USER'S GUIDE

PERSONAL LIBRARIAN

Version 4.1 for Windows

PERSONAL LIBRARY SOFTWARE

Personal Library Software
2400 Research Boulevard
Suite 350
Rockville, MD 20850
USA
Tel. 301-990-1155
FAX. 301-963-9738
e-mail. support@pls.com

CONTENTS

PART 2 SEARCHING

CHAPTER 5 VIEWING SEARCH RESULTS 47

CHAPTER 6 VIEWING RECORDS 63

CHAPTER 9 TABLE OF CONTENTS SEARCHING 135

CHAPTER 10 LINKING TO OTHER INFORMATION 145

PART 3 DATABASE ADMINISTRATION

CHAPTER 14 CREATING A DATABASE 201

CHAPTER 15 BUILDING AND UPDATING A DATABASE 233

PART 4 APPENDICES

APPENDIX A QUICK REFERENCE 257

APPENDIX B CUSTOMIZING PERSONAL LIBRARIAN 265

PREFACE

Welcome to the Personal Librarian User's Guide. This manual will instruct you in searching and authoring large text databases using, respectively, Personal Librarian (a.k.a. PL) and its database administration module, PL-Admin.

ORGANIZATIONAL OVERVIEW

Take a moment to browse through the table of contents, in order to get a feel for how this manual is organized. Chapters are grouped into the following partitions:

Part 1: Getting Started. This section discusses installation of Personal Librarian and includes an introductory chapter which makes important points about the application’s searching philosophy.

Part 2: Searching. From these chapters you will learn techniques for extracting and manipulating information from your databases. Among the topics covered are query composition, review of search results, navigation within a database, online searching tools, and use of hypertext links.

Part 3: Database Administration. These chapters show you how to build databases in ways that facilitate more productive and user-friendly searching. Included are discussions of database design and creation, preparation of text files, integration of hypertext links, and database maintenance.

Part 4: Appendices. These appendices provide quick-reference materials and customization instructions, along with a discussion of advanced topics, including networking, CD-ROM databases, DLL links, DDE capabilities, and batch processing of administration tasks.

CONVENTIONS

This manual uses a consistent form of shorthand for referring to keyboard and mouse actions. This style of shorthand is illustrated by the examples in Table 1.

Table 1 Keyboard and Mouse Actions

Action	Explanation
SHIFT+F7	Hold down SHIFT, and press F7.
Alt+F, B, N	Press and release ALT and F in unison, then press B and N in succession.
click	Press the left mouse button.
double-click	With the mouse stationary, press the left mouse button twice rapidly.
control-click	Hold down CTRL, and press the left mouse button.
shift-click	Hold down SHIFT and press the left mouse button.
drag	Hold down the left mouse button while moving the mouse.

ASSUMPTIONS

This manual assumes that you are familiar with the Windows operating system. If you don't know what a directory is, or the difference between copying and moving a file—if you don't know how to point-and-click, drag, double-click or open a pull-down menu—then you should spend some time with a Windows User's Guide. It is also assumed that you know how to operate an ASCII text editor (e.g., Windows Notepad).

This manual further assumes that your computer is equipped with a mouse; keyboard equivalents exist for many PL functions, but a mouse is needed to take advantage of all features.

PART 1

GETTING STARTED

1

CHAPTER 1

INTRODUCTION

This chapter introduces you to principles involved in using Personal Librarian to search text databases. Also discussed are basic database terminology and specific PL features that augment your search activities.

SEARCHING WITH PERSONAL LIBRARIAN

Searching is a process, not an event. This should be your guiding principle when using Personal Librarian. Searching a database is not about spending time and mental energy formulating a "golden query" that will retrieve all desired information in a single stroke. In practice, searching usually consists of issuing a succession of queries until you are satisfied with the results. After each query, you evaluate its success by asking:

> *"Did I find what I was looking for?"*
>
> *"Is better information still out there?"*
>
> *"How can I refine my query to find better information?"*

Issuing multiple queries can be frustrating or rewarding, depending on the overall time it takes to find what you really need. With Personal Librarian, you can expedite successful retrieval by getting advice from the software and using the results of prior searches to formulate new queries. This interactive process is called *feedback searching*, a rapid and intuitive approach to getting the information you want.

The following topics include a discussion of basic database terminology, along with descriptions of the tools Personal Librarian provides for feedback searching.

TEXT DATABASE FUNDAMENTALS

DATABASES, RECORDS, AND FIELDS

A text database is a collection of related documents assembled into a single searchable unit. The individual documents can be massive or minuscule, but they should bear some relation to each other.

A database is composed of smaller units called records. In a text database, a record can be an entire document, a section within a document, a single page or a fragment of text within a page. When you search a database, you will retrieve one or more records containing information that satisfies your query.

A record can contain smaller regions of data called fields. A field usually defines a particular type of data common to several or all records within a database. For example, in a database of corporate memos, wherein each memo makes up a record, the following fields might be used: *TO*, *FROM*, *DATE*, *SUBJECT*, and *TEXT*. You can narrow the scope of a search by restricting it to one or more fields. In this example, you might limit your search to the *FROM* field when searching for a sender's name. Only those records with the specified name in that field would be retrieved.

STOPWORDS

As opposed to a keyword-based system, Personal Librarian is full-text retrieval software, meaning that it indexes every word in a document, with the exception of *stopwords*. Stopwords are those terms that PL is programmed to ignore during the indexing and retrieval processes, in order to prevent the retrieval of extraneous records. Generally, a stopword list includes articles, pronouns, adjectives, adverbs, and prepositions (*the*, *they*, *very*, *not*, *of*, etc.) that are most common in the English language. Use of stopwords is not required. However, after reading the following discussion of relevance ranking, you'll understand why a stopword list is used by default.

RELEVANCE RANKING

The most powerful weapon in the Personal Librarian arsenal is relevance ranking. Simply put, relevance ranking arranges a set of retrieved records so that those most likely to be relevant to your request are shown to you first. That is, after Personal Librarian retrieves all documents that satisfy your search query, it uses relevance ranking to arrange them based on a measurement of similarity between your query and the content of each record.

What determines the likelihood of relevance? Personal Librarian performs a content analysis of records in your database by using a combination of the following indicators:

Breadth of Match. The more distinct query terms that appear in a record, the higher the weight of relevance.

Inverse Document Frequency. Rare terms (within the entire database) receive a higher weight of relevance.

Frequency. The number of times a query term occurs in a document.

Density. The comparable length of retrieved documents.

Consideration of these combined criteria produces intelligent on-the-fly evaluation of a record's likelihood of satisfying the intent behind your query.

This allows you to find more relevant information with less effort. Regardless of how many records your search query retrieves, you will have to review relatively few of them, because moving down the ranking means moving toward less relevant records. With relevance ranking, you will spend less time reviewing search results before deciding whether they are satisfactory.

Additionally, you are free from the burden of composing complex logical queries, which are used to reduce the amount of retrieved data to manageable proportions. You don't have to care about how many records are retrieved, as long as you know that the best information floats to the top.

Note: Now you can see why stopwords exist. Since words like *the* and *of* are not meaningful search terms but are so commonly used in English, their presence could artificially boost the relevance of what are actually irrelevant documents. Stopwords also provide a speed advantage, since a search for *the* would probably retrieve every record in a database.

INTELLIGENT SEARCHING

Since relevance ranking relieves you of constructing complex query syntax, you get to enter *natural language queries*. This means that you can describe the information you want in the same terms you would use in speaking naturally. For example, if you wanted to find information on offshore oil drilling in the Gulf of Mexico, you could enter the following natural language query in Personal Librarian:

```
offshore oil drilling in the Gulf of Mexico
```

A comparable query using Boolean logical syntax would be entered as:

```
offshore OR oil OR drilling OR Gulf OR Mexico
```

As you can see, Boolean searching makes you accommodate the computer by speaking its language; PL's intelligent processing of your natural language input allows the computer to accommodate you by understanding your native tongue. Since you don't have to spend the time figuring out the correct logic and phrasing that Boolean queries require, you can find what you need faster.

Of course, there are times when you will need the narrow focus and precision that a Boolean query provides. For this reason, Personal Librarian fully supports Boolean searching.

ADVANCED SEARCHING TOOLS

With easily composed natural language queries and relevance ranking, you will be put in touch with more relevant documents in a shorter period of time. Since overall searching time is reduced, you will be able to take advantage of several powerful feedback tools that Personal Librarian makes available.

FEEDBACK SEARCHING

Personal Librarian promotes spontaneity by allowing an interactive search process. When you see results from one search, they may inspire new ideas for other information that you didn't originally know you wanted. You can then use PL's tools to incorporate these new ideas into your query. As you combine methods to refine your query, you quickly get closer to the information that is most important to you. You will find that feedback searching usually yields more satisfactory results than traditional Boolean strategies. See *Including Words from the Hitlist in a Query* on page 52 and *Including Text from a Record in a Query* on page 70.

EASY INCLUSION OF NEW QUERY WORDS

Many times you will notice on the screen a word or phrase that didn't occur to you, but would make an excellent search term. Personal Librarian makes it easy to include that text in your query. By eliminating the need for manual entry, this feature prevents typing mistakes and reduces overall searching time.

CONCEPT SEARCHING

This tool lets you search based on a general idea. In cases where you can't immediately articulate your information needs, you can perform a concept search and build on the results. See *Concept Search: Exploring a General Idea* on page 39 and *Word-Level Concept Searching* on page 104.

ADVISORS

Personal Librarian features tools called *advisors*, which serve as online assistants. As part of the feedback searching process, you can use the advisors to gain added insight into the nature of a database's contents.

The Relate Advisor can suggest appropriate search terms that you might never have considered. It produces a list of words that, in the context of the database, are related to the words in your current query. See *Finding Words Related to Your Query* on page 90.

The Dictionary Advisor can verify the existence and popularity of a query word and/or its variants within a database. With this advisor, you can decide if a word might yield good results, or if it should be excluded. See *Viewing a Database's Dictionary* on page 88.

If you want to reiterate an earlier query or include it as part of a new one, you can use the History Advisor. This tool keeps track of all queries issued during a search session, and lets you include them (whole or in part) in a new query. See *Reusing Previous Queries* on page 95.

The Fuzzy Match Advisor scans the database for words with spelling similar to that of a query term. It can identify correct spelling—if you're in doubt about yours—and instances where scanned material has been incorrectly interpreted during OCR (optical character recognition) processing. See *Fuzzy Matching: Identifying Words with Similar Spelling* on page 92.

You can use the Thesaurus Advisor to have a single query word represent many synonyms, saving yourself the trouble of manually entering a multitude of related or equivalent terms. See *Viewing a Database's Thesaurus* on page 94.

SEARCH OPERATORS

When formulating queries, you can take advantage of the wide variety of operators supported by Personal Librarian. Using operators, you can impose Boolean logic to produce specific search results, find occurrences of adjacency/proximity of search terms, search for exact phrases, impose algebraic logic on complex queries, restrict a search to one or more fields, etc. See *What is a Query Operator?* on page 101.

FIELD-RESTRICTED SEARCHING

For databases containing records with fielded information, you can perform field-restricted searching, wherein PL's search considers only the contents of one or more specified fields. See *Field-Restricted Searching* on page 125.

HYPERTEXT LINKS

With PL, you can use hypertext links to call up supplemental information related to a record. By activating a link, you can view an image, display another record, execute a new search, or start up an external application (word processor, spreadsheet, etc.). See *What are Hypertext Links?* on page 145.

CROSS-PLATFORM COMPATIBILITY

Personal Librarian shares a common interface for the Windows and Macintosh operating systems. Moreover, PL database files are interoperable (i.e., a Windows user can search a database built on a Macintosh, and vice versa).

CONCURRENCY OF SEARCHING AND ADMINISTRATION

Maintaining a PL database does not require taking it off-line (i.e, making it unavailable for searching). You can update a database concurrent with it being searched by someone else. Therefore, you can keep information as current as possible (being able to update it at any time) while providing round-the-clock search access.

CHAPTER 2

INSTALLING PERSONAL LIBRARIAN

This chapter tells you how to install Personal Librarian from the distribution diskettes included in the product package.

BEFORE YOU BEGIN…

You should make backup copies of your PL distribution diskettes. Place the originals in a safe place, and when installing the software, do so from the backup disks. Keep the serial number from disk #1 handy by writing it on the backup disks or on the back of this manual's cover page.

SYSTEM REQUIREMENTS

In order to run Personal Librarian and PL-Admin, your computer:

- must have a 386 or higher CPU.
- must be running with MS-DOS 3.3 or higher.

- must be running with Microsoft Windows 3.1 or higher; Windows must be running in enhanced mode.
- must have 2MB of RAM; 4MB or more is recommended.
- must have an EGA or higher video adapter; VGA is recommended.
- must have 6.5MB of free hard disk space for installation of all PL software; if you elect not to install the sample database shipped with PL, only 5MB of hard disk space will be required.
- must have adequate free hard disk space for creation of temporary and database files—see *Preparing to Index Files* on page 236.
- should have a mouse; keyboard equivalents exist for many PL functions, but a mouse is needed to take advantage of all features.
- should have a color monitor—some features may be difficult to use on a monochrome monitor.

OPTIMIZING YOUR SYSTEM ENVIRONMENT

You can take the following steps to expedite processing of tasks in Personal Librarian and PL-Admin:

- Increase your system's RAM.
- Remove any unnecessary device drivers.
- Utilize disk caching (e.g., Smartdrive).

THE INSTALLATION PROCESS

OVERVIEW

When installing Personal Librarian, you will select one of two installation methods: express or custom.

If you select express installation, you will have the option of installing either Personal Librarian (for database searching) or PL-Admin (for database building), or both. The software will be installed in a `\plwin` directory on your hard drive, along with a sample database of information excerpted from this manual.

During custom installation, you will get to make the following choices:

- The destination hard drive and directory for installation of PL applications.
- The destination hard drive—local or networked—and directory for installation of shared PL configuration files. In a network environment, this directory can be used to share software settings among multiple users.
- The destination hard drive and directory for storage of temporary files that are created by PL applications.
- Which application modules will be installed—Personal Librarian, PL-Admin, or both.
- Whether the User’s Guide sample database will be installed.

INSTALLATION INSTRUCTIONS

You can cancel the installation process at any time by clicking on the current dialog box’s Cancel button. During installation, you should fully read all dialogs, as they contain supplemental information about your particular installation circumstances.

Express Installation

1. Insert PL distribution diskette #1 in your floppy drive.

2. Select Run from the Program Manager’s File menu.

3. In the Run dialog’s Command Line box, type ***x*:\winstall**, where *x* is the letter of your floppy drive, and click the OK button.

4. An introductory installation dialog will appear, with the express installation method as the default selection. After reading the dialog, click the OK button.

5. With the next dialog, you will determine which applications to be installed and whether the sample User’s Guide database, which comprises information excerpted from this manual, will be installed. By default, both applications—Personal Librarian and PL-Admin—and the sample database are selected for installation; you can de-select an item by clicking on it. When you are satisfied with the selections, click the OK button.

6. The next dialog will summarize your selected installation options. At this point you can elect to proceed with installation or change options. After selecting a course of action, click the OK button.

7. When prompted to do so, insert the other PL distribution diskettes.

8. When installation is complete, you will have a Personal Librarian program group in the Program Manager window. Before running Personal Librarian or PL-Admin, you should use a text editor (e.g., Notepad) or word processor to review the `readme.txt` file located in the PL home directory; this file can contain last-minute information, not available when this manual went to press, concerning operation of Personal Librarian applications.

Custom Installation

1. Insert PL distribution diskette #1 in your floppy drive.

2. Select Run from the Program Manager's File menu.

3. In the Run dialog's Command Line box, type ***x*:\winstall**, where *x* is the letter of your floppy drive, and click the OK button.

4. An introductory installation dialog will appear, with the express installation method as the default selection. After reading the dialog, select the custom installation option and click the OK button.

5. A short series of dialogs will summarize your system resources. You can progress through these messages by clicking the OK button.

6. The Enter Target Drive dialog will list all hard drives—local or networked—available for installation of PL applications. After selecting the desired drive, click the OK button.

7. You can use the Enter Target Subdirectory dialog to name the directory into which PL applications will be installed; the default directory name is `\plwin`. When you are satisfied with the specified directory name, click the OK button.

Note: If the specified directory already exists, a supplemental dialog will give you the option of either overwriting or leaving intact any existing PL files that are found there.

8. With the next series of dialogs, you will specify the location of shared PL configuration files. If you and other networked PL users wish to share the same databases and software settings, you can all specify the same shared directory during the installation process. After reading the introductory dialog for this installation stage, click the OK button.

9. The Enter Target Shared Drive dialog will list all hard drives—local or networked—available for installation of the shared files. After selecting the desired drive, click the OK button.

10. You can use the Enter Target Shared Directory dialog to name the directory into which shared files will be placed; the default directory name is `\plwin\shared`. When you are satisfied with the specified directory name, click the OK button.

Note: If the specified directory already exists, a supplemental dialog will give you the option of either overwriting or leaving intact any existing shared files that are found there.

11. With the Enter Temporary Drive dialog, you can specify the hard drive on which temporary files created by PL applications will be stored. After selecting the desired drive, click the OK button.

12. You can use the Enter Temporary Directory dialog to name the directory that will serve as a repository for PL's temporary files; the default directory name is `\plwin\temp`. When you are satisfied with the specified directory name, click the OK button.

13. With the next dialog, you will dictate which applications are to be installed and whether the sample User's Guide database, which comprises information excerpted from this manual, will be installed. By default, both applications—Personal Librarian and PL-Admin—and the sample database are selected for installation; you can de-select an item by clicking on it. When you are satisfied with the selections, click the OK button.

14. The Selected Options Review dialog will summarize your selected installation options. At this point you can elect to proceed with installation or change options. After selecting a course of action, click the OK button.

15. When prompted to do so, insert the other PL distribution diskettes.

16. When installation is complete, you will have a Personal Librarian program group in the Program Manager window. Before running Personal Librarian or PL-Admin, you should use a text editor (e.g., Notepad) or word processor to review the `readme.txt` file located in the PL home directory; this file can contain last-minute information, not available when this manual went to press, concerning operation of Personal Librarian applications.

CHAPTER 3

Starting Personal Librarian and PL-Admin

This chapter explains how to start PL applications and how to access online help from within the applications.

Starting Personal Librarian

To run Personal Librarian, double-click the PL icon, shown in Figure 1, in the Personal Librarian program group.

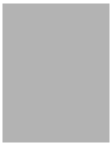

Figure 1 Personal Librarian icon

After the startup dialog, you will see the database *ShortList*, similar to the example in Figure 2.

Figure 2 The database ShortList

The ShortList is a menu of databases available for searching. It is extracted from the file `dblist.cpl`, located in PL's `\shared` subdirectory; `dblist.cpl` contains the names and locations of frequently accessed databases. If you and others users are using the same `\shared` directory, you will all share a common ShortList.

To open a listed database, highlight its name by clicking it, then click the OK button. This will place you in the main Personal Librarian workspace, where you may begin searching. See *The Search Window* on page 32.

Tip: You can also open a database from the ShortList by double-clicking its name.

CAUTION: If, during installation, you elected not to install the User's Guide sample database, you will not be able to run PL until you have created a database and added it to the ShortList with PL-Admin.

CLOSING PERSONAL LIBRARIAN

To exit PL, select Exit from the File menu (ALT+F4) or double-click the application window's control box.

STARTING PL-ADMIN

To run PL-Admin, double-click its icon, shown in Figure 3, in the Personal Librarian program group.

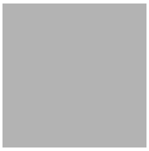

Figure 3 The PL-Admin icon

A startup dialog will appear; clicking its OK button will place you in the main PL-Admin window.

CLOSING PL-ADMIN

To exit PL-Admin, select Exit from the File menu (ALT+F4) or double-click the application window's control box. If you have any incomplete administration transactions pending, a dialog box will prompt you to confirm that the transactions are to be discarded.

USING ONLINE HELP

Personal Librarian uses the native Windows help viewer to display help topics. To learn how to navigate among help topics with this viewer, you should consult your Windows User's Guide.

You can access help topics through the Help menu, found in both PL and PL-Admin; this menu comprises the following commands:

Introduction. Selecting this command displays an introductory topic in the online help system.

Contents. Selecting this command displays an outline of major topic browse sequences within the online help system; each browse sequence heading serves a hypertext link that will display the first topic in the sequence.

Index. Selecting this command displays a dialog box with which you can search for specific help topics based on keywords associated with the topics.

CONTEXT-SENSITIVE HELP

In addition to the Help menu, PL applications feature *context-sensitive* help, which displays a help topic related to the functional area with which you are currently involved. For example, if you summon context-sensitive help while a record window is active, PL will display the help topic dealing with record windows.

While working in Personal Librarian or PL-Admin, you can display a context-sensitive help topic by pressing F1.

EXITING THE HELP SYSTEM

To dismiss the online help viewer, select Exit from the File menu or double-click the application window’s control box.

PART 2

SEARCHING

2

CHAPTER 4

PERFORMING A SEARCH

This chapter discusses fundamental searching techniques, including opening a database, composing basic queries, and initiating a search based on a query.

OPENING A DATABASE

When you start Personal Librarian, a database ShortList is displayed, whereupon you select a database for searching. See *Starting Personal Librarian* on page 21.

Once a search session has begun, you can open databases in the following manner:

To open a database:

1. Select Open Database from the File menu (CTRL+O).
2. When the ShortList dialog is displayed, highlight the desired database by clicking its name; you can select a contiguous or non-contiguous range of databases by, respectively, shift-clicking or control-clicking on their names.
3. Click the OK button.

Tip: You can also open a database from the ShortList by double-clicking its name or by highlighting its name and pressing ENTER.

USING A VIRTUAL DATABASE

You can have multiple databases open for searching simultaneously; this is known as a *virtual database*. With a virtual database, you can search across several databases with a single query.

Note: You can have up to a maximum of ten databases open simultaneously, depending on your system's available resources.

To create a virtual database for searching, simply open additional databases in the normal manner. The ShortList will display those databases that are not already open.

CAUTION: Searching a virtual database may demand more of your system resources—RAM and free hard drive space—than searching a single database. Also, a given search, when performed on a virtual database, may be slightly slower than when performed on a single database.

OPENING A RESTRICTED DATABASE

PL databases can have access restrictions for security purposes. If you try to open a restricted database, a password entry dialog similar to Figure 4 will be displayed.

Figure 4 The Password Entry dialog

If you are authorized to search the database, enter the password and click the OK button.

CLOSING A DATABASE

To close a database:

1. Select Close Database from the File menu (CTRL+W).

2. If the Search Results Window, any images, or any records are open, the dialog in Figure 5 will be displayed. Click the OK button.

Figure 5 Closing a database discards current search results.

3. If only one database is open, it will close after Step 2; if multiple databases are open, a dialog similar to Figure 6 will be displayed. Highlight the database you want to close by clicking its name; you can select a contiguous or non-contiguous range of databases by, respectively, shift-clicking or control-clicking on their names.

Figure 6 The Close Database dialog

4. Click the OK button.

THE SEARCH WINDOW

When you open a database in PL, the Search Window shown in Figure 7 will be displayed.

Figure 7 The Search Window

Think of this window as the point from which you will coordinate most of your search activities. You will use it to enter queries, initiate searches, activate online Advisors (see *Activating the Advisors* on page 87), and perform concept searching (see *Concept Search: Exploring a General Idea* on page 39).

In searching a database, you will enter a single query or a series of queries. A query is a word or phrase that describes the information you are seeking. A query word is a continuous string of characters containing, with the exception of search operators, no spaces or punctuation marks. You will enter your search queries in the long, horizontal text box that fills the upper half of the Search Window.

Personal Librarian is not case-sensitive in accepting queries. You can enter query words in upper-case, lower-case or a combination of the two.

Once you have entered a query, you can execute a search for records containing those words by clicking the Search button or pressing ENTER.

NATURAL LANGUAGE QUERIES

You are strongly encouraged to use natural language queries, an example of which is shown below.

```
gold medal winners in the 1992 Olympics
```

A natural language query is one that is expressed using normal conversational syntax; that is, you phrase your query as if making a spoken or written request to another person. There are no syntax rules or conventions for you to learn.

Natural language queries generally find more relevant information in less time than traditional Boolean queries, which, while precise, require strict interpretation that can often exclude information that is relevant to your interests.

USING A WORD MULTIPLE TIMES IN A QUERY

By using a particular term more than once in a single query, you can cause the term to be weighed more heavily in the relevance ranking process. Consider the following queries:

```
Clinton Health Plan
```

```
Clinton Clinton Clinton Health Plan
```

During relevance ranking of retrieved records for the first query, *Clinton*, *Health*, and *Plan* will be weighed equally. For the second query, PL will consider *Clinton* more significant than *Health* or *Plan* when assigning relevance scores to retrieved records.

UNDERSTANDING THE DEFAULT OPERATOR

Unless you enter an explicit logical search operator in your query, Personal Librarian interprets the query using a predetermined *default operator*. The operators that can be used as the default are *AND*, *OR*, and *ADJ*.

By default, *OR* is the default operator; it is automatically used in the absence of any manually-entered operators. If you enter a multiple-word query, PL retrieves all records that contain at least one of the words. For example, the following query:

```
tribal politics in South Africa
```

is interpreted as:

```
tribal OR politics OR South OR Africa
```

Any record in the current database that contains one or more of those terms satisfies the conditions of the search and therefore, is retrieved.

Tip: If you want to define an operator other than OR as the default, see *Customizing Search Performance* on page 41.

EDITING QUERIES

Editing of queries in the Search Window is consistent with most text-editing conventions. The DELETE and BACKSPACE keys function normally; you can manually position the insertion point to insert text; you can select a range of characters by dragging the cursor, etc.

PASTING TEXT FROM EXTERNAL APPLICATIONS INTO A QUERY

You can refine a query using words or text selections from external applications. For example, you can transfer a portion of a word processor document to the Search Window. The only requirement for this process is that the source application be able to copy text to your operating system's clipboard.

To transfer text from another application to your search query:

1. Make the external program your currently active application.

2. Using methods common to that application, select the text you wish to import to your query.

3. Copy the selected text to the clipboard.

4. Make Personal Librarian the active program.

5. Make the Search Window the active window within PL.

6. Select Paste from the Edit menu (CTRL+V). The copied text will be added to your query.

Note: This process can be reversed; you can copy all or part of a query to the clipboard, and paste the material into an external application.

CAUTION: If you paste very large amounts of text into the Search Window, the selection may be truncated. Also, some characters and punctuation marks contained in pasted text may be interpreted as search operators by PL. To learn about the operators used in PL's query language, see *Using Operators* on page 101 and *Query Language Operators* on page 257.

CLEARING THE SEARCH WINDOW

To instantly erase your entire query, click the Search Window's Clear All button.

WHEN A WORD IS NOT IN THE DATABASE

Personal Librarian actually begins to evaluate your query as you are composing it. If, while typing, you enter a word that does not exist in the current database—single or virtual—or in PL's stopword list, you will receive a message similar to Figure 8.

Figure 8 PL alerts you to extraneous query terms.

This helps you formulate streamlined queries that are free of words for which no records will be retrieved.

Note: This message is strictly informational; it does not prevent you from searching on the word.

Tip: If you want to disable this advisory feature, see *Customizing Search Performance* on page 41.

USING ONLY PART OF A QUERY FOR A SEARCH

If you have selected a block of text within your query, the Search Window, as shown in Figure 9, will display a message indicating that only the selected portion of the query will be used for searching. That is, a search will look only for records that contain the highlighted words; any words outside the selected range of text will be ignored.

Figure 9 Search activities will be confined to the selected portion of the query.

Note: Selecting a portion of a query also restricts the scope of operation of the Dictionary, Relate, and Fuzzy Match advisors.

CANCELING A SEARCH IN PROGRESS

While a search is being processed, a dialog similar to Figure 10 is displayed. If you want to interrupt the search, click the dialog’s Cancel button or press ESC.

Figure 10 Click the Cancel button to halt a search in progress.

CONCEPT SEARCH: EXPLORING A GENERAL IDEA

Because of the richness of language, a topic or idea can be described with a wide variety of words or expressions. For this reason, a conventional search may be too limited when you want to research a broad topic area. If other records discuss the desired topic, but do so in terms other than those included in your query, they may not be retrieved.

Rather than formulating an all-encompassing query, you can perform *concept searching*. To perform a concept search, enter a simple query describing your area of interest and click the Search Window's Concept button.

In a concept search, PL first generates a list of terms that are statistically related to the words in your query. This part of the process is equivalent to the operation of the Relate Advisor—see *Finding Words Related to Your Query* on page 90. Those words that have a significant degree of co-occurrence with your query words are deemed related within the context of the database. If you are searching a virtual database, related words will be identified for all open databases.

The concept search operation then performs a conventional search using the original query words and the 20 most significant related terms. You will find that many of the records retrieved, while perhaps not having occurrences of your original query words, will nonetheless contain information that is relevant to your search interests.

Note: When generating a related word list, PL considers the contents of all indexed fields, not just the contents of fields that are defined as searchable. See *Assigning Field Attributes* on page 207.

WHEN A SEARCH RETRIEVES NOTHING

If you execute a search and no records in the current database satisfy your query, PL displays a message to that effect. The message is more common when a query includes the AND operator, proximity operators (e.g., ADJ, NEAR, SAME), or field restriction.

If your query uses the default OR operator, the message means that none of the current database's records contain occurrences of any of the words in the query. In that case, you might want to compose a more general query.

CUSTOMIZING SEARCH PERFORMANCE

You can modify certain search performance parameters to conform to your personal preferences. To customize these parameters, select Searching from the Prefs menu. The Searching Preferences dialog, shown in Figure 11, will be displayed.

Figure 11 The Searching Preferences dialog

If, when executing a new search, you want to receive a warning that the current contents of the Search Results Window will be replaced with new results, enable the Warn Before Clearing Old Results option.

Including Words from the Hitlist in a Query on page 52*Including Text from a Record in a Query* on page 70To define the default operator for your current search session, make a selection from the Default Search Operator list. If you define an operator other than OR as the default, the alternative you choose will be displayed in the Search Window's title bar.

With the checkboxes under When a Word is Not in the Database, you can modify or disable the warning that is issued when a query word is not in the current database.

To accept the current settings in the Searching Preferences dialog, click the OK button.

Note: Changes to default searching preferences normally apply only to your current search session. To retain your current preference settings for use in future search sessions, select Save Settings from the Prefs menu.

Restoring Default Preference Settings

To restore all preferences to their system-wide default settings, select Default Settings from the Prefs menu.

MODIFYING FIELD SEARCHABILITY

By default, Personal Librarian searches all record fields that are defined as searchable by the database administrator—see *Assigning Field Attributes* on page 207. However, you can customize the searchability of individual fields during a search session.

To modify field searchability:

1. Select Fields from the Prefs menu. The Set Field Attributes dialog, shown in Figure 12, will be displayed. If you are searching a virtual database, a supplemental dialog will precede the Set Field Attributes dialog; select from that dialog the currently open database for which you want to modify field searchability.

Figure 12 The Set Field Attributes dialog

Listed in the dialog’s Individual Attributes window are the names of all fields defined for the current database, along with parenthetical abbreviations for their attributes. The presence of an *S* attribute indicates that a field is defined as searchable.

2. Click the name of the field you want to modify. When the field is highlighted, its attributes are reflected in the checkboxes to the right of the Individual Attributes window, as shown in Figure 13.

Figure 13 Use the checkboxes to modify the selected field's searchability.

3. Click the Search checkbox to enable or disable searchability for the selected field.

4. Select other fields and modify searchability, as desired.

5. When you are satisfied with the searchability settings, click the OK button.

Note: Changes you make with the Set Field Attributes dialog apply only during the current search session.

SELECTING THE SEARCH WINDOW FONT

To select a font for query text in the Search Window:

1. Make the Search Window active by clicking it or selecting Search Window from the Windows menu.

2. From the Font menu, select the desired typeface, size, and formatting.

CHAPTER 5

VIEWING SEARCH RESULTS

This chapter introduces you to Personal Librarian's method of displaying search results. You can use the displayed information to evaluate the success of your search strategy.

THE SEARCH RESULTS WINDOW

Upon completion of a search, Personal Librarian displays the Search Results Window, which shows the total number of records retrieved and their relative relevance, along with a short preview of each record's contents.

