Microsoft Office Online

Microsoft Office Access

Creating Your Database and Tables

Applies to: Microsoft Office Access 2003

Hide All

Applies to

Microsoft Office Access 2003

This article was adapted from Chapter 4, "Creating your database and tables," in the book *Microsoft Office Access 2003 Inside Out* by John L. Viescas. Visit Microsoft Learning for information about how to buy this book.

Sections in this article

After you design the tables for your database, defining them in an Access 2003 desktop database (.mdb file) is incredibly easy. This article shows you how it's done. You'll learn how to:

- Create a database application by using a database template.
- Create an empty database for your own custom application.
- Create a simple table by entering data directly in the table.

Get a jump start on defining custom tables by using the Table Wizard to:

- Define your own tables from scratch by using table Design view.
- Select the best data type for each field.
- Define the primary key for your table.
- Set validation rules for your fields and tables.
- Tell Access what relationships to maintain between your tables.
- Optimize data retrieval by adding indexes.
- Set options that affect how you work in table Design view.
- Print a table definition.

Note All the screen images in this article were taken on a computer running Microsoft Windows XP with the display theme set to Windows XP.

About the sample databases referenced in this article

The articles based on this book refer to three sample databases — the Wedding List database, the Housing Reservations database, and the LawTrack Contacts database. Of the three, the LawTrack Contacts sample database is available for download on Office Online.

If you are interested in exploring the other two databases, you can buy the Microsoft Office Access 2003 Inside Out book.

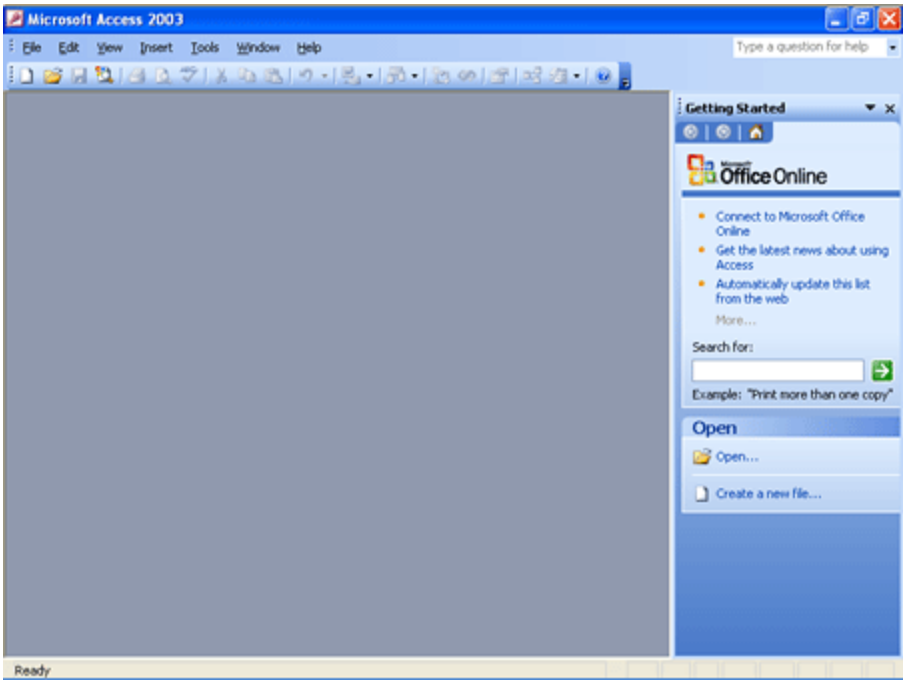
- Download the sample application

Download and open the sample application to follow along with the examples mentioned in this article.

Creating a new database

When you first start Access, you see a blank work area on the left and the Home task pane on the right, as shown in the following illustration.

Original page: http://office.microsoft.com/en-us/access/HA011245541033.aspx



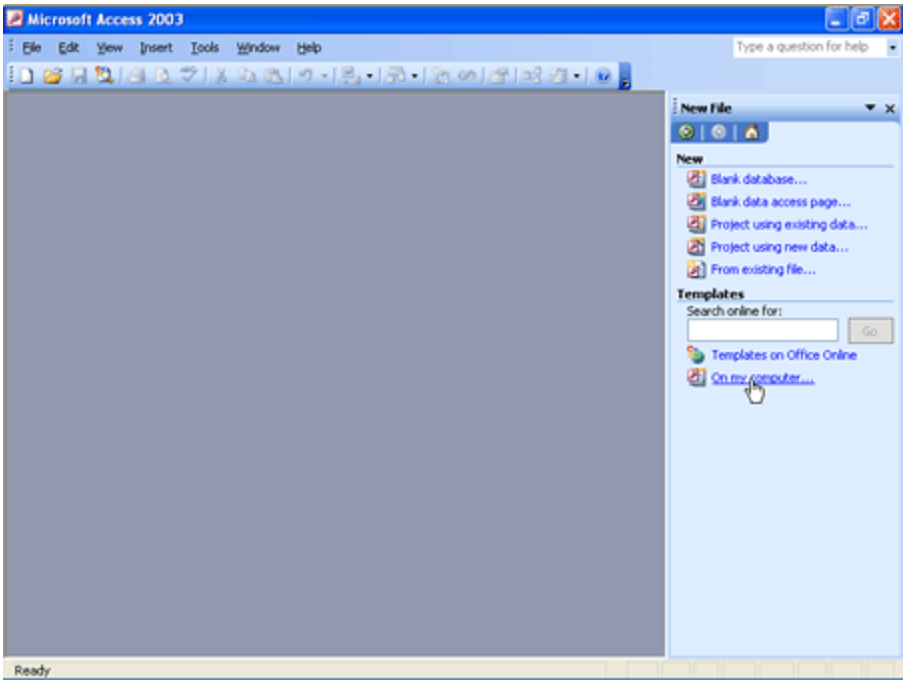
If you've previously opened other databases, such as the Northwind Traders sample database that is included with Access, you also see a most recently used list of up to nine database selections under **Open** in the top part of the task pane.

Using a database template to create a database

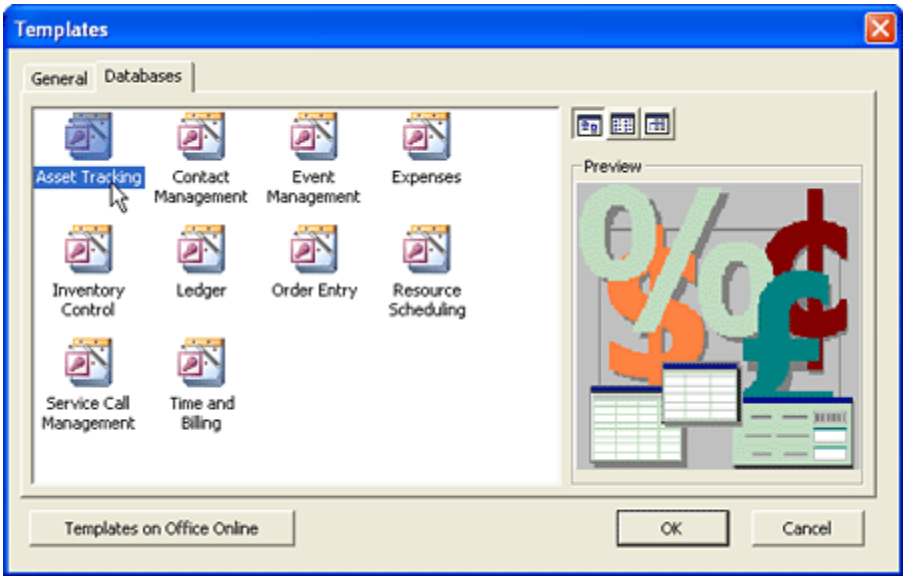
Just for fun, let's explore the built-in database templates first. If you're a beginner, you can use the templates included with Access to create one of several common applications without needing to know anything about designing database software. You might find that one of these applications meets most of your needs right off the bat. As you learn more about Access, you can build on and customize the basic application design and add new features.

Even if you're an experienced developer, you might find that the application templates save you lots of time in setting up the basic tables, queries, forms, and reports for your application. If the application that you need to build is covered by one of the templates, the wizard that builds an application by using one of the templates can take care of many of the simpler design tasks.

When you start Access, click **New** (the leftmost button on the toolbar in the following illustration) on the **Database** toolbar; then, in the **New File** task pane, under **Templates**, click **On my computer**.



When you click **On my computer**, Access opens the **Templates** dialog box. Click the **Databases** tab of this dialog box to see a list of the 10 available templates, as shown in the following illustration.



You work with all the templates in the Database Wizard in the same way. This example will show you the steps that are needed to build an Asset Tracking database.

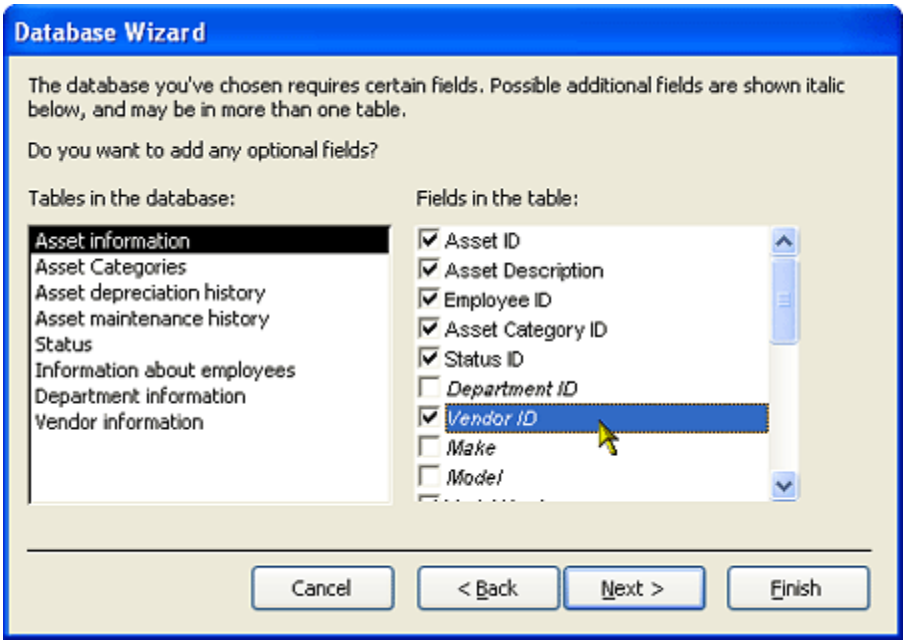
Where to find more templates

In addition to the 10 templates supplied with Access, you can find dozens of database examples at Microsoft Office Online Templates. You can also go directly to this Web site by clicking **Templates on Office Online** in the **Templates** dialog box, as shown in the previous illustration. To find all the database examples, perform a search on the word "Access" on this Web site.

Note that the examples you find on the Web site are completed .mdb files, not template files that require the Database Wizard to build a sample database.

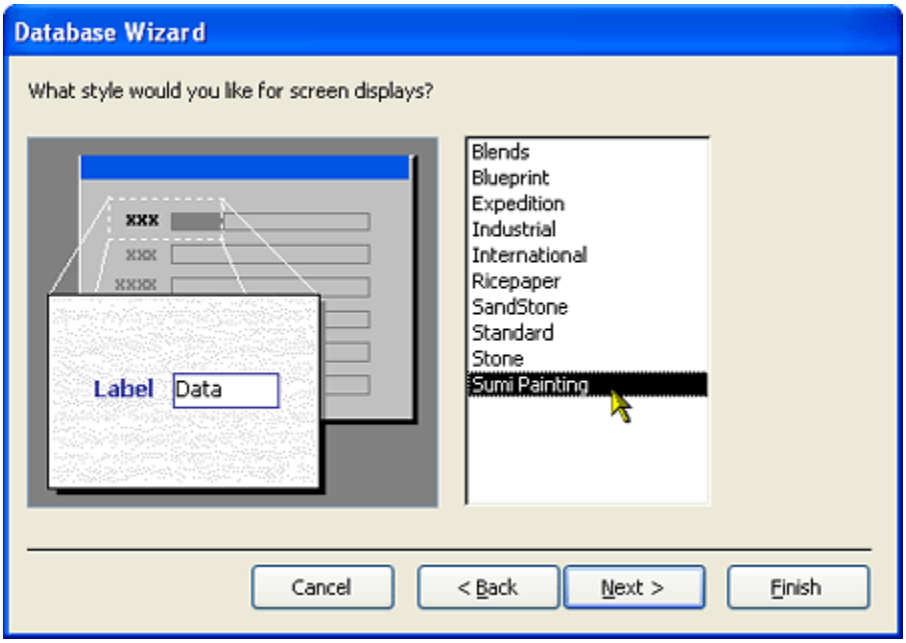
Scan the list of available templates on the **Databases** tab of the **Templates** dialog box. When you click a template icon, Access shows a preview graphic to give you another hint about the purpose of the template. You start the Database Wizard by selecting a template and then clicking **OK**. You can also double-click a template icon. Access opens the **File New Database** dialog box and suggests a name for your new database file. You can modify the name and then click **Create** to launch the wizard.

The wizard takes a few moments to initialize and to create a blank file for your new database application. The wizard first displays a page with a few more details about the capabilities of the application you are about to build. If this isn't what you want, click **Cancel** to close the wizard and delete the database file. You can click **Finish** to have the wizard quickly build the application with all the default options. Click **Next** to proceed to a window that provides options for customizing the tables in your application, as shown in the following illustration.



In this window, you can see the names of the tables the wizard plans to build. As you select each table name in the list on the left, the wizard shows you the fields it will include in that table in the list on the right. For many of the tables, you can have the wizard include or exclude certain optional fields (which appear in italic). In the Asset Tracking application, for example, you might be interested in keeping track of the vendor for each asset. When you click the optional Vendor ID field in the Asset information table, you'll be able to specify from which vendor you acquired the asset. Click **Next** when you finish selecting optional fields for your application.

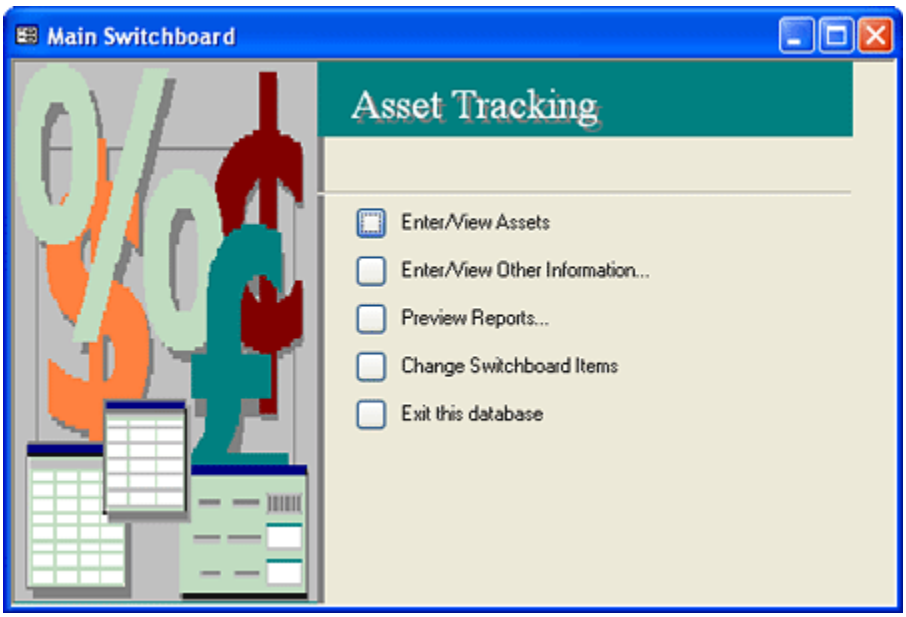
In the next window, shown in the following illustration, you select one of several styles for the forms in your database. Forms are objects in your database that are used to display and edit data on your screen. As you click each style name, the Database Wizard shows you a sample of that style on the left. Some of the styles, such as **Expedition** or **Ricepaper**, are quite whimsical. The **Standard** style has a very businesslike gray-on-gray look. In this case, select the **Sumi Painting** style, which has a bit more character but is still businesslike.



After you select a form style, click **Next** to proceed to the window to select a report style. You might want to select **Bold**, **Casual**, or **Compact** for personal applications. **Corporate**, **Formal**, and **Soft Gray** are good choices for business applications. Again, you can see a sample of the style on the left as you click on each available style on the right. Select an appropriate report style, and then click **Next**.

In the next page of the Database Wizard, you specify the title that will appear in the Access title bar when you run the application. You can also include a picture file such as a company logo in your reports. This picture file can be a bitmap (.bmp), a Windows metafile (.wmf), or an icon file (.ico). Click **Next** after you supply a title for your application.

