Reports

We are going to construct an invoice to show details of a particular order for a particular customer.

It's a good idea to plan what your reports should look like before you start trying to program them. The invoice needs to show customer details, together with the selected order and order lines. It will also need to have calculations for VAT, possible discount and totals.

Invoice ***Invoice Date :*** Friday, May 08, 1998

Customer **Richard Jones**

Address **University of Greenwich**
Wellington Street
Woolwich
London
SE18 PF **Phone** **0181 331**

Order Date *26-Nov-1997* ***Order_Id*** *4* ***Payment Method*** *C*

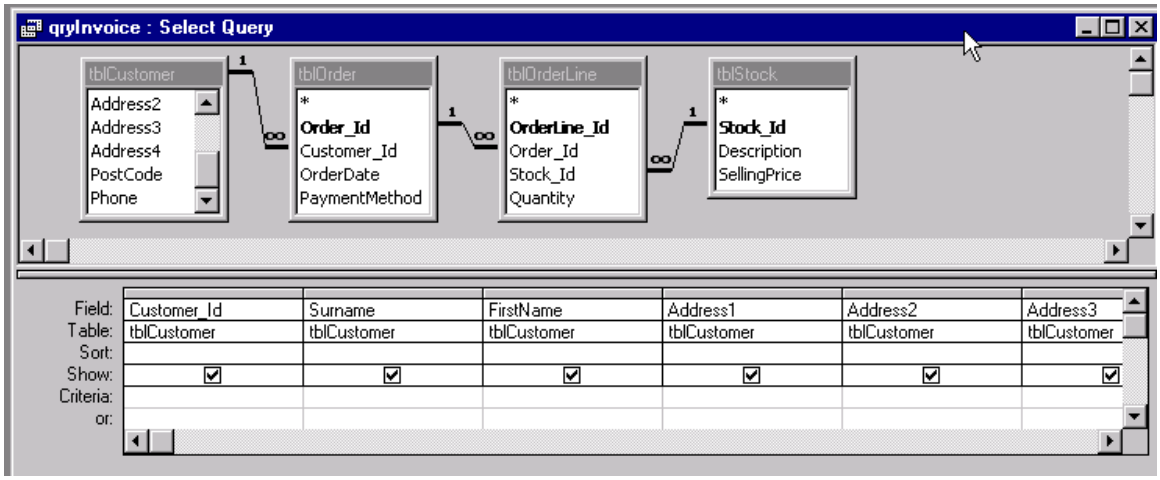
Quantity	***Description***	***Selling Price***	***Value***
1	Leaded Window	£35.00	£35.00

Total Value	£35.00
Vat @ 17.5%	£6.13
Total Value inc. VAT	£41.13
10% Cash Discount	£4.11
Total Payable	£37.01

Friday, May 08, 1998 ***Page 1 of 1***

You first need to create a query pulling together all the stored data that is needed.

Here is an extract showing the tables required:



Include the following fields

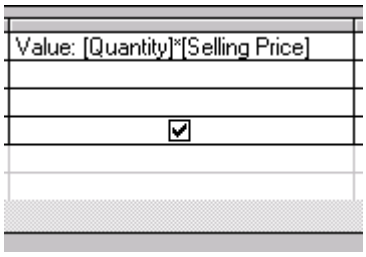
tblCustomer
- All fields

TblOrder
- Order_Id
- OrderDate
- PaymentMethod

tblOrderLine
- Quantity

tblStock
- Description
- SellingPrice

Create a calculated field, showing the value of an order:



Save the query as qryInvoice and check it works by running it.

Creating the Report

Practical 7---Reports

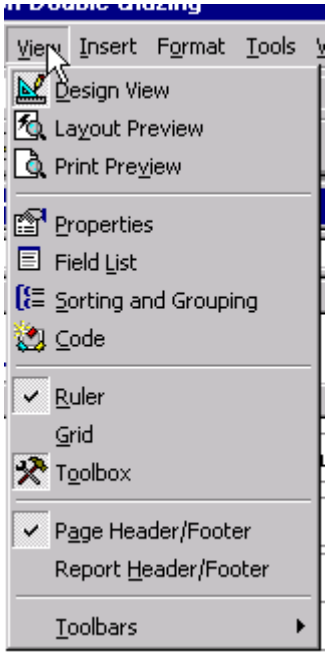
You can use the Report wizard to create the report for you, but for a complex layout like this, it is often better to start with a blank report.

In the Database window, click the Reports tab, New and Design View.

Choose qryInvoice as the query where the report will get its data.

You will now get a blank report.

Open the View menu and make sure that the Page Header is on and the Report Header off.

