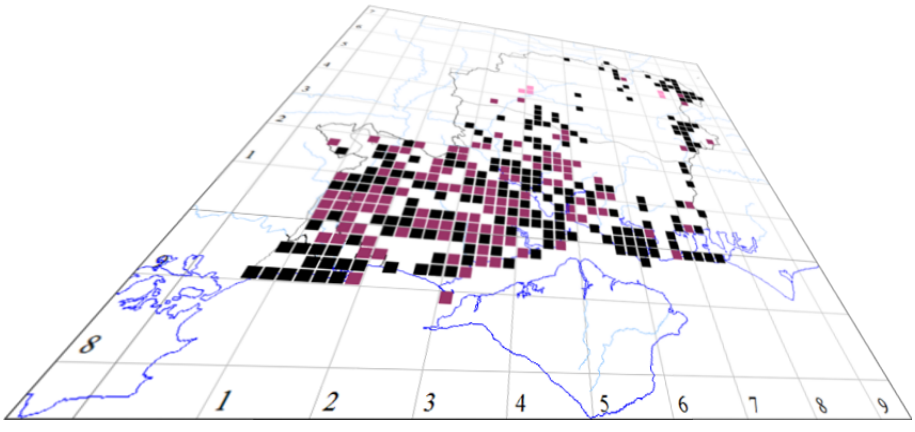
Chapter 4
Reporting and Maintaining Data



The author of this handbook is pleased to have errors pointed out and to receive suggestions for improvement and other comments.

Please send all communications to:
vc11recorder@hantsplants.net

CHAPTER 4 REPORTING AND MAINTAINING DATA

Introduction

In this chapter we are going to look at some of the options for examining the data you already have in your MapMate database. We start with some very simple facilities for retrieving data "by example", then proceed to look at the large library of interrogation options that come as standard with MapMate. Finally we look at how you can make up your own, more specialised enquiries. There is a simple (but limited) way to do this that requires very little computer technical knowledge; and there is a much more versatile way that requires you to immerse yourself in the technicalities of computer software. We shall explore this latter to a limited extent at the end of the chapter, because it is so useful; but to be honest, the topic really deserves a separate book. Hopefully, if you decide to tackle this part of the chapter, it will give you enough background information to be able to make use of one of the excellent online support forums for MapMate users for further help.

One thing you will quickly come to realise is that there are no facilities in MapMate for designing reports for printing, or even for using pre-designed printed reports. The MapMate developers have taken the view that they should concentrate their efforts on the biological recording and mapping aspects, and let you choose your own software for document presentation. So there are really only three things you can do with data you interrogate from your database:

- View it in a "grid" layout on screen.
- Copy it to the Windows clipboard for pasting into another program.
- Save it to a standard format of file which can be used to load the data into other programs such as spreadsheets, word processors and general-purpose databases.

If you want to produce reports or extracts for other people who are not MapMate users, you are clearly going to need one or more of these other programs, unless you are just going to send them the raw data saved by MapMate using the last of these options, letting them sort things out for themselves. In the illustrated examples later, I use the *OpenOffice* office suite which has several things in its favour. If you don't already own software such as *Microsoft Office*, you may well want to consider it for your own use.

- It's very good and very versatile, with capabilities generally matching and sometimes exceeding commercial software like *Microsoft Office*.
- It's completely free to download and use. (Although you have to pay for technical support.)
- It can generate Acrobat Portable Document Format (PDF) files, making it very easy to distribute your documents to other people, whatever their computer system.
- It can read and write a wide range of other document formats, including *Microsoft Office* documents.

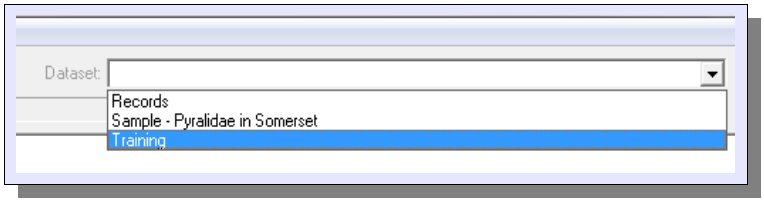
If you want technical support for *OpenOffice*, then you have to pay for it. But recently a group of independent software professionals have taken on its further development as a completely free package called *LibreOffice*. This has all the capabilities of the original, and then some more. Support comes from the community of developers and users through discussion groups.

If you do already have spreadsheet and word processing software, you should find it quite straightforward to apply the examples to your own circumstances. The process of moving data across is similar for most programs. In Chapter 9 I shall go into much more detail about exchanging data with other software, including other mapping software.

Using the Training Dataset

Finally, if you are going to follow our examples for querying MapMate data, you will need some data to query. I'm assuming that you haven't already entered up a large set of your own data; in any case, the examples will be easier to follow if you use the Training dataset. At this point, if you didn't install the Training dataset as described in Chapter 2, you will need to go back and follow the instructions there.

Now, on the main MapMate screen, place your cursor over the box to the right of the caption that reads **Dataset:** and click. You should see a drop-down list like the following.



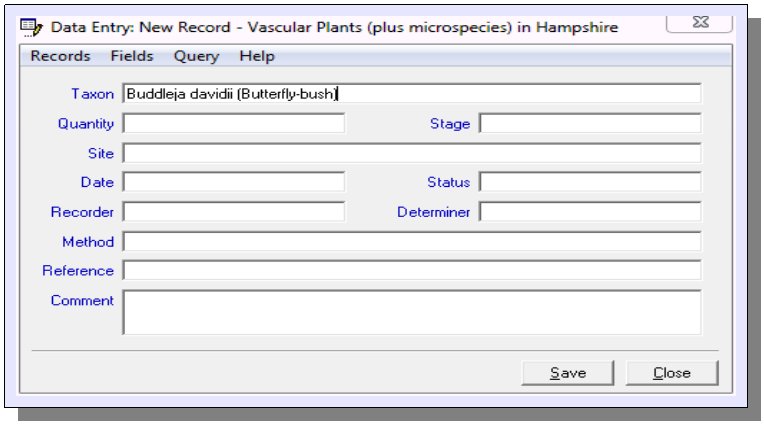
Select the **Training** dataset from the list. You will then have a set of about 50,000 records for South Hampshire (vice-county 11) to examine.

This ability to hold multiple data sets in MapMate is convenient for illustrative purposes like this, but not much use operationally, for several reasons.

- Closing and re-opening MapMate will always return you to the main **Records** dataset.
- Certain other operations within MapMate will do the same.
- You can only exchange data with other MapMate users via the **Records** dataset.
- You can only import data from non-MapMate sources into the **Records** dataset.
- There are no tools for migrating or sharing data between datasets.

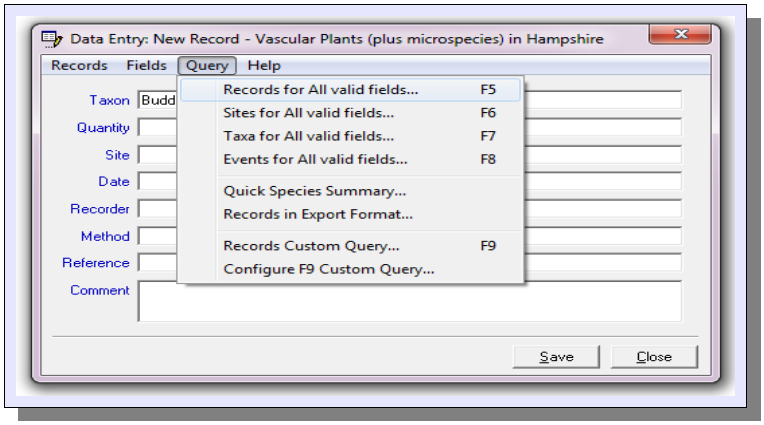
So, if you share data with other people with different interests (for instance, different geographical areas, different species groups) then keeping separate data sets is not really an option at present. There are other techniques for dealing with this, as we shall see in Chapter 5.

Simple Ad Hoc Queries: Listing Records



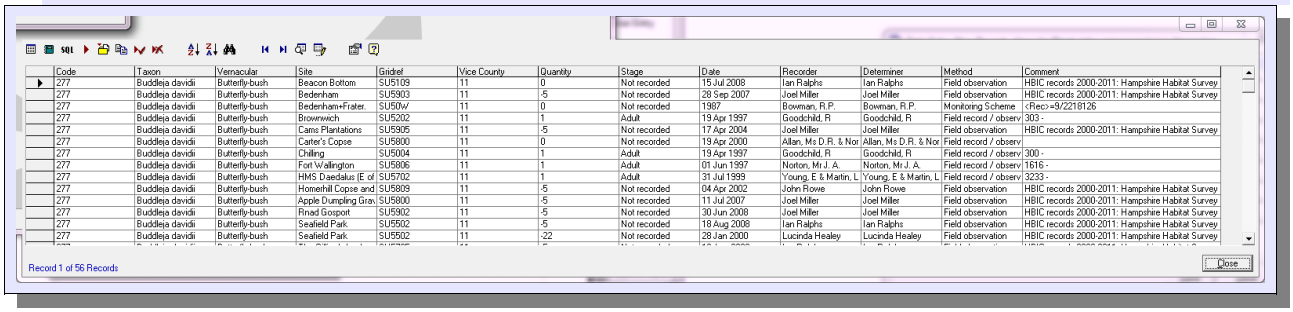
We are now going to look at some simple ways of listing data from MapMate, and for this you will need to re-open the Data Entry form that we used extensively in Chapter 3.

We'll start by inspecting all the records we currently hold for *Buddleja davidii*. Using any of the techniques you learnt in Chapter 3, select that into the **Taxon** field and make sure it is validated.



Now, from the **Query** menu at the top of the form, select **Records for All valid fields...** You'll see for future reference that there is a shortcut function key (F5) for this.

The Data Browser



Code	Taxon	Vernacular	Site	Gridref	Vice County	Quantity	Stage	Date	Recorder	Determiner	Method	Comment
277	Buddleja davidii	Butterfly-bush	Beacon Bottom	SU5109	11	0	Not recorded	15 Jul 2008	Ian Ralphs	Ian Ralphs	Field observation	HBIC records 2000-2011: Hampshire Habitat Survey
277	Buddleja davidii	Butterfly-bush	Bedenham	SU5903	11	-5	Not recorded	28 Sep 2007	Joel Miller	Joel Miller	Field observation	HBIC records 2000-2011: Hampshire Habitat Survey
277	Buddleja davidii	Butterfly-bush	Bedenham+Frater.	SU50W	11	0	Not recorded	1987	Bowman, R.P.	Bowman, R.P.	Monitoring Scheme	<Rec>=9/2218126
277	Buddleja davidii	Butterfly-bush	Brownwich	SU5202	11	1	Adult	19 Apr 1997	Goodchild, R	Goodchild, R	Field record / observ	303 -
277	Buddleja davidii	Butterfly-bush	Cams Plantations	SU5905	11	-5	Not recorded	17 Apr 2004	Joel Miller	Joel Miller	Field observation	HBIC records 2000-2011: Hampshire Habitat Survey
277	Buddleja davidii	Butterfly-bush	Carter's Copse	SU5800	11	0	Not recorded	19 Apr 2000	Allan, Ms D.R. & Nor	Allan, Ms D.R. & Nor	Field record / observ	
277	Buddleja davidii	Butterfly-bush	Chilling	SU5004	11	1	Adult	19 Apr 1997	Goodchild, R	Goodchild, R	Field record / observ	300 -
277	Buddleja davidii	Butterfly-bush	Fort Wallington	SU5806	11	1	Adult	01 Jun 1997	Norton, Mr.J. A.	Norton, Mr.J. A.	Field record / observ	1616 -
277	Buddleja davidii	Butterfly-bush	HMS Daedalus (E of	SU5702	11	1	Adult	31 Jul 1999	Young, E & Martin, L	Young, E & Martin, L	Field record / observ	3233 -
277	Buddleja davidii	Butterfly-bush	Homehill Copse and	SU5809	11	-5	Not recorded	04 Apr 2002	John Rowe	John Rowe	Field observation	HBIC records 2000-2011: Hampshire Habitat Survey
277	Buddleja davidii	Butterfly-bush	Apple Dumpling Gra	SU5800	11	-5	Not recorded	11 Jul 2007	Joel Miller	Joel Miller	Field observation	HBIC records 2000-2011: Hampshire Habitat Survey
277	Buddleja davidii	Butterfly-bush	Rnad Gosport	SU5902	11	-5	Not recorded	30 Jun 2008	Joel Miller	Joel Miller	Field observation	HBIC records 2000-2011: Hampshire Habitat Survey
277	Buddleja davidii	Butterfly-bush	Seafield Park	SU5502	11	-5	Not recorded	18 Aug 2008	Ian Ralphs	Ian Ralphs	Field observation	HBIC records 2000-2011: Hampshire Habitat Survey
277	Buddleja davidii	Butterfly-bush	Seafield Park	SU5502	11	-22	Not recorded	28 Jan 2000	Lucinda Healey	Lucinda Healey	Field observation	HBIC records 2000-2011: Hampshire Habitat Survey

Record 1 of 56 Records

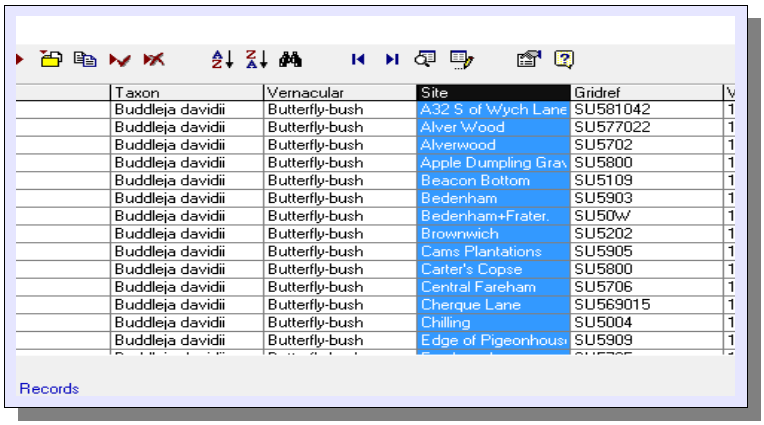
Close

You should now have something like the above on your screen – MapMate refers to this as the **Data Browser**. The grid which forms the main part of this shows one record in each row; the columns show the data items which MapMate considers the ones you are most likely to want to see. (We shall see how you can change that later.) Notice that the **Quantity** column shows the coded number that MapMate stores in its database, rather than the interpretation such as 'Rare'.

If you can't see all the information properly in any column, you can adjust its width. Move your cursor over the grey caption line at the top, and position it over the 'crack' between the right-hand end of your undersized column and the next column to the right. The cursor should change to a vertical bar with arrows pointing in either direction. Now push down the left mouse button and hold it down, while dragging the mouse to the right. A moveable bar appears, allowing you to widen the column. Let go the mouse button when you are satisfied. Of course this also works for making columns narrower.

Sorting and Searching

The first point of interest here is the row of icons arranged above the grid in four groups. Ignore the first group for now; we'll come back to these later. The second group is for sorting and searching the list. The first two sort in forward and reverse dictionary order; in order to use them, you'll first need to highlight a column to sort by.

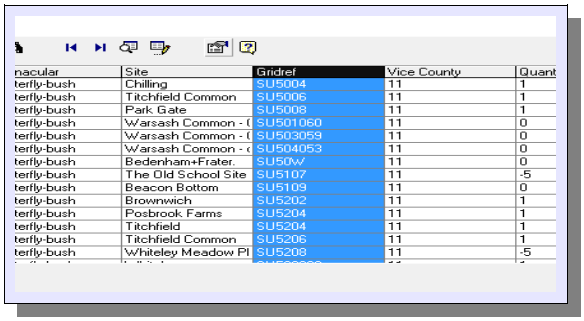


Taxon	Vernacular	Site	Gridref	V
Buddleja davidii	Butterfly-bush	A32 S of Wych Lane	SU581042	1
Buddleja davidii	Butterfly-bush	Alver Wood	SU577022	1
Buddleja davidii	Butterfly-bush	Alverwood	SU5702	1
Buddleja davidii	Butterfly-bush	Apple Dumpling Gra	SU5800	1
Buddleja davidii	Butterfly-bush	Beacon Bottom	SU5109	1
Buddleja davidii	Butterfly-bush	Bedenham	SU5903	1
Buddleja davidii	Butterfly-bush	Bedenham+Frater.	SU50W	1
Buddleja davidii	Butterfly-bush	Brownwich	SU5202	1
Buddleja davidii	Butterfly-bush	Cams Plantations	SU5905	1
Buddleja davidii	Butterfly-bush	Carter's Copse	SU5800	1
Buddleja davidii	Butterfly-bush	Central Fareham	SU5706	1
Buddleja davidii	Butterfly-bush	Cherque Lane	SU569015	1
Buddleja davidii	Butterfly-bush	Chilling	SU5004	1
Buddleja davidii	Butterfly-bush	Edge of Pigeonhous	SU5909	1

Records

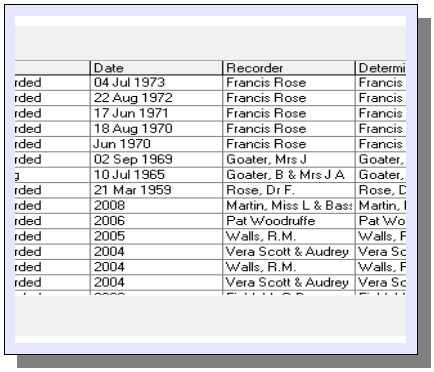
Do this by clicking anywhere in the column you want to sort by, or on the caption at the head of the column; for instance, if you want to see Site names in alphabetical order, click somewhere in the **Site** column. Then click on the first of the sorting icons.