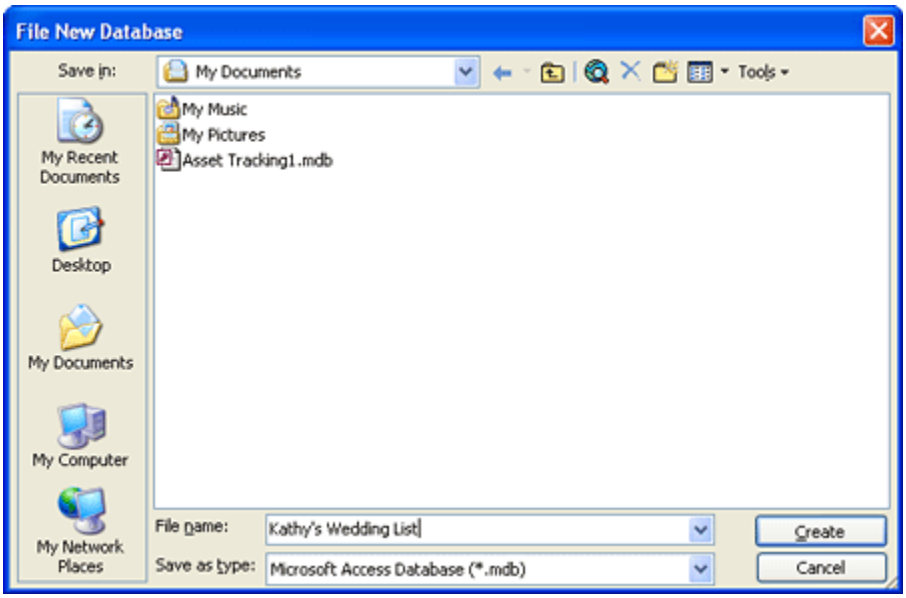
In the final window, you can choose to start the application immediately after the wizard finishes building it. You can also choose to open a special set of Help topics to guide you through using a database application. Select the **Yes, Start the database** option and click **Finish** to create and then start your application. The following illustration shows the Main Switchboard form for the Asset Tracking database application.



Once you use one of the built-in templates, Access lists that template under **Recently used templates** in the **New File** task pane in case you want to use that template again.

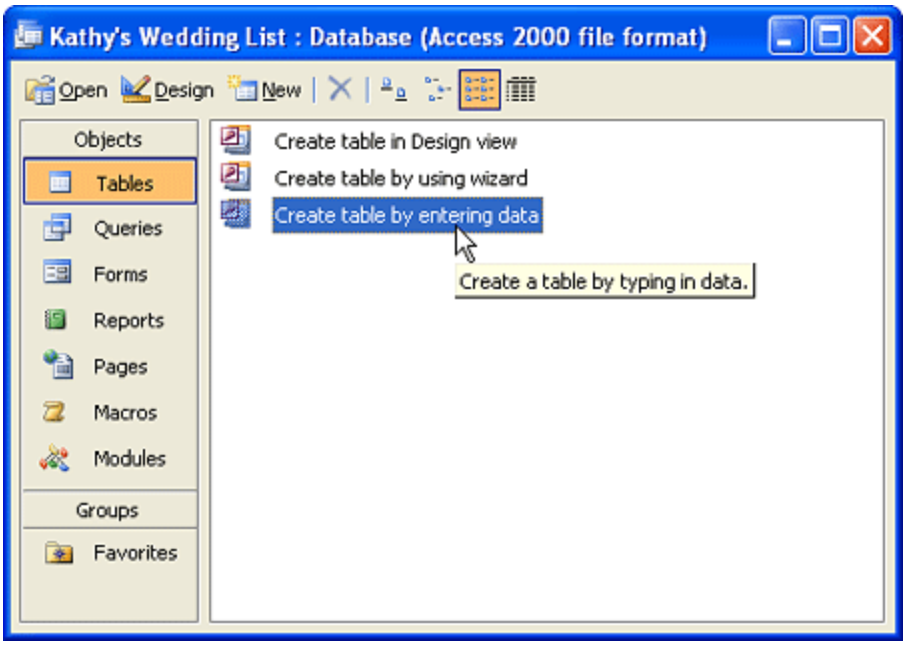
Creating an empty database

To begin creating a new empty database when you start Access, on the **File** menu, click **New**, and then, in the **New File** task pane, click **Blank Database**. This opens the **File New Database** dialog box, shown in the following illustration.



Select the drive and folder you want from the **Save in** drop-down list. In this example, select the **My Documents** folder on My Computer. Finally, go to the **File name** box and type the name of your new database. Access appends an .mdb extension to the file name for you. (Access uses a file with an .mdb extension to store all your database objects, including tables, queries, forms, reports, data access pages, macros, and modules.) For this example, create a new sample database named Kathy's Wedding List to experiment with one way to create a database and tables. Click the **Create** button to create your database.

Access takes a few moments to create the system tables in which to store all the information about the tables, queries, forms, reports, data access pages, macros, and modules that you might create. After Access completes this process, it displays the Database window for your new database, as shown in the following illustration.



When you open a database (unless the database includes special startup settings), Access selects the button under **Objects** that you last chose in the Database window for that database. For example, if the last time you were in this database you worked on queries, Access highlights the **Queries** button on the left and shows you the last query that you selected in the pane on the right the next time you open the database. Each button under **Objects** displays the available objects of that type.

Because this is a new database and no tables or special startup settings exist yet, you see a Database window with no objects defined. The items you see under **Tables** are simply shortcuts to three ways to create a table. The following sections show you how to use each of these.

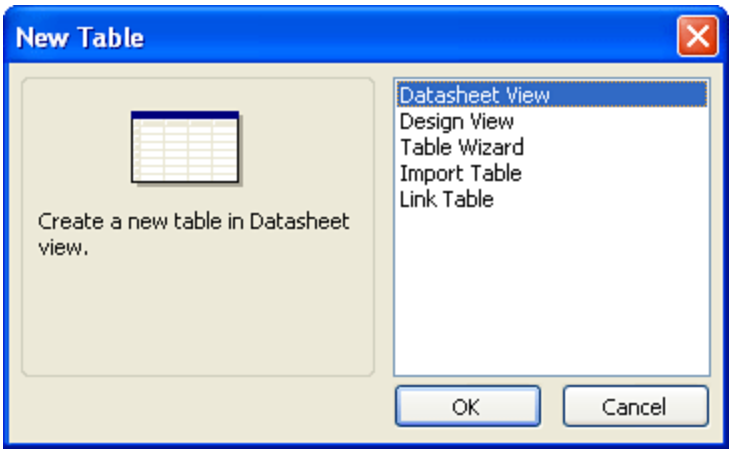
Wait a minute! Why do I see "(Access 2000 file format)" at the top of the Database window? I thought I was working in Access 2003!

Access 2003 fully supports two different file formats to provide you maximum flexibility if your organization still has some users who have Microsoft Access 2000 or Access 2002 installed. By default, any new database you create is in the "lowest common denominator" format —Access 2000. See the section Setting table design options for information about how to change this default.

Back to top

Creating your first simple table by entering data

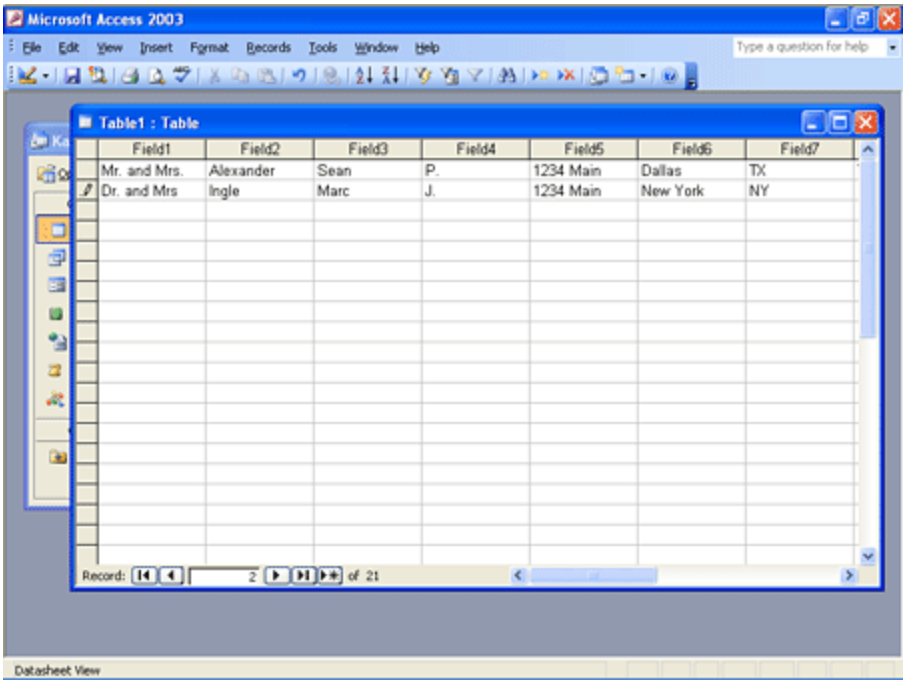
If you've been following along to this point, you should still have your new Kathy's Wedding List database open with the Database window displaying the **Tables** pane, as shown in the previous illustration. (You can also follow these steps in any open database.) Make sure the **Tables** button under **Objects** is selected, and then click the **New** button in the Database window to open the **New Table** dialog box, shown in the following illustration.



Select **Datasheet View** in the list, and then click **OK** to get started. (You can also just double-click **Datasheet View** in the list.) What you see next is an empty datasheet, which looks quite similar to a spreadsheet. You can enter just about any type of data you want — text, dates, numbers, currency. But unlike in a spreadsheet, in a datasheet you can't enter any calculated expressions. However, you can easily display a calculated result using data from one or more tables by entering an expression in a query.

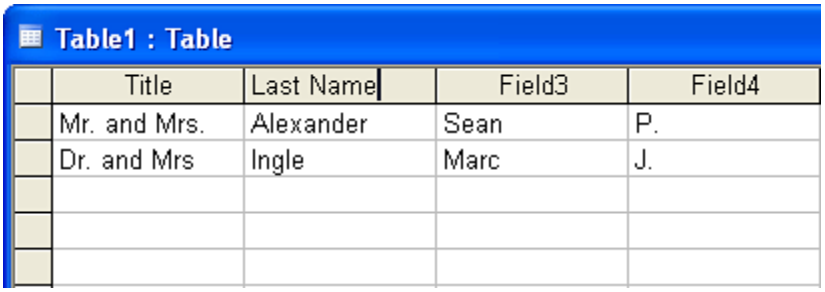
Because we're starting a list of wedding invitees, we'll need columns containing information such as title, first name, middle initial, last name, street address, city, state, postal code, number of guests invited, number of guests confirmed, gift received, and a gift acknowledged indicator. Be sure that each column contains the same type of data in every row. For example, enter the city name in the sixth column for every row.

You can see some of the data entered for the wedding invitee list in the following illustration. When you start to type in a field within a row, Access displays a pencil icon on the row selector at the far left to indicate that you're adding or changing data in that row. Use the TAB key to move from column to column. When you move to another row, Access saves what you typed. If you make a mistake in a particular row or column, you can click the data you want to change and type over it or delete it.



If you create a column of data that you don't want, click anywhere in the column and click **Delete Column** on the **Edit** menu. If you want to insert a blank column between two columns that already contain data, click anywhere in the column to the right of where you want to insert the new column and then click **Column** on the **Insert** menu. To move a column to a different location, click the field name at the top of the column to highlight the entire column, and then click again and drag the column to a new location. You can also click an unselected column and drag your mouse pointer through several adjacent columns to highlight them all. You can then move the columns as a group.

You probably noticed that Access named your columns Field1, Field2, and so forth — not very informative. You can enter a name for each column by double-clicking the column's field name. You can also click anywhere in the column and then click **Rename Column** on the **Format** menu. As you can see in the following illustration, the first name has already been renamed, and the second one is in the process of getting renamed.



After you enter several rows of data, it's a good idea to save your table. You can do this by clicking the **Save** button on the toolbar or by clicking **Save** on the **File** menu. Access displays a **Save As** dialog box, as shown in the following illustration.

Type an appropriate name for your table, and then click **OK**. Access displays a message box warning you that you have no primary key defined for this table and offering to build one for you. If you accept the offer, Access adds a field called ID and assigns it a special data type named AutoNumber that automatically generates a unique number for each new row you add. See the Understanding field data types section for details about the AutoNumber feature. If one or more of the data columns you entered would make a good primary key, click **No** in the message box. In this case, click **Yes** to build a field called **ID** that will serve as the primary key.

Back to top

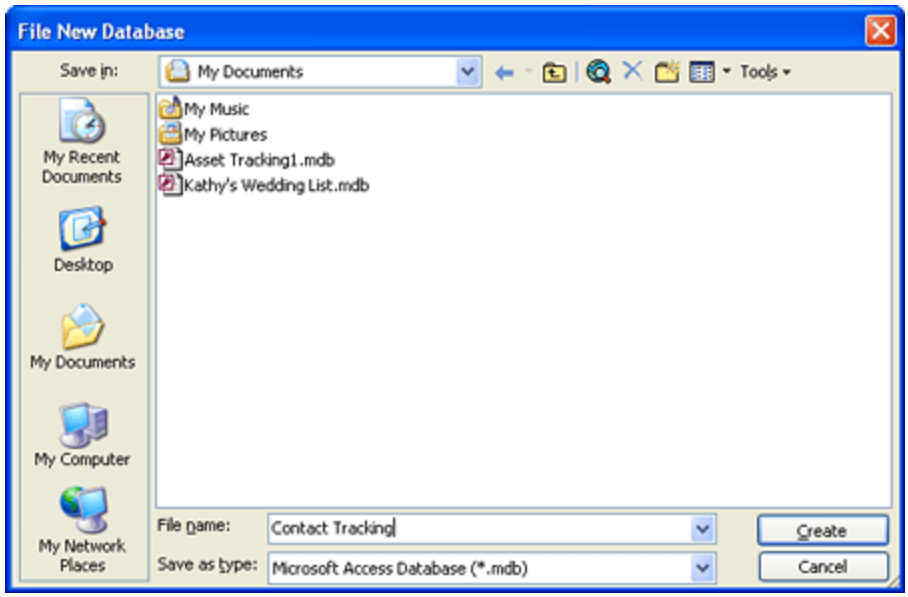
Creating a table by using the Table Wizard

If you look in the Wedding List sample database (WeddingList.mdb), you'll find it very simple, with one main table and a few supporting tables for data such as titles, cities, and postal codes. Most databases are usually quite a bit more

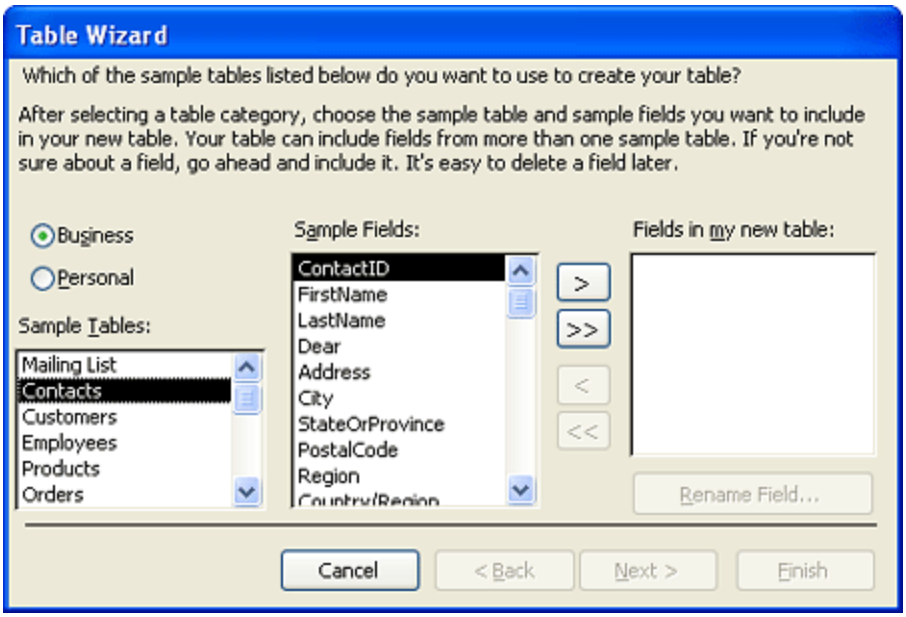
complex. For example, the Housing Reservations sample database contains six main tables, and the LawTrack Contacts sample database contains more than a dozen tables. If you had to create every table by hand, it could be quite a tedious process.

Note The Wedding List and Housing Reservations sample databases are not available for downloading. If you are interested in exploring these databases, you can buy the Microsoft Office Access 2003 Inside Out book.

Fortunately, Access comes with a Table Wizard to help you build many common tables. Let's move on to a more complex task — building tables like those you find in LawTrack Contacts. For this exercise, create a new blank database and give it the name Contact Tracking, as shown in the following illustration.



To build a table by using the Table Wizard, open the Database window, click the **Tables** button, and then click the **New** button. In the **New Table** dialog box, select **Table Wizard** from the list and click **OK**. You can also double-click the **Create table by using wizard** shortcut shown near the top of the Database window. You'll see the opening page of the Table Wizard, shown in the following illustration.



Toward the middle left of the window are two option buttons —**Business** (to select business-oriented tables) and **Personal** (to select personal tables). You can find an entry for a Contacts sample table in the Business category. When you select this table, the wizard displays all the fields from the Contacts sample table in the **Sample Fields** list.

