flexible search™

Data Source Reference Manual

(Click on a link to access the desired section)

Version 2.0

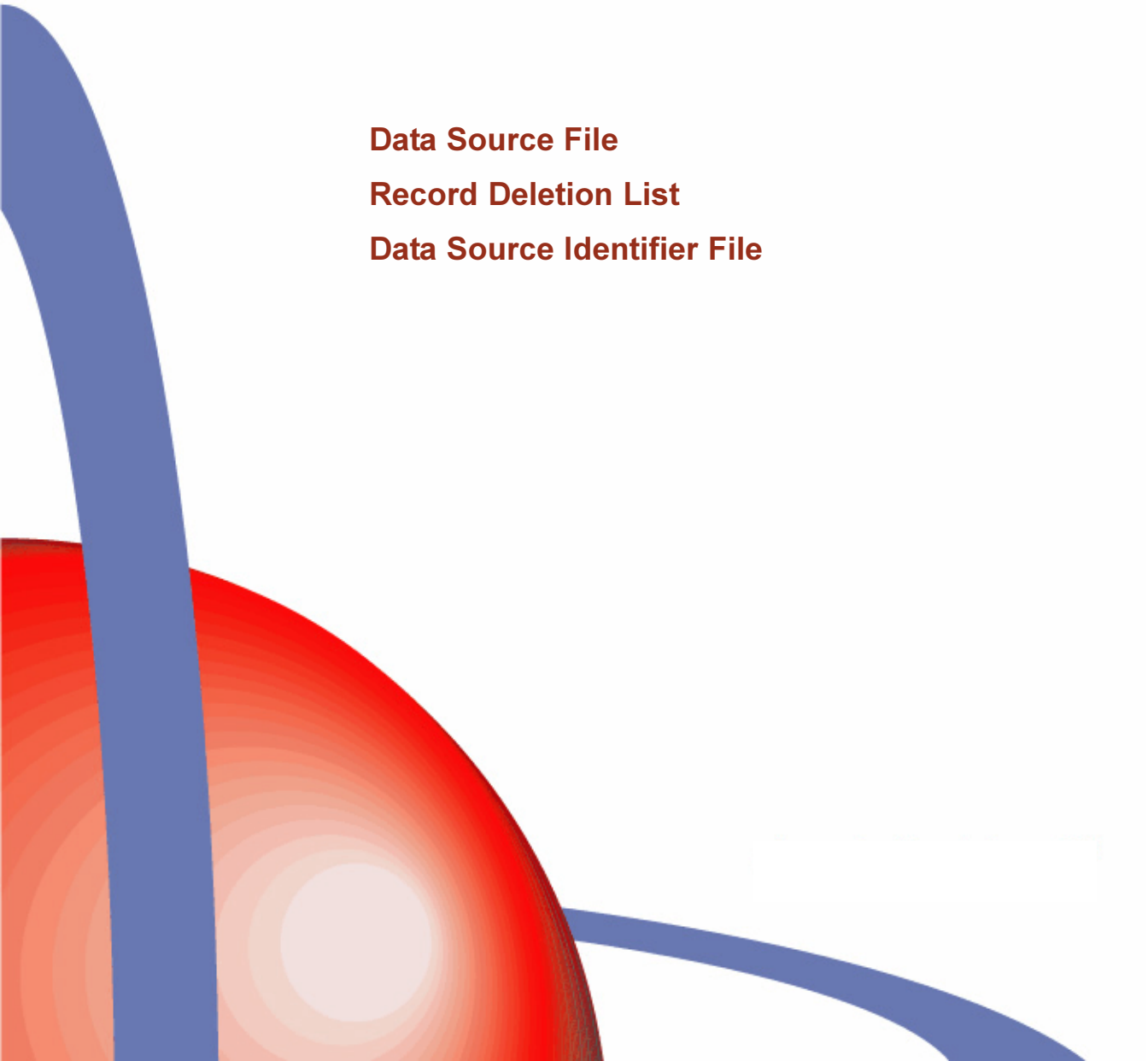
SLICCWARE™

Publication Date -- January 24, 2003

Table of Contents


SLICCWARE™

Section 1 – Database Data Source



Data Source File

Record Deletion List

Data Source Identifier File

Section 1.1 – Data Source File

A Data Source File is one method of inserting or updating records within the search engine load. The actual record data for each record to be updated is stroed within the *Data Source File*. Each record being updated by the Data Source File must be complete. No partial updates are allowed. Each time a record is updated, all previous information is discarded, and only the new information provided by the Data Source File is retained.