You may be wondering whether you can select more than one of the columns and sort first on one, and then on another within that (for instance, by Sites and then by record dates at each Site). Unfortunately, within MapMate you can't use more than one column.



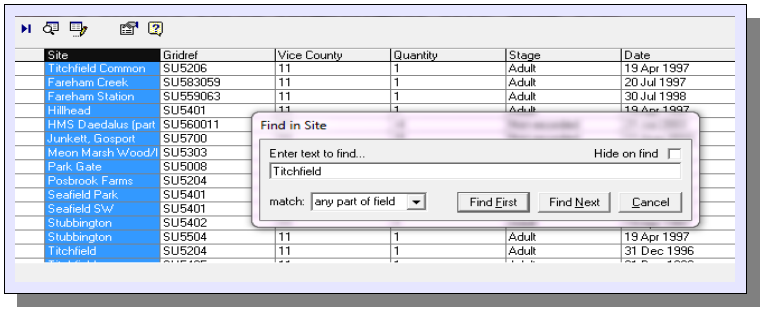
nacular	Site	Gridref	Vice County	Quant
terfly-bush	Chilling	SU5004	11	1
terfly-bush	Titchfield Common	SU5006	11	1
terfly-bush	Park Gate	SU5008	11	1
terfly-bush	Warsash Common - (	SU501060	11	0
terfly-bush	Warsash Common - (	SU503059	11	0
terfly-bush	Warsash Common - (	SU504053	11	0
terfly-bush	Bedenham+Frater.	SU50W	11	0
terfly-bush	The Old School Site	SU5107	11	-5
terfly-bush	Beacon Bottom	SU5109	11	0
terfly-bush	Brownwich	SU5202	11	1
terfly-bush	Posbrook Farms	SU5204	11	1
terfly-bush	Titchfield	SU5204	11	1
terfly-bush	Titchfield Common	SU5206	11	1
terfly-bush	Whiteley Meadow Pl	SU5208	11	-5

All the sorting is done strictly on alphabetic order by the text that appears in the browser. That may give you a couple of surprises. If you sort on grid reference and you have references at different precisions, the order won't make any sense geographically. For instance, in this illustration you will see that **SU5008** appears after **SU5006** (which is to the south of it), and before **SU501060** (which is also to the south of it). Similarly, **SU50W** is to the east of its predecessor **SU504053**, but also to the east of its successor **SU5107**.



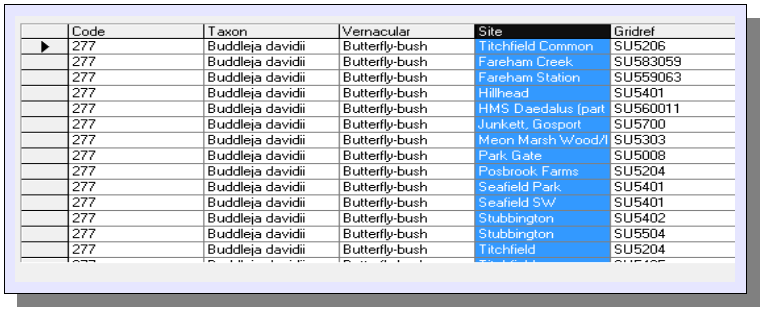
	Date	Recorder	Determi
rded	04 Jul 1973	Francis Rose	Francis
rded	22 Aug 1972	Francis Rose	Francis
rded	17 Jun 1971	Francis Rose	Francis
rded	18 Aug 1970	Francis Rose	Francis
rded	Jun 1970	Francis Rose	Francis
rded	02 Sep 1969	Goater, Mrs J	Goater,
g	10 Jul 1965	Goater, B & Mrs J A	Goater,
rded	21 Mar 1959	Rose, Dr F.	Rose, D
rded	2008	Martin, Miss L & Bas	Martin, I
rded	2006	Pat Woodruffe	Pat Wo
rded	2005	Walls, R.M.	Walls, F
rded	2004	Vera Scott & Audrey	Vera Sc
rded	2004	Walls, R.M.	Walls, F
rded	2004	Vera Scott & Audrey	Vera Sc

Also, if you try sorting by Date order, dates which comprise just a year date will sort out separately from exact dates. This illustration shows dates sorted latest first, using the second of the two sorting buttons. You can see that all dates signifying just a year sort out earlier than any date which is precise. You just need to be aware of this when scanning the list.



Site	Gridref	Vice County	Quantity	Stage	Date
Titchfield Common	SU5206	11	1	Adult	19 Apr 1997
Fareham Creek	SU583059	11	1	Adult	20 Jul 1997
Fareham Station	SU559063	11	1	Adult	30 Jul 1998
Hillhead	SU5401	11	1	Adult	19 Apr 1997
HMS Daedalus (part	SU560011				
Junkett, Gosport	SU5700				
Meon Marsh Wood/I	SU5303				
Park Gate	SU5008				
Posbrook Farms	SU5204				
Seafield Park	SU5401				
Seafield SW	SU5401				
Stubbington	SU5402				
Stubbington	SU5504	11	1	Adult	19 Apr 1997
Titchfield	SU5204	11	1	Adult	31 Dec 1996

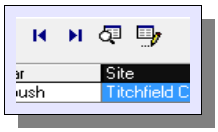
The third button in the group allows you to search for text in any of the columns. Again, you must first click somewhere in a column you want to search in. For instance, to find records in Titchfield, first highlight the **Site** column, then click on the 'binoculars' icon. Type in the text you are looking for; notice that if you want to be more strict in your selection, you have options to limit the result to records where your text appears as the first part or the whole of the name. This is like the use of the 'wildcard' (*) character we saw in Chapter 3 when searching for existing items like Taxa and Sites. You may be tempted to try more sophisticated searches using '*' and one of these options in combination. Only one of these really works – that is, two pieces of text separated by '* ' on any part of field will find anything that contains both fragments in the order specified. For instance, **alv*wood** finds records for both **Alver Wood** and **Alverwood**. From this you can also see that searches are case-insensitive.



	Code	Taxon	Vernacular	Site	Gridref
▶	277	Buddleja davidii	Butterfly-bush	Titchfield Common	SU5206
	277	Buddleja davidii	Butterfly-bush	Fareham Creek	SU583059
	277	Buddleja davidii	Butterfly-bush	Fareham Station	SU559063
	277	Buddleja davidii	Butterfly-bush	Hillhead	SU5401
	277	Buddleja davidii	Butterfly-bush	HMS Daedalus (part	SU560011
	277	Buddleja davidii	Butterfly-bush	Junkett, Gosport	SU5700
	277	Buddleja davidii	Butterfly-bush	Meon Marsh Wood/I	SU5303
	277	Buddleja davidii	Butterfly-bush	Park Gate	SU5008
	277	Buddleja davidii	Butterfly-bush	Posbrook Farms	SU5204
	277	Buddleja davidii	Butterfly-bush	Seafield Park	SU5401
	277	Buddleja davidii	Butterfly-bush	Seafield SW	SU5401
	277	Buddleja davidii	Butterfly-bush	Stubbington	SU5402
	277	Buddleja davidii	Butterfly-bush	Stubbington	SU5504
	277	Buddleja davidii	Butterfly-bush	Titchfield	SU5204

If you click on **Find First**, MapMate will display the first record in the list matching your criteria by bringing it on display in the Data Browser window and putting a little chevron by it. There are no prizes for guessing what **Find Next** will do.

Navigation

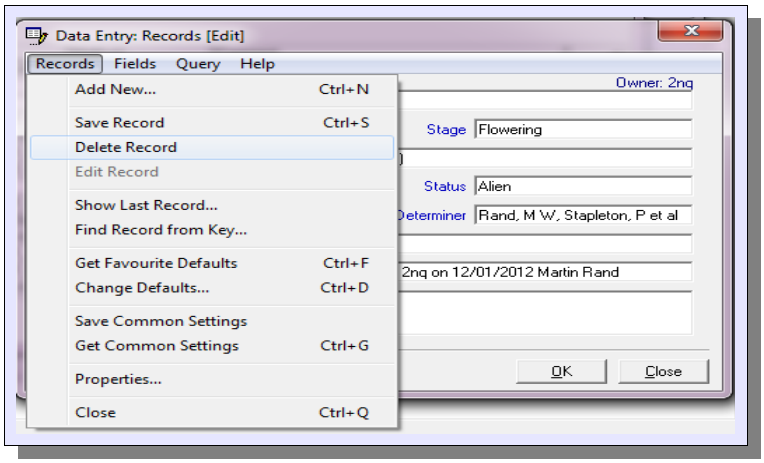


Now let's look at the third group of buttons on the Data Browser. These allow you to navigate quickly to one end or other of the list, and to extract an individual record with full detail. The chevron pointing to the left goes to the beginning of the list; that pointing to the right goes to the end.

The next button is for viewing the complete detail for a record, and it requires you to select a record in the grid first. You can do this simply by clicking anywhere on the record in the grid, or by using the facilities for finding a record that we introduced in the previous section. Clicking on the icon brings up a Data Entry form for the record, but it is in 'view only' mode; you won't be able to change any of the detail.

Modifying and Removing Records

The final button in this group also gets you to the Data Entry form, but in this case you **will** be able to edit the existing data. If you hold only your own data on your MapMate database, you will always be able to do this. If you exchange data with other people, however, you may notice that the icon is 'greyed out' at times as you navigate through the list of records. This indicates that this is someone else's record, and you are not allowed to edit it. MapMate enforces the principle that only the person who entered the record has the right to modify it. This is a good principle in general, but can make life rather awkward when someone dies or for other reasons gives up maintaining their own data. Fortunately this problem can be overcome, as we shall relate in Chapter 10.



Once the Data Entry form is opened, you have all the facilities for entering and changing data that you had when you first entered it. At this point I would like to point out a feature in the **Records** menu that we mentioned briefly in the last chapter: **Delete Record**. This will remove that single record from your database.

Before making any deletion, it's worth pausing and taking a deep breath. There is no 'Undo' function in MapMate, so if you remove records by mistake you will have to go through the process of re-entering them. Once you have read Chapter 6 you will know the value of regular backups, but you still have to weigh the effort of recreating deleted records against the loss of any information you entered since the last backup.

Deleting records singly from the Data Entry form is all very well, but there will be times when you want to remove a whole batch at once. Also, you may have reasons for wanting to remove records entered by someone else, and as we've just seen, you can't do this by first going to the Data Entry form. Fortunately there is another technique that covers both these eventualities.

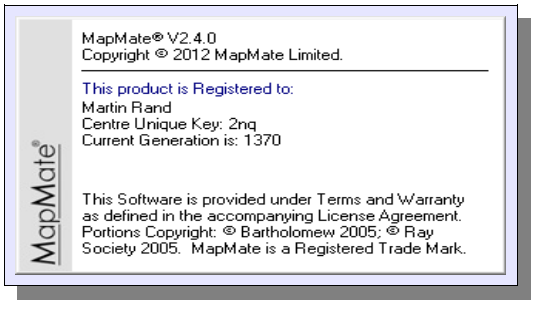
What Does Removing a Record Mean in MapMate?

You may be surprised to learn that removing a record actually means two different things in MapMate. Which one comes into play depends on who entered the record in the first place – you, or someone who sent you data using the standard MapMate data exchange facility, which we'll learn more about in Chapter 5.

For now, let's be clear about what this implies. If someone else actually made the record in the first place, but you typed it into your copy of MapMate, then they are the Recorder of the record, but **you** are the owner. You have the ability to modify that record. But if they entered the record on their copy of MapMate and then sent it to you using MapMate's data exchange facilities, they are the owner.

When you remove a record that **you** own, MapMate doesn't actually take it out of your database. It blanks out a lot of the information, but it leaves behind a 'skeleton' record with a special marker to say it has been deleted. Why? So that if you then pass on information to other people through MapMate's data exchange, the record will also be removed from their copies. MapMate can recognise the 'removed' record and send an instruction that the equivalent record should be taken out at their end.

You can also remove records from other owners from your own copy of MapMate, if you are not satisfied with the record or if for some reason you end up with duplicates. But then, MapMate actually does delete the entire record from your own data. This does not get passed on to any of the people you share data with; only your copy of MapMate will ever be affected. This is sometimes convenient, especially if you have an urgent report to produce or you can no longer make contact with the record owner. But if there is something wrong with the record, it's always best to get the owner to put it right at source and then exchange the update with you. That ensures that anyone whom they, and you, exchange data with will eventually get the amendment.



If you do exchange data with others, by now you may be wondering how you can be sure who owns a particular record when you are looking through a large archive. This is determined by a coded piece of information called the 'Centre Unique Key' ('CUK' for short) that is attached to every record, of whatever type, held in MapMate. If you can't remember your own CUK, click on the **Help** menu of the MapMate main screen and select **About MapMate...** This will show you who you are.

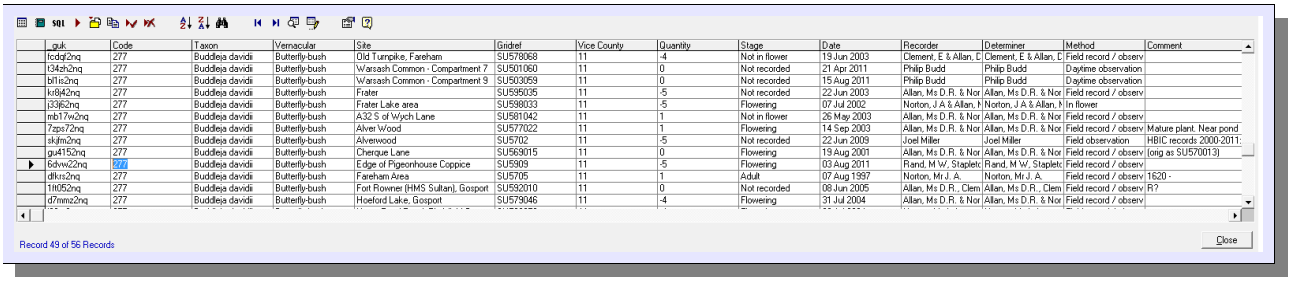
While looking at the Data Browser, you can find out whether you are the owner of a given record in the list by highlighting it, then using the **View Record** button. The owner of a record is always shown in the top right-hand corner of the record viewing / editing form.

Record Identities

If you want to check ownership of a whole batch of records, bringing them up on view individually can get tedious. There is another trick you can use to see the owners of all the records in the Browser. You have already seen how to alter the width of columns on the

Browser. Now move your cursor to the crack on the **left** side of the left-most column (in this case, **Code**). The cursor should again change to a vertical bar, similar to when you are adjusting the width, but with just one arrow pointing to the right. Hold down the left mouse button as before and drag the cursor rightwards, far enough to allow a dozen or so characters to display in a new column.

When you release the button, the Browser should be showing something like this.



The strange garble that appears in the newly revealed column is what MapMate calls the 'Global Unique Key' of the record, or 'GUK' for short. MapMate manufactures one of these for every record created in its database and, as you might hope from the name, guarantees its uniqueness throughout all the universe of computers running MapMate. I talked in Chapter 2 about the use of pointers in databases so that each biological Record didn't have to keep its own copy of all the Site information, Recorder information and so on, potentially leading to massive duplication. GUKs are what MapMate uses as pointers. The first five characters of the GUK are arbitrary as far as we're concerned, but the last three are the Centre Unique Key (CUK) of the copy of MapMate where the data was entered, or if you prefer, the identity of the Owner. Since any individual copy of MapMate can't know what arbitrary five-character sequences of GUK codes have already been assigned on all other computers running MapMate, sticking the CUK on the end is its way to guarantee that the code really is unique for all MapMates. And it means that you can look at a display like this and recognise whether it is one of the records you own, or one entered and sent on to you by someone else. In Chapter 5 we'll also get to see how you find out which 'someone else'.

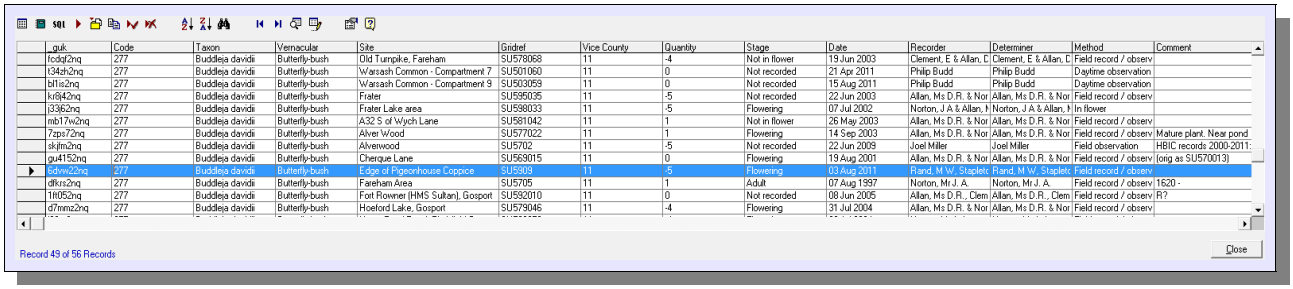
Just in case you were curious, this method of producing a unique key means that you have a theoretical limit of over 6 billion data records on your computer. But in fact, limitations in the underlying database mean that you would never get near this number even if you had super-recorder powers. The practical limit at present depends on how much detail you include in your records, but is probably going to be around 10 million Records with their associated data – and that includes both your own and those that other people have sent you. On this reckoning, if no MapMate user ever shared data with another, that would give a potential world-wide capacity of 450 billion Records – probably enough spare to allow quite a lot of sharing, in fact.