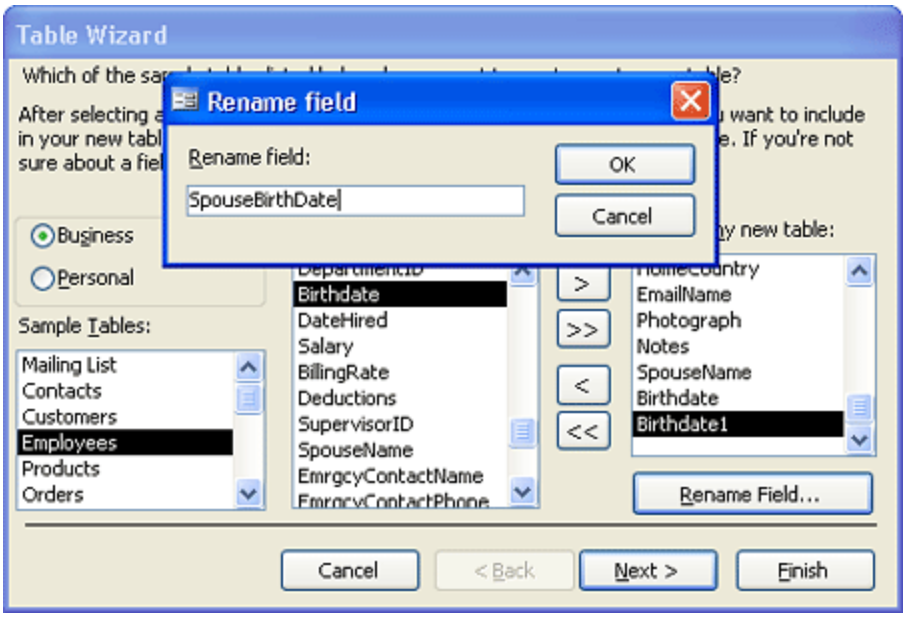
To select a field, click its name in the **Sample Fields** list, and then click the single right arrow (**>**) button to move it to the **Fields in my new table list**. (You can also select a field by double-clicking its name.) You define the sequence of fields in your table on the basis of the sequence in which you select them from the **Sample Fields** list. If you add a field that you decide you don't want, select it in the **Fields in my new table** list and click the single left arrow (**<**) button to remove it. If you want to start over, you can remove all fields by clicking the double left arrow (**<<**) button. If you pick fields in the wrong sequence, you must remove any field that is out of sequence: Click the field above where you want the field inserted, and then select that field again.

Many of the fields in the Contacts sample table are fields you'll need in the Contacts table for your Contact Tracking database. You can pick ContactID, FirstName, LastName, Address, City, StateOrProvince, PostalCode, Country/Region, WorkPhone, WorkExtension, and EmailName directly from the Contacts sample table.

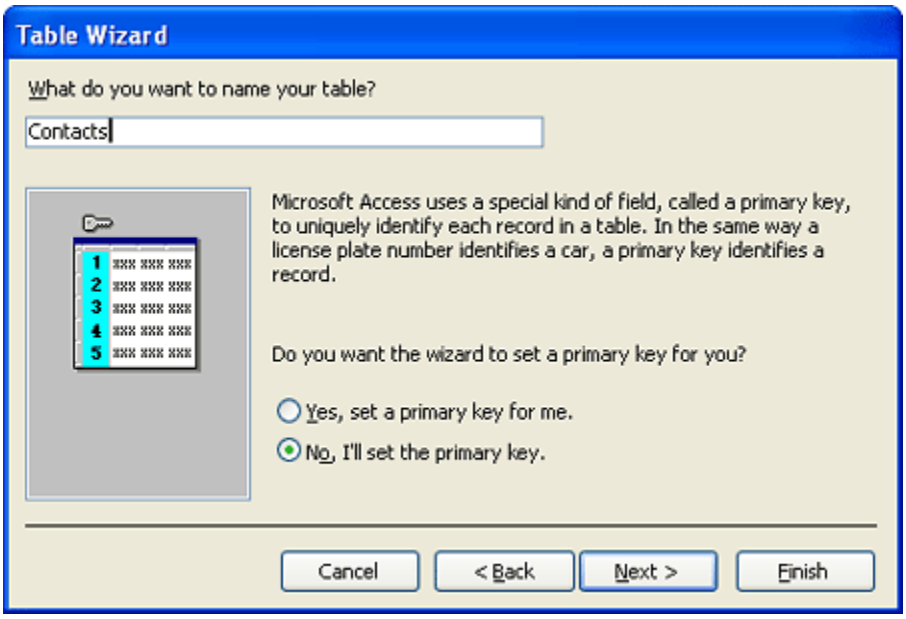
In this Contact Tracking database, you want to track some home contact information, so you need to find a second set of address fields in another sample table to use in your new Contacts table. You also need a field for middle name or middle initial. In the **Sample Tables** list, select the **Employees** sample table. You can see a **MiddleName** field here that you can use, and some more address fields. Click the **LastName** field in the **Fields in my new table** list to indicate that you want to insert a field after **LastName**, and then double-click the **MiddleName** field to insert it after **LastName**. Similarly, click the **WorkExtension** field in the **Fields in my new table** list, and then add **Address** from the **Sample Fields** list (which the wizard will rename Address1 because you already have an Address field), **City**, **StateOrProvince**, **PostalCode**, and **Country/Region** —you'll use these fields to store the contact's home address information. Also, below **EmailName**, add **Photograph**, **Notes**, **SpouseName**, and two copies of the **Birthdate** field from the **Employees** sample table. (The wizard will name the second one Birthdate1 for now.)

Now you need to rename some of the fields. Rename the first Address field to WorkAddress by clicking that field in the **Fields in my new table** list and then clicking the **Rename Field** button. The Table Wizard opens a dialog box to allow you to type a new name. You need to rename the rest of the first set of address fields with a **Work** prefix, and then

correct the names of the second set of fields to use as home address information by adding a **Home** prefix. While you're at it, also remove **/Region** from the **Country/Region** fields. The following illustration shows how to rename the second Birthdate field to SpouseBirthDate. As you can see, it's easy to mix and match fields from various sample tables and then rename the fields to get exactly what you want.



Click the **Next** button to see the window shown in the following illustration.



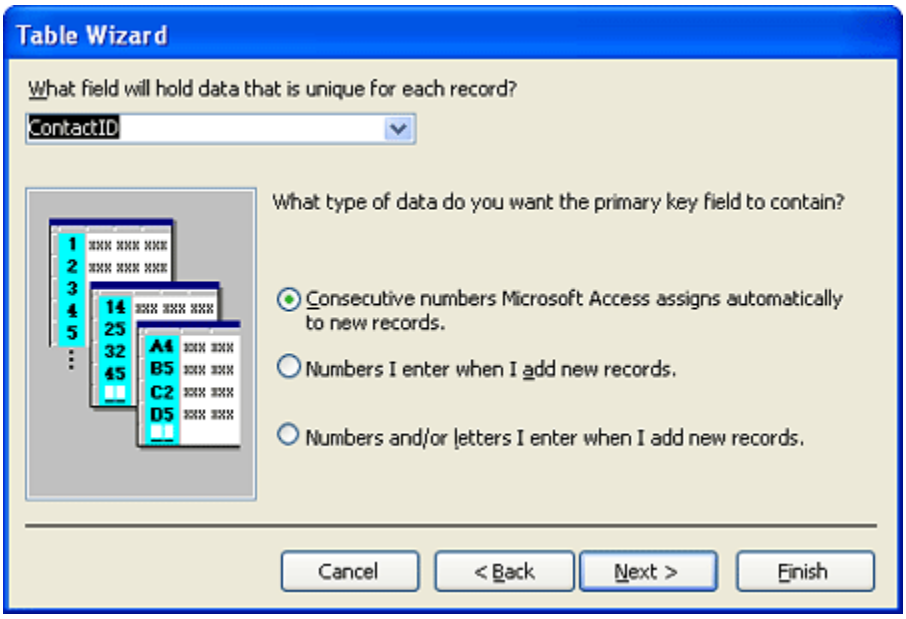
In this window, you can specify a name for your new table. (Because you started by choosing fields from the Contacts sample table, the wizard suggests that you name the table "Contacts," which is just fine.) You can also ask the wizard to set a primary key for you, or you can define your own primary key. In many cases, the wizard chooses the most logical field or fields to be the primary key, but you can override this by clicking the **No, I'll set the primary key** option. If the wizard can't find an appropriate field to be the primary key, it creates a new primary key field that uses a special data type called "AutoNumber." As you'll learn later in this article, the AutoNumber data type ensures that each new row in

your table will have a unique value as its primary key.

Why you should always pick the primary key in the Table Wizard

The Table Wizard might pick the wrong field as the primary key, particularly when you're using the wizard to define a table that contains both an identifier field for the table in addition to fields that you'll use to link to other tables. For example, if you build an Employees table and include both the EmployeeNumber field (that you intend to use as the primary key) and the DepartmentID field (to link later to a Departments table), the wizard incorrectly picks DepartmentID as the primary key of the table. The wizard's default behavior seems to be that it chooses fields that end in "ID" as primary keys over ones that end in "Number."

Go ahead and choose the option to pick your own primary key and then click **Next**. The Table Wizard appears, as shown in the following illustration.



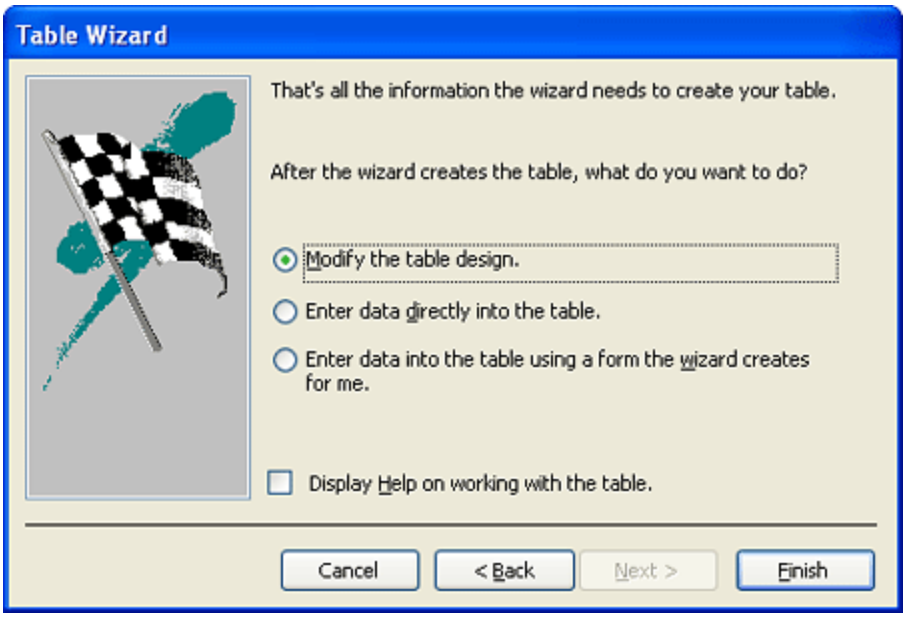
You can open the list at the top of the window to select any field that you defined. This is the Contacts table, so ContactID is the appropriate field to choose. As shown in the previous illustration, if you choose the first option for the type of key, the wizard uses an AutoNumber data type. The second option causes the wizard to create a number field into which you must enter a number for each record, and the third option gives you a text field for each record. In this case, click the first option (**Consecutive numbers Microsoft Access assigns automatically to new records.**).

For details about the different data types that you can assign to fields in your tables, see the section Defining fields.

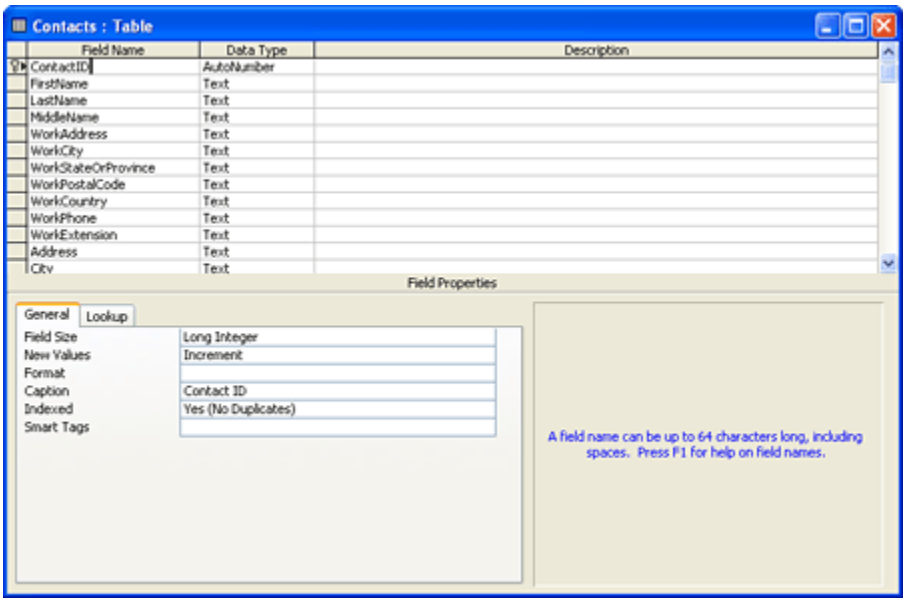
Click the **Next** button to move to the next page of the wizard. If you have other tables already defined in your database, the Table Wizard shows you a list of those tables and tells you whether it thinks your new table is related to any of the existing tables. If the wizard finds a primary key in another table with the same name and data type as a field in your new table (or vice versa), it assumes that the tables are related. If you think the wizard has made a mistake, you can prevent it from creating a relationship (a link) between your new table and the existing table. You'll learn how to define your own relationships between tables later in this article.

Because this is the first and only table in this database, you won't see the **Relationships** page in the Table Wizard.

Instead, the wizard shows you a final page in which you can choose to modify the table design, open it as a datasheet, or call another wizard to build a form to edit your data, as shown in the following illustration.



Select the **Modify the table design** option and click **Finish** to let the wizard build your table. The table will open in Design view, as shown in the following illustration. For now, close the table Design view so that you can continue building other tables that you need.



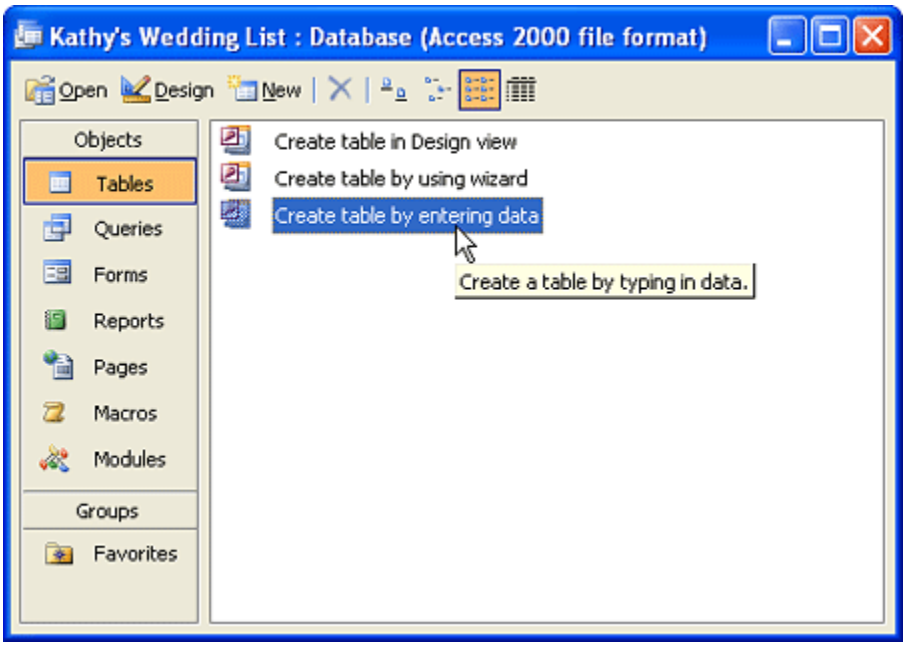
Back to top

Creating a table in Design view

You could continue to use the Table Wizard to build some of the other tables in the Contact Tracking database to mimic

those in LawTrack Contacts. For example, you could use the Customers sample table from the Business sample tables in the Table Wizard to build the Companies table or the Products sample table to get a jump-start on the Products table. However, you'll find it very useful to learn the mechanics of building a table from scratch, so now is a good time to explore Design view and learn how to build tables without using the Table Wizard. You'll also see many additional features that you can use to customize the way your tables (and any queries, forms, or reports built on these tables) work.

To design a new table in a database, open the Database window, as shown in the following illustration.



Click the **Tables** button under **Objects**, and then click the **New** button. Access displays the **New Table** dialog box. Select **Design View** and click **OK**. You can also double-click the **Create table in Design view** shortcut in the Database window. Access displays an empty table in Design view.

In Design view, the upper part of the table displays columns in which you can enter the field names, the data type for each field, and a description of each field. After you select a data type for a field, Access allows you to set field properties in the lower-left area of the table. In the lower-right area of the table is a box in which Access displays information about fields or properties. The contents of this box change as you move from one location to another within the table.

For details about data type values, see the section Understanding field data types.

Back to top

Defining fields

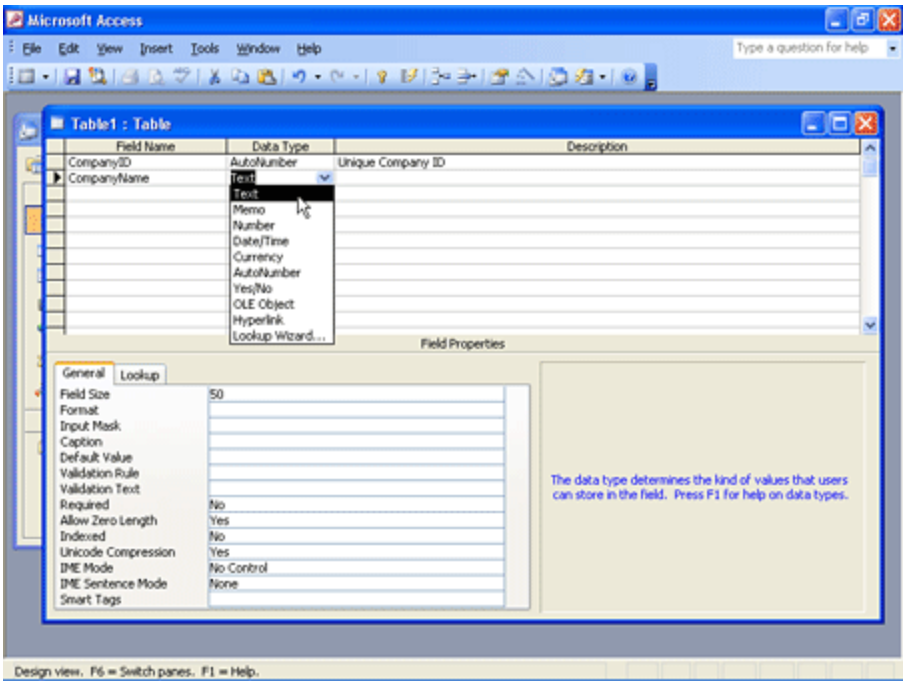
You can now define the fields for the Companies table. Be sure the insertion point is in the first row of the **Field Name** column, and then type the name of the first field, **CompanyID**.

