# Room to manoeuvre

**Mark Whitehorn reports on building a bigger text file, and looks at smaller database engines.**

You need big text files? I can get you big text files. How big do you need? I have recently been building a data warehouse, a process that involves importing huge text files which are 'report' outputs from legacy systems.

Reports are typically structured with several lines of header, then multiple lines of detail, then another header, more detail, and so on. For example, the report from a student information system might consist of: header containing the name of the course; names of 23 students doing that course; header containing the name of the next course; names of 65 students doing that course. And so on.

**When this information** is imported into the warehouse, it is often necessary to extract information from the header and apply it to each of the following lines so that we end up with, for example:

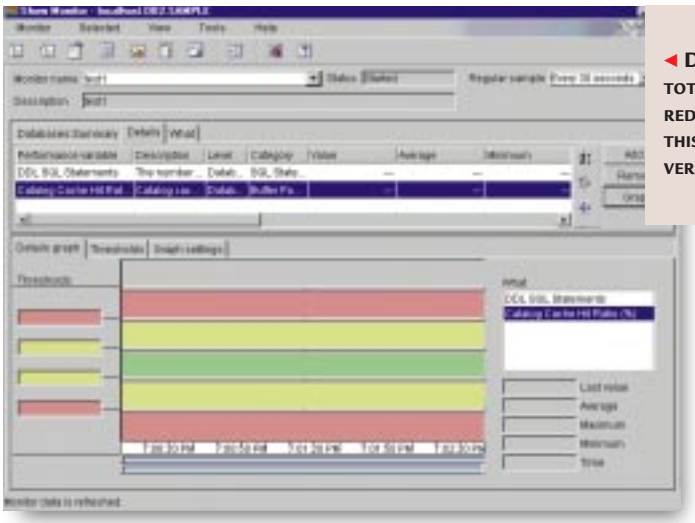
| Course | Student Name |
|---|---|
| Biology | Fred Smith |
| Biology | Sally Jones |
| Biology | Helen James |
| History | Fred Smith |
| History | Alan Weston |
| History | Bert Samson |

(Yes, all of the information is de-normalised, but that's not a sin in a data warehouse.)

I needed to rough out the code for importing the data before the reports were made available to me, so I faced the prospect of hand-writing a sample text file, a prospect that didn't appeal. However, I suddenly realised that the DOS command

```
Dir/s > Penguin.txt
```

generates a file of precisely this format which is perfect for testing. It happens to be fixed length, which is what I needed, but if you require a sample file that is comma separated, simply generate the file and then use Word, or something similar, to perform the necessary search and replace.



**DB2 WILL BE TOTALLY REDESIGNED FROM THIS GROWN-UP VERSION**

## Mobile databases

Last month we started to look at mobile access to corporate data. This process generates three delectable challenges: first, the size of the RDBMS engine; second, the size of the data; and third, how to resolve conflicts during data replication.

Let's start with the size of the engine: RDBMS engines are typically huge, complex, lumbering pieces of software — so how can one possibly fit on a portable device? The trite answer is that laptops are becoming unbelievably powerful, so use one of those as the mobile device.

But this is to miss the point: laptops are very expensive, and we have the technology to run on much smaller, and much cheaper, devices.

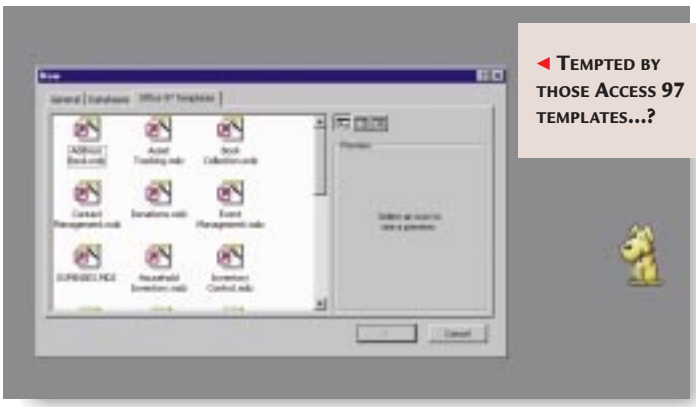
***Unlike Satellite, DB2e is a redesign which sits on the file structure of the target device***

Ask yourself why a client-server RDBMS is so big: the answer lies mainly in two areas — multi-user access to the data, and security. To ensure that you and I don't edit exactly the same data at the same time, the RDBMS has complex locking mechanisms: on a single-user device, all of this can be shed. Secondly, RDBMS designers are paranoid about data security. Every transaction is logged, the log files are kept on separate disks, and the data is periodically backed up. Why? Because this database is the datastore for the entire company, and a lunched database could literally mean the end of the company. Losing all of the data on a mobile device is sad, but unlikely to bring the entire company to its knees. So, we can trim that fat off the RDBMS engine as well.