Displaying the GUK like this works for many of the queries you can run in the Data Browser, but not all, as we shall see when we come to the **Analysis** menu and Stock Queries. It depends on whether the designers of the report saw fit to include the GUK as part of the information to be returned. If they didn't, you will find another restriction on the Data Browser; you won't be able to view or edit the individual record, or carry out the removal

operation we're about to describe. Without the GUK as a unique 'handle' to grab the relevant information by, MapMate can't guarantee to choose the right record every time.

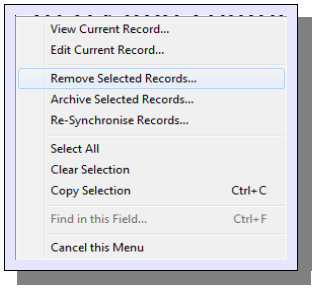
Back to Removing Records...

Now we shall see how to select one or more records to remove from your data. First you need to highlight the entire record showing in the Browser. Do this by clicking on the grey rectangle to the left of the data.



From here you can go on to select multiple records, either adding further records to your selection one at a time by holding down the Control key as you click; or selecting a single block of records by holding down the Shift key and clicking alongside the last one you want to include in the block. If you've used other Windows programs with similar facilities, you may want to try and combine these methods to select multiple discrete blocks of records; but you'll find that doesn't work in MapMate. What you can do, if you select a record in error, is Control-click again alongside the record to deselect it.

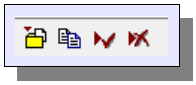
Having made your selection, you then need to click with the right mouse button (unless you're left-handed like me and have switched buttons around), somewhere over the actual detail of the selected record or records.



That should bring up a pop-up menu like this. You can then choose the **Remove Selected Records...** option. (I don't recommend that you do this right now with the Training data set.) If you don't get the pop-up, your cursor was probably hovering over the wrong part of the Browser display.

Getting Your Data Out Of MapMate

You can't do much more with the data in the Data Browser than what we've seen so far; it's really a tool for quick inspection and for managing your data set. You can't format your data into a nice-looking report; you can't do extra calculations and derivations from what you see on screen.

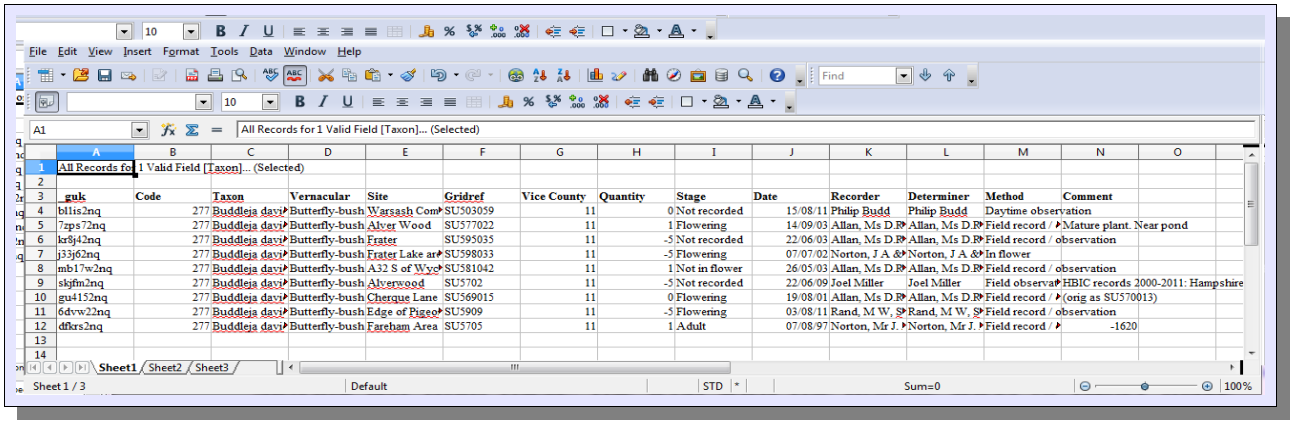


So now it's time to look at the facilities for getting that data out of MapMate and into another application. This is part of the first group of buttons on the Browser toolbar. We've already seen how to select individual records, or a contiguous run of records, while learning how to remove records just now. The two red chevrons on the

right provide extra shortcuts; the tick selects all the records listed in the Browser, the cross deselects them all.

Moving left from these, the icon with the double sheet of paper makes a copy of the selected records to the Windows clipboard. From here you can paste them into any other program that can handle such data. To demonstrate this, I suggest you now open up your favourite spreadsheet program (I'm opening up *OpenOffice Calc.*) Make a selection of some or all of the *Buddleja* records we've been displaying, and click on this button.

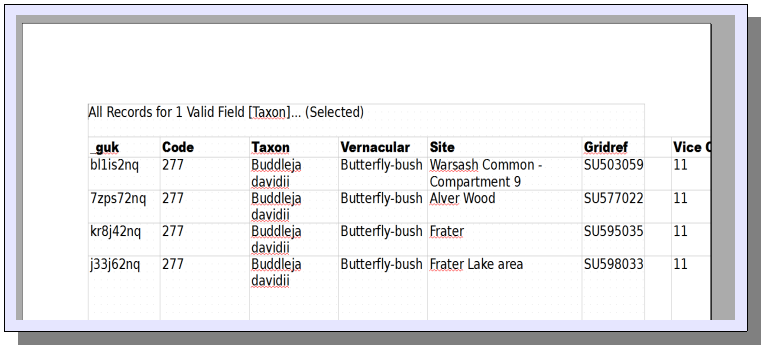
Now go to the spreadsheet and use the Windows 'Paste Clipboard' function. It's likely to be **Edit / Paste** on your menu bar, or **Control-V** as a shortcut key. This is what I get; your layout may be a little different.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	All Records for 1 Valid Field [Taxon]... (Selected)														
2															
3	_guk	Code	Taxon	Vernacular	Site	Gridref	Vice County	Quantity	Stage	Date	Recorder	Determiner	Method	Comment	
4	bl1is2nq	277	Buddleja davi	Butterfly-bush	Warsash Com	SU503059	11	0	Not recorded	15/08/11	Philip Budd	Philip Budd	Daytime observation		
5	7zps72nq	277	Buddleja davi	Butterfly-bush	Alver Wood	SU577022	11	1	Flowering	14/09/03	Allan, Ms D.R	Allan, Ms D.R	Field record /	Mature plant. Near pond	
6	kr8j42nq	277	Buddleja davi	Butterfly-bush	Frater	SU595035	11	-5	Not recorded	22/06/03	Allan, Ms D.R	Allan, Ms D.R	Field record /	observation	
7	j33j62nq	277	Buddleja davi	Butterfly-bush	Frater Lake ar	SU598033	11	-5	Flowering	07/07/02	Norton, J A &	Norton, J A &	In flower		
8	mb17w2nq	277	Buddleja davi	Butterfly-bush	A32 S of Wyc	SU581042	11	1	Not in flower	26/05/03	Allan, Ms D.R	Allan, Ms D.R	Field record /	observation	
9	skjfm2nq	277	Buddleja davi	Butterfly-bush	Alverwood	SU5702	11	-5	Not recorded	22/06/09	Joel Miller	Joel Miller	Field observa	HBIC records 2000-2011: Hampshire	
10	gu4152nq	277	Buddleja davi	Butterfly-bush	Cherque Lane	SU569015	11	0	Flowering	19/08/01	Allan, Ms D.R	Allan, Ms D.R	Field record /	(orig as SU570013)	
11	6dvw22nq	277	Buddleja davi	Butterfly-bush	Edge of Pigeo	SU5909	11	-5	Flowering	03/08/11	Rand, M W, S	Rand, M W, S	Field record /	observation	
12	df4rs2nq	277	Buddleja davi	Butterfly-bush	Fareham Area	SU5705	11	1	Adult	07/08/97	Norton, Mr J.	Norton, Mr J.	Field record /	-1620	
13															
14															

Mapmate always puts one or two lines of header at the start of the data when it's copied this way, which document how it was derived. You may not want this, but obviously it's an easy matter to delete the rows.

You'll see that MapMate sends the column captions over at the head of the data. If you didn't go through the Intermediate section on record identities earlier on, you may be wondering what the column headed **_guk_** is. If you got this and weren't expecting it, you can just remove the entire column. If you did follow through that section, be aware that the GUK is exported whenever you've displayed it in the Data Browser as described there. If you didn't display it, you won't get it.



All Records for 1 Valid Field [Taxon]... (Selected)

_guk	Code	Taxon	Vernacular	Site	Gridref	Vice C
bl1is2nq	277	Buddleja davidii	Butterfly-bush	Warsash Common - Compartment 9	SU503059	11
7zps72nq	277	Buddleja davidii	Butterfly-bush	Alver Wood	SU577022	11
kr8j42nq	277	Buddleja davidii	Butterfly-bush	Frater	SU595035	11
j33j62nq	277	Buddleja davidii	Butterfly-bush	Frater Lake area	SU598033	11

If you copy the clipboard into a word processor such as *Microsoft Word* or *OpenOffice Write*, you will get a text table, like this. With many of the standard MapMate reports, the number and size of columns mean that it won't fit conveniently onto a standard page width, and you will often have to play about with page layout, column widths and font sizes in your word processor to get the desired effect.

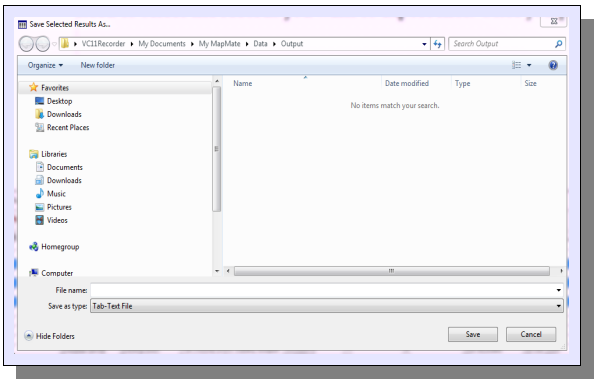
What you do with the data once you've got it out of MapMate is of course only limited by your ingenuity and the capabilities of the program you put it into. For example, MapMate does have some facilities, that we'll see shortly, for totalling up population sizes across

Records, but it fails to handle the negative values that are used to code abundance estimates properly. In a spreadsheet you can use formulae to weed these out from 'proper' numbers. We won't go into more details at the moment, but Chapter 9 has a number of illustrative tasks that you may find useful in themselves, or that suggest new avenues to you.

When you are copying data from the MapMate Data Browser to the clipboard, you may be tempted to use the **Control-C** shortcut key, as you do in just about any other Windows program, instead of the toolbar button. But this often doesn't give you what you expect – if it gives you anything at all. Also, don't expect the data you copied to stay in the clipboard for later once you've closed the Data Browser window. It doesn't – Mapmate tidies it away.

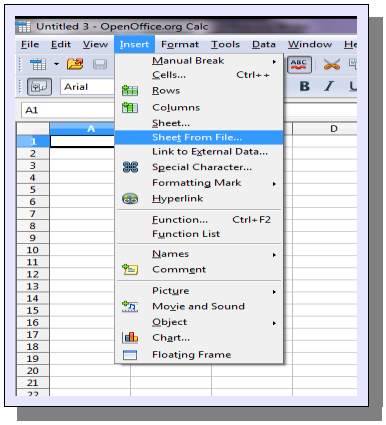
Another Way to Transfer Data

The final button in this group (sheet of paper going into a folder) is also used to export data to the outside world, but by creating a file on your computer that can then be re-used at leisure. To use this you will also need to select the records you want to export first.



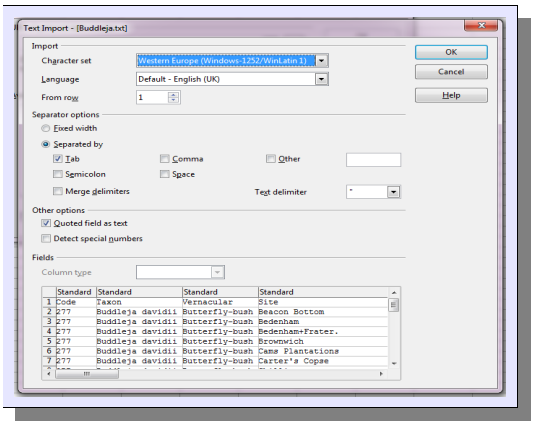
Once you press the button, MapMate will ask you where to save the file and require you to supply a file name. By default, it will put the file into the **Output** sub-folder of your MapMate **Data** folder. You can change this, and I usually do, as I want to distribute data to various places for different purposes. Rather annoyingly, if you do change the destination MapMate doesn't remember this, even in the same session; you have to declare it again for each file you save.

The contents of the file can also be brought into other office applications such as spreadsheets. Here's how to do it in the case of *OpenOffice Calc*. There are similar facilities in Microsoft *Excel*, although the techniques are slightly different.

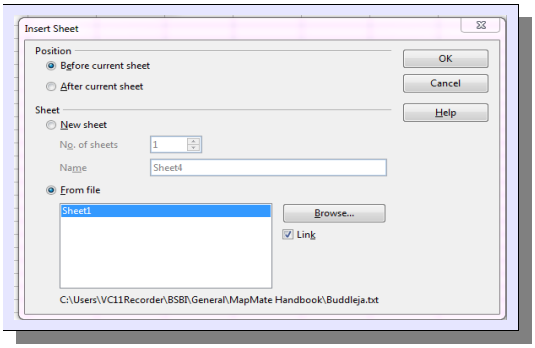


From the **Insert** menu select **Sheet from File...**

Then select the file you have just saved from MapMate.



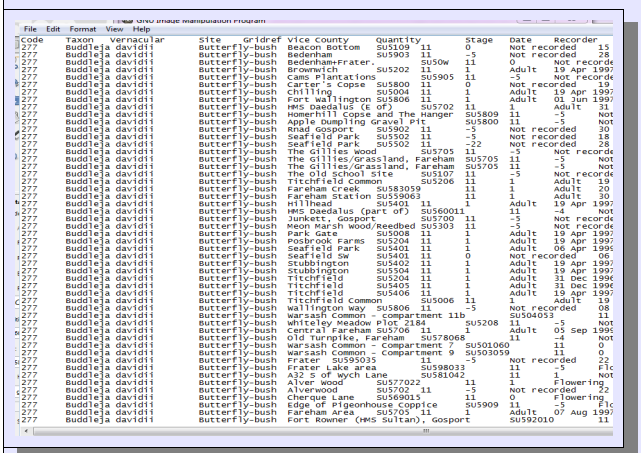
The software then examines the content of the file and makes decisions on how it has been formatted. These are presented in the form shown here. In my case the assumptions are all correct and I can go ahead and click on the **OK** button. Depending on circumstances and the program that you are using, you may get a different set of assumptions; in that case, change the options until they look like this. The important thing to check is that the sample data layout shown at the bottom is organised in columns in the way that you would expect from MapMate.



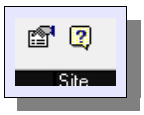
OpenOffice Calc then expects you to put these into a new worksheet. If you tick the box marked **Link**, then when you save your spreadsheet it will keep a record of the link to the original file from MapMate. This means that if you want to provide an update to the data, all you need to do is overwrite the same text file, in the same location on your computer, with a new run of the report in MapMate. When you next open your spreadsheet, it will ask you whether you want to update the link.

The MapMate Data Export Format

The format of file that MapMate generates when you export data from the Data Browser is known as 'Tab-delimited Text'. It's a fairly unsophisticated file format, but it does the job it's required to do – that is, move rows and columns of data from one program to another, preserving distinctions between text, numbers and dates as required. Because it's a text file, it's reasonably readable in its raw state by humans and you can look at the contents with a text editor such as *Notepad* which comes with Windows. Here's a part of one. You'll see that it includes the column headings from MapMate as the first row. Each row occupies one line in the text; each column is separated from its neighbours by a 'Tab' (tabulator) character. Because the amount of text for any given column differs in different rows, and because text editors like Notepad aren't concerned with fancy layout, things don't line up nicely. But if you get a file that doesn't load properly into your spreadsheet or other document, it can be useful to look for things in the data that have upset the format – like text lines breaking in the wrong place. Sometimes users manage to get unexpected characters into the Comment field of a Record, that affect the layout.



Changing the Filters on Reported Data



You'll remember that when we embarked on data entry, we used **Change defaults...** to limit the taxonomic and geographical areas of interest that we were currently working on. The Data Browser has a similar facility in the fourth group of buttons. This is the 'hand on document' icon.

When you make a change to your defaults, any data currently on display in your Data Browser doesn't automatically refresh to take account of the new settings. To see any changes, you need to re-run the query. To do that, you don't need to go back to where you started; in the first group of buttons there is a right-facing chevron that will simply redisplay whatever you originally asked for, using the new criteria.

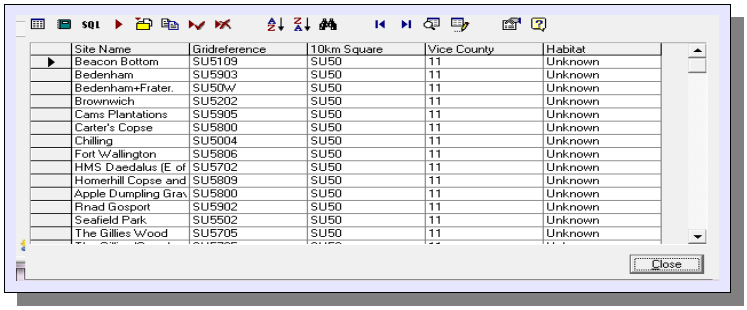
When you change the default filter settings on the areas of interest, they apply to anything you are currently doing in MapMate, wherever you do it from. For instance, if you change them here in the Data Browser, they will affect the picking lists and validation checks that operate on any data entry you subsequently do.