Figure 14 The Search Results Window

This window, shown in Figure 14, always represents the most recently completed search. The key functional elements of the Search Results Window are the Hitlist, the bar chart, and the title bar.

THE HITLIST

Any term within a record that satisfies the conditions of a search, thereby causing the record to be retrieved, is referred to as a hit, or hit word. A record containing a hit word is known as a hit record. The *Hitlist*, which occupies the upper portion of the Search Results Window, consists of summaries for hit records retrieved by the most recently executed search. These summaries are ranked in order of relevance to the search query, and a number beside each summary indicates that record's position in the ranking. If you are searching a virtual database, hit records from all open databases are combined and ranked in a single Hitlist.

You can use the Hitlist to analyze the success of your search, viewing the summaries to gauge whether the retrieved records contain the kind of information you are seeking.

To highlight a Hitlist entry, click it or navigate to it with the up or down arrow keys. You can scroll through the list with its scroll bar or the PAGEUP and PAGEDOWN keys.

You can open a record from the Hitlist by double-clicking its summary or, having highlighted its summary, pressing ENTER. When opening a record from the Hitlist, you can close all currently open records by shift-clicking the record's Hitlist summary or pressing SHIFT+ENTER.

Note: By default, Hitlist entries are listed in order of decreasing relevance. If you want to base Hitlist order on other criteria, see *Sorting Hit Records by Field* on page 50 and *Listing Hit Records in Database Order* on page 52.

THE BAR CHART

Occupying the lower portion of the Search Results Window is a bar chart, which shows the relative relevance of records in the Hitlist; from left to right, it represents the Hitlist from top to bottom. The y-axis in this graph is calibrated as follows: the record with the highest degree of relevance—hit record #1—is assigned a 100% value, with all other hit records scaled relative to the first. If you are searching a virtual database, hit records from all open databases are combined and ranked in a single bar chart.

Note: The bar chart represents only the first 60 records of a lengthy Hitlist.

You can open a record from the bar chart by double-clicking its corresponding bar. When a record is opened, its bar becomes shaded and remains so until that record window is closed.

You can use the bar chart for moving through the Hitlist; when you click a bar in the chart, its respective summary will be displayed and highlighted in the Hitlist. Conversely, for the currently highlighted Hitlist summary, a dashed line will appear over the corresponding bar in the chart.

You should use the bar chart to determine which retrieved records are suitable for review. If the chart shows a sharp decline in relevance at any point, you probably don't want to look at records beyond the drop-off; the less relevant records may contain relatively few hits.

If the slope of the bar chart is mostly level, it signals only a small distinction in relevance between retrieved records. If the slope steepens as you add words to your query, it implies that you have isolated more useful information (i.e., records that are more distinctly relevant than others).

You can use the bar chart to employ a form of feedback searching: open the most relevant hit records and leave them open as you search with a refined query. If the shaded bars that represent the open records are displaced by the bars of more relevant records, you may be getting closer to your search goals; if the open records' bars remain the most relevant, it indicates that your revised query may not have retrieved information that is more relevant to your interests.

THE TITLE BAR

In addition to displaying the query that yielded the current results, the Search Results Window's title bar shows the total number of records that have been retrieved.

SORTING HIT RECORDS BY FIELD

While Hitlist entries are listed in order of decreasing relevance by default, you have the option of sorting the Hitlist based on the contents of one or more fields. When sorting the Hitlist in this manner, PL considers only the first term found in a field. Alphabetic and alphanumeric terms are sorted as words; numeric strings are sorted as numbers.

Note: Hitlist sorting criteria will only apply to the current search session; they cannot be saved.

To sort the Hitlist by fields:

1. Select Sort Hitlist By Field Contents from the Prefs menu. A dialog similar to Figure 15 will be displayed.

Figure 15 The Select Fields To Sort On dialog

2. Click the field you want to use for top-level sorting.

3. Click the button with rightward-pointing arrows. This moves the selected field to the Fields to Sort On window and enables the Reverse Sort Order checkbox.

4. By default, sorting is done in ascending order—A before B, 1992 before 1993, etc. To sort in descending order for the currently highlighted sort field, click the Reverse Sort Order checkbox.

5. Repeat steps 2–4 to select fields for lower sorting levels.

Tip: You don't have to select sort fields in sort-level order. Because a new sort field will be added immediately below the currently highlighted field in the Fields to Sort On window, you can, prior to moving a new sort field to that window, highlight the desired destination in the existing sort order hierarchy.

6. To enable field-based sorting of subsequent Hitlists—any existing search results will be unaffected by the new sorting criteria—click the OK button.

LISTING HIT RECORDS IN DATABASE ORDER

You can disable relevance ranking of search results, in which case the Hitlist displays records in database order (i.e., the order in which they were added to the database). This is tantamount to sorting retrieved records by *record ID*—when added to a database, records are assigned distinct logical ID numbers. ID 1 indicates the first record added; ID 2, the second, etc.

To disable relevance ranking, select Sort Hitlist/Don't Sort from the Prefs menu.

INCLUDING WORDS FROM THE HITLIST IN A QUERY

Feedback searching requires that you be able to easily refine your query based on current search results; you can direct a search strategy by continually expressing your opinion of what is good or bad information. PL facilitates feedback searching by enabling you to quickly incorporate words from the Search Results Window in your search.

If, while viewing the Hitlist, you spot a word that you think would be a meaningful search term, you can append it to your current query by control-clicking it. With this technique, you can rapidly add to your query useful words that might not have originally occurred to you. Besides being fast, this technique saves you the trouble of typing new terms and reduces the likelihood of typographic errors.

DISPLAYING INTERMEDIATE SEARCH RESULTS

To monitor how PL evaluates a query when a search is executed, select Show Search Detail from the Prefs menu; this opens the Search Detail Window, an example of which is shown in Figure 16. This window always represents the most recently completed search.

Figure 16 The Search Detail Window

At the top of the Search Detail Window is a summary that shows a total number of records retrieved from each open database, along with a combined total of records retrieved from all open databases.

The window also displays, for individual databases, the results of each operation within a query. Immediately below the database heading, the default operator for the query is listed.

Note: You may notice PL has searched for variants of your query terms. This is due to search stemming—see *Understanding Search Stemming* on page 67. Also, if you search for the term *x**, where *x* is a single alphanumeric character, the Search Detail Window may post the combined results for the first few hundred hit words under an entry called *array*.

PRINTING THE HITLIST

To print Hitlist summaries:

1. Select Print Results from the File menu. The Print Results dialog, shown in Figure 17, will be displayed.

Figure 17 The Print Results dialog

2. By default, the entire Hitlist will be printed. If you want to print only a range of summaries, click the From radio button and enter the relevance ranking values of the first and last summaries in the desired range. By default, the range will begin with the rank of the record currently highlighted in the Hitlist.

3. Click the OK button.

CONFIGURING A PRINTOUT

To configure a print job—select printer, set page size/orientation, etc.—select Print Setup from the File menu.

PRINTING TO AN ASCII TEXT FILE

When printing the Hitlist, you have the option of saving the information to an ASCII text file. Moreover, if you print to a text file, you can choose either to save the data to a new file or to append it to an existing one.

To print the Hitlist to an ASCII text file:

1. Select Print Results from the File menu. The Print Results dialog, shown in Figure 17, will be displayed.

2. By default, the entire Hitlist will be printed. If you want to print only a range of summaries, click the From radio button and enter the relevance ranking values of the first and last summaries in the desired range. By default, the range will begin with the rank of the record currently highlighted in the Hitlist.

3. Click the Print to File checkbox. By default, the Hitlist will be saved to a new file, but you can elect to append the results to an existing text file by clicking that option's radio button.

4. Click the OK button. A standard file-saving dialog will be displayed, whereupon you can name and situate the new text file or specify the existing file to which the Hitlist information will be appended.

CUSTOMIZING THE DISPLAY OF SEARCH RESULTS

You can modify certain search results display parameters to conform to your personal preferences. To customize these parameters, select Search Results from the Prefs menu. The Search Results Preferences dialog, shown in Figure 18, will be displayed.

Figure 18 The Search Results Preferences dialog

To select top-to-bottom or side-by-side orientation of the Hitlist and bar chart, click the corresponding radio button under Window Alignment.

To automatically open the first record in the Hitlist upon completion of a search, click the Always Open First Record checkbox.

If, for each Hitlist entry, you want the corresponding record’s ID number displayed, click the Show Record IDs checkbox.

Note: When you are searching a virtual database, ID numbers displayed in the Hitlist may not be unique (i.e., the list may contain two records that have the same ID number in their respective databases).

To display field labels in the Hitlist, click the Show Field Labels checkbox. With this option enabled, field labels also appear in printouts of the Hitlist. By default, field labels are underlined. To have them enclosed in a box instead, click that option's radio button under Field Label Highlight.

To accept the current settings in the Search Results Preferences dialog, click the OK button.

Note: Changes to default search results preferences normally apply only to your current search session. To retain your current preference settings for use in future search sessions, select Save Settings from the Prefs menu.

Restoring Default Preference Settings

To restore all preferences to their system-wide default settings, select Default Settings from the Prefs menu.

SELECTING FIELDS FOR HITLIST DISPLAY

By default, the Hitlist will display only those fields with the Hitlist display attribute—see *Assigning Field Attributes* on page 207—that appear in the first 15 lines of a retrieved record. However, you can customize which fields are displayed in the Hitlist during a search session.

To modify Hitlist display of fields:

1. Select Fields from the Prefs menu. The Set Field Attributes dialog, shown in Figure 19, will be displayed. If you are searching a virtual database, a supplemental dialog will precede the Set Field Attributes dialog; select from that dialog the currently open database for which you want to modify Hitlist display of fields.

Figure 19 The Set Field Attributes dialog

Listed in the dialog's Individual Attributes window are the names of all fields defined for the current database, along with parenthetical abbreviations for their attributes. The presence of a *T* attribute indicates that a field is defined for display in the Hitlist.

2. Click the name of the field you want to modify. When the field is highlighted, its attributes are reflected in the checkboxes to the right of the Individual Attributes window, as shown in Figure 20.

Figure 20 Use the checkboxes to modify the selected field's display setting.

3. Click the Results List checkbox to enable or disable Hitlist display of the selected field.

4. Select other fields and modify their display settings, as desired.

5. When you are satisfied with the Hitlist display settings, click the OK button.

Note: Changes you make with the Set Field Attributes dialog apply only during the current search session.

ADJUSTING RELATIVE HITLIST & BAR CHART SIZES

To modify how space within the Search Results Window is allocated between the Hitlist and bar chart, drag the heavy border between them up or down; if the Hitlist and bar chart are displayed vertically, drag the border left or right.

SELECTING FONTS FOR THE SEARCH RESULTS AND SEARCH DETAIL WINDOWS

To select a font for text in the Hitlist or Search Detail Window:

1. Make the window active by clicking it or selecting it from the Windows menu.

2. From the Font menu, select the desired typeface, size, and formatting.

Note: The font selected for Hitlist display will also be used when the Hitlist is printed.

CHAPTER 6

VIEWING RECORDS

From this chapter you will learn how to interact with individual records in your Personal Librarian database. Topics discussed include opening records, navigating among retrieved records, browsing within databases, printing one or more records, annotating records, and establishing bookmarks for random access of individual records.

OPENING A RECORD…

…FROM THE HITLIST

To open a record from the Hitlist, double-click its summary or, having highlighted its summary, press ENTER.

Tip: By pressing SHIFT when opening a record from the Hitlist, you can close all other open records.

…FROM THE BAR CHART

To open a record from the bar chart, double-click its corresponding bar.

...WITH ITS HITLIST POSITION

To open a record with a specified rank in the Hitlist:

1. Select Record at Rank from the Go To menu (CTRL+R).

2. In the dialog that appears, enter the rank number of the desired record.

3. Click the OK button.

...WITH ITS ID NUMBER

You can open a record, whether it's in the Hitlist or not, by referencing its *record ID*—when added to a database, records are assigned distinct logical ID numbers. ID 1 indicates the first record added; ID 2, the second, etc. Each ID number is unique within a single database.

To open a record with a specified ID number:

1. Select Record ID from the Go To menu (Ctrl+D).

2. In the dialog that appears, enter the ID number of the desired record.

3. If you are searching a virtual database, the dialog will list all open databases; select the database containing the desired record. Because ID numbers are not necessarily unique in a virtual database, you must specify, along with the desired ID number, the database from which the record is to be retrieved.

4. Click the OK button.

Note: If you enter an invalid ID number (e.g., the record assigned that ID has been deleted from the database), the dialog's OK button will be disabled.

THE RECORD WINDOW

When you open a record, its contents will appear in a *record window*, an example of which is shown in Figure 21.

Figure 21 A record window

The key functional elements of a record window are the button bar and the title bar.

THE BUTTON BAR

At the extreme left of the button bar is the Print button, with which you can print the contents of the record window. See *Printing a Record* on page 71.

At the center of the button bar is a button with a single rightward-pointing arrow: the Next Hit button, with which you can quickly cover the distance between hit words. See *Rapidly Locating Hit Words in a Record* on page 68.

Flanking the Next Hit button are browse buttons, which you can use to move forward or backward, in ID order, through a database's records. See *Browsing Through Records in Database Order* on page 68.

At the right end of the button bar are buttons with which you can locate occurrences of a specified text string in a record. See *Locating a Word or Phrase Within a Record* on page 69.

THE TITLE BAR

A record window's title bar contains the following information about the record being displayed: the name of the parent database, the record's ID number, and, if the record is part of the current search results, the Hitlist rank. The Hitlist rank is expressed as *x of y*, where *x* is the record's position within a set of *y* retrieved records.

You have the option of modifying the information displayed in a record window's title bar. See *Customizing Record Window Display* on page 81.

HOW HIT WORDS ARE IDENTIFIED

When viewing a record which has been retrieved by a search, you will notice that some words in the record are underlined. This indicates that they are *hit words* (i.e., words that satisfied the conditions of the search and caused the record to be retrieved).

UNDERSTANDING SEARCH STEMMING

In a retrieved record's window, some words highlighted as hit words may vary somewhat from your original query terms. This is due to a process called *stemming*, wherein Personal Librarian strips query words of common suffixes that indicate plurality, verb tense, etc. After query words undergo stemming, a search is performed for variants of the resulting root words.

Stemming is performed by PL automatically, in order to accommodate the variety and imprecision of the English language. If you use the word *run* as a query word, stemming causes variants like *runs* and *running* to be sought, so you retrieve desirable information that would be overlooked under a strict interpretation of your query.

You have the option of turning stemming off—see *Setting the Default for Search Stemming* on page 275

NAVIGATING WITHIN AND BETWEEN RECORDS

MOVING WITHIN A RECORD

Personal Librarian's record windows conform to standard window navigation conventions. You can use the scroll bars, cursor control keys, and PAGEUP or PAGEDOWN to move up, down, or side-to-side within a record.

To move directly to the beginning of a record, press CTRL+HOME; to move to the end of a record, press CTRL+END.

RAPIDLY LOCATING HIT WORDS IN A RECORD

You can use a shortcut to rapidly locate and display the hit words in a retrieved record. The shortcut, a *jump-to-hit*, displays the next hit word not currently visible in the active record window.

To perform a jump-to-hit, make the desired record window active by clicking it, and select Next Hit from the Go To menu (F4) or click the Next Hit button, located at the center of the record window's button bar.

MOVING BETWEEN RETRIEVED RECORDS

With either the Search Results Window or a record window active, you can open the next record in the Hitlist by selecting Next Ranked Record from the Go To menu (F9). To open the preceding record in the Hitlist, select Previous Ranked Record from the Go To menu (SHIFT+F9).

Note: The Next Ranked Record and Previous Ranked Record commands are relative to the currently highlighted record in the Hitlist.

BROWSING THROUGH RECORDS IN DATABASE ORDER

You can view consecutive records in a database by *browsing*. To browse forward, open a record window and select Next Record ID from the Go To menu (F7) or click the Forward Browse button—a button with two rightward-pointing arrows—located in the record window's button bar. To browse backward, select Previous Record ID from the Go To menu (SHIFT+F7) or click the Reverse Browse button—the button with two leftward-pointing arrows.

The numbers beside the browse buttons indicate the ID numbers of the records logically adjacent to the currently displayed record.

Note: When you browse from one record to another, the original record window will be closed as the new one is opened.

LOCATING A WORD OR PHRASE WITHIN A RECORD

To find a specified word or phrase in a record:

1. Make the desired record window active by clicking it or selecting it from the Window menu.

2. Select String from the Go To menu (CTRL+F) or click the Find String button, located at the right end of the record window's button bar.

3. In the dialog that appears, enter the word or phrase that you want to find; wildcards—see *Using Wildcards* on page 108—may not be used to represent variable characters.

4. Click the OK button. If the text string is found, it will be displayed and highlighted in the record window; moreover, the Next String button, located at the extreme right of the record window's button bar, will become active.

5. To locate any subsequent occurrences of the string, select Next String from the Go To menu (F3) or click the Next String button.

Note: This feature is not case-sensitive; the string search is not restricted to occurrences of text that match your entry's combination of upper- and lower-case characters.

INCLUDING TEXT FROM A RECORD IN A QUERY

Feedback searching enables you to refine your query based on current search results; you can direct a search strategy by continually expressing your opinion of what is desirable about the information that has been retrieved. PL facilitates feedback searching by enabling you to incorporate words or phrases from record windows in your search.

INCLUDING A SINGLE WORD

If, while viewing a record, you spot a word that you think would be a meaningful search term, you can append it to your current query by control-clicking it. With this technique, you can rapidly add to your query useful words that might not have originally occurred to you. Besides being fast, this technique saves you the trouble of typing new terms and reduces the likelihood of typographic errors.

INCLUDING A PHRASE OR RANGE OF TEXT

To append a range of text from a record to your query:

1. In the record window, select the range of text you want to append to your query by dragging through it or by clicking and shift-clicking at, respectively, the beginning and end of the range.

2. Select Paste Selection to Query from the Edit menu (CTRL+=).

PRINTING RECORDS

PRINTING A RECORD

To print a record from the record window:

1. Make the desired record window active by clicking it or selecting it from the Window menu.

2. Select Print this Record from the File menu (Ctrl+P) or click the record window's Print button, located at the extreme left of the window's button bar.

To print a record from the Hitlist:

1. Click the Hitlist summary of the record to be printed.

2. Select Print Record at this Rank from the File menu (CTRL+P).

PRINTING A RANGE OF RECORDS

To print a range of records:

1. Select Print Records from the File menu. The Print Records dialog, shown in Figure 22, will be displayed.

Figure 22 The Print Records dialog

2. By default, the range you specify will be interpreted as a range of consecutive records from the database. To specify a range of records from the Hitlist, click the Rank in Hitlist radio button.
3. In the boxes under Record Range, enter the ID numbers of the first and last records in the desired range; if you are specifying a range from the Hitlist, enter the relevance ranking values of the first and last records.
4. If you are searching a virtual database, the dialog will list all open databases. If you are printing a range of records from a database—not from the Hitlist—select the database containing the desired records.
5. Click the OK button.

CONFIGURING A PRINTOUT

To configure a print job—select printer, set page size/orientation, etc.—select Print Setup from the File menu.

PRINTING TO AN ASCII TEXT FILE

When printing records, you have the option of saving the information to an ASCII text file. Moreover, if you print to a text file, you can choose to either save the data to a new file or append it to an existing one.

To print records to an ASCII text file:

1. Select Print Records from the File menu. The Print Records dialog, shown in Figure 22, will be displayed.

2. By default, the range you specify will be interpreted as a range of consecutive records from the database. To specify a range of records from the Hitlist, click the Rank in Hitlist radio button.

3. In the boxes under Record Range, enter the ID numbers of the first and last records in the desired range; if you are specifying a range from the Hitlist, enter the relevance ranking values of the first and last records.

4. If you are searching a virtual database, the dialog will list all open databases. If you are printing a range of records from a database—not from the Hitlist—select the database containing the desired records.

5. Click the Print to File checkbox. By default, the records will be saved to a new file, but you can elect to append them to an existing text file by clicking that option’s radio button.

6. Click the OK button. A standard file saving dialog will be displayed, whereupon you can name and situate the new text file or specify the existing file to which the records will be appended.

MANAGING MULTIPLE OPEN WINDOWS

A record window will remain open until you close it. After opening a succession of records from the Search Results Window, you will have several record windows open simultaneously. To avoid a cluttered workspace, you may want to use the following techniques for managing the display of multiple open windows.

CLOSING A WINDOW

To close the currently active window—record, image, advisor, Search Results, or Search Detail—select Close Window from the File menu (CTRL+F4) or double-click the window's control menu box.

Note: You cannot close the Search Window.

MOVING A WINDOW

You can move a window by dragging its title bar.

RESIZING A WINDOW

To resize a window, drag its border. You can enlarge a window to fill the PL workspace or reduce it to icon form by clicking, respectively, its maximize and minimize buttons.

Note: You cannot resize the Search Window.

ARRANGING WINDOWS AND ICONS

To overlap windows to so that the title bar of each is visible, select Cascade from the Window menu.

To arrange windows top-to-bottom so that each window is visible and none overlap, select Tile Horizontally from the Window menu.

To arrange windows side-by-side so that each window is visible and none overlap, select Tile Vertically from the Window menu.

To arrange icons—windows that have been minimized—so that they line up neatly and do not overlap, select Arrange Icons from the Window menu.

ANNOTATING RECORDS

Using Personal Librarian's annotation feature, you can attach your own notation to the text in a record. Because these electronic "sticky notes" are unique for each user, your annotations will be personalized.

CREATING ANNOTATIONS

To add a note to a record:

1. In the record window, make visible the line of text to which you want to attach notation.

2. Move the cursor into the margin to the left of the text.

3. When the cursor changes to resemble a pencil, click the margin. The dialog shown in Figure 23 will be displayed.

Figure 23 The Annotation dialog

4. Type your note in the dialog's text window.

5. Click the Save button.

After creating a notation, you will notice that its location in a record is indicated by a marker, as shown in Figure 24.

Figure 24 A triangle marks an annotation's location.

You may also notice a blackened area at the left end of the band immediately under the record window's button bar, as shown in Figure 25.

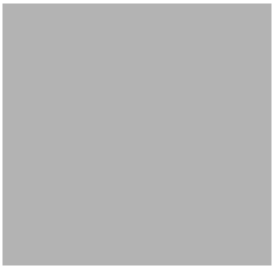

Figure 25 This mark indicates the presence of annotations in the record.

This indicates that a record contains one or more annotations. Even if no annotation markers are currently visible in the record window, this black band will alert you to the presence of notes within the record.

VIEWING AND EDITING ANNOTATIONS

To view an annotation, click its marker.

You can edit a note while it is displayed in the Annotation dialog. To clear the dialog’s text window, click the Clear button.

To delete an annotation, open it and click the Annotation dialog’s Delete button.

Note: You can assign only a single annotation to each line in a record.

PLACING BOOKMARKS IN RECORDS

With PL's bookmark feature, you can mark your current position within a database or record, in order to return to that point in the future. Activating a bookmark does more than just instantly display the desired record; it displays the position within the record that was visible when the bookmark was defined.

CREATING A BOOKMARK

To create a bookmark:

1. Open the desired record

2. In the record window, make visible the portion of the record for which a bookmark is to be defined.

3. Select Bookmarks/New from the File menu.

4. In the dialog box that appears, enter a description of the material being marked. Bookmark names must be unique; you cannot enter a description that duplicates an existing bookmark.

Note: If you have any record text highlighted, it will be entered as the default bookmark description.

5. Click the OK button.

ACTIVATING A BOOKMARK

When you use a bookmark to open a record, the record window will display the part of the record that was visible when the bookmark was created.

To open a bookmarked record:

1. Select Bookmark from the Go To menu. A dialog listing all of your bookmarks, similar to Figure 26, will be displayed.

Figure 26 The Go to Bookmark dialog

2. Click the desired bookmark.

3. Click the OK button. If the database containing the bookmarked record is not currently open, a supplemental dialog will prompt you to confirm that you want to open it.

RENAMING A BOOKMARK

To rename a bookmark:

1. Select Bookmarks/Rename from the File menu. A dialog similar to Figure 27 will be displayed.

Figure 27 The Rename Bookmark dialog

2. In the list of existing bookmarks, click the one you want to rename.

3. Type a new description in the Enter New Name text box.

4. Click the OK button.

DELETING A BOOKMARK

To delete a bookmark:

1. Select Bookmarks/Delete from the File menu.

2. In the dialog that appears, click the bookmark you want to delete.

3. Click the OK button.

CUSTOMIZING RECORD WINDOW DISPLAY

You can modify certain record display parameters to conform to your personal preferences. To customize these parameters, select Record Display from the Prefs menu. The Record Display Preferences dialog, shown in Figure 28, will be displayed.

Figure 28 The Record Display Preferences dialog

When you open a record in the Hitlist, its window by default displays the beginning of the record. To have the first hit word displayed instead, click the Open Record at First Hit Term radio button.

Note: When you open a record by its ID number or by browsing, this setting has no effect; the beginning of the record will be displayed.

By default, when you browse from one record to another, the first window is closed as the second one is opened; when you open records by any other means, any currently open records will remain open until you close them manually. If, when opening a record, you want any currently open records closed, click the Close Other Records radio button. To have open records left open when you open a record by any means—even by browsing—click the Leave Other Records Open radio button.

Tip: When opening a record from the Hitlist, you can temporarily override the Multiple Records setting by shift-clicking the record's Hitlist summary or pressing SHIFT+ENTER.

By default, hit words are underlined in retrieved records. To have them enclosed in a box instead, click that option's radio button under Highlighting. To disable hit word highlighting altogether, click the Highlight Hit Terms checkbox.

When you select a font size larger than 12 points for record windows, there is a risk that some text may be truncated when the record is printed. To have a warning of this possibility displayed when fonts larger than 12 points are selected for record windows, click that option's checkbox.

To customize the information displayed in a record window's title bar, click the Record Title button. The Record Title Components dialog, shown in Figure 29, will be displayed.

Figure 29 The Record Title Components dialog

With the Record Title Components dialog, you can define a prefix for the title bar and determine what combination of record-specific information—parent database name, Hitlist rank, and ID number—is displayed after the prefix. Your title bar preferences will be reflected in the Sample Record Title. To accept the current settings in the Record Title Components dialog, click the OK button.

To accept the current settings in the Record Display Preferences dialog, click the OK button.

Note: Changes to default record display preferences normally apply only to your current search session. To retain your current preference settings for use in future search sessions, select Save Settings from the Prefs menu.

Restoring Default Preference Settings

To restore all preferences to their system-wide default settings, select Default Settings from the Prefs menu.

SELECTING FIELDS FOR RECORD DISPLAY

By default, record windows will display only those fields with the record display attribute—see *Assigning Field Attributes* on page 207. However, you can customize which fields are displayed in record windows during a search session.

To modify record display of fields:

1. Select Fields from the Prefs menu. The Set Field Attributes dialog, shown in Figure 30, will be displayed. If you are searching a virtual database, a supplemental dialog will precede the Set Field Attributes dialog; select from that dialog the currently open database for which you want to modify record display of fields.

Figure 30 The Set Field Attributes dialog

Listed in the dialog's Individual Attributes window are the names of all fields defined for the current database, along with parenthetical abbreviations for their attributes. The presence of a *D* attribute indicates that a field is defined for display in record windows.

2. Click the name of the field you want to modify. When the field is highlighted, its attributes are reflected in the checkboxes to the right of the Individual Attributes window, as shown in Figure 31.

Figure 31 Use the checkboxes to modify the selected field's display setting.

3. Click the Display checkbox to enable or disable record window display of the selected field.

4. Select other fields and modify their display settings, as desired.

5. When you are satisfied with the record window display settings, click the OK button.

Note: Changes you make with the Set Field Attributes dialog apply only during the current search session.

MODIFYING DISPLAY OF FIELD LABELS IN RECORD WINDOWS

By default, field labels are displayed in record windows. However, you can disable this display of field labels during a search session.

To modify field label display in record windows:

1. Select Fields from the Prefs menu. The Set Field Attributes dialog will be displayed.

2. Click the Show Field Labels in Records checkbox.

3. Click the OK button.

Note: Changes you make with the Set Field Attributes dialog apply only during the current search session.

SELECTING A RECORD WINDOW FONT

To select a font for record window text:

1. Make a record window active by clicking it or selecting it from the Windows menu.

2. From the Font menu, select the desired typeface, size, and formatting.

Note: A font selection made for a single record window will apply to all record windows. The font selected for record window display will also be used when records are printed.

CHAPTER 7

USING ADVISORS

You can think of Personal Librarian's advisors as on-line assistants. In addition to giving you added insight into the nature of a database's contents, they provide you with powerful feedback searching tools for refining your search strategy.

ACTIVATING THE ADVISORS

The Search Window features icons, shown in Figure 32, for PL’s advisors. You can activate an advisor by clicking its icon. If an advisor’s icon is blank, that advisor is not currently enabled.

Figure 32 The Search Window features an icon for each advisor.

Not all advisors will be enabled when you start PL. The Relate and Fuzzy Match Advisors become active only when query text is entered in the Search Window. The Thesaurus Advisor is enabled when you open a database for which a thesaurus has been defined—see *Defining a Thesaurus* on page 215. Once you have executed a search, the History Advisor will become active.

LAUNCHING A DETOUR SEARCH

During feedback searching with PL's advisors, you may encounter a word for which you want to perform a search. If you want to see what kind of records will be retrieved for that word, but don’t want to discard the query currently in the Search Window, you can perform a *detour search*.

A detour search is essentially a departure from your current search strategy; you can see what kind of information is retrieved for a particular word, then return to using your previous query, which remains unaltered.

To execute a detour search, double-click on the desired word in an advisor window. A new Hitlist is generated, but the contents of the Search Window do not change.

Note: Advisor windows are the only windows from which you can perform detour searching.

VIEWING A DATABASE'S DICTIONARY

By clicking the Dictionary Advisor icon, you open a window containing an alphabetical listing of all unique terms in the currently open database’s index; an example is shown in Figure 33. If you are searching a virtual database, an intermediate dialog will prompt you to select the database for which you want to open a Dictionary window; a Dictionary window can be opened for each open database.

Figure 33 A Dictionary Advisor window

Beside each index term is a posting number, which indicates the number of times the term occurs within the database.

Note: The Dictionary Advisor window does not display stopwords or words occurring in fields defined as non-indexed by the database administrator—see *Assigning Field Attributes* on page 207.

The Dictionary Advisor is useful for verifying the existence and popularity of terms within a database; if a search for a particular word yields no hits, and you believe that the word is in the database, the Dictionary Advisor can confirm or deny your suspicions. This advisor also shows what variants of a given word exist in a database.

You can use the Dictionary Advisor to see if a particular word would be a meaningful query term. If a word is shown to have an exceptionally large number of occurrences within a database, it may be unsuitable as a query term, because using it may cause retrieval of the majority of the database's records.

If you activate the Dictionary Advisor while no query terms are in the Search Window, the first database term that begins with an alphabetic character will be highlighted in the advisor window. If you have part of your query highlighted, the Dictionary window will open to where the first selected word does, or would, appear.

In addition to using the scroll bars, cursor control keys, and PAGEUP/PAGEDOWN keys, you can navigate through a Dictionary window by typing a character or word while the window is active. Doing this will reposition the window to display the first index term, if any, that matches what you have typed.

To execute a detour search for a word in a Dictionary Advisor window, double-click the word or, having highlighted the word, press ENTER.

INCLUDING DICTIONARY ADVISOR ENTRIES IN YOUR QUERY

To append a word in a Dictionary Advisor window to your current query, control-click the word.

FINDING WORDS RELATED TO YOUR QUERY

The Relate Advisor is enabled only when you have entered query text in the Search Window; if you have part of your query highlighted, the advisor's operation will be based only on the selected words. When you activate the Relate Advisor, it dynamically generates a list of words in the currently open database that are statistically related to the words in your query; words are deemed related if they exhibit a significant degree of co-occurrence within records. If you are searching a virtual database, a Relate Advisor window, an example of which is shown in Figure 34, displays a merged list of related words from all open databases.

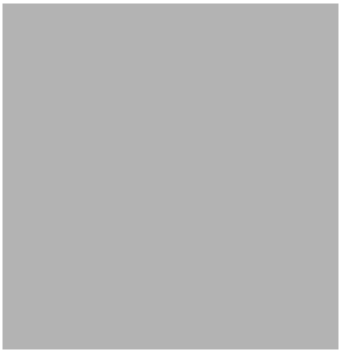

Figure 34 A Relate Advisor window

A Relate Advisor window lists up to 50 terms related to the words in your query, ranked by degree of co-occurrence. The window's title bar displays the query terms that were evaluated and the number of related terms found is shown at the bottom of the window.

