In this chapter

- Learn how to add lookup fields to your tables
- Run the Table Analyzer to get hints on making your design more efficient
- Create and modify relationships between tables
- Use referential integrity to protect your data

6

Tapping the Power of Relationships

The average person might define the term *relationship* as a connection of blood or marriage. In the relational database world, the term *relationship* simply refers to an association between two records. That's the good news—you don't have to send your Access database flowers when you mess up. The bad news is that you might find Access relationships just as frustrating as your blood relations. An irate mother-in-law has nothing on an Access relationship run amuck.

The truth is that relationships are like everything else in Access. If you learn the basics and apply them correctly, they will serve you, your data, and your users well. In fact, they're the foundation on which the entire database stands. After you have the hang of it, you'll see that creating the correct relationship between two tables is rather intuitive.

You can think of a relationship as a connection between fields in two related tables where the two fields share common values. By matching the values, Access can combine records from those related tables to display related data. That's the real power behind relationships—the capability to display just the data you need when you need it. For example, if you're storing catalogs in one table and plants in another, you can use a relationship to tell Access how to figure out which plants came from which catalog.

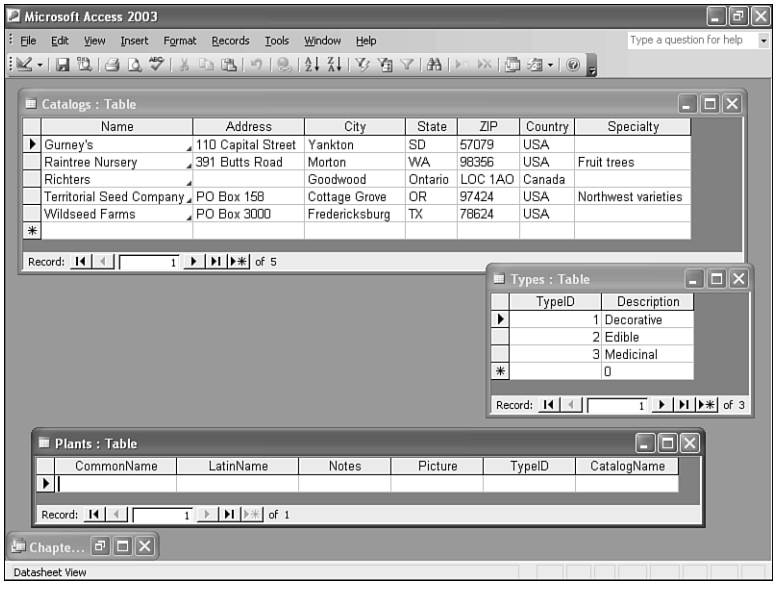
You've already completed the hardest section of this book—you learned about primary and foreign keys (in Chapter 4, "Planning a Database") and created the tables (in Chapter 5, "Building Your First Tables"). In this chapter, we'll show you how to create relationships between those tables. The result will be that you can quickly learn from which catalog you ordered your last batch of cosmos seeds. Or, you can list all the edible plants you're currently growing and so on.

You can download the `Chapter 6.mdb` sample file, which is inclusive of all the examples in this chapter, from `http://www.quepublishing.com/`. If you want to follow along with our examples, download `Chapter 5.mdb`.

Using the Lookup Wizard

In Chapter 5, you created the three tables in the sample database. You also entered three type records and a few catalog addresses, as shown in Figure 6.1. At this point, you should be ready to enter a few plant records.

FIGURE 6.1
You created these three tables in Chapter 5.



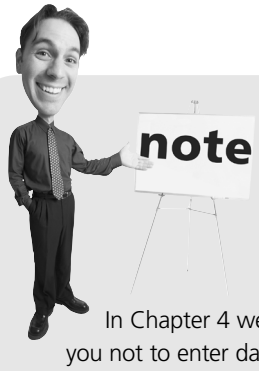
Let's open the empty Plants table and enter records for a few of the latest season's new plants, which are listed in Table 6.1.

TABLE 6.1 Plant Records

Common Name	Latin Name	Catalog	Type
Yarrow	Achillea millefolium	*Wildseed Farms*	Decorative
Purple Coneflower	Echinacea purpurea	*Wildseed Farms*	Medicinal
Cosmos	Cosmos bipinnatus	*Gurney's*	Decorative
Black-eyed Susan	Rudbeckia hirta	*Wildseed Farms*	Decorative
Rocket Larkspur	Delphinium ajacis	*Wildseed Farms*	Decorative
German Chamomile	Matricaria recutita	*Gurney's*	Medicinal
Calendula	Calendula officinalis	*Richters*	Decorative

Just to refresh your memory, we'll help you enter the first record:

1. Select the first field in the first row and enter **`Yarrow`**.
2. Press the right arrow key and enter **`Achillea millefolium`**.
3. Skip the next two fields and enter the TypeID value for decorative. This is where data entry becomes a bit challenging. You probably don't have these values memorized, so open the Types table. After viewing the table, you can clearly see that the TypeID value for decorative type is `1`.
4. Return to the Plants table and enter **`1`** in the TypeID field for yarrow.



In Chapter 4 we advised you not to enter data directly into a table. During the design and development stage, it's common to play with a few records, just so you can test your design. Later, you'll create data entry forms that you'll use when you actually put the application to work.

