Designing Forms for Efficient and Accurate Data Entry

3

IN THIS CHAPTER

Data entry is one of those tasks that I describe as "dangerous" because it's a chore that's both tedious and important. It's tedious because entering dozens or hundreds of records is no one's idea of fun. It's important because the data must be entered accurately; otherwise, any analysis of the data becomes at best misleading and at worst just plain wrong. The danger, then, lies in the fact that data entry is prone to errors but can't afford to have any.

As a forms designer, you can help to reduce this danger by setting up your forms so that data entry is both as efficient as possible and as accurate as possible. In some cases you can achieve both goals with a single technique. For example, asking someone to type a customer name manually is both slow and prone to misspellings. However, suppose you already have a Customers table with a `CustomerName` field. If you relate the current table with the Customers table (using, say, a common `CustomerID` field), Access adds the `CustomerName` field to the current form using a drop-down list that contains all the customers. This makes data entry more efficient (the users just select a name from the list instead of typing it) and more accurate (the users can't misspell the customer name).

This chapter introduces you to several techniques that serve to either make data entry less of a chore, or to reduce or eliminate data entry errors (or both).

Preventing Errors by Validating Data

If, as the cooks say, a recipe is only as good as its ingredients, a database is only as good as its data. Viewing, summarizing, and analyzing the data are meaningless if the table you're working with contains erroneous or improper data. For basic data errors (for example, entering the wrong date or transposing a number's digits), there's not a lot you can do other than exhorting yourself or the people who use your forms to enter data carefully. Fortunately, you have a bit more control when it comes to preventing improper data entry. By "improper," I mean data that falls in either of the following categories:

- Data that is the wrong type. For example, entering a text string in a cell that requires a number.
- Data that falls outside an allowable range. For example, entering 200 in a cell that requires a number between 1 and 100.

NOTE

To stress the importance of data entry, consider the story told to me by computer book author Greg Perry. Greg used to work for a large Fortune 500 company, and he says that the company made its data entry clerks enter all data *twice*: One clerk would enter the data in a file and then, when finished, another clerk would enter the same data. Then a comparison would be run to find exceptions where data didn't match, and that would then be reconciled. The lesson, he says, is that to the company, accuracy was far less costly than paying its employees to do the same job twice.

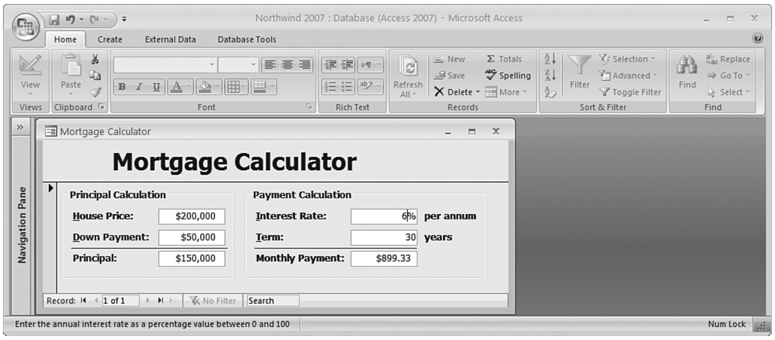
The next few sections show you several techniques that can help you reduce these types of errors.

Helping Users with Text Prompts

You can prevent improper entries to a certain extent by adding text that provides details on what is allowable inside a particular cell. You have two choices:

- Add status bar text. This is a string that appears in the Access status bar when users enter the field. You specify this text by opening the field's property sheet, displaying the Other tab, and then entering the string in the `Status Bar Text` property.
- Add a label. Place a `Label` control near the field and use it to enter text that describes the field's data requirements or shortcut keys. For example, if the field requires a date, the label might say `Press Ctrl+; to enter today's date.`

For example, Figure 3.1 shows the Mortgage Calculator form. Notice the labels added beside the Interest Rate and Term text boxes that specify to the users that they must enter the interest rate per annum and the term in years. Note, too, the status bar text that appears when the users enter the Interest Rate field.



Mortgage Calculator

Principal Calculation

House Price:	$200,000
Down Payment:	$50,000
Principal:	$150,000

Payment Calculation

Interest Rate:	6%	per annum
Term:	30	years
Monthly Payment:	$899.33	

Enter the annual interest rate as a percentage value between 0 and 100

Figure 3.1
Use form labels and status bar text to give the users text prompts about the data they must enter.

Preventing Errors with Data Validation Expressions

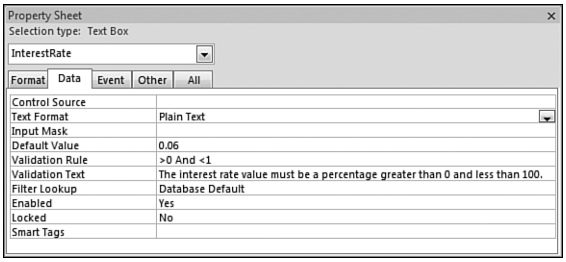
The problem with text prompts is they require other people to both read *and* act on the text. The better solution for preventing data entry errors is the Access data validation feature. With data validation, you create *rules* that specify exactly what kind of data can be entered and in what range that data can fall. You can also specify pop-up input messages that appear when a cell is selected, as well as error messages that appear when data is entered improperly.

Follow these steps to define the settings for a data validation rule:

1. Display the property sheet of the field to which you want to apply the data validation rule.
2. Click the Data tab.
3. Click inside the `Validation Rule` property.
4. Enter a formula that specifies the validation criteria. You can either enter the formula directly into the property box, or you can click the ellipsis (…) button and enter the formula using the Expression Builder.
5. If you want a dialog box to appear when the users enter invalid data, click inside the `Validation Text` property and then specify the message that appears.
6. Close the property sheet to apply the data validation rule.

For example, suppose you want the users to enter an interest rate. This quantity should be positive, of course, but it should also be less than 1. (That is, you want users to enter 6% as 0.06 instead of 6.) Figure 3.2 shows the property sheet for a field named `InterestRate` that meets these criteria by defining the following expression in the `Validation Rule` property:

```
>0 And <1
```

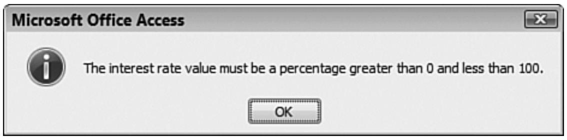


Property Sheet
Selection type: Text Box
InterestRate

Control Source	
Text Format	Plain Text
Input Mask	
Default Value	0.06
Validation Rule	>0 And <1
Validation Text	The interest rate value must be a percentage greater than 0 and less than 100.
Filter Lookup	Database Default
Enabled	Yes
Locked	No
Smart Tags	

Figure 3.2 Use the `Validation Rule` property to enter a data validation expression for a field.

Figure 3.2 also shows a string in the `Validation Text` property. If the users enter invalid data (that is, any value for which the `Validation Rule` expression returns False), the `Validation Text` appears in a dialog box, as shown in Figure 3.3.



Microsoft Office Access

The interest rate value must be a percentage greater than 0 and less than 100.

