zero or more document stores. Typically, there is one document store manager for each server where document indexing occurs. The document store manager for each document store that you create sets up document indexing on the different servers in the system.

As the administrator, you can import document stores and resize the query data cache.

Resizing the query data cache

Query data cache allows more queries to be processed faster by caching commonly requested search and metadata terms in RAM. You can resize the maximum capacity of the query data cache to best meet the requirements of your environment.

❖ **To resize the query data cache**

1 From the Document Store Manager page, select a document store manager.

2 Click Resize. The Query Data Cache Capacity page appears.

3 Enter a value in megabytes that you want to increase or decrease the query data cache to. The value you enter must be at least 1MB.

4 Click Change. The query data cache is updated, and Sybase Search returns you to the Document Store Manager page.

Grouping document stores

You can group document stores into document groups. A **document group** lets users filter search results by the document stores defined in the selected document groups.

For example, a Sybase Search environment might include three document store managers on three separate machines; each document store manager has a “CV resumé” document store. You can create a “CV resumé” document group to include all three CV resumé document stores. You can then use the document group as a search parameter to indicate that search results should only come from the “CV resumé” document group.

Table 2-5 lists the properties of the Document Groups page.

Table 2-5: Document Groups properties

Property	Description
Name	The name of the document group
ID	The unique document group ID assigned to the group
Document Stores	The document stores that are members of the group

Creating, editing, and removing document groups

You can create, edit, and remove document groups.

❖ **To create a document group**

1 Click Document Management. The Document Stores Summary page appears.

2 Click Document Groups. The Document Groups page appears.

3 Click Create. The Create Document Group page appears.

4 In the Name field, enter a name used to uniquely identify the document group.

5 Select the document stores that you want to include in the document group from the Document Store Non-Members and click Add.

6 Click Create. Sybase Search adds the new document group to the list and returns you to the Document Groups page.

❖ **To edit a document group**

1 From the Document Groups page, select the document group that you want to edit and click Edit. The Edit Document Group page appears.

2 Make the changes. You can change the name of the document group. You can also add or remove document stores from the document group. For more information, see "To create a document group" on page 36.

3 Click Save Changes. Sybase Search saves the changes and takes you to the Document Groups page.

❖ **To remove a document group**

1 From the Document Groups page, select the document group that you want to remove.

2 Click Remove. Sybase Search prompts you to confirm whether you want to remove the document group.

3 Click Yes. The document group is removed from the system.

Categorizing documents

Sybase Search lets you set up categories to help facilitate information retrieval. A **category** groups documents by content, independent of location or type of document store. You use categories to filter search results. You can also view lists of documents for each category. By setting up a well-organized category strategy, you can manage information by grouping documents of similar content.

By categorizing documents, you can create groups of documents on behalf of your users. Instead of searching for documents, a user could be presented with a pre-defined set of categories. The user can then browse the documents in each category.

Creating, editing, and removing categories

You can set up categories by defining an initial query and a relevance threshold. You can also include metadata filtering. A category must have at least one search term or at least one metadata expression.

Using the initial query and the given threshold of the document's relevance percent, Sybase Search assigns a document to the category if its relevance percent is equal to or greater than the threshold.

For example, a query that consists of search terms and a minimum document relevance creates a category of documents that are grouped by their relevance to search terms defined in the given query. The use of the document relevance helps ensure that the documents in the category are valid matches.

Another example is a category query that consists of only metadata, such as “fileType = HTML”. This creates a category that only contains HTML documents.

Users can search within a category about a certain subject, such as “England World Cup football.” Or, the user can simply use a category to filter search results, such as searching within a category of HTML documents.

Another way of categorizing documents is based on the content from one or more training documents. In this method, Sybase Search extracts the most relevant content from the training documents and uses this information as a new internal query to generate matching documents. This process is similar to the "find similar" feature except, with category training, relevant content is extracted from more than one document, ensuring that the extracted content is relevant to each of the training documents.

This method has the following benefits:

- Categories are created automatically based on example documents and without a base query.

  For example, a recruitment company wants to create a category based on a sample Java programmer’s resumé. Without the ability to create category based on category training, the company would need to produce a category "seed" query, which can vary depending on the individual who was creating the category. With training documents, Sybase Search extracts the most relevant content and creates newly training Java programmer category relevant to the example Java programmer’s resumé.

- Retrains a category that was originally created using a seed query. Categories can be trained repeatedly on new training documents to achieve the best results.

  For example, creating a category using the seed query "football team" can contain documents on English football or American football, depending on the documents that have been indexed by the system. Retraining this category on a few sample documents about American football ensures that the documents in the category are more relevant.

- Removes the need for manual tagging and continual maintenance of non-trained categories.

❖ **To create a category using a base query**

1 Click Document Management. The Document Stores Summary page appears.

2 Click Categories. The Categories page appears.

3 Click Create. The Create Category page appears.

4 In the Category Query Terms field, enter a natural-language query. The more information you provide, the more accurate your results are. For more information, see “Searching across documents” on page 96.

5 In the Not Terms field, enter terms to indicate concepts dissimilar to those for which you are searching. For more information, see “Searching across documents” on page 96.

6 Select the Details tab.

7 In the Name field, enter a text to distinguish the category from others.

8 In the Description field, enter text to further describe the category.

9 Select the Document Groups tab.

10 From the group list, select one or more document groups to restrict your search.

11 Select the Metadata tab. To include metadata in the category:

a Select a metadata parameter from the metadata list. You can add as many as five metadata parameters to the category. Click Add to add more metadata parameters.

b Select an operator. All metadata types support the equal to (=) operator. The integer and date types also support greater than or equal to(>=) and the less than or equal to (<=) operators.

c Enter a value for the metadata parameter. Table 6-1 on page 98 lists the predefined metadata parameters and types.

d If the metadata parameter contains a value that consists of more than one term, select the Within expression operator when the metadata parameter contains a value that consists of more than one term. When you set the operator to AND, every term must be present in the document metadata for the match to succeed. When you set the operator to OR, only one of the terms must be present in the document metadata for the match to succeed.

e If you have defined at least two metadata parameters, select the Across Expressions operator. When you set the operator to AND, both metadata parameters must succeed for the match to succeed. When you set the operator to OR, only one of the metadata parameters must succeed.
For more information, see, “Searching across documents” on page 96.

12 Select the Result Options tab. To set up result options:

a From the Minimum document relevance list, select a percentage. The percentage you select defines the minimum relevance ranking that a document must score for it to be included within the category. Documents with scores lower than the percentage that you enter are not included.

b Select the Score unknown terms check box to specify that terms unknown to the system—which, therefore do not exist in any indexed document—are considered by the scoring algorithm.

c Under Training Options, specify the number of results to display per page and the number of paragraphs to display for each document. Select the Term highlighting check box to highlight the query terms in the search results.

Note The fields under Training Options assist category training and have no effect on category creation. The values specified in these fields are not saved during category creation.

13 Click Create. Sybase Search creates the category, assigns a unique system-generated numeric ID to it, and automatically adds documents that match the category criteria. Sybase Search displays the new category and list of relevant documents on the View Category page.

❖ **To create a category using training documents**

1 Perform steps 1 through 12 of “To create a category using a base query” on page 39.

2 Click Run Category Query. The search results are displayed in the search results pane. Each search result contains the Add to training documents link.

3 From the search results, determine the training document that matches the information you are searching and click Add to training documents. Name or title of the specified document appears in the Training documents box. You can add up to five training documents.

4 Select the Training Documents option.

5 Click Train Category. The search result displays documents that fall into the category, sorted by relevance.

6 Click Create. Sybase Search creates the category and returns you to the Categories page.

❖ **To edit a category**

1 From the Categories page, determine the category that you want to edit.

2 Click Edit.
The Edit Category page appears.

3 Make the required changes. For more information about each category property, see "To create a category using a base query" on page 39.

4 Click Save. Sybase Search updates the category properties and returns you to the Categories page.

❖ **To remove a category**

1 From the Categories page, determine the category that you want to remove.

2 Click Remove. A message appears, asking you to verify whether to remove the category.

3 Click OK. Sybase Search removes the category and returns you to the Categories page.

Managing the category tree

The category tree allows you to view and organize categories in a tree structure. The categories are listed in the alphabetical order on the category tree. You can rearrange the category tree to add a selected category as a child category of another category.

❖ **To view the category tree**

- From the Categories page, click Tree. The Category Tree Admin page appears and displays the categories in the tree view.

❖ **To organize categories within the tree**

1 From the Categories page, click Tree. The Category Tree Admin page appears.

2 On the category tree, click the category you want to rearrange.

3 Click Move to. The Select Parent Category page appears.

4 Click the category on the category tree to which you want to add the selected category as a child.

5 Click Submit. Sybase Search adds the selected category as a child and returns you to the Category Tree Admin page.

❖ **To reset the category tree**

1 From the Category Tree Admin page, click Reset. The Sybase Search prompts you to confirm whether you want to reset.

2 Click OK. Sybase Search resets the category tree to the default view where the categories are listed alphabetically.

Viewing the contents of a document

You can view the contents of a document, open the document, and find documents that are similar to a selected document. You can view and open documents imported from a file system.

You only view and open source documents imported from a database if they have been imported as DOC_LINK references to file system documents. If a document is composed of multiple DOC_LINK file system documents, the first referenced document is returned for viewing.

Documents that have been imported from a Web site contain links to both the original URL and a cached version of the page.

Note The Web robot saves only the textual content of each indexed page. Web content such as images, JavaScript, and style sheets are not saved. Therefore, the cached Web page can appear different from the original Web page.

❖ **To view the contents of a document**

1 From the View Category page, determine which document you want to view.

2 Click View Text to open the plain text of a document in a read-only browser.

3 Click View File to open the document in its native application.

4 Click Find Similar to display documents that contain similar content.

CHAPTER 3

Configuring Sybase Search

This chapter describes the key configuration parameters for containers, the hub, and modules. It includes tips on using the configuration files, and changing the parameters.

Configuring the container XML file

Each container has an XML configuration file that determines if the container loads the hub and lists the modules to be loaded. You also use the configuration files to set system properties for the JVM in which the container runs. The hub and modules run in containers, and thus share some configuration parameters.

The XML is formed with a root container tag enclosing zero or more System Property tags, exactly one Hub tag, zero or more Module tags, and zero or one Data tag.

The format is as follows:

```
<Container id="1" port="8001">
   <SystemProperty name="exampleName"
   value="exampleValue" />
   <Hub local="true" host="127.0.0.1" port="7000"
   bindName="Hub" logEvents="true" />
   <Module id="101" class="com.omniq.xmp.ExampleModule"
   name="Example Module" />
   <Data directory="G:\example\data" />
</Container>
```

The modules can contain zero or more *HttpHandler* tags, which in turn can contain zero or more *Param* tags. For example:

```
<Module id="101" class="com.omniq.xmp.ExampleModule"
   name="Example Module">
   <HttpHandler class="com.omniq.xmp.ExampleHandler"
   resourceURI="/handler/example">
       <Param name="exampleName" value="exampleValue"/>
   </HttpHandler>
</Module>
```

For more information, see "Developing and configuring HTTP handlers" on page 83.

The hub

The hub is a special module that is the global coordinator of Sybase Search. The container that loads the hub also runs a Java Remote Method Invocation (RMI) registry to listen for remote requests. Satellite containers load a hub facade to handle communication with the real hub. All queries and administration requests are negotiated by the hub.

Configuration and ID conventions

Sybase Search example configuration files can be obtained upon request; however, the quickest way to obtain configuration files is to install the required container.