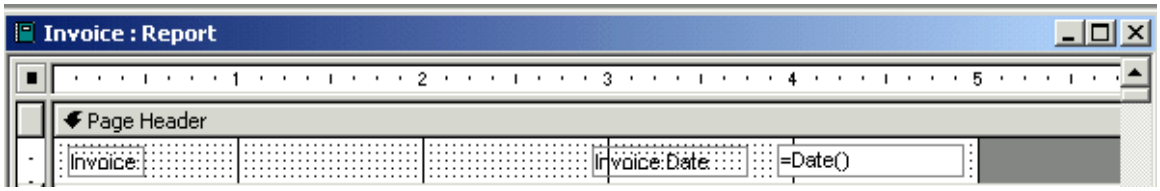


In the Page Header:

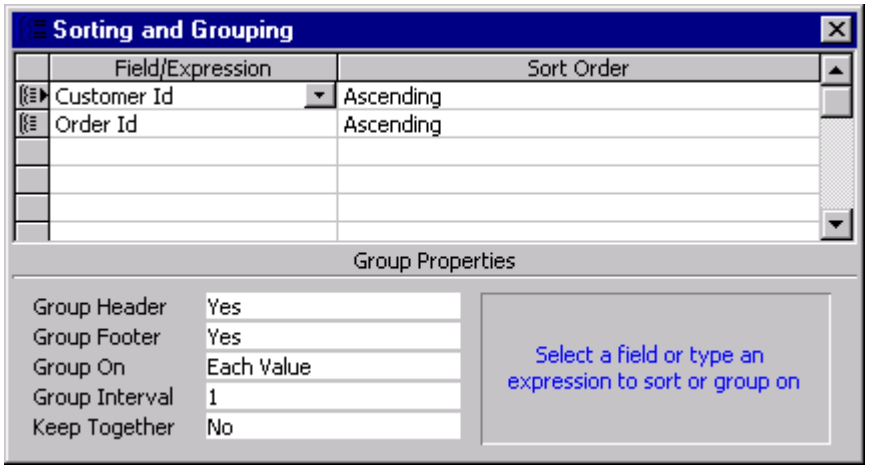
Add a Label
Caption *Invoice*

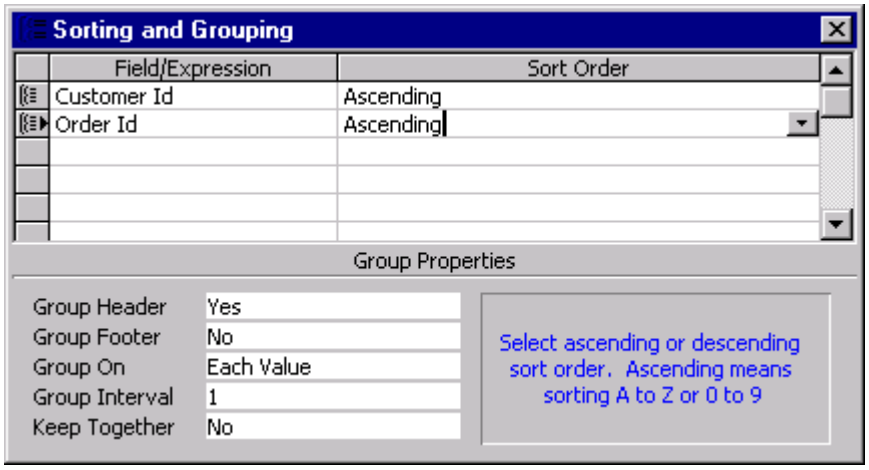
Add a text box
Control Source = *Date()*
Format General Date

