Qddb User's Guide

Version 2

Herrin Software Development, Inc.

2001

Copyright (C) 1997-2001 Herrin Software Development, Inc.
All rights reserved.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Qddb is powerful database software. Using *Qddb*, you can quickly and easily create database applications. For example, using *nxqddb*(a program distributed with *Qddb*) you can create a complete database application in less than five minutes. Larger applications can also be developed using Fx, a powerful Tcl/Tk toolkit. If you're looking for a way to manage database information over the web through a CGI interface, consider using Hx — another toolkit that comes with the *Qddb* distribution.

This manual is for users that want to learn how to use *nxqddb*. In particular, we describe the steps to create a *Qddb* database and then delve into how to manage that database using the *nxqddb* program. If you would like to learn how to program with *Qddb*'s interfaces(i.e. Fx and Hx), you should reference the Programming Guide to *Qddb*.

Once you take the time to understand *Qddb*, you'll find it to be a rewarding, time-saving and productive tool. You'll be surprised at just how easy it is to create your own applications — personal, or commercial!

1.1 Why *Qddb*?

Qddb was started back in 1989 as a project to manage student records at the University of Kentucky. Since this time, *Qddb* has been significantly improved with new features, a few of which include:

- *Fx* — a powerful Tcl/Tk graphical user interface library that you can use in your own programs to customize your application's interface.

- *nxqddb* — an *Fx*-based program for managing database information

- *Hx* — an easy to use *Tcl* library for developing scripts for CGI online databases

- *QddbScript* — a very simple language (like HTML) that enables users to easily design the graphical user interfaces to their Qddb databases. A quick and easy, complete template is automatically generated to reduce the amount of typing required.
- *Report generator* — Produces listings, e-mail, postcards, letters, fancy barcharts, and graphs. Barcharts and graphs can be built using a plethora of time summary configurations, ranges, and multiple nested searches.
- *Import CSV* — Import flat CSV tables directly into a Qddb database from the convenient user interface.
- *Export* – Export reports and search results in a variety of formats for convenient import into other popular applications.
- *Postscript printing* - print out long/wide reports using a very configurable format on any postscript printer. Straight ASCII printers are also supported.
- text based tools — We have maintained the original text based tools.

Does it work? You bet! The underlying database technology makes creating and using databases easy, especially when you use the *Fx*-based *nxqddb* program and the new QddbScript.

1.2 Minimum requirements

Currently, *Qddb* runs on most *UNIX* operating systems including: *Linux*, *BSDI*, *FreeBSD*, *SunOS*, *Solaris*, etc... We are currently porting *Qddb* to *Windows NT*; this *NT* port will be available in 1997/1998.

1.3 What you need to know

You'll easily be able to follow along with this guide as long as you're familiar with the basic UNIX commands (e.g. *ls*, *mkdir*, *pwd*, etc...) and can use a simple text editor such as *vi* or *emacs*.

1.4 What's in this guide

Because *Qddb* is relatively new and uses some very powerful technology, we have included a lot of explanatory material along with many pictures illustrating the graphical user interface screens. We discuss how to create and manage information in *Qddb* with an emphasis on the Tcl/Tk-based GUI *nxqddb* and other applications built with the Fx library.

1.5 Typefaces and icons used in this book

Shadow boxes are used throughout the manual for defining terms or to bring attention to hints or key concepts.

This is a shadow box.

Examples, hints and exercises are noted by the following special symbols:

☞ **EXAMPLE**:
☞ **HINT**:
☞ **EXERCISE**:

Buttons, menu buttons and other widgets are shown in double quotes, as in “Edit/Clear” or “Ok” and “Cancel”.

Italics indicate values that you key into entries. We use italics to denote programs, such as *nxqddb*, and to sometimes emphasize certain words.

1.6 How to contact HSDI

HSDI offers a wide variety of services including:

- Software development
- Customer support
- Training
- Hardware consulting and support
- Internet services such as online support, web site host

1.6.1 Visit our Web Page

You may visit our web page on the Internet, **http://www.hsdi.com**. From our web page, you can explore new products, updates, documentation or contact us through electronic mail at **info@hsdi.com**.

1.6.2 Write a letter

If you prefer, write a letter with your comments and mail to:

```
Herrin Software Development
Qddb Software Division
41 South Highland Ave.
Prestonsburg, Kentucky  41653
```

Chapter 2

Developing an *nxqddb* addressbook

To illustrate just how easy it is to create and manage your information in a *Qddb* database, this chapter describes the process. In particular, we'll walk through the steps required to create an address database. In section 2.1 we'll keep the database simple by just having the following fields: (1) first and last name, (2) street, (3) city, (4) state, (5) zip code, and (6) phone number. In section 2.2, we'll expand on the addressbook by allowing multiple addresses and phone numbers.

2.1 Creating a simple addressbook

The first step to creating your database is to determine where you want to store the information. Many users find it convenient to store all of their databases in one directory. For example, you could create your AddressBook database in your Databases subdirectory by executing the commands:

```
1) $ cd ~
2) $ mkdir Databases
3) $ export QDDBDIRS=~/Databases
4) $ cd ~/Databases
5) $ qnewdb AddressBook
6) $ vi AddressBook/Schema
```

Figure 2.1: Steps to create an AddressBook *Qddb* database

Lines 1 and 2 set up your Databases' directory. Line 3 establishes a search path so *Qddb* programs can find any relations in that directory. You can put this line in your .profile so that you will have a nice path set up each time you log in. We have shown the syntax for the *bash* shell; other shells establish environment variables by other syntax. Lines 4 and 5 create a new relation in a subdirectory called AddressBook. Line 6 invokes a text editor for you to create the Schema.

> The Schema file contains a description of your database. In this Schema file, you enter all the field names that you want to store in *Qddb*.

For our sample database, we would enter the following in the Schema file:

```
Firstname verbosename ‘‘First Name’’
Lastname verbosename ‘‘Last Name’’
Street
City
State
Zipcode verbosename ‘‘Zip Code’’
Phonenumber verbosename ‘‘Phone Number’’
```

Figure 2.2: Simple Addressbook Schema

> The optional verbosename keyword provides the label text that appears on the *nxqddb* window. The field names (e.g. Firstname, Lastname, Zipcode, Phonenumber) cannot have spaces; hence, we use the verbosename option to specify how the label should appear.

At this point, you've completed the steps to building your database application. To use your new database, just type the following at your command prompt:

```
$ nxqddb AddressBook
```

The *nxqddb* window(see figure 2.3) will appear to allow you to start entering information into your addressbook. Reference chapter 3 for details on how to use *nxqddb* windows.

2.2 Improving the simple addressbook

Let's improve our addressbook that we built in the previous section. In particular, let's provide the capability to have multiple addresses and phone numbers. And just for fun,

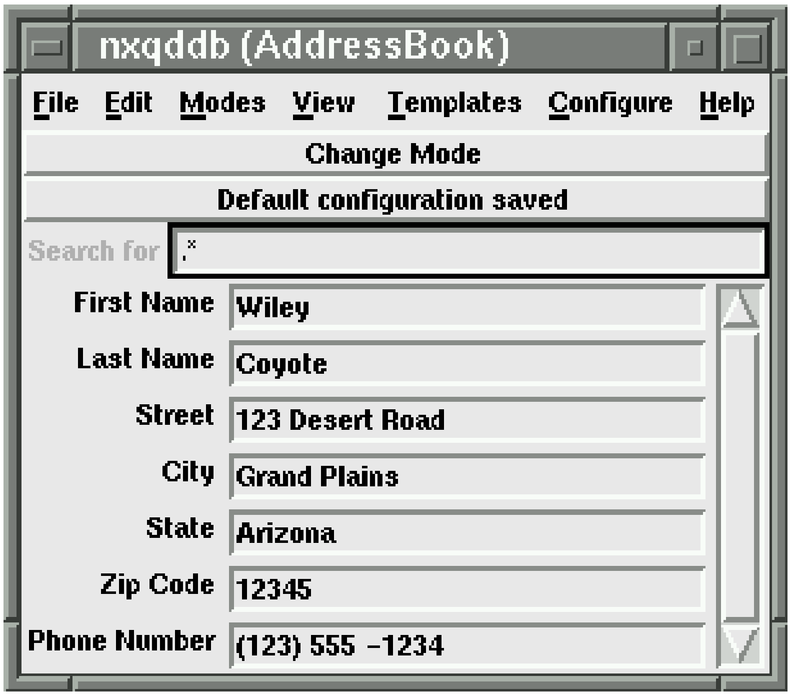


Figure 2.3: A simple address book - *nxqddb* interface

let's add some new fields. We'll add a sex field to show how to dynamically change text entries to radiobuttons and a birthdate field to illustrate some of *nxqddb*'s date functionality. Also, we want to group like fields together. To accomplish these tasks, we modify our Schema file just a little bit. Let's call this version AddressBook2. To create AddressBook2, follow these steps:

```
1) $ cd ~/Databases
2) $ qnewdb AddressBook2
3) $ vi AddressBook2/Schema
```

Figure 2.4: Creating AddressBook2 database

> Note, you should already have the Databases directory setup from the previous section. Also, you should already have the environment variable, QDDBDIRS pointing to your /Databases directory.

In your editing session of AddressBook2/Schema create the schema shown in figure 2.5:

```
Name ( First Middle Last )
Address ( Street City State Zip Email HomePage ) *
Phone ( Description AreaCode Number ) *
Sex
Relation
Birthday type date
Comments *
```

Figure 2.5: AddressBook2 Schema containing Structured Expandable Attributes

There are several things to note about the syntax of the schema shown in figure 2.5:

- We have introduced structured attributes for grouping the Name, Address and Phone data. You specify structured fields by enclosing the sub-fields within parenthesis.
- We have also made the Address and Phone information expandable. Meaning, we can now have multiple addresses and phone numbers. You specify expandable fields by placing an asterisk(*) at the end of the field definition.
- We have added Email and HomePage attributes to the Address grouping.
- We have added an additional Comments field for entering free-form comments about each person.

You can type your schemas free-form. Note in figure 2.5 how we have included more than one field on the same line. For example, First Middle Last are on the same line. You could just have easily created the schema to look like the schema shown in figure 2.6.

After entering this new schema for AddressBook2, we display the *nxqddb* window shown in figure 2.7 by typing "nxqddb AddressBook2"

2.3 Adding some final touches to our addressbook

In this section, we'll refine our AddressBook with the following updates:

1. make the First and Last Name fields mandatory. It doesn't make much sense to have the address without a name, does it?
2. turn the Sex field into radiobuttons: Male, Female and "Do not Know"
3. turn the Relation field into radiobuttons: Family, Friend, Bus., and Other
4. turn the Comments field into a text entry and raise the height to four(4) lines

```
Name (
    First
    Middle
    Last
)
Address (
    Street
    City
    State
    Zip
    Email
    HomePage
) *
Phone (
    Description
    AreaCode
    Number
) *
Sex
Relation
Birthday type date
Comments *
```

Figure 2.6: Another syntax that gives the same results

2.3.1 Making the first and last name fields mandatory

On the *nxqddb* window, position the mouse over the "First" label and then `<Control-3>` click the mouse. To perform this operation, you press and hold the control key and then click the right(or left for left handed users) mouse button. The configuration window shown in figure 2.8 will be displayed.

To make the First name field mandatory, click the "Mandatory" checkbutton and then press the "Ok" button to dismiss the configuration window. When you return back to your *nxqddb* window, you'll notice that the First name entry now has a darker shaded background color. This visual cue tells you that you cannot enter a record in your addressbook without first populating the required First name field.

> Required fields are visibly obvious in *nxqddb* because they have a darker shaded entry background than non-required fields. Before saving or modifying a record, you must be sure that all required fields contain text(at least one character).

To make the Last name field mandatory, repeat the same steps as described for the First name except initially `<Control-3>` click over the Last name field label.

2.3.2 Sex radiobuttons: Male, Female, Do not know

As in the previous section, we want to display the field configuration window by `<Control-3>` clicking the mouse on the "Sex" label. By now, you'll be familiar with the configuration window shown in figure 2.8.

To turn the Sex field into a radiobutton, perform these steps:

1. Click the "Radiobutton" radiobutton.
2. Enter **Male Female "Do not know"** in the Values entry. Note, you must separate the words **Male** and **Female** with at least one space. If your radiobutton name has spaces, then you must surround the value with double quotes. For example, if you wanted to choose Sex based on (1) Male, (2) Female and (3) Do not know, you would enter the values in the radiobutton text entry as shown in figure 2.9:
3. Click the Ok button.

Follow the same steps mentioned above to convert the "Relation" attribute into radiobuttons: Family, Friend, Business and Other.

2.3.3 Aligning radiobutton labels

Notice in figure 2.11, the radiobuttons do not line up with the rest of your *nxqddb* fields. To align radiobuttons, simply change the height of the radiobuttons to 2 or more rows. For example, to ensure that the "Sex" radiobuttons are vertically aligned, do these steps:

1. `<Control-3>` click the mouse on the Sex label to display the configuration window.
2. Enter a value of "2" in the width entry for the "Radiobutton" entry.
3. Click the "Ok" button

nxqddb will reformat your window such that the "Sex" radiobutton label is vertically aligned with the rest of the fields.

2.3.4 Changing Comments field to text area with height of 4 lines

`<Control-3>` click the mouse on the Comments label to bring up the familiar field configuration window. To make the Comments field a text area with a height of four(4) lines, just do these simple steps:

1. Click the "Text" radiobutton

2. Enter a value of "4" in the Height entry

3. Click the "Ok" button

2.3.5 Resizing your *nxqddb* display window

After reconfiguring fields, your *nxqddb* window will be automatically updated to reflect your recent changes. You may have to resize the window to view any hidden changes that may occasionally occur when you enlarge widgets(i.e. increase text area height). To quickly resize the window, left-click on the maximize window icon(see figure 2.10).

After making the changes from this section, your "nxqddb AddressBook2" window should resemble the one shown in figure 2.11.

2.3.6 Saving your changes

In this chapter, we have made several changes to the display configuration of our addressbook. In order to have these changes present after we exit *nxqddb*, you must be sure to save your changes.

You can save (1) personal or (2) global configuration or (3) default settings by selecting the appropriate Cofigure menubutton. When you click one of these "Configure/Save" buttons, the status bar will be updated accordingly (see figure 2.12). Reference section 5.2 for details about each of these configurations.

If you have"Auto-Save Personal Configuration" turned on, *nxqddb* will automatically save any configuration changes that you have made in your current session before you exit *nxqddb*.

Reference chapter 5 for a complete discussion of saving/loading configuratons as well as a discussion of all of the *nxqddb* configuration features.

2.4 Redesigning the interface using QddbScript

You can use *QddbScript* to change the look of your addressbook. Just to give you a gentle introduction to *QddbScript*, follow these steps to make your AddressBook2 look like the one shown in figure 2.14.

1. cd /Databases

2. qddbscript autogen AddressBook2

3. vi AddressBook2/QddbScript

4. nxqddb AddressBook2

Line 1 places you in your Databases directory that contains your AddressBook2 relation. Line 2 runs the qddbscript command to autogen (automatically generate) a QddbScript file contained under the AddressBook2 directory. Line 3 says to edit the QddbScript file. Edit your QddbScript file to look like the one shown in figure 2.13.

Now, when you run *nxqddb AddressBook2* the window shown in 2.14 will be displayed. Of course, you can edit the AddressBook2/QddbScript file to rearrange the fields, modify or add new field settings, or even add new Tcl code.

To download a fancier addressbook that uses QddbScript or to see other QddbScript applications, visit *http://www.hsdi.com/qddb/applications*

For complete details on how to use *QddbScript* reference the *QddbScript in a Nutshell* guide.

2.5 Learn by doing

At this point, you have created a reasonably nice addressbook that is completely functional through the *nxqddb* program.

If you haven't already, experiment with *nxqddb* by adding new entries, searching for existing entries, and configuring various *nxqddb* features. If you're like many users, the *nxqddb* interface is easily learned just by playing around a little bit. Of course, we continue our explanation of **Using** *nxqddb* in the next chapter if you prefer to follow along in this guide.

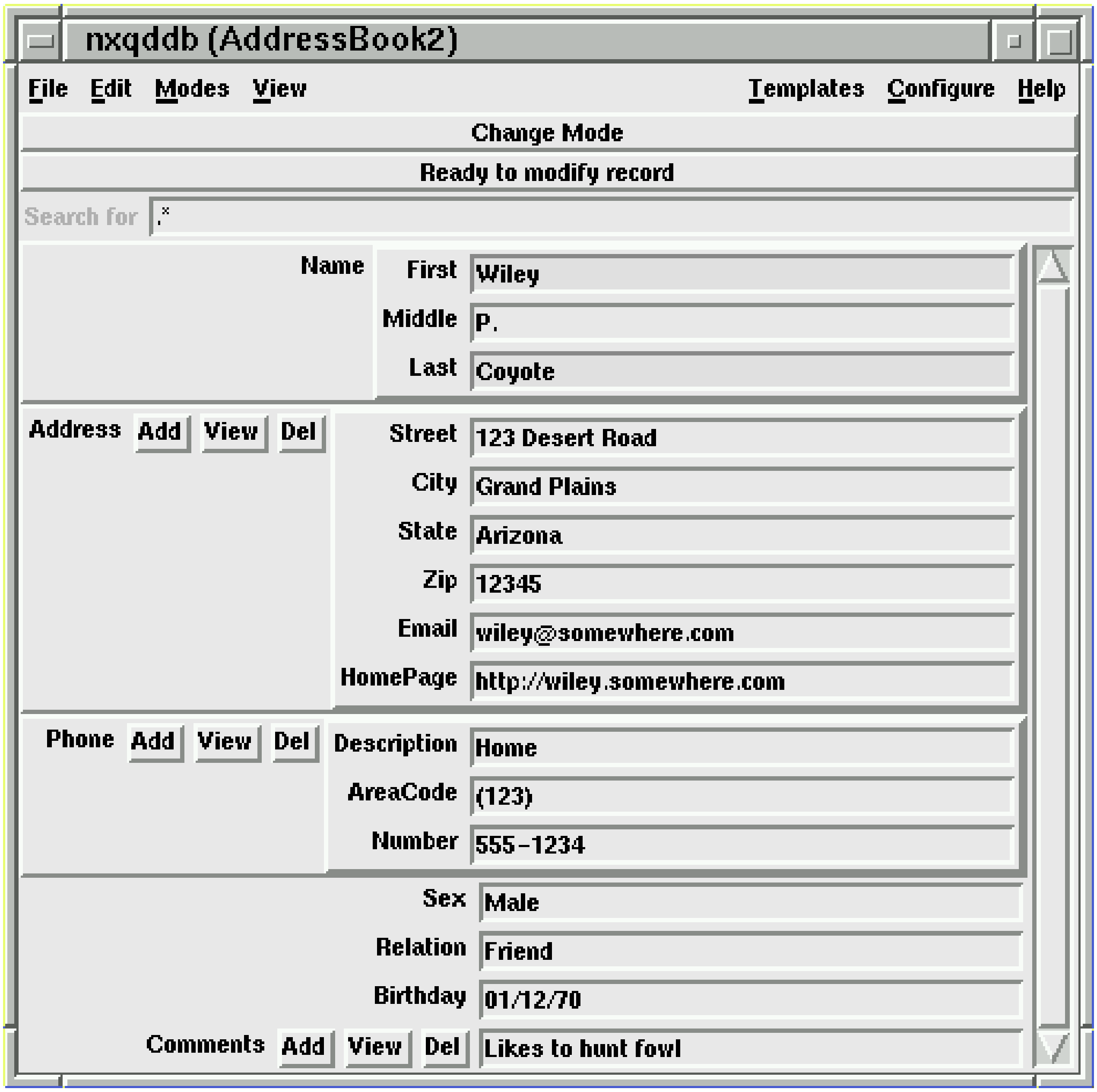


Figure 2.7: An improved Addressbook

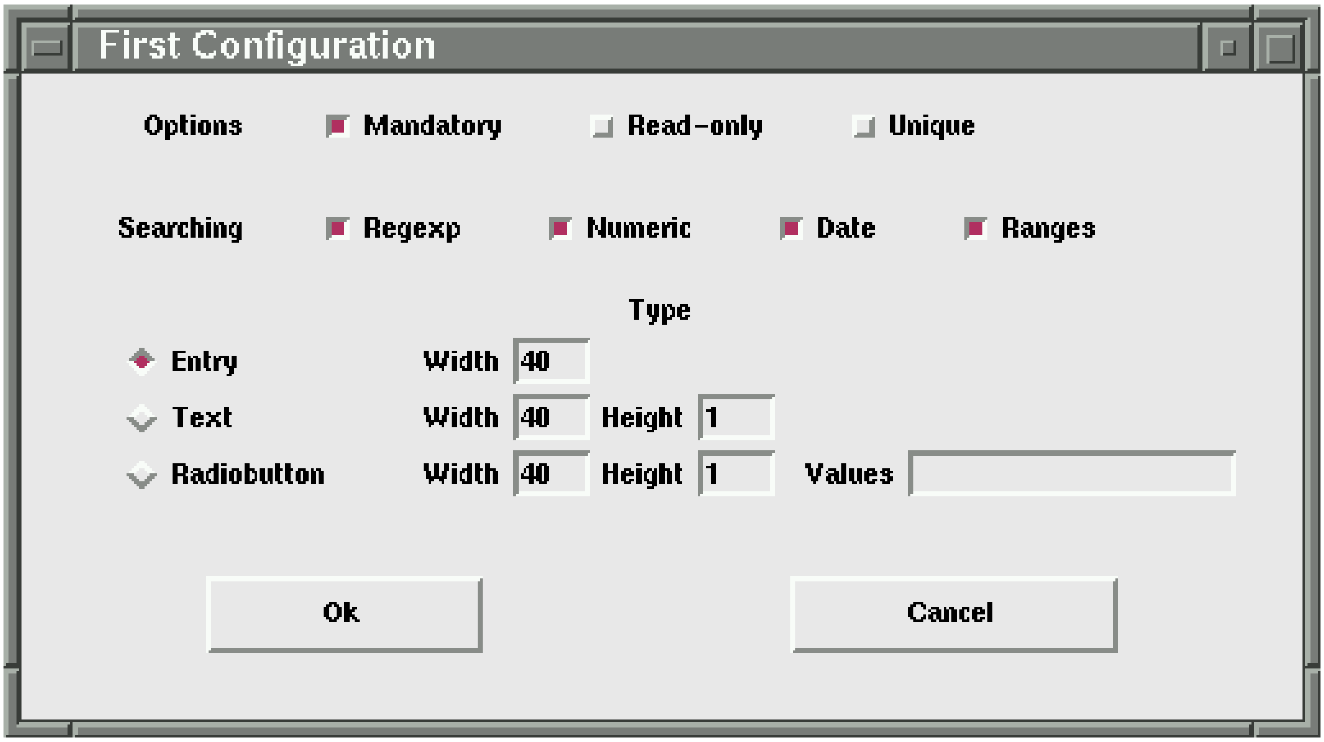


Figure 2.8: Configuring individual fields

```
Male  Female  ``Do not know''
```

Figure 2.9: Entering radiobutton names that have spaces

Figure 2.10: Iconify Button

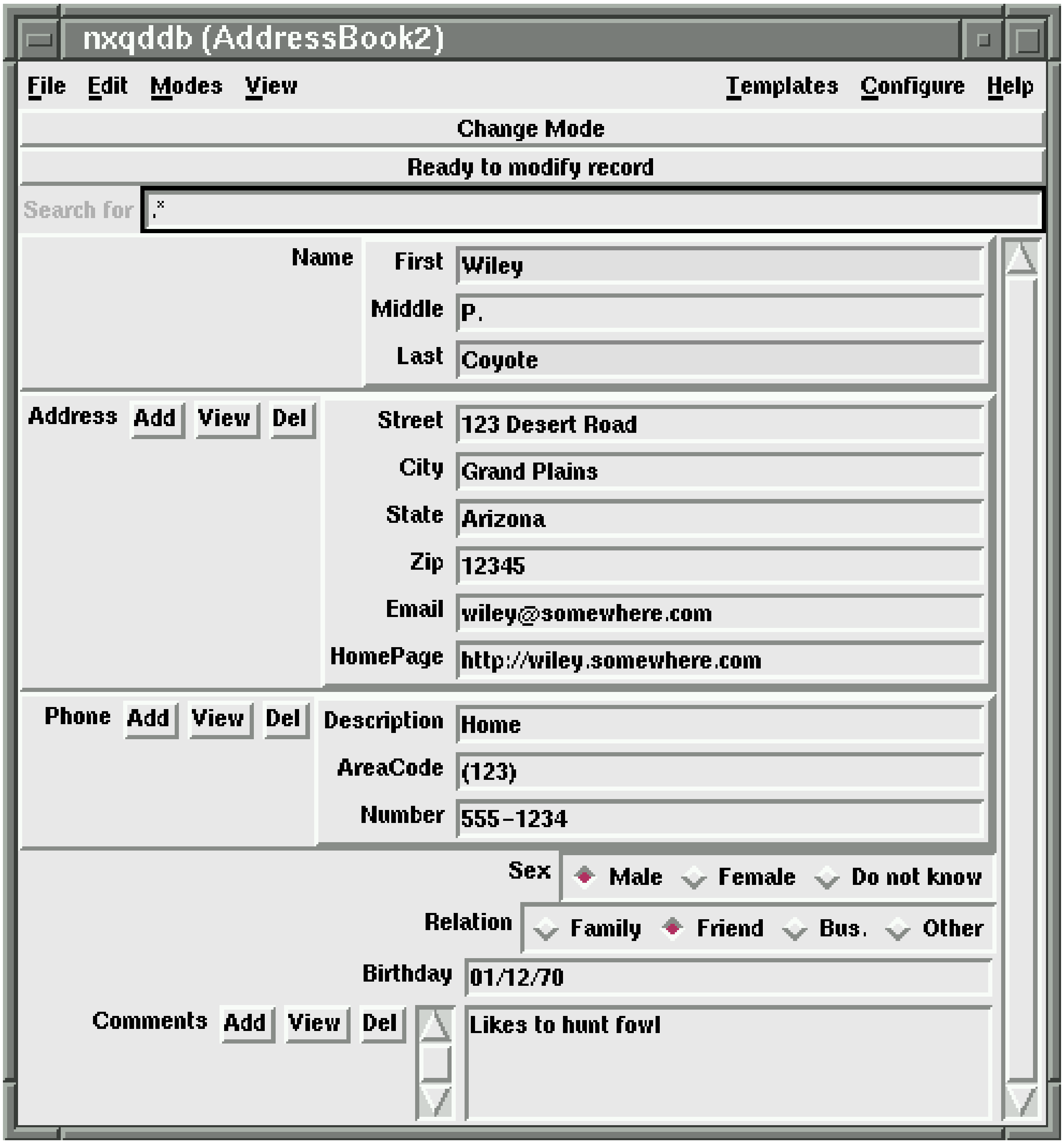


Figure 2.11: Revised AddressBook2

Figure 2.12: Personal Configuration Saved

```
[prologue]
[menubar]
[options autoalign]
[frame attr=Name border=3 relief=groove]
[entry attr=Name.First label="First" width=10]
    [entry attr=Name.Middle label="Middle" width=10]
    [entry attr=Name.Last label="Last" width=10]
    [break]
[end frame]
[break]
[frame attr=Address border=3 relief=groove]
    [entry attr=Address.Street label="Street" width=10]
    [break]
    [entry attr=Address.City label="City" width=10]
    [entry attr=Address.State label="State" width=2 noexpand]
    [entry attr=Address.Zip label="Zip" width=10]
    [break]
    [entry attr=Address.Email label="Email" width=10]
    [break]
    [entry attr=Address.HomePage label="Home page"width=10]
[end frame]
[break]
[frame attr=Phone border=3 relief=groove]
    [entry attr=Phone.Description width=10]
    [entry attr=Phone.AreaCode width=5 noexpand]
    [entry attr=Phone.Number width=10 noexpand]
[end frame]
[break]
[frame border=3 relief=groove]
    [radiobutton attr=Sex value="Male" value="Female" value="Unknown" noexpand]
    [break]
    [radiobutton attr=Relation value="Relative" value="Friend"
        value="Colleague" value="Other" noexpand]
    [break]
    [entry attr=Birthday width=10]
    [break]
[end frame]
[frame border=3 relief=groove]
    [options noautoalign]
    [text attr=Comments height=4 width=50]
    [code value="pack [=Comments cget -w].f_0 -side top -anchor nw"]
[end frame]
[epilogue]
```

Figure 2.13: Sample QddbSript file for AddressBook2

Figure 2.14: AddressBook2 using QddbScript

Chapter 3

Using *nxqddb*

This chapter focuses on how to use *nxqddb* — *Qddb*'s generic graphical user interface program. Remember from the previous chapter, you were able to run *nxqddb*(see figure 2.7) on your AddressBook by typing the command shown in figure 3.1.:

```
$ nxqddb AddressBook2
```

Figure 3.1: Command to start *nxqddb* program

3.1 Adding an entry

To add new entries, follow these steps:

1. Go to "Add Mode"; click the "Modes/Add Mode" menubutton.
2. Enter the appropriate information.
3. Choose File/Save to store the new information.

When *nxqddb* first appears, you will be placed in "Search Mode", the default mode. In order to add an entry, you must first get into "Add Mode". To tell *nxqddb* to switch to "Add Mode", click the "Modes/Add Mode" menubutton from the menubar. You'll know you are in add mode when the status bar(see figure 3.2) displays the add mode message.

3.1.1 Editor keys

All *nxqddb* entries uses emacs-style key bindings. See table 3.1 for a list of the *nxqddb* key bindings.

	Action taken for various widgets		
Keystrokes	Entry box	Text box	Radiobutton
`<Tab>, <DownArrow>`	Go to the next field or radiobutton		
`<Shift-Tab>, <UpArrow>`	Go to the previous field or radiobutton		
`<Control-a>`	Go to beginning of line		
`<Control-b>, <LeftArrow>`	Go back one character		
`<Control-d>`	Delete one character to the right		
`<Control-e>`	Go to the end of the line		
`<Control-f>, <RightArrow>`	Go forward one character		
`<Control-h>`	Delete one character to the left		
`<Control-k>`	Delete to the end of line		
`<Control-o>`		Open a blank line	
`<Control-u>`	Clear the current line		
`<Control-w>`	Delete the current selection into cut buffer		
`<Control-y>`	Paste the cut buffer at current cursor position		
`<Alt-w>`	Copy the current selection into cut buffer		
`<Button-2-Motion>`	Scroll horizontally	Scroll vertically	
`<Button-3>`	Paste the current selection		
`<Return>`	(Search Mode) Start the search	Go to next line	Set value
`<space>`	blank space	blank space	Set value
`<Shift-Return>`		Go to next line	Unset value
`<Shift-space>`	blank space	blank space	Unset value

Table 3.1: Key interpretations

Figure 3.2: Add Mode status

3.1.2 Navigating within Add Mode

In add mode you can move among the entries by:

1. pressing the tab or down-arrow keys to move down a field.

2. pressing the shift-tab or up-arrow keys to move up a field.

3. left-clicking with the mouse on a field's entry box.

3.1.3 Saving your entry

Once you have keyed in the information for the record you wish to add, the "File/Save" menubar option(see figure 3.3) becomes enabled. Click this menubar button to save your

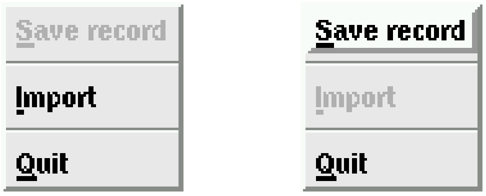


Figure 3.3: File Pull-down Menu

entry. *nxqddb* will save your information and the status bar will be updated to reflect that you've added a record(see figure 3.4). Additionally, the window will be prepared for your

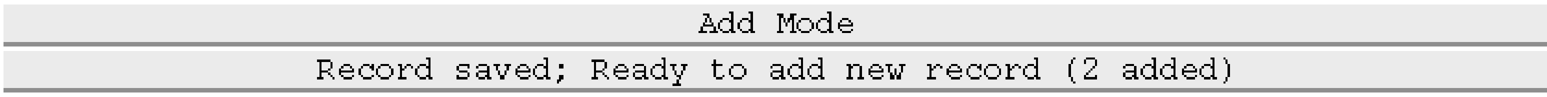


Figure 3.4: Service saved status message

next entry.

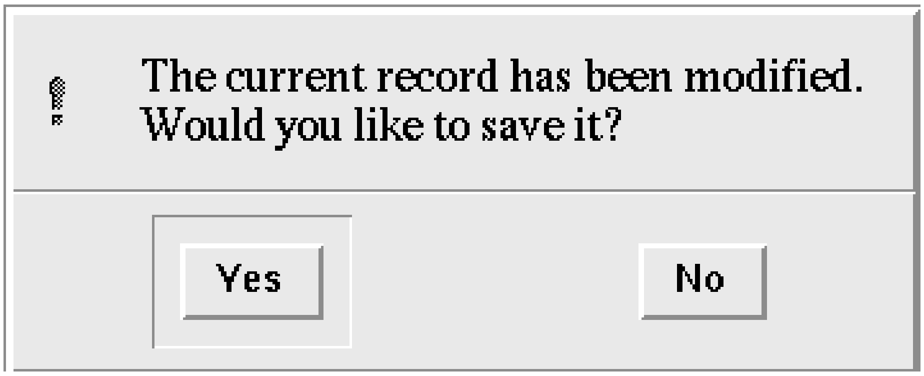


Figure 3.5: Confirm Save Dialog

3.1.4 Exiting Add Mode

Once you've entered your record(s), you can change to another mode by clicking one of the Modes' menubuttons or exit the *nxqddb* application by choosing the File/Quit menubutton. If you've entered any text while in Add mode, *nxqddb* will prompt you with a confirmation dialog box(see figure 3.5). This confirmation dialog box serves as a precaution to warn you before leaving Add mode without saving your input. You can choose not to have this dialog box appear by turning on the "Auto-Save Tuples" option (see chapter 5.12 for details.)

3.2 Searching for an entry

In *nxqddb*, searching for information is fast and easy. Here are the steps to searching:

1. Go to Search Mode by clicking the "Modes/Search Mode" menubutton.
2. Tab to the field(s) that you want to search on and type in the search value(s).
3. Press the **Enter** key or click the "Edit/Search" menubutton.

3.2.1 Types of search results

There are three basic results of a search; these results are described below:

1. If there are no matches found, *nxqddb* will display "Mo matches found" in the status bar. You can then refine your search criteria and try again.
2. If there is exactly one match found, the matching record's information will be displayed in the *nxqddb* window and the mode will switch to "Change Mode" — you will then be able to view, modify or delete the found record.

3. If there are two or more matches found, *nxqddb* will display a listbox containing all the matches found from your search. You can choose a record to view from the results listbox window by clicking on the desired record and then clicking the "Ok" button to dismiss the results listbox window.

Please reference chapter 4 for a complete discussion of searching.

3.3 Modifying an entry

To modify an existing record in your *Qddb* database, follow these steps:

1. Go to Search Mode and enter the search criteria to find the record. Once the record is found, you should be in "Change mode" with the information displayed in the *nxqddb* window.

2. Update the fields that need to be changed.

3. Choose "File/Save". *nxqddb* will display "Record has been saved." in the status bar to inform you that the entry has been saved.

3.4 Deleting an entry

To delete a record from your *Qddb* database, follow these steps:

1. Go to Search Mode and enter the search criteria to find the record. Once the record is found, you should be in "Change mode" with the information displayed in the *nxqddb* window.

2. Click the "Edit/Delete" menubutton. At this point, *nxqddb* will display a confirmation dialog box(see figure 3.6) to get confirmation that you really do want to delete the record. Click "Cancel" if you do not want to delete the record; click "Ok" to go ahead and delete the record permanently from the *Qddb* database.

3. After you delete the record, *nxqddb* will return back to "Search mode".

3.5 Accelerator Keys

With *nxqddb*, certain actions have accelerator keys that you can press instead of using the mouse to invoke the action. Each menu selection in the menubar has a letter that is underlined. This visual clue lets you know that the selection has an accelerator key defined by pressing(and holding) the **ALT** key while pressing the key that is underlined. For

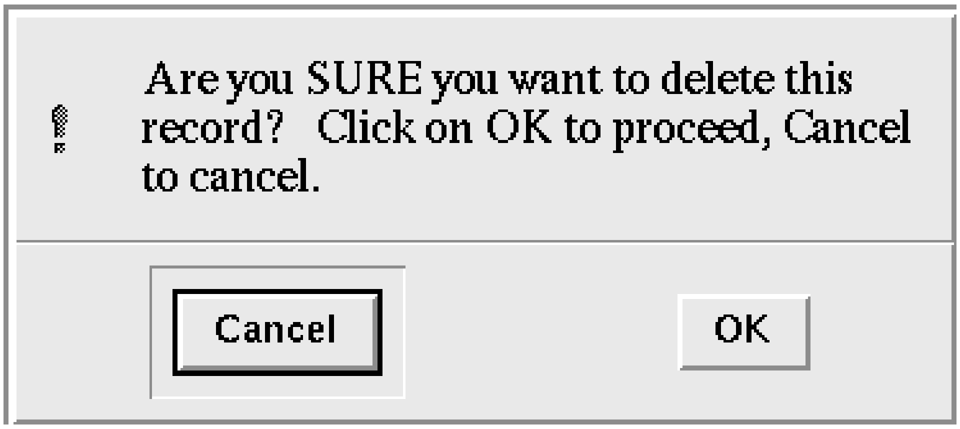


Figure 3.6: Dialog box to receive confirmation before deleting a record

example, if you press the <ALT-F> key, the **File** pulldown menu will be displayed. Pressing the <ALT-E> key displays the **Edit** menu and so on.

Once the pulldown menu is displayed, you can also use accelerators to launch a particular action on the menu by pressing the <ALT> key combination specified to the right of the menu label.

☞ **EXERCISE**: To see how to use accelerators, lets switch back and forth between Change Mode and Read-Only Mode. To perform this example, first find a record and get into change mode. Then, with the record in change mode, do this key sequence:

ALT-M to display the Modes menu

ALT-R to go to Read-Only Mode

ALT-M to display the Modes menu

ALT-C to go to Change Mode

Chapter 4

Searching

This chapter provides information on performing simple searches within an *nxqddb* application. In particular, this chapter discusses:

- How to perform a search
- Narrowing your search criteria
- Configuring search results
- Generating quick reports using the simple search
- Expert searching

4.1 Introduction to Searching

Let's first understand what exactly a **search** means in *nxqddb*.

> A search is simply a term used to describe the action that a user takes to finding information from a *Qddb* database.

Detailed steps on how to perform, narrow, and even configure your searches are discussed in the following sections. This chapter also provides many sample searches for you to learn by example.

4.1.1 Steps to searching

Searching is extremely easy in *nxqddb*. And, since all *nxqddb* windows share the same look and feel, once you learn how to search in one window you'll have mastered all *nxqddb* windows. The basic steps to performing a search are:

1. Be sure that you are in Search Mode. The default mode is search mode for all *nxqddb* windows; meaning, when you first open a window, you will be placed in Search Mode. The mode bar will read *Search Mode* to let you know that you are in Search Mode. If you're not in Search Mode, click the *Modes/Search Mode* menubutton.

2. Key in the values in the appropriate fields and then press the *Enter* key. You also have the option of choosing the *Edit/Search* menubutton to perform the search. However, users find that it is easier to press the *Enter key*.

3. After pressing the *Enter* key to invoke the search, *nxqddb* will search its database to find the requested information. If *nxqddb* finds more than one instance, then a listbox will be displayed to allow you to choose the particular instance desired. Choosing a row in the instance listbox will populate that instance in the search window and the mode will change from *Search Mode* to *Change Mode* (or *Read-only Mode* if you've configured Read-only on the window).

4. If no instances are found that match your search criteria, *nxqddb* will display *No matches found.* in the Status bar and you will be returned to the window to try again.

5. If exactly one instance is found that matches your search criteria, *nxqddb* will populate the window with the found instance and you'll be placed in *Change Mode* (or *Read-only Mode* if you've configured the window for *Read-only Mode*).

4.1.2 An example of a common search

Suppose you wanted to view a particular person in your addressbook; but you only remember that his first name is *John*. From within the *nxqddb* window, you would type *John* in the First name field and then perform the search by pressing the *Enter* key. All of the people that have a first name of *John* would be found and displayed to you in a **Search results** listbox. You would then be able to choose from the list to find the person for whom you were searching.

4.1.3 *nxqddb* searching is case insensitive

nxqddb performs case insensitive searches. This just means that *nxqddb* finds all occurrences of a word whether it is capitalized or not. For example, suppose you search on *McDonald*.

nxqddb will return all clients that have any combination of McDonald such as *mcdonald*, *Mcdonald*, *McDonald*, *Mcdonald*, *MCDonald*, or *MCDONALD*.