The *install_location\OmniQ\config\Container.1.xml* file contains single-server configuration.

Multiple-server configuration requires more than one file. One configuration file is for the hub container, and one configuration file is required for each container.

The files for multiple-server configuration are:

- *install_location\OmniQ\config\Container.1.xml*
- *install_location\OmniQ\config\Container.2.xml*

Containers, hub facades, and modules are not automatically assigned unique IDs (UIDs)—you must configure them manually. The UIDs must be within the range of 1 to the UID Generator's seed value, which is 10,000 by default. See "Setting Unique ID (UID) Generator parameters" on page 51.

If a container or module is assigned an ID greater than the seed value, it may conflict with an internally generated ID and cause an unexpected error later.

Because these UIDs are split across several files, it is recommended that you employ a numbering convention. The example two-server configuration files use the following conventions:

- Container ID – a value from 1– 99.
- Container XML – includes the container ID in its name, for example, `Container.1.xml`.
- HTTP listener – the container's HTTP listener binds to the port number 8000 plus the container's ID. For example, the port is 8001 for container 1 and 8002 for container 2.
- Hub container – always binds the RMI registry on port 7000.
- Hub facade ID – on satellite containers this is 100 times the container ID. For example, the hub facade ID for container 2 is 200.

  Note An exception is that the default Web application always allocates its hub facade ID as 999 as it does not need to follow the other conventions.

- Modules – each module has the ID of 100 times the container ID + *N*. For example, the first module ID on container 1 is 101, the second is 102, the third is 103 and so on.

If the Sybase Search installation requires more than 99 servers, a different convention is necessary.

Table 3-1 shows the attributes for the container tag.

Table 3-1: Container tag attributes

Attribute	Default value	Description
id	None	The unique ID of the container. This value identifies the container when it registers itself with the hub.
port	None	The TCP/IP port on which the container's embedded HTTP server listens.

Table 3-2 shows the attributes for the *SystemProperty* tag. The system properties include JVM settings and global indexing and querying parameters for modules loaded within the container.

Table 3-2: SystemProperty tag attributes

Attribute	Default value	Description
name	None	The name of the Java system property to set. In other words, the name you use within the Java process when using the java.lang.System.getProperty (java.lang.String) method.
value	None	The string value to associate with the property name.

Table 3-3 shows the attributes for the hub tag.

Table 3-3: Hub tag attributes

Attribute	Default value	Description
local	false	When set to true, the real hub is loaded into the current container. Otherwise, the container loads a hub facade.
id	None	The unique ID of the hub facade, which is used when the hub facade registers itself with the real hub. If the hub is local, this attribute is not required.
host	127.0.0.1	If the hub is not local, the hub facade uses this value to contact the real hub on the RMI registry.
port	None	The TCP/IP port on which the RMI registry started by the hub container is bound. When the hub is local, the port is used when starting the RMI registry. When the hub is not local, the port is used to connect to the RMI registry to access the real hub.
bindName	Hub	The name by which the hub is bound on the RMI registry. When the hub is local, bindName is used to bind the hub. When the hub is not local, bindName is used to look up the hub.
logEvents	false	Indicates whether the event log should be enabled. The location of the hub is irrelevant.
logDirectory	*<data.directory>\log*	The full path of the directory in which events logs should be written. If *logEvents* is false, this attribute is not required.

Table 3-4 shows the attributes for the module tag.

Table 3-4: Module tag attributes

Attribute	Default value	Description
id	None	The unique ID of the module, used to identify the module when it is registered with the hub.
name	None	The name of the module.
class	None	The name of the Java class that is the module.
enabled	true	If set to false, the module is not loaded.

Table 3-5 shows the attributes for the HttpHandler tag.

Table 3-5: HttpHandler tag attributes

Attribute	Default value	Description
class	None	The name of the Java class that is the HTTP handler (the resource).
resourceURI	None	The HTTP URI of the HTTP handler resource. This is used to complete the URL, for example, http://*container.host*:*container.port*/*resourceURI*.
name	None	The name of the parameter to pass to the HTTP handler.
value	None	The string value to associate with the parameter name.

Configuring modules

This section describes the Sybase Search modules and their configurations. Each module runs in a container and, with a few restrictions, can either be run in its own separate container on different servers, or grouped with other modules within a single container.

The available modules are:

- Setting Unique ID (UID) Generator parameters
- Setting Document Group Manager parameters
- Setting Text Manager parameters
- Setting Term Lexicon Manager parameters
- Setting Term Lexicon Manager Delegate parameters
- Setting Metadata Manager parameters
- Setting Metadata Manager Delegate parameters
- Setting Query Manager parameters
- Setting Repository Manager parameters
- Setting Filter Factory parameters
- Setting Category Manager parameters
- Setting Database Import Manager parameters

- Setting File System Import Manager parameters
- Setting Passive Import Manager parameters
- Setting Web Robot Manager parameters
- Setting Category Tree Manager parameters

Setting Unique ID (UID) Generator parameters

The Unique ID Generator settings are loaded through the *UIDGeneratorModule.default.xml* configuration file. Table 3-6 shows the parameters in this file.

Table 3-6: UIDGeneratorModule.default.xml parameters

Parameter	Default	Description
filename	uid.dat	The file that stores the next unique ID.
alwaysOpen	false	If set to true, the underlying Java class leaves the file handle open to the file name above.
seed	10,000	The UID Generator seed starts from 10,000, because numbers less than 10,000 are reserved by Sybase Search as module IDs.

Setting Document Group Manager parameters

The Document Group Manager module does not have a configuration file associated with it, because initially the system contains no predefined document groups.

Setting Text Manager parameters

The Text Manager module settings are loaded through the *TextModule.default.xml* configuration file. Table 3-7 shows the parameters in this file.

Table 3-7: TextModule.default.xml parameters

Parameter	Default	Description
min.term.length	2	The minimum term length considered for indexing. This is not taken into account in the list of preserved terms and does not apply to single-digit terms.

Parameter	Default	Description
max.term.length	20	The maximum term length considered valid for indexing. This value must match the Term Lexicon Manager parameter term.length.max.
custom.term.weight.tag.start	ctw{	Indicates the start of the custom term weighting parameter among search terms. For example, the format for a complete parameter, for the search term "Sybase" with the weighting increased 5 times, is ctw{Sybase,5}.
custom.term.weight.tag.delim	,	The delimiter used to separate search terms from the custom weight.
custom.term.weight.tag.end	}	Indicates the end of a custom term weighting parameter.
stopwords.filename	*locale/Stopwords_en.xml*	Contains a list of stopwords to remove during the indexing and querying processes to improve system performance. See "Defining the list of stopwords" on page 64.
preserved.terms.filename	*locale/PreservedTerms_en.xml*	Contains a list of preserved terms that are not stemmed during indexing. The list can also include terms less than the minimum term length defined in the *min.term.length* parameter. See "Defining the list of preserved terms" on page 64.
term.splitter.class	*com.isdduk.text.BreakIteratorSplitter*	Specifies the Java class used to break text into separate words. The default *BreakIteratorSplitter* handles all double-byte character sets.
term.stemmer.class	*com.isdduk.text.Porter2Stemmer*	Specifies the Java class used for term stemming. The default *Porter2Stemmer* is for English text.
query.augmentor.filename	*locale/QueryAugmenter_en.xml*	Contains a list of synonyms and acronyms. See "Augmenting queries" on page 65.
query.augmentor.verboseLoad	false	Indicates logging the details of cases where configurations cannot be strictly adhered to. For example, if the synonyms "transport" and "transportation" are provided, the *QueryAugmentor* creates a log stating that "transportation" will collapse to "transport," so the synonyms will not be loaded.
parsers.filename	*Parsers.xml*	The name of the file in the *config* directory that contains the list of text parsers.

You can set the term splitter and stemmer classes to language-independent classes or to language-specific classes. Language-specific stemmers allow an increase in system performance when Sybase Search is going to index documents in one language only.

Setting Term Lexicon Manager parameters

The Term Lexicon Manager settings are loaded through the *TermLexiconModule.default.xml* configuration file. Table 3-8 shows the parameters for the Term Lexicon Manager.

Table 3-8: Term Lexicon Manager parameters

Parameter	Default	Description
term.length.max	20	The maximum term length considered valid for indexing. This value must match the Text Manager parameter max.term.length. See "Setting Text Manager parameters" on page 51.
cache.capacity	131,072	The number of terms stored in memory to improve indexing and querying performance.
cache.useRootChildrenCache	true	If set to true, the underlying term lexicon data structures cache some of their structure in memory to improve indexing and querying performance.
unify.size.threshold	10,000	Determines how many terms in each term lexicon segment are stored in memory before being written to disk.
unify.idle.threshold	120,000	The time, in milliseconds, that the Term Lexicon Manager remains idle before unifying the pending terms. The idle time is restarted when a new term is added, or when an existing term is looked up.
number.of.segments	20	The number of term lexicon segments. For maximum efficiency, the value should be equal to the term.length.max.
minimization.factor	50	The branching factor of the underlying term lexicon segments. **Warning!** This parameter affects the lookup performance of the Term Lexicon Manager. Do not change this value without consulting Sybase Technical Support.

Setting Term Lexicon Manager Delegate parameters

This module is for use on the containers that do not host the Term Lexicon Manager. The Term Lexicon Manager Delegate is a cache of terms and their unique IDs. Each time the Term Lexicon Manager Delegate fetches a value from the Term Lexicon Manager, the value is cached. This reduces the amount of RMI communication between the two hosting containers.

The Term Lexicon Manager Delegate settings are loaded through the *TermLexiconModuleDelegate.default.xml* configuration file.

The only parameter in the configuration file is cache.capacity, which represents the number of terms that can be cached locally.

Setting Metadata Manager parameters

The Metadata Manager settings are loaded through the *MetadataModule.default.xml* configuration file. Table 3-9 shows the parameters in this file.

Table 3-9: MetadataModule.default.xml parameters

Parameter	Default	Description
metadata.filename	*Metadata.ser.gz*	The name of the file to where the metadata fields are serialized. It is recommended that you do not change this parameter.
uid.filename	uid.dat	The name of the file that stores the next unique ID, which is used when creating new metadata fields. It is recommended that you do not change this parameter.

Setting Metadata Manager Delegate parameters

This module is for use on the containers that do not host the Metadata Manager. The Metadata Manager Delegate is a cache of terms and their unique IDs. Each time the Metadata Manager Delegate fetches a value from the Metadata Manager, the value is cached. This reduces the amount of RMI communication between the two hosting containers.

The Metadata Manager Delegate settings are loaded through the *MetadataModuleDelegate.default.xml* configuration file.

Table 3-10: Metadata Manager Delegate parameter

Parameter	Default	Description
metadata.filename	*Metadata.ser.gz*	The name of the file to where the metadata fields are serialized. Sybase recommends that you do not change this parameter.

Setting Query Manager parameters

The Query Manager settings are loaded through the *QueryModule.default.xml* configuration file. Table 3-11 shows the Query Manager parameters.

Table 3-11: Query Manager parameters

Parameter	Default	Description
cache.termStats.capacity	131,072	The number of term statistics stored in memory to improve querying performance.
queryRunnerPool.size	20	The number of concurrent threads used to run queries.
queryParsers.filename	*QueryParsers.xml*	The name of the file in the *config* directory that contains the list of query parsers.

Setting Repository Manager parameters

The Repository Manager has no configuration settings and is used to allow other containers to pass on the text from documents located in other containers.

Setting Filter Factory parameters

The Filter Factory parameters are loaded through the *FilterFactory.default.xml* configuration file.