Change the text box label to *Invoice Date*

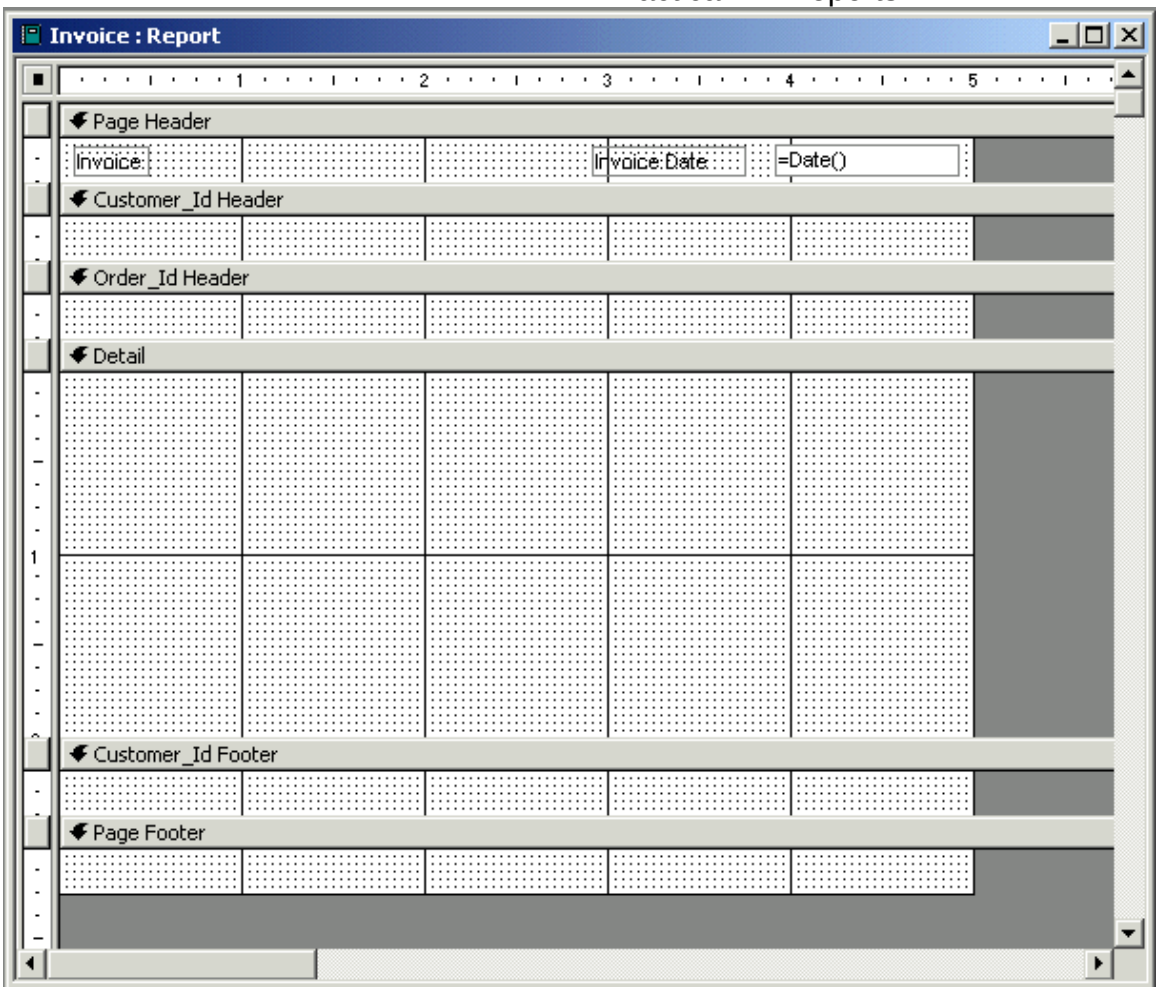


From the View menu, select Sorting and Grouping, and change as shown:





Your report will now have a Customer_Id header and footer, and an Order_Id header.



Sections of the Report

Page Header
This will appear at the top of each page of the report

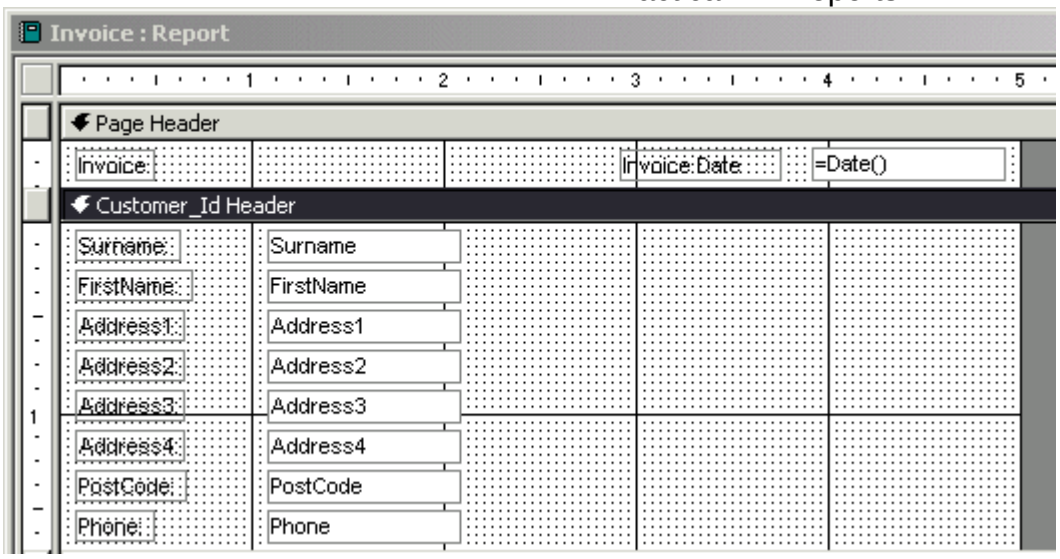
Customer_Id Header & footer
The contents of these will change for each group of records dependant on the Customer_Id

Order_Id Header & footer
The contents of these will change for each group of records dependant on the Order_Id

