

# CHAPTER 15

## DESIGNING CUSTOM MULTITABLE FORMS

### In this chapter

- Expanding Your Form Design Repertoire 570
- Understanding the Access Toolbox 570
- Access's Control Wizards, Builders, and Toolbars 574
- Using the Toolbox to Add Label and Text Controls 577
- Adding Option Groups with the Wizard 590
- Using the Clipboard to Copy Controls to Another Form 595
- Adding Combo and List Boxes 596
- Working with Tab Controls 616
- Optimizing the Form's Design 621
- Adding a History Subform to a Tab Control Page 623
- Adding New Records in the HRActionEntry Form 628
- Overriding the Field Properties of Tables 630
- Adding Page Headers and Footers for Printing Forms 631
- Troubleshooting 632
- In the Real World—Access Wizardry 632

## EXPANDING YOUR FORM DESIGN REPERTOIRE

The controls that the Form Wizard adds to the forms it creates are only a sampling of the 17 native control objects offered by Access 2003. *Native controls* are built into Access; you also can add various ActiveX controls to Access forms. Until now, you used the Form Wizard to create the labels, text boxes, and subform controls for displaying and editing data in the HRActions table. These three controls are sufficient to create a simple transaction-processing form.

The remaining 14 controls described in this chapter let you take full advantage of the Windows graphical user environment. You add controls to the form by using the Access Toolbox. List and combo boxes increase data-entry productivity and accuracy by letting you select from a list of predefined values instead of requiring you to type the value. Option buttons, toggle buttons, and check boxes supply values to Yes/No fields. You can place option buttons, toggle buttons, and check boxes in an option group. Inside an option group, the control you click sets the numeric Value property of the option group control. The Image control supplements the Bound and Unbound Object Frame controls for adding pictures to your forms. Page breaks determine how forms print. Access 2003's tab control lets you create tabbed forms to display related data on forms and subforms in a space-saving and more clearly organized fashion. Command buttons usually execute VBA event-handling procedures.

### NOTE

The form-design techniques you learn in this chapter also apply to Access Data Projects (ADP). ADP forms are identical to conventional Access forms, except that the forms and controls bind to objects in SQL Server 2000—not Jet 4.0—databases.

## UNDERSTANDING THE ACCESS TOOLBOX

-  The Access Toolbox is based on the Toolbox that Microsoft first created for Visual Basic. Essentially, the Access Toolbox is a variety of toolbar. You select one of the 20 buttons that appear in the Toolbox to add a native control—represented by that tool's symbol—to the form. Selecting a tool lets you select a control, enable or disable the Control Wizards, or add a Microsoft or third-party ActiveX control to the form. When you create a report, the Toolbox serves the same purpose—although tools that require user input, such as combo boxes, seldom are used in reports.

### CONTROL CATEGORIES

Three control object categories apply to Access forms and reports:

- *Bound controls* are associated with a field in the data source for the form or subform. Binding a control means connecting the control to a data source, such as a field of a table or a column of a query, which supplies the current value to or accepts an updated value from a control. Bound controls display and update values of the data cell in the associated field of the currently selected record. Text boxes are the most common bound control. You can display the content of graphic objects or play audio files embedded in a table with a bound OLE object. You can bind toggle buttons and check boxes to Yes/No fields. Option button groups bind to fields with numeric values. All bound controls have associated labels that display the Caption property of the field; you can edit or delete these labels without affecting the bound control.
- *Unbound controls* display data you provide that is independent of the form's or subform's data source. You use the image or unbound OLE object control to add a drawing or bitmapped image to a form. You can use lines and rectangles to divide a form into logical groups or simulate boxes used on the paper form. Unbound text boxes are used to enter data that isn't intended to update a field in the data source but is intended for other purposes, such as establishing a value used in an expression. Some unbound controls, such as unbound text boxes, include labels; others, such as unbound OLE objects, don't have labels. Labels also are unbound controls.
- *Calculated controls* use expressions as their source of data. Usually, the data source expression includes the value of a field, but you also can use values created by unbound text boxes in calculated control expressions.

## THE ACCESS TOOLBOX

 The Toolbox appears only in Design view for forms and reports, and it appears only if you click the Toolbox button on the toolbar or toggle the **V**iew, **T**oolbox menu choice. When the Toolbox is visible, the **T**oolbox menu choice is checked; Figure 15.1 shows the Toolbox in its default mode—a two-column floating toolbar. You can select one of the 17 controls and three other buttons, whose names and functions are listed in Table 15.1.

**Figure 15.1**  
The Access Toolbox lets you add 17 different native controls to forms and reports.

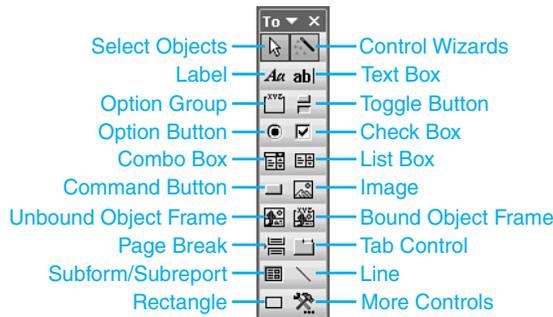


TABLE 15.1 NAMES AND FUNCTIONS OF ACCESS TOOLBOX BUTTONS

Tool	Name	Function
	Select Objects	Changes mouse pointer to the Object Selection tool. Deselects a previously selected tool and returns the mouse pointer to normal selection function. Select Objects is the default tool when you open the Toolbox.
	Control Wizards	Turns the Control Wizards on and off. Control Wizards aid you in designing complex controls, such as option groups, list boxes, and combo boxes.
	Label	Creates a box that contains fixed descriptive or instructional text.
	Text Box	Creates a box to display and allow editing of a data value and also creates a corresponding label, which you can choose to delete.
	Option Group	Creates a frame of adjustable size in which you can place toggle buttons, option buttons, or check boxes. Only one of the objects within an object group frame can be selected. When you select an object within an option group, the previously selected object is deselected.
	Toggle Button	Creates a button that changes from On to Off when clicked. The On state corresponds to Yes (-1), and the Off state corresponds to No (0). When used within an option group, toggling one button On toggles a previously selected button Off. You can use toggle buttons to let the user select one value from a set of values.
	Option Button	Creates a round button (originally called a <i>radio button</i> ) that changes from the Off to On state when you select it. Option buttons are most commonly used within option groups to select between values in a set in which the choices are mutually exclusive.
	Check Box	Creates a check box that toggles On and Off. Multiple check boxes should be used outside option groups so that you can select more than one check box at a time.
	Combo Box	Creates a combo box with an editable text box where you can enter a value, as well as a list from which you can select a value from a set of choices.
	List Box	Creates a drop-down list box you can select a value from. A list box is the list portion of a combo box.

Tool	Name	Function
	Command Button	Creates a command button that, when clicked, triggers an event that can execute an Access VBA event-handling procedure.
	Image	Displays a static graphic on a form or report. This is not an OLE image, so you can't edit it after placing it on the form.
	Unbound Object Frame	Adds an OLE object created by an OLE server application, such as Microsoft Chart or Paint, to a form or report.
	Bound Object Frame	Displays the content of an OLE field of a record if the field contains a graphic object. If the field contains no graphic object, the icon that represents the object appears, such as the Sound Recorder's icon for a linked or embedded .wav file.
	Page Break	Causes the printer to start a new page at the location of the page break on the form or report. Page breaks don't appear in form or report Run mode.
	Tab Control	Inserts a tab control to create tabbed forms. (The tab control looks like the tabbed pages you've seen in the Properties windows and dialogs throughout this book.) Pages of a tab control can contain other bound or unbound controls, including subform/subreport controls.
	Subform/Subreport	Adds a subform or subreport to a main form or report, respectively. The subform or subreport object you intend to add must exist before you use this control.
	Line	Creates a straight line that you can size and relocate. You can change the color and width of the line by using the Formatting toolbar buttons or the Properties window.
	Rectangle	Creates a rectangle that you can size and relocate. The border color, width, and fill color of the rectangle are determined by selections from the palette.
	More Controls	Clicking this tool opens a scrolling list of ActiveX controls that you can use in your forms and reports. The ActiveX controls available through the More Controls list aren't part of Access; ActiveX controls are supplied as .ocx or .dll files with Office 2003, Visual Basic, and various third-party tool libraries.

Using controls in the design of reports is discussed in Chapter 16, “Working with Simple Reports and Mailing Labels” and Chapter 17, “Preparing Advanced Reports.” Using command buttons to execute VBA code is covered in Part VII of this book, “Programming and Converting Access Applications.”

## ACCESS’S CONTROL WIZARDS, BUILDERS, AND TOOLBARS

Access provides a number of features to aid you in designing and using more complex forms. Three of these features—Control Wizards, Builders, and customizable toolbars—are described in the three sections that follow.

### ACCESS CONTROL WIZARDS

Much of the success of Access is attributable to the Form Wizard, Report Wizard, and Chart Wizard that simplify the process of creating database objects. The first wizard appeared in Microsoft Publisher, and most of Microsoft’s productivity applications now include a variety of wizards. Chapter 14, “Creating and Using Access Forms,” introduced the Form Wizard; the Report Wizard is discussed in Chapter 16 and the Chart Wizard is described in Chapter 18, “Adding Graphs, PivotCharts, and PivotTables.” In this chapter, you’re introduced to a Control Wizard each time you add a control for which a wizard is available.

### ACCESS BUILDERS

Builders are another feature that makes Access easy to use. You use the Expression Builder, introduced in Chapter 5, “Working with Jet Databases and Tables,” to create expressions that supply values to calculated controls on a form or report. The Query Builder creates the Jet SQL or Transact-SQL statements you need when you create list boxes or combo boxes whose Row Source property is an SQL statement that executes a select query. The Query Builder is described in the “Using the Query Builder to Populate a Combo Box” section near the end of this chapter.

### CUSTOMIZABLE TOOLBARS

The preceding chapters demonstrated that Access toolbars include many shortcut buttons to expedite designing and using Access database objects. Access 2003, like most other contemporary Microsoft applications, lets you customize the toolbars to your own set of preferences. Access stores customized toolbar preferences in System.mdw. Toolbars that you create yourself, called *command bars*—are stored in a hidden system table—MSysCmdbars—in each database.

By default, the Toolbox is a floating toolbar. To anchor the Toolbox, also called *docking* the toolbar, follow these steps:

1. Press and hold down the mouse button while the mouse pointer is on the Toolbox's title bar, and drag the Toolbox toward the Form Design toolbar. When the Toolbox reaches the toolbar area, the dotted outline changes from a rectangle approximately the size of the Toolbox into a wider rectangle only as high as a toolbar.
2. Release the mouse button to change the Toolbox to an anchored toolbar positioned below the standard Form Design toolbar.

**TIP**

You can anchor or dock a toolbar to any edge of Access's main window. Press and hold down the mouse button on an empty area of the toolbar (not covered by a button), and drag the toolbar until its outline appears along the left, right, or bottom edge of the window. If you drop the toolbar within the confines of Access's main window, it becomes a floating toolbar. In addition, double-clicking the title bar automatically docks a toolbar to its last docked position.

## ADD OR DELETE TOOLBARS

You can add or delete buttons from toolbars with the Customize Toolbars dialog. To add form design utility buttons to the Toolbox toolbar (whether it's docked or floating), do the following:

1. Choose View, Toolbars, Customize to display the Customize dialog. Alternatively, click the down arrow of the Toolbox's title bar, choose Add or Remove Buttons from the context menu, and select Customize from the button list.

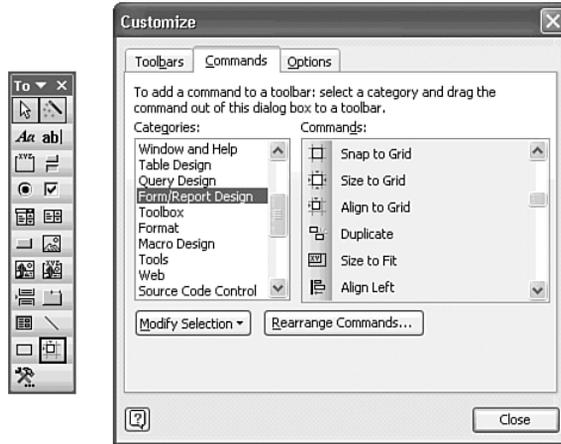
**TIP**

You can also open the Customize dialog by right-clicking any part of a toolbar and choosing Customize from the context menu.



2. Click the Commands tab, and select Form/Report Design from the Categories list. The optional buttons applicable to form design operations appear in the Commands list, as shown in Figure 15.2.
3. The most useful optional buttons for form design are control alignment and sizing buttons. Click and hold down the mouse button on the Align to Grid command, drag this button to the Toolbox toolbar, and drop it to the right of the Rectangle button. The right margin of the Toolbox toolbar expands to accommodate the new button (if you customize the Toolbox while it's floating, the window expands to accommodate the new button). You can drag the Align to Grid button slightly to the right to create a gap between the new button and the Rectangle button.

**Figure 15.2**  
Add Align to Grid and  
other Form/Report  
Design toolbar buttons  
to the Toolbox  
with the Customize  
dialog.



4. Repeat step 3 for the Size to Fit, Size to Grid, and Align Left commands, dropping each button to the right of the preceding button. You now have four new and useful design buttons available in your Toolbox. The Customize dialog for toolbars provides the following additional capabilities:
  - To remove buttons from the toolbar, open the Customize dialog; click and drag the buttons you don't want, and drop them anywhere off the toolbar.
  - To reset the toolbar to its default design, open the Customize dialog, and click the Toolbars tab. In the Toolbars list, select the toolbar you want to reset, and click the Reset button. A message box asks you to confirm that you want to abandon any changes you made to the toolbar.
  - To create a button that opens or runs a database object, open the Customize dialog, display the Commands page, and scroll the Categories list to display the All *Objects* items. When, for example, you select All Tables, the tables of the current database appear in the Commands list. Select a table name, such as Employees, and drag the selected item to an empty spot on a toolbar. The ScreenTip for the new button displays Open Table 'Employees'. (If you select All Macros and drag a macro object to the toolbar, the button you add runs the macro when clicked.)
  - To substitute a different image for the picture on the buttons, open the Customize dialog. Right-click the button you want to change (on the toolbar, not in the Customize dialog) to display the button shortcut menu. Click Change Button Image to display the Change Button Image dialog. Click one of the images offered. To edit the button's image, click Edit Button Image.
  - To create a new empty toolbar that you can customize with any set of the supplied buttons you want, open the Customize dialog and select Utility 1 or Utility 2 on the Toolbars page. If there's space to the right of an existing toolbar, the empty toolbar

appears in this space. Otherwise, Access creates a new toolbar row for the empty toolbar. The Utility 1 and Utility 2 toolbars and the changes you make to them are available in any Access database you open.

- To create a custom toolbar that becomes part of your currently open database, open the Customize dialog and click New on the Toolbars page. The New Toolbar dialog appears, requesting a name for the new toolbar (Custom 1 is the default). Access creates a new floating tool window to which you add buttons from the Commands page of the Customize dialog. You can anchor the custom tool window to the toolbar, if you want.
- To delete a custom toolbar, open the Customize dialog, select the custom toolbar on the Toolbars page, and click the Delete button. You are requested to confirm the deletion. The Delete button is disabled when you select one of Access's standard toolbars in the list.

As mentioned earlier, custom toolbars to which you assign names become part of your database application and are stored in the current database file; they are available only when the database in which you store them is open. Built-in Access toolbars that you customize are stored in System.mdw and are available in any Access session.

## USING THE TOOLBOX TO ADD LABEL AND TEXT CONTROLS

Using the Form Wizard or the AutoForm feature of Access 2003 simplifies the generation of standard forms for displaying and updating data in tables. Creating forms from scratch in Form Design view by adding controls from the Toolbox provides much greater design flexibility than automated form generation. The examples in this chapter use the HRActions table that you created in Chapter 5, and a query, qryHRActions, which you create in the next section.

→ For more information on creating the data source for this chapter and establishing the correct relationships, see “Creating the HRActions Table,” p. 182.



### TIP

If you haven't created the HRActions table, you can import it from the Forms15.mdb database in your \Program Files\Seua11\Chaptr15 folder or from the \Seua10\Chaptr15 folder of the accompanying CD-ROM.

## CREATING THE QUERY DATA SOURCE FOR THE MAIN FORM

The HRActions table identifies employees uniquely by their sequential ID numbers, located in the EmployeeID field. As before, you need to display the employee's name and title on the form to avoid entering records for the wrong person. The form design example in this chapter uses a one-to-many query to provide a single source of data for the new, custom HRActions form.