The list of default filters in the configuration file are:

- HTML filter
- RFC822 filter
- TXT filter

Each filter specifies a number of settings that determine which class is loaded for the filter, which paragraph extractor is used, and the MIME types to which the filter applies. Table 3-12 shows the filter setting parameters.

Table 3-12: Filter setting parameters

Parameter	Default	Description
className	None	The Java class that defines the filter.
extractorClassName	None	The Java class used for extracting paragraphs from the filtered text.
mimeTypes	None	The list of MIME types that are associated with the filter.
timeout	45,000	Indicates the time in milliseconds the filter waits while filtering a document. If the filter exceeds the given time, the filter aborts.

Parameter	Default	Description
keepTempFiles	false	If set to true, the filter keeps any temporary files produced during the filtering process.

In addition to the filter-specific settings, there are a number of general filter settings that help the extractors determine the paragraphs. The filter ensures that each paragraph is between the minimum and maximum lengths and aims for the ideal paragraph length. Table 3-13 shows the paragraph length settings.

Table 3-13: Paragraph setting parameters

Parameter	Default	Description
default.minParaLen	250	The minimum number of characters in a paragraph
default.idealParaLen	500	The ideal number of characters in a paragraph
default.maxParaLen	1,000	The maximum number of characters in a paragraph

Setting the HTML filter parameters

The HTML filter is a filter used to parse HTML files. Table 3-14 shows a list of the parameter settings used by the HTML filter.

Table 3-14: HTML filter parameters

Parameter	Value
className	com.omniq.filter.html.HTMLFilter
extractorClassName	com.omniq.filter.StandardExtractor
mimeTypes	text/html
timeout	N/A
keepTempFiles	N/A

Setting the RFC822 filter parameters

The RFC822 filter is a filter used to parse RFC822 files, a format commonly used by e-mail systems. Table 3-15 shows the parameter settings used by the RFC822 filter.

Table 3-15: RFC822 filter parameters

Parameter	Value
className	com.omniq.filter.rfc822.RFC822Filter
extractorClassName	com.omniq.filter.StandardExtractor
mimeTypes	message/rfc822
timeout	N/A
keepTempFiles	N/A

Setting the TXT filter parameters

The TXT filter is used to parse plain text files. Table 3-16 shows a list of parameter settings used by the text filter.

Table 3-16: TXT filter parameters

Parameter	Value
className	com.omniq.filter.txt.PlainTextFilter
extractorClassName	com.omniq.filter.StandardExtractor
mimeTypes	text/html
timeout	N/A
keepTempFiles	N/A

Setting Category Manager parameters

The Category Manager settings are loaded through the *CategoryModule.default.xml* configuration file. Table 3-17 shows the parameters in the configuration file.

Table 3-17: Category Manager parameters

Parameter	Default	Description
categoryRunnerPool.size	20	The number of concurrent threads used to run category queries
queryParsers.filename	QueryParsers.xml	The name of the file in the *config* directory that contains the list of query parsers

Setting Database Import Manager parameters

The Database Import Manager settings are loaded through the *DBDocumentStoreModule.default.xml* configuration file. Table 3-18 shows the parameters in the configuration file.

Table 3-18: Database Import Manager parameters

Parameter	Default	Description
cache.queryData.capacityInBytes	52428800	The maximum amount of memory allowed for Document Store Manager query data cache
database.config.filename	DBConfig.xml	The name of the database presets XML configuration file

Setting File System Import Manager parameters

The File System Import Manager settings are loaded through the *FSDocumentStoreModule.default.xml* configuration file. Table 3-19 shows the parameters in the configuration file.

Table 3-19: File System Import Manager parameters

Parameter	Default	Description
cache.queryData.capacityInBytes	52428800	The maximum amount of memory allowed for the Document Store Manager's query data cache

Setting Passive Import Manager parameters

The Passive Import Manager settings are loaded through the *EMDocumentStoreModule.default.xml* configuration file. Table 3-20 shows the parameters in the configuration file.

Table 3-20: Passive Import Manager parameters

Parameter	Default	Description
cache.queryData.capacityInBytes	52428800	The maximum amount of memory allowed for the Document Store Manager's query data cache

Setting Web Robot Manager parameters

The Web Robot Manager module does not have a configuration file associated with it, because initially the system contains no predefined Web robots.

Setting Category Tree Manager parameters

The Category Tree Manager module does not have a configuration file associated with it.

Optimizing Sybase Search

You can optimize Sybase Search if you know that all the source documents are in one language. This optimization includes using stemming algorithms, stopwords, and preserved terms.

Configure these settings using the Text Manager and Query Manager.

Processing metadata values

Parsers are used for processing metadata values, which are generally received as string key/value pairs. Although document body text is processed by the system term splitter and stemmer, metadata often must be handled differently (because metadata values can be not only strings but also numeric and date types). The parsers loaded by the Text Manager are referenced in the metadata field parser and query parser XML configuration files.

There are four types of parsers:

- String
- Numeric decimal
- Numeric integer
- Date (time)

You can build custom parsers and plug them into the system if necessary. Table 3-21 shows the attributes for the Parser tag

Table 3-21: Parser tag attributes

Attribute	Default	Description
identifier	None	The Parser instance's identifier. This must be a name and a unique ID separated by an underscore (_).
class	None	The Java implementation class.

Table 3-22 shows the attributes for the Param tag.

Table 3-22: Param tag attributes

Attribute	Default	Description
name	None	The name of the parameter to pass to the parser.
value	None	The string value to associate with the parameter name.

Sybase Search comes with the preconfigured parsers, shown in Table 3-23, which are adequate for most common metadata types.

Table 3-23: Preconfigured parsers

Item	Description
Name	float_1
Class	**com.isdduk.text.SimpleFloatParser** This class parses strings representing decimal numbers into actual decimal numbers. For example, the string "3.142" is parsed into Java float 3.142.
Name	integer_2

Item	Description
Class	**com.isdduk.text.IntegerParser** This class parses strings representing an integer number into an actual integer number; any floating-point information is discarded. For example, both “3” and “3.142” are parsed into Java int 3.
Name	dateUK_3
Class	com.isdduk.text.DateFormatParser
Name	dateMs1970_4
Class	com.isdduk.text.Ms1970DateParser
Parameter	Name – roundTo. Value – choose a year, month, day, hour, minute, second, or any other value to denote no rounding should take place. This class is date parser, which effectively parses strings representing long integer (64-bit) numbers, which themselves represent dates as the number of milliseconds since 1 January 1970. The preconfigured instance rounds dates to the nearest day (UTC).
Name	intB2KB_5
Class	**com.isdduk.text.B2KBIntParser** This class parses strings representing byte-size numbers and converts them into kilobyte-size numbers. For example, the string “2048” (bytes) is parsed as Java int 2 (kilobytes).
Name	datePDF_6
Class	**com.isdduk.text.PDFDateParser**
Parameter	Name – roundTo. Value – choose a year, month, day, hour, minute, second, or any other value to denote that no rounding should take place. This class handles the PDF date format, in which dates are formatted “D:20030602143803+01'00'”. The preconfigured instance rounds dates to the nearest day (UTC).
Name	url_7
Class	**com.isdduk.text.URLTermParser** This class splits URL strings into their constituent elements, namely, protocol, host, port, path, extension, and query. Optionally, each element can be indexed separately. The *options* parameter of the parser determines the elements that the parser returns. All elements are not indexed, by default. For example, the protocol and port elements are not indexed by default, because their values are usually the same for all URLs, which is http and 80 respectively. Thus, these values are typically not important in URL matching.

Item	Description
Class	**com.isdduk.text.IntegerParser** This class parses strings representing an integer number into an actual integer number; any floating-point information is discarded. For example, both “3” and “3.142” are parsed into Java int 3.
Name	dateUK_3
Class	com.isdduk.text.DateFormatParser
Name	dateMs1970_4
Class	com.isdduk.text.Ms1970DateParser
Parameter	Name – roundTo. Value – choose a year, month, day, hour, minute, second, or any other value to denote no rounding should take place. This class is date parser, which effectively parses strings representing long integer (64-bit) numbers, which themselves represent dates as the number of milliseconds since 1 January 1970. The preconfigured instance rounds dates to the nearest day (UTC).
Name	intB2KB_5
Class	**com.isdduk.text.B2KBIntParser** This class parses strings representing byte-size numbers and converts them into kilobyte-size numbers. For example, the string “2048” (bytes) is parsed as Java int 2 (kilobytes).
Name	datePDF_6
Class	**com.isdduk.text.PDFDateParser**
Parameter	Name – roundTo. Value – choose a year, month, day, hour, minute, second, or any other value to denote that no rounding should take place. This class handles the PDF date format, in which dates are formatted “D:20030602143803+01'00'”. The preconfigured instance rounds dates to the nearest day (UTC).
Name	url_7
Class	**com.isdduk.text.URLTermParser** This class splits URL strings into their constituent elements, namely, protocol, host, port, path, extension, and query. Optionally, each element can be indexed separately. The *options* parameter of the parser determines the elements that the parser returns. All elements are not indexed, by default. For example, the protocol and port elements are not indexed by default, because their values are usually the same for all URLs, which is http and 80 respectively. Thus, these values are typically not important in URL matching.

Item	Description
Parameter	Name – options Value – choose the value that is the sum of the bits that represent the elements the URL parser should return: • PROTOCOL – 1 • HOST – 2 • PORT – 4 • PATH – 8 • EXTENSION – 16 • QUERY – 32 For example, if the URL parser should only return the path and extension URL elements, set the *options* parameter to 24 (8+16). If this parameter is used for parsing the URL http://www.sybase.com/about/jobs.html, the parser returns "about", "jobs," and "html."

Configuring parsers to query metadata fields

The settings of query parsers are loaded through the configuration file as specified in the *queryParsers.filename* parameter in the Query Manager configuration file.

The configuration file allows the Sybase Search administrator to configure parsers for querying metadata fields. In some cases, you might want to search metadata fields using formats different from those that were indexed. For example, a date metadata field might be indexed in the YYYY-MM-DD format but searched on using a DD/MM/YYYY format. Table 3-24 shows the metadata field tag attributes.

Table 3-24: Metadata field tag attributes

Attribute	Default value	Description
name	None	The internal name of the metadata field to which this setting applies.
parser	None	The identifier of the parser used to parse the query metadata values.

Note You can change the query parser configuration at any time; however, changes take effect only after you restart the container hosting the Query Module.

Defining the list of stopwords

Stopwords are common words such as “I,” “a,” “an,” “the,” and so on, that are ignored during the indexing or querying process. Removing the most common words during the indexing process keeps index sizes smaller, which enhances performance.

You can change the list of stopwords in one of two ways:

- Edit the list of words in the default stopwords file, *Stopwords_en.xml*, available in *install_location\OmniQ\config\locale*, or
- Create a new stopwords file and configure the Text Manager to read from the new file by editing *install_location\OmniQ\config\TextModule.default.xml* and changing the value of the stopwords.filename parameter to point to the new file.

The format of *Stopwords_en.xml* is as follows:

```
<Stopwords xml:lang="en">
       <Key>the</Key>
       etc
</Stopwords>
```

Note Because the words on the stoplist are ignored when you index documents (in other words, the document is indexed as if the words on the stoplist did not exist), you must make any changes to the stoplist before you index. If you have already indexed your documents and you add new stopwords, the words are not included in your query, but the disk space consumed by that word’s associated data is not reclaimed until you index your documents again.

Removing stopwords after you have already indexed your documents has no effect until you index your documents again.