Each record within the Data Source File consists of a series of token:value pairs followed by an end-of-record indicator. A token:value pair is a token -- used to identify the field, followed by a colon, followed by the data to be stored within the field. The association between the tokens and the fields they identify is made within the package definition section of the *Load Definition File*. For more information see section 3.3 of the Load Definition File Reference.

keywords: pizza italian food pasta spaghetti

The data may be of any size, but must not contain the termination character except to terminate the data. The default termination character is a newline character, ASCII 10. However, a different termination character may be identified within the *Load Definition File* for any data source. Multiple token:value pairs may be used to set a single field within the same record. The data from each successive token:value pair will be concatenated to the previous data with a single blank in between.

The following two token:value pair entries

keywords: pizza italian food pasta
keywords: rigatoni ravioli wine

result in a single field containing the following data

pizza italian food pasta rigatoni ravioli wine

Exactly one of the token:value pairs within each record must contain a unique value used to identify the record. This value, known as the identity value, allows each record may be inserted, referenced, updated and deleted independently of all other records.

Record Deletion List is used to remove records from the search engine load. Records to be deleted are identified by the record indentity values. One identity value or an identity value range may be defined on each line. An identity value range is signified by two identity values seperated by a hyphen. An identity value range includes the two end-point identity values.

Contents of a *Record Deletion List* used to delete records 1234, 1275 and records 1302 through 1317

```
1234
1275
1302-1317
```

A *Record Deletion List* can be used to remove records that are part of a *Data File Section* as defined by the *Load Definition File*. For more information see section 3.3 of the Load Definition File Reference.

Only records that are to be deleted should be placed into the *Record Deletion List*. Records that are to be updated should not be placed into the *Record Deletion List*. They should, instead, be placed only into the *Data Source File*. The update process will guarantee removal of the old data and index links before inserting the new data and index links resulting from the update.

Section 1.3 – Data Source Identifier File

The Data Source Identifier File is used to identify the *Data Source Files* and *Record Deletion Lists* which are available for processing. One entry is made for each *Data Source File* or*Record Deletion List*. Each entry occupies a single line and may be in either of two formats: simple format or redirection format.

Simple Format is used to identify *Data Source Files* and *Record Deletion Lists* following the load update file naming convention and stored within the "./data" directory. In this format, each *Data Source File* or *Record Deletion List* is identified by only the file name, with no path information, -- one *Data Source File* or *Record Deletion List* per a line.

Also, in this format, each *Data Source File* and *Record Deletion List* must exist within the "./load/data" directory. In addition, each *Data Source File* name must match the core file name defined within the package section for which it is intended; and each *Record Deletion List* name must match the core file name defined within the package section for which it is intended with the added extension of ".del".

SourceFile
SourceFile.del

Redirection Format is used to identify *Data Source Files* and *Record Deletion Lists* not following the load update file naming convention or not stored in the "./data" directory. In this format the *Data Source Files* and *Record Deletion Lists* may be stored anywhere which is accessible by the local file system, and may be named anything. Each entry -- one per a line -- identifies a single *Data Source File* or *Record Deletion List*. A single entry consists of two fields seperated by an equal sign.

For *Data Source Files,* the first field is the core file name associated with the package section to be updated. For *Record Deletion Lists,* the first field is the core file name associated with the package section to be updated plus the extension ".del".

