Chapter 4 – Queries

Understanding queries

Queries are one of the most important tools in a database for converting raw data into useful information. A query can be thought of as a request to find all records satisfying certain criteria so that they can be displayed in a form or record. The terminology used here is rather confusing, since the word *query* in Access refers not only to the question (“Find all the customers on Round 1”), but also to the resulting *object*, a set of records. (This used to be referred to as a *dynaset* in Access 2). Queries may be made up of records that were created from fields from several tables.

You can use queries in much the same way that you use tables – for example, a form or report may be based on either a *table* or a *query*. You can also use a query to update tables, to perform global ‘search and replace’ tasks, to append, delete and analyse data.

Types of query

There are six different types of query in Access, each used for different purposes. They are:

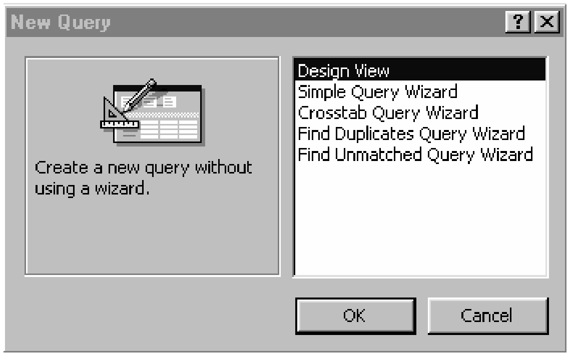
- Select query;
- Crosstab query;
- Make-table query;
- Update query;
- Append query;
- Delete query.

Task 4.1: Create a query involving only one table

We will start with the simplest, the **Select query**.

The simplest use of a query is to show selected fields and records from one table. We’ll create a query that displays the names, addresses and holiday dates of all customers on Round 1.

- In the database window click **Queries**, select **Create Query in Design View** (*not in Access 97*) and then click **New**. The following window appears:



Create a new query without using a wizard.

Figure 4.1: Creating a new query

- Double-click **Design View** to open a query window. A query window appears with the **Show Table** dialogue box floating on top of it.

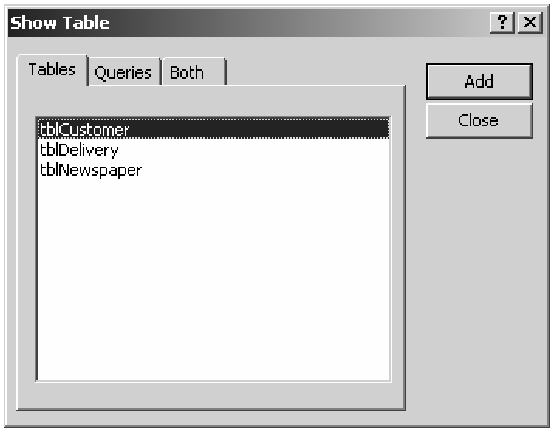


Figure 4.2: The Show Table dialogue box

- With **tblCustomer** selected, click **Add**, and then **Close**. A table pops up on the query window as shown below:

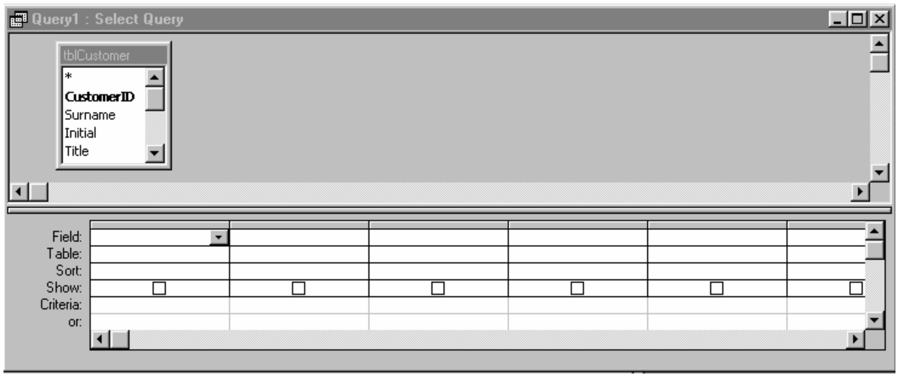


Figure 4.3: The Query window

Selecting fields

The next step is to select the fields that you want to see in the query. We'll select **Surname**, **Initial**, **Address1**, **HolsBegin** and **HolsEnd**. The fields can be placed on the query grid in a number of different ways:

- Double-click the field name;
- Drag the field name from the table onto the top line of the query grid;
- Click the Down arrow in the Field cell in the query grid to display a list of field names from which you can select.

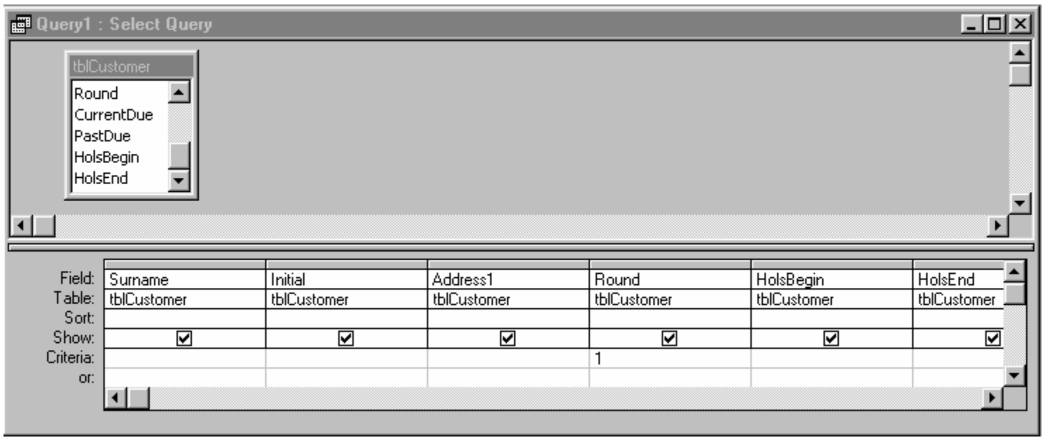
- Double-click **Surname, Initial**, **Address1**, **Round**, **HolsBegin** and **HolsEnd** in turn to place them in the query grid.