Defining the list of preserved terms

You can use preserved terms to ensure that some terms are *not* removed as part of the indexing and querying processes. For example, the term “US” would be removed from any extracted text if you entered the term “us” in the list of stopwords. The case-sensitive list of preserved terms ensures that “us” will be removed, but “US” is indexed and made available to the query calculations.

You can change the list of preserved terms using one of two methods:

- Edit the list of words in the default preserved terms file, *PreservedTerms_en.xml*, available in *install_location\OmniQ\config\locale*, or
- Create a new preserved terms file and configure the Text Manager to read from the new file by editing *install_location\OmniQ\config\TextModule.default.xml* and changing the value of the preserved.terms.filename parameter to point to the new file.

The format of *preserved_terms_en.xml* is as follows:

```
<PreservedTerms xml:lang="en">
       <Key>US</Key>
        etc
</PreservedTerms>
```

Note The preserved term list must be changed before you index any documents, because preserved terms require special handling during indexing. If you have already indexed documents, changing the preserved terms has no effect because the terms must still be queried exactly as before to produce matches (because the terms are fixed in the indexes).

Augmenting queries

In Sybase Search, the use of synonyms and acronyms is collectively called **query augmentation**. Synonyms are implemented as lists of words that are considered to have the same meaning. For example:

- Drowsy, lethargic, listless, sleepy
- Holiday, vacation

When a term featured in a list is used as a query parameter, all the other words in the list are appended to the query. For example, the query `The medicine made me drowsy`, when augmented using the previous synonym examples, becomes the following query:

`The medicine made me drowsy, lethargic, listless, sleepy`.

Acronyms are implemented as a list of keys (or a single key) with a corresponding list of values. In the following example the keys “HTML” and “HTM” have the values “Hypertext Markup Language”:

- HTML, HTM = Hypertext Markup Language

- USA, US = United States of America

Acronyms can augment a query in two ways:

- Acronym expansion – when a term featured in an acronym key list is found in a user's search terms, all the corresponding values are added to the original query. For example, the query `How to write HTML documents`, when augmented with the previous acronym examples, becomes the following query:

  ```
  How to write HTML Hypertext Markup Language documents.
  ```

- Acronym resolution – when a list of terms featured as an acronym values list is found in a user's search terms, all the corresponding keys are added to the original query. For example, the query `How to write Hypertext Markup Language documents`, when augmented with the previous acronym examples, becomes the following query:

  ```
  How to write Hypertext Markup Language HTML HTM documents.
  ```

You can change the list of synonyms and acronyms using one of two methods:

- Edit the list of words in the default synonyms and acronyms file, *QueryAugmentor_en.xml*, available in *install_location\OmniQ\config*, or
- Create a new query augmentation file and configure the Text Manager to read from the new file by editing *install_location\OmniQ\config\TextModule.default.xml* and changing the value of the query.augmentor.filename parameter to point to the new file.

The format of *QueryAugmentor_en.xml* is as follows:

```
<QueryAugmentor xml:lang="en">
        <Synonym>
                <Word>vacation</Word>
                <Word>holiday</Word>
        </Synonym>
        etc

        <Acronym>
                <Word>UK</Word>
                <Phrase>united kingdom</Phrase>
        </Acronym>
        etc
</QueryAugmentor>
```

Note Synonyms and acronyms are processed at runtime, so you can edit the augmentation list without having to index any documents again. However, you must restart Sybase Search to load the new synonym and acronym lists.

Configuring metadata fields

Sybase Search supports four types of metadata fields:

- TEXT
- DATE
- FLOAT
- INT

Each of these types supports a number of different parsers, which format or modify the metadata in different ways. For example, different DATE parsers parse the date value differently depending on the date format specified. See “Processing metadata values” on page 60.

Each metadata field is configured to be one of these four types and use one of the valid parsers.

Sybase Search cannot automatically extract new metadata fields from documents, because the metadata type and parser must be specified by the Sybase Search administrator in advance.

Setting TEXT metadata fields

TEXT metadata corresponds to any metadata field that contains words or characters. If any date or numeric information is treated as text, then each digit or number is treated simply as another character.

There are two special, internal text parsers that can be assigned to text metadata fields, namely, TEXT_STANDARD, which parses text metadata in the same manner as text content; and TEXT_FILENAME which is specialized in parsing document paths.

TEXT metadata fields do not support range searching.

Setting DATE metadata fields

DATE metadata corresponds to any metadata field that can be parsed into a date. The exact parsing of the date depends on the date parser's settings. For example, the parser may have the format DD/MM/YYYY or YY-MM-DD.

DATE metadata fields support range searching.

Setting FLOAT metadata fields

FLOAT metadata corresponds to any metadata field that can be parsed into a numeric value. The exact parsing of the numeric value depends on the parser's settings.

FLOAT metadata fields support range searching.

Setting INT metadata fields

INT metadata corresponds to any metadata field that can be parsed into an integer value. The exact parsing of the numeric value depends on the parser's settings. For example, the com.isdduk.text.IntegerParser parser simply attempts to convert the metadata to an integer, while the com.isdduk.text.B2KBIntParser parser attempts to convert the metadata to an integer and then divide the result by 1024 to get the value expressed in terms of kilobytes (KB) instead of bytes (B).

INT metadata fields support range searching.

Defining metadata fields

The list of metadata fields is loaded through the configuration file as specified in the metadata.filename parameter (*Metadata.xml*) in the Metadata Manager configuration file. Each metadata field in *Metadata.xml* is specified by the parameters shown in Table 3-25.

Table 3-25: Metadata field parameters

parameter	Default	Description
name	None	The internal name of the metadata field; one word with no spaces. This name is used in XML queries over HTTP.
displayName	None	The human-readable name for the metadata field.
type	None	The metadata field type, which can be one of TEXT, DATE, FLOAT and INT.
parser	None	The parser used to format the metadata field into the format Sybase Search will use. The parser must be defined in the *Parsers.xml* file.
indexable	false	If set to true, Sybase Search indexes any document data found for this metadata field.

Adding new metadata fields

When source documents contain metadata fields that are not listed in the default set, you can add them:

❖ **To add metadata fields to the default set**

1 Using a text editor, open the *Metadata.xml* file available in *install_location\OmniQ\config*.

2 Anywhere within the XML Metadata tag, add a new field tag specifying the following attributes:

- Name – the name of the metadata field inside the document.
- DisplayName – specifies the way the metadata field displays on the search page.
- Type – specifies whether the metadata type is TEXT, DATE, FLOAT, or INT.
- Parser – the name of the parser used to parse the metadata field for indexing. For TEXT, use one of the internal parsers (TEXT_STANDARD or TEXT_FILENAME). Otherwise, use a parser listed in the *Parsers.xml* file).
- Indexable – set this property to true to index this metadata field.

3 Save and close the file.

For example, if the new metadata field is Customer ID, the new field tag would look similar to this:

```
<Field name="custId" displayName="Customer ID" type="INT"
parser="integer_2" indexable= "true" />
```

Note You must add the new metadata field before indexing any documents. If any Sybase Search containers are running, they must be restarted.

If the metadata field requires parsing from a nonstandard string (for example, to change the customer ID portion of the string "CUST-98334" to an INT), refer to "Developing and configuring customized parsers" on page 87.

If you need to search the new metadata field in a different format from that in which it was indexed, you must configure a new query parser. For example, if INT metadata field values have been parsed from strings such as "CUST-98334" but you do not want to type the "CUST-" prefix for searching, you can add a new query parser configuration as follows:

1 Using a text editor, open *QueryParsers.xml*, which is available in the *install_location\OmniQ\config* directory.

2 Anywhere within the XML QueryParsers tag, add a new MetadataField tag, specifying the following attributes:

- Name – the internal name of the metadata.
- Parser – the name of any parser listed in the *Parsers.xml* configuration file.

3 Save and close the file.

4 Restart Sybase Search.

Using the previous example, the new MetadataField tag might look like this:

```
<MetadataField name="custId" parser="integer_2" />
```

Note You can change the query parser configuration at any time.

Configuring MIME types

The list of MIME types that Sybase Search can index is in the *MimeTypeMap.default.xml* configuration file. In the configuration file, each MIME type is assigned a type and is marked as to whether or not it is supported.

A MIME type might have several extensions, each of which may or may not be indexable. The list of MIME types allows Sybase Search to index only those document types that may contain valid text data. Common formats such as plain text and HTML are indexable by default. Executable MIME types are not are indexable by default.

You can add custom MIME types and the appropriate text filter in the *FilterFactory.default.xml* file.

Configuring modules using system parameters

Many shared module settings are configured using SystemProperty tags in the container XML configuration file. These properties are set as JVM system properties and are accessible to all classes loaded in the container. Properties set in this manner are “container-global.”

You can enter numeric parameters using several formats:

- Plain integers – for example, 20
- K – for example, 20K = 20 x 1000
- M – for example, 20M = 20 x 1000K
- KB – for example, 20KB = 20 x 1024 bytes
- MB – for example, 20MB = 20 x 1024K
- GB – for example, 20GB = 20 x 1024MB

These formats allow high values to be entered as parameters while keeping the parameters easy to read. They also avoid the “missing zero” problem that can sometimes occur when entering parameters that have a high number of trailing zeros.

Indexing processes

Sybase Search stores its data in a number of proprietary data structures, generically called indexes. For more information, see “Indexing document stores” on page 31.

Indexing involves three different processes. The first two are fundamental indexer processes and might occur numerous times during one indexing session. The third process occurs as a maintenance operation. These processes are:

- Filtering, or parsing, documents and extracting data in memory
- Writing processed data to index stripes on disk
- Unifying index stripes on disk

Extracting data into memory

The first indexing process has a threshold for restricting the amount of memory the extracted data buffer can consume before the data is written to disk (process 2). The greater the memory allocation, the more efficient the entire indexing process is, because more data can be handled at once.

Parameters that affect the extraction process are shown in Table 3-26.

Table 3-26: General upload parameters

Parameter	Default	Description
omniq.index.buffer.maxMemory	10MB	The indexing process is more efficient if many documents are indexed in a batch. The buffer's maximum memory allocation determines how many documents are processed in each batch.
omniq.indexer.maxDocumentSize	10MB	Sets the maximum document size to be indexed by Sybase Search. **Note** Very long documents have an adverse effect on the query results.

Writing data to disk

The second process is when the buffered data is written to the indexes. There are two main sets of parameters that affect this stage—the rate at which the data is written to the indexes (to reduce CPU and disk contention), and the index settings themselves. Parameters that affect the write process are shown in Table 3-27.

Table 3-27: Index parameters

Parameter	Default	Description
omniq.indexer.sleepDurationMillis	20	The time, in milliseconds, the indexer thread sleeps during indexing to allow other CPU-intensive applications to run.
omniq.indexer.sleepFrequency	20	Indicates the number of omniq.indexer.sleepFrequency cycles the indexer thread will sleep.
omniq.index.term.numSegments	5	The number of segments helps to distribute the indexed data across a number of files, reducing the seek times of large files.
omniq.index.term.minimizationFactor	20	The branching factor of each index segment. This parameter affects the lookup performance of the index segment.
omniq.index.term.useRootChildrenCache	true	If set to true, the index segments cache some of their structure in memory to improve indexing and querying performance.
omniq.index.metadata.numSegments	2	The number of segments helps to distribute the indexed data across a number of files, reducing the seek times of large files.
omniq.index.metadata.minimizationFactor	10	The branching factor of each metadata index segment. This parameter affects the lookup performance of the metadata index segment.
omniq.index.metadata.useRootChildrenCache	true	If set to true, the metadata index segments cache some of their structure in memory to improve indexing and querying performance.
omniq.lexicon.document.maxKeyLength	256	The maximum document file path length deemed valid for indexing.
omniq.lexicon.document.minimizationFactor	20	The branching factor of each document lexicon segment. This parameter affects the lookup performance of the document lexicon segment and should not be changed without consulting with Sybase Technical Support.
omniq.lexicon.document.useRootChildrenCache	true	If set to true, the document lexicon segments will cache some of their structure in RAM to improve indexing and querying performance.
omniq.lexicon.reverseDocument.numSegments	4	The number of segments helps to distribute the indexed data across a number of files, reducing the seek times of large files.