Repeat steps 1–4 to enter the record for Purple Coneflower, which we've planted for its medicinal value. Even though you just looked at the Types table, you'll have to return to that table to learn the value for medicinal plants. To do so, find the Types table on the Windows taskbar and browse the records. This time, the appropriate value is `3`, so return to the Plants table and enter **`3`** in the Purple Coneflower's TypeID field.

This routine could quickly become annoying, not to mention that it's horribly inefficient. If working with related tables is so efficient, why are you working so hard just to enter the few records shown in Figure 6.2?

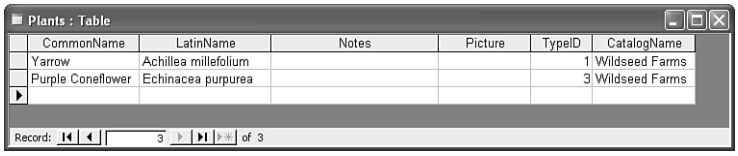


FIGURE 6.2
Remembering the appropriate type value for each record is difficult.

One way to solve this problem is to create what's known as a *lookup field* in the plant table. A lookup field refers to a field that displays one value but stores another. The best part is that a wizard is available that will help you create the lookup field.

As is, the TypeID values are meaningless, and remembering which value to enter is difficult. If your table had hundreds of records, it would be impossible, so let's convert the plant table's TypeID field to a lookup field. To do so, follow these steps:

1. Open the plant table in Design view by selecting it in the Database window and then clicking **Design** on the Database Window toolbar.
2. Currently, the TypeID data type is Number. Click the right side of the TypeID field's Data Type column to display its drop-down list.
3. Select **Lookup Wizard**, as shown in Figure 6.3.

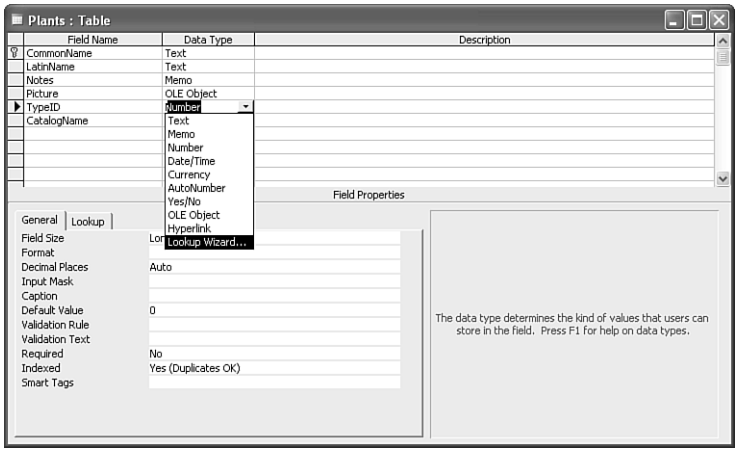


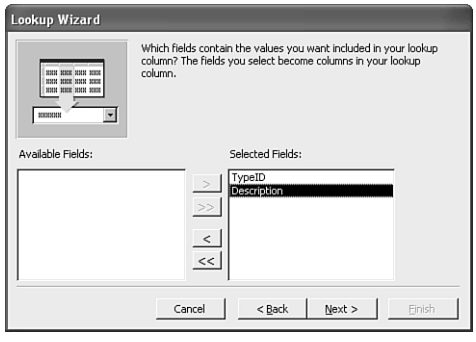
FIGURE 6.3
Select Lookup Wizard from the Data Type column's drop-down list.

4. The wizard's first panel lets you choose between displaying existing values from a table (or query) and entering a list of items. You want to display the descriptive entries from the Types table, so accept **I Want the Lookup Column to Look Up the Values in a Table or Query**, which is the default option. Then click **Next**.
5. When you want to display existing values, the wizard displays a list of tables. The values you want to display are in the Types table, so select **Table: Types**.

Notice the View panel at the bottom of the wizard window. If you want to choose values from a query, select the **Queries** option to update the wizard's list. Or, you can display both tables and queries by clicking the **Both** option. Click **Next** to continue.

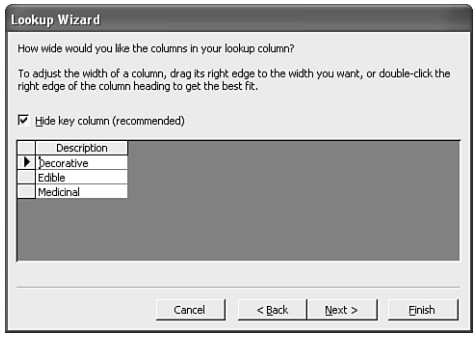
6. The next panel displays all the fields in the table or query you selected in the previous window. Generally, you should select the field that is the primary key of the lookup table and the field that contains the values you want to display. In the case of our example, click the double arrow button to move both fields to the Selected Fields list, as shown in Figure 6.4. TypeID is the primary key of the table, and Description is the field whose values you want to display to the user. Click **Next** when you're ready to continue.

FIGURE 6.4
Move both fields to the Selected Fields list.



