

# WebSPIRS 4.0

## JavaScript Search Software

*Péter Jacsó*

---

---

*This article discusses the major features of the WebSPIRS 4.0 information retrieval software. It analyzes its interface, browsing, searching, output, and navigational features in detail. Selection of search terms, thesaurus look-up, field qualification and limiting of queries, definition of display formats, and sorting of results are discussed and amply illustrated by screenshots. Except for the limit function and some aspects of the indexes, the software gets a high rating.*

---

---

SilverPlatter's original CD-ROM software version for DOS in 1986 was a pioneer and has been popular for its simplicity. The Windows versions have added much functionality and grace to the CD-ROM software, and so did later the Client version and the browser-based version. The new WebSPIRS 4.0 software further enhances the synergy of powerful functionality and intuitive interface and proves what can be achieved using JavaScript.

WebSPIRS is available for subscribers to SilverPlatter on-line databases and also makes SilverPlatter databases mounted on the server of a university or a corporation accessible through a Web browser. Although users need Netscape 3.1 or Internet Explorer 4 or higher browser, the convenience of remote access at any time from anywhere—without installing a client software—is well worth the browser upgrade for those who have been sticking to earlier versions of browsers.

The overall design and the interface is top notch (with the notable exception of the very inappropriate limit functions). The navigation and help features are excellent; the thesaurus handling capabilities are very good but would greatly benefit from displaying posting information about the terms. The introduction of the ADJ positional operator is a welcome

development to increase precision, especially in full-text databases. The output facilities, including on-the-fly format customization by the user, are powerful and intuitive.

WebSPIRS 4.0 still does not offer perfect index browsing capabilities, although it is improving. The revitalized term mapping (known as Suggest feature) also needs improvement and database-specific fine tuning in the algorithm for generating a better list of descriptor terms from a user query. Multiple database searching is a rare asset of WebSPIRS 4.0. Finding the most promising database(s) for a topic with the database selection utility is fast and simple. However, the problems caused by the ill-designed limit function in single-database searching just multiply and become painfully inadequate in multiple-database searching, which would also need deduplication facility.

### Interface and Navigational Features

The navigational system in WebSPIRS is excellent. Despite the fact that there are many icons, function buttons, pull-down menus, pop-up windows, radio buttons, and check boxes, the screen layouts are clear and efficient (with the exception of the limit options that deserve bashing on every occasion). One of the tricks of the good design is the use of narrow typeface style that makes the text eminently legible. The other is the systematic and mostly consistent arrangement of navigational signs. The labeling of buttons and boxes is clear, but if it would not suffice, a small explanatory box pops up as you move the mouse over the buttons. Buttons that are not operational in a given context are grayed out. The button that launches the most likely next step displays the label text in green. For example, on the query form, the Start Search label appears in green as it is the

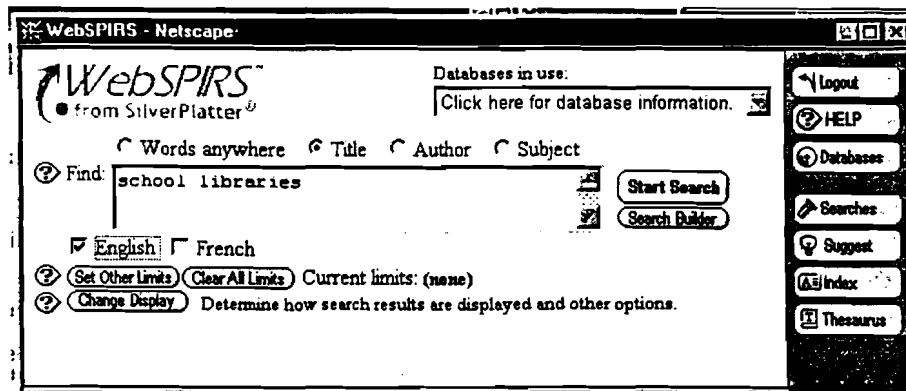


Figure 1. Search screen with function buttons, check boxes, and query cell.

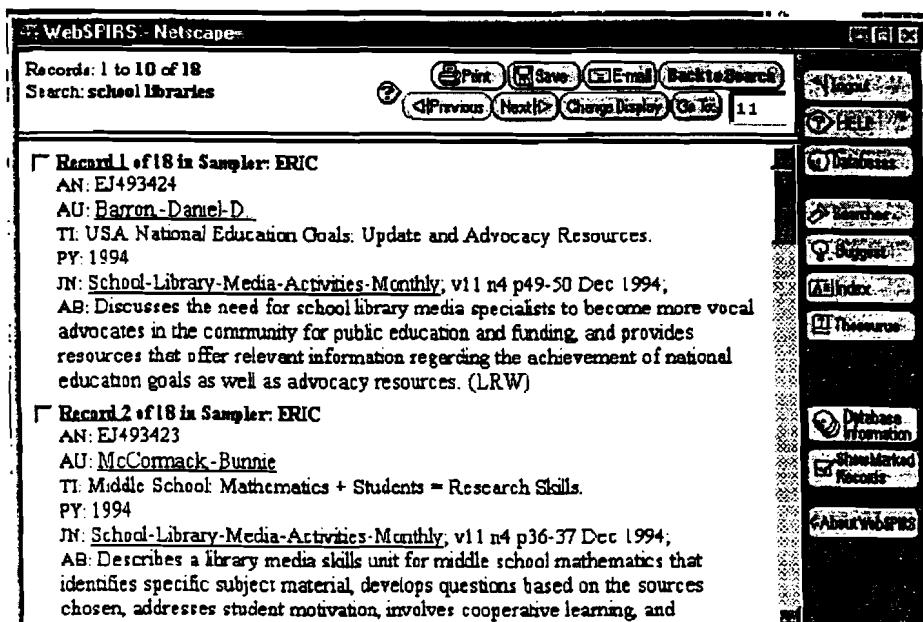



Figure 2. Consistent navigational and action buttons.

next most likely step (see Figure 1). If you have seen users waiting in front of the computer for something to happen but they forgot to press the Enter key, you realize how important such a prompt can be.

The results of the query—along with the query itself—are automatically displayed in a format predefined

by the system administrator. The navigational buttons remain on the screen, and the label on the Back to Search button becomes green. The user may choose another action (such as displaying the next ten results), but the button of the most likely action gets the green light (see Figure 2).

## WebSPIRS<sup>™</sup> Session in Process from SilverPlatter<sup>®</sup>

WebSPIRS is running in a different browser window. Click one of the buttons below, or click  for help.

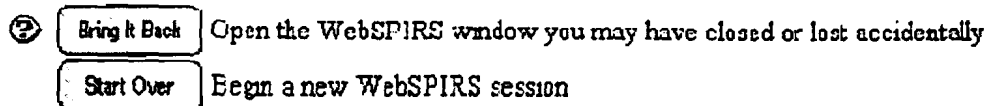


Figure 3. Bring back my screen.

It is also a godsend that if you would accidentally navigate yourself out of the system, a screen pops up offering to bring back the screen that you just left. This guardian angeling will spare a lot of agony (see Figure 3).

Beyond the intuitive and self-explanatory action buttons, the radio buttons and the check boxes prompt the users to restrict the terms to specific indexes (for example, to the title) or to limit the query to English language documents. The appropriateness of the choice of these radio buttons and check boxes will be discussed below.

### Help System

There is detailed on-line help documentation. Although the Help button is not strictly context sensitive (that is, it does not take you automatically to the help information about search options when you invoke help from the query template), the table of contents screen gives a good hint which section or subsection of the help file you should click on for help (see Figure 4).

In addition, next to some buttons there is a question mark symbol that takes you directly to the help information about the Find command, limit setting, and changing of display format. This is a nice combination. The help screen appears in a separate window that overlays (but does not cover) the active screen from where you asked for help (see Figure 5).

The hyperlinks from within a help subsection take you to examples. The related help topics at the bottom of the help screens (see Figure 6) guide you to other relevant help sections. The Close Window button helps the inexperienced user get rid of the window. This is as good as it gets for navigating in a well-structured and informative help file.

From within the Help table of contents section, a button takes you to database-specific help information. It is also available from the main search screen in a pull-down menu. The pull-down menu is needed when you do multi-database searching to specify which database you need help about. In

single database search mode, the name of the database should be prominently displayed without a pull-down menu. WebSPIRS 4.0 earns a high score for its "how can I help you" attitude and for its capability of really rendering, not just offering, help.

### Query Template and Basic Search

The query template provides more than sufficient space for entering even a very long search question. The query cell itself is scrollable to accommodate more lines. This is hardly needed, and the space could be used for an extra line of options above the query cell. The four radio buttons offer the possibility of restricting the search to one of the four indexes (see Figure 7). The query templates are somewhat different depending on the database(s) selected, but the layout is identical.

Check boxes could be more appropriate for restricting queries because the user could choose both the Title and the Subject check boxes to search for the query terms in the title and/or the descriptor fields at once. The radio button solution makes the choice mutually exclusive, that is, either in the title or in the subject. It is another matter that the search can be done in two steps and then combining the sets or that the experienced searcher can achieve this in the command mode using the statement: (school libraries) in ti,de.

The check boxes below the query cell for limiting subject searches are obviously database specific. In the ERIC sample, the two languages are appropriate for check box limiting (although Spanish may be a more likely candidate). More important, a check box would be badly needed for journal articles (CIJE documents for the connoisseurs) to limit the search to the preferred document type if a user wishes to do so. I would also like to see odometer boxes for publication range setting right under the query template. (They are available under the Additional Limit options as filters but would be much better under the query cell. As I

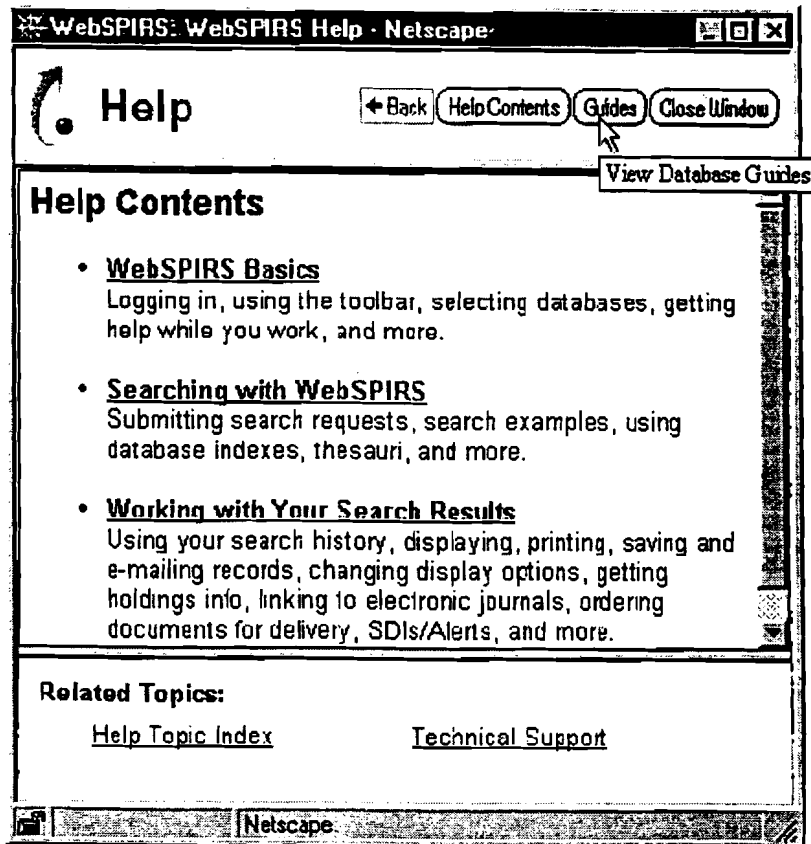


Figure 4. Help table of contents.

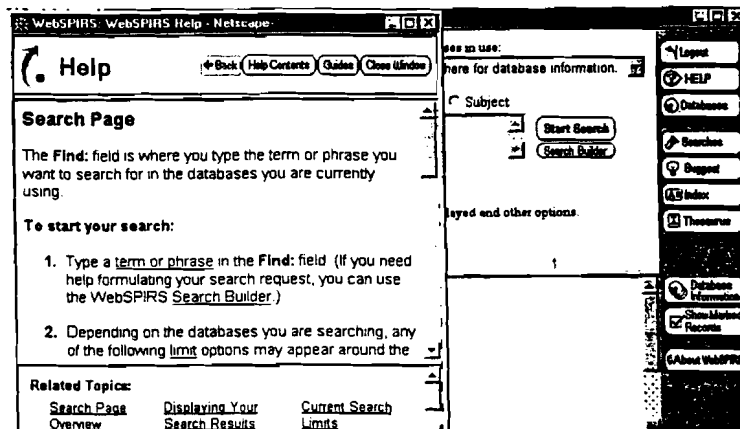


Figure 5. Help section for Find command.