Figure 3.3 If the users enter invalid data in the field, Access displays a dialog box such as this one, which uses the string entered into the `Validation Text` property.

Using Input Masks for Consistent and Accurate Data Entry

One of the major headaches that database administrators have to deal with is data entered in an inconsistent way. For example, consider the following phone numbers:

```
(123)555-6783
(123) 555-6783
(123)5556783
123555-6783
1235556783
```

These sorts of inconsistencies might appear trivial, but they can cause all kinds of problems, from other users misreading the data to improper sorting to difficulties analyzing or querying the data. And it isn't just phone numbers that cause these kinds of problems. You also see them with Social Security numbers, ZIP codes, dates, times, account numbers, and more.

One way to avoid such inconsistencies is to add a label or status bar message that specifies the correct format to use. As with data validation, however, these prompts are not guaranteed to work every time (or even most of the time).

A better solution is to apply an *input mask* to the field. An input mask is a kind of template that shows the users how to enter the data and prevents them from entering incorrect characters (such as a letter where a number is required). For example, here's an input mask for a phone number:

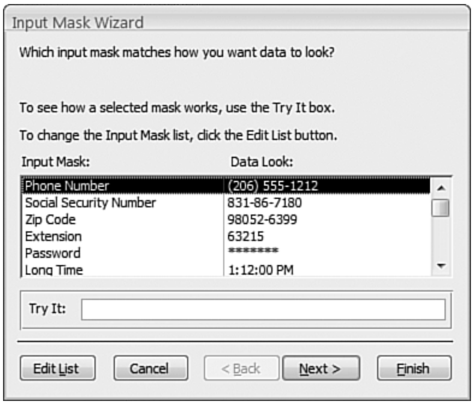
(___)___-____

Each underscore (_) acts as a placeholder for (in this case) a digit, and the parentheses and dash appear automatically as the user enters the number.

Using the Input Mask Wizard

The easiest way to create an input mask is to use the Input Mask Wizard. Here are the steps to follow:

1. Display the property sheet of the field to which you want to apply the input data.
2. Click the Data tab.
3. Click inside the `Input Mask` property.
4. Click the ellipsis (…) button to start the Input Mask Wizard, shown in Figure 3.4.



Which input mask matches how you want data to look?

To see how a selected mask works, use the Try It box.

To change the Input Mask list, click the Edit List button.

Input Mask:	Data Look:
Phone Number	(206) 555-1212
Social Security Number	831-86-7180
Zip Code	98052-6399
Extension	63215
Password	*******
Long Time	1:12:00 PM

Figure 3.4
Use the Input Mask Wizard to choose a predefined input mask or to create your own input mask.

5. In the Input Mask list, click the input mask you want (or that's close to what you want) and then click Next.
6. Use the Input Mask box to make changes to the mask (see "Creating a Custom Input Mask Expression," next, for the specifics of which symbols to use); use the Placeholder Character list to choose the character you want to appear in the input mask as a placeholder; click Next.
7. Click the option that matches how you want the field data stored in the table (click Next after you've made your choice):
   - **With the Symbols in the Mask**—Click this option if you want the extra symbols (such as the parentheses and dash in a phone number mask) stored along with the data.
   - **Without the Symbols in the Mask**—Click this option to store only the data.
8. Click Finish.

Creating a Custom Input Mask Expression

If your data doesn't fit any of the predefined input masks, you need to create a custom mask that suits your needs. You do this by creating an expression that consists of three kinds of characters:

- **Data placeholders**—These characters are replaced by the actual data typed by the users. The different placeholders specify the type of character the users must enter (such as a digit or letter) and whether the character is optional or required.
- **Modifiers**—These characters aren't displayed in the mask; instead, they're used to modify the mask in some way (such as converting all the entered characters to lowercase).
- **Literals**—These extra characters appear in the mask the same as you enter them in the expression. For example, you might use parentheses as literals to surround the area code portion of a phone number.

Table 3.1 lists the data placeholders you can use to build your input mask expressions.

Table 3.1 Data Placeholders to Use for Custom Input Masks

Placeholder	Data Type	Description
`0`	Digit (0–3)	The character is required; the users are not allowed to include a plus sign (+) or minus sign (–).
`3`	Digit or space	The character is optional; the users are not allowed to include a plus sign (+) or minus sign (–).
`#`	Digit or space	The character is optional; the users are allowed to include a plus sign (+) or minus sign (–).
`L`	Letter (*a*–*z* or *A*–*Z*)	The character is required.
`?`	Letter (*a*–*z* or *A*–*Z*)	The character is optional.
`a`	Letter or digit	The character is required.
`A`	Letter or digit	The character is optional.
`&`	Any character or space	The character is required.
`C`	Any character or space	The character is optional.

Table 3.2 lists the modifiers and literals you can use to build your input mask expressions.

Table 3.2 Modifiers and Literals to Use for Custom Input Masks

Modifier	Description
`\`	Displays the following character as a literal; for example, `\(` is displayed as `(`.
`"text"`	Displays the string *`text`* as a literal; for example, `"MB"` is displayed as `MB`.
`.`	Decimal separator.

Modifier	Description
`,`	Thousands separator.
`: ; - /`	Date and time separators.
`<`	Displays all the following letters as lowercase.
`>`	Displays all the following letters as uppercase.
`!`	Displays the input mask from right to left when you have optional data placeholders on the left.
`Password`	Displays the characters as asterisks so that other people can't read the data.

You can enter your input mask expressions directly into the `Input Mask` property, or you can modify a predefined input mask using the Input Mask Wizard.

For example, suppose your company uses account numbers that consist of four uppercase letters and four digits, with a dash (-) in between. Here's an input mask suitable for entering such numbers:

```
>aaaa\-0000
```

Note, too, that input masks can contain up to three sections separated by semicolons (;):

`first``;`*`second`*`;`*`third`*

`first`—This section holds the input mask expression.

`second`—This optional section specifies whether Access stores the literals in the table when you enter data. Use `0` to include the literals; use `1` (or nothing) to store only the data.

`third`—This optional section specifies the placeholder character. The default is the underscore (_).

For example, here's an input mask for a ZIP code that stores the dash separator and displays dots (.) as placeholders:

```
00000\-3333;0;.
```

Using Controls to Limit Data Entry Choices

Data entry always trips over two unfortunate facts of life: Humans are fallible creatures, and typing is an error-prone activity. Expert data entry operators can't achieve 100% accuracy (although some come remarkably close), and the rest of us can only hope for the best. In short, if your form relies on other people (or yourself, for that matter) typing in field values, it's death-and-taxes certain that your table will end up with errors.

It stands to reason, then, that you can greatly reduce the number of errors by greatly reducing the amount of typing. The best way to do that is by taking advantage of controls to generate field values automatically. Here are some examples:

- If you have a `Yes/No` field that uses a text box, the users must enter the unintuitive values `-1` (for `Yes`) and `0` (for `No`). A more intuitive approach is to use a check box (or toggle button) that the users either activate (for `Yes`) or clear (for `No`).
- Suppose you have a field that can take only one of a small set of values (say, two to five values). For example, an invoice form might offer the users three choices for freight or four choices for credit cards. Again, instead of having the users type the freight choice or credit card name, you can populate the form with option buttons representing the choices.
- Suppose you have a field that can take one of a relatively large set of values (more than five). For example, the field might hold a customer name or a product name. Instead of making the users look up (time-consuming) and then type (inaccurate) the value, it's both faster and more accurate to place all the possible values in a drop-down list.