Tip: *If you place a wrong field by mistake, move the cursor over the top margin of the column (the column selector) until it changes to a downward pointing arrow, click to select the column and press the* ***Delete*** *key to delete it. You can move a column by dragging the column selector once the column is selected.*

Setting the criteria

We only want to see the customers for Round 1.

- In the Criteria row, type *1* in the column for **Round**. The query grid now appears as shown:

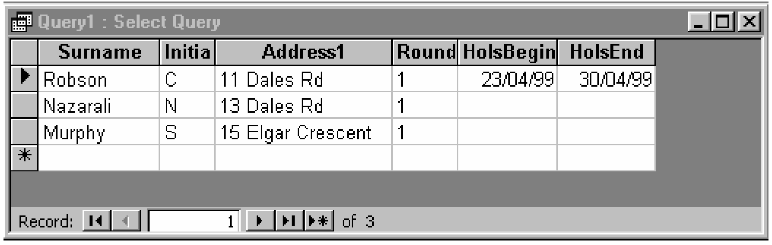


Field:	Surname	Initial	Address1	Round	HolsBegin	HolsEnd
Table:	tblCustomer	tblCustomer	tblCustomer	tblCustomer	tblCustomer	tblCustomer
Sort:						
Show:	☑	☑	☑	☑	☑	☑
Criteria:				1		
or:						

Figure 4.4: Selecting fields and setting criteria

Running the query

- Click the **Run** button on the toolbar. The query results appear:



Surname	Initia	Address1	Round	HolsBegin	HolsEnd
Robson	C	11 Dales Rd	1	23/04/99	30/04/99
Nazarali	N	13 Dales Rd	1		
Murphy	S	15 Elgar Crescent	1		

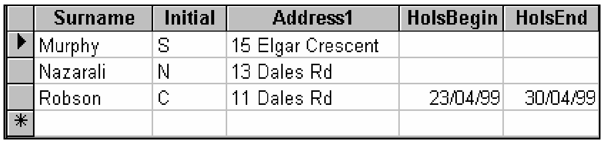
Record: 1 of 3

Figure 4.5: Query results

Sorting records and hiding fields

You can control certain aspects of the query result table, such as the sort order and which fields you want to see. For example you may want the results displayed in alphabetical order of surname, and you may not want the Round field displayed since you know all these customers are on Round 1.

- Click the **Design View** tool to return to design view.
- In the **Surname** column, click in the **Sort** row and select **Ascending** from the drop-down list.
- In the **Round** column, click the check box in the **Show** row to deselect it.
- Run the query again. You can adjust the column widths of the resultant query so that the results appear as follows:



Surname	Initial	Address1	HolsBegin	HolsEnd
Murphy	S	15 Elgar Crescent		
Nazarali	N	13 Dales Rd		
Robson	C	11 Dales Rd	23/04/99	30/04/99

*Figure 4.6: Query results after sorting on **Surname** and hiding **Round** field*

- Save the query, naming it *qryRound1*.

Setting special criteria

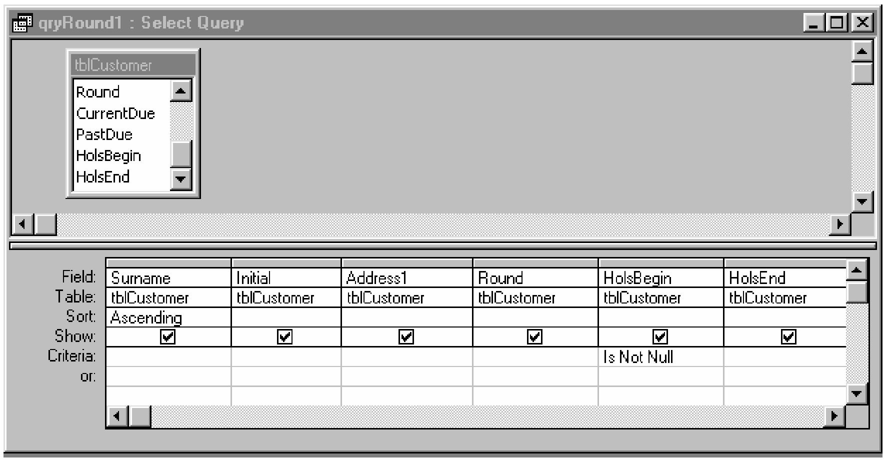
If you want to look for all records with a constant value in a particular column, you simply type the value in the appropriate column, as you did when searching for all customers on Round 1. You can also:

- Look for all records with a blank in a particular field;
- Look for records that satisfy a range of values;
- Use expressions involving Visual Basic functions such as **Date()**.

Looking for blanks

Suppose you want to look for all customers who have entries in the **HolsBegin** field.

- Return to Design view.
- Delete the criteria from the **Round** field so that we have a few more records to choose from, and check its Show box to select it.
- In the criteria for **HolsBegin**, type *Is not null* as shown below:



Field:	Surname	Initial	Address1	Round	HolsBegin	HolsEnd
Table:	tblCustomer	tblCustomer	tblCustomer	tblCustomer	tblCustomer	tblCustomer
Sort:	Ascending					
Show:	☑	☑	☑	☑	☑	☑
Criteria:					Is Not Null	
or:						

*Figure 4.7: Searching for records with a non-blank **HolsBegin** field*

- Run the query. This time you should see only the records for Newbourne and Robson.