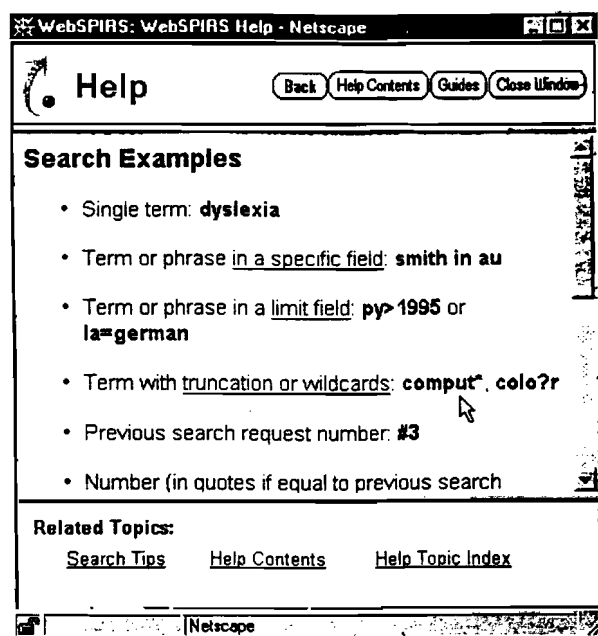


Figure 6. Examples for help information.

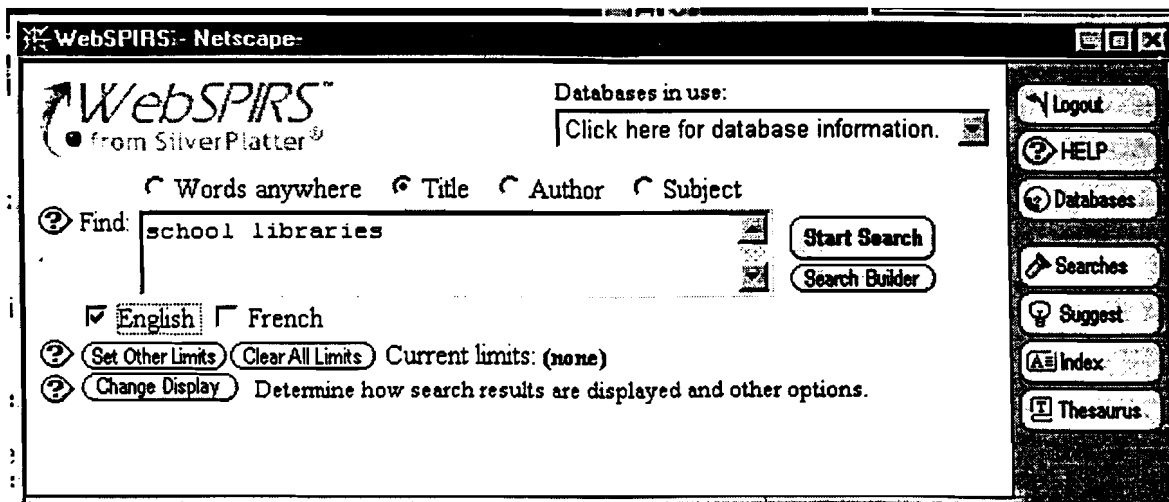


Figure 7. Query form.

understand from the Implementor's Guide, the publication year limit may be placed on the primary search screen by the system implementor. This is a good idea and could be extended to the choice of check boxes and radio buttons.)

### Additional Limits

Additional filters can be selected through pull-down menus after clicking on the Set Additional Limits button,

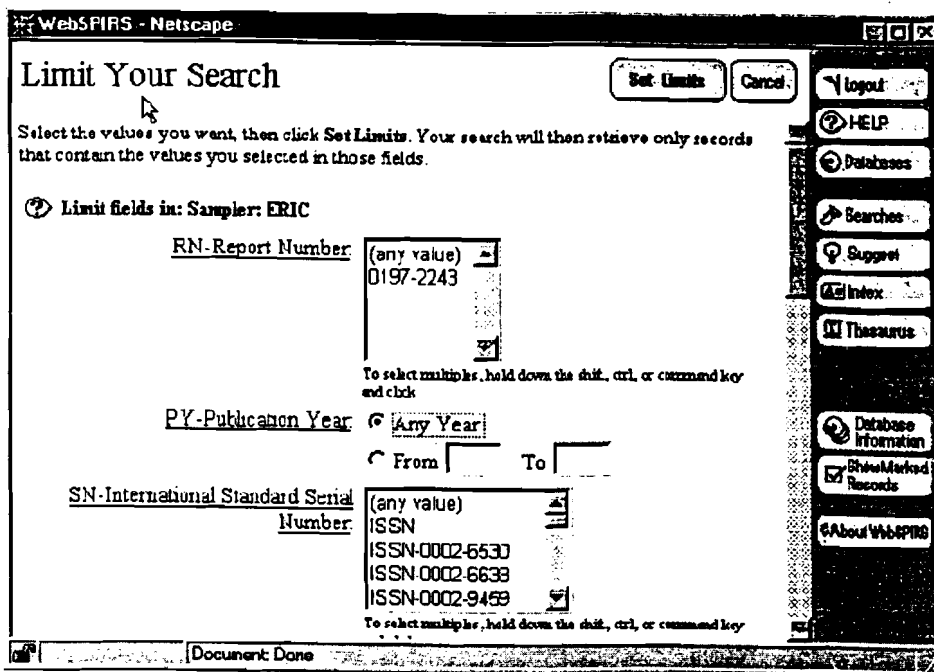


Figure 8. Ill-chosen and ill-implemented scroll down windows for limit options.

but it is a less efficient way to do it for two reasons. Based on my nonscientific observations, many users would not venture to these extra limits. Even if they do, they would see poorly laid out limit options (see Figure 8). All of a sudden, the exquisite screen design we saw before disappears. (In the examples, ignore the values in most of these windows as they were generated from a tiny subset of ERIC records.) The point is the software aspect of limiting a topical (or author) query by additional criteria.

The layout of the limit screen is inadequate. With many types of limit options, it would be essential to put them in a dual column format. Much space is wasted by displaying the name of the limit field to the left of the scroll-down window. These should appear on top of the limit windows and would allow side-by-side accommodation of two types of limits.

The choice of limit options and their sequencing is not appropriate. Neither is the use of codes for limits instead of spelled-out limit values. I cannot envision many users who want to limit a topical search by report number. If they know the report number, they should be able to search for it directly by calling up a number search query form (as it is so nicely done in the Library of Congress Experimental Search System). Simply put, report number is not a limit criterion, not a search filter. Neither is Abstract issue or ISSN for mere mortals. I would be the first to push file producers to include the ISSN in every record and in a consistent format for journals that have an ISSN. It could then make it worth it for the knowledgeable user to look up the ISSN from an on-line

journal index or approach the search with one or more ISSNs in hand. Until then, it is an unnecessary limit option that pushes down the far more important ones such as target audience, clearing house, and document type, which are indeed search filters, especially if they would appear with limit values that users can understand instead of enigmatic codes (see Figure 9). I have been badgering SilverPlatter for this for a decade to no avail. I don't give up.

The only reasonable and well-done limit fields on this long screen are the Target Audience and Language limits in the ERIC database. In the underlying records, all of the limit fields may appear with a code for space-saving reasons. This, however, does not mean that the limit options should use the codes. The programmer should convert these codes into sensible limit terms in the scroll-down windows (as it is done for the Target Audience and the Language limits) and in all the output records.

It is inconsiderate to assume that users would know the meaning of the two-character Clearinghouse codes and Document Type Numbers. I just can't envision users—even if grown up on 90210—who would look for some good 051 from clearinghouse FL instead of some good monographs added by the clearinghouse in charge of Language and Linguistics materials.

The Source File indication with its big window is a waste of precious screen estate: it can be either EJ (for journal articles) or ED (for other resources, like reports and theses) and should appear as a check box right below the query

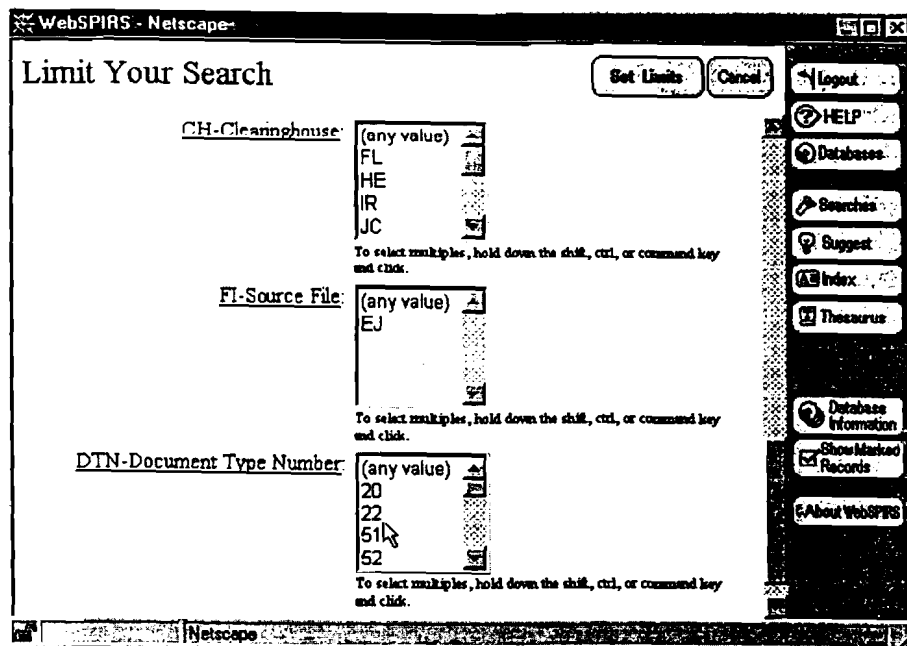


Figure 9. Limit options with enigmatic codes.

template. There are similar problems in many other database implementations of SilverPlatter, and it would not take much effort to correct them.

### Command Mode Searching

Although formally there is no explicit option for rookies and pros, the latter can shoot from the hip by entering a command directly (see Figure 10). The results screen (see Figure 11) smartly echoes the command that is created behind the scene from this query that uses field qualification (restriction) command and language limit through a check box. Recurring users would learn the command language by looking at the command echo box. I mean recurring docent users.

The query log (query history) appears in the lower part of the screen (intentionally not shown in the screen shots until now) with its own function buttons (see Figure 12). It shows the queries entered with their set numbers. These can be marked and combined with the Boolean radio buttons. The NOT operator is not offered as a radio button (which is a good decision given the consequences of this operation if used with subject terms), but in the command mode and under the Search Builder mode, it is available.

It is unusual but very delightful that the current query is on the top of the query log and the older ones scroll out of view (but don't disappear; they can be displayed by scrolling down). It is a minor design issue, but I would shift the

Save History and Load History button pair a little further down to make the set combination part stand out more prominently. These two button labels are highlighted when at least one former query set from the log is marked. The label of the Boolean button is highlighted when at least two sets are marked on the log. This selectivity and context sensitivity is as smart and elegant as the entire interface.

The previous sets can be easily displayed without re-executing the search first. The Remove Checked button helps to clean up the query log, and the Re-type Checked button recalls into the query cell the marked query for editing. It is like the prior query callback function of Dialog-Web. While the query limits can be cleared with the click of a button, the query itself needs manual "sponging" if you change your mind while formulating the query. Executing the query empties the query cell automatically.

### Thesaurus Management

SilverPlatter does not implement the thesaurus for every database, but it does for most of them. Thesaurus browsing is activated by the click of a button. First, you enter the permuted index term list of the thesaurus (see Figure 13), which acts as an antechamber. It is a good idea to let you see in how many subject headings a potential term may occur. This immediately reminds the user that instead of just searching the very broad term *Ability* in ERIC, it may make more

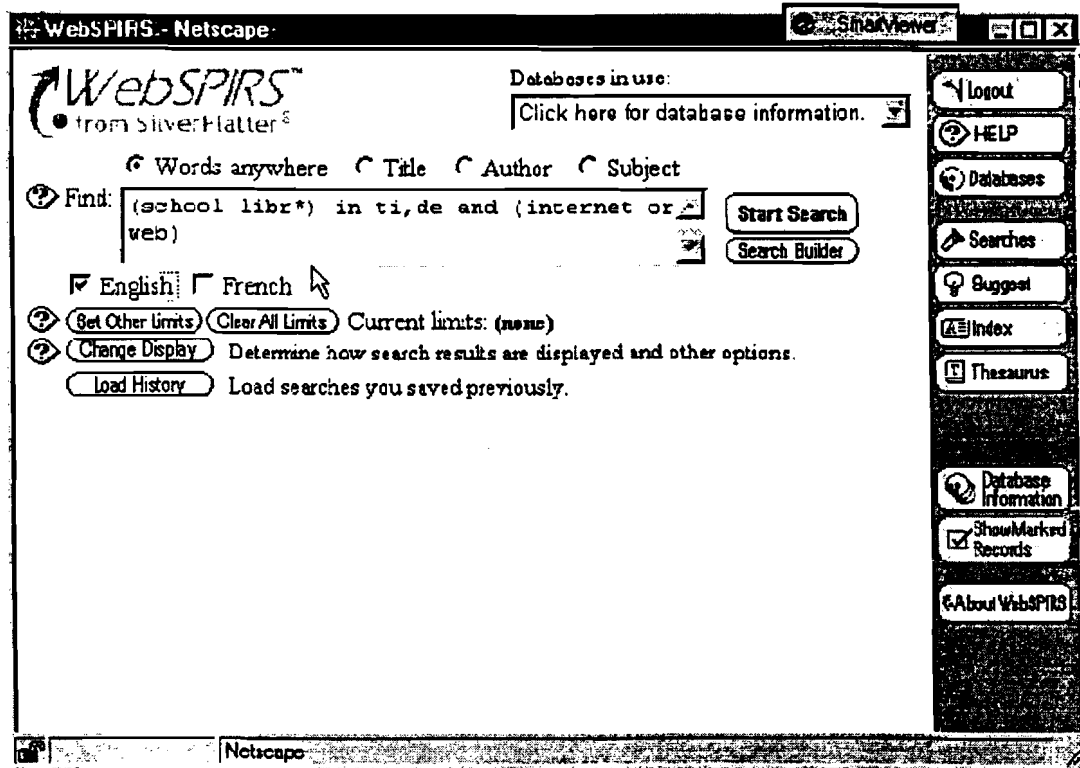


Figure 10. Query in command mode (with a language check box enabled).