Detail
This will display data from each of the records in the underlying query

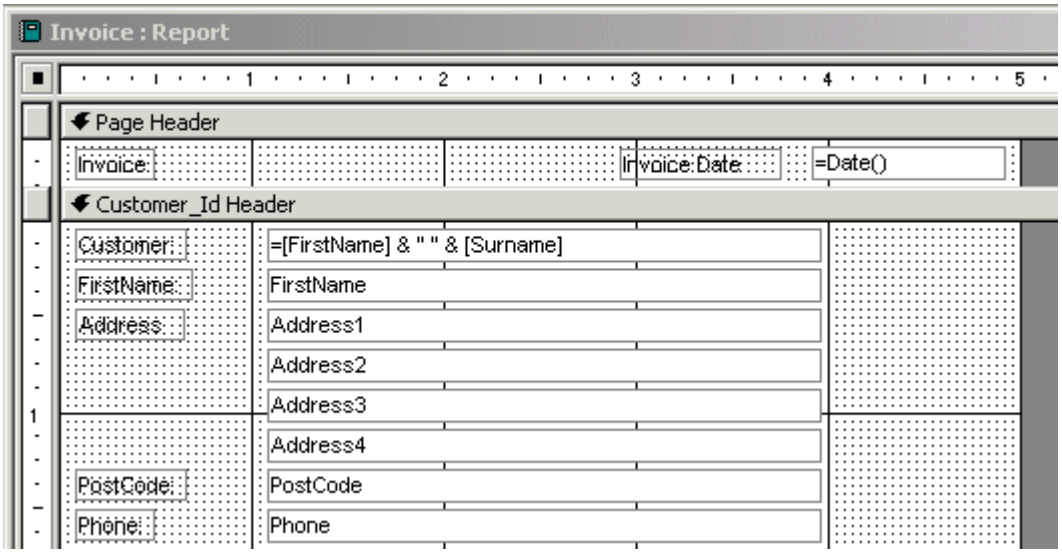
In this particular case, the detail section will show the data for all the orders for a particular customer. Each order will have its own data in the Order_Id header and footer, and each customer their own data in the Customer_Id header and footer.

From the View menu open the field list, select the customer details and drag them into the Customer_Id header.

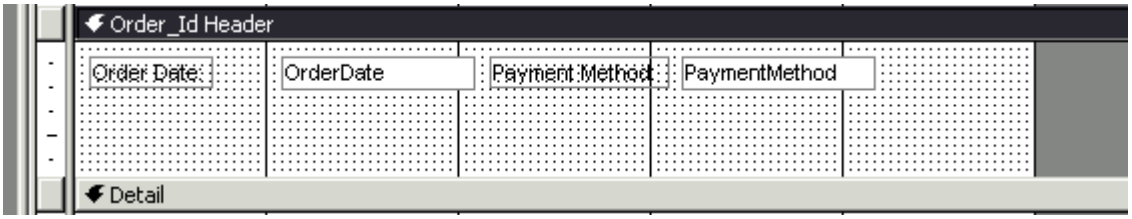


Create a new text box in which the FirstName and Surname are concatenated in the same way as they were on the CustomerOrder form.

Change the address text box labels.

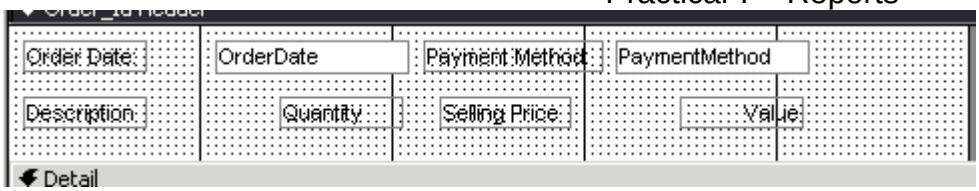


Select the Order details from the Field List and drag them into the Order_Id Header.
Edit the layout as shown below.



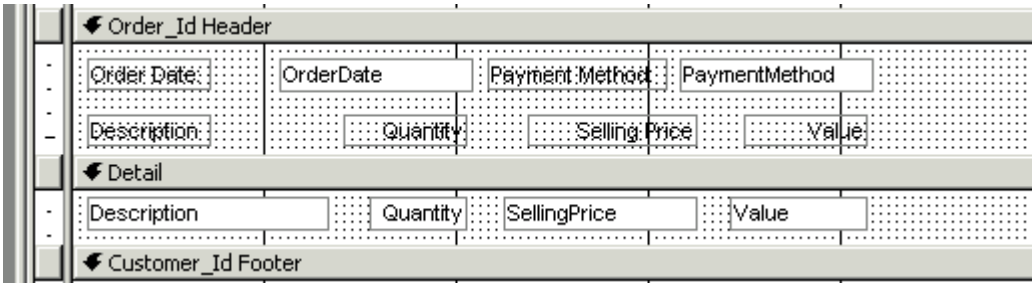
(You can alter a combo box to a text box from the toolbar : Format \ Change Format)

Add some labels in the Order_Id header for Quantity, Description, Selling Price and Value.



Select the Order Line details from the Field List and drag them into the Detail section.

Delete the accompanying labels from the Detail section.