7. The next panel allows you to decide which fields should be used to sort the list of data. In this case, the default sort order is fine, so click **Next**.
8. The panel shown in Figure 6.5 displays the values the lookup field will display. Notice that the Hide Key Column option is checked by default; that means Access won't display the primary key values (refer to Figure 6.2). Instead, the list will display only the descriptive values shown in the current list. If you need to, adjust the width of the column so you can completely see each entry. To continue, click **Next**.

FIGURE 6.5
Adjust the width of the column if necessary.



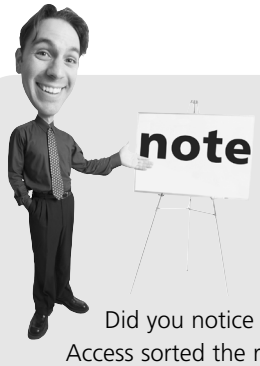
9. Finally, the wizard borrows the field's name for the new lookup field. Accept the wizard's suggested name and click **Finish**.
10. When prompted to save the table, click **Yes**. If you click No, the wizard will discard the lookup field properties you just created.

To see the new lookup field, view the table in Datasheet view by clicking **View** on the Table Design toolbar. You might recall that the original TypeID field contained numeric values (refer to Figure 6.2). Now that field displays descriptive text instead. Specifically, the lookup field automatically displays the appropriate description for any existing records instead of the value it's actually storing.

The new lookup field also lets you easily enter the foreign key values for each record in the plant table. Refer to Table 6.1 to enter the next record, the one for cosmos. When you get to the TypeID field, click the arrow to open the new lookup field's drop-down list, as we've done in Figure 6.6.

You can use the list to enter a type value for a new record by simply clicking the value in the list. Select **Decorative** to finish the record for cosmos and enter the remaining records in Table 6.1. When you're done, your table should resemble the one shown in Figure 6.7. (Close and reopen the table to see your records sorted as shown.)

Open the Plants table in Design view so you can examine the lookup field properties. (Click the **View** button on the Table Datasheet toolbar.) Next, select any field in the TypeID field row and click the **Lookup** tab in the Field Properties pane.



Did you notice that Access sorted the records after you added the lookup field? The sort really has nothing to do with the lookup field; it just appears that way because the sort showed up after you added the lookup field, which is just a coincidence. Access is actually sorting the records by the primary key values. Because the primary key field is a text field (CommonName), Access sorts the records alphabetically by the values of that field.

It might not matter to you right now, but forms and reports inherit a table's lookup fields. That means any forms or reports bound to the Plants table will use a combo box—not a text box control, which is the default control—to display the `TypeID` field's contents. This will make more sense to you in Chapter 8.

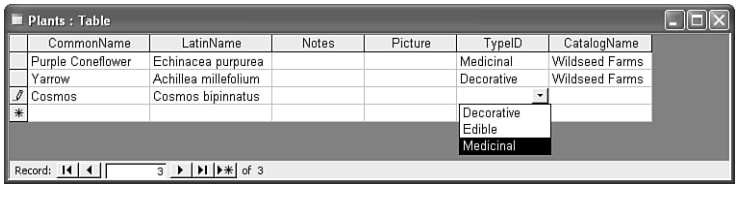
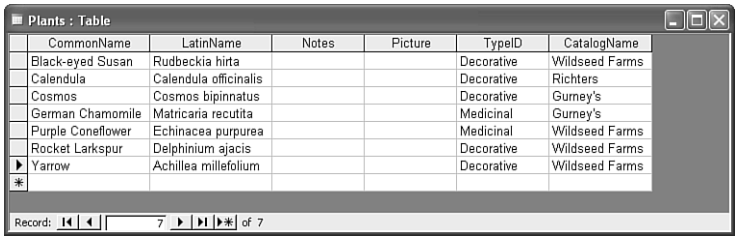


FIGURE 6.6
Open the lookup field's drop-down list to see the data items you can enter.

FIGURE 6.7
Seven plants now appear in the plant table.


Plants : Table

CommonName	LatinName	Notes	Picture	TypeID	CatalogName
Black-eyed Susan	Rudbeckia hirta			Decorative	Wildseed Farms
Calendula	Calendula officinalis			Decorative	Richters
Cosmos	Cosmos bipinnatus			Decorative	Gurney's
German Chamomile	Matricaria recutita			Medicinal	Gurney's
Purple Coneflower	Echinacea purpurea			Medicinal	Wildseed Farms
Rocket Larkspur	Delphinium ajacis			Decorative	Wildseed Farms
Yarrow	Achillea millefolium			Decorative	Wildseed Farms

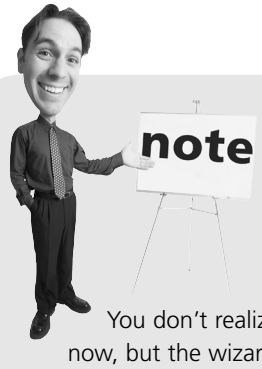
Record: 7 of 7

Notice that the Display Control is a combo box control. You haven't been introduced to controls yet, but a *combo box* is a complex control with a text box for entering data and a list. You can enter data directly into the text box component, or you can select an item from the control's drop-down list. You'll learn more about creating and using the various controls that Access offers in Chapter 8, "Creating and Using Data Entry Forms," and Chapter 13, "Customizing Forms."