To create the HRActions query that serves as the data source for your main form, follow these steps:

1. Close any open Northwind forms, click the Tables shortcut in the Database window, and select HRActions in the table list.
2. Click the New Object: Query toolbar button to open the New Query dialog, and click OK with Design View selected to open Query1 with the HRActions table added. (Don't worry if your query's name contains a different number.)
3. Click the Show Table toolbar button to open the Show Table dialog, and add the Employees table to your query. Click the Close button to close the Show Table dialog.
4. If you defined relationships for the HRActions table as described in Chapter 5, the upper pane of the query window appears as shown in Figure 15.3. The line connecting the two tables indicates that a many-to-one relationship exists between the EmployeeID field in the HRActions table and the EmployeeID field of the Employees table.

**Figure 15.3**

The upper pane of the Query Design window displays the many-to-one relationship between the Employees and HRActions tables.



**TIP**

If you didn't define any relationships, the join line doesn't appear. In this case, you need to drag the EmployeeID field from the HRActions field list to the EmployeeID field of the Employees field list to create a join between these two fields.

5. Click the \* field of the HRActions table, and then drag and drop it in the first column of the Query Design grid. This adds all the fields of the HRActions table to your query.
6. From the Employees table, click and drag the LastName, FirstName, Title, HireDate, Extension, ReportsTo, and Notes fields to columns 2 through 8 of the Query grid, respectively, as shown in Figure 15.4.
7. To simplify finding an employee, click the Sort row of the LastName column and select an Ascending sort.
8. Click the Run toolbar button to check your work, and then close the new query. Click Yes when the message box asks if you want to save the query.
9. In the Save As dialog, name the query **qryHRActions** and click OK.

**Figure 15.4**

The query includes all fields (\*) from the HRActions table and seven fields from the Employees table.

Field:	HRActions.*	LastName	FirstName	Title	HireDate	Extension	ReportsTo	Notes
Table:	HRActions	Employees						
Sort:								
Show:	<input checked="" type="checkbox"/>							
Criteria:								
or:								

Now that you've created the query that provides a unified data source for the main form, you're ready to begin creating your custom multitable form.

## CREATING A BLANK FORM WITH A HEADER AND FOOTER

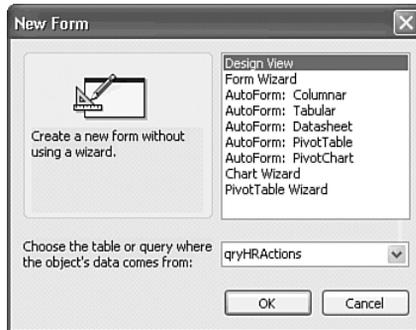
When you create a form without using the Form Wizard, Access opens a default blank form to which you add controls that you select from the Toolbox. To open a blank form to begin duplicating the form you created with the Form Wizard in Chapter 14, do the following:



1. In the Database window, click the Forms shortcut, and click the New button to open the New Form dialog.
2. With the default Design view selected in the upper list of the New Form dialog, select qryHRActions in the lower drop-down list (see Figure 15.5). Click OK.

**Figure 15.5**

Select qryHRActions as the data source for the new form in the New Form dialog.



3. Access creates a new blank form with the default title Form1. By default, the Toolbox and the field list for the qryHRActions query open. Click the Maximize button of the Form Design window to expand the form to fill the document window.

### TIP



If the Toolbox or field list isn't visible, click the appropriate button on the Form Design (top) toolbar. If the grid doesn't appear on the form, choose View, Grid.

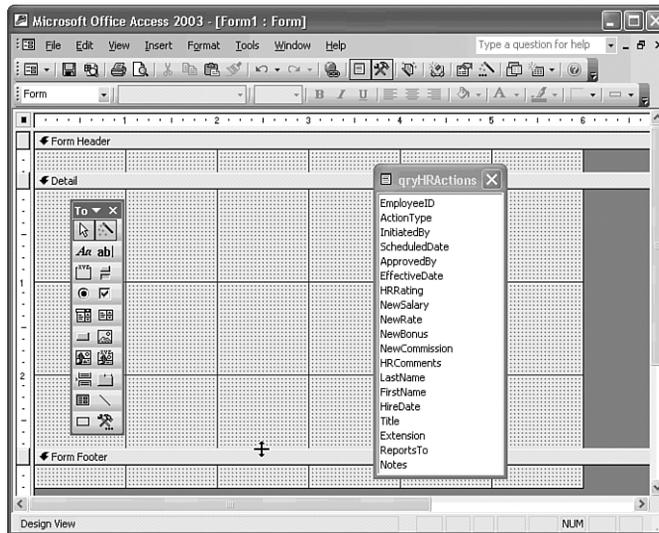
4. Choose View, Form Headers/Footer to add a header and footer to the form.

The default width of blank forms is 5 inches. The default height of the Form Header and Footer sections is 0.25 inch, and the height of the Detail section is 2 inches. To adjust the height of the form's Detail section and the width of the form, do this:

1. Place the mouse pointer on the top line of the Form Footer bar. The mouse pointer becomes a double-headed arrow with a line between the heads. Hold down the left mouse button and drag the bar to create a Detail section height of about 3.0 inches, measured by the left vertical ruler. The active surface of the form, which is gray with the default 24×24 grid dots, expands vertically as you move the Form Footer bar downward, as shown in Figure 15.6.

**Figure 15.6**

When you drag a section header or margin, the mouse pointer changes to a line with a double-headed arrow.



2. Minimize the Form Footer section by dragging the bottom margin of the form to the bottom of the Form Footer bar.
3. Drag the right margin of the form to 6 inches as measured by the horizontal ruler at the top of the form.

## ADDING A LABEL TO THE FORM HEADER

The label is the simplest control in the Toolbox to use. By default, labels are unbound and static, and they display only the text you enter. Static means that the label retains the value you originally assigned as long as the form is displayed, unless you change the Caption value with VBA code. To add a label to the Form Header section, complete the following steps:

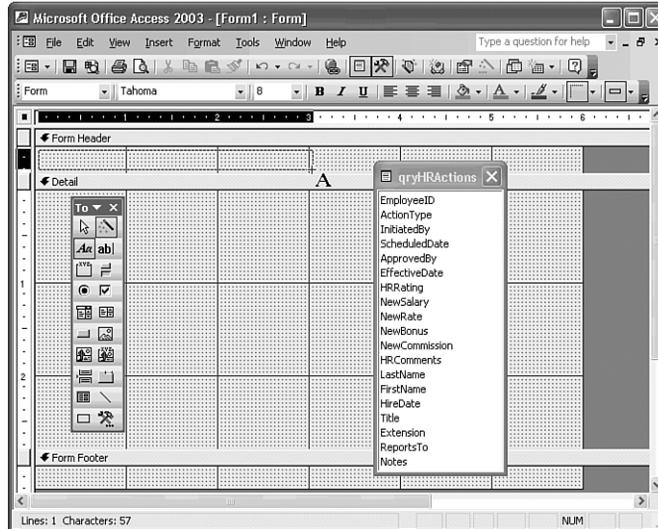
1. Click the Toolbox's Label button. When you move the mouse pointer to the form's active area, the pointer becomes the symbol for the Label button, combined with a

crosshair. The center point of the crosshair defines the position of the control's upper-left corner.

2. Locate the crosshair at the upper-left of the Form Header section. Press and hold down the left mouse button while you drag the crosshair to the position for the lower-right corner of the label (see Figure 15.7).

**Figure 15.7**

Drag the symbol for the control, a label in this example, from the upper left to the lower right to define a rectangle that represents the size of the control.



**NOTE**

As you drag the crosshair, the outline of the container for the label follows your movement. The number of lines and characters that the text box can display in the currently selected font appears in the status bar.

3. If you move the crosshair beyond the bottom of the Form Header section, the Form Header bar moves to accommodate the size of the label after you release the left mouse button. When the label is the size you want, release the mouse button.
4. The mouse pointer becomes the text-editing cursor inside the outline of the label. Type **Human Resources Action Entry** as the text for the label, and click anywhere outside the label to finish its creation. If you don't type at least one text character in a label after creating it, the box disappears the next time you click the mouse.
5. Choose **File, Save**, and type the name **frmHRActionEntry** in the Form Name text box of the Save As dialog. Click **OK**.

→ For tips on manipulating elements of a form, see "Selecting, Moving, and Sizing a Single Control," p. 546.

You use the basic process described in the preceding steps to add most of the other types of controls to a form. (Some Toolbox buttons, such as the graph and command buttons, launch a Control Wizard to help you create the control if the Control Wizards button is activated.)

After you add the control, you use the anchor and sizing handles described in Chapter 14 to move the control to the desired position and to size the control to accommodate the content. The location of the anchor handle determines the Left (horizontal) and Top (vertical) properties of the control. The sizing handles establish the control's Width and Height property values.

## FORMATTING TEXT AND ADJUSTING TEXT CONTROL SIZES

 When you select a control that accepts text as the value, the typeface and font size combo boxes appear on the toolbar (refer to Figure 15.7). To format the text that appears in a label or text box, do the following:

-  1. Click the Human Resources Action Entry label you created in the preceding section to select the label. If the Properties window isn't open, click the Properties toolbar button. Alternatively, double-click the unselected label.

### NOTE



Access 2002 added a drop-down list to the Properties window that lets you select any object on the form or report. Selecting objects from the Properties window's list is faster than selecting from the toolbar's Object list.

2. Open the Font list on the Formatting toolbar and select the typeface family you want.

 Tahoma is Access 2003's default font. (MS Sans Serif was the default typeface in Access 97 and earlier.)

-  3. Open the Font Size list and select 14 points.

-  4. Click the Bold attribute button on the toolbar.

-  5. The size of the label you created probably isn't large enough to display the larger font. To adjust the size of the label to accommodate the content of the label, click the Size to Fit button—if you added it to the Toolbox—or choose **F**ormat, **S**ize, **T**o **F**it. Access resizes the label's text box to display the entire label; if necessary, Access also increases the size of the Form Header section.

### TIP

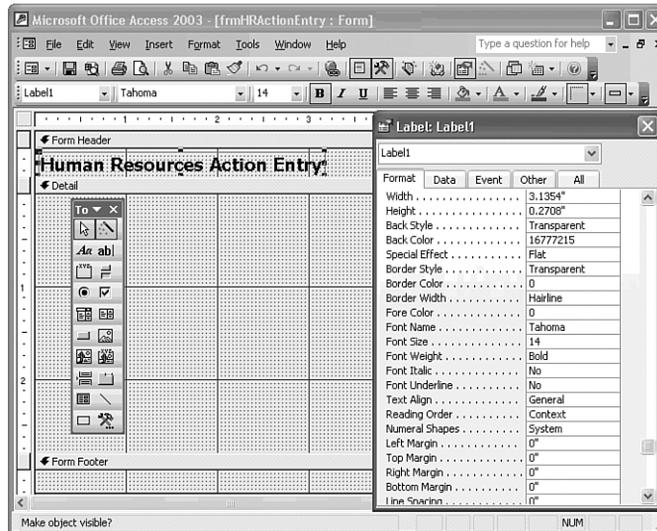


The two **F**ormat commands—**S**ize, **T**o **G**rid and **S**ize, **T**o **F**it—work slightly differently, depending on whether one or more controls are selected. If one or more controls are selected when you execute one of the sizing commands, the command is applied to the selected control(s). If no controls are selected, the chosen sizing command applies as the default to all objects you subsequently create, move, or resize.

When you change the properties of a control, the new values are reflected in the Properties window for the control, as shown in Figure 15.8. If you move or resize the label, you see the label's Left, Top, Width, and Height property values change in the Properties window. You usually use the Properties window to change the property values of a control only if a toolbar button or menu choice isn't available.

**Figure 15.8**

The Properties window reflects changes you make to the property values of a control with toolbar controls.



## NOTE

You can select different fonts and the Bold, Italic, and Underline attributes (or a combination) for any label or caption for a control. Good design practices dictate use of a single font family, such as Tahoma, for all controls on a form. If the PC running your Access application doesn't have the font family you specified, Windows selects the closest available match—usually Arial for sans serif fonts. Changes you make to the formatting of data in controls doesn't affect the data's display in Datasheet view.

## TIP

Change the default format of a control, such as a label, by doing this: Select the control in the toolbox and click the Properties button or press F4 to open the Default *ControlName* window. Click the Format tab and change the property values—such as Font Name, Font Size, and Font Weight—to apply a standard format to the new labels or other controls you add.

## CREATING BOUND AND CALCULATED TEXT BOXES

Following are the most common attributes of Access text boxes:

- *Single-line text boxes* usually are bound to controls on the form or to fields in a table or query.
- *Multiline text boxes* usually are bound to Memo field types and include a vertical scroll bar to allow access to text that doesn't fit within the box's dimensions.
- *Calculated text boxes* obtain values from expressions that begin with = (equal sign) and are usually a single line. Most calculated text boxes get their values from expressions that manipulate table field or query column values; but the =Now expression to supply the current date and time also is common. Calculated text boxes are unbound and read-only. You can't edit the value displayed by a calculated text box.
- *Unbound text boxes* can be used to supply values—such as limiting dates—to Access VBA procedures. As a rule, an unbound text box that doesn't contain a calculation expression can be edited. (An unbound text box control can be set to inhibit editing, but doing so negates the control's purpose in most cases.)

The following sections show you how to create the first three types of text boxes.

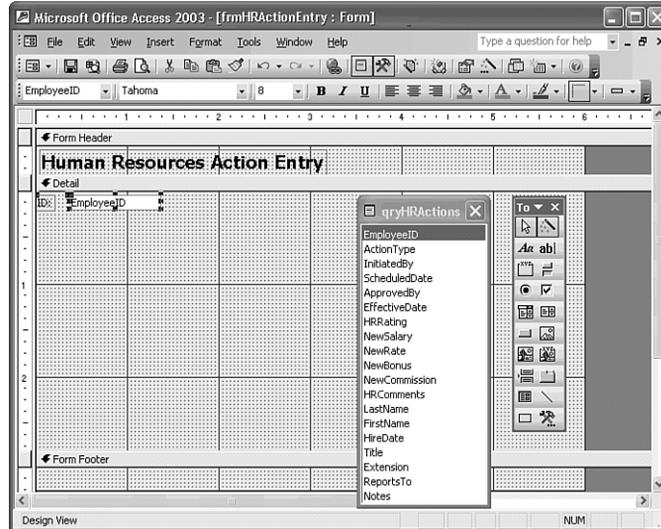
### ADDING BOUND TEXT BOXES

The most common text box used in Access forms is the single-line bound text box that makes up the majority of the controls for the frmHRActions form of Chapter 14. To add a bound text box do the following:

-  1. If necessary, click the Field List button on the toolbar to redisplay the field list.
2. Click and drag the EmployeeID field in the field list window to the upper-left corner of the form's Detail section. When you move the mouse pointer to the active area of the form, the pointer becomes a field symbol, but no crosshair appears. (Notice that Access creates two controls with this action: the text box and its label control.) The position of the field symbol indicates the upper-left corner of the text box, not the label, so drop the symbol in the approximate position of the text box anchor handle, as shown in Figure 15.9.
3. Drag the text box by the anchor handle closer to the ID label, and decrease the box's width.
-  4. Small type sizes outside a field text box are more readable when you turn the Bold attribute on. Select the ID: label and click the Bold button.

**Figure 15.9**

Add a bound text box control by dragging the field name to the position where you want the text box to appear.

**NOTE**

When Access creates a label for a text box that's associated with a form control, the bound object's name is the value for the text label. If the form control is bound to a table object, such as a field, that has a Caption property (and the Caption property isn't empty), Access combines the value of the Caption property with a colon suffix (the colon is a default you can inhibit) as the default value for the text label of the bound form control. When you created the HRActions table in Chapter 5, you set the Caption property for each field name. The EmployeeID field has a Caption property set to ID, so the label for the text box bound to the EmployeeID field is also ID plus a colon.

5. Drag the HRComments field from the list box to the form about 0.75 inch below the ID label, delete the label, and resize the text box to about 1×2 inches, as shown later in Figure 15.10. When you add a text box bound to a memo field, Access automatically sets the Scrollbars property to Vertical, and they appear when the memo is longer than the text box space or you place the cursor in the text box.
6. Press Ctrl+S to save your work.

When you drag fields from the Field list in this manner, you automatically create a bound control. By default, however, all controls you add with the Toolbox are *unbound* controls. You can bind a control to a field by creating an unbound control with a tool and selecting a field in the Control Source property dropdown list (reach the Control Source list by clicking the Data tab in the Properties window for the control).