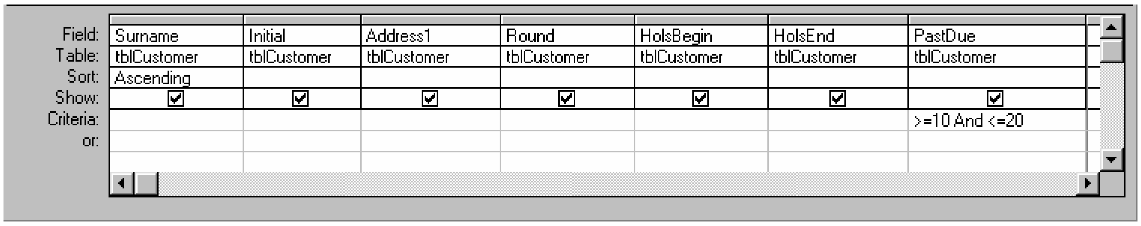
Looking for a range of values

Now supposing you want to find all customers whose holidays begin on or after April 27th.

- Return to Design view.
- Delete the criteria **Is Not Null** in the **HolsBegin** field and type *>=27/04/99* instead.

- Run the query, and this time you should see only the record for Newbourne.

You can include the words ‘and’ and ‘or’ in your query criteria. For example if you want to find all customers whose **PastDue** amount is between £10 and £20 inclusive, you could add the **PastDue** field to the list of selected fields by dragging it onto the query grid, and type *>=10 and <=20* as the criteria in the **PastDue** field, as shown below. (You can also type *Between 10 and 20* for the criteria. This is a more convenient format, especially for dates.)



Field:	Surname	Initial	Address1	Round	HolsBegin	HolsEnd	PastDue
Table:	tblCustomer	tblCustomer	tblCustomer	tblCustomer	tblCustomer	tblCustomer	tblCustomer
Sort:	Ascending						
Show:	☑	☑	☑	☑	☑	☑	☑
Criteria:							>=10 And <=20
or:							

Figure 4.8: Looking for a range of values in a field

Using wildcards

Suppose you wanted to find all the customers living in Dales Road.

- Type **Dales** in the criteria for **Address1**. (* is known as a *wildcard* character. You can look up other wildcard characters in the Access Help system.)
- If you want to check the result of this query, first delete any other criteria you may have set. The records for Nazarali and Robson should be displayed.

Using a function to find records

Date functions combined with mathematical operators such as + and - are sometimes useful to find, for example, all dates more than 30 days ago.

The following are examples of expressions that calculate or manipulate dates, reproduced from the MS Access Help system.

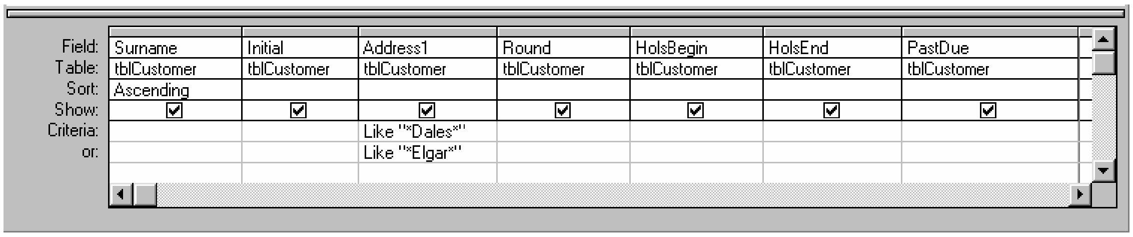
Field	Expression	Description
RequiredDate	Between Date( ) And DateAdd("m", 3, Date( ))	Uses the Between...And operator and the DateAdd and Date functions to display orders required between today’s date and three months from today’s date.
OrderDate	< Date( ) - 30	Uses the Date function to display orders more than 30 days old.
OrderDate	Year([OrderDate]) = 1996	Uses the Year function to display orders with order dates in 1996.
OrderDate	DatePart("q", [OrderDate]) = 4	Uses the DatePart function to display orders for the fourth calendar quarter.
OrderDate	DateSerial(Year([OrderDate]), Month([OrderDate])+1, 1)-1	Uses the DateSerial, Year, and Month functions to display orders for the last day of each month.
OrderDate	Year([OrderDate]) = Year(Now()) And Month([OrderDate]) = Month(Now())	Uses the Year and Month functions and the And operator to display orders for the current year and month.

Table 4.1: Date functions used in criteria

AND and OR queries

Select queries can include multiple criteria to find records. If you need to use more than one criterion in the same field, you can separate the criteria with the word *and.* If the criteria are in different fields, enter each one in the appropriate column.

To use an OR query, put the criteria on different rows. The query below finds all customers who live in Dales Road or Elgar Crescent. The criteria are entered on two separate rows as *Like "*Dales*", Like "*Elgar*"*. Note that the use of the wildcard * allows you to find all records containing the word **Dales** or **Elgar**.



Field:	Surname	Initial	Address1	Round	HolsBegin	HolsEnd	PastDue
Table:	tblCustomer	tblCustomer	tblCustomer	tblCustomer	tblCustomer	tblCustomer	tblCustomer
Sort:	Ascending						
Show:	☑	☑	☑	☑	☑	☑	☑
Criteria:			Like "*Dales*"				
or:			Like "*Elgar*"				

Figure 4.9: Using OR in query criteria

The two parts of the OR do not necessarily have to be in the same column – you could, for example, look for all customers who either live in Dales Road OR are on Round 1, by putting the criteria on different rows in the appropriate columns.