The rest of this chapter shows you how to use check boxes, toggle buttons, option buttons, lists, and other controls to build faster and more accurate forms. In each case, the idea is to move the users away from typing values and toward selecting them via a familiar and easily used control.

TIP

Another way to ensure data accuracy is to set up a field with a default value that Access enters into the field automatically when the user starts a new record. This can be a literal value such as 0 for a numeric field, or a formula such as `=Date()` for a date/time field. In the control's Property Sheet, display the Data tab and type the value in the Default Value property.

CAUTION

This is as good a place as any to warn you against what I call "form complacency." This is the attitude (which I've succumbed to myself on more than one occasion) which assumes that once *you* are happy with your form's layout, format, and data validation, then other people will automatically be happy with those things, too. Probably not! Other people will almost certainly approach the form differently, and they'll almost always have trouble figuring out how it works and what's expected of them. In other words, *always* "test drive" your form by letting other users take their best shots at it. It only takes a little extra time, and the suggested changes they come up with (and there *will* be suggestions, believe me) will save you time in the long run.

Working with `Yes/No` Fields

You use `Yes/No` fields in tables when you have a quantity that you can represent in one of two states: on (`Yes`, `True`, or `-1`) or off (`No`, `False`, or `0`).

When you create a `Yes/No` field in the table Design view, the `Display Control` property (it's in the Lookup tab) defaults to `Check Box`. This means that when you add a `Yes/No` field to a form, Access automatically represents the field with a check box control (along with a label that displays the name of the field or the field's `Caption` property). However, it's possible that the `Display Control` property has been set to `Text Box`, either by design or by accident. As I mentioned earlier, you want to avoid users having to enter `-1` or `0` into a text box, so you should never use a text box for a `Yes/No` field on your forms. Instead, you have two choices:

- If you have access to the table's design, change the `Yes/No` field's `Display Control` property to `Check Box`. After you've done that, return to the form, delete the `Yes/No` field's text box and label (if they're already on the form), and then add the field back to the form to get the check box version.
- If you can't change the table design, use a check box or toggle button control bound to the `Yes/No` field. The next two sections show you how to do this.

Using Check Boxes

Here are the steps to follow to insert a check box and bind it to a `Yes/No` field:

1. In the Design tab's Controls group, click the Check Box button.
2. Draw the check box on the form.
3. Edit the text of the label control that Access adds to the right of the check box. (For clarity, it's best to use the name of the `Yes/No` field.)
4. Click the check box and then choose Design, Property Sheet to open the Property Sheet pane.
5. In the Data tab, use the `Control Source` property to choose the name of the `Yes/No` field you want bound to the check box.
6. In the `Default Value` property, enter the initial value for new records: `Yes`, `True`, or `-1`; or `No`, `False`, or `0`.

CAUTION

Many form designers like to use an option group as a way of "framing" a number of related controls. This is often a good idea (I discuss it in more detail in Chapter 4, "Designing Forms for Business Use"), but you need to be careful: If you add the option group and then insert the check boxes within the group, Access treats the check boxes as mutually exclusive options. That is, the users can activate only one check box at a time. To avoid this situation, add the check boxes to the frame first and then draw the option group around them.

It's worth pointing out here that check boxes (and toggle buttons, discussed next) can insert only one of two values into a field: `-1` or `0`. You can't use a check box for other two-state choices, such as `"male"` and `"female"` or `"Pepsi"` and `"Coke"`. For fields that can take only one of two values other than `0` and `-1`, use option buttons instead (as described later in this chapter).

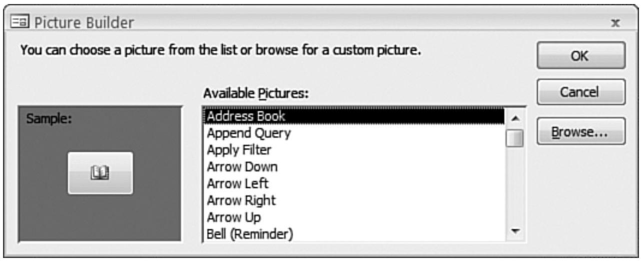
Using Toggle Buttons

A toggle button is a cross between a check box and a command button: Click it once, and the button stays pressed; click it again, and the button returns to its normal state. The button can display either a caption or a picture. Here are the steps to follow to insert a toggle button and bind it to a `Yes/No` field:

1. In the Design tab's Controls group, click Toggle Button.
2. Draw the toggle button on the form.
3. Choose Design, Property Sheet to open the Property Sheet pane.
4. In the Format tab, you have two choices that determine what appears on the face of the button:
   - `Caption`—Use this property to specify text that appears on the face of the button. (For clarity, it's best to use the name of the `Yes/No` field.)
   - `Picture`—Use this property to specify an image that appears on the button face. Click the ellipsis button (…) to display the Picture Builder dialog box, shown in Figure 3.5. Either use the Available Pictures list to click an image or click Browse to choose an image from the Select Picture dialog box (although note that Access can only use BMP or icon files).

CAUTION

If you want to use a custom picture, bear in mind that if the image is larger than the toggle button, Access won't shrink the image to fit inside the button—it just centers the image in the button and displays as much as will fit. Therefore, always choose a bitmap or icon that's the same size or smaller than the toggle button.


Picture Builder

You can choose a picture from the list or browse for a custom picture.

Sample:

Available Pictures:
Address Book
Append Query
Apply Filter
Arrow Down
Arrow Left
Arrow Right
Arrow Up
Bell (Reminder)

Figure 3.5
Use the Picture Builder dialog box to choose an image to appear on the face of the toggle button.

5. In the Data tab, use the `Control Source` property to choose the name of the `Yes/No` field you want bound to the toggle button.
6. In the `Default Value` property, enter the initial value for new records. For the "pressed" state, use `Yes`, `True`, or `-1`; for the "unpressed" state, use `No`, `False`, or `0`.

Using Option Buttons to Present a Limited Number of Choices

Option buttons are a good choice if the underlying field accepts only a limited number of possible numbers: at least two but no more than about five or six. (If you have more possible values, use a list box or combo box, discussed later in this chapter.)

How does having multiple option buttons on a form enable you to store a single value in a field? There are two components to consider:

- **The option buttons**—You assign each option button a value from among the list of possible values that the field can take.

NOTE Option button values must be numeric. Therefore, you can use option groups and option buttons only with numeric fields.

- **The option group**—This is a separate control that you use to organize the option buttons. That is, if you insert multiple option buttons inside a group, Access allows the users to activate only one of the options at a time. (You can also use check boxes or toggle buttons, but option buttons are best because most users are familiar with them and know how to operate them.)