SourceFile=/data/source1/updates
SourceFile.del=/data/source1/deletes

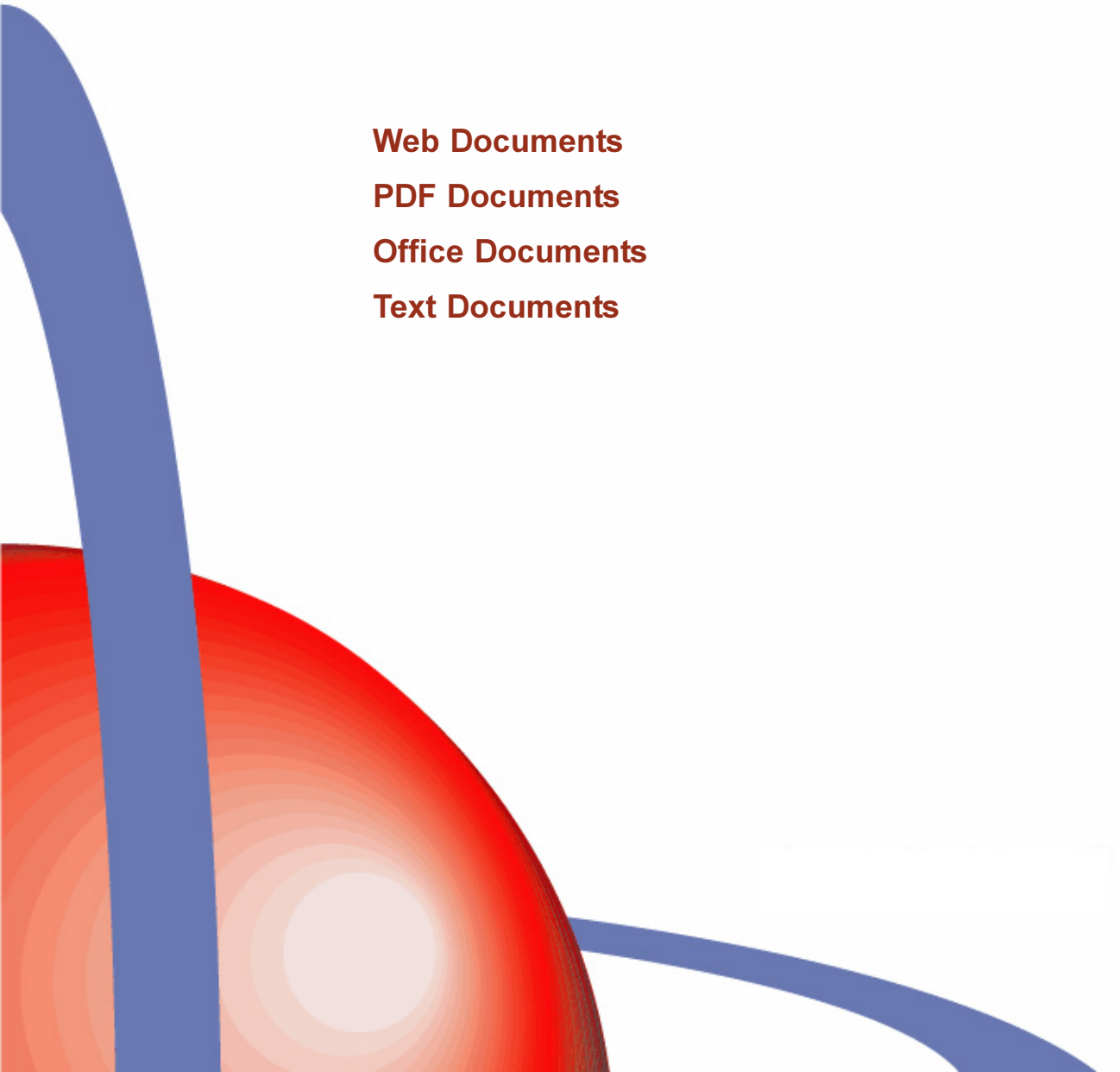
The second field is the fully qualified path to the actual file.

SourceFile=/data/source1/updates
SourceFile.del=/data/source1/deletes

With either format, all *Data Source Files* and *Record Deletion Lists* which are to be processed should be completely written out before they are added to the *Data Source Identifier File*.


SLICCWARE™

Section 2 – Document Data Source



Web Documents

PDF Documents

Office Documents

Text Documents

Section 2.1 – Web Documents

Web documents consist of HTML documents, *Active Server Pages,* and other XML based documents. Web documents are comprised of text data and tags. For the most part it is the text data within the document that is indexed. However, certain tags play a part in the indexing as well. In almost all instances, information within tags is not indexed.

Meta-tags. A specialized tag, called a meta-tag is used to identify meta-data associated with the document. It is the only type of HTML tag which contains information that is indexed. Each meta-data item is identified as a name-value pair within the meta-tag. Meta-data can be defined as information about, or relating to the content of a document, but not directly part of the content.

```
<meta name="author" value="John Q. Smith">
```

The name portion of the meta-tag contains the name assigned to the meta-data item, which can be associated with a specific field within the data-store. The association between the meta-data name and the field it identifies is made within the package definition section of the *Load Definition File*. For more information see section 3.4 of the Load Definition File Reference.

The value portion of the meta-tag contains the actual data to be stored or indexed.

Anchor tags. Another type of tag which is of importance is the anchor tag. This tag becomes important when used to define a hyperlink. A hyperlink consists of an anchor tag containing an HREF statement, a text portion which is selectable by the reader, and a terminating anchor tag.

```
<a href="http://192.168.1.1">Local Home</a>
```

No portion of the hyperlink, including the text -- which is shown in red in the example, is indexed. This prevents documents which list other major stories as hyperlinks from being incorrectly indexed as containing information about those stories.

Section tags. Another tag that is of interest is the section tag. A section tag is a specialized comment tag used by *flexible* **scan™** to subdivide the HTML document into sections for individual storage and indexing. The start of the section is identified by a comment string consisting of a single word. The end of the section is identified by an almost identical comment string that has the single word preceded by a slash. The data within the sections is loaded into data fields or temporary fields associated with the single word

identifier found within the comment string. The association is made within the *Load Definition File* in the same way as that for meta-data. For more information see section 3.4 of the Load Definition File Reference.

```
<!-- SYNOPSIS -->
<p>Out of the country for twenty years, Smith . . .
<!-- /SYNOPSIS -->
```

In the example, the data between the two comment strings is loaded into the data field or temporary field associated with the term "SYNOPSIS" within the *Load Definition File*.