Deleting a Lookup Field

To delete the lookup field, select **Text Box** from the Display Control property field's drop-down list, as shown in Figure 6.8. But don't do so right now—you need to keep the lookup field you just added to the TypeID field. Close the plant table without making any changes to the lookup field.

Wouldn't it be much easier to just select an item from a drop-down list than type the entire entry yourself? Not only is it easier and more efficient, a drop-down list is a more reliable data-entry solution because it eliminates typos and other human errors. We recommend you limit the choices a user can enter in this manner as often as possible.



You don't realize it now, but the wizard created your first relationship. Later in this chapter, you'll get a closer look at that relationship.

While you were entering records into the Plants table in the last section, did you think that it might be nice to also add a lookup table to the CatalogName field? It certainly would be more efficient, and it would also eliminate typos. You probably had to correct at least an entry or two. (Stay tuned because we'll use another method for displaying items in a drop-down list in Chapter 13.)

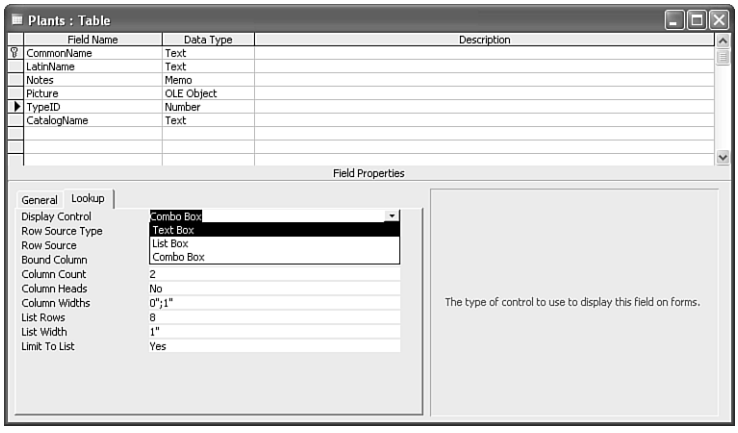


FIGURE 6.8
To delete the lookup field, select Text Box from the Display Control property.

Using the Table Analyzer to Create Relationships

At this point, are you completely satisfied with the design? Let's see whether the Table Analyzer agrees with you. This utility reviews current entries and makes suggestions on improving the design.

In a nutshell, the utility looks for repeated data and helps you copy that data to a new and related table. Not every table will need changes, but it doesn't hurt to run the analyzer just to see what it proposes. In addition, until you're more familiar with design rules, you can rely on the Table Analyzer to help you through the process of creating properly normalized tables. (Refer to Chapter 4 for a definition of normalization.) To analyze the Plants table, follow these steps:

1. Select **Analyze** from the Tools menu and select **Table** from the resulting submenu.
2. The first pane in the wizard offers to show you a sample problem that you might experience with duplicate data. Feel free to review the example. Click **Next** when you're ready to continue.
3. The next pane offers to explain, by example, how the analyzer will split a table. View the example, and when you're ready to continue, click **Next**.
4. Select the table you want to analyze—we selected Plants. Click **Next** when you're ready to continue to the next step.
5. While learning, you'll probably want to let the wizard do as much work as possible. Let the wizard choose which fields to split into a new table by accepting the default option **(Yes, Let the Wizard Decide)**. Click **Next** to continue. We won't even look at the other option; if you knew how to split the table, you wouldn't need the wizard!

6. The next pane, shown in Figure 6.9, displays the wizard's suggestions for splitting the table. Does the solution look familiar? It might because this is exactly the route you took in Chapter 4, when you designed the database. Right now, there's no relationship between the Catalogs and Plants tables, so the wizard has no way of knowing that you already have a catalog table.

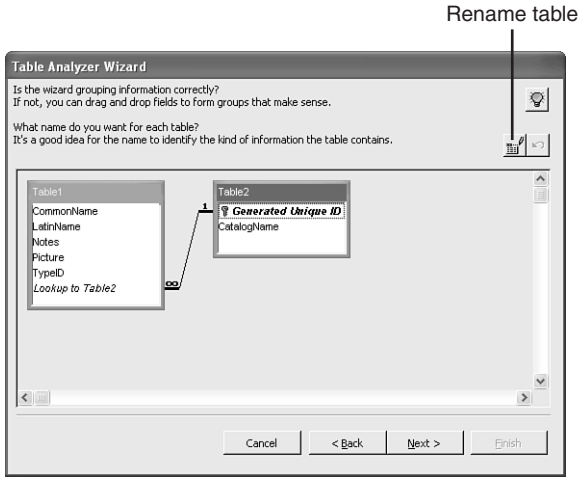


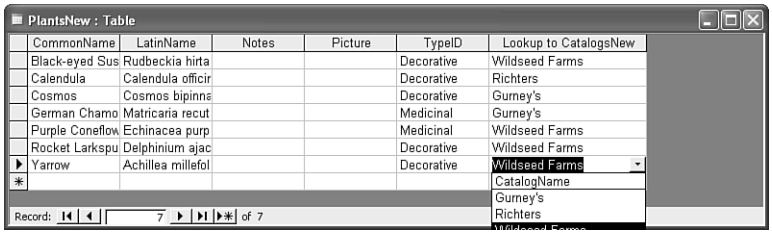
FIGURE 6.9
The wizard offers its solution.