You should change the Text Align property for all numeric fields to Right, and align right their labels (select the text box and the label and click the right button to find the Align functions)

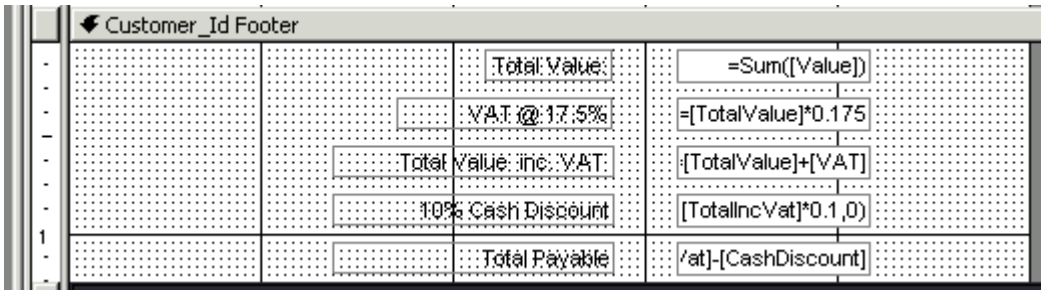
Note: it is better to have the Order line labels in the Order_Id Header so that they only appear once per order.

In the Customer_Id Footer, add text boxes as shown.
Change the Name and ControlSource properties of each of these is as follows:

Label	*Name*	*ControlSource*
Total Value	TotalValue	=Sum([Value])
Vat @ 17.5%	Vat	=[TotalValue]*0.175
Total Value Inc.Vat	TotalIncVat	=[TotalValue]+[VAT]
10% Cash Discount	CashDiscount	=IIf([PaymentMethod]="C", [TotalIncVat]*0.1,0)

Note: be careful to correctly type IIF, not IFF (which is a common error to make).

Total Payable	TotalPayable	=[TotalIncVat]-[CashDiscount]



The Control Source of the Text Box *CashDiscount* means :

IF Payment Type is “C” THEN
CashDiscount is [TotalIncVAT]*0.1
ELSE
CashDiscount is 0
ENDIF

In the Page Footer, add two text boxes. Change the Control Sources as shown, and delete the accompanying labels.

In the Customer_Id Footer properties sheet, change the *Force New Page* property to *After Section.* This will mean that every time the Customer_Id changes, a new page will be forced.

Save the report as *Invoice*

Check the report by clicking the preview button. 

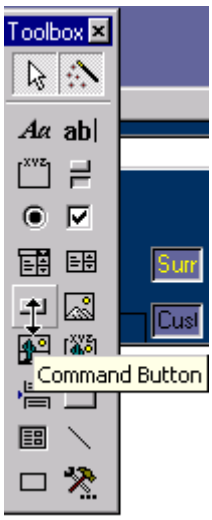
You will find that you have a page of the invoice for every customer, and that if the customer has multiple orders, they will all appear on the same invoice.

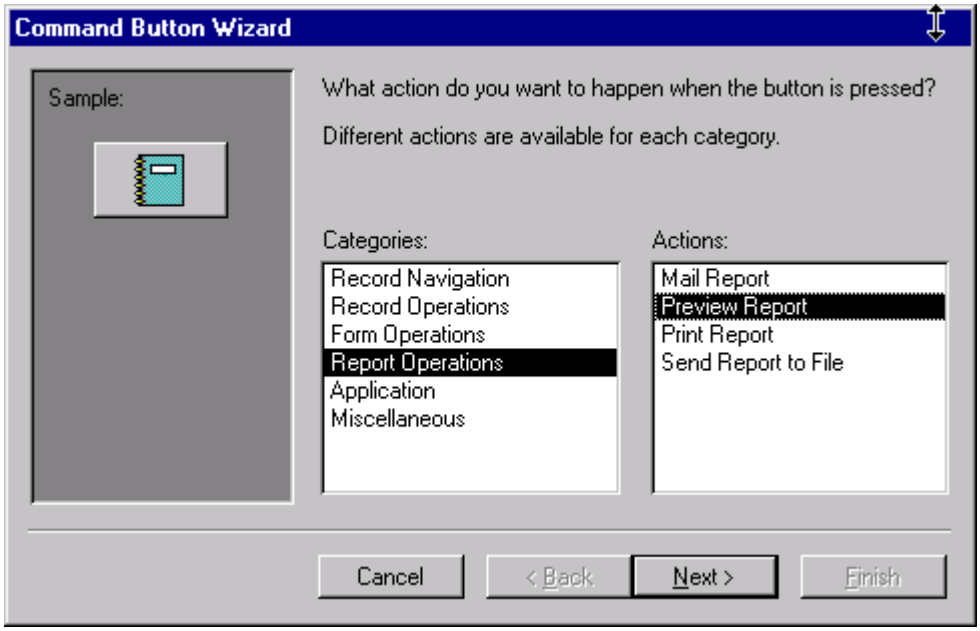
What we need is to enable the user to choose a customer’s order and then print the invoice just for that one order.