Choosing field names

Access gives you lots of flexibility when it comes to naming your fields. A field name can be up to 64 characters long and can include any combination of letters, numbers, spaces, and special characters except a period (.), an exclamation point (!), an accent grave (`), and brackets ([ ]); however, the name cannot begin with a space and cannot include control characters (ANSI values 0 through 31). In general, you should give your fields meaningful names and should use the same name throughout for a field that occurs in more than one table. You should avoid using field names that might also match any name internal to Access or Visual Basic for Applications (VBA). For example, all objects have a **Name** property, so it's a good idea to qualify a field containing a name by calling it **CustomerName** or **CompanyName**. You should also avoid names that are the same as built-in functions, such as **Date**, **Time**, **Now**, or **Space**. See Access Help for a list of all the built-in function names.

Although you can use spaces anywhere within names in Access, you should try to create field names and table names without embedded spaces. Most SQL databases (SQL database: A database that is based on Structured Query Language (SQL).) to which Access can attach do not support spaces within names. If you ever want to move your application to a client/server environment and store your data in an SQL database, such as Microsoft SQL Server™ 2000 or Oracle, you'll need to change any names in your database tables that contain an embedded space character. Also, table field names propagate into the queries, forms, reports, and data access pages that you design using these tables, so any name you decide to change later in a table must also be changed in all your queries, forms, reports, and data access pages. See Setting table design options for details about options to automatically propagate changes.

Press the TAB key once to move to the **Data Type** column. A button with a down arrow appears on the right side of the **Data Type** column. Here and elsewhere in Access, this type of button signifies the presence of a drop- down list. Click the down arrow or press ALT+DOWN ARROW to open the list of data types, shown in the following illustration.



In the **Data Type** column, you can either type a valid value or select from the list of values in the drop-down list. Select **AutoNumber** as the data type for **CompanyID**.

In the **Description** column for each field, you can enter a descriptive phrase. Access displays this description on the

status bar (at the bottom of the Access window) whenever you select this field in a query in Datasheet view or in a form in Form view or Datasheet view.

Why setting the **Description** property is important

Entering a **Description** property for every field in your table helps document your application. Because Access also displays the description on the status bar, paying careful attention to what you type in the **Description** field can later pay big dividends as a kind of mini-help for the users of your database. Also, since this data propagates automatically, you probably don't want to type something nonsensical or silly. Typing **I don't have a clue what this field does** is probably not a good idea — it will show up later on the status bar!

Tab down to the next line, enter **CompanyName** as a field name, and then choose **Text** as the data type. After you select a data type, Access displays some property boxes in the **Field Properties** area in the lower part of the table. These boxes allow you to set **properties** — settings that determine how Access handles the field —and thereby customize a field. The properties that Access displays depend on the data type you selected; the properties appear with some default values in place, as shown in the previous illustration.

For details about the values for each property, see Setting field properties.

Understanding field data types

Access supports nine types of data, each with a specific purpose. You can see the details about each data type in the following table.

Access data types

Data type	Usage	Size
Text	Alphanumeric data	Up to 255 characters
Memo	Alphanumeric data — sentences and paragraphs	Up to about 1 gigabyte (but controls to display a memo are limited to the first 65,535 characters)
Number	Numeric data	1, 2, 4, 8, or 16 bytes
Date/Time	Dates and times	8 bytes
Currency	Monetary data, stored with 4 decimal places of precision	8 bytes
AutoNumber	Unique value generated by Access for each new record	4 bytes (16 bytes for Replication ID)
Yes/No	Boolean (true/false) data; Access stores the numeric value zero (0) for false, and minus one (-1) for true.	1 bit
OLE Object	Pictures, graphs, or other ActiveX objects from another Windows-based application	Up to about 2 gigabytes
Hyperlink	A link address to a document or file on the World Wide Web, on an intranet, on a local area network (LAN), or on your local computer	Up to about 1 gigabyte

Access also gives you a tenth option, Lookup Wizard, to help you define the characteristics of foreign key fields that link to other tables.

For each field in your table, select the data type that is best suited to how you will use that field's data. For character data, you should normally select the **Text** data type. You can control the maximum length of a **Text** field by using a field property, as explained later. Use the **Memo** data type only for long strings of text that might exceed 255 characters or that might contain formatting characters such as tabs or line endings (carriage returns).

When you select the **Number** data type, you should think carefully about what you enter as the **Field Size** property because this property choice will affect precision as well as length. (For example, integer numbers do not have decimals.) The **Date/Time** data type is useful for calendar or clock data and has the added benefit of allowing calculations in seconds, minutes, hours, days, months, or years. For example, you can find out the difference in days between two **Date/Time** values.

Understanding what's inside the Date/Time data type

Use the **Date/Time** data type to store any date, time, or date and time value. It's useful to know that Access stores the date as the integer portion of the **Date/Time** data type and the time as the fractional portion — the fraction of a day, measured from midnight, that the time represents, accurate to seconds. For example, 6:00:00 AM internally is 0.25. The day number is actually the number of days since December 30, 1899, and can be a negative number for dates prior to that date. When two **Date/Time** fields contain only a date, you can subtract one from the other to find out how many days are between the two dates.

You should generally use the **Currency** data type for storing money values. Currency has the precision of integers, but with exactly four decimal places. When you need to store a precise fractional number that's not money, use the **Number** data type and choose **Decimal** for the **Field Size** property.

The **AutoNumber** data type is specifically designed for automatic generation of primary key values. Depending on the settings for the **Field Size** and **New Values** properties you choose for an AutoNumber field, you can have Access create a sequential or random long integer. You can include only one field using the AutoNumber data type in any table. If you define more than one AutoNumber field, Access displays an error message when you try to save the table.

Use the **Yes/No** data type to hold Boolean (true or false) values. This data type is particularly useful for flagging accounts paid or not paid or orders filled or not filled.

The **OLE Object** data type allows you to store complex data, such as pictures, graphs, or sounds, which can be edited or displayed through a dynamic link to another Windows-based application. For example, Access can store and allow you to edit a Microsoft Word document, a Microsoft Excel spreadsheet, a Microsoft PowerPoint presentation slide, a sound file (.wav), a video file (.avi), or pictures created using the Microsoft Paint or Draw application.

The **Hyperlink** data type lets you store a simple or complex "link" to an external file or document. (Internally, **Hyperlink** is a memo data type with a special flag set to indicate that it is a link.) This link can contain a Uniform Resource Locator (URL) that points to a location on the World Wide Web or on a local intranet. It can also contain the Universal Naming Convention (UNC) name of a file on a server on your local area network (LAN) or on your local computer drives. The link can point to a file that is in Hypertext Markup Language (HTML) or in a format that is supported by an ActiveX application on your computer.

Setting field properties

You can customize the way Access stores and handles each field by setting specific properties. These properties vary according to the data type you choose. The following table lists all the possible properties that can appear on a field's **General** tab in a table's Design view, and the data types that are associated with each property.

Field properties on the General tab

Data type	Options, description
Field Size Property	
Text	Text can be from 0 through 255 characters long, with a default length of 50 characters.
Number	**Byte** A single-byte integer containing values from 0 through 255.
	Integer A 2-byte integer containing values from -32,768 through +32,767.
	Long Integer A 4-byte integer containing values from -2,147,483,648 through +2,147,483,647.
	Single A 4-byte floating-point number containing values from -3.4E+38 through +3.4E+38 and up to seven significant digits.
	Double An 8-byte floating-point number containing values from -1.797E+308 through 1.797E+308 and up to 15 significant digits.
	Replication ID A 16-byte globally unique identifier (GUID).
	Decimal A 12-byte integer with a defined decimal precision that can contain values from -1E+28 through 1E+28. The default precision (number of decimal places) is 0 and the default scale is 18.
AutoNumber	**Long Integer** A 4-byte integer containing values from -2,147,483,648 through +2,147,483,647 when **New Values** is **Random** or from 1 to +2,147,483,647 when **New Values** is **Increment**.
	Replication ID A 16-byte globally unique identifier (GUID).
New Values Property	
AutoNumber only	**Increment** Values start at 1 and increment by 1 for each new row.
	Random Access assigns a random long integer value to each new row.
Format Property	
Text, Memo	You can specify a custom format that controls how Access displays the data. For details about custom formats, see the article Format Property - Text and Memo Data Types, at the Microsoft Developer Network (MSDN) Library.
Number (except Replication ID), Currency, AutoNumber	**General Number (default)** No commas or currency symbols; the number of decimal places shown depends on the precision of the data.
	Currency Currency symbol (from the regional settings in Windows Control Panel) and two decimal places.

	Euro Euro currency symbol (regardless of Control Panel settings) and two decimal places.
	Fixed At least one digit and two decimal places.
	Standard Two decimal places and separator commas.
	Percent Percentage — moves displayed decimal point two places to the right and appends a percentage (%) symbol.
	Scientific Scientific notation (for example, 1.05E+06 represents 1.05 × 106).
	You can specify a custom format that controls how Access displays the data. For details about custom formats, see the article Format Property - Number and Currency Data Types, at the MSDN Library.
Date/Time	**General Date (default)** Combines Short Date and Long Time format (for example, 4/15/2003 5:30:10 PM).
	Long Date Uses Long Date Style from the regional settings in Windows Control Panel (for example, Tuesday, April 15, 2003).
	Medium Date 15-Apr-2003.
	Short Date Uses Short Date Style from the regional settings in Windows Control Panel (for example, 4/15/ 2003).
	Long Time Uses Time Style from the regional settings in Windows Control Panel (for example, 5:30:10 PM).
	Medium Time 5:30 PM.
	Short Time 17:30.
	You can specify a custom format that controls how Access displays the data. For details about custom formats, see the article Format Property - Date/Time Data Type, at the MSDN Library.
Yes/No	**Yes/No (default)**
	True/False
	On/Off
	You can specify a custom format that controls how Access displays the data. For details about custom formats, see the article Format Property - Yes/No Data Type, at the MSDN Library.
Precision Property	
Number, Decimal	You can specify the maximum number of digits allowed. The default value is 18, and you can specify an integer value between 1 and 28.
Scale Property	
Number, Decimal	You can specify the number of decimal digits stored. This value must be less than or equal to the value of the Precision property.
Decimal Places Property	
Number (except Replication ID),	You can specify the number of decimal places that Access displays. The default specification is Auto, which causes Access to display two decimal places for the Currency, Fixed, Standard,

Currency	and Percent formats and the number of decimal places necessary to show the current precision of the numeric value for General Number format. You can also request a fixed display of decimal places ranging from 0 through 15.

Input Mask Property

Text, Number (except Replication ID), Date/Time, Currency	You can specify an editing mask that the user sees while entering data in the field. For example, you can have Access provide the delimiters in a date field such as __/ __/__, or you can have Access format a U.S. phone number as (###) 000-0000. See the section Defining input masks for details.

Caption Property

All	You can enter a more fully descriptive field name that Access displays in form labels and in report headings. **Tip** If you create field names with no embedded spaces, you can use the **Caption** property to specify a name that includes spaces for Access to use in labels and headers associated with this field in queries, forms, and reports.

Default Value Property

Text, Memo, Date/Time, Hyperlink, Yes/No	You can specify a default value for the field that Access automatically uses for a new row if no other value is supplied. If you don't specify a default value, the field will be Null if the user fails to supply a value. See also the Required property.
Number, Currency	Access sets the property to 0. You can change the setting to a valid numeric value. You can also remove the setting, in which case the field will be Null if the user fails to supply a value. See also the Required property.

Validation Rule Property

All (except OLE Object, Replication ID, and AutoNumber)	You can supply an expression that must be true whenever you enter or change data in this field. For example, **<100** specifies that a number must be less than 100. You can also check for one of a series of values. For example, you can have Access check for a list of valid cities by specifying **"Chicago" Or "New York" Or "San Francisco"**. In addition, you can specify a complex expression that includes any of the built-in functions in Access. See the section Defining simple field validation rules for details.

Validation Text Property

All (except OLE Object, Replication ID, and AutoNumber)	You can specify a custom message that Access displays whenever the data entered does not pass your validation rule.

Required Property

All (except AutoNumber)	If you don't want to allow a Null value for the field, set this property to **Yes**.

Allow Zero Length Property

Text, Memo	You can set the field equal to a zero-length string ("") if you set this property to **Yes**. See Nulls and zero-length strings for more information.

Indexed Property

All except OLE Object	You can ask that an index be built to speed access to data values. You can also require that the values in the indexed field always be unique for the entire table. See Adding indexes for details.

Unicode Compression Property

Text, Memo, Hyperlink	As of version 2000, Access stores character fields in an .mdb file using a double-byte (Unicode) character set to support extended character sets in languages that require them. The Latin character set required by most Western European languages (such as English, Spanish, French, or German) requires only one byte per character. When you set Unicode Compression to **Yes** for character fields, Access stores compressible characters in one byte instead of two, thus saving space in your database file. However, Access will not compress Memo or Hyperlink fields that will not compress to fewer than 4,096 bytes. The default for new tables is **Yes** in all countries where the standard language character set does not require two bytes to store all the characters.

IME Mode Property, IME Sentence Mode Property

Text, Memo, Hyperlink	On machines with an Asian version of Windows and appropriate Input Method Editor (IME) installed, these properties control conversion of characters in Kanji, Hiragana, Katakana, and Hangul character sets.

Smart Tags Property

All data types except Yes/No, OLE Object, and Replication ID	Indicates the registered smart tag name and action that you want associated with this field. When the user views this field in a table datasheet, a query datasheet, or a form, Access displays a smart tag available indicator next to the field. The user can click on the indicator and select the smart tag action to perform.

Don’t specify a validation rule without validation text

If you specify a validation rule but no validation text, Access generates a cryptic message that your users might not understand:

"One or more values are prohibited by the validation rule '<your expression here>' set for '<table name.field name>'. Enter a value that the expression for this field can accept."

Unless you like getting lots of support calls, it is recommended that you always enter a custom validation text message whenever you specify a validation rule.

Completing the fields in the Companies table

You now know enough about field data types and properties to finish designing the Companies table in this example. (You can also follow this example using the tblCompanies table from the LawTrack Contacts sample database.) Use the information listed in the following table to design the table shown in the following illustration.

Nulls and zero-length strings

Relational databases support a special value in fields, called a **Null**, that indicates an unknown value. In contrast, you

can set Text or Memo fields to a **zero-length string** to indicate that the value of a field is known but the field is empty.

Why is it important to differentiate Nulls (unknown values) from zero-length strings? Here's an example: Suppose you have a database that stores the results of a survey about automobile preferences. For questionnaires on which there is no response to a color-preference question, it is appropriate to store a Null. You don't want to match responses based on an unknown response, and you don't want to include the row in calculating totals or averages. On the other hand, some people might have responded "I don't care" for a color preference. In this case, you have a known "no preference" answer, and a zero-length string is appropriate. You can match all "I don't care" responses and include the responses in totals and averages.

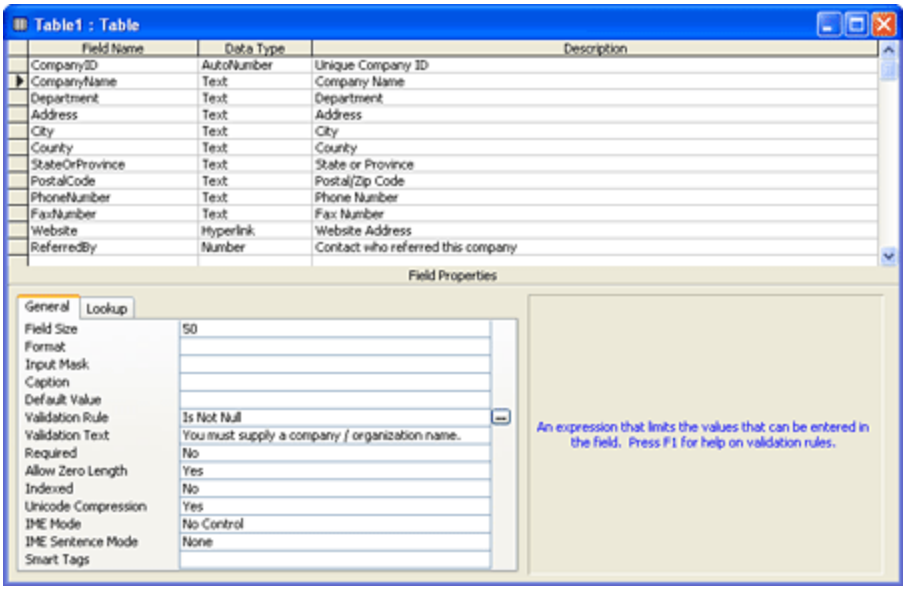
Another example might be fax numbers in a customer database. If you store a Null, it means you don't know whether the customer has a fax number. If you store a zero-length string, you know the customer has no fax number. Access gives you the flexibility to deal with both types of "empty" values.

You can join tables on zero-length strings, and two zero-length strings will compare to be equal. However, for Text, Memo, and Hyperlink fields, you must set the **Allow Zero Length** property to **Yes** to allow users to enter zero-length strings. (**Yes** became the default in Access 2002.) Otherwise, Access converts a zero-length or all-blank string to a Null before storing the value. If you also set the **Required** property of the Text field to **Yes**, Access stores a zero-length string if the user enters either **""** (two double quotes with no space) or blanks in the field.