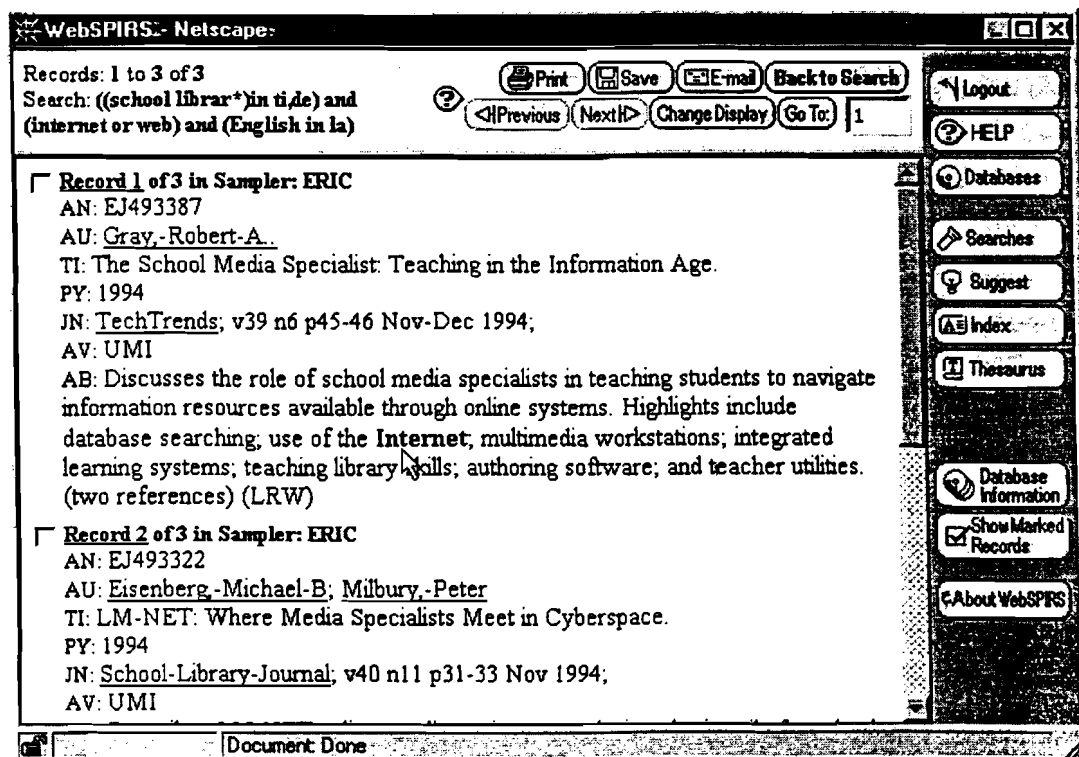


Figure 11. Results showing the behind-the-scenes full command.

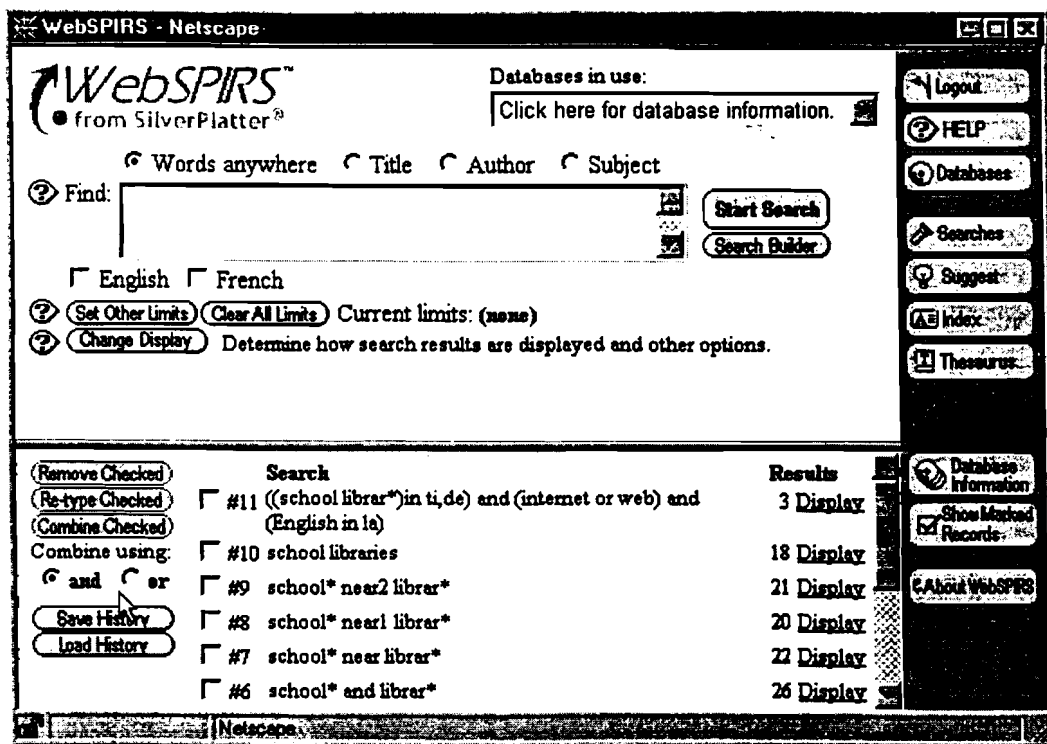


Figure 12. Query log.

sense to go for the more specific terms, such as *Academic Ability* or *Nonverbal Ability*.

It would be even more useful if posting information was provided to indicate in how many records the descriptors occur. Thesaurus entry terms that are not descriptors provide a "see" reference to the preferred descriptor, such as *Mental Ability*, see *Cognitive Ability*. It is a nice touch that you may initiate the search from the nondescriptor term without paying a courtesy visit to the preferred term and bow.

Typing a term or the beginning of a term takes you to the alphabetic list of thesaurus entry terms (see Figure 14). Instead of typing in a new term, you may scroll up or down by five, ten, or twenty terms at one click of the Previous or Next button. One or more terms can be selected for searching directly. These terms are automatically ORed together for the search. This is reasonable as you are more likely to create an OR relationship between these adjacent thesaurus terms than an AND relationship, but it would not hurt to offer the AND operator as it is done in Ovid.

The information about a descriptor term includes a scope note (a definition), the year when the term was introduced as a descriptor (or promoted from an identifier status), and an additional usage note. For example, with the very general term *Libraries*, the usage note recommends the use of more specific terms (see Figure 15). Using all uppercase letters in the definition makes the scope note hard to read. This is an

odd trait in this system that excels in good typographic choice.

It is a minor issue but is against consistency that instead of offering the function button (Return to Permuted Term Index), it reads Cancel, which might make users more hesitant in their next step given its terminating connotation.

When a descriptor has a narrower term, the Explode button is displayed, allowing the user to select a descriptor and all of its narrower descriptors to generate a query by one click. Again, posting information would orient the user to whether this is a smart step or not.

Another button that would allow the user to restrict the search to records where the descriptor is assigned as a major descriptor would be very useful. It can be done using the command language, for example (school libraries in dem), but it is far less obvious and efficient. It can be found out from the excellently structured, dual-panel database guide (see Figure 16), but such a detour detracts from the search.

Broader, narrower, and related terms are listed alphabetically within this grouping (see Figure 17). Posting information would be badly needed to foresee the implication of choosing a narrower or broader term without initiating a search. This is more important considering the sometimes sluggish Internet connection, which makes you think twice before each clicking if it is worth it to take a licking, and because of the limitation that only one term can be selected at a time from the thesaurus.

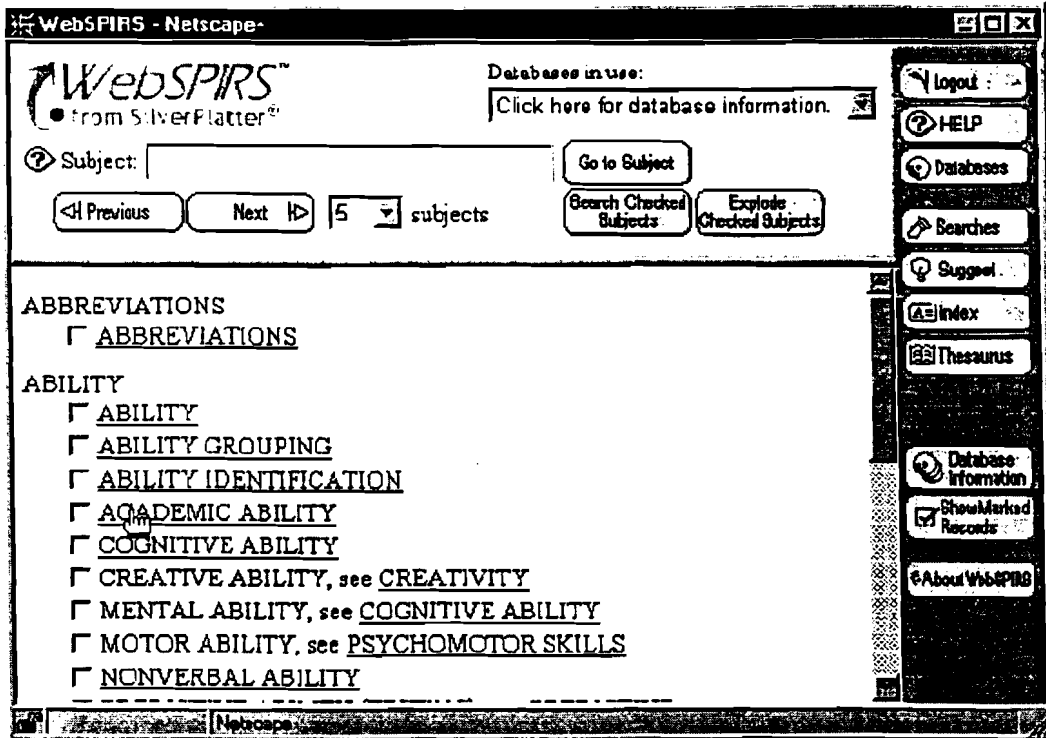


Figure 13. Permuted Term Index of the thesaurus (permindx1).

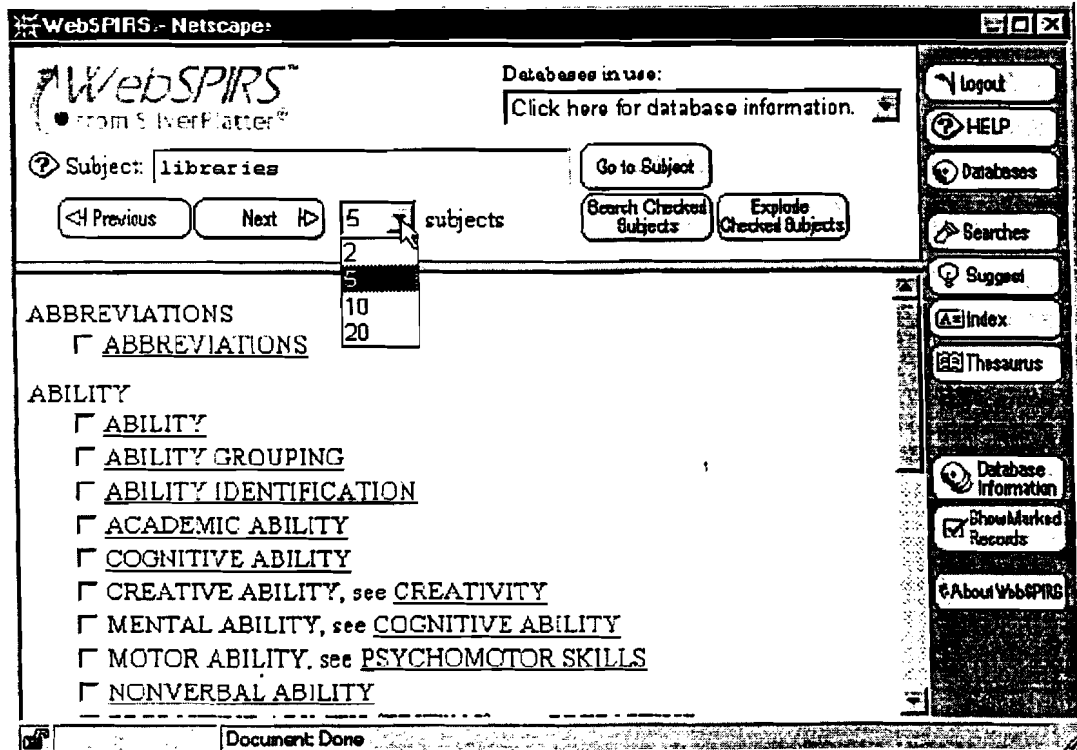


Figure 14. Jumping in the Permuted Term Index of the thesaurus (permindx2).