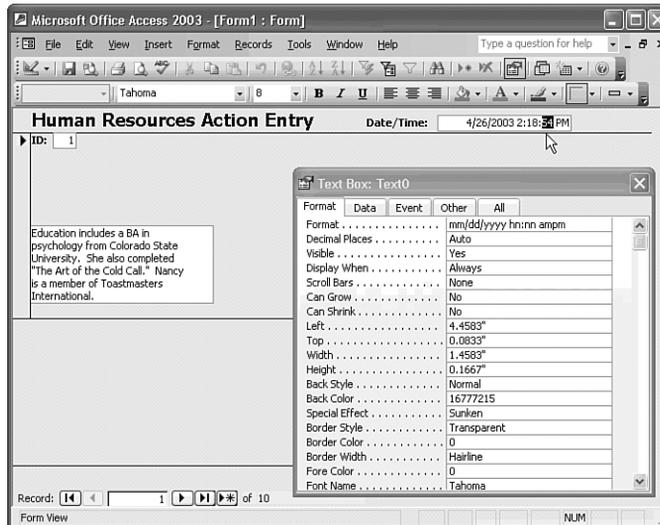
## ADDING A CALCULATED TEXT BOX AND FORMATTING DATE/TIME VALUES

You can display the result of all valid Access expressions in a calculated text box. An expression must begin with = and can use VBA functions to return values. To create a calculated text box that displays the current date and time, do the following:

1. Close the Properties window, click the Text Box tool in the Toolbox, and draw an unbound text box at the right of the Form Header section of the form.
2. Edit the label of the new text box to read **Date/Time:**, and relocate the label so that it is adjacent to the text box. Apply the Bold attribute to the label.
3. Type =Now in the text box to display the current date and time from your computer's system clock; Access adds a trailing parentheses pair for you. Adjust the width of the label and the text box to accommodate the approximate length of the text.
4. Change to Form view and inspect the default date format, MM/DD/YYYY HH:MM:SS AM/PM for North America.
5. To delete the seconds value, open the Properties window for the text box, and click the Format tab. Select the Format property and type mm/dd/yyyy hh:nn ampm in the text box.

Your reformatted Date/Time text box appears as shown in Figure 15.10. Access lets you alter properties of text boxes and other controls in Form *and* Form Design views. When you change the focus to another control, the format string, mm/dd/yyyy hh:nn ampm for this example, properly reformats the text box.

**Figure 15.10**  
The format string you specify for a text box doesn't apply when the Properties window for the text box is open in Form or Form Design view until you change the focus.



**NOTE**

When you return to Design view, the Human Resources Action Entry label you added to the form header has an error correction flag. An error correction flag is a small green triangle in the top left corner of the control. The “Accepting or Declining Control Error Correction” section, which follows shortly, describes dealing with error correction on forms.

**USING THE CLIPBOARD WITH CONTROLS**

You can use the Windows Clipboard to make copies of controls and their properties. As an example, create a copy of the Date/Time control using the Clipboard by performing the following steps:



1. Return to Form Design mode and select the unbound Date/Time control and its label by clicking the field text box. Both the label and the text box are selected, as indicated by the selection handles on both controls.



2. Copy the selected control to the Clipboard by pressing Ctrl+C.



3. Click the Detail bar to select the Detail section, and paste the copy of the control below the original version by pressing Ctrl+V. (Access pastes the control into the top-left corner of the section, so you’ll need to reposition it.)

4. Click the Format property in the Properties window for the copied control, and select Long Date from the drop-down list.



5. To check the appearance of the controls you’ve created, click the View button on the toolbar.



6. Return to Design view and delete the added Date/Time text box and label. To do so, enclose both with a selection boundary created by dragging the mouse pointer across the text boxes from the upper-left to the lower-right corner. Press Delete. (You need the Date/Time text box only in the Form Header section for this form.)

**ACCEPTING OR DECLINING CONTROL ERROR CORRECTION**

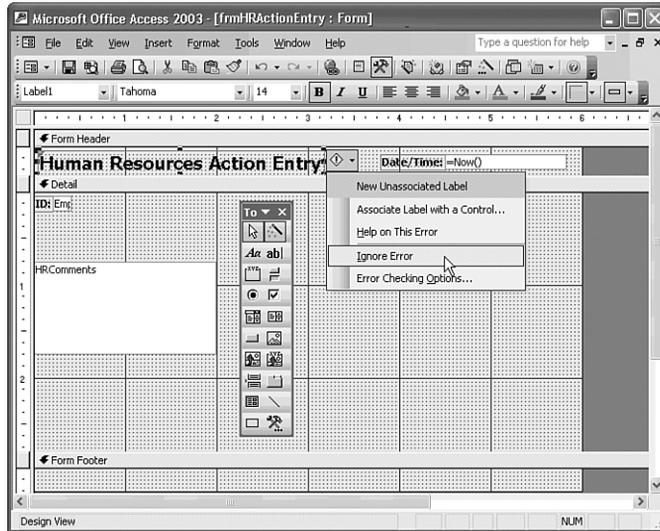
After you add a text box control to the form header section, Access 2003’s new control error correction feature becomes evident in Form Design view. The Human Resources Action Entry label sports a green flag in its upper left corner. When you select a control with an error flag, the error checking smart tag icon—a diamond-shaped sign with an exclamation point—appears to the right of the control.

When you pass the mouse pointer over the icon, an error message screen tip describing the problem appears—in this case: “This is a new label and is not associated with a control.” Clicking the icon’s arrow opens the following list (see Figure 15.11):

- *Error description*—New Unassociated Label for this example.

**Figure 15.11**

When you add an unbound label control that's not associated with a text box, Access 2003 flags the control for error correction and offers these selections to ignore or correct the purported error.



- *Corrective action(s)*—Associate Label with Control for this example, which opens a list of text box(es) in the section to which the label can be associated.
- Help on This Item opens a Microsoft Access Help window with the topic relating to the error.
- Ignore Error removes the flag from the selected control.
- Error Checking Options lets you specify the errors to be flagged or turn off error checking (see Figure 15.12). Clearing the New Unassociated Labels check box and clicking Apply removes the flag from the selected control. Clicking OK with a check box cleared prevents further error checking for the selection.

**Figure 15.12**

The error checking Options dialog lets you specify the types of errors to be flagged by this new feature.



Adding new unassociated labels is a common task, so removing this error check is a good form and report design practice. Changes you make in the error checking Options dialog apply to all databases.

## CHANGING THE DEFAULT VIEW AND OBTAINING HELP FOR PROPERTIES

A form that fills Access's Design window might not necessarily fill the window in Run mode. Run mode might allow the beginning of a second copy of the form to appear. A second copy appears if the Default View property has a value of Continuous Forms. Forms have the following five Default View property values from which you can choose:

- *Single Form* displays one record at a time in one form.
- *Continuous Forms* displays multiple records, each record having a copy of the form's Detail section. You can use the vertical scroll bar or the record selection buttons to select which record to display. Continuous Forms view is the default value for subforms created by the Form Wizard.
- *Datasheet* displays the form fields arranged in rows and columns.
- *PivotTable* displays an empty PivotTable design form, unless you've previously designed the PivotTable.
- *PivotChart* displays an empty PivotChart design form, unless you've previously designed the PivotChart.



### NOTE

PivotTable and PivotChart views of the data source for a form seldom are useful. These views require aggregate values, which are uncommon except in decision-support forms. Rather than use a PivotTable or PivotChart view of the data, add these controls to a form. Chapter 18 describes how to add PivotTable and PivotChart controls to forms.

To change the form's Default View property, do the following:



1. Click the View button on the toolbar to return to Form Design view, if necessary.
2. Press Ctrl+R or choose Edit, Select Form.
3. Click the Properties button if the Properties window isn't visible. Click the Format tab.
4. Click the Default View property and open the list.
5. Select the value you want for this property for the current form. For this exercise, select Single Form (the default) from the list.
6. While Default view is selected, press F1 to open the Help window for the Default View property. This Help window also explains how the Default View and Views Allowed properties relate to each another.

### NOTE

The vertical scroll bar disappears from the form in Run mode if a single form fits within its multiple document interface (MDI) child window.



You can verify your changes, if any, to the Default View property by clicking the View button to review the form's appearance.

## ADDING OPTION GROUPS WITH THE WIZARD

Option buttons, toggle buttons, and check boxes ordinarily return only Yes/No (–1/0 or True/False) values when used by themselves on a form. These three controls also can return Null values if you change the TripleState property value to Yes. Individual bound option button controls are limited to providing values to Yes/No fields of a table or query. When you place any of these controls within an option group, however, the buttons or check boxes can return a number you specify for the value of the control's Option Value property.

The capability to assign numbers to the Option Value property lets you use one of the preceding three controls inside an option group frame for assigning values to the HRRating field of the HRActions table. Option buttons are most commonly used in Windows applications to select one value from a limited number of values.

The Option Group Wizard is one of three Control Wizards that take you step-by-step through the creation of complex controls. To create an option group for the HRRating field of the HRActions table with the Option Group Wizard, follow these steps:



1. Click the Control Wizards tool to turn on the wizards if the toggle button isn't On (the default value).

### NOTE

Access's toggled Toolbox (and toolbar) buttons indicate the On (True) state by a border with a colored background under Windows XP or a white background under Windows 2000. This differs from toggle buttons on forms, which use a very light gray background and a sunken effect to indicate the On (pressed) state. Background colors differ if you've applied a Windows XP desktop theme.



2. Click the Option Group tool, position the pointer where you want the upper-left corner of the option group, and click the mouse button to display the first dialog of the Option Group Wizard.
3. For this example, type five of the nine ratings in the Label Names datasheet (pressing Tab, not Enter): **Excellent**, **Good**, **Acceptable**, **Fair**, and **Poor** (see Figure 15.13). Click Next.

**Figure 15.13**

Type the caption for each option button of the option group in the first dialog of the Option Group Wizard.

Option Group Wizard

An option group contains a set of option buttons, check boxes, or toggle buttons. You can choose only one option.

What label do you want for each option?

Label Names:
Excellent
Good
Acceptable
Fair
Poor

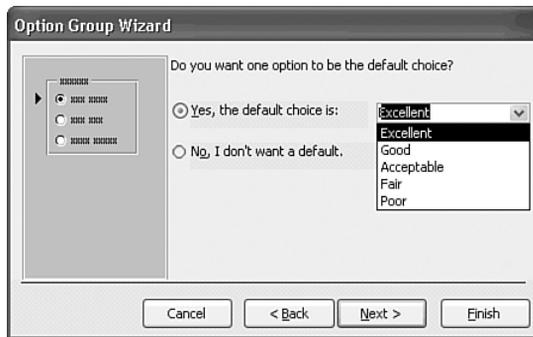
Cancel < Back Next > Finish

**TIP**

You can specify accelerator keys in the captions of your option buttons by placing an ampersand (&) before the letter to be used as an accelerator key. Thereafter, pressing Alt in combination with that letter key selects the option when your form is in Run mode. To include an ampersand in your caption, type &&.

- The second dialog lets you set an optional default value for the option group. Select the option named Yes, the Default Choice Is, and open the drop-down list. Select Good, as shown in Figure 15.14, and click Next. If you need to, you can return to any previous step by clicking Back one or more times.

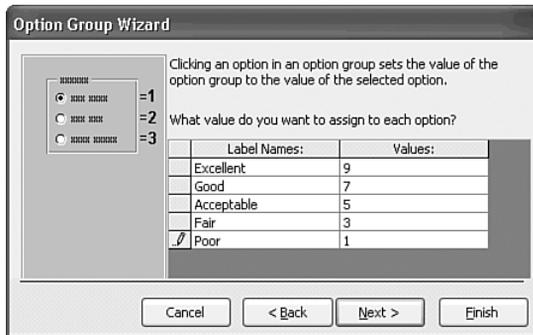
**Figure 15.14**  
Select a default value in the second Option Group Wizard dialog.



**NEW**  
2002

- The third dialog of the Option Group Wizard provides for the assignment of option values to each option button of the group. The default value is the numbered sequence of the buttons. Type 9, 7, 5, 3, and 1 in the five text boxes, as illustrated in Figure 15.15, and click Next.

**Figure 15.15**  
Assign a numeric value to each option button in the group. In Form view, clicking an option button assigns its value to the option frame.

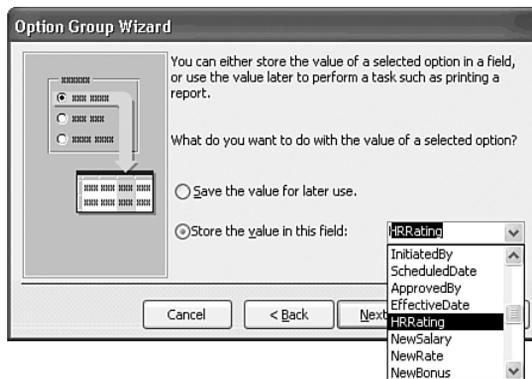


**NOTE**

The domain integrity rule for the HRRating field provides for nine different ratings. Nine option buttons, however, occupy too much space on a form, so this example uses only five of the nine ratings. (Only 5 of the 9 options are included here for the sake of simplicity. In the real world, you wouldn't just eliminate options because there are too many.)

- The fourth Wizard dialog lets you bind the option group to a field of a table or a column of a query that you specified as the Record Source property value of the bound form. Select the HRRating column of the qryHRActions query to which your form is bound (see Figure 15.16). Click Next.

**Figure 15.16**  
Bind the option group value.



- The fifth dialog lets you determine the style of the option group, as well as the type of controls (option buttons, check boxes, or toggle buttons) to add to the option group. You can preview the appearance of your option group and button style choices in the Sample pane. For this example, select Option Buttons and Sunken (see Figure 15.17). The sunken effect matches the default effect applied to text boxes.

**Figure 15.17**  
The fifth Wizard dialog lets you choose the option frame's control type and appearance.



**NOTE**

Check boxes are an inappropriate choice for controls in an option group. Windows programming standards reserve multiple check boxes for situations in which more than one option choice is permissible.

The sunken and raised styles of option groups, option buttons, and check boxes are applicable only to control objects on forms or option groups with a Back Color property other than white.

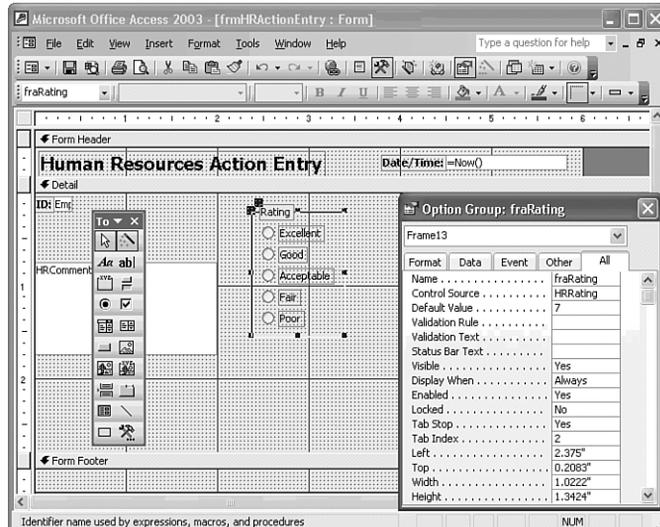
8. The last dialog provides a text box for entering the Caption property value of the label for the option group. Type **Rating**, as shown in Figure 15.18, and click Finish to let the Wizard complete its work.

**Figure 15.18**  
Add the caption for the option group in the last Wizard dialog.



9. Open the Properties dialog for the option frame, and assign the frame a name, **fraRating** for this example. Figure 15.19 shows the completed Rating option group and its properties window in Form Design view.

**Figure 15.19**  
The Properties dialog for the fraRating option frame displays the property values assigned by the Wizard.



**TIP**

Name the controls you add to identify their use, rather than accepting Access default value for the Name property. This book uses object-naming conventions that consist of a three-letter, lowercase abbreviation of the object type—*fra* for frames, *txt* for text boxes, *frm* for forms, and the like—followed by a descriptive name for the control. Using a consistent object naming convention makes it much easier to write (and later interpret) VBA code for automating your application.

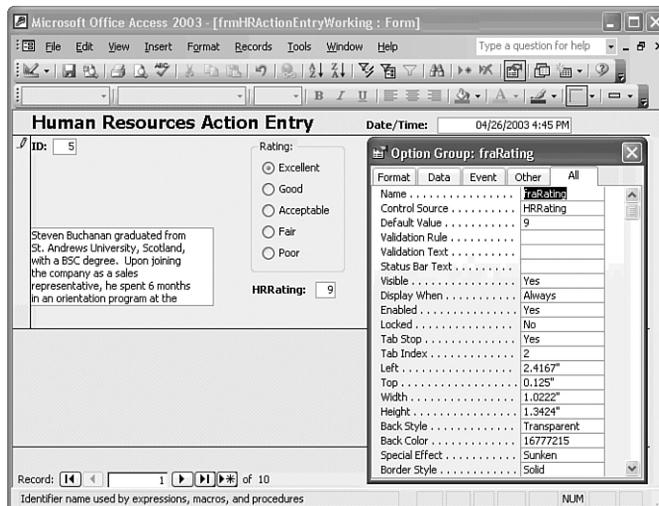
Access 2000 added the capability to change property values in Form view. However, you can change the Name property value of an object only in Form Design view.

→ For more information on Access and VBA naming conventions, see “Typographic and Naming Conventions Used for VBA,” p. 1152.

**abi** To test your new bound option group, select the Text Box tool and drag the HRRating field from the field list to the form to add a text box that’s bound to the HRRating column. Figure 15.20 shows the option group in Form view with the Bold attribute applied to the option group label and the Rating text box added. Click the option buttons to display the rating value in the text box. Although your entry on the form tentatively updates the value onscreen, the value in the table doesn’t change until you move the record pointer or change the view of the form. Press Ctrl+S to save your form.