> *nxqddb*'s case insensitive search feature is extremely helpful if data has been entered into your database with inconsistent capitalization.

4.1.4 Meaningless search fields

Sometimes it becomes convenient to exclude fields from a search when the results of searching the field are meaningless. In *Qddb*, you can specify whether or not to exclude a field from being searchable. Meaning, if you edit your *Schema* file to enter the keyword **exclude** next to a field name, *Qddb* will not store any index information for that field. And, the focus will skip over that field if the user tabs to the field while in Search mode.

☞ **EXERCISE**: To see the "exclude" feature at work, modify your AddressBook schema by typing the word **exclude** next to a field name. Then, restart *nxqddb* AddressBook2 and observe how the cursor skips over the excluded field while in search mode.

4.1.5 Clearing search fields

To clear all search fields, click the *Edit/Clear* menubutton.

> A good habit to develop is to always clear the search fields prior to performing a search. To clear the search fields, click the **Edit/Clear** menubutton.

If your search does not return what you think it should, it may be possible that you have fields populated from a previous search. It may also be possible that one of the notebooks not currently displayed contains a search field value that you do not want to include in your search. By always clicking the *Edit/Clear* menubutton prior to entering your search criteria, you are more likely to have a successful search.

4.1.6 Sample searches

For the following sample searches, you should be in the *nxqddb* AddressBook2 window in Search Mode:

Find person named John Key in *John* in the First name field and press the *Enter* key. If there are no *John*s, then a status message stating *No matches found* will be displayed and you'll return to the *nxqddb* window to try again. If there is a person whose first name is *John*, then the information corresponding to *John* will be displayed in the *nxqddb* window. Of course, if there is more than one *John*, then the list of people whose first name is *John* will be displayed for you to choose from. Chances are(if you're database is large), there will be more than one person that has a first name of *John*. So, you will likely want to narrow your search by keying in a last name as well.

Find person named John Doe Key in *John* in the First name field and *Doe* in the Last name field; press *Enter*.

Find all people that live in a particular state, say Ky Key in *Ky* in the State field and press enter. Note, this search result would likely yield a large number of people if you live in Kentucky. So, you would probably want to narrow the search to include a city or zip code.

Find all people that live in the city of Louisville Key in *Louisville* in the City field and press *Enter*. All the people that reside in the entered city will be displayed in a listbox for you to view, print, or choose to display in the client window.

Find all people that reside in Zip Code 41017 Key in *41017* in the Zip Code field and press *Enter*.

Find all Females Click the *Female* radiobutton and then press *Enter*.

Find people named Butch, Billy, and Bobby who live on Main street Key in *Butch , Billy , Bobby* in First Name field and key in *main* in the Street field; then press *Enter*. Remember, capitalization is not required; typically, users find it convenient just to type all searches in lowercase letters.

> By entering more than one word in a search field separated by a comma(,) *nxqddb* returns the union of those instances that match either word. In this case, the returned instances will be all Main street residents that have one of the names: Butch, Billy or Bobby.

Find a person named John Doe who likes basketball Key in (1) *John* in the first name field, (2) *Doe* in the Last Name field and (3) *Basketball* in the Comments field and then press the *Enter* key. Be sure to press the enter key outside of the Comments field. Meaning, tab out of the comments field prior to pressing return so that *nxqddb* will execute the search. Remember, pressing the return key from a text area widget has a different meaning; namely, the cursor is placed on the next line.

Find all people born on May 2, 1986 Key in *05/02/86* in the Birthday field and press *Enter*.

Find all people born in Month of May for 1975 Key in *05/01/75-05/31/75* in the Birthday field and press *Enter*.

> By keying a dash in a search field, *nxqddb* knows to perform a range search. The search results will include those instances that match the values that fall within the range specified, including the two end dates - in this case, 05/01/96 and 05/31/96.

Find all people born today Key in *today* in the Birthday field and press *Enter*.

Find all people born yesterday Key in *today -1 day* or *yesterday* in the birthday field and press *Enter*. Note, there must be a space after the word "today" but no space between the dash and the number one(1).

4.2 More Advanced searches

In addition to performing the simple searches discussed in the previous sections, *nxqddb* provides some special types of searching. These more advanced search techniques are discussed in this section.

4.2.1 The Generic Search for field

Each *nxqddb* window contains a *Search for* field(see figure 4.1). This field is expressly used for searching and is therefore only active while in Search mode.

Client Name	*Address*	*Comments*
John Williams	123 Main Street	Pet named **Rex**
Tim McGrath	33 **Rex** Drive	Plays Poker
Barb **Rex**	22 Vine Street	Running partner

Table 4.1: Search data example

Figure 4.1: The Search for Field

This **Search for** field provides a special searching capability. Values entered in this field are compared against all the fields. If a match is found in any of the fields, the record is returned.

For example, suppose you type *Rex* in the **Search for** field. The people returned would be those records that contained **Rex** in any field. Suppose you had the following people(see table 4.1), each one of these people would be returned from the search since they each have the value *Rex* in at least one field.

4.2.2 Searching on Multiple fields

nxqddb provides the capability to search on more than one field. There were some sample searches discussed earlier that illustrated this technique.

Steps to searching multiple fields

To search on more than one field, you follow these steps:

1. As in any other search, be sure your window is in Search Mode. If you're not in Search Mode, click the *Edit/Search Mode* menubutton to enter Search Mode.

2. For each field that you'd like to search, key in the appropriate search value.

3. Press the *Enter* key.

Example of searching multiple fields

For example, suppose you would like to find a person named *John Doe* who lives in *Ludlow*. You could find this information by following these steps:

1. Key in *John* in the first name field.

2. Tab to the Last Name field and key in *Doe*.

3. Tab to the City field and then key in *Ludlow*.

4. Once you've entered all the fields, press the *Enter* key.

The results will be displayed in a search results listbox if there are more than one instance. Again, if there is only one instance, then the instance data will be displayed in the search window and the Mode will change from *Search Mode* to *Change Mode*.

Note, the results returned will be those instances that match on all three fields. That is, you'll receive only those instances where the Name is *John Doe* who lives in the city of *Ludlow*. In other words, the result is the intersection of instances whose (1) first name equals *John*, (2) last name equals *Doe* and (3) city equals *Ludlow*.

There is no limit to the number of fields that you can search on. You can search on any field displayed in the window, provided that you can tab into the field while in Search mode.

4.2.3 Searches with more than one value(Comma Delimiter)

You can specify more than one search field value simply by delimiting the values by commas. For example, suppose you wanted to find all the people whose name is either *Timothy* **OR** *Carol*. You would do these steps:

1. On the window, be sure you're in Search mode. If you're not in Search mode, click the *Edit/Search Mode* menubutton to go to Search Mode.

2. Press the *Edit/Clear* menubutton to be sure that no other fields are populated.

3. Tab to the first name field.

4. Key in *Timothy , Carol*

5. Press the *Enter* key

Of course, you can have multiple search fields whereby each search field can specify one or more search values. This type of search basically combines the above two search techniques. Lets clarify this with an example.

Suppose you'd like to find all people that have a last name of either *Simpson* or *Flanders* and who live in the city *Springfield* or *Chicago*. To find these instances, you would follow these steps:

1. Tab to the last name field.
2. Key in *simpson , flanders*. (Remember, case is insignificant in search mode)
3. Tab to the city field.
4. Key in *springfield , chicago*
5. Press the *Enter* key.

There is no limit to the number of fields or the number of search field values.

4.2.4 Specifying search values in a Date Field

You can do searches on date fields. The format for entering dates is mm/dd/yy where (1) mm denotes the month, (2) dd denotes the year, and (3) yy denotes the year.

Valid date entries are shown below:

- 01/01/96 denotes January 1, 1996
- 03/22/95 denotes March 22, 1995
- 11/04/97 denotes November 4, 1997

Date entries must not contain any spaces.

You may also enter dates related to some date. *nxqddb* provides some keywords (or short cuts) for specify dates. These shortcuts are listed in table 4.2.

You may also use the day-of-week keywords **Monday**, **Tuesday**, **Wednesday**, **Thursday**, **Friday**, **Saturday**, and **Sunday**. Month names are also fine. For example, the following are all valid input to a date attribute:

next tuesday
next december
next 9 dec
december 1, 1995 + 1 week
9 dec 1995
wednesday

Date entry keywords		
Keyword	Meaning	Example
tomorrow	24 hours	tomorrow 12:00AM
today	current date/time	today 1:00PM
yesterday	current date/time - 24 hours	yesterday 1:00PM
last	prefix (-1 unit)	last week
next	prefix (+1 unit)	next year
ago	suffix (-X units)	3 years ago
epoch	beginning of time	epoch + 1 year
year(s)	unit of time	last year + 1 week
month(s)	unit of time	next month
week(s)	unit of time	next week
day(s)	unit of time	today + 3 days
hour(s)	unit of time	now + 3 hours
minute(s)	unit of time	now - 20 minutes
second(s)	unit of time	now - 1000 secs

Table 4.2: Generic Search For Field

The input “**next tuesday**” does not mean this coming (or current) Tuesday, but the one after. The keyword “**tuesday**” means the coming (or current) Tuesday.

4.2.5 Specifying Ranges for Numeric and Date Field

nxqddb provides the ability to perform range searches on Numeric and Date fields. Since our guestbook doesn't have any numeric fields, we'll create a new schema for handling, say invoices. When we refer to the “Invoices” window, we'll be referring to the *nxqddb* window displayed using the schema shown in figure 4.2.

Actually, the schema shown in figure 4.2 is taken directly from a commercial application — Faunix. If you'd like to view the schemas(and complete source code for that matter), visit **http://www.hsdi.com** on the internet.

A range search is performed by keying the lower limit(or value) followed by a dash followed by the upper limit.

Range search on a numeric field

For example, suppose you wanted to find all the invoices in the range 100 through 200. The following steps show you how to find this range of invoices including the invoice 100 and invoice 200. *nxqddb* range searches return the lower and upper limits.

1. On the Invoices window in Search Mode.
2. Tab to the Invoice Number field.
3. Key in *100-200* and press the *Enter* key.

Range search on a date field

You can also perform range searches on date fields. Similarly to range searching on numeric fields, you specify a range search by keying in the lower limit value (or starting date) followed by a dash, followed by the upper limit(or ending date).

The following examples illustrate range searches on date fields:

Find invoices created in July, 1996 From the Invoices window, tab to the Date field, Key in *07/01/96-07/31/96* and press the *Enter* key.

Find invoices created for 1996 From the Invoices window, tab to the Date field, Key in *01/01/96-12/31/96*, and press the *Enter* key.

@firstofmonth(today) and @lastofmonth(today)

There are many times that you may want to find information for an entire month. To simplify this task, *nxqddb* provides two special functions: *@firstofmonth(today)* and *@lastofmonth(today)*.

These two functions find the first/last day of the month. For example, suppose today's date was say 10/11/96. Then, *nxqddb* would convert the following:

- *@firstofmonth(today)* would convert to 10/01/96
- *@lastofmonth(today)* would convert to 10/31/96

```
Use Reduced Attribute Identifiers
Use Cached Hashing
CacheSize = 500
HashSize = 20000

Number verbosename "Invoice Number" type integer
Client (
        ID verbosename "Client #" type integer
        Name
)
Date type date
Time exclude
DayOfWeek verbosename "Day of Week"
Pet (
        PetID verbosename "Animal #" type integer
        PetName verbosename "Animal Name"
        Doctor
        Personnel
        EntryCode verbosename "Entry Code"
        Items (
                Qty type integer
                Code
                SvcType
                Desc verbosename "Description"
                Taxable exclude
                Price type real format "%.2f" exclude
                Total verbosename "Line Total" type real format "%.2f" exclude
                DiscountedTotal verbosename "Discounted Total" type
                        real format "%.2f" exclude
        )*
        Subtotal type real format "%.2f" exclude
)*
Subtotal type real format "%.2f" exclude
Discounts type real verbosename "Discounts" format "%.2f" exclude
Tax type real verbosename "Sales Tax" format "%.2f"
Total verbosename "Invoice Total" type real format "%.2f" exclude
PaymentType verbosename "Type of Payment"
PaymentAmount verbosename "Payment Amount" type real format "%.2f" exclude
PreviousBalance verbosename "Previous Balance" type real format "%.2f" exclude
AmountDue verbosename "Total amount due" type real format "%.2f"
```

Figure 4.2: Invoice Schema

These two functions are especially useful for storing search templates that are run every month. Using these special functions allows you to not have to re-key the date fields each time you perform your search.

4.2.6 Exclusion using exclamation point(!) delimiter

nxqddb allows you to search for instances where a field does **NOT** equal some specified value.

☞ **EXAMPLE**: Suppose you wanted to find all the invoices for March, 1996 but excluding those from March 5, 1996. You could accomplish this by keying *03/01/96-03/31/96 ! 03/05/96* in the Invoice Date field on the Invoices window.

Suppose you would like to find all people except for those named Smith. To accomplish this, follow these steps:

1. From the AddressBook2 window, in Search Mode, tab to Last Name field.

2. Key in *.* ! Smith* and press the *Enter* key.

> Note, the special characters **.*** is a regular expression that tells *nxqddb* to search for all instances. Suffixing the .* with the *! Smith* then tells *nxqddb* to exclude all the instances that contain the word Smith. The end result is that all clients not named *Smith* will be found.

4.2.7 Regular expression searches

nxqddb provides regular expression searches. A regular expression search is a way to specify wild cards in your search. To accomplish this searching technique, you use the dot-star character combination(.*).

If you enter .* in a search field, *nxqddb* will return all the instances in the database. For example, if you're in the AddressBook2 window and enter .* in any field, then all the people will be displayed in the search results listbox.

> Searching using the .* may result in long searches if you use no other search constraints. Typically, you would not want to just search on .* only. Rather, you would combine the .* with other search values.

Here are some sample searches and their results using the .* wild-card.

Find all people Tab to any field, key in **.***, and then press the *Enter* key.

Find all people whose first name starts with the letter A Tab to first name field, key in **A.***, and then press the *Enter* key.

Find all people whose first name starts with the letters Al Tab to first name field, key in **Al.***, and then press the *Enter* key. If your addressbook had people with names Al, Albert, Alfred, Allen, and Alberta — all of these names would be returned.

Find all people whose first name starts with the letter A and ends in the letter n Tab to first name field, key in **A.*N**, and then press the *Enter* key. If your addressbook had people with names Al, Albert, Alfred, Allen, and Alberta — only Allen would be returned since its the only name that starts with an A and ends with an N.

Find all people living in zip codes starting with 405 Tab to zip code field, key in **405.***, and then press the *Enter* key. All zip codes starting with 405 would be returned. If your addressbook had people living in 40502, 40503, 40521 — all of these people would be displayed in the Search results listbox for you to browse, update or print.

4.2.8 Grouping search values using curly braces ({})

If you want to build more complex searches, you can group search values using curly braces({}).

☞ **HINT**: To search for "Billy Ray", Tommy, and Jack you would (1) Tab to the first name field. (2) key in **{Billy Ray} , Tommy , Jack** (3) press the *Enter* key.

The braces surrounding { *Billy Ray* } are required since there is a blank between Billy and Ray.

4.3 Search Templates

nxqddb provides the ability for you to store/retrieve your search to/from files. This feature enables you to quickly retrieve common search values from a file(stored on disk) rather than manually typing the information each time you wish to perform a search.

4.3.1 Steps to storing a search template

Follow these steps to store a search template:

1. From Search Mode, click the *Edit/Clear* menubutton to clear the fields. Doing this step is a good habit to develop.
2. Key in your search criteria for the fields you wish to search.
3. Click the Templates/Write Template menubutton.
4. A dialog box(see figure 4.3) will appear showing the current templates that you've previously created.
5. Key the name of the file in the *Write to file:* entry.
6. Click the *Write* button.
7. At this point, you've stored your search in the specified file.

You'll now be able to retrieve this search template at any time since it has been stored on the hard disk.

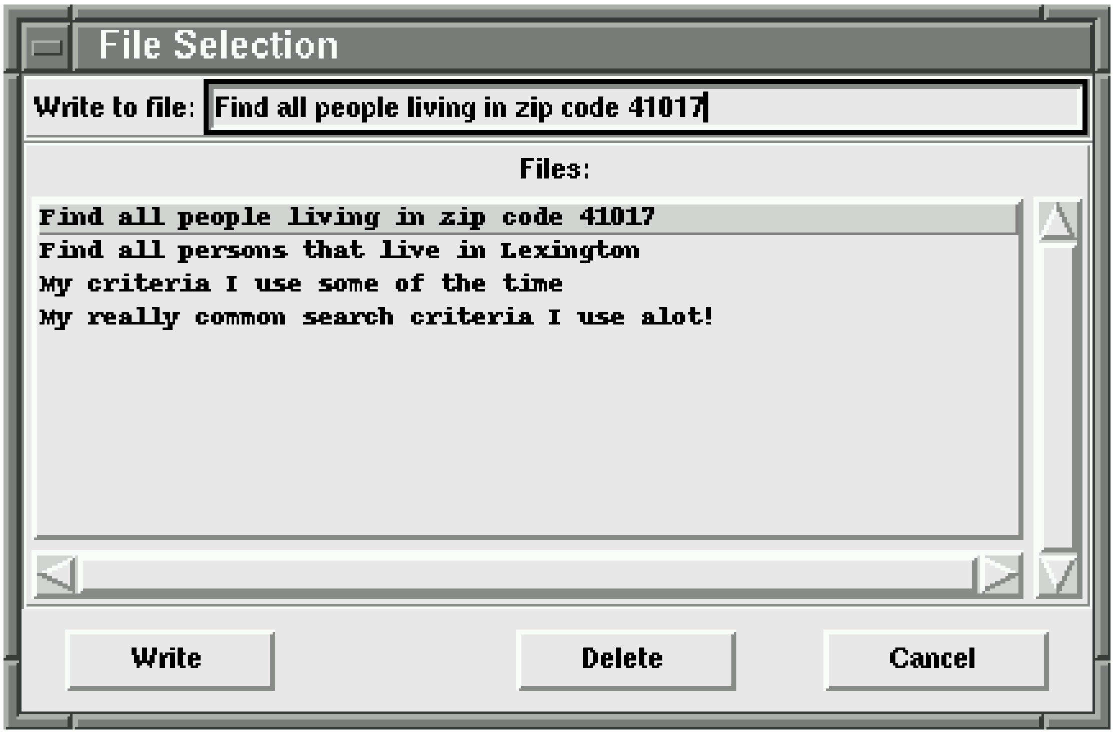


Figure 4.3: Template Write Dialog Box

4.3.2 Steps to retrieving a search template

Follow these steps to retrieve a search template:

1. While in Search Mode, click the Templates/Read Template menubutton.
2. A dialog box(see figure 4.4) will appear showing the current templates that you've previously created.
3. Click the name of the search template file that you'd like to use. The row will become highlighted.
4. Click the *Select* button.
5. At this point, the template dialog will be removed and the search fields will be populated.

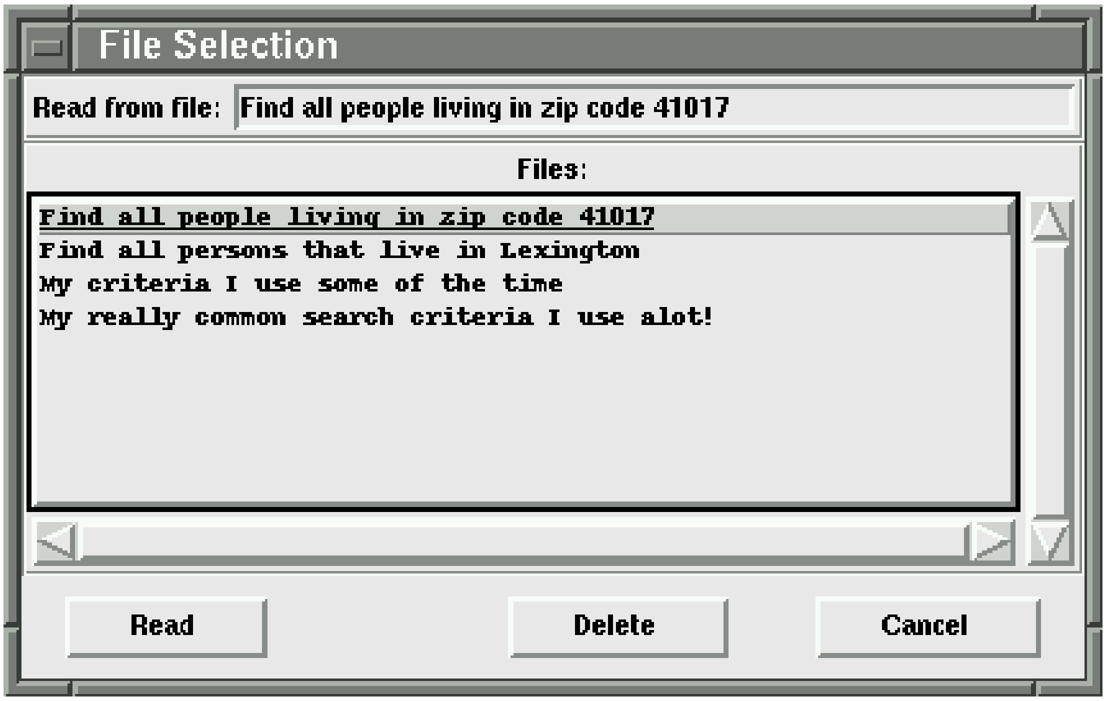


Figure 4.4: Template Read Dialog Box

Note, you do not have to clear the fields prior to reading a search template. When you retrieve a search template, all the fields are set to the values that were present when you first created the search template.

4.3.3 Insert Last Template

nxqddb provides a convenient method to retrieve the most recently used search template. To get the most recent template, just click the *Templates/Insert Last Template* menubutton.

After you click this menubutton, the fields will be populated with the values that belong to the template that was used most recently.

4.3.4 Deleting a search template

nxqddb provides the ability to delete search templates. You may decide that you no longer need to use a template that you created. If so, then you may wish to remove the template so that your list of search templates does not get cluttered with files that you no longer need.

Follow these steps to delete a search template from the hard disk:

1. From any Mode (Search, Add, Change, or Read-only) choose either *Templates/Read Template* or *Templates/Write Template*.
2. A dialog box will appear(see figure 4.4) showing the currently defined search templates.
3. Click the search template row that you would like to delete.
4. The row will become highlighted.
5. Click the *Delete* button
6. At this point, the row will be removed from the listbox and the search template will be permanently removed from the hard disk.

4.4 Customizing a view and/or a search results dialog box

nxqddb provides the ability for you to customize the look and feel of any expandable field's view dialog box. There are reasonable defaults for all expandable fields; so, you may never need to customize these view dialog boxes. However, if for some reason you have a need to change a particular view box, *nxqddb* provides a friendly and powerful interface to accomplish your customizations.

You can also customize the search results dialog box. User's find that they customize the search results dialog box more frequently than they do a view dialog box. The techniques to customizing both the view dialog box and the search results dialog box are exactly the same. The only difference between the two is how the configuration is displayed:

- To display the configure window for a view dialog box, you press and hold the *Control* key while you click the right mouse button over the expandable field label.
- To display the configure window for a search results box, you click the *Configure/Search Results* menubutton.

You should see a window similar to the one shown in figure 4.5 if you chose to configure the Address view dialog box. And, you should see a window similar to the one shown in figure 4.6 if you clicked the *Configure Search Results* within the people window. Notice how these two windows are identical except for the contents of the listboxes. Hence, once you learn this one window, you've mastered the technique for configure both the view dialog box and the search results dialog box.

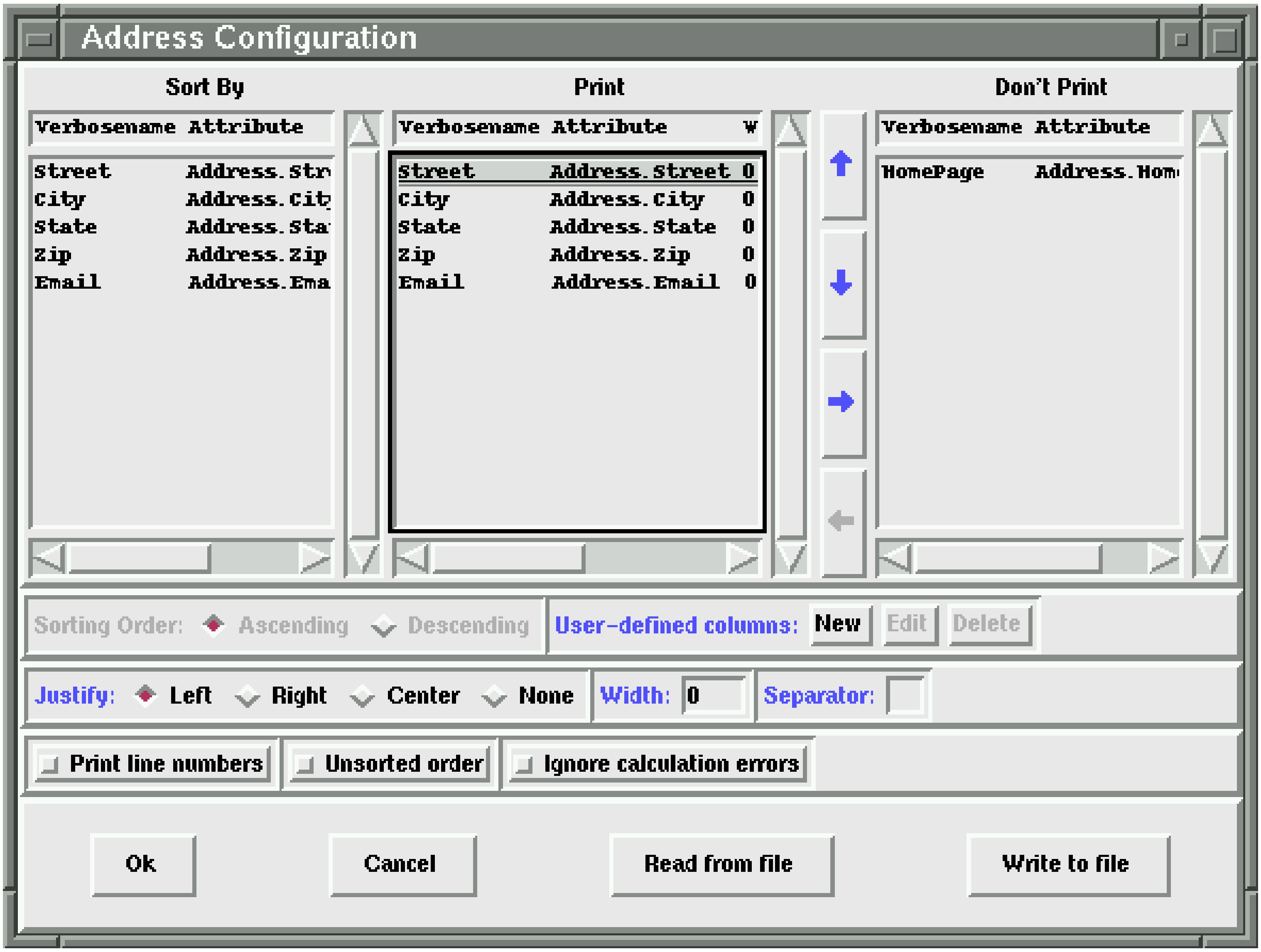


Figure 4.5: Configure view dialog box

From this configuration window, you can specify the following:

Sort Order You can choose the sorting order of the displayed rows.

Ascending or Descending You can specify the field to be sorted Ascending or Descending.

Fields to Print You can choose which fields to display in the view dialog box. Some expandable items may contain many fields; hence, you can decide which fields are important for you to view.

Fields not to print *nxqddb* provides a *Don't Print* listbox with all the fields that are currently not being displayed in the view dialog box. You can easily move fields from this listbox to the Print listbox and vice versa.

Width You can specify the maximum width of each column. If you specify a zero width, *nxqddb* will automatically set the column width to the widest field value. By setting a reasonable width, you can avoid having view dialog boxes that are too wide for your screen.

Justify You can tell *nxqddb* to left or right justify the field values within the column.

Separator By default, *nxqddb* uses the space character to separate the field values. However, you can tell *nxqddb* to use a different character if you so desire.

Unsorted You can tell *nxqddb* not to sort the rows in the view dialog box.

Read/Write from file You can tell *nxqddb* to read a previously saved configuration or write a new configuration to a file.

4.4.1 Sorting

You can specify the sort order of the fields displayed in the view box. To rearrange the sort order, click the field that you want to be sorted first. The field will become highlighted.

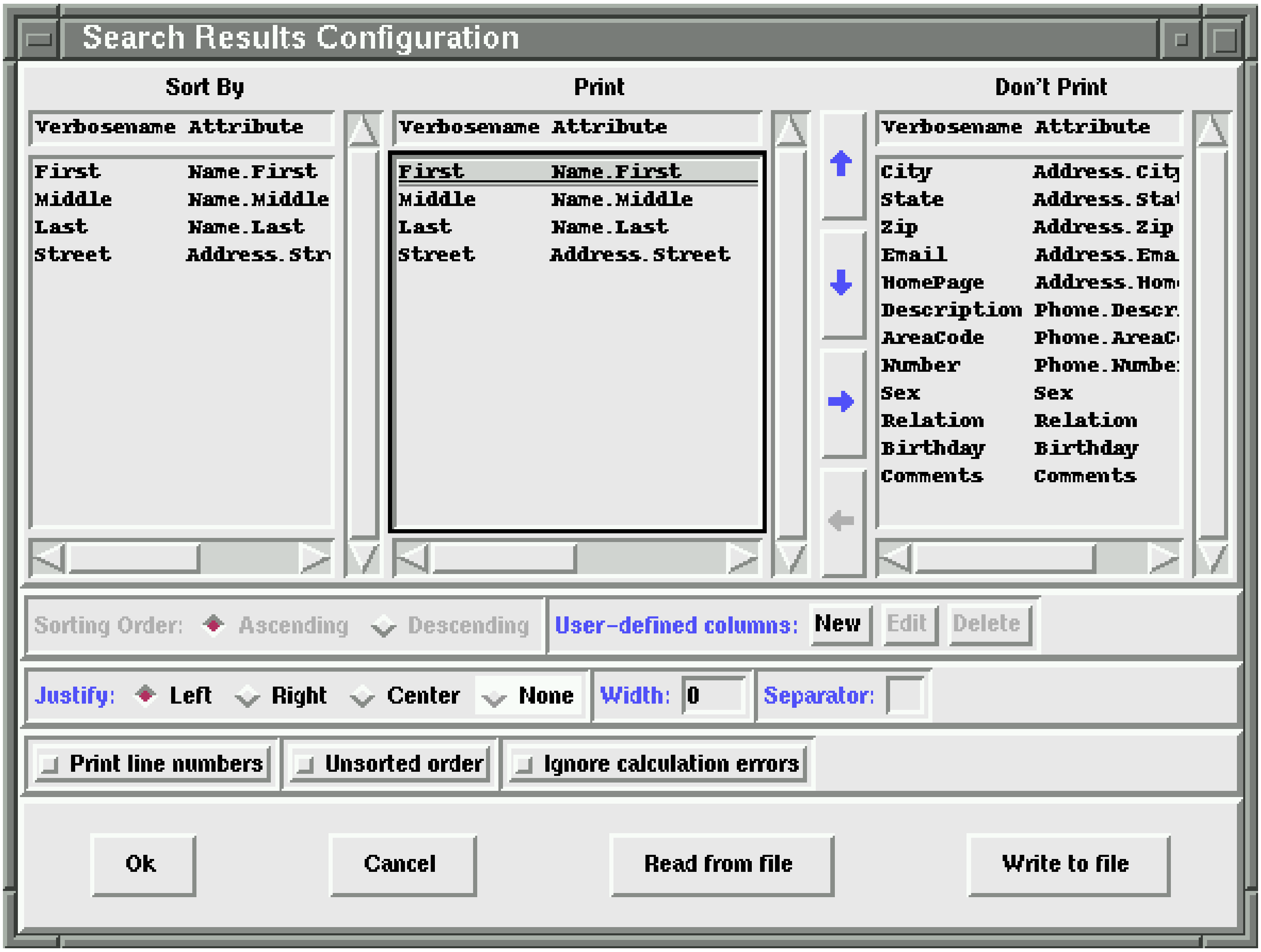


Figure 4.6: Configure search results dialog box

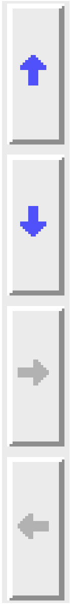


Figure 4.7: Configure view dialog box arrows

Now, simply click on the blue arrows to position the field at the top of the list box.

> Blue arrows are active; grey arrows are inactive. To make an arrow active, you must first select a row from the appropriate listboxes. To make the up and down arrows active(or blue), choose a row from the *Sort by* listbox. To make the left and right arrows active, choose a row from the *Print* and *Don't Print* listboxes, respectively.

nxqddb sorts the rows in the view dialog box based on the position of the field in the list box. For example, in figure 4.8, *nxqddb* will first sort on *Date Due*, then *Code*, *Description*, *Number of reminders sent* and finally *Last Done*.

Figure 4.8: Ascending or Descending sort order

Each field can be sorted in either ascending or descending order. By default, a field will be sorted in ascending order. To toggle to ascending or descending, click the appropriate radiobutton as shown in figure 4.8.

4.4.2 Printing

You can specify what fields you want to display in the view box; the display fields are those fields that are listed in the *Print listbox*.

4.4.3 Adding a display field to the view dialog box

To display a field in the view dialog box, do the following:

1. From the *Don't Print* listbox, select the desired field. The field will become highlighted and the disabled grey left arrow will become blue highlighted.

2. Click the blue left arrow. At this point, the field will be removed from the *Don't Print* listbox and added to the *Print* listbox.

3. Click the Ok button to save your results.

4.4.4 Removing a field from the view dialog box

To remove a field from being displayed in the view dialog box, do the following:

1. From the *Print* listbox, select the desired field to be removed from the view dialog box. The field will become highlighted and the right arrow will become blue highlighted.

2. Click the blue right arrow. At this point, the field will be removed from the *Print* listbox and added to the *Don't Print* listbox.

3. Click the Ok button to save your results.

4.4.5 Choosing the width of a column in the view dialog box

You can choose the width of each field. A zero width informs *nxqddb* to make the column width large enough to display the widest field value. Zero is the default width; however, you can change this width by entering a number in the width entry (see figure 4.9).

Figure 4.9: Column Justification

4.4.6 Choosing left or right justification

The *Justify* radiobuttons allow you to justify the values in each column to be aligned on the left or right column margin. For example, for currency fields such as balance, you would likely choose right justification. Most other fields would likely be left justified. For this reason, the default is for fields to be left justified. To specify a field justification:

1. Choose a field in the *Print* listbox. The Justify label will become blue highlighted to let you know that you can now click on either the left or right radio button.

2. Click the *Left* radiobutton to left justify the field; or, choose the *Right* radiobutton to right justify the field.

3. Click the Ok button to save your results.

4.4.7 Choosing a column separator

By default, *nxqddb* separates the columns in the view dialog box with a blank space. By using a space character, the view dialog box columns are easier to read. However, if for some reason you would like to choose a different separator, follow these simple steps:

1. Choose a field in the *Print* listbox. The Separator label will become blue highlighted to let you know that you can now enter a separator character in the entry.

2. Enter a separator character in the entry.

3. Click the Ok button to save your results.

4.4.8 Reading/Writing Configuration Files

You can store your expandable field view dialog configuration(s) to files. Storing configuration files is useful if you change your view dialog box setup frequently.

☞ **EXAMPLE**: For example, suppose you have setup a database for managing your pet information. Perhaps you have 2 ways that you like to view a person's pets. Maybe the first way is to view the *Animal Number* and *Pet Name* sorted by the *Animal Number*. And, the second way is to view the *Pet Name*, *Species* and *Rabies Tag Number* sorted by *Tag Number*. Each of these two views have different columns and a different sorting order. Rather than setting up a new configuration each time you swap between these two different views, you can store both configurations to a file and read the desired configuration when you need it.

To **store** a view dialog configuration, click the *Write to File* button.

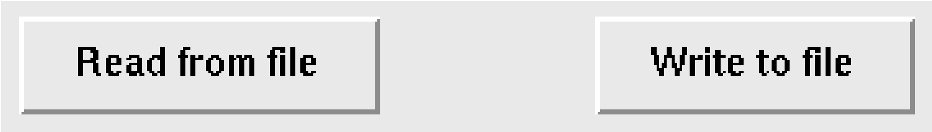


Figure 4.10: Read/Write buttons

The standard File Save dialog box will appear as shown in figure 4.11. This window allows you to type a new filename to store the configuration. Or, you can choose an existing configuration file to replace with the new configuration.

The three button operations from this window are:

Write Saves the current configuration to the filename specified in the *Write to file* entry.

Delete From this window, you can remove any configuration files that you no longer need. To remove a configuration file, press the *Delete* button on the *Write to file* window.

Cancel If after pressing the *Write to File* button, you decide not to save your new configuration, you can cancel your operation by choosing the *Cancel* button on the *Write to file* window.

To **read** a previously saved configuration, you click the *Read from File* button. The standard *Read from File* dialog (see figure 4.12) will be displayed.

From this window, you can select from a list of previously stored configurations. Unlike, the write operation, you must select a configuration file from the listbox. After you've made a selection you click the *Select* button to retrieve this configuration.

The three button operations from this window are:

Read Reads the selected configuration file and updates the configuration to match these settings.

Delete From this window, you can remove any configuration files that you no longer need. To remove a configuration file, press the *Delete* button on the *Write to file* window.

Cancel If after pressing the *Read from File* button you decide not to save your new configuration, you can cancel your operation by choosing the *Cancel* button on the *Write to file* window.

4.5 Expert Searches

In addition to the searches discussed so far, *nxqddb* provides a powerful mechanism to perform any complex search on your practices database.

This mechanism is named **Expert Search**.

Figure 4.11: Writing a view dialog box configuration file

4.5.1 Steps to performing an expert search

To perform an expert search, you follow these steps:

1. Determine which window you need to search from. As explained earlier in this chapter, you can perform searches in any window except for the Setup window.

2. Be sure you're in Search Mode. Remember, to go to search mode, click the *Modes/Search Mode* menubutton.

3. Click the *Edit/Expert Search* menubutton.

4. At this point, the Expert Search window will display for you to create an expert search. Details on how to create an expert search are discussed in this section.

5. If you want to store your expert search criteria to a file to be used over and over at a later date, you can click the *Write to File* button to save your expert search.

6. Once you've completed you're expert search, you click the *Search* button.

Figure 4.12: Reading a previously saved configuration file

7. At this point, *nxqddb* will perform your search and display the search results.

The Expert Search window is shown in figure 4.13.

4.5.2 A simple expert search

Lets walk through an example of a simple expert search. For this example, lets assume we want to find all the people whose first name starts with the letter "A". There are two ways that this can be accomplished. First, you could just do a simple search by 1) populating "a.*" in the First Name field or 2) populating "aa-az" (a word range search). However, we want to learn how to do an expert search so we'll take this simple search and comprise an expert search.

Displaying the expert search window

The first step to creating an expert search is to display the expert search window. From search mode, click the "Edit/Expert Search" menubutton to display the Expert Search window(see figure 4.13).

> The Expert Search window is a modal dialog. Modal dialogs require that you complete your work within the window before doing anything else. Meaning, while the Expert Search window is displayed, you will not be able to do any other operation outside of the Expert Search window.

Since the Expert Search window is modal, you'll notice that if you move your mouse back to the *nxqddb* window the cursor will change to an hourglass. This visual clue lets you know that the Expert Search is a modal window. Once you're finished with the Expert Search window, you'll be able to work in the *nxqddb* window once again.

Choosing your search field(or attribute)

Click the choicebox (or "Down Arrow") pushbutton as shown in figure 4.14. Clicking this icon will display a popup window(see figure 4.15) to allow you to:

- Choose the field that you want to search upon
- Choose the search type, one of: (1) Parse, (2) Word, (3) Word Range, (4) Numeric, (5) Numeric Range, (6) Date, (7) Date Range, and (8) Regular Expression.
- Delete a node. (More about what a node is in the next section)
- Expand a node.

Next, we need to choose the "First name" field (or attribute).

> Attribute is another name for a field. Attributes are database fields that store values. Examples of attributes are a person's name, address, pet name, rabies tag number and so on.

To specify the attribute, click the "down arrow" to the right of the attribute label; doing so will display the attribute listbox (see figure 4.16) for you to select the First name attribute.

☞ **HINT**: Use the horizontal and vertical scrollbars to view any text that is hidden from view.

Choosing the Search Type

In our example, we want to perform a word-range search on the first name field. So, you should click the "Word range" radiobutton.

Entering the search values

After you've chosen the search field(or attribute), and have clicked the search type radiobutton(e.g. Word-range), position the cursor in the entry boxes to enter your word range. Type the text "Aa" in the first entry box. Then, press the tab key to position the cursor in the rightmost entry box. Now type the text "Az".