You can use the Relate Advisor to identify potentially valuable query words that might not have originally occurred to you. In the feedback searching process, this advisor can suggest terms with which you can refine your query.

For example, if you were searching a database of news articles and used the Relate Advisor for the following query:

```
Saddam Hussein
```

a list similar to Figure 34 might be generated. The list of related terms contains some words that might not be immediately obvious to you—Kurds, supergun, intervention, etc.—but could be useful in a search.

You can see how the Relate Advisor's feedback can aid you in refining a query by suggesting valuable search terms that you otherwise might not consider.

To execute a detour search for a word in a Relate Advisor window, double-click the word or, having highlighted the word, press ENTER.

Note: When finding related words in a database's records, the Relate Advisor considers all indexed fields, not just the contents of fields defined as searchable by the database administrator—see *Assigning Field Attributes* on page 207.

INCLUDING RELATE ADVISOR ENTRIES IN YOUR QUERY

To append a word in a Relate Advisor window to your current query, control-click the word.

FUZZY MATCHING: IDENTIFYING WORDS WITH SIMILAR SPELLING

The Fuzzy Match Advisor is enabled only when you have entered query text in the Search Window. When you activate this advisor, it displays a list of words in the currently open database that have spelling similar to the first term in your query; an example is shown in Figure 35. If you have part of your query highlighted, the advisor will try to match the first selected term. If you are searching a virtual database, an intermediate dialog will prompt you to select the database in which you want to find similarly-spelled words.

Figure 35 A Fuzzy Match Advisor window

A Fuzzy Match Advisor window's title bar displays the query term that was evaluated and the number of similarly-spelled terms found is shown at the bottom of the window.

The Fuzzy Match Advisor can be helpful in finding word variants or misspellings in a database—the result of typographical mistakes or errors during OCR (optical character recognition) processing. Also, if you have doubts about the spelling of a query word, you can use the Fuzzy Match Advisor to verify it. For example, if you were searching a database of news articles and wanted to verify spelling of the name of the leader of South Africa's Inkatha Freedom Party, you could enter the following approximation:

```
Butalezi
```

and activate the Fuzzy Match Advisor, which would display a list similar to that shown in Figure 35.

To execute a detour search for a word in a Fuzzy Match Advisor window, double-click the word or, having highlighted the word, press ENTER.

Note: If you want adjust the selectivity of PL's fuzzy matching algorithm, see *Adjusting Fuzzy Matching Selectivity* on page 276.

INCLUDING FUZZY MATCH ADVISOR ENTRIES IN YOUR QUERY

To append a word in a Fuzzy Match Advisor window to your current query, control-click the word.

VIEWING A DATABASE'S THESAURUS

The Thesaurus Advisor is enabled only when you open a database for which a thesaurus has been defined—see *Defining a Thesaurus* on page 215. When you activate the Thesaurus Advisor, it opens a window that displays the contents of the database's thesaurus, similar to the example shown in Figure 36. If you are searching a virtual database and multiple open databases have thesauri defined, an intermediate dialog will prompt you to select the database for which you want to view the associated thesaurus.

Figure 36 A Thesaurus Advisor window

A Thesaurus Advisor window displays main entries on the left, followed by their respective synonyms.

In addition to using the scroll bars, cursor control keys, and PAGEUP/PAGEDOWN keys, you can navigate through a Thesaurus window by typing a character or word while the window is active. Doing this will reposition the window to the first main entry, if any, that matches what you have typed.

To execute a detour search for a synonym in a Thesaurus Advisor window, double-click the word. Double-clicking a main entry will execute a detour search for all of its synonyms.

INCLUDING THESAURUS ADVISOR ENTRIES IN YOUR QUERY

To append a word in the Thesaurus Advisor window—main entry or synonym—to your current query, control-click the word. If you use this technique to include a main entry in your query, the thesaurus operator—see *Using a Thesaurus* on page 120—will be automatically appended to it in the Search Window.

REUSING PREVIOUS QUERIES

The History Advisor is enabled after you execute a search. When you activate this advisor, it opens a window, an example of which is shown in Figure 37, that lists all queries used during the current search session.

Figure 37 A History Advisor window

You can use the History Advisor to review your search strategy or to recycle previous queries.

To execute a detour search for a query in the History Advisor window, double-click the query.

INCLUDING HISTORY ADVISOR ENTRIES IN YOUR QUERY

To append an entry in the History Advisor window to your current query, control-click the entry; the entire query will be added to the Search Results Window.

SAVING A SEARCH HISTORY

If you like the results produced by your current search, you can save the search history for reuse in future search sessions. If you have to periodically repeat searches in a database that changes over time, this feature will save you the trouble of rebuilding routine queries.

To save your current search history:

1. Select History/Save from the File menu. A standard file-saving dialog will be displayed.

2. Use the dialog to name and situate the history file. By default, the history file will be saved to PL's `\personal` subdirectory.

3. Click the OK button.

RESTORING A PREVIOUS HISTORY

If you have a previously saved search history, you can import it for use in your current search session.

To restore a history from an earlier search session:

1. Select History/Restore from the File menu.

2. A dialog will prompt you to indicate whether you want the saved history to replace or supplement your current search history. If you want to replace your current history with the saved one, click the Replace button. If you want the saved history appended to your current one, click the Append button.

3. A standard file-opening dialog will be displayed. Use it to find and select the desired history file.

4. Click the OK button.

CUSTOMIZING ADVISOR WINDOW DISPLAY

You can modify certain advisor display parameters to conform to your personal preferences. To customize these parameters, select Advisors from the Prefs menu. The Advisor Preferences dialog, shown in Figure 38, will be displayed.

Figure 38 The Advisor Preferences dialog

By default, multiple windows for a single advisor can be open simultaneously (e.g., two or more Relate Advisor windows can be open at the same time). To disable this condition, click the Allow Duplicate Advisor Windows checkbox.

By default, an advisor window opens with the same size and position it had when it was last closed. To disable this condition, click the Remember Advisor Positions checkbox.

When advisor windows propagate, they are arranged by default in a tiled formation. To have advisor windows overlapped with their title bars visible, click the Cascade radio button.

To accept the current settings in the Advisor Preferences dialog, click the OK button.

Note: Changes to default advisor display preferences normally apply only to your current search session. To retain your current preference settings for use in future search sessions, select Save Settings from the Prefs menu.

Restoring Default Preference Settings

To restore all preferences to their system-wide default settings, select Default Settings from the Prefs menu.

SELECTING ADVISOR WINDOW FONTS

To select a font for advisor window text:

1. Make an advisor window active by clicking it or selecting it from the Windows menu.

2. From the Font menu, select the desired typeface, size, and formatting.

Note: A font selection made for a single advisor window type will apply to all windows of that type.

CHAPTER 8

USING OPERATORS

This chapter discusses the functions of Personal Librarian's query language operators, which you can use to customize the processing of search queries.

WHAT IS A QUERY OPERATOR?

A query operator consists of one or more characters that act not as a query word, but as an instruction on how a query is to be processed. An operator can work at word-level, where it applies to a single query term, or at query-level, where its presence affects the processing of the entire query.

Note: When you apply a query operator to a stopword (e.g., `the!`), the operator is ignored.

FUNCTIONAL OVERVIEW

Table 2 Query Operator Functions

Operator	Function
default operator	Used in absence of explicit operator—see page 34
concept operator	Searches for idea a word represents—see page 104
OR operator	Imposes Boolean OR logic—see page 105
AND operator	Imposes Boolean AND logic—see page 105
NOT operator	Imposes Boolean NOT logic—see page 105
single character wildcard operator	Matches single variable character—see page 108
optional character wildcard operator	Matches single variable character or no character—see page 109
string wildcard operator	Matches variable character string—see page 110
proximity operator	Searches for unidirectional proximity of two words—see page 111
adjacency operator	Searches for unidirectional adjacency of two words—see page 112
near operator	Searches for bidirectional proximity of two words—see page 113
same paragraph operator	Searches for bidirectional paragraph-based proximity of two words—see page 114
notsame paragraph operator	Imposes Boolean NOT logic on search for directional paragraph-based proximity of two words—see page 115

Table 2 Query Operator Functions

Operator	Function
exact match operator	Searches for exact word match—see page 116
exact phrase operator	Searches for exact phrase match—see page 117
stemming operator	Forces search for word variants—see page 118
fuzzy search operator	Searches for similar spelling—see page 118
thesaurus operator	Searches for defined synonyms—see page 120
document operator	Searches for similar records—see page 123
query operator	Reiterates previous query—see page 124
word-level field restriction operator	Restricts word search to field(s)—see page 125
query-level field restriction operator	Restricts word full query search to field(s)—see page 126
equivalence operator	Searches field for specified value—see page 129
range operator	Searches field for specified value range—see page 129
scope of operation delimiters	Impose algebraic grouping logic—see page 133

WORD-LEVEL CONCEPT SEARCHING

With the *concept operator*—the exclamation point—you can perform a concept search for an individual query word, as opposed to using the Search Window's Concept button, which considers your entire query. See *Concept Search: Exploring a General Idea* on page 39.

The concept operator dynamically generates a list of terms statistically related to the word to which the operator has been appended; the list of related terms is not displayed as it is when you use the Concept button. A search is then performed for the original query word and the 20 most significant related terms.

Syntax:

```
word!
```

Example:

```
tribal! politics in South Africa!
```

In this example, concept searches will be performed for the words *tribal* and *Africa*, but not for *politics* or *South*.

If search stemming is on, as it is by default—see *Understanding Search Stemming* on page 67, the following query:

```
run!
```

will search for terms related to variants of the word—*run*, *runs*, *running*, etc. If stemming is off, the query will search for terms related only to *run*.

If search stemming is on, the following query:

```
run"!
```

will search for terms related only to *run*.

Note: When generating a related word list, PL considers the contents of all indexed fields, not just the contents of fields that are defined as searchable. See *Assigning Field Attributes* on page 207.

SEARCHING WITH BOOLEAN LOGIC

Personal Librarian supports use of the *Boolean logic operators*, OR, AND, and NOT. In the absence of explicit operators, PL uses the default OR in evaluating queries. If you want to change the default operator, see *Customizing Search Performance* on page 41.

Tip: You should rely on natural language queries; they will usually retrieve useful information that Boolean queries can exclude. With relevance ranking at work, you need not be concerned with reducing the number of retrieved records. Consider using Boolean queries only when a precisely restricted search is appropriate.

Note: You can enter query operators in upper- or lower-case. They are capitalized in the following examples only for purposes of clarity.

THE OR OPERATOR:

The *OR operator* searches for records that contain either of the words it separates.

Syntax:

word1 OR *word2*

Example:

```
heaven OR hell
```

This query will retrieve any record that contains an occurrence of at least one of the two words.

Note: By default, OR is used as the default operator; you don't need to enter it explicitly unless a different default operator has been defined.

THE AND OPERATOR:

The *AND operator* searches for records that contain both of the words it separates.

Syntax:

word1 AND *word2*

Example:

```
Burke AND Hare
```

This query will retrieve only those records that contain both of the words.

THE NOT OPERATOR:

The *NOT operator* can be used in binary or unary form. In binary form, it searches for records that contain the query term that precedes it but do not contain the term that follows it. In unary form, it searches for all records that do not contain the term that follows it.

Syntax (binary):

```
word1 NOT word2
```

Syntax (unary):

```
NOT word
```

Examples:

```
media NOT television
NOT Gordian
```

The first query will retrieve only records that contain *media* and are without any occurrences of *television*.

The second query will retrieve any record that contains no occurrences of *Gordian*.

COMBINING BOOLEAN OPERATORS:

You can combine Boolean operators in a single query.

Example:

```
New AND York NOT City
```

This query will retrieve records that contain both *New* and *York*, but do not contain *City*.

CAUTION: When combining Boolean operators in a query, you should keep in mind the logic imposed on the query by PL's operator precedence rules—see *Operator Precedence Rules* on page 132. Precedence rules govern the order in which a query's operations are processed; the relative precedence of different operators may cause a query to be processed with logic that is not immediately obvious.

Tip: When combining operators in a query, you can control the order in which operations within the query are processed by using parentheses as delimiters—see *Composing Complex Queries* on page 133.

USING WILDCARDS

With *wildcard operators*, you can search for words with variable characters. You can position wildcard operators at the beginning, middle, or end of a query word, and you can combine them within a word (e.g., `?ffect*`).

Note: Search stemming is disabled for a query term when you use a wildcard at the beginning or in the middle of the term—see *Understanding Search Stemming* on page 67.

MATCHING A SINGLE CHARACTER

You can use the *single character wildcard*—the question mark—to represent a single variable character in a query.

Syntax:

```
string?
str??ing
???string
```

Examples:

```
medic???
1?4
R??2
```

The first query will search for terms like *medicine, medicate,* etc.

The second query will search for terms like *114*, *144*, etc.

The third query will search for terms like *R2D2,* etc.

MATCHING ONE OR ZERO CHARACTERS

The *optional character wildcard*—the dollar sign—is similar to the single character wildcard, but it represents one *or zero* variable characters.

Syntax:

string$
str$$*ing*
$$$*string*

Examples:

```
colo$r
V$TOL
electron$$$
```

The first query will search for terms like *color, colour,* etc.

The second query will search for terms like *VTOL, VSTOL*, etc.

The third query will search for terms like *electron, electrons, electronic, electronics,* etc.

MATCHING A CHARACTER STRING

You can use the *string wildcard*—the asterisk—to represent a variable string of zero or more characters in a query. The string wildcard is equivalent to an infinite series of adjacent optional character wildcards.

Syntax:

```
string*
string1*string2
*string
```

Examples:

```
medic*
m*n
*ane
1*5
```

The first query will search for terms like *medic*, *medics*, *medical*, *medicine*, *medicate*, *medically*, *medication*, etc.

The second query will search for terms like *man*, *men*, *mean*, *moon*, *melon*, *moron*, *maroon*, *Manhattan*, *manifestation*, etc.

The third query will search for terms like *bane*, *lane*, *crane*, *plane*, *profane*, *insane*, *membrane*, etc.

The fourth query will search for terms like *15*, *175*, *1995*, etc.

Note: A query consisting of only the string wildcard operator will return a Hitlist comprising all records in a database; the records will have identical relevance ranking scores and will be sorted in descending ID order.

SEARCHING FOR WORDS THAT ARE NEAR EACH OTHER

Note: You can enter query operators in upper- or lower-case. They are capitalized in the following examples only for purposes of clarity.

THE PROXIMITY OPERATOR

You can use the *proximity operator* to search for word pairs in which the pair's second term occurs within a specified number of words after the first.

Note: The proximity operator does not work across field boundaries; you cannot use it to search for a word pair in which the words occupy separate fields within a record.

Syntax:

```
word1 W/n word2
```

where *n* is the number of words within which *word2* must occur after *word1*. The variable *n* can be any integer greater than 0—within reason, that is; the actual limit is between 1,000,000,000 and 2,000,000,000. Stopwords and punctuation do not count as words in the range specified by *n*.

The proximity operator is unidirectional from left to right. It retrieves only records in which *word2* occurs within *n* words after *word1*. Occurrences of *word1* within *n* words after *word2* are not considered hits.

Example:

```
amphibian W/5 DNA
```

This query will retrieve records in which *DNA* occurs within five words after *amphibian*.

THE ADJACENCY OPERATOR

The *adjacency operator*—ADJ—is equivalent to a proximity operator with a defined range of one word (i.e., W/1). Certain punctuation marks—hyphen, apostrophe, comma, and period—function as adjacency operators when they appear in the middle of a character string; they do not function as such if immediately preceded or followed by a character not recognized by PL during indexing (e.g., a space).

Note: The adjacency operator does not work across field boundaries; you cannot use it to search for a word pair in which the words occupy separate fields within a record.

Syntax:

```
word1 ADJ word2
str-ing
str'ing
str,ing
str.ing
```

The adjacency operator is unidirectional from left to right. It retrieves only records in which *word2* follows *word1*. Occurrences of *word1* that follow *word2* are not considered hits.

Examples:

```
great ADJ white
337-81-4417
O'Hara
2,001
Carcharodon.carcharias
```

THE NEAR OPERATOR

The *near operator* duplicates the functions of the proximity and adjacency operators, with one exception: it is bidirectional—you can use it to search for word pairs in which the second term occurs within a specified number of words before or after the first.

If you specify a word range with it, the near operator functions as a bidirectional proximity operator—see *The Proximity Operator* on page 111. If no word range is specified, it serves as a bidirectional adjacency operator—see *The Adjacency Operator* on page 112.

Note: The near operator does not work across field boundaries; you cannot use it to search for a word pair in which the words occupy separate fields within a record.

Syntax:

word1 NEAR/*n* *word2*
word1 NEAR *word2*

where *n* is the number of words within which *word1* must occur before or after *word2*. The variable *n* can be any integer greater than 0—within reason, that is; the actual limit is between 1,000,000,000 and 2,000,000,000. Stopwords and punctuation do not count as words in the range specified by *n*.

Examples:

```
Whitewater NEAR/5 indictment
tax NEAR increase
```

The first query will retrieve records in which *Whitewater* occurs within five words of *indictment*.

The second query will retrieve records in which *tax* occurs immediately before or after *increase*.

THE SAME PARAGRAPH OPERATOR

You can use the *same paragraph operator* to search for word pairs that occur—in any order—in the same paragraph or in a specified range of paragraphs. In a text file, paragraphs are delineated by two consecutive end-of-line characters.

Note: The same paragraph operator does not work across field boundaries; you cannot use it to search for a word pair in which the words occupy separate fields within a record.

Syntax:

```
word1 SAME/n word2
word1 SAME word2
```

where *n* is the number of paragraphs within which *word1* must occur before or after *word2*. The variable *n* can be any integer greater than 0—within reason, that is; the actual limit is between 1,000,000,000 and 2,000,000,000. Stopwords and punctuation do not count as words in the range specified by *n*.

Note: If the database being searched does not have the statement `index_size=full` in its definition file—see *Enabling Same Paragraph Proximity Searching* on page 217—the same paragraph operator will retrieve records in which *word1* and *word2* occupy the same field.

Examples:

```
Africa SAME/3 Congo
Australia SAME outback
```

The first query will retrieve records in which both *Africa* and *Congo* occur within a range of no more than three paragraphs.

The second query will retrieve records in which *Australia* occurs in the same paragraph as *outback*.

THE NOTSAME PARAGRAPH OPERATOR

You can use the *notsame paragraph operator* to search for records that contain paragraphs in which the query term that precedes the operator occurs and the term that follows the operator does not. In a text file, paragraphs are delineated by two consecutive end-of-line characters.

Note: The notsame paragraph operator does not work across field boundaries; it considers only paragraphs that occupy the same field within a record.

Syntax:

word1 NOTSAME *word2*

Note: If the database being searched does not have the statement `index_size=full` in its definition file—see *Enabling Same Paragraph Proximity Searching* on page 217—the notsame paragraph operator will retrieve records in which *word1* occurs, excluding those in which *word1* and *word2* occupy the same field.

Example:

```
plague NOTSAME buboes
```

This query will retrieve records that contain *plague*, excluding those in which *plague* and *buboes* occupy the same paragraph.

SEARCHING FOR AN EXACT WORD

If search stemming is on, as it is by default, you can override it for an individual query term by using the *exact match operator*—the double quotation mark—to force a search for only the exact word. Variants for a term to which the operator has been appended will not be sought. See *Understanding Search Stemming* on page 67.

Syntax:

```
word"
```

CAUTION: Do not use the exact match operator with a query term that includes wildcard operators—see *Using Wildcards* on page 108; this causes the wildcards to be interpreted literally, rather than as variables.

Example:

```
run"
```

This query will search only for *run*; variants like *runs*, *runner*, and *running* will not be sought.

SEARCHING FOR AN EXACT PHRASE

The *exact phrase operator* combines characteristics of the adjacency and exact match operators—see *The Adjacency Operator* on page 112 and *Searching for an Exact Word* on page 116. By bordering a sequence of query terms with this operator—the single quotation mark—you can search for records that contain exact matches of that phrase.

Syntax:

```
'word1 word2 word3'
```

This syntax is equivalent to the following:

```
word1" ADJ word2" ADJ word3"
```

CAUTION: Do not use the exact phrase operator with a query term that includes wildcard operators— see *Using Wildcards* on page 108; this causes the wildcards to be interpreted literally, rather than as variables.

Examples:

```
'Project Blue Book'
'Council on Foreign Relations'
```

The first query will retrieve records that contain the phrase *Project Blue Book*.

The second query will retrieve records that contain the phrase *Council on Foreign Relations*. Because *on* is a stopword, occurrences in records of the phrase *Council on Foreign Relations* are read by PL as *Council Foreign Relations*, which satisfies the conditions of the query.

Note: If you include in the enclosed phrase any operator that is not defined as a stopword—see *Stopword Customization* on page 161—it will be interpreted as a query term, not as an operator.

SEARCHING FOR WORD VARIANTS

If search stemming is turned off, you can use the *stemming operator*—the plus sign—to force stemming of an individual query term. PL will search for variants of the term to which the operator has been appended. See *Understanding Search Stemming* on page 67.

Syntax:

```
word+
```

Example:

```
run+
```

This query will search for *run* and alternate forms of the word, such as *runs*, *runner,* and *running*.

SEARCHING FOR WORDS WITH SIMILAR SPELLING

You can use the *fuzzy search operator* to retrieve records that contain words with spelling similar to that of a query term. That part of a query term that will be matched exactly is determined by the placement of the operator—a tilde—within the word; the character string that precedes the operator will be anchored, meaning that it must be identical in all terms qualified by PL as having similar spelling.

The fuzzy search operator can be helpful in finding word variants or misspellings in a database—the result of typographical mistakes or errors during OCR (optical character recognition) processing.

Note: Regardless of where you position the fuzzy search operator in a query term, the first character of the term will be anchored.

Syntax:

```
string1~string2
```

where *string1* is the part of the query term that must be matched exactly and *string2* is the part that will be matched with similar spelling patterns.

Example:

```
Kir~gizia
cyclo~hexene
```

The first query will retrieve records that contain *Kirgizia* or alternate spellings of the word, such as *Kirghizia*.

The second query will retrieve records that contain *cyclohexene* or similarly spelled words like *cyclooctene* and *cyclohexane.*

CAUTION: Use of this operator may result in a search for words relating to entirely different subjects. For example, the query `anti~` might search for *antics*, *antique*, and *Antigua*.

Note: If you want to adjust the selectivity of PL's fuzzy matching algorithm, see *Adjusting Fuzzy Matching Selectivity* on page 276.

USING A THESAURUS

You can use the *thesaurus operator*—the "at" sign (@)—to search for predefined synonyms for a query term. By appending this operator to a query term, you cause PL to search not for the term, but for synonyms stored in a thesaurus file—see *Defining a Thesaurus* on page 215.

If a query term to which the thesaurus operator has been appended is not a main entry in a currently open database's thesaurus, or if a thesaurus has not been defined for any open databases, the operator will be ignored and the term will be processed normally.

Note: Because concept searching and the Relate Advisor make it easy to identify and search with words related to query terms, you may not need to use a thesaurus—see *Concept Search: Exploring a General Idea* on page 39, *Word-Level Concept Searching* on page 104, and *Finding Words Related to Your Query* on page 90. However, a thesaurus is ideal for databases that include many acronyms; thesaurus searching can minimize the need to correctly interpret acronyms or manually enter the words they represent.

Syntax:

```
word@
```

Example:

```
FBI@ director
```

If the thesaurus defined for an open database contains the following entries:

```
CIA   central intelligence agency CIA
FBI   federal bureau investigation FBI
NATO  north atlantic treaty organization NATO
congress    {congress house senate legislature}
```

PL will replace *FBI@* with that term's synonyms—*federal*, *bureau*, *investigation*, and *FBI*—and process the query as:

```
federal bureau investigation FBI director
```

Note: If you want PL to automatically perform thesaurus expansion for all query terms—to implicitly append the thesaurus operator to each word—see *Activating Automatic Thesaurus Operation for All Query Terms* on page 276.

UNDERSTANDING THESAURUS ENTRY FORMATS

When using the Thesaurus Advisor to view the contents of a database's thesaurus—see *Viewing a Database's Thesaurus* on page 94—you will see synonyms formatted in one of three ways. It is important to understand how each format affects the processing of synonyms. The following example shows standard formatting:

```
MARX  Groucho Chico Harpo Zeppo Gummo Karl
```

The main entry, *MARX*, is followed by its synonyms: *Groucho*, *Chico*, *Harpo*, *Zeppo*, *Gummo*, and *Karl*. Using a thesaurus that contains this entry, the following query:

```
Marx@
```

will be processed as:

```
Groucho OR Chico OR Harpo OR Zeppo OR Gummo OR Karl
```

The term *Marx* is simply replaced by its synonyms. If AND, rather than OR, is the default operator, the query will be processed as:

```
Groucho AND Chico AND Harpo AND Zeppo AND Gummo AND
Karl
```

Alternate Thesaurus Entry Format #1

You may also see thesaurus entries formatted in a manner similar to the following example:

```
STOOGES     {Moe Larry Curly Shemp}
```

Synonyms enclosed in braces are evaluated as a nested OR operation, regardless of any other logic imposed on a query. Using a thesaurus that contains this entry, the following query:

```
Healy AND stooges@
```

will be processed as:

```
Healy AND (Moe OR Larry OR Curly OR Shemp)
```

Alternate Thesaurus Entry Format #2

Thesaurus entries can contain nested expressions, as in the following example:

```
STOOGES     Moe Larry (Curly NOT Shemp)
```

The nested expression, along with other synonyms, will replace the main entry in a query; the expression will retain its internal logic within the context of a larger query. Using a thesaurus that contains this entry, the following query:

```
Healy AND stooges@
```

will be processed as:

```
Healy AND Moe AND Larry AND (Curly NOT Shemp)
```

SEARCHING BY EXAMPLE

By specifying a particular record with the *document operator*, you can search for other records that are similar in content. When you use this operator in a query, all statistically significant words in the specified record are included in the query.

Note: You can enter query operators in upper- or lower-case. They are capitalized in the following examples only for purposes of clarity.

Syntax:

D_*n*

where *n* is the ID number of the record to be used as an example of the kind of information you are seeking.

Example:

```
D_20
```

This query will search for records that contain significant words from the database's twentieth record.

Note: Use of this operator usually retrieves the specified record as the most similar record.

REUSING A PREVIOUS QUERY

By using the *query operator* to refer to an earlier query from the current search session, you can include it in your current query; this is equivalent to including an entry from the History Advisor window—see *Including History Advisor Entries in Your Query* on page 96. You can use this operator by itself—to replicate the results of a previous query—or as part of a larger query.

Note: You can enter query operators in upper- or lower-case. They are capitalized in the following examples only for purposes of clarity.

Syntax:

Q_*n*

where *n* is the position of the previous query in your current search history.

Note: If *n* specifies an invalid query number (e.g., *n* is 15 and the current search history comprises only five queries), the operator will be treated as a term that does not exist in the database.

Example:

```
Q_5
Q_7 Trilateral Commission
```

The first example duplicates the fifth query in the current history.

The second example includes all elements of the current history's seventh query.

FIELD-RESTRICTED SEARCHING

When searching a database in which records have been organized into fields, you can formulate queries that consider only a specified field or range of fields within each record. Moreover, you can search particular fields for terms that match a specified alphanumeric value or fall within a specified range of values.

SEARCHING SPECIFIC FIELDS FOR A QUERY WORD

You can use the *word-level field restriction operator*—the colon—to search one or more fields for a particular query term.

Syntax:

```
word:field1,field2...
```

where *word* is the query term to be sought and *field* specifies one or more fields to which the search will be restricted; if you specify multiple fields, their names must be separated by commas. You can use partial field names and fieldlists with this operator—see *Using Partial Field Names* on page 128 and *Field Restriction Shorthand: Using Fieldlists* on page 128.

The word-level field restriction operator will be distributed through scope of operation delimiters—see *Composing Complex Queries* on page 133. For example, the following query:

```
(Harding AND Gillooly):defendant
```

is equivalent to:

```
Harding:defendant AND Gillooly:defendant
```

Note: Any indexed field specified with this operator will be searched, even if it has been defined as non-searchable by the database administrator—see *Assigning Field Attributes* on page 207.

Examples:

```
Renfield:name,supervisor
Alhazred:author AND Necronomicon:title
```

The first query will retrieve records that contain *Renfield* in either of the *name* or *supervisor* fields.

The second query will retrieve records that contain *Alhazred* in the *author* field and *Necronomicon* in the *title* field.

RESTRICTING AN ENTIRE QUERY TO SPECIFIC FIELDS

You can use the *query-level field restriction operator* to restrict your entire query to one or more specified fields. This is a convenient alternative to using the word-level operator in conjunction with every word in a query.

Syntax:

```
query/f:field1,field2...
query /f: ALL
```

where *query* is your entire query and *field* specifies one or more fields to which the search will be restricted; if you specify multiple fields, their names must be separated by commas; a single space may be placed immediately before or after the operator. You can use partial field names and fieldlists with this operator—see *Using Partial Field Names* on page 128 and *Field Restriction Shorthand: Using Fieldlists* on page 128.

If you use the word *ALL* instead of field names, all indexed fields in a database will be searched.

As with the word-level field restriction operator, the query-level operator will be distributed through scope of operation delimiters—see *Composing Complex Queries* on page 133.

Note: Any indexed field specified with this operator will be searched, even if it has been defined as non-searchable by the database administrator—see *Assigning Field Attributes* on page 207.

Examples:

```
Renfield Seward Harker/f:name
Miskatonic ADJ University /f: text,synopsis
```

The first query will search only the *name* field for occurrences of *Renfield*, *Seward*, or *Harker.*

The second query will retrieve records in which *Miskatonic* occurs immediately before *University* in either of the *text* or *synopsis* fields.

COMBINING WORD-LEVEL AND QUERY-LEVEL OPERATORS

You can combine word-level and query-level field restriction operators in a single query; the query-level operator will not apply to any terms with word-level restriction.

Example:

```
Seuss:author eggs AND ham /f:title
```

This query will retrieve records in which either *Seuss* occurs in the *author* field or both *eggs* and *ham* occur in the *title* field.

USING PARTIAL FIELD NAMES

When specifying field names in queries, you can save time and typing by using *partial field names*. A partial field name is a beginning fragment of a field name; the fragment must contain enough characters to uniquely distinguish it among the fields defined for the database being searched.

For example, while searching a database for which the fields *publisher*, *pubdate*, *review*, and *recdate* have been defined, you could compose a query using *publ* to specify the *publisher* field or *rev* to specify the *review* field.

Note: If you use a partial field name that is too short—with too few characters to uniquely identify a particular field—PL will use the first matching field found in the database's definition file.

FIELD RESTRICTION SHORTHAND: USING FIELDLISTS

If you routinely restrict searches of a particular database to a given set of fields, you may find it convenient to use a *fieldlist* instead of field names; a fieldlist is a single name with which you can specify multiple fields. See *Defining a Fieldlist* on page 209.

Like partial field names—see *Using Partial Field Names* on page 128—partial fieldlist names may be used in a query.

Example:

```
Lovecraft Arkham Dunwich /f:abstract
```

If, for the current database, *abstract* is a fieldlist defined to be equivalent to the fields *author*, *publisher*, and *title*, this query will be restricted to those fields.

SEARCHING FIELDS FOR VALUES AND VALUE RANGES

Value Searching

You can use the *equivalence operator*—the "equals" sign (=)—to retrieve records in which a specified field contains a specified alphanumeric value.

Note: If you specify a numeric value in conjunction with the equivalence operator, PL considers only numeric terms in the specified field; if you specify an alphabetic or alphanumeric value, all terms in the specified field are considered.

Syntax:

```
field=value
value=field
```

where *field* is the name of the field being searched and *value* is the alphanumeric value being sought.

Note: When searching for a numeric value with this operator, you must express *value* as a whole number.

Examples:

```
author=Seuss
1897=year
```

The first query will retrieve records that contain *Seuss* in the *author* field.

The second query will retrieve records that contain *1897* in the *year* field.