**Figure 15.20**

Clicking an option button displays its value in the HRRating text box and makes a tentative change to the HRRating field of the current record of the HRActions table.



## CHANGING ONE CONTROL TYPE TO ANOTHER

Access lets you “morph” a control of one type to become a control of a compatible type. You can change an option button to a check box, for example, or you can change a toggle button to an option button. You can’t, however, change a text box to an object frame or other control with a different field data type. To change a control to a different type, follow these steps:

1. In the form's Design view, select the control whose type you want to change.
2. Choose **Format**, **Change To** to see a submenu of form control types. Only the submenu choices for control types that are compatible with the selected control are enabled.
3. Select the control type you want from the submenu. Access changes the control type.

## USING THE CLIPBOARD TO COPY CONTROLS TO ANOTHER FORM

Access's capability of copying controls and their properties to the Windows Clipboard lets you create controls on one form and copy them to another. You can copy the controls in the header of a previously designed form to a new form and edit the content as necessary. The form that contains the controls to be copied need not be in the same database as the destination form in which the copy is pasted. This feature lets you create a library of standard controls in a dedicated form that is used only for holding standard controls. If your library includes bound controls, you can copy them to the form, and then change the field or column to which they're bound.

The Date/Time calculated text box is a candidate to add to Chapter 14's frmHRActions form. You might want to add a Time/Date text box to the Form Header or Detail section of all your transaction forms. To add the Date/Time control to the frmHRActions form, assuming both forms are in the same database, do the following:



1. With the frmHRActionEntry form open, click the Design View button, and select the Date/Time control and its label by clicking the text box.
2. Press Ctrl+C to copy the selected control(s) to the Clipboard.
3. Press Alt+W, 1 and open the frmHRActions form from the Database window in Design view.
4. Click the Detail section selection bar, and press Ctrl+V. A copy of the control appears in the upper-left corner of the Detail section.
5. Position the mouse pointer over the copied text box so that the pointer becomes a hand symbol.
6. Hold down the mouse button and drag the text box and its label to a position to the right of the Title label and text box.



7. Click Form view to display the modified frmHRActions form (see Figure 15.21).
8. Return to Form Design view, click the Save button to save your changes, and close the frmHRActions form.



*If you receive an error message when the focus moves to controls you've copied to another form, see "Error Messages on Copied Controls" in "Troubleshooting" near the end of this chapter.*

**Figure 15.21**  
Copying a previously formatted control and its label from one form to another saves design time.

Type	Init. By	Scheduled	Appr. By	Effective	Rating	Salary	Rate	Bonus	% Comm	HRComments
H	1	05/01/1992	1	05/01/1992		2,000				Hired
Q		04/26/2003		05/24/2003						

## ADDING COMBO AND LIST BOXES

Combo and list boxes both serve the same basic purpose by letting you pick a value from a list, rather than type the value in a text box. These two kinds of lists are especially useful when you need to enter a code that represents the name of a person, firm, or product. You don't need to refer to a paper list of the codes and names to make the entry. The following list describes the differences between combo and list boxes:

- *Combo boxes* consume less space than list boxes in the form, but you must open these controls to select a value. You can allow the user to enter a value in the text box element of the drop-down combo list or limit the selection to just the members in the drop-down list. If you limit the choice to members of the drop-down list (sometimes called a pick list), the user can still use the text box to type the beginning of the list value—Access searches for a matching entry. This feature reduces the time needed to locate a choice in a long list.
- *List boxes* don't need to be opened to display their content; the portion of the list that fits within the size of the list box you assign is visible at all times. Your choice is limited to values included in the list.

In most cases, you bind the combo or list box to a field so that the choice updates the value of this field. Two-column controls often are the most common. The first column contains the code that updates the value of bound field, and the second column contains the name associated with the code. A multiple-column list is most useful when assigning supervisor and manager employee ID numbers to the InitiatedBy and ApprovedBy fields in the frmHRActionEntry form, for example.

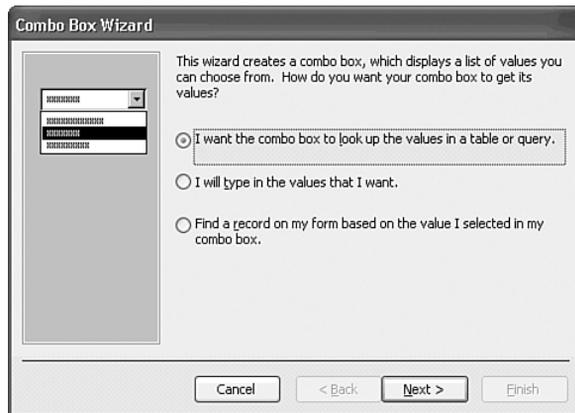
### USING THE COMBO BOX WIZARD

Designing combo boxes is a more complex process than creating an option group, so you're likely to use the Combo Box Wizard for every combo box you add to forms. Follow these steps to use the Combo Box Wizard to create the cboInitiatedBy drop-down list that lets you select from a list of Northwind Traders' employees:

1. Open the frmHRActionEntry form (that you created and saved earlier in this chapter) from the Database window in Form Design view if it isn't presently open.
2. Click the Control Wizards button, if necessary, to turn on the wizards.
3. Click the Combo Box tool in the Toolbox. The mouse pointer turns into a combo box symbol while on the active surface of the form.
4. Click the Field List button to display the Field List.
5. Drag the InitiatedBy field to a position at the top and rightmost edge of the form's Detail section, opposite the EmployeeID field (look ahead to Figure 15.26). The first Combo Box Wizard dialog opens.
6. You want the combo box to look up values in the Employees table, so accept the default option (see Figure 15.22). Your selection specifies Table/Query as the value of the Row Source Type property of the combo box. Click Next.

**Figure 15.22**

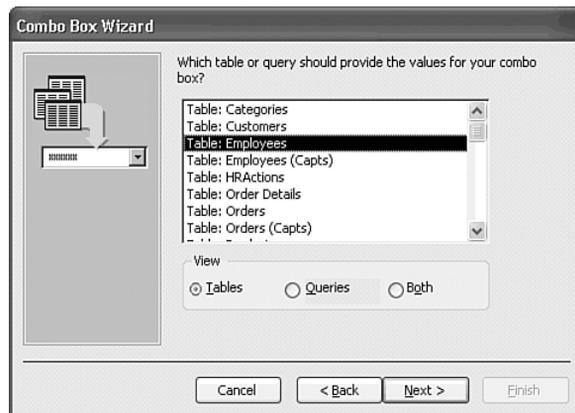
The first Combo Box Wizard dialog lets you select the type of combo box to create. This example uses a lookup-type combo box.



7. In the second Wizard dialog, select Employees from the list of tables (see Figure 15.23), and click Next.

**Figure 15.23**

Select the table or query to provide the list items of the combo box in the Wizard's second dialog. Use a base table, Employees for this example, to assure that the list doesn't contain multiple entries for a single lookup value.



8. For this example, the combo box needs the EmployeeID and LastName fields of the Employees table. EmployeeID is the field that provides the value to the bound column of the query, and your combo box displays the LastName field. EmployeeID is selected in the Available Fields list by default, so click the > button to move EmployeeID to the Selected Fields list. LastName is then selected automatically, so click the > button again to move LastName to the Selected Fields list. Your Combo Box Wizard dialog appears as shown in Figure 15.24. This selection generates the Jet SQL `SELECT` query that serves as the value of the combo box's Row Source property and populates its list. Click Next.

**Figure 15.24**

In the third Wizard dialog, add the bound column and one or more additional columns to display in the combo box list.



**TIP**

If two or more employees have the same last name, add the FirstName field to the combo list. Unlike conventional combo and list boxes, Access controls can display multiple columns.



9. Access 2003 adds a new sorting dialog to the Combo Box Wizard. To sort the list by last name, open the first list and select the LastName field (see Figure 15.25). Selecting a sort on one or more fields adds an `ORDER BY` clause to the combo box's `SELECT` query.
10. The fifth dialog displays the list items for the combo box. Access has successfully determined that the EmployeeID field is the key field of the Employees table and has assumed (correctly) that the EmployeeID field binds the combo box.

**NOTE**

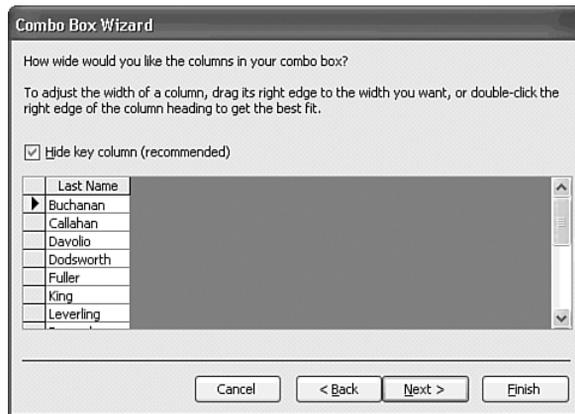
The Hide Key Column check box is selected by default; this option causes Access to hide the bound column of the combo box. You've selected two columns for the combo box, but only one column (the LastName field) displays in the combo box's list. The EmployeeID column is hidden and used only to supply the data value for the InitiatedBy field.

**Figure 15.25**  
In the new Wizard sorting dialog, select the field(s) on which to apply an ascending or descending sort. Clicking an Ascending button toggles an Descending or Ascending sort.



11. Resize the LastName column by dragging the right edge of the column to the left—you want the column wide enough to display everyone’s last name but not any wider than absolutely necessary (see Figure 15.26). Click Next.

**Figure 15.26**  
The Wizard queries the combo box’s data source (the Employees table) and displays the control’s list items. Double-click the right edge of the list to size the list’s width to fit the list items.

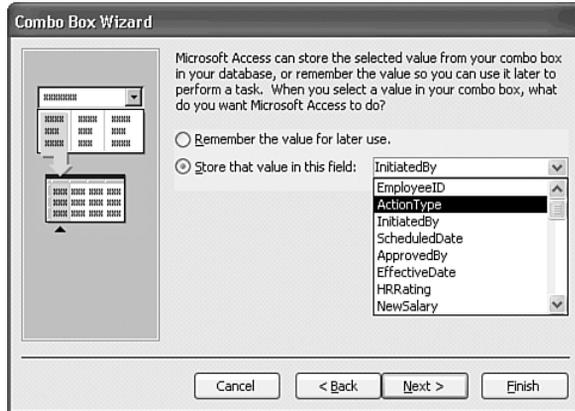


**NOTE**

Resizing the list width doesn’t accomplish its objective in Access 2003 or the previous two versions. The Wizard adds a combo box of the size you created when dragging the tool on the form, regardless of the width you specify at this point.

12. Your combo box updates the InitiatedBy field the EmployeeID value corresponding to the name you select. You previously specified that the Control Source property is the InitiatedBy column when you dragged the field symbol to the form in step 5. The Combo Box Wizard uses your previous selection as the default value of the Control Source property (see Figure 15.27), so accept the default by clicking the Next button to display the sixth and final dialog.

**Figure 15.27**  
The fifth Wizard dialog specifies the column of the query to be updated by the combo box selection.



13. The last dialog lets you edit the label associated with the combo box (see Figure 15.28). Type **Initiated by:** and click Finish to add the combo box to your form.

**Figure 15.28**  
Type the label caption for the combo box for the sixth and last Wizard dialog.

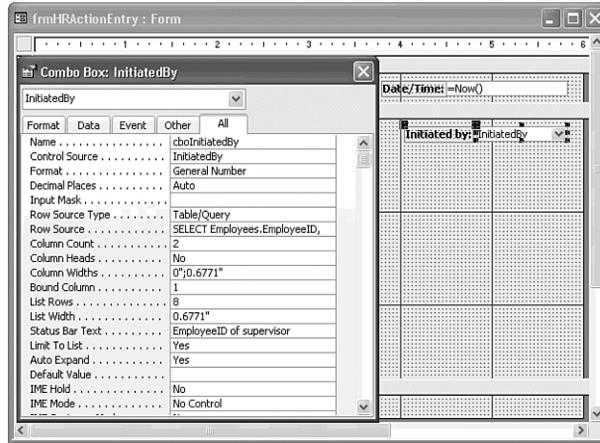


14. Apply the bold attribute to the combo box label, and adjust the width and position of the label. Open the Properties window for the combo box, and change its name to **cboInitiatedBy**. Figure 15.29 shows the new combo box in Form Design view.

#### NOTE

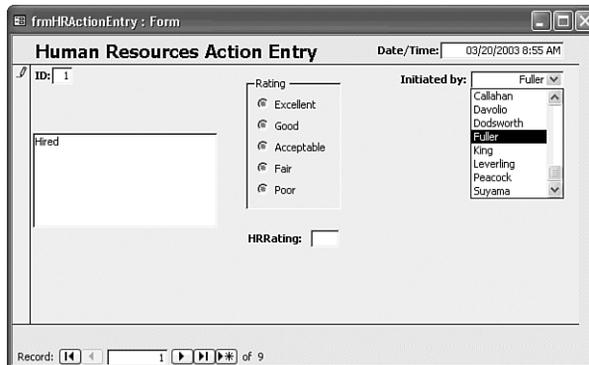
The Row Source property is the SQL `SELECT` statement that fills the combo box's list. Specifying a Column Width value of 0 hides the first column. The Description property of the EmployeeID field provides the default Status Bar Text property value.

**Figure 15.29**  
The Combo Box Wizard sets the property values for the combo box.



- 15. Change to Form view to test your combo box. Change the Initiated by value to another person, and then use the navigation buttons to move the record pointer and make the change permanent. Return to the original record, and verify that the combo box is bound to the InitiatedBy field (see Figure 15.30).

**Figure 15.30**  
The combo box in Form view displays a list with the default maximum of eight items. You can change the depth of the list by specifying a different value for the List Rows property.



**Jet SQL**

The Jet SQL statement generated by the Combo Box Wizard for cboInitiatedBy is

```
SELECT Employees.EmployeeID, Employees.LastName
FROM Employees
ORDER BY [LastName];
```

**TIP**

If you don't use the Wizard to generate the combo box, you can select an existing table or query to serve as the Row Source for the combo box.

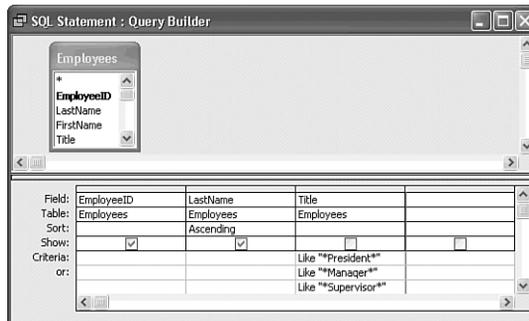
## USING THE QUERY BUILDER TO POPULATE A COMBO BOX

If the Row Source Type property for a combo box is Table/Query, you can substitute a custom SQL statement for a named table or query as the value of the Row Source property. For either tables or queries, you can choose only the fields or columns you want for the text box, eliminating the need to hide columns. In addition, you can specify a sort order for the list element of your combo box and specify criteria to limit the list.

To invoke Access's Query Builder and create an SQL statement for populating a manually added Approved by combo box, follow these steps:

1. Return to or open frmHRActionEntry in Design view, and click to disable the Toolbox's Control Wizards button to add the combo box manually. Click the Field List button, if necessary, to display the field list.
2. Click the Combo Box button in the Toolbox, and then drag the ApprovedBy field to add a new combo box under the Initiated By combo box you added in the preceding section. Select the new control and open the Properties window if necessary.
3. Select the Row Source property, and click the Build button to launch the Query Builder. The Query Builder window is identical in most respects to the Query Design window, but its title and behavior differ.
4. Add the Employees table to the query, and then close the Show Table dialog. Drag the EmployeeID, LastName, and Title fields to the Query Design grid.
5. You want an ascending sort on the LastName field, so select Ascending in the Sort check box. Only presidents, vice-presidents, managers, and supervisors can approve HR actions, so type **Like \*President\*** in the first Criteria row of the grid's Title column, **Like \*Manager\*** in the second, and **Like \*Supervisor\*** in the third. Access adds the quotation marks surrounding the Like argument for you. Clear the Show check box of the Title column. Your query design appears as shown in Figure 15.31.

**Figure 15.31**  
This query design limits approval to employees whose titles include President, Manager, or Supervisor.



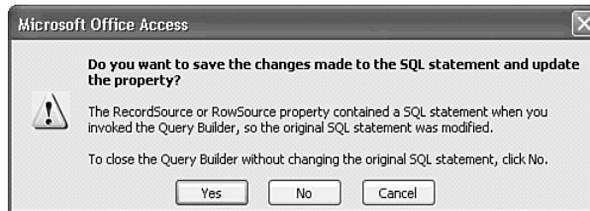
**TIP**



Test the results of your query by clicking the Run button on the toolbar. Access executes the query and displays a Datasheet view of the query's results. For this example, only Mr. Buchanan and Dr. Fuller meet the criteria.