> Remember, *nxqddb* searches are case insensitive. So, you can type in Aa, AA, aa, or aA and *nxqddb* treats all of these entries the same. That is, *nxqddb* will return instances that match any of the above patterns.

At this point your expert window should resemble the one shown in figure 4.17

Notice how the two topmost entries "Current search type" and "Current attribute" are updated to reflect the values chosen for the current field. The current field is the field that contains the cursor.

Saving your expert search

Click the "Write to file" pushbutton to save the expert search that was just created. The standard "Write to file" dialog box will be displayed. See section 4.4.8 for details on how to store this search to a file.

Reading a previously stored expert search

At any time after saving an expert search to a file, you can retrieve that search and save some time from manually keying in the search. To retrieve a previously stored expert search, you click the "Read from file" button. See section 4.4.8 for details on how to use the standard *nxqddb* read/write dialog box.

Executing the Search

Click the "Search" button to invoke the expert search. After you click the "Search" button, *nxqddb* will start retrieving the instances that match your search criteria. In this example, *nxqddb* will return all the people that have a first name starting with the letter "A". The Expert Search modal dialog window will disappear and you will be returned to the *nxqddb* window. Note, while *nxqddb* is performing the search, you may see a *watch* cursor and the status bar may read "Searching ...".

> The watch cursor is a visual clue that *nxqddb* is working. You must wait until the watch cursor changes back to an arrow cursor before continuing your work.

Clearing an expert search

At any time, you can erase(or clear) your current expert search by clicking the "Clear" pushbutton. Note, this button only clears the current session. If you've retrieved the expert search from disk, the disk file will **not** be removed. If you want to remove the search from the disk file, you must click the "Delete" button from either the *Read from file* or *Write to file* dialog boxes(see section 4.4.8).

Documenting your expert search with comments

nxqddb allows you the opportunity to store comments with each expert search that you store to a file. Since expert searches can get complicated, you may wish to enter comments explaining what the search accomplishes. To enter comments, just type any text in the Comments text entry box prior to clicking the "Write to file" pushbutton. Your comment text will be retrieved into the comments text entry box the next time you retrieve your expert search.

4.5.3 Extending the simple expert search example

In the previous section, we created a simple expert search that returned all the people whose first name began with the letter "A". Now, lets add to our search criteria.

Suppose, we also want to find those people whose last name starts with the letter "B". That is, we want all people whose first name starts with an "A" OR whose last name starts with a "B" Meaning, suppose our records contained the following data as shown in table 4.3.

In our search we want to produce the matches as illustrated in table 4.4.

First Name	*Last Name*
Alvin	Smith
Roger	Bennet
Tim	Carr
Allen	Boone
Dudley	Doolittle

Table 4.3: Input data for expert search example

First Name	*Last Name*
Alvin	Smith
Roger	Bennet
Allen	Boone

Table 4.4: Results from expert search example

Notice, the above entries contain **either** a first name starting with the letter "A" **or** a last name starting with the letter "B". In the case of *Allen Boone*, this instance meets both criteria.

> The search criteria discussed in this extended example can only be accomplished using the Expert search. You can not get the same results using a simple search; because, when two fields(e.g. First and Last Name) are populated, *nxqddb* will take the intersection. This search example takes the union of two different search field values.

Appending a node

In the first expert search created, the **node** was:

- First name field
- Word-range search type
- First name attribute

Now, we want to append to this node the criteria to also include last names starting with a "B".

To do this, click the "Append Node" pushbutton. A brand new node(see figure 4.18) will display with a default operation of **And**. That is, to the right of the "Down Arrow" icon, you'll see the word **And**.

However, in our search, we do not want an **And** operation; instead, we want to perform an **Or** operation. We'll see how to change this operation when we enter our second search node – that is, all names starting with the letter "B".

Entering second node information

Click the choicebox "Down Arrow" pushbutton to the right of the "And" label to display the window shown in figure 4.19.

From the window shown in figure 4.19, perform the following:

- Choose the **Or** radiobutton to denote an Or operation. See how *nxqddb* automatically updates the screen to show that you've changed the operation from **And** to *Or*. Try clicking other operations to verify that indeed the screen is updated. Return the operation to be equal to "Or" to carry on with our exercise.
- Click the Word-range search type radiobutton
- Click the attribute "Down arrow" pushbutton and then choose the Last Name field from the list of available attributes.
- Click the "Dismiss" button to return to the second node's text entries.
- Key in "Ba" in the leftmost bottom entry on the screen.
- Tab to the rightmost bottom entry and key in "Bz".
- Tab to the Comments text entry box and enter some comments: Find people whose first name starts with "A" or whose last name starts with "B".

At this point, you should have a window that looks like the one shown in figure 4.20.

Viewing the Search Node information

At any time within the Expert Search window, you can view or modify an existing node. To view the search type and search attribute, position the mouse in the desired field; the top two read-only entries "Current search type" and "Current attribute" will become populated to show the values.

You can also click the "Down Arrow" choicebox button to the left of the node's entry boxes to view or modify the node's logic operation, search type or search attribute.

Deleting a node

Suppose you accidentally click the "Append node". There are two ways to remove this unwanted node. You can click the "Clear" button to start over. This may not be a good move; especially, if you've already created several nodes.

The alternative is to *Delete a node*. To delete a node, you click the "Down Arrow" pushbutton to the left of the node that you want removed. The node's edit window will display to allow you to modify the contents. To actually remove the node, just click the "Delete" pushbutton. After doing so, *nxqddb* will remove the node from the Expert Search window and you may continue with your work.

Expanding a node

The "Expand node" pushbutton allows you to **group** search nodes. Suppose for example that you would like to perform the following query: Find all cats and dogs but NOT ones named **Gus**. Here's the steps to creating this search using the expand node:

1. Create the search node for cats. Choose the Species attribute and enter a Word search. Click the "Dismiss" pushbutton. Key the value **cat** in the text entry.

2. Now, click the "Down Arrow" button. On the window, click the "Expand node" pushbutton button. At this point, a new node will be generated on the screen and **grouped** with the node you've just entered (i.e. the **Cat** search node).

3. Click the "Down arrow" on the bottommost node and populate like so: (1) Choose **Word** search type, (2) Choose **Species** attribute. and (3) Choose **Or** logic radiobutton.

4. Click the dismiss button. Key in Dog in the bottom left text entry.

5. Now, click the bottommost "Append node" pushbutton.

6. Populate the third and final search node like so: (1) Choose the **Not** logic radiobutton, (2) Choose **Pet Name** attribute and (3) Click the dismiss button, (4) Key in **Gus** in bottommost entry.

7. Tab to the comments field and enter an appropriate comment describing this query.

8. At this point, you're ready to execute or save the search criteria.

The resulting expert search window should look like the one shown in figure 4.21.

Dismissing the Expert Search Window

The expert search window will be automatically dismissed when you click the "Search" pushbutton.

> The Dismiss button will remove the expert search window and return you to the main window where you launched the expert search operation. Be sure to save your search results (if you so desire), before choosing the dismiss button.

Alternatively, you may cancel your expert search at any time by simply clicking the "Dismiss" button.

4.5.4 Unlimited Search Capability

The two expert search examples discussed in this chapter have been relatively simple. However, using the expert search tool, you have the capability to perform any type of complex search. In doing so, you may have virtually unlimited amount of search fields and levels of search nodes. However, as the number of search fields and levels increase, so does the complexity of your search. For example, reference figure 4.22 for an example that contains a more complicated expert search than the two we've seen so far in this chapter.

4.5.5 Learn the expert search by experimentation

The best way to master expert searching is through experimentation. Start off with simple searches and progress into more complicated searches. With a little time and effort, you'll quickly become familiar and will appreciate *nxqddb*'s expert search technique.

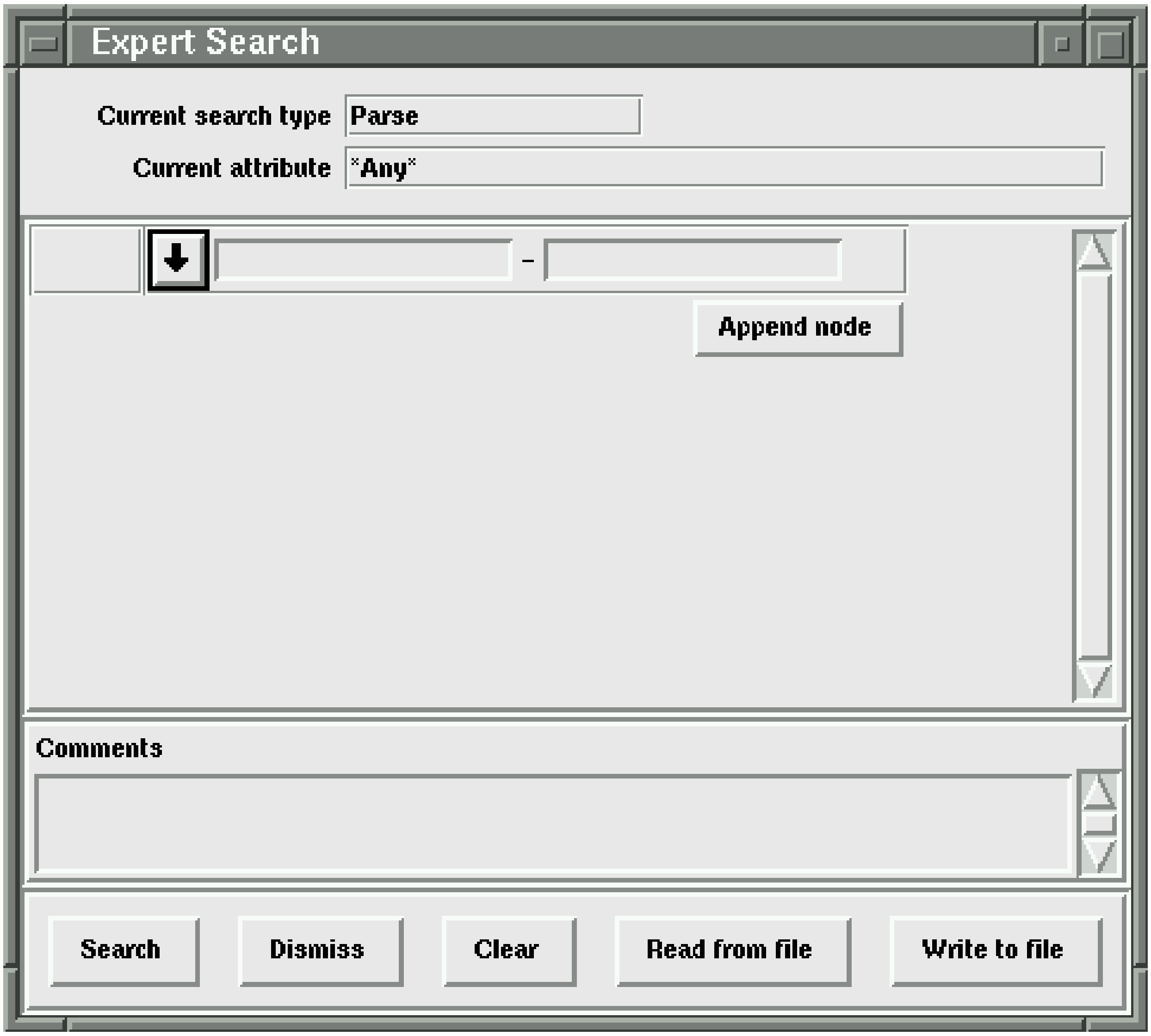


Figure 4.13: Expert Search Window

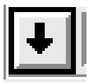


Figure 4.14: Attribute Choicebox button

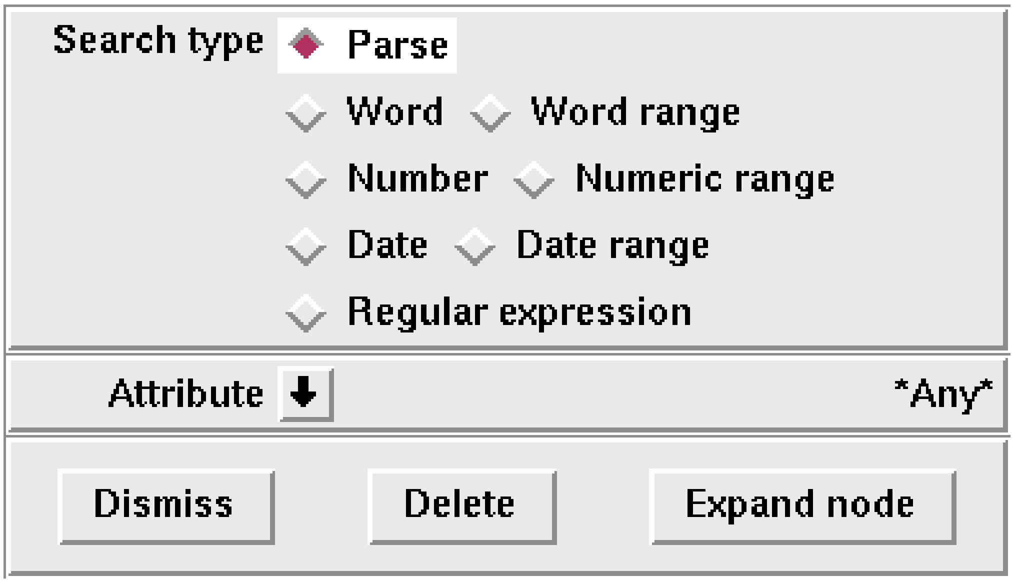


Figure 4.15: Window to choose attribute(or field)

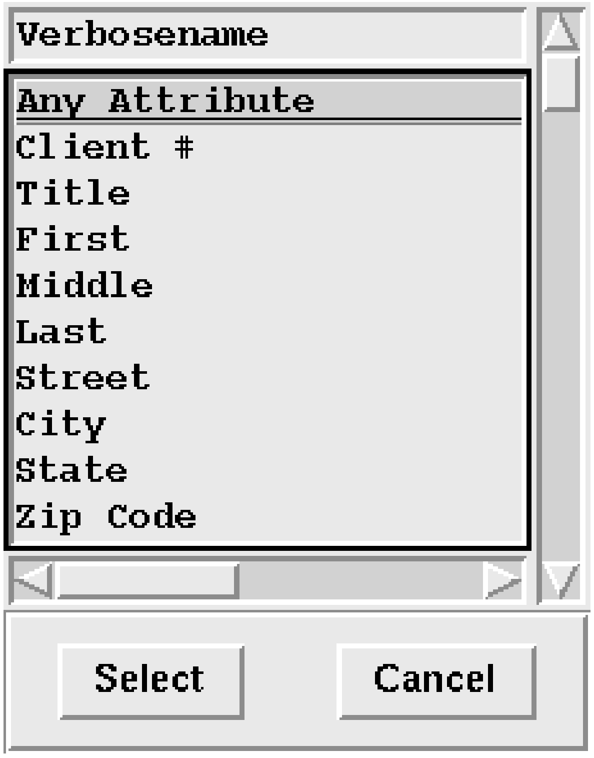


Figure 4.16: Selecting an Attribute (or Field)

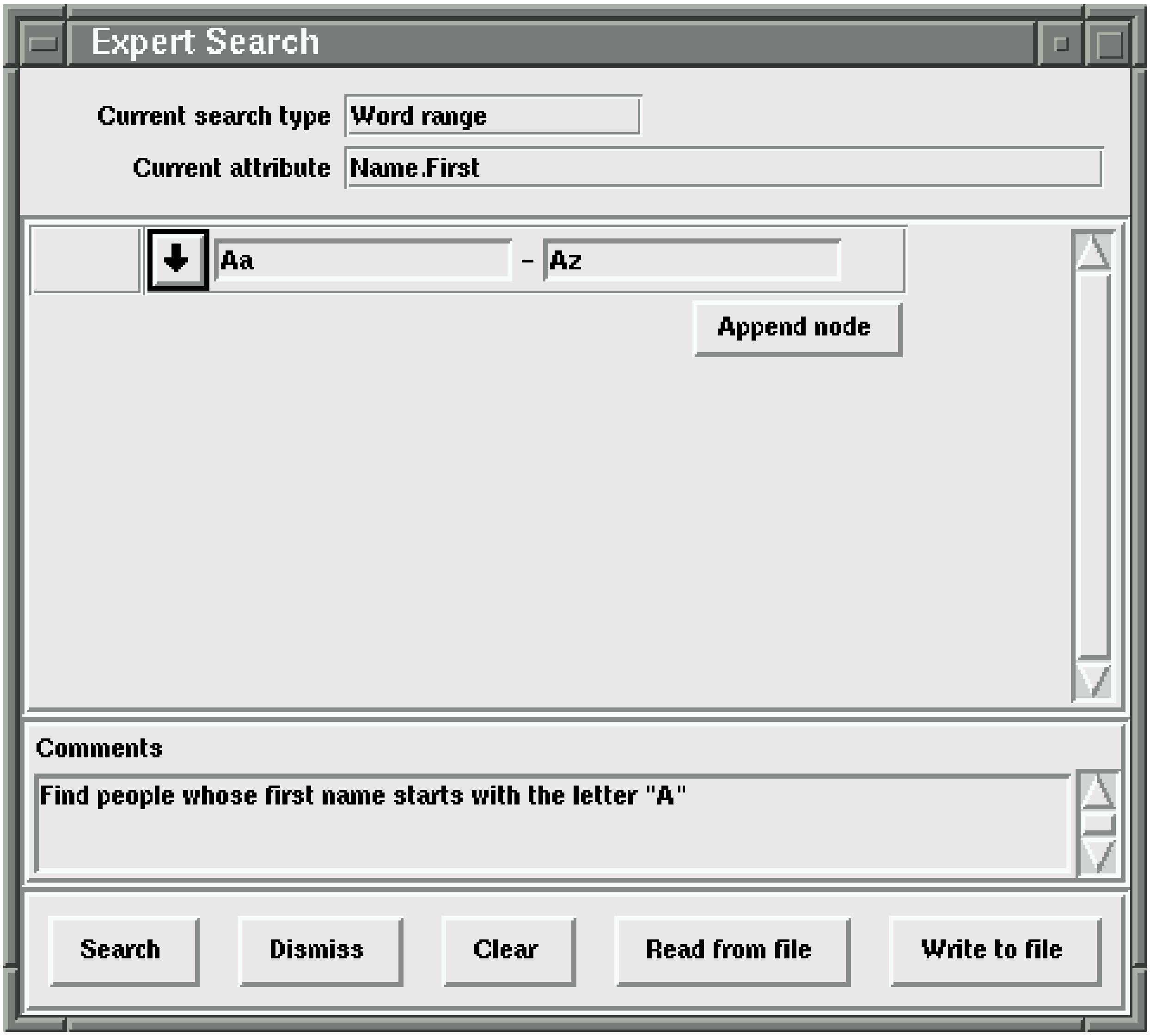


Figure 4.17: Sample Expert Search - Find people whose first name begins with an A

Figure 4.18: Default Append Node

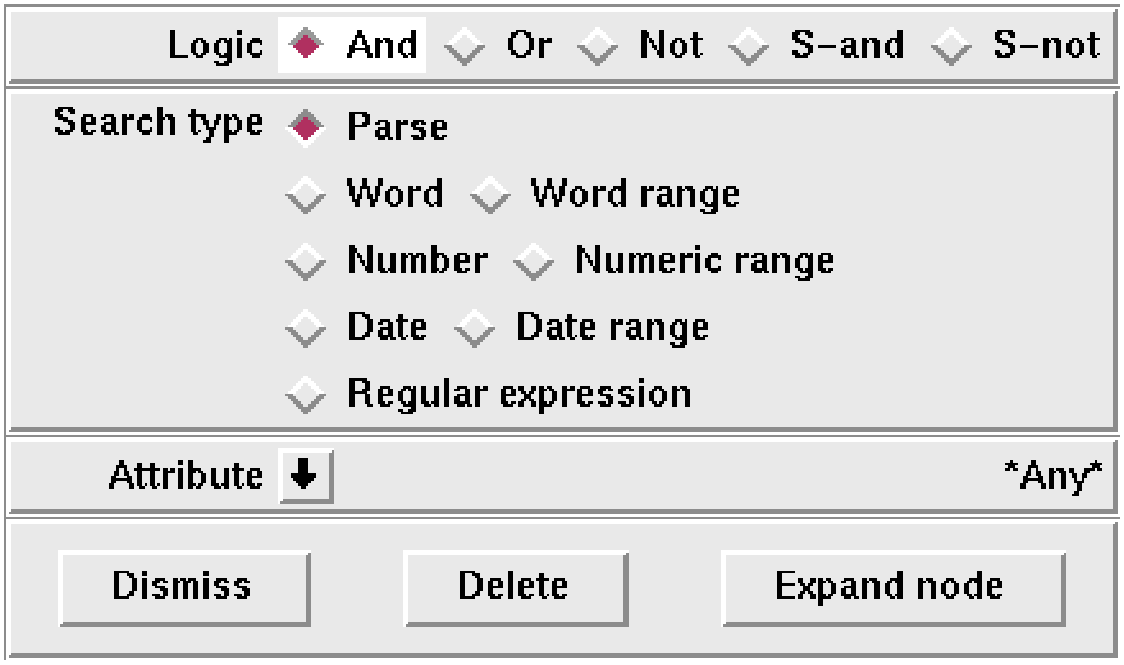


Figure 4.19: Defining the second search node

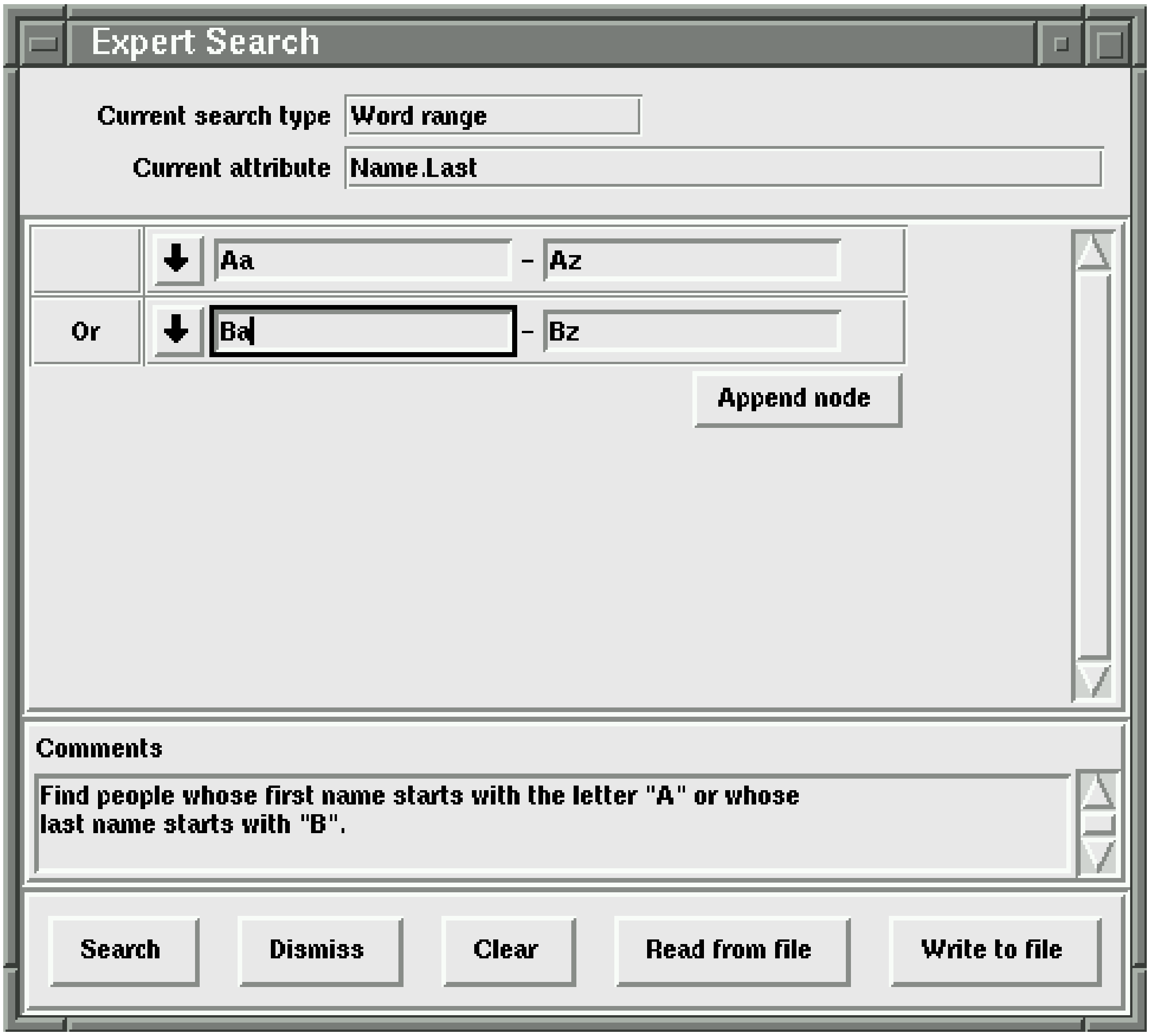


Figure 4.20: Extended Expert Search Example

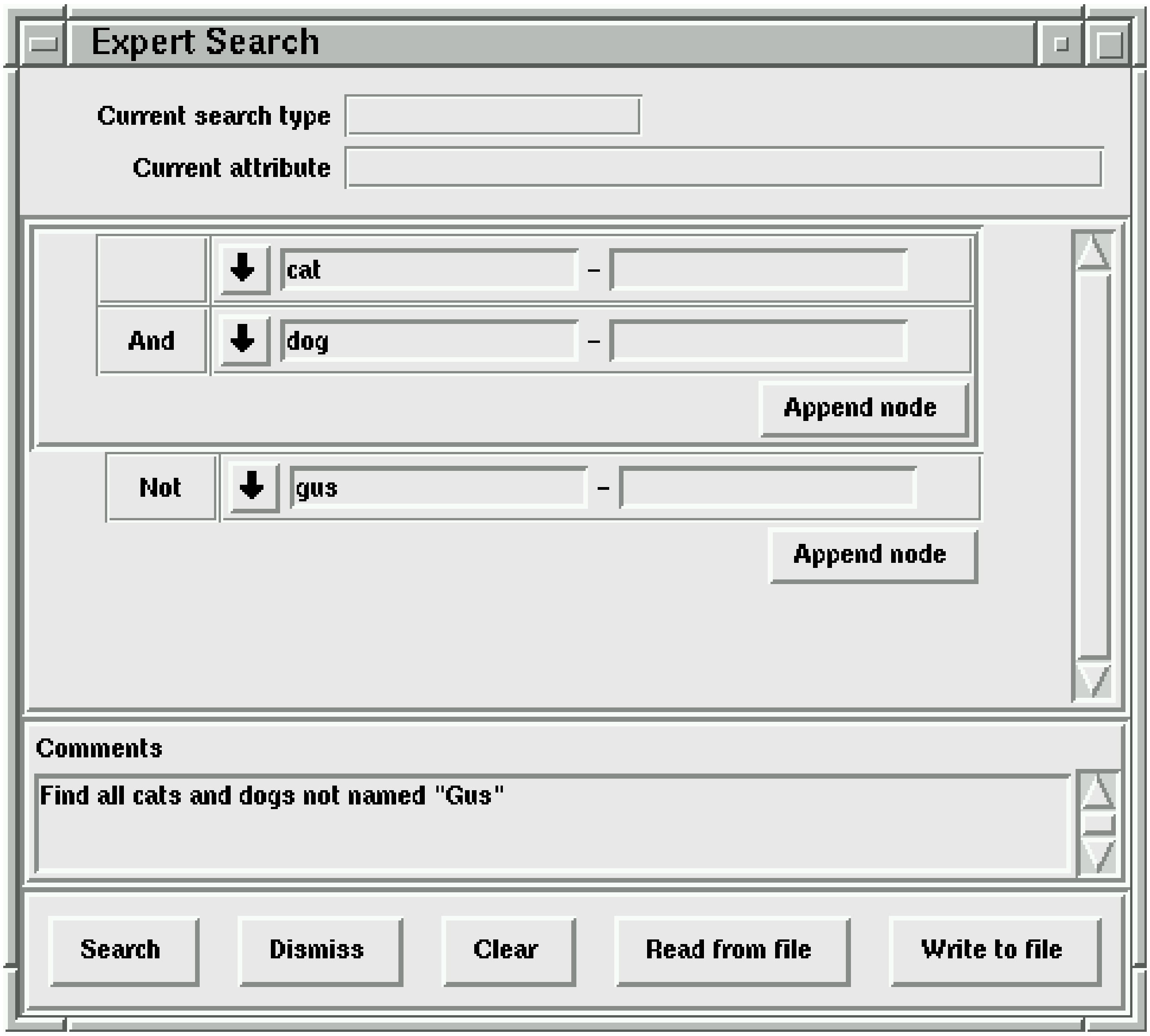


Figure 4.21: Grouped Expert search using Expand node

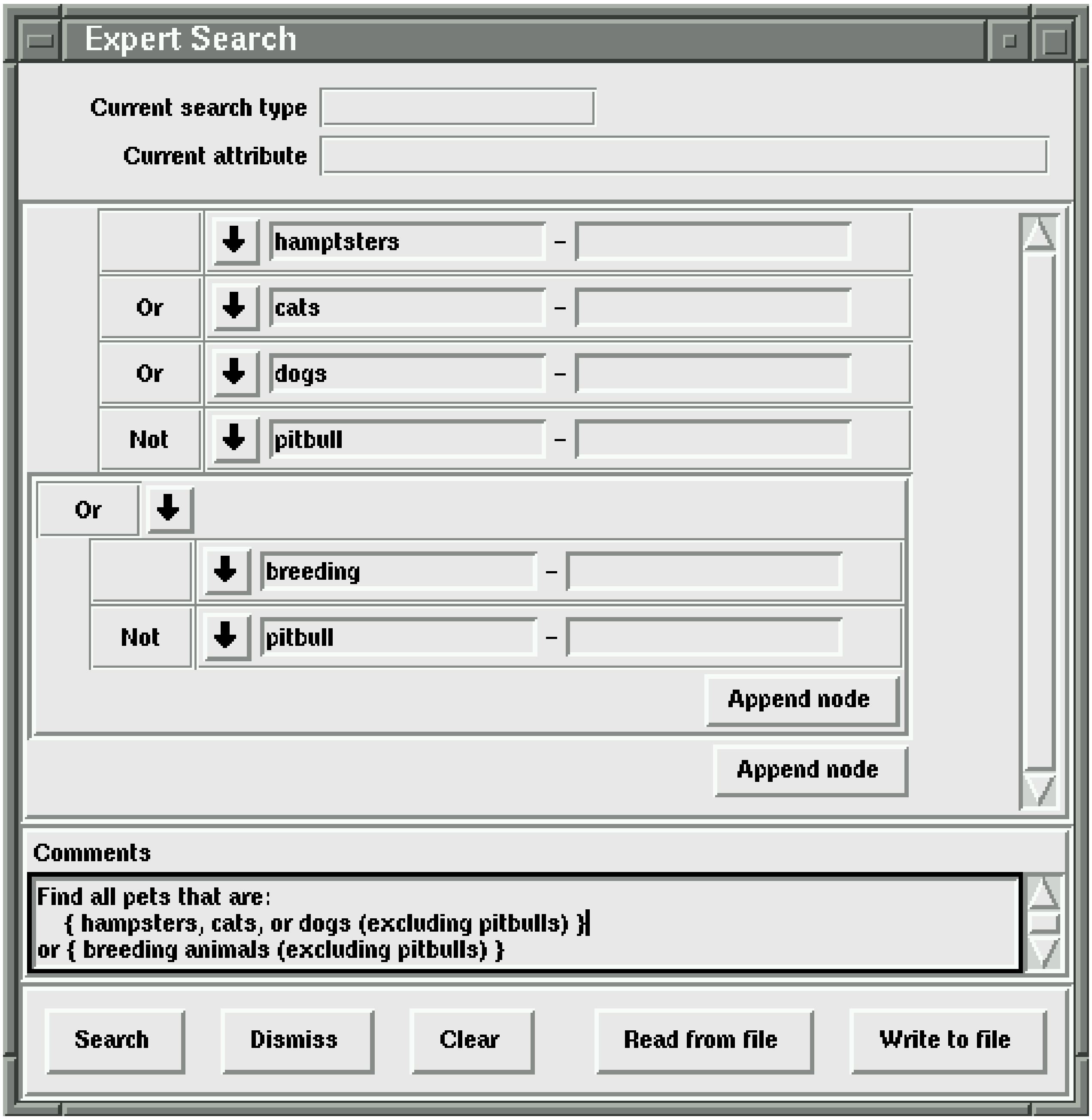


Figure 4.22: More complicated expert search

Chapter 5

Configuring *nxqddb*

You can easily configure many *nxqddb* options to suit your individual preferences. Each of the *nxqddb* configure options is discussed in this chapter.

5.1 Configuring Individual Fields

To configure a particular field, position the mouse over the field's label and then `<Control-3>` click the mouse. That is, press and hold the control key and then click the third mouse button. When you `<Control-3>` click on a field label, a field configuration window will be displayed as shown in figure 5.1.

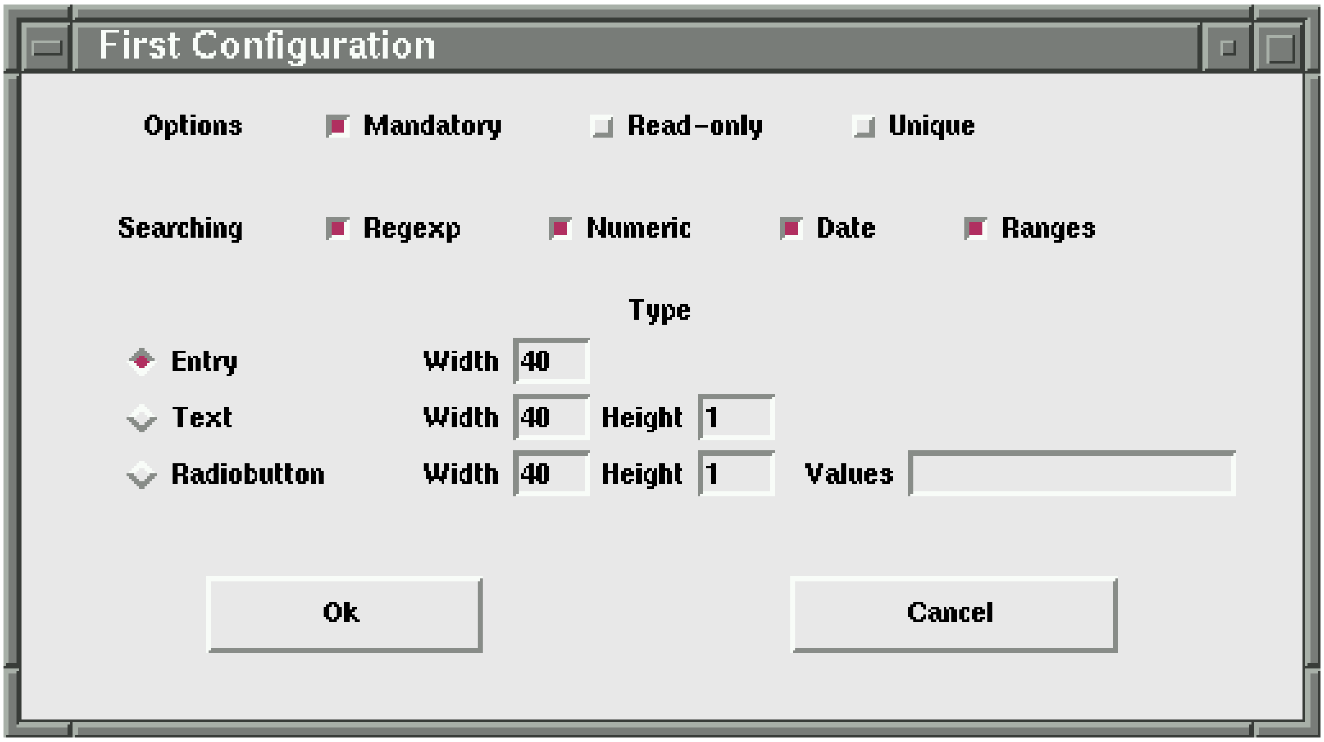


Figure 5.1: Field Configuration Window

5.1.1 Setting the add mode options

From this window you can set the following Add Mode options:

Mandatory If checked, the field must contain text before adding or modifying a record.

Read-only If checked, the user will not be able to add or modify the field's value.

Unique If checked, *nxqddb* will prevent you from entering a value that has already been saved in the database. For example, suppose you made the first name field a unique field, and had previously saved a person named "Gilbert". The next time you try to add someone named "Gilbert", *nxqddb* will display a dialog box as shown in figure 5.2. You will not be allowed to leave the field until you enter a non-unique name.

5.1.2 Setting the searching options

The "Searching" options include:

Figure 5.2: Unique Value Dialog Box

Regexp Turns on/off regular expression searching capability.

Numeric Turns on/off numeric searching capability.

Date Turns on/off date searching capability.

Ranges Turns on/off range searching.

For more details on *nxqddb* searching capability, please reference chapter 4.

5.1.3 Setting the type of widget

You can also decide to change the type of widget from this configuration window. From the configuration window(see 5.1), you have the option of making the field one of:

Entry If you choose entry, you can also set the width of the entry.

Text In addition to specifying a width, you can also set the height of the text area. For example, if you specify a height of five(5), then *nxqddb* will display a text area that will visibly display five rows of data. Of course, you can use the scrollbars to view text that is beyond the view area.

Radiobutton Like the text widget, you can specify the width and height. Additionally, you specify the radiobuttons for the user to select from. You enter these radiobutton values in the "Values" entry. For example, let's suppose you want to change the "Sex" field from an entry to a radiobutton, you would do these steps:

1. Click the "Radiobutton" radiobutton.
2. Enter **Male Female** in the Values entry.

3. Click the Ok button.

You must specify a valid width. That is, you must enter a valid number. If you do not specify a valid width upon leaving the configuration window,

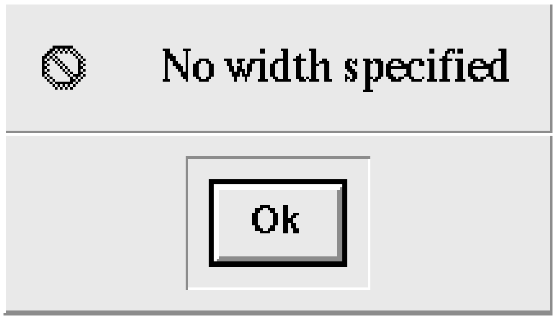


Figure 5.3: No width warning dialog box

nxqddb will display the dialog shown in figure 5.3.

5.2 Saving/Loading your configuration settings

In *nxqddb*, you have three options to save your configuration settings to your hard disk:

Personal configuration stores all the nxqddb settings in your user id's `.nxqddb_conf` directory using the file name convention

`~/.nxqddb_conf/.yourhomedirectory.yourdatabase`

The settings stored in your "Personal" configuration override global and default settings.

Global configuration stores all the nxqddb settings in the relation's directory. So, if you have a database in /u/yourhome/YourDatabase, the global configuration settings will be stored in /u/yourhome/YourDatabase/.nxqddb_conf These global settings override global settings but do not override the personal configuration settings.

Default configuration stores only the common settings(e.g. fonts, colors) that can be shared across all qddb databases. The default configuration settings are stored using the file name convention

`~/qddb_fxrc`

where the tilda represents your home directory.

You must choose the *Save* menubutton from one of the above options in order for nxqddb to write your current session's configuration settings to the hard disk. Or, optionally you can turn on the "Auto-save" feature so that when you exit your session nxqddb will automatically store your session's configuration settings. Once you have saved your settings, the next time you start nxqddb, you can load the previously stored settings. To save/load your settings, choose one of the options beneath the "Configure" menubutton shown in figure 5.4.

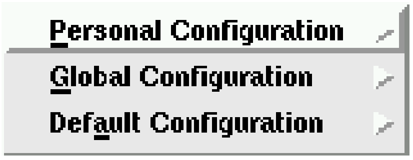


Figure 5.4: Saving/Loading Configuration Settings

At startup, *nxqddb* loads the configurations in the following order:

1. Default configuration settings
2. Global configuration settings
3. Personal configuration settings

Hence, settings from the *Global* configuration(if present) override any *Default* settings. And, the *Personal* configuration settings (if present) override both global and default settings.

5.3 Auto-Save Personal Configuration

After you make configuration changes, you must explicitly tell *nxqddb* to save your changes before you exit the window. If you forget to save your settings *nxqddb* will disregard your changes — unless of course, you have the *Auto-Save* turned on. Many users find this checkbutton feature a nice option to keep turned on; especially if you're the kind of user who frequently makes configuration changes and forgets to save your results.