Allowing the end-user to enter criteria

It is often useful when you are developing a customised application to be able to save a query but let the user type in the criteria when the query is run. For example, you could allow the user to specify the Round number in the original query instead of having the query always find customers in Round 1.

There are two ways of doing this.

Method 1: Use a form to enter query criteria

Create a dialogue box which the user opens (see Task 7.8 in chapter 7) to specify criteria. The user then clicks a command button (labeled say, **OK**) to run the query. For example you could create a dialogue box named **fdlgEnterRound** which asks the user to enter the Round number in a text box named **RoundNumber**. You would have to have previously designed the query with *Forms![fdlgEnterRound].[RoundNumber]* in the Criteria row of the Round column.

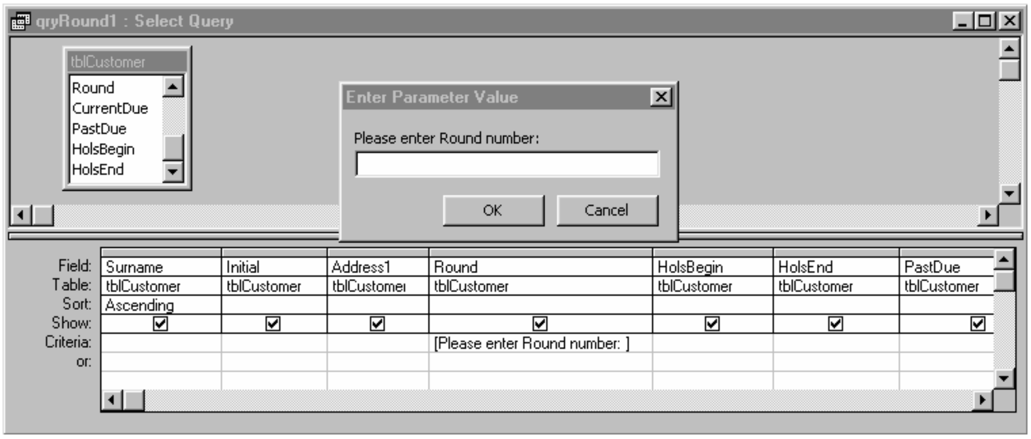
(The Help system explains this further, with examples. Type *Entering Criteria* and then select **Create a form to enter report criteria**.)

Method 2: Enter a prompt to the user in the Criteria row of the query.

The prompt must be enclosed in square brackets. This method is quicker to implement but does not give you the flexibility of using a dialogue box, in which you can use list boxes or option groups to limit the user's entries and thus eliminate the possibility of an invalid entry (e.g. a non-existent Round 4).

You can try this method now, as shown below:

- In Design view, delete any existing criteria and in the **Criteria** row of the **Round** column type *[Please enter Round number:]*. Make sure **Show** is selected in this column.
- Press the **Run** button to run the query. The message you typed is displayed asking the user to enter the Round number:



Please enter Round number:

Field:	Surname	Initial	Address1	Round	HolsBegin	HolsEnd	PastDue
Table:	tblCustomer	tblCustomer	tblCustomer	tblCustomer	tblCustomer	tblCustomer	tblCustomer
Sort:	Ascending						
Show:	☑	☑	☑	☑	☑	☑	☑
Criteria:				[Please enter Round number:]			
or:							

Figure 4.10: Allowing the user to specify criteria

- Enter *1* and press **OK**. The query finds all customers on Round 1.

Saving a query with a different name

- From the menu select **File, Save As** to leave the original **Round1** query unchanged. Save the new query as *qryRound*.
- Close the query to return to the database window.

Task 4.2: Query linked tables and create a form from a query

Queries can be used to bring together fields from several related tables. We will use a query to display newspaper deliveries to a customer.

- Make sure **Queries, Create query in design view** is selected in the Database window and then click **New**.
- Select Design View and click **OK**.
- In the Show Table dialogue box, select and add each table in turn. (You can hold down the **Shift** key while you select the bottom table to select all three tables and add them in one operation.)

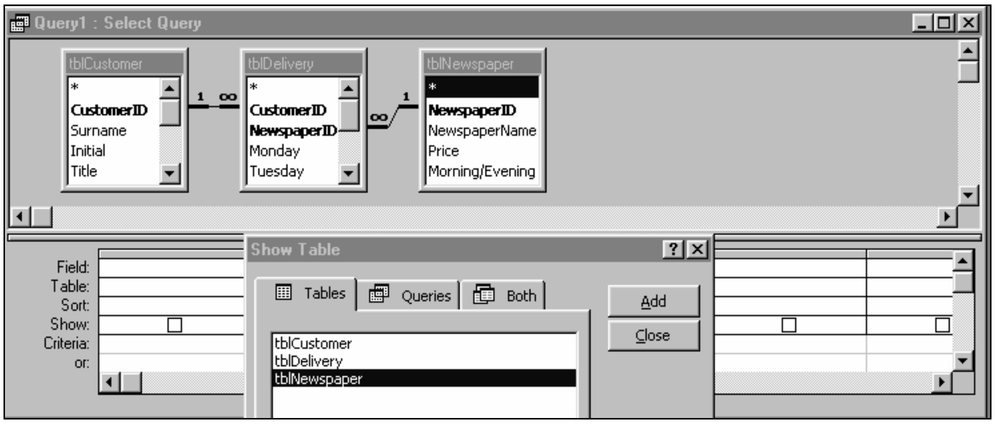


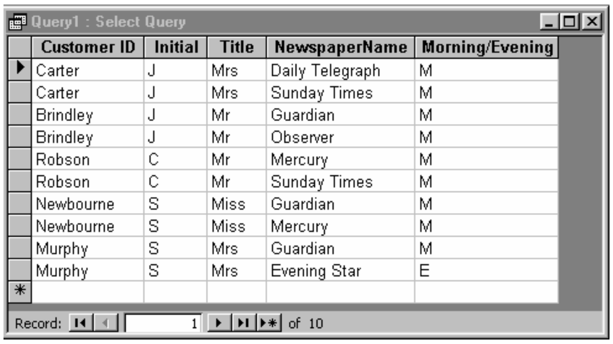
Figure 4.11: Adding all tables to the Select Query window

- Click **Close** in the Show Table dialogue box.