At this point, you could abandon your task, but let's continue because you're probably unfamiliar with how this utility works. Just remember: You had the right design; all you're lacking is the relationship between the two tables.

7. Select **Table1** and click the **Rename Table** button to the right of the window (refer to Figure 6.9). Enter the name `PlantsNew`, and then click **OK**.
8. Repeat step 7 and rename Table2 `CatalogsNew`. Then click **Next** to continue.
9. The next pane enables you to reset the suggested primary key. The wizard suggests adding an AutoNumber data type field to the CatalogsNew table and using it as the primary key. However, the wizard doesn't define a primary key for PlantsNew, so you must do so. Select the **CommonName** field in the **PlantsNew** list and click the **Set Unique Identifier** button. In response, the wizard displays a primary key icon next to that field. Click **Next** to continue.
10. In the final pane, the wizard offers to create a query. You're not ready for queries yet, so select the **No, Don't Create the Query** option. You'll probably want to deselect the Display Help on Working with the New Tables or Queries options. Otherwise, you'll just have an additional window to close. Click **Finish** to complete the changes.

The two new tables, PlantsNew and CatalogsNew, are for the most part, just like Plants and Catalogs. The two fields in CatalogsNew are transposed, but that doesn't matter. In addition, PlantsNew has a second lookup field. Select any value in the CatalogName field to display that field's new lookup field, as shown in Figure 6.10. The Table Analyzer created it automatically.

FIGURE 6.10
The Table Analyzer has added a second lookup field to the Plants table.



At this point, you have two relationships in your database:

- Between the TypeID fields in Plants and Types
- Between ID and CatalogsNew_ID (created by the Table Analyzer)

We completed this exercise just to show you how the analyzer works. Now you'll know what to expect when you need it. For now, delete both PlantsNew and CatalogsNew. To delete a table, simply select it in the Database window and then press the **Delete** key. After deleting the two tables the Table Analyzer created, just one relationship exists—the one between the Plants and Types table the Lookup Wizard created in the first section. It's okay to leave the new tables in the database if you want to study them later. They won't interfere with other objects.

Using the Relationships Window

It's fine to use the Lookup Wizard or the Table Analyzer, but you can create relationships yourself. First, open the Relationships window shown in Figure 6.11 by clicking the **Relationships** button on the Database toolbar. If that button is not visible, press **F11** to give focus to the Database window. Then, the button should be available.

The window contains two field lists, one for Plants and one for Types. (If your window is empty, click the **Show All Relationships** button on the Relationship toolbar.) In addition, there's a line between the two tables, which is known as a *join line*. When you created the lookup field for the TypeID field in the Plants table, the wizard created this relationship.

When you need to create a relationship, drag a field from one list to another. In almost all cases, you'll drag the primary key field to its counterpart in the related table. If you look closely at the existing join line, you'll see that it connects the two TypeID fields in both tables.

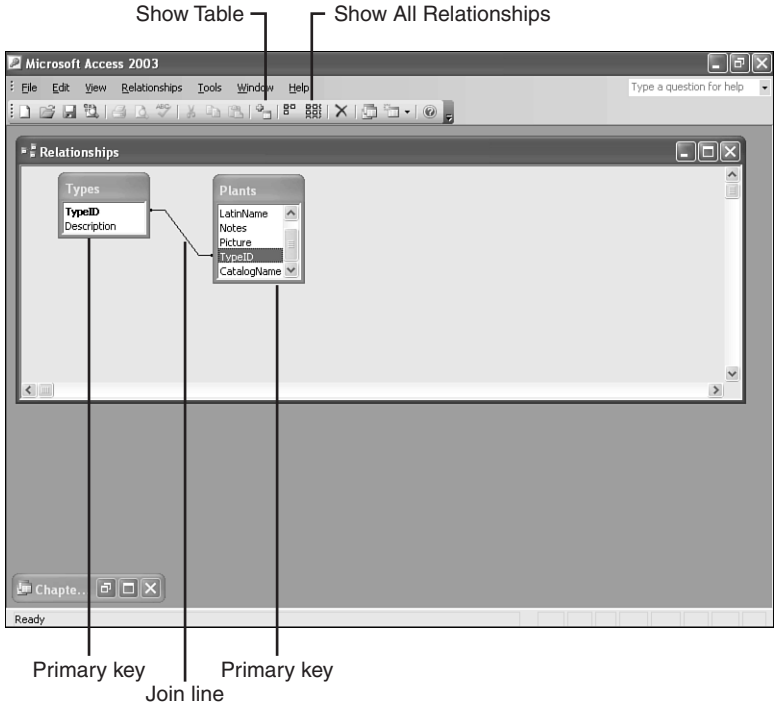


FIGURE 6.11
Open the Relationships window.

You can easily find the primary key fields in both tables because Access displays the primary key fields in bold text. Most of the time, you'll rely on the Relationships window to create permanent relationships between tables.