5.4 Search Results

Please reference section 4.4 for details on this configuration option.

5.5 Report Format Defaults

Clicking the "Configure/Report Format Defaults" menubutton displays the cascading menu shown in figure 5.5.

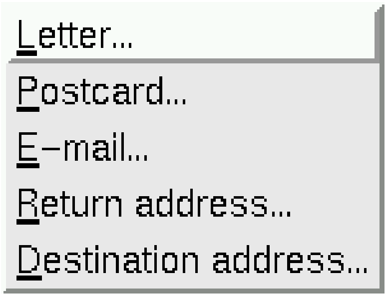


Figure 5.5: Menu for setting report format values

From the menu shown in figure 5.5 you can access the screens that tell *nxqddb* how to:

Format letters Choose the format for your letters. For example you can specify (1) the number of lines, (2) address location, etc.

Format Postcards Similar to the letter setup window, you can customize the appearance of your postcards.

Handle email Tell *nxqddb* what attribute holds the email address and what mail program to use on your system.

Return address Tell *nxqddb* what return address to include on postcards and letters.

Destination address Tell *nxqddb* what destination address to include on postcards and letters.

Each of these options are discussed in detail in the following sections.

5.5.1 Letter Format

Choosing the "Configure/Report Format Defaults/Letter format" menubutton displays the window shown in figure 5.6.

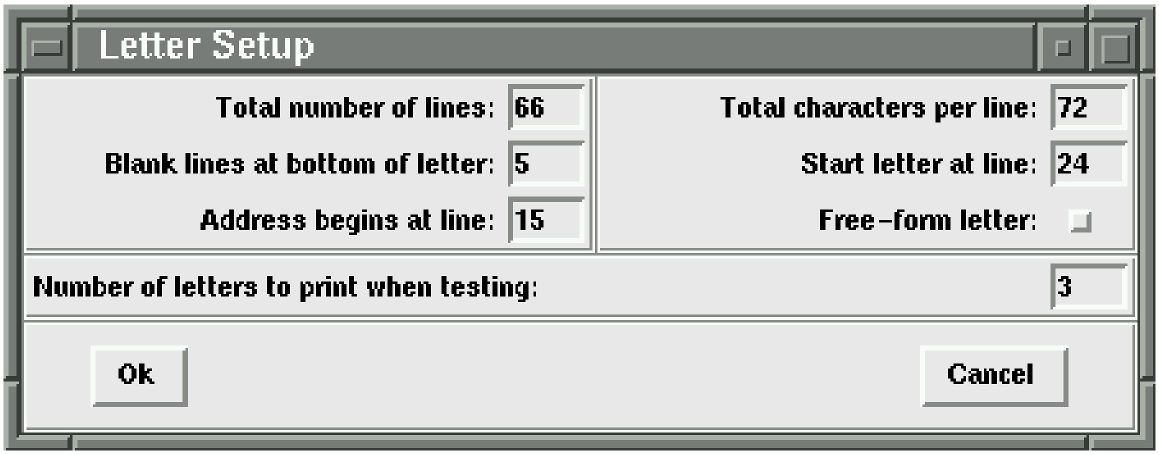


Figure 5.6: Window to customize appearance of letters generated by *nxqddb*

Each of the fields shown in the reminder card setup window is described below:

Total number of lines This field tells *nxqddb* how many lines are contained on the letter page.

Blank lines at bottom of letter This field tells *nxqddb* how many lines to leave blank at the the bottom of the letter page.

Address begins at line This field tells *nxqddb* on what line the destination address should begin printing.

Total characters per line This field tells *nxqddb* how wide your letter is; in other words, this field tells *nxqddb* how many characters fit on a line.

Start letter at line This field tells *nxqddb* at what line to start the letter. In other words, the value of these field tells *nxqddb* how many blank lines should appear at the top of the letter.

Free-form letter This checkbutton tells *nxqddb* to produce *free-form letters* when the checkbutton is turned on.

Number of letters to print when testing This field tells *nxqddb* how many letters should be printed when the user clicks the *Test* button from the Report Generation window. This number should be a small number, usually less than five or so. This feature is nice because you can generate a test print run to check your letter setup and to verify that the printouts are working correctly.

After you've entered your setup information, you can click the *Ok* button to save your setup or click the *Cancel* to cancel without saving your setup.

5.5.2 Postcard Format

Choose the "Configure/Report Format Defaults/Postcard Format" menubutton to display the window shown in figure 5.7.

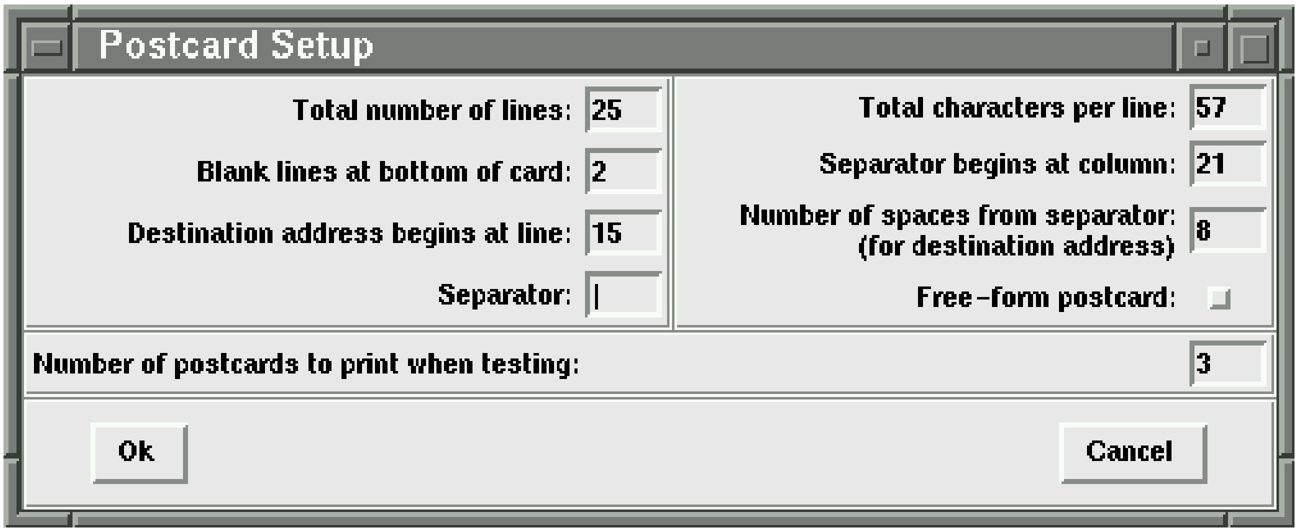


Figure 5.7: Window to customize appearance of postcards generated by *nxqddb*

The options to formatting postcards are similar to the options of formatting letters. Each of the fields shown in the postcard setup window is described below:

Total number of lines This field tells *nxqddb* how many lines are contained on the postcard.

Blank lines at bottom of card This field tells *nxqddb* how many lines to leave blank at the the bottom of the postcard.

Destination address begins at line This field tells *nxqddb* on what line the destination address should begin printing.

Separator The separator is a character that separates the postcard text from the destination address. *nxqddb* defaults the separator to a vertical bar(—).

Total characters per line This field tells *nxqddb* how wide your card is; in other words, this field tells *nxqddb* how many characters fit on a line.

Separator begins at column This field tells *nxqddb* where to place the vertical separator on the reminder card.

Number of spaces from separator(for destination address) This field tells *nxqddb* how many spaces to place to the right of the separator before printing the destination address.

Free-form postcard This checkbutton tells *nxqddb* to produce *free-form postcards* when the checkbutton is turned on.

Number of postcards to print when testing This field tells *nxqddb* how many postcards should be printed when the user clicks the *Test* button from the reports window. This number should be a small number, usually less than five or so. This feature is nice because you can generate a test run of reminder printouts to check your postcard setup and to verify that the printouts are working correctly.

After you've entered your setup information, you can click the *Ok* button to save your setup or click the *Cancel* to cancel without saving your setup.

5.5.3 Email Format

Choosing the "Configure/Report Format Defaults/E-mail format" menubutton displays the window shown in figure 5.8.

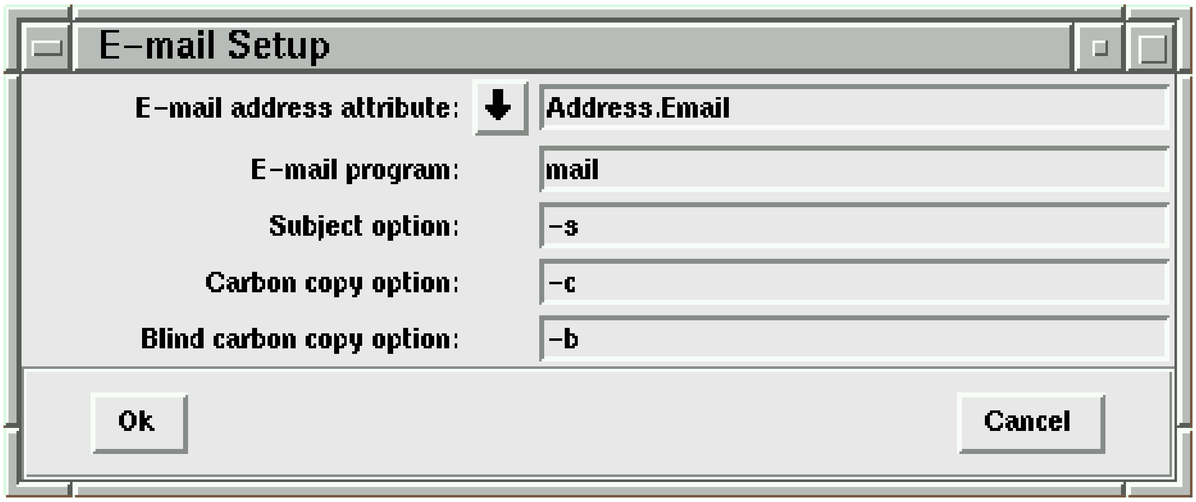


Figure 5.8: Window to customize E-mail settings

Email address attribute

From the *E-mail Setup* window, you have the ability to specify which attribute contains the Email address(es) that you want to receive mail generated from *nxqddb* reports. To specify an attribute, you simply key the fully qualified attribute name in the text entry. Or, you can click the “down arrow” button to the left of the entry to choose from a list of

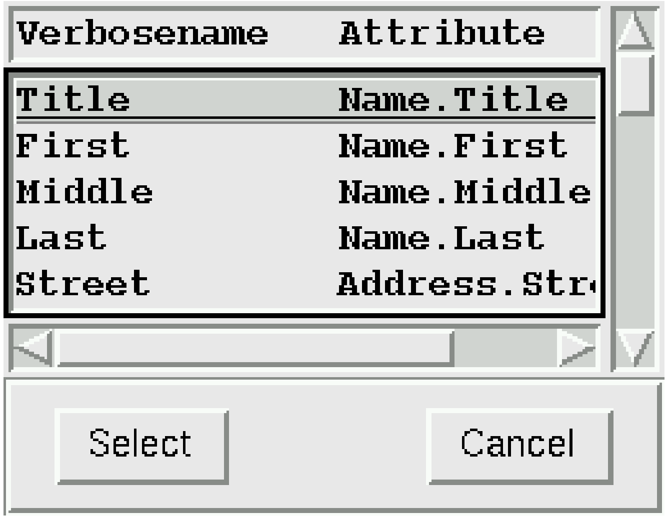


Figure 5.9: Pulldown listbox of available attributes to choose from

attributes(see figure 5.9). Typically, this attribute is usually called something like "Email" or a similar name that you used when you created your Schema file.

E-mail program

You can tell *nxqddb* which mail program you want to be launched whenever you create Email correspondence reports. Typically, most users will be okay with the default *mail* program; however, you can use whatever mail program you desire.

E-mail subject, carbon copy, blind carbon copy options

This field tells *nxqddb* what switch option your mail program uses for the subject content. The default value is "-s"; this value works just fine for the standard UNIX mail program. If your mail program expects a different switch option for specifying the subject, you would enter that value here. At the time you run your Email report, *nxqddb* will prompt you for the subject and content of the mail message(s) along with the carbon copy and blind carbon copy distribution list.

If you click the "Ok" button, your settings will be remembered for that session. Remember, you must choose "Configure/Save Personal Configuration" or "Configure/Save Global Configuration" to permanently remember your new settings.

If you click the "Cancel" button, your modifications will be disregarded.

5.5.4 Setting your return address

Choosing the "Configure/Report Format Defaults/Return address" menubutton displays the window shown in figure 5.10.

Figure 5.10: Window to customize your return address

From this window, you simply enter the text for a return address. For example, suppose you live on Hermes Avenue in Covington, Kentucky; then you might enter the following text in the window's text entry and then choose the "Ok" button to save your results:

```
John and Jane Doe
3205 Hermes Avenue
Covington, Kentucky  41016
```

Remember, you must choose "Configure/Save Personal Configuration" to permanently save your settings.

5.5.5 Setting your destination address layout

You can tell *nxqddb* the attributes that comprise your destination address for your reports. To accomplish this, choose the "Configure/Report Format Defaults/Destination address" to display the window shown in figure 5.11. There is text contained on this window to instruct you how to complete your destination address entry. Simply perform the steps listed in this window to tell *nxqddb* how to generate and print the destination addresses found from producing an *nxqddb* report. For details on how to generate reports, please reference chapter 7 *Generating Reports.*

View Attributes and aliases

nxqddb provides a convenient method for you to complete your return address. Following these steps makes it easy for you to enter your destination address layout and also reduces the chance of you making any typing mistakes:

1. From the *Destination Address Setup* window, click the "View Attributes and aliases" button. The window shown in figure 5.12 will be displayed to allow you to view all of the attributes defined in your database.

2. For each field in your destination address, do these steps: (1) Click the attribute in the "View Attributes and Aliases" window (the row will become highlighted), (2) Position the mouse in the "Destination Address Setup" window where you want to place the attribute and then click the middle mouse button to paste the attribute into the entry.

3. After completing your destination address, choose the "Ok" button to tell *nxqddb* to save your setting. Remember, you must choose "Configure/Save Personal Configuration" to permanently save your settings when you leave *nxqddb*. If you choose "Cancel", *nxqddb* will disregard your modifications to the destination address.

5.6 Fonts

Clicking the "Configure/Fonts" menubutton displays the "Font Configuration" window shown in figure 5.13. You can individually customize a particular widget type font's (1) family, (2) slant, (3) weight, and (4) Size. To set all the widget fonts together, follow these steps:

1. Set the Default font's family, slant, weight and size values. To set these values, left-click on the word. For example, from figure 5.13, clicking on the Default font's Family value will display a popup window for you to choose a font family (see figure 5.14). Repeat this action for the slant, weight and size attributes.

2. Click the "Set all fonts to default" pushbutton. Notice that all the widget fonts are immediately set to your chosen default font.

3. At this point, click the Ok button to save your settings and return to your *nxqddb* window.

At any time, you can choose the "Cancel" button to return to the *nxqddb* window with your fonts unchanged.

☞ **HINT**: You'll notice that there are only two options available for the Listbox' family selection: (1) courier, and (2) fixed. These two fonts have a fixed character size (that is, each character occupies the same width) so that listbox columns will always be aligned.

5.7 Colors

To change to a different background color, choose the "Configure/Colors" menubutton. The window shown in figure 5.15 will be displayed. From this window, you can choose from one of the "Symbolic Color Names" from the listbox; or, you can set your own color by adjusting the individual Red, Green or Blue sliders. This window contains the following buttons:

Ok saves your current selection.

Cancel aborts the operation.

Apply applies your selection to the *nxqddb* window. Note, you must click "Ok" to save the change.

Revert sets the color prior to opening the "Color Configuration" window.

5.8 Setting Soft and Hard Search Limits

There are two limits that you may set for your searches:

Soft limit This limit causes *nxqddb* to display a warning dialog box(see figure 5.19) if the number of matches found from your search exceeds the specified limit.

Hard limit This limit tells *nxqddb* to abort the search operation if the number of matches found exceeds the specified limit. A dialog box(see figure 5.20) will be displayed to tell you that you've exceeded your hard search limit. Of course, if you really need to view a search that exceeds your current limit, you can reset your limit to a higher (or unlimited) amount and repeat your search.

5.8.1 Soft Limit

The Soft Limit is useful to warn you that perhaps your search criteria is too general. That is, the results of the search may return more rows than you care to view. You have the option to continue on with the search, or abort the search to return back to your search window.

To set soft or hard limits, you press the "Configure/Soft limit" or the "Configure/Hard limit" menubutton as shown in the pull-down menu of figure 5.16.

Notice that both the "Configure/Soft limit" and "Configure/Hard limit" both have a small "right arrow" to the right of the menubutton. This arrow serves as a visual clue that the option is a cascading menu. Clicking the "Hard Limit" and "Soft Limit" menubuttons display the cascading menus shown in figures 5.17 and 5.18.

> A cascading menu is a menu that contains one or more options that when clicked displays another menu.

When you click the "Configure/Soft limit" menubutton, you're presented with a cascading menu that contains the limits ranging from 50 to 500, and an unlimited option.

If you choose **unlimited**, then there will be no soft limit. That is, no warning dialog box will be displayed, regardless of how many matches are found from your searches.

For example, clicking the 400 radiobutton will instruct *nxqddb* to display a warning box(see figure 5.19) prompting you to continue or abort the search.

5.8.2 Hard Limit

The hard limit is useful to interrupt searches that may take a long time to complete. Unlike the soft limit search, you will not be able to continue after the hard limit dialog box is displayed(see figure 5.20). Instead, after clicking the "Ok" button, you will be returned back to the window from which you launched the search.

After receiving a hard limit dialog box, you have two options:

- Perhaps your search was too general; maybe you can narrow your search by entering more specific search criteria.
- Alternatively, you could increase your hard limit.

5.8.3 Typical Settings for the Soft and Hard Limits

Since *nxqddb* searches are fast, you typically don't want to set too low of a limit for either soft or hard limits. The soft limit is a more friendly type of event since the resulting dialog allows you to continue the search or to cancel; whereas, the hard limit requires you to abort the search.

Also, the hard limit should be higher than the soft limit. If the hard limit is not higher than the soft limit, then you will never see the soft limit warning dialog. *nxqddb* first checks to see if the hard limit is exceeded before prompting you with the soft limit. And, if the hard limit is exceeded, then you will be prompted with the hard limit dialog box.

Whatever value you choose for your soft limit is totally up to your preference. However, you may find that setting the limit too low quickly becomes tedious because you'll be prompted more frequently. Since, *nxqddb* searches are so fast, we recommend setting a soft limit of 500 (the default setting.)

A common setting for soft and hard limit values is shown below:

- Soft limit equal to 500

- Hard limit equal to 4000

5.9 Keyboard Traversal Options

By default, pressing the navigation keys(e.g. tab, shift-tab and up/down arrows) automatically skips the (1) scrollbars, (2) add, (3) view, and (4) del buttons. However, *nxqddb* provides checkbuttons to allow you to configure whether or not you want these widgets to be included in the tabbing sequence. To activate these widgets, click the appropriate checkbutton from the "Configure/Keyboard Traversal" menu as shown in figure 5.21. If any of the keyboard traversal checkbuttons are on (i.e. the checkbutton square is red), then *nxqddb* will not skip over the widget when you tab to it. Most users tend to like the default option — scrollbars, add, view and del buttons turned off.

☞ **EXERCISE**: Try this experiment to see the behavior of the keyboard traversal. First, be sure that the keyboard traversal is turned off for the scrollbars, add, view and del buttons. Then position the cursor in the topmost entry and tab through all the fields. Notice how the focus does not stop on the scrollbar or the pushbuttons. Now, turn on the keyboard traversal for the Add and View buttons. Then, tab through the fields again. Notice how the focus visits the Add and View buttons since you turned on the keyboard traversal for these buttons.

5.10 Date style option

You can specify one of two date style options by clicking one of the radiobuttons from the "Configure/Date style" menu as shown in figure 5.22. The two choices are described below:

US format displays the date in the month/day/year format. For example, "January 28, 1996" would be displayed as "01/28/96".

European format displays the date in the day/month/year format. For example, "January 28, 1996" would be displayed as "28/01/96".

> Note, the "Date style" option is available only in "Search mode". So, you must go to "Search mode" in order to change the date style. The "Date style" option remains stippled in modes other than "Search mode".

The environment variable shown in figure 5.23 tells *Qddb* how to format your dates.

The environment variable overrides the compiled default(you can set the compiled default during Qddb installation). The user's selection overrides the environment variable setting.

5.11 Auto-Clear option

If you have Auto-Clear turned on (see figure 5.24), *nxqddb* will automatically clear the entries when you save your entry. The Auto-Clear menubutton will have a red square when the option is turned on.

In most cases, you'll want to have Auto-Clear turned on. However, if most of the information is duplicated among your entries, you'll likely want to turn Auto-Clear off so that you can save some typing.

The best way to see how Auto-Clear works is to try adding records with (1) Auto-Clear turned on and then with (2) Auto-Clear turned off and observe the different behavior.

5.12 Auto-Save Tuples option

nxqddb provides an "Auto-Save Tuples" option that is turned off by default. If you turn "Auto-Save Tuples" on, *nxqddb* will not prompt you with a confirmation dialog box when you exit add mode with unsaved changes. Instead, *nxqddb* will go ahead and save your record. Most users prefer to leave the "Auto-Save Tuples" off. To set the "Auto-Save Tuples" option, choose the "Configure/Auto-Save Tuples" menubutton(see figure 5.24).

5.13 Force Field Formatting

This option, when turned on, tells *nxqddb* to automatically format the entries when you tab out of an entry. For example, date fields can accept certain keywords(e.g. today, yesterday, etc.). If you have Force Field Formatting turned on, *nxqddb* will automatically convert "today" to the current day's date (e.g. 12/25/96 if it was Christmas, 1996) when you tab out of the field.

Many users choose Force Field Formatting turned on as their default setting because *nxqddb* immediately displays the date before saving your entry.

5.14 Force Read-Only Mode

If turned on, the Force Read-Only Mode will automatically cause *nxqddb* to switch to Read-Only mode when the user selects a search result. In this mode, you will not be allowed to modify the record. You can tell that you're in Read-Only mode by observing that the status bar is updated as shown in figure 5.25.

Also, the "Modes/Change Mode" option is disabled. To go from Read-Only mode, you must first turn off the "Configure/Force Read-Only Mode" checkbutton; doing so will enable the "Modes/Change Mode" menubutton and therefore allow you to switch to change mode.

By default, the Force Read-Only Mode checkbutton is turned off.

If you have another *nxqddb* program running that currently has an entry in change mode, this entry can not be modified by other *nxqddb* programs. Instead, the other *nxqddb* programs will "view" the record in change-mode.

☞ **EXERCISE**: Try this exercise to see that only one *nxqddb* window can have a certain entry in change mode at one time. Bring up two "*nxqddb* AddressBook2" programs. First, find an entry in change mode. Then, in the second *nxqddb* program, select the same record. In the second *nxqddb* program, the record will be in "Read-Only Mode" and you will not be able to go into change mode until after the first *nxqddb* program either (1) switches to Read-Only mode or (2) goes into search mode or exits. To verify this is indeed the case, change the first *nxqddb* program to Read-Only mode; at this point, you'll be able to go to change mode within the second *nxqddb* program.

This powerful locking feature is handy in that it prevents other users from editing the same record at the same time.

5.15 Entry Widths

nxqddb defaults the entry widths to a length of 30 characters. There are two ways to change an entry's width:

- You can choose the "Configure/Entry Widths" menubutton and select from one of the 10 through 90 radiobuttons.
- You can set the entry width by `<Ctrl-3>` clicking(hold the ctrl key while clicking) the mouse on the text entry's label. Doing so will display a window for you to set various options — one of which is to reset the length of the field. For details on this individual field configuration, see section 5.1.

Choosing a radiobutton from the "Configure/Entry Widths" will immediately update the display to set the widths to the length you specify.

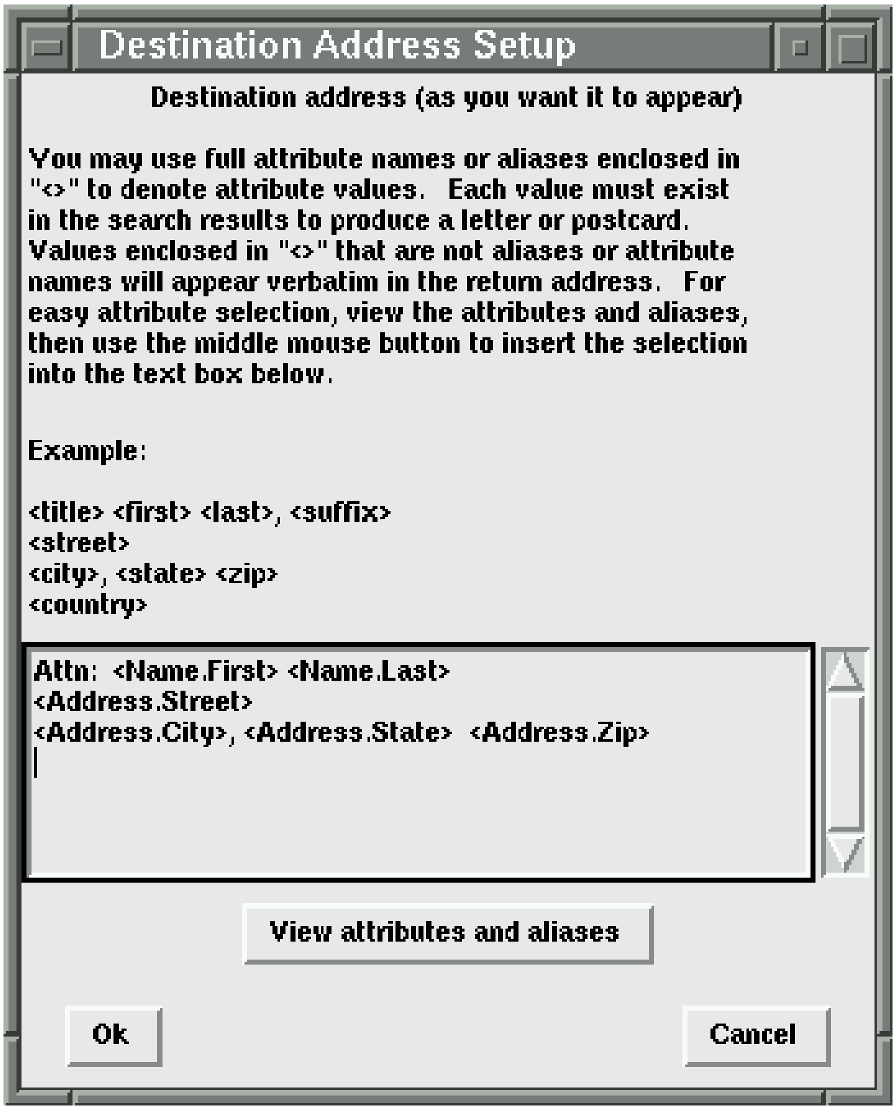


Figure 5.11: Window to customize your destination address

Figure 5.12: View Attributes and Aliases Window

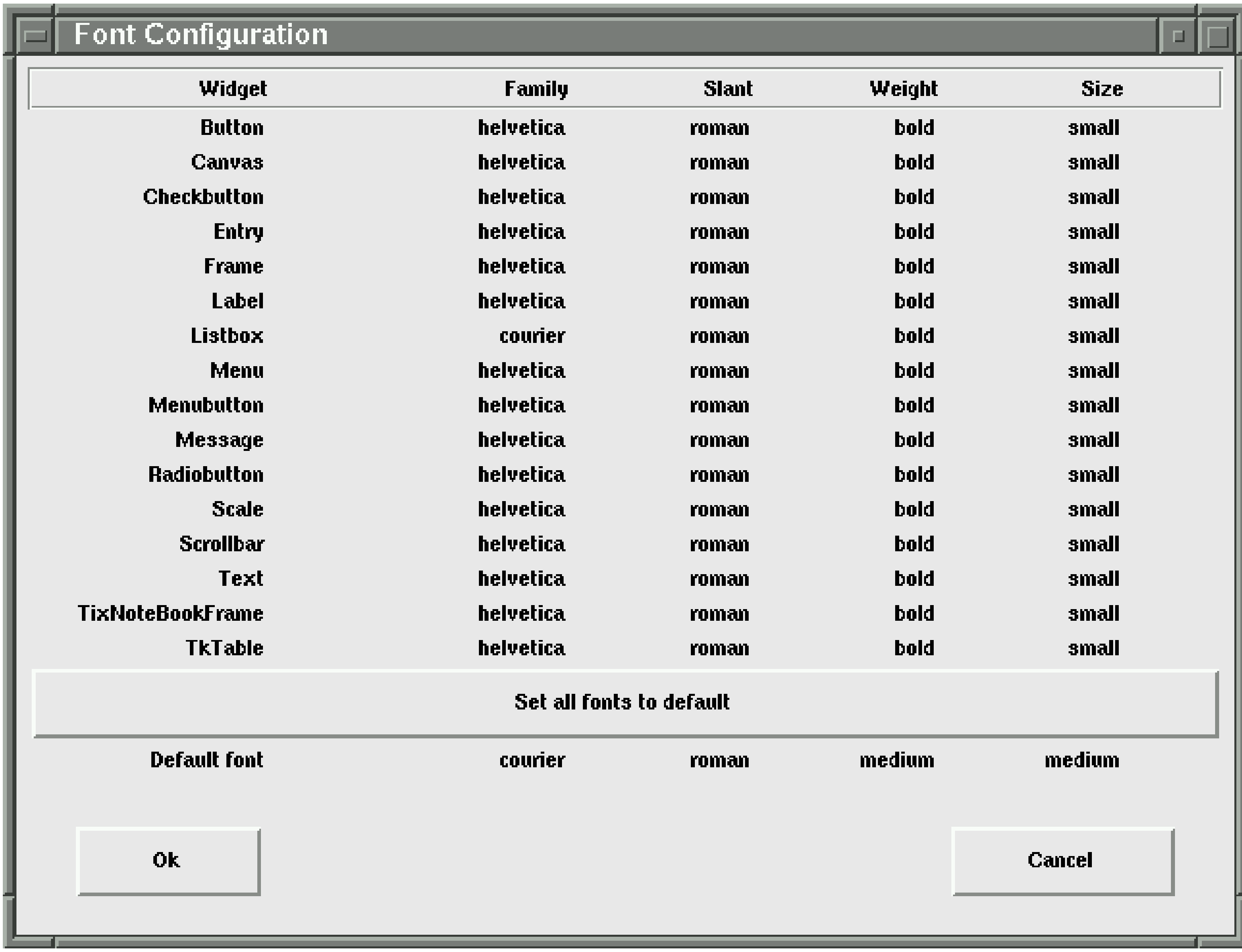


Figure 5.13: Font Configuration Window

Figure 5.14: Font's family options

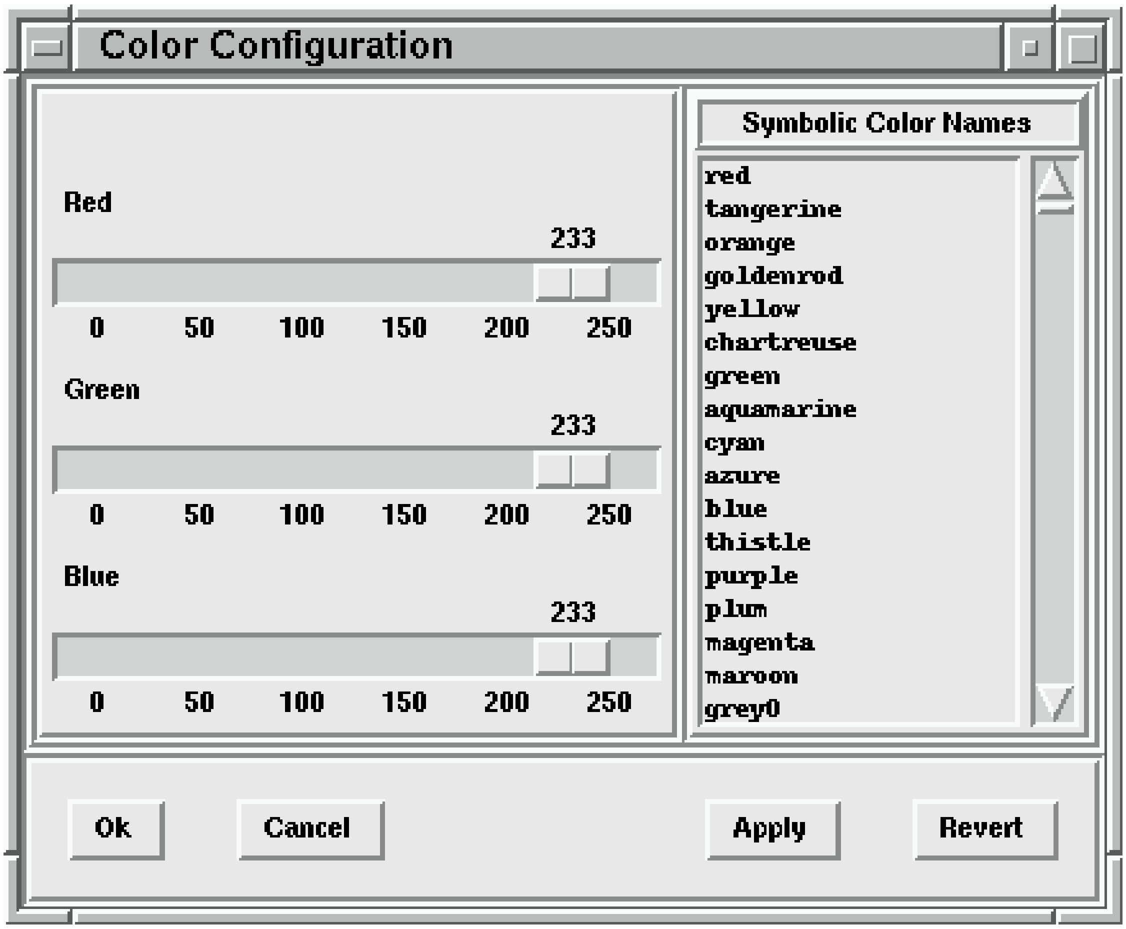


Figure 5.15: Configure Background Color

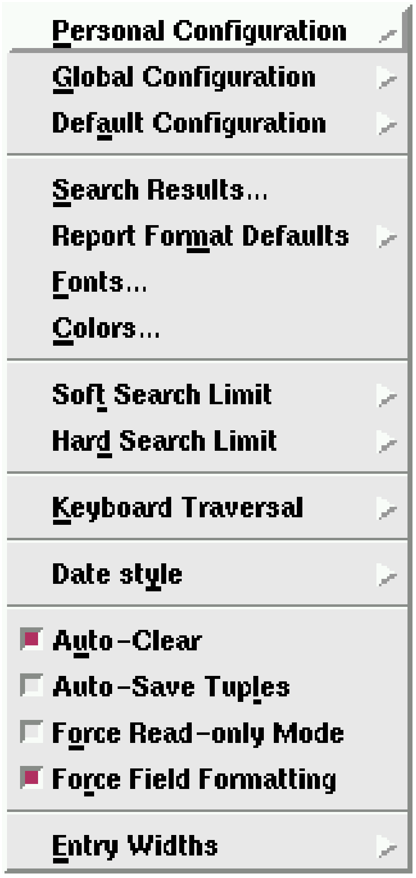


Figure 5.16: Configure pull-down menu

Figure 5.17: Configure Soft Limit

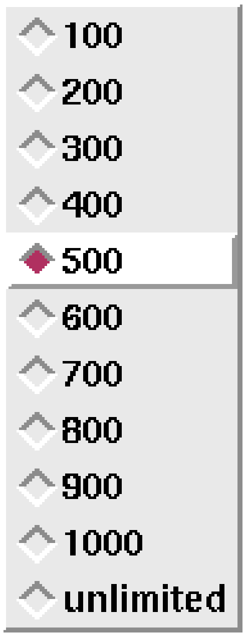


Figure 5.18: Configure Hard Limit

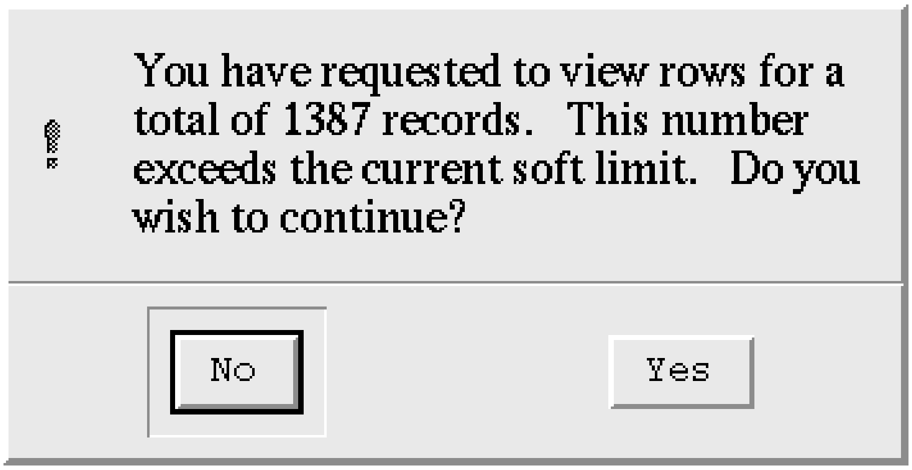


Figure 5.19: Soft limit warning dialog box

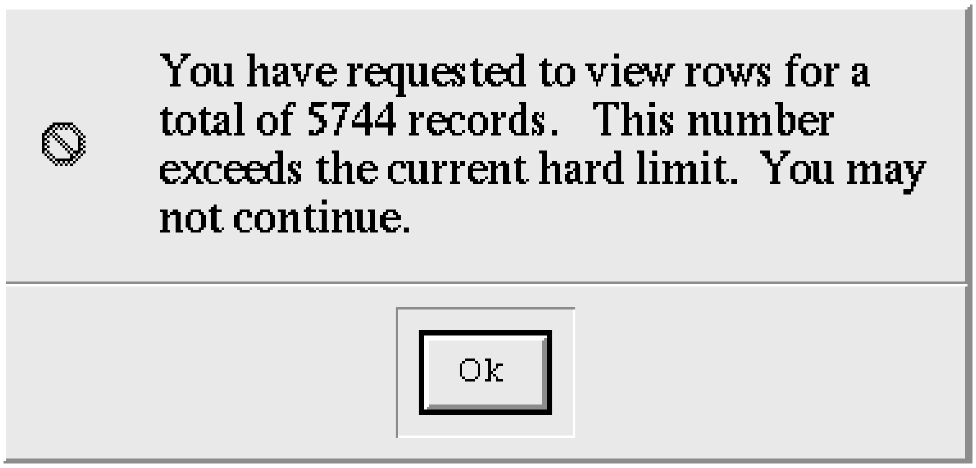


Figure 5.20: Hard limit warning dialog box

Figure 5.21: Set tabbing for scrollbars, add, view and delete buttons

Figure 5.22: Date style options

```
export QDDB_DATESTYLE=us
export QDDB_DATESTYLE=european
```

Figure 5.23: Set environment variable to override compiled default

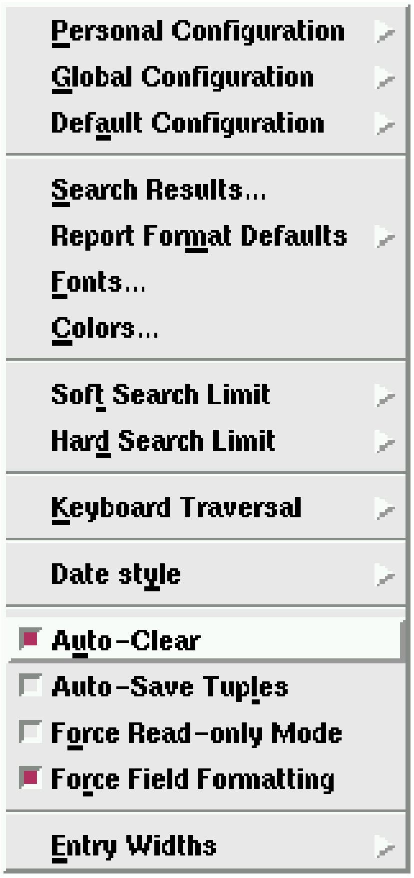


Figure 5.24: Setting the Auto-Clear option

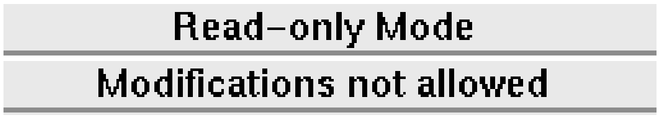


Figure 5.25: Read-Only Mode Status Bar

Chapter 6

Printing

Printing from nxqddb is controlled via a standard print dialog shown in figure 6.1.