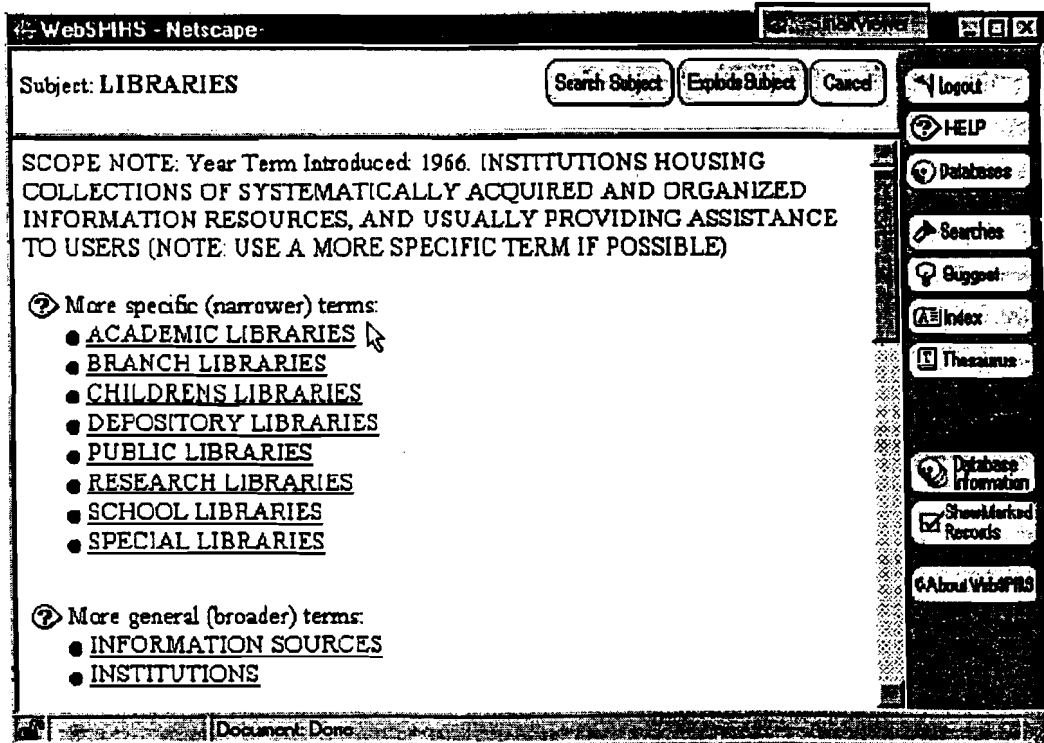


Figure 15. Thesaurus entry—Part I.

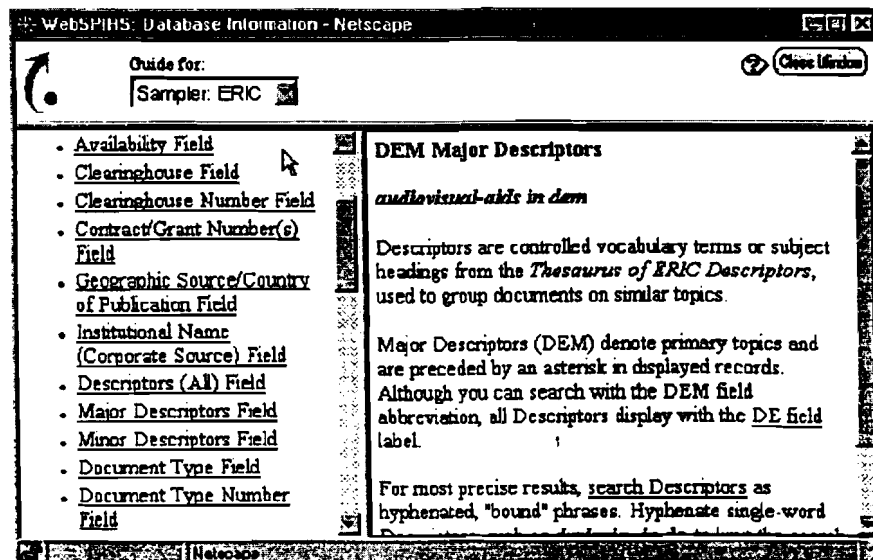


Figure 16. Database guide for ERIC.

### Other Indexes

Index browsing—except for the very nice permuted index list of the thesaurus—has always been SilverPlatter's

weakest point. It has improved since the early DOS days, but it is still far from perfect. The underlying problem is the lack of appropriate field-specific indexes. In the early DOS days, there was only a single browsable index. It is still there in the

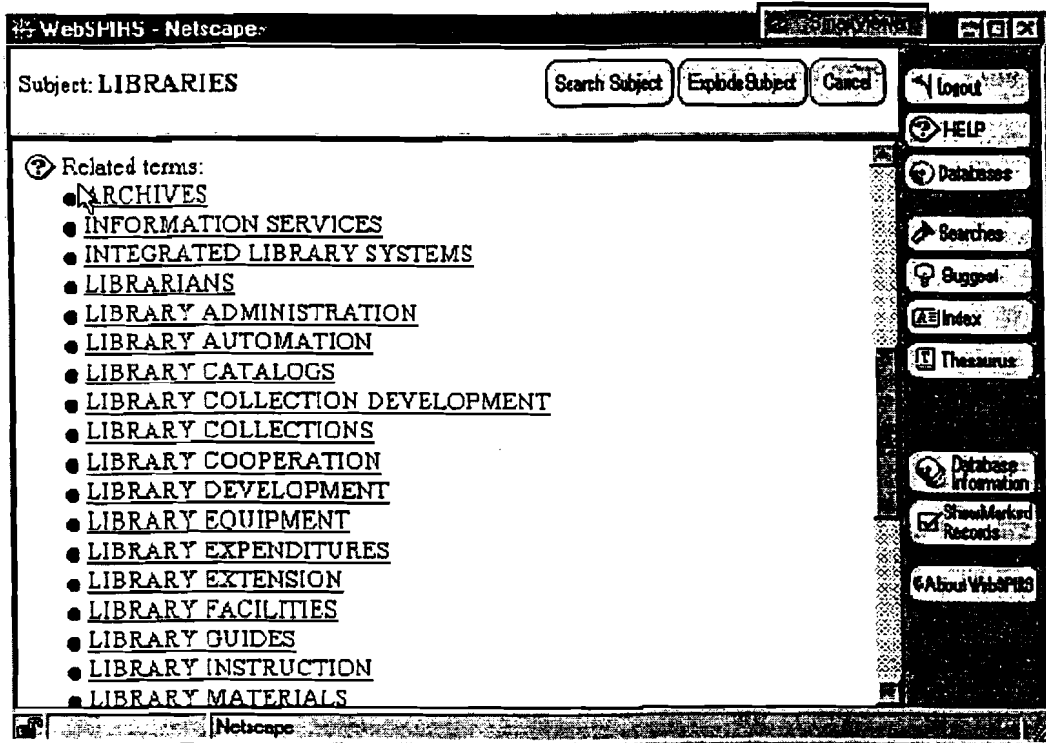


Figure 17. Thesaurus entry—Part II.

most current WebSPIRS 4.0 version as General Index (see Figure 18).

Terms are generated from the title, abstract, descriptor, journal name, author affiliation, and many other fields—without visible distinction of the origin for the user. In such a lump-index, the entry *Child-Development* may be a journal title as well as a descriptor or an identifier. The hyphen between the words of a compound term may give a hint, but the hyphen at the end of a single word is certainly confusing for the casual user and does not make much sense. It is to distinguish a single term as a descriptor (or identifier) from a single term generated from the title or the abstract. Once the librarian explains this to the users or they dig it up from the help file, they may understand the difference that *Seasons* is a term from the abstract and/or the title, and *Seasons-* is a descriptor or identifier or a journal name (see Figure 19).

The excessive use of hyphens to keep parts of compound terms together is further confused by the omission of other special characters, such as the apostrophe or the & sign. The journal name *Library Resources & Technical Services* becomes *Library-Resources—Technical-Services*, which looks like a subject heading with subdivision. Terms like *childrens* or *womans* in the index would be an eyesore even for an undergraduate who would fail before the first round of Spelling Bee prequalifiers. Actually, they are correctly spelled in the records as *children's* and *woman's*, and the weird indexing conventions make them look as incorrect plurals.

Most users don't know what the document type names are, let alone the preferred words (book or monograph?) and the language names (Romanian or Rumanian?), unless they look them up in the help file that interrupts the rhythm of browsing and searching. At least the indexes include posting information, and multiple terms can be selected at the same time. These are then ORed together.

The lump index in DOS and the General Index in the latest Web version are identical for all practical purposes. The big difference is that at least in the Windows/Web versions of the software, some additional indexes are also browsable. I call these field-specific indexes because the entries are generated from clearly identified, specific fields, such as publication year, language, and so on.

The choice of browsable fields is database specific. After looking at a number of databases, it is fair to generalize that the choice of browsable index fields, and their content, is not good enough. They have the same problems as the pull-down menus of limit fields. Author name, journal name, and author affiliation are data fields that are often used for searching, and they should be made browsable in their own field-specific indexes to find the inaccuracies, inconsistencies, and other idiosyncrasies of data entry, and to make better searches.

Given the poor authority control of many databases, searching without browsing is like buying apples from a cart without touching and looking at them. The availability of a large number of field-specific indexes (see Figure 20) with

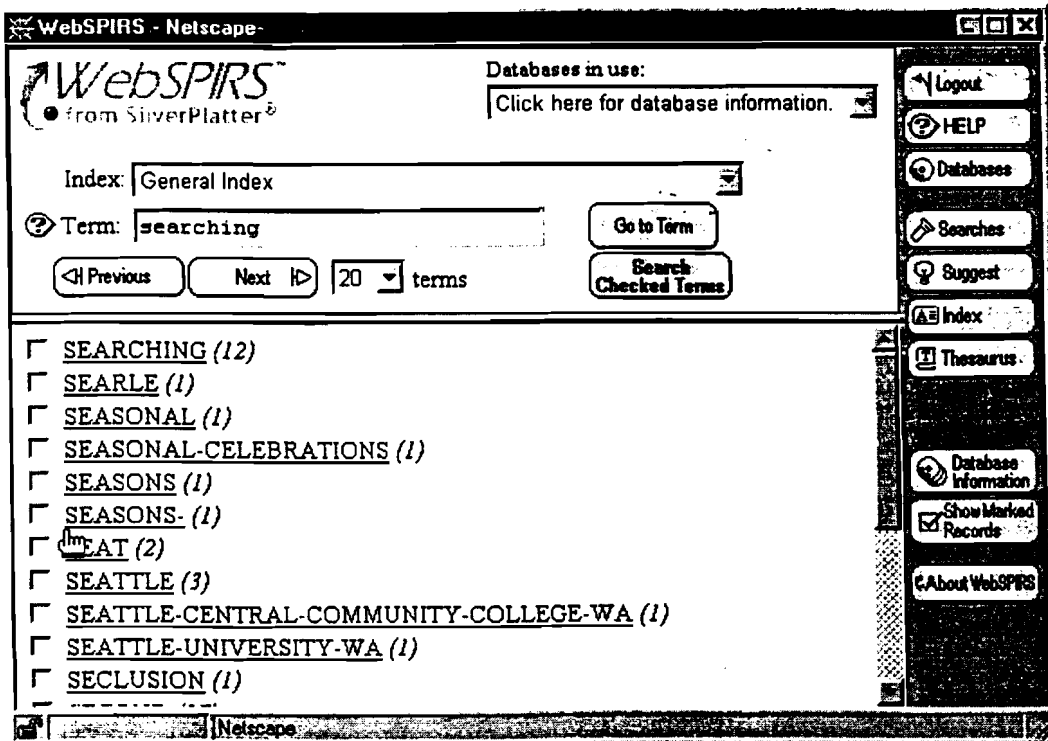


Figure 18. General Index—What is in a hyphen.

