

Lesson 14: Subforms

14.1 Introduction: The advantages of forms within forms

A columnar (single-column) main form with a tabular subform is a natural way of displaying data from tables that participate in a one-to-

many relationship. For example, the form shown in Figure 14.1 is really two forms: the **main form** contains information about a specific order; the **subform** shows all the order details associated with the order.

FIGURE 14.1: A typical form/subform combination.

The main part of the form is columnar (one record per page) and displays information about **Orders**.

Although the **Orders** table is the foundation for the main form, certain information from the **Customers** table is also shown as a convenience. This is achieved by basing the form on a join query.

Product ID	Description	Unit	Qty on hand	Qty ordered	Qty shipped	Actual price	Extended price
57 3826	Spatula, 6" "Cuisipro"	EA	65	12	12	\$4.00	\$48.00
57 3828	Spatula, 8" "Cuisipro"	EA	20	12	12	\$4.25	\$51.00
57 4966	Mixing bowl, 16 qt.	EA	0	6	6	\$12.50	\$75.00
74 4539	Meat tenderizing hammer	EA	0	2	0	\$2.50	\$0.00
74 6083	Spring form pan, 9" non stick	EA	8	5	5	\$7.50	\$37.50
74 6102	Deluxe measuring spoon set	4PC	11	12	11	\$3.50	\$38.50
74 6191	Potato ricer, tinned	EA	0	1	0	\$2.50	\$0.00

The subform is a separate tabular form that displays information from the **OrderDetails** table.

Because a link is established between the main form and the subform, only the order details that belong with **OrderID** = 1 are displayed in the subform.

In the case of an order form with an order details subform, the **OrderID** field provides a link between the two forms. The connection through **OrderID** allows Access to **synchronize** the forms, meaning:

- when you move to another order, only the order details associated with the currently visible order are shown in the subform;
- when you add a new order detail, the foreign key in the **OrderDetails** table is automatically filled in (in fact, there is no need to even show **OrderID** in the subform).

(lesson14-1.avi)

Although you will quickly learn to take a feature such as form/subform synchronization for granted, it is worthwhile to consider what this feature does and what it would take if you had to implement the same feature yourself using a programming language.

14.2 Learning objectives

- understand form/subform synchronization
- create a form/subform combination
- manually link a subform to its main form

14.3 Exercises

Although there are a number of different ways to create a subform within a main form, the recommended procedure is the following:

1. create and save two forms—a columnar main form and a tabular subform (recall the distinction between columnar and tabular forms from [Figure 13.11](#));
2. drag the subform on to the main form; and,
3. verify the linkage between the two forms.

14.3.1 Creating the main form

- ➔ If you did not do so in [Section 10.5](#), create a query that joins the **Orders** and **Customers** tables and save it as **qryOrders**.

(lesson14-2.avi)



Since the purpose of the **qryOrders** query is to facilitate the population of the **Orders** table, ensure you understand the implications of [Section 10.4.2](#). Specifically, ensure that all the field from **Orders** are projected into the query.

- ➔ Use the wizard to create a columnar form based on the **qryOrders** query.

(lesson14-3.avi)

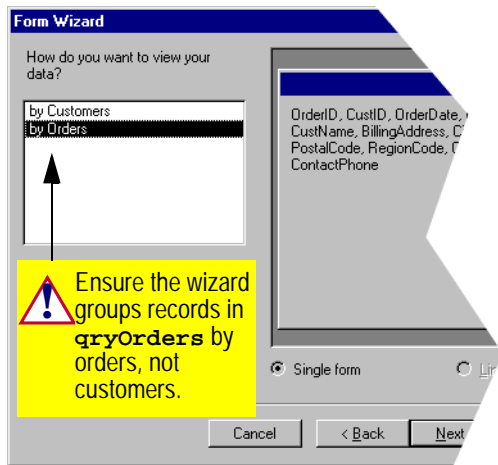


- When you get to the wizard step shown in [Figure 14.2](#), make sure you select the “by Orders” option.