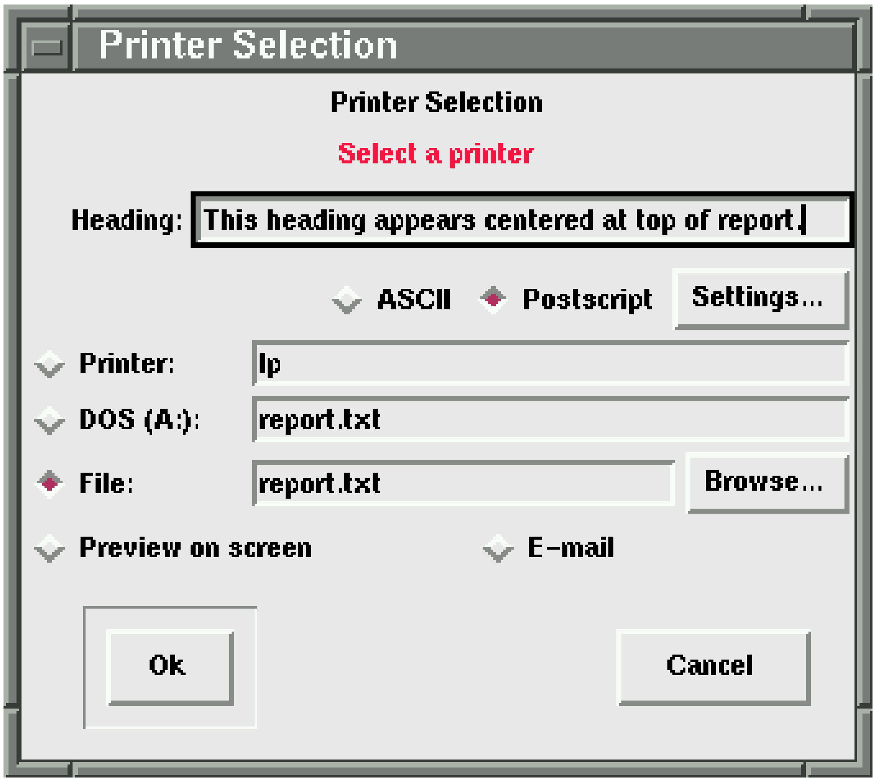


Figure 6.1: Printer Selection Dialog Box

From the print dialog, you can direct your output to any printer destination, floppy disk, hard drive, preview it on screen or Email the output to others. You can select the report to be in ascii or postscript.

If you select the postscript format, the “Settings...” button will become enabled for you to set various postscript options. The postscript settings are shown in figure 6.2

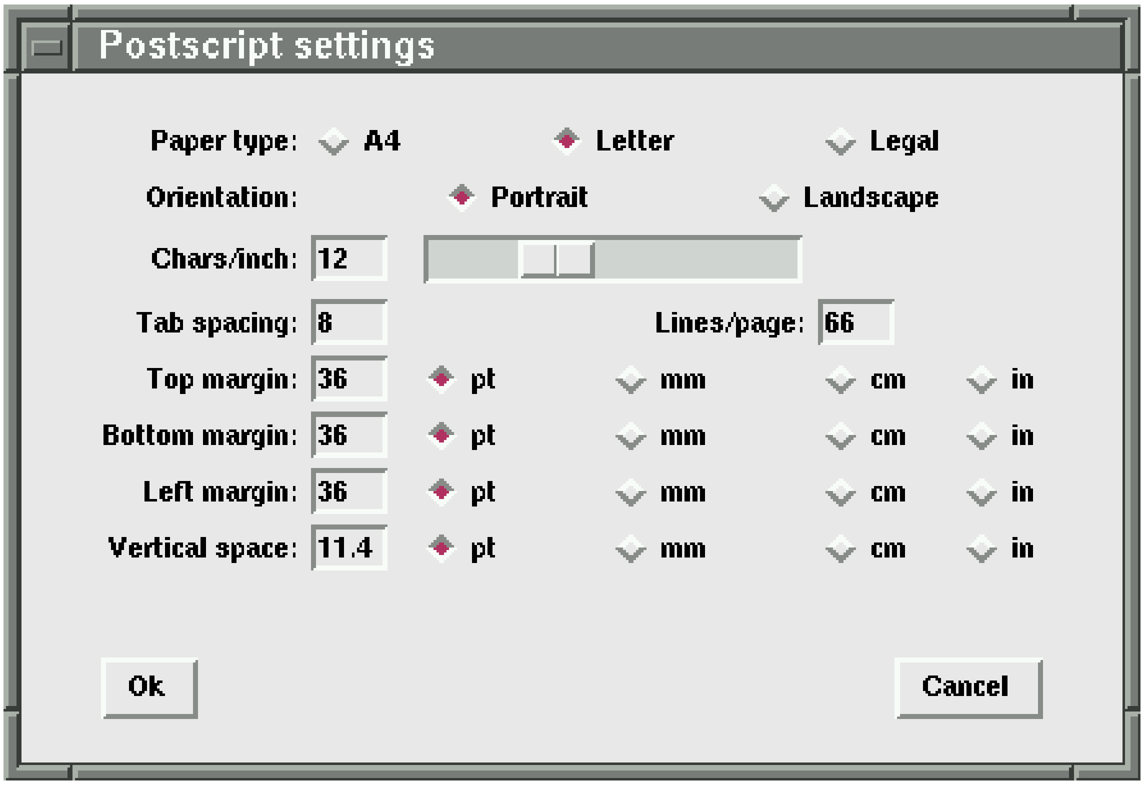


Figure 6.2: Postscript settings window

If you wish to save your report to your hard disk, choosing the “File” radiobutton enables the “Browse” button. Choosing this button displays a standard Tk browse window(see figure 6.3) to allow you to easily find a file of your choice.

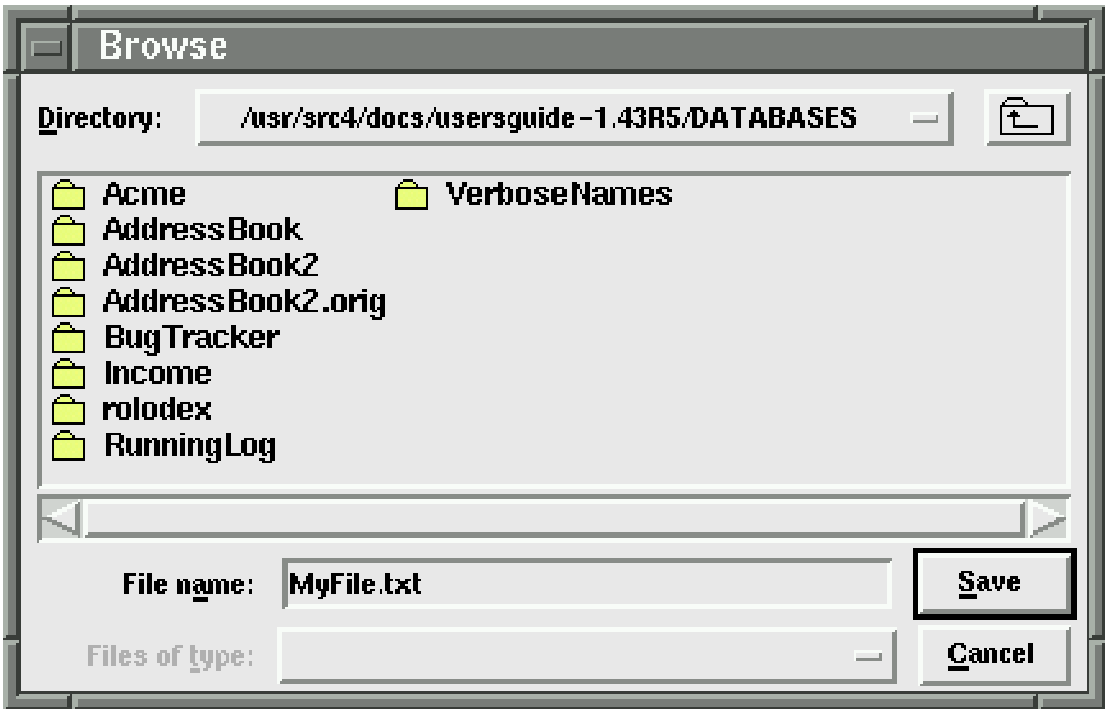


Figure 6.3: Standard browse window

Chapter 7

Generating reports

This chapter describes how to use the *nxqddb* report generator. In particular, we'll cover the steps to creating the various types of reports. We'll walk through an example of creating a report listing and postcards for our AddressBook2 database that we created earlier. We'll also describe how to create each of these types of reports:

1. reports through *nxqddb* search mode
2. report listing
3. postcards
4. letters
5. email
6. graphs
7. barcharts

7.1 Reports through *nxqddb* search mode

You may already know how to create reports if you've been playing with *nxqddb* searching. To create a report from search mode, follow these steps:

1. Click the "Configure/Search Results". The "Search Results Configuration" window(see figure 4.6) is displayed to allow you to choose the columns, sort order, new user-defined columns, column justification, column widths and separators, and whether or not you wish to see line numbers on your report. Remember, if this search criteria is often used, you may wish to write the template to a file by clicking the

"Write to file" button. For details on operating this configuration window, reference section 4.4.

2. Go to search mode by choosing "Modes/Search Mode"

3. Choose the "Edit/Clear" menubutton to clear out any previous search criteria. Of course, this step is optional.

4. Tab to the field(s) that you wish to search on and key the search value(s). Alternatively, if you've previously saved a search template, you can read the template into your search window by choosing the "Templates/Read Template" menubutton. See chapter 4 for details on how to search within the *nxqddb* window.

5. After you've completed keying in your search criteria, start the search by either (1) pressing the enter key, or (2) clicking the "Edit/Search" button.

6. If only one record is found that matched your search criteria, the record will be displayed in the *nxqddb* window and you'll be in "Change Mode". From "Change Mode", you can click the "File/Print" menubutton to generate a report containing this one record.

7. If more than one record is found, a window similar to

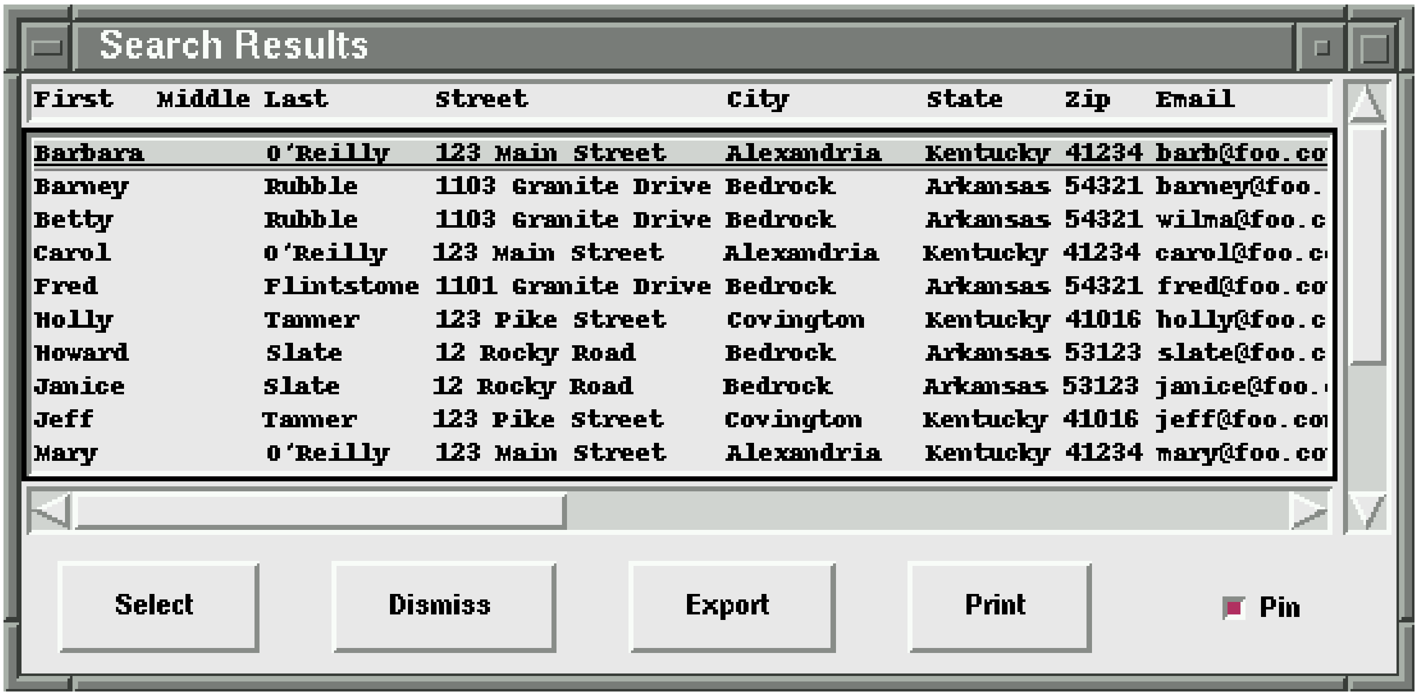


Figure 7.1: Search Results Window

figure 7.1 is displayed. From this window, you choose the "Print" button to display the standard print dialog box. From the print dialog box, you can direct the output to (1) a specific printer, (2) floppy disk, (3) hard disk, or (4) preview on screen.

Remember, you can configure the layout of these reports by configuring the search results.

7.2 Using the Expert Search for more complicated searches

For more complex reports, you can use the "Expert Search" window from within the *nxqddb* main window. Follow the same steps as you followed in the previous section for creating simple reports — only use the "Expert Search" window to generate your search criteria. To invoke the "Expert Search" window, click the "Edit/Expert Search" button. See section 4.5 for details on how to perform expert searches.

7.3 Displaying the Reports Window

If you want more sophisticated reports, including the capability to generate graphs, bar-charts, and automated mass mailings via email, letters and postcards — use the *nxqddb* reports window. To display the reports window, choose the "Edit/Reports"

Figure 7.2: The Edit Submenu Options

menubutton (see figure 7.2) to display the window shown in figure 7.3.

Figure 7.3: Window to generate custom reports

> You must be in Search Mode to have access to the "Edit/Reports" menubutton. To enable the "Edit/Reports" submenu, go to Search Mode by choosing the "Modes/Search Mode" menubutton.

7.4 Steps to creating a report

From figure 7.3, you can see that the window is divided into six major categories:

Report Title Enter text that you want to appear at the top of your report.

Search Criteria and Results Format From this frame, you tell *nxqddb* what search results you desire and how to print the results. This two steps are identical to what you've already learned from the (1) expert search section (see section 4.5) and (2) search results configuration (see section 4.4)

Listing options You can include the *Sum*, *Average*, *Standard Deviation*, *Minimum* or *Maximum* values for numeric columns.

> A "numeric column" means that the attribute must have been defined as "real" or "integer" in the Schema when you created your database.

Choose the Printing Options You can choose to generate any combination of reports from (1) Listings, (2) Postcards, (3) Letters, (4) E-mail, or even (5) Graph and (6) Bar charts.

Define the report setup If you want a specific setup, other than the default setup you created, you can specify the setup for each of (1) Letters, (2) Postcards, (3) E-mail, (4) Graph and (5) Barchart.

Click the appropriate button The last step is to generate the report by choosing the "Run report" button. Alternatively, you can choose "Dismiss" to tell *nxqddb* to disregard your input.

7.5 Generating a report from AddressBook2 database

This section walks through an example, step by step, on how to create a simple report. Using the AddressBook2 database created earlier, we'll generate some birthday announcements to send to all our "friends" and "family members". In our scenario, we want to generate two types of reports. First, we want a listing of all the people we want to invite to the party. This listing will contain the following columns:

1. First name

2. Last name

3. Street
4. City
5. State
6. Zip
7. Phone number (Area and Number)

This listing will serve as a checklist for when we mail out the invitations (a.k.a postcards). The listing will also serve as a handy contact reference since we've included phone number information.

Secondly, we want to generate a postcard for every "Family" or "Friend" that appears in our database.

7.5.1 Entering a Report Title

As mentioned earlier, the first step is to display the reports window as shown in figure 7.3. At the top of this window is an entry for you to specify the report title. For our report, we'll add a title "Elaine's 30th Birthday Party!!" as seen in figure 7.4.

Figure 7.4: Entering a report title

7.5.2 Entering Search Criteria

The second step to creating a report is to select those records from the database that you want to include on your reports. (If you've already read chapter 4 on "Searching", this information will be familiar to you.)

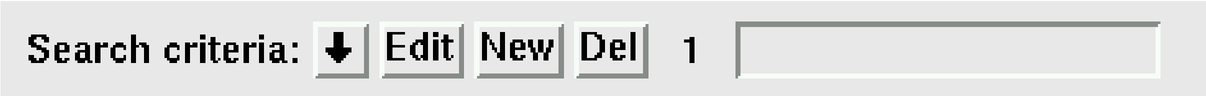


Figure 7.5: Entering your search criteria

Click the "New" pushbutton (see figure 7.5) to create the search criteria. The familiar expert search window will appear as shown in figure 7.6.

For a detailed discussion of how to use this expert search window, please refer to section 4.5. In our scenario, Elaine is celebrating her 30th birthday. So, we want to send out invitations to all "friends" and "family" (Its a really big party!!) For this search criteria, we want to restrict our report to those records where *Relation* = *"friend"* **OR** *Relation* = *"family"*.

Enter a title on the expert search, for example "Elaine's Birthday Invitation". Then, click on the down arrow to display the

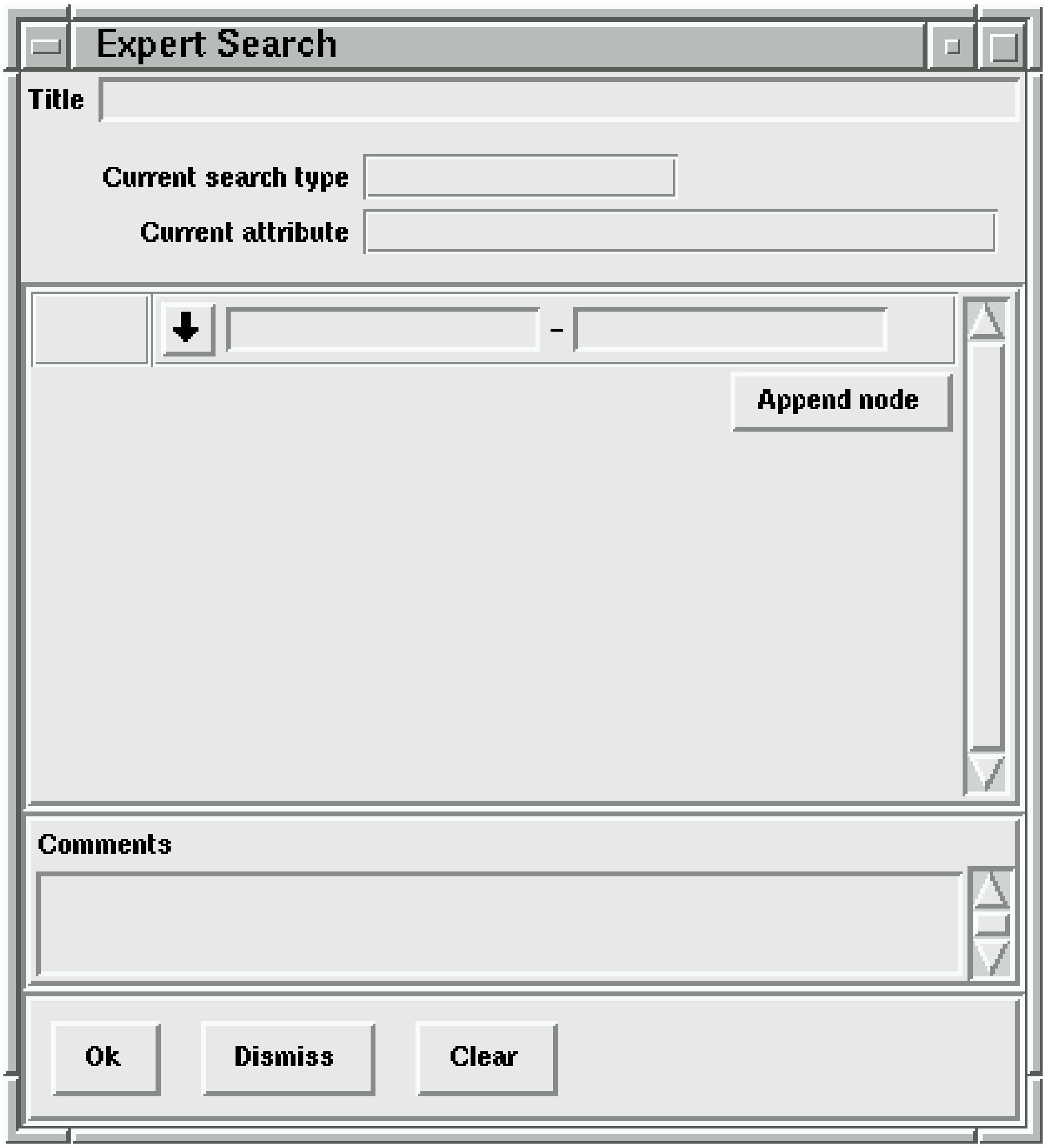


Figure 7.6: Expert search window used to capture your search criteria

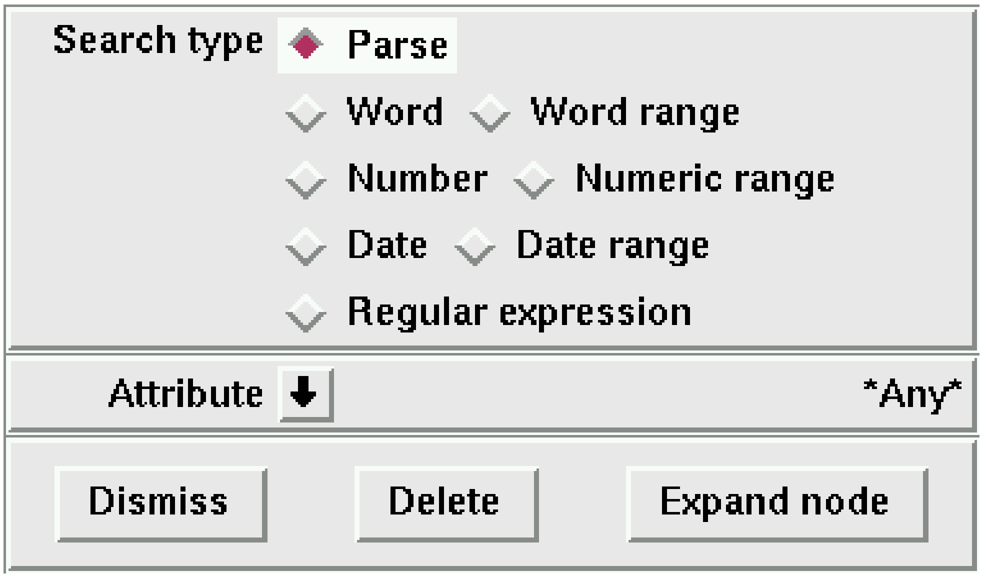


Figure 7.7: Enter the search criteria for Elaine's party

window shown in figure 7.7. Choose the "Parse" radiobutton and then click the "Attribute down arrow" button to view a list of the attributes defined in our AddressBook2 schema. From this list, use the scrollbar to scroll to the bottom. You should see the "Relation" attribute. Click the Relation row to select that attribute. Remember, the relation attribute can be one of "Family", "Friend", "Bus." or "Other". Now, position the cursor in the entry field and enter the value **family , friend**.

The criteria **family , friend** tells *nxqddb* to return only those people whose relation radiobutton was selected as either "Family" or "Friend".

If you want, you can enter text in the comments field to help you remember your search. Since this search was such a simple one, comments may not be necessary; however, many users still like to add comments for documenting their search criteria.

Click the Ok button to save your entered search criteria and return back to the "Reports" window. If you click the "Dismiss" button, *nxqddb* will disregard your search criteria and return you back to the "Reports" window. Clicking the "Clear" button simply erases all of your input on the "Expert Search" window for you to start entering your criteria from scratch.

When you return back to the "Reports" window, you'll notice that the report title entered on the "Search Criteria" window, "Elaine's Birthday Invitation" appears in the label just to the right of the *Edit*, *New* and *Del* pushbuttons. Immediately to the right of the pushbuttons is the number "1" to denote the report. You can define any number of search criteria per report using the "down arrow", "Edit", "New" and "Del" buttons.

However, for our scenario we just want to produce one search criteria to invite family and friends to Elaine's party.

7.5.3 Entering the Results Format

Click the "Results format" button to bring up the **Column definitions** window shown in figure 7.8.

If this window looks familiar, then you've obviously read the chapter on searching or have been experimenting with *nxqddb* on your own. For details on how to operate this window, please reference chapter 4. For our announcements, we want to be sure to include the following attributes:

- First and Last name attributes
- Address attributes including: Street, City, State, and Zip
- Email
- Relation
- AreaCode and Number (so we can phone people if plans change!)

In order to produce letters or postcards within *nxqddb*, you must include those fields in the column definition that comprise the destination address. See chapter 5, for information pertaining to the configuration of the destination address.

If you omit one or more fields from the column definition that is setup in the destination address, *nxqddb* will stipple the (1) postcards, and (2) letters checkbuttons that allow you to generate these mass mailings.

Additionally, if you decide to generate E-mail, you must be sure that the Email attribute (configured from the "Configure/Report Format Defaults/E-mail" option) is also defined.

7.5.4 Listing options (summary of numeric columns)

Since we're only generating a mailing list, we really have no need to produce any statistical information from our columns. In fact, since the listing options work only on numeric columns (i.e. those attributes defined with type "integer" or "real"), we couldn't produce this information with our AddressBook2 database even if we wanted to. Remember, our AddressBook2 schema did not define any fields with type "integer" or type "real".

7.5.5 Printing options

In our scenario, we want to produce a listing and postcards. The "Listing" checkbutton is turned on by default.

> As a visual clue, you know that a checkbutton is turned on when the tiny checkbutton square is the color red and is sunken.

Since the "Listing" checkbutton is likely already turned on, you shouldn't need to click this button.

Click the Postcards checkbutton to tell *nxqddb* that you want to generate postcards.

> The Postcards and Letters checkbuttons will be available only if you've included every attribute in your "Results Format" that also appears in the destination address. That is, *nxqddb* compares your default destination address setup to the fields that you specified in the "Results format" operation where you specified the column definitions. If there is one or more fields in the default destination address that do not appear in your column definitions, then *nxqddb* will not allow you to produce postcards or letters. This safety check helps to prevent you from creating invalid mailing addresses. Remember, you set up the destination address from the "Configure/Report Format Defaults" menu.

7.5.6 Creating the postcard message

From the "Reports" window, there is a setup panel with pushbuttons to allow you to customize the content and layout for (1) letters, (2) postcards, (3) E-mail, (4) graphs, and (5) barcharts.

For our scenario, we only want to produce a listing and postcards. So, let's setup our postcard invitations. Click the "Postcards" button to display the "Postcard Message" window as shown in figure 7.9.

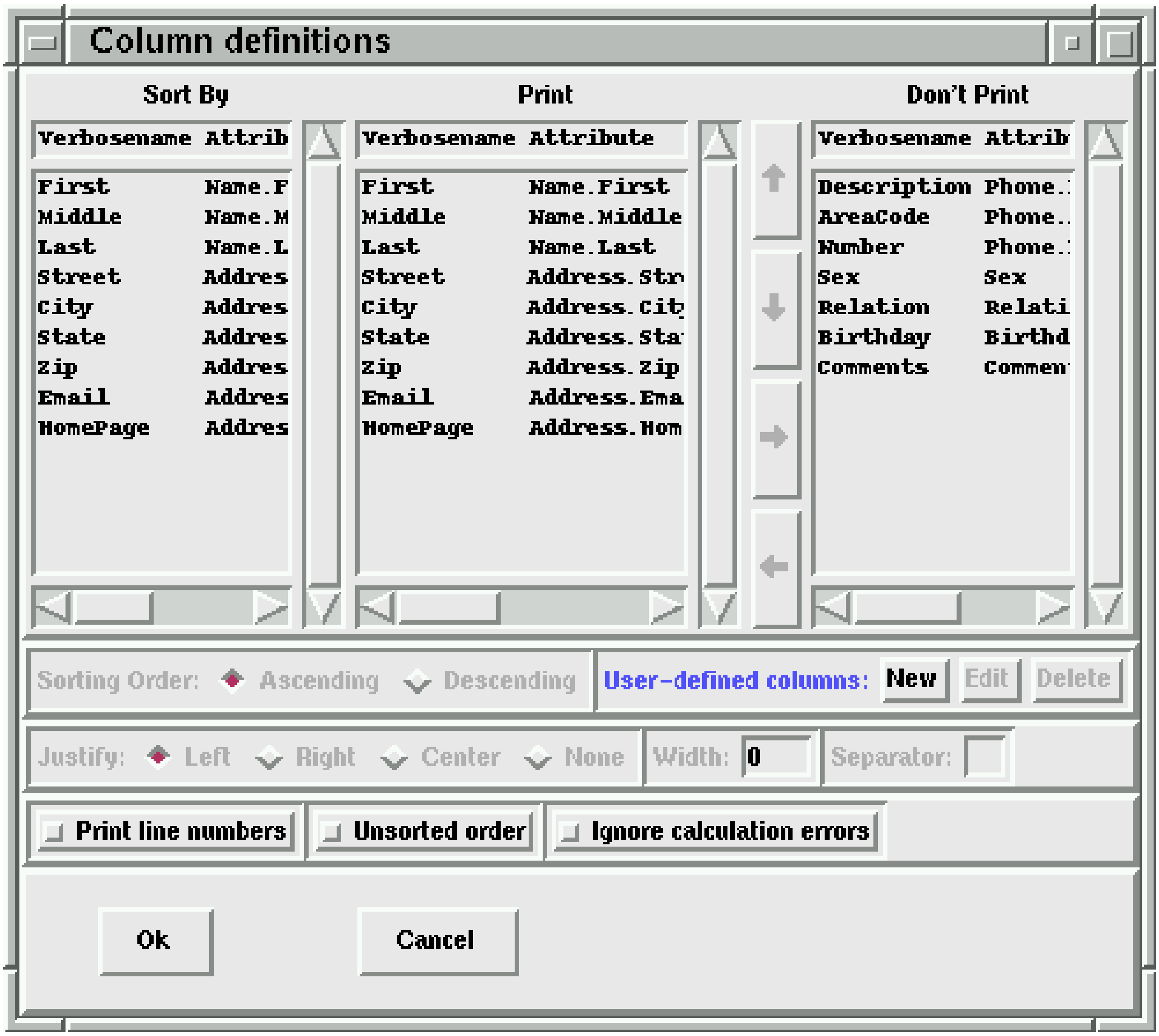


Figure 7.8: Window to define the report's columns

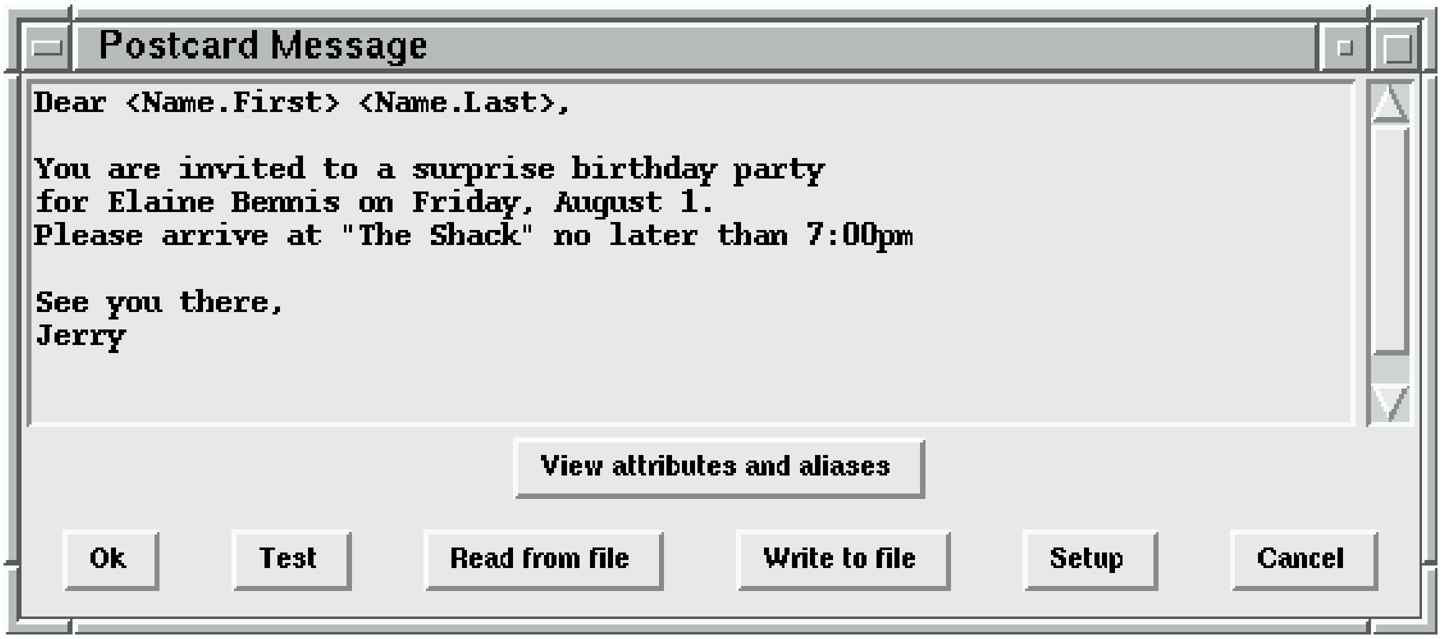


Figure 7.9: Window to enter postcard message

Create your custom postcard message by entering the appropriate text in the text entry(see figure 7.10).

```
Dear <Name.First> <Name.Last>,

You are invited to a surprise Birthday party
for Elaine Bennis on Friday, August 1.
Please arrive at The Shack no later than
7:00pm.

See you there,
Jerry
```

Figure 7.10: Sample postcard message for Elaine's Birthday Party

To avoid silly input errors when creating your postcard message, *nxqddb* provides a "View attributes and aliases" button to allow you to conveniently paste the attributes into your message entry. To do so, follow these steps: (1) Click the attribute in the "View Attributes and Aliases" window (the row will become highlighted), (2) Position the mouse in the "Destination Address Setup" window where you want to place the attribute and then click the middle mouse button to paste the attribute into the entry.

nxqddb will automatically format your postcard with the destination address, and vertical separator that you specified in the "Configure/Report Format Defaults" — provided you did *not* choose the *Free-form postcard* checkbutton.

If you chose the *Free-form postcard* option on the Postcard Setup window, then you must enter the destination address and vertical separator (if desired). Most users find it convenient to just let *nxqddb* generate the postcard destination addresses automatically.

Saving/Retrieving postcard messages

nxqddb provides "Read from file" and "Write to file" pushbuttons to allow you to read/write postcard messages to and from your hard disk. This feature is convenient because you can avoid typing a postcard message from scratch each time you do a mass mailing. For example, with the AddressBook2 database, you may choose to create generic birthday announcements, club meeting notices, weekly golf statistics, bridge meetings or whatever need you have for mass mailings. Then, when it comes time to do the mass mailing, you can read a previously saved postcard message template and make any necessary changes — instead of retyping the message from scratch. For details on how to read/write templates, please reference section 4.4.8.

Modifying the postcard setup

To modify your default postcard setup for Elaine's invitations, click the "Setup" pushbutton on the "Postcard Message" window to display the "Postcard Setup" window shown in figure 7.11. For details on this setup window, reference section 5.5.2.

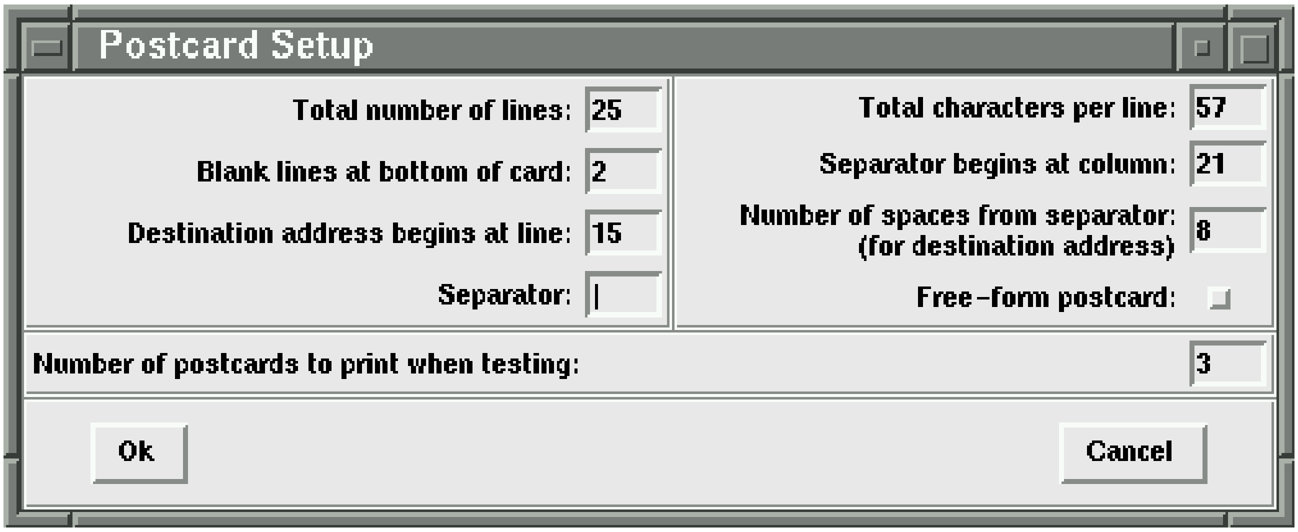


Figure 7.11: Window to adjust the postcard settings

7.5.7 Running the report

Once you've entered in the search criteria, results format, and created your postcard message, click the "Run report" button to create your report.

Standard Print Dialog Box

At this point, *nxqddb* generates your report and if matches are found, the standard print dialog box will be displayed. See the chapter on printing (chapter 6) for details on how to operate the standard print dialog window.

Preview on screen

Your output, when previewed on screen, should resemble that shown in figure 7.12

From the preview window, you can print the report by clicking the "print" button or dismiss the window by clicking the "Close" button. If you click the print button, the standard *nxqddb* print dialog will appear to prompt you for a destination.

After you leave the listing window, *nxqddb* continues to the next print option that you have selected. If you only chose the listing option, then you'll be returned back to the main "Reports" window. However, in our scenario we also chose to generate postcards; so, *nxqddb* will now do the postcards. The "Postcard Message" window will be displayed as shown in figure 7.9. You can retrieve the text entered earlier by choosing the "Read from file" pushbutton to grab the postcard message recorded earlier. After reading your saved postcard message, your window should resemble the one shown in figure 7.13.

Once you have the message text the way you like, choose the "Ok" button to start generating your postcards. When finished generating your postcards, *nxqddb* will display the familiar standard print dialog box (see figure 6.1) to prompt you for a printer destination. Previewing the results should yield a window similar to the one shown in figure 7.14.

Once you've finished generating your postcards, you'll be returned back to the reports window. At this point, you've successfully generated both a report listing and a printed postcard for each person in your AddressBook2 database that satisfied your "family/friend" search criteria.

7.5.8 Saving your report

After running your report, you may save your report setup to a file on your hard disk by choosing the "Write to file" pushbutton. Then, if you ever need to run this report (or one similar), you can save time by retrieving the file later when you wish to run the report.

7.6 Postcard Reports

The steps to creating mass mailings on postcards is described in detail in the previous sections (see section 7.5).

7.7 Letter Reports

Similarly to postcards, you can produce mass mailings on letters. Like postcards, you follow the same steps to produce letters for each person found in your report's search results.

7.8 Email Reports

In the previous section, you learned how to generate a listing report as well as mass mailings for postcards. This section discusses *nxqddb*'s useful Email reporting feature.

You can setup a report such that each record(i.e. person) found on the report is automatically E-mailed a customized mail message. An overview of the steps to perform Email mailings is described below:

1. Click "Configure/Report Format Defaults/E-mail format" to display the "E-mail Setup" window shown in figure 5.8. Please reference section 5.5.3 for details on how to configure your E-mail setup.

2. Click the "Edit Reports" menubutton to display the "Reports" window(see figure 7.3).
3. Follow the instructions as described in section 7.4.
4. Be sure to click the E-mail checkbutton from within the "Printing options" panel.

> For the E-mail checkbutton to become highlighted, you must be sure that you include your "E-mail" in the "Print" dialog box when you performed the "Results format" operation. Remember, you specify the E-mail attribute when you configured *nxqddb* via the "Configure/Report Format Defaults/E-mail format" menubutton.