The option group is bound to the field in the underlying table. Therefore, when you activate an option button, the value assigned to that button is stored in the field. This form of data entry brings many advantages to the table (literally!):

- **It's quick.** The users don't have to look up the possible values elsewhere.
- **It's accurate.** The field value is stored "behind the scenes," so the users can't enter the wrong value.
- **It's intuitive.** The option button captions can be as long as you like (within reason), so you can provide users with a helpful description or title for each option.
- **It's familiar.** All Windows users know how to operate option buttons, so no extra training is required.

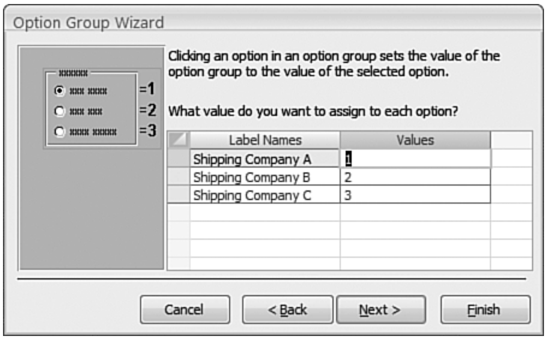
The next two sections show you how to create option buttons using a wizard and by hand.

Running the Option Group Wizard

The easiest way to create an option group and its associated option buttons is to use the Option Group Wizard, as described in the following steps:

1. In the Design tab's Controls group, make sure the Control Wizards button is activated and then click the Option Group button.
2. Draw the option group on the form. Access launches the Option Group Wizard.

3. For each option button you want, type the label in the Label Names list and press Tab. When you're done, click Next.
4. To select a default choice (the option that Access activates automatically when the user starts a new record), leave the Yes, The Default Choice Is option activated and then choose the option label from the list. Click Next.
5. Use the Values column to assign a numeric value for each option, as shown in Figure 3.6. Note that each value must be unique. Click Next when you're done.



Clicking an option in an option group sets the value of the option group to the value of the selected option.

What value do you want to assign to each option?

Label Names	Values
Shipping Company A	1
Shipping Company B	2
Shipping Company C	3

Figure 3.6
Use this Option Group Wizard dialog box to assign a unique numeric value to each option.

6. Specify where you want the option group value stored (click Next when you're done):
   - **Save the Value for Later Use**—Click this option to have Access save the option group value. This is mostly used by VBA programmers—the current value of the option group is stored in the `Frame` object's `Value` property.
   - **Store the Value in This Field**—Click this option and then select a field from the list to have Access store the option group value in the field.
7. Click the type of control you want to use in the option group: Option Buttons, Check Boxes, or Toggle Buttons. You can also select the special effect used by the option group border (Etched, Flat, and so on). Click Next to continue.
8. Edit the option group caption (the text that the users see along the top border of the option group frame; use the field name or something similar) and then click Finish to complete the wizard.

TIP

If you already have an "unframed" option button on your form, you can still insert it into an option group. Select the button, cut it to the Clipboard, select the option group (by clicking its frame), and paste. Access adds the button to the option group.

Creating an Option Group By Hand

If you'd rather create the option group yourself, here are the steps to follow:

1. In the Design tab's Controls group, make sure the Control Wizards button is deactivated and then click the Option Group button.
2. Draw the option group on the form.
3. In the Design tab's Controls group, click Option Button.
4. Draw the option button inside the option group.
5. Choose Design, Property Sheet to display the option button's property sheet.
6. In the Data tab, use the `Option Value` property to specify the numeric value associated with the option.
7. Use the drop-down list to choose the label associated with the option button. (It's the control that is one number greater than the option button. For example, if the option button name assigned by Access is `Option10`, the associated label will be named `Label11`.)
8. In the Format tab, use the `Caption` property to specify text that appears alongside the option button.
9. Repeat steps 3–8 for the other option buttons you want to add to the option group.
10. Use the drop-down list to choose the option group (it's named Frame*n*, where *n* means it was the *n*th control added to the form).
11. In the Data tab, use the `Control Source` property to choose the field in which you want the value of the selected option button stored.
12. If you want one of the option buttons to be activated when the users start a new record, use the `Default Value` property to enter the value of the corresponding option button.
13. Close the property sheet.

CASE STUDY

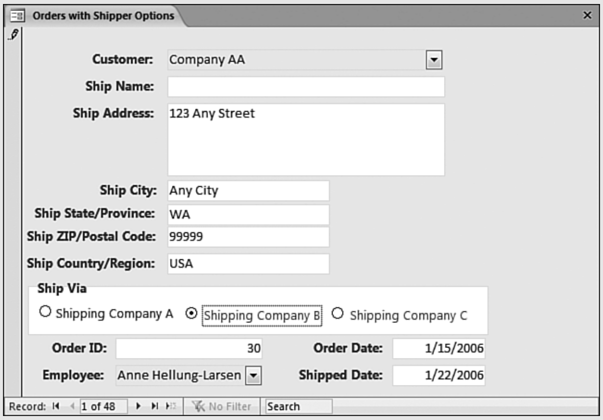
Using an Option Group to Select the Shipper

In the Northwind 2007 database, the Orders table has a `Shipper ID` field that specifies which shipping company to use. There are three shipping companies that the users can select: Shipping Company A, Shipping Company B, and Shipping Company C. (The person at Microsoft who put together the Northwind 2007 database was singularly uncreative when it came to names.) Option buttons can take only numeric values, so you can't use them to assign a text value such as `"Shipping Company A"` to the `Shipper ID` field. That's not a problem because the `Shipper ID` field is designed to store a number: `1` for Shipping Company A, `2` for Shipping Company B, and `3` for Shipping Company C. These numbers correspond to the `ID` field in the Shippers table. The Shipper and Orders tables have a one-to-many relation based on the `ID` and `Shipper ID` fields.

A field that takes one of three numeric choices is perfect for an option group. You set things up as follows:

- Create an option group and bind it to the `Shipper ID` field.
- Add three option buttons for Shipping Company A, Shipping Company B, and Shipping Company C, and assign them the values `1`, `2`, and `3`, respectively.

Figure 3.7 shows the resulting option group in the form.



Orders with Shipper Options

Customer:	Company AA
Ship Name:	
Ship Address:	123 Any Street
Ship City:	Any City
Ship State/Province:	WA
Ship ZIP/Postal Code:	99999
Ship Country/Region:	USA

Ship Via: ( ) Shipping Company A [✓ Shipping Company B] ( ) Shipping Company C

Order ID:	30	Order Date:	1/15/2006
Employee:	Anne Hellung-Larsen	Shipped Date:	1/22/2006

Figure 3.7
This form uses an option group to choose the shipping method for each order.

Using Lists to Present a Large Number of Choices

Option buttons have three main disadvantages:

- If a field can take more than about five or six values, option buttons become too unwieldy and confusing for the users.
- Option buttons can't work with non-numeric values.
- Users can't enter unique values. This is normally a good thing, but in some instances you might want to give the users the flexibility to choose either a predefined value or to enter a different value.