Unifying index stripes

The unifying process is for maintenance and optimization. This can take place only after a document store has been indexed again, which in turn produces new Index Stripes. For more information, see “Unifying an index stripe” on page 34.

Parameters, which affect the unifying process are shown in Table 3-28.

Table 3-28: Unifying parameters

Parameter	Default	Description
omniq.unifier.sleepDurationMillis	20	The time, in milliseconds, that the unifier thread sleeps during unifying to allow other CPU-intensive applications to run.
omniq.unifier.sleepFrequency	100	Indicates the number of *omniq.unifier.sleepFrequency* cycles the unifier thread will sleep.
omniq.unifier.termMapSizeSoftLimit	40K	The limit of the number of terms processed in each unifying batch.
omniq.unifier.termMapSizeInBytesSoftLimit	32MB	The memory limit used for processing the terms in each unifying batch.
omniq.unifier.metadataMapSizeSoftLimit	40K	The limit to the number of metadata processed in each unifying batch.
omniq.unifier.metadataMapSizeInBytesSoftLimit	32MB	The memory limit used for processing the metadata in each unifying batch.

Warning! The index and lexicon parameters are critical to how the system performs. Do not modify them without consulting with a Sybase Search support engineer.

Setting Query parameters

All queries run against Sybase Search are affected by the query parameters. The document scores can be scaled up or down using the confidence parameter, and the linking parameters affect all “find similar” queries. Table 3-29 describes the query parameters.

Table 3-29: Query parameters

Parameter	Default	Description
omniq.query.termLimit	30	The maximum number of terms in a query. If the user's query exceeds this number, Sybase Search selects the most important *omniq.query.termLimit* number of terms from the query to use as the internal query.
omniq.query.confidence	125	Sybase Search generates its own scaling factor when converting internal document relevance scores to a more user-friendly percentage score. This scaling can be influenced by the *omniq.query.confidence* and has the effect that a higher confidence lowers the overall scores, while a lower confidence raises the overall scores.
omniq.query.linking.default.minDocRel	5	The minimum document relevance for a linking query can be specified on a per-query basis, but this value is used when the minimum document relevance is not specified.
omniq.query.linking.minTerms	5	The minimum number of terms that are generated automatically by Sybase Search to be used as a linking query.
omniq.query.linking.maxTerms	10	The maximum number of terms that are generated automatically by Sybase Search to be used as a linking query.
omniq.query.linking.confidence	50	The omniq.query.linking.confidence parameter works in the same way as *omniq.query.confidence does*, except for linking queries instead of normal queries. Linking queries tend to generate lower document scores, as the generated linking query can cover many different topics. To compensate, the confidence parameter is low to raise the overall linking query scores.
omniq.query.training.maxDocs	5	The maximum number of training documents used to train a category; the minimum is one.
omniq.query.training.minTerms	5	The minimum number of terms the category query generator will use when creating a new category query. The lower the value, the more generic the generated queries become.
omniq.query.training.maxTerms	10	The maximum number of terms the category query generator will use when creating a new category query. The higher the value, the more specific the generated queries become.

Setting up metadata paragraph files

The metadata paragraph files (MPFs) are where Sybase Search stores the metadata and text of the files it has indexed. This data is used in constructing result sets and for generating plain-text versions of the indexed documents.

The MPF is a custom data structure – each contains the metadata and body text of a number of indexed documents (“Configuring MPFs” on page 76) in a compressed format. The first group of MPFs is created in the 0 (zero) directory, and subsequent groups are numbered sequentially beginning with 1.

Configuring MPFs

The MPF classes utilize a strategy to best compress all the paragraphs from documents, favoring documents of average length (where the average length is implied from the MPF configuration). Each paragraph is written to disk in one of two ways:

- The paragraph is entered into a paragraph group that is compressed as a whole and written to disk.
- The paragraph is compressed and written to disk individually.

The first technique is employed initially, as the compression scheme works better with more data; as a result the paragraphs take up less space on disk. The second technique is used when the paragraph group allocation is exhausted.

The paragraphs are not all written together, as it is often necessary to read individual paragraphs from disk (and compressing all the paragraphs together forces the application to read and decompress all paragraphs to access the sole paragraph required). The grouping provides a balance between data compression and disk I/O.

The number of paragraphs in any one paragraph group is not fixed; groups accept new paragraphs until the data buffer’s soft limit is reached. “Soft” indicates that a limit can be exceeded, but the group is then closed. The ideal scenario is when all the paragraphs from a document fit exactly within the allocated number of paragraph groups. Unused paragraph groups result in redundancy.

You can configure the paragraph grouping using the MPF parameters shown in Table 3-30. The MPF parameters are defined for all document stores in a container and are set in the main container file called *Container.<uid>.xml*.

Table 3-30: MPF parameters

Parameter	Default	Description
omniq.index.mpf.docsPerFile	20	The number of documents stored in each MPF.
omniq.index.mpf.filesPerFolder	250	The number of MPFs stored in each directory.
omniq.index.mpf.foldersPerFolder	50	The number of MPF directories stored per directory.
omniq.index.mpf.maxParagraphGroups	5	The maximum number of paragraph groups to allocate per document.
omniq.index.mpf.maxTotalGroupEntries	50	The maximum number of paragraphs from any one document that can be in a paragraph group.
omniq.index.mpf.bufferSoftLimit	8192	The ideal number of bytes an uncompressed paragraph group can consume before it is closed, compressed, and written to disk. This limit is usually slightly exceeded by design.

CHAPTER 4

Configuring Web Administration

This chapter describes the key configuration parameters for the Hyena servlet container, which is provided as a component of Sybase Search Web administration. The Hyena servlet container is a standalone lightweight HTTP server for use only with Sybase Search. You can use the Hyena servlet container, or you can integrate Sybase Search with any J2EE application server, such as Apache Tomcat.

Changing the Hyena configuration

The initial Hyena configuration takes place while installing the Web administration component of Sybase Search. You can change the configuration of the Hyena servlet container by editing the Hyena configuration file called *server.xml*, which is available in *install_location\Hyena\config* directory. Use a text editor to edit the file. Table 4-1 shows the attributes for the HTTP server tag.

Table 4-1: HTTP Server tag attributes

Attribute	Default value	Description
port	None	The TCP/IP port on which Hyena listens for connections.
host	*localhost*	The name or IP address of the host on which the Hyena servlet container resides.
stdOutput	false	All standard output (for example, printed to *java.lang.System.out* and *java.lang.System.err*) is always redirected to the Hyena log file. When set to true, the output is sent to the original standard output (usually the console) as well.

Table 4-2 shows the attributes for the Request-Handler tag.

Table 4-2: Request-Handler tag attributes

Attribute	Default value	Description
minThreads	10	The minimum number of server threads that Hyena uses to serve connections.

Attribute	Default value	Description
maxThreads	75	The maximum number of server threads that Hyena uses to serve connections.
maxIdleTime	10000	The number of milliseconds an idle server thread is kept alive before being destroyed. This parameter applies only when the current number of server threads exceeds the minimum.
debug	false	If set to true, request handling debug information is written to standard output for every HTTP connection received.

Table 4-3 shows the attributes for the Request-Parser tag.

Table 4-3: Request-Parser tag attributes

Attribute	Default value	Description
maxHeaderLength	None	The maximum number of characters accepted in any one HTTP request header (including *GET* parameters). Requests using headers longer than this are denied. Requests that send large parameter values should use the POST method.
maxNumberOfHeaders	None	The maximum number of request headers accepted as part of any single request. Requests formed using more headers than this are denied.

Table 4-4 shows the attributes for the Request-Keep-Alive tag.

Table 4-4: Request-Keep-Alive tag attributes

Attribute	Default value	Description
enabled	false	When set to true, HTTP keep-alive is used with all HTTP clients that support it.
maxRequests	None	The maximum number of requests that are served by any one connection.
timeout	None	The number of milliseconds the server waits for further requests on an open connection before breaking it.

Table 4-5 shows the attributes for the Remote-Admin tag.

Table 4-5: Remote-Admin tag attributes

Attribute	Default value	Description
enabled	false	When set to true, authorized stop and start commands sent through HTTP are accepted.
authCode	None	The authorization code required by the remote administration listener.

Table 4-6 shows the attributes for the Logging tag.

Table 4-6: Logging tag attributes

Attribute	Default value	Description
enabled	false	If set to true, HTTP requests are logged.
directory	None	Designates the directory in which log files are written.
prefix	None	The standard prefix for all log file names (appears before the date).
suffix	None	The standard suffix to use for all log file names (appears after the date).
timestamp	false	If set to true, the time of each HTTP request is written to the log.

Table 4-7 shows the attributes for the container tag.

Table 4-7: Container tag attributes

Attribute	Default value	Description
debug	false	Prints context loading debug to the standard output stream, which includes the context name and path, the Java libraries it uses, and details of each servlet that is loaded.

Table 4-8 shows the attributes for the JSP-handler tag.

Table 4-8: JSP-handler tag attributes

Attribute	Default value	Description
vigilance	0	The number of seconds that must elapse before the last modified date of the *.JSP* file is checked to determine whether it needs to be re-compiled. This is set to zero or less to represent "don't ever check." If this is set to a positive value, then you must load the Java compiler library. For more information about obtaining and loading a Java compiler library, contact Sybase Technical Support.

Table 4-9 shows the attributes for the error-template tag.

Table 4-9: Error-template tag attributes

Attribute	Default value	Description
path	*install_location**config**error_template.htm*	Defines the path to an HTML template with which error messages are formatted to display to clients when an application error is encountered.

Table 4-10 shows the attributes for the context tag.

Table 4-10: Context tag attributes

Attribute	Default value	Description
name	None	All context resource URIs implicitly start with this value; it must begin with a forward slash. For example, the context named */omniq* can have its home page at */omniq/index.html*.
path	None	The full path of the directory that contains the context.

The MIME-mapping tag

The MIME-mapping tag defines no attributes but has two other tags nested within each opening and closing pair:

- Extension – its node value represents a file extension, for example, “htm” for HTML documents.
- MIME-type – its node value represents a MIME type, for example, “text/html” for HTML documents.

Use MIME configuration when setting the content-type HTTP response header for requested files.

CHAPTER 5

Customizing Sybase Search

This chapter contains information for developing, configuring, and using custom HTTP handlers, filters, parsers, and text splitters.

Developing and configuring HTTP handlers

An HTTP handler is a Java object designed to service HTTP requests, similar to a simplified Java servlet. Sybase Search allows allocation of any number of HTTP handlers to modules at configuration time. This means you can develop and plug in custom HTTP handlers as necessary. Sybase Search ships with five HTTP handlers, four XML handlers, and a generic file serving handler (for the XML Schema Definitions).

For more information about configuring HTTP handlers, see "Configuring the container XML file" on page 45.