TblDelivery is the table that links the other two tables together. Suppose we want to display for a particular customer, their Surname, Initial and Title, and for each newspaper they have delivered, the newspaper name and whether it is a morning or evening paper. It makes no difference whether you select **CustomerID** and **NewspaperID** from **tblDelivery** or from **tblCustomer** and **tblNewspaper** respectively. (Which records appear when you run the query depends on the Join type, as described in the note below.) **Surname** and **Initial** will be selected from **tblCustomer** and **NewspaperName** and **Morning/Evening** from **tblNewspaper**.

> Note: There are different types of Join that can be used in a query. If you double-click the line between **tblCustomer** and **tblDelivery**, you will see that the default type of Join is **Only include rows where the joined fields from both tables are equal**. This means that if you had customers on file who have no deliveries, they would not be listed when the query was run, which is probably what you want in this case and in most other cases.

- Double-click **CustomerID** in the **tblDelivery** table (not **tblCustomer**) to add it to the query grid.
- Because we used the Lookup wizard to define the field for **CustomerID** when the table structure was created in Chapter 2, the Surname rather than the ID number will automatically be displayed. Just run the query as it is now to convince yourself of this fact, and then return to the Design window.
- Double-click **Initial** and **Title** on **tblCustomer**, and **NewspaperName** and **Morning/Evening** on **tblNewspaper**.
- Run the query. Results are displayed as follows (they may be in a different order):



Query1 : Select Query

Customer ID	Initial	Title	NewspaperName	Morning/Evening
Carter	J	Mrs	Daily Telegraph	M
Carter	J	Mrs	Sunday Times	M
Brindley	J	Mr	Guardian	M
Brindley	J	Mr	Observer	M
Robson	C	Mr	Mercury	M
Robson	C	Mr	Sunday Times	M
Newbourne	S	Miss	Guardian	M
Newbourne	S	Miss	Mercury	M
Murphy	S	Mrs	Guardian	M
Murphy	S	Mrs	Evening Star	E

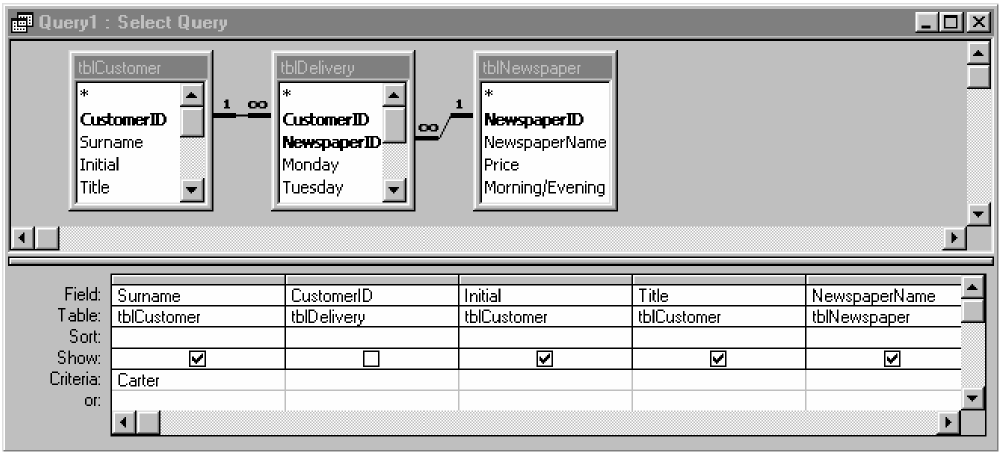
Record: 1 of 10

Figure 4.12: Query combining fields from different tables

Now suppose we want to display only the records for Carter. Notice that the heading at the top of the first column is **CustomerID** even though surnames are displayed. You won't be able to put the criteria "Carter" in the first column because this is an AutoNumber field, in spite of the fact that the Surname is displayed.

- Return to Design view.
- Drag **Surname** from **tblCustomer** on to the first column of the query grid. This automatically inserts a new column to the left of column 1.
- Type the criteria *Carter* in the **Surname** column.

- Uncheck the **Show** box in the **CustomerID** column, as shown below.