Simple Ad Hoc Queries: Other Types of Data, Other Ways of Showing Data

At this point, you know most of the things you can do with the Data Browser. All the operations you have learnt about up to this point can be applied to anything that appears in the Browser grid, regardless of the method by which you put it there. The next sections of this chapter will look at other ways of extracting that data, starting with other ad hoc queries.

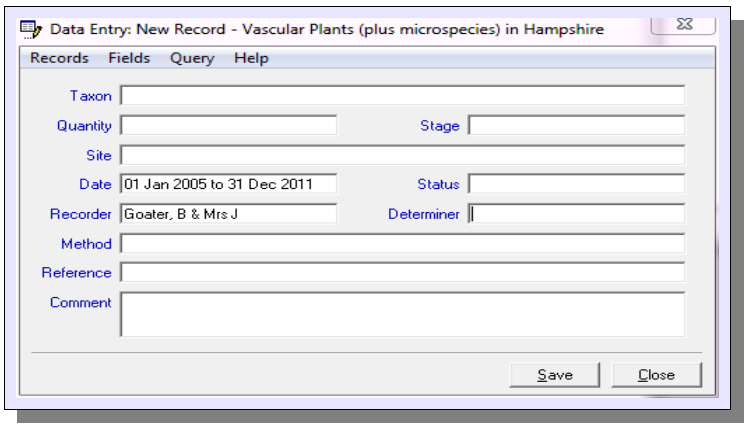
Listing Sites

Close down the Data Browser and return now to the Data Entry form. Suppose that you don't want to see every individual record for Butterfly-bush, but you would like a list of the Sites where it's been recorded. Make sure you still have *Buddleja davidii* as a valid Taxon entry. Select **Query** from the form's menu bar, and this time choose **Sites for All valid fields...**

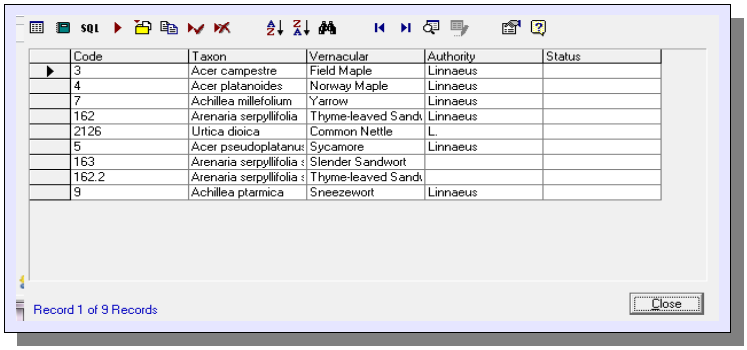


This should bring up the Data Browser again, but now you will see a different set of data.

Listing Taxa



Let's now say we want to see all the Taxa that have been recorded by a given recorder during a given period. Clear the **Taxon** field on the Data Entry form, and type **2005 to 2011** into the **Date** field. (Don't forget to press **Enter** to validate it!) Type **Goater** into the **Recorder** field, and press **Enter**. From the list offered, select **Goater, B & Mrs J**. Your Data Entry form should now look like the illustration to the left.

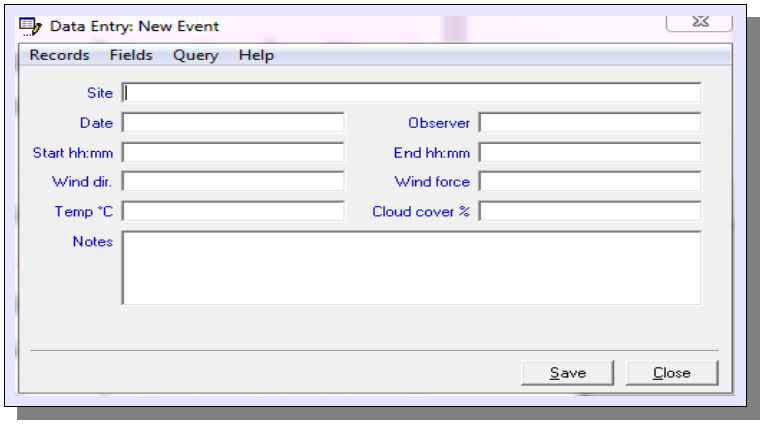


Now select **Taxa for All valid fields...** from the **Query** menu. You should see something like this in the Data Browser.

There are many other combinations of selection criteria with the three top queries in the **Query** menu list, and I won't labour the point; you can explore them for yourself, knowing that you can't do any harm to the data just by running the queries. If you try to run **Taxa for All valid fields...** with a single Taxon selected in the form, you won't get anything in the Data Browser; the listing would obviously be trivial. The same goes for Sites and **Sites for All valid fields...** .

Events

The fourth item in the **Query** menu, **Events for All valid fields...**, brings us to something that we didn't cover in Chapter 3. MapMate allows you to record details of a specific recording event, separately from the Site, Recorder and Date details you put into the record. This is really a lot more use for people who record mobile organisms like birds or butterflies.



If you want to see what it offers, select **Add New...** from the Data Entry form's **Records** menu, then select **Events**.

However, even if you do decide this is a wonderful thing, there are still shortcomings in MapMate; it looks like a half-implemented feature.

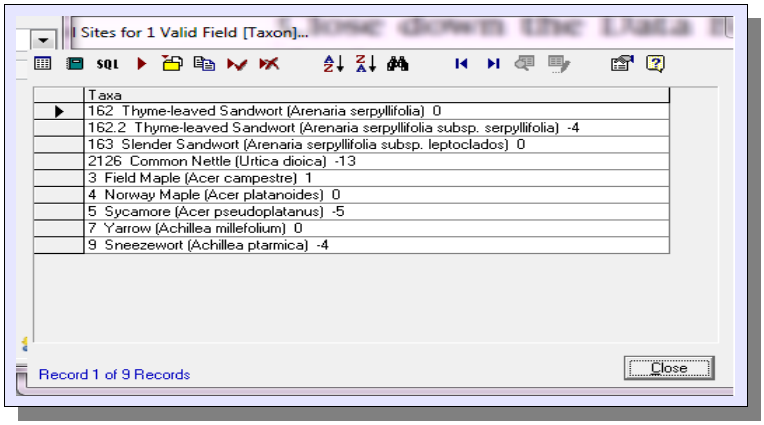
- First, there's no neat trick for linking a Record to the Event during which it was recorded. You'll notice that there is no 'Associated Event' item on the Record Data Entry form, and there is no other fully automated way to associate a Record with an Event.
- Instead, it is up to you to make the link, by making the Site defined in the Record identical to the Site defined in the Event, the Date defined in the Record identical to the Date defined in the Event, and (I would say, although MapMate doesn't suggest this in its Help files) the Recorder defined in the Record identical to the Observer defined in the Event.
- Having done all that, there is still no standard way in MapMate to report on all the Records associated with a given Event. The Help files suggest how you can create your own report, but this involves advanced techniques, discussed at the end of this chapter - something you may not want to tackle for yourself. In fact there is an easier way, using the **Custom Query** facility from the Data Entry **Query** menu, which we'll come to shortly. But whichever way you tackle it, there are weaknesses in how the related data is brought up. One you may have already spotted is that there's no means of naming an Event with a meaningful name like 'The Great British Garden Willow-herb Watch 2012'. (OK, you can put it in the **Notes** box, but then it's just free

text and it's easy to make a slight mis-spelling when you try to use the same name for another record.) If you had two Events on the same day at the same Site at different times of day, they would both get listed in the reports offered; that can be overcome if you're prepared to make further modifications to the query that MapMate suggests you build for yourself, but not to the Custom Query available here. Alternatively, you may prefer to export the data to a spreadsheet and do the filtering or segregation there.

Suffice it to say that over several years of using MapMate, I've never felt the need to make use of the Events facility for botanical recording.

Quick Reports

Earlier we listed all the Taxa for a given recorder and date range. There is a 'quick and dirty' way to get a similar but abbreviated report; having entered your selection criteria, use the **Quick Species Summary...** item in the **Query** menu.



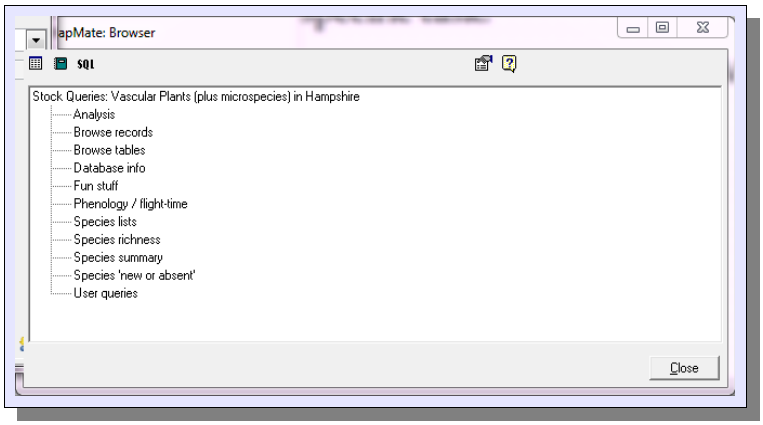
This puts together various items from the data (BRC recording code, vernacular name, scientific name, population total) into a single column. The 'quick' bit of this will be self-evident; for the 'dirty' bit, note first of all that records are sorted by their BRC code, but not in any sensible way; large numbers that start with '1' or '2' come before the numerically smaller '3', '4' etc. Then look at the counts at the end. Because the recorders have often used abundance codes in their records, these cause the totals to go negative, because MapMate doesn't filter these out. Those are the obvious errors. But does the Field Maple total represent one record with a single individual, or does it (say) represent one record with a total of 5 individuals, one record where it was recorded 'Occasional' (-4), and three more records where it was recorded simply as 'Present' (0)?

Again, this is a facility I've rarely felt the need to make use of. There are better ways of handling the data concerned. But as a way of getting a list out rapidly into another document you may find value in it.

Now we shall leave the **Query** menu of the Data Entry form for a while. It's very simple to use as far as it goes, but it has obvious limitations on what you can report on and how you can select subsets of data. The next section will introduce you to the battery of additional report formats that MapMate provides. You can close the Data Entry form at this point.

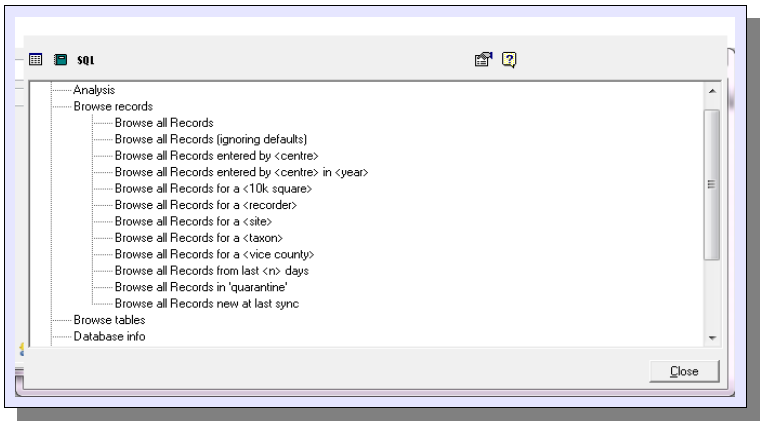
Stock Queries and the Analysis Menu

Sooner or later you will find that querying from the Data Entry form doesn't give you something you want, especially if you what you want is to summarise your data. Your next port of call is the **Analysis** menu from the main MapMate window. Here you will find a large number of pre-designed reports (referred to in MapMate as 'Stock Queries') each doing one specific task.

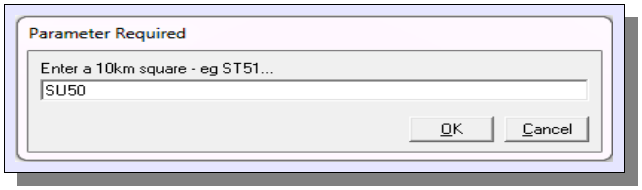


Click on **Analysis** on the MapMate main toolbar, and this is what you should see. The Stock Queries are arranged in categories, and if you double-click on a category, the display will expand to list all available queries under that category.

Browsing Records: an Introduction

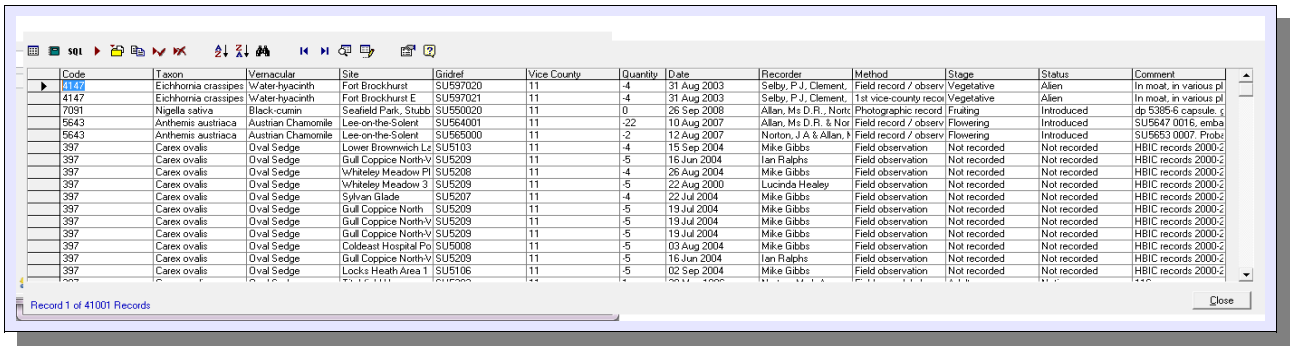


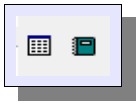
Let's start by expanding the **Browse records** category. You'll notice that several of these appear to duplicate things you could do from the Data Entry **Query** menu. We'll pass over those for the moment, and have a look at one of the others.



Start with **Browse all Records for a <10k square>**. This will prompt you for a 'Parameter': in other words, a piece of information you supply up front which then becomes a given for the operation you're about to perform. It's the equivalent of one of the items you filled in on the Data Entry form before you ran a query from there. When you run an Analysis query, however, you will be prompted for them one by one.

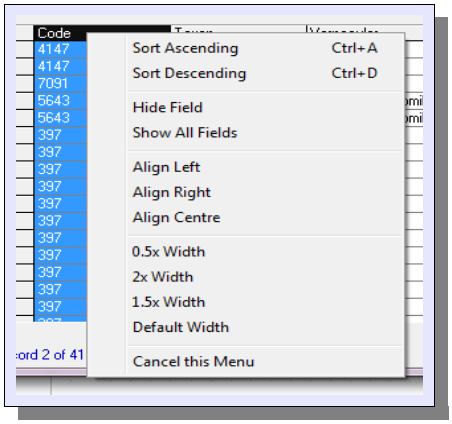
Enter the 10km square illustrated above, and click on **OK**. This will bring up the familiar Data Browser window.





At this point it's worth introducing a couple of the toolbar buttons that we haven't seen yet, over on the far left-hand side of the first group. The 'binder' icon on the right (**View Query List**) will take you back to the list of Stock Queries if you want to run another one. If, having got there, you decide you want to return to the last results you displayed, use the 'grid' icon on the left (**View Results**).

I'll also mention a few more things you can do with the layout of the grid, since what you get by default is what MapMate thinks is good for you. To carry out these operations, you need to position your cursor over the **caption** above a column, then click with the right mouse button.



You should see this pop-up menu; try it for **Code**.

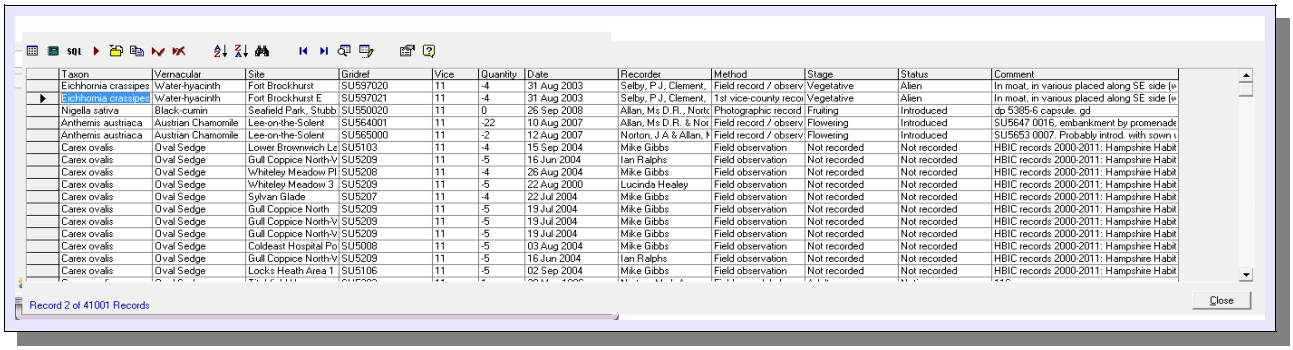
I'm rarely interested in seeing the BRC code for a Taxon, except when I'm entering data from a BRC recording sheet. So I will select **Hide Field** from this menu.

The **Vice County** column is taking up a lot of space, so I'll call up the same menu for that column and choose **0.5x Width**.

It would be more readable to have the **Quantity** values right-aligned, so in this case I use **Align Right**.