5. Click the "E-mail" button from within the "Setup" panel to display the "E-mail Message" window(see figure 7.15). From this window, you specify a subject by keying into the "Subject" entry. You customize your mail message by typing into the text entry. To ease the task of specifying your attributes, *nxqddb* provides the "View attributes and aliases" window.

> To avoid silly input errors when creating your E-mail message, *nxqddb* provides a "View attributes and aliases" button to allow you to conveniently paste the attributes into your message entry. To do so, follow these steps: (1) Click the attribute in the "View Attributes and Aliases" window (the row will become highlighted), (2) Position the mouse in the text entry window where you want to place the attribute and then click the middle mouse button to paste the attribute into the entry.

7.9 Creating new columns

Faunix provides the ability for you to create new columns to display on a report. Please be warned, this section is for advanced users and is quite technical.

To create a new column, click the "New" button to the right of the "User-defined columns" label. The window shown in figure 7.16 is displayed. As you can see from figure 7.16, there are a lot of built-in commands to help you construct your new column. Clicking a button on this window enters the corresponding text in the field construction window. This section describes the formulas to use in creating a user-defined column.

There are four types of subexpressions:

@range(row,col:row,col)
@row(row)
@col(col)
@cell(row,col)
@(row,col?:row,col?)

Table 7.1: Ranges

1. numeric constants/expressions
2. ranges
3. range functions
4. numeric functions

Ranges can only be used as arguments to a range function. Range functions take ranges as arguments and produce real numbers; numeric functions take numbers as arguments and produce real numbers. All calculations and values are represented in double precision floating point.

7.9.1 Numeric expressions

Numeric expressions may include any of the following operators and may be fully parenthesized. Precedence is the same as the C programming language:

```
- + * / %
```

For example, the following are all valid numeric expressions:

```
(2+3)/4
2+3/4
97*2+98*2/3-42.0
```

7.9.2 Ranges

A range can be any of the following as shown in table 7.1

Row and column names may be used as arguments to ranges, as well as numeric expressions and the following constants:

- @thisrow

@sum	sum
@prod	product
@avg	average (mean)
@stddev	standard deviation
@count	num of cells in range
@min	minimum cell value
@max	maximum cell value

Table 7.2: Range functions

- @thiscol
- @maxrow
- @maxcol

The constants are evaluated in the current table context (that is, the calculated field's implicit table). Each range produces a list of cells to be evaluated by a range function. @range produces a list of all cells in the rectangular region specified by the four numeric arguments (row,col:row,col). Ranges have no meaning by themselves and must be used as arguments to a range function. Cell and cell ranges (@range and @cell) may be abbreviated with just '@'.

7.9.3 Range functions

A range function accepts a list of ranges separated by commas and produces a numeric result. The result of a range function is a numeric value that can be used in any expression, or as the argument to a numeric function. All range functions have the form:

```
range_func(range1 ?, range2, ...?)
```

The range functions are shown in table 7.2.

7.9.4 Numeric functions

A numeric function accepts numeric values as arguments and produces a numeric result. No table context is required for numeric functions. The numeric functions accept any numeric expression as an argument. See table 7.3 for a brief description of the numeric functions.

@cellval(e1,e2)	cell value @(e1,e2)(equivalent to "@sum(?t,? @(e1,e2))")
@sqrt(e)	square root of 'e'
@exp(e)	exponential function of 'e'
@ln(e)	natural log of 'e'
@log(e)	log base 10 of 'e'
@floor(e)	largest integer not greater than 'e'
@ceil(e)	smallest integer not less than 'e'
@rnd(e)	round 'e' to nearest integer
@round(e,n)	round 'e' to 'n' decimal places
@abs(e)	absolute value of 'e'
@pow(e1,e2)	'e1' raised to the power of 'e2'
@pi	a constant close to pi
@degrees(e)	convert 'e' radians to degrees
@radians(e)	convert 'e' degrees to radians
@sin(e)	sin of 'e'
@cos(e)	cos of 'e'
@tan(e)	tan of 'e'
@asin(e)	asin of 'e'
@acos(e)	acos of 'e'
@atan(e)	atan of 'e'
@atan2(e1,e2)	atan of e1/e2
@hypot(e1,e2)	computes sqrt(e1*e1 + e2*e2) in such a way that underflow will not happen

Table 7.3: Numeric functions

=	equal to
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
!=	not equal to
&	logical and
—	logical or
!	logical not (unary)

Table 7.4: Logical expressions

7.9.5 Logical expressions

A logical expression performs various logical operations and returns the numeric value 1.0 if true, and 0.0 if false. Logical operations may be fully parenthesized. See table 7.4 for a brief definition of each logical expression. Logical expressions are used as the first argument to the @if function:

```
@if(condition, e1, e2)
```

If the logical expression is true, then @if evaluates to the second argument **e1**. If the logical expression is false, it evaluates to the third argument **e2**. See Section 7.9.6 for an example of @if function usage.

7.9.6 Examples

Year to date invoice totals

Let's assume you have the following simple Schema for invoice total summaries:

```
Date type date
Comments
NumberOfInvoices type integer
InvoiceTotals type real format "%.2f"
PaymentTotals type real format "%.2f"
```

Now you want to create a column for a yearly report that calculates the year-to-date for invoices and payments. The first step is to create a new calculated field by clicking the "New" button in the search results configuration window. Next, for summing InvoiceTotals, we can enter one of the following:

```
@sum(@range(1,InvoiceTotals:@thisrow,InvoiceTotals))
```

The title of this column could be "Invoice Totals YTD". Now you can build several separate searches for your report, each of which builds a table for a full year (say 1995, 1996, and 1997).

Block ranges

Using our same Schema, now let's build a new column that shows the average YTD of payments and invoice totals on a daily basis. Again, we build a column by selecting the "New" button in the search results configuration window. Now we can perform the following calculation:

```
@avg(@range(1,InvoiceTotals:@thisrow,PaymentTotals))
```

Note that we must print the InvoiceTotals and PaymentTotals in adjacent columns, InvoiceTotals first, for this particular range to work properly. Block ranges are calculated in row order, left to right.

Conditional statements

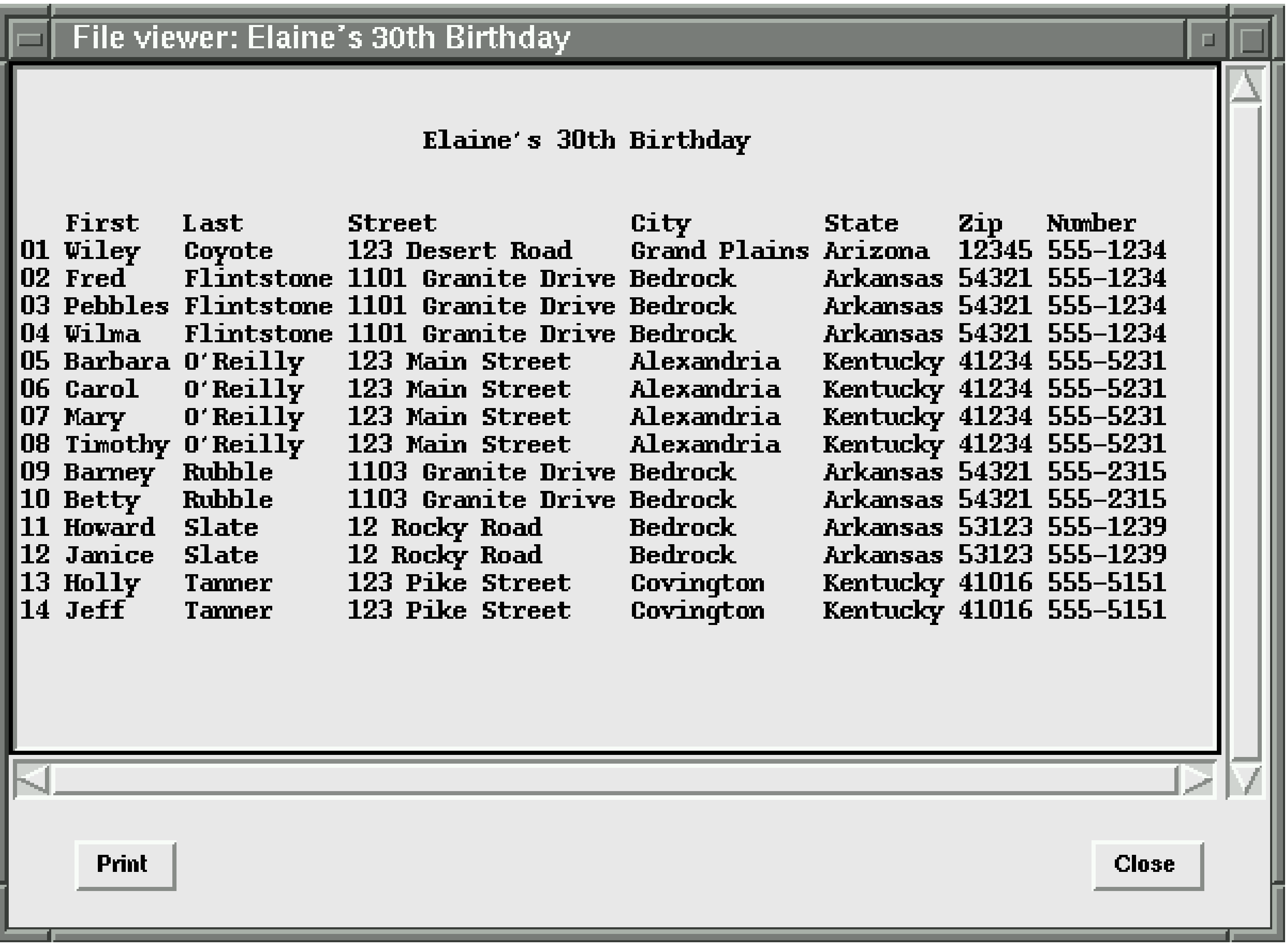
Suppose now that we want to see how many days for which the PaymentTotals field is larger than the InvoiceTotals. We can set up a new column as follows:

```
@if(@cellval(@thisrow,InvoiceTotals) <
    @cellval(@thisrow,PaymentTotals), 1.0, 0.0)
```

Now, for our report, we can ask for a "sum" in the listing options to get the summation of the user-defined column at the end of our report.

7.9.7 Notes

Calculations are free-format; that is, you may insert newlines and spaces throughout to improve readability. Attribute names (that is, column names) may be used interchangably with column numbers. We recommend using attribute names to improve readability and guard against human error.



Elaine's 30th Birthday

	First	Last	Street	City	State	Zip	Number
01	Wiley	Coyote	123 Desert Road	Grand Plains	Arizona	12345	555-1234
02	Fred	Flintstone	1101 Granite Drive	Bedrock	Arkansas	54321	555-1234
03	Pebbles	Flintstone	1101 Granite Drive	Bedrock	Arkansas	54321	555-1234
04	Wilma	Flintstone	1101 Granite Drive	Bedrock	Arkansas	54321	555-1234
05	Barbara	O'Reilly	123 Main Street	Alexandria	Kentucky	41234	555-5231
06	Carol	O'Reilly	123 Main Street	Alexandria	Kentucky	41234	555-5231
07	Mary	O'Reilly	123 Main Street	Alexandria	Kentucky	41234	555-5231
08	Timothy	O'Reilly	123 Main Street	Alexandria	Kentucky	41234	555-5231
09	Barney	Rubble	1103 Granite Drive	Bedrock	Arkansas	54321	555-2315
10	Betty	Rubble	1103 Granite Drive	Bedrock	Arkansas	54321	555-2315
11	Howard	Slate	12 Rocky Road	Bedrock	Arkansas	53123	555-1239
12	Janice	Slate	12 Rocky Road	Bedrock	Arkansas	53123	555-1239
13	Holly	Tanner	123 Pike Street	Covington	Kentucky	41016	555-5151
14	Jeff	Tanner	123 Pike Street	Covington	Kentucky	41016	555-5151

Figure 7.12: Report listing of invitees to Elaine's birthday party

Figure 7.13: Populated Postcard Message Window

Figure 7.14: Previewing postcards for Elaine's birthday party invitations

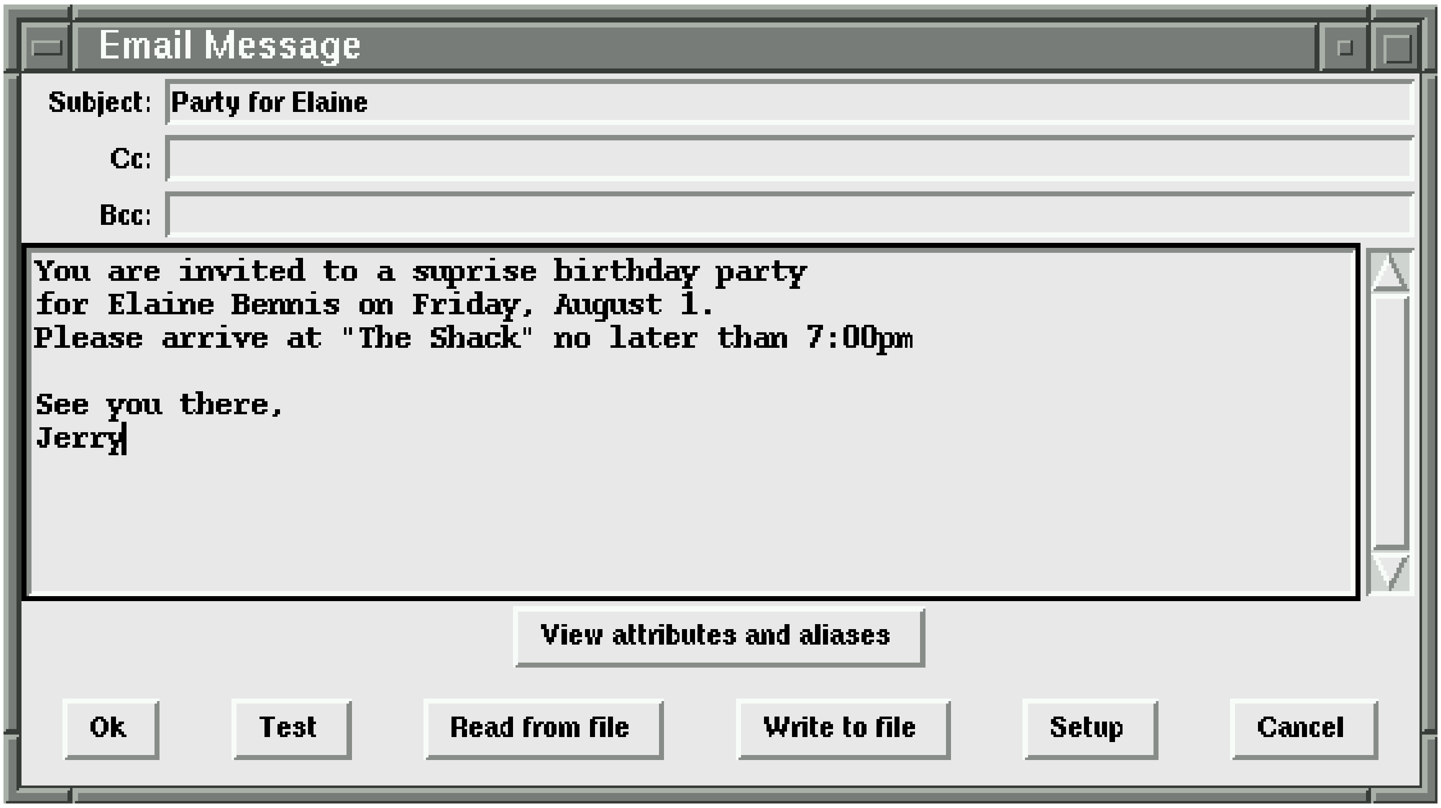


Figure 7.15: Sample Email Message Setup

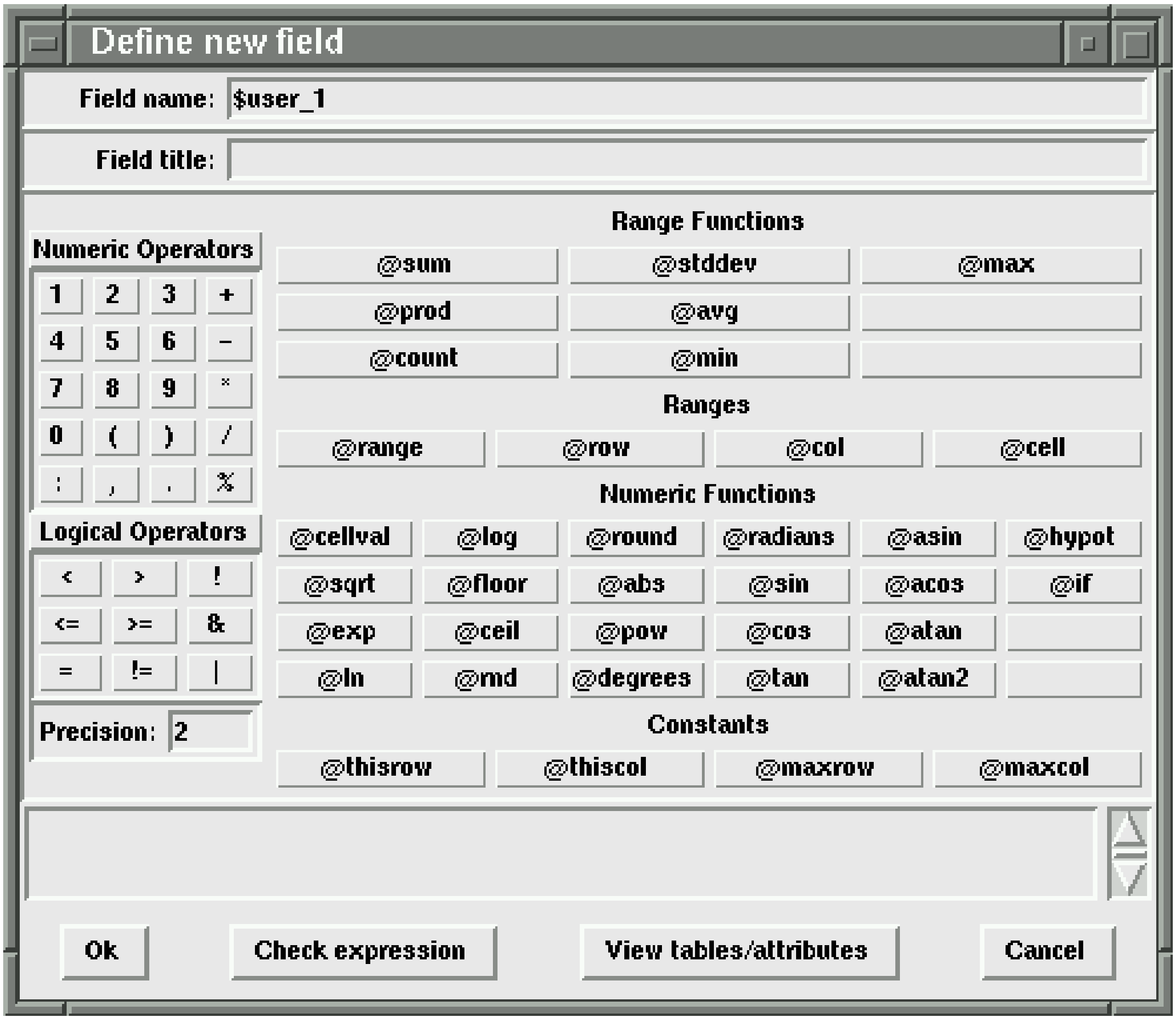


Figure 7.16: Define a new column on your report

Chapter 8

Graphs

In this chapter, we present the steps to creating graphs and barcharts within *nxqddb*. In particular, we walk through the process of creating simple X-Y graphs for a fictitious software company — Acme. We also present range analysis examples for Acme's web site hits.

8.1 Acme's schema and database for the graph scenario

Assume Acme has four types of classifications for its software bugs: (1) nasty, (2) cosmetic, (3) fluff and (4) documentation. The goal of this exercise is to plot the number of bugs reported over time for each of the four classifications.

For this scenario, assume Acme's database was created with the schema shown in figure 8.1(intentionally kept simple for illustrating graphs).

```
Classification
NumberOfDaysSinceLaunch type integer
CumulativeNumberOfBugs type integer
```

Figure 8.1: Acme's bug tracking schema

The attributes for Acme's schema are described below:

Classification This field will contain one of: (1) nasty, (2) cosmetic, (3) fluff and (4) documentation.

NumberOfDaysSinceLaunch This field will be a count of the number of days since the Acme product was launched. This field will be used for the X-axis in our graph.

CumulativeNumberOfBugs This field will be a cumulative total for the number of bugs for the specific classification entered. This field will be used for the Y-axis in our graph.

Also, assume the contents of Acme's bug tracking database contains the records displayed in table 8.1.

8.2 Steps to creating graphs

Here are the steps to creating a graph in *nxqddb*:

1. Bring up the reports window by clicking the "Edit/Reports" menubutton.

2. From the Reports window, enter a search criteria for each graph item. For example, in our Acme scenario, we'll create four(4) search criteria, one for each bug classification: nasty, cosmetic, fluff and cosmetic.

3. Choose the "Results format". In our scenario, we just click the "Results format" button and accept the default settings.

4. After you've chosen the search criteria and set the results format, the graph checkbutton will become enabled in both the "Printing options" and "Setup" panels.

5. Click the "Graph" checkbutton on the "Printing options" panel.

6. Click the "Graph" pushbutton on the "Setup" panel to tell *nxqddb* how to graph your information.

7. The final step is to click the "Run report" button. *nxqddb* will display the standard print dialog box to prompt you for a printer destination.

8.3 Entering the search criteria

From the "Reports" window, click the "Edit" button to display the familiar expert search window. By now, you should be familiar with how to create searches within *nxqddb*. If you're unsure how to complete an expert search criteria, reference chapter 4.

You should create four search criteria for the Acme scenario. To create more than one search criteria, simply click the "New" button followed by the "Edit" button. See table 8.2 for the information that you should enter for each of the four search criteria.

Classification	*NumberOfDaysSinceLaunch*	*CumulativeNumberOfBugs*
nasty	1	1
nasty	2	4
nasty	3	10
nasty	4	16
nasty	5	18
nasty	6	19
nasty	7	19
nasty	8	19
nasty	9	19
nasty	10	19
cosmetic	1	1
cosmetic	2	2
cosmetic	3	3
cosmetic	4	10
cosmetic	5	10
cosmetic	6	12
cosmetic	7	15
cosmetic	8	28
cosmetic	9	36
cosmetic	10	36
fluff	1	5
fluff	2	5
fluff	3	5
fluff	4	8
fluff	5	8
fluff	6	8
fluff	7	8
fluff	8	9
fluff	9	10
fluff	10	11
documentation	1	10
documentation	2	12
documentation	3	14
documentation	4	16
documentation	5	18
documentation	6	19
documentation	7	20
documentation	8	21
documentation	9	22
documentation	10	23

Table 8.1: Input data for Acme's bug tracking graph example

Title	*Search type*	*Attribute*	*value*
nasty	word	Classification	nasty
cosmetic	word	Classification	cosmetic
fluff	word	Classification	fluff
documentation	word	Classification	documentation

Table 8.2: Search criteria values for Acme graph example

8.4 Entering the Results Format

Click the "Results Format" button to set the format of your report. Just choose "Cancel" button on the "Column definitions" window to accept the default *nxqddb* settings.

8.5 Printing options

For our scenario, we just want to produce a graph, so click the "Graph" checkbutton in the "Printing options" panel.

8.6 Setting up your graph

Click the "Graph" button in the Setup panel to display the "Graph setup" window shown in figure 8.2.

There are many options that you can configure for your graphs. Each of these options is briefly described below:

8.6.1 General graph options

Graph title Sets the title of the graph. The title appears centered above the graph.

Graph title font Clicking the "Font" pushbutton to the right of the "Graph title" entry box displays a dialog box(see figure 8.3) to allow you to choose the font for the graph title. From this dialog, you choose (1) Family, (2) Slant, (3) Size and (4) Weight by clicking the pull-down menu adjacent to the setting labels. To activate your new font, choose the "Ok" button. To cancel your modifications, choose the "Cancel" button.

Background Sets the background color of the graph, including the margins and legend but not the plotting area.

Plot background Sets the background color of your plotting area.

Graph width Sets the requested width of the graph window. For example, a value of "4i" sets the width of the window to four inches.

Graph height Sets the requested height of the graph window

8.6.2 Configuring options for the X/Y-Axis

Attribute Use the drop-down listbox to choose the attribute that you want to plot. In our Acme scenario, we chose "NumberOfDaysSinceLaunch" for the X-axis attribute and "CumulativeNumberOfBugs" for the Y-axis attribute.

Descending Checkbutton to denote whether or not the values along the axis are monotonically increasing or decreasing. If you want the values to increase, do not check the "Descending" checkbutton. By default, the "Descending" option is not turned on.

Color Sets the color of the axis components(label, ticks, etc).

Label Sets the title of the axis. By default, *nxqddb* populates this value with the attribute name that you've chosen for the Axis. However, you have the ability to modify this value if you so desire.

Title font Sets the font of the title using the standard font selection dialog box(see 8.3).

Tick font Sets the font of the axis ticks using the standard font selection dialog box(see 8.3).

Range Analysis This checkbutton turns on/off range analysis. If turned on, then *nxqddb* will perform the range analysis based on the settings you choose. If turned off, then *nxqddb* will ignore your settings and exclude range analysis.

Range Dates and strings can be used on the X-axis for performing range comparisons. Dates can be displayed in any format that strftime can handle. You can also choose to specify ranges for summaries; some examples are listed in table 8.3

Dates are processed with the range specification, and like values are lumped together for sum, avg, stddev, min, max, count, or prod operations on the Y values. The X-axis is sorted by ASCII comparison. Some examples of aestheticly pleasing range/format pairs are illustrated in table 8.4.

Format Tell *nxqddb* how you want your data formatted.

Range analysis Drop-down options Choose the type of range operation from the pull-down menu shown in figure 8.4.

Tick width Sets the width of the axis and tick lines. *nxqddb* defaults this value to one(1).

Format	*What it means*
%m	month of year
%d	day of month
%y	year
%y/%m	year/month
%y/%m/%d	year/month/day
%w	day of week
%W	week of year
%H	hour of day

Table 8.3: Sample range values

Range	*Format*
%m	%b
%w	%a
%y/%m/%d	%d/%m/%Y

Table 8.4: Aestheticly pleasing range values

Rotate label You can specify the degrees to rotate the axis labels.

Log scale If turned on, the scale of the axis is logarithmic. If turned off, the scale is linear. The default scale is linear.

Loose fit If turned on, *nxqddb* makes the limits of the axis loose fitting at the outer tick intervals. The default *nxqddb* loose fit option is turned off.

Show ticks If turned on, *nxqddb* will display minor ticks. If turned off, minor ticks will not be displayed. The default is to show ticks.

Tick subdivisions Indicates the number of minor axis ticks that are displayed. If number is two, only one minor tick is displayed. If number is one, then no minor ticks are displayed. *nxqddb* defaults this option to two.

Major tick length Sets the length of major ticks. If the value is negative, the axis will be inverted with ticks drawn pointing towards the plot. *nxqddb* defaults this option to 0.1i

Step size Sets the step size between the major axis ticks. You must enter a value greater than zero and less than the axis range or your entry will be ignored and the step size is automatically calculated based on the element data points.

8.6.3 Configuring options for each individual graph

In our Acme graph scenario, we created four separate search criteria. Hence, on the "Graph Setup" window, we'll see our four graphs. We can configure each of these graphs separately by setting the following options:

Legend Enter the text to appear as the legend. In our example, we entered (1) nasty, (2) cosmetic, (3) fluff and (4) documentation as our 4 legends.

Color Clicking the "Color" button displays a window for you to specify a color for the graph

Width Sets the line width.

Symbol Drop down list of symbols to choose from including: none, circle, cross, diamond, plus, etc.

Style Sets the style of line connecting the plotted points. The style may be one of solid line or dash style lines chosen from the drop-down choice box.

Show Checkbutton that when turned on displays the legend. If turned off, then the legend will not be shown.

8.7 Generating the Acme Graph

After you've completed the graph setup, click the "Run report" button on the "Reports" window. *nxqddb* will redisplay the "Graph Setup" window to allow you to make any changes to the graph setup before generating the graph. Click the Ok button to continue generating the graph. *nxqddb* will then display the standard print dialog box to prompt you for the print destination. Choose "Preview on screen"; you should see a window similar to the one shown in figure 8.5.

8.7.1 Graph display options: grid, cross-hairs, zooming, Invert X/Y

On the graph window shown in figure 8.5, notice that *nxqddb* provides three checkbuttons:

Grid This checkbutton turns on/off a grid.

Cross-hairs This checkbutton turns on/off cross-hairs. If turned on, you'll see an intersecting vertical and horizontal line. This option is useful to locate the X and Y values for a point on the graph.

Zooming This checkbutton turns on/off zooming. To zoom in on a part of your graph, follow these steps:

1. Check the "Zooming" checkbutton. The checkbutton is turned on when the red checkbutton square is depressed.
2. Locate where you want the upper left corner of your desired zoom area and then click the mouse. Now move the mouse to the lower right corner — notice how a rectangle follows your mouse movement highlighting your area. Click the mouse a second time. At this point, *nxqddb* will zoom on the area you selected. You can continue zooming in on your graph. To return back to the original, just click the "Zooming" checkbutton.

Invert X/Y This checkbutton toggles which attribute is on the X/Y axis.

Once you've displayed the graph shown in figure 8.5 you have the option to (1) make further customizations using the "Apply" button or (2) print the graph.

8.8 Applying customizations to your graph

Notice that when you run your graph and are presented with the "Graph" window as shown in figure 8.5, you have a "Setup" button available. Clicking the "Setup" button displays the "Graph Setup" window (see figure 8.2). You can change any of the setup options from this window and then immediately see the results by choosing the "Apply" button. The **Apply** button is available on the "Graph Setup" window only when you have your graph displayed on screen.

Try changing an option on the "Graph Setup" window, say the *Background* and then click the "Apply" button. Notice how the graph is immediately updated to reflect your change.

8.9 Printing your graph

To print your graph, click the "Print" button on the "Graph" window (see figure 8.5). *nxqddb* will display the "Printing options" window shown in figure 8.6 *nxqddb* will remember your settings from the previous time you printed a graph. The settings are fairly self-explanatory:

Orientation You can print your graph in either Portrait or Landscape mode.

Expand to full page If turned on, this option will tell *nxqddb* to stretch your graph to fill the entire page.

Center graph on page If turned on, *nxqddb* will automatically center your graph on the page.

Backgrounds/Borders If turned on, *nxqddb* will print graph backgrounds and borders.

Colors You can choose one of three available color output: (1) full color, (2) gray scale, and (3) black/white.

Page width You can specify the page width. For example, "8.5i" tells *nxqddb* that your pages are 8.5 **i**nches wide.

Page height You can specify the page height. The default is *11.0i* (eleven inches).

Graph width You can tell *nxqddb* to resize your graph to a certain width (shrink or expand) by specifying a non-zero value in this field. A zero value tells *nxqddb* to print a normal graph width.

Graph height Like the graph width, you can tell *nxqddb* to resize your graph height. A zero value tells *nxqddb* to print a normal height.

Left/right border This field tells *nxqddb* an amount of space to leave as left and right borders.

Top/bottom border This field tells *nxqddb* an amount of space to preserve as top and bottom borders.

Pressing the "Cancel" button will abort the print and return you to the "Reports" window. Pressing the "Print" button displays the "Printer Selection" dialog box to allow you to direct your output to the printer or other medium.

8.10 Range Analysis

The steps to performing range analysis are described below:

1. Be sure that you have a properly defined schema. If you want to perform date range analysis, then your schema must contain a field that has the keywords "**type date**". If you want to perform a numeric range analysis, then you must have a field that contains either "**type integer**" or "**type real**".

2. From the graph/barchart setup window, select the X/Y-axis attributes that you want to receive the range analysis. For example, if you want to analyze web hits for 1994 through 1996, you would select your Date attribute for the X-axis and your WebHit attribute for the Y-axis.

3. Click the "*Range analysis*" checkbutton shown in figure 8.7. You must have this checkbutton turned on for *nxqddb* to perform range searching. If this button is turned off, then *nxqddb* omits any range calculations.

4. Enter the range and format parameters. If your X-axis attribute is a date field, then *nxqddb* provides a range/format pull-down menu as shown in figure 8.8. Clicking one of the options populates the range and format entry boxes. Of course, if you choose "User-defined" value from the pull-down, *nxqddb* places the cursor in the range entry to allow you to define your own range and format values. Most users will likely opt to use one of the range/format combinations from the listbox. If your X-axis is not a date field, then you must specify a positive integer in the range entry. For attributes defined as real, the integer value you enter in the format field tells *nxqddb* the precision (i.e. the number of digits to the right of the decimal) of your Y-axis. If the Y-attribute was not typed as integer, then the format field is ignored.

5. Run your report normally.

6. After your report has run, you can quickly modify your range/format settings and "Apply" your updates instantly while the graph is still displayed. *Qddb* is able to quickly display new ranges and formats because of the way it generates intermediate data. The details of how *Qddb* does this data management is beyond the scope of this text; however, you can appreciate the speed of these graph updates. For example, earlier we mentioned looking at say, web hits for 1994-1996. You can quickly analyze the data over several views such as monthly, yearly or even look at what the hits were on each day of the week. Of course, you can look at the sum, average, or any of the other operations available from the pull-down menu.

8.11 Example: range analysis of web hits

Suppose Acme has a web site and wants to view information on who and how often people visit their web pages. Also, suppose Acme has saved all of its log files for the past three years.

The Acme system administrator could write a quick *Qtcl* (i.e. Qddb tcl script using Fx interface) script that parses the web log files and populates a *Qddb* database. Let's assume the system administrator created a schema shown in figure 8.9.

This schema allows you to look at (1) total number of hits, (2) number of hits by host and (3) number of hits by domain.

Let's assume you already wrote your *Qtcl* script and now have a *Qddb* database populated with all the hits over the past three years.

By now, you should be familiar with generating reports. If you need more practice with generating reports, refer to chapter 7.

For all of the graphs and barcharts in this section, choose the following settings:

Range analysis checkbutton should be turned on.

X-axis attribute should be set to **Hosts.Accesses.Date**

Y-axis attribute should be set to **Hosts.Number**

Now, for each different graph we'll just change the Range/Format parameters by clicking a different pull-down option. The following sections illustrate a few of the ways to view your web site data hits.

8.11.1 Graph of daily hits by month

Choose the **Month of year (3-letter) option** to display the graph shown in figure 8.10.

8.11.2 Graph of average daily hits by day of week

To obtain the graph shown in figure 8.11, choose the "avg" value from the range operations pull-down menu and then choose the **Day of week (3-letter)** option.

8.11.3 Barchart of yearly summaries

Choose the "sum" value from the range operations pull-down menu. Then, choose the "Year" pull-down menu option. After applying these changes, you'll see a chart similar to one shown in figure 8.12.

8.11.4 Barchart of summary by month

Be sure the "sum" value is chosen in the operations pull-down menu and then choose "Month of year (3-letter)". After applying these settings, your barchart should resemble the one shown in figure 8.13.

8.11.5 Barchart of summary by day of week

Be sure the "sum" value is chosen in the operations pull-down menu and then choose "Day of week (3-letter)". Your barchart should resemble the one shown in figure 8.14.

8.11.6 Summary

This section has provided the steps to performing range analysis in creating graphs and barcharts. We have presented a detailed scenario of how to create useful representations of data for a fictitious company. Specifically, we looked at a sample schema for maintaining web site log information and how to generate different views of this data using *Qddb*'s powerful range analysis tool.