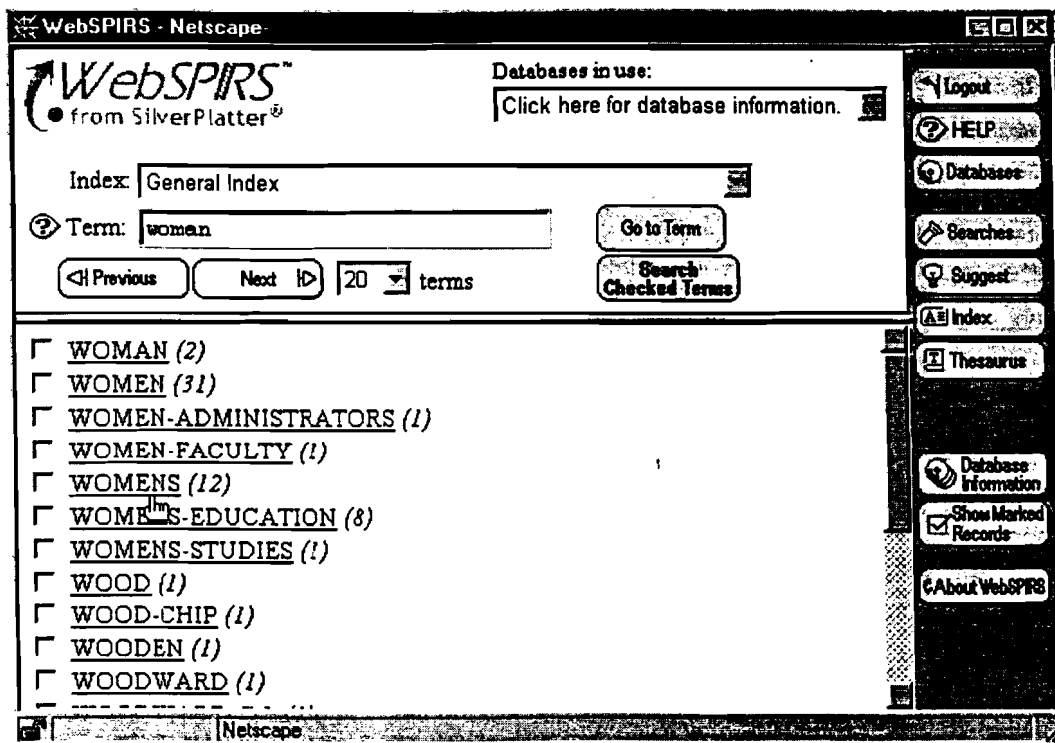


Figure 19. Is that a bunch of misspelled plurals in your index, or did women just put a spell on you?

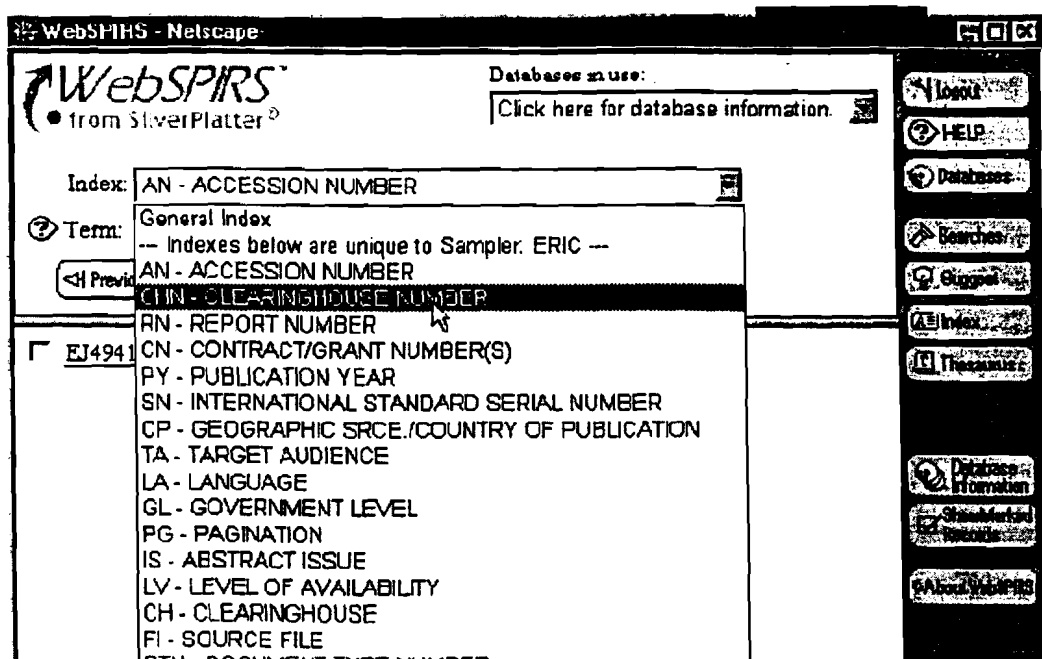


Figure 20. Choice of browsable indexes.

little relevance does not compensate for the lack of essential ones. The software has the capabilities of doing word or phrase indexing or both. It is a human decision which fields are used to create field-specific indexes. It seems as though SilverPlatter made a pledge in the mid-1980s that some of the data elements that begged for browsing were not to be indexed separately to create their own field-specific indexes, and it has adhered to this pledge. Data file producers may be happy with it as the deficiencies in these fields don't become immediately obvious when browsing in the General Index, but it is a great disservice to the users. There is hope, however, as in the PsycLit database, for example; the journal name data element has its field-specific index.

As opposed to author or journal names, or author affiliations, the user is unlikely to browse report numbers, accession numbers, or ISSNs. Still, these are among the field-specific indexes. The user either knows them or doesn't. Browsing these numbers is not like impulse buying at the cheese shelf. Looking at the numbers does not bring inspiration to pick one over the other (see Figure 21). Even some of the relevant field-specific indexes are rendered useless by displaying codes in the index instead of the spelled-out name of the clearinghouse responsible for a subject area (such as Language and Linguistics instead of FL, or Information and Technology instead of IR).

### Truncation and Masking

WebSPIRS does not use automatic truncation but offers the user the option of specifying limited or unlimited

truncation. For example, the query term *art?* would retrieve *art* or *arts*. The query term *art\** retrieves *art*, *arts*, *artist*, *artists*, and *artistical*, as well as *artificial* and *artifact* or *artifacts*. Masking can also be limited to a single character usually needed for British/American differences (*organization* versus *organisation*), or irregular plurals such as *wom?n*. Alternatively, using multiple masking symbols, up to the specified number of characters may be specified such as *lab???r* to retrieve both *labor* and *labour*.

### Field-Specific Searching

Field-specific searching is possible—to a limited extent—through the radio buttons of the search template, and fully through the command language. The former allows the restriction of a query to the title, author, subject, or keyword field by clicking on the appropriate radio button. It could be better to use these as check boxes to allow the user to restrict the search alternatively to the title or the subject field, for example. This means that instead of single-field qualification at a time, the query template could offer multiple-field restriction. Additionally, the abstract field should be offered as a qualifier, especially in the case of full-text databases where the occurrence of search terms in the subject, title, and abstract field is more likely to yield relevant results than their occurrence in the keyword index that may include the full text depending on the database (see Figure 22).

There are more options for field qualification of a query when you click on the Search Builder button that brings up

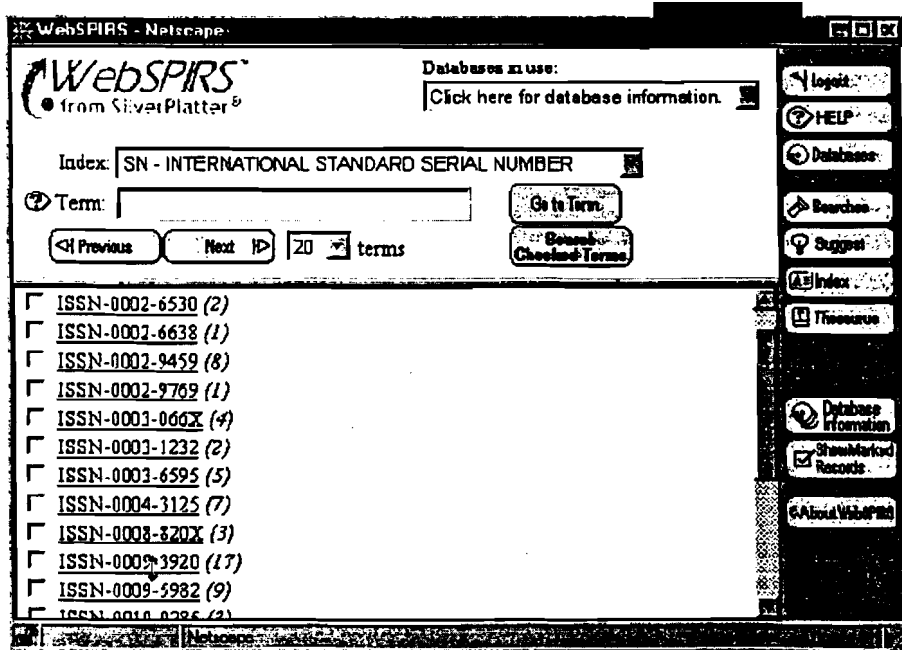


Figure 21. ISSN index content.

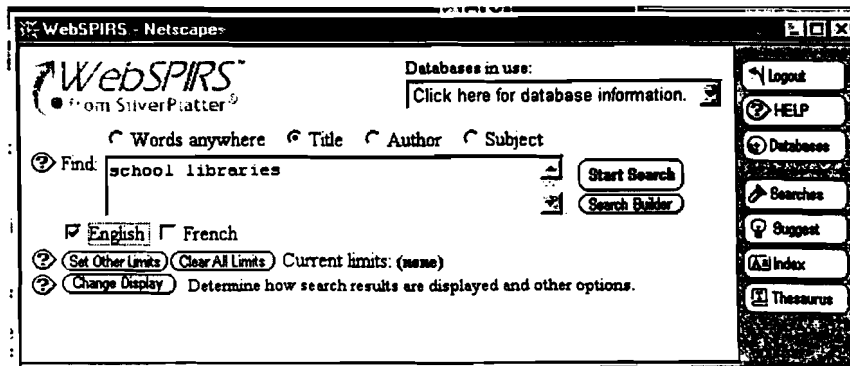


Figure 22. Field-specific searching through radio buttons.

the list of fields available on two window panes (see Figure 23). The terms entered in the query cells can be restricted to one or more of the fields listed and can be combined with one of the three Boolean operators. It is appropriate to offer the NOT operator here because the user may wish to exclude records of, say, certain document types.

The choice of fields is very good (see Figure 24). It includes the following fields: citation, personal author, title, corporate source, sponsoring agency, journal name, descriptive note, document type, descriptors (distinguishing major and minor), identifiers (again, distinguishing major and minor), and abstract. The journal name field is referred to as

journal citation and may make it difficult for the user to distinguish it from the citation field. Renaming this field Journal Name would solve the problem.

Multiple fields can be selected, so one of the query terms can be restricted to the title and abstract fields, and the other one to the major descriptor or major identifier field. The help information above the query cell does not make it clear that pressing the Shift key will select all the fields between the two picked by the user, whereas the Ctrl key would select just the ones that are under the cursor when pressing the key. For computer savvy users, this may be obvious, but not for the casual patrons. PC users looking for the Command key

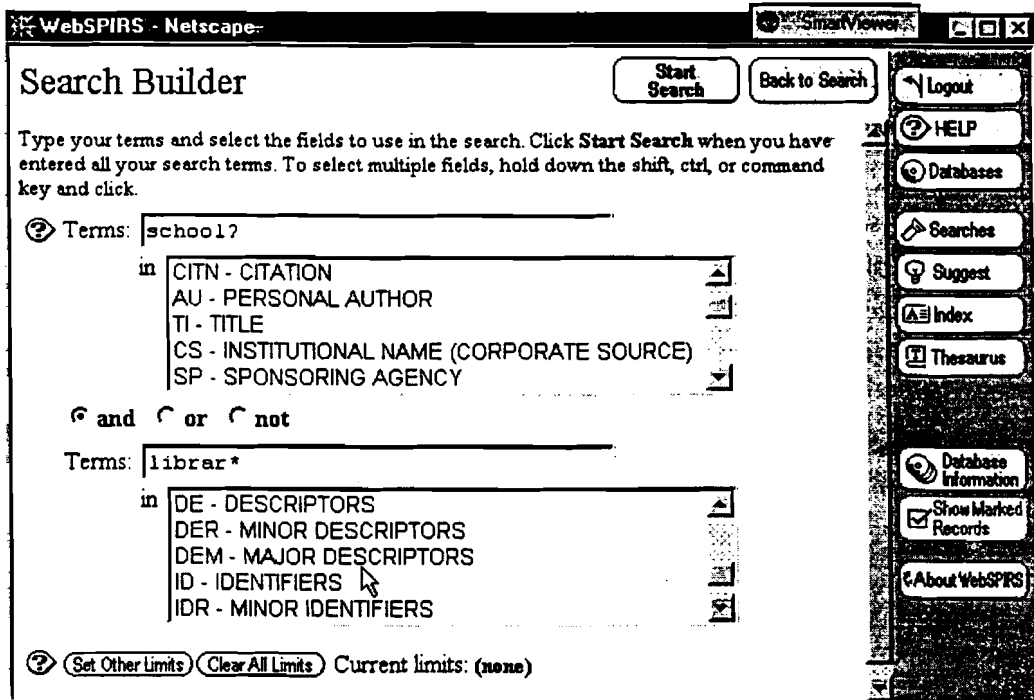


Figure 23. Field specification template with dual panes (fldqual1).

will be baffled as only the Mac computers have it. Of course, there is a limit to how much help you can squeeze onto this screen, but some good compromise could be made.

The command mode offers the option to restrict the search to several fields in a single search statement such as school libraries in ti,de. In both modes, it is also possible to combine truncation and field qualification or masking—a feature that has not been possible in the DOS versions.

It would enhance the choices for field-specific searching on the main search screen if there were two rows of check boxes: one for subject fields and another for nonsubject fields or not directly subject fields, such as author name, journal name, and affiliation name.