Finally, it would be nice to see a bit more of the **Comment** column, so I'll use **2x Width**. You might think that you could then use it again and get four times the original width, but sadly it doesn't work like that; all these scaling operations work against a 'default width' that MapMate has worked out for the column, so you just get the same result. If you want more radical expansions or finer tuning, use the the method described earlier in this chapter to drag the right-hand edges of the column in or out.

If you follow through these steps, you should end up with the Browser grid looking like this. (By the way, if you want to do bulk adjustments to several columns at once, you can do this by using the standard Windows Control-click and Shift-click facilities for multiple selection.)

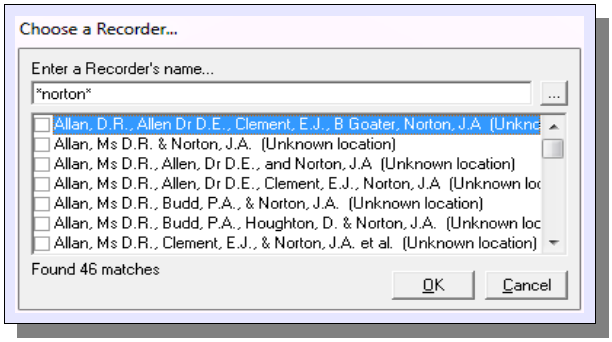


When you hide certain columns like this, you will find a bonus if you are exporting Browser data for use in another program like a spreadsheet. None of the hidden columns are exported. So this is a simple way of suppressing data that you don't want from a Stock Query.

If you want to run the same Stock Query again but with a different parameter, you don't need to go back to the catalogue of queries. For instance, to get the list of records for SU42, simply click on the right-pointing chevron (**Re Run Query**) to the right of the 'SQL' icon. MapMate will once again prompt you to enter the 10km square to report on.

Getting Parameters to Work for You

Return to the catalogue of queries, and this time select **Browse all Records for a <recorder>**. I take records from many MapMate users, and I inherited a large database of records from my predecessor. As a result, the same person is often identified by several forms of name, and also crops up in combination with other people. How can I get to see all the Records that one individual participated in recording? We couldn't have done it through the Data Entry **Query** menu; there we had to select just one incarnation of the Recorder name.



In the **Analysis** queries, you have the chance to enter an ambiguous name. When prompted for a Recorder name, try typing ***norton*** (in other words, anything with 'norton' anywhere in the name). That should bring up a pick list of candidate names like this.

If you have only two or three specific names of interest, you can tick the checkboxes against those names and click on **OK**. That will ensure that you get Records for those names only. Unfortunately, once you tick more names than this, the retrieval process doesn't seem to filter them correctly.

However there is a work-around that allows you to retrieve **all** the matches. Click back in the box that reads ***norton***, and then click **OK**. That will retrieve all records for any Recorder with 'norton' anywhere in their name, in upper or lower case.

This has its shortcomings, as will be apparent if you have other Recorders called 'D L Norton' or 'Mrs J A Norton-Byker'. However it will usually get you close enough that you can export your records to a spreadsheet and filter them down to what you want without too much pain.

Browsing Records: Other Useful Facilities

I don't intend to go through all the Stock Queries for browsing Records; you can explore these for yourself. But there are a few that are worth pointing out. Go back to the list of queries.

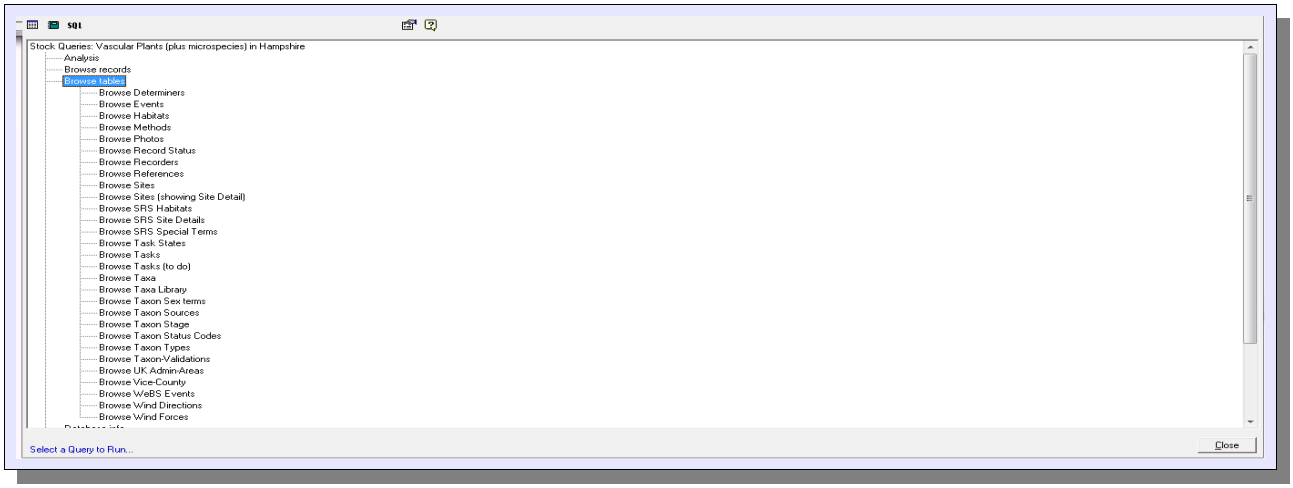
Browse all Records (ignoring defaults) will by-pass whatever filtering you have in place for taxonomic groups and geographical areas. If you record molluscs as well as plants, for instance, you will see those in the Data Browser, even if you've set your defaults for Vascular Plants.

Browse all Records entered by <centre> allows you to look at those records entered up on a specific MapMate user's computer. To use it, you need to know the Centre Unique Key of the person concerned. You can find out your own by selecting **Help** then **About...** from the MapMate main menu. If you exchange data with other people, we'll see in the next chapter how to find other people's.

Browse all Records new at last sync is useful if you exchange data with other people and you want to review what they have recently sent you. We'll talk more about this in the next chapter, but any time you go through an exchange of data (which MapMate refers to as a 'sync') the event is recorded. This report isn't quite perfect for the purpose outlined, since along with the data they sent you, you will also see any data you yourself have entered since you accepted theirs; and if you don't keep a strict 'receive and send' exchange protocol going with your partners, you may see that mixed up with data you received earlier from other partners. But it is generally good enough.

Browsing Other Data

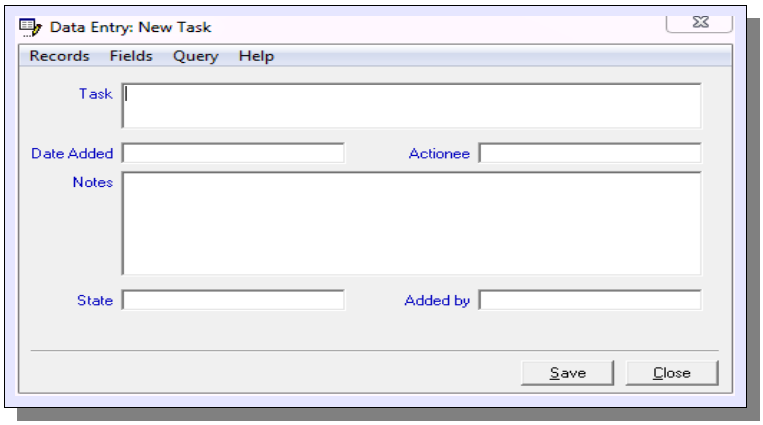
Return now to the list of queries, and expand the **Browse tables** category. You will see that there are reports for all the categories of data other than Records. These break down into data that you enter as part of your own information (such as Sites and Recorders), and reference data provided by the system (for instance, Taxa and UK Admin Areas).



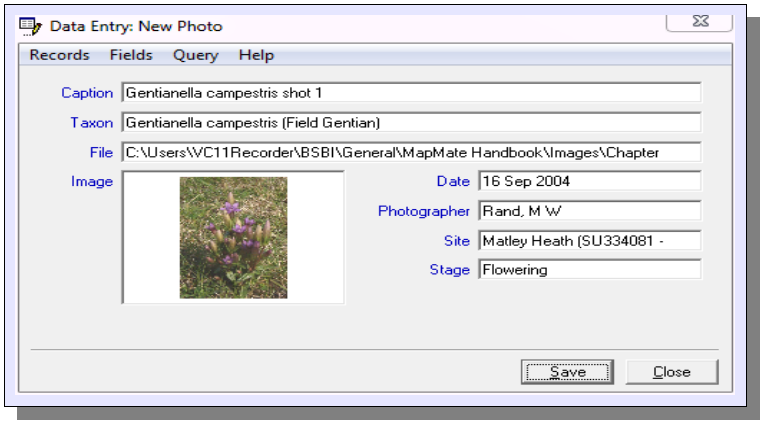
I won't go into detail exploring these; you can do that for yourself. There will be a few things that are unfamiliar to you in this list. Some of them have to do with specific national recording schemes such as the Spider Recording Scheme.

An Aside on Some Other Data Recording Facilities

A few of the data categories listed are facilities in MapMate that we haven't discussed up to this point. I'll mention a couple of them now. If you want to put your own information into these, then open the Data Entry form, select **Add New...** from the **Records** menu, and select the category of choice.



Tasks: MapMate provides a very simple Task Manager that does nothing more than list things you need to do. As you'll see, there is no way of setting a timetable for a task, and consequently no way of getting MapMate to alert you about deadlines. Don't try and envisage complicated workflows, either: the options for **State** are 'To Do' and 'Completed'.



Photos: MapMate provides the means to include photographs in your data archive. The photograph is associated with a Taxon, a Date, a Recorder (i.e. **Photographer**), and a Site, which opens up the possibility of querying your photo archive using any combination of these criteria.

It would be nice if one could do this simply from the **Query** menu of the Data Entry form, but unfortunately there isn't an entry there for Photos. Since the Analysis query list only includes a blanket one for retrieving all Photos, that means you would have to develop a set of selective queries the hard way – which we shall come to later.

You could type the file path and name in full in the **File** entry on the Photos form, but you will probably want to use the **List Possibles** (**F2**) facility. Rather than bringing up a list of known files, this displays a browser for the folders on your computer. MapMate doesn't recognise all image file formats (for instance, TIFF files are not supported), but most of the main ones including BMP, GIF and JPEG are there.

Once you've selected a file, a thumbnail will appear in the Image window as shown. Double-clicking on this image will open it in the image viewing software that is the default for this purpose on your computer.

MapMate's photo facilities offer rather less than many photo organiser programs, or even Windows' own file management, and are rather more cumbersome to use. Before committing yourself to it, it's worth considering the alternatives.

Other Groups of Queries

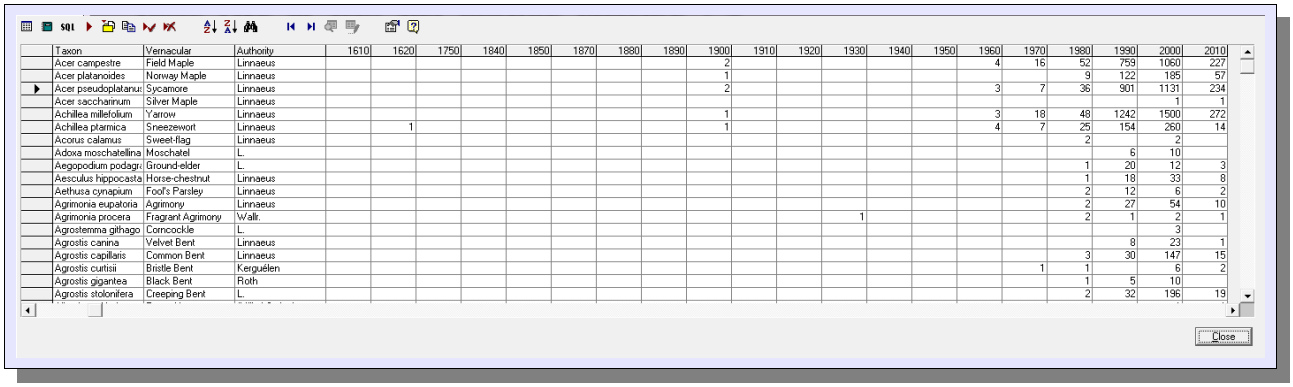
Database info: We shall skip over this category for now. It includes some very useful facilities if you are trying to trace the history of data exchange with colleagues, but we'll come back to that in Chapter 5.

Fun stuff may stretch your definition of fun a bit, but more importantly, it is one of several groups that include a report of abundance (**Top 20 most abundant species**). Unless every record you make is a formal census of "non-repeating" individuals such as butterflies, this serves little purpose. Not only is it biased by including the 'Present' designation and all the abundance codes in the calculation, it takes no account of the fact that the same individuals may be re-recorded on successive visits – clearly a likely occurrence in botany. If you run this query against the Training dataset I think you will soon see what is wrong with it. The same goes for any other query with an abundance statistic in it.

With this caveat, I can leave you to explore **Species lists**, **Species richness** and **Species new or absent** for yourself. The last of these includes several queries that are useful for highlighting records for verification, and some that are handy when you are compiling annual reports.

Phenology / flight-time is problematic. For one thing, the reports that claim to show number of records in a given period actually show the number of individuals. Since, as elsewhere, these take in zero and negative presence / abundance codes, they give no sort of meaningful profile where these have been used, and consequently can't be used for phenology graphs. Also, if any of your records use 'year only' or 'year range' dates, MapMate interprets these as specific dates running from 1st January to 31st December. The queries provided make no allowance for this, so your reports will be polluted by spurious late and early dates.

It's worth looking at one of the **Species summary** reports in a bit more detail, **Summary by species by decade**. If you run this, you should get a result like this (after adjusting column widths).



Taxon	Vernacular	Authority	1610	1620	1750	1840	1850	1870	1880	1890	1900	1910	1920	1930	1940	1950	1960	1970	1980	1990	2000	2010
Acer campestre	Field Maple	Linnaeus									2						4	16	52	759	1060	227
Acer platanoides	Norway Maple	Linnaeus									1								9	122	185	57
Acer pseudoplatanus	Sycamore	Linnaeus									2						3	7	36	901	1131	234
Acer saccharinum	Silver Maple	Linnaeus																			1	1
Achillea millefolium	Yarrow	Linnaeus									1						3	18	48	1242	1500	272
Achillea ptarmica	Sneezewort	Linnaeus		1							1						4	7	25	154	260	14
Acorus calamus	Sweet-flag	Linnaeus																	2		2	
Adoxa moschatellina	Moschatel	L.																		6	10	
Aegopodium podagra	Ground-elder	L.																	1	20	12	3
Aesculus hippocastanum	Horse-chestnut	Linnaeus																	1	18	33	8
Aethusa cynapium	Fool's Parsley	Linnaeus																	2	12	6	2
Agrimonia eupatoria	Agrimony	Linnaeus																	2	27	54	10
Agrimonia procera	Fragrant Agrimony	Wallr.												1					2	1	2	1
Agrostemma githago	Corncockle	L.																			3	
Agrostis canina	Velvet Bent	Linnaeus																		8	23	1
Agrostis capillaris	Common Bent	Linnaeus																	3	30	147	15
Agrostis curtisii	Bristle Bent	Kerguélen																1	1		6	2
Agrostis gigantea	Black Bent	Roth																	1	5	10	
Agrostis stolonifera	Creeping Bent	L.																	2	32	196	19

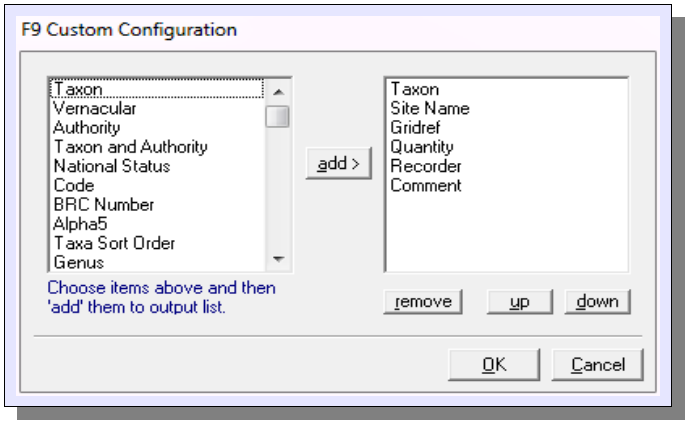
Close

This is an example of a style of report often referred to as a 'Crosstab' (i.e. cross-tabulation) or, more obscurely, a 'pivot table'. It takes two discrete arguments (in this case, individual Taxa and year ranges) as the axes of the table and displays a statistic for matching values (such as a count, sum or average) at each position. Mercifully, in this case it uses a count of records rather than a count of individuals. I'm sure that the resulting display is perfectly intuitive to grasp and doesn't need further explanation; but as we shall see later in the

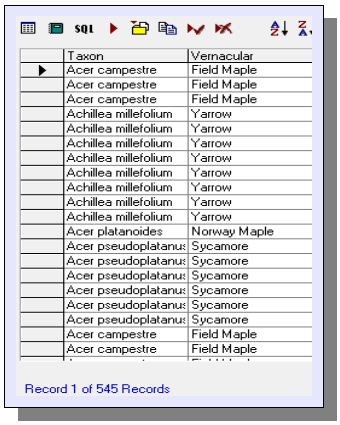
chapter, it is possible to construct this kind of report for yourself if you don't mind grappling with the technicalities.