To solve all these problems, Access offers two different list controls that enable you to present the users with a list of choices:

- A list box presents a list of choices. These choices are *static*, meaning that users can't enter any different values.
- A combo box enables users to either select a value from a drop-down list or (optionally) enter a different value using the associated text box.

NOTE

Another consideration you need to bear in mind when deciding between a list box and a combo box is the size of each control on the form. A list box is usually large enough to show at least three or four items in the list, whereas a combo box always shows only a single item (the users click the list to choose another). Therefore, the list box always takes up quite a bit more room than the combo box, so keep that in mind when designing your form. If you don't have much room, but you don't want the users to be able to add different values to the field, you'll see later that it's possible to restrict the combo box to just the values in the list.

In both cases, the item the users choose from the list (or the item the users enter in the combo box) is the value that is stored in the bound field. This means that you can use list and combo boxes for any type of value, including numeric, string, and date values.

It's important to note that Access defaults to a combo box when you add to the form a field that is used as part of a relationship with another table. Specifically, if the relationship is one-to-many and the current table is the "many" side, adding the field that corresponds to the common field on the "one" side creates a list that contains all the values from that field.

For example, the Products table has a one-to-many relationship with the Order Details table via the `common ID` and `Product ID` fields, respectively. If you're putting together a form based on the Order Details table and you add the `Product ID` field, Access creates a combo box list and populates it with the values from the Products table's `Product Name` field. Why `Product Name` and not `Product ID`? The reason is that in the design for the Order Details table, the `Product ID` field's `Row Source` property (in the Lookup tab) specifies an SQL statement that selects the `Product Name` field from the Products table:

```
SELECT ID, [Product Name] FROM Products ORDER BY [Product Name]
```

The next few sections show you various ways to work with both controls.

Starting the List Box or Combo Box Wizard

The List Box Wizard and Combo Box Wizard make it easy to create a bound list control. Here are the steps to follow to get started with these wizards:

1. In the Design tab's Controls group, make sure the Control Wizards button is activated.
2. Click either Combo Box or List Box.
3. Draw the box on the form. Access starts either the List Box Wizard or the Combo Box Wizard.

These wizards work identically, but the steps you take vary dramatically depending on which option you choose in the initial dialog box. The next three sections take you through the details of each option.

Getting List Values from a Table or Query Field

The most common list scenario is to populate the list box or combo box with values from a field in a specified table or query. For example, if you're putting together an orders form, you'll probably want to include a list that contains all the customer names, so you'll populate the list with the values from the Customers table's `Company` field.

The following steps show you how to continue with the List Box or Combo Box Wizard to populate a list with values from a table or query field:

1. In the first wizard dialog box, click the I Want the List Box to Look Up the Values in a Table or Query option and then click Next.
2. Click the table or query that contains the field you want to use for the list and then click Next.
3. In the Available Fields list, select the field you want to use and then click > to add it to the Selected Fields list. Click Next.
4. If you want the list sorted, use the drop-down list to choose the field you selected, click the Ascending (or Descending) toggle button, and then click Next.
5. Click and drag the right edge of the column header to set the width of the list column and then click Next.
6. To create a bound list box or combo box, select the Store That Value in This Field option, choose the field you want to use from the drop-down list, and then click Next.
7. In the final wizard dialog box, use the text box to edit the label text that appears above the list and then click Finish.

Specifying Custom List Values

If the items you want to appear in your list don't exist in another table or query, you need to specify them by hand. Here are the steps to follow to continue with the List Box or Combo Box Wizard and populate a list with custom values:

1. In the first wizard dialog box, click the I Will Type in the Values That I Want option and then click Next.
2. For each value you want to add, type the item text and press Tab. Click Next when you're done.
3. To create a bound list box or combo box, select the Store That Value in This Field option, choose the field you want to use from the drop-down list, and then click Next.
4. In the final wizard dialog box, use the text box to edit the label text that appears above the list and then click Finish.

Getting List Values from the Current Table

Sometimes the values you want in your list already exist in the form's underlying table or query. For example, if your form uses the Customers table, you might want to set up a list for the `Job Title` field and use the unique values in that to populate the list. (This example illustrates when you might want to use a combo box, because a new customer contact could have a title other than the ones in the list.) Note, however, that the list you create using this method will always be an *unbound* control.

The following steps show you how to continue with the List Box or Combo Box Wizard to populate a list with values from a field in the form's current data source:

1. In the first wizard dialog box, click the Find a Record on My Form Based on the Value I Selected in My Combo Box option and then click Next.
2. In the Available Fields list, select the field you want to use and then click > to add it to the Selected Fields list. Click Next.
3. Click and drag the right edge of the column header to set the width of the list column and then click Next.
4. In the final wizard dialog box, use the text box to edit the label text that appears above the list and then click Finish.

Creating a Multiple-Column List

Sometimes displaying a single column of values in a list might not be enough. For example, if you're working with data from the Northwind 2007 Products table, displaying just the `Product Name` field might not give the users enough information. Instead, you might also want to show the users the corresponding Category or Supplier value (using an inner join query for the latter) for each product.

➔ To learn about inner joins, **see** "Establishing Table Relationships," **p. 267**. (Chapter 12)

You can do this by adding one or more columns to the list and then specifying which of those columns contains the value you want to store in your form's bound field. Here are the steps to follow:

1. Draw a list box or combo box on the form to launch the List Box or Combo Box Wizard.
2. In the first wizard dialog box, select the I Want the List Box to Look Up the Values in a Table or Query option and then click Next. (Note that you can also display multiple columns using the Find a Record on My Form Based on the Value I Selected in My Combo Box option.)
3. Select the table or query that contains the field you want to use for the list and then click Next.
4. In the Available Fields list, for each field you want to display in the list, select the field and then click > to add it to the Selected Fields list. Click Next.
5. You sort the list on multiple fields by using separate drop-down lists to choose each field and its sort order. Click Next.

6. Click and drag the right edge of each column header to set the width of the list columns. Note, too, that you can also change the column order by clicking and dragging the column headers left or right. Click Next.
7. To create a bound list box or combo box, select the Store That Value in This Field option, choose the field you want to use from the drop-down list, and then click Next.
8. In the final wizard dialog box, use the text box to edit the label text that appears above the list and then click Finish.

Figure 3.8 shows a form that uses a two-column combo box to display both the `Product Name` field and the `Category` field from the Products table.

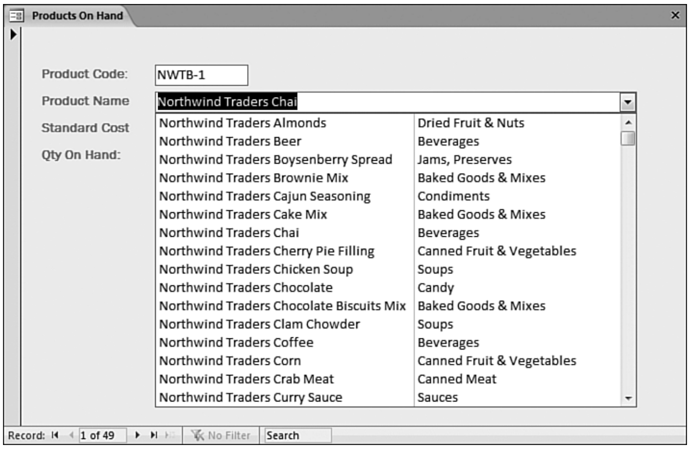


Figure 3.8
This combo box uses multiple columns to display both the `Product Name` field and the corresponding `Category` field.