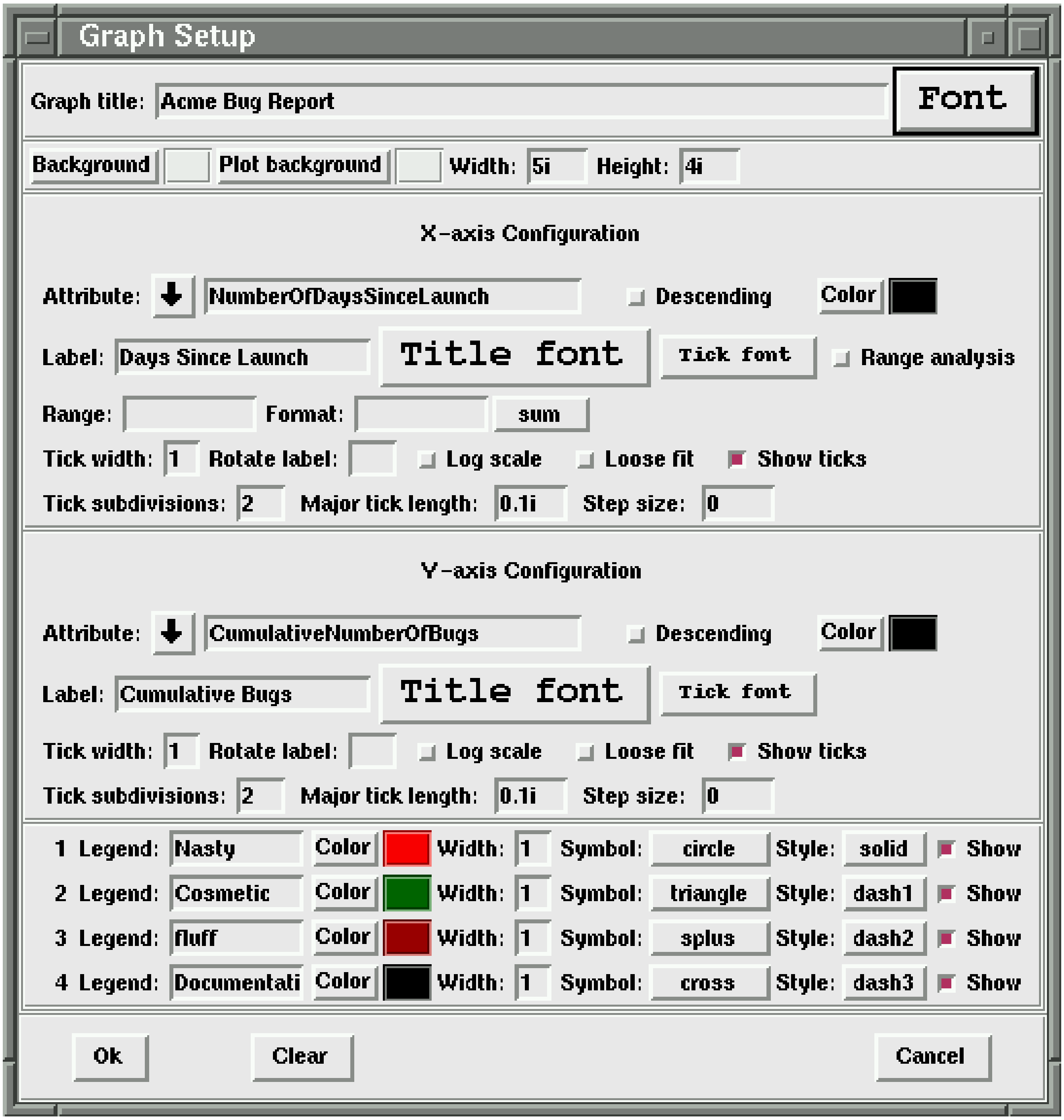


Figure 8.2: Window to setup your Acme graphs

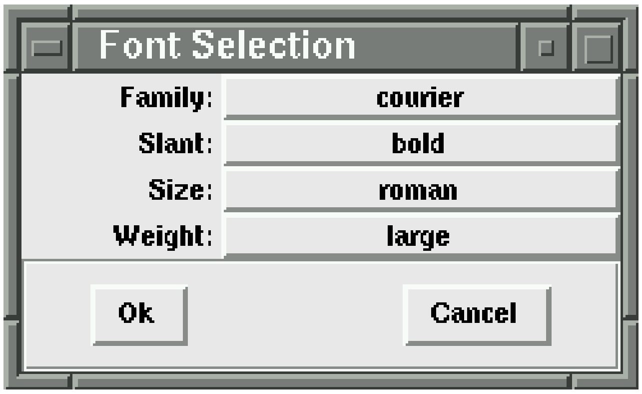


Figure 8.3: Font dialog box for graphs/barchart settings

Figure 8.4: Range operation pull-down menu

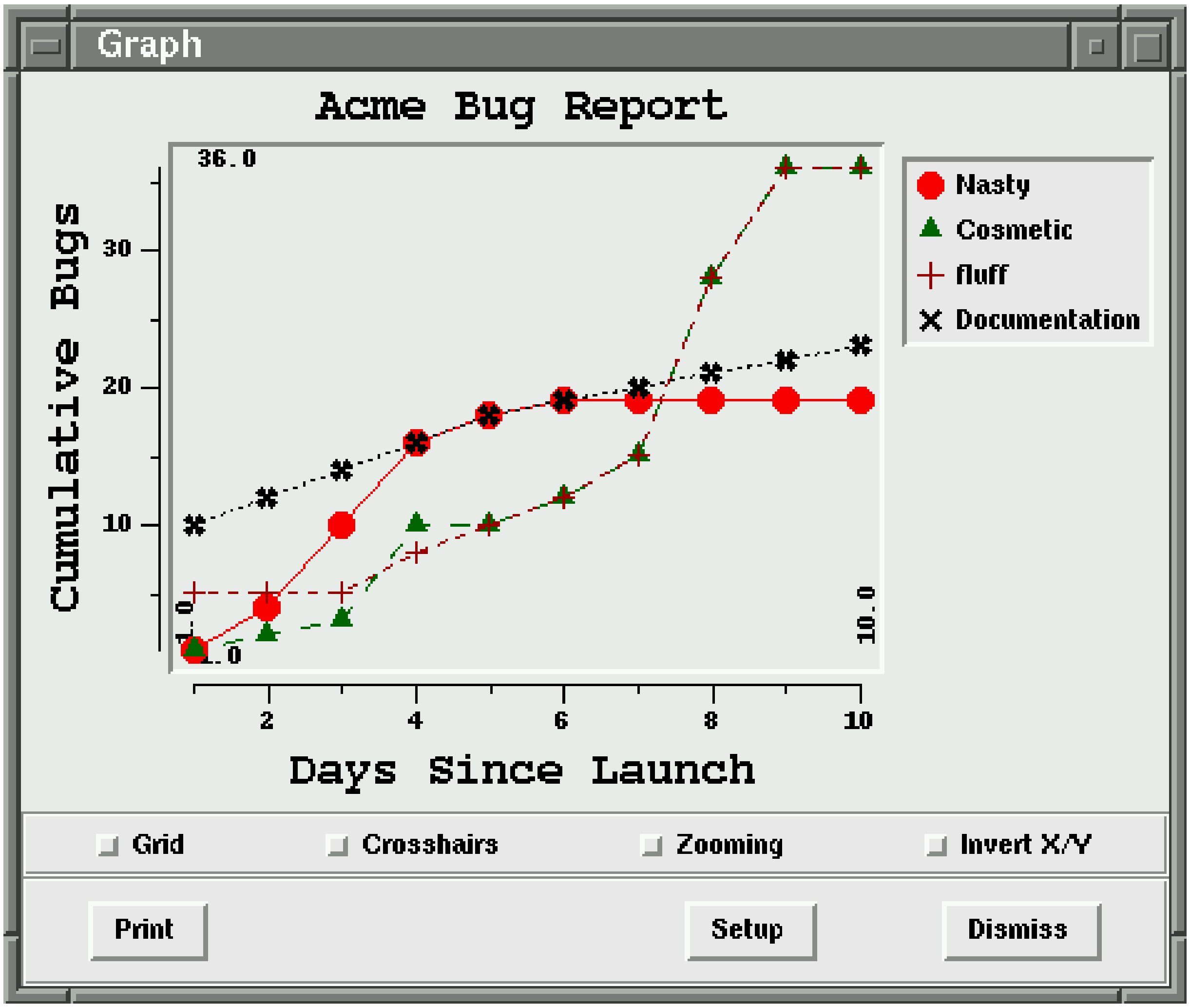


Figure 8.5: Acme graph of software bugs

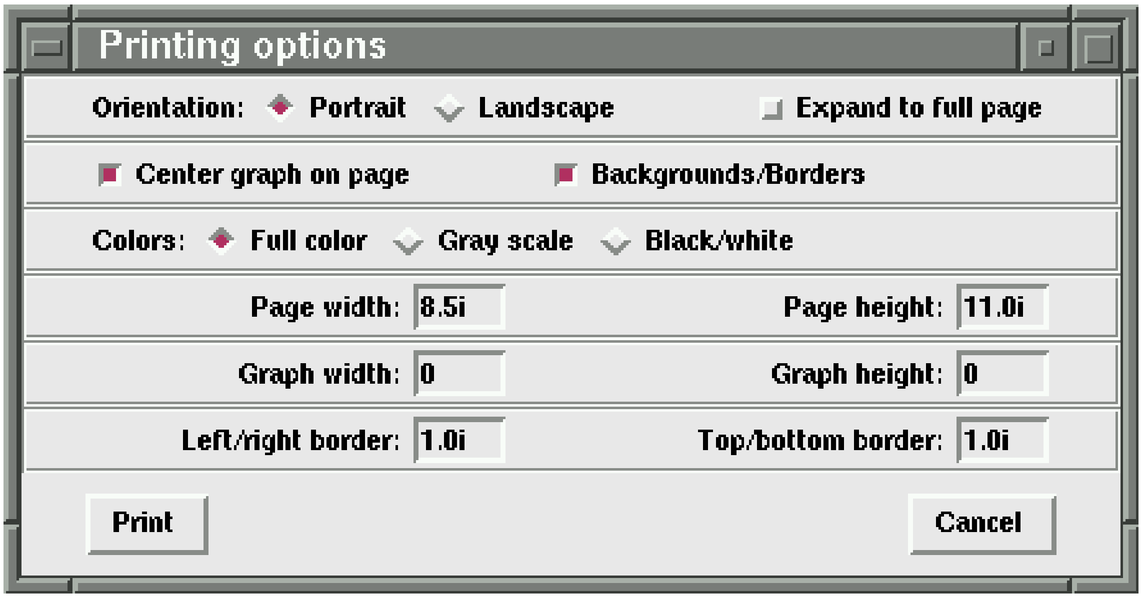


Figure 8.6: Printing Options Window

Figure 8.7: Range analysis checkbutton

Figure 8.8: Range/format options for date range analysis

```
Use cached hashing
CacheSize=100
HashSize=20000
Use reduced attribute identifiers
Use Cached Secondary Search

Domain separators ""
Hosts (
    Name separators ""
    Accesses (
        Date type date verbosename "Date/Time"
            format "%m/%d/%y %H:%M:%S"
            Page separators "  /"
    )*
    Number type integer verbosename "Number accesses (host)"
)*
Number type integer verbosename "Number accesses (domain)"
Comments
```

Figure 8.9: Web hit schema

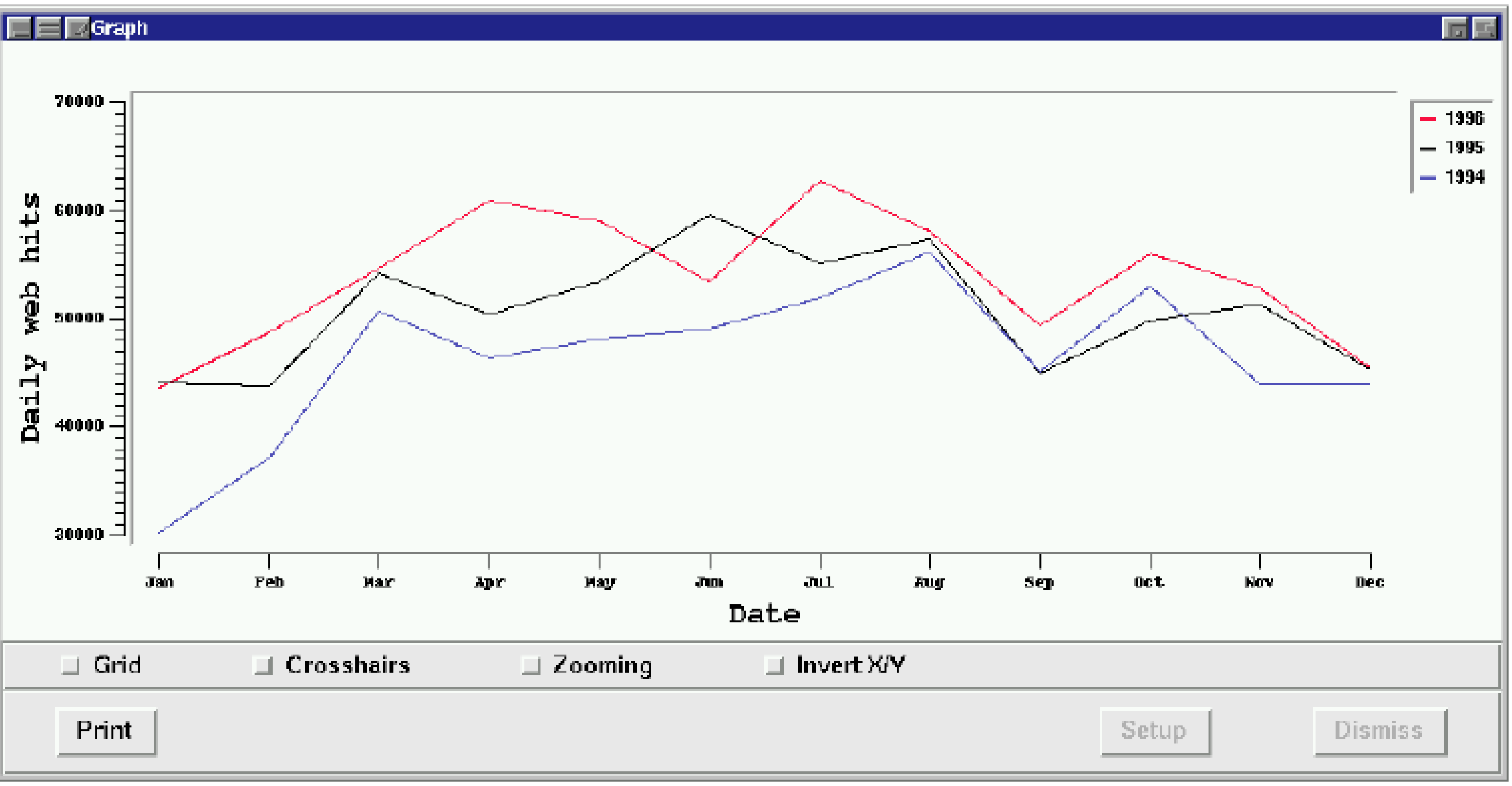


Figure 8.10: Generated from “Month of year (3-letter)” option

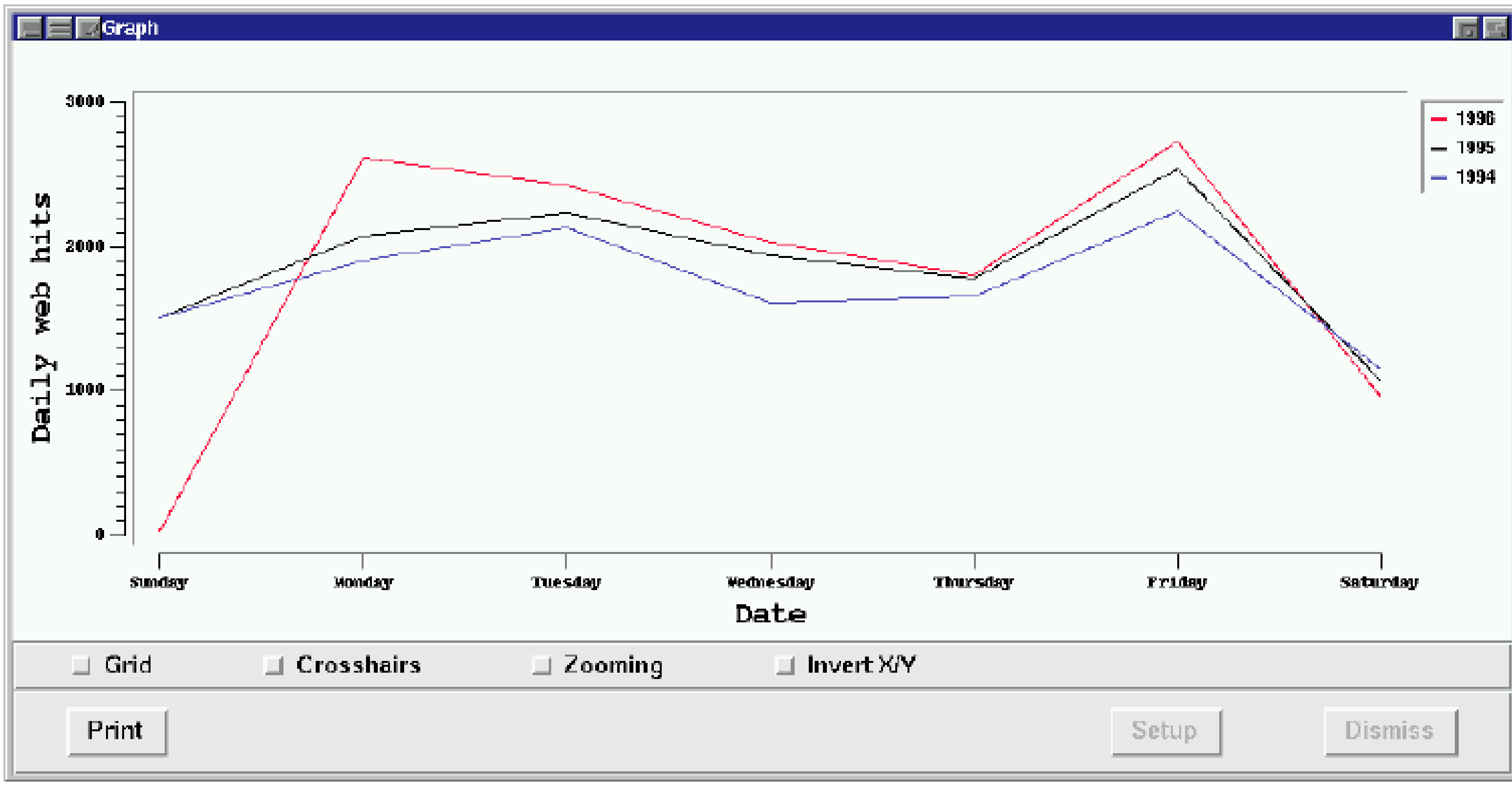


Figure 8.11: Generated from “Day of week (3-letter)” option

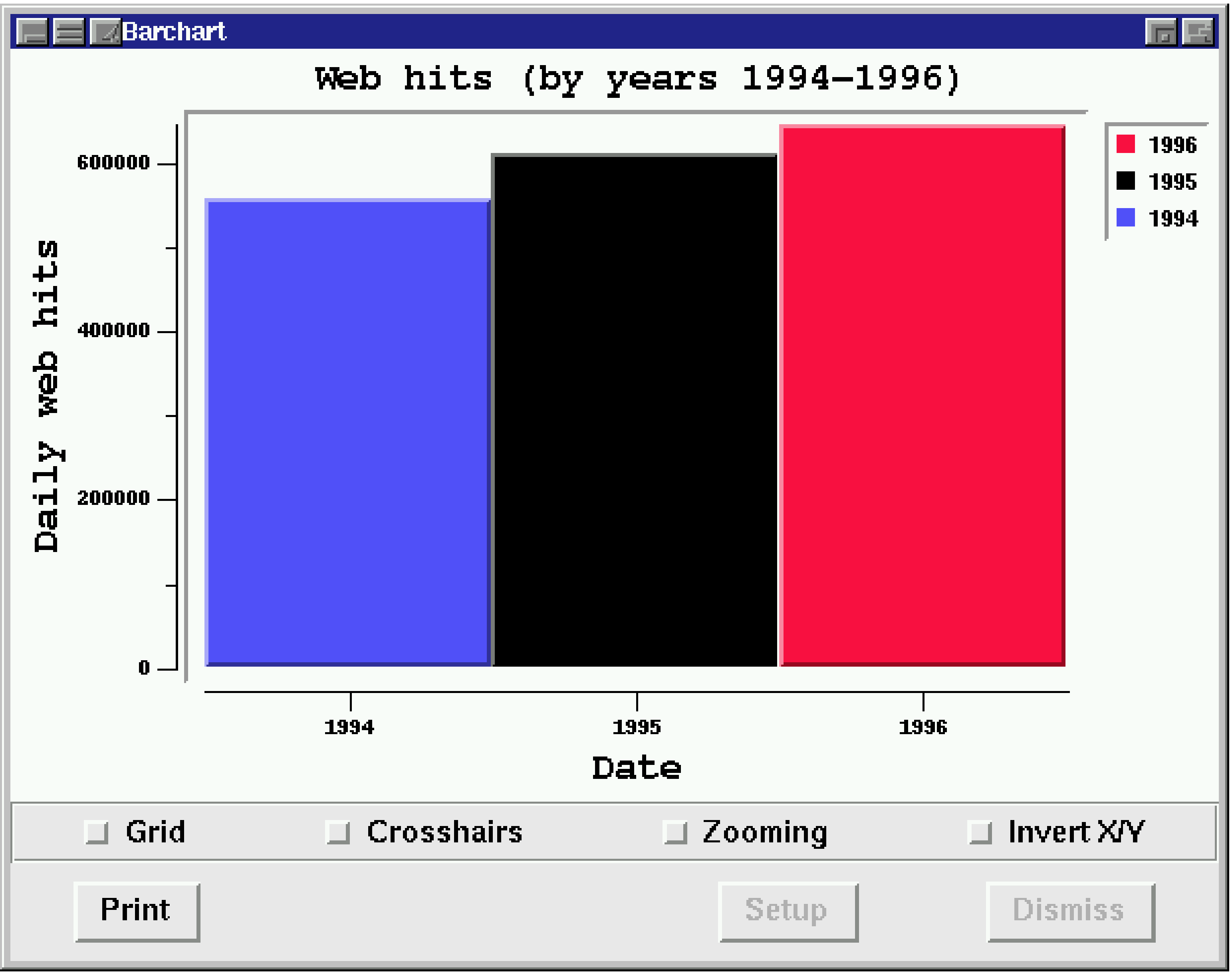


Figure 8.12: Generated from “Year” pull-down menu option

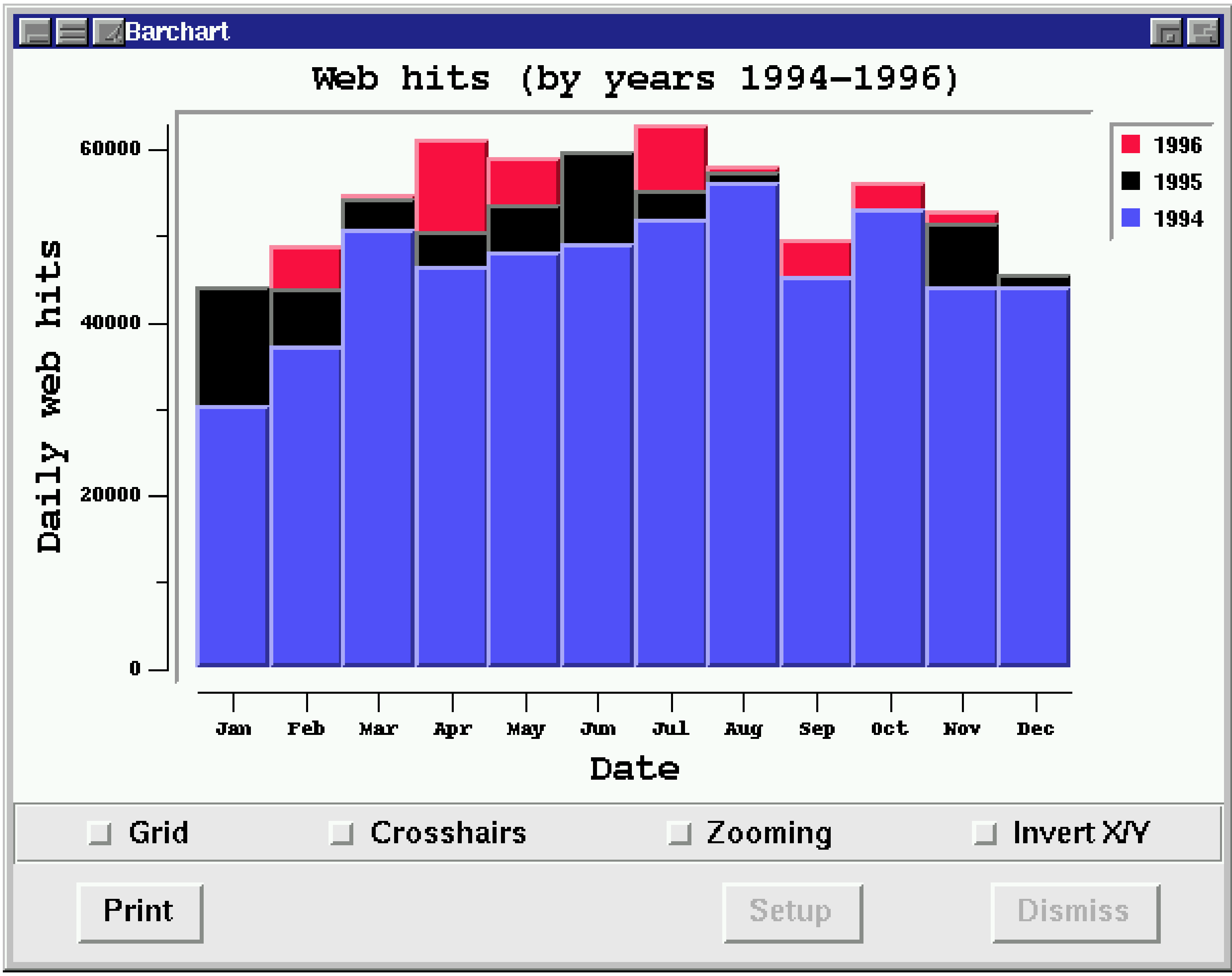


Figure 8.13: Generated from "Month of year (3-letter)" option

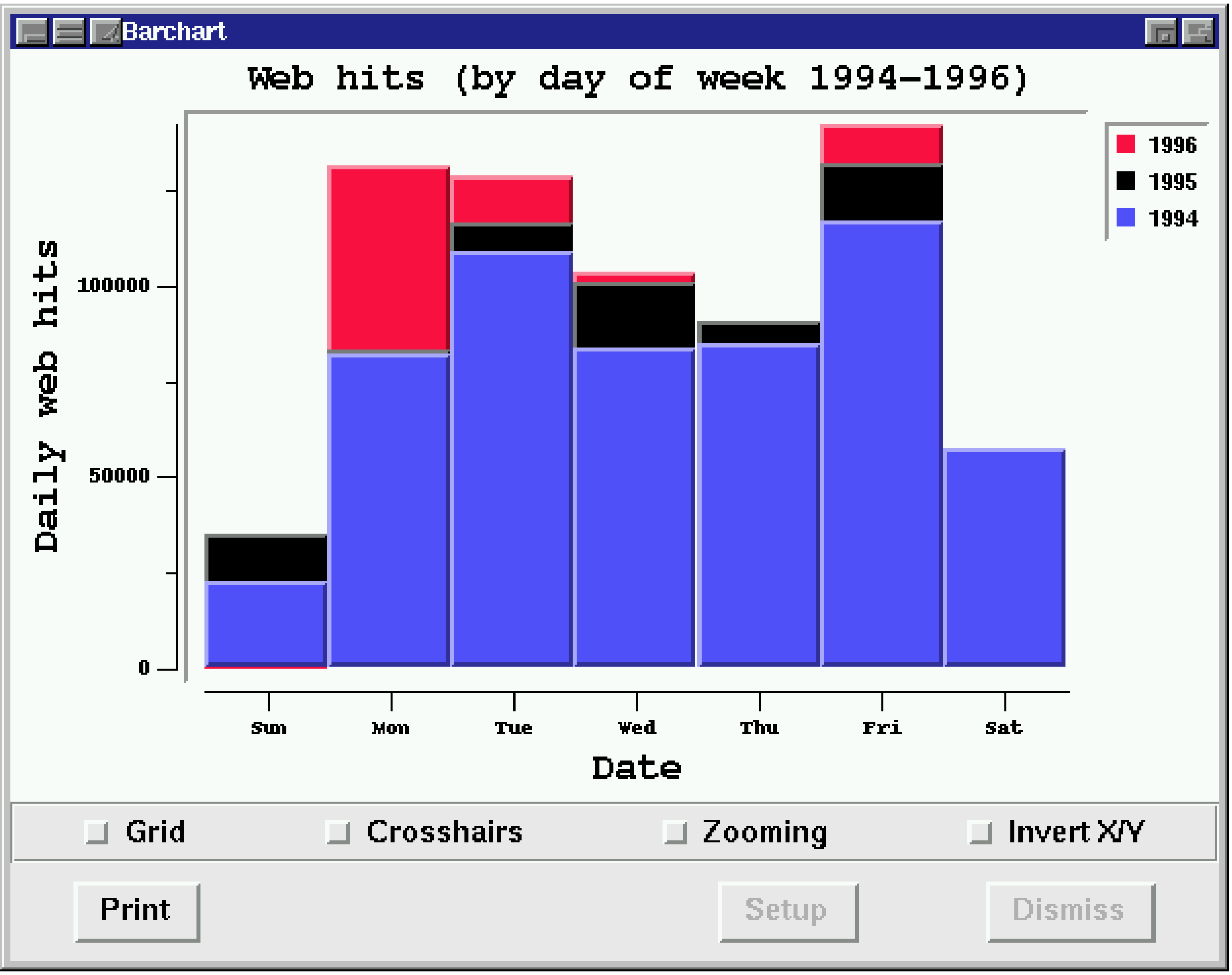


Figure 8.14: Generated from “Day of week (3-letter)”

Chapter 9

Barcharts

Barcharts are created just like graphs. So, if you've read the previous chapter on graphs, you already know most of what you need in order to generate barcharts. There's only a few differences and these differences reside on the "Barchart Setup" window. Basically, you follow the same steps that you do for graphs; except, you click the "Barchart" buttons instead of the "Graph" buttons. And, you set your barchart options accordingly from the "Barchart Setup" window.

9.1 Barchart Setup Window

Clicking the "Barchart" button on the Reports "Setup" panel displays the barchart window shown in figure 9.1. The resulting output barchart from the figure 9.1 settings is shown in figure 9.2.

There are many options that you can configure for your barcharts; each of these options are briefly described in the next section.

9.1.1 General barchart options

Barchart title You can specify a title to appear at the top of your barchart.

Barchart title font Clicking the "Font" pushbutton to the right of the "Barchart title" entry box displays a dialog box(see figure 8.3)

Background Sets the background color of the barchart, including the margins and legend but not the plotting area.

Plot background Sets the background color of your plotting area.

Barchart width Sets the requested width of the barchart window. For example, a value of "4i" sets the width of the window to four inches.

Barchart height Sets the requested height of the barchart window

Bar alignment You can specify one of three bar alignments: (1) normal, (2) aligned, and (3) stacked. The stacked alignment is particularly useful for viewing the individual charts as a grand total.

Bar relief You can specify one of three different appearances for the bars: (1) raised, (2) flat and (3) sunken. This setting affects every bar in the graph. You also have the option to specify the stippling, color, and width for each individual bar (see section 9.1.3 for details on the individual bar settings).

9.1.2 Configuring options for the X/Y-Axis

Attribute Use the drop-down listbox to choose the attribute that you want to plot. In our Acme scenario, we chose "NumberOfDaysSinceLaunch" for the X-axis attribute and "CumulativeNumberOfBugs" for the Y-axis attribute.

Descending Checkbutton to denote whether or not the values along the axis are monotonically increasing or decreasing. If you want the values to increase, do not check the "Descending" checkbutton. By default, the "Descending" option is not turned on.

Color Sets the color of the barchart

Label Sets the title of the axis. By default, *nxqddb* populates this value with the attribute name that you've chosen for the Axis. However, you have the ability to modify this value if you so desire.

Title font Sets the font of the title using the standard font selection dialog box(see 8.3).

Tick width Sets the width of the axis and tick lines. *nxqddb* defaults this value to one(1).

Log scale If turned on, the scale of the axis is logarithmic. If turned off, the scale is linear. The default scale is linear.

Loose fit If turned on, *nxqddb* makes the limits of the axis loose fitting at the outer tick intervals. The default *nxqddb* loose fit option is turned off.

Show ticks If turned on, *nxqddb* will display minor ticks. If turned off, minor ticks will not be displayed. The default is to show ticks.

Tick subdivisions Indicates the number of minor axis ticks that are displayed. If number is two, only one minor tick is displayed. If number is one, then no minor ticks are displayed. *nxqddb* defaults this option to two.

Major tick length Sets the length of major ticks. If the value is negative, the axis will be inverted with ticks drawn pointing towards the plot. *nxqddb* defaults this option to 0.1i

9.1.3 Configuring options for each individual bar

In our Acme barchart scenario, we created four separate search criteria. Hence, on the "Barchart Setup" window, we'll see our four barcharts. We can configure each of these bars separately by setting the following options:

Legend Enter the text to appear as the legend. In our example, we entered (1) nasty, (2) cosmetic, (3) fluff and (4) documentation as our 4 legends.

Color Clicking the "Color" button displays a window for you to specify a color for the barchart.

Width Sets the line width.

Stipple Sets the style of the bar. The style may be solid or one of nine different dashed stippling styles chosen from the drop-down choice box.

Show Checkbutton that when turned on displays the bar. If turned off, then the individual bar will not be shown.

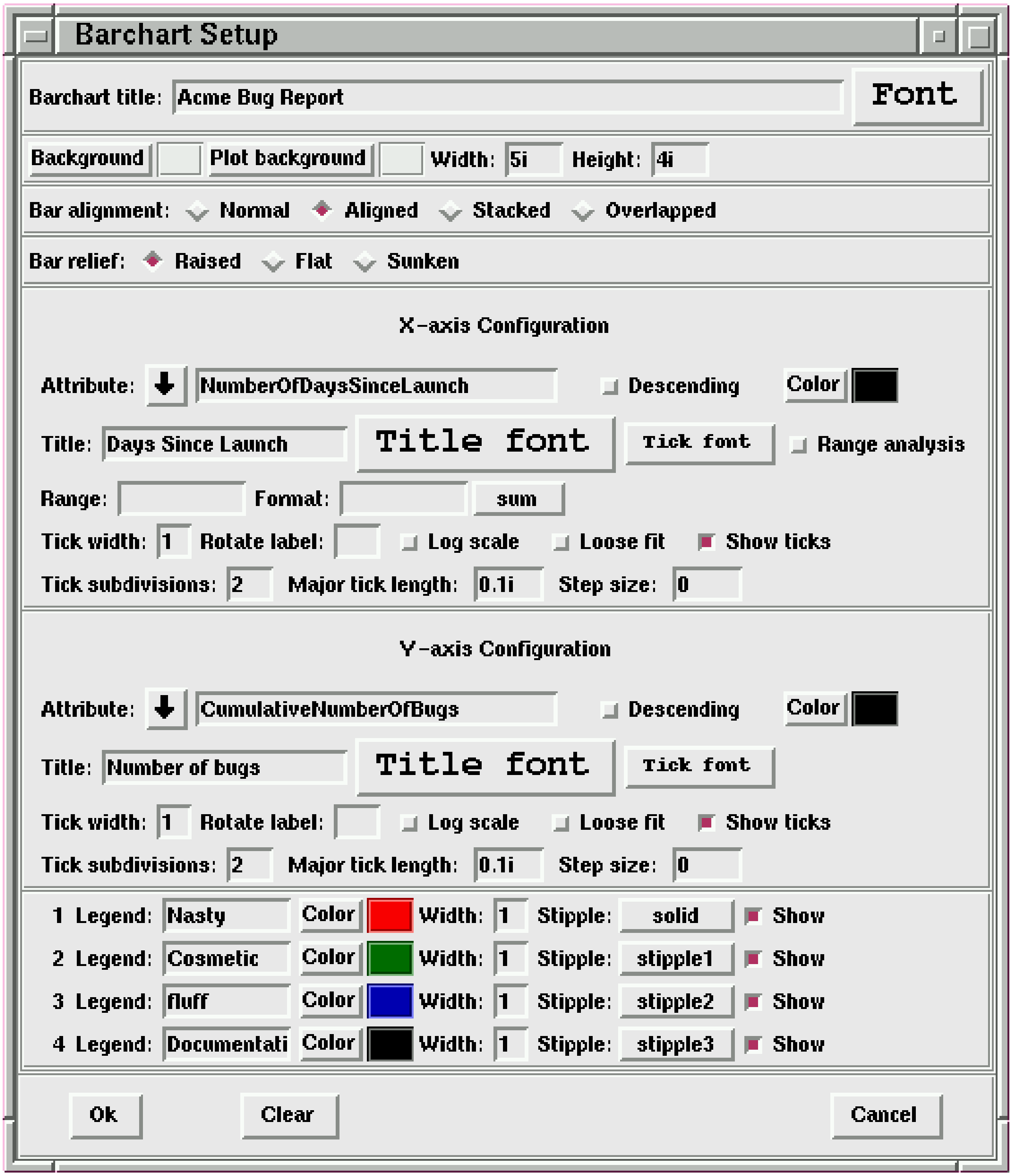


Figure 9.1: Barchart setup window

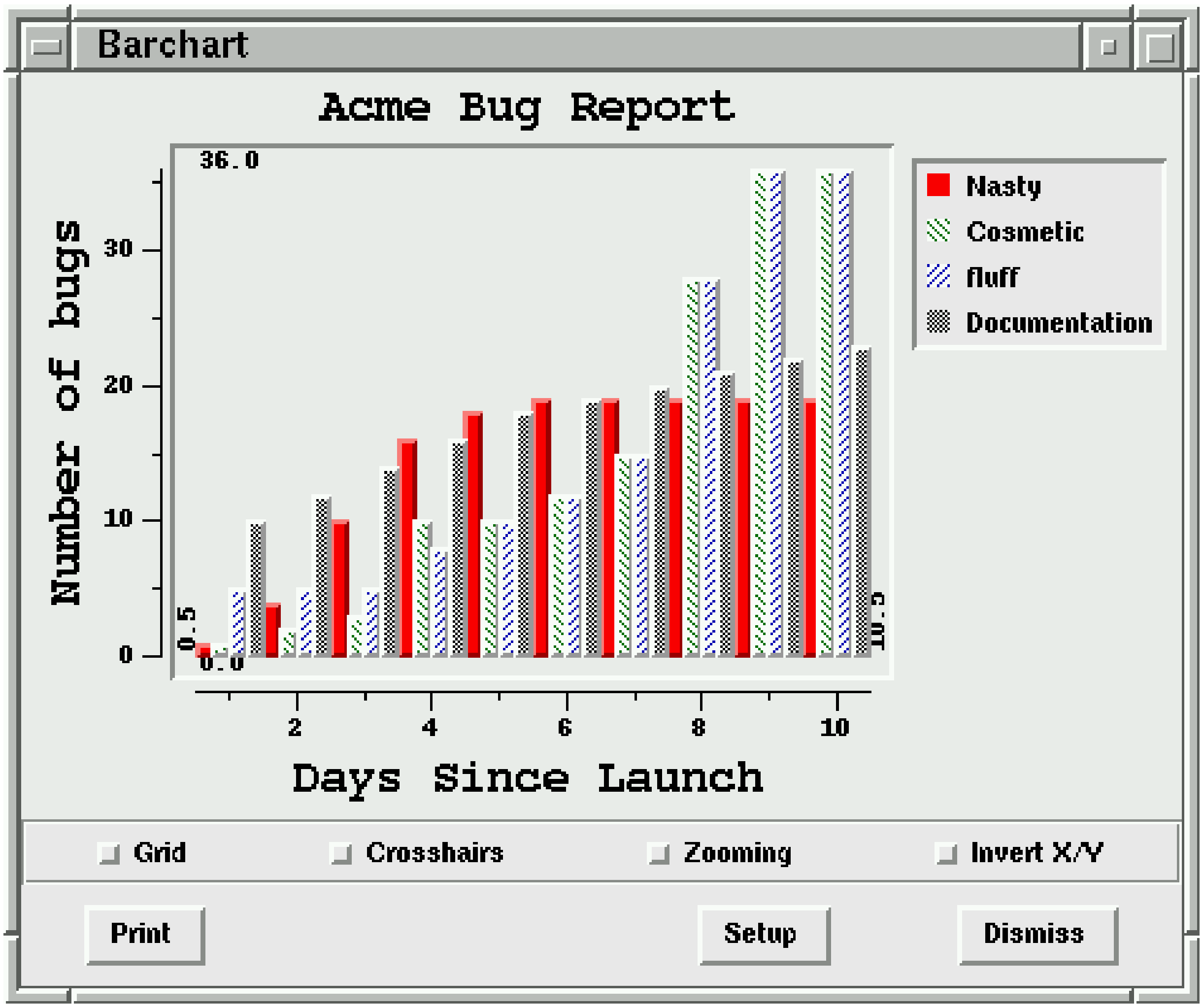


Figure 9.2: Barchart for Acme example

Chapter 10

Importing/Exporting

10.1 Importing data into your Qddb database

Qddb 1.44 includes a simple CSV (comma separated values) import mechanism for importing flat tables into your Qddb database. CSV format uses double quotes (") to denote string data, and commas to separate the fields. Double quotes within a string value must be backslashed. Qddb can import almost any date format, and dates may be encapsulated by double quotes if desired. Dates of the form "01/02/98" are interpreted by the current date style setting under the Configure-"Date Style" menu.

The following is a valid CSV format file with five columns. Note that Qddb is somewhat flexible when it comes to reading the values. Numbers, dates, and strings may all be encapsulated in double quotes. Any row may have fewer or more items than the other rows. The values are imported from left to right given the selected fields.

```
01/02/96,67,2213.32,42,"George"
01/03/96,32,1193.32,9,"Jackie"
01/04/96,43,1492.48,13,"Sam"
01/05/96,47,2127.89,1,"Gilbert"
01/06/96,23,881.32,42,"Ricky"
01/07/96,0,0.00,0.00,"Liz"
01/08/96,11,334.01,0.00,"Larry"
01/09/96,24,895.85,0.00,"Sharon"
01/10/96,46,1937.56,0.00,"Sally"
01/11/96,36,1774.73,0.00,"Martha"
01/12/96,32,1627.10,0.00,"Debbie"
01/13/96,18,544.41,0.00,"Jimbo"
```

To begin the import process, you first click the File-Import menu item. You will see the screen shown in Figure 10.1. Only one import format is currently available: CSV. You can read your CSV file from a floppy (assuming your system is equipped with mtools) or from a file. After you have selected a file, click "Ok" and you will see the window shown in Figure 10.2 after the CSV file has been read.

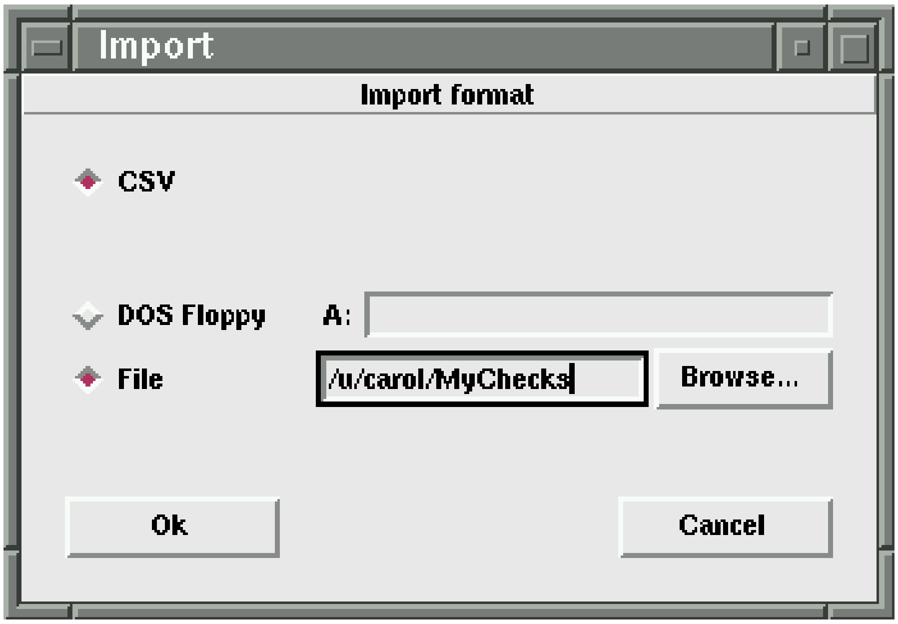


Figure 10.1: Importing data into your Qddb database

To select columns for import, simply click the attributes in the "Available Attributes" listbox that correspond to the columns in the file. The expected import format of the CSV file is listed in the "Import Columns" listbox; the topmost items correspond to the leftmost columns of the file. If you frequently import files, you can save the import settings by choosing the "Write to file" button and reload them later with the "Read from file" button. Choosing "Ok" at this point completes the import process and creates the new records.

10.2 Exporting data from your Qddb database

Qddb 1.44 allows you to export the contents of reports or the search results in either CSV or AS (Applix^{tm}[1] Spreadsheets) format. More popular export formats will appear on a regular basis during the 1.44 release cycle.

The export window allows you to select the format, as well as the style of any dates you

[1] Applix^{tm} is a registered trademark of Applix Inc.

might be exporting. You can omit column titles from the output by clicking the "Column Titles" checkbutton to the off position. Note that not all date styles are supported by all export formats. Qddb's export tool allows you to export only legal date styles for the selected format (note that the listbox containing the available date styles changes when you change the export selection). CSV format is the most restrictive, and is not completely standard. Qddb attempts to closely follow the defacto CSV standard by not exporting double quotes in string values or date styles containing commas.