The form wizard recognizes the one-to-many relationship between **Customers** and **Orders** and offers to create a subform for you automatically. We prefer to create the subform ourselves.

FIGURE 14.2: Prevent the form wizard from creating a customer/order subform.



An annoying problem that occurs in ACCESS version 8 is that the form wizard builds

the form on an “ad hoc” query, rather than the saved query you specify. For example, instead of binding the form to **qryOrders** (as specified in Step 1), the wizard sets the form’s **Record Source** property to an embedded SQL statement, as shown in [Figure 14.3](#).

- Verify that the **Record Source** property for your new form is correct. If not, delete the embedded SQL statement and bind the form to the appropriate saved query.




If the form is bound to an embedded SQL query, any changes you make to **qryOrders** (such as sorting by **OrderDate**) will not be reflected in the form.

- Rearrange the fields so that they make efficient use of the top part of the form, as shown in [Figure 14.1](#).
- Save the form as **frmOrders**.

14.3.2 Creating the subform

Once the main form is created, you create a second, tabular form using essentially the same procedure. A subform is like any other form, except that it will ultimately be nested within a main form.

Since you will want to show information about products (such as quantity on hand and description) when you add order details, the subform should be bound to the **qryOrderDetails** query that you completed in [Section 11.5](#).

 If you use the wizard to create a form based on a query and then make significant changes to the query (e.g., add or delete columns), you will also have to make changes to the form. For this

reason, it is good practice to test your queries and make sure you are 100-percent happy with them before creating your forms.

➔ Use the wizard to create a tabular subform based on **qryOrderDetails**, as shown in [Figure 14.4](#) and [Figure 14.5](#)
(Lesson14-4.avi)

Save the form as **sfrmOrderDetails**. Subforms created by the wizard typically require some

FIGURE 14.3: Set the form's **Record Source** property to the desired saved query.

1 Click on the intersection of the vertical and horizontal rules to select the form as a whole.

2 Right-click to bring up the properties sheet for the form (the name of the selected object shows in the title bar of the properties sheet).

3 Note the embedded SQL statement. Use the pull-down list to set the **Record Source** property to **qryOrders**.

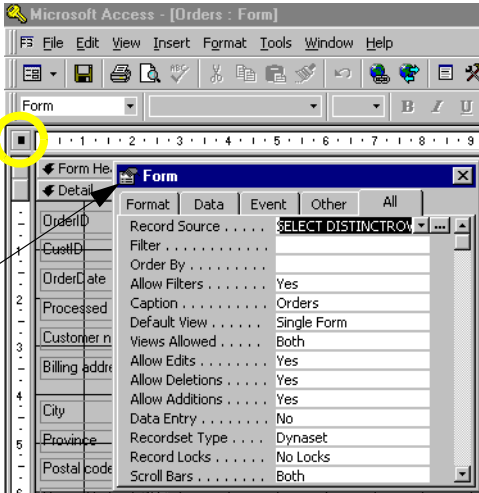
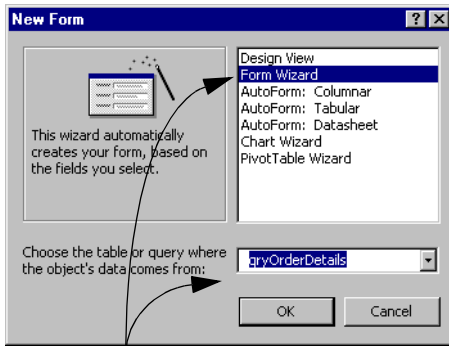
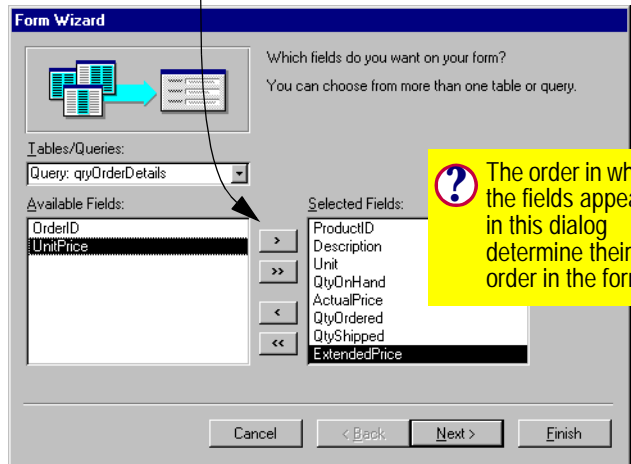


FIGURE 14.4: Use the wizard to create the **OrderDetails** subform (part 1).