Tailor-made Reports for Records: the Easy Way

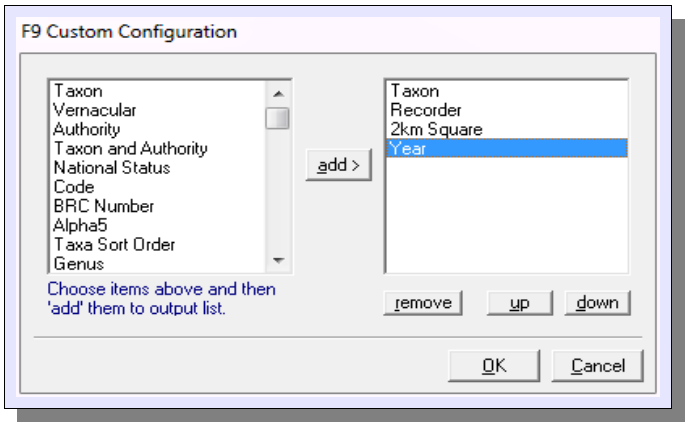
There may come a point when you would like something different from anything the Data Entry **Query** menu or the **Analysis** queries can give you. There are two ways of tackling this in MapMate, and we're now going to look at the first, called the Custom Query. It's simple, but it's very limited in its capabilities.



Open up the Data Entry form, and from the **Query** menu select **Configure F9 Custom Query...** You should then see a form like this. This is where you get to select which items of data you would like to appear on your report. The right-hand window shows what is currently reported; the left-hand window shows what is available to report.

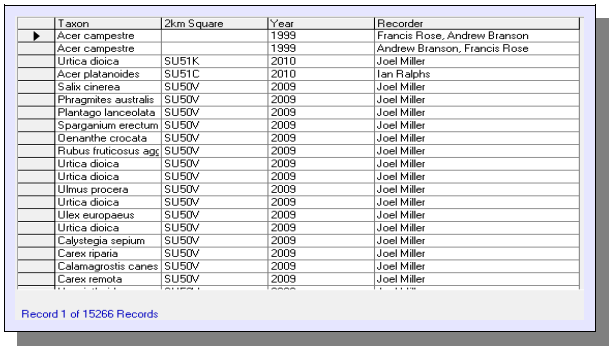


The first thing to be aware of is that the Custom Query is a report on Records, and not on any other type of data. Even if you remove all the items of data from the right-hand side that relate to the specific recording incident and then run the Custom Query, you wouldn't be able to get, for instance, a straightforward Sites or Taxa report. In this illustration I've removed everything from the Custom Query list except **Taxon** and **Vernacular**, and then run it against a specific Recorder, but it still provides an entry for every separate Record.



With that understood, we can now proceed to create a tailor-made query. Let's assume that we would like a rather more summary report of Records, that lists just the **Taxon**, the tetrad (**2km Square**), the year of the Record, and the **Recorder**. First, select in turn the **Site Name**, **Gridref**, **Quantity** and **Comment** items on the right-hand side (or whatever else is showing there, apart from Taxon and Recorder) and click on the **Remove** button for each. Then hunt through the list on the left for **2km Square** and **Year** and use the **add>** button to put them on the right-hand side, as shown.

The order in which the items on the right appear in the list determines the order in which the columns appear left to right in the Data Browser. Try putting the Recorder down to the bottom of the list by highlighting **Recorder** and clicking on the **down** button twice. Then click **OK**.



	Taxon	2km Square	Year	Recorder
▶	Acer campestre		1999	Francis Rose, Andrew Branson
	Acer campestre		1999	Andrew Branson, Francis Rose
	Urtica dioica	SU51K	2010	Joel Miller
	Acer platanoides	SU51C	2010	Ian Ralphs
	Salix cinerea	SU50V	2009	Joel Miller
	Phragmites australis	SU50V	2009	Joel Miller
	Plantago lanceolata	SU50V	2009	Joel Miller
	Sparganium erectum	SU50V	2009	Joel Miller
	Oenanthe crocata	SU50V	2009	Joel Miller
	Rubus fruticosus agg	SU50V	2009	Joel Miller
	Urtica dioica	SU50V	2009	Joel Miller
	Urtica dioica	SU50V	2009	Joel Miller
	Ulmus procera	SU50V	2009	Joel Miller
	Urtica dioica	SU50V	2009	Joel Miller
	Ulex europaeus	SU50V	2009	Joel Miller
	Urtica dioica	SU50V	2009	Joel Miller
	Calystegia sepium	SU50V	2009	Joel Miller
	Carex riparia	SU50V	2009	Joel Miller
	Calamagrostis canes	SU50V	2009	Joel Miller
	Carex remota	SU50V	2009	Joel Miller

Record 1 of 15266 Records

Now we're ready to put our Custom Query into action. Let's try listing all the Records that have been recorded simply as 'Present' on the database, with neither a specific quantity nor an estimate of abundance. In the Data Entry form, type **P** in the **Quantity** box and press **Enter**. That should bring up **Present** in the box. Now from the **Query** menu select **Records Custom Query...**, or press the **F9** function key. You should see a listing like this in the Data Browser.

And that just about covers the capabilities of the Custom Query. There is a very long list of data items you can include, and although the query only reports on Records, many of these clearly come from related data tables like Sites and References. Sometimes the name in the configuration form makes this obvious, sometimes it doesn't, and some of the names there won't be familiar to you yet. My advice is: if you're not sure what a given data item is, put it in the Custom Query and see what you get!

Apart from the fact that you can only report on Records by this means, there are other obvious limitations. There are no summarising facilities like those we've seen in some of the **Analysis** Stock Queries. And MapMate only allows you one Custom Query at any one time. There's no way to save one under its own name and then develop another one; so if you have several requirements for reporting this way, you will need to fiddle with the configuration form each time you switch.

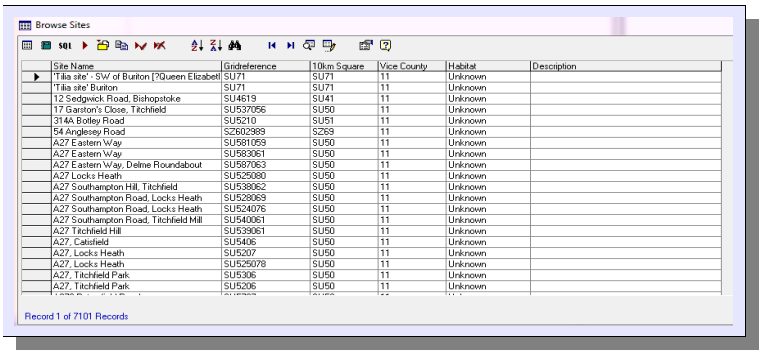
You can go quite a long way with the Custom Query with little effort, so it's worth making it your first port of call for any report that fits into its mould. For any more sophisticated enhancements to reporting, you will need to consider User Queries, which are the subject of the rest of this chapter. Fortunately there is a large library of custom queries available through MapMate support sites on the Internet, and it's always worth investigating these first; there's no point in re-inventing the wheel. We shall give pointers to some of these in the next section. If you don't find what you want, you may be able to get help from the "MM-Users" Yahoo group, which has several very knowledgeable and generous contributors. After that, you will have to consider crafting your own; but be warned that this demands a step change in your computer skill level.

User Queries: Tailor-made Reports the Hard Way

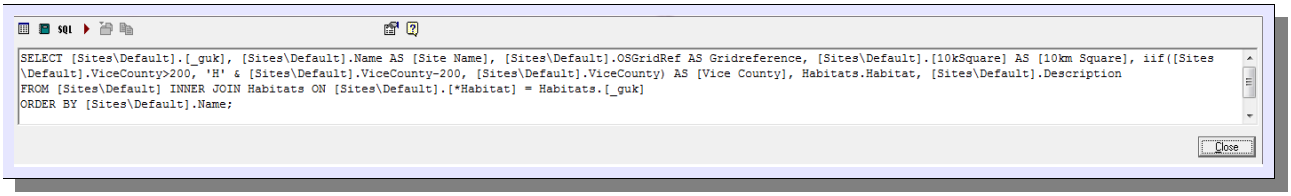
A Bleak Introduction (but it gets better later)

The **Analysis** menu provides you with a long list of predefined queries, as you have seen; but it also allows you to develop your own. That's the good news; now let's go on from that.

Bring up the **Analysis** menu again from the MapMate main window. Notice that the last of the headings for query categories is **User queries**; but don't go there just yet. Instead, go to **Browse tables** and select **Browse Sites**.



There's one button in the left-hand group that we haven't yet looked at; that's the **SQL** icon. Click on this now.



If you've never met this sort of thing before, you are probably wondering what on earth it can be. A little background explanation is due.

Structured Query Language

You will be well aware that if you want a computer to do something for you that someone else hasn't already designed and implemented, you need to talk to it in a language it can understand. Such languages are formal, and their rules must be followed strictly.

There is no end of computer languages, but MapMate uses a general-purpose database management system (currently Microsoft *Access*) for storing your data, and there is one very widely used language for manipulating databases: Structured Query Language, or SQL for short. Many different database systems can interpret instructions passed to them in SQL to retrieve data, manage data or even modify the structure of the database. Each database supplier may implement a slightly different dialect of SQL, but the core features are much the same.

SQL is not a particularly rich or extensive computer language, but its constructs can sometimes seem complex and hard to follow – especially if you're a novice.

So, almost anything that a program like MapMate does to its database eventually comes down to passing instructions in SQL to the database software. For most of the things you do with MapMate, such as data entry or removing records, this goes on in the bowels of MapMate and is well hidden from you.

When it comes to making your own queries, it's a different story. Here's the bad news:

- Most general-purpose database management systems these days provide facilities called 'query by example'. Instead of writing SQL programming code directly, you are shown a schematic of the data in the database. From this you can select the data items you want to include on your report, fairly easily specify how you want the data ordered, and set up criteria for selecting a subset of data. The system then writes the SQL for you from your specification. MapMate has a very limited range of 'query by example' tools, which you've already met; these are the **Query** facilities from the Data Entry form. That's as far as it goes.
- If you want something more than that, you will be obliged to write your own SQL code, either from scratch or by taking one of the existing Stock Queries, making a copy of it and then modifying it.
- General database management systems let you do that, too; a database developer or administrator will probably be fluent in SQL, but the system will help them out by imposing a proper layout on SQL instructions, and when they do make a mistake, by giving clear hints as to where it is and what the nature of the problem is. MapMate expects you to get your own layout correct to the letter, and if you make a mistake it isn't always very explicit. Sometimes it just tells you that you've made an error - not what or where. However, both these aspects seem to have improved a lot in recent versions, so it's worth keeping up to date with the latest version if you're going to develop queries.
- You may be wondering whether a general-purpose database management package can be used to query the MapMate database instead of writing SQL? This is certainly possible, but a design feature of MapMate makes it harder than it might be. Many database designs wrap up all of their data tables within a single database. MapMate puts different tables to one of several different databases - which, for *Access*, means different files on Windows. For example, all the data relating to Records live in a separate database from all the data related to Sites. (There are reasons for this, partly to do with maximising the space available for data.) Many of the general-purpose programs for managing and reporting data won't cope with this, since they will only let you deal with one database connection at a time. I'm not aware of a way of overcoming this in *OpenOffice*, for instance. The same goes for other free and open-source solutions. The best options are probably a professional report generator such as Crystal Reports, which allows simultaneous access to more than one database, or Microsoft Access itself, which allows you to set up links from one database to another. Clearly these require an investment of money; and if you're not already familiar with them, of time.

Manipulating data out of MapMate without having to write your own SQL is a topic we'll return to in Chapter 9. Just in case you choose to go ahead meanwhile and experiment with a tool such as *Access*, there are a couple of important caveats.

- The aim of the exercise is to retrieve data. General-purpose tools will allow you also to modify data, delete data and change the very organisation of the database. **Don't be tempted to do any of those things.** Unless you know exactly how MapMate handles its own data, you run a real risk of making it unusable.

- MapMate uses an old *Access* database format; later versions have changed the format to exploit new features. If you open a MapMate database in one of the later versions of the program, it will probably ask you whether you want to update it to the new format. **Never, ever do that**. If you do, you will no longer be able to use the database in MapMate.

Some Good News: The Kindness of Strangers

If you decide that writing computerese is not for you, there is still one other strategy you can pursue: get someone else to do it for you!

A very good starting point is the Web site of Martin Harvey, who is the producer of a set of outstanding training videos that complement this book. At his Web site (http://sites.google.com/site/kitenetter/Home/mapmate/sql) you will find the largest available collection of extra queries available for download.

Then there is a *Yahoo!* discussion forum on the Internet called *MM-Users,* accessible through the Web site http://uk.groups.yahoo.com/. It is a moderated group and you will need to sign up to it: the email address for subscribing is MM-Users-subscribe@yahoogroups.com.

There are several extremely helpful and knowledgeable people on this forum (including Martin Harvey), but they also have professional lives to lead. The etiquette for seeking help is:

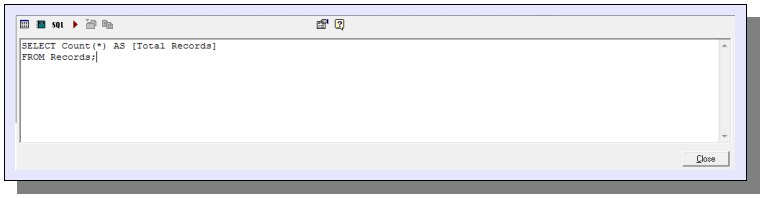
- Check whether Appendix B to this document contains a solution.
- Check whether your problem has been solved by Martin, or by a previous posting to the forum. A huge number of common requirements have been discussed already. Also check the 'Files' area for additional queries.
- State your needs as specifically and clearly as you can.
- Make sure that you are familiar with the mechanics of pasting queries into MapMate, as described in the next section of this chapter. A lot of problems arise from stray formatting characters like line breaks creeping into copied text.

I would also recommend that you read the next section after that, even if you don't intend to write your own queries. If you have some understanding of the rudiments of a SQL instruction, it will help you to do your own checking if something goes amiss when copying another person's query.

How to Create a User Query from Someone Else's Copy

In the **Analysis** Stock Query list, navigate to **User queries**, open it up and then select **<new user query>**.

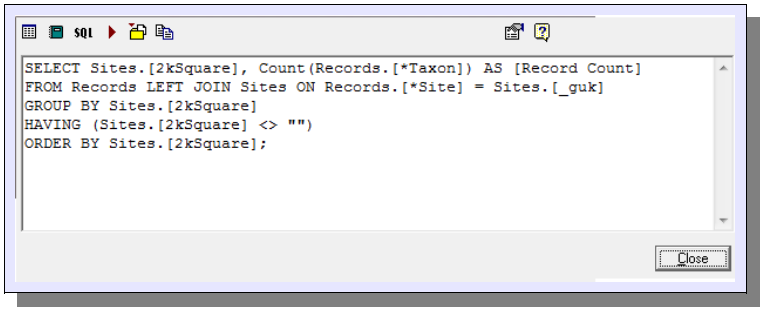
You will be prompted for a name for the new query: in the example I'm about to give, a suitable name would be **Count records by tetrad**.



The SQL editing window will appear, displaying a basic query. Highlight all the text. Press the **Delete** key to clear the SQL window.

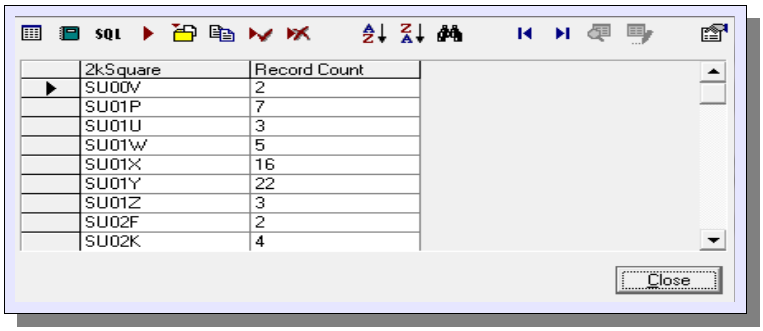
Now copy the text from your source document to the clipboard. In the example I'm giving, copy the following block of text.

```
SELECT Sites.[2kSquare], Count(Records.[*Taxon]) AS [Record Count]
FROM Records LEFT JOIN Sites ON Records.[*Site] = Sites.[_guk]
GROUP BY Sites.[2kSquare]
HAVING (Sites.[2kSquare] <> "")
ORDER BY Sites.[2kSquare];
```



Paste this into the SQL editing window; the result should look like this. Be careful not to omit anything. If you have a very small editing window open, line breaks may appear where the text wraps around from one line to the next; but otherwise, try to avoid extra line breaks. In fact, the later versions of MapMate seem to handle extra and missing line breaks pretty well as long as they don't appear in the middle of "words", but earlier versions were more fussy.

Now click the **Save Query** button, followed by **OK**. If you do get an error message, you will need to go back and carefully examine the code for any transcription errors.



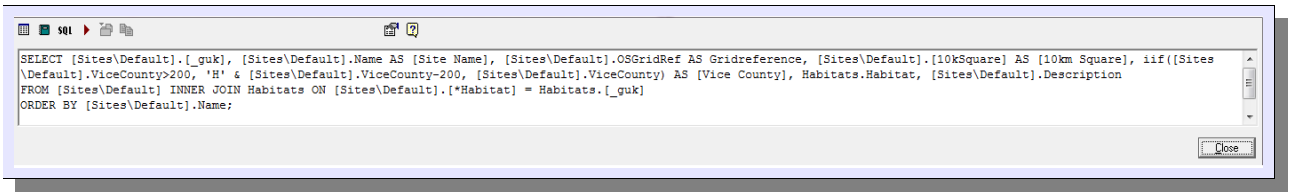
Finally, click on the **Re Run Query** button to see the results. Hopefully you will get something like this illustration from our example.