Outside Value Searching

The *non-equivalence operator*—the "equals" sign preceded by the exclamation point (!=)—can be used in binary or unary form. In binary form, it searches for records in which a specified value does not occur in a specified field. In unary form, it searches for all records that do not contain a specified value.

Note: If you specify a numeric value in conjunction with the non-equivalence operator, PL considers only numeric terms in the specified field; if you specify an alphabetic or alphanumeric value, all terms in the specified field are considered.

Syntax (binary):

field`!=`*value*
value`!=`*field*

Syntax (unary):

`!=`*value*

where *field* is the name of the field being searched and *value* is the alphanumeric value being excluded.

Note: When excluding a numeric value with this operator, you must express *value* as a whole number.

Examples:

```
author!=Seuss
1897!=year
!=C57D
```

The first query will retrieve records in which *Seuss* does not occur in the *author* field.

The second query will retrieve records in which *1897* does not occur in the *year* field.

The third query will retrieve records that do not contain the term *C57D*.

Value Range Searching

With the *range operators*—the “greater than” angle bracket (>), the “less than” angle bracket (<), or either bracket followed by the “equals” sign (>=, <=)—you can search for records in which a specified field contains a specified value.

Note: If you specify a numeric value in conjunction with a range operator, PL considers only numeric terms in the specified field; if you specify an alphabetic or alphanumeric value, all terms in the specified field are considered.

Syntax:

```
field>value
field<value
field>=value
field<=value
```

where *field* is the name of the field being searched and *value* is the upper or lower limit of the range.

Tip: Range operators work much faster when they are part of a bounded expression—when you specify upper *and* lower limits to the range being considered; only in a bounded expression may *value* precede a range operator.

Note: When searching for numeric values with these operators, you must express the values as whole numbers.

Examples:

```
population>900
zipcode<=20895
Karloff<name<Pratt
```

The first query will retrieve records in which values greater than *900* occur in the *population* field.

The second query will retrieve records in which values of *20895* or lower occur in the *zipcode* field.

The third query will retrieve records in which the *name* field contains values alphabetically between *Karloff* and *Pratt*.

WORD AND WORDPERFECT DOCUMENTS...

Word and WordPerfect documents cannot be subdivided into fields recognized by Personal Librarian. When documents in these word processor formats are added to a PL database, their full contents are considered to be resident in the default *TEXT* field. Unless a field-restricted query specifies *ALL* or the *TEXT* field, it will ignore records in these file formats.

OPERATOR PRECEDENCE RULES

Because search queries can include different types of operators in a variety of combinations, Personal Librarian uses precedence rules to govern the order in which a query’s operations are processed.

When processing queries, PL evaluates operators in the following order—all operators of equal precedence are evaluated from left to right:

- All operators not specified below, in order from left to right.
- Proximity operators—W/*n*, ADJ and its surrogates (hyphen, apostrophe, comma, period), NEAR/*n*, SAME—from left to right.
- Boolean NOT logic operators—NOT, NOTSAME—from left to right.
- The Boolean AND operator.
- The Boolean OR operator.
- The query-level field restriction operator (/f:).

COMPOSING COMPLEX QUERIES

When you formulate a complex query—a query that includes more than one type of operator—you can use parentheses as *scope of operation delimiters*. If you are familiar with algebra, you will recognize the use of these delimiters as a way to establish the scope of a particular mathematical operation. In a query, they can delineate the range of terms that a particular operator influences, or control the order in which operations within the query are evaluated.

PL queries support infinite levels of parenthetical nesting—one set of delimiters inside another. Nested expressions are evaluated in order from the lowest nesting level to the highest.

Most query operators do not distribute through scope of operation delimiters. For example, both of the following queries:

```
(Buttafuoco Gillooly)!
(Buttafuoco Gillooly)"
```

will be interpreted as:

```
Buttafuoco OR Gillooly
```

The exceptions to this rule are field restriction operators, which will distribute through a nested query expression. See *Field-Restricted Searching* on page 125.

Examples:

```
United ADJ (States Nations)
(space shuttle) W/2 (program research)
```

In the first query, the adjacency operation will apply to *United* and the results of the delimited OR operation. Without the scope of operation delimiters, PL's precedence rules will cause the query to be evaluated as:

```
(United ADJ States) OR Nations
```

In the second example, the proximity operation will be applied to all possible results of the two delimited OR operations. Without the delimiters, the higher precedence of the proximity operator will cause the query to be evaluated as:

```
space OR (shuttle W/2 program) OR research
```

Tip: You should rely on natural language queries; they will usually retrieve useful information that Boolean queries can exclude. With relevance ranking at work, you need not be concerned with reducing the number of retrieved records. Unless you are an expert at Boolean searching, you will generally get better results from natural language queries.

CHAPTER 9

Table of Contents Searching

A database can be structured like a book, with chapters and subchapters. You can search such a database as you would a book, by selecting the desired section from a table of contents. When searching a database with a table of contents, you can establish a search domain, which restricts standard searches to a single chapter or a range of chapters.

The Table of Contents Window

When you open a database that has a table of contents, a table of contents window, similar to the example in Figure 39, will be opened automatically. If an open database's table of contents window has been closed, you can open it by selecting Table of Contents from the Go To menu (F5).

Figure 39 A table of contents window

You can also use the Go To menu’s Table of Contents command to make an open table of contents window active. If you select this command while searching a virtual database, an intermediate dialog will prompt you to select the database for which you want to view the associated table of contents.

A database can contain multiple books, in which case the top-level headings in the table of contents window will be individual book titles; for a database that consists of a single book, the top-level headings can represent sections or chapters.

A table of contents can comprise up to seven hierarchical levels of book titles, chapters, and sub-chapters. Headings for chapters that contain sub-chapters are distinguished by triangular pointers, as shown in Figure 40.

You can open a separate table of contents window for each open database that has a table of contents.

EXPANDING AND COLLAPSING HIERARCHY LEVELS

To expand table of contents hierarchy in order to see a chapter’s constituent sub-chapters, click the chapter’s triangular pointer. The pointer will turn downward and the hierarchy will expand, as shown in Figure 40.

Figure 40 Clicking a chapter’s pointer will display its sub-chapters.

Chapter pointers act as toggles; clicking them will alternately expand and collapse a chapter’s hierarchy.

Note: If a table of contents heading contains more than a hundred chapters in the immediately subordinate hierarchy level (e.g., Part 1 contains Chapters 1–200), it may be relatively slow to expand or collapse.

OPENING RECORDS FROM A TABLE OF CONTENTS

A table of contents chapter can comprise a single record or a range of consecutive records. To view the first record in a chapter, double-click the chapter heading in the table of contents window. The record will be opened and you can use the record window's Forward Browse button to navigate through other records in the chapter.

DISPLAYING A RECORD'S LOCATION IN A TABLE OF CONTENTS

If you select Table of Contents from the Go To menu while either a record window or the Search Results Window is active, the table of contents will highlight the active record's parent chapter and, if that chapter is a sub-chapter, the chapter(s) containing it will be displayed in expanded form, as shown in the example in Figure 41.

Figure 41 The table of contents expands to show a record's parent chapter.

You can use this feature to instantly ascertain a record's orientation within a database's table of contents hierarchy. You can disable this feature—see *Customizing Table of Contents Expansion to Active Record Location* on page 142.

DISPLAYING A RECORD'S GENEALOGY

Another way to get information about a record's position within a table of contents hierarchy is to display the record's *genealogy*.

If genealogy display is enabled for an open database—see *Adding Genealogy Markup* on page 192 and *Enabling Genealogy Display for Records* on page 229—record windows for that database will feature a Genealogy button, shown in Figure 42.

Figure 42 The Genealogy button

To display an open record's genealogy, click the record window's Genealogy button. A genealogy window, similar to Figure 43, will be displayed.

Figure 43 A genealogy window

RESTRICTING A STANDARD SEARCH TO A RANGE OF CHAPTERS

You can restrict a search to a single chapter or a range of chapters in a database by establishing a search *domain* in the table of contents window. When you set a domain, a standard search executed from the Search Window will consider only records that occupy chapters included in the domain; if you are searching a virtual database, the search will also consider all records in any open database for which a domain has not been established. In a virtual database, you can establish a domain for each open database that has a table of contents.

Note: You can set a domain only at the top level of a table of contents hierarchy.

To set a search domain:

1. With the table of contents window active, move the cursor into the margin to the left of the top-level heading that you want to include in the domain.

2. When the cursor changes to resemble a lorgnette (i.e., opera glasses), click the margin. As shown in Figure 44, a lorgnette symbol will appear in the margin (and beside any currently displayed sub-chapters) to indicate that the chapter is part of a search domain, and the window's title bar will change to reflect the search restriction.

Figure 44 Standard searches will be restricted to chapters in a domain.

3. Repeat the preceding steps for each top-level chapter to be included in the search domain.

Note: Domain markers act as toggles; to remove a chapter from the current domain, click on its domain marker.

SAVING AND REUSING SEARCH RESTRICTIONS

You can save a search domain for use in future searches of the same database. Saved domains are unique for each user.

To save a search domain:

1. With a domain set in the active table of contents window, select Domains/Save from the File menu.

2. In the dialog that appears, enter a name for the domain.

3. Click the OK button.

To reuse a saved domain:

1. With the appropriate table of contents window active, select Domains/Restore from the File menu. A dialog listing all saved domains for the database is displayed.

2. Click the name of the desired domain.

3. Click the Restore button.

To delete a saved domain:

1. With the appropriate table of contents window active, select Domains/Delete from the File menu. A dialog listing all saved domains for the database is displayed.

2. Click the name of the domain that you want to delete.

3. Click the Delete button.

4. A dialog will prompt you to confirm that you want to delete the domain. Click the Delete button.

CUSTOMIZING TABLE OF CONTENTS EXPANSION TO ACTIVE RECORD LOCATION

By default, when you select Table of Contents from the Go To menu, the table of contents window will expand to show the location of the active record's parent chapter in the hierarchy. To disable this feature, select Table of Contents from the Prefs menu. The Table of Contents Preferences dialog, shown in Figure 45, will be displayed.

Figure 45 The Table of Contents Preferences dialog

To disable the feature, click the Open Table of Contents at Active Record Location checkbox and click the OK button.

Note: Changes to default table of contents preferences normally apply only to your current search session. To retain your current preference settings for use in future search sessions, select Save Settings from the Prefs menu.

Restoring Default Preference Settings

To restore all preferences to their system-wide default settings, select Default Settings from the Prefs menu.

SELECTING FONTS FOR THE TABLE OF CONTENTS AND GENEALOGY WINDOWS

To select a font for table of contents and genealogy text:

1. Make a table of contents window active by clicking it or selecting it from the Windows menu.

2. From the Font menu, select the desired typeface, size, and formatting.

Note: A font selection made for a single table of contents window type will apply to all subsequently opened windows of that type.

CHAPTER 10

LINKING TO OTHER INFORMATION

This chapter discusses the use of hypertext links, a mechanism Personal Librarian provides for rapidly accessing supplemental information during a search.

WHAT ARE HYPERTEXT LINKS?

A hypertext link is a coded instruction, accessed as part of an individual record or as part of the Personal Librarian interface, that directs PL to perform a task.

LINK ACCESS METHODS

There are three methods of accessing and activating hypertext links in Personal Librarian; all three methods support the same range of link functions.

Embedded Links. Named because they appear as highlighted portions of the text within a record, embedded links can be activated only when they are visible within a record window. See *Creating Embedded Links* on page 174.

Fielded Links. Like embedded links, fielded links reside in individual records; they differ from embedded links in that they must occupy specially designated fields that may not be visible in the record window. However, because a fielded link can be accessed through the Go To menu, you can activate it while any part of its parent record is visible in the active record window. See *Creating Fielded Links* on page 175.

Action Links. Accessed through the Actions menu, action links can be global or database-specific. The availability of global action links is constant; you can activate database-specific action links only when their associated databases are open. See *Creating Database-Specific Action Links* on page 225 and *Creating Global Action Links* on page 265.

LINK FUNCTIONS

Regardless of how they are accessed, hypertext links in Personal Librarian can be one of the following functional types:

Document Links. This type of link serves as a cross-reference from one record to another. When you activate a document link, it opens a record related to the link's parent record. See *Linking to a Specific Record Within a Database* on page 176.

Target Links. Similar to a document link, a target link connects to one or more specific locations within other records or within its own parent record. See *Linking to Specific Locations Within Records* on page 177.

Search Links. This type of link executes a predefined search of all open databases. See *Executing a Search* on page 179.

Image Links. When you activate an image link, it displays a predefined image using PL's integrated image viewer or a third-party viewer. See *Displaying an Image* on page 179.

Command Links. By activating a command link, you can run an external application—word processor, spreadsheet, etc.—from within PL. See *Executing an External Application* on page 182.

DLL Links. This type of link executes a function in an external dynamic link library. See *Executing a DLL Function* on page 183.

ACTIVATING EMBEDDED LINKS

The presence of an embedded link in a record is immediately obvious when the portion of the record containing the link is displayed in a record window. As shown in Figure 46, an embedded link is prominently highlighted with underlining, overlining, and inward-pointing angle brackets.

Figure 46 An embedded link is easily distinguished from surrounding text.

To activate an embedded link, click its *hot spot*—the area between the angle brackets.

DISPLAYING AN EMBEDDED LINK'S FUNCTION

Before activating an embedded link, you can find out what it will do upon execution. To display a message describing the link's underlying function, click *and hold* on the link's hot spot. A message similar to Figure 47 will be displayed.

Figure 47 Clicking and holding on a link's hot spot will display its function.

ACTIVATING FIELDED LINKS

A fielded link is available when the record containing the link is open and its window is active.

To activate a fielded link:

1. With the desired record window active, select Fielded Links from the Go To menu. A dialog listing all fielded links in the record, similar to Figure 48, will be displayed.

Figure 48 The Follow Fielded Link dialog lists all fielded links in the active record.

2. Click the link that you want to activate.

3. Click the OK button.

ACTIVATING ACTION LINKS

Global action links are accessible through the Actions menu, as shown in Figure 49, at all times during a search session, even if no databases are open. To activate a global action link, select it from the Actions menu.

Figure 49 Global action links are always available during a search session.

A database-specific action link is available only when its associated database is open. If action links have been defined for an open database, the database’s name will appear in the Actions menu, as shown in Figure 50.

Figure 50 Database-specific action links are listed below global links.

To activate a database-specific action link, select its parent database's name from the Actions menu. A fly-out menu listing all action links for the database, as shown in Figure 51, will be displayed, whereupon you can select the link that you want to execute.

Figure 51 A fly-out menu lists individual action links for the database.

Note: If you are searching a virtual database, a database-specific action link that functions as a search link or target link may apply to all open databases.

USING A TARGET LINK WITH MULTIPLE TARGETS

If you activate a target link for which multiple targets exist among open databases, a window containing summaries for all target records will be displayed; this window is functionally equivalent to the Search Results Window's Hitlist—see *The Hitlist* on page 48. To open the desired target record from the list, double-click its summary or, having highlighted its summary, press ENTER.

REVERSING HYPERTEXT LINKS

When you activate document or target links, which move you from one record to another, you create link chains through which you can travel backward. A hypertext link chain can represent one or more document or target links.

To move reverse hypertext links:

1. Make active the record window that was opened by a link.
2. Select Reverse Link from the Go To menu (CTRL+B).
3. Repeat until you reach the desired record in the link chain.

Note: If you close a record that serves as an intermediate step in a hypertext link chain, the original chain will be broken into two separate chains.

USING THE IMAGE VIEWER

If image links are not configured to use a third-party application—see *Displaying an Image* on page 179—they display images with Personal Librarian's integrated image viewer. The integrated viewer displays an image in a window similar to the example in Figure 52.

Figure 52 An image displayed with PL's integrated viewer.

PL's integrated viewer can display monochrome images in the following formats: uncompressed TIFF, compressed TIFF group 3, and compressed TIFF group 4. When the integrated viewer is in use, PL's Image menu becomes active. You can use the commands in this menu to:

- increase image magnification (Ctrl+").
- decrease image magnification (Ctrl+-).
- scale the image vertically or horizontally within its window.
- rotate an image 90 degrees clockwise (Ctrl+>), 90 degrees counter-clockwise (Ctrl+<), or 180 degrees.

- invert the image's black and white values.
- modify the weight of lines in the image.

MAGNIFYING A SELECTED IMAGE AREA

To magnify a selected portion of an image:

1. Make the image window active by clicking it or selecting it from the Window menu.
2. Drag the mouse, as shown in Figure 53, to circumscribe the area that you want to magnify.

Figure 53 Drag over the area that you want to enlarge.

3. When you have outlined the desired portion of the image, release the mouse button. The selected area will be enlarged so that its longest axis fits the image window, a shown in Figure 54.

Figure 54 The selected area is magnified to fit the image window.

PRINTING AN IMAGE

While an image window is active, you can print the image by selecting Print Image from the File menu.

PART 3

DATABASE ADMINISTRATION

3

CHAPTER 11

EXPRESS DATABASE SETUP

This chapter outlines the most streamlined method for creating a Personal Librarian database from your ASCII text, Word or WordPerfect files. This "cookbook" approach is presented without explanation; each step is detailed in other chapters.

CREATING A BASIC PL DATABASE

1. Run PL-Admin—see *Starting PL-Admin* on page 23.

2. Select New from the File menu. A standard file selection dialog will be displayed—see *Creating a New Database* on page 202.

3. Navigate to the directory that you want the database's index, definition, and control files to occupy.

4. In the File Name window, enter a name for the database's definition file; do not enter a filename extension.

5. Click the OK button. The Define Database dialog will be displayed.

6. In the Database Name window, enter a ShortList name for the database—see *Assigning a ShortList Name* on page 204.

7. Click the OK button.

8. A dialog will appear, prompting you to decide whether the new database is to appear in the ShortList. Click the Yes button.

9. In the PL-Admin window, click the Add Files button. A standard file selection dialog will be displayed—see *Adding Files* on page 236.

10. Navigate to a directory occupied by text files that you want to add to the database.

11. From the list of files in the directory, select those that you want to index.

12. Click the OK button. The selected files will be listed in PL-Admin's Pending Changes window.

13. Repeat Steps 9–12 to select files in other directories for indexing.

14. When the Pending Changes window lists all files that you want to include in the database, click the PL-Admin window's OK button.

15. When indexing is finished, a dialog will summarize the changes that were made to the database's index. Click the OK button.

16. Exit PL-Admin.

You can now search your new database with Personal Librarian. Each text file added to the database constitutes a single record; the text in each record occupies the default *TEXT* field.

CHAPTER 12

DESIGNING A DATABASE

One of the most important parts of the database administrator's role is database design. The design of a database profoundly influences how much, and how well, it gets used. Prior to building a database, you should consider the design principles discussed in this chapter.

HOW MANY DATABASES?

Your database design process should start with an examination of your goals in creating a database. From there, it should evolve into a general plan for achieving those goals and accommodating the needs of the database users. Your first decision involves the most basic question of structure: should you create a single database or build a group of separate databases?

Central to this decision is a consideration of how users are going to access the data. For example, if usage patterns indicate that users would routinely search separate databases simultaneously, you could facilitate more efficient retrieval by combining that total body of information in a single database.

A good way to determine the number of databases that you need is to study how the information is organized and used in a non-database environment: observe or interview users of the information in its current format. You might include the following questions in your research:

- How is the information currently stored and used?
- What is its original format and purpose?
- Are the database-related purposes different (e.g., current awareness vs. research)?
- Does a difference in purpose suggest a difference in format?
- What improvements are expected through use of Personal Librarian by the database users and administrators?

DO YOU NEED FIELDS?

When designing a database, you need to determine how many fields, if any, should be part of its record structure. By subdividing records into multiple fields, you enable users to employ field-restricted searching—see *Field-Restricted Searching* on page 125. If your data are in a very simple format and have no natural category breakdowns, you may choose to have only one field in each record.

You can do many things with fields to help users get the most out of the database. Consider the following scenarios:

- Your fields can follow the structure of the source information. For example, a book-like database might have title, author, chapter, section, and text fields, or a transcript of legal testimony might include one question and one answer per record or one witness per record.

- Your fields can define record types. For example, in a database containing job opening and applicant information, you might create two record types (openings and applicants) that contain different, but possibly overlapping, sets of fields. Job opening records might include the following fields: *TITLE*, *PROFESSION*, *SALARY_RANGE*, *SUPERVISOR*, *SUP_PHONE*, *QUALIFICATIONS*, and *DATE_OPEN*. Job applicant records might consist of the following fields: *NAME*, *ADDRESS*, *PHONE*, *EDUCATION*, *EXPERIENCE*, *SALARY_REQUIRED*, *JOB_DESIRED*, and *AVAILABILITY*. In this scenario, all job-search information can be kept in the same database but can be searched separately. You can search the openings record type without retrieving information on candidates who have already applied, and you can limit users to one type of record by setting specific groups of fields for search and display or by defining fieldlists.

OTHER CONSIDERATIONS

Certain customization options can be implemented after a database is built; they affect only PL's searching performance. Others options, discussed in the following sections, affect database indexing—the process whereby a database is built or updated. Because changing pre-indexing options for an existing database requires rebuilding or reorganizing its index, you can avoid such inconvenience by considering the options prior to building a database.

STOPWORD CUSTOMIZATION

To prevent the retrieval of extraneous records during searching, Personal Librarian disregards certain stopwords within records—see *Stopwords* on page 5.

You can define new stopwords by adding them to PL's global stopword list, which is contained in the ASCII text file `stopword.cpl`, located in PL's `\shared` subdirectory. This file, prior to any modification, contains the terms listed in Table 3 on page 163.

Note: When considering stopwords, you should bear in mind that any word listed in `stopword.cpl` will not be used for retrieval—it will not be recognized within a search query. Depending on the nature of your data and search interests, this can yield results that seem to run counter to reasonable user expectations. For example, with the word *a* listed in `stopword.cpl`—as it is by default—you cannot search a pharmacology database for occurrences of the phrase *vitamin a*.

When editing `stopword.cpl`, you must observe the following guidelines:

- Words must be separated by an end-of-line character.
- If changes to the stopword list will apply to an existing database, you must rebuild the database after modifying the list.

Tip: Because all existing databases—excluding those with database-specific stopword lists—must be reindexed when the global stopword list is altered, you should make all changes to this list before building any databases.

Words in PL's global stopword list are ignored during indexing and searching of all databases. You can create a database-specific stopword list, which will supersede the global list for a particular database.

To create a database-specific stopword list:

1. Place a copy of `stopword.cpl` in the database's *library directory*—the eventual location of the database's index and control files.

2. Edit `stopword.cpl` to include the desired stopwords for the database.

3. Build the database.

Table 3 PL's Default Global Stopword List

a	about	above	across
adj	after	again	against
all	almost	alone	along
also	although	always	am
among	an	and	another
any	anybody	anyone	anything
anywhere	apart	are	around
as	aside	at	away
be	because	been	before
behind	being	below	besides
between	beyond	both	but
by	can	cannot	could
deep	did	do	does
doing	done	down	downwards
during	each	either	else
enough	etc	even	ever
every	everybody	everyone	except
far	few	for	forth
from	get	gets	got

Table 3 PL's Default Global Stopword List

had	hardly	has	have
having	her	here	herself
him	himself	his	how
however	i	if	in
indeed	instead	into	inward
is	it	its	itself
just	kept	many	maybe
might	mine	more	most
mostly	much	must	myself
near	neither	next	no
nobody	none	nor	not
nothing	nowhere	of	off
often	on	only	onto
or	other	others	ought
our	ours	out	outside
over	own	p	per
please	plus	pp	quite
rather	really	said	seem
self	selves	several	shall
she	should	since	so
some	somebody	somewhat	still
such	than	that	the
their	theirs	them	themselves
then	there	therefore	these
they	this	thorough	thoroughly
those	through	thus	to
together	too	toward	towards

Table 3 PL's Default Global Stopword List

under	until	up	upon
v	very	was	well
were	what	whatever	when
whenever	where	whether	which
while	who	whom	whose
will	with	within	without
would	yet	young	your
yourself			

TABLE OF CONTENTS SEARCHING

If you want to organize a database's contents like a book, with chapters and sub-chapters, you must add table of contents markup to text files prior to adding them to the database. See *Adding Table of Contents Markup* on page 187.

SAME PARAGRAPH PROXIMITY SEARCHING

You can enable users of a database to search for word pairs that occur in the same paragraph or in a specified range of paragraphs by adding a statement to the database's definition file prior to indexing. See *Enabling Same Paragraph Proximity Searching* on page 217.

DATABASES STORED ON SLOWER MEDIA

You can optimize search and display performance for a database stored on relatively slow media (e.g., CD-ROM) by building a portion of the database's index on a faster medium; you can achieve this partitioning of the index by placing a particular statement in the database's definition file prior to indexing. See *Building Index Partitions on Faster Media* on page 217.

SPECIAL CHARACTERS

By modifying rules in a *tokenizer file*, you can determine which characters and character combinations are treated as readable text by Personal Librarian. If you customize tokenization rules, you will have to rebuild any existing databases. See *Making Special Characters Searchable in a Database* on page 220 and *Making Special Characters Searchable* on page 297.

CHAPTER 13

Preparing Text Files for Indexing

The text files you add to a Personal Librarian database are referred to as source files. Source files can be in ASCII, Word for Windows 2.0, or WordPerfect 5.0/5.1 format. From this chapter you will learn how to use a text editor or word processor to subdivide ASCII source files into multiple records, subdivide records into multiple fields, add embedded or fielded hypertext links to records, and add table of contents and genealogy information to records.

Adding Records Without Markup

You can add a source file to a PL database without any record or field markup, in which case its contents will constitute a single record and occupy the default *TEXT* field.

ADDING RECORD MARKUP

You can divide an ASCII source file into multiple records by inserting a *record tag*—the word *end* bordered by hyphens—between the end of one record and the beginning of the next, as shown in the following example:

```
-text-
Searching is a process, not an event…
-end-
-text-
A text database is a collection of related
documents assembled into…
-end-
-text-
As opposed to a keyword-based system, Personal
Librarian is full-text retrieval software…
-end-
```

Note: Source files must be in ASCII format in order to accommodate record markup.

While Personal Librarian imposes no limits on record length, you are encouraged to keep it at one to five pages. This facilitates easier viewing within record windows; users will be able to view a record's contents without excessive vertical scrolling. To minimize the need for horizontal scrolling, you should limit the width of lines to 80 or fewer characters.

When editing a source file to add record markup, you should observe the following guidelines:

- A record tag must begin at the first character position in a line.

- Record tags can be in upper-, lower-, or mixed-case.
- A record tag must begin and end with a hyphen.
- Extra lines after the last record tag in a source file will produce an empty record.

ADDING FIELD MARKUP

You can divide a record into multiple fields by adding field tags—field names bordered by hyphens—as shown in the following example:

```
-DATES-
19911219
-HEADLINE-
BCCI Seen Pleading Guilty in Agreement
-BYLINE-
Rob Wells
-DATELINE-
NEW YORK
-TEXT-
NEW YORK - Bank of Credit and Commerce
International could plead guilty to…
-END-
```

Note: Source files must be in ASCII format in order to accommodate field markup.

When editing a source file to add field markup, you should observe the following guidelines:

- Within its first 15 lines, every record with field markup must contain a field defined for Hitlist display—see *Assigning Field Attributes* on page 207. If a record does not have such a field within that range of lines, it will be represented by a blank summary in the Search Results Window's Hitlist. If you want to modify the default scanning depth, see *Setting the Number of Lines per Record Scanned for Hitlist Information* on page 277. You should put fields that will be displayed in the Hitlist as close to the beginning of records as possible; this will expedite Hitlist generation.

- In a source file that contains multiple fielded records, a record tag—see *Adding Record Markup* on page 168—must be placed after the contents of the last field in each record.

- A record can contain up to 255 field tags; this is the maximum number of fields that you can define for a database.

- Field tags are not required in every record.

- A field tag must begin and end with a hyphen.

- A field tag must begin at the first character position in a line.

- A field tag must precede the text that will occupy the corresponding field (e.g., the field tag `-date-` must be placed before text that will reside in the *DATE* field).

- An end-of-line character is not required between a field tag and the text that follows it; the preceding example has field tags separated from field text only for the sake of clarity.

- Field tags can be in different order from record to record. However, this is discouraged because it may disorient users who are accustomed to looking in a particular place for a particular field.

- Between its hyphens, a field tag must contain a field name that is—or will be—defined for the database that will contain the field's parent record. See *Defining Fields* on page 205.

- A field name—the text between the hyphens in a field tag—may be up to 29 characters long, including punctuation.
- The underscore (_) and the forward slash (/) are the only punctuation marks that may be used in a field name.
- A field name may not contain spaces.
- Field names can be in upper-, lower-, or mixed-case.
- A field name must begin with an alphabetic character; subsequent characters can be alphanumeric.
- A field name may not begin with another complete field name (e.g., you cannot use *date* and *date_published* as field names in the same database).

FORMATTING DATES

Because Personal Librarian evaluates numeric text strings from right to left (i.e., as numbers), you should consider formatting dates in such a way that they will be subject to sorting, value searches, and range searches. See *Sorting Hit Records by Field* on page 50, *Value Searching* on page 129, and *Value Range Searching* on page 131.

Syntax:

YYYYMMDD

Example:

```
19941201
```

This example shows ideal formatting for December 1, 1994.

ADDING MARKUP FOR BOLD & ITALIC TEXT

By marking text in a source file with formatting tags, you can have it displayed as bold/italic in record windows.

When editing a source file to add text formatting tags, you should observe the following guidelines:

- A formatting tag must include a beginning marker (e.g., <b>) and an end marker (e.g., </b>).
- An end-of-line character may not exist between a formatting tag's beginning and end markers.
- Multiple formatting tags may be placed in the same line of text.

BOLD TEXT

Syntax:

`<b>`*`text`*`</b>`

where *text* is the text that you want displayed in bold format.

Example:

```
We can <b>rebuild him</b>.
```

In a record window, this example will be displayed as:

`We can` **`rebuild him.`**

ITALIC TEXT

Syntax:

`<it>`*`text`*`</it>`

where *text* is the text that you want displayed in italic format.

Example:

```
We <it>have the technology</it>.
```

In a record window, this example will be displayed as:

We *have the technology*.

BOLD & ITALIC TEXT

Syntax:

`<bi>`*`text`*`</bi>`

where *text* is the text that you want displayed in bold and italic format.

Example:

```
We can make him better than he was: <bi>better...
stronger...faster</bi>.
```

In a record window, this example will be displayed as:

We can make him better than he was: ***better...
stronger...faster***.

ADDING HYPERTEXT LINKS

As database administrator, you must create the hypertext links that will be available to searchers of the database. You can provide links with varied functions and access methods—see *Link Functions* on page 146 and *Link Access Methods* on page 145.

Because embedded and fielded links must exist within the body of a source file, you create them when preparing source files for addition to a database. Techniques with which you can create these links are discussed in the following sections.

Because action links—both global and database-specific—exist independently of source files, their creation is not part of source file preparation. If you want to create action links, see *Creating Database-Specific Action Links* on page 225 and *Creating Global Action Links* on page 265.

CREATING EMBEDDED LINKS

When editing a source file to add embedded hypertext links, you should observe the following guidelines:

- An embedded link must be placed in a field that is defined for record window display—see *Assigning Field Attributes* on page 207.

- A link must include a beginning marker (e.g., `<D<…>>`) and an end marker (e.g., `</D>`).

- An end-of-line character (e.g., a carriage return) may not exist between a link’s beginning and end markers.

- Multiple embedded links may be placed in the same line of text.

- No part of a link may be positioned beyond the 256th character in a line of text.
- An embedded link does not require—but should use—descriptive text that will occupy the link's hot spot.

CREATING FIELDED LINKS

When editing a source file to add fielded hypertext links, you should observe the following guidelines:

- A fielded link must be placed in a field that is defined for fielded link containment—see *Assigning Field Attributes* on page 207.
- A fielded link must be placed within the first 30 lines of a record. If you want to modify this default scanning depth, see *Setting the Number of Lines per Record Scanned for Fielded Links* on page 277.
- The tag for a field that contains a fielded link should use a field name that describes the link's function; the field name will represent the link in the dialog that is displayed upon selection of Fielded Links from the Go To menu. The field name must conform to the rules listed in *Adding Field Markup* on page 169.
- Only one fielded link may exist in a field.
- A field that contains a fielded link may contain no other text.
- A fielded link must include a beginning marker (e.g., `<D<…>>`) and an end marker (e.g., `</D>`).
- An end-of-line character (e.g., a carriage return) may not exist between a link's beginning and end markers.
- No part of a link may be positioned beyond the 256th character in a line of text.