1 Select the form wizard and bind the new form to the **qryOrderDetails** table.

2 Use the arrows to select the fields to show in the form. Note that **OrderID** and **UnitPrice** should not be shown.



? The order in which the fields appear in this dialog determine their order in the form.

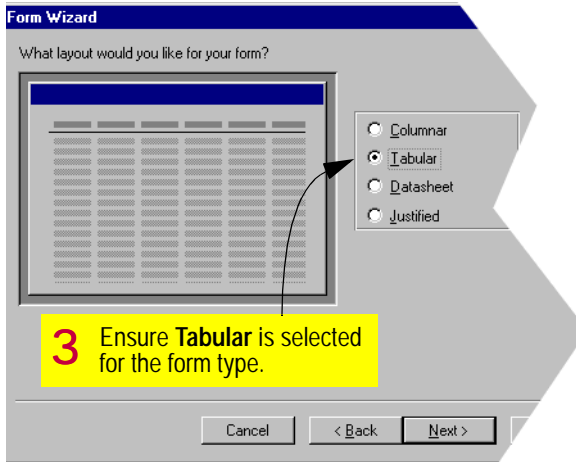
fine tuning in order to reduce the amount of space they occupy. A number of editing issues are highlighted in [Figure 14.6](#).

➔ Switch to design mode and rearrange the fields on your subform as required. You should not waste too much time on this. Set the **Enabled** property to False for all the

fields that the user should not be changing during order entry. For example, there is no reason for the user to change the **Description** or **QtyOnHand** fields when entering order details.



FIGURE 14.5: Use the wizard to create the **OrderDetails** subform (part 2).




Remember, if you also change the **Locked** property to True, the coloring of the text box will be normal instead of grayed-out.

14.3.3 Linking the main form and subform

To create a subform, you drag and drop the icon for the tabular form on to the columnar form.

(lesson14-5.avi)

- Open the main form (**frmOrders**) in design mode.
- Use the **Window** menu to bring the database window into the foreground. Alternatively, you can press the database window icon () on the tool bar.
- Perform the steps shown in [Figure 14.7](#) to drag the subform on to the main form. The result of the drag-and-drop operation is shown in [Figure 14.8](#).



In ACCESS 2000, the blank subform control is replaced by a live window on to the design view of the subform. This feature permits you to edit both your main form and your subform(s) at the same time.



The advantage of the drag-and-drop method of creating a sub form is that the width of the subform control (the white window) is automatically set to equal the width of the tabular subform. Naturally, if you change the size of the tabular subform later on, you will have to manually adjust the size of the subform control on the main form (or delete the subform control and repeat the drag-and-drop procedure).



FIGURE 14.6: Edit the subform to reduce the amount of space it uses.

1 Reduce the horizontal space used by the headings and fields.

? To split the headings into two or more lines, place the cursor at the desired split location and press **Shift-Enter**.

? Drag a "selection box" around multiple objects. The objects can then be moved as a group.

2 Reduce the vertical space by moving the fields up to the "detail band" and bringing the "form footer" band up against the fields (to move a band, drag it using the mouse).

3 Resize the detail area to minimize unused grey space.

14.3.4 Linking forms and subforms manually

If both the form and the subform are based on tables, and if relationships have been declared between the tables, ACCESS normally has no problem determining which fields "link" the information on the main form with the information in the subform. However, when the forms are built on queries, ACCESS has no

relationship information to rely on. As such, you have to specify the form/subform links manually.

