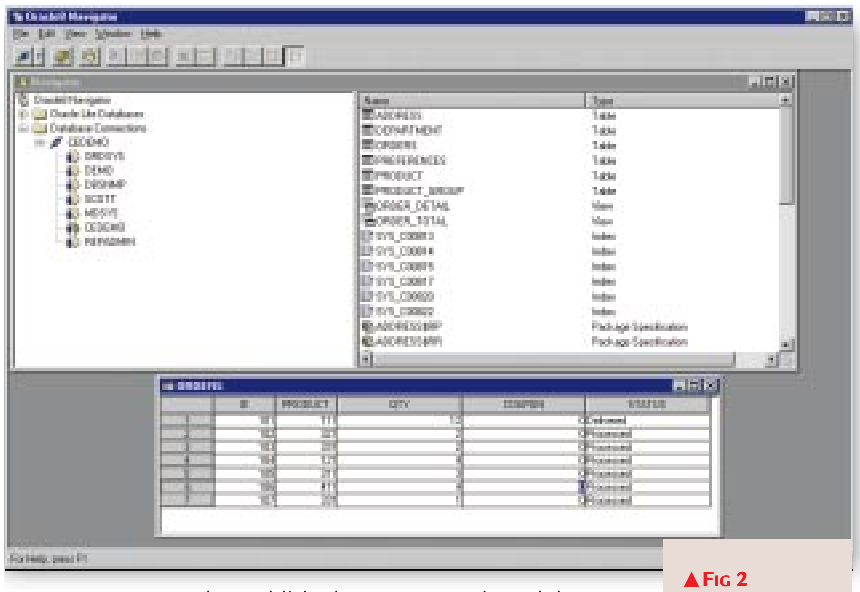
Working model

Mark Whitehorn presents a simpler way to calculate working days.



▲**FIG 2 THE 'CORPORATE' DATABASE**

Last month, I published a way of calculating the number of working days (where a working day is [Mon, Tue, Wed, Thu, Fri]). However, I also wrote that the problem sounded hauntingly familiar and that I had come across other algorithms which were likely to be faster — albeit more complex.

To give you some idea of how long ago this was, the code I dredged up from my archives is in dBASE for DOS — ask your grandparents for details!

The good news is that dBASE is so English-like that the code reads reasonably clearly and is relatively easy to translate into other languages. On our cover-mounted CD, in a text file called CODE.TXT, is an algorithm that I developed. This is documented and reasonably rapid in execution.

By all means examine it for interest's sake but do not bother to implement it because reader Charli Langford recently came up with a more elegant solution that is even faster: 'A less complex solution is to divide the TotalDays by seven to get the number of weeks and a remainder number of days over. The remainder can then be matched against FirstDate to see how many workdays are in the leftover part-week, and the extra day due to include both Firstdate and SecondDate can be added in here, too.'

The reference she makes to adding in the extra day refers to the fact that the original specification required both the start and the end dates to be included in the calculation, which doesn't happen when you subtract one date from another.

Charli produced a block of pseudo code which, in Fig 1, I have translated into dBASE code:

I did not test this rigorously, but the logic seems fine and it worked with the 50-or-so dates that I threw at it. Like most good algorithms, as soon as I looked at it I thought 'Why didn't I do it that way?'

The logic seems fine, and it worked

It is left as an exercise for the reader — just to keep you on your toes — to define how it works.

However, a couple of hints may help you here.

1 First of all, expressions such as this one:

```
WorkDays = WorkDays – 1
```

would be more readably, but less elegantly, expressed as;

```
WorkDays = WorkDays – 2
WorkDays = WorkDays + 1
```

In other words, two operations are going on here. One is the removal of two days because the 'remainder' contains two weekend days. Then a day is added for the reasons discussed, above.

2 You might get to worrying about what happens...

```
If Remainder + FirstDay equals 7
```

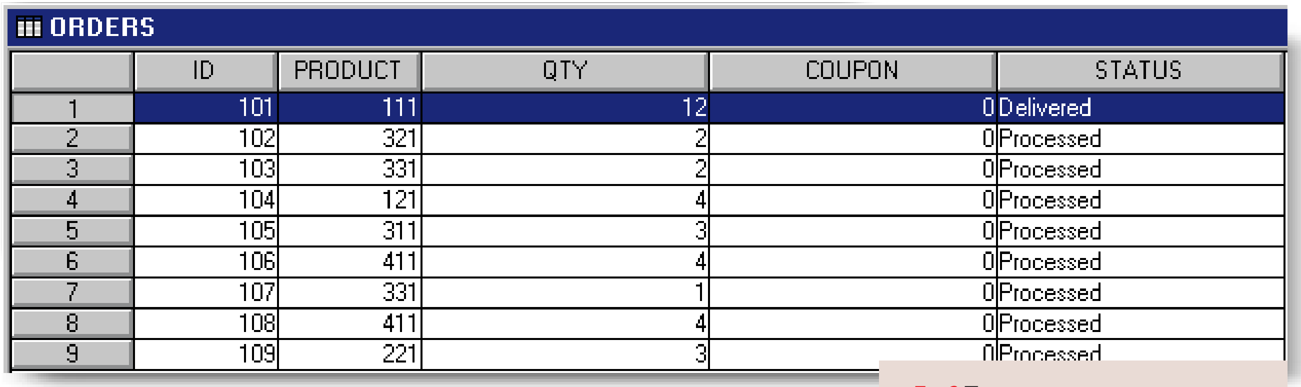
That possibility requires that we first subtract 1 and then add 1 to 'remainder'.