Query1 : Select Query

tblCustomer: *, CustomerID, Surname, Initial, Title

tblDelivery: *, CustomerID, NewspaperID, Monday, Tuesday

tblNewspaper: *, NewspaperID, NewspaperName, Price, Morning/Evening

Field:	Surname	CustomerID	Initial	Title	NewspaperName
Table:	tblCustomer	tblDelivery	tblCustomer	tblCustomer	tblNewspaper
Sort:					
Show:	☑	□	☑	☑	☑
Criteria:	Carter				
or:					

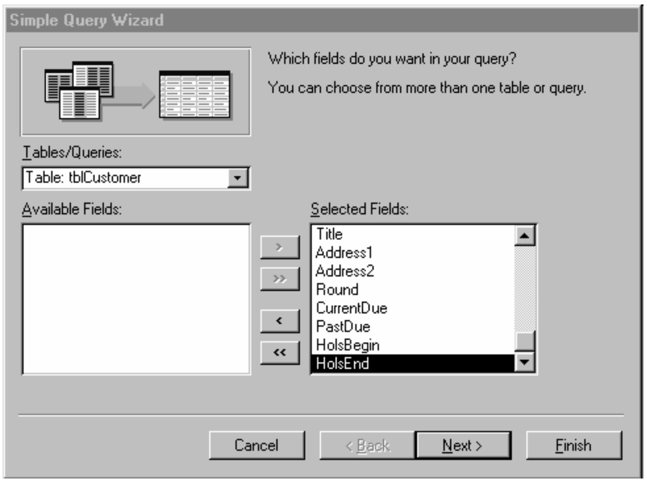
Figure 4.13: Setting the Surname criteria

- Run the query.
- Save the query as *qryDelivery* and close it to return to the Database window.

Using the Simple Query Wizard

The simple Query Wizard can be used when you do not need to enter any criteria.

- In the Database window, make sure **Queries, Create query by using wizard** is selected and click **New**.
- This time we will use the Simple Query Wizard to create the basic query, which we can then edit. Select **Simple Query Wizard** and click **OK**.
- In the next window, select **Table: tblCustomer** in the **Tables/Queries** box and click the double arrow between the list boxes to transfer all the fields from the **Available Fields** to the **Selected fields** list boxes. Click **Next**.



Simple Query Wizard

Which fields do you want in your query?

You can choose from more than one table or query.

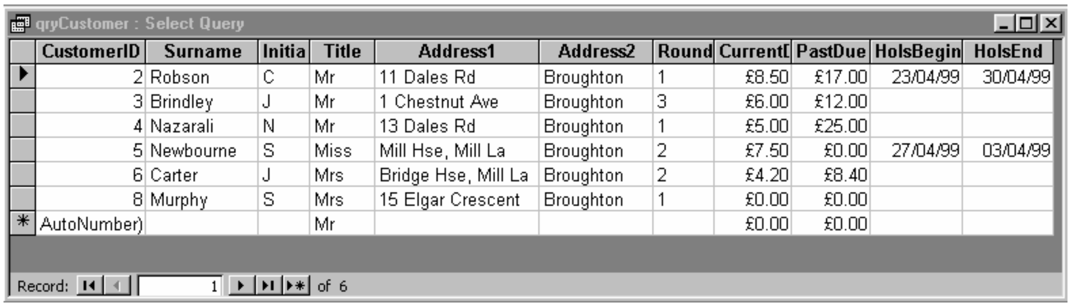
Tables/Queries: Table: tblCustomer

Available Fields:

Selected Fields: Title, Address1, Address2, Round, CurrentDue, PastDue, HolsBegin, HolsEnd

Figure 4.14: Using the Simple Query Wizard

- On the next screen, make sure **Detail (shows every field of every record)** is selected, and click **Next**.
- Name the query *qryCustomer* and click **Finish**.
- The records appear in Datasheet view. (Column widths have been adjusted in the figure below.)



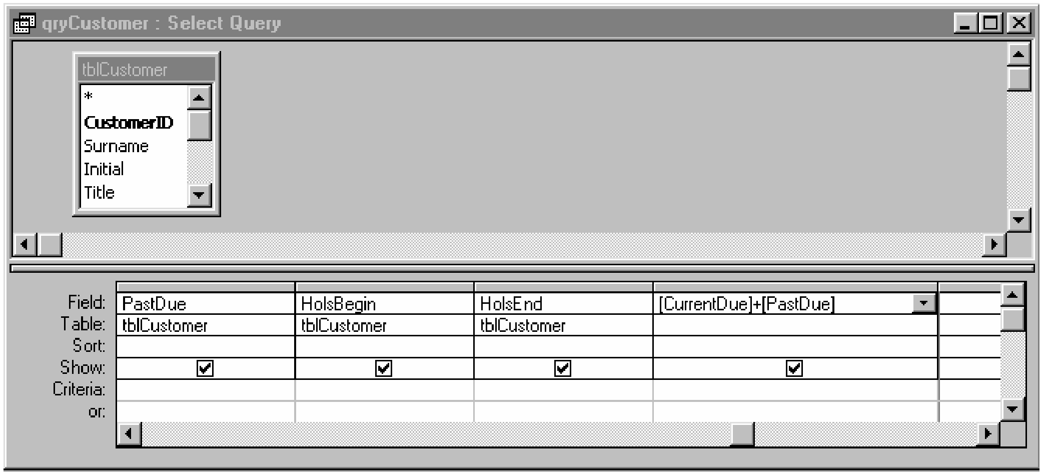
	CustomerID	Surname	Initia	Title	Address1	Address2	Round	CurrentI	PastDue	HolsBegin	HolsEnd
▶	2	Robson	C	Mr	11 Dales Rd	Broughton	1	£8.50	£17.00	23/04/99	30/04/99
	3	Brindley	J	Mr	1 Chestnut Ave	Broughton	3	£6.00	£12.00		
	4	Nazarali	N	Mr	13 Dales Rd	Broughton	1	£5.00	£25.00		
	5	Newbourne	S	Miss	Mill Hse, Mill La	Broughton	2	£7.50	£0.00	27/04/99	03/04/99
	6	Carter	J	Mrs	Bridge Hse, Mill La	Broughton	2	£4.20	£8.40		
	8	Murphy	S	Mrs	15 Elgar Crescent	Broughton	1	£0.00	£0.00		
*	AutoNumber)			Mr				£0.00	£0.00		

Figure 4.15: Query results

Creating calculated fields with a query

You can add extra fields to a query to show values calculated from other fields. In the next example we will add a field to *qryCustomer* that will calculate the total amount that each customer owes by adding together the **PastDue** and the **CurrentDue** amounts.