6. Close the Query Builder. The message box shown in Figure 15.32 appears to confirm your change to the Row Source property value, instead of asking if you want to save your query. Click Yes and the SQL statement derived from the graphical Query Design grid becomes the value of the Row Source property.

**Figure 15.32**  
This query design supplies the corresponding Jet SQL statement as the value of the combo box's Row Source property.



7. In the combo box's Properties window, change the name of the combo box to **cboApprovedBy**. Change the Column Count property value to **2** and type **0.2;0.8** in the Column Widths text box. You specify column widths in inches, separated by semicolons, Access adds the units—double quotes (“) for inches—to the widths. (Metric users specify column widths in cm.) Finally, change the Limit to List value to **Yes**.

**TIP**

You can display only the LastName field in the combo box, making the combo box similar in appearance to that for the InitiatedBy field, by setting the first Column Width value to **0**.



8. Change the label caption to **Approved by:** and apply the Bold attribute.



9. Switch to Form view to test the effect of adding the sort (the ORDER BY clause) and criteria (the WHERE clause) to the query (see Figure 15.33). Press Ctrl+S to save your form changes.

**Jet SQL**

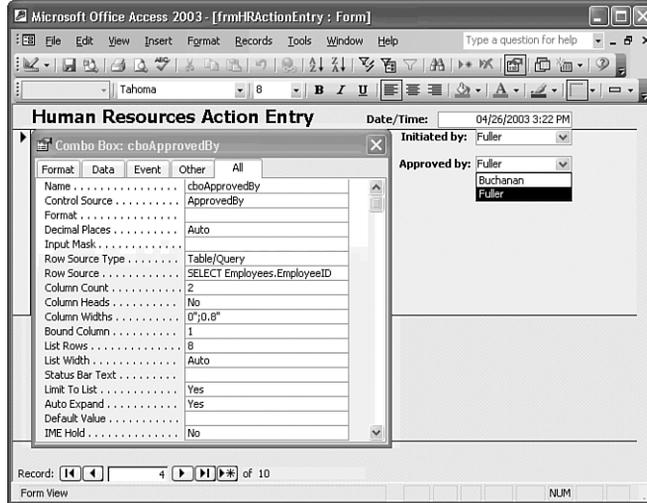
The Jet SQL statement generated by the Query Builder is

```
SELECT Employees.EmployeeID, Employees.LastName
FROM Employees
WHERE (((Employees.Title) Like "**President*")) OR
(Employees.Title) Like "**Manager*") OR
(Employees.Title) Like "**Supervisor*")
ORDER BY Employees.LastName;
```

Jet SQL uses the DOS and UNIX \* and ? wildcards for all characters and a single character, respectively. T-SQL requires the ANSI SQL wildcards % and \_, and surrounds character strings with a single-quote rather than double-quotes. The table name prefixes aren't needed, and the parentheses in the WHERE clause are superfluous.

**Figure 15.33**

The combo box list contains items for employees whose titles comply with the Like criteria.



## T-SQL

The simplified T-SQL equivalent of the preceding Jet SQL statement for ADP is

```
SELECT EmployeeID, LastName
FROM dbo.Employees
WHERE Title LIKE '%President%' OR
       Title LIKE '%Manager%' OR
       Title LIKE '%Supervisor%'
ORDER BY LastName
```

The `dbo.` prefix—called the *schema* component of the table name—in the preceding statement is optional, but is a common practice in T-SQL statements.

It's a more common practice for ADP to use SQL Server 2000 views, stored procedures, or table-returning functions to provide the Row Source for forms, combo boxes, and list boxes.

## CREATING A COMBO BOX WITH A LIST OF STATIC VALUES

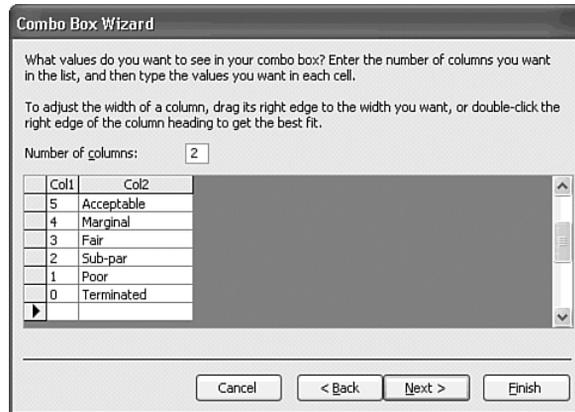
Another application for list boxes and combo boxes is picking values from a static list of options that you create. A drop-down list to choose a Rating value saves space in a form compared with the equivalent control created with option buttons within an option group. As you design more complex forms, you find that display “real estate” becomes increasingly valuable.

The option group you added to the `frmHRActionEntry` form provides a choice of only 5 of the possible 10 ratings. To add a drop-down list with the Combo Box Wizard to allow entry of all possible values, do the following:



1. Change to Form Design view, and click the Control Wizards button in the Toolbox to enable the Combo Box Wizard.
2. Open the Field List window, and then click the Combo Box tool in the Toolbox. Drag the HRRating field symbol to a position underneath the cboApprovedBy combo box you added previously.
3. In the first Wizard dialog, select the I Will Type in the Values That I Want option (refer to Figure 15.22), and then click Next to open the second dialog.
4. The Rating combo box requires two columns: The first column contains the allowable values of HRRating, 0 through 9, and the second column contains the corresponding description of each rating code. Type 2 as the number of columns.
5. Access assigns value-list Row Source property values in column-row sequence; you enter each of the values for the columns in the first row and then do the same for the remaining rows. Type **9 Excellent, 8 Very Good, 7 Good, 6 Average, 5 Acceptable, 4 Marginal, 3 Fair, 2 Sub-par, 1 Poor, 0 Terminated** (use the Tab key and don't type the commas).
6. Set the widths of the columns you want by dragging the edge of each column header button to the left, as shown in Figure 15.34. If you don't want the rating number to appear, drag the left edge of column 1 fully to the left to reduce its width to 0. When you've adjusted the column widths, click Next to open the third dialog.

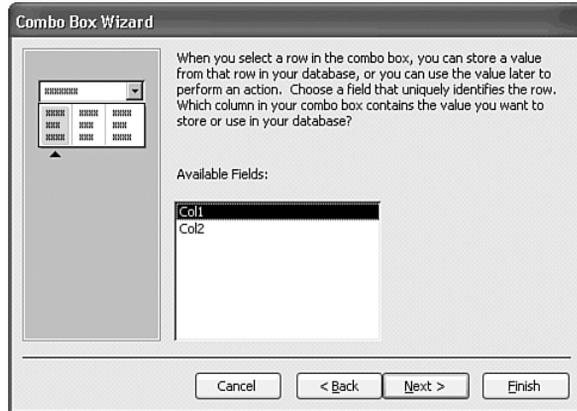
**Figure 15.34**  
Type the values for the two columns in the list, and then adjust the column widths to suit the list's contents.



7. Select Col1, the HRRating code, as the bound column for your value list—that is, the column containing the value you want to store or use later (see Figure 15.35); this column must contain unique values. Click Next to open the fourth dialog.

**Figure 15.35**

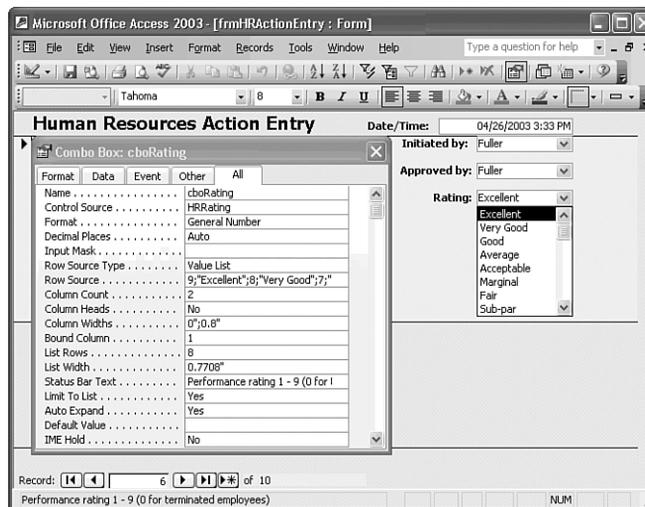
Select the column that contains the unique value to identify the rows of the list, in most cases, Col1.



8. Accept the default value (the HRRating column) in the fourth dialog, and click Next to open the final dialog of the Combo Box Wizard.
- B** 9. Type **Rating:** as the label for the new combo box control, apply the Bold attribute to the label, and then click Finish to complete the combo box specification and return to Form Design view.
-  10. Open the Properties window for the combo box, change the Name to **cboRating**, and then click the Data tab in the Properties window. Set Limit to List to Yes to convert the drop-down combo to a drop-down list. Quickly review the Row Source property. Notice that the Wizard has added semicolons between the row entries, and quotation marks to surround the text values in the Row Source property. You use this format when you enter list values manually.
-  11. Change to Form view. The open Rating static-value combo box and its Properties window appear as shown in Figure 15.36.

**Figure 15.36**

The value-list version of the cboRating combo box closely resembles the cboApprovedBy combo box.



Another opportunity to use a static-value combo box is as a substitute for the Type text box. Several kinds of performance reviews exist: Quarterly, Yearly, Bonus, Salary, Commission, and so on, each represented by an initial letter code.

**TIP**



You can improve the appearance of columns of labels and associated text, list, and combo boxes by right-aligning the text of the labels and left-aligning the text of the boxes. Select all the labels in a column with the mouse, and click the Align Right button on the toolbar. Then select all the boxes and click the Align Left button.

### CREATING A COMBO BOX TO FIND SPECIFIC RECORDS

The Combo Box Wizard includes a third type of combo list box that you can create—a combo list that locates a record on the form based on a value you select from the list. You can use this type of combo box, for example, to create a Find box on the frmHRActionEntry form that contains a drop-down list of all last names from the Employees table. Thus, you can quickly find HRActions records for employees.

To create a combo box that finds records on the form based on a value you select from a drop-down list, follow these steps:



1. Change to Design view, and click the Control Wizards button in the Toolbox, if necessary, to enable the Combo Box Wizard.



2. Click the Combo Box tool in the Toolbox, and then click and drag on the surface of the form's Detail section to create the new combo box in a position underneath the cboRating combo box you created previously. Release the mouse, and the first Combo Box Wizard dialog appears. When you don't drag a column name to the form, you create an unbound combo box.
3. Select the Find a Record on My Form Based on the Value I Selected in My Combo Box option, and click Next (refer to Figure 15.22).
4. In the second Wizard dialog, scroll the Available Fields list until the LastName field is visible. Click to select this field, and then click the > button to move it to the Selected Fields list (see Figure 15.37). Click Next to open the third dialog.

**TIP**

When creating a combo box to find records, select only one field. The combo box won't work for finding records if you select more than one field for the combo box's lists.

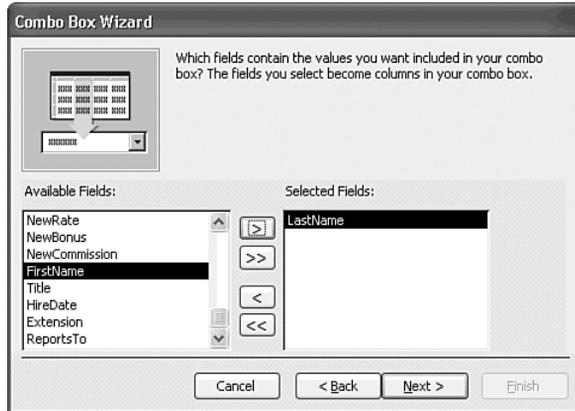
If the record source contains more than one person with the same last name, you need to add a calculated FullName query column to use the find-record combo box version.

For this example the expression to create a FullName query column is `FullName:`

```
[LastName] & ", " & [FirstName].
```

**Figure 15.37**

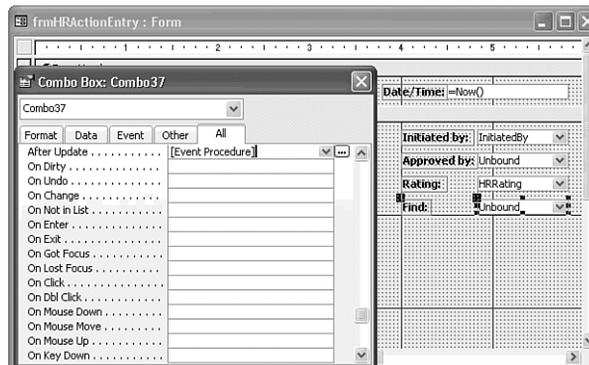
Select the name of the field to search in the Available Fields list, and click > to add the entry to the Selected Fields list.



5. The Combo Box Wizard now displays a list of the field values from the column you just selected. Double-click the right edge of the LastName column to get the best column-width fit for the data values in the column, and then click Next to go to the fourth and final step of the Wizard.
- B** 6. Type **Find:** as the label for the new combo box, and then click Finish to complete the new combo box control. After applying the bold attribute to the label and adjusting its size, your form appears as shown in Figure 15.38.

**Figure 15.38**

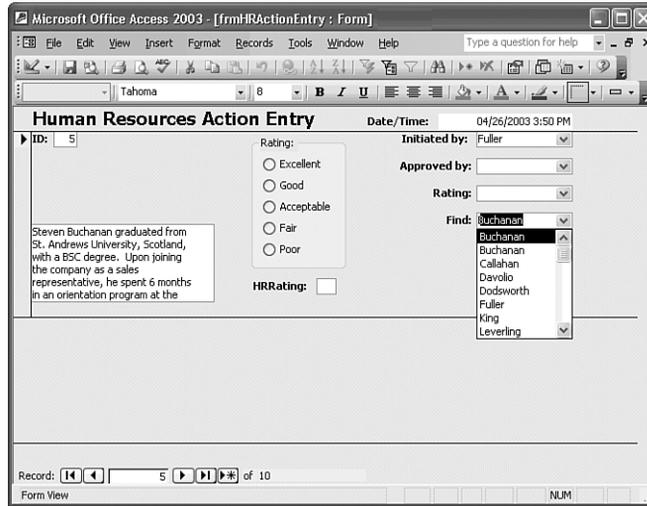
The record-finding version of the combo box uses an event procedure to move the record pointer to the first record matching the combo box selection.

**CAUTION**

Don't change the name of the new combo box at this point. If you change the name at this point, the Find combo box won't work in Form view.

- ☐** 7. Click the Form View button on the toolbar to display the form. The open Find: combo box appears as shown in Figure 15.39.
8. Press Ctrl+S to save your work so far.

**Figure 15.39**  
The combo box finds the records for last name Buchanan. If you have more than one record for an employee, multiple instances of the LastName value appear in the list at this point.



**TIP**

Always use unbound combo box controls for record selection. If you bind a record-selection combo box to a field, the combo box updates field values with its value.

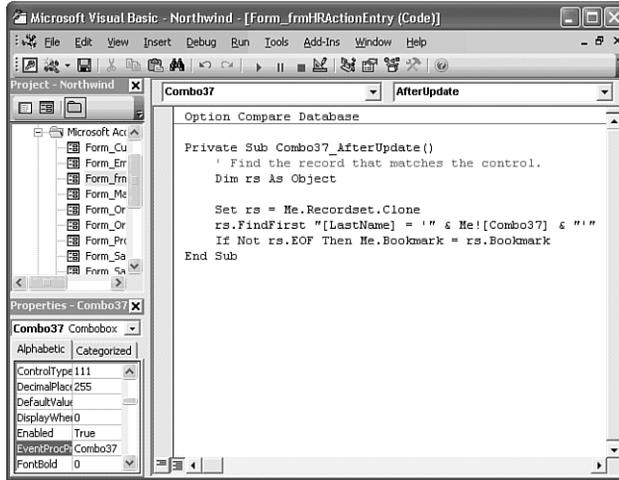
When you create this type of combo box, the Combo Box Wizard automatically creates a VBA event subprocedure for the After Update property of the combo box (refer to the Property window in Figure 15.35). An event subprocedure is a VBA procedure that Access executes automatically whenever a particular event occurs—in this case, updating the combo box. Chapter 27, “Learning Visual Basic for Applications,” describes how to write Access VBA code and Chapter 28, “Handling Events with VBA 6.0” describes how to write event-handling subprocedures.



To view the event procedure code that the Wizard created for your new combo box, change to Design view, open the Properties window for the Name: combo box, click the Events tab in the window, select the After Update property text box, and then click the Build button. Access opens the VBA Editor window shown in Figure 15.40. After you’ve looked at the code, close the VBA Editor and return to Design view.

To use a combo box of this type, select a value from the list. As soon as you select the new value, Access updates the combo box’s text box, which then invokes the VBA code for the After Update event procedure. The VBA code in the After Update procedure finds the first record in the form’s Recordset with a matching value and displays it. You can use this type of combo box only to find the first matching record in a Recordset.