You still have a relationship you need to create—the one between the Catalogs and Plants tables. So, let's get started:

1. Click the **Show Table** button on the Relationship toolbar.
2. In the resulting Show Table dialog box, select **Catalogs**, and then click **Add**. Alternatively, you could double-click **Catalogs**.
3. Close the Show Table dialog box by clicking **Close**.
4. Select the **Name** field in the Catalogs list, but don't release the mouse. Access will display the field pointer shown in Figure 6.12.

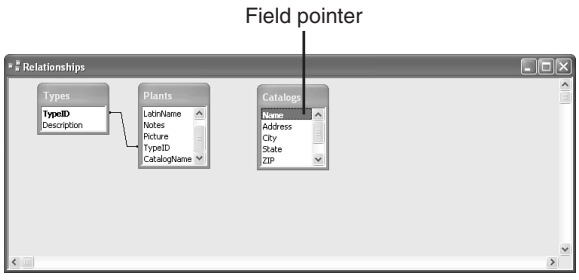


FIGURE 6.12
Access displays the field pointer when you drag a field from one list to another.

5. Still holding down the mouse, drag the Name field to the CatalogName field in the Plants list and then release the mouse. Access displays the Edit Relationships dialog box.
6. Check the **Enforce Referential Integrity** option, as shown in Figure 6.13; then click the **Create** button. (We'll introduce referential integrity in the next section.)

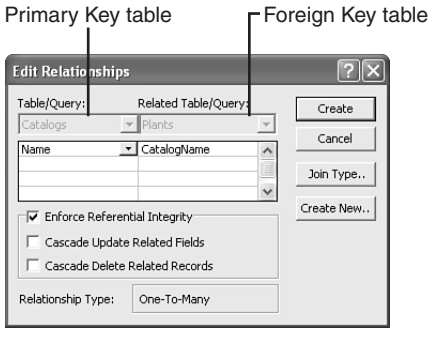


FIGURE 6.13
Specify relationship properties in the Edit Relationships dialog box.

Figure 6.14 shows the new relationship between the two tables. Notice that the join line is very different from the one connecting the Plants and Types tables:

- **The 1 next to the Catalogs list defines the primary key table, or the one side of the relationship**—Only one matching record exists in this table, which makes sense because the relationship is based on the primary key value, which as you've already learned, must be a unique value.
- **The infinity symbol next to the Plants list indicates the many sides of this relationship**—Many related records might exist in the Plants table that match any given record in the Catalogs table.

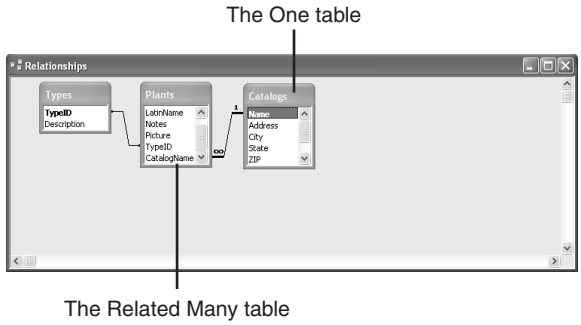


FIGURE 6.14
The join line reflects the relationship you just created.

The Many Sides to Relationships

We just introduced you to a new concept—a *one-to-many* relationship between the Plants and Catalog tables. There are three types of relationships. A thorough discussion of the relationship types is beyond the scope of this book, but you should be familiar with the terms:

- **One-to-one**—Has only one matching record in both tables. You won't see these relationships very often.
- **One-to-many**—The most common relationship. Each record in the primary key table can have many records in the related table. For instance, each catalog can match many records in the actual Plants table, but each plant matches only one catalog.
- **Many-to-many**—Both tables can have many records in this relationship. For instance, you could add a table of colors to describe the bloom on each plant. Each color could refer to many plants, and each plant could consist of more than one color.

Don't spend too much time thinking about the types of relationships right now. Fortunately, Access does a good job of interpreting the relationships between tables.

Modifying a Relationship

Now that you know how to create a relationship, let's modify one. Specifically, let's turn on *referential integrity* for the relationship between the Plants and Types tables. Referential integrity refers to a set of rules that protects data from changes that don't make sense. For example, think about the relationship between plants and types. What would happen if you deleted the first row from the Types table? You'd no longer be able to look up the descriptions of plants with a TypeID of 1. That's the sort of problem referential integrity prevents. We'll demonstrate how this works after adding referential integrity to the relationship.

To display the Edit Relationships dialog box for the relationship between the Plants and Types tables, double-click the join line between those two tables. In the resulting dialog box, check the **Enforce Referential Integrity** option. Then click **OK**, which is a little different from the last time when you clicked Create when you were done (refer to Figure 6.13).

After you modify the relationship between the Plants and Types tables, Access updates the join line accordingly, as shown in Figure 6.15. It's easy to see that the relationship between the two tables enforces referential integrity in a one-to-many relationship.

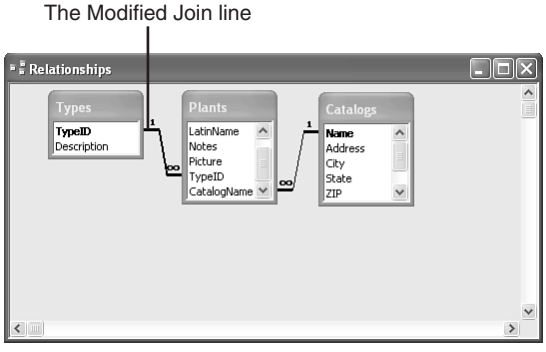


FIGURE 6.15
Access updates the join line between the Plants and Types tables.