LINKING TO A SPECIFIC RECORD WITHIN A DATABASE

A *document link* opens a record with a specified ID number; if a field is specified in the link, the opened record window will display that field. By placing document links in source files, you can facilitate convenient cross-referencing between records in a database.

Embedded Document Link Syntax:

`<D<`*`recID`*`,`*`fieldnum`*`>>`*`description`*`</D>`

Fielded Document Link Syntax:

`<D<`*`recID`*`,`*`fieldnum`*`>></D>`

where:

- *recID* is the ID number of the record that will be opened when the document link is activated. When added to a database, records are assigned distinct ID numbers. Each ID number indicates the order in which its corresponding record was added to the database—ID 1 indicates the first record added; ID 2, the second, etc. If you know how many records exist in each source file and the order in which you will add the files to a database, you can extrapolate the ID number of a record to be opened with a document link. For example, if you are going to add files A and B—in that order—to a database, and each file contains five records, you can use the number 8 as *recID* in a document link that will open the third record in file B.

- *fieldnum* is a number indicating the position—in the database's definition file—of the definition of the field that will be displayed in the record window opened by the link. This parameter is optional; if you do not include it in a document link (e.g., `<D<recID>>`*`description`*`</D>`), the linked record will open at its first displayable field.

- *description* is text that will appear in the embedded link's hot spot.

Embedded Document Link Example:

```
For information about the Clinton Legal Defense
Fund, <D<110,5>>click here</D>.
```

Fielded Document Link Example:

```
-ClintonFundSpecs-
<D<110,5>></D>
```

These links will open a record window showing the 5th defined field in the record with ID number 110.

LINKING TO SPECIFIC LOCATIONS WITHIN RECORDS

You can use a *target link* to open a record window that displays a specific portion of the record's text. For example, if you activate a target link that points to a table situated in the middle of a multi-page record, the record window will open at the location of the table.

A target link consists of two components: the link and its target, or *endpoint*, which is placed at the link's destination point. In the aforementioned example, the link's endpoint is positioned at the location of the table in the linked record.

A target link can point to one or more endpoints in other records or in its own parent record. Moreover, multiple target links can point to a single endpoint. A target link endpoint is not visible in a record window.

Embedded Target Link Syntax:

<D<!*label*>>*description*</D>

Fielded Target Link Syntax:

<D<!*label*>></D>

Target Link Endpoint Syntax:

<L<!!*label*>></D>

where:

- *label* is an identifying text string shared by the target link and its endpoint.
- *description* is text that will appear in the embedded link's hot spot.

When creating the *label* string, you should observe the following guidelines:

- The string cannot contain spaces.
- Characters in the string can be in upper-, lower-, or mixed-case.

Note: These rules apply for default settings in PL's tokenizer file—see *Making Special Characters Searchable* on page 297. You can change the tokenizer so that special characters (e.g., spaces) or character combinations are recognizable by PL, in which case they may be used in the *label* string.

Embedded Target Link Example:

```
<D<!FldAttribTable>>Click here</D> to view a
summary of field attributes.
```

Fielded Target Link Example:

```
-View_Field_Attributes-
<D<!FldAttribTable>></D>
```

Target Link Endpoint Example:

```
<L<!!FldAttribTable>></D>Field attributes are
summarized in the following table:
```

These links will open the target record at the location of the endpoint and nearby table.

EXECUTING A SEARCH

A *search link* executes a search, using a predefined query, of all open databases.

Embedded Search Link Syntax:

```
<S<query>>description</D>
```

Fielded Search Link Syntax:

```
<S<query>></D>
```

where:

- *query* is the query upon which the search will be based.
- *description* is text that will appear in the embedded link's hot spot.

Embedded Search Link Example:

```
<S<United States occupation Haiti>>Click here</D>
for more information about the U.S. occupation of
Haiti.
```

Fielded Search Link Example:

```
-Search-
<S<United States occupation Haiti>></D>
```

These links will execute a search based on the following query:

```
United States occupation Haiti
```

DISPLAYING AN IMAGE

You can use an *image link* to open an image file. An image link can run PL's integrated viewer—which displays monochrome images in uncompressed TIFF, compressed TIFF group 3, and compressed TIFF group 4 formats—or a third-party imaging application. If you want to configure PL to use a third-party viewer, see *Defining Alternate Viewers for Image Links* on page 266.

Embedded Image Link Syntax:

`<I<`*`type,filename,title`*`>>`*`description`*`</D>`

Fielded Image Link Syntax:

`<I<`*`type,filename,title`*`>></D>`

where:

- *type* specifies the viewer that will display the image. If you want to use PL's integrated viewer, set *type* to be 1. If you want the image displayed by a third-party viewer, use the image type associated with the viewer in `plwdefs.ini`.

- *filename* specifies the path and filename of the image file that will be displayed. If you specify a relative path, it must be relative to the index of the database that contains the link. If the link uses a third party viewer, *filename* will be appended to the command line defined for *type* in `plwdefs.ini`.

Tip: When creating a portable database (e.g., on CD-ROM), you should use relative paths in image links; full paths may not be consistently interpreted in different environments.

- *title* determines what information will appear in the image window's title bar. This parameter applies to PL's integrated viewer; it is ignored in an image link that activates a third-party viewer. Valid values for the *title* parameter are summarized in Table 4.

- *description* is text that will appear in the embedded link's hot spot.

Table 4 Valid Values for *Title* Parameter in Image Links

Value	Image Window TItle Bar Display
==	First 80 characters of *description*
text string	Word or phrase of up to 80 characters, including punctuation
(blank - no parameter)	Image filename

Embedded Image Link Examples:

```
<I<1,grafix\report.tif,==>>Reporting Process</D>

<I<1,satpix\sat58.tif,Satellite View>>Click</D> to
display a satellite photograph of this region.

<I<1,grafix\ecu.tif>>Click here</D> to see a close-
up of this component.
```

Fielded Image Link Examples:

```
-Image-
<I<1,grafix\report.tif,==>>Reporting Process</D>

-Image-
<I<1,satpix\sat58.tif,Satellite View>>Click</D> to
display a satellite photograph of this region.

-Image-
<I<1,grafix\ecu.tif>>Click here</D> to see a close-
up of this component.
```

The first embedded and fielded links will open an image window that displays *Reporting Process* in its title bar.

The second embedded and fielded links will open an image window that displays *Satellite View* in its title bar.

The third embedded and fielded links will open an image window that displays the filename `ecu.tif` in its title bar.

EXECUTING AN EXTERNAL APPLICATION

You can use a *command link* to launch an external application—word processor, spreadsheet, etc.—from within Personal Librarian.

Note: In order for a command link to work, your computer must have sufficient memory to run both PL and the application that the link will open.

Embedded Command Link Syntax:

`<C<`*`command`*`>>`*`description`*`</D>`

Fielded Command Link Syntax:

`<C<`*`command`*`>></D>`

where:

- *command* is a character string that will be passed as a command to your computer's operating system. If you specify a relative path to the executable, it must be relative to the index of the database that contains the link.

Tip: When creating a database that is portable (e.g., on CD-ROM) or interoperable among various operating systems, you should use relative paths in command links; full paths may not be consistently interpreted in different environments. If the linked application requires a full path as part of its command line syntax, a command link may not be portable among different operating systems.

- *description* is text that will appear in the embedded link's hot spot.

Embedded Command Link Example:

```
<C<winword docs\rfp.doc>>Click here</D> to edit the
proposal.
```

Fielded Command Link Example:

```
-WinWord-
<C<winword docs\rfp.doc>></D>
```

These links will open the file `rfp.doc` in Word for Windows.

EXECUTING A DLL FUNCTION

A *DLL link* will execute a function in a dynamic link library. Implementation of DLL links should be done by users who have some programming experience. If you want to use this type of link, see *Using DLL Links and Callbacks* on page 284.

LINKING AND SEARCHING FROM AN EXTERNAL APPLICATION

If you are running an application from which commands can be passed to your computer's operating system, you can execute a search or activate certain types of links in Personal Librarian. To issue a command to the operating system, you can use Windows' *WinExec( )* subroutine; in a Visual Basic application, you can use the *Shell* command.

Note: In order for PL to be run from an external application, your computer must have sufficient memory to run both applications.

Searching with a Single-Word Query

Command Syntax:

```
plpath listname /e S<query>
```

where:

- *plpath* is the path and filename of the PL executable, `plwin.exe`. If the directory containing the executable is in your `autoexec.bat` file's PATH statement, you do not have to specify a path to `plwin.exe`.

- *listname* is the ShortList name of the database that you want to search.

Note: If listname includes spaces, it must be enclosed in double quotation marks.

- *query* is the word that will be sought in the database specified by *listname*.

Examples:

```
c:\plwin\plwin Crime /e S<DeSalvo>

plwin "True Crime" /e S<Whitechapel>
```

Searching with a Multiple-Word Query

Command Syntax:

```
plpath listname /e "S<query>"
```

where:

- *plpath* is the path and filename of the PL executable, `plwin.exe`. If the directory containing the executable is in your `autoexec.bat` file's PATH statement, you do not have to specify a path to `plwin.exe`.
- *listname* is the ShortList name of the database that you want to search.

Note: If listname includes spaces, it must be enclosed in double quotation marks.

- *query* is the series of query terms that will be sought in the database specified by *listname*.

Examples:

```
c:\plwin\plwin UFO /e "S<Roswell Incident>"

plwin "Dreamland Dossier" /e "S<Area 51 Nellis>"
```

You can include query operators in *query*, as shown in the following example:

```
plwin Cryptozoology /e "S<Lake ADJ Champlain w/20
Ogopogo NOT Loch ADJ Ness>"
```

Activating a Document Link

Command Syntax:

```
plpath listname /e D<recID,fieldnum>
```

where:

- *plpath* is the path and filename of the PL executable, `plwin.exe`. If the directory containing the executable is in your `autoexec.bat` file's PATH statement, you do not have to specify a path to `plwin.exe`.

- *listname* is the ShortList name of the database that you want to search.

Note: If listname includes spaces, it must be enclosed in double quotation marks.

- *recID* is the ID number of the record that will be opened when the document link is activated.
- *fieldnum* is a number indicating the position—in the database's definition file—of the definition of the field that will be displayed in the record window opened by the link. This parameter is optional; if you do not include it in the command (e.g., D<*recID*>), the linked record will open at its first displayable field.

Examples:

```
c:\plwin\plwin FedCode /e D<55,2>

plwin "Federal Code" /e D<91,3>
```

Activating a Target Link

Command Syntax:

plpath listname /e D<!*label*>

where:

- *plpath* is the path and filename of the PL executable, `plwin.exe`. If the directory containing the executable is in your `autoexec.bat` file's PATH statement, you do not have to specify a path to `plwin.exe`.

- *listname* is the ShortList name of the database that you want to search.

Note: If listname includes spaces, it must be enclosed in double quotation marks.

- *label* is the identifying text string contained in the desired endpoint.

Examples:

```
c:\plwin\plwin DKheavy /e D<!Cobblepot>

plwin "Rogues Gallery" /e D<!Nigma>
```

ADDING TABLE OF CONTENTS MARKUP

If you want to make a database searchable as if it were a book—see *Table of Contents Searching* on page 135—you must add table of contents markup to the text files that the database will contain.

CREATING TABLE OF CONTENTS SECTIONS

You can determine where in the table of contents hierarchy a record occurs by adding to the record a field designated for that purpose. The field will contain a heading that specifies the portion of the table of contents hierarchy that the parent record will occupy.

Heading Field Syntax:

```
-heading-
<hn>heading</h>
```

where:

- *n* is the level of the table of contents hierarchy that the field's parent record will occupy. Valid values for *n* are the integers from 0 to 6. A record in which *n* is 0 will occupy the table of contents' top level; a record with *n* equal to 1, the second level, etc.

- *heading* is a word or phrase that will be displayed for the record in the table of contents and genealogy windows.

Example:

```
-heading-
<h0>Front Matter</h>
-text-
The software in this document is furnished under…
-end-
-heading-
<h1>Preface</h>
-text-
Welcome to the Personal Librarian User’s Guide…
-end-
-heading-
<h2>Organizational Overview</h>
-text-
Take a moment to browse through the table of…
-end-
-heading-
<h2>Conventions</h>
-text-
This manual uses a consistent form of shorthand…
-end-
-heading-
<h2>Assumptions</h>
-text-
This manual assumes that you are familiar with…
-end-
-heading-
<h0>Part 1: Getting Started</h>
-end-
-heading-
<h1>Chapter 1: Introduction</h>
-text-
This chapter introduces you to principles…
-end-
-heading-
<h2>Searching with Personal Librarian</h>
-text-
Searching is a process, not an event…
-end-
```

A table of contents created from the preceding example will appear as shown in Figure 55.

Figure 55 A table of contents based on the source file example.

When adding table of contents markup to a source file, you should observe the following guidelines:

- A `-heading-` field is not required in every record. A record without heading information will not be explicitly represented in a table of contents window. Moreover, a genealogy cannot be created or displayed for a record that does not contain a table of contents heading.
- If a database is a collection books, the first record in each book should have a top-level heading (i.e., `<h0>...</h>`).
- The `-heading-` field tag must begin and end with a hyphen.
- The `-heading-` field tag can be in upper-, lower-, or mixed-case.
- With the exception of target link endpoints, which may be placed in the same line as the `-heading-` field tag, no text other than a heading may exist in a `-heading-` field.
- No blank lines may exist between a `-heading-` field tag and the heading that follows it.
- Only one table of contents heading may exist in a record.
- The beginning marker in a heading must be immediately preceded by an end-of-line marker (e.g., a carriage return).

- A heading must include a beginning marker (`<h`*n*`>`) and an end marker (`</h>`).
- An end-of-line character (e.g., a carriage return) may not exist between a heading's beginning and end markers.
- A heading can be in upper-, lower-, or mixed-case.

Tip: You can contrive "dummy" records that serve only to establish top-level divisions in a table of contents. A "dummy" record contains only a `-heading-` field with a top-level heading (i.e., `<h0>...</h>`). If you use such records in your databases, you can configure PL to skip each one—and open the next record—when you double-click its heading. See *Disabling Opening of Top-Level Records from the Table of Contents* on page 231.

Note: If you mark up a source file so that a chapter contains more than 100 sub-chapters in the immediately subordinate hierarchy level (e.g., a level two chapter contains 200 level three chapters), the parent chapter may be relatively slow to expand or collapse in a table of contents window.

ADDING COPYRIGHT INFORMATION

You can configure PL so that when a record is printed, an additional record that contains copyright information will be printed automatically—see *Enabling Printout of Record Copyright Information* on page 230. When you exercise this option, you must identify one or more records that contain copyright information in each database; you can select one of the following criteria: the first valid record in a database, each record with a top-level table of contents heading, or the record that immediately follows each record with a top-level heading.

If you plan to use this feature, you should include copyright information in the appropriate record—or records—in each database.

ADDING GENEALOGY MARKUP

If you want to be able to display a genealogy for a record, you must add genealogy information to it. Fortunately, you do not have to add this information manually. Instead, you can process source files with the `gen.exe` utility, located in PL's `\gen` subdirectory.

The `gen.exe` program uses information from a record's table of contents heading to create a genealogy field that is automatically added to the record; genealogy information is not added to any record that lacks a table of contents heading.

Genealogy processing must be done before a source file is added to a database.

The `gen.exe` utility can process only source files that contain a single top-level heading (i.e., `<h0>...</h>`).

Note: To enable display of genealogy information, you must set preferences in PL-Admin so that a genealogy button is present in record windows—see *Enabling Genealogy Display for Records* on page 229.

Syntax:

```
gen rules divider input output log
```

where:

- *rules* is the name of the file that governs the appearance and layout of information in genealogy windows; *rules* can include a path. See *Creating a Rules File* on page 193.

- *divider* is the name of the file that contains characters that are inserted verbatim between the original contents of a record and the genealogy field; *divider* can include a path. See *Creating a Divider File* on page 195.

- *input* is the name of the source file to which genealogy information will be added; *input* can include a path and must use the `.txt` filename extension. This file will not be altered by `gen.exe`.

- *output* is the name of the post-processing source file that will contain genealogy information; output can include a path and should use the `.src` filename extension. If you do not specify *output*, the file will be named *`input`*`.src`.

- *log* is the name of the diagnostic file that contains the output message from `gen.exe`. If you do not specify *log*, the file will be named *`input`*`.log`.

Note: The filenames used in *rules*, *divider*, *input*, *output*, and *log* can contain only letters, numbers, or the underscore character (_).

Example:

```
gen genrules.rul gendiv.div fedreg01.txt
fedreg01.src regs01.log
```

Creating a Rules File

Before processing a source file with `gen.exe`, you must use a text editor or word processor to create a rules file, which determines how genealogy information will be formatted for records in the source file.

Syntax:

```
genfield
heading,offset,text,arrow,hdglength
```

where:

- *genfield* is the name of the genealogy field that will be added to records.

- *heading* specifies a table of contents heading level that will be represented in a genealogy.

- *offset* specifies the column position at which the heading will start in a genealogy window; *offset* must be greater than the ending column number in *arrow*.

- *text* specifies the kind of formatting that will be used for the heading. Value values are *n* for normal, *b* for bold, *it* for italic, and *bi* for bold italic.

- *arrow* specifies the column boundaries for the arrow that will point to the heading in a genealogy window. The beginning and ending column numbers must be separated by a hyphen; the ending column number must be greater than the beginning column number. If you leave this parameter blank, no arrow will be inserted in the genealogies of records at the level specified by *heading*.

- *hdglength* is the maximum number of characters the heading, including *offset*.

Example:

```
Overview
h0,1,n,,80
h1,5,n,1-3,75
h2,9,n,1-7,70
h3,13,n,1-11,65
h4,17,n,1-15,60
```

A record genealogy based on these example rules will be similar to the example shown in Figure 56.

Figure 56 This genealogy is based on the preceding rules example.

When creating a rules file, you should observe the following guidelines:

- The file must be in ASCII text format.
- The filename must have the `.rul` extension.
- *genfield* must be on the first line in the file.
- Lines in the file may not include spaces.
- Parameters in each line must be separated by commas.
- Lines may be in upper-, lower-, or mixed-case.

Creating a Divider File

Before processing a source file with `gen.exe`, you must use a text editor or word processor to create a divider file, which contains text that will be inserted verbatim after each genealogy field.

If you review a source file after processing it with `gen.exe`, the divider text will serve to distinguish genealogy information from a record's pre-processing contents.

Syntax:

```
divfield
string
```

where:

- *divfield* is a field tag that prevents the text in *string* from becoming part of the genealogy field. If you do not specify *divfield*, *string* will be visible in genealogy windows.

- *string* is the text that will be inserted verbatim after each genealogy field.

Example:

```
-divider-
=================================================
```

When creating a divider file, you should observe the following guidelines:

- The file must be in ASCII text format.

- The filename must have the `.div` extension.

- Text in the file may be in upper-, lower-, or mixed-case.

- Blank lines will be ignored.

Interpreting the Log File

The log file output by `gen.exe` is a ASCII text file containing the ID numbers of records that have table of contents markup errors—subordinate headings without parent headings, etc.—or headings longer than the limit established in the rules file.

Example:

```
Over 80 characters
      Record 11
      Record 43
```

ADDING MARKUP FOR A TABLE SUMMARY

Before processing a source file with `gen.exe`, you can add markup that will be used by that program to create records that contain target links to tables in the source file. When you add the source file to a database, these records will be directly accessible from the database's table of contents.

In a source file, you should insert table summary markup before each table that you want to be accessible from the table summary.

Syntax:

```
<L<!!label>></D>
<TC<summary>>description</TC>
```

where:

- *label* is an identifying text string that will be shared by the target link and its endpoint.
- *summary* is the text that will represent the target link's parent record in the table of contents' table summary.
- *description* is text that will appear in the embedded target link's hot spot.

When adding table summary markup, you should observe the following guidelines:

- The target link endpoint must conform to the guidelines listed in *Linking to Specific Locations Within Records* on page 177.
- The table summary tag must conform to the guidelines listed in *Creating Embedded Links* on page 174.
- A single end-of-line character (e.g., a carriage return) must exist between the target link endpoint and the table summary tag.
- There can be up to nine distinct values for *summary* in a source file when it is processed by `gen.exe`.

- *summary* cannot contain spaces.
- *summary* can be in upper-, lower-, or mixed-case.

Example:

```
<L<!!keybd>></D>
<TC<Tables>>Keyboard and Mouse Actions</TC>
```

When `gen.exe` processes a source file that contains table summary markup, it creates one or more records, similar to the following example, that contain embedded target links to all endpoints that immediately precede table summary tags.

Example:

```
-Heading-
<h1><b>Tables</b></h>
-Text-
                                     Tables
<D<!keybd>>Keyboard and Mouse Actions</D>
<D<!opfunctions>>Query Operator Functions</D>
<D<!fldattrib>>Field Attributes</D>
-end-
```

By default, a table summary is displayed at the top level of a table of contents, as shown in Figure 57.

Figure 57 An expanded table summary

If you want a table summary to occupy a lower level of a table of contents hierarchy, see *Setting a Table Summary's Hierarchy Level* on page 199.

By double-clicking an entry in a table summary, you can open a record that contains embedded target links, similar to Figure 58.

Figure 58 This record contains links to records with table summary markup.

Setting a Table Summary's Hierarchy Level

By default, `gen.exe` uses a top-level heading called "Table Summary" for table summary records. You can specify a heading and hierarchy level for table summaries by inserting a line that includes those parameters after the first line in the rules file that `gen.exe` will use.

Syntax:

```
heading,n
```

where:

- *heading* is a word or phrase that will be displayed for the summary in the table of contents window.
- *n* is the level of the table of contents hierarchy that the table summary record will occupy.

Both *heading* and *n* must conform to the guidelines listed in *Creating Table of Contents Sections* on page 187.

Example:

```
Overview
Tables & Figures,2
h0,1,n,,80
h1,5,n,1-3,75
h2,9,n,1-7,70
h3,13,n,1-11,65
h4,17,n,1-15,60
```

PREPARING WORD PROCESSOR FILES

As mentioned in this chapter’s introduction, you can add files in native Word for Windows 2.0 or WordPerfect 5.0/5.1 format to a Personal Librarian database. Source files in these word processor file formats are subject to the following conditions:

- Record markup is not supported; when added to a PL database, each file becomes a single record.
- Field markup is not supported; the complete contents of a file will be assigned to the default *TEXT* field.
- Because fields are not recognized in a file, fielded links are not supported.
- A file can contain embedded links. See *Creating Embedded Links* on page 174.
- A file cannot be appended to an existing database file. See *Appending New Files to an Indexed File* on page 238.

If these conditions seem too restrictive, you should consider saving your word processor files as ASCII text and using the ASCII versions as source files.

CHAPTER 14

CREATING A DATABASE

From this chapter you will learn how to use PL-Admin to prepare a database for the addition of text files. Also discussed are optional techniques that you can employ to customize database performance.

THE DATABASE DEFINITION PROCESS

Database definition—the process whereby you create a database and define its performance characteristics—begins when you select New from PL-Admin's File menu (CTRL+N). The definition process consists of the following stages:

Creating the Database. When creating a new database, you must select a name and location for its control files. See *Creating a New Database* on page 202.

Assigning the Database's ShortList Name. You have the option of specifying the name that will represent the new database in the ShortList. See *Assigning a ShortList Name* on page 204.

Importing Definitions. If you want use an existing database's definitions in the new database, you can import them.

Defining Fields. If the records in your source files are subdivided into fields, you must establish corresponding field definitions for the database. See *Defining Fields* on page 205.

Adding Definition Statements. You have the option of adding definition statements that influence how the new database will be built or searched. See *Adding Optional Definition Statements* on page 208.

Archiving the Index. If the new database will not be updated after the first time files are added to it, you can use archival formatting to reduce its index to the smallest possible size. See *Selecting Archival Index Formatting* on page 221.

Adding the Database to the ShortList. You can elect to have the database included in the ShortList that is used to open databases in both Personal Librarian and PL-Admin. See *Adding a Database to the ShortList* on page 221.

CREATING A NEW DATABASE

To create a new database:

1. In PL-Admin, select New from the File menu (CTRL+N). A standard file-saving dialog will be displayed.

2. Use the dialog to name and situate the index file; do not include an extension in the filename. PL-Admin will use the name you enter as the base filename for the database's control files—its index file (`*.pls`), definition file (`*.def`), etc.

Note: The directory occupied by a database's index file is the database's *library directory*, which also contains the other control files associated with the database.

3. Click the OK button. The Define Database dialog, shown in Figure 59, will be displayed.

Figure 59 The Define Database dialog

You will use the Define Database dialog to:

- Specify the database's ShortList name—see *Assigning a ShortList Name* on page 204.
- Import definitions—see *Importing Definitions from an Existing Database* on page 204.
- Define fields—see *Defining Fields* on page 205.
- Add definition statements—see *Adding Optional Definition Statements* on page 208.
- Store the index in archival form—see *Selecting Archival Index Formatting* on page 221.

To accept the current settings in the Define Database dialog, click the OK button.

ASSIGNING A SHORTLIST NAME

In the Define Database dialog's Database Name text box, you can enter the name that will represent the new database in the ShortList, which is displayed when you open a database in Personal Librarian or PL-Admin. The base name of the database's index file will be entered as the default ShortList name.

When specifying a ShortList name in the Define Database dialog, you should observe the following guidelines:

- The name must begin with an alphabetic character.
- If the name includes spaces, it must be enclosed in double quotation marks.
- The name can be up to 31 characters in length. If the name is enclosed in double quotation marks, the marks are not included in the 31-character limit.
- The name can be in upper-, lower-, or mixed-case.

IMPORTING DEFINITIONS FROM AN EXISTING DATABASE

If you are creating a database that will have definitions similar or identical to those of an existing database, you can import the latter's definitions for use with the former. This will save you time and typing if you create several databases with similar definitions.

To import definitions from an existing database:

1. Click the Define Database dialog's Import button. A standard file-selection dialog will be displayed.

2. By default, the dialog will list files with the `.def` filename extension that is common to all database definition files. Use the dialog to find and select the desired definition file, which will have the same base name as the existing database's index file.

3. Click the OK button.

DEFINING FIELDS

When creating a database, you must define fields that correspond to the field markup in the source files that the database will contain—see *Adding Field Markup* on page 169. If your source files are not subdivided into fields, each file's contents will occupy the *TEXT* field, which is defined by default.

Currently defined fields are listed in the Define Database dialog's Fields list box. A field's performance attributes are abbreviated parenthetically; if a field is highlighted, its attributes are reflected in the checkboxes beside the Fields list box.

Note: If your source files contain table of contents markup—see *Adding Table of Contents Markup* on page 187—you must define the `-heading-` field for the database to which the files will be added. Likewise, if your source files contain genealogy markup—*Adding Genealogy Markup* on page 192—or have dividers in separate fields, you must define fields that correspond to the files' genealogy and divider field tags.

To define a new field:

1. Click the Add button in the Fields region of the Define Database dialog. The Add New Field dialog will be displayed.

2. Enter a field name; you should observe the following guidelines:

- A field name should correspond to a field tag used in the source file(s) that will be added to the database.

- Fields do not have to be defined in the order in which their corresponding fields appear in records.

- A field name may be up to 29 characters long, including punctuation.

- The underscore (_) and the forward slash (/) are the only punctuation marks that may be used in a field name.

- A field name may not contain spaces.

- Field names can be in upper-, lower-, or mixed-case.

- A field name must begin with an alphabetic character; subsequent characters can be alphanumeric.

- A field name may not begin with another complete field name (e.g., you cannot use *date* and *date_published* as field names in the same database).

3. Click the OK button. The new field will appear in the Fields list box.

4. Repeat the preceding steps to define additional fields as needed.

Note: You can define a maximum of 255 fields for a database.

To remove a field definition:

1. In the Fields list box, click the field definition that you want to remove.

2. Click the Remove button in the Fields region of the Define Database dialog.

ASSIGNING FIELD ATTRIBUTES

For each field listed in the Define Database dialog's Fields list box, you can define attributes that control the field's performance during indexing and searching. The attributes that you can assign to a field are summarized in Table 5.

Table 5 Field Attributes

Name	Abbreviation	Performance Characteristics
Display	D	The field will be displayed in record windows.
List	T	If it is within the first 15 lines of a record, the field will be displayed in the Hitlist.
Search	S	The field will be searched, unless excluded by a field-restricted query. Fields without this attribute *will* be searched when specified in a field-restricted query.
No Index	N	The field will not be indexed and therefore, will never be searchable.
Hypertext Link	H	The field contains a hypertext link.

By default, the Display, List, and Search attributes are assigned to a field.

To modify a field's attributes:

1. In the Fields list box, click the field you want to modify.

2. Click the checkboxes of the attributes that you want to enable or disable.

When assigning field attributes, you should observe the following guidelines:

- At least one field with the List attribute must be defined for a database.
- At least one field with the Search attribute should be defined for a database; otherwise, any query that is not field-restricted will return no hits.
- A field may not have both the Search and No Index attributes.

Tip: If, in source files with genealogy markup, the divider exists in its own field, you can keep dividers from being displayed in genealogy windows by not assigning the Display attribute to the divider field.

ADDING OPTIONAL DEFINITION STATEMENTS

During database definition, you can customize the new database's performance during indexing or searching by entering *definition statements*. Definition statements are database-specific, they affect only the database for which they are defined. As their name implies, optional definition statements are not required as part of a database's definitions.

To enter an optional definition statement:

1. Click the Add button in the Optional Definition Statements region of the Define Database dialog. The Add Optional Definition dialog will be displayed.

2. Enter one of the statements documented in the following sections. Be certain to employ proper syntax; if you add a statement with incorrect syntax, unpredictable results may be experienced when the database is opened.

3. Click the OK button. The new statement will appear in the Optional Definition Statements list box.

4. Repeat the preceding steps to add additional statements as needed.

To remove an optional definition statement:

1. In the Optional Definition Statements list box, click the statement that you want to remove.

2. Click the Remove button in the Optional Definition Statements region of the Define Database dialog.

DEFINING A FIELDLIST

When defining a database, you can add a definition statement that will enable searchers of the database to use a fieldlist—see *Field Restriction Shorthand: Using Fieldlists* on page 128.

Syntax:

`fieldlist=`*`listname field`*`01,`*`field`*`02...`

where:

- *listname* is the fieldlist name; fieldlist names are subject to the field name guidelines in *Defining Fields* on page 205.

- *field* is the name of a field that the fieldlist will represent within a query.

Examples:

```
fieldlist=abstract title,author,publisher

fieldlist=staff author,artist,editor
```

When defining a fieldlist, you should observe the following guidelines:

- You can define multiple fieldlists for a database.
- A field name can appear in multiple fieldlists.
- The fieldlist must be separated from its corresponding field names by a space or tab.

Note: Defining a fieldlist does not require that the associated database be reindexed; you can add, modify, or delete a fieldlist at any time after building a database—see *Modifying a Definition File after Database Creation* on page 222.

DISPLAYING FILENAMES FOR RECORDS IN THE HITLIST

You can arrange for the Hitlist to display, for each retrieved record from a particular database, the name of the source file that contains the record, as shown in Figure 60. To do this, you must include a statement to that effect in the database's definitions.

Syntax:

```
hitlist=container
```

Figure 60 A Hitlist that displays the name of each record’s source file.

DISPLAYING A RECORD’S TABLE OF CONTENTS POSITION IN THE HITLIST

If a database has a table of contents, you can arrange for the Hitlist to display, for each retrieved record from the database, the position of the record in the table of contents, as shown in Figure 61. To do this, you must include a statement to that effect in the database’s definitions.

Syntax:

hitlist=hierarchy *TOCfile*

where *TOCfile* is the database's table of contents file (`*.hir`), located in the database's library directory.

Figure 61 A Hitlist that displays records' table of contents positions

Note: A maximum of four hierarchy levels will be displayed for each record.

Example:

```
hitlist=hierarchy pl41ug.hir
```

DISPLAYING CUSTOM RECORD SUMMARIES IN THE HITLIST

When defining a database, you can add a definition statement that causes Hitlist summaries to be extracted from a text file that you have created, rather than from the database's records. With this option, you can customize a summary for every record in the database.