Nulls have special properties. A Null value cannot be equal to any other value, not even to another Null. This means you cannot join (link) two tables on Null values. Also, the question "Is A equal to B?" when A, B, or both A and B contain a Null, can never be answered "yes." The answer, literally, is "I don't know." Likewise, the answer to the question "Is A not equal to B?" is also "I don't know." Finally, Null values do not participate in aggregate calculations involving such functions as Sum or Avg. You can test a value to determine whether it is a Null by comparing it to the special keyword NULL or by using the **IsNull** built-in function.

Field definitions for the Companies table

Field name	Data type	Description	Field size
CompanyID	AutoNumber	Unique Company ID	
CompanyName	Text	Company Name	50
Department	Text	Department	50
Address	Text	Address	255
City	Text	City	50
County	Text	County	50
StateOrProvince	Text	State or Province	20
PostalCode	Text	Postal/Zip Code	10
PhoneNumber	Text	Phone Number	15
FaxNumber	Text	Fax Number	15
WebSite	Hyperlink	Website address	
ReferredBy	Number	Contact who referred this company	Long Integer



Defining simple field validation rules

To define a simple check on the values that you allow in a field, enter an expression in the **Validation Rule** property box for the field. Access won't allow you to enter a field value that violates this rule. Access performs this validation for data entered in a table in Datasheet view, in an updateable query, or in a form. You can specify a more restrictive validation rule in a form, but you cannot override the rule defined for the field in the table by specifying a completely different rule in the form.

In general, a field validation expression consists of an operator and a comparison value. If you do not include an operator, Access assumes you want an "equals" (=) comparison. You can specify multiple comparisons separated by the Boolean operators **OR** and **AND**.

It is good practice to always enclose text string values in quotation marks. If one of your values is a text string containing blanks or special characters, you must enclose the entire string in quotation marks. For example, to limit the valid entries for a City field to the two largest cities in the state of California, enter **"Los Angeles" Or "San Diego"**. If you are comparing date values, you must enclose the date constants in number sign (#) characters, as in **#01/15/2004#**.

You can use the comparison symbols to compare the value in the field to a value or values in your validation rule. Comparison symbols are summarized in the following table. For example, you might want to ensure that a numeric value is always less than 1000. To do this, enter **<1000**. You can use one or more pairs of comparisons to ask Access to check that the value falls within certain ranges. For example, if you want to verify that a number is in the range of 50 through 100, enter either **>=50 And <=100** or **Between 50 And 100**. Another way to test for a match in a list of values is to use the **IN** comparison operator. For example, to test for states surrounding the U.S. capital, enter **In ("Virginia", "Maryland")**. If all you need to do is ensure that the user enters a value, you can use the special comparison phrase **Is Not Null**.

Comparison symbols used in validation rules

Operator	Meaning

NOT	Use before any comparison operator except IS NOT NULL to perform the converse test. For example, NOT > 5 is equivalent to <=5.
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
=	Equal to
<>	Not equal to
IN	Test for **equal to** any member in a list; comparison value must be a comma-separated list enclosed in parentheses.
BETWEEN	Test for a range of values; comparison value must be two values (a low and a high value) separated by the **AND** operator.
LIKE	Test a Text or Memo field to match a pattern string.
IS NOT NULL	Requires the user to enter a value in the field.

A more friendly way to require a field value

When you set the **Required** property to **Yes** and the user fails to enter a value, Access displays an unfriendly message: "The field '<tablename.fieldname>' cannot contain a Null value because the **Required** property for this field is set to **True**. Enter a value in this field."

It is recommended that you use the **Validation Rule** property to require a value in the field and then use the **Validation Text** property to generate your own more specific message.

If you need to validate a Text, Memo, or Hyperlink field against a matching pattern (for example, a postal code or a phone number), you can use the **LIKE** comparison operator. You provide a text string as a comparison value that defines which characters are valid in which positions. Access understands a number of **wildcard characters**, which you can use to define positions that can contain any single character, zero or more characters, or any single number. These characters are shown in the following table.

LIKE wildcard characters

Character	Meaning
?	Any single character
*****	Zero or more characters; used to define leading, trailing, or embedded strings that don't have to match any specific pattern characters.
#	Any single digit

You can also specify that any particular position in the Text or Memo field can contain only characters from a list that you

provide. You can specify a range of characters within a list by entering the low value character, a hyphen, and the high value character, as in **[A-Z]** or **[3-7]**. If you want to test a position for any characters except those in a list, start the list with an exclamation point (!). You must enclose all lists in brackets ([ ]). You can see examples of validation rules using **LIKE** here.

Validation rule	Tests for
LIKE "#####" or	A U.S. 5-digit ZIP Code
LIKE "#####-####"	A U.S. 9-digit ZIP Code
LIKE "[A-Z]#[A-Z] # [A- Z]#"	A Canadian postal code
LIKE "###-##-####"	A U.S. Social Security Number
LIKE "Smith*"	A string that begins with **Smith**
LIKE "*smith##*"	A string that contains **smith** followed by two numbers, anywhere in the string
LIKE "??00####"	An eight-character string that contains any first two characters followed by exactly two zeros and then any four digits
LIKE "[!0-9BMQ] *####"	A string that contains any character other than a number or the letter B, M, or Q in the first position and ends with exactly four digits

Defining input masks

To assist you in entering formatted data, Access allows you to define an **input mask** for Text, Number (except Replication ID), Date/Time, and Currency data types. You can use an input mask to do something as simple as forcing all letters entered to be uppercase, or as complex as adding parentheses and hyphens to phone numbers. You create an input mask by using the special mask definition characters shown in the following table. You can also embed strings of characters that you want displayed for formatting or stored in the data field.

Input mask definition characters

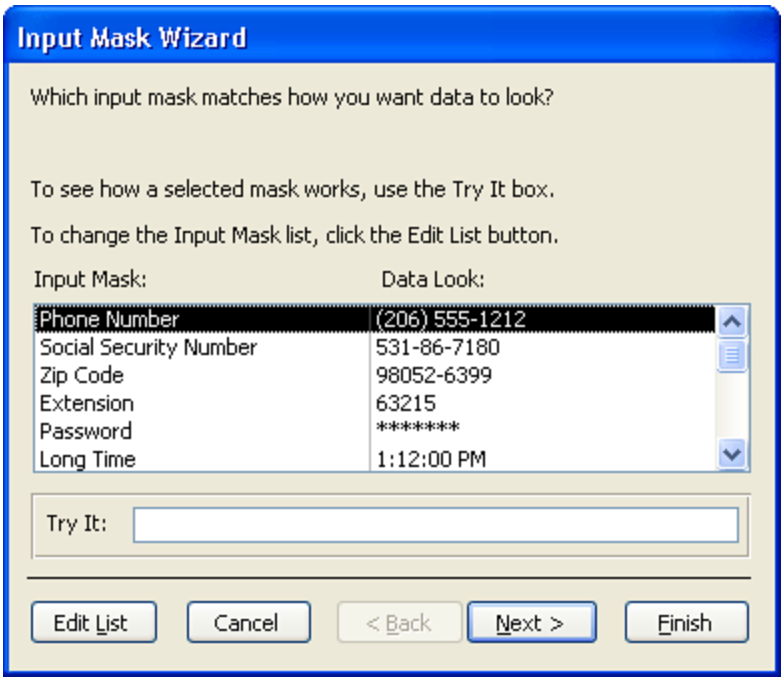
Mask character	Meaning
0	A single digit must be entered in this position.
9	A digit or a space can be entered in this position. If the user skips this position by moving the cursor past the position without entering anything, Access stores nothing in this position.
#	A digit, a space, or a plus or minus sign can be entered in this position. If the user skips this position by moving the cursor past the position without entering anything, Access stores a space.
L	A letter must be entered in this position.
?	A letter can be entered in this position. If the user skips this position by moving the cursor past the position without entering anything, Access stores nothing.
A	A letter or a digit must be entered in this position.

a	A letter or a digit can be entered in this position. If the user skips this position by moving the cursor past the position without entering anything, Access stores nothing.
&	A character or a space must be entered in this position.
C	Any character or a space can be entered in this position. If the user skips this position by moving the cursor past the position without entering anything, Access stores nothing.
.	Decimal placeholder (depends on the setting in the regional settings in Windows Control Panel).
,	Thousands separator (depends on the setting in the regional settings in Windows Control Panel).
: ; - /	Date and time separators (depends on the settings in the regional settings in Windows Control Panel).
<	Converts to lowercase all characters that follow.
>	Converts to uppercase all characters that follow.
!	Causes the mask to fill from right to left when you define optional characters on the left end of the mask. You can place this character anywhere in the mask.
\	Causes the character immediately following to be displayed as a literal character rather than as a mask character.
"literal"	You can also enclose any literal string in double quotation marks rather than use the \ character repeatedly.

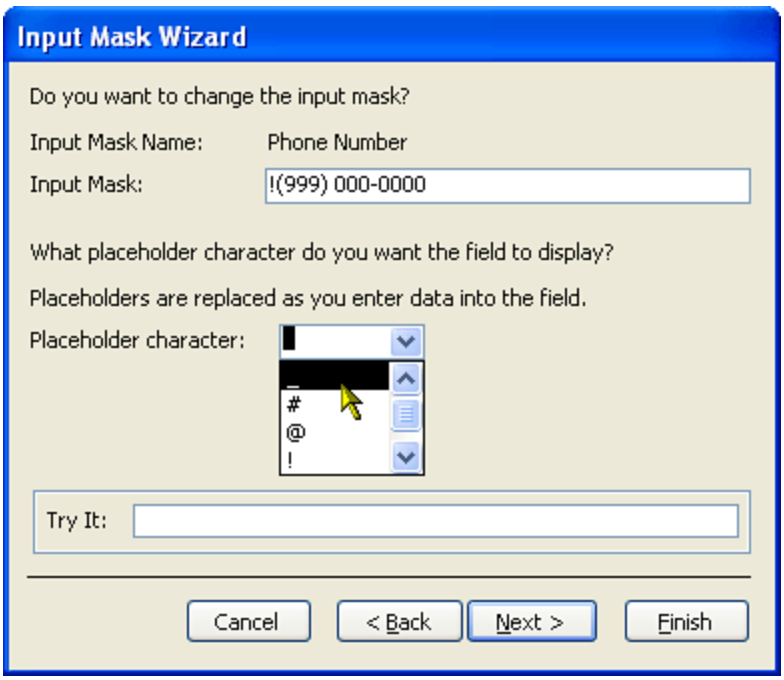
An input mask consists of three parts, separated by semicolons. The first part defines the mask string using mask definition characters and embedded literal data. The optional second part indicates whether you want the embedded literal characters stored in the field in the database. Set this second part to **0** to store the characters or to **1** to store only the data entered. The optional third part defines a single character that Access uses as a placeholder to indicate positions where data can be entered. The default placeholder character is an underscore (_).

Perhaps the best way to learn to use input masks is to take advantage of the Input Mask Wizard. In the Companies table of the Contact Tracking database, the PhoneNumber field could benefit from the use of an input mask. Click the PhoneNumber field in the upper part of the table in Design view, and then click in the **Input Mask** property box in the lower part of the window. You should see a small button with three dots on it (called the **Build** button) to the right of the property box.

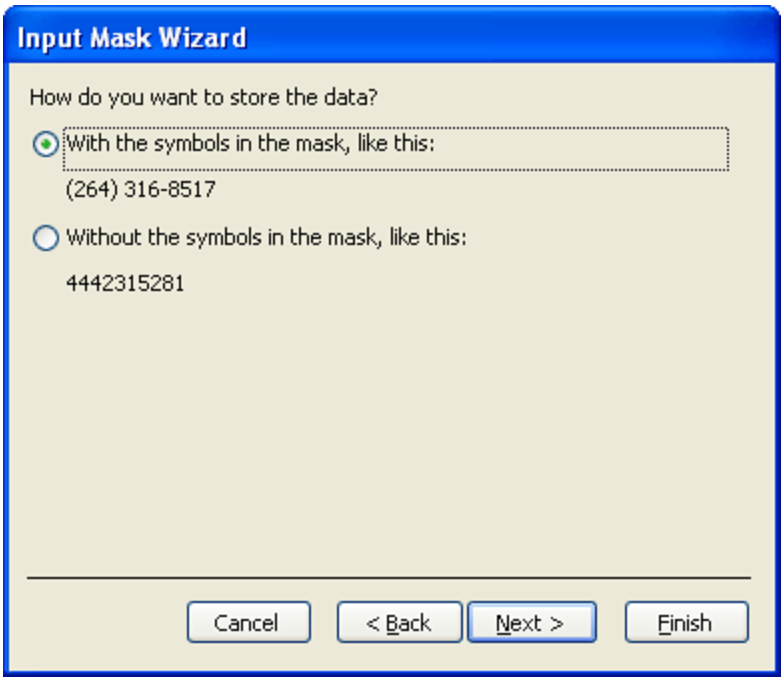
Click the Build button to start the Input Mask Wizard. If you haven’t already saved the table, the wizard will suggest that you do so. Save the table and name it "Companies." When Access warns you that you have not defined a primary key and asks if you want to create a primary key now, click **No**. You’ll define a primary key in the next section. In the first page of the Input Mask Wizard, you have a number of choices for standard input masks that can be generated for you. In this case, click the first one in the list — **Phone Number**, as shown in the following illustration. Note that you can type something in the **Try It** box below the Input Mask selection box to try out the mask.



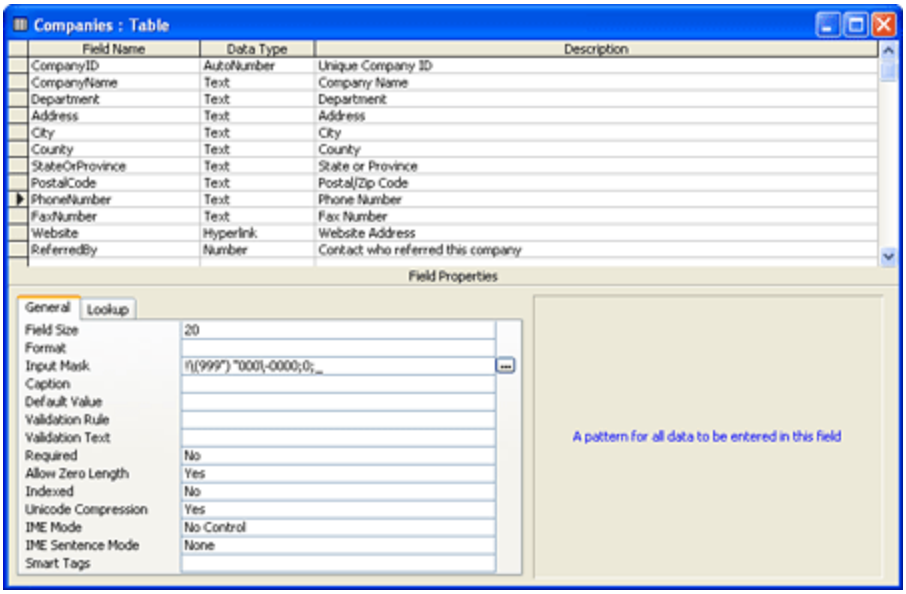
Click the **Next** button to go to the next window. In this window, shown in the following illustration, you can see the mask name, the proposed mask string, a drop-down list from which you select the placeholder character, and another **Try It** box. The default underscore character (_) works well as a placeholder character for phone numbers.



Click **Next** to go to the next window, where you can choose whether you want the data stored without the formatting characters (the default) or stored with the parentheses, spaces, and hyphen separator. In the following illustration, you're indicating that you want the data stored with the formatting characters.



Click **Next** to go to the final window, and then click the **Finish** button in that window to store the mask in the property setting. The following illustration shows the resulting mask in the PhoneNumber field. You'll find this same mask handy for any text field that is meant to contain a U.S. phone number (such as the phone number fields in the Contacts table).



Caution Although an input mask can be very useful to help guide the user to enter valid data, if you define an input mask incorrectly or do not consider all possible valid values, you can prevent the user from entering necessary data. For example, the input mask for a U.S. telephone number you just built would prevent someone from entering a European phone number correctly.

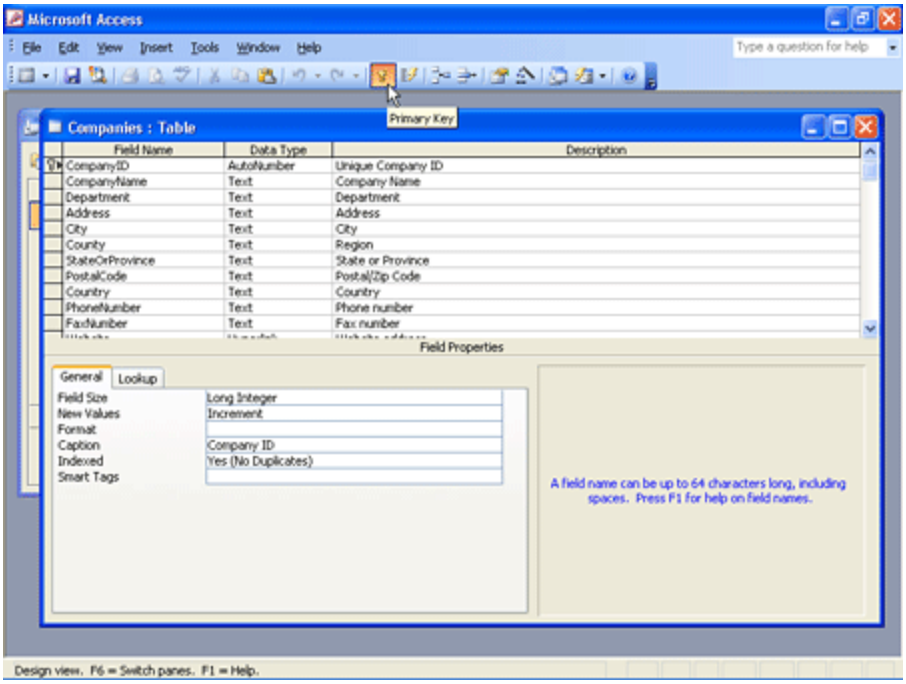
Back to top

Defining a primary key

Every table in a relational database should have a primary key. If you use the procedure outlined in the article Designing your database application you should know what fields must make up the primary key for each of your tables.