You probably noticed that several other buttons are available in the Edit Relationship dialog box (refer to Figure 6.13). Let's take a brief look at the remaining edit possibilities:

- **Table/Query**—Always lists the primary key side of a relationship. Specifies the appropriate key field(s) in the cells just below this control. Access usually defaults to the correct fields.
- **Related Table/Query**—Always lists the foreign key side of a relationship. Specifies the appropriate key field(s) in the cells just below this control. Access usually defaults to the correct fields.
- **Join type**—Displays another dialog box that lets you modify the type of join. This is an advanced operation that you won't need in this book.
- **Create New**—Offers more field possibilities for multiple field keys. (See Chapter 4 to learn more about primary and foreign keys.)
- **Cascade Update Related Fields**—A referential integrity feature that's available only when the Enforce Referential Integrity option is selected. This option automatically updates any related foreign key values when you change the value of a primary key. We recommend you not use this option unless you have a specific reason to do so.

- **Cascade Delete Related Fields**—A referential integrity feature that's available only when the Enforce Referential Integrity option is selected. This option automatically deletes any related foreign key records when you delete a primary key record. We recommend you not use this option unless you have a specific reason to do so.
- **Relationship Type**—Lists the type of relationship between the tables.

Close the Relationships window. When Access prompts you to save changes, click **Yes**.

Using Referential Integrity

Previously, we told you to select the Enforce Referential Integrity option in the Edit Relationships dialog box (refer to Figure 6.13). Referential integrity is simply a set of rules that protects your data because Access restricts the records you can add and delete. With referential integrity turned on

- You can't change a primary key value if a related record exists in another table.
- You can't enter a foreign key value if that value doesn't already exist as a primary key in the related table.

These rules will make more sense if you see them in action, so let's return to your tables and make a few changes. First, open the Catalog table and try to delete the first record. When you do, Access displays the warning message shown in Figure 6.16. Click **OK** to clear the message.

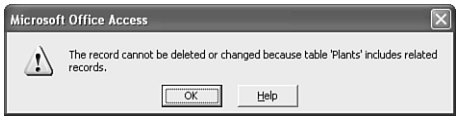
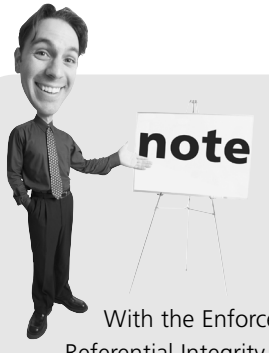


FIGURE 6.16
Referential integrity won't allow you to delete the record for *Gurney's* catalog.



With the Enforce Referential Integrity option turned off, Access enables you to update any value or add and delete any records as long as you don't violate any data type or validation rules (which you'll learn about in Chapter 11, "Customizing Your Tables"). You must turn on referential integrity to use it. We recommend you not use either of the cascading options unless you have a specific reason to do so, and then we suggest you never turn them on permanently.

You can't delete the record for *Gurney's* catalog because two plants, cosmos and German Chamomile, are related to that catalog. As long as even one of those records is there, Access won't let you delete the record for the *Gurney's* catalog.

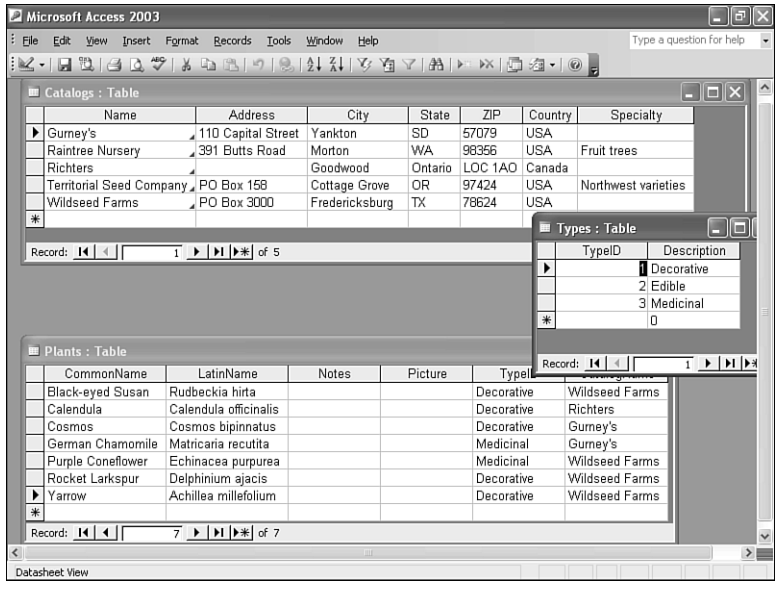
Deleting that record would create what's known as an *orphan*—related records where the foreign key value (in this case, Gurney's) doesn't match a record in the related table. In other words, if you deleted the record for the *Gurney's* catalog, there would be no way to know from which catalog you purchased chamomile and cosmos seeds. This doesn't seem too terribly important, but can you imagine not knowing which customer placed a particular order? That would be a much more serious problem.