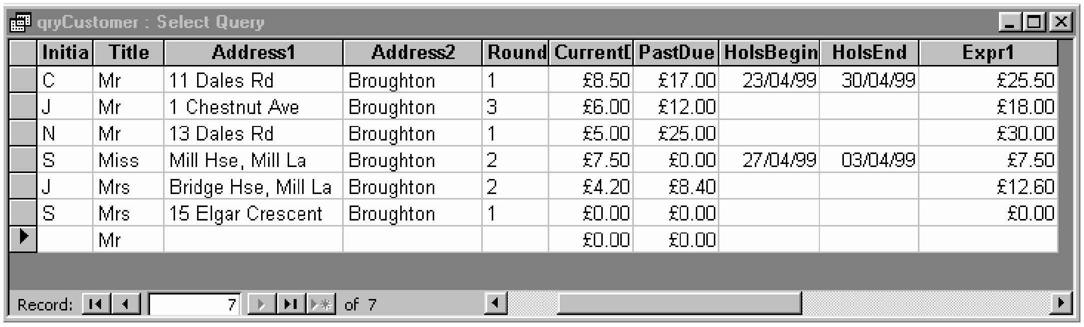
- Return to Design View.
- Scroll along the query grid until you can see a blank column. In the top row type as shown below
  [CurrentDue] + [PastDue]
- As soon as you press **Enter** or tab out of the column, Access adds **Expr1**: as the field name to be used for your calculation. (Widen the column if necessary so that you can see the whole expression.)



Field:	PastDue	HolsBegin	HolsEnd	[CurrentDue]+[PastDue]
Table:	tblCustomer	tblCustomer	tblCustomer	
Sort:				
Show:	☑	☑	☑	☑
Criteria:				
or:				

Figure 4.16: Adding a calculated field

- Click either the **Run** button or the **View** button to see the query in Datasheet view. You'll see the new column headed **Expr1** appear after **HolsEnd**. You may need to scroll along to see it.



Initia	Title	Address1	Address2	Round	Current[	PastDue	HolsBegin	HolsEnd	Expr1
C	Mr	11 Dales Rd	Broughton	1	£8.50	£17.00	23/04/99	30/04/99	£25.50
J	Mr	1 Chestnut Ave	Broughton	3	£6.00	£12.00			£18.00
N	Mr	13 Dales Rd	Broughton	1	£5.00	£25.00			£30.00
S	Miss	Mill Hse, Mill La	Broughton	2	£7.50	£0.00	27/04/99	03/04/99	£7.50
J	Mrs	Bridge Hse, Mill La	Broughton	2	£4.20	£8.40			£12.60
S	Mrs	15 Elgar Crescent	Broughton	1	£0.00	£0.00			£0.00
	Mr				£0.00	£0.00			

Figure 4.17: Datasheet view

Moving columns

We’ll move the new column so that it appears before the **HolsBegin** field.

- Place the pointer in the top margin above the new calculated column so that it appears as a down arrow, and click to select the column.
- Now click again in the same place, and the pointer changes to an upward pointing arrow. Drag the boundary to between **PastDue** and **HolsBegin**.

Changing a column caption

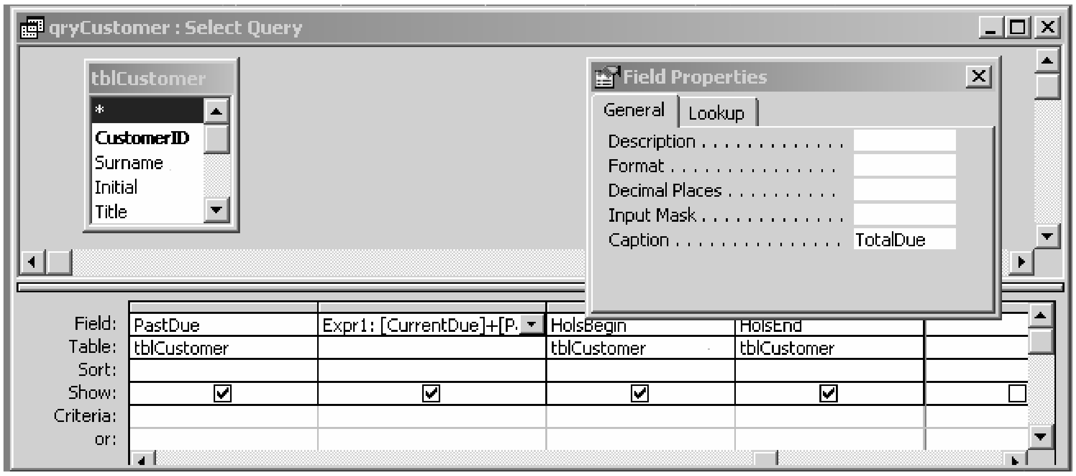
You can change the caption at the head of the column in either of two ways:

- by changing the **Caption** property of the field, or
- by changing the name in the Design grid.

We’ll change the caption of the **Expr1** field to **TotalDue** using the first method.

- Return to Design View, and right-click in the **Expr1** column. Select **Properties**.
- The Properties sheet is displayed. Type *TotalDue* in the **Caption** property and press **Enter**.

  (Note that this does not change the heading **Expr1** in the query, but the column caption will be changed to **TotalDue** when you run the query.)