**Figure 15.40**  
The Combo Box Wizard generates the Combo37\_AfterUpdate VBA subprocedure to find the record.



### TIP

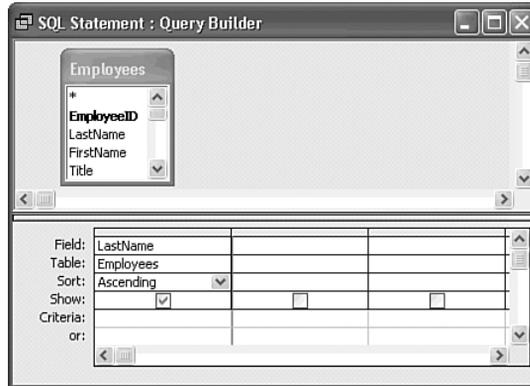
If you change the name of the combo box to comply with the naming convention mentioned earlier, you must also change the name of the VBA procedure. For example, replace the two instances of `Combo37` in the VBA code shown in Figure 15.40 with **cboFind**, close the VBA code editor, then change the Name property value of the combo box to **cboFind**, and finally set the After Update event's value to [Event Procedure]. Name AutoCorrect, which is enabled by default in the General Page of the Options dialog, doesn't change the names of VBA event procedures to correspond to changes of object names, or vice-versa.

Because the field on the form is based on the `LastName` column of the form's underlying query, you see an entry in the list for every last name entry in the Recordset produced by the `qryHRActions` query. If, for instance, more than one Personnel Action record exists for Steve Buchanan, Buchanan appears in the combo list as many times as there are records for him. If an employee doesn't have a record in the `qryHRActions` query result set, the name doesn't appear in the list. To display a unique list of all employee last names, change the Row Source property to obtain the `LastName` field values for the combo box list through an SQL statement based on a query from the `Employees` table.



To change the Row Source property, follow the procedure you learned in the "Using the Query Builder to Populate a Combo Box" section, earlier in this chapter: Open the Properties window of the Name: combo box, click the Data tab, select the Row Source text box, and then open the Query Builder. Change the query so that it uses the `LastName` field of the `Employees` table, add an ascending sort, as shown in Figure 15.41, and change the Limit to List property value to Yes.

**Figure 15.41**  
 Changing the Row Source of the combo box to a query against the Employees table eliminates duplicate items in the Find: combo box.



## USING BOUND OBJECT FRAMES

Access OLE Object fields let you embed or create a link to graphic images, sound or video files, or any other type of object for which you have an OLE 2.0 server installed and registered on your computer. OLE Object fields are unique to Access, and other applications (such as Visual Basic) can't directly read the data the fields contain. Access adds a special "OLE wrapper" to the data that identifies the OLE 2.0 server used to create and display or play the data.

The tab control that you add later in the chapter includes a bound object frame to display a photo for each employee. To use a bound object frame, you must add a field of the OLE Object type to the Employees table, and then insert the images from the files into the Employees table. The sections that follow describe how to add an OLE Object field to a table, insert objects into the field, and test displaying bitmap objects in a temporary bound object frame.

**NOTE**

Storing images in OLE Object fields isn't considered a good database design practice, especially if you expect to handle a large number of images. This isn't an issue, however, with the nine Northwind employee photos that are used in the following example.

## ADDING AN OLE OBJECT FIELD TO THE EMPLOYEES TABLE



The Employees table of early versions of Access used an OLE Object field to hold employee photos, and the Employees form displayed the images in a bound object frame. Access 2002 changed the Photo field to a field of the Text data type, which contains the names of individual bitmap files—EmpID1.bmp through EmpID9.bmp—stored in the ...\\Office10\\Samples folder. The Employees form contains VBA code to display the appropriate image in an image control.

**NOTE**

Microsoft's objective in substituting linked .bmp files for embedded bitmaps isn't clear. The reason might have been to make the Employees table compatible with SQL Server, which doesn't support OLE Object fields. However, the Categories table has a Picture OLE Object field. Even less clear is the reason for using the .bmp format, which consumes much more storage space than a compressed image format, such as Graphics Interchange Format (.gif) or Joint Photographic (Experts) Group (.jpg).

To add a new OLE Object field to the Employees table, do this:

1. Close the frmHRActionEntry form and any open queries against the Employees table.
2. In the Database window, create a backup copy of the Employees table by selecting it, pressing Ctrl+C, and then pressing Ctrl+V. Type a name for the backup, such as **Employees\_Orig**, in the Paste Table As dialog, and click OK.
-  3. Open the Employees table in Design view, and select the Notes field.
4. Press Insert to add a new field, type **PhotoOLE** as the field name, and set the Field data type to OLE Object.
-  5. Change to Datasheet view, and save your change to the table design.

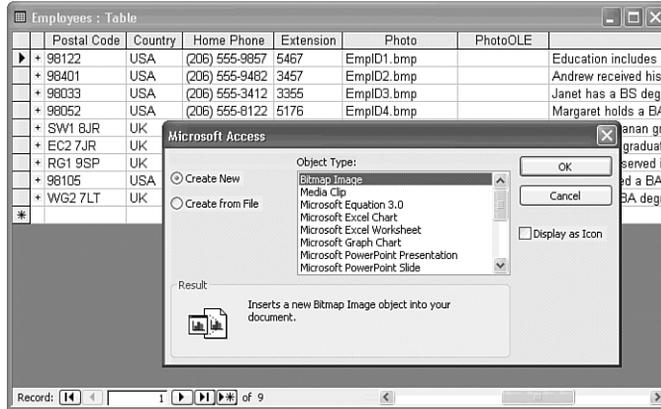
## EMBEDDING OR LINKING IMAGES IN AN OLE OBJECT FIELD

Embedding the object's data is safer than creating an OLE link to the object's source file, because someone might move the source files. Linking the source files doesn't save space in the database, because the OLE Object field stores the last version of the image, called its *presentation*. Linking assures that modifications to the image's source file appear when you display the image in a bound object frame. This example uses embedded data from the nine sample EmpID?.bmp files, but the process is identical for any file type that has an association with an OLE 2.0 server.

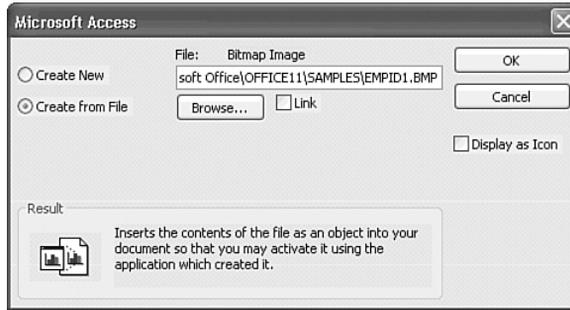
To embed or link the sample bitmap files to the PhotoOLE field, do this:

-  1. With the Employees table open in Datasheet view, navigate to the PhotoOLE field of the first record.
2. Right-click the PhotoOLE cell and choose Insert Object to open a Microsoft Access dialog with an Object Type list, which includes all OLE 2.0 servers registered by your computer (see Figure 15.42).
3. Select the Create from File option, click Browse, and navigate to the \Program Files\Microsoft Office\Office 11\Samples folder.
4. Double-click Empid1.bmp in the folder to add the file to the File: Bitmap Image text box (see Figure 15.43). If you want to link, rather than embed, the data, click Link before clicking OK.

**Figure 15.42**  
Right-clicking an OLE Object field and choosing Insert Object opens a dialog with a list of registered OLE 2.0 servers.



**Figure 15.43**  
Selecting the Create from File option changes the dialog to provide a text box to enter the name of the file to embed or link.



5. Move to each of the remaining eight records in succession, repeating steps 2, 3, and 4 to add Empid2.bmp through Empid9.bmp to the PhotoOLE field.



6. After you've added all nine bitmaps, double-click one of the Bitmap Image cells to display the image in Microsoft Paint (see Figure 15.44).

**Figure 15.44**  
Double-clicking a cell of an OLE Object field opens the object in its OLE 2.0 server, in this case Microsoft Paint.



**NOTE**

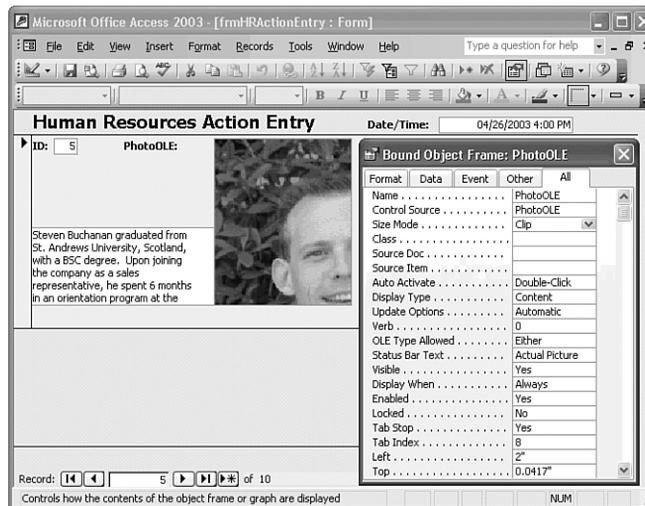
If .bmp files are associated with another OLE 2.0 server on your computer, such as Adobe Photoshop, the associated server opens. Microsoft Paint is the default server for .bmp files, if another application hasn't assumed this role.

**DISPLAYING OLE OBJECT BITMAPS IN A BOUND OBJECT FRAME**

Bound object frames display or play OLE objects in a form, and print bitmap and vector-based images in reports. To add a temporary bound object frame to the frmHRActionEntry form that displays the bitmap objects in the PhotoOLE field of the query, do the following:

1. Close the Employees table, if it's open, and open qryHRActions in Query Design view.
2. Drag the PhotoOLE field from the Employees table to the empty column to the right of the Notes column of the grid. Close and save your changes.
3. Open frmHRActionEntry in Form Design view, and display the Toolbox and Field List.
4. Click the Bound Object Frame control in the Toolbox, and drag the PhotoOLE field from the Field List near the upper right corner of the fraRatings option frame.
5. Adjust the size of the bound object frame to about 1.5×1.7 inches.
6. Change to Form view, and open the Properties window. The default Size Mode property of the control is Clip, so a cropped image of an employee photo opens in the frame (see Figure 15.45).

**Figure 15.45**  
When you open a bitmapped or other image in a bound object frame, the default mode is Clip. Clip displays a cropped version of the full-size image.

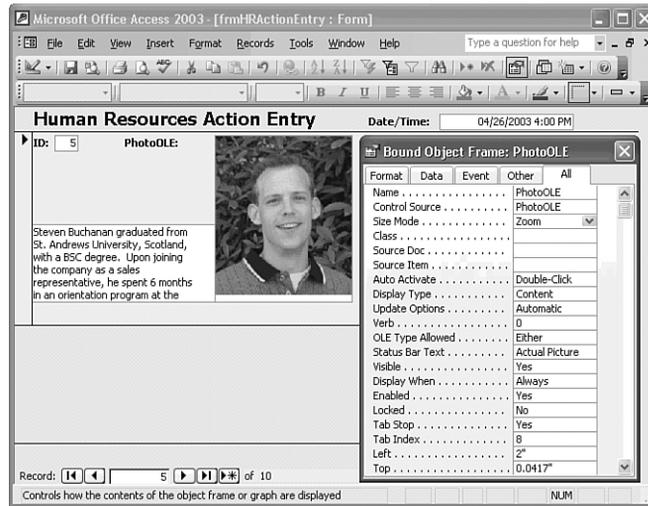


**TIP**

If the employee photo doesn't appear in the bound object frame, close the form, save your changes, and reopen it in Form view.

7. Change the Size Mode property value to Zoom, which sizes the image to fit within the frame but doesn't change its aspect ratio—the ratio of width to height (see Figure 15.46). Stretch mode expands both the width and height of the image to fit the frame, which can result in distortion of the image.

**Figure 15.46**  
Change the Size Mode property value from Clip to Zoom to fit the image within the frame without changing the image's aspect ratio.



8. Double-click the image to edit the bitmap *in situ* with Windows Paint. Paint's menu is grafted to Access's menu, the Paint toolbox opens to the left of the form, an OLE activation border surrounds the Clip mode version of the image, and Paint's palette appears at the bottom of the form (see Figure 15.47).
9. Click outside the image to deactivate the object, then right-click the frame, and choose Bitmap Image **O**bject, **O**pen to open the object in a linked instance of Paint. It's usually easier to edit images in the server's window than in the smaller in-situ frame.
10. To prevent users from activating the image with a double-click, click the Other tab (in the Properties window), and set the Auto Activate property value to Manual.
11. After you've experimented with the bound object frame, return to Form Design view and delete the frame.



The process for adding a static image to an unbound object frame control is similar to that for a bound object frame. You add the unbound frame to the form, right click the frame, and choose Insert Object to add the static image to the control. Charts you create with the Office Chart Wizard in Chapter 18 display in an unbound object frame, but you also can store copies of static charts in an OLE Object field.

**Figure 15.47**  
Activating the image with a double-click enables in-situ editing of the object with the designated OLE 2.0 server.



## WORKING WITH TAB CONTROLS

The tab control lets you easily create multipage forms in a tabbed dialog, similar to the tabbed pages you've seen in the Properties window, in the Options dialog, and elsewhere in Access. The tab control is a very efficient alternative to creating multipage forms with the Page Break control. You can use the tab control to conserve space onscreen and show information from one or more tables. The sections that follow show you how to add images to a new OLE object field of the Employees table, add a tab control to a form, and display images in a bound image control on a tab control page. You also learn to set the important properties of the tab control as a whole, as well as the properties of individual pages of the tab control.

### ADDING THE TAB CONTROL TO A FORM

To add a tab control to the frmHRActionEntry form, follow these steps:

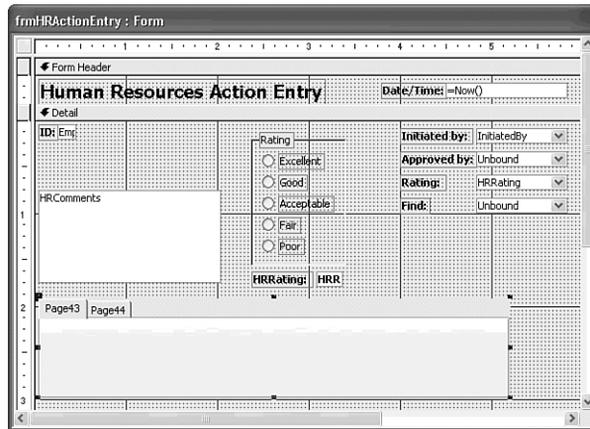


1. Click the Design View button on the toolbar if the frmHRActionEntry form isn't already in Design view. No wizard for the tab control exists, so the status of the Control Wizards button doesn't matter.



2. Click the Tab Control tool in the Toolbox; the mouse cursor changes to the Tab Control icon while it's over the active surface of the form.
3. Click and drag on the surface of the form's Detail section to create the new tab control near the bottom center of the form (see Figure 15.48).

**Figure 15.48**  
Access's default tab control has two pages.



By default, Access creates a tab control with two pages. Each page's tab displays the name of the page combined with a sequential number corresponding to the number of controls you placed on your form in this work session. The next few sections describe how to change the page tab's caption, add or delete pages in the tab control, add controls to the pages, and set the page and tab control properties.

## ADDING TAB CONTROL PAGES

Depending on the data you want to display and how you want to organize that data, you might want to include more than two pages in your tab control. To add a page to a tab control, follow these steps:

1. In Design view, right-click the tab control to open the context menu.
2. Choose **I**nsert Page; Access inserts a new page in the tab control to the right of the last page.

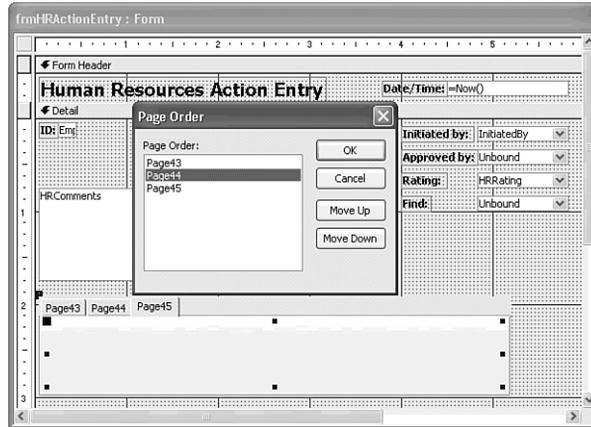
## CHANGING THE PAGE ORDER

Because Access adds a new page only after the last page, it isn't possible to add a new page at the beginning or middle of the existing tab pages. As a result, if you want the new tab control page to appear in another location in the tab control, you must change the order of pages in the tab control. You might also want to change the order of tab control pages as you work with and test your forms—in general, you should place the most frequently used (or most important) page at the front of the tab control.