Telling Access how to define the primary key is quite simple. Open the table in Design view, and then select the first field for the primary key by clicking the row selector to the left of that field's name. If you need to select multiple fields for your primary key, hold down the CTRL key and click the row selector of each additional field you need.

After you select all the fields you want for the primary key, click the **Primary Key** button on the toolbar or click the **Primary Key** command on the **Edit** menu. Access displays a key symbol to the left of the selected field(s) to acknowledge your definition of the primary key. To eliminate all primary key designations, see the section titled Adding Indexes. When you've finished creating the Companies table for the Contact Tracking database, the primary key should be the CompanyID field, as shown in the following illustration.



Be sure to click the **Save** button on the toolbar to save this latest change to your table definition.

Back to top

Defining a table validation rule

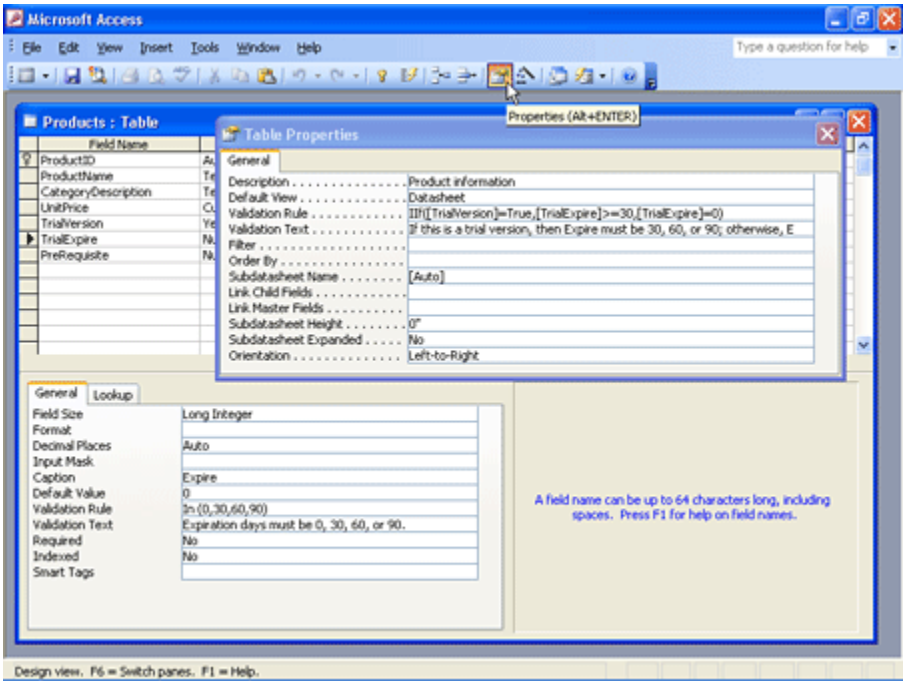
The last detail to define is any validation rules that you want Access to apply to any fields in the table. Although field validation rules get checked as you enter each new value, Access checks a table validation rule only when you save or add a row. Table validation rules are handy when the values in one field are dependent on what's stored in another field. You need to wait until the entire row is about to be saved before checking one field against another.

One of the tables in the Contact Tracking database — Products — needs a table validation rule. Define that table now using the specifications in the following table. Be sure to define ProductID as the primary key and then save the table and name it Products.

Field definitions for the Products table

Field name	Data type	Description	Field size
ProductID	AutoNumber	Unique product identifier	
ProductName	Text	Product description	100
CategoryDescription	Text	Description of the category	50
UnitPrice	Currency	Price	
TrialVersion	Yes/No	Is this a trial version?	
TrialExpire	Number	If trial version, number of days before expiration	Long Integer
PreRequisite	Number	Customer must own this product	Long Integer

To define a table validation rule, be sure that the table is in Design view, and then click the **Properties** button on the toolbar or click the **Properties** command on the **View** menu to open the Table Properties window, as shown in the following illustration.



On the **Validation Rule** line in the Table Properties window, you can enter any valid comparison expression, or you can use one of the built-in functions to test your table field values. In the Products table, you want to be sure that any trial version of the software expires in 30, 60, or 90 days. Zero is also a valid value if this particular product isn't a trial version. As you can see in the previous illustration, a field validation rule for TrialExpire on the **General** tab is already entered to make sure the TrialExpire value is always 0, 30, 60, or 90 — IN(0, 30, 60, 90). But how do you make sure that

TrialExpire is zero if TrialVersion is False, or one of the other values if TrialVersion is True? For that, you need to define a **table-level** validation rule in the Table Properties window.

To refer to a field name, enclose the name in brackets (**[ ]**), as shown in the previous illustration. You'll use this technique whenever you refer to the name of an object anywhere in an expression. In this case, you're using a special built-in function called **Immediate If** (or **IIF** for short) in the table validation rule to perform the test on the TrialExpire and TrialVersion fields. The **IIF** function can evaluate a test in the first argument and then return the evaluation of the second argument if the first argument is true or the evaluation of the third argument if the first argument is false. You must separate the arguments in a function with commas. Note that the **evaluation of the argument—** means the additional tests can be entered, even another **IIF**, in the second and third arguments.

So, the first argument uses **IIF** to evaluate the expression **[TrialVersion] = True** — is the value in the field named TrialVersion True? If this is true (this is a trial version that must have a nonzero number of expiration days), **IIF** returns the evaluation of the second argument. If this is not a trial version, **IIF** evaluates the third argument. Now all you need to do is type in the appropriate test based on the true or false result on TrialVersion. If this is a trial version, the TrialExpire field must be 30 or greater (you'll let the field validation rule make sure it's exactly 30, 60, or 90), so you need to test for that by entering **[TrialExpire] >= 30** in the second argument. If this is not a trial version, you need to make sure TrialExpire is zero by entering **[TrialExpire] = 0** in the third argument. Got it? If TrialVersion is True, then [TrialExpire] >= 30 must be True or the validation rule will fail. If TrialVersion is False, then **[TrialExpire] = 0** must be True. As you might imagine, once you become more familiar with building expressions and with the available built-in functions, you can create very sophisticated table validation rules.

On the fourth line in the Table Properties window, enter the text that you want Access to display whenever the table validation rule is violated. You should be careful to word this message so that the user clearly understands what is wrong. If you enter a table validation rule and fail to specify validation text, Access displays the following message when the user enters invalid data.

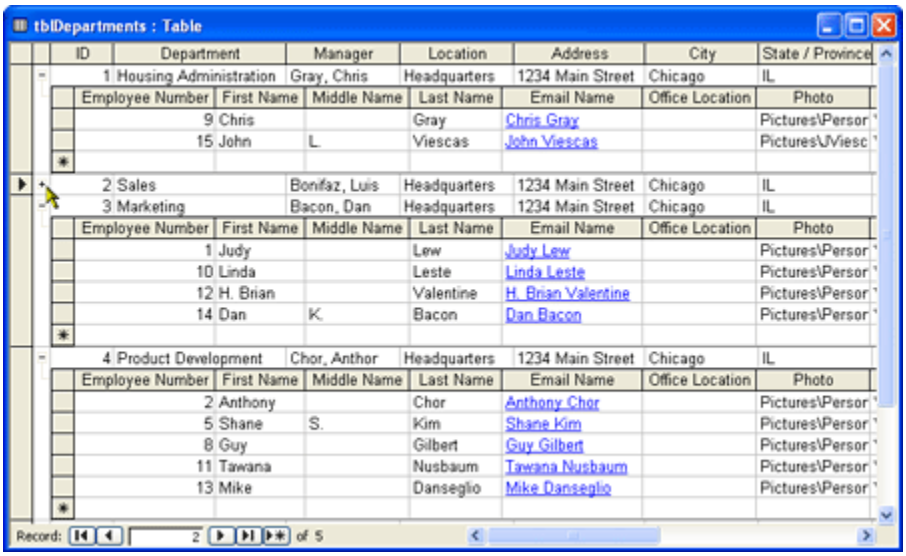
One or more values are prohibited by the validation rule '< your validation rule expression here >' set for '<table name>'. Enter a value that the expression for this field can accept.

Not very pretty, is it? And you can imagine what the user will say about your **IIF** expression!

Back to top

Understanding other table properties

As you can see in the following illustration, Access provides several additional table properties that you can set in Design view. You can enter a description of the table on the first line, and you'll see this description in the Database window if you select the details view. For **Default View**, you can choose from Datasheet (the default) view, PivotTable view, or PivotChart view.



The **Filter** property lets you predefine criteria to limit the data displayed in the Datasheet view of this table. You can use **Order By** to define one or more fields that define the default display sequence of rows in this table when in Datasheet view. If you don't define an **Order By** property, Access displays the rows in primary key sequence.

The next five properties — **Subdatasheet Name**, **Link Child Fields**, **Link Master Fields**, **Subdatasheet Height**, and **Subdatasheet Expanded** — are all related. Access 2000 introduced a feature that lets you see information from related tables when you view the datasheet of a table. For example, in the Contacts Tracking database you have been building, you can set the **Subdatasheet** properties in the definition of Contacts to also show you related information from ContactEvents or ContactProducts. In the Housing Reservations sample database, you can see Departments and their Employees or Employees and their Reservation Requests. The previous illustration shows you the Departments table in Housing.mdb open in Datasheet view. For this table, a subdatasheet is defined to show related Employee information for each department.

Notice the small plus and minus signs at the beginning of each department row. Click on a plus sign to expand the subdatasheet to show related employees. Click on the minus sign to shrink the subdatasheet and show only department information. The following table explains each of the **Table Property** settings that you can specify to attach a subdatasheet to a table.

Table properties for defining a subdatasheet

Property name	Setting	Description
Subdatasheet Name	[Auto]	Creates a subdatasheet using the first table that has a **many** relationship defined with this table.
	[None]	Turns off the subdatasheet feature.
	Table.**<name>** or Query.**<name>**	Uses the selected table or query as the subdatasheet.
Link Child Fields	Name(s) of the foreign key field(s) in the related table, separated by semicolons	Defines the fields in the subdatasheet table or query that match the primary key fields in this table. When you choose a table or query for the **Subdatasheet Name** property, Access uses an available relationship definition or matching field names and data types to

		automatically set this property for you. You can correct this setting if Access has guessed wrong.
Link Master Fields	Name(s) of the primary key field(s) in this table, separated by semicolons	Defines the primary key fields that Access uses to link to the subdatasheet table or query. When you choose a table or query for the **Subdatasheet Name** property, Access uses an available relationship definition or matching field names and data types to automatically set this property for you. You can correct this setting if Access has guessed wrong.
Subdatasheet Height	A measurement in inches	If you specify zero (the default), each subdatasheet expands to show all available rows when opened. When you specify a nonzero value, the subdatasheet window opens to the height you specify. If the height is insufficient to display all rows, a scroll bar appears to allow you to look at all the rows.
Subdatasheet Expanded	Yes or No	If you specify Yes, all subdatasheets appear expanded when you open the table datasheet. No is the default.

Don’t set subdatasheet properties in a table

For a production application, it’s a good idea to set **Subdatasheet Name** in all your tables to **[None]**. First, when Access opens your table, it must not only fetch the rows from the table but also fetch the rows defined in the subdatasheet. Adding a subdatasheet to a large table can negatively impact performance.

Also, any production application should not allow the user to see table or query datasheets because you cannot enforce complex business rules. Any data validation in a table or query datasheet depends entirely on the validation and referential integrity rules defined for your tables because you cannot define any VBA code behind tables or queries.

The last property available in the Table Properties window is **Orientation**. The default in most versions of Access is **Left-to-Right**. In versions that support a language that is normally read right to left, the default is **Right-to-Left**. When you use the **Right-to-Left** value, field and table captions appear right-justified, the field order is right to left, and the tab sequence proceeds right to left.

Back to top

Defining relationships

After you have defined two or more related tables, you should tell Access how the tables are related. You do this so that Access will be able to link all your tables when you need to use them in queries, forms, data access pages, or reports.

You have seen how to build the main subject tables of the Contact Tracking database — Companies, Contacts, and Products. Before you define the relationships in the sample database you’ve been building, you need to create a couple of "linking" tables that define the many-to-many relationships between Companies and Contacts and between Products and Contacts. The following table shows you the fields you need for the Company Contacts table that forms the "glue" between the Companies and Contacts tables.

Field definitions for the Company Contacts table

Field name	Data type	Description	Field size
CompanyID	Number	Company/organization	Long Integer
ContactID	Number	Person within company	Long Integer
Position	Text	Person's position within the company	50
DefaultForContact	Yes/No	Is this the default company for this contact?	

Define the combination of CompanyID and ContactID as the primary key for this table by clicking the selection button next to CompanyID and then holding down the CTRL key and clicking the button next to ContactID. Click the **Primary Key** button on the toolbar to define the key and save the table as CompanyContacts.

The following table shows you the fields you need to define the Contact Products linking table between the Contacts and Products tables.

Field definitions for the Contact Products table

Field name	Data type	Description	Field size
CompanyID	Number	Company/organization	Long Integer
ContactID	Number	Related contact	Long Integer
ProductID	Number	Related product	Long Integer
DateSold	Date/Time	Date product sold	
SoldPrice	Currency	Price paid	

As you might remember from the Designing your database application article, the primary key of the Contact Products table is the combination of CompanyID, ContactID, and ProductID. You can click **CompanyID** to select it, then hold down the SHIFT key while you click **ProductID** (if you defined the fields in sequence) to select all three fields. Click the **Primary Key** button on the toolbar to define the key, and save the table as **ContactProducts**.

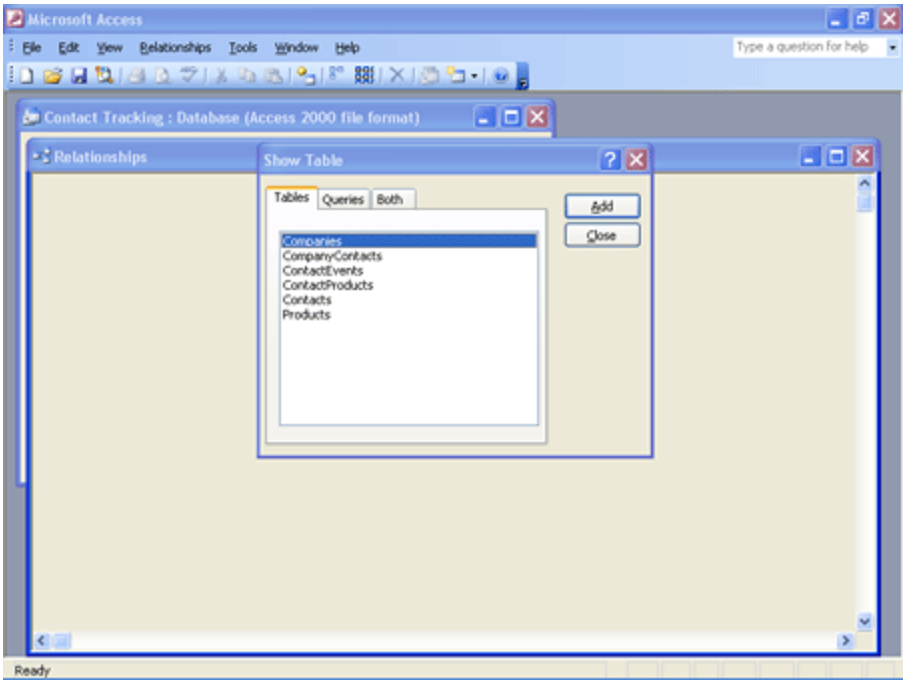
You need one last table, the Contact Events table, to define all the major tables you'll need for Contact Tracking as you designed it in the Designing your database application article. The following table shows you the fields you need. The primary key for this table is the combination of ContactID and ContactDateTime. Note that you took advantage of the fact that a Date/Time data type in Access can store both a date and a time, so you don't need two separate date and time fields.

Field definitions for the Contact Events table

Field name	Data type	Description	Field size
ContactID	Number	Related contact	Long Integer
ContactDateTime	Date/Time	Date and time of the contact	

ContactNotes	Memo	Description of the contact
ContactFollowUpDate	Date/Time	Follow-up date

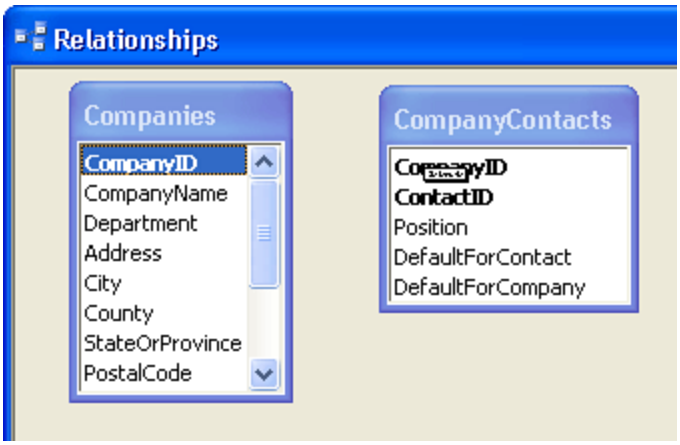
Now you're ready to start defining relationships. To define relationships, you need to return to the Database window by closing any tables that are open and then clicking in the Database window to make it active. Click the **Relationships** command on the **Tools** menu to open the Relationships window. If this is the first time you have defined relationships in this database, Access opens a blank Relationships window and then opens the **Show Table** dialog box, as shown in the following illustration.