Products On Hand

Product Code: NWTB-1
Product Name: Northwind Traders Chai
Standard Cost
Qty On Hand:

Northwind Traders Almonds	Dried Fruit & Nuts
Northwind Traders Beer	Beverages
Northwind Traders Boysenberry Spread	Jams, Preserves
Northwind Traders Brownie Mix	Baked Goods & Mixes
Northwind Traders Cajun Seasoning	Condiments
Northwind Traders Cake Mix	Baked Goods & Mixes
Northwind Traders Chai	Beverages
Northwind Traders Cherry Pie Filling	Canned Fruit & Vegetables
Northwind Traders Chicken Soup	Soups
Northwind Traders Chocolate	Candy
Northwind Traders Chocolate Biscuits Mix	Baked Goods & Mixes
Northwind Traders Clam Chowder	Soups
Northwind Traders Coffee	Beverages
Northwind Traders Corn	Canned Fruit & Vegetables
Northwind Traders Crab Meat	Canned Meat
Northwind Traders Curry Sauce	Sauces

Record: 1 of 49 No Filter Search

Modifying List Box and Combo Box Properties

If you want a bit more control over the list layout and data, you need to tweak the control properties. Here's a list of the properties to work with:

- `Control Source` **(Data tab)**—The field in which the selected list item will be stored.
- `Row Source Type` **(Data tab)**—Choose `Table/Query` for values that come from a table or query field; choose `Value List` for values that you enter by hand; choose `Field List` to populate the list with field names from a table or query.
- `Row Source` **(Data tab)**—This value depends on the `Row Source Type` value:
  - `Table/Query`—Enter an SQL `SELECT` statement that specifies the field you want to use to populate the list (along with any criteria you want to use). Alternatively, click the ellipsis button (…) and use the Query Builder to specify the table, field, and criteria. When you close the Query Builder, Access converts your selections into an SQL `SELECT` statement.
  - `Value List`—Enter the values with which you want to populate the list, separated by semicolons.

- `Field List`—Enter the name of the table or query that contains the field names with which you want to populate the list.

NOTE

If you want to display a multiple-column list, specify each field that you want to include in the list after the `SELECT` verb in the SQL statement, as in this example:

```
SELECT CategoryName, Description FROM Categories;
```

Alternatively, use the Query Builder to add each field to the criteria grid.

- `Bound Column` **(Data tab)**—If `Row Source Type` is `Table/Query` and the `Row Source` `SELECT` statement specifies only a single field, the `Bound Column` value should always be `1`. If the `Row Source` proeprty specifies two or more fields (for a multiple-column list), set `Bound Column` to the number of the field that contains the value you want to store in the current table (`1` is the first field, `2` is the second field, and so on).
- `Limit To List` **(Data tab)**—This is a combo box-only property. When the value is `Yes`, the users can only select values from the list; when the value is `No`, the users can enter new values.
- `Column Count` **(Format tab)**—The number of columns in the list box.
- `Column Heads` **(Format tab)**—If this property is `Yes`, the list columns are displayed with headers, whereby each header contains the name of the field.
- `Column Widths` **(Format tab)**—The width, in inches, of each column, separated by semicolons.
- `List Rows` **(Format tab)**—This is a combo box-only property, and it specifies the number of items the users see when they click the list.
- `Multi Select` **(Other tab)**—This is a list box-only property. If this property is `None`, users can select only one item at a time; if this value is `Simple`, users can select multiple items by clicking them; if this value is `Extended`, users must hold down the Ctrl key to select multiple items (or hold down Shift to select multiple items that appear consecutively in the list).

Entering Data with ActiveX Controls

The controls you see in the Design tab's Controls group will likely satisfy most of your form needs. However, you might have noticed that some controls that are commonly seen in Windows dialog boxes aren't available in the Controls group. For example, many Windows programs use spin buttons for entering numeric values, whereas others use a "calendar" control to enable users to choose dates via the mouse.

These and many other controls are available on your system as separate components that either come with the default Windows installation or are added to the system when you install Microsoft Office. (Other programs might also add their own controls to the system.) There are dozens of these so-called ActiveX controls, although only a few are suitable to be used on an Access form. The next three sections show you how to use three of them: a spin button, a scrollbar, and a calendar control.

Entering Numbers Using a Spin Button

A spin button comes with up and down arrows that the users can click to increment or decrement a value. Most spin buttons have a text box control beside them to show the current value. In most cases, the text box also gives users the choice of entering the number directly or selecting the number by using the spin button arrows. However, as you'll see, to use a spin button on an Access form, you can't make the text box editable, so the users must use the spin button arrows. Therefore, this control is useful only for fields that require relatively small numbers (to minimize the amount of clicking the users must do to get the required value).

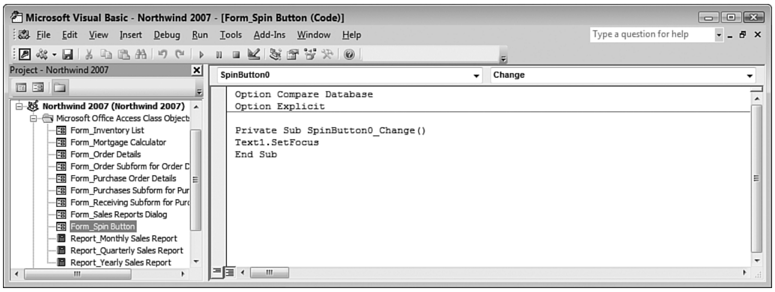
Here are the steps required to add a spin button and companion text box to a form:

1. In the Design tab's Controls group, click the Insert ActiveX Control button. Access displays the Insert ActiveX Control dialog box.
2. In the Select an ActiveX Control list, click Microsoft Forms 2.0 SpinButton and then click OK.
3. Adjust the dimensions of the control as needed. Note that if you make the control taller than it is wide, you get up and down arrows; if you make the control wider than it is tall, you get left and right arrows.
4. In the Design tab's Controls group, click the Text Box button.
5. Draw the text box on the form beside the spin button. Access adds the text box and an associated label. Make a note of the text box name because you need it later in these steps.
6. Edit the text box label, as necessary.
7. Click the spin button and choose Design, Property Sheet to display the control's property sheet.
8. In the Data tab, use the `Control Source` property to specify the field in which you want the spin button value stored.
9. In the Other tab, customize the spin button using the following properties:
    - `Min`—Sets the minimum value of the spin button.
    - `Max`—Sets the maximum value of the spin button.
    - `SmallChange`—Sets the amount that the spin button value changes when the users click one of the arrows.
10. Ensure that the text box gets updated with the new spin button value whenever the users click an arrow. To do this, right-click the spin button, click Build Event, click

Code Builder, and then click OK. Access opens the Visual Basic Editor and adds a stub for the `Updated` event, which you can delete. In the procedure list (the one on the right at the top of the module), click Change.