### Proximity and Positional Operators

WebSPIRS has enhanced its capabilities for positional operations in this release. It always had proximity operators to let the user specify the distance between two or more words in a query, such as *drug NEAR3* being used to retrieve *use of alcohol and drug; drug use; drug, tobacco and alcohol use*, that is, irrespective of the order of the words. The number after the NEAR operator indicates the maximum distance of words. Although the documentation does not mention it, NEAR in itself (without a number) seems to be equivalent to NEAR6 proximity. The WITH operator specifies that the connected words be in the same field in any order. The field need not be specified so it is different from

the query (*school?* and *librar\**) in ti. For the full-text databases, it would be helpful to have an operator to limit the query terms to the same sentence.

The new ADJ operator prescribes not only that two or more words be next to each other but also that they be in the specified order. For example, *computer ADJ assisted ADJ instruction* will retrieve the records that mention *computer-assisted instruction* but not the record with the sentence "a computer assisted the school in planning the instruction program." A variation of the ADJ operator with a number such as ADJ2 would be useful to retain word order but not direct adjacency. For example, *school ADJ2 librar\** would retrieve *school and public libraries*, but not *library school*.

The size of the retrieved set is reduced by using more and more restrictive queries as to the distance and order of the words. It is an asset of the software that if a compound term is entered such as *school library* without any proximity or positional operators, the system assumes NEAR6 proximity. This is a far better solution than DIALOG's approach, which assumes that if there is a space between two words, the phrase must be an exact descriptor.

### Sort Options

Records can be sorted in ascending or descending order by a single criterion. It would be nicer to offer two criteria (such as journal name and within journal name by year in descending order), but the single field sort is quick and easy

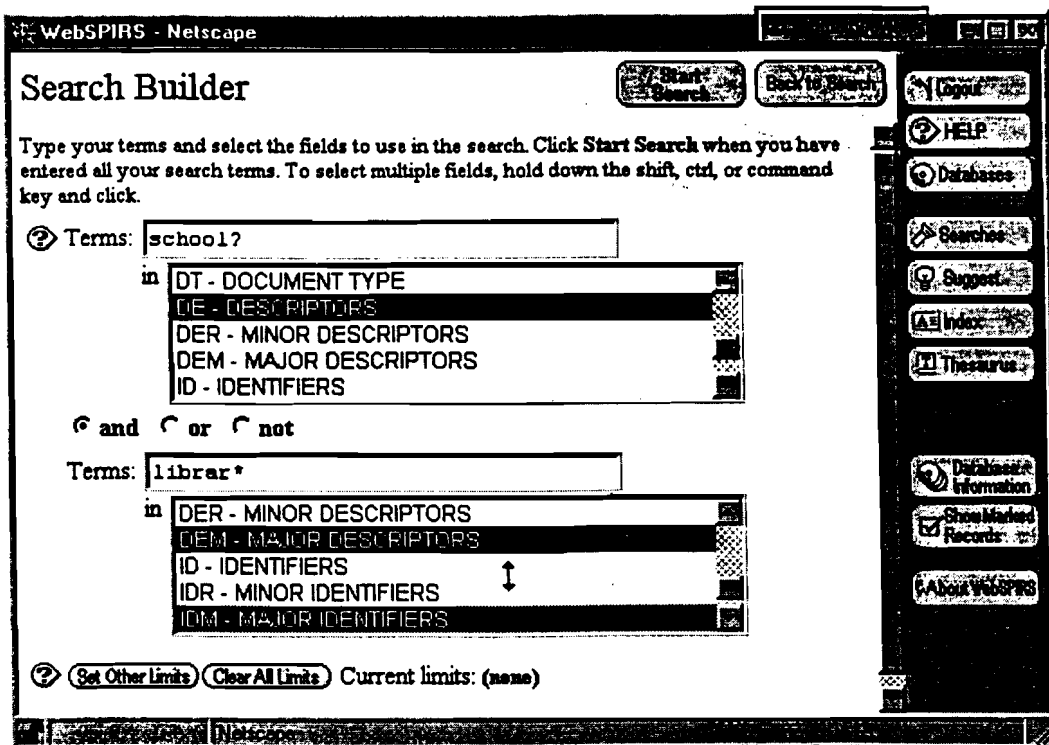


Figure 24. Choice of fields for field qualification.

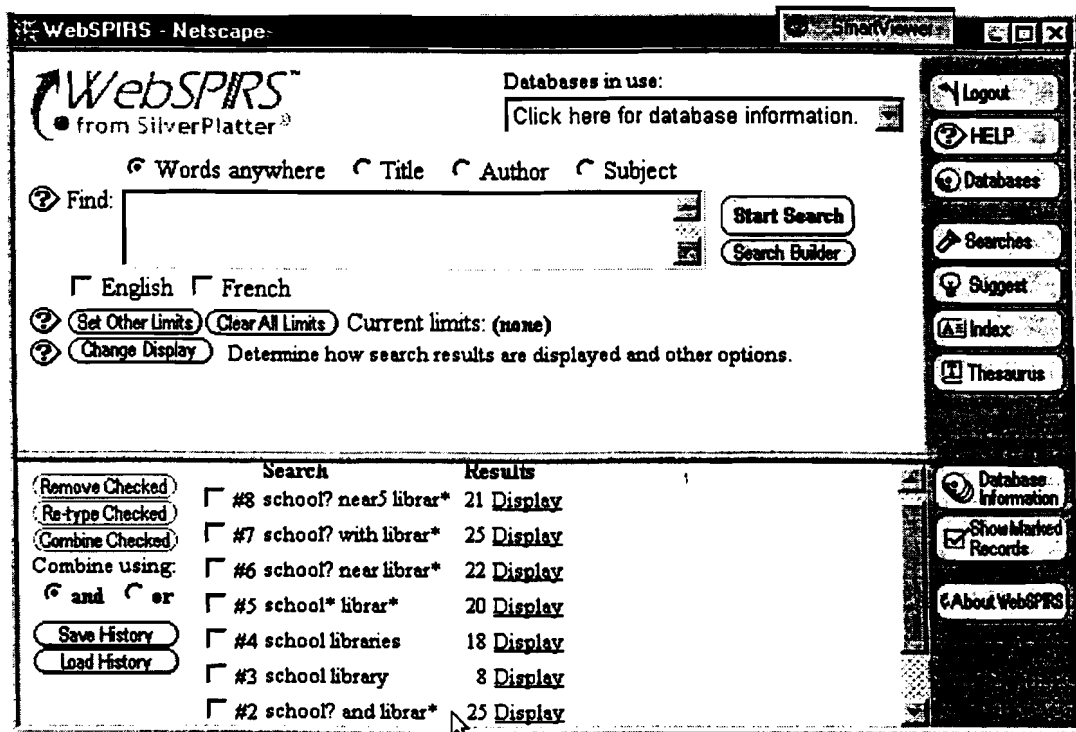


Figure 25. Proximity and positional operators.

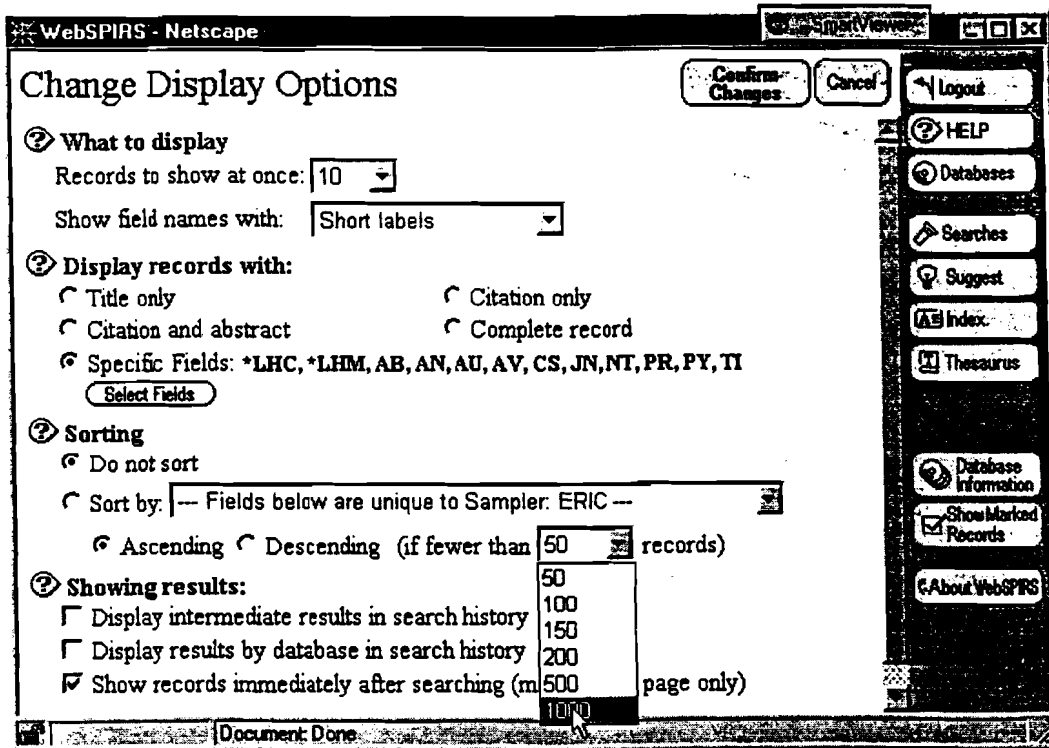


Figure 26. Sort options.

and the choice of sort criteria is very good. The fields that can be used as sort criteria are displayed in a pop-up menu (see Figure 26). I am delighted to see that SilverPlatter increased the puny 100-record limit to 1,000 for sorting.

### Output Content

The system administrator defines the content of the default output format that would appear automatically after the search is done, but the user may change it very easily. There is no predefined hit-list format that would show in a compact list of one-liner entries, that is, some key elements of each record retrieved (such as forty characters from the title, fifteen characters from the author's name, the publication year, and fifteen characters from the journal name). However, there are a number of predefined output formats that include title only, citation only, citation and abstract, or complete record. These can be selected by clicking on the appropriate radio button (see Figure 27). It would be useful to have a predefined radio button option to show fields that include the query term(s). It is possible to specify this in the user-defined format, but it would be more convenient as a predefined option.

In addition, the user may select any fields of the database from a pop-up list of field names. After changing the

selection of fields in the user-defined format, the field tags are listed (see Figure 28).

### Marking Records

Records displayed can be marked to pick the best ones from a large set and to display, print, or download in another format. Records that are marked across several screens are retained (as opposed to what occurs in some other systems, including DIALOG). Assuming that the result of a search includes more relevant than irrelevant records, it could be more efficient to mark records for exclusion from the set, but it is a rare feature used by a few systems, including EBSCO. With the proliferation of the Web, positive marking is certainly more common because from the avalanche of "hits" that most search engines throw at you, it is easier to pick the promising ones by marking them (see Figure 29).

### Output Actions

The user may define how many items should appear on one "page." It is tempting to specify the highest value (100), but it may take too long for the software to build such a long page before displaying anything. The default value (10) may make it necessary to click on the Next button more often

WebSPIRS - Netscape: [X] [O] [Z]

## Change Display Options

**What to display**

Records to show at once:

Show field names with:

**Display records with:**

Title only                       Citation only  
 Citation and abstract            Complete record  
 Specific Fields: +LHC, +LHM, AB, AN, AU, AV, CS, JN, NT, PR, PY, TI

**Sorting**

Do not sort  
 Sort by: -- Fields below are unique to Sampler: ERIC --  
 Ascending    Descending (if fewer than  records)

**Showing results:**

Display intermediate results in search history  
 Display results by database in search history  
 Show records immediately after searching (main search page only)

Show Archived Records

Netscape

Figure 27. Predefined output options with radio buttons.

WebSPIRS - Netscape: [X] [O] [Z]

## Select Fields

Select the fields you want to use, then click **Select Fields**. Click **Cancel** to return without making a selection.

**Fields from: Sampler: ERIC**

\*H - Fields With Hits  
 CITN - CITATION (AN, AU, TI, CS, PY, JN, AV, NT, PR)  
 AN - ACCESSION NUMBER  
 CHN - CLEARINGHOUSE NUMBER  
 AU - PERSONAL AUTHOR  
 TI - TITLE  
 CS - INSTITUTIONAL NAME (CORPORATE SOURCE)  
 SP - SPONSORING AGENCY  
 RN - REPORT NUMBER  
 CN - CONTRACT/GRANT NUMBER(S)  
 PY - PUBLICATION YEAR  
 JN - JOURNAL CITATION  
 \*LHM - Library Holdings Message  
 \*LHC - Call Number  
 SN - INTERNATIONAL STANDARD SERIAL NUMBER

Show Archived Records

Netscape

Figure 28. User-defined field selection for output.