Syntax:

```
hitlist=file sumfile
```

where *sumfile* is the path and filename of the text file that contains custom Hitlist summaries for each record in the database. If you specify a relative path, it must be relative to the database's library directory.

Example:

```
hitlist=file c:\db\recsums.txt
```

Tip: You can use this technique to expedite searching of CD-ROM-based text files; if you place the separate summary file on a local hard drive, PL can access Hitlist information more quickly than by reading from the records.

Using a text editor or word processor, you must create the ASCII text file that contains custom record summaries. The file's contents must be in the following format:

Syntax:

```
CplDiskArray Class Version 1.00/June, 1991
90         maxbytes  nsums    sum10    sum20…
```

where:

- *CplDiskArray Class Version 1.00/ June, 1991* must occupy character positions 0–42.

- *90*, a constant that must begin at character position 60, specifies the position of the first record summary.

- *maxbytes*, which must begin at character position 70, specifies the maximum number of characters that will be used for each summary.

- *nsums*, which must begin at character position 80, specifies the total number of record summaries that exist in the file.

- *sum1* and *sum2*, which must respectively begin at character positions 90 and 90+*maxbytes*, are summaries for the first and second records in the database.

- 0 terminates summaries that are shorter than *maxbytes* and assures that spaces that are between the end of one summary and the beginning of the next are ignored.

Example:

```
CplDiskArray Class Version 1.00/June, 1991
                 90        40        2000
Personal Librarian User's Guide0        About this
manual0…
```

In this example, each summary must be exactly 40 characters long and the file contains 2000 summaries.

When creating the summary file, you should observe the following guidelines:

- The file must be in ASCII format.

- Carriage returns may be used; under Windows and DOS, a carriage return comprises two characters.

- Summaries that are shorter than *maxbytes* must be terminated by a zero; text that is between the zero and the beginning of the subsequent summary will be ignored.

- The file must include a summary for each record in the database; you can add new summaries as new records are added to the database.

HIDING FIELD LABELS IN RECORD WINDOWS

By default, Personal Librarian displays field labels in record windows. You can suppress the display of field labels for records in a database by including a statement to that effect in the database's definitions.

Syntax:

```
hide_fields=x
```

where *x* is capital Y or N, depending on whether you want to, respectively, hide or display field labels in the database's records.

Examples:

```
hide_fields=Y

hide_fields=Y
```

The first example will override the factory default setting and suppress the display of field labels in record windows.

If the system-wide default has been set to hide field labels—see *Modifying Display of Field Labels in Record Windows* on page 279—the second example will override it and cause field labels to be displayed.

Note: This statement will also govern the display of field labels in printed records.

DEFINING A THESAURUS

In order for searches of a database to use a thesaurus, you must add a statement that defines a thesaurus to the database's definitions. You must create a thesaurus file before it can be defined for use with a database—see *Creating A Thesaurus* on page 227.

Syntax:

```
thesaurus=x thesname
```

where *thesname* is the path and filename of the thesaurus file that you want to associate with the database. If you specify a relative path, it must be relative to the database's library directory.

Example:

```
thesaurus=x c:\db\thesauri\fedthes.txt
```

Tip: You may define a single thesaurus file for multiple databases.

Note: Because concept searching and the Relate Advisor make it easy to identify and search with words that are related to query terms, you may decide that your database does not need a thesaurus. However, a thesaurus is useful for searching a database that includes many acronyms; thesaurus searching can minimize the need to correctly interpret acronyms or manually enter the words that they represent.

ADDING COMMENTS TO A DEFINITION FILE

In a database's definition file, you can add a line of commentary or disable a statement by inserting a pound sign (#) and a space at the beginning of the line.

Syntax:

```
# string
```

where *string* is your commentary text or the definition statement that you want to disable.

Example:

```
# Un-comment the line that specifies the
# thesaurus you want to use for searching.
# thesaurus=x c:\db\thesauri\fedthes.txt
# thesaurus=x c:\db\thesauri\chemthes.txt
```

ENABLING SAME PARAGRAPH PROXIMITY SEARCHING

By including a statement in a database's definitions, you can enable searchers of the database to use the same paragraph and notsame paragraph operators—see *The Same Paragraph Operator* on page 114 and *The Notsame Paragraph Operator* on page 115.

Syntax:

```
index_size=full
```

When you add source files to the database, this statement will cause PL-Admin to track paragraph termination—the presence of two consecutive end-of-line characters—in the files.

CAUTION: If it is used, this statement should be present in a database's definition file before the database is built. If you add the statement to the definition file for a database to which text files have already been added, you must completely rebuild the database in order for the aforementioned operators to work correctly.

BUILDING INDEX PARTITIONS ON FASTER MEDIA

When you build a database on a capacious but relatively slow storage medium—CD-ROM, magneto-optical diskette, etc.—you have the option of storing parts of the index file on a faster medium (e.g., your local hard drive) in order to improve searching and display performance.

By default, database index information is unified in a single file: `*dbname*.pls`. The alternative process of storing parts of the index in files that are separate from `*dbname*.pls` is called *partitioning*. You can create index partitions by including a statement to that effect in a database's definitions prior to adding files to the database. You can partition two components of a database index and save them on a faster storage medium.

The first component is the *dictionary*, which contains an alphabetical list of all words indexed in the database and a posting count—number of occurrences—for each word. Partitioning the database's dictionary to a faster medium can improve searching speed. The size of a dictionary partition—in bytes—is approximately 20 times the number of unique words in the database.

The second component consists of the database's *storage keys*, which list the location and document class information for each record in a database. Partitioning the database's storage key to a faster medium can improve the speed at which records are displayed. The size of a storage keys partition—in bytes—is approximately 12 times the number of records in the database.

CAUTION: If it is used, a partition statement should be present in a database's definition file before the database is built. If you add the statement to the definition file for a database to which text files have already been added, you must completely rebuild the database or reorganize its index—see *Reorganizing a Database's Index* on page 244.

Creating a Dictionary Partition

Syntax:

```
partition=partfile dic
```

where *partfile* is the path and filename of the dictionary partition file. If you specify a relative path, it must be relative to the application's current working directory.

Example:

```
partition=acctlaw.dic dic
```

Creating a Storage Keys Partition

Syntax:

```
partition=partfile storage_keys
```

where *partfile* is the path and filename of the storage keys partition file. If you specify a relative path, it must be relative to the application's current working directory.

Example:

```
partition=acctlaw.stk storage_keys
```

Combining Partitions

If you are partitioning both the dictionary and storage keys for a database, you can store the partitions in separate files, as discussed in the preceding sections, or you can combine them in a single file by defining them in the same partition statement.

Syntax:

```
partition=partfile storage_keys,dic
```

where *partfile* is the path and filename of the partition file that contains the database's dictionary and storage keys. If you specify a relative path, it must be relative to the application's current working directory.

Example:

```
partition=acctlaw.par storage_keys,dic
```

MAKING SPECIAL CHARACTERS SEARCHABLE IN A DATABASE

During the indexing process, PL-Admin ignores punctuation and non-alphanumeric characters in source files; this makes special characters—ampersand (&), percent sign (%), etc.—non-searchable. Moreover, these characters are stripped from queries during searching in Personal Librarian.

If you want certain special characters to be searchable for a particular database—and recognized within queries when the database is searched—use the following procedure:

1. Place a copy of the `tknztbld.cpl` file, located in PL's `\shared` subdirectory, in the database's library directory.

2. Make the desired changes to the copy of `tknztbld.cpl`—see *Making Special Characters Searchable* on page 297.

3. Include the following statement in the database's definitions:

   ```
   Tokenizer=CplTabledRomanceTokenizer
   ```

4. Build the database.

CAUTION: If it is used, this statement should be present in a database's definition file before the database is built. If you add the statement to the definition file for a database to which text files have already been added, you must completely rebuild the database.

SELECTING ARCHIVAL INDEX FORMATTING

By clicking the Define Database dialog's Archive Format checkbox during database definition, you can elect to compress the database's index to the smallest possible size when files are added to it for the first time; this has no effect on your ability to search the database in Personal Librarian. Because a PL database's index is self-optimizing—unused index space is eliminated or minimized when files are added, removed, etc.—the benefits of archival formatting are marginal.

CAUTION: You should use this option only for a database that will never be updated after initial indexing; updating an archived database is extremely wasteful of hard drive space.

ADDING A DATABASE TO THE SHORTLIST

At the conclusion of the database definition process, a dialog will prompt you to determine if the database will be included in the ShortList. If you want to add the database to the ShortList, click the Yes button; if not, click the No button.

If you do not include a database in the ShortList when it is created, you can add it later with the following procedure:

1. In PL-Admin, open the database—see *Opening a Database* on page 233.
2. Select Add Database to ShortList from the Options menu.

REMOVING A DATABASE FROM THE SHORTLIST

To remove a database from the ShortList:

1. In PL-Admin, open the database—see *Opening a Database* on page 233.
2. Select Remove Database from ShortList from the Options menu.

MODIFYING A DEFINITION FILE AFTER DATABASE CREATION

You can change definitions for an existing database by using a text editor or word processor to modify its definition file (`*.def`), which is located in the database's library directory. The file must remain in ASCII format.

Certain changes to database definitions can be made at any time; they affect only searching performance. Other modifications, which affect indexing, should be made before any files are added to a database; otherwise, the database must be completely rebuilt.

MODIFYING OR ADDING FIELD STATEMENTS

In a definition file, fields are defined by statements in the following format:

```
name  (attributes)
```

where:

- *name* is the field name.
- *attributes* is a string of abbreviations for the field's attributes.

When entering or modifying field statements in a definition file, you should observe the following guidelines:

- A field statement must begin at the first character position in a line.
- A field's name and parenthetical attributes must be separated by a space or tab.
- A field statement must conform to the guidelines for field names—see *Defining Fields* on page 205—and field attributes—see *Assigning Field Attributes* on page 207.
- If you define new fields for a database to which files have already been added, you must rebuild the database.
- If you enable or disable the No Index attribute for a field in a database to which files have already been added, you must rebuild the database.

MODIFYING OR ADDING OPTIONAL DEFINITION STATEMENTS

When entering or modifying optional definition statements in a definition file, you should observe the following guidelines:

- Optional definition statements must be positioned below field statements.
- A statement must begin at the first character position in a line.

- A statement can be in upper-, lower-, or mixed-case.
- Statements must conform to the syntax and guidelines in *Adding Optional Definition Statements* on page 208.

DEFINING A PASSWORD FOR SEARCH ACCESS RESTRICTION

You can restrict search access to a database by assigning it a password; only users who know the password will be able to open the database in Personal Librarian. Moreover, you can determine whether authorized searchers of the database will be able to output their search results in either printed or electronic form.

To restrict search access for a database:

1. In PL-Admin, open the database—see *Opening a Database* on page 233.
2. Select Access Restrictions from the Options menu (CTRL+A). The Access Restrictions dialog, shown in Figure 62, will be displayed.

Figure 62 The Access Restrictions dialog

3. Click the User Password Required checkbox.

4. In the Password text box, enter a password of up to 12 alphanumeric characters.

5. If you want to prohibit authorized users from printing records or the Hitlist to either hard copy or ASCII files, click the User Allowed to Save and Print checkbox.

6. Click the OK button.

To remove access restrictions from a database:

1. In PL-Admin, open the database—see *Opening a Database* on page 233.

2. Select Access Restrictions from the Options menu (CTRL+A).

3. Click the Access Restrictions dialog's OK button.

CAUTION: Any PL-Admin user with write access to a database's library directory can create, modify, or remove access restrictions for the database. You may want to restrict distribution of PL-Admin among your users.

CREATING DATABASE-SPECIFIC ACTION LINKS

To define action links that are available only when a particular database is open in Personal Librarian—see *Activating Action Links* on page 149—you must use a text editor or word processor to create an action link file for the database.

When creating an action link file for a database, you should observe the following guidelines:

- The file must be in ASCII text format.
- The file must occupy the database's library directory—the directory that contains the database's index file (`*.pls`).
- The filename must have the extension `.act` and the same base name as the database's index file. For example, if a database's index file is named `fedcode.pls`, its action link file must be named `fedcode.act`.
- Links in the file must be formatted as embedded links—see *Creating Embedded Links* on page 174.
- A link's *description* parameter will represent the link in PL's Actions menu.
- Each link in the file must be on its own line.
- Each link in the file must begin at the first character position in a line.

The following is an example of a database-specific action link file's contents:

```
<s<GATT:headline>>Find GATT Headlines</d>
<s<Whitewater!>>Whitewater Concept Search</d>
```

If this example action file is associated with a database, the links will be accessible from the Actions menu, as shown in Figure 63, when the database is open in Personal Librarian.

Figure 63 Database-specific action links

CREATING A THESAURUS

By creating a thesaurus file and associating it with a database—see *Defining a Thesaurus* on page 215—you can enable searchers of the database to use the thesaurus operator—see *Using a Thesaurus* on page 120—and the Thesaurus Advisor—see *Viewing a Database's Thesaurus* on page 94.

A thesaurus file is an ASCII text file that contains a list of main entries along with their respective synonyms—see *Understanding Thesaurus Entry Formats* on page 121. The following is an example of a thesaurus entry.

```
MARX  Groucho Chico Harpo Zeppo Gummo Karl
```

The main entry, *MARX*, is followed by its synonyms: *Groucho, Chico, Harpo, Zeppo, Gummo,* and *Karl*.

When using a text editor or word processor to create a thesaurus file, you should observe the following guidelines:

- The file must be in ASCII text format.
- A main entry must begin at the first character position in a line.
- A main entry must be a continuous string of text; you cannot use a phrase as a main entry.
- A main entry cannot be in mixed-case; it must be in all upper- or lower-case.
- Main entries must be in alphabetical order; synonyms may be in any order.

- A main entry and its first synonym must be separated by a tab; no spaces may exist between a main entry and its first synonym.

CAUTION: Do not use DOS's `edit.com` to create a thesaurus file; it saves tabs as spaces.

- Synonyms may be in upper-, lower-, or mixed-case.
- If more than one line of synonyms is needed for a main entry, a tab must be inserted at the beginning of each additional line.
- The last synonym in the file must be followed by an end-of-line character (e.g., a carriage return).
- You can use a thesaurus operation as a synonym. For example, based on the following thesaurus:

  ```
  STOOGES  Moe Larry Curly Shemp Joes@
  JOES     Besser DeRita
  ```

  the following query:

  ```
  stooges@
  ```

  will be processed as:

  ```
  Moe Larry Curly Shemp Besser DeRita
  ```

- If you want to force a series of synonyms to be evaluated as a nested OR operation—see *Alternate Thesaurus Entry Format #1* on page 122—you must enclose it in braces, as shown in the following example:

  ```
  STOOGES  {Moe Larry Curly Shemp}
  ```

- If a series of synonyms constitutes a query that includes operators, and you want the query to be evaluated as a nested expression—see *Alternate Thesaurus Entry Format #2* on page 122—you must enclose the expression in parentheses, as shown in the following example:

```
STOOGES  Moe Larry (Curly NOT Shemp)
```

- If you want a main entry to be included in a thesaurus search, you must define the entry as its own synonym, as shown in the following example:

```
vampire  nosferatu succubus incubus vampire
```

ENABLING GENEALOGY DISPLAY FOR RECORDS

You can enable users to display record genealogies—see *Displaying a Record's Genealogy* on page 138—by adding the Genealogy button to PL's record windows. If you exercise this option

To enable genealogy display for records:

1. In PL-Admin, select General from the Prefs menu. The General Preferences dialog, shown in Figure 64, will be displayed.

Figure 64 The General Preferences dialog

2. Click the Record Windows Have Genealogy Button checkbox.

3. Click the OK button.

ENABLING PRINTOUT OF RECORD COPYRIGHT INFORMATION

If copyright information is contained in one or more records in each of your databases—see *Adding Copyright Information* on page 191—you can configure PL to print the copyright information automatically when any record is printed.

To enable automatic printing of copyright records:

1. In PL-Admin, select Copyright Record from the Prefs menu. The Copyright Record Preferences dialog, as shown in Figure 65, will be displayed.

Figure 65 The Copyright Record Preferences dialog

2. Click the Print Copyright Record With Each Printed Record checkbox.

3. Click the radio button of the copyright record location that applies to your databases.

The First Valid Record in the Database option is selected by default. If the first record in each of your databases contains copyright information for all records in its parent database, you should leave this option selected.

If your databases contain copyright information in those records that have top-level table of contents headings, click the First Record in the Top Level Domain of the Printed Record radio button.

If your databases contain copyright information in records that immediately follow records with top-level headings, click the Second Record in the Top Level Domain of the Printed Record radio button.

4. Click the OK button.

DISABLING OPENING OF TOP-LEVEL RECORDS FROM THE TABLE OF CONTENTS

When you double-click a top-level heading in a table of contents—see *Opening Records from a Table of Contents* on page 137—the record that contains the top-level heading markup is opened by default. You can configure PL so that when a top-level table of contents heading is double-clicked, the record that immediately follows the top-level heading record is opened.

To skip top-level records in a table of contents:

1. In PL-Admin, select General from the Prefs menu. The General Preferences dialog, shown in Figure 64, will be displayed.

2. Click the Skip First Record in Top Level Domains checkbox.

3. Click the OK button.

DELETING A DATABASE

To delete a database:

1. In PL-Admin, open the database that you want to delete—see *Opening a Database* on page 233.

2. Select Delete Index from the File menu (CTRL+D).

3. A dialog will prompt you to confirm that you want to delete the database. Click the Delete button.

Note: When you delete a database, you are not putting any source files at risk; the process only deletes most of the database's control files and removes the database from the ShortList. Because you may want to recycle the deleted database's definitions in future databases, the definition file is preserved.

CHAPTER 15

BUILDING AND UPDATING A DATABASE

When using PL-Admin to perform database administration tasks, you can take advantage of a feature called concurrency, which allows simultaneous searching and maintenance of a database.

OPENING A DATABASE

To open a database:

1. In PL-Admin, select Open Database from the File menu (CTRL+O).

2. When the ShortList dialog is displayed, highlight the desired database by clicking its name.

3. Click the OK button.

Tip: You can also open a database from the ShortList by double-clicking its name.

The ShortList name of the currently open database will be displayed at the bottom of the PL-Admin window.

OPENING A DATABASE THAT IS NOT IN THE SHORTLIST

A database does not have to be included in the ShortList in order to be opened in PL-Admin.

To open a database that is not in the ShortList:

1. In PL-Admin, select Open Database from the File menu (CTRL+O). PL-Admin's ShortList dialog, similar to the example in Figure 66, will be displayed; unlike PL's ShortList, it features a Browse button.

Figure 66 The PL-Admin ShortList dialog

2. Click the Browse button.

3. A standard file-selection dialog will be displayed. Use it to find and select the desired database's index file (`*.pls`).

4. Click the OK button.

CLOSING A DATABASE

To close a database:

1. Select Close Database from the File menu (Ctrl+W).

2. If any transactions are in the Pending Changes list box, a supplemental dialog will prompt you to confirm that you want to close the database. If the pending transactions can be discarded, click the Close button.

THE INDEXING PROCESS

The process whereby PL-Admin adds source files, deletes source files or records, or replaces records is called *indexing*. During indexing, PL-Admin reads the contents of source files and records its findings in a database's index file (`*.pls`); source files are not altered in any way by this process.

When you build a database, PL-Admin scans the designated source files—also known as *container files,* because each contains one or more records—and records the number of occurrences and location of all words, except stopwords, in the database's index file. Conversely, when you delete files or records from a database, PL-Admin removes references to their constituent words from the index.

When a new record is added to a database, it is assigned an *ID number,* which indicates the order in which the record was added—ID 1 indicates the first record added; ID 2, the second, etc. Within a database, each record's ID number is unique.

You can combine different types of administration activities in a single indexing run. For example, you can have statements for file addition, filed deletion, record deletion, and record replacement listed simultaneously in the Pending Changes list box; when indexing is started, the tasks will be performed in the order in which they are listed.

Preparing to Index Files

Before adding files to a database, you must have sufficient free hard disk space to accommodate the temporary files that will be created during the indexing process. The recommended amount of available space is equal to the amount of data being indexed (e.g., to add one megabyte of data, you should have one megabyte of free hard disk space). A small fraction of this free space overhead is used permanently; the rest is used for temporary file storage and will become free when indexing is finished.

Because source files should not be moved after they have been added to a database, it is recommended that you situate them in permanent locations prior to indexing.

Building a Database

Adding Files

To add one or more source files to a database:

1. In PL-Admin, open the database to which you want to add files.

2. Click the Add Files button.

3. A standard file-selection dialog will be displayed. Use it to find and select the desired source files. By default, the dialog lists files with the `.src` filename extension. In a single directory, you can select a range of files in the manner that is standard to your operating system.

4. Click the OK button. For each selected file, a statement in the Pending Changes list box will specify the administration activity that will be performed.

Note: Files will be added to the database in the order in which they appear in the Pending Changes list box.

5. Repeat Steps 2–4 to select additional files from other directories as needed.
6. If you reconsider adding a file to the database, select the corresponding statement in the Pending Changes list box and click the Remove Transaction button.

Tip: To remove all pending transactions, click the Clear button.

7. When you are satisfied with the pending transactions, click the Go button.
8. When indexing is finished, a dialog will summarize the changes that have been made to the database. Click the OK button.

The list box that occupies the upper half of the PL-Admin window will list the source files that have been added to the database.

Note: If you add a source file more than once to the same database, it will be listed only once in the source files list box.

Appending New Files to an Indexed File

When adding ASCII source files to a database, you can elect to append their contents to a previously indexed ASCII file. If you do this, the original copies of the files may be deleted, as they will not be required for searching.

Note: Source files in word processor formats cannot be appended to a previously indexed source file.

To append source files to an indexed file:

1. In PL-Admin, open the database to which you want to add files.

2. In the database files list box, select the file to which the contents of new source files will be appended.

Note: When a database file is highlighted, its path, filename, and size are displayed at the bottom of the PL-Admin window.

3. Click the Add Files Into button.

4. A standard file-selection dialog will be displayed. Use it to find and select the source files that will be added. By default, the dialog lists files with the `.src` filename extension. In a single directory, you can select a range of files in the manner that is standard to your operating system.

5. Click the OK button. For each selected file, a statement in the Pending Changes list box will specify the administration activity that will be performed.

Note: Files will be added to the database—and appended to the previously indexed source file—in the order in which they appear in the Pending Changes list box.

6. Repeat Steps 3–5 to select additional files from other directories as needed.

7. If you reconsider adding a file to the database, select the corresponding statement in the Pending Changes list box and click the Remove Transaction button.

Tip: To remove all pending transactions, click the Clear button.

8. When you are satisfied with the pending transactions, click the Go button.

9. When indexing is finished, a dialog will summarize the changes that have been made to the database. Click the OK button.

BUILDING A TABLE OF CONTENTS

After building a database from source files that contain table of contents markup—see *Adding Table of Contents Markup* on page 187—you can build a table of contents file (`*.hir`) for the database by selecting Add Table of Contents from the Options menu (Ctrl+T) while the database is open in PL-Admin.

UPDATING A DATABASE

REMOVING SOURCE FILES FROM A DATABASE

Removing a source file from a database deletes references to the file and its contents from the database's index; the source file is not altered in any way.

CAUTION: When you remove a source file from a database, you delete the records that the file comprises; any bookmarks or annotations that are associated with the deleted records will be rendered invalid.

To remove source files from a database:

1. In PL-Admin, open the database from which you want to remove files.

2. In the database files list box, select the source file that you want to remove.

Note: When a database file is highlighted, its path, filename, and size are displayed at the bottom of the PL-Admin window.

3. Click the Remove Container button. A statement in the Pending Changes list box will specify the administration activity that will be performed for the selected source file.

4. As needed, repeat Steps 2–3 to select additional source files for removal.

5. If you reconsider removing a source file from the database, select the corresponding statement in the Pending Changes list box and click the Remove Transaction button.

Tip: To remove all pending transactions, click the Clear button.

6. When you are satisfied with the pending transactions, click the Go button.

7. When indexing is finished, a dialog will summarize the changes that have been made to the database. Click the OK button.

DELETING RECORDS

CAUTION: When you delete a record from a database, any bookmarks or annotations that are associated with the record will be rendered invalid.

To delete records from a database:

1. In PL-Admin, open the database from which you want to delete records.

2. Click the Delete Records button. The Delete Records dialog, shown in Figure 67, will be displayed.

Figure 67 The Delete Records dialog

3. In the corresponding text boxes, enter the first and last ID numbers in the range of consecutive records that you want to delete; if you want to delete a single record, enter its ID number in both boxes.

4. Click the OK button. A statement in the Pending Changes list box will specify the administration activity that will be performed for the selected record(s).

5. As needed, repeat Steps 2–4 to select additional records for deletion.

6. If you reconsider deleting any records or record ranges, select the corresponding statement in the Pending Changes list box and click the Remove Transaction button.

Tip: To remove all pending transactions, click the Clear button.

7. When you are satisfied with the pending transactions, click the Go button.

8. When indexing is finished, a dialog will summarize the changes that have been made to the database. Click the OK button.

Note: If a deleted record or record range comprises the last record in the database, the ID number(s) of the deleted record(s) will be used for records that are added to the database in the future. Otherwise, ID numbers of deleted records are not recycled.

REPLACING RECORDS

You can update a record by replacing it with the contents of a specified text file.

Note: An updated record will retain its original ID number.

To replace records:

1. In PL-Admin, open the database in which you want to update records.

2. Click the Replace Record button. The Replace Record dialog, shown in Figure 68, will be displayed.

Figure 68 The Replace Record dialog

3. In the Replace Record ID text box, enter the ID number of the record that you want to replace. The first few lines of the specified record will be displayed in the dialog’s window; this excerpt is useful for confirming that you have specified the correct ID number.

4. In the With File text box, enter the path and filename of the text file that will replace the specified record; if you don’t know the full path or name of the file, you can search for it by clicking the Browse button.

5. If you want to append the contents of the specified text file to the end of the source file that contains the specified record, click the Make Copy in Existing Source File checkbox. If you enable this option, you can discard the original copy of the text file when indexing is finished; it will not be required for searching.

6. Click the OK button. A statement in the Pending Changes list box will specify the administration activity that will be performed for the specified record.

7. As needed, repeat Steps 2–6 to specify other records for updating.

8. If you reconsider updating any records, select the corresponding statement in the Pending Changes list box and click the Remove Transaction button.

Tip: To remove all pending transactions, click the Clear button.

9. When you are satisfied with the pending transactions, click the Go button.

10. When indexing is finished, a dialog will summarize the changes that have been made to the database. Click the OK button.

REORGANIZING A DATABASE'S INDEX

Because PL databases are self-optimizing, you should consider reorganizing a database's index only under the following conditions:

The database will never be updated. If the contents of a database will not change in the future, you can maximize available hard disk space by performing archival reorganization, which compresses the database's index to the smallest possible size.

The database has undergone extensive updating. If free hard disk space is at a premium, you can reclaim the small amount of unused space that exists in a frequently updated database's index by reorganizing it.

You want to partition the database's index. If, after adding files to a database, you decide to partition its index, you can implement partitions during the reorganization process.

To reorganize a database's index:

1. In PL-Admin, open the database that you want to reorganize.
2. If you want the reorganized index to be in archival format, select Compact/Archival Reorg from the Options menu; if not, select Compact/Reorg.

CAUTION: You should select archival formatting only if the database will never be updated after reorganization; updating an archived database is extremely wasteful of hard drive space.

3. A standard file-saving dialog will be displayed. Use it to situate and name the reorganized index file; do not specify a filename extension.

Tip: It is recommended that you save the reorganized index to the database's library directory, which contains the original index; if you do not save it to that directory, you will have to move it there after reorganization is finished.

4. Click the OK button.

5. A dialog will prompt you to indicate if you want to use the original database's definitions for the reorganized version. Click the Yes button.

6. The Define Database dialog will be displayed. If you want to partition the reorganized index, add the appropriate definition statement—see *Building Index Partitions on Faster Media* on page 217. If the original index was partitioned, you must replace the original partition statement with one that specifies a different partition filename.

7. Click the OK button.

8. If you did not save the reorganized index to the directory occupied by the original index, move it there, along with its definition file (`*.def`) and any partitions.

9. Rename the reorganized index and its definition file to have the filenames of the originals.

MOVING AND/OR RENAMING SOURCE FILES AFTER INDEXING

To move and/or rename an indexed source file:

1. Using methods standard to your operating system, move or rename the file.

2. In PL-Admin, open the database that contains the altered source file.

3. Double-click the file's corresponding line in the database files list box. A dialog similar to the example in Figure 69 will be displayed.

Figure 69 Enter the source file's new path or filename.

4. As needed, enter the file's new path or name. If you specify a relative path, it must be relative to the database's library directory.

CAUTION: If you intend for the database to be portable, do not specify an absolute path to a source file.

5. Click the OK button.

6. A dialog will inform you that PL-Admin has not altered the file—if you did not modify the file's name/location at the beginning of this procedure, you must do so now.

7. Click the OK button. The database files list box will reflect the file's new name/location.

MOVING A DATABASE

You can move database files using methods that are standard to your operating system. In doing so, you should observe the following guidelines:

- A database is most portable when its control files—index, definition file, table of contents file, etc.—and source files occupy the same directory.
- If you intend for a database to be portable, you should, prior to indexing, position all source files in a central location or in locations that you can manage easily. When you move a database's index file, source files must maintain their position relative to it.
- If you specify absolute paths for source files when using `pladd.exe` or `plupdate.exe` to build a database, you must rebuild the database after moving it. See *Batch Processing of Administration Tasks* on page 310.
- If you move a database, you must update the ShortList by editing the appropriate line in `dblist.cpl`.

EDITING DBLIST.CPL

The ShortList used in Personal Librarian and PL-Admin is extracted from an ASCII text file called `dblist.cpl`, which is located in PL's `\shared` subdirectory. By using a text editor or word processor to modify the contents of this file, you can accomplish the following:

- Add an existing database to the ShortList.

- Remove a database from the ShortList.
- Change a database's ShortList name.
- Specify a new location for a database's index, in the event that you move the database.
- Specify the location of a database's definition file, in the event that it is not in the database's library directory.

Each line in `dblist.cpl` conforms to the following format:

```
listname    idxpath     defpath
```

where:

- *listname* is the alias that represents a database in the ShortList.
- *idxpath* specifies the path and base name of a database's index file (`*.pls`).
- *defpath* specifies the path and filename of a database's definition file (`*.def`); this parameter is optional.

Example:

```
Cryptozoology C:\DB\CRYPTZOO\CRYPTZOO
Xenobiology C:\DB\EBE\EBE C:\DB\BEM\EBE.DEF
“Fortean Phenomena” C:\DB\FORTEAN\FORTEAN
```

When editing `dblist.cpl`, you should observe the following guidelines:

- Text can be in upper-, lower-, or mixed-case.
- Each line can be up to 99 characters in length.
- Comment lines must be preceded by the pound sign (#); comments and blank lines are ignored.

- Parameters—*listname*, *idxpath*, *defpath*—must be separated by a single space or a single tab.

- The *listname* parameter must begin at the first character position in a line.

- If you change a database's ShortList name, you must do the following before opening the database in Personal Librarian: open the database in PL-Admin, select Access Restrictions from the Options menu, and click the Access Restrictions dialog's OK button.

- If you change a database's ShortList name, all annotations and bookmarks that are associated with the database will be lost.

- If *listname* includes spaces, it must be enclosed in double quotation marks.

- *listname* can be up to 31 characters in length. If the name is enclosed in double quotation marks, the marks are not included in the 31-character limit.

- *idxpath* cannot include a filename extension.

- If *idxpath* contains a relative path, it must be relative to PL's home directory.

- *defpath* must include an absolute path.

- *defpath* must include the `.def` filename extension.

- If you use the *defpath* parameter to specify a definition file location other than the database's library directory, any relative paths in definition statements must be relative to the definition file.

- Even if the *defpath* parameter is used for a database, a definition file with the same base name as *idxpath* must exist in the database's library directory.

VERIFYING INDEX INTEGRITY

If, when searching a database in Personal Librarian, you notice erratic performance—inconsistent results for a query, Dictionary Advisor terms that are listed in incorrect order, etc.—it may indicate that the database's index has been corrupted. In this event, you can use PL-Admin to verify the integrity of the database's index.