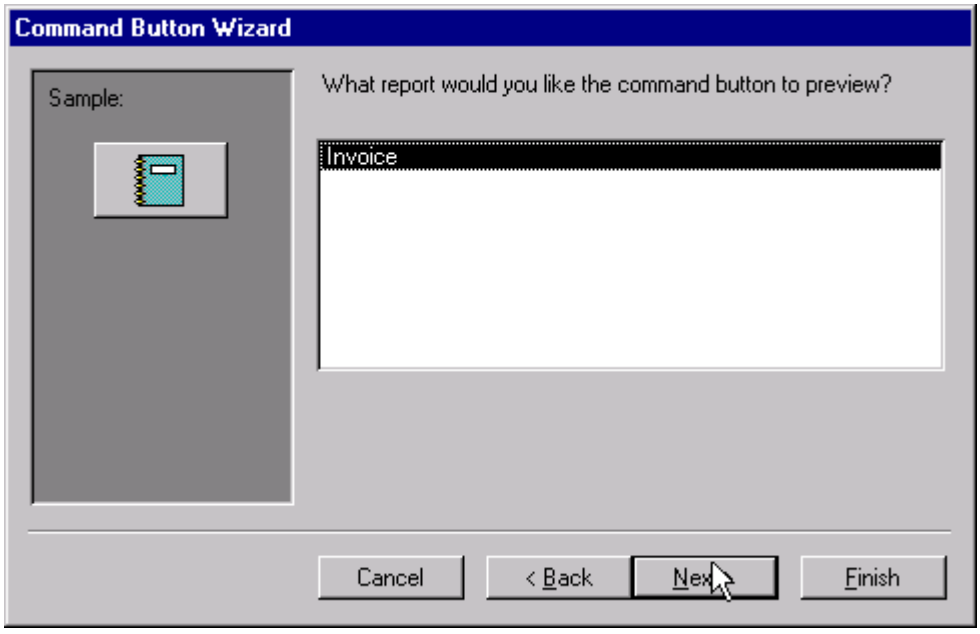
Printing the correct Invoice for an Order

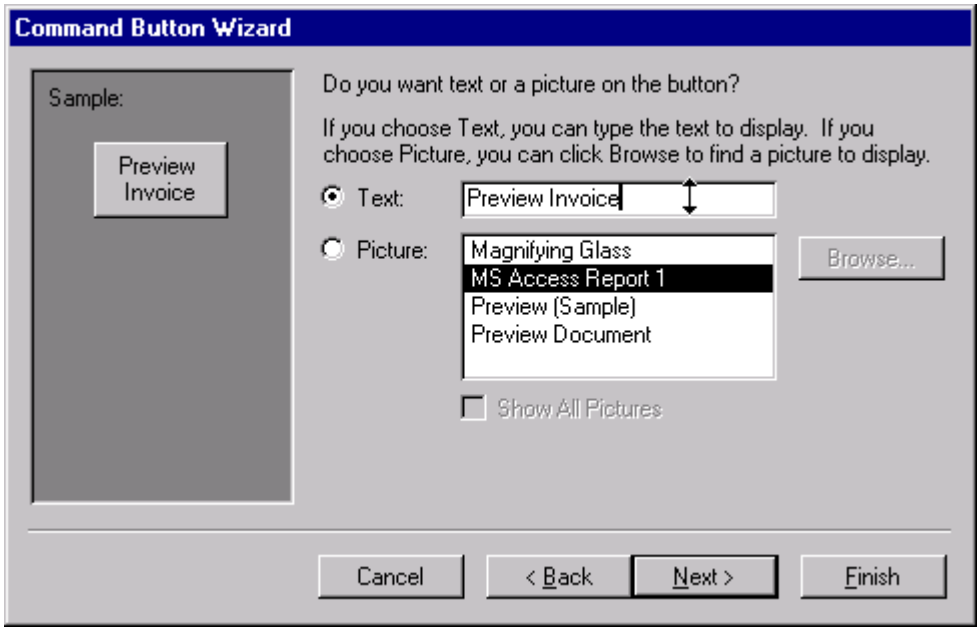
Open the OrderView form in design view.

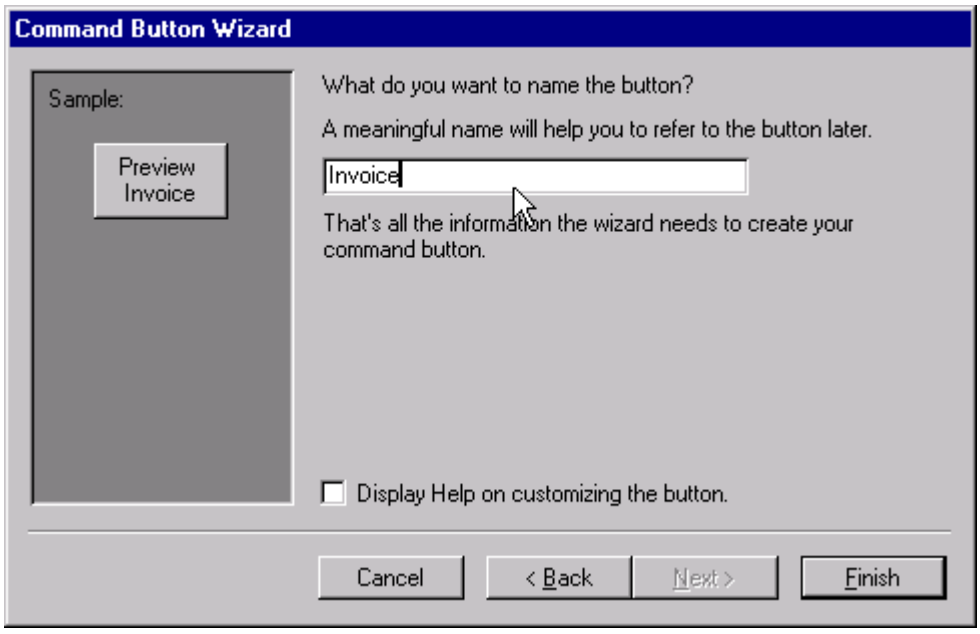
We are going to use the Button wizard to add a command button to the Order View form which will open the Invoice Report.