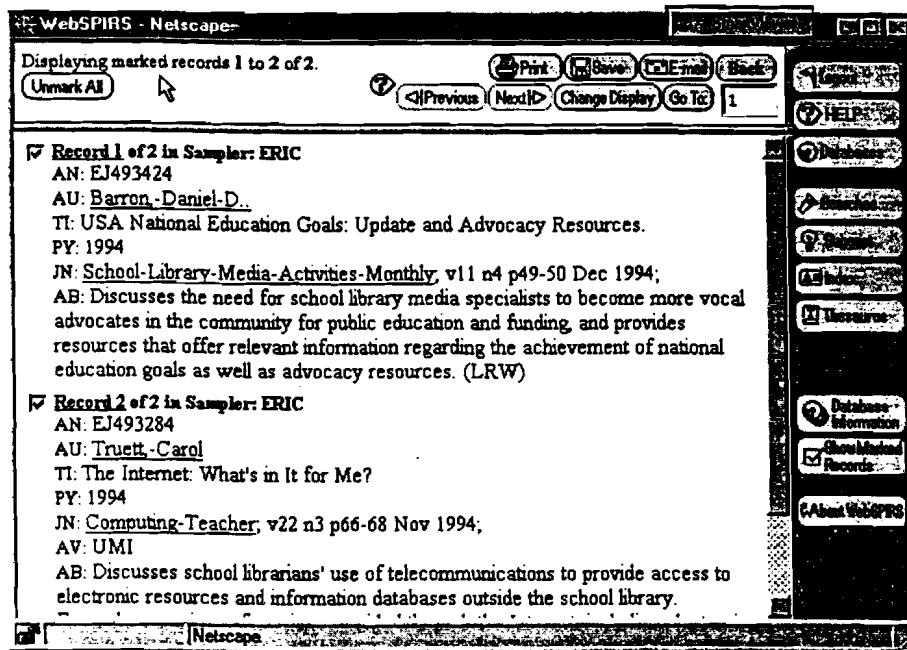
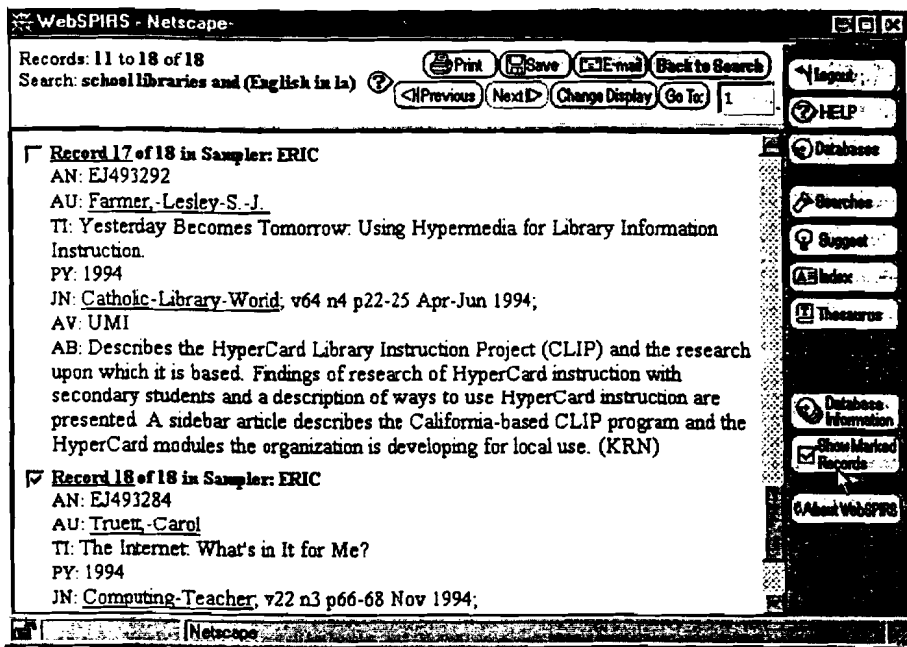


Figure 29. Marked records from different pages.

than you wish and to start another transaction. Users can set their own preferences ranging from 5 to 100, although this remains valid only for the current session.

Records can be saved in text format, but not in HTML format. (The page itself can be saved as HTML using the browser itself, but you must make sure that the right frame is active, otherwise you save the function buttons frame.) Printing and saving pure bibliographic records is possible

with or without the search log. Other parameters for the printing and saving actions (that are almost identical) can also be specified (see Figure 30). Saving in HTML format would be useful in case records are to be included in a Web bibliography (observing the copyright rules, of course, if applicable).

E-mailing results in plain text format is possible, and the output can be nicely customized (see Figure 31). It is a

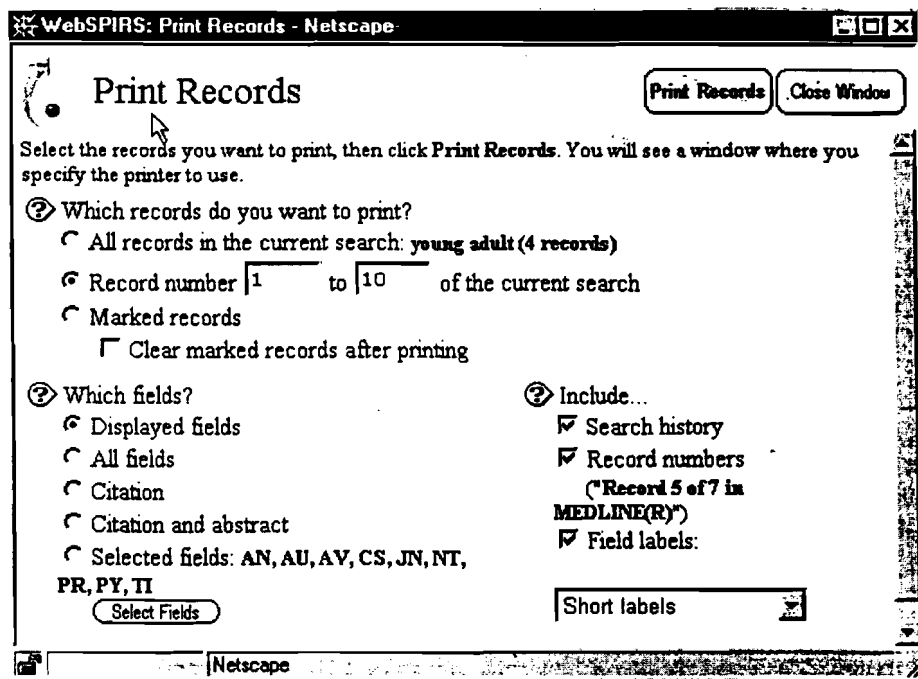


Figure 30. Print and save options.

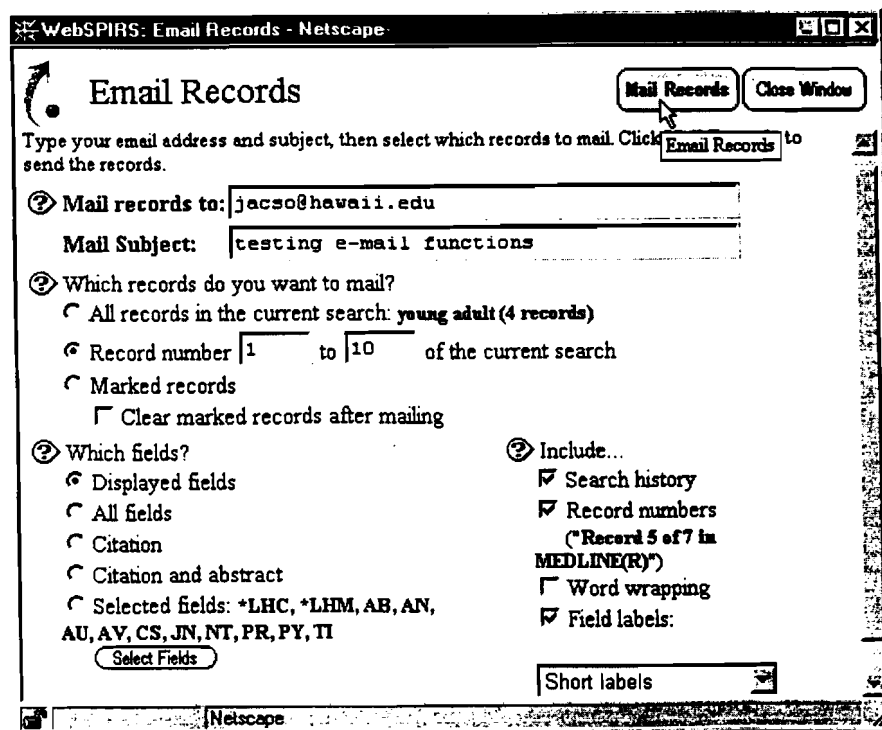


Figure 31. E-mail options.

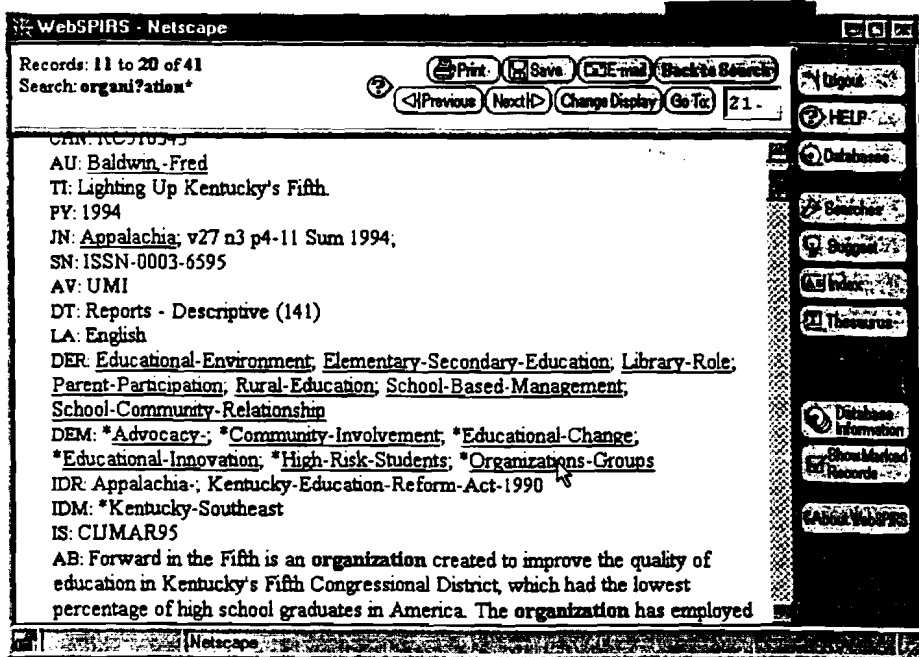


Figure 32. Hot-linked data elements.

comforting feature that the successful sending of the mail is confirmed right on the screen.

Hot searching from the records displayed is particularly attractive as such links make the Web a web. Indeed, SilverPlatter had excelled in this area well before the birth of the World Wide Web. Data elements that can be used as hot links are highlighted and underlined and appear in blue according to the Web hot-linking convention (see Figure 32). The choice of hotlinks is very good; it includes the author name, the journal name, and the descriptor fields. I could not check whether URLs embedded in bibliographic records could be hot-linked, but it is very likely to be possible.

### Advanced Features

SilverPlatter has two features that are rarely available in on-line systems. One is mapping of terms from the user's query into controlled vocabulary terms, and the other is multiple database searching. The former is offered by Ovid, the latter by DIALOG and Datastar. The Experimental Search System of the Library of Congress uses an interesting mapping technique for presenting the search results and for showing a list of Library of Congress Subject Heading terms in the result set.

The purpose of term mapping (known as Suggest feature in WebSPIRS) is to find controlled vocabulary subject terms based on the words the user has entered in the query. This is

particularly useful when the thesaurus is not implemented with the database, or when the thesaurus does not include the term at least as an entry that provides a "see" reference to the preferred term. For example, the concept of quitting smoking is not a term in this thesaurus either, but the Suggest feature lists Smoking Cessation as the second term when the user enters quit smoking and asks for suggestions (see Figure 33).

The exact mapping algorithm is a closely held secret of the developers, but from the results, it can be guessed to some extent. It seems that the program looks at the descriptors assigned to the records that include, for example, the words *puerperal* and *depression* and selects the ten most often appearing ones as suggested terms. It is not known what priority is given to records; are those preferred where the two words appear next to each other in the given order? or the ones that include the words in the title in any order. In 1997, I compared the term-mapping features of Ovid and WinSPIRS, and the former proved to be far better. The Suggest feature was then withdrawn and surfaced again in WebSPIRS 4.0. In testing the new version of the Suggest feature, I discovered that the mapping algorithm still needs fine tuning. In the example above, *Adulthood* is the number 1 suggested term. This is obviously a very generic term that makes it a useless suggestion. This can be corrected easily by removing the Age Group terms that are assigned to every record and too often become ranked first. There are other simple changes that could help. The easiest would be to see if the stems of the words of the query appear anywhere in the

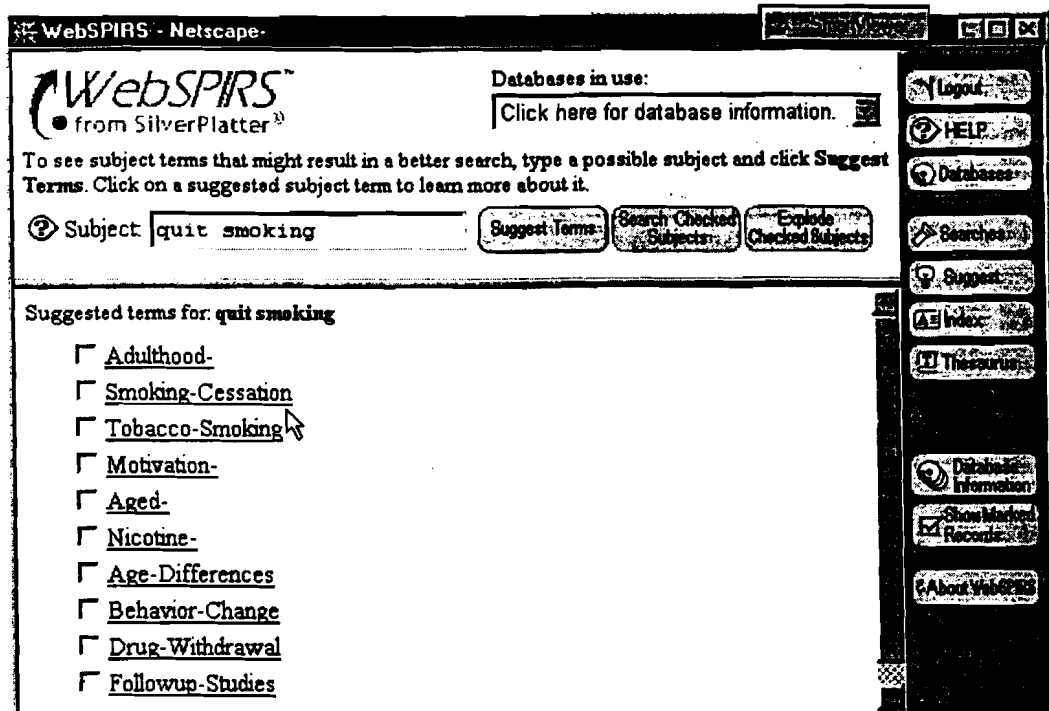


Figure 33. Mapping user terms into descriptors.

thesaurus and offer that term as a possible search term. Without this, the query *libraries in schools* yields no terms at all (see Figure 34) even though *school libraries* is a descriptor. The query *school library* fares better because the descriptor term in plural format shows up third on the list of ten suggested terms (see Figure 35).