To export the search results, simply click the "Export" button in the Search Results window. A window similar to the one shown in Figure 10.3 will appear. Choose your export format and date style, then select your export destination. Clicking "Ok" will complete the export.

To export from the report generator, click the "Export" checkbutton from the report options. You will be prompted for the export type and date style with same window shown in Figure 10.3.

AS format exports each search in a report to a separate "tab" in the spreadsheet. The search title provides the name of the tab; the default is used if no name is provided.

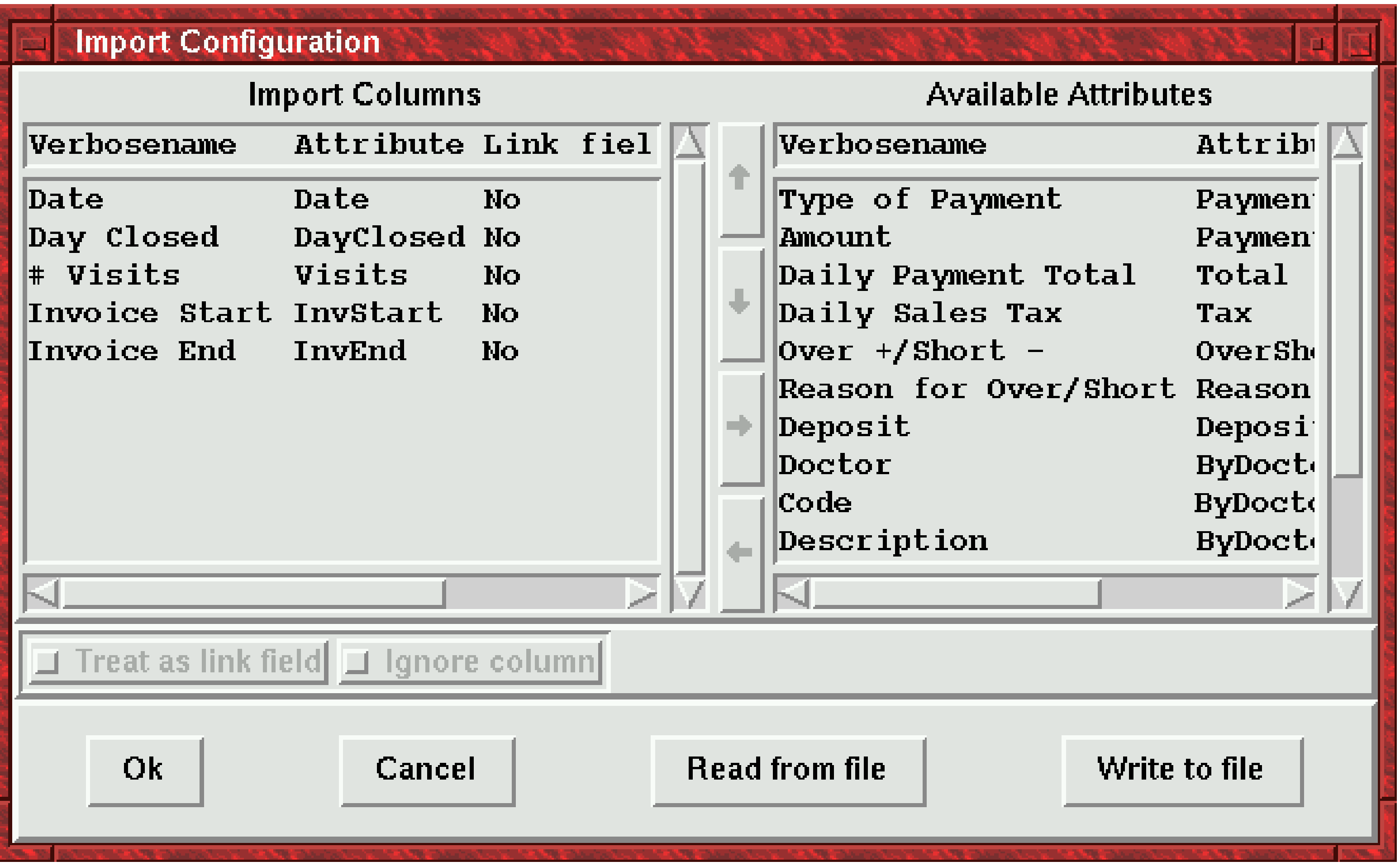


Figure 10.2: Selecting columns for import

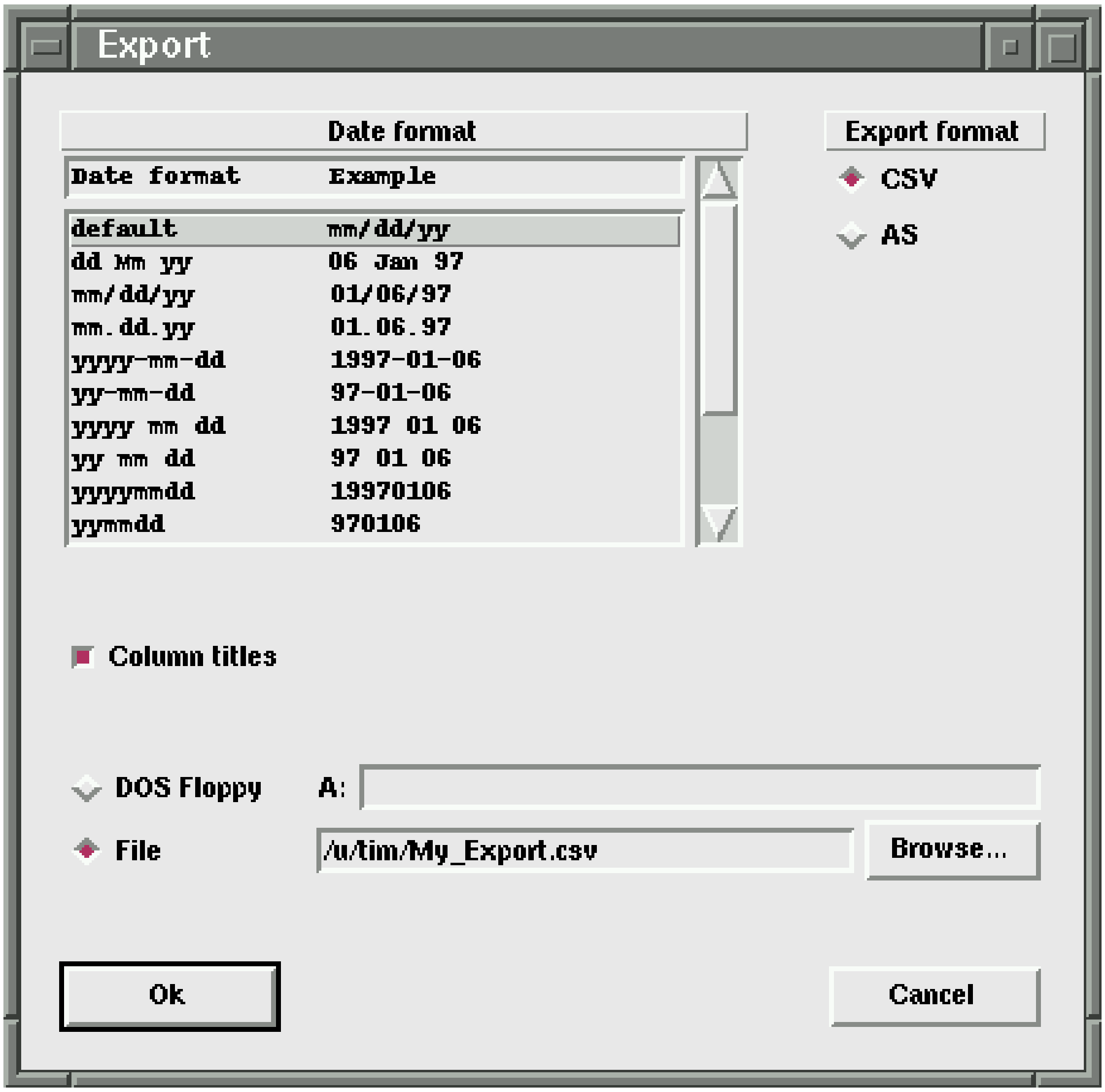


Figure 10.3: Exporting data from your Qddb database

Chapter 11

Schemas

11.1 Creating a *Qddb* Database

The Schema file is the only file that you need to edit in order to setup a *Qddb* database. Remember from chapter 2, your Schema file (along with the other *Qddb* files) is automatically generated when you enter the command: "`$ qnewdb <DatabaseName>`"

For example, if you enter the commands shown in figure 11.1

```
$ cd ~/Databases
$ qnewdb MyRecords
```

Figure 11.1: Steps to creating a new Database(Schema)

Qddb will automatically create the "`~/Databases/MyRecords`" subdirectory. Inside this subdirectory, *Qddb* creates several directories/files; each of these files are briefly described below:

Schema Immediately after performing the "`qnewdb MyRecords`" command you need to edit the Schema file (using your normal editor such as vi or emacs) to describe your database. *Qddb* uses this Schema file to setup your database.

Additions This directory initially contains only one file — NextNumber. NextNumber is a file used by *Qddb* to determine the "next number" of the filename that gets stored in the Additions directory. This number is actually a sequential number that gets incremented each time an entry is added to your database. Each time you add a new record to your database, *Qddb* determines what number this entry is (by reading the "NextNumber" file) and then stores the information in the file beneath the Additions directory. The filename is the number of the entry. For example, the first time you save an entry from *nxqddb*, a file named "1" is created in the Additions directory. The second entry has the filename "2", the third "3", and so forth. These files are added to the "Database" file and removed from the Additions directory whenever you do a "qstall" command.

Database This file is initially empty when you first do a "`qnewdb`" command. The Database file gets populated when you perform the "qstall" command. The qstall (short for *Qddb* stabilize all) command stabilizes the *Qddb* database. Meaning, *Qddb* reads all the Additions' files to add to the database, and then regenerates its indexes. If you *really* know what you're doing, you can even edit this Database file (however we don't recommend this since its easy enough to do within *nxqddb*!)

Indexing files *Qddb* will create its indexing files and store them in your Database's subdirectory: HashTable, KeyIndex, Database.key, Index and NumericIndex. You should *not* modify these files.

Changes *Qddb* creates new files in the Changes subdirectory to store any modifications that you make to existing tuples. You should not make any modifications to the files stored in this directory.

11.2 The format

The `Schema` file begins with optional storage format options:

- `HashSize = <Number>`: Sets the number of buckets in the HashTable file. If you leave this number out, Qddb will use a default of 20,000.
- `HashType = <Number>`: Sets the hash function to `<Number>`, where `<Number>` may currently be *0* or *1*. *0* is the default hash function; *1* is a newer function designed for data with repetitive prefixes/suffixes. If your data contains numbers, WWW URLs, or source code, you may find that HashType *1* performs much better than the default. You can experiment with your own HashType by editing the function `Qddb_HashValue` in `Lib/LibQddb/Hash.c`.
- `Use Cached Hashing`: Instructs *qstall* and *qindex* to build a HashTable file suitable for caching hash values instead of reading the entire file at startup.
- `CacheSize = <Number>`: Sets the maximum number of hash values to cache. Ignored unless "`Use Cached Hashing`" is specified.
- `Use Reduced Attribute Identifiers`: Instructs *qstall* to build a RedAttrIndex structure file holding the full attribute identifiers. This option reduces the size of

the Database and Index files, because the lengthy attribute identifiers and instance numbers are represented by a shorter base-36 number. You can add and delete this option from the Schema at any time, but you must immediately restabilize with *qstall*. *qstall* will notice that you are converting from reduced attribute identifiers to full attribute identifiers (or vice-versa) and make the appropriate conversion.

- `DateFormat = "Format String"`: Sets the default format string for outputting dates (this relation only). Any format suitable for strftime(3) is allowed.

- `Use Cached Secondary Search`: This option significantly speeds up the searching of the instable parts of the database at the expense of some memory and an extra lookaside file "SecondaryCache" (see the Schema.5 man page)

- `Use ExcludeWords`: Instructs *Qddb* to read the file "ExcludeWords" for words that should be excluded from the index files. Searching for words contained in this file will yield "No matches found." since the words are omitted from the index files.

11.3 Entering your attributes

To define your database, you simply key the attribute names in the Schema file.

11.3.1 Valid attribute names

The attribute names must begin with an alphabetic character ("a" through "z", either upper or lower case) and may continue with any alphanumeric characters.

11.3.2 Unique Attribute Names

Each attribute name must be unique within the schema. However, you can share the same name across two different structured groupings. For example, in figure 11.2, we use the attribute name "Description" to describe (1) a phone number description and (2) an address description.

11.4 Schema options

This section describes the various Schema options that are available.

```
#
# Sample schema illustrating that two attributes can have
# the same name --- provided the names are in different
# subgroups.
#
Name
Phone (
  Description verbosename "Phone Description:"
  Number
)
Address (
  Description verbosename "Address Description:"
  Street City State Zipcode
)
```

Figure 11.2: Names can be the same across different structured groups

11.4.1 Free-form comments are allowed (encouraged) in your schema files

Comments may be placed anywhere on a line after a pound sign (“`#`”). You can begin lines with a “`#`” to provide block comments or place the “`#`” at the end of each line to document lines separately. *Qddb* ignores any text following the “`#`” character. See figure 11.5 for a sample schema with comments.

11.4.2 verbosename

The verbosename keyword provides an option for you to specify a more verbose name for your attribute. In particular, the verbosename value is used by *nxqddb* to determine the label that appears to the left of the entry on the *nxqddb* window. For example, consider the schema shown in figure 11.3, and its accompanying *nxqddb* window in figure 11.4. In this schema, we created verbosenames for each attribute. Remember, attribute names cannot have spaces. Hence, we use the verbosename option to create labels that have spaces. We also added colons to the end of the verbosename to help serve as a delimiter to the entry field.

```
name verbosename "Full Name: "
number verbosename "Phone Number: "
ssn verbosename "Social Security Number: "
```

Figure 11.3: Schema with verbosenames

Figure 11.4: *nxqddb* window using a schema with verbosenames

11.4.3 type

The default type for an attribute is string. You can change an attribute's type by keying the phrase "`type <TYPEVALUE>`" next to an attribute. Figure 11.5 provides example schemas that illustrate how to define the various types.

```
#
# Schema to illustrate "types''
#
Name                            # this field defaults to type string
Birthday type date              # date field
Cash type real format "%6.2f"  # currency field, 6 digits with 2 to right of decimal
Number type integer             # define an integer field (accepts positive or negative integer
```

Figure 11.5: Schema illustrating how to define the various types of attributes

So, what does it mean to declare something as string versus, say an integer? If you declare an attribute as an integer, *nxqddb* will only allow you to type numeric characters in the entry field. Meaning, *nxqddb* restricts your input to a valid integer. Similar validation is performed on real, and date attributes.

11.4.4 format

The **format** keyword tells *nxqddb* how to print/display attributes that are defined as (1) **type date**, (2) **type integer** and (3) **type real**. If you don't have one of the above three

integer formats	
`format "%10d"`	10 digits, pad on left
`format "%.4d"`	Zero-fill if less than 4 digits

Table 11.1: Examples of formatting integer types

real formats	
`format "%10.2f"`	10 digits, two to the right of decimal
`format "%.2f"`	Arbitrary digits, two to the right of decimal

Table 11.2: Examples of formatting real types

types of attributes, then *nxqddb* will just ignore the format qualifier. The format keyword has no impact on how you enter the data; remember, the validation is handled based on the type.

integer format

For integers, you may specify any of the standard "non-"C" programmers, this format allows you to specify things like padding, zero-fill, etc. The default format is "To learn more about how to format integers, reference your online man page by entering the following at your command prompt: `man 3 printf` Common integer formats are illustrated in figure 11.1

real format

Similarly to integers, you can specify the format for real numbers. Users commonly specify the precision of the real number using the " The default format for real types is "

date format

For formatting date attributes, you may specify almost any combination of the ones illustrated in figure 11.3. For further details on the date format, see the man pages for *strftime*(3), *asctime*(3), and *mktime*(3). The default format for dates is either "installation of *Qddb* was configured

You may enter dates in almost any format, except that "nn/nn/nn" is interpreted as "%m/%d/%y". You can override this interpretation when you install *Qddb* so that such input will be interpreted "%d/%m/%y". Some combinations of date format, such as "%d;%b;%y", are not currently recognized by Qddb. If you provide a date format that

Date formats	
%A	replaced by the full weekday name.
%a	replaced by the abbreviated weekday name, where the abbreviation is the first three characters.
%B	replaced by the full month name.
%b or %h	replaced by the abbreviated month name (the abbreviation is the first three characters.)
%C	equivalent to "%a %b %e %H:%M:%S %Y"
%c	equivalent to "%m/%d/%y".
%D	replaced by the date in the format "'mm/dd/yy"'.
%d	replaced by the day of the month as a decimal number (01-31).
%e	replaced by the day of month as a decimal number (1-31); single digits are preceded by a blank.
%H	replaced by the hour (24-hour clock) as a decimal number (00-23).
%I	replaced by the hour (12-hour clock) as a decimal number (01-12).
%j	replaced by the day of the year as a decimal number (001-366).
%k	replaced by the hour (24-hour clock) as a decimal number (0-23); single digits are preceded by a blank.
%l	replaced by the hour (12-hour clock) as a decimal number (1-12); single digits are preceded by a blank.
%M	replaced by the minute as a decimal number (00-59).
%m	replaced by the month as a decimal number (01-12).
%p	replaced by either "AM" or "PM" as appropriate.
%R	equivalent to "%H:%M"
%S	replaced by the second as a decimal number (00-60).
%T or %X	equivalent to "%H:%M:%S".
%U	replaced by the week number of the year (Sunday as the first day of the week) as a decimal number (00-53).
%W	replaced by the week number of the year (Monday as the first day of the week) as a decimal number (00-53).
%w	replaced by the weekday (Sunday as the first day of the week) as a decimal number (0-6).
%Y	replaced by the year with century as a decimal number.
%y	replaced by the year without century as a decimal number (00-99).
%Z	replaced by the time zone name.

Table 11.3: Date format options

Date entry keywords		
Keyword	Meaning	Example
tomorrow	24 hours	tomorrow 12:00AM
today	current date/time	today 1:00PM
yesterday	current date/time - 24 hours	yesterday 1:00PM
last	prefix (-1 unit)	last week
next	prefix (+1 unit)	next year
ago	suffix (-X units)	3 years ago
epoch	beginning of time	epoch + 1 year
year(s)	unit of time	last year + 1 week
month(s)	unit of time	next month
week(s)	unit of time	next week
day(s)	unit of time	today + 3 days
hour(s)	unit of time	now + 3 hours
minute(s)	unit of time	now - 20 minutes
second(s)	unit of time	now - 1000 secs
@firstofyear(date)	function	first day of year containing "date"
@lastofyear(date)	function	last day of year containing "date"
@firstofmonth(date)	function	first day of month containing "date"
@lastofmonth(date)	function	last day of month containing "date"

Table 11.4: Keywords available for date attributes

Qddb cannot interpret, you will see an error when entering dates. Most common formats are supported.

You may also enter dates relative to some date(see table 11.4).

You may also use the day-of-week keywords `Monday`, `Tuesday`, `Wednesday`, `Thursday`, `Friday`, `Saturday`, and `Sunday`. Month names are also fine. For example, the following are all valid input to a date attribute:

```
next tuesday
next december
next 9 dec
december 1, 1995 +1 week
9 dec 1995
wednesday
@firstofmonth(this month)
@lastofmonth(last month -1 year)
```

The input "`next tuesday`" does not mean this coming (or current) Tuesday, but the one after. The keyword "`tuesday`" means the coming (or current) Tuesday.

11.4.5 alias

Aliases provide a way to provide a simple (often much shorter) name to an attribute. The use of aliases are especially helpful if you are programming with Qtcl and you have many levels of nested structured attributes. Also, using aliases, you can save some typing whenever you create your custom Email, letters and postcards through the *nxqddb* report generator. The alias syntax is simple, for example to alias a "SocialSecurityNumber" field, you might enter `SocialSecurityNumber alias ssn`.

11.4.6 defaultvalues

You can tell *nxqddb* to prepopulate an entry with a default value when you are in "Add Mode". For example, consider a "City" attribute. If most of your entries live in say "Cincinnati", you could specify a default value of "Cincinnati" and this value would appear in the *nxqddb* window whenever you go to "Add Mode".

Many users find it convenient to prepopulate ZipCode attributes with a partial default value. For example, suppose you happen to have say 95of the five zip codes 41016, 41042, 41017, 41011 and 41015. In this case, you would create the ZipCode schema like so: `ZipCode defaultvalues ''410''`

11.4.7 exclude

The exclude option tells *Qddb* to **exclude** indexing on the attribute. If you specify the exclude option on a field, you will not be able to perform searching on the field. Typically, you want to use the exclude option only on fields that you know will never be used as search criteria fields. For example, fields that you may want to exclude from searching might be: sales tax, subtotals, or any other field that would yield meaningless search results.

11.4.8 separators

Separators tell *nxqddb* how your data values are to be parsed into words during a word search. If you don't specify the key word "separators", then *Qddb* abides by the following rules when parsing words:

1. *Qddb* uses all non-alphanumeric characters as delimiters.
2. If the attribute has "**type real**", **+-.** are not considered delimiters.
3. If the attribute has "**type integer**" or "**type date**", +- are not considered delimiters
4. A newline is always a delimiter

The default separators are shown below:

string The string default separators are:

```
" 0r!@#$%^&*()_+-={}[];':
```

real The default separators for real are the same as string except for the minus(-) and plus(+) characters.

integer Same as string except for the minus(-) and plus (+).

date Same default separators as integer type attributes.

Example of specifying a separator

Suppose you have data that contains the delimiters: ":" and "/". For example, say you have a string like "foo:bar/whatever this is". To handle this scenario, you would set the separators to ":/" as shown in figure 11.6

```
#
# Schema containing separators
#
MyField separators ":/"
```

Figure 11.6: Schema to illustrate the use of separators

Keywords
verbosename
type
separators
format
alias
exclude
defaultvalues

Table 11.5: Keywords that must not be used for attribute names

11.5 Example Schemas

This section provides some sample schemas to give you an idea for the style and types of schemas that you can create.

11.5.1 Schemas are free-form

You can enter your database description in the Schema file using any style you choose. *Qddb* only requires the following:

1. You can choose any name you want for your fields, except for those shown in table 11.5
2. You must separate field names and keywords by one or more whitespace characters.

> Whitespace characters can be either a (1) tab, (2) blank or (3) newline character.

All of the schemas shown in figure 11.7 are equivalent.

```
a ( b c ) * d          a (             a
                           b           (
                           c             b c
                       ) *             ) * d
                       d
```

Figure 11.7: Equivalent schemas

11.5.2 Deeply Nested Schema

There is no limit to the number of nested structured attributes. For example, the schema shown in figure 11.8

```
a ( b ( c ( d ( e ( f ( g ( h ( i ( j ( k ) ) ) ) ) ) ) ) ) ) )
```

Figure 11.8: A deeply nested schema example

will work just fine — although, if you have this much nesting in your database, you may want to rethink your design.

11.5.3 Sample Schema illustrating all the attribute options

Figure 11.9 shows a schema using all the options. It represents a family that might have many members, addresses and phone numbers. The attributes `n`, `a`, and `p` are expandable so you can have multiple names, addresses and phone numbers grouped together in one record.

11.6 Adding new fields after data has been entered

Qddb allows you to add new attributes to existing schemas — provided you add the attribute name to the tail of the schema or structured attribute.

> If you add new fields to a schema, you must add the fields to the end of the schema, or structured attribute group.

For example, suppose you have the schema shown in figure 11.10

You could add the attributes "Name.Middle", but only if you put "Name.Middle" after "Name.Last" as shown in figure 11.11 Similarly, "Address.Zip" must be added at the end of the Address structured attribute as shown in figure 11.11

You can even add structured/expandable attributes to the end of schema files as shown in figure 11.12

```
# Database of families I know
HashSize = 2000
CacheSize = 100
Use Cached Hashing
Use reduced attribute identifiers
DateFormat = "%d %B %Y" # "27 February 1956"
n verbosename "Name" (
   t verbosename "Dr./Mr./Mrs./Ms./Miss"
    f verbosename "First"
    m verbosename "Middle" exclude
    l verbosename "Last"
)*
a verbosename "Address" (
    street city state # three attributes on one line
    zip verbosename "zip code" type integer
)*
e verbosename "E-mail" separators "@!%"
p verbosename "Phones" (
    d verbosename "Desc"
    a type integer verbosename "" alias pa format "%10d"
    p verbosename "" alias pp type integer
    s verbosename "" type integer
)*
z verbosename "Extra stuff" (
    a ( a b c ) b c d e f g h
)
c verbosename "Comments"
```

Figure 11.9: An example of a schema showing all options

```
Name (
  First
  Last
)
Address (
  Street
  City
  State
)
```

Figure 11.10: Sample schema before adding new attributes

```
Name (
  First
  Last
  Middle
)
Address (
  Street
  City
  State
  Zip
)
```

Figure 11.11: Sample schema after adding new attributes to existing database

Notice how we added the structured/expandable attribute "Phone" to the tail of the Schema file.

Of course, if you have not stored any records yet, you can change the Schema however you like. If you have entered data and really cannot enter the fields at the end of the schema or structured attribute, you can always create new databases and then write Tcl scripts to copy the information from one database to another.

The only drawback to being forced to add attributes to the end of the schema (or structured attribute grouping) is that the attribute will be the last attribute in the *nxqddb* window. However, you can always write a short Fx-based Tcl program to layout your own screen. For details on how to program Fx/Qddb/Tcl applications, reference the *Qddb* Programming Guide.

11.7 Stabilizing your databases

Periodically, for performance reasons, you should stabilize your database. To stabilize your database, follow these steps:

```
Name (
  First
  Last
  Middle
)
Address (
  Street
  City
  State
  Zip
)
Phone (
  Description
  Number
) *
```

Figure 11.12: Adding attributes to end of schema file for existing database

1. Be sure that no one is accessing your database and ensure no one will be trying to access during the stabilization process. For this reason, users find it convenient to stabilize their database during off-peak hours such as evenings or weekends.

2. From the command prompt, type "`$ qstall YourDatabaseName`". Do not interrupt the stabilization process and be sure the stabilization is complete before accessing your database.

Appendix A

Schemas for common applications

We provide some schemas in this appendix for you to use. Of course, you can modify these schemas to fit your needs or create new ones.

"Rolodex" Schema

```
Name (
        Title alias title
        First alias firstname
        Middle alias middlename
        Last alias lastname
)
Address (
        Street alias street
        City alias city
        State alias state
        Zip alias zip
        Country alias country
) *
Phone (
        Description alias phonedescription
        Area verbosename "Area Code" alias areacode
        Number alias phonenumber
) *
Email *
Homepage verbosename "WWW Home Page" alias homepage
Sex
Category
Birthday type date
Employer
Occupation
Hobbies
Comments
```

Figure A.1: Schema: Rolodex

“Library” Schema

```
# Qddb relation for personal library
Book (
    ISBN Title Authors * Subject
    Keywords verbosename "Key Words"
    Publisher *
    Series verbosename "Book Series"
    Year Edition Printing Abstract
)
Borrowers (
    Name
    Address (
        Street City State
        Zip verbosename "Zip Code"
    )*
    Phones (
        Desc verbosename "Description"
        Number (
            Area type integer
            Prefix type integer
            Suffix type integer
        )
    )*
    Borrowed verbosename "Date Borrowed" type date
    Returned verbosename "Date Returned" type date
)*
```

Figure A.2: Schema: Library

“RunningLog” Schema

```
DateOfRun verbosename "Date: " type date
Course verbosename "Course: "
Time verbosename "Time: " type date format "%H:%M:%S"
Distance verbosename "Distance (miles): " type real format "%3.1f"
WeatherConditions verbosename "Weather conditions: "
Comments *
```

Figure A.3: Schema: RunningLog

"Bug Tracking System" Schema

```
status verbosename "Status: "   # open, closed, working, deferred
TrackingCode verbosename "Issue number: "
RelatedIssues verbosename "Related issues: "
Abstract verbosename "Abstract: "
Description verbosename "Bug description: "

Originator (
  Submittedby verbosename "Name: "
  Company verbosename "Company: "
  DateSubmitted verbosename "Submitted on: " type date
  TimeSubmitted verbosename "Time submitted: " type date format "%H:%M:%S"
  Email verbosename "Email: "
)

Hardware (
  Processor verbosename "Processor: "
  GraphicsCard verbosename "Graphics card: "
  Monitor verbosename "Monitor: "
)

Software (
  OperatingSystem verbosename "Operating system: "
  Release verbosename "Software release: "
  Configuration verbosename "Configuration: "
)

Comments verbosename "Comments: " *
ProgrammerNotes verbosename "Programmer notes: " *
```

Figure A.4: Schema: BugTracker

Appendix B

Faunix: Schemas from a commercial application

This section lists the main schemas used in Faunix — a commercial application for managing a veterinary practice. The schemas are presented here to give you a feel for how to create schemas for your application.

Faunix: "Setup" Schema

```
Use Cached Hashing
HashSize = 500

Doctors ( Name ) *
Personnel ( Name )*
EntryCodes verbosename "Entry Codes" (
        Code
        Description
) *
Rabies verbosename "Next Rabies Tag #"
InvoiceNumber verbosename "Next Invoice #" type integer
ClientID verbosename "Next Client #" type integer
PetID verbosename "Next Pet #" type integer
ServiceChargeRate verbosename "Service Charge Rate %"
        type real format "%.2f"
DefCity verbosename "Default City"
DefState verbosename "Default State"
DefZip verbosename "Default Zip Code"
Clinic verbosename "Clinic Information" (
        Name
        Address (Street City State Zip)
        Phone verbosename "Primary phone:"
)
TaxRate verbosename "Tax rate %" type integer
PaymentTypes verbosename "Types of Payment Accepted"
EstimateNumber verbosename "Next Estimate #" type integer
MinimumServiceCharge verbosename "Minimum Service Charge $"
        type real format "%.2f"
RxNumber verbosename "Next Rx #" type integer
AgingType verbosename "Account Aging" defaultvalue "Automatic"
AgingDayOfMonth verbosename "Day of month for aging (1-28)"
        type integer defaultvalue "1"
AutomaticBackups verbosename "Automatic Backups" defaultvalue "On"
```

Figure B.1: Faunix: Setup schema

Faunix: "Clients" Schema

```
Use reduced attribute identifiers
Use cached secondary search
Use Cached Hashing
HashSize = 20000

ID verbosename "Client #" type integer
Client (
    Name (Title First Middle Last)*
    Address (
        Street City State Zip verbosename "Zip Code"
        Billing verbosename "Billing" defaultvalue "Y"
        Reminder verbosename "Reminders" defaultvalue "Y"
    )*
    Phones (Desc verbosename "Desc" Number)*
    Employer Occupation
    SS verbosename "S.S. #"
    BillingPercent verbosename "Invoice %"
)*
net  verbosename "Net Bal." type real format "%.2f"
MonthlyPayments verbosename "This month's payments" type real format "%.2f"
MonthlyCharges verbosename "This month's charges" type real format "%.2f"
curr verbosename "Current" type real format "%.2f"
d30 verbosename "30 days" type real format "%.2f"
d60 verbosename "60 days" type real format "%.2f"
d90 verbosename "90 days" type real format "%.2f"
CreditHold verbosename "Credit hold" defaultvalue "N"
LastPayment verbosename "Last Payment Date" type date
FirstDate verbosename "1st Date" type date
LastDate  verbosename "Last Date" type date
ReferenceDisc verbosename "Ref. Disc. %" type real format "%.2f"
MultiPet verbosename "Multi-Pet Disc. %" type real format "%.2f"
Comments
Referral verbosename "Referred by" (ClientName ClientNumber)
```

Figure B.2: Faunix: Clients schema - "Part A"

```
Pets (
    ID verbosename "Animal #" type integer
    Name Species Breed Markings Color Sex
    Died defaultvalue "N"   Breeding verbosename "Breeding An."
    Birthday type date
    FirstVisit verbosename "First Visit" type date
    LastVisit verbosename "Last Visit" type date
    NextVisit verbosename "Next Visit" type date
    WeightDate verbosename "Date Taken" type date
    WeightType verbosename "Weight Type" defaultvalue "lbs"
    Weight verbosename "Weight"
    Rabies (
        Date verbosename "Date" type date
        Tag verbosename "Tag #"
        Manufacturer verbosename "Mfg."
    )*
    Comments
    MedHist verbosename "Medical History" (
        Date type date
        EntryCode verbosename "Entry Code"
        Doctor
        Items verbosename "Services/Items" (
            Qty type integer defaultvalue ""
            SvcType
            Code
            Desc exclude
            Price type real format "%.2f" defaultvalue "" exclude
            Total type real format "%.2f" defaultvalue "" exclude
            Unit exclude
            Comments
        )*
        Comments
    )*
    Reminders (
        DueWhen verbosename "Date Due" type date
        Code
        Desc exclude
        RemindersSent verbosename "# reminders sent" type integer
        DoneWhen verbosename "Last Done" type date
    )*
    Recalls (
        EnteredDate verbosename "On (date)" type date
        Date verbosename "recall/recheck (date)" type date
        Code
        Desc verbosename "Description"
        Doctor
        Comments
    )*
)*
```

Figure B.3: Faunix: Clients schema - “Part B”

```
Statements (
        Date verbosename "Statement date" type date
        net verbosename "Net Bal." type real format "%.2f" defaultvalue ""
        curr verbosename "Curr. Bal." type real format "%.2f" defaultvalue ""
        d30 verbosename "30 Days" type real format "%.2f" defaultvalue ""
        d60 verbosename "60 Days" type real format "%.2f" defaultvalue ""
        d90 verbosename "90 Days" type real format "%.2f" defaultvalue ""
        ServiceCharge verbosename "Service charge"
                type real format "%.2f" defaultvalue ""
)*
```

Figure B.4: Faunix: Clients schema - “Part C”

Faunix: "Invoices" Schema

```
Use Reduced Attribute Identifiers
Use Cached Hashing
CacheSize = 500
HashSize = 20000

Number verbosename "Invoice Number" type integer
Client (
        ID verbosename "Client #" type integer
        Name
)
Date type date
Time exclude
DayOfWeek verbosename "Day of Week"
Pet (
        PetID verbosename "Animal #" type integer
        PetName verbosename "Animal Name"
        Doctor
        Personnel
        EntryCode verbosename "Entry Code"
        Items (
                Qty type integer
                Code
                SvcType
                Desc verbosename "Description"
                Taxable exclude
                Price type real format "%.2f" exclude
                Total verbosename "Line Total" type real format "%.2f" exclude
                DiscountedTotal verbosename "Discounted Total" type
                        real format "%.2f" exclude
        )*
        Subtotal type real format "%.2f" exclude
)*
Subtotal type real format "%.2f" exclude
Discounts type real verbosename "Discounts" format "%.2f" exclude
Tax type real verbosename "Sales Tax" format "%.2f"
Total verbosename "Invoice Total" type real format "%.2f" exclude
PaymentType verbosename "Type of Payment"
PaymentAmount verbosename "Payment Amount" type real format "%.2f" exclude
PreviousBalance verbosename "Previous Balance" type real format "%.2f" exclude
AmountDue verbosename "Total amount due" type real format "%.2f"
```

Figure B.5: Faunix: Invoices schema

Faunix: "Inventory" Schema

```
Use Cached Hashing
HashSize = 500

Category
Barcode separators ""
Desc verbosename "Description"
Msg verbosename "Print Message"
Size
Tax
Price type real format "%.2f"
Group
QtyInStock type integer verbosename "Quantity in stock"
Date verbosename "Date last changed"
Dynamic verbosename "Variable pricing"
Message verbosename "Message To Print"
Actions
Code separators ""
```

Figure B.6: Faunix: Inventory schema

Faunix: "Estimates" Schema

```
Use Reduced Attribute Identifiers
Use Cached Hashing
CacheSize = 500
HashSize = 20000

Number verbosename "Estimate Number" type integer
Client (
ID verbosename "Client #" type integer
Name
)
Date type date
Time exclude
DayOfWeek verbosename "Day of Week"
Pet (
PetID verbosename "Animal #" type integer
PetName verbosename "Animal Name"
Doctor
Items (
Qty type integer
Code
SvcType
Desc verbosename "Description" exclude
Taxable exclude
Price type real format "%.2f" exclude
Total verbosename "Line Total" type real format "%.2f" exclude
DiscountedTotal verbosename "Discounted Total" type
real format "%.2f" exclude
)*
Subtotal type real format "%.2f" exclude
)*
Subtotal type real format "%.2f" exclude
Discounts type real verbosename "Discounts" format "%.2f" exclude
Tax type real verbosename "Sales Tax" format "%.2f" exclude
Total type real format "%.2f" exclude
PaymentType verbosename "Type of Payment" exclude
PaymentAmount verbosename "Payment Amount" type real format "%.2f" exclude
```

Figure B.7: Faunix: Estimates schema

Faunix: "Services" Schema

```
Use Cached Hashing
HashSize = 500

Category
Code separators ""
Barcode separators ""
Desc verbosename "Description"
Unit
Reminders verbosename "Reminders in" type integer defaultvalue "0"
Area
SvcType verbosename "Type"
AI
Msg verbosename "Print a message on the invoice"
Tax defaultvalue "N"
Price type real format "%.2f"
Group verbosename "Additional codes for automatic entry"
Message verbosename "Message To Print"
Actions verbosename "Programming Actions"
Invoice verbosename "Print on invoice" defaultvalue "Y"
ReminderUnits verbosename " " defaultvalue "Months"
Dynamic verbosename "Variable pricing" defaultvalue "N"
Recall verbosename "Recall/recheck in" type integer defaultvalue "0"
RecallUnits verbosename " " defaultvalue "Days"
# programming actions
RabiesCode verbosename "Should this service generate a rabies tag?" defaultvalue "N"
SexChangeCode verbosename "Does this service change the animal's sex?" defaultvalue "N"
SexChangeValue verbosename "If so, to what?"
DeadCode verbosename "Does this service indicate the animal has died?" defaultvalue "N"
EquivalentCodes verbosename "Equivalent codes for reminders"
RecallComments verbosename "Prompt for comments?" defaultvalue "N"
MedHistComments verbosename "Prompt for medical history comments?" defaultvalue "N"
```

Figure B.8: Faunix: Services schema

Faunix: "Bonus" Schema

```
Use Cached Hashing
HashSize = 500

Title verbosename "Report title"
From type date verbosename "Report from (date)"
To type date verbosename "Report to (date)"
RabiesCodes verbosename "Rabies Codes"
ComparisonCodes verbosename "Codes for comparison to rabies" (
Desc verbosename "Description"
Codes
Target type real format "%.2f" verbosename "Target (%)"
)*
GroomingCodes verbosename "Grooming Codes"
Expected type real format "%.2f" verbosename "Expected % increase from last year"
Excess type real format "%.2f" verbosename "Excess (per visit) bonus $"
Days type integer verbosename "Working days"
Bonus type real format "%.2f" verbosename "Bonus (%)"
```

Figure B.9: Faunix: Bonus schema

Faunix: "Summaries" Schema

```
Use Cached Hashing
HashSize = 3000

Date type date
DayClosed verbosename "Day Closed"
Visits verbosename "# Visits" type integer
InvStart verbosename "Invoice Start" type integer
InvEnd verbosename "Invoice End" type integer
Payments verbosename "Daily Payments" (
PaymentType verbosename "Type of Payment"
Amount type real format "%.2f"
)*
Total verbosename "Daily Payment Total" type real format "%.2f"
Tax verbosename "Daily Sales Tax" type real format "%.2f"
OverShort verbosename "Over + / Short -" type real format "%.2f"
Reason verbosename "Reason for Over/Short"
Deposit verbosename "Deposit" type real format "%.2f"
ByDoctor verbosename "Services by doctor" (
Doctor
Services (
Code
Desc verbosename "Description"
Total type real format "%.2f"
Qty type integer
)*
)*
Fees verbosename "Daily Invoice Total" type real format "%.2f"
```

Figure B.10: Faunix: Summaries schema

Faunix: "VetStats" Schema

```
Use Cached Hashing
HashSize = 500

Title verbosename "Report title"
From type date verbosename "Report from (date)"
To type date verbosename "Report to (date)"
StatementDate type date verbosename "Statement Date (8/1/95)"
QtyCodes verbosename "Areas and codes for quantity breakdown" (
Desc verbosename "Description"
Areas
Codes
)*
RevenueBreakdown verbosename "Areas and codes for revenue breakdown" (
Desc verbosename "Description"
Areas
Codes
)*
```

Figure B.11: Faunix: VetStats schema

Index