Toolbox
Surr
Cust
Command Button

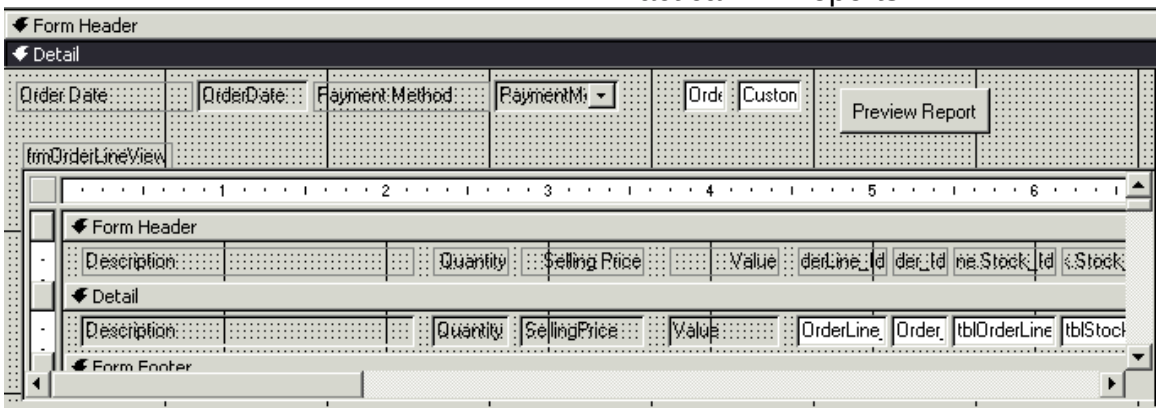

Command Button Wizard
Sample:
What action do you want to happen when the button is pressed?
Different actions are available for each category.
Categories:
Record Navigation
Record Operations
Form Operations
Report Operations
Application
Miscellaneous
Actions:
Mail Report
Preview Report
Print Report
Send Report to File
Cancel
< Back
Next >
Finish


Command Button Wizard
Sample:
What report would you like the command button to preview?
Invoice
Cancel
< Back
Next >
Finish





Your form should now look like this:



The Button Wizard creates some Visual Basic to perform this task.
You will need to edit it.

On the view menu, click View Code.

Find the code associated with the button. Remember, you chose on the last screen to give the button a meaningful name!

```
Option Compare Database

Private Sub Invoice_Click()
On Error GoTo Err_Invoice_Click

    Dim stDocName As String

    stDocName = "Invoice"
    DoCmd.OpenReport stDocName, acPreview

Exit_Invoice_Click:
    Exit Sub

Err_Invoice_Click:
    MsgBox Err.Description
    Resume Exit_Invoice_Click

End Sub
```

The line *DoCmd.OpenReport stDocName, acPreview* is the one that does the business. You need to edit it so that the report relating to the chosen customer is shown.

Add two commas to the end of the line *DoCmd.OpenReport stDocName, acPreview* and type the line "[Order_Id] = " & Me![Order_Id]

The complete line should now be :
DoCmd.OpenReport stDocName, acPreview, , "[Order_Id] = " & Me![Order_Id]

The final parameter of this command is the equivalent of a SQL WHERE clause. Here, you are concatenating the value of the Order_Id on the form to the text "[Order _Id] = ".

When the command is executed, the WHERE clause will look something like "[Order _Id] = 45". The report will then be opened only for the customer with the Order with the Order _Id of 45.

Save the OrderView form and close it.

Open the CustomerOrder form, select a customer with an order and try out the invoice button.

You could now try to make the invoice look more presentable!

CHECKPOINT: SDG Check7

ACCESS REFERENCE

Indexing and Queries

Indexing a field speeds up Queries. When setting a field's Indexed property, you will be asked to choose between :

Indexed No Duplicates
Use this when the field's value is unique to a particular record e.g. Customer_Id in the Customer File.

Indexed Duplicates OK
Use this when many records may have the same value in this field e.g. Customer_Id in the Order file.