XML document groups HTTP handler

This handler returns a list of document groups in an XML format compliant with its XML Schema Definition (XSD), available in the *install_location/OmniQ/config/xsd/DocumentGroups.xsd* file.

It lists each document group's ID for use as a search parameter, as well as its name and the names and addresses of each of the document stores' members for display and integration purposes.

In a default installation of the Sybase Search, the XML handler and its XSD handler are available in the following directories, respectively:

- *http://install_location:<container-port>/xml/documentgroups*

- *http://install_location:<container-port>/xsd/documentgroups*

XML metadata HTTP handler

This handler returns a list of all indexable metadata Fields in an XML format compliant with its XSD, available in the *install_location/OmniQ/config/xsd/Metadata.xsd* file.

This handler lists each Metadata Field internal name (for use as a search parameter) as well as its display name and type for display and integration purposes.

In a default installation of the Sybase Search, the XML handler and its XSD handler are available in the following directories, respectively:

- *http://install_location:<container-port>/xml/metadata*
- *http://install_location:<container-port>/xsd/metadata*

XML query HTTP handler

This handler takes query parameters over HTTP (GET or POST) and returns a result set in XML compliant with the result set XSD, available in the *install_location/OmniQ/config/xsd/ResultSet.xsd* file.

In a default installation of the Sybase Search, the XML handler and its XSD handler are available in the following directories, respectively:

- *http://install_location:<container-port>/xml/query*
- *http://install_location:<container-port>/xsd/resultset*

The XML query HTTP handler parameters are shown in Table 5-1.

Table 5-1: XML query HTTP handler parameters

Parameter	Description
Normal query parameters	
terms	A natural language query string describing the concepts that all documents should contain.
notTerms	A natural language query string describing the concepts documents that should not contain.

Parameter	Description
termHi	A value used to indicate whether returned terms should be highlighted. The default is bold. The opening and closing tags are: • termHiTagOpen • termHiTagClose
Linking query parameters	
linkingDocAddr	The address of the document to use to create a `find-similar` query.
Linking query with external document parameters	
targetDSMID	The target Document Store Manager ID. The chosen document store is used by Sybase Search to obtain its initial term statistics when calculating the linking query.
linkingDocPath	The full path to the external document (from the Document Store Manager's perspective) used to create a `find-similar` query.
Common parameters	
documentGroupIds	A comma-delimited list of document group IDs. When present, only documents that are members of these groups are returned.
metadata	A metadata search expression that takes the form: < *name*><*operator*><*value*> where <*name*> is the internal name of the metadata field; <*operator*> is "=," ">=," or "<=" (the latter two are only supported by numeric and date types) and <*values*> is the criteria of the metadata search.
metadataOpWithinFields	Valid values are AND and OR. • When this parameter is AND and two or more values are presented for any one metadata field, all must match for the query to succeed. • When this parameter is OR, only one of any number of values needs to match for the query to succeed.
metadataOpAcrossFields	Valid values are AND and OR. • When this parameter is AND and two or more metadata parameters are present, all must succeed for the query to succeed. • When this parameter is OR, only one of any number of metadata parameters needs to succeed for the query to succeed.
minDocRel	The minimum relevance, expressed as a percentage, that a document must achieve to be included within a result set.
numParas	The number of document excerpts to return for each document returned in the page of results.
scoreUnknownTerms	When set to true, terms present in a query yet unknown to Sybase Search (for example, terms not present in any indexed document) are represented in the scoring algorithm. When set to false, unknown terms are ignored

Parameter	Description
resultsOffset	An integer value that represents the place in the result set the page of results should begin. The first document in the result-set (for example, the top scoring document) is at offset zero (0). If the offset is greater than the number of results found, Sybase Search returns an empty page of results.
resultsLength	An integer value that represents the number of documents to include in the page of results.
maxResultsNeeded	An integer value that represents the maximum number of results required by the caller (the minimum value is implicitly resultsOffset + resultsLength). This yields performance benefits when queries are cached for returning on a page-by-page basis.
categoryIds	A comma-delimited list of category IDs. When present, only documents that are members of the listed categories are returned.

XML document HTTP handler

This handler returns the text from an indexed document in an XML format compliant with its XSD, available in the *install_location/OmniQ/config/xsd/Document.xsd* file.

In a default installation of Sybase Search, the XML handler and its XSD handler are available in the following directories, respectively:

- *http://install_location:<container-port>/xml/document*
- *http://install_location:<container-port>/xsd/document*

The handler accepts the following parameters shown in Table 5-2.

Table 5-2: XML document HTTP handler parameters

Parameter	Description
address	The document address of the document to fetch as XML. The document address format is <DSM-ID>-<DS-ID>-<DOC-ID> (Document Store Manager ID, document store ID, and document ID).
useParagraphs	When this parameter is true, the body text of the document is broken into paragraphs and formatted between extra paragraph XML tags. If set to false, the entire body text is returned in a large, unbroken block.

XML categories HTTP handler

This handler returns a list of all categories in an XML format compliant with its XSD, available in the *install_location/OmniQ/config/xsd/Categories.xsd* file.

Each category lists the properties ID, document count, display name, and definition. Definition is the query and metadata parameters used to create it.

In a default installation of Sybase Search, the XML handler and its XSD handler are available in the following directories, respectively:

- *http://install_location:<container-port>/xml/categories*
- *http://install_location:<container-port>/xsd/categories*

Developing and configuring customized parsers

This section provides information for Java developers about developing, configuring, and using custom parsers.

Parser classes

You can create custom date, int, and float parsers and plug them into the system.

All parsers implement this common base interface:

```
com.isdduk.text.Parser
```

```
getId() : short
setId(short) : void
getName() : java.lang.String
setName(java.lang.String) : void
init(com.isdduk.util.map.FastMap): void
```

This interface defines methods that facilitate tracking and displaying information about parser instances loaded in Sybase Search, mainly simple GET and SET methods. There is also an initialization method, which takes a map of parameters if the parser require any—this method is guaranteed to be called before parsing commences. You can extend the convenience base class *com.isdduk.text.AbstractParser*.

Regardless of whether the convenience base class is used, each custom parser class must provide a no-arguments constructor and implement the appropriate one of these four specialized parser interfaces:

```
com.isdduk.text.DateParser
     parse(java.lang.String, com.isdduk.util.set.LongSet) : boolean
     parse(java.lang.String) : com.isdduk.util.set.LongSet
     format(long) : java.lang.String

com.isdduk.text.IntParser
     parse(java.lang.String, com.isdduk.util.set.IntSet) : boolean
     parse(java.lang.String) : com.isdduk.util.set.IntSet
     format(int) : java.lang.String

com.isdduk.text.FloatParser
     parse(java.lang.String, com.isdduk.util.set.FloatSet) : boolean
     parse(java.lang.String) : com.isdduk.util.set.FloatSet
     format(float) : java.lang.String

com.isdduk.text.TermParser
     parse(java.lang.String source) : com.isdduk.util.set.StringSet
     parse(java.lang.String source, com.isdduk.util.set.StringSet result :
           boolean
     format(java.lang.String term) : java.lang.String
```

The parse method, which returns a Boolean result, should contain the parsing logic; the other parse method should simply create a suitable set object and delegate the call, because it is a convenience method for when there is no suitable set object in its scope. The format method should reverse the parse process and return the date, int, or float value as a string (although this is not always possible).

Note The date parser turns date strings into long values—the number of milliseconds that have passed since the 1st of January 1970 in Coordinated Universal Time (UTC).

Adding the new parser

After you have compiled the new parser class, you must make it available to the system. The easiest way to do this is by adding the class to a Java Archive (JAR) file and placing the JAR file in the *install_location\OmniQ\lib* directories.

Note You must place the JAR file into every container's library directory.

You must add the new parser to the internal set of parsers. Edit the *install_location\OmniQ\config\Parsers.xml* configuration file and add a new Parser tag.

For example, a parser that parses user IDs from strings might have a configuration like this:

```
<Parser identifier="intUserId_8" class="com.mycompany.IntParserImpl">
     <Param name="base" value="16" />
</Parser>
```

You can load multiple instances of the same class. This example assumes the parser class can parse different integer bases (such as octal, decimal, hexadecimal, and so on), but the configured instance expects the hexadecimal format.

Note Because Sybase Search is a distributed system, it is important to configure the new parsers for the container instance that loads the Text Manager.

Using the new parser for metadata indexing

You must configure each metadata field that requires the new parser. To do this:

1 Using a text editor, open the *Metadata.xml* configuration file available in *install_location**OmniQ**config*\.

2 Locate the appropriate fields and change the parser attribute to the value of the new parser identifier. For example:

```
<Field name="userid" displayName="User ID" type="INT"
parser="intUserId_8" indexable="true" />
```

If it is a new metadata field and does not exist in this file, add it.

Note Because Sybase Search is a distributed system, it is important to perform metadata changes for the container instance that loads the Metadata Manager.

Using the new parser for querying

If the metadata field requires metadata query values to be handled by the new parser, it is necessary to override the default behavior by editing the *QueryParsers.xml* file available in *install_location**OmniQ**config*. If you assume the querying format is the same as it is for indexing, then the query parser would have the following entry:

```
<MetadataField name="userid" parser="intUserId_8" />
```

Note Because Sybase Search is a distributed system, it is important that the metadata changes are performed for the container instance that loads the Query Manager.

Developing and configuring customized text splitters

This section provides information for Java developers about developing, configuring, and using custom text splitters.

All values for document body text and textual metadata (excluding file paths) are passed through the configured term splitter to be broken into individual terms. Each term that is not preserved, not a stopword, and is neither too short nor too long, is passed to the configured term stemmer to be reduced to its root form. Both the term splitter and term stemmer can be reimplemented and reconfigured where necessary. For more information on preserved terms and stopwords, see:

- "Defining the list of preserved terms" on page 64
- "Defining the list of stopwords" on page 64

Term splitting turns extracted plain text into words. Term stemming reduces words to their common roots. Term splitting and term stemming are language-specific; therefore, for optimum performance, when you know documents and searches are to be performed in a single language, you can customize the term splitter and term stemmer algorithm to make best use of the language.

For example, an English stemming algorithm converts "singing," "sings," and "singer" to the stem "sing"; however, this algorithm is not appropriate for French or Chinese.

The default configuration splitter class com.isdduk.text.BreakIteratorSplitter handles all double-byte characters by using the underlying default Java class java.text.BreakIterator. The Java *BreakIterator* class uses punctuation and word delimiters to split single-byte languages into words. For double-byte languages, however, the Java *BreakIterator* class samples the glyphs in pairs and tries to determine where the end of the words are likely to be.

If you intend to run Sybase Search with documents containing glyph-based languages, Sybase recommends that you write your own custom term splitter (described in "Configuring the term splitter"). Term splitting algorithms designed for a single language out-perform the Java *BreakIterator*, which is designed to handle multiple languages, particularly glyph-based languages.