Since both the forms created in [Section 14.3.3](#) were built on queries, ACCESS was unable to automatically determine the linking fields.



FIGURE 14.7: Drag the subform on to the main form.

1 Open the main form in design mode.

2 Drag the form footer down to make room for the subform.

3 Position the database window so that the subform's target destination is visible.

4 Drag the subform on to the main form.

➤ Verify the link between the form and the subform by examining the property sheet of the subform control, as shown in Figure 14.8.

If the link fields are incorrect or empty (as they are in this case), you can do one of two things:

1. Type in the name of the link field(s) manually. If more than one field is required

to define the link, separate the field names with semicolons, for example:

DeptName; CourseNo.

2. Use the builder provided in version 7 and greater to have Access make a guess at how the main form and subform are related.

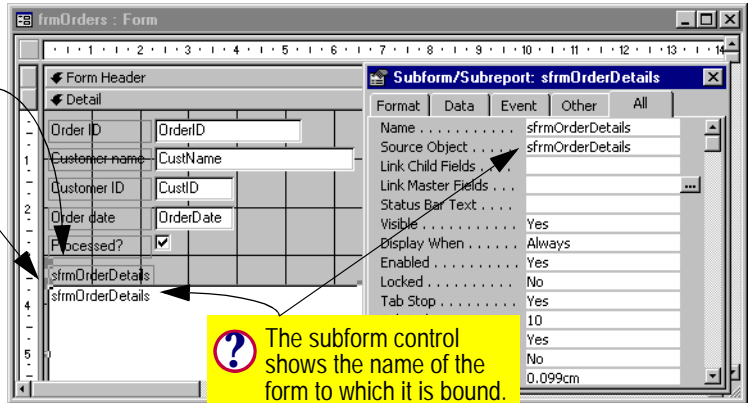


FIGURE 14.8: The drag-and-drop operation creates a subform control.

1 Delete the label associated with the subform control. In this case, the contents of the subform are obvious and the label merely adds clutter.

2 Select the subform control and bring up its property sheet.

? The white area is a "subform control". It is essentially a window through which the tabular subform shows.



? The subform control shows the name of the form to which it is bound.

! The terms "link child field" and "link master field" are specific to the form design tool in MICROSOFT ACCESS. However, you should recognize that "link child field" and "link master field" are identical to "foreign key" and "primary key" respectively. The main form is the master ("one" side) and the subform is the child ("many" side).

➔ Use the builder to specify the link fields between the main form (**frmOrders**) and

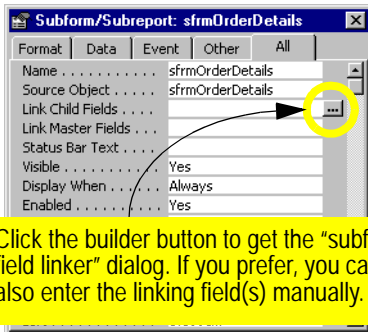
the subform (**sfrmOrderDetails**). This is shown in [Figure 14.9](#).

2 Version 2.0 does not have a builder for linking fields.

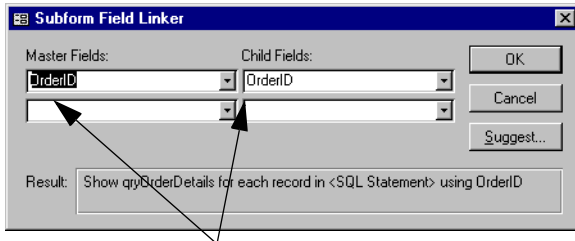
➔ View the resulting form. Notice that as you move from order to order, the number of order details shown in the subform changes. This indicates that form/subform synchronization is working.



FIGURE 14.9: Set the link fields for the form/subform.



1 Click the builder button to get the “subform field linker” dialog. If you prefer, you can also enter the linking field(s) manually.



2 In most cases, ACCESS can identify the correct linking fields. Use the “Subform Field Linker” dialog to verify that the correct fields are used.