Properties

Objects have properties that you can set to make them look and behave the way you want.
Refer to the Help system for a full reference to the properties available for each type of object.

Control Properties

Control properties can be changed by double clicking on a control in design view.

Useful properties

You should experiment with the use of the following properties :

Control Source	Field or calculation from which the control gets its value
Enabled	Allows / Prevents the user from selecting the field
Locked	Allows / Prevents the user from changing data in the field
Fore / Back Colour	Changes appearance
Font Name / Size	Changes appearance
Border Colour/Style	Changes appearance
Text Align	Left, Right, Centre
After Update	Can trigger an event after the contents of the control are changed
On Click	Can trigger an event when the user clicks on the control
Visible	Governs whether the control is visible on the form
Status Bar Text	What appears in the status bar at the bottom of the form when the mouse enters the control

Form properties

Play with these :

Back Color
DefaultEditing
Modal
Navigation Buttons
On Activate
On Open
Record Selectors
Scroll Bars

Activate the property box by double-clicking the white box in the top left of a form. (This is only visible when the Ruler is on.)

Bound & Unbound Controls

Bound

A bound control is one that is tied to an underlying element in the Database such as a Table or Query field.

For example, a *Text Box* may be bound to the Customer Surname field in the Customer Table. When data is entered into the Text Box, the field content is updated.

Unbound

Unbound controls are not tied to any data. They can be used to enter data which is to be used temporarily but not stored in a table e.g. the date to be used on a letter.

Referring to objects

Objects names should be enclosed in square brackets e.g: *[Customer Surname]*

To refer to an object or a value, start with the object and identify each element in turn. Use the ! operator before a Table, Form or Query name.

- For example, to refer to a form that you've named Orders, use the expression : *Forms![Orders]*
- To refer the Customer_Id control on the Orders form, use the expression
  - *Forms![Orders]![Customer_Id]*
- To refer to a property of a control, use the . operator. For example, to refer to the Enabled control on the Customer_Id control, use the expression *Forms![Orders]![Customer_Id].Enabled*

As you can see, the syntax involved can be difficult and it is better to use the Expression Builder (...) to help select the correct expression.

Form Design Standards

Many companies adopt standards for the design of software. These following guidelines are typical, and should be applied in your application.

Menu forms

Menu forms should not have extraneous features such as Navigation Buttons, Record Selectors or Scroll Bars.

Clearly Marked Exits

Forms should have a clearly marked exit.
Close Form buttons should unload the form, not re-open a previous form.

Titles

Forms should have a title to let the user know which system they are using and where they are in that system. It may be appropriate to have the date / time visible.

Data presentation

All data controls should be clearly labelled using simple and natural language

The dialogue should be expressed clearly in words, phrases and concepts familiar to the user rather than in system-oriented terms

All information should appear in a natural and logical order

Labelling should be consistent. Users should not have to wonder whether different words, situations or actions mean the same thing

The tab order should follow the physical screen order.

Data should be grouped together in a natural and logical manner. Rectangles can be used to emphasise groupings and avoid transgressing the 7+or+2 rule

Any control bound to data which the user is not allowed to edit should be disabled.

Computer generated codes should not be displayed

Data Input

Form design for data input should reduce the chances of user error.

Aspects which can help the user are :

- Clear labels
- Combo boxes
  For selecting data from reference tables or lists
- Default values

Giving an example of correct format
Reducing time spent on data entry

- Templates
  Visually indicating the format the data should be in e.g. dd/mm/yyyy

Dates

All dates should be entered with the full century i.e. 1997 *not* 97 in order to comply with Year 2000 compliance standards.

Form size

All screens should be capable of being displayed on the lowest screen resolution on which the application might be run. For example, for 640 X 480, screens should be a maximum of 6" wide and 4" deep.

Colours

Only the initial 16 Windows colours should be used. This is to prevent colour hatching which may occur when additional colours are used which are not supported by the user's video card.

Background colours should be restrained.

Fonts

Many companies restrict the number of fonts installed on their systems. Companies often have a company standard font.

Only the following fonts should be used in your application:

- MS Sans Serif
- Arial
- Times New Roman

Data should be displayed in font sizes 8, 10 or 12

Alignment

Controls and labels should be aligned in a regular manner both vertically and horizontally. Care should be taken over text alignment. It is, for example, usual to Right align text which is being used as a column heading for numeric data (which defaults to right alignment)

Customisation

In order to accommodate a wide range of users and their expectations and skill level, you may want to to provide alternatives to the default interface

For example :

Provide menus as well as buttons
Provide key shortcuts
Turn off help messages

In the final analysis, it will be the user who decides what interface best suits them. The RAD or prototyping method lends itself well to getting the interface right.

User Help and Error Messages

Validation text should be used rather than relying on Access' default error messages. The text should suggest what the user needs to do in order to avoid the error message.

On screen prompts should be provided for all data entry. This can be achieved either through the Status Bar Text property or by displaying a message triggered by the On MouseMove event.