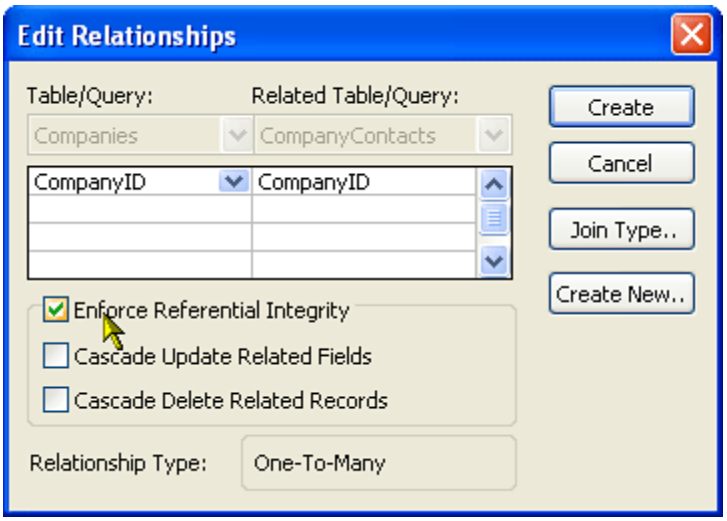
In the **Show Table** dialog box, select each table and click the **Add** button. Click **Close** to close the **Show Table** dialog box.

Defining your first relationship

If you remember the design work you did in the Designing your database application article, you know that a company can have several contacts, and any contact can belong to several companies or organizations. This means that companies have a many-to-many relationship with contacts. You should also recall that defining a many-to-many relationship between two tables requires a linking table. Link the Companies and Contacts tables by defining the first half of the relationship — the one between Companies and the linking table, CompanyContacts. You can see that for the CompanyID primary key in the Companies table, there is a matching CompanyID foreign key in the CompanyContacts table. To create the relationship you need, click in the CompanyID field in the Companies table and drag it to the CompanyID field in the CompanyContacts table, as shown in the following illustration.



When you release the mouse button, Access opens the **Edit Relationships** dialog box, shown in the following illustration.



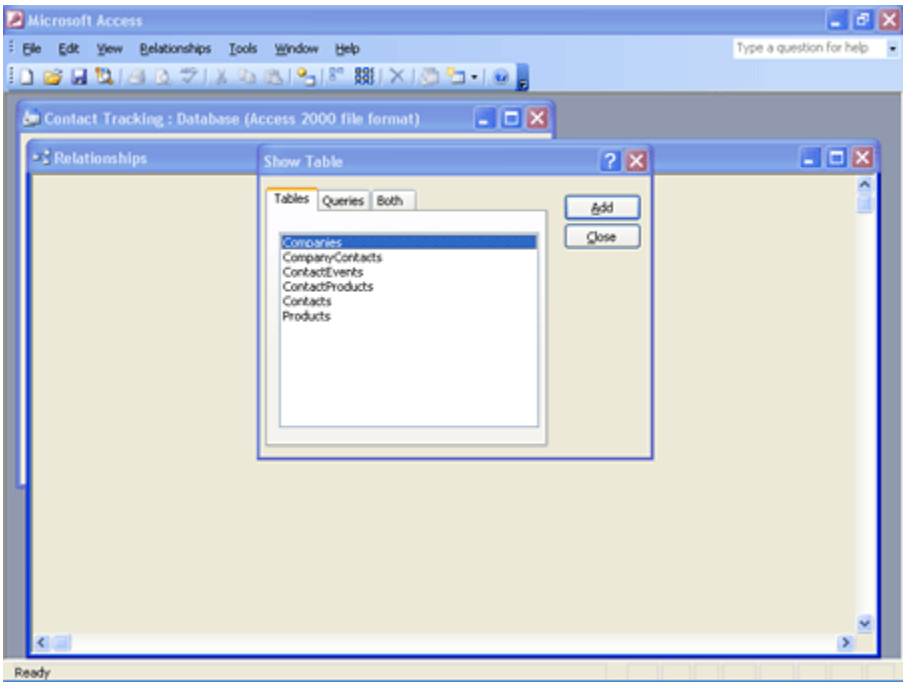
You can also click **Edit Relationship** on the **Relationships** menu to create a new relationship, but you have to fill in the table and field names yourself. Dragging and dropping does some of this work for you.

You'll notice that Access has filled in the field names for you. If you need to define a multiple-field relationship between two tables, use the additional blank lines to define those fields (you'll do that in just a second). Because you probably don't want any rows created in CompanyContacts for a nonexistent company, select the the **Enforce Referential Integrity** check box. When you do this, Access ensures that you can't add a row in the CompanyContacts table containing an invalid CompanyID. Also, Access won't let you delete any records from the Companies table if they have contacts still defined.

Note that after you select the **Enforce Referential Integrity** check box, Access makes two additional options available: **Cascade Update Related Fields** and **Cascade Delete Related Records**. If you select **Cascade Delete Related Records**, Access deletes child rows (the related rows in the **many** table of a one-to-many relationship) when you delete a parent row (the related row in the **one** table of a one-to-many relationship). For example, if you removed a company from the database, Access would remove the related Company Contact rows. In this database design, the CompanyID field has the AutoNumber data type, so it cannot be changed once it is set. However, if you build a table with a primary key that is Text or Number (perhaps a ProductID field that could change at some point in the future), it might be a good

idea to select **Cascade Update Related Fields**. This option requests that Access automatically update any foreign key values in the "child" table (the "many" table in a one-to-many relationship) if you change a primary key value in a "parent" table (the "one" table in a one-to-many relationship).

You might have noticed that the **Show Table** dialog box, shown in the following figure, gives you the option to include queries in addition to tables.



Sometimes you might want to define relationships between tables and queries or between queries so that Access knows how to join them properly. You can also define what's known as an **outer join** by clicking the **Join Type** button in the **Relationships** dialog box and selecting an option in the **Join Properties** dialog box. With an outer join, you can find out, for example, which companies have no contacts or which products haven't been sold.

Avoid defining a relationship with an outer join

It is recommended that you do not define an outer join relationship between two tables. Access automatically links two tables you include in a query design by using the relationships you have defined. In the vast majority of cases, you will want to include only the matching rows from both tables. If you define the relationship as an outer join, you will have to change the link between the two tables every time you include them in a query.

It is also not recommended that you define relationships between queries or between a table and a query. If you have done a good job of naming your fields in your tables, the query designer will recognize the natural links and define the joins for you automatically. Defining extra relationships adds unnecessary overhead in your database application.

Click the **Create** button to finish your relationship definition. Access draws a line between the two tables to indicate the relationship. Notice that when you ask Access to enforce referential integrity, Access displays a **1** at the end of the relationship line, next to the "one" table, and an infinity symbol (∞) next to the "many" table. If you want to delete the relationship, click the line and press the DELETE key.

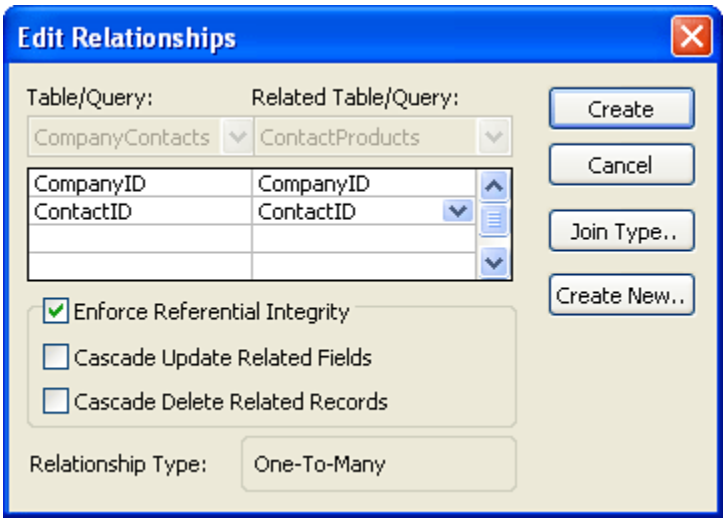
You now know enough to define the additional one-to-many simple relationships that you need. Go ahead and define a

relationship on ContactID between the Contacts and CompanyContacts tables to complete the other side of the many-to-many relationship between Companies and Contacts, a relationship on ContactID between the Contacts and Contact Events tables, and a relationship on ProductID between the Products and ContactProducts tables.

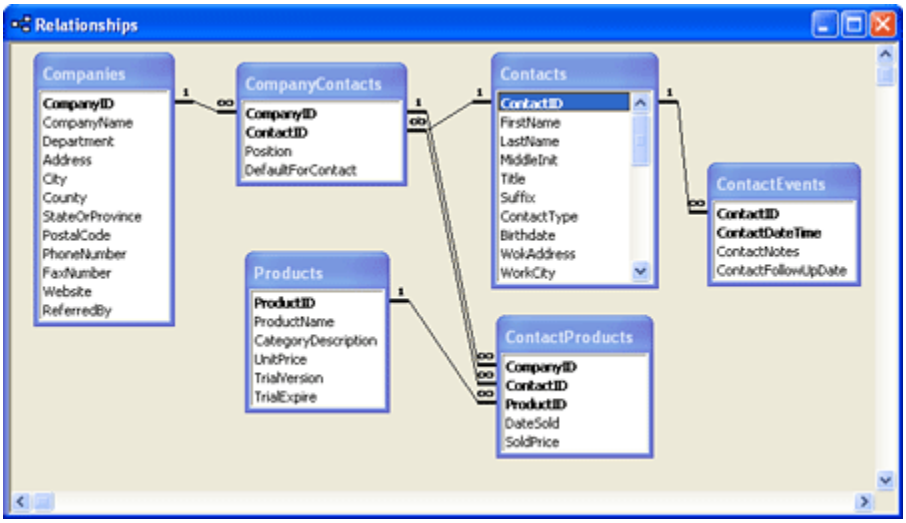
Creating a relationship on multiple fields

There's one last relationship you need to define in the Contact Tracking database between CompanyContacts and ContactProducts. The relationship between these two tables requires multiple fields from each table. You can start by dragging the CompanyID field from the CompanyContacts table to the ContactProducts table. Access opens the **Edit Relationships** dialog box.

When you first see the **Edit Relationships** dialog box for the relationship you are defining between CompanyContacts and ContactProducts, Access shows you only the CompanyID field in the two lists. To complete the relationship definition on the combination of CompanyID and ContactID, you must click in the second line under both tables and select ContactID as the second field for both tables, as shown in the following illustration. Select **Enforce Referential Integrity** as shown and click **Create** to define the compound relationship.



The following illustration shows the Relationships window for all the main tables in your Contact Tracking database. Notice that there are two linking lines that define the relationship between CompanyContacts and ContactProducts.



If you want to edit or change any relationship, double-click the line to open the **Edit Relationships** dialog box again. If you want to remove a relationship definition, click on the line linking two tables to select the relationship (the line appears highlighted) and press the DELETE key. Access presents a warning dialog box in case you are asking it to delete a relationship in error.

Note that once you define a relationship, you can delete the table or query field lists from the Relationships window without affecting the relationships. To do this, click the table or query list header and press the DELETE key. This can be particularly advantageous in large databases that have dozens of tables. You can also display only those tables that you're working with at the moment. To see the relationships defined for any particular table or query, include it in the Relationships window by using the **Show Table** dialog box, and then click the **Show Direct Relationships** button on the toolbar or click **Show Direct** on the **Relationships** menu. To again display all relationships, click the **Show All Relationships** button on the toolbar or click **Show All** on the **Relationships** menu.

When you close the Relationships window, Access asks whether you want to save your layout changes. Click **Yes** to save the relationships you've defined. That's all there is to it.

Additional features in the Relationships window

You can right-click any table in the Relationships window and then click **Table Design** on the shortcut menu to open that table in Design view. You can also click **Print Relationships** on the **File** menu while viewing the **Relationships** window to create a report that prints what you laid out in the window.

Back to top

Adding indexes

The more data you include in your tables, the more you need indexes to help Access search your data efficiently. An **index** is simply an internal table that contains two columns: the value in the field or fields being indexed and the physical location of each record in your table that contains that value.

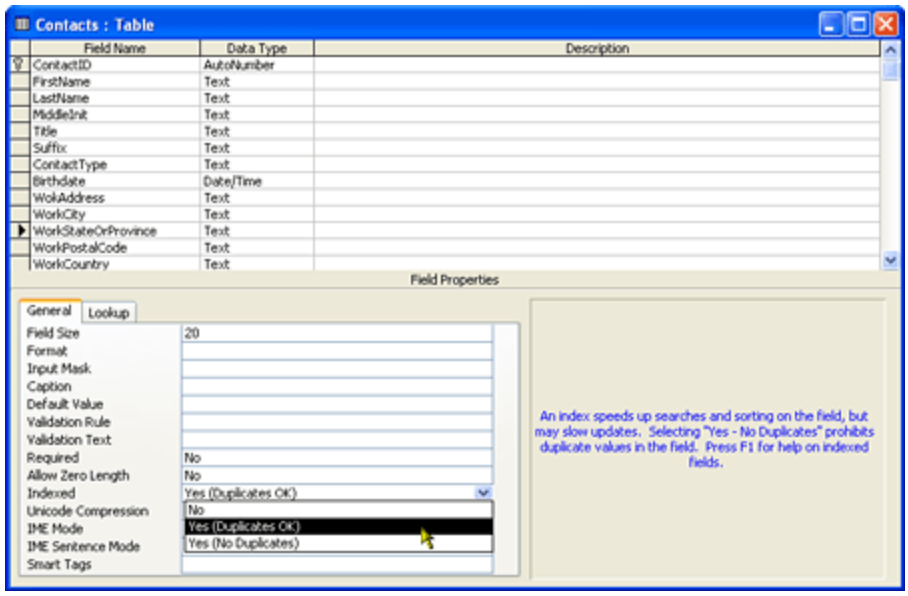
Assume that you often search your Contacts table by city. Without an index, when you ask Access to find all the contacts who live in the city of Chicago, Access has to search every record in your table. This search is fast if your table includes

only a few contacts but very slow if the table contains thousands of contact records collected over many years. If you create an index on the HomeCity field, Access can use the index to find more rapidly the records for the contacts in the city you specify.

Single field indexes

Most of the indexes you'll need to define will probably contain the values from only a single field. Access uses this type of index to help narrow down the number of records it has to search whenever you provide search criteria on the field —for example, **HomeCity = Chicago** or **HomePostalCode = 60633**. If you have defined indexes for multiple fields and provided search criteria for more than one of the fields, Access uses the indexes together (using a technology called Rushmore from Microsoft FoxPro) to find the rows you want quickly. For example, if you have created one index on HomeCity and another on LastName and you ask for **HomeCity = Austin** and **LastName = Viescas**, Access uses the entries in the HomeCity index that equal **Austin** and matches those with the entries in the LastName index that equal **Viescas**. The result is a small set of pointers to the records that match both criteria.

Creating an index on a single field in a table is easy. Open the Contacts table in Design view and select the field for which you want an index —in this case, WorkStateOrProvince. Click the **Indexed** property box in the lower part of the table, and then click the down arrow to open the list of choices, as shown in the following illustration.



When you create a table from scratch (as you did earlier in this article for the Companies table), the default **Indexed** property setting for all fields except the primary key is **No**. If you use the Table Wizard to create a table (as you did for the Contacts table in this article), the wizard indexes fields that might benefit from an index. If you followed along earlier by using the Table Wizard to build the Contacts table, you will find that the wizard built indexes for the EmailName, LastName, WorkPostalCode, and HomePostalCode fields.

If you want to set an index for a field, Access offers two possible Yes choices. In most cases, a given field will have multiple records with the same value —perhaps you have multiple contacts in a particular state or province or multiple products in the same product category. You should select **Yes (Duplicates OK)** to create an index for this type of field. By selecting **Yes (No Duplicates)** you can use Access to enforce unique values in any field by creating an index that doesn't allow duplicates. Access always defines the primary key index with no duplicates because, as you learned in the Designing your database application article, all primary key values must be unique.

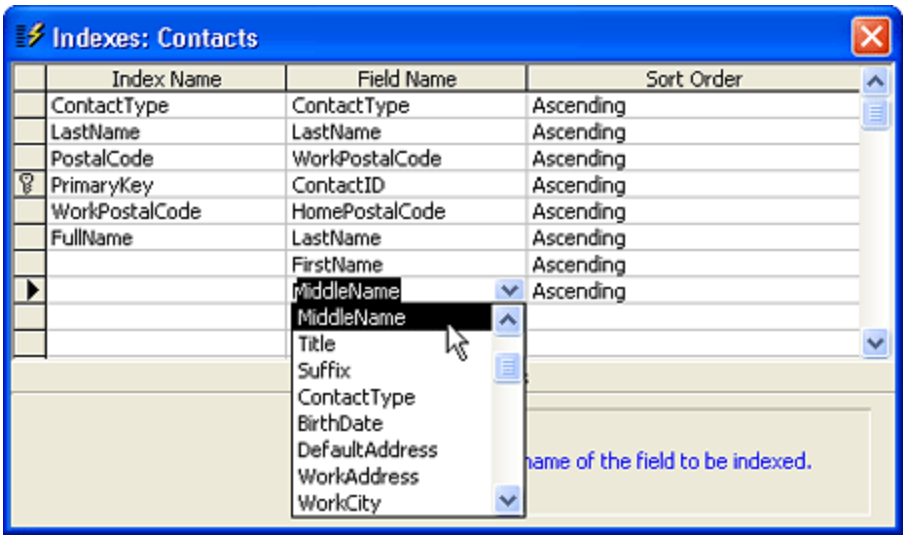
Note You cannot define an index using an OLE Object field.