To change the order of pages in a tab control, follow these steps:

1. Right-click one of the tabs and choose **P**age Order to open the Page Order dialog shown in Figure 15.49.

**Figure 15.49**  
Change the left-to-right sequence of the tabs with the Page Order dialog.



2. In the Page Order list, select the page whose position you want to change.
3. Click the Move Up or Move Down buttons, as appropriate, until the page is in the position you want.
4. Repeat steps 3 and 4 until you have arranged the tab control pages in the order you want, and then click OK to close the Page Order dialog and apply the new page order to the tab control.

## DELETING A TAB CONTROL PAGE

At some point, you might decide that you don't want or need a page in a tab control. The `frmHRActionEntry` form needs only two pages in its tab control. If you added a page to the tab control by following the steps at the beginning of this section, you can delete a page from the tab control by following this procedure:

1. Right-click the page tab of the page you want to delete; Access brings that page to the front of the tab control.
2. Choose Ddelete Page; Access deletes the currently selected tab control page.

## SETTING THE TAB CONTROL'S PROPERTIES

Two sets of properties govern the appearance and behavior of a tab control. A set of properties exists for the entire tab control, and a separate set of properties exists for each page in the tab control. The following list summarizes the important properties of the tab control and its pages; the remaining property settings for the tab control and its pages are similar to those you've seen for other controls (height, width, color, and so on):

- *Caption* is a text property, which controls the text that appears on the page's tab and applies to individual tab control pages only. If this property is empty (the default), then the page's Name property is displayed on the page's tab.

- *MultiRow* is a Yes/No property, which applies to the tab control as a whole and determines whether the tab control can display more than one row of tabs. (The Options dialog, reached by choosing **T**ools, **O**ptions, is an example of a multirow tabbed dialog.) The default setting is No; in this case, if there are more tabs than fit in the width of the tab control, Access displays a scroll button in the tab control. If you change this property to Yes and there are more page tabs than will fit in the width of the tab control, Access displays multiple rows of tabs.
- *Picture* displays an icon in any or all the page tabs. You can use any of Access's built-in icons or insert any bitmapped (.bmp) graphics file as the page's tab icon.
- *Style* applies to the tab control as a whole and controls the style in which the tab control's page tabs are displayed. The default setting, **T**abs, produces the standard page tabs you're accustomed to seeing in the Properties window and in various dialogs in Access and Windows. Two other settings are available: **B**uttons and **N**one. The **B**uttons setting causes the page tabs to be displayed as command buttons in a row across the top of the tab control. The **N**one setting causes the tab control to omit the page tabs altogether. Use the **N**one setting if you want to control which page of the tab control has the focus with command buttons or option buttons located outside the tab control. However, using command buttons external to the tab control to change pages requires writing Access VBA program code. You should use the default **T**abs setting unless you have a specific reason for doing otherwise—using the **T**abs setting ensures that the appearance of your tab controls is consistent with other portions of the Access user interface. Using this setting also saves you the effort of writing VBA program code.
- *Tab Fixed Height* and *Tab Fixed Width* apply to the tab control as a whole and govern the height and width of the page tabs in the control, respectively. The default setting for these properties is 0. When these properties are set to 0, the tab control sizes the page tabs to accommodate the size of the **C**aption for the page. If you want all the page tabs to have the same height or width, enter a value (in inches or centimeters, depending on your specific version of Access) in the corresponding property text box.



To display the Properties window for the entire tab control, right-click the edge of the tab control, and choose **P**roperties from the resulting context menu. Alternatively, click the edge of the tab control to select it (clicking the blank area to the right of the page tabs is easiest), and then click the Properties button on the toolbar to display the Properties window.

To display the Properties window for an individual page in the tab control, click the page's tab to select it, and then click the Properties button on the toolbar to display the page's Properties window.

The tab control in the frmHRActionEntry form uses one page to display current information about an employee: the employee's job title, supervisor, company telephone extension, hire date, and photo. The second tab control page displays a history of that employee's HRActions in a subform you add later in the chapter.



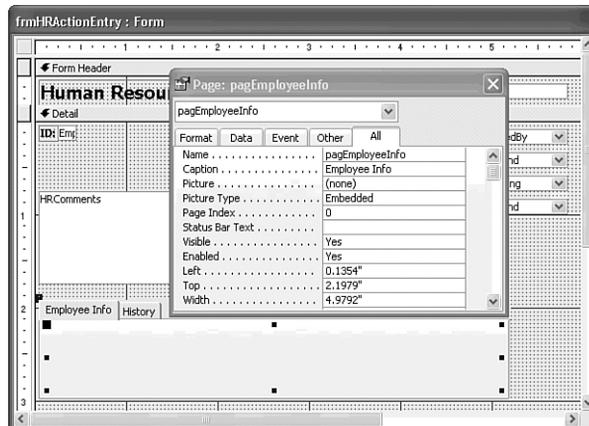
Follow these steps to set the Caption property for the frmHRActionEntry form's tab control:

1. Open the frmHRActionEntry form, and change to Form Design view, if necessary.
2. Click the first page of the tab control to select it, and then click the Properties button on the toolbar to display that page's Properties window.
3. Click the Format tab, if necessary, to display the Format properties for the tab control page.
4. Type **Employee Info** in the Caption property's text box.
5. Click the Other tab and change the Name property value to **pagEmployeeInfo**.
6. Click the second page of the tab control to select it; the contents of the Property dialog change to show the properties of the second tab control page. Click the Format tab.
7. Type **History** in the Caption property text box for the second page of the tab control, type **pagHistory** in the Name property of the Other page, and close the Properties window.
8. Click outside the tabbed region to select the entire tab control, and type **tabHRAction** as the name of the control.

Figure 15.50 shows the tab control with both page captions set and the first page of the tab control selected. Notice that the sizing handles visible in the tab control are inside the control—this position indicates that the page, not the entire control, is currently selected. When the entire tab control is selected, the sizing handles appear on the edges of the tab control.

**Figure 15.50**

Set the Page properties by clicking the tab of one of the pages. Click the empty area to the right of the tabs to set the properties of the entire tab control.



## PLACING OTHER CONTROLS ON TAB PAGES

You can place any of Access's 16 other types of controls on the pages of a tab control—labels, text boxes, list boxes, even subforms. To add a control of any type to a tab control's page, follow this procedure:

1. In Design view, click the page tab you want to add the control to; Access selects the page and brings it to the front of the tab control.
2. Add the desired control to the tab control's page using the techniques presented earlier in this chapter for creating controls on the main form.

Alternatively, you can copy controls from the same or another form and paste them into the tab control's pages by using the same techniques you learned for copying and pasting controls on a form's Detail and Header/Footer sections. You can't drag controls from the form's Detail or Header/Footer sections onto the tab control's page or vice-versa.

As you proceed with the examples in this chapter and complete the frmHRActionEntry form, you place various bound and unbound controls on the pages of the tab control.

## OPTIMIZING THE FORM'S DESIGN

The preceding sections of this chapter have shown you how to use Toolbox controls without regard to positioning the controls to optimize data entry operations. In this section, you add more controls from the qryHRActions query's field list to the main form's Detail section and the Company Info page of the tab control. You place new controls for adding or editing fields of the HRActions table on the main form, and relocate the controls you added earlier into a logical data-entry sequence. The Employee Info page of the tab control displays reference data from the Employees table. Multi-page tab controls are especially effective for displaying data that's related to the entries you make on the main form.

To add and rearrange the forms controls to optimize data entry, follow these steps (look ahead to Figure 15.51 for control placement and formatting):



1. Return to Design view, if necessary, and delete the Rating option group and the HRRating text box. When you delete an option group, you automatically delete the option buttons within the frame.



2. Click the Field List button on the toolbar to open the Field List dialog if it isn't already open.
3. Drag the LastName field from the Field List to a position to the right of the ID field text box. Edit the field's label to read **Name:**.
4. Drag the FirstName field from the Field List to a position to the right of the LastName field; delete the FirstName field's label.
5. Drag the ActionType field from the Field List to a position at the right of the FirstName field to add a combo box.

6. Repeat step 5 for the ScheduledDate, EffectiveDate, NewSalary, and NewBonus fields (see Figure 15.51 for field positioning and sizing). You must move the InitiatedBy, ApprovedBy, HRRating, and Name text boxes that you placed on the form earlier in this chapter.
7. Resize the HRComments field so that it's underneath the EmployeeID and name fields (see Figure 15.51). Next, resize the tab control so that it fills the width of the form and extends from an area below the HRComments field to the bottom of the form. The tab control needs to be as large as possible to display as much data as possible in the sub-form that you add later to its second page.
8. Click the first tab of the tab control to bring it to the front, and then drag the Title field from the Field List to a position near the top-left corner of the Company Info page.
9. Repeat step 8 for the ReportsTo, Extension, and HireDate fields (see Figure 15.51 for field placement).
10. Drag the PhotoOLE field onto the right side of the tab control's first page, and delete its label (the fact that this field displays a photo of the employee is enough to identify the field). Size and position the Photo field at the right edge of the tab control's page; you might need to resize the tab control and the form after inserting the Photo field.
11. Right-click the PhotoOLE field, and choose Properties to display its Properties window, click the Format tab, and select the Size Mode property's text box. Select Zoom from the drop-down list to have the employee photo scaled down to fit the photo field's size.

**TIP**

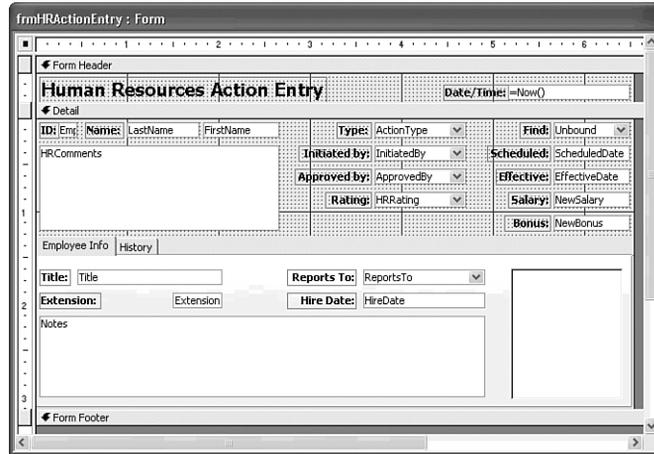
Use the Format Painter to format the text labels of the fields.

→ For more information on creating a uniform appearance with the Format Painter, see "Using the Format Painter," p. 551.

12. Drag the Notes field onto the bottom left side of the tab control's first page and delete its label.
13. Use the techniques you learned in Chapter 14 to move, rearrange, and change the label formats to match the appearance of Figure 15.51. (All labels are bold, sized-to-fit, and right-aligned.)
14. Change the Format property value of the ScheduledDate and EffectiveDate text boxes to **mm/dd/yyyy** to assure Y2K compliance, and change the Format property value of the Salary and Bonus text boxes to **\$#,##0** to add a dollar sign.
15. To replace number with text in the Type, ApprovedBy, and Rating combo boxes, change the Column Widths property value from 0.2";0.8" to **0";0.8"**.
16. Click the Select Form button or press Ctrl+R, and set the Caption property value of the form to **Human Resources Action Entry**.

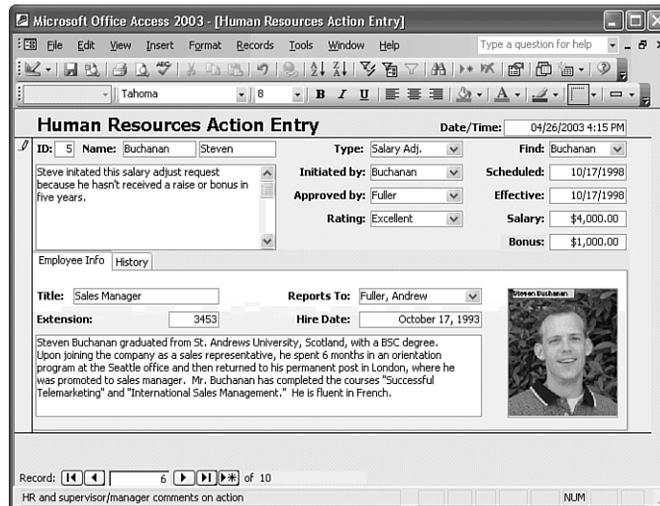


**Figure 15.51**  
Here's the final design of the form with new controls and formatting.



- 17. Press Ctrl+S to save your changes, and test your new and modified controls by changing to Form view (see Figure 15.52).

**Figure 15.52**  
The form is now complete, except for the addition of a subform to the History page of the tab control.



## ADDING A HISTORY SUBFORM TO A TAB CONTROL PAGE

The frmHRActionEntry form needs a subform to display the history of HRActions for the employee displayed in the main part of the form. The HRActions table provides the data source for the subform. Access's Subform/Subreport control offers a Subform Wizard, which lets you quickly add a new subform. The Subform Wizard of prior Access versions

added a datasheet-style subform; the Access 2002 Subform Wizard added a columnar subform, despite the image of a tabular subform in the Wizard's first dialog. This change rendered the Access 2002 Subform Wizard useless for all but very specialized applications. Access 2003's Subform Wizard lets you add an existing form as a subform, as described in the following sections.

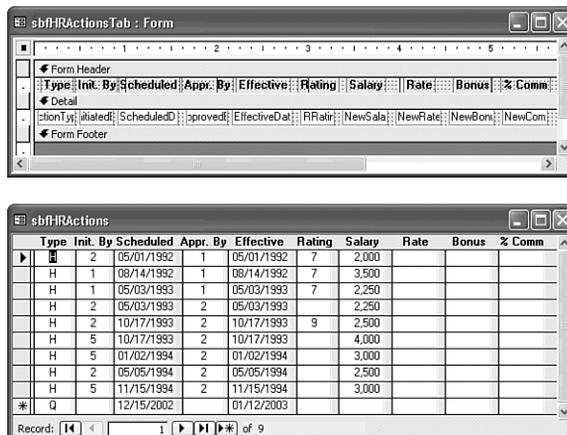
## CREATING A MODIFIED HRACTIONS SUBFORM

If you didn't create the sbfHRActions subform in the preceding chapter, import the sbfHRActions subform from your \Program Files\Seua11\Chaptr14\Forms14.mdb database.

Follow this drill to adapt the sbfHRActions subform for use on the History page of the tab control:

1. Select sbfHRActions in the Forms page of the Database window, and press Ctrl+C and Ctrl+V to create a copy of the subform named **sbfHRActionsTab** and open it in Design view.
2. Delete the HRComments header and text box.
3. Right-click the ActionType combo box, and choose **Change To, Text Box**.
4. Shift-click to select the ActionType, InitiatedBy, ApprovedBy, and HRRating text boxes, and click the toolbar's Center button to center the text and labels.
5. To minimize the height of the subform's header select all labels and drag them to the top of the Form header section. Click the Bold button of the Formatting toolbar to apply the bold attribute to all labels. Drag the Detail section bar up to the bottom of the labels.
6. Select all text boxes and drag them to the bottom of the Detail bar. Drag the Form Footer bar up to the bottom of the text boxes.
7. Reduce the width of the subform to about 5.75 inches, and save your changes (see Figure 15.53).

**Figure 15.53**  
Form Design and Form views show the changes you make to the layout for the sbfHRActionsTab subform.



## ADDING THE sbfHRACTIONS TAB SUBFORM WITH THE WIZARD

To add the sbfActionsTab subform to the History page of the tab control with the Subform Wizard, do the following:



1. With the Wizards activated, open frmHRActionEntry in Design view, click the History tab of the tab control, and select the Subform/Subreport tool in the toolbox.
2. Drag the mouse pointer icon, which assumes the shape of the Subform/Subreport tool, to the History page. The icon changes to a pointer when you reach the active region of the History page, which changes from white to black.
3. Release the mouse to open the Subform Wizard's first dialog.
4. Select the Use an Existing Form option and select sbfHRActionEntryTab in the list (see Figure 15.54). Click Next.

**Figure 15.54**  
The Subform Wizard's first dialog lets you select the existing form to use as a subform.

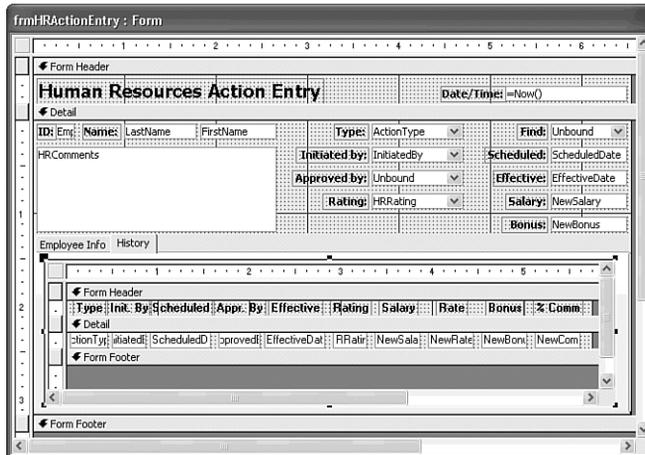


5. In the second Wizard dialog, accept the Choose from a List option and the Show HRActions for Each Record in qryHRActions Using Employee[ID] (see Figure 15.55). Click Next.
6. Accept sbfHRActionEntryTab as the name of the subform, and click Finish to dismiss the Wizard.
7. Delete the label and adjust the size of the subform to occupy most of the available area of the History page (see Figure 15.56).
8. Change to Form view, select Buchanan in the Find list, and click the History tab to display the hired entry for Steven Buchanan and a tentative append record (see Figure 15.57).