14.3.5 Non-synchronized forms

In this section, you will delete the link fields created in [Figure 14.9](#) in order to explore some of the problems associated with non-synchronized forms.



Failure to verify and set link fields is very common mistake for neophyte form designers. As you will see, however, unsynchronized forms can cause all kinds of trouble when it comes time to enter data. *Always* verify your links when creating a subform.

(lesson14-6.avi)

- Return to form design mode, bring up the subform’s property sheet. Delete the link fields (highlight the text and press the **Delete** key).
- View the form. When you attempt to add a new order (see [Figure 14.10](#)) all order details (not just those associated with a particular order) still show in the subform.
- Use the record selector buttons at the bottom of the form to return to the first order and attempt to add a new order detail (use **ProductID** = “51 5012”) to the bottom of the list. This is shown in [Figure 14.11](#).



FIGURE 14.10: Adding a new order using an unsynchronized form/subform.

Product ID	Description	Unit	Qty	Price
573826	Spatula, 6" "Cuisipro"	EA	65	
573829	Spatula, 8" "Cuisipro"	EA	20	\$4.20
574966	Mixing bowl, 16 qt.	EA	7	12.50



Although the order is new (the AutoNumber has not even been set yet) order details associated with other orders show in the subform.



Because the forms are not synchronized, ACCESS cannot automatically set the **OrderID** of the new order detail to that of the current order (in this case 1). By default, **OrderDetails.OrderID** is equal to zero, hence the referential integrity error in [Figure 14.11](#).

➔ Return to design view, re-establish the correct link fields, and save the form.

FIGURE 14.11: Adding a new order using an unsynchronized form/subform.

Product ID	Description	Unit	Qty	Price
515012	Water jug, s.s. w/ice gu...	EA	36	\$0.00

14.3.6 Aesthetic refinements

In this section, you will modify the properties of several form objects (including the properties of the form itself) to make your form more attractive and easier to use.

In [Figure 14.12](#), the basic form created in the previous sections is shown and a number of shortcomings are identified.



FIGURE 14.12: A form/subform in need of some basic aesthetic refinements.

The caption of the form shows the form's name. A more attractive/descriptive caption is required.

The subform control should be made taller so that more order details can be shown on the main form

Product ID	Description	Unit	Qty on hand	Price	Qty ordered	Qty shipped	Extended Price
57 3826	Spatula, 6" "Cuisipro"	EA	65	\$4.00	12	12	\$48.00
57 3828	Spatula, 8" "Cuisipro"	EA	20	\$4.25	12	12	\$51.00
57 4966	Mixing bowl, 16 qt.	EA	7	12.50	6	6	\$75.00

Since the subform control was automatically sized to fit the underlying form, space for a horizontal scroll bar is not necessary.

The navigation buttons for the subform are too easily confused with the navigation buttons for the main form.

14.3.6.1 Changing the form's caption

- In design mode, click on the point at which the horizontal and vertical rulers intersect, as shown in Figure 14.13. This selects the form object.
- Change form's **Caption** property to "Order Entry" or something similar.

14.3.6.2 Eliminating unwanted scroll bars and navigation buttons

Scroll bars and navigation buttons are also form-level properties. However, in this case, you need to modify the properties of the subform rather than the main form.

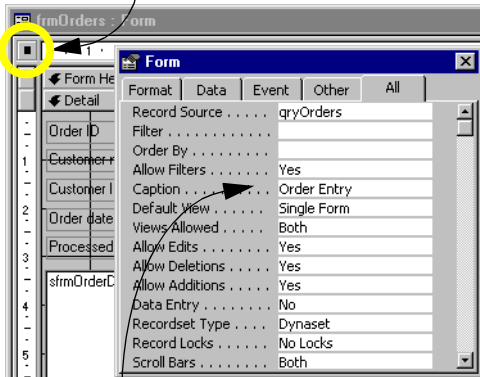
(lesson14-7.avi)

- To quickly open the subform in design mode, double-click the subform control when viewing the main form in design mode (this takes some practice).