11. Inside the `Change` procedure, type the following statement, where *`TextBox`* is the name of the text box you added in step 6 (see Figure 3.9):

```
TextBox.SetFocus
```


Microsoft Visual Basic - Northwind 2007 - [Form_Spin Button (Code)]
File Edit View Insert Debug Run Tools Add-Ins Window Help
Type a question for help
Project - Northwind 2007
Northwind 2007 (Northwind 2007)
Microsoft Office Access Class Objects
Form_Inventory List
Form_Mortgage Calculator
Form_Order Details
Form_Order Subform for Order D
Form_Purchase Order Details
Form_Purchases Subform for Pur
Form_Receiving Subform for Purc
Form_Sales Reports Dialog
Form_Spin Button
Report_Monthly Sales Report
Report_Quarterly Sales Report
Report_Yearly Sales Report
SpinButton0
Change

```
Option Compare Database
Option Explicit

Private Sub SpinButton0_Change()
Text1.SetFocus
End Sub
```

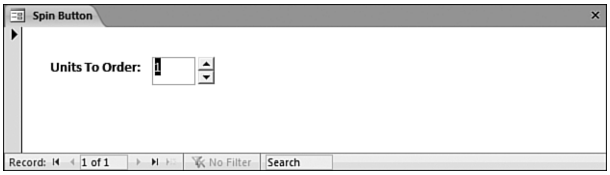
Figure 3.9
Use a simple Visual Basic for Applications statement to keep the text box updated with the current spin button value.

12. Choose File, Close and Return to Microsoft Office Access (or press Alt+Q or Alt+F11).
13. Use the Property Sheet pane's drop-down list to choose the text box.
14. In the Data tab, set the text box value equal to the spin button value by using the `Control Source` to add the following expression (where *`SpinButton`* is the name of the spin button control):

```
=SpinButton
```

15. Close the property sheet.

Figure 3.10 shows a form with a spin button and associated text box.



Units To Order: 1

Figure 3.10
When you click the spin button arrows, the value displayed inside the text box changes accordingly.

Entering Numbers Using a Scrollbar

Scrollbars are normally used to navigate windows, but by themselves you can use them to enter values between a predefined maximum and minimum. In this context, they are very similar to spin buttons, so the procedure for adding them to your form is more or less the same:

1. In the Design tab's Controls group, click the Insert ActiveX Control button. Access displays the Insert ActiveX Control dialog box.
2. In the Select an ActiveX Control list, click Microsoft Forms 2.0 ScrollBar and then click OK.
3. Adjust the dimensions of the control as needed. Note that if you make the control taller than it is wide, you get a vertical scrollbar; if you make the control wider than it is tall, you get a horizontal scrollbar.
4. In the Design tab's Controls group, click the Text Box button, draw the text box on the form beside the scrollbar, and edit the label. Remember to make note of the text box name.
5. Click the scrollbar and choose Design, Property Sheet to display the control's property sheet.
6. In the Data tab, use the `Control Source` property to specify the field in which you want the scrollbar value stored.
7. In the Other tab, customize the scrollbar using the following properties:
   - `Min`—Sets the minimum value of the scrollbar.
   - `Max`—Sets the maximum value of the scrollbar.
   - `SmallChange`—Sets the amount that the scrollbar value changes when the users click one of the scroll arrows.
   - `LargeChange`—Sets the amount that the scrollbar value changes when the users click between the scroll box and one of the scroll arrows.
8. Right-click the scrollbar, click Build Event, click Code Builder, and then click OK. Access opens the Visual Basic Editor and adds a stub for the `Updated` event, which you can delete. In the procedure list (the one on the right at the top of the module), click Change.
9. Inside the `Change` procedure, type the following statement, where *`TextBox`* is name of the text box you added in step 4:
   ```
   TextBox.SetFocus
   ```
10. Choose File, Close and Return to Microsoft Office Access (or press Alt+Q or Alt+F11).
11. Use the property sheet drop-down list to choose the text box.
12. In the Data tab, set the text box value equal to the scrollbar value by using the `Control Source` property to add the following expression (where *`ScrollBar`* is the name of the scrollbar control):
    ```
    =ScrollBar
    ```
13. Close the property sheet.

Figure 3.11 shows a form with a scrollbar and associated text box.

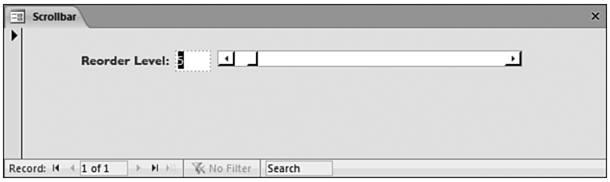


Figure 3.11
When you click the scrollbar arrows or drag the scroll box, the value displayed inside the text box changes accordingly.

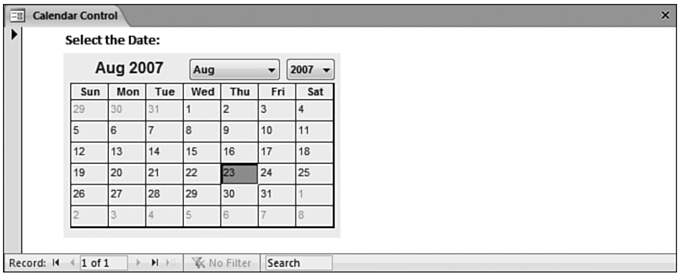
Entering Dates Using a Calendar

Entering dates can be problematic because users might not use the proper format. For example, they might reverse the month and day, they might use an unrecognizable month abbreviation, they might leave out a date separator, and so on. To avoid these common data entry scenarios, you can add a calendar control to your form. The users enter a date by selecting the month and year and then just clicking the day of the month.

Follow these steps to add a calendar control to your form:

1. In the Design tab's Controls group, click the Insert ActiveX Control button. Access displays the Insert ActiveX Control dialog box.
2. In the Select an ActiveX Control list, click Calendar Control 12.0 and then click OK.
3. Adjust the size and position of the control to ensure that the calendar is displayed properly.
4. Click the calendar and choose Design, Property Sheet to display the control's property sheet.
5. In the Data tab, use the `Control Source` property to specify the date field in which you want the calendar value stored.
6. In the Other tab, use properties such as `DayFontColor` and `GridCellEffect` to format the calendar.
7. Close the property sheet.

Figure 3.12 shows a form with a calendar control added.



Sun	Mon	Tue	Wed	Thu	Fri	Sat
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

Figure 3.12
With the calendar control, use the drop-down lists to select a month and year and then click the date you want.

Collecting Form Data via Email

NEW

Although small databases are often managed by a single person, larger databases require input from multiple people. If those people are on your network, such collaboration is most easily achieved either by placing the database in a shared network location or by moving some or all of the data to a SharePoint site. However, what do you do if your collaborators don't have network access? One solution would be to export the necessary tables and forms into another database and then email that database to the collaborators. When the data is returned, you could then import it back into your database.