Now, let's see what happens when you try to modify a primary key value. In the Catalogs table, try to change `Gurney's` to `Gurneys` (delete the apostrophe character). You'll have to move the insertion point to another row to complete the action. When you do, Access displays another warning message: `The record cannot be deleted or changed because table 'Plants' includes related records`. Click **OK** to clear the message and then press **Esc** to clear the change. The problem is the same as before. Changing `Gurney's` to any other value would strand orphans—cosmos and German Chamomile—in the Plants table.

Subdatasheets—A Product of Relationships

Did you notice the plus signs (+) to the left of each record in the Catalogs table as you worked with it? Access displays those signs in the primary key table when a relationship exists between two tables. If you open all three tables, as we've done in Figure 6.17, you'll see that the Types and Catalogs tables both have a column of plus signs, but the Plants table doesn't.

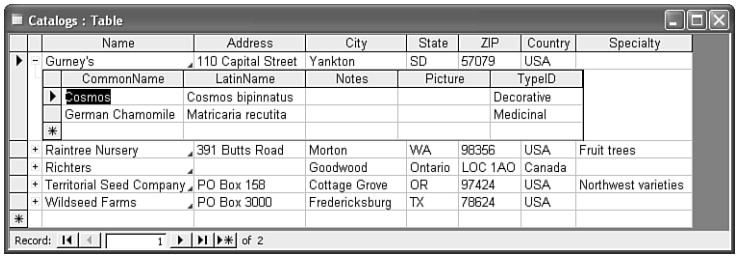
FIGURE 6.17
The plus signs in a table indicate that Access has related records available.



The plus sign means that the corresponding record has at least one related record in another table. If a relationship exists between two tables but no related record exists for a particular primary key value, Access displays a minus sign (–) instead of the plus sign.

Click the plus sign to the left of Gurney's in the Catalogs table. The result is shown in Figure 6.18. Specifically, a subdatasheet displays the records for cosmos and German Chamomile. A *subdatasheet* displays related values via an embedded datasheet (table).

FIGURE 6.18
Click a record's plus sign to display related records.



You can disable this feature if you like. To do so, follow these steps:

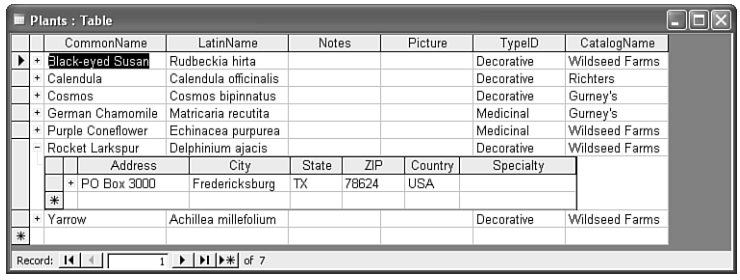
1. Open the table in Design view (click the **View** button on the Table Datasheet toolbar).
2. Click the **Properties** button on the Table Design toolbar.
3. In the resulting dialog box, select **[None]** from the Subdatasheet Name property's drop-down list. The default option is [Auto], which automatically displays one-to-many related records.

If you know the table has related records, but the plus signs aren't visible, you can update the table's properties to display subdatasheets. For instance, the Plants table doesn't display subdatsheets by default because the primary key values have no related records. All the relationships are between that table's foreign key values and the other two tables. To display subdatasheets in the Plants table, do the following:

1. Open the **Plants table** in Table Datasheet view.
2. Select **Subdatasheet** from the Insert menu to display the Insert Subdatasheet dialog box.
3. Select **Catalogs** in the Tables tab.
4. Select **Name** from the Link Child Fields control.
5. Select **CatalogName** from the Link Master Fields control, and click **OK**.

The result of this change is shown in Figure 6.19. Click any plus sign to display that record's corresponding catalog information. You don't want to display subdatasheets in the Plants table, so close the table without saving the change you just made. (Click **No** when Access prompts you to save your changes.)

FIGURE 6.19
The Plants table now displays related records in sub-datasheets.



With all the extra records and fields displayed, navigating can be a bit awkward. Refer to Table 6.2 for shortcut keys for working with and navigating subdatasheets.

TABLE 6.2 Shortcut Keys

Press	Result
Ctrl+Shift+Down Arrow	Expands a record's subdatasheet
Ctrl+Shift+Up Arrow	Collapses a subdatasheet
Tab	Enters the subdatasheet from the last field of the previous record in the datasheet
Shift+Tab	Enters the subdatasheet from the first field of the following record in the datasheet
Ctrl+Tab	Exits the subdatasheet and moves to the first field of the next record in the datasheet
Ctrl+Shift+Tab	Exits the subdatasheet and moves to the last field of the previous record in the datasheet
Tab	Enters the next field in the datasheet from the last field in the subdatasheet

THE ABSOLUTE MINIMUM

The relationships between tables are the cornerstones on which all relationship database systems thrive. As you saw in this chapter, relationships can help you find related data from one table to another. Coupled with referential integrity, they can also help protect your data from accidental changes or deletions. In this chapter, you learned how to

- Create a lookup field and run the Table Analyzer
- Create a relationship between two tables
- Enforce referential integrity and why you should
- Display subdatasheets for browsing related records