To verify index integrity:

1. In PL-Admin, open the database that you want to verify.

2. Select Verify from the Options menu. The Verify Database dialog, shown in Figure 70, will be displayed.

Figure 70 The Verify Database dialog

3. You can select from three levels of verification, which are summarized in Table 6; Basic Verification is selected by default. Click the radio button of desired verification level.

Table 6 Index Verification Options

Verification Level	Description
Basic	Verifies that all words in the database's dictionary are in alphabetical order.
Intermediate	Performs Basic verification and verifies that postings in the dictionary match those in the index and that index postings are in correct order—postings for record ID 1 precede those for record ID 2, etc.
Advanced	Performs Basic and Intermediate verification and verifies that each word occurs the correct number of times in each record.

Note: Because Advanced verification attempts to open each record in a database, it can take considerably longer than the other verification levels. The Intermediate option provides nearly the same level of verification in a fraction of the time.

4. By default, the verification process is terminated after 30 errors are detected. To disable this feature, click the corresponding checkbox.

5. Click the OK button.

6. A dialog that summarizes the results of the verification process will be displayed. Click the OK button.

7. If errors were found, and you are not sure whether or not you should rebuild the database, contact the PLS customer support department.

RECOVERING FROM INDEXING FAILURE

PL-Admin features *transaction integrity*, which means that the previous version of a database is not affected in the event of indexing failure. If indexing is interrupted, only the effects of the pending administration transactions will be lost.

If a failure occurs during the indexing process, PL-Admin may create a *lock file* (`*.lck`), which prevents updating of the database with which it is associated. If a lock file is generated for a database, you must delete it before trying to update the database.

If a failure occurs during the creation of a database, you should delete the new database's index—see *Deleting a Database* on page 232.

To gain insight to the possible cause of indexing failure, you can use a text editor or word processor to review the contents of `aderrors.txt`, which is located in PL's `\shared` subdirectory. This file keeps a record of failed PL-Admin operations; in the event that you need to contact the PLS customer support department, the contents of the file will be useful in diagnosing any problems.

PART 4

APPENDICES

4

APPENDIX A

QUICK REFERENCE

This appendix summarizes the operators used in Personal Librarian's query language, and lists the accelerator keystrokes that are available in PL and PL-Admin.

QUERY LANGUAGE OPERATORS

Operator	Description	Examples
space	**Default operator.** Is assumed between query terms separated by a space. See page 34.	`heaven hell`
!	**Concept operator.** Generates a list of words related to the query term to which it is appended; searches for the query term and the related terms See page 104.	`Whitewater! investigation`
OR	**Boolean OR logic operator.** Searches for records that contain either of the query terms it separates. See page 105.	`succubus OR incubus`

Operator	Description	Examples
AND	**Boolean AND logic operator.** Searches for records that contain both of the query terms it separates. See page 106.	`Burke AND Hare`

NOT	**Boolean NOT logic operator.** In binary form, searches for records that contain the query term that precedes it but do not contain the term that follows it.
	In unary form, searches for all records that do not contain the term that follows it.
	See page 106.
		`media NOT television`
		`NOT Gordian`

?	**Single character wildcard.** Matches a single variable character.
	See page 108.
		`medic???`
		`1?4`
		`R??2`

$	**Optional character wildcard.** Matches one or zero variable characters.
	See page 109.
		`colo$r`
		`V$TOL`
		`electron$$$`

*	**String wildcard.** Matches a string of zero or more variable characters.
	See page 110.
		`medic*`
		`m*n`
		`*ane`
		`1*5`

W/	**Proximity operator.** Searches for word pairs in which the second term occurs within a specified number of words after the first. See page 111.	`amphibian W/5 DNA`

Operator	Description	Examples

ADJ
-
'
,
.
	Adjacency operator. Searches for word pairs in which the second term occurs immediately after the first.
	May be replaced by a hyphen, apostrophe, comma, or period with no leading or trailing spaces.
	See page 112.
		`great ADJ white`
		`337-81-4417`
		`O'Hara`
		`2,001`
		`Carcharodon.carcharias`

NEAR/	**Near operator.** Searches for word pairs in which the second term occurs within a specified number of words before or after the first.
	With no specified range, serves as bidirectional adjacency operator
	See page 113.
		`Whitewater NEAR/5 scandal`
		`tax NEAR increase`

SAME/
SAME
	Same paragraph operator. Searches for word pairs that occur in a specified range of paragraphs.
	With no specified range, searches for pairs that occur in the same paragraph.
	See page 114.
		`Africa SAME/3 Congo`
		`Australia SAME outback`

NOTSAME	**Notsame paragraph operator.** Searches for records that contain paragraphs in which the term that precedes the operator occurs and the term that follows it does not. See page 115.	`plague NOTSAME buboes`

Operator	Description	Examples
"	**Exact match operator.** Searches for the exact term to which it is appended; disables stemming for the term. See page 116.	run"
' '	**Exact phrase operator.** Searches for exact matches of the phrase that it borders. See page 117.	'Project Blue Book' 'League of Women Voters'
+	**Stemming operator.** Forces stemming of the query term to which it is appended. See page 118.	run+
~	**Fuzzy search operator.** Searches for words with spelling similar to the query term with which it is used. See page 118.	Kir~gizia cyclo~hexene
@	**Thesaurus operator.** Searches for synonyms of the query term to which it is appended. See page 120.	CIA@ FBI@ NATO@ congress@
D_	**Document operator.** Searches for records that are similar in content to the record with a specified ID number. See page 123.	D_20
Q_	**Query operator.** Recycles the query from a specified search in the current session. See page 124.	Q_5 Q_7 Trilateral Commission

Operator	Description	Examples

:	**Word-level field restriction operator.** Restricts search for a specified query term to one or more specified fields.
	See page 125.
		`Simpson:defendant`
		`Renfield:name,supervisor`
		`Alhazred:author AND Neconomicon:title`

/f:	**Query-level field restriction operator.** Restricts entire query to one or more specified fields.
	See page 126.
		`Renfield Harker/f:name`
		`Miskatonic OR Arkham /f: text,synopsis`

=	**Equivalence operator.** Searches for a specified value in a specified field.
	See page 129.
		`author=Seuss`
		`1897=year`

!=	**Non-equivalence operator.** In binary form, searches for records in which a specified value does not occur in a specified field.
	In unary form, searches for all records that do not contain a specified value.
	See page 130.
		`author!=Seuss`
		`1897!=year`
		`!=C57D`

>
<
>=
<=
	Range operator. Searches for records in which the field that precedes the operator contains a value that falls within the range that follows the operator.
	See page 131.
		`population>900`
		`zipcode<=20895`
		`Karloff<name<Pratt`

( )	**Scope of operation delimiters.** Impose algebraic grouping logic on complex queries. See page 133.	`United ADJ (States Nations)`

KEYBOARD SHORTCUTS

You can use the accelerator keystrokes in Table 7 and Table 8 to replace or complement mouse actions in Personal Librarian and PL-Admin.

Table 7 Accelerator Keystrokes in Personal Librarian

Keystroke	Action
CTRL+O	Open database
CTRL+W	Close database
ALT+F4	Exit Personal Librarian
CTRL+A	Select all text in current window
CTRL+X	Cut selection
CTRL+C	Copy selection
CTRL+V	Paste clipboard contents
CTRL+=	Paste selection to query
CTRL+HOME	Go to beginning of current record
CTRL+END	Go to end of current record
F4	Go to next hit in current record
F9	Go to next ranked record
SHIFT+F9	Go to previous ranked record
F7	Go to next record ID
SHIFT+F7	Go to previous record ID
CTRL+D	Go to specified record ID
CTRL+R	Go to record at specified rank
CTRL+F	Find text string in current record
F3	Go to next string occurrence

Table 7 Accelerator Keystrokes in Personal Librarian

Keystroke	Action
CTRL+B	Reverse hypertext link
F5	Go to table of contents
CTRL+P	Print current record/image/search details
CTRL+F4	Close current window
CTRL+F7	Select normal text style for current window
CTRL+F8	Select font family, style, an size for current record
CTRL+"	Zoom in to current image
CTRL+-	Zoom out from current image
CTRL+>	Rotate current image 90 degrees clockwise
CTRL+<	Rotate current image 90 degrees counter-clockwise
F1	Display context-sensitive help
SHIFT+F1	Display help contents

Table 8 Accelerator Keystrokes in PL-Admin

Keystroke	Action
CTRL+N	Create new database
CTRL+O	Open database
CTRL+W	Close database
ALT+F4	Exit PL-Admin
CTRL+T	Build table of contents
CTRL+A	Set access restrictions
CTRL+R	Verify index integrity
CTRL+D	Delete database
F1	Display context-sensitive help
SHIFT+F1	Display help contents

APPENDIX B

CUSTOMIZING PERSONAL LIBRARIAN

This appendix documents techniques with which you can modify Personal Librarian's system-wide performance settings.

CREATING GLOBAL ACTION LINKS

To define action links that are always available in Personal Librarian—see *Activating Action Links* on page 149—you must use a text editor or word processor to create a global action link file.

When creating a global action link file, you should observe the following guidelines:

- The file must be in ASCII text format.
- The file must occupy PL's `\personal` subdirectory.
- The file must be named `actlinks.pl`.
- Links in the file must be formatted as embedded links—see *Creating Embedded Links* on page 174.
- A link's *description* parameter will represent the link in PL's Actions menu.

- Each link in the file must be on its own line.

- Each link in the file must begin at the first character position in a line.

The following is an example of a global action link file’s contents:

```
<c<notepad>>Open Text Editor</d>
<i<1,grafix\usmap.tif,==>>Display U.S. Map</d>
<i<1,grafix\worldmap.tif,==>>Display World Map</d>
```

If these example links are defined in a global action link file, the Actions menu in Personal Librarian will appear as shown in Figure 71.

Figure 71 Global action links

DEFINING ALTERNATE VIEWERS FOR IMAGE LINKS

By using a text editor or word processor to modify `plwdefs.ini`, which is located in PL’s `\shared` subdirectory, you can define one or more third-party image viewers for use with image links—see *Displaying an Image* on page 179.

To define a viewer that will be launched by corresponding image links, add the following lines to `plwdefs.ini`:

```
[ImageTypes]
type=command
```

where:

- *type* is an alias that associates a viewer's command line with a link that contains the alias in its syntax.
- *command* is the command line that PL will pass to your operating system when an image link of the corresponding *type* is activated. An image link's *filename* parameter will be appended to the command line.

Example:

```
[ImageTypes]
bmp=c:\windows\pbrush
pcx=c:\windows\pbrush
gif=c:\winapps\lview31
jpg=c:\winapps\lview31
```

With these image types defined in `plwdefs.ini`, image links can be composed with syntax similar to the following examples:

```
<I<bmp,grafix\logo.bmp>>Click here to see logo.</D>

<I<pcx,ocr\clip51.pcx>>Click to see facsimile.</D>

<I<gif,maps\area51.gif>>Groom Lake aerial view.</D>

<I<jpg,oolala\paulina8.jpg>>Click this!</D>
```

When defining alternate viewers in `plwdefs.ini`, you should observe the following guidelines:

- The file must remain in ASCII format.
- You can insert blank lines; they will be ignored.
- The `[ImageTypes]` section heading must begin at the first character position in a line.
- Image type definition statements must begin at the first character position in a line.

- *type* can be up to 4 characters in length.
- The application specified in *command* must accept arguments in its command line.
- If you specify a relative path in *command*, it must be relative to the index file of the database that contains the image link.

SPECIFYING DATABASES TO OPEN AT STARTUP

By using a text editor or word processor to modify `plwdefs.ini`, which is located in PL's `\shared` subdirectory, you can have one or more databases opened automatically when you run Personal Librarian.

To define databases that will be opened at startup, add the following lines to `plwdefs.ini`:

```
[Startup]
Databases=listname+listname+listname...
```

where *listname* is the ShortList name of a database that will be opened automatically.

Example:

```
[Startup]
Databases=Fortean Phenomena+Cryptozoology
```

When adding this definition to `plwdefs.ini`, you should observe the following guidelines:

- The file must remain in ASCII format.
- You can insert blank lines; they will be ignored.

- The [Startup] section heading and the definition statement must begin at the first character position in a line.
- If multiple databases are defined, their ShortList names must be separated by a plus sign (+).

ADDING SUPPLEMENTAL ONLINE COPYRIGHT INFORMATION

By using a text editor or word processor to modify `cpyrt.inf`, which is located in PL's `\shared` subdirectory, you can add up to five lines of additional text to the dialogs that are displayed when Personal Librarian and PL-Admin are started or when About is selected from the Help menu in either application. To get an idea of how supplemental copyright information is displayed, see the example in Figure 72.

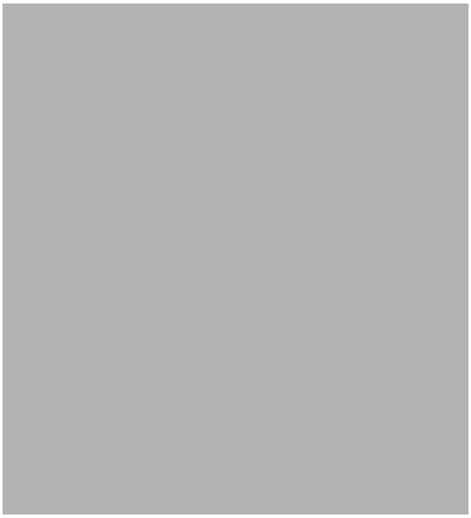

Figure 72 Supplemental copyright information in the PL startup dialog

To add supplemental online copyright information:

1. With a text editor or word processor, open `cpyrt.inf`.

2. Delete the file's current contents.

3. Enter up to five lines of supplemental copyright information.

4. Save the edited version of `cpyrt.inf`.

5. Exit Windows or shell out to DOS.

6. From PL's `\cpyrt` subdirectory, enter the following command:

   ```
   hide c:\plwin\shared\cpyrt.inf
   ```

Note: If you did not accept the default directory names during installation, you will have to enter a different path to `cpyrt.inf`.

When editing `cpyrt.inf`, you should observe the following guidelines:

- The file must remain in ASCII format.

- Blank lines can be included.

- Text may exist below the fifth line, but it will not be displayed.

- Because copyright information is displayed in a proportional font, the maximum number of displayable characters per line depends on which characters are used. For example, a line of more than 24 upper-case W's will be truncated, while a line of 80 lower-case i's will not. You can use the example in Figure 72 to estimate how your copyright information should be formatted.

ADDING ONLINE RESELLER INFORMATION

By using a text editor or word processor to modify `startinf.cfg`, which is located in PL's `\shared` subdirectory, you can define reseller-specific information that will be displayed in its own dialog, similar to Figure 73, when Personal Librarian is started.

Figure 73 A reseller information dialog

To add online reseller information:

1. With a text editor or word processor, open `startinf.cfg`.

2. Delete the file's current contents.

3. Enter the desired reseller information.

4. Save the edited version of `startinf.cfg`.

5. Exit Windows or shell out to DOS.

6. From PL's `\cpyrt` subdirectory, enter the following command:

   ```
   hide c:\plwin\shared\startinf.cfg
   ```

Note: If you did not accept the default directory names during installation, you will have to enter a different path to `startinf.cfg`.

The contents of `startinf.cfg` must be in the following format:

```
DispText=x
DlgTitle
VARinfo
```

where:

- *x* determines whether no reseller dialog is displayed, the dialog is displayed automatically, or the dialog is displayed at the user's request. Valid values for the *x* parameter are summarized in Table 9.

Table 9 Valid Values for *x* Parameter in `startinf.cfg`

Value	Reseller Dialog Behavior
n	Not displayed
y	Displayed automatically after Personal Librarian startup dialog
o	Displayed when user clicks More button that is added to Personal Librarian startup dialog, as shown in Figure 74

- *DlgTitle* is text that will appear in the reseller dialog's title bar.

- *VARinfo* is one or more lines of text that will be displayed in the reseller dialog.

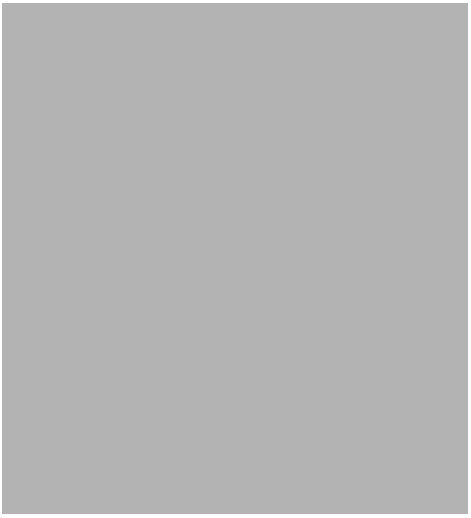

Figure 74 A user can view reseller information by clicking the More button.

Example:

```
DispText=y
Personal Library Software
The software in this document is furnished under a
license agreement and may be used only in
accordance with the terms of that agreement.

Copyright © 1995 Personal Library Software, Inc.
All Rights Reserved.

Personal Librarian is a trademark of Personal
Library Software, Inc.

Personal Library Software
2400 Research Boulevard
Suite 350
Rockville, MD 20850
USA
Tel. 301-990-1155
FAX. 301-963-9738
e-mail. support@pls.com
```

When editing `startinf.cfg`, you should observe the following guidelines:

- The file must remain in ASCII format.
- The file can contain a maximum of 64,000 characters.
- Blank lines can be included.

EDITING DEFAULTS.CPL

The following sections discuss performance parameters that you can modify by changing settings in the ASCII text file `defaults.cpl`, which is located in PL's `\shared` subdirectory.

This file contains numbered strings of text, each of which dictates a system-wide configuration default for Personal Librarian. While certain settings can be superseded at the database level, you can modify a few system-wide defaults by changing the corresponding lines in `defaults.cpl`.

When editing `defaults.cpl`, you must observe the following guidelines:

- The file must remain in ASCII format.
- You must not change any `defaults.cpl` strings that are not discussed in this document.
- You may change only a string's setting; the string number and the single space that follows it must remain intact.
- If the valid values for a string are *yes* and *no*, the string's setting must be expressed as capital Y or N.

CUSTOMIZING SEARCH PERFORMANCE

SETTING THE DEFAULT FOR SEARCH STEMMING

`Defaults.cpl` string 287 determines whether search stemming—see *Understanding Search Stemming* on page 67—is on or off; by default, it is turned on. If you want to disable search stemming, change string 287's setting to *N*.

ADJUSTING FUZZY MATCHING SELECTIVITY

The selectivity of Personal Librarian's fuzzy matching algorithm—see *Fuzzy Matching: Identifying Words with Similar Spelling* on page 92 and *Searching for Words with Similar Spelling* on page 118—is determined by `defaults.cpl` string 138, which has a default value of 60. You can increase or decrease fuzzy matching selectivity by, respectively, raising or lowering the string's setting. The valid values for string 138 are 1–100.

Note: Slight adjustments to this string's setting can result in wide variations in fuzzy matching selectivity.

ACTIVATING AUTOMATIC THESAURUS OPERATION FOR ALL QUERY TERMS

By modifying `defaults.cpl` string 227, you can arrange for the thesaurus operator—see *Using a Thesaurus* on page 120—to be implicitly appended to all words in a query. By default, this feature is disabled; to enable it, change string 227's setting to *Y*.

When this feature is enabled, the following query:

```
lions AND tigers AND bears
```

is processed as:

```
lions@ AND tigers@ AND bears@
```

SETTING THE NUMBER OF LINES PER RECORD SCANNED FOR FIELDED LINKS

With `defaults.cpl` string 257, you can specify the number of lines from the beginning of each record that PL will check for fielded links; a fielded link that occupies a line beyond this limit cannot be accessed. By default, the first 30 lines in a record are scanned.

CUSTOMIZING HITLIST DISPLAY

SETTING THE NUMBER OF LINES PER RECORD SCANNED FOR HITLIST INFORMATION

With `defaults.cpl` string 238, you can specify the number of lines from the beginning of each record that PL will scan for fields that contain Hitlist information; by default, the first 15 lines of a record are scanned. The contents of a field that is defined for Hitlist display—see *Assigning Field Attributes* on page 207—will not appear in the Hitlist unless the text is within the range specified by string 238.

CAUTION: Setting this string to a high number may retard Hitlist display, especially for data that is stored on CD-ROM.

SETTING THE NUMBER OF LINES PER RECORD DISPLAYED IN THE HITLIST

With `defaults.cpl` string 239, you can specify the maximum number of lines that will be displayed for any record in the Hitlist; by default, a maximum of 8 lines per record are displayed.

Note: This string's setting defines a maximum limit; fewer lines may be displayed for individual records, depending on a database's field definitions, the presence or absence of fields defined for Hitlist display, and the length of those fields.

SETTING THE NUMBER OF LINES PER FIELD DISPLAYED IN THE HITLIST

With `defaults.cpl` string 240, you can specify the maximum number of lines that will be displayed for any field in the Hitlist; by default, a maximum of 2 lines per field are displayed.

Note: This string's setting defines a maximum limit; fewer lines may be displayed for individual fields, depending on the length of the fields.

CUSTOMIZING RECORD WINDOW DISPLAY

MODIFYING DISPLAY OF FIELD LABELS IN RECORD WINDOWS

`Defaults.cpl` string 185 determines whether field labels are displayed or hidden in record windows; by default, field labels are displayed. If you want field labels to be hidden by default, change string 185's setting to *Y*.

Note: This setting also governs the display of field labels in printed records.

Tip: If you want to override this setting for a particular database, see *Hiding Field Labels in Record Windows* on page 214.

APPENDIX C

ADVANCED TOPICS

If you are an advanced user, you can extend PL's functionality by employing the techniques that are documented in this appendix.

USING PERSONAL LIBRARIAN ON A NETWORK

When using Personal Librarian in a network environment, you should observe the following guidelines:

- PL applications should be run locally, rather than from a network server.
- The directory that serves as a repository for PL applications' temporary files—see *The Installation Process* on page 15—should be located on a local hard drive, rather than on a network server.
- To be most accessible to workstations in a network, databases and their constituent source files should be placed on network file servers.
- Network bandwidth should be 10 megabits/second or higher; retrieval performance improves as bandwidth increases.

If multiple workstations define the same `\shared` subdirectory during installation—see *The Installation Process* on page 15—they will share the following:

- a common ShortList.
- system-wide search performance and display settings, as defined in `defaults.cpl`—see *Customizing Search Performance* on page 275, *Customizing Hitlist Display* on page 277, and *Customizing Record Window Display* on page 279.
- global stopword definitions—see *Stopword Customization* on page 161.
- global tokenization rules—see *Making Special Characters Searchable* on page 297.
- third-party image viewer definitions—see *Defining Alternate Viewers for Image Links* on page 266.
- databases that are opened automatically when Personal Librarian is started—see *Specifying Databases to Open at Startup* on page 268.
- supplemental online copyright information—see *Adding Supplemental Online Copyright Information* on page 269.
- VAR-specific information that is displayed when PL applications are started—see *Adding Online Reseller Information* on page 271.

The following are unique for each workstation:

- annotations—see *Annotating Records* on page 75.
- bookmarks—see *Placing Bookmarks in Records* on page 78.
- global action links—see *Activating Action Links* on page 149.

USING PERSONAL LIBRARIAN ON CD-ROM

When using Personal Librarian for a CD-ROM application, you should observe the following guidelines:

- The Personal Librarian application should be run from a local hard drive, rather than from a CD-ROM.

- Source files should not reside in deeply nested directories on a CD-ROM.

- All ASCII data should reside in a single source file; if this data is indexed in multiple batches, it should be appended, rather than indexed in place—see *Appending New Files to an Indexed File* on page 238.

- The maximum source file size should be 200–385MB.

- If a database's index is on CD-ROM, its dictionary and storage keys should be partitioned to a local hard drive—see *Building Index Partitions on Faster Media* on page 217.

- If a CD-ROM-based database has index partitions on a local hard drive, the partition statements in the database's definition file must specify the hard drive location. If this requires moving the definition file to a writeable medium, the database's line in `dblist.cpl` must specify the definition file's new location—see *Modifying a Definition File after Database Creation* on page 222.

- Because absolute paths may not be consistently interpreted on different platforms, image links and command links in a CD-ROM-based database should use relative paths.

USING DLL LINKS AND CALLBACKS

You can use a *DLL link* to call functions in a DLL. These functions can execute callback functions to retrieve and manipulate data from the Personal Librarian application. For example, a DLL link could invoke a DLL function that serves as a report generator; the function might print the contents of certain fields for each record retrieved by the most recent PL search.

You must provide the DLL that contains the function called by a DLL link. You will find a sample DLL in PL's `\dlllinks\example` subdirectory.

Embedded DLL Link Syntax:

`<X<`*`DLLname`*`,`*`function`*`,`*`arguments`*`>>`*`description`*`</D>`

Fielded DLL Link Syntax:

`<X<`*`DLLname`*`,`*`function`*`,`*`arguments`*`>></D>`

where:

- *DLLname* is the name of your DLL.
- *function* is the name of a function in *DLLname*.
- *arguments* is text that will be passed as a single LPSTR (string) argument to *function*; the DLL function is responsible for interpreting the string. This parameter can include spaces and separators other than the "greater than" angle bracket (>). The *arguments* parameter must exist in a DLL link; if *function* does not require arguments, a space can be used.
- *description* is the text that will appear in the embedded link's hot spot.

When creating DLL links, you should observe the guidelines in *Creating Embedded Links* on page 174 and *Creating Fielded Links* on page 175.

Embedded DLL Link Example:

```
<X<reprtdll,goReport,null>>Write Report</D>
```

Fielded DLL Link Example:

```
-Write_Report-
<X<reprtdll,goReport,null>></D>
```

Note: While PL databases are interoperable across multiple platforms, DLL links will function only in a Windows environment. In a non-Windows environment, an embedded DLL link's description will be displayed without highlighting, which indicates that the link is not active.

PL CALLBACKS

Initialization

Your DLL must implement an initialization function, through which PL communicates the addresses of available callback functions. Your initialization function should store these pointers so that the linked function can make use of the callbacks as needed. The prototype of the initialization function is as follows:

```
void FAR _export PASCAL
initPLCallbacks(PLDLLCallbacks FAR *p, size_t s);
```

This function must be a C function, not a C++ function. If you use a C++ compiler, you should put the function declaration within extern "C" { } brackets to tell the compiler to generate a C signature.

The simplest implementation of this initialization function would be as follows:

```
static PLDLLCallbacks FAR *pCb = NULL;

void FAR _export PASCAL
initPLCallbacks(PLDLLCallbacks FAR *p, size_t s)
{
      pCb=p;
}
```

Your functions should now be able to use the PL callback functions by calling:

```
      pCb->function(...)
```

An example is provided in PL's `\dlllinks\example` subdirectory.

Callback Functions

A function in a linked DLL can use one or more of the following callbacks to retrieve information about the current state of the Personal Librarian application:

```
BOOL _export FAR PASCAL plGetDBName (LPSTR dbname);

BOOL _export FAR PASCAL plGetContainer
                              (LPSTR container);

long _export FAR PASCAL plGetDocID        (void);

BOOL _export FAR PASCAL plGetLine
                        (char *buffer, int buflen);

BOOL _export FAR PASCAL plSeekFieldName
                              (char *fieldname);

BOOL _export FAR PASCAL plSeekFieldID     (int id);

BOOL _export FAR PASCAL plNextRecord      (void);
BOOL _export FAR PASCAL plPrevRecord      (void);

BOOL _export FAR PASCAL plSeekRecord
                                    (long id);

BOOL _export FAR PASCAL plNextHit         (void);

BOOL _export FAR PASCAL plPrevHit         (void);

BOOL _export FAR PASCAL plSeekHit   (long offset);

BOOL _export FAR PASCAL plSortHits
                              (LPSTR fieldname);

BOOL _export FAR PASCAL plGetQuery  (LPSTR query);
```

Table 10 Operational Descriptions of PL Callbacks

Callback Function	Operation
`plGetDBName()`	Retrieves name of current record's parent database.
`plGetContainer()`	Retrieves name of current record's parent source file.
`plGetDocID()`	Retrieves ID number of current record.
`plGetLine()`	Retrieves current line of text.
`plSeekFieldName()`	Seeks to field name in current record.
`plSeekFieldID()`	Seeks to field ID in current record.
`plNextRecord()`	Seeks to next record in current database.
`plPrevRecord()`	Seeks to previous record in current database.
`plSeekRecord()`	Seeks to record with specified ID number in current database.
`plNextHit()`	Seeks to next record in Hitlist.
`plPrevHit()`	Seeks to previous record in Hitlist.
`plSeekHit()`	Seeks to record with specified rank in Hitlist.
`plSortHits()`	Sorts Hitlist by specified field; modifies Hitlist display; irreversible.
`plGetQuery()`	Retrieves query from most recent search.

The following conditions apply to PL callbacks:

- When searching for a linked DLL, PL checks the following directories in order: the PL home directory, the Windows directory, and all directories that are specified by the PATH environment variable.
- The `BOOL` return values for all callbacks except `plGetLine()` and `plGetDocID()` are TRUE (1) for success and FALSE (0) for failure.
- As long as `plGetLine()` returns a line from the field to which you are seeking, the function returns TRUE (1); when a field boundary is crossed, the function returns FALSE (0).
- `plGetDocID()` returns the ID number (a long) of the current record; if an error occurs, the function returns -1.
- All callback functions are relative to the current record in PL. At invocation time, the current record is the one that is currently active in the interface. In order for a record to be active in the PL interface, either a record window or the Search Results Window must have focus. If the Search Results Window has focus, the highlighted record in the Hitlist is the active record. After invocation, your DLL function can change the current record—by calling `plNextHit()`, etc.
- All lists are zero-based (e.g., `plSeekHit(1)` specifies the second record in the Hitlist).
- The number specified with `plSeekFieldID()` is the position of the desired field in the current database's definition file (e.g., `plSeekField(2)` specifies the field listed third in the definition file).
- `plGetLine()` must be initialized; before calling this function, call `plSeekField()`.
- The `plSortHits()` function modifies the state of the PL interface. If this function is called, the Hitlist will be sorted irreversibly; ranking by relevance cannot be restored.

USING PERSONAL LIBRARIAN AS A DDE SERVER

This section presents a brief overview of Personal Librarian's ability to act as a DDE server. DDE links allow other applications to communicate directly with PL, where the communication can be initiated and controlled by the other application.

Unless stated otherwise, all features (items) discussed are for the "PLData" topic. This discussion assumes familiarity with the DDE protocol. For specific information about the implementation of a DDE client, refer to the appropriate documentation available from Microsoft and other vendors. In addition, this topic is discussed in detail in many books about Microsoft Windows programming and is available from a variety of publishers.

APPLICATION SCENARIOS

- If you want an application to automatically open the source file that contains the currently active record in PL, the application can request the filename from PL.

- If you want an application to track the records that are opened during a PL search session, the application can request that PL notify it each time a record is opened. The application can use the information from PL to create a file that lists the opened records.

- If you want an application to create a list of the ten highest ranked records found for each search in PL, the application can request that PL notify it each time a search is performed, at which point the application can query PL for the ten highest ranked records.

- If a database is updated hourly, and you want to be notified when a routine query retrieves new records, you can create an application that executes hourly searches in PL and checks the results for the presence of new records.

INITIATING AND TERMINATING A CONVERSATION

A conversation with PL is established using the standard DDE protocol. Your application can elect to use either the low level WM_DDE message protocol or the DDE Management Library (DDEML) that is part of Microsoft Windows 3.1.

The Service—or Application—name of Personal Librarian is "PL". The only application topics currently supported by PL are "PLData" and the generic "System" topic. In order to succeed, a request to establish a DDE conversation must supply these service and topic strings.

PL will respond to WILDCONNECT requests, and can support any number of simultaneous conversations. Because a separate “state” object is created in PL for each new conversation, different applications can communicate with PL without interfering with each other.

REQUESTING DATA

After establishing a conversation, an application can request data from PL.

PL recognizes a total of eight items, three of which refer to a *current record*. The current record is maintained as a state supporting the DDE conversation within PL. Before the notion of a current record is valid, a request must be synchronized with the PL interface; this is done with a command that is discussed in a later section. During synchronization, the current record is set to the currently active record on the screen, which may be either an open record window or a highlighted Hitlist summary.

The other five supported items refer to the most recent action of a particular type.