Multiple database searching offers great convenience for the user. Although the test search was done in a group of small sampler databases, it is obvious that running the query across a dozen databases and ranking them by decreasing number of hits is an excellent start to find the most promising databases about, for example, rheumatoid arthritis (see Figure 36).

After this very useful, fast, and easy starter, you may choose two or more databases for searching. As long as you don't want to set limits or use the Set Builder function, multiple database queries remain efficient. The deficiencies of the limit functions discussed above multiply the problem in cross-database searching. You have more limit fields of highly questionable use (such as the CODEN field) and cryptic codes (such as the EMTAG code), and the menu-based field-qualification search becomes unwieldy owing to the convoluted and lengthy screens that list the common and the different searchable fields (see Figures 37 and 38).

Most of the deficiencies could be easily corrected by having an experienced user review each database for the really relevant limit fields and assess in what priority they should

be listed. Replacing the coded values with spelled-out terms that are useful for casual and even experienced users could be done by a junior programmer who reads the database user manual. Field-specific indexes could be added with minimal efforts. Deduplication is a more complicated task but not for those who can design such an overall attractive and powerful user interface and search engine. Because of the power and grace of this innovative JavaScript version, its blemishes should not take away from its appeal. The changes suggested could be proposed for the other versions and would have a ripple effect on the client and the desktop Windows versions as well.

### About the Author

*Péter Jacsó is an associate professor in the University of Hawaii Department of Information and Computer Sciences' Library and Information Science Program. He teaches courses on database searching, database design, abstracting and indexing, and other information technology related courses. He has written for scholarly journals and has regular columns in Database, Information Today, and Computers in Libraries. He won the 1998 Louis Shores/Oryx Press Award from ALA's Reference and User Services Association for his discerning database reviews, as well as the 1998 ALISE/Pratt-Severn Faculty Innovation Award.*

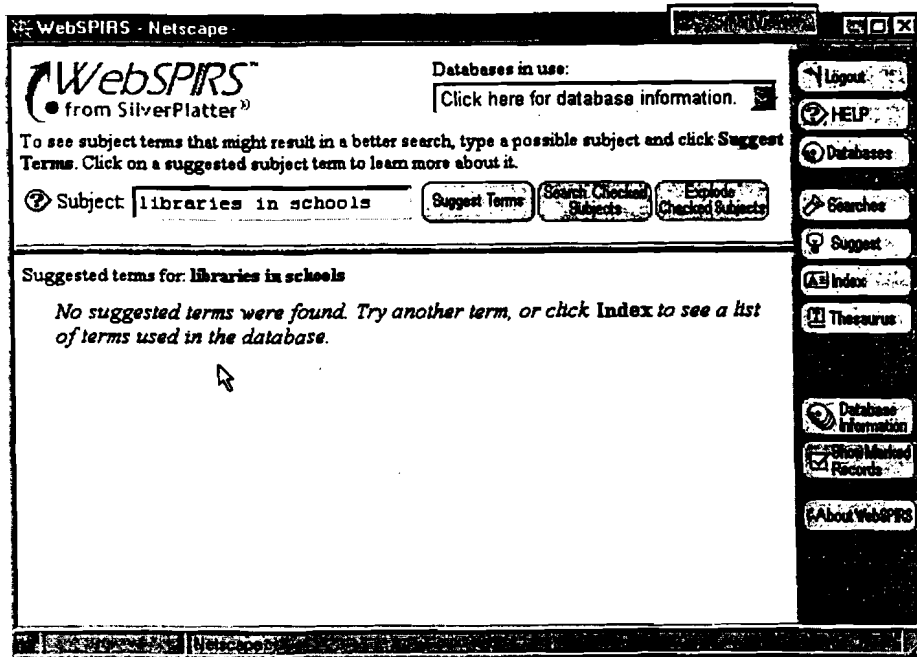


Figure 34. Unsuccessful mapping.

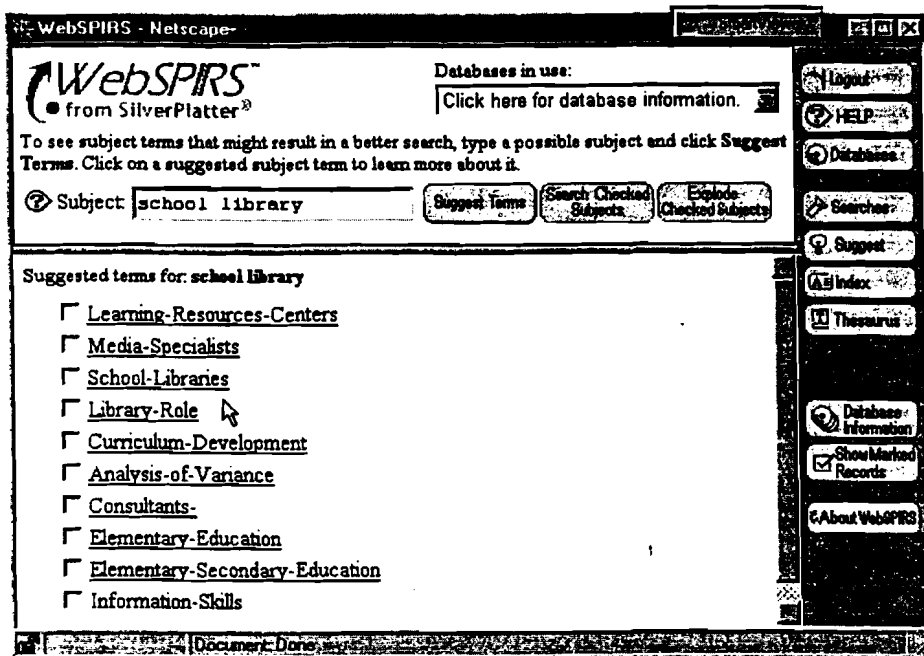


Figure 35. Somewhat successful mapping.

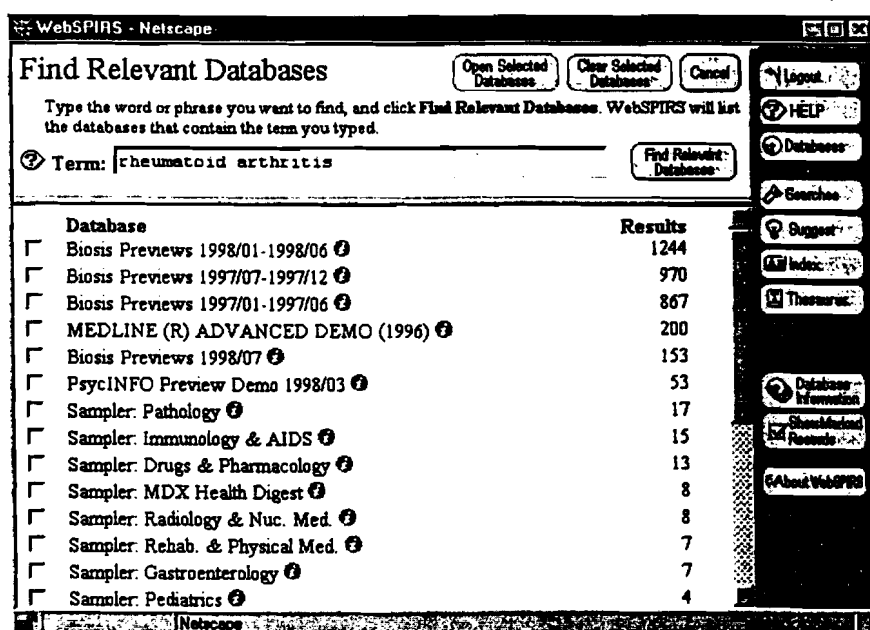


Figure 36. Finding the most promising databases for a subject.

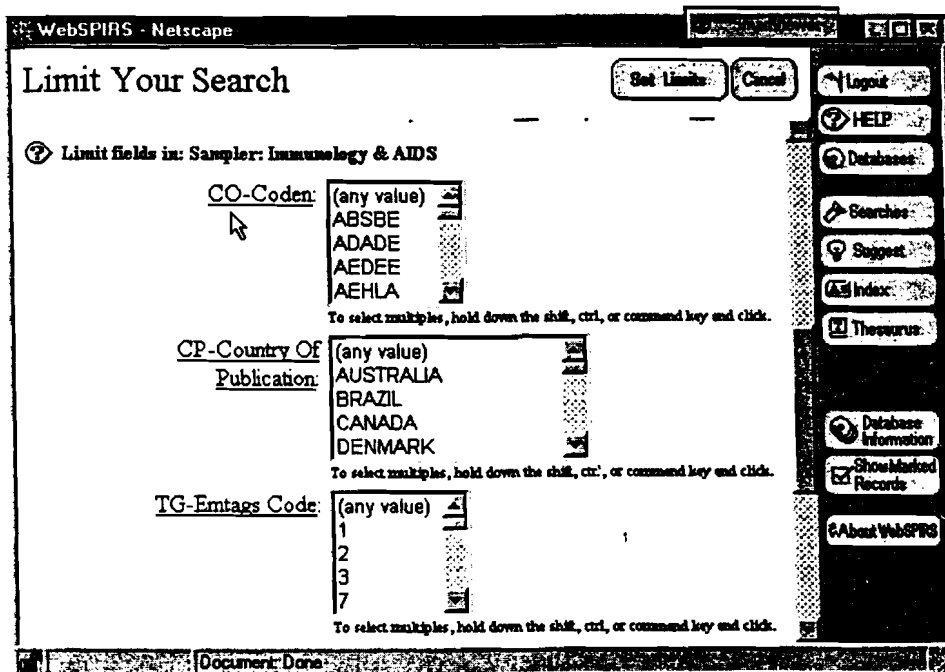


Figure 37. More of the questionable limit fields.

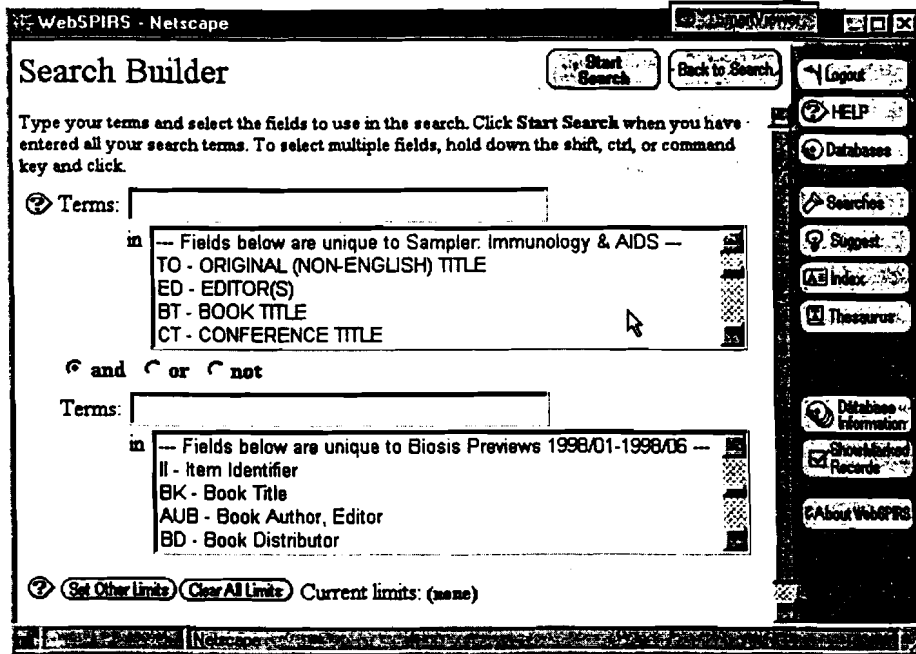


Figure 38. Lengthy window for field qualification.