Legacy section tags. A second method of sectioning HTML files using comment tags is also supported. This method is supported primarily to allow for upward compatibility with installations designed around other search engines. Using this method, a section to be indexed is preceded by a specialized comment string. The comment string has the format:

<!-- flagtoken -->

where **flag** is the token-flag defined within section 3.4 of the *Load Definition File*.
token is the single word associated by the *Load Definition File* with the data field or temporary field destined to hold the data.

The section is terminated by the specialized comment string beginning the next section, or a comment string containing the token-flag and nothing else, the end of the file.

```
<!-- plsfield:author -->
<i>by John Doe</i>
<!-- plsfield:content -->
Today at 3:45 eastern time, two men entered . . .
<!-- plsfield: -->
```

In addition to content and meta-data found within the document, ***flexible* scan™** also supports storing and indexing of other information such as document name, path components, complete URL, unique id, and current date or file date. This data is accessed by associating special tokens with data fields or temporary fields within the *Load Definition File*. For more information about those special tokens, see section 3.4 of the Load Definition File Reference.

Section 2.2 – PDF Documents

PDF documents are comprized of formatting information, individual characters, images, and meta-data. A PDF document has no concept of stories, sentences, or even words. When scanning a PDF document, ***flexible* scan™** must recreate words, sentences, and stories from the individual characters by analyzing how they are positioned on the page. ***flexible* scan™** then, uses those words, sentences and stories to generate the content for storage and indexing.

In addition to content, ***flexible* scan™** also supports storing and indexing the meta-data found within a PDF document. Each meta-data item within the PDF document has assigned to it, a name, which can be associated with a specific field within the data-store. The association between the meta-data name and the field it identifies is made within the package definition section of the *Load Definition File*. For more information see section 3.4 of the Load Definition File Reference.

Among the meta-data names recognized are the following:

creationDate	The date the file was create.
moddate	Last time the file was modified.
producer	Producer of the document.
author	Author of the document.
creator	Original creator of the document.
title	The title of the document.

In addition to content and meta-data found within the document, ***flexible* scan™** also supports storing and indexing of other information such as document name, path components, complete URL, unique id, and current date or file date. This data is accessed by associating special tokens with data fields or temporary fields within the *Load Definition File*. For more information about those special tokens, see section 3.4 of the Load Definition File Reference.

Office documents are comprized of formatting control characters, identifiable text content, and meta-data. In order to properly scan office documents, meta-data must be properly extracted, and the text content must be accurately seperated from the formatting control characters. To accomplish this task, ***flexible* scan™** relies on Microsoft software present within the more recent Microsoft operaring systems. For those older supported Microsoft operating systems which may not contain the software, a special Microsoft DLL is included with the installation package.

Just as with HTML and PDF documents, the names identifying the various meta-data items can be associated with data fields and temporary fields within the *Load Definition File*. For more information see section 3.4 of the Load Definition File Reference.

Among the meta-data names recognized are the following:

DocTitle	Title of document.
DocSubject	Subject of document.
DocAuthor	Author of document.
DocKeywords	Document keywords.
DocComments	Comments about document.
DocTemplate	Name of template for document.
DocLastAuthor	Most recent user to edit document.
DocRevNumber	Current version number of document.
DocEditTime	Total time spent editing document.
DocLastPrinted	Time document was last printed.
DocCreatedTm	Time document was created.
DocLastSavedTm	Time document was last saved.
DocPageCount	Number of pages in document.
DocWordCount	Number of words in document.
DocCharCount	Number of characters in document.
DocThumbnail	Thumbnail
DocAppName	Name of application that created the file.
DocSecurity	Control permissions.

n addition to content and meta-data found within the document, ***flexible* scan™** also supports storing and indexing of other information such as document name, path components, complete URL, unique id, and current date or file date. This data is accessed by associating special tokens with data fields or temporary fields within the *Load Definition File*. For more information about those special tokens, see section 3.4 of the Load Definition File Reference.

Section 2.4 – Text Documents

Text documents are comprized of only text context and the simple formatting control characters: tabs, line feeds and carriage returns. Text documents do not contain any meta-data. Nor do they contain any specialized strings such as the various tags found in web documents. As a result, all text within a text document is indexed when the document is indexed. No special meta-tag handling or partitioning of the document is supported.

In addition to content found within the document, *FlexibleScan* also supports storing and indexing of other information such as document name, path components, complete URL, unique id, and current date or file date. This data is accessed by associating special tokens with data fields or temporary fields within the *Load Definition File*. For more information about those special tokens, see section 3.4 of the Load Definition File Reference.

INDEX

D

H

I

M

O

P

R

S

T

W