Configuring the term splitter

The term splitter interface defines numerous methods, many of which must be the same, regardless of the splitting algorithm used. To simplify implementing new term splitters, Sybase Search includes an abstract base class that you can extend to inherit much of the required functionality:

```
com.isdduk.text.AbstractTermSplitter
```

The convenience base class does not implement any splitting algorithms. The various split methods defined by the term splitter interface are as follows (see the Javadocs for the full interface method listing):

```
com.isdduk.text.TermSplitter
     split(java.lang.String source) : com.isdduk.util.set.StringSet
     split(java.lang.String source, boolean validate) :
          com.isdduk.text.StringList
     splitWords(java.lang.CharSequence source) :
          java.util.SortedSet<com.isdduk.text.TermSplitter.WordIndex>
     splitWords(java.lang.CharSequence source, int offset, int length) :
          java.util.SortedSet<com.isdduk.text.TermSplitter.WordIndex>
     splitFrequencies(java.lang.CharSequence source,
          com.isdduk.util.map.FastTermMap insertInto) : void
```

Configuring the term stemmer

The term stemmer interface is much simpler than its splitter counterpart. It defines only three methods:

```
com.isdduk.text.TermStemmer
     stem(com.isdduk.text.Term term) : com.isdduk.text.Term
     hasNormalize() : boolean
     normalize(com.isdduk.text.Term term) : com.isdduk.text.Term
```

The stem method takes a term argument and returns a stemmed version of it, which is in many cases the same object, although perhaps with a different length. The normalize method caters to terms that are not sent through the stem method (which should incorporate normalization as part of its routine)—it ensures the term conforms to a single standard of representation (for example, a German stemmer may normalize the sharp S “ß” to its equivalent “ss” or vice versa). Terms may bypass the stem method occasionally, when their lengths exceed the maximum allowed (and are therefore “force stemmed” to fit).

Replacing the system text and term splitters

After you have compiled the new term splitter class, you must make it available to the system. The easiest way to do this is by adding the class to a JAR file and placing the JAR file in the *install_location\OmniQ\lib* directories.

Note Because Sybase Search is a distributed system, you must place the JAR file into every container's library directory.

The module responsible for loading the term splitter and stemmer is the Text Manager module. Edit the *TextModule.default.xml* configuration file available in *install_location\OmniQ\config* and change the *term.splitter.class* property and the *term.stemmer.class* property as necessary.

Note You must perform the metadata changes for the container instance that loads the Text Manager.

CHAPTER 6

Using Sybase Search

This chapter describes how to access Sybase Search and search across documents.

Accessing Sybase Search

You can access the Sybase Search from any computer that runs a Web browser.

Note The following procedure describes how to access Sybase Search for searching across documents. To perform Sybase Search administrative tasks, you must log in to the Sybase Search administration pages. See "Accessing administration pages" on page 15.

❖ **To access Sybase Search**

1 Open a Web browser.

2 Enter the following URL in the address bar of the Web browser:

`http://`*`hostname`*`:`*`port`*`/omniq/search/`

where:

- *hostname* is the name or IP address of the machine hosting the Sybase Search Web application
- *port* is the port number for the J2EE application server hosting the Sybase Search Web application. The default port number is 8111.

The Search page displays.

Searching across documents

From the Search page, you can search across all of the documents that have been indexed in Sybase Search. Various options are provided to help you get accurate search results.

To search across documents:

- From the Search page, enter appropriate values for the following fields and click Search:
  - **Specifying search terms** – you can specify the terms that are to be included or excluded from the search criteria.
    - Search Terms – enter a natural-language query in the Search Terms field. The more information you provide, the more accurate your results are. For more information, see "Optimizing search strategies" on page 4.
    - Not Terms – enter terms to indicate concepts dissimilar to those for which you are searching. Unlike the Boolean NOT operator, documents that contain the Not Terms are considered for retrieval. However, the number of Not Terms a document contains is considered by the scoring algorithm, and its relevance score is downgraded accordingly based on the weight of the Not Terms it contains.

      For example, a search for "operating systems" with Not Terms "Windows XP" would not discount a document for containing the phrase "opens in a new window."
  - **Selecting categories** – categories are a set of documents grouped by content, independent of location or type of document store. You can use categories to filter search results. You can also view lists of documents for each category. For more information, see "Categorizing documents" on page 37.
  - **Selecting document groups** – limit your search to one or more predefined document groups by selecting specific groups from the Document Groups list. Only documents from the chosen document groups are included in the search results.

- **Specifying metadata** – select from a list of predefined metadata parameters to include metadata in the search. Sybase Search supports text, integer, and date metadata types.

  Note Some metadata parameters are document-specific. For example, a Microsoft Word document can have a Word Count, whereas a plain text document cannot and an HTML document most likely does not. Metadata parameters that are guaranteed to be searchable for all documents are described as being reliable. When the parameter searched on is not supported or not present in a document, it is automatically excluded from the results.

  You select an operator for each metadata parameter. All metadata types support the equal to (=) operator. The integer and date types also support greater than or equal to(>=) and the less than or equal to (<=) operators.

  You also enter a value for each metadata field that you define. The values for text types are processed as search text. In other words, search terms are stemmed and augmented through synonyms and acronyms (except file path fields). The numeric type values are numbers and the date type values should be in the format configured by the Sybase Search system administrator, for example, `dd/mm/yyyy`. Fore more details, see “Configuring parsers to query metadata fields” on page 63. Table 6-1 on page 98 lists the predefined metadata parameters and types.

Table 6-1: Predefined metadata parameters

Name	Type	Reliable
Author	TEXT	No
Character Count	INT	No
Client	TEXT	No
Comment	TEXT	No
Company	TEXT	No
Creation Date	TEXT	No
Description	TEXT	No
Document Name	TEXT	Yes
Document Path	TEXT	Yes
Document Size (KB)	INT	Yes
Document Type	TEXT	No
Editor	TEXT	No
File Type	TEXT	Yes
Keywords	TEXT	No
Language	TEXT	No
Last Modified	DATE	Yes
Page Count	INT	No
Project	TEXT	No
Publisher	TEXT	No
Reference	TEXT	No
Second Author	TEXT	No
Status	TEXT	No
Subject	TEXT	No
Title	TEXT	No
URL	TEXT	No
Word Count	INT	No

Metadata Combination Operators – you can select two combination operators:

- Within Expression – use when there is at least one metadata parameter with a value that consists of more than one term. When you set the operator to AND, every term must be present in the document metadata for the match to succeed. When you set the operator to OR, only one of the terms must be present in the document metadata for the match to succeed.

For example, when the metadata parameter is `Author = "John Smith"`, the Within Expression operator differentiates the two possible interpretations, which are `Author = "John AND Smith"` or `Author = "John OR Smith"`.

Note Sybase Search supports only one Within Expression operator, so you cannot perform a metadata search for `Author = "John AND (Smith OR Roberts)"`. However, Sybase Search processes each Equals expression individually; therefore, you can achieve the same effect by using two separate expressions and using the OR Within Expression operator and the AND Across Expression operator. For example, `Author = "John" AND Author = "Smith, Roberts"` returns documents only authored by John Smith or John Roberts.

- Across Expressions – use when you have defined at least two metadata parameters. When you set the operator to AND, both metadata parameters must succeed for the match to succeed. When you set the operator to OR, only one of the metadata parameters must succeed.

  For example, when the metadata parameters are `Author = "Smith," Title = "Algebra,"` the Across Expressions operator differentiates the two possible interpretations as:

  - `Author = "Smith" AND Title = "Algebra"`
  - `Author = "Smith" OR Title = "Algebra"`

Note Sybase Search supports only one Across Expressions operator, so you cannot perform a metadata search for multiple Across Expressions operators.

The predefined metadata parameters are primarily to be used in conjunction with the Search Term and Not Term search criteria to refine the search, but you can also use them independently for a metadata search. Search results from a pure metadata search have no meaningful relevance scores.

- **Setting result options** – you can further refine search results by defining values for the following options from the Result Options tab:
  - Minimum Document Relevance – define the minimum relevance ranking that a document must score for it to be included within the search results. Documents with scores lower than the percentage that you enter are not returned.

- Number of Results – define the number of document results to display for each page by selecting a value from the Number of Results list.
- Number of Paragraphs – define the number of document paragraphs to display for each result document by selecting a value from the Number of Paragraphs list.
- Score Unknown Terms – include unknown terms by selecting the Score Unknown Terms check box. When selected, terms unknown to the system (and therefore, do not exist in any indexed document) are considered by the scoring algorithm.
- Term Highlighting – specify whether to highlight terms in the search results by selecting the Term Highlighting check box. When selected, terms from the query are highlighted in the result paragraphs and in the plain-text versions of the matching documents, as shown by the view text links.

APPENDIX A

Generated Files

Each module contains its own directory where it stores files. These can be serialized Java object files or proprietary data structures. The format of each directory is the short name of the module followed by its unique module ID.

Module files

Table A-1: Module generated file locations

Module	File location
Document Group Manager	*install_location*\OmniQ\data\DGM-<*uid*> where <*uid*> is its unique module ID.
Filter factory	*install_location*\OmniQ\data\FilterFactory-<*uid*>
Metadata Manager	*install_location*\OmniQ\data\MetadataModule-<*uid*> where <*uid*> is its unique module ID.
Metadata Manager Delegate	*install_location*\OmniQ\data\MetadataModuleDelegate-<*uid*> where <*uid*> is its unique ID.
Query Manager	*install_location*\OmniQ\data\QueryModule-<*uid*> where <*uid*> is its unique ID.
Repository Module	*install_location*\OmniQ\data\RepositoryModule-<*uid*> where <*uid*> is its unique ID.
Term Lexicon Manager	*install_location*\OmniQ\data\TermLexiconModule-<*uid*> where <*uid*> is its unique ID.
Term Lexicon Manager Delegate	*install_location*\OmniQ\data\TermLexiconModuleDelegate-<*uid*> where <*uid*> is its unique ID.
Text Manager	*install_location*\OmniQ\data\TextModule-<*uid*> where <*uid*> is its unique ID.
Unique ID generator	*install_location*\OmniQ\data\UID-<*uid*> where <*uid*> is its unique ID.

Module	File location
Document Store Manager	*install_location*\OmniQ\data\DSM-<*uid*> where <*uid*> is its unique ID.
Document Stores	*install_location*\OmniQ\data\DSM-<*uid1*>\DS-<*uid2*> where <*uid1*> is the unique ID of the Document Store Manager, and <*uid2*> is the ID of the Document Store.
Web Robot Manager	*install_location*\OmniQ\data\WRM-<*uid*> where <*uid*> is the unique ID of the Web Robot Manager.

APPENDIX B

Sybase Search Content Adapter

This appendix describes how to install and configure the Sybase Search Content Adapter for use with Sybase Search component of Sybase® Data Integration Suite.

Introduction

Sybase Search Content Adapter includes the third-party Stellent document filter that enables searching across proprietary file formats such as Microsoft Office documents and Adobe PDF.

Earlier version of Sybase Search, which was part of Sybase Data Integration Suite (DI Suite) 1.0, included Stellent document filters in the same orderable. In DI Suite 1.1, Stellent filters and Sybase Search are packaged separately.

You must purchase the Sybase Search Content Adapter add-on option and install it separately. The installation procedures are described in “Installation” on page 105.

Note If you are using earlier versions of Sybase Search, you must obtain the license for Sybase Search Content Adapter add-on option to use it with Sybase Search 3.2. For more information about obtaining the license, see “License information” on page 104.

For more information about Sybase Search Content Adapter, see the *Sybase Search 3.2 New Features* guide.

License information

Sybase Search Content Adapter uses the Sybase Software Asset Management (SySAM) licensing mechanism for license administration and asset management tasks. After you have purchased the Sybase Search Content Adapter add-on option, go to the SPDC Web site at http://sybase.subscribenet.com to generate and download the license. For more information about SySAM, see the Sybase Software Asset Management User's Guide.

Sybase Search Content Adapter supports the same license models as the DI Suite Sybase Search. Therefore, to install Content Adapter you must generate the license similar to that generated for Sybase Search. For example, if Sybase Search uses the unserved license model, then you must generate unserved license for Content Adapter.