FIGURE 14.13: Change the form's caption.

- 1 Click on the square where the vertical and horizontal rulers intersect and right-click to bring up the property sheet for the form.



- 2 Set the **Caption** property to "Order Entry" (or whatever text you want showing in the title bar of your form).



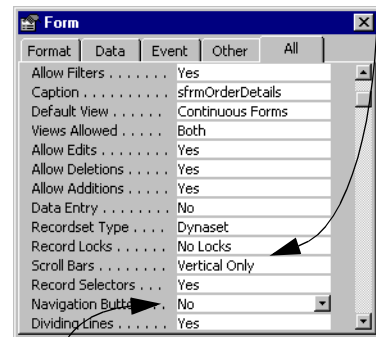
In ACCESS 2000, you can edit the subform directly through the subform control. However, this assumes the subform control's window is large enough that you can see what you are doing. If you prefer to work in a full-sized window, you can use the database window and open the

subform in design view in the normal manner.

- ➔ Bring up the property sheet for the form and scroll down to change its **Scroll Bars** and **Navigation Buttons** properties, as shown in Figure 14.14.

FIGURE 14.14: Change the scroll bars and navigation buttons of the subform.

- 1 Set the **Scroll Bar** property to "Vertical Only".



- 2 Set the **Navigation Buttons** property to "No".

The net result, as shown in Figure 14.1, is a more attractive, less cluttered form.



14.4 Discussion

14.4.1 Add, edit, and view modes

One problem with the order form you have created is that there is nothing to stop users from making changes to the order information once the order has been entered and processed. Clearly, the ability to change the order once the physical goods have been shipped and a paper copy of the invoice has been sent to the customer is dangerous.

This is a common problem: you need to give users the ability to add and edit information when entering transactions. But once the transaction is processed, the information should be cast in stone.¹ There are two basic approaches to achieving this type of functionality:

1. **Two forms** — You can create two forms for your users: one for adding new orders and one for viewing orders that have already

¹ This is especially true for financial systems in which every transaction must be stored “as is”. If a mistake is made in such systems, the transaction cannot be deleted or changed. Instead, the incorrect transaction must be flagged as void, an offsetting transaction must be made to “undo” the error, and the correct transaction must be re-entered. Any other approach would be impossible to audit.

been placed. Clearly, the latter form should be read-only.

2. **A single “smart” form** — If you know something about event-driven programming, you can create a single form that behaves differently depending on whether the transaction showing on the screen has been processed. The advantage of this approach is that you do not have to worry about maintaining two forms.

In either approach, control over the users’ interaction with the data can be controlled by setting some simple form properties.

14.4.2 Form properties for controlling user access

If you examine the property sheet in [Figure 14.14](#), you will see four form-level properties that can be used to control what the user can and cannot do using the form:

1. **Allow Edits**: this is similar to the **Locked** property for fields except that it applies to the form as a whole.
2. **Allow Deletions**: when this property is set to No, a user cannot use the form to delete a record once it is saved.
3. **Allow Additions**: when this property is set to No, users cannot use the form to add new records.



4. **Data Entry**: when this property is set to Yes, the form opens to a new record and prevents the user from moving back to previously saved records. That is, the user can add and edit records created in that particular session. But once the form is closed, all saved records are inaccessible.

Like most properties in ACCESS, these properties can be set via a programming language while the form is in use. As such, it is easy to envision an event-driven programming scenario in which the user presses a button to process an order and, at the same time, the **Allow Edits** property is set to No. You will learn more about event-driven programming in [Lesson 19](#).

14.5 Application to the project

- ➔ Complete the order form you started in this lesson. Since the form is the foundation of your application, you should ensure that basic features (such as form/subform synchronization) are working properly before you continue.



The form you created in this lesson is not quite ready for entering orders. In the remaining tutorials, you will be enhancing its functionality with bound controls and adding a feature to automatically update inventory when the order is complete. For

now, the form is only really useful for viewing any orders you have added manually.