Such a solution is workable, but a little too unwieldy to be practical. Fortunately, Access 2007 comes with a new feature called Access Data Collection (ADC) that makes email-based data collection much easier. With this feature, you create a form that includes fields for the data you want to collect, place that form in an HTML email message, and then send that message to every person from whom you want to collect the data. Each person fills in the form and returns the message, which is then saved in a special Outlook folder called Access Data Collection Replies. You then synchronize Access (by hand or automatically) with those replies, and the data is added to the underlying table.

CAUTION

Many people set up their email clients to read messages in plain text, and that's not good for ADC. First, viewing the ADC message in plain text prevents the user from seeing the form at all. Second, even if the user converts the message to HTML (by clicking the Information bar and then clicking Display as HTML), Access will perceive this as a "change" to the form, and it won't process the reply. That is, in the Access Data Collection Replies folder, the Data Collection Status column for the reply will say the following:

```
Failure: Cannot process this e-mail message. The form in
this e-mail message is either corrupt or has been modified.
```

The user must turn off the Outlook option to read messages in plain text (choose Tools, Trust Center, click E-mail Security, and then click to deactivate the Read All Standard Mail in Plain Text check box), reply to and fill in the ADC form, and then reset the plain text option.

Sending the Access Data Collection Email Message

Unlike the other forms you've seen in this chapter, an ADC form cannot be created by hand. Instead, Access runs a wizard that builds the form step by step, as shown in the following procedure:

1. Use the Navigation pane to click the table you want to use to store the collected data.
2. Choose External Data, Create E-mail. Access starts the ADC Wizard.

3. In the initial wizard dialog box, click Next. The wizard asks whether you want to use an HTML form or an InfoPath form.
4. Click HTML Form and then click Next.
5. If the table already contains data, the wizard asks whether you want to collect new information or update existing information. Click one of the following options and then click Next:
   - **Collect New Information Only**—Click this option to send a blank form for new data.
   - **Update Existing Information**—Click this option to send existing data for the recipient to edit. The record that contains the recipient's address is the record the recipient will edit.
6. For each field you want to include in the form, click the field and then click > (or click >> to add all the fields). Click Next.
7. If you want Access to synchronize with Outlook automatically when the replies arrive, click to activate the Automatically Process Replies and Add Data to *Table* check box (where *Table* is the name of the table you chose in step 1) and then click Next.
8. Choose how you want to specify the message recipients (click Next after you've made your choice):
   - **Enter the E-mail Addresses in Microsoft Office Outlook**—Click this option to enter the recipients by hand in the Outlook message window that appears later. Skip to step 10.
   - **Use the E-mail Addresses Stored in a Field in the Database**—Click this option if you have the recipients' addresses stored in the current database. Proceed to step 9.
9. Specify the addresses in the database using one of the following options (click Next when you are done):
   - **The Current Table or Query**—Click this option if the email addresses are stored in the table you're using with ADC. Use the associated list to click the field that contains the addresses.
   - **An Associated Table**—Click this option if the addresses reside in another table that's related to the current table. First, use the associated list to click the field in the current table upon which the relationship is based. Second, when Access displays a list of fields in the related table, use the list to click the field that contains the email addresses.
10. Edit the message Subject and Introduction, as needed. If the addresses came from the Access database, click where you want the addresses added: the To Field, Cc Field, or Bcc Field. Click Next.

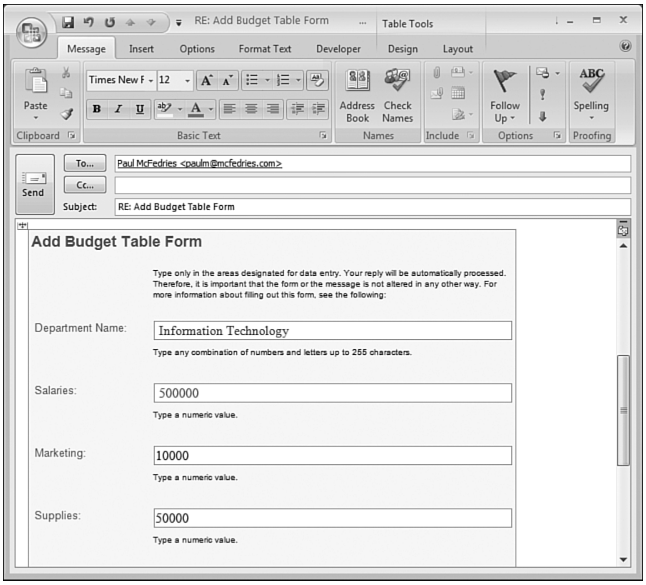
11. You now have two ways to proceed:

- If you choose an Access field for the recipient addresses, click Next. Access displays a list of the recipients with check boxes for each address. Leave the check boxes activated for the recipients you want to receive the message. When you are done, click Send.
- If you will be specifying recipients via Outlook, click Create to create the message, select the recipients, and then click Send.

Replying to an Access Data Collection Email Message

If you receive an ADC message, you need to fill in the fields and return the message. Here are the steps to follow:

1. Click the Access Data Collection message and then click Reply. Access displays the message window.
2. Scroll down the message body until you see the form, as shown in Figure 3.13.



Paul McFedries <paulm@mcfedries.com>

Subject: RE: Add Budget Table Form

Add Budget Table Form

Type only in the areas designated for data entry. Your reply will be automatically processed. Therefore, it is important that the form or the message is not altered in any other way. For more information about filling out this form, see the following:

Department Name:	Information Technology Type any combination of numbers and letters up to 255 characters.
Salaries:	500000 Type a numeric value.
Marketing:	10000 Type a numeric value.
Supplies:	50000 Type a numeric value.

Figure 3.13
When you reply to an ADC message, fill in each form field in the body of the reply.

3. Click inside a form field and type the data.
4. Repeat step 3 for each field.
5. After you've filled in each field, click Send.

Managing the Access Data Collection Replies

As I mentioned earlier, when you receive replies to your messages, they are automatically routed to the Access Data Collection Replies folder in Outlook. (This is a subfolder of the Inbox folder.) If you didn't set up Access to handle the replies automatically, follow these steps to handle a reply manually:

1. In Outlook, open the reply.
2. Click Export to Access. Outlook asks you to confirm.
3. Click OK. Outlook exports the data.
4. Click OK.

From Here

- ➔ To learn how to work with the `Label` control, **see** "Adding Labels to the Form," **p. 41**. (Chapter 2)
- ➔ To learn how to work with the `Text Box` control, **see** "Adding Text Boxes to the Form," **p. 43**. (Chapter 2)
- ➔ For the specifics of the Expression Builder, **see** "Working with the Expression Builder," **p. 256**. (Chapter 11)
- ➔ To learn about inner joins, **see** "Establishing Table Relationships," **p. 267**. (Chapter 12)
- ➔ For the details on building SQL `SELECT` statements, **see** "Using SQL to Perform a Select Query," **p. 342**. (Chapter 15)