PL recognizes the following items:

(1)	"DOCID"	ID of current record
(2)	"DBNAME"	name of db that contains current record
(3)	"CONTAINER"	name of source file that contains current record
(4)	"QUERY"	last performed query
(5)	"DBOPEN"	last opened database (name)
(6)	"DBCLOSE"	last closed database (name)
(7)	"DOCOPEN"	last opened record (dbname / docid)
(8)	"DOCCLOSE"	last closed record (dbname / docid)
(9)	"TopicItemList"	tab-delimited list of items supported under the PLData topic
(10)	"Formats"	tab-delimited list of formats supported (currently TEXT only)

The data returned from a request for items 1–6 return a single item of information. Data items 7 and 8 return both a database and a record ID number. The two data items are returned as a single string and are separated by " / " (space, forward slash, space). PL is not case sensitive with regard to the item name of requests.

ADVISE LOOPS

PL supports DDE advise loops as part of a conversation. Each conversation can have multiple advise loops, as long as each advise loop is initiated on a different item. Advise loops in different conversations can be initiated on the same item and can be either warm or hot. If your application establishes a warm loop on an item, it will be notified that a change to that item has occurred but has to explicitly request a new value from PL using a request discussed in the previous section. If the application establishes a hot advise loop, the new value of the item will be sent automatically whenever it changes. Advise loops can be established for the following items:

```
(1)   "DBOPEN"    notify of any db opened (name)

(2)   "DBCLOSE"   notify of any db closed (name)

(3)   "DOCOPEN"   notify of any rec opened
                  (dbname / docid)

(4)   "DOCCLOSE"  notify of any rec closed
                  (dbname / docid)

(5)   "QUERY"     notify of any new search (query)
```

The format of the data supplied for advise items 3 and 4 is the same as for the request items 7 and 8 discussed in the previous section. PL is not case sensitive with regard to the item name of advise loops.

EXECUTING COMMANDS

You can ask PL to execute certain commands; some of the supported commands are part of the normal data exchange between a custom application and PL; other commands can be used to execute links within PL. Requests for values of items 1, 2, and 3—see *Requesting Data* on page 290—are only valid if the current record was previously synchronized with the state of the interface. To perform this synchronization, an application must send the "RESET_DOC" command to PL.

```
(1)   "RESET_DOC" reset current record to currently
                  active window
```

This command will check the PL interface to see if either a record window or the Search Results Window is active (i.e., has focus) and set the current record accordingly. If neither window is active, the database name will be set to "" (empty string), and the record ID to -1 (invalid rec). Once a current record is set, your custom application can request data from PL by issuing requests. After the current record is set, it can be modified with the following commands:

```
(2)   "NEXT_HIT"  set current record to next hit in
                  results list

(3)   "PREV_HIT"  set current record to previous
                  hit in results list

(4)   "NEXT_DOC"  set current record to next rec in
                  db

(5)   "PREV_DOC"  set current record to previous
                  rec in db
```

This allows the application to iterate over the records in a results list or over the records in a database.

In addition to issuing commands that support data requests, you can also ask PL to execute links. The links that can be executed through DDE are similar to the hypertext links in PL. Each link has a corresponding DDE command, with the exception of command links, which are not supported; external applications can be launched directly from your custom application.

The following items are supported for executing links:

```
(6)   "EXEC_XLINK"      execute a DLL link
```

(7)	"EXEC_SLINK"	execute a search link
(8)	"EXEC_DLINK"	execute a document link
(9)	"EXEC_ILINK"	execute an image link
(10)	"EXEC_TLINK"	execute a target link

All of these link commands require arguments. The argument passing mechanism requires one additional execute command:

(11)	"RESET_ARGS"	Initialize the argument stack

The technique of passing arguments is discussed in the following section. PL is not case sensitive with regard to commands.

POKING ARGUMENTS

Poking is supported to supply arguments to functions. For each conversation, PL implements an internal stack, onto which arguments can be pushed. The normal order of operation when issuing a command that requires arguments is as follows:

1. Execute a "RESET_ARGS" command to reset the argument stack
2. Poke individual arguments
3. Execute the link command (commands (6) through (10))

To poke an argument, the following item name must be supplied:

(1)	"ARG"	supply an argument to a function

and the data is the actual argument being pushed onto the stack.

Items are retrieved from the stack in the same order in which they were pushed onto the stack. If you want to execute a function in the following format:

```
EXEC_FUNC (arg1, arg2)
```

you must first push arg1, then arg2 onto the stack. All arguments are passed as strings. The commands 6–10, listed in the previous section, require the following parameters:

```
EXEC_XLINK  DLL name
            function name
            arguments to the function

EXEC_SLINK  query

EXEC_DLINK  record ID
            field ID (optional, defaults to 0)
            database name (optional, defaults to
            current)

EXEC_ILINK  full path of the image to open
            image title (optional, defaults to
            image path)

EXEC_TLINK  label
```

When an argument is listed as optional, it is not required. PL is able to determine the number of arguments passed, and use default arguments where needed.

The execution of a function uses arguments *destructively*; that is, you cannot pass the arguments to a function once and then execute the function multiple times. Arguments have to be resupplied before the next command is issued. It is possible to push arguments onto the stack for more than one function and then call those functions without poking arguments in between. However, this requires you to supply the maximum number of arguments to those functions that take optional arguments.

The following example outlines the procedure with which you can open the current database' 17th record at the field that is defined first in the database's definition file:

1. Execute a "RESET_AGS" command.

2. Poke item "ARG" with value "17".

3. Poke item "ARG" with value "1".

4. Execute an "EXEC_DLINK" command.

SYSTEM TOPIC

The following items are recognized for the "System" topic. Please refer to third-party documentation for more detailed information.

(1)	"Topics"	tab-delimited list of topics supported
(2)	"SysItems"	tab-delimited list of items supported under the "System" topic
(3)	"Formats"	tab-delimited list of formats supported (currently TEXT only)
(4)	"TopicItemList"	tab-delimited list of items supported under the "System" topic
(5)	"Status"	readiness of the server ("Busy" or "Ready")

MAKING SPECIAL CHARACTERS SEARCHABLE

CUSTOMIZING GLOBAL TOKENIZATION RULES

During the indexing process, PL-Admin ignores punctuation and non-alphanumeric characters in source files; this makes special characters—ampersand (&), percent sign (%), etc.—non-searchable. Moreover, these characters are stripped from queries during searching in Personal Librarian.

If you want certain special characters to be searchable in all databases—and recognized within queries when the databases are searched—use the following procedure:

1. Using a text editor or word processor, change string 33 in `defaults.cpl`, located in PL's `\shared` subdirectory, to read exactly as follows:

   ```
   33 CplTabledRomanceTokenizer
   ```

2. Make the desired changes to `tknztbld.cpl`, located in PL's `\shared` subdirectory—see *Indexing Special Characters* on page 298.

3. *Indexing Special Characters* on page 298*Indexing Special Characters* on page 298Build your databases.

CAUTION: If you will be customizing global tokenizer settings, you should do so before building any databases. If you modify tokenization rules after building databases, you must completely rebuild them.

Note: A database with its own tokenization rules—see *Making Special Characters Searchable in a Database* on page 220—will not be affected by global tokenizer settings.

INDEXING SPECIAL CHARACTERS

You, as a database administrator, can customize the rules that dictate which characters can be part of "words" and which will be interpreted as word-separator characters. In previous versions, PL-Admin's indexing process ignored any characters that were not numbers or letters; if a non-alphanumeric character (any punctuation mark or symbol) appeared in the middle of a text string, the tokenizer would have broken the string into separate "words"; for example, a phone number such as *234-5678* would have been indexed as two words: *234* and *5678*; however, by customizing the tokenizer, you can now control the rules that determine word continuation.

CAUTION: If you want to make searchable any characters or character sequences that are used as operators in PL's query language, contact the PLS customer support department.

Functionality

As the indexing process begins, PL's tokenizer reads your source files and compares each character with the tokenization rules. During this process, the tokenizer determines which characters are indexable, which should stay together, which should be broken into individual strings, and which are separator characters (e.g., the space character). Such determinations are based on four sets of rules that define tokenization.

1. Character Class Definitions—see page 299.

2. Character Classification Map—see page 301.

3. Word Continuation Rules—see page 303.

4. Canonization Map—see page 308.

Note: The order of these rules is critical and must remain the same in the `tknztbld.cpl` file.

Before You Begin...

As a PL user, you have as your default tokenizer the *Romance Tokenizer*. Using this tokenizer, you cannot customize your indexing preferences; however, by changing your tokenizer to the *Tabled Romance Tokenizer*, you may establish your own tokenization rules. You can make your new tokenization rules apply to a specific database or to all Personal Librarian databases by following the instructions in, respectively, *Making Special Characters Searchable in a Database* on page 220 and *Customizing Global Tokenization Rules* on page 297.

Character Class Definitions

Characters that you want to be searchable in your database(s) must be assigned to one of the classes that you define in the character class definitions; for example, if you know that your database will include letters, numbers, and dollar signs, you will want to define classes such as *letter*, *number,* and *dollar* in order for these letters to be searchable.

With a few exceptions, you may name your classes whatever you wish, although descriptive names will aid your organization; however, you may not label your classes the following five names because they are reserved values that have specific meaning to the tokenizer:

- Invalid
- EndRule
- EndOfDefs
- EndBuff
- Break

Only those classes defined in the Character Class Definitions section of the file and the five reserved class names can be used in the tokenizer definition; undefined names used later in the Character Class Map will cause an error. You may define up to 250 character classes, although for most applications, ten or fewer classes will be sufficient. All class names are case sensitive.

The following is an example of a character class definition set. Note that you must complete your definition list with the default value *EndRule*.

```
Letter
Number
Dollar
EndRule
```

Performance Hints

- List first the classes that are represented most frequently in your database; for example, if characters of the *letter* class occur most frequently in your database, list the *letter* class first.

- To maximize the tokenizer's speed, define as few classes as possible.

Character Classification Map

Because the Character Classification Map associates ASCII decimal character codes with the classes that you defined in the Character Class Definitions, you can assign a particular character to one of your defined character classes. When assigning characters to classes, only assign those characters that you want to be indexed. All other characters in your database will automatically be assigned to the *break* class and will not be indexed; ASCII NULL will be mapped as the EndBuff class.

When creating your Character Classification Map, note that each classification statement must be on its own line and include the ASCII decimal value of the character and the character's class definition. In addition, each classification statement may include an optional comment; for example, the comment statements in the following map document the name of the characters. Separate each classification statement with tabs or spaces. Precede comments with the pound character (#) and, at the last line of the map, include an end-of-list decimal value and class: *-1* and *EndOfRule*, respectively. The following table is an example of a character classification map set:

Example:

```
# Decimal Value   Class       Comment
# Special Characters
      33          Dollar      # Character $
# Digits
      48          Number      # Character 0
      49          Number      # Character 1
      50          Number      # Character 2
      51          Number      # Character 3
      52          Number      # Character 4
      53          Number      # Character 5
      54          Number      # Character 6
      55          Number      # Character 7
      56          Number      # Character 8
      57          Number      # Character 9
# Upper-case Letters
      65          Letter      # Character A
      66          Letter      # Character B
      67          Letter      # Character C
      68          Letter      # Character D
      69          Letter      # Character E
      70          Letter      # Character F
      71          Letter      # Character G
      72          Letter      # Character H
      73          Letter      # Character I
      74          Letter      # Character J
      75          Letter      # Character K
      76          Letter      # Character L
      77          Letter      # Character M
      78          Letter      # Character N
      79          Letter      # Character O
      80          Letter      # Character P
      81          Letter      # Character Q
      82          Letter      # Character R
      83          Letter      # Character S
      84          Letter      # Character T
      85          Letter      # Character U
      86          Letter      # Character V
      87          Letter      # Character W
      88          Letter      # Character X
      89          Letter      # Character Y
      90          Letter      # Character Z
# Lower-case Letters
      97          Letter      # Character a
      98          Letter      # Character b
      99          Letter      # Character c
      100         Letter      # Character d
      101         Letter      # Character e
      102         Letter      # Character f
      103         Letter      # Character g
      104         Letter      # Character h
      105         Letter      # Character i
      106         Letter      # Character j
      107         Letter      # Character k
      108         Letter      # Character l
      109         Letter      # Character m
      110         Letter      # Character n
      111         Letter      # Character o
      112         Letter      # Character p
      113         Letter      # Character q
      114         Letter      # Character r
      115         Letter      # Character s
      116         Letter      # Character t
      117         Letter      # Character u
      118         Letter      # Character v
      119         Letter      # Character w
      120         Letter      # Character x
      121         Letter      # Character y
      122         Letter      # Character z
      -1          EndOfDefs   # Predefined value
```

Word Continuation Rules

The Word Continuation Rules specify which sequences of characters cannot be separated from one another when the tokenizer is breaking a stream of data into words or logical strings (i.e., which character sequences continue words and which sequences end them). Before you begin defining your Word Continuation Rules you must have established both the Character Class Definitions and the Character Classification Map, as previously explained.

Each Word Continuation Rule must include the Character Class Names (that you have defined) separated either by tabs or by spaces. Even though you will not have listed it in the character class names, the asterisk (*) character may also be included in your word continuation rules because it is a special indicator to the tokenizer to read one or more characters of the given class. In addition, the value *EndRule* must always end your list of Word Continuation Rules.

A rule defines which characters from the specified classes may not be separated during indexing. Table 11 lists several Word Continuation Rule examples; they are based on the class definitions *Letter*, *Number*, and *Dollar* and on the input text `123READ45$$LOG$6F78`.

Table 11 Word Continuation Rule Examples

Rule	Explanation	Tokenized Text
`Letter`	Each occurring Letter character may be indexed separately.	`R E A D L O G F`

`Letter *`
`Letter Letter`
	A single Letter character may be indexed separately and any number of Letter characters in a row continues a word during indexing.	`READ LOG F`

`Number`	Each occurring Number character may be indexed separately.	`1 2 3 4 5 6 7 8`

`Number *`
`Number Number`
	A single Number character may be indexed separately and any number of Number characters in a row continues a word during indexing.	`123 45 6 78`

`Dollar`	Each occurring Dollar character may be indexed separately.	`$ $ $`

Table 11 Word Continuation Rule Examples

Rule	Explanation	Tokenized Text

`Dollar *`
`Dollar Dollar`
	A single Dollar character may be indexed separately and any number of Dollar characters in a row continues a word during indexing.	`$$ $`

`Letter Number`	A Letter character followed by a Number character continues a word during indexing.	`D4 F7`
`Letter * Number *`	Any number of Letter characters followed by any number of Number characters continues a word during indexing.	`READ45 F7`
`Letter Dollar`	A Letter character followed by a Dollar character continues a word during indexing.	`G$`
`Letter * Dollar *`	Any number of Letter characters followed by any number of Dollar characters continues a word during indexing.	`LOG$`
`Number Letter`	A Number character followed by a Letter character continues a word during indexing.	`3R 6F`
`Number * Letter *`	Any number of Number characters followed by any number of Letter characters continues a word during indexing.	`123READ 6F`
`Number Dollar`	A Number character followed by a Dollar character continues a word during indexing.	`5$`

Table 11 Word Continuation Rule Examples

Rule	Explanation	Tokenized Text
`Number * Dollar`	Any number of Number characters followed by any number of Dollar characters continues a word during indexing.	`45$$`
`Dollar Letter`	A Dollar character followed by a Letter character continues a word during indexing.	`$L`
`Dollar * Letter *`	Any number of Dollar characters followed by any number of Letter characters continues a word during indexing.	`$$LOG`
`Dollar Number`	A Dollar character followed by a Number character continues a word during indexing.	`$6`
`Dollar * Number *`	Any number of Dollar characters followed by any number of Number characters continues a word during indexing.	`$6`

`Letter *`
`Number *`
`Letter Number`
`Number Letter`
	Any number of Letter characters followed by any number of Number characters and any number of Number characters followed by any number of Letter characters continues a word during indexing. Any number of Letter characters or any number of Number characters continues a word during indexing.	`123READ45 LOG 6F78`

Table 11 Word Continuation Rule Examples

Rule	Explanation	Tokenized Text

`Number *`
`Dollar *`
`Number Dollar`
`Dollar Number`
	Any number of Number characters followed by any number of Dollar characters and any number of Dollar characters followed by any number of Number characters continues a word during indexing. Any number of Number characters or any number of Dollar characters continues a word during indexing.	`123 45$$ $6 78`

`Letter *`
`Number *`
`Dollar *`
`Letter Number`
`Number Letter`
`Number Dollar`
`Dollar Number`
`Dollar Letter`
`Letter Dollar`
	Any Letter string followed by a Number string, any Number string followed by a Letter string, any Number string followed by a Dollar string, any Dollar string followed by a Number string, any Dollar string followed by a Letter string, and any Letter string followed by a Dollar string continues a word during indexing. Any number of Letter characters, any number of Number characters, or any number of Dollar characters continues a word during indexing.	`123READ45$$LOG$6F78`

Remember that, although it is not shown at the end of the examples, you must complete your last rule with the default value *EndRule*; you may never include this value as part of any other rule.

Note: In the chart, the examples are independent of one another; however, the last three examples do functionally link to one another and illustrate the typical rules that you will create.

Performance Hints

- Keep rules as simple as possible. The simpler the rule, the faster the word-continuation tests. If a rule uses a few character classes and contains only a few items, it is simple. If a rule uses more than two character classes or uses two or more classes in different permutations, it is complex.

- To maximize the tokenizer's speed, define as few rules as possible.

Canonization Map

After you have established the tokenizer's Character Classes, Classification Map, And Word Continuation Rules, the tokenizer is able to return a word based on your definitions; however, to make your searches faster and to reduce the size of your database dictionary, you should now define the Canonization Map.

The Canonization Map takes all typographic variances of a word and translates them to a regularized or, canonical, form. The canonizer is useful, for example, for converting all lower-case letters to upper case so that, when you are doing a search, the word *NeXT* will match the words *next*, *NEXT*, and *nExt*; however, only the canonical form of the word that is defined in Canonization Map will appear in the database dictionary.

When customizing the Canonization Map, only include in the table those characters that you want to have translated before tokenization. Each canonization statement must be on its own line and include the input ASCII decimal value and the output ASCII decimal value. In addition, it may include an optional comment that defines the mapping conversion. Precede comments by the pound character (#), and separate each statement item with either tabs or spaces.

The following Canonization Map (which is the default map for the tokenizer) converts all lower case characters to upper case. Note that the value *-1* occurs at the end of the map; you must always end your mapping table with this value.

Canonization Map:

```
# Map a-z to the canonical A-Z so that all letters
# will be upper case.
#
# Input     Output
# Decimal   Decimal
# Value     Value       Comment
  97        65          # a canonizes to A
  98        66          # b canonizes to B
  99        67          # c canonizes to C
  100       68          # d canonizes to D
  101       69          # e canonizes to E
  102       70          # f canonizes to F
  103       71          # g canonizes to G
  104       72          # h canonizes to H
  105       73          # i canonizes to I
  106       74          # j canonizes to J
  107       75          # k canonizes to K
  108       76          # l canonizes to L
  109       77          # m canonizes to M
  110       78          # n canonizes to N
  111       79          # o canonizes to O
  112       80          # p canonizes to P
  113       81          # q canonizes to Q
  114       82          # r canonizes to R
  115       83          # s canonizes to S
  116       84          # t canonizes to T
  117       85          # u canonizes to U
  118       86          # v canonizes to V
  119       87          # w canonizes to W
  120       88          # x canonizes to X
  121       89          # y canonizes to Y
  122       90          # z canonizes to Z
  -1        -1          # End-of-canonization value
```

After the tokenizer completes the default canonization, all other values remain unchanged unless you define other characters for mapping; for example, if you have a database that includes foreign words, you may want to convert all special characters to a canonical form.

Example:

```
# Map a-z to the canonical A-Z so that all letters
# will be upper case.
#
# Input     Output
# Decimal   Decimal
# Value     Value       Comment
  194       65          # Â canonizes to A
  239       73          # ï canonizes to i
  199       67          # Ç canonizes to C
  233       101         # é canonizes to e
```

BATCH PROCESSING OF ADMINISTRATION TASKS

If you are an advanced user and are comfortable working with the DOS operating system, you can build databases without PL-Admin. By using utilities that can be installed separately from PL, you can perform many database administration tasks with commands that you issued directly to your operating system. These commands can be entered at the system prompt or as part of a batch script.

CAUTION: A database that is built with these batch administration utilities can be searched in Personal Librarian. However, it is *not* interoperable with PL-Admin—you cannot use PL-Admin to perform any administration for a database that is built with the batch utilities, and vice versa.

FUNCTIONAL OVERVIEW OF BATCH ADMINISTRATION UTILITIES

Table 12 summarizes the functions of PL's batch administration utilities.

Table 12 Batch Administration Utilities

Program	Description
`plbuild.exe`	Creates a database index file and adds the new database to the ShortList.
`placcess.exe`	Imposes password restrictions on a database.
`pladd.exe`	Adds source files to an existing database.
`pldelete.exe`	Deletes records from a database.
`plupdate.exe`	Replaces records in a database.
`plreorg.exe`	Removes unused space from a database's index file; also partitions index as needed.
`plcrypdb.exe`	Encrypts and decrypts source files in a database.
`plhier.exe`	Builds a table of contents file for a database.

VERIFYING SUCCESSFUL UTILITY OPERATION

Verifying Successful Utility Operation

When each of the utility programs stops, it returns an exit code to the operating system; a code of 0 indicates successful operation, while 1 means failure. The code that the system holds will always be from the most recently executed program. After each program has run, you can verify its operation by checking the current value of this code.

Example:

```
plcreate “Federal Law” -f c:\db\fedlaw
if ERRORLEVEL 1 goto fail
echo Database created successfully
goto end
:fail
echo Database could not be created
goto end
:end
```

CREATING A DATABASE

You can create a PL database by invoking the utility program `plbuild.exe`. Before running this utility, you must create the database's definition file (`*.def`) by using techniques discussed in *Modifying a Definition File after Database Creation* on page 222.

Modifying a Definition File after Database Creation on page 222The database you create with `plbuild.exe` will be empty. You can add records to it and manage its contents by scripting with the other utility programs, which are covered in later sections.

Note: When you create a database with `plbuild.exe`, a reference to it is added to the copy of `dblist.cpl` that occupies the batch utilities directory. To make the database available for searching in Personal Librarian, you must copy the database's reference line from that file and add it to the copy of `dblist.cpl` that resides in PL's `\shared` subdirectory.

Syntax:

```
plbuild listname [-f dbpath]
```

where:

- *listname* is the ShortList name of the database that will be created. This parameter is subject to the ShortList name guidelines listed in *Assigning a ShortList Name* on page 204. If this parameter includes spaces, it must be enclosed in double quotation marks.

- *dbpath* specifies the path and name of the database's index file; this optional parameter cannot include a filename extension.

You must use the optional -f argument to specify a path for the index file—unless you run `plbuild.exe` from the database's library directory—or to give the file a name other than *listname*. If the -f argument is absent, the base name of the database's index file will be *listname*.

Examples:

```
plbuild fedlaw
plbuild “Federal Law” -f c:\db\fedlaw
```

Note: If *dbpath* specifies a filename that is already used by a file in the specified directory, the command will fail. It will also fail if *listname* is already used in the ShortList.

DEFINING A PASSWORD FOR SEARCH ACCESS RESTRICTION

You can restrict search access to a database by assigning it a password with `placcess.exe`; only users who know the password will be able to open the database in PL.

Syntax:

placcess *listname* [-p *password*] [-v]

where:

- *listname* is the ShortList name of the database to which access will be restricted. If this parameter includes spaces, it must be enclosed in double quotation marks.
- *password* is the password that a user will have to enter in order to open the database in PL. If you want to impose password restrictions, you must include the -p argument and *password* in the command. This optional parameter can be a string of up to 12 alphanumeric characters.

By including the -v argument, you can prohibit authorized users from printing records or the Hitlist to either hard copy or ASCII files.

Example:

```
placcess fedlaw -p opensesame
```

ADDING FILES

You can use the `pladd.exe` program to add source files to an existing database.

Note: You cannot remove source files from a database with a batch administration utility. If you want to remove a source file from a database that was built with the batch utilities, you must delete the database and rebuild it without the file.

Syntax:

```
pladd [-a inxfile] [-f list] listname [addfile…]
```

where:

- *inxfile* specifies the indexed source file to which records that are contained in *addfile* will be appended if the -a argument is used. This parameter must include the same path that was specified when the indexed file was added to the database.
- *list*, used in conjunction with the -f argument, specifies the path and name of an ASCII text file that contains the names of source files that will be indexed.
- *listname* is the ShortList name of the database to which source files will be added. If this parameter includes spaces, it must be enclosed in double quotation marks.
- *addfile* specifies the path(s) and name(s) one or more source files that will be indexed. If you anticipate moving the database after adding files to it, you should include only a relative path in this parameter. This parameter is not required if the -f argument is used; if this parameter and the -f argument are both used, the files specified by *addfile* will be indexed first.

If you want to append the contents of new ASCII source files to a previously indexed ASCII file, you can use the -a argument. When you append files in this manner, the original copies of the files may be deleted, as they will not be required for searching.

By using the -f argument, you can direct `pladd.exe` to index files that are listed in an ASCII text file. When creating the text file named in *list*, you should observe the following guidelines:

- One or more file names must be listed.
- File names must be separated by a space, tab, or end-of-line character (e.g., a carriage return).
- Each file name may include a path.

Examples:

```
pladd fedlaw c:\db\fl.001

pladd -a c:\db\fl.001 fedlaw c:\db\fl.002

pladd -f c:\db\lawdocs.txt fedlaw
```

Tip: You should, when adding multiple files, index them with a single iteration of `pladd.exe` rather than devote a separate iteration to each individual file. This makes for more efficient indexing by reducing the number of unique words indexed in relation to the total amount of data processed.

DELETING RECORDS

You can delete records from an existing database with the `pldelete.exe` program.

Syntax:

pldelete [-f *list*] *listname* [*delID…*]

where:

- *list*, used in conjunction with the -f argument, specifies the path and name of an ASCII text file that contains the ID numbers of records that will be deleted.
- *listname* is the ShortList name of the database from which records will be deleted. If this parameter includes spaces, it must be enclosed in double quotation marks.
- *delID* specifies the ID number(s) of one or more records that will be deleted. This parameter is not required if the -f argument is used; if this parameter and the -f argument are both used, the records specified by *delID* will be deleted first.

By using the -f argument, you can direct `pldelete.exe` to delete records that are specified by ID number in an ASCII text file. When creating the text file named in *list*, you should observe the following guidelines:

- One or more record ID's must be listed.
- ID numbers must be separated by a space, tab, or end-of-line character (e.g., a carriage return).

Examples:

```
pldelete fedlaw 98 99 143

pldelete -f c:\db\recs2del.txt fedlaw
```

Tip: You should, when deleting multiple records, use a single iteration of the `pldelete.exe` command rather than devote a separate iteration to each individual record. This makes for more efficient indexing by reducing the number of unique words indexed in relation to the total amount of data processed.

REPLACING RECORDS

You can update a record in an existing database by using the `plupdate.exe` program to replace it with the contents of a specified file.

Words found in the original record but not in the replacement are removed from the database's index; likewise, words found in the replacement but not in the original are added to the index.

Syntax:

```
plupdate [-a] listname recID newrecfile
```

where:

- *listname* is the ShortList name of the database in which records will be replaced. If this parameter includes spaces, it must be enclosed in double quotation marks.
- *recID* is the ID number of the record that will be updated.
- *newrecfile* specifies the path and name of the file that will replace the record specified by *recID*. The contents of *newrecfile* must constitute a single record and be compatible with the database’s field definitions. If you anticipate moving the database after updating its records, you should include only a relative path in this parameter.

By using the -a argument, you can direct `plupdate.exe` to append the contents of an ASCII text file that is specified by *newrecfile* to the end of the ASCII source file that contains the record specified by *recID*. When you append files in this manner, the original copies of the files may be deleted, as they will not be required for searching.

Example:

```
plupdate fedlaw 99 c:\db\fl.099
```

REORGANIZING A DATABASE'S INDEX

You can use the `plreorg.exe` program to reorganize a database's index. You should consider reorganizing a database only under the conditions described in *Reorganizing a Database's Index* on page 244.

Reorganizing a Database's Index **on page 244Syntax:**

```
plreorg [-{c|u}] listname reorgname
```

where:

- *listname* is the ShortList name of the database that will be reorganized. If this parameter includes spaces, it must be enclosed in double quotation marks.

- *reorgname* specifies the path and base filename to which the reorganized index file will be saved; this parameter must include a path.

You can use the -c argument to compact the database to archived form, in which it occupies the smallest possible amount of disk space. This option cannot be used in conjunction with the -u option.

With the -u argument, you can unarchive a previously archived database prior to updating it. By doing so, you can avoid the prohibitive disk space overhead that is required when updating a database that is in archived form. You should use this option only with a database that has been archived; it cannot be used in conjunction with the -c option.

CAUTION: Updating an archived database is extremely wasteful of disk space, as compared to doing so to an unarchived one. Before updating an archived database, you should first unarchive it by running `plreorg.exe` with the -u option.

Examples:

```
plreorg fedlaw c:\db\lawopt01

plreorg -c "Federal Law" c:\db\lawopt01
```

To reorganize a database:

1. In the path that is specified by *reorgname*, place a copy of the definition file for the database that is specified by *listname*.

2. Rename the duplicate definition file to have the base name that is specified in *reorgname* and the `.def` extension.

3. Run the `plreorg.exe` program.

4. If the reorganized database was not saved to the directory occupied by the original index, move it there, along with its definition file and any partitions.

5. Rename the reorganized index and its definition file to have the filenames of the originals.

ENCRYPTING DATABASE SOURCE FILES

You can use the `plcrypdb.exe` program to encrypt all of a database's source files so that PL is the only avenue through which their contents can be accessed. Only ASCII source files can be encrypted.

Note: You must decrypt an encrypted database prior to updating it.

Syntax:

```
plcrypdb -{e|d} listname
```

where *listname* is the ShortList name of the database that will have its constituent source files encrypted or decrypted. If this parameter includes spaces, it must be enclosed in double quotation marks.

If you use the -e argument, all of the specified database's source files will be encrypted. Use of the -d argument will decrypt the specified database's source files.

Tip: You should maintain unencrypted copies of the source files in an encrypted database. This will ensure that you have the elements for a functioning version of the database in the unlikely event that file damage occurs during encryption or decryption.

BUILDING A TABLE OF CONTENTS

You can use the `plhier.exe` program to build a table of contents file (`*.hir`) for an existing database.

Syntax:

```
plhier listname [-f textfile]
```

where:

- *listname* is the ShortList name of the database for which a table of contents file will be built. If this parameter includes spaces, it must be enclosed in double quotation marks.
- *textfile*, used in conjunction with the -f argument, specifies the path and name of an ASCII text file from which heading information will be extracted.

Examples:

```
plhier fedlaw

plhier fedlaw -f c:\db\lawtoc.txt
```

By using the -f argument, you can direct `plhier.exe` to read heading information from a specially formatted ASCII text file, rather than from heading fields that are embedded in the database's source files. Each line in the text file must conform to the following format:

```
LHeading                      begID        endID        1
```

where:

- *L* specifies the level that a heading will occupy in the table of contents hierarchy. This parameter must occupy character position 0; valid values for this parameter are the integers from 1 to 7.

- *Heading* is the heading text that will be displayed in the table of contents. This parameter must begin at character position 1 and can extend as far as character position 132.

- *begID* specifies the ID number of the first record that occupies a heading's domain. This parameter must begin at character position 133 and can extend as far as character position 139.

- *endID* specifies the ID number of the last record that occupies a heading's domain. This parameter must begin at character position 140 and can extend as far as character position 146.

- 1 is a constant that must occupy character position 147.

When creating the text file, you should observe the following guidelines:

- The first line in the file must specify a top level heading.

- A line of heading information must exist for every record in the database.

- Every record must occupy the ID number range defined for its parent heading.

Example:

```
1Front Matter                       1     3     1
2Preface                            1     3     1
3Organizational Overview            1     1     1
3Conventions                        2     2     1
3Assumptions                        3     3     1
1Part 1: Getting Started            4     6     1
2Chapter 1: Introduction            4     6     1
3Searching with Personal Librarian  4     4     1
3Text Database Fundamentals         5     6     1
4Databases, Records, and Fields     5     5     1
4Stopwords                          6     6     1
1Part 2: Searching                  7     7     1
1Part 3: Database Administration    8     8     1
1Part 4: Appendices                 9     9     1
```

INDEX