Anatomy of a SQL Query

Now it's time to learn something about the language itself. Although SQL is a highly artificial language governed by strict rules of syntax, it does have some similarities to a natural language. Each query is a statement in the form of an instruction. Like commands in English, it can be very short and simple ("Get out!") or long, detailed and with several qualifying clauses ("If you've finished all the exercises, place your exercise book before leaving in the red box I'm going to put on the table at the back when we've cleared up the mess from this experiment.")

The main difference is in the latitude you have for expressing the same intention in different ways. Clauses in a SQL statement always have to appear in the correct order, although some of them are optional and so needn't appear at all in a given instance. A clause has to be introduced with a specific term (a "keyword") that unambiguously establishes what kind of clause it is. And all references to objects like data items in the database have to use the exact phrasing and wording that was assigned to them by the database designer, with no ambiguity and no spelling mistakes.

To put some flesh on this, let's go back to the **Browse Sites** example we started the section with. This query is made up of three clauses, and each starts on a new line in the text.



Selecting Data Items

The first clause in this statement is introduced by the **SELECT** keyword. **SELECT** states that we want to retrieve some information from the database (our current purpose being to display it, but that's not always the case).

Everything that follows on this same line and the next specifies what we're going to retrieve, and in some cases what we'll do with it before returning it for display.

Each item of data is specified by a name couplet with a dot ('.') between the two elements. The element before the dot says where in the database to look for the item of data. Often this is the name of a database table, and you will see the name **Habitats** used here; that is one of the database tables built into MapMate.

But the other one (**[Sites\Default]**) is a bit subtler. You can also nominate an existing query (as long as it's a **SELECT** query that returns data itself) as the place to look for the data item. That means that you can get data pre-digested for you by the query. This is a built-in query that you won't find in the **Analysis** query list. If we used **Sites** (the database table) rather than **[Sites\Default]**, the **Browse Sites** query would look at all the Sites defined in our database, regardless of where they were. The **[Sites\Default]** query limits the search to the ones we said we were interested in when we set the Defaults for our reporting.

That is why you can use the **Change Defaults** button in the Data Browser toolbar, change the area of interest, re-run the query and get a different result. Most Stock Queries (but not all) use the built-in 'Default' queries to obtain their results for Sites and Taxa.

Why is this name enclosed in square brackets? Simply because it contains a punctuation character ('\'). Any name that has punctuation (and that includes spaces) must be in square brackets.

The item names are exactly those that have been assigned to the item by the database designer. Sometimes these are not very human-friendly readable names, and that is why some of the items are followed by an **AS** expression, for example

```
[Sites\Default].[10kSquare] AS [10km Square]
```

The name following the **AS** is an alias, which will be assigned to that item of data when it is displayed. You can check these aliases out by switching between the SQL editing window and the Results display, where you will see them appearing as the column headings. You may question whether some of them in this query are much of an improvement!

In the middle of the data item list is the following rather formidable bit of code.

```
iif([Sites\Default].ViceCounty>200, 'H' & [Sites\Default].ViceCounty-200,
[Sites\Default].ViceCounty) AS [Vice County]
```

This is an example of doing some work on the raw data from the database before passing it on for use. The **iif** word is the name of a built-in function that performs a test on the data it is given, and yields one of two results depending on the outcome of the test. To understand what's going on here, you need to know that MapMate internally stores all the UK vice-counties with their usual vice-county numbers, but stores the Irish vice-counties not with the usual 'H' prefix but by adding 200 to their vice-county number. This means that vice-county references can all be treated consistently as numeric codes by the program. But of course it's better to show them to users in the format which will be familiar to them.

```
iif([Sites\Default].ViceCounty>200, 'H' & [Sites\Default].ViceCounty-200,
[Sites\Default].ViceCounty) AS [Vice County]
```

The **iif** requires three pieces of information: the first is the test to be performed. Simply, this tests "Is the **ViceCounty** code more than 200?".

```
iif([Sites\Default].ViceCounty>200, 'H' & [Sites\Default].ViceCounty-200,
[Sites\Default].ViceCounty) AS [Vice County]
```

The second is what to return in the case that the **ViceCounty** code is more than 200. It stitches together a piece of text made up of an 'H' followed by the code with 200 subtracted. For instance, '215' in the database becomes 'H15' on the displayed result.

```
iif([Sites\Default].ViceCounty>200, 'H' & [Sites\Default].ViceCounty-200,
[Sites\Default].ViceCounty) AS [Vice County]
```

And the third is what to return in the case that **ViceCounty** code isn't more than 200: that is, just the simple value found in the database.

Specifying How the Data Is To Be Selected and Extracted

```
FROM [Sites\Default] INNER JOIN Habitats ON [Sites\Default].[*Habitat] =
Habitats.[_guk]
```

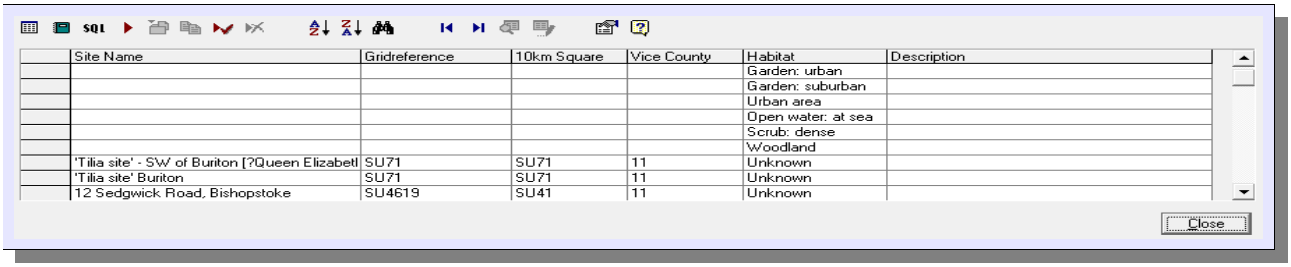
We know from the list of data items that we need data from two sources in the database: the Sites (with our current Defaults applied) and the Habitats list. But in order to know which Habitat entry to display for a given Site, we have to say how they are related, and this is what the second clause specifies. There is a ***Habitat** field in every Site record, which is a pointer to a record in the Habitats list. If you need to refresh your memory on how database tables are interrelated through pointers, turn back to Chapter 2. (By convention, whenever MapMate uses a pointer to cross-reference one data table to another, it puts an asterisk ('*') in front of its name. That isn't something required by the database rules, but it's a useful convention enabling one to pick out the pointer items when one is looking

through data item names.) So we are looking for the Habitat record where our cross-reference from the Site matches its unique identifier (the Global Unique Key, or GUK for short). Notice that in the database, GUKs are always given the name **_guk**.

What if not every Site has been tied to a Habitat definition? (That won't happen in any MapMate data created in the last few years, but it can occasionally crop up in legacy data.) In that case, such Sites won't get listed by this query. The reason for that is the two words **INNER JOIN**. These stipulate that the individual records for Sites will only be listed if there **is** a reference to a Habitat designation in the Site record. And it works symmetrically: if there are Habitats that have never been associated with a Site, they won't get listed either.

If I change this expression to **LEFT JOIN**, I will then get a listing of all the records on the "left hand side" of the **JOIN** (i.e. the Sites) even if they don't have a pointer to a Habitat. As it happens, there aren't any of those in the Training dataset.

If I change it to **RIGHT JOIN**, I will still get a listing of all the Sites with a Habitat reference (and only those) – but now I will also get listings for Habitats that don't have an association with any Site. There are in fact quite a few of those.



Site Name	Gridreference	10km Square	Vice County	Habitat	Description
				Garden: urban	
				Garden: suburban	
				Urban area	
				Open water: at sea	
				Scrub: dense	
				Woodland	
'Tilia site' - SW of Buriton [?Queen Elizabeth	SU71	SU71	11	Unknown	
'Tilia site' Buriton	SU71	SU71	11	Unknown	
12 Sedgwick Road, Bishopstoke	SU4619	SU41	11	Unknown	

Close

As a general rule, I will tell you never to edit any of the Stock Queries without first making a new copy of it to work on, as described in the next main section. It's too easy to make a mistake in the SQL code and render it unusable. But in this case the alteration is so simple that I recommend you try making these one-word changes to the query, saving it and re-running it to see the results. Don't forget to put it back to an **INNER JOIN** afterwards! If you are nervous about this, see the next main section on making a copy of a query.

Specifying What Order the Results Will Arrive In

```
ORDER BY [Sites\Default].Name;
```

The final clause in this query specifies the order in which the results are to appear. This simple example says that they should be listed by site name. Blanks come first, and most punctuation characters and all decimal digits come before letters, which explains the sequence you can see in the last illustration. With more complex SQL, it's possible to specify a hierarchy of items to sort by (only relevant, of course, if the first ones in order can include duplicate values) and also whether the sorting is in normal or reverse order.

The semicolon, by the way, is an indicator that this is the end of the query statement.

Summing Up

This is a relatively simple query, and it certainly hasn't covered all the different types of qualifying clause that are available, or even all the complexities of the ones we've looked at.

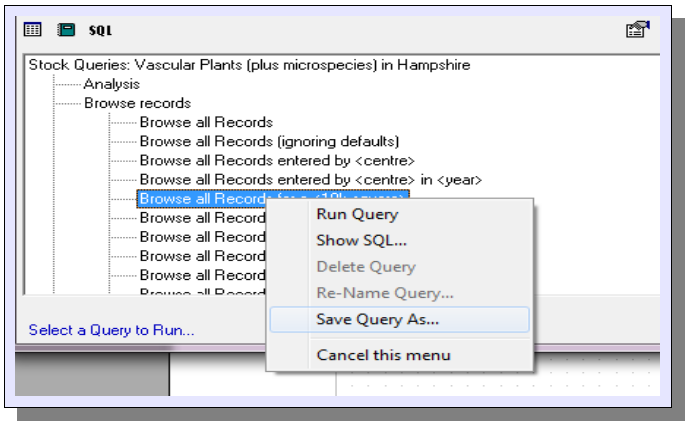
Hopefully it will have given you a flavour of SQL and made it a little more comprehensible. In the following two sections we shall look at another couple of Stock Queries, and these will bring in more features. But if you decide you do want to go more deeply into queries after that, Appendix D will suggest some sources of information to explore.

Using an Existing Query as a Template for A New One

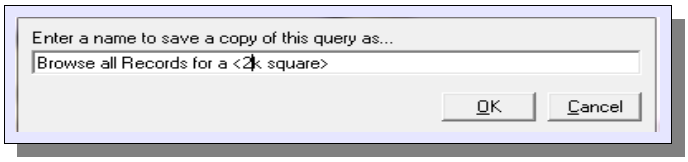
In this section I am going to show you how to take an existing Stock Query, make a copy of it, and edit the copy to do a related useful job that is not covered by any of the standard offerings.

The query we're going to take as our model is actually a pretty complex one (**Browse all Records for a <10k square>**). But by seeing how queries are broken down into clauses that each specify one aspect of the job, you will be able to ignore a lot of this complexity by picking out the relevant clauses where you need to make limited changes.

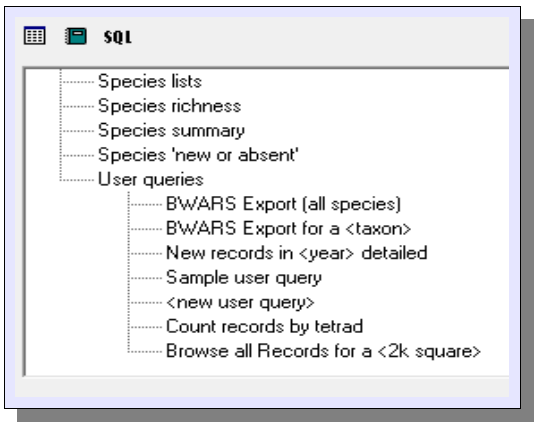
Making a Copy of the Query



Find the query in the list, and right-click on its caption. The option to choose is **Save Query As...** This will save a copy of the Stock Query as a User Query.



When prompted for the name of the new query, change it as shown.



To find your new query, you will need to look under the **User queries** category. Whenever you create a new one, it appears at the bottom of this list. The next time you open MapMate, you will find the list has been alphabetically sorted, so it's worth adopting a consistent naming convention that brings related or similar queries together in the list.

Now you need to click on the **SQL** button to make amendments to the query.

Analysing and Modifying the Query

```
PARAMETERS [Enter a 10km square - eg ST51] Text;
SELECT Records.[_guk], [Taxa\Default].Code AS Code,
[Taxa\Default].Taxon, [Taxa\Default].Vernacular,
[Sites\Default].Name AS Site, [Sites\Default].OSGridRef AS
Gridref, IIf([Sites\Default].ViceCounty>200,'H' &
[Sites\Default].ViceCounty-200,[Sites\Default].ViceCounty) AS
[Vice County], Records.Quantity & IIf([*Sex]<>'u',' ' &
[*Sex],'') AS Quantity, IIf([Records].DateTo-
[Records].Date<27,Format([Records].Date,"dd mmm
yyyy"),IIf([Records].DateTo-
[Records].Date>32,Format([Records].Date,"yyyy"),Format([Records]
.Date,"mmm yyyy"))) AS [Date], Recorders.Name AS Recorder,
Methods.Method, TaxonStage.Stage, RecordStatus.Status,
Records.Comment
FROM (((((Records INNER JOIN [Taxa\Default] ON Records.[*Taxon]
= [Taxa\Default].[_guk]) INNER JOIN [Sites\Default] ON Records.
[*Site] = [Sites\Default].[_guk]) INNER JOIN Methods ON Records.
[*Method] = Methods.[_guk]) INNER JOIN Recorders ON Records.
[*Recorder] = Recorders.[_guk]) INNER JOIN TaxonStage ON
Records.[*Stage] = TaxonStage.[_guk]) INNER JOIN RecordStatus ON
Records.[*Status] = RecordStatus.[_guk]
WHERE ((([Sites].[10kSquare]) Like [Enter a 10km square - eg
ST51]));
```

This is actually made up of two programming statements. The first is one you haven't met yet.

```
PARAMETERS [Enter a 10km square - eg ST51] Text;
```

The **PARAMETERS** keyword introduces a list of data values that your query users will have to provide each time they run the query. They will be used to select the data of interest. If you make use of this statement it needs to appear before the actual query statement where those data values will be used.

```
PARAMETERS [Enter a 10km square - eg ST51] Text;
```

The next element of the statement is a name for the parameter, by which it can be referred to in the main query statement. You may think this is a very verbose name, but there's a good reason for that: it is also the text that MapMate will display as a prompt to the user when the query is run. So, use a name that is meaningful and helpful.

```
PARAMETERS [Enter a 10km square - eg ST51] Text;
```

The final element here is the type of data that the parameter represents. There are data types not only for text but for numbers in various representations, money, date and time, and 'Yes/No' choices. A complete specification can be found among the references in Appendix D. Appendix A will tell you what data type MapMate uses for every element of its database.

If you need more than one parameter to run the query, you can specify a list of name / data type pairs separated by a comma (',') between each pair.

This is where we need to make our first change. Since our new query is going to provide data for a 2km square rather than a 10km square, the wording needs to reflect that so as not to confuse the user. Change the line to read:

```
PARAMETERS [Enter a 2km square - eg SU50A] Text;
```

The next clause brings in the main statement of the query, and as before, it is the **SELECT** clause that specifies the data items required.

```
SELECT Records.[_guk], [Taxa\Default].Code AS Code, [Taxa\Default].Taxon,
[Taxa\Default].Vernacular, [Sites\Default].Name AS Site,
[Sites\Default].OSGridRef AS Gridref,
IIf([Sites\Default].ViceCounty>200,'H' & [Sites\Default].ViceCounty-200,
[Sites\Default].ViceCounty) AS [Vice County], Records.Quantity &
IIf([*Sex]<>'u',' ' & [*Sex],'') AS Quantity, IIf([Records].DateTo-
[Records].Date<27,Format([Records].Date,"dd mmm
yyyy"),IIf([Records].DateTo-
[Records].Date>32,Format([Records].Date,"yyyy"),Format([Records].Date,"mmm
yyyy"))) AS [Date], Recorders.Name AS Recorder, Methods.Method,
TaxonStage.Stage, RecordStatus.Status, Records.Comment
```

It's even more complex than the previous query we examined, and there are two reasons for that: it brings in more data from more tables; and there are several more functions for formatting the raw data coming from the database – particularly for dealing with the date.

Fortunately, we don't need to change any of this, because we want our query to return data to the Browser in exactly the same format as the original one. If you like an intellectual puzzle you are welcome to tease this out. Most of us, when we want something similar for date formatting, simply copy and paste it out of an existing query.

The next clause doesn't need to be changed either. Because the query gets data out of many tables, all the individual relationships between tables have to be specified. This actually follows a common pattern: link one table to another, then link the result to the next, and so on. The round brackets help to specify the order in which this is done; but in fact, in many cases this is arbitrary.

```
FROM (((((Records INNER JOIN [Taxa\Default] ON Records.[*Taxon] =
[Taxa\Default].[_guk]) INNER JOIN [Sites\Default] ON Records.[*Site] =
[Sites\Default].[_guk]) INNER JOIN Methods ON Records.[*Method] = Methods.
[_guk]) INNER JOIN Recorders ON Records.[*Recorder] = Recorders.[_guk])
INNER JOIN TaxonStage ON Records.[*Stage] = TaxonStage.[_guk]) INNER JOIN
RecordStatus ON Records.[*Status] = RecordStatus.[_guk]
```

So we are left with the short last clause, and this is the only other bit we have to change!

```
WHERE ((([Sites].[10kSquare]) Like [Enter a 10km square - eg ST51]));
```

Last time we had an **ORDER BY** clause that specified how the results would be sorted. That's lacking this time; the purpose of the **WHERE** clause that replaces it is to specify that we require only a limited subset of the data, and to state the rules for selecting that sub-set.