[FIG 1] Charli's code in dBASE

```
Procedure Charli
 If SecondDate < FirstDate
   ?"Twit"
 EndIf
 TotalDays = SecondDate – FirstDate
 FirstDay = DOW(FirstDate)
 Weeks = Int(TotalDays/7)
 Remainder  = TotalDays – (Weeks * 7)
 WorkDays = (Weeks * 5) + Remainder

 If remainder = 6
   WorkDays = WorkDays – 1
 Else
  If FirstDay > 1
    If Remainder + FirstDay > 7
       WorkDays = WorkDays – 1
    Else
       If Remainder + FirstDay < 7
        WorkDays = Workdays + 1
       EndIf
    EndIf
  EndIf
EndIf

? 'Total days  = ',(SecondDate+1-
FirstDate)
? 'Weekdays   = ',WorkDays
RETURN
```


ORDERS

	ID	PRODUCT	QTY	COUPON	STATUS
1	101	111	12	0	Delivered
2	102	321	2	0	Processed
3	103	331	2	0	Processed
4	104	121	4	0	Processed
5	105	311	3	0	Processed
6	106	411	4	0	Processed
7	107	331	1	0	Processed
8	108	411	4	0	Processed
9	109	221	3	0	Processed

▲Fig 3 The same database after records have been 'replicated' back from a WinCE client. The user of the WinCE machine (in this case, me) has added two records which now appear in the corporate database

We have to do nothing if this happens to be true, so there is no code there to do it!

3 It is worth knowing that dBASE assumes that Sunday is the first day of the week.

If any reader implements this in a modern RDBMS send it in and, if it's good, I will pop it onto the cover disk.

Mobile databases

The explosive growth in mobile computing — both for laptops and PDAs — has led to a corresponding increase in interest in mobile databases. The allure, at least for commercial organisations, is overwhelming.

Imagine that you have a mobile workforce. Now imagine that you can give each person a copy of the corporate database on a mobile device. They can then go out, take orders, refer to corporate information, show statistical information to the client and use the database in whatever way is appropriate to their particular job.

Once they return to their hotel room at night, all they need do is connect into the main database using a modem, upload the changes they've made, download the changes that others have made during the day, and everything is hunky-dory. Ah! You've spotted that there may be just one or two problems in working like this. It's a shame, really, because mobile access to corporate data has so much promise.

The good news is that we now have the technology. The even better news is that some of this technology can only be used if a human brain is there to guide it. Astute database administrators can therefore benefit from an acquaintance with the potential problems.

The challenges associated with mobile computing fall into three categories:

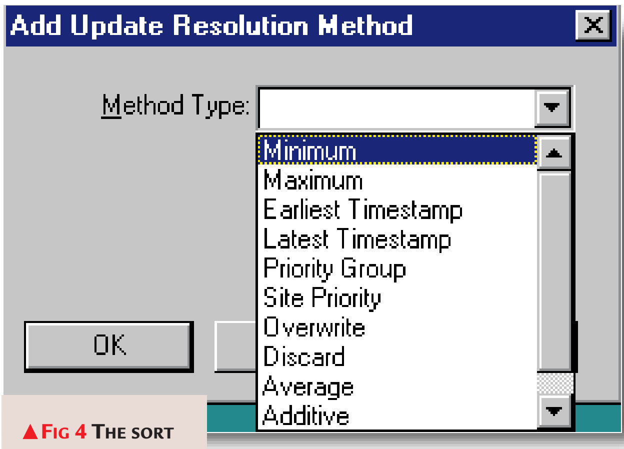
1 RDBMSes are big beasts requiring serious computing power, memory and disk space. This is not an accurate description of the average PDA — although laptops can now provide this.

2 The same can be said of databases themselves, as they are often big.

3 A single word, 'conflict', embraces a huge class of problems that arise when two or more people are allowed edit access to data that is not held centrally. In the simplest example, you are at base, I am on the road. I edit an existing record, increasing a customer's credit rating, say. Meanwhile, back at the ranch, you *lower* that same customer's credit rating. What happens when I upload my changes?

What are the simple answers to these problems? Companies such as Sybase, Oracle and IBM are producing RDBMS engines with tiny footprints. By tiny, I mean well under 1Mb. In fact, IBM has announced Everywhere which hasn't even got a footprint: it has a fingerprint of an unbelievable 50K.

You subset the corporate database and only give the user the data they need. You may still end up with a quantity that requires a laptop rather than a PDA but at least it can be done. The answer to conflict is, of course, a mechanism called Conflict Resolution. Clearly there is more to know about all of these so we will look at each, in more detail, in future columns.



▲Fig 4 The sort of conflict resolution options that Oracle offers

Oracle Lite on WinCE

Just to prove that this is not all just theory, I have been playing with Oracle Lite, a version of Oracle that runs on WinCe. Oh, and it also runs on Windows 95/98/NT and 2000. Oh, and on EPOC, PalmOS, QNX... Oracle is certainly serious about this market place.

Fig 2 shows an Oracle database running on a server. You can see that the table has seven records. If you flip to *Hands On PDAs* (p230), you will find screenshots showing a sample application running on a WinCE machine.

After the data from the PDA has been 'replicated' back to the server, lo and behold, there are nine records [Fig 3]. And as for a taster of how Oracle manages conflict replication, see Fig 4.

PCW CONTACTS

Mark Whitehorn welcomes your feedback on the Databases column. Contact him via the PCW editorial office (address, p10) or email database@pcw.co.uk