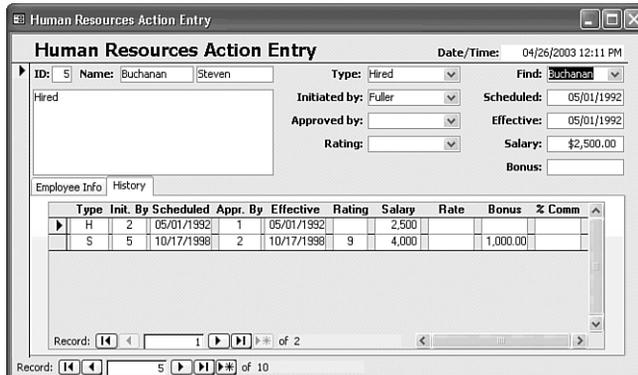
**Figure 15.55**  
The second dialog is where you specify values of the LinkChildFields and LinkMasterFields property value, EmployeeID for this example.



**Figure 15.56**  
After the Wizard adds the subform, adjust its dimensions to suit the active area of the tab control's page.



**Figure 15.57**  
Form view with the History page selected displays HRActions record(s) for the employee selected in the Find combo box.





**NOTE**

The frmHRActionEntry and sbfHRActionEntryTab forms, the qryHRActions query, the HRActions table, and the Employees table with the added PhotoOLE field are included in the \Seau11\Chaptr15\Forms15.mdb database on the accompanying CD-ROM.

## MODIFYING THE DESIGN OF CONTINUOUS FORMS

The default design of the History page’s subform as created by the Subform Wizard lets you edit records of the HRActions table. The term History implies read-only access to the table in the tab control. Therefore, you should alter the properties of the subform to make the form read-only and remove unnecessary controls. For example, the vertical scroll bar lets you display any HRActions record for the employee, so you don’t need record navigation buttons, nor do you need record selectors.

Access 2003’s in-situ subform editing feature lets you change many of the properties of subforms with the main form open in Design view. Unfortunately, you can’t change property values, such as Record Selectors and Navigation Buttons, that affect the structure of the subform. Therefore, you must change the design of the subform independently of its main form container. Access 2003 has a command—Subform in New Window—that provides a shortcut for changing subform properties.

To further optimize the design of the sbfHRActionEntryTab subform, follow these steps:



1. Return to Design view, click the History tab, select and right-click the subform, and choose Subform in New Window to open sbfHRActionEntryTab in Design view, and then click the Properties button to display the properties of the subform.
2. In the Format page of the Properties window, set Scroll Bars to Vertical Only, Record Selectors to No, and Navigation Buttons to No.
3. In the Data page, set the Recordset Type to Snapshot. Doing so has the same effect as setting Allow Edits, Allow Deletions, and Allow Additions to No. Your subform is now read-only because all snapshot-type Recordsets are read-only.
4. Select all text boxes and set the Tab Stop property to No for the group. Close the subform and save your changes.
5. In the now-empty History page, reduce the width of the subform and the tab control by about 1/8-inch to reflect removal of the Record Selector buttons. You can reduce the width of the subform container only when the subform isn’t open for in-situ editing.
6. In the main form, select the tab control, and set its Tab Stop property to No. Do the same for the LastName and FirstName text boxes.
7. Set the tab order by choosing View, Tab Order to open the Tab Order dialog. Click Auto Order to set the tab order of the controls for which the Tab Order property is Yes. The default control tab order is top-right to bottom-left. Click OK to close this dialog.



- Return to Form view and click the History tab to verify your changes to the subform (see Figure 15.58).

**Figure 15.58**

Form view reflects subform linking on the EmployeeID field and the changes you made to the design and dimensions of the subform.

Type	Init. By	Scheduled	Appr. By	Effective	Rating	Salary	Rate	Bonus	% Comm
H	2	05/01/1992	1	05/01/1992		2,500			
S	5	10/17/1998	2	10/17/1998	9	4,000		1,000.00	

### TIP

Removing text boxes and other controls from the tab order that you seldom or can't edit speeds data entry. To further optimize data entry, set the Tab Stop property of all controls on both pages of the tab control to No. Labels don't have a tab stop control; if you multi-select all controls on a page, use Shift+Click to deselect the labels to set Tab Stop to No for all other controls.

## ADDING NEW RECORDS IN THE HRACTIONENTRY FORM

Unlike Chapter 14's frmHRActions form, the Record Source for the main frmHrActionEntry form is a query. The query design takes advantage of Access' row fix-up feature when you add a new record to the HRActions table in the form. Row fix-up works in this case, because the source of the qryHRActions query's EmployeeID column is the HRActions table.

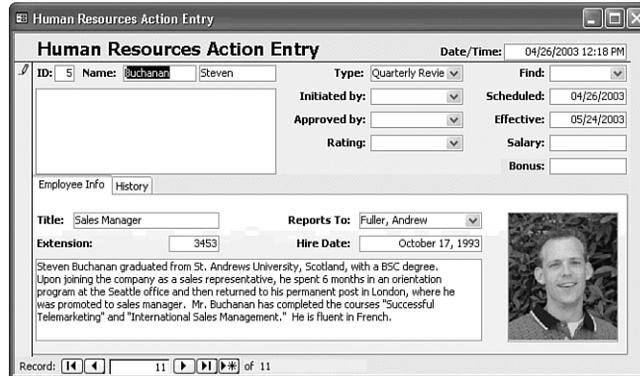
- For a review of row fix-up in one-to-many queries, see "Taking Advantage of Access's Row Fix-Up Feature," p. 431.

To add a new record to the HRActions table, do the following:



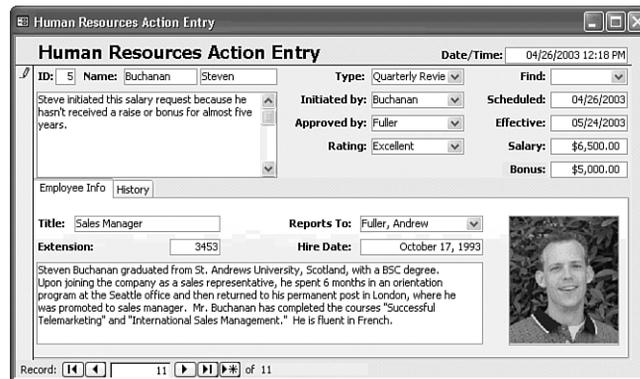
- Open frmHrActionEntry in Form view, and click the tentative append (new) record button to add a new record. All data disappears from the form.
- Type a number (other than 1, the default) in the [Employee]ID text box; for this example type 5. Press Tab or Enter to add data in the query columns from the Employees table and default values from the HRActions table to the form's controls (see Figure 15.59).

**Figure 15.59**  
Adding a new record and typing an employee ID in the ID text box fills the form with employee data and default values from the qryHRActions query.



3. Make selections in the combo boxes, change the default dates, if necessary, and add a note regarding the action. Choose Records, Save Record or, if your cursor isn't in the Notes multi-line text box, press Shift+Enter to save the record (see Figure 15.60). Shift+Enter in a multi-line text box adds a newline character and doesn't save the record.

**Figure 15.60**  
Completing the record addition requires only a few combo box selections, changes to dates, and an optional note.



4. Verify that you added the record correctly by selecting the employee for whom you added the action—in this case Steve Buchanan—in the Find combo box. Click the History tab to display the employees entries (see Figure 15.61). The History subform's snapshot Recordset must be refreshed to display the added record.

At this point in its development, frmHRActionEntry would be dangerous to release for use by data entry operators. For example, all fields in the main form are updatable. Thus, an operator could change the FirstName, LastName, and other values of the Employees table, as well as the EmployeeID for an existing HRActions record. You can set the Locked property to Yes for all controls linked to the Employees table, but you can't lock the



EmployeeID text box that's required to specify the EmployeeID of a new record. You can control the locked status of form controls by adding VBA event-handling code for the form's Before Insert and After Insert events.

**Figure 15.61**

After you refresh the History subform by selecting the employee in the Find combo box, the newly added record appears.

Type	Init. By	Scheduled	Appr. By	Effective	Rating	Salary	Rate	Bonus	% Comm
H	2	05/01/1992	1	05/01/1992		2,500			
S	5	10/17/1998	2	10/17/1998	9	4,000		1,000.00	
Q	5	04/26/2003	2	05/24/2003	9	6,500		5,000.00	

### TIP

You can make frmHRActionEntry safe for data entry operators by setting the Data Entry property of the form to Yes. Specifying Data Entry prevents operators from viewing existing records and only allows them to enter new records. In this case, you must change the Recordset Type property value of the subform to Dynaset, and set the AllowEdits, AllowDeletions, and AllowAdditions property values to Yes. Otherwise, the added record won't appear in the History subform.

## OVERRIDING THE FIELD PROPERTIES OF TABLES

Access uses the table's property values assigned to the fields as defaults. The form or subform inherits these properties from the table or query on which the form is based. You can override the inherited properties, except for the Validation Rule property, by assigning a different set of values in the Properties window for the control. Properties of controls bound to fields of tables or queries that are inherited from the table's field properties are shown in the following list:

- Format
- Decimal Places
- Status Bar Text
- Typeface characteristics (such as Font Name, Font Size, Font Bold, Font Italic, and Font Underline)
- Validation Rule
- Validation Text
- Default Value

Values of field properties that you override with properties in a form apply only when the data is displayed and edited with the form. You can establish validation rules for controls

bound to fields that differ from properties of the field established by the table, but you can only narrow the rule. The table-level validation rule for the content of the HRType field, for example, limits entries to the letters H, Q, Y, S, R, B, C, and T. The validation rule you establish in a form can't broaden the allowable entries; if you add F as a valid choice by editing the validation rule for the HRType field to `InStr("HQYSRBC TF", [HRType]) > 0`, you receive an error when you type F.

However, you can narrow the range of allowable entries by substituting `InStr("SQYB", [HRType]) > 0`. Notice that you can use expressions that refer to the field name in validation-rule expressions in forms; such expressions aren't permitted in field-level validation-rule expressions in Access 2003.

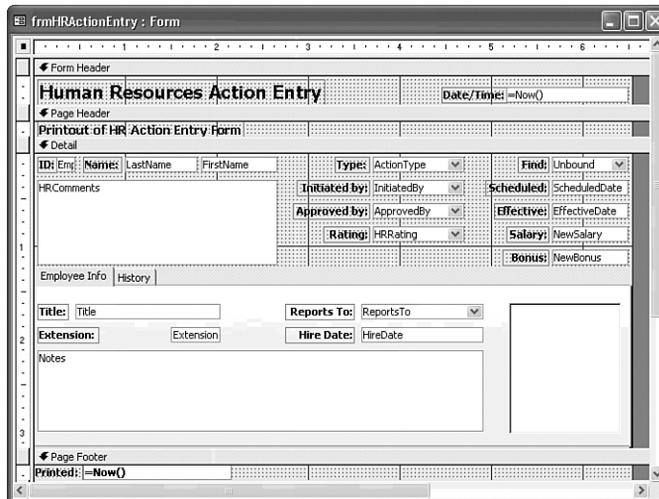
## ADDING PAGE HEADERS AND FOOTERS FOR PRINTING FORMS

Access lets you add a separate pair of sections, Page Header and Page Footer, that appear only when the form prints. You add both of these sections to the form at once by choosing **View, Page Header/Footer**. The following list shows the purposes of Page Headers and Footers:

- Page Header sections enable you to use a different title for the printed version. The depth of the Page Header can be adjusted to control the location where the Detail section of the form is printed on the page.
- Page Footer sections enable you to add dates and page numbers to the printed form.

Page Header and Page Footer sections appear only in the printed form, not when you display the form onscreen in Form view. Figure 15.62 shows the frmHRActionEntry form in Design view with Page Header and Page Footer sections added.

**Figure 15.62**  
Page Headers and Page Footers appear when you print the form, but not in Form view.



With the Display When (Format) property of the Properties window for the Form Header and Form Footer sections, you can control whether these sections appear in the printed form. In Figure 15.62, the Form Header duplicates the information in the Page Header (except for the Date/Time label and text box), so you might not want to print both. To control when a section of the form prints or is displayed, perform the following steps:

1. Double-click the title bar of whichever section of the form you want to change; this opens the related Properties window. (The Page Header and Page Footer sections don't have a Display When property; these sections appear only when printing.)
2. Click the Format tab if the formatting properties aren't already showing. Click to drop down the Display When list.
3. To display but not print this section in Form view, select Screen Only.
4. To print but not display this section, select Print Only.

## TROUBLESHOOTING



### ERROR MESSAGES ON COPIED CONTROLS

*A control copied to another form throws error messages whenever that control gets the focus.*

When you copy a control to a form that uses a data source different from the one used to create the original control, you need to change the Control Source property to correspond with the field the new control is to be bound to. Changing the Control Source property doesn't change the Status Bar Text, Validation Rule, or Validation Text properties for the new control source. You must enter the appropriate values manually.

## IN THE REAL WORLD—ACCESS WIZARDRY

Access 1.0 had only a few wizards; Access 2003 has 46, the same number as Access 2002. Microsoft defines a wizard as “A Microsoft Access tool that asks you questions and creates an object according to your questions.” The “Wizards, add-ins, and Builders in Microsoft Access 2003” online help topic lists 51 wizards, but four items in the list—Documentor, Macro-To-Module Converter, Subform/Subreport Field Linker, and Switchboard Manager—don't carry the Wizard suffix and are better classified as utilities.

Form-related wizards are the most numerous. The following nine wizards, listed in alphabetical order, assist you in creating custom forms:



- *AutoForm Wizard* automatically generates a form. Access 2002 added PivotTable and PivotChart types.
- *AutoFormat Wizard* applies a specific format to a form.
- *Chart Wizard* adds to a form a graph or chart bound to a table or query.

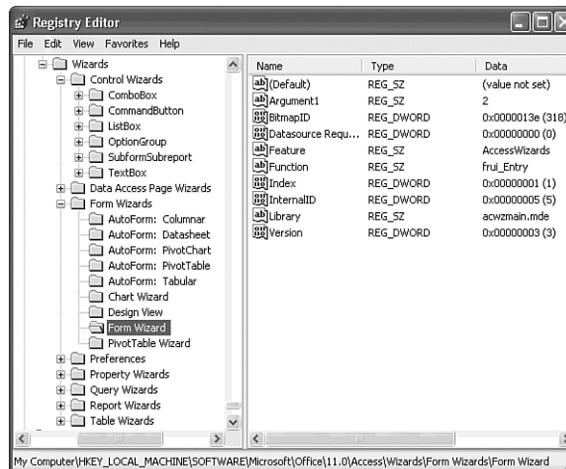


- *Combo Box Wizard* generates one of three classes of bound and unbound combo box controls on a form.
- *Command Button Wizard* adds a command button control.
- *Form Wizard* generates a new form. Access 2002 added the capability to generate a PivotTable or PivotChart form.
- *Option Group Wizard* adds a group of option buttons.
- *Subform/Subreport Field Linker* creates or alters links between a main form and subform.
- *Subform/Subreport Wizard* adds a new subform.

One of the reasons that Access 2003 has the largest wizard population of all Office 2003 applications is that Access is the most complex of the Office 2003 members—from both the user and developer standpoint. Access’s complexity, compared with Word, Excel, PowerPoint, and Outlook, undoubtedly is the reason that Microsoft doesn’t include Access 2003 in the Standard or Small Business editions of Office 2003. The omission of Access from the Small Business edition is surprising, because establishing and maintaining databases is crucial for almost every enterprise, regardless of size.

Wizards are classified as Access add-ins, which also include Builders, menu add-ins, and a new class of Component Object Model (COM) add-ins. The standard set of wizards and builders that come with Access appear in the `HKEY_LOCAL_MACHINE\Software\Microsoft\Office\11.0\Access\Wizards` key of the Registry. Figure 15.63 shows the top-level Registry keys for the Control and Form Wizards used in this and the preceding chapter. Microsoft classifies Access Builders as Property Wizards in the Registry.

**Figure 15.63**  
The Registry includes keys for each Access wizard and Builder.



Most of the wizards are contained in the Acwzmain.mde file in your ...\\Office11 folder; the Acwztool.mde Advanced Wizards file includes the Add-In Manager and some Builders; and Acwzlib.mde holds the Import/Export Wizards. You can open the Acwz... .mde files in Access 2003, but you can't make changes to objects. It's unfortunate that Microsoft uses the .mde format to prevent viewing the wizard VBA code; Access wizards are excellent examples of VBA power programming in action.