The clause as it stands states that the **10kSquare** value recorded for a site should match what the user entered. The **Like** keyword, used rather than an **=**, allows for matches with user entries that include wildcards. You can try running this on the original query when we've finished our edits: for instance, try entering **SU4***.

Since we're dealing with 2km squares, we need to change this to read:

```
WHERE ((([Sites].[2kSquare]) Like [Enter a 2km square - eg SU50A]));
```

Make sure that what you type into the long name **exactly** matches what you put as a name in the **PARAMETERS** statement.

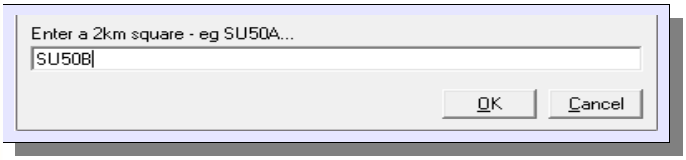
This works because whenever you enter a grid reference for a Site into MapMate, it does a calculation and works out what the 10 km and 2 km squares are for that reference (as long as the reference is at least as precise as 10 kilometres and 2 kilometres respectively, of course). It then stores this derived information as part of the Site record. This makes for much faster retrieval of the data whenever we want Sites selected or ordered by one of these criteria. It also makes it easier to write queries using these criteria.

You may be wondering if this also works with 1 km squares. Unfortunately, MapMate doesn't go that far in deriving data. It's possible, but it entails writing a more complex query that runs more slowly on large sets of data. I've included one that does this in Appendix B.

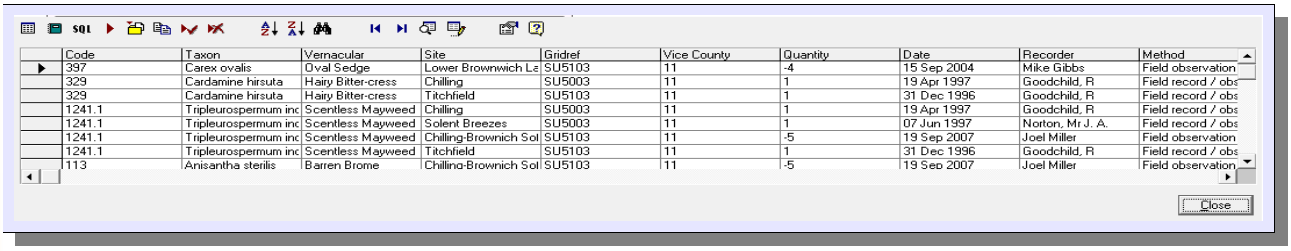
You may have noticed an inconsistency here which was inherited from the supplied Stock Query. The rest of the query uses **[Sites\Default]** as the source of the Sites data (the Sites filtered down to our chosen geographic Defaults settings). Here it refers to the (unfiltered) **Sites** table directly. This actually works, because using the **[Sites\Default]** built-in query implicitly links to the Sites table, and the necessary filtering still happens. But it's an anomaly that you might want to tidy up.

Saving and Running the Modified Query

We've now made all the changes that are needed. Click on the **Save Query** button above the editing window, and confirm the save process with **OK**. Hopefully this will go smoothly, but if you get an error message you will need to go back and examine your code carefully, including punctuation and capitalisation.

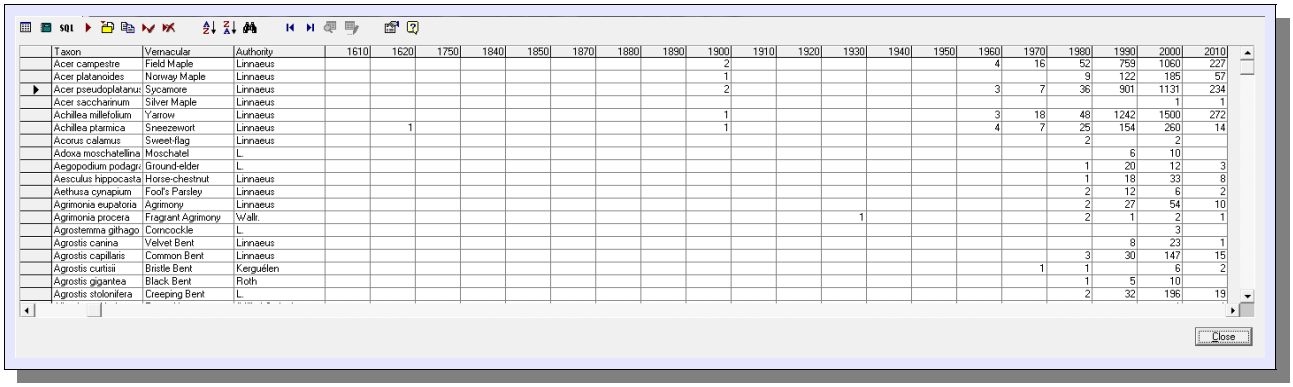


You can then **Re Run Query** and you should see this prompt. Don't actually enter **SU50A** (it's in the sea). Try **SU50B** instead. The Data Browser should give you something like this.



Analysing a Crosstab Query

To deepen our knowledge of SQL, for the next part of this chapter we're going to look at the SQL statements for a crosstab query. You may remember that we looked at **Summary by species by decade,** one of the **Species summary** reports we considered when working through the range of Stock Queries. It produced a result like this, which shows results summarised by two sets of data categories.



Taxon	Vernacular	Authority	1610	1620	1750	1840	1850	1870	1880	1890	1900	1910	1920	1930	1940	1950	1960	1970	1980	1990	2000	2010
Acer campestre	Field Maple	Linnaeus									2						4	16	52	759	1060	227
Acer platanoides	Norway Maple	Linnaeus									1								9	122	185	57
Acer pseudoplatanus	Sycamore	Linnaeus									2						3	7	36	901	1131	234
Acer saccharinum	Silver Maple	Linnaeus																			1	1
Achillea millefolium	Yarrow	Linnaeus									1						3	18	48	1242	1500	272
Achillea ptarmica	Sneezewort	Linnaeus		1							1						4	7	25	154	260	14
Acorus calamus	Sweet-flag	Linnaeus																	2		2	
Adoxa moschatellina	Moschatel	L.																		6	10	
Aegopodium podagr	Ground-elder	L.																	1	20	12	3
Aesculus hippocasta	Horse-chestnut	Linnaeus																	1	18	33	8
Aethusa cynapium	Fool's Parsley	Linnaeus																	2	12	6	2
Agrimonia eupatoria	Agrimony	Linnaeus																	2	27	54	10
Agrimonia procera	Fragrant Agrimony	Wallr.												1					2	1	2	1
Agrostemma githago	Corncockle	L.																			3	
Agrostis canina	Velvet Bent	Linnaeus																		8	23	1
Agrostis capillaris	Common Bent	Linnaeus																	3	30	147	15
Agrostis curtisii	Bristle Bent	Kerguélen																1	1		6	2
Agrostis gigantea	Black Bent	Roth																	1	5	10	
Agrostis stolonifera	Creeping Bent	L.																	2	32	196	19

Close

Navigate back to this query, and click on the **SQL** button to examine its code.

```
TRANSFORM Count(Records.Quantity) AS CountOfQuantity
SELECT [Taxa\Default].Code AS Code, [Taxa\Default].Taxon,
[Taxa\Default].Vernacular, [Taxa\Default].Authority
FROM (Records INNER JOIN [Taxa\Default] ON Records.[*Taxon] =
[Taxa\Default].[_guk]) INNER JOIN [Sites\Default] ON Records.
[*Site] = [Sites\Default].[_guk]
GROUP BY [Taxa\Default].Taxon, [Taxa\Default].[**Parent],
[Taxa\Default].Code, [Taxa\Default].Vernacular,
[Taxa\Default].Authority
PIVOT Int(Year(Date)/10)*10;
```

You'll see that there are five clauses to this statement, and three of them are of types we haven't met before.

`TRANSFORM Count(Records.Quantity) AS CountOfQuantity`

The **`TRANSFORM`** clause comes before the **`SELECT`**, and specifies what summarising statistic will go into the main body of the Data Browser grid. You can use counts, totals, averages, maxima and minima here. Because we're using a count of Records, it doesn't actually matter what data item in the Record we specify to be counted – it's just necessary to give one of the items that occurs in a Record definition. The query could just as well have stated **`Count(Records.[_guk])`** or **`Count(Records.Comment)`** here. We're not actually doing anything with the **`Quantity`** value found in the record. If we'd used any of the other statistics, like totals or averages, it would have been a different matter.

The next clause is familiar by now. Here the query needs to list anything else that we want to see displayed in each row of the Data Browser. As usual, the SELECT also determines the order in which these items are displayed.

```
SELECT [Taxa\Default].Code AS Code, [Taxa\Default].Taxon,
[Taxa\Default].Vernacular, [Taxa\Default].Authority
```

And the clause after that also follows a familiar pattern, linking the three data tables together.

```
FROM (Records INNER JOIN [Taxa\Default] ON Records.[*Taxon] =
[Taxa\Default].[_guk]) INNER JOIN [Sites\Default] ON Records.[*Site] =
[Sites\Default].[_guk]
```

Now we come to something new: as we are aggregating results, we need to specify the criteria by which we group them to yield sub-totals. You **must** include anything you said you want to display in the `SELECT` clause. For instance, `[Taxa\Default].Vernacular` (the vernacular name) will generally change at the same time as the scientific name (`[Taxa\Default].Taxon`). So it doesn't contribute anything new to the subtotalling logic. But we must still include it, because we have asked for it to be returned for display.

```
GROUP BY [Taxa\Default].Taxon, [Taxa\Default].[**Parent],
[Taxa\Default].Code, [Taxa\Default].Vernacular, [Taxa\Default].Authority
```

On the other hand we can include other things here which aren't part of the displayed data. `[Taxa\Default].[**Parent]` is a case in point. This is an example of a pointer that references not into another data table, but back into the same table – to be precise, to provide a link to the species' family. The MapMate developers have followed the convention that self-referring pointers have two asterisks in front of their name. Since there are four different (and independent) protocols for the scientific naming of organisms, it's possible for the same scientific name to be legitimate for different organisms at the lower end of the hierarchy in separate groups. But for most of the time, the inclusion of this item doesn't bring much to the feast.

Finally, we need to specify the other dimension of the table - the category and its subdivisions by which the totals will be split across the columns. That is the function of the `PIVOT` clause.

```
PIVOT Int(Year(Date)/10)*10;
```

In order to achieve the split by decades, this query manipulates the date of the Record. In calculations like this, just as in mathematical formulae, the expressions in the innermost parentheses are evaluated first, then the next outer evaluation is applied, and so on. Expressions at the same level of bracketing get evaluated left to right.

`Date`	That retrieves the specific calendar date for the Record from the database: for example, 12 Feb 2012.
`Year(Date)`	That gets us just the year part of the date: for example, 2012
`(Year(Date)/10)`	That gets us a value from the date which has a fractional part: for instance 201.2
`Int(Year(Date)/10)`	That discards the fractional part and gets us back to an integer, with no rounding up: in this example, 201
`Int(Year(Date)/10)*10`	That gets us back to a year value, namely the first year of the decade: in this example, 2010.

Creating a New Crosstab Query from Scratch

Finally, we'll increase our repertoire of queries with a worked example of another crosstab report.

Go back to the **User queries** part of the query list and select **<new user query>**. Enter **Annual count of records by hectad** as the new query name. That describes pretty well what it is going to do for us.

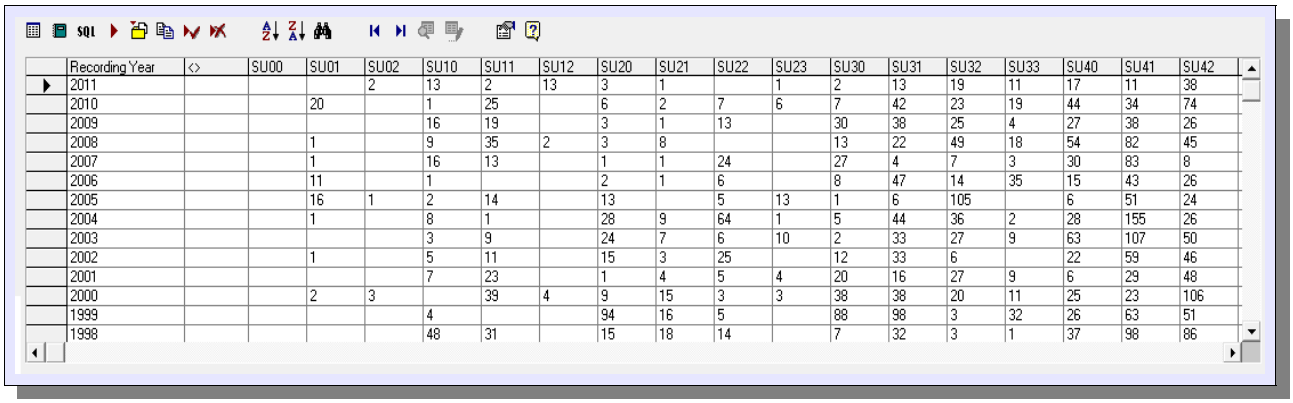
As before, delete the query code that MapMate helpfully provides "for free" for all new queries, and replace it with this.

```
TRANSFORM Count(Records.[_guk])
SELECT Year(Records.[Date]) AS [Recording Year]
FROM Records INNER JOIN [Sites\Default] ON Records.[*Site] =
[Sites\Default].[_guk]
GROUP BY Year(Records.[Date])
ORDER BY Year(Records.[Date]) DESC
PIVOT [Sites\Default].[10kSquare];
```

After the last example, this should be fairly easy to interpret. Note that in the **SELECT** clause we alias the year part of the date to a name that is clear for the reader of the display.

The only novelty here is the reintroduction of the **ORDER BY** clause (which must come between a **GROUP BY** clause or anything that takes precedence over that in its absence, and the **PIVOT** clause). The **DESC** is shorthand for 'descending order', and ensures that we see the most recent recording years at the top of the Data Browser display.

If you save this query and run it, you should see something like this. (I've narrowed the year columns down.)



Recording Year	<>	SU00	SU01	SU02	SU10	SU11	SU12	SU20	SU21	SU22	SU23	SU30	SU31	SU32	SU33	SU40	SU41	SU42
2011				2	13	2	13	3	1		1	2	13	19	11	17	11	38
2010			20		1	25		6	2	7	6	7	42	23	19	44	34	74
2009					16	19		3	1	13		30	38	25	4	27	38	26
2008			1		9	35	2	3	8			13	22	49	18	54	82	45
2007			1		16	13		1	1	24		27	4	7	3	30	83	8
2006			11		1			2	1	6		8	47	14	35	15	43	26
2005			16	1	2	14		13		5	13	1	6	105		6	51	24
2004			1		8	1		28	9	64	1	5	44	36	2	28	155	26
2003					3	9		24	7	6	10	2	33	27	9	63	107	50
2002			1		5	11		15	3	25		12	33	6		22	59	46
2001					7	23		1	4	5	4	20	16	27	9	6	29	48
2000			2	3		39	4	9	15	3	3	38	38	20	11	25	23	106
1999					4			94	16	5		88	98	3	32	26	63	51
1998					48	31		15	18	14		7	32	3	1	37	98	86

There are a couple of things to observe about this display. The first is that there is a column for a blank 10km square. If you scroll down through the years to 1931, you will see that there is one record associated with that, denoting that the site associated with the record was not even given a 10-kilometre precision.

As you scrolled down to 1931, you may also have noticed that if there were no records in a particular year, the year doesn't get a row entry at all in the grid. That keeps the listing compact, but it can have its drawbacks – particularly if you are trying to compare results from two sets of data. Getting the empty years listed as well with the techniques available

in this sort of query is not at all straightforward, and entails writing a hierarchy of queries that are then used inside other queries (like the [Sites\Default] query). Rather than getting bogged down with this in MapMate, you may prefer to move your raw data out into another software package and use the rather more user-friendly facilities it provides for summarising data. We come back to this topic in Chapter 9.

In Conclusion

The latter part of this chapter is probably the most technically challenging material you will have to deal with in the whole of this book. You may have decided that producing your own queries is not for you. I hope, however, that you have at least read through the Intermediate sections once, and got an overview of how queries are put together.

In the first place, this should enable you to deploy other people's published queries and get them working without tears. Secondly, reporting is not the only place where you will encounter SQL (Structured Query Language). For instance, when making Atlases (Chapter 7), there are some refinements that you can only achieve by making minor changes to the SQL code that MapMate uses to put dots on your map. Having the confidence and understanding to make these changes puts a lot more flexibility in your hands.

You can also customise your user Defaults, and control more precisely what data you exchange with other people, by designing and deploying Filters – a topic we shall come to in Chapter 5. Filters also require you to use SQL.

Perhaps you have decided that SQL is a wonderful tool, and you are ready to tackle the challenges of designing your own queries for all manner of new data extractions. The first thing to appreciate is that we have barely scratched the surface of the SQL language in this chapter. You will need to follow up the leads in Appendix D to give you more information.

Next, while many things are feasible with SQL, it is designed for a specific purpose and it has its limitations. Some things it simply won't do for you – at least in the dialect of SQL that *Access* supports. And some things can only be done with lengthy and convoluted coding. Before ruining your home life by trying to achieve the near-impossible in SQL, it's worth looking at Chapter 9 and considering whether it wouldn't be better to extract the data in a more 'raw' form and then use a more appropriate tool to do the rest of the job.

Finally, don't forget the *MM-Users* Internet forum mentioned earlier in the chapter!