Multiple-field indexes

If you often provide multiple criteria in searches against large tables, you might want to consider creating a few multiple-field indexes. This helps Access narrow the search quickly without having to match values from two separate indexes. For example, suppose you often perform a search for contacts by last name, first name, and middle name. If you create an index that includes all these fields, Access can satisfy your query more rapidly.

To create a multiple-field index, you must open the table in Design view and open the Indexes window by clicking the **Indexes** button on the toolbar or by clicking the **Indexes** command on the **View** menu. You can see the primary key index and the index that you defined on the WorkStateOrProvince field in the previous section in addition to the indexes defined by the Table Wizard. Each of these indexes comprises exactly one field.

To create a multiple-field index, move the cursor to an empty row in the Indexes window and type a unique name. In this example, you want a multiple-field index using the LastName, FirstName, and MiddleName fields, so "FullName" might be a reasonable index name. Select the LastName field in the Field Name column of this row. To add the other fields, skip down to the next row and select another field without typing a new index name. When you're done, your Indexes window should look like the one shown in the following illustration.



Tip To insert a row in the middle of the list in the Indexes window, right-click in the Index Name column, and then click **Insert Rows** on the shortcut menu. Do not click the **Insert Rows** button on the main toolbar — that inserts rows into the main table design.

You can remove an existing single-field index by changing the **Indexed** property of a field to **No**. The only way to remove a multiple-field index is via the Indexes window. To remove a multiple-field index, highlight the rows (by holding down the CTRL key as you click each row selector) that define the index and then press the DELETE key. Access saves any index changes you make when you save the table definition.

Access can use a multiple-field index in a search even if you don't provide search values for all the fields, as long as you provide search criteria for consecutive fields starting with the first field. Therefore, with the FullName multiple-field index

shown in the previous illustration, you can search for last name; for last name and first name; or for last name, first name, and middle name. There's one additional limitation on when Access can use multiple-field indexes: only the last search criterion you supply can be an inequality, such as >, >=, <, or <=. In other words, Access can use the index shown in previous illustration when you specify searches such as these:

```
LastName = "Smith"
LastName > "Franklin"
LastName = "Buchanan" And FirstName = "Steven"
LastName = "Viescas" And FirstName >= "Bobby"
```

But Access will not use the FullName index shown in the previous illustration if you ask for

```
LastName > "Davolio" And FirstName > "John"
```

because only the last field in the search string (FirstName) can be an inequality. Access also will not use this index if you ask for

```
FirstName = "John"
```

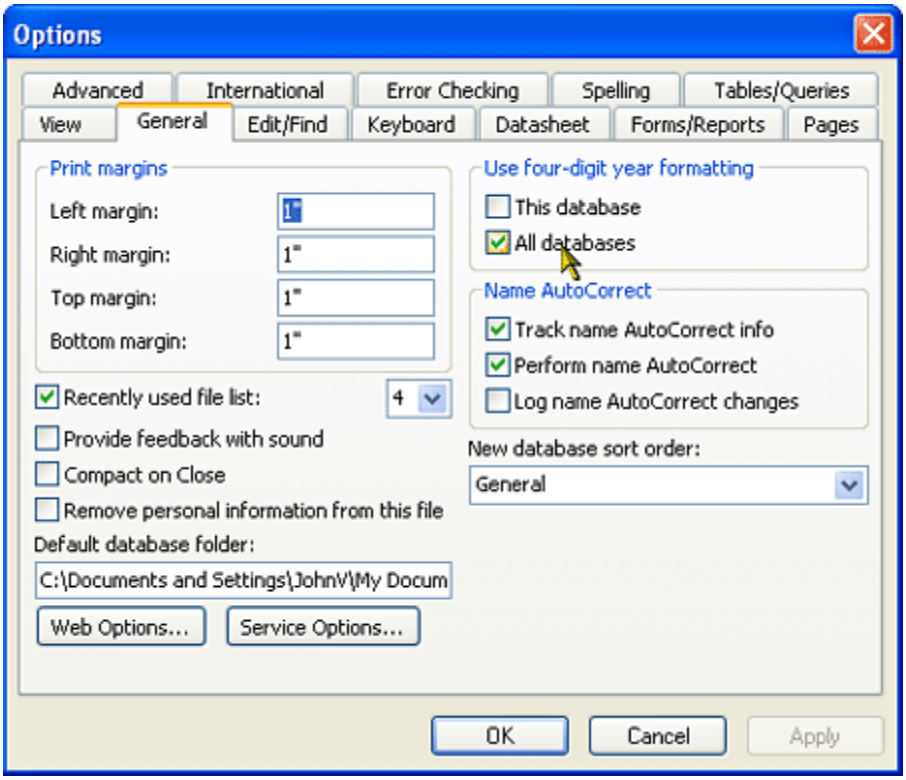
because the first field of the multiple-field index (LastName) is missing from the search criterion.

Back to top

Setting table design options

Now that you understand the basic mechanics of defining tables in your desktop database, it's useful to take a look at a few options you can set to customize how you work with tables in Design view. Close any open tables so that all you see is the Database window. On the **Tools** menu, click **Options** to take a look at all the custom settings offered.

You can find the first options that affect table design on the **General** tab, as shown in the following illustration.



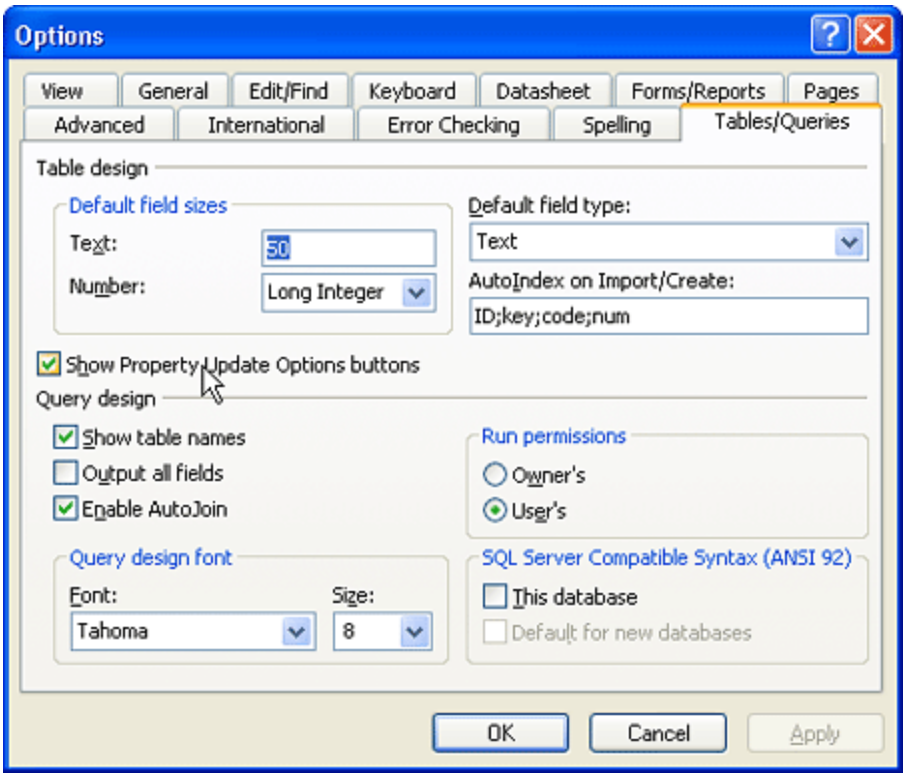
One option that is highly recommended that you use is four-digit year formatting. When you set four-digit year formatting, Access displays all year values in date/time formats with four digits instead of two. This is important because when you see a value (in two-digit medium date format) such as 15 MAR 12, you won't be able to easily tell whether this is March 15, 1912 or March 15, 2012. Although you can affect the display of some formats in your regional settings in Windows Control Panel, you won't affect them all unless you set four-digit formatting in Access.

As you can see in the previous illustration, you have two options under **Use four-digit year formatting** on the **General** tab. If you choose **This database**, the setting creates a property in the database you currently have open and affects only that database. If you choose **All databases**, the setting creates an entry in your Windows registry that affects all databases that you open on your machine.

On this tab, you can also choose a feature that was introduced in Access 2000 called Name AutoCorrect that asks Access to track and correct field name references in queries, forms, and reports. If you choose **Track name AutoCorrect info**, Access maintains a unique internal ID number for all field names. This allows you to use the new Object Dependencies feature. It also allows you to select the next option, **Perform name AutoCorrect**.

If you choose **Perform name AutoCorrect**, when you change a field name in a table, Access automatically propagates the name change to other objects (queries, forms, reports, and pages) that use the field. However, **Track name AutoCorrect info** requires some additional overhead in all your objects, so it's a good idea to carefully choose names as you design your tables so that you won't need to change them later. Finally, if you choose **Log name AutoCorrect changes**, Access logs all changes it makes in a table called AutoCorrect Log. You can open this table to verify the changes made by this feature. (Access doesn't create the table until it makes some changes.)

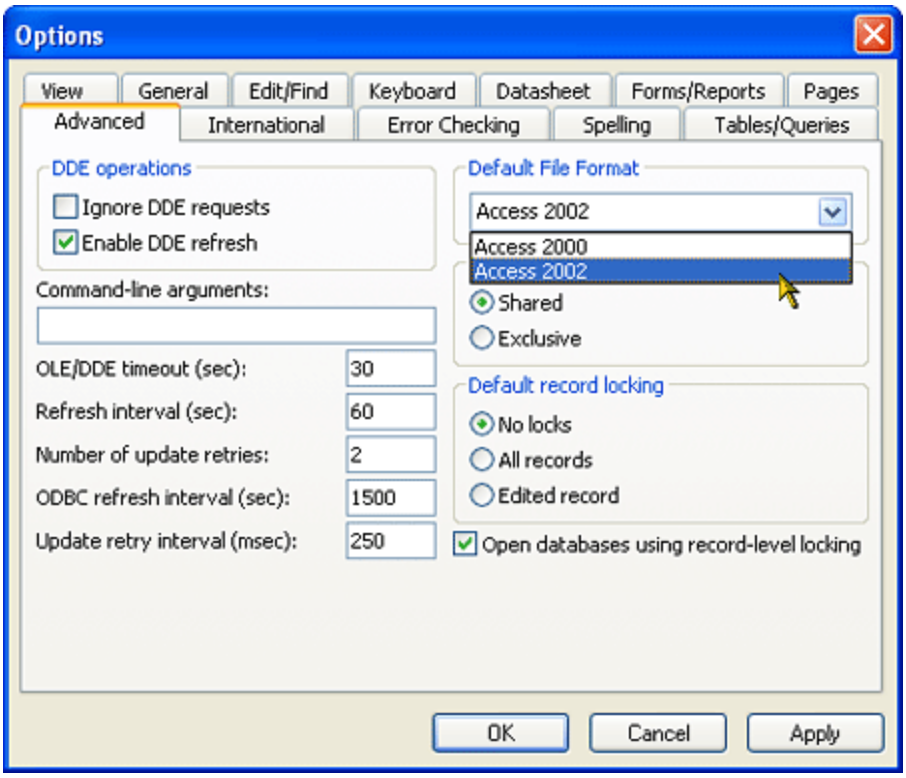
The next tab that contains useful settings that affect table design is the **Tables/Queries** tab. Click that tab to see the settings shown in the following illustration.



In the upper-left corner of this tab, you can set the default field sizes for Text and Number fields. When you choose a data type of Text, Access will automatically fill in the length you choose. When you choose a data type of Number, Access sets the number size to your choice of **Byte**, **Integer**, **Long Integer**, **Single**, **Double**, **Decimal**, or **Replication ID**. In the upper-right corner, you can choose the default data type that Access selects when you type in a new field name in table design and then tab to the Data Type column. Use the **AutoIndex on Import/Create** box to define a list of field name prefixes or suffixes for which Access automatically sets the **Index** property to **Yes (Duplicates OK)**. In the default list, for example, any field that you define with a name that begins or ends with "ID" will automatically have an index.

The last item on this tab that affects how you work in table Design view is **Show Property Update Options buttons**. If you choose this option, a smart tag appears that offers to automatically update related properties in queries, forms, and reports when you change certain field properties in a table design.

You can find the last option that affects how your tables are stored (and, in fact, all objects in your database) on the **Advanced** tab, shown in following illustration.

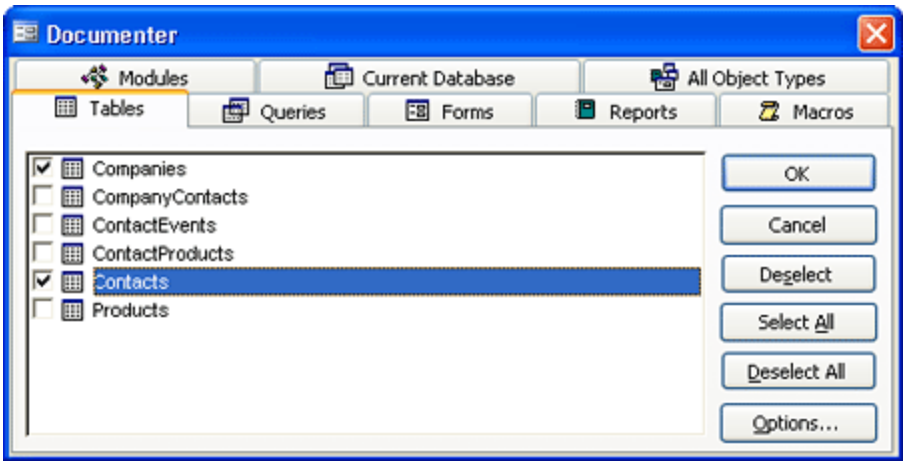


When you create a new database in Access 2003, you actually have a choice of two different file formats. You would think that you would see this option in the **File New Database** dialog box, but it's actually buried away in the **Options** dialog box. You should use the Access 2000 format if others with whom you might share this database are still using Access 2000. Choosing the Access 2002 format ensures maximum compatibility of what you build in Access with future versions of the product.

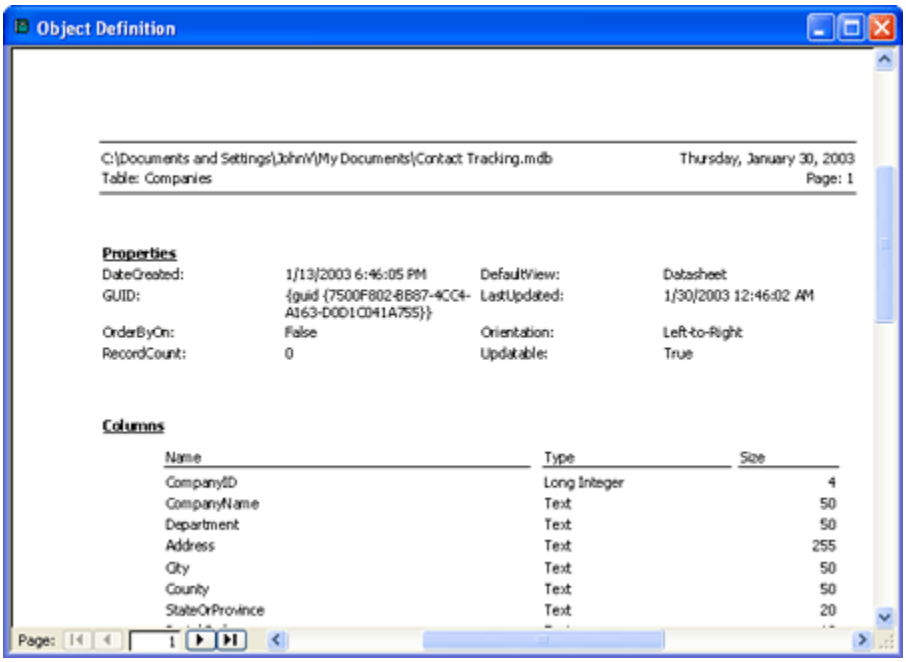
Back to top

Printing a table definition

After you create several tables, you might want to print out their definitions to provide a permanent paper record. You can do this by pointing to **Analyze** on the **Tools** menu, and then clicking **Documenter**. Access displays several options in the **Documenter** dialog box, as shown in the following illustration.



You can select not only the type of object you want to document but also which objects you want to document. Click the **Options** button to select what you want reported. For example, you can ask for the properties, relationships, and permissions for a table; the names, data types, sizes, and properties for fields; and the names, fields, and properties for indexes. Click **OK** in the **Documenter** dialog box to produce the report and view it in Print Preview, as shown in the following illustration.



Back to top

Database limitations

As you design your database, you should keep in mind the following limitations:

- A table can have up to 255 fields.
- A table can have up to 32 indexes.

Important Keep in mind that defining relationships with **Referential Integrity** turned on creates one additional index in each participating table that counts toward the 32-index limit per table.

- A multiple-field index can have up to 10 fields. The sum of the lengths of the fields cannot exceed 255 bytes.
- A row in a table, excluding memo fields and ActiveX objects, can be no longer than approximately 4 kilobytes.
- A memo field can store up to 1 gigabyte of characters, but you can't display a memo larger than 64 kilobytes in a form or a datasheet.
- An ActiveX object can be up to 2 gigabytes in size.
- There is no limit on the number of records in a table, but an Access database cannot be larger than 2 gigabytes. If you have several large tables, you might need to define each one in a separate Access database and then attach them to the database that contains the forms, reports, macros, and modules for your applications.

Back to top