Description . . . .
Format . . . .
Decimal Places . . . .
Input Mask . . . .
Caption . . . . TotalDue

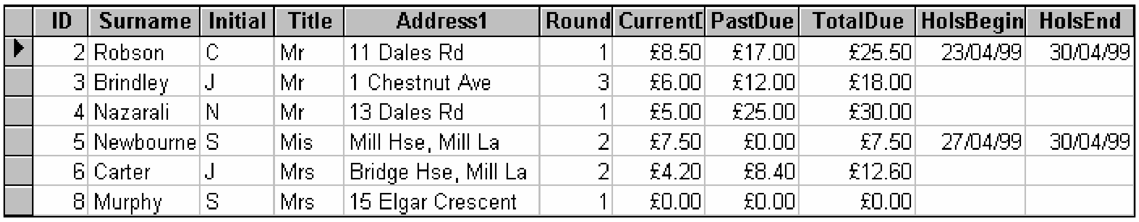
Field:	PastDue	Expr1: [CurrentDue]+[P.	HolsBegin	HolsEnd	
Table:	tblCustomer		tblCustomer	tblCustomer	
Sort:					
Show:	☑	☑	☑	☑	□
Criteria:					
or:					

Figure 4.18: Changing the column caption

- Close the Field Properties box.

  We'll use the second method to change the caption of the **CustomerID** column to *ID*.

- In Design View, place the cursor to the left of the field name CustomerID and type *ID:* (Note: You must type the colon between **ID:CustomerID**).
- Run the query.
- Right-click the **Address2** column header and click **Hide Columns**. (Note: To restore a column, select **Format, Unhide Columns**.)
- The query appears as shown below, after adjusting column widths.



	ID	Surname	Initial	Title	Address1	Round	CurrentD	PastDue	TotalDue	HolsBegin	HolsEnd
▶	2	Robson	C	Mr	11 Dales Rd	1	£8.50	£17.00	£25.50	23/04/99	30/04/99
	3	Brindley	J	Mr	1 Chestnut Ave	3	£6.00	£12.00	£18.00		
	4	Nazarali	N	Mr	13 Dales Rd	1	£5.00	£25.00	£30.00		
	5	Newbourne	S	Mis	Mill Hse, Mill La	2	£7.50	£0.00	£7.50	27/04/99	30/04/99
	6	Carter	J	Mrs	Bridge Hse, Mill La	2	£4.20	£8.40	£12.60		
	8	Murphy	S	Mrs	15 Elgar Crescent	1	£0.00	£0.00	£0.00		

Figure 4.19: Column captions have been changed and a column hidden

- Save **qryCustomer** and close the query.

You can look up more detail on renaming fields and the effects of doing this if you need to, by asking the Office Assistant. Type *How do I change the field name in a query?* and the paperclip will be pleased to instruct you.

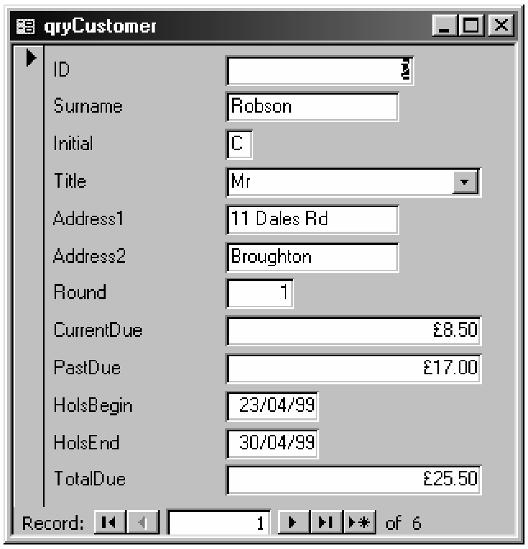
Creating a form from a query

Queries are often used as the basis for a form. This means for example that fields from two or more separate tables can first be combined using a multi-table query, and then a form using all the fields can be created and used for data entry. The data will automatically be stored in the correct table. You can also use a query to display calculated fields, change captions, sort on a particular field etc and then create a form which reflects these operations.

For these reasons, you should consider basing *all* your forms on queries rather than on tables, as you will have much more control over what you can put on the form. You will be able to amend your forms more easily too if you later decide you need to display a calculated field.

We'll create a customer form based on the query **qryCustomer** that we have just created.

- In the Database window, click **Forms, Create form by using wizard** and click **New**.
- In the New Form dialogue box, select **AutoForm: Columnar**. Select **qryCustomer** as the **Table or query where the object's data comes from**, and click **OK**.
- The new form is automatically created and appears as shown below:



qryCustomer

ID	1
Surname	Robson
Initial	C
Title	Mr
Address1	11 Dales Rd
Address2	Broughton
Round	1
CurrentDue	£8.50
PastDue	£17.00
HolsBegin	23/04/99
HolsEnd	30/04/99
TotalDue	£25.50

Record: 1 of 6

Figure 4.20: A form created from the query

Note that:

- The label for the **CustomerID** field appears as **ID** because the field was renamed in the query;
- **TotalDue** (or **Expr1**) is based on an expression and cannot be edited.
- The form caption by default shows the source of the form, **qryCustomer**. You can customise this by altering the form's **Caption** property in Design view.

- Add a new record for a customer named Miss J.Patel, 3 Elgar Crescent, Broughton, Round 1, with **CurrentDue** of £5.00 and **PastDue** of £3.75. You'll notice that **TotalDue** is calculated automatically as soon as you tab out of the **PastDue** field.
- Save your form as *frmCustomerDues* and close it to return to the Database window.

In Chapter 5 we'll look at other types of query, and in Chapter 6 we'll look at more advanced form design features.