**I discussed Oracle Lite** in the *Hands On PDAs* column last month, but IBM has been doing even more drastic liposuction on DB2 [*pictured, above*]. There are two versions coming: DB2 Satellite is a cut-down version of the standard DB2 Universal database and runs in under 1Mb on Windows NT, 95 and 98, with the delivery of the server-side tools being on NT and AIX.

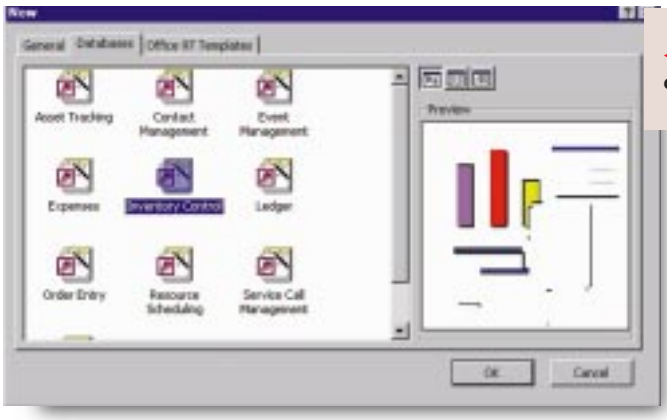
But there is the unbelievably tiny DB2 Everywhere (coyly abbreviated to DB2e) which is designed for Windows CE, Epoc-32 and PalmOS. This runs in 50K — tiny, or what? Unlike Satellite, DB2e is a total redesign which sits on top of the file structure of the target device. All IBM has added is a thin layer that understands basic SQL (Select, Insert, Update and Delete). For the synchronisation back to the server, IBM Mobile Connect is used.

So, small RDBMS engines are possible after all. More next month...



◀ TEMPTED BY THOSE ACCESS 97 TEMPLATES...?


▲ WELL, DON'T BE...




◀ GREAT GRAPHICS, GUYS...

## Record numbering

In the June issue I talked about a way to number records on a report. Many people emailed in about this, but the following letter from Paul is typical:

*'With regard to your comments on record numbering in a report, might I suggest that you can in fact take a slight shortcut from your version and not have to use Select 1 AS Test. Simply add an unbound text field to your report and set its control source to =1, then set the running sum as normal.'*

PWALKER@INNOGISTIC.CO.UK

Paul is, of course, correct. Although I haven't done any timings, I suspect that his solution is faster too, if we use a big data set.

However, one of the reasons I chose the solution I did is that it can be more versatile. If you put the 'calculation' in a query, then all of the reports based on that query can make use of the numbering system.

This isn't meant to imply that 'my' solution is in some way better, just that the different solutions have pros and cons. But that's just one reason why I love databases; they're such intricate tools.

## Bugs in Access 2000

I have also received several emails about Access 2000. You probably all know that I love Access dearly and 2000 is still a great product. So it pains me to report that, in my opinion, Access 2000 is the most buggy version that Microsoft has ever shipped. We aren't talking data-corrupting bugs here; we're talking about careless annoyances. But the trouble with annoyances is that they are annoying — very annoying.

***We aren't talking data-corrupting bugs here; we're talking careless annoyances***

For example, the graphics for the database wizards are mangled on both the machines on which I have tested Access. Then there are the Access 97 templates [*see screenshots, above*]. These appear listed as database wizards for Access 2000, which is good, but if you try to use one, you're told that you can't, which is bad.

Even more galling is the fact that the UK specific postcode and phone number input masks that Microsoft supplies still don't work. This has been the case since version 2.0.1 (and I'm sure other users have reported the fault with each new version). I know this annoys new users especially, because they use the wizards and then can't enter data into the databases they create. And, of course, being new to the game, they blame themselves, rather than Access.

So, I fired off another bug report: 'Since Access 2.0, UK customers have highlighted that the input mask for postcodes does not work for UK codes. Microsoft apparently chooses not to listen to them and resolve this issue for UK clients.'

The following is part of the reply from the US: 'Our knowledge base on the web provides articles that outline the great flexibility for formatting postal codes. In case you haven't seen it, I have attached the text of the KB article which explains how to implement it. <<Manipulating Zip Codes.doc>>.'

**As you can guess** from the title, this document is all about American postal codes. Sigh... Such are the pitfalls of responding to the particular problem of a customer with a formulated answer: I ask specifically about UK postcodes, and Microsoft replies that zip codes can be handled efficiently. It could be described as customer lip service.

Bearing in mind that a major part of my complaint was that Microsoft doesn't listen, I'll let you know how I get on.

## PCW CONTACTS

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