For more information on license models that DI Suite supports, see Chapter 1, "Before You Begin," in the *Sybase Data Integration Suite 1.1 Installation Guide*.

License installation steps

The license installation steps are different for served and unserved license model.

❖ **To install using a served license model**

1 Copy the license file to the *licenses* directory on the network license server machine.

2 Refresh or restart the license server.

❖ **To install using an unserved license model**

- Copy the license file to the *install_location\SYSAM-2_0\licenses* directory.

Note By default, the SYSAM *licenses* directory is located in the *install_location\SYSAM-2_0* directory. If the container on which you want to install the Content Adapter is configured with a different SySAM *licenses* directory, you can find the directory location by looking up the *sysam.license.dir* system property in the *install_location/Search-3_2/OmniQ/config/Container.ID.xml* file, where *ID* is the container ID of the container on which you want to install the Content Adapter.

Installation

Installation directory

The Sybase Search Content Adapter installer installs the Content Adapter files in the *Search-3_2* directory.

Installation mode

You can install Sybase Search Content Adapter in one of the following modes:

- GUI mode – the Sybase Search Content Adapter is installed through a graphical user interface. This is the default installation mode and Sybase recommends you to install the Sybase Search Content Adapter in this mode. For more information, see “Installing in GUI mode” on page 105.
- Console mode – the Sybase Search Content Adapter is installed through a command line interface. For more information, see “Installing in console mode” on page 106.

Pre-installation tasks

Before you install Sybase Search Content Adapter:

- Make sure that Sybase Search is installed and the *Search-3_2* directory exists on the installation machine.
- Make sure that you have the necessary SySAM licenses available in your installation environment. See “License installation steps” on page 104.
- Make sure that the container on which you want to install Content Adapter is stopped. For information about stopping the container, see “Starting and stopping Sybase Search” on page 12.

Installing in GUI mode

This section describes installing Sybase Search Content Adapter in GUI mode.

❖ **To install Sybase Search Content Adapter in GUI mode**

1 Insert the Sybase Search Content Adapter installation media.

- On Windows, the setup program should start automatically. If it does not, start the program manually by selecting Start | Run from the Windows Start menu. Browse to *sysearch_ca.exe*.
- On AIX, Solaris, and Linux, go to the directory where *sysearch_ca.bin* is available and enter the following at the command prompt:

```
./sysearch_ca.bin
```

2 The Welcome window appears. Click Next to proceed.

3 The Readme window appears. It displays information about the platforms supported by Sybase Search Content Adapter, related documentation, and technical support. Read the information, and click Next.

4 The License Agreement window appears. Select I accept the terms of the license agreement option and click Next.

5 The Installation Directory window appears, displaying the default installation directory. Verify the installation directory and click Next.

6 The Installation Summary window appears, displaying the installation directory you specified and the size of the installation files. Review the information and click Install. The installer proceeds with the installation and displays the installation progress.

7 The Install Summary window appears, indicating whether the installation was successful or not. Click Finish to exit

Installing in console mode

In cases where no graphics display device is available, or if you want to run the installer without the GUI, you can run the installer in console mode. The flow of the installation is identical to a regular GUI installation, except that the display is written to a terminal window and responses are entered using the keyboard. For more information, see “Installing in GUI mode” on page 105.

❖ **To install the Content Adapter in console mode**

1 On Windows, run the installer with the -console command line argument:

```
sysearch_ca.exe -console
```

2 On AIX, Solaris, and Linux, run the installer with the -console command line argument:

```
./sysearch_ca.bin -console
```

Silent installation

A silent installation (sometimes referred to as an unattended installation) is done by running the installer and providing a response file that contains answers to all of the installer questions. The response file is a text file that you can edit to change any responses before using it in any subsequent installations.

There are two ways to create a response file:

- Record mode
- Template mode

Creating a response file using record mode

In this mode, the installer performs the installation and records all your responses and selections in the specified response file. You must complete the installation to generate a response file.

To create a response file, enter the following command, where *respFileName* is the absolute path of the file name you choose for the response file:

- On Windows:

```
sysearch_ca.exe -options-record respFileName
```

- On AIX, Solaris, and Linux:

```
./sysearch_ca.bin -options-record respFileName
```

You can also use the console mode to generate a response file without using the GUI. To create a response file in console mode, enter the following command, where *respFileName* is the absolute path of the file name you choose for the response file:

- On Windows:

```
sysearch_ca.exe -is:javaconsole -console -options-record respFileName
```

- On AIX, Solaris, and Linux:

```
./sysearch_ca.bin -is:javaconsole -console -options-record respFileName
```

The following are the results:

- An installation of Content Adapter
- A response file containing all of your responses from the installation

If you use this response file for a silent installation, the resulting installation is identical to the one from which the response file was created: the same installation location, same feature selection, and all the same remaining information.

Creating a response file using template mode

In this mode, installer creates a response file containing commented-out values for all required responses and selections. However, you need not install the Content Adapter, and you can cancel the installation after the response file has been created.

To create this template file, enter the following command, where *respFileName* is the absolute path of the file name you choose for the response file:

- On Windows:

```
syssearch_ca.exe -is:javaconsole -options-template respFileName
```

- On AIX, Solaris, and Linux:

```
./syssearch_ca.bin -is:javaconsole -options-template respFileName
```

If run in console mode, as shown in the previous example, the installer displays a message indicating that the template creation was successful. If run in GUI mode, the installer does not display the message.

If you use this response file for a silent installation, the default values for all responses are used. Edit the template with the values you want to use during installation.

Installing interactively using a response file

An interactive installation using a response file allows you to accept the default values from the response file, or to change any of those values for the specific installation. This is useful when you have multiple similar installations that have minor differences that you want to change at installation time.

Enter the following at the command line, where *respFileName* is the absolute path of the response file.

- On Windows:

```
syssearch_ca.exe -options respFileName
```

- On AIX, Solaris, and Linux:

```
./syssearch_ca.bin -options respFileName
```

Installing in silent mode

A silent mode installation allows you to install the Content Adapter with all responses being taken from the response file that you have set up. There is no user interaction. This is useful when you want multiple identical installations, or you want to automate the installation process.

Enter the following at the command line, where *respFileName* is the absolute path of the response file. The -W option specifies that you agree with the Sybase License Agreement text.

- On Windows:

```
syssearch_ca.exe -is:javaconsole -silent -options respFileName -W
SybaseLicense.agreeToLicense=true
```

- On AIX, Solaris, and Linux:

```
./syssearch_ca.bin -is:javaconsole -silent -options respFileName -W
SybaseLicense.agreeToLicense=true
```

Except for the absence of the GUI screens, all actions of the installer are the same, and the result of an installation in silent mode is exactly the same as one performed in GUI mode with the same responses.

Post-installation tasks

After installing the Content Adapter, you must restart the container on which Content Adapter was installed. For information about starting the container, see “Starting and stopping Sybase Search” on page 12.

Testing the installation

To test the installation and working of Sybase Search Content Adapter:

- Verify that a directory named *sx* is created under the *installlocation\search-3_2* directory and that this directory is not empty.
- Verify that the Content Adapter license is applied correctly to the container:
  a Log in to Web administration pages. For more information, see “Accessing administration pages” on page 15.
  b Select System | Environment. The Environment page appears. Depending on the installation type, it displays one or more containers and the corresponding details.
  c Click License Information on the container on which Content Adapter is installed. The License Information page appears.

d Review the license information. If the Content Adapter license is applied correctly, the Content Adapter license information must appear in a separate table for the container to which you applied this license. Also, in this table, the feature name should appear as SYSEARCH_CONTENT_ADAPTER.

- Verify that you can create and index a document store that contains Microsoft Word or Adobe PDF documents. For more information, see:
  - "Creating, editing, and removing document stores" on page 18
  - "Indexing document stores" on page 31

Uninstallation

Before uninstalling Sybase Search Content Adapter:

- Log out of Sybase Search Web administration pages.
- Stop the container on which Content Adapter is installed. For more information about stopping the container, see "Starting and stopping Sybase Search" on page 12.

Uninstalling in GUI mode

❖ **To uninstall in GUI mode**

1 Invoke the uninstall program:

- On Windows, select Start | Settings | Control Panel | Add or Remove Programs. When the Add or Remove Program window appears, select Sybase Search Content Adapter 1.1 and click Change/Remove.
- On AIX, Solaris, and Linux, enter the following at the command prompt:

```
$SYBASE/Search-3_2/sx/_uninst/uninstaller.bin
```

2 The Welcome window appears. Click Next to continue.

3 The Uninstall window appears, displaying the following features that you can uninstall:

- Common Container Config – replaces the FilterFactory and MimeTypeMap settings with the base Sybase Search configuration settings, which were stored in a backup file created during the Content Adapter installation.
- Content Adapter – uninstalls the Sybase Search Content Adapter.

Select the feature you want to uninstall and click Next.

4 The Uninstallation Summary window appears. Review the information and click Uninstall. The Uninstall Progress window appears, indicating that the uninstallation process is being performed.

5 When the uninstallation is completed the Uninstall Summary window appears. Click Finish to exit.

Uninstalling in console mode

❖ **To uninstall in console mode**

1 Invoke the uninstaller:

- On Windows, change directory to *install_location**sx**_uninst*\ and enter the following command:

```
uninstaller.exe -console
```

- On AIX, Solaris, and Linux, change directory to *install_location**sx**_uninst*\ and enter the following command:

```
./uninstaller.bin -console
```

2 Select the feature you want to uninstall. The selected feature is uninstalled.

Configuring Sybase Search Content Adapter

Sybase Search uses the Content Adapter, which includes Stellent document filters for parsing many document formats. The Stellent document filter is a multifilter—in other words, the same filter instance handles all supported MIME types. Thus, the Stellent filter is configured to handle the MIME type */*, indicating that it can filter text from documents of any MIME type presented to it.

When Sybase Search obtains a filter for a document, it first identifies its MIME type from the file extension. For example, *C:\document.pdf* has the MIME type "application" and the subtype "pdf" (application/pdf). Sybase Search then requests a filter from the Filter Factory to handle documents with the identified MIME type.

The filter look-up is performed in this order:

1 If a filter is configured to handle a specific MIME type, that filter instance is returned.

2 If a multifilter (*/*) is configured, that filter instance is returned.

3 No filter is returned, denoting "not indexable."

Setting Filter Factory parameters for Content Adapter

The Filter Factory parameters are loaded through the *FilterFactory.default.xml* configuration file.

For more information, see "Setting Filter Factory parameters" on page 55.

The Sybase Search Content Adapter related filters in the configuration file are:

- SearchML export multifilter – the default filter used to parse all other MIME types that do not have their own specific filter. The output from the SearchML export multifilter is an XML document that contains the raw text and associated document metadata.
- SearchML filter – an internal filter used to parse the XML output from the SearchML export multifilter as given above.

Table B-1 shows the filter factory settings for the Content Adapter.

Table B-1: Filter Factory parameters for Content Adapter

Parameter	Value
General settings	
timeout	45,000
keepTempFiles	false
SearchML export multi-filter settings	
className	com.omniq.filter.stellent.SearchMLFilterExport
extractorClassName	com.omniq.filter.StandardExtractor
mimeTypes	*
timeout	45000
keepTempFiles	false

Parameter	Value
SearchML parameters	
className	com.omniq.filter.stellent.SearchMLFilter
extractorClassName	com.omniq.filter.StandardExtractor
mimeTypes	text/x-searchml
timeout	N/A
keepTempFiles	N/A

Index

N

O

P

Q

R

S

T

U

W

X