FlashFiler 2.0

A new version of TurboPower's client/server database engine

Reviewed by Chris Frizelle

Back in June 1998 I reviewed a bunch of database engines for Delphi in *Developers Review*. The aim was to examine the alternatives to the Borland Database Engine (BDE) that came in the box with Delphi. One of those engines was FlashFiler 1.0, which at the time was only just out. TurboPower has recently brought out a new version of FlashFiler. The database development goalposts have definitely changed since 1998, and it's time to see how the new version of FlashFiler fares.

The BDE is definitely on its way out (stop cheering at the back there!) and more and more folk are turning to client/server style databases from the older shared file approach. And there's more available for free: MySQL is a client/server database which has been around for some time now and of course there's the Open Source version of InterBase 6.0.

FlashFiler 2.0 comes with a nice thick printed manual: noticeably thicker than for version 1.0. The installer does everything for you, including setting up FlashFiler within Delphi and installing the help into Delphi. Great: please take note, other companies!

The most important thing about FlashFiler is that it is a client/server database, not a shared file system. This means that it is inherently scaleable (although I couldn't find any information on maximum users, etc) and includes goodies such as transaction management. The manual claims tables can now be up to Terabytes in size! (limited by the operating system capabilities and disk size). Unlike some client/server systems, however, it is fairly easy to deploy: just copy the server application (1.2Mb) and some configuration files, get your application to do some alias setup, and you are away. This makes it well suited to use within vertical market software ('embedded' databases). The server can live on the same machine as the application(s), on a LAN server, or even across the internet somewhere.

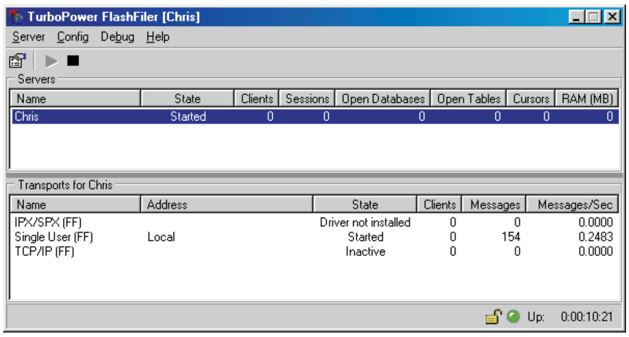
New in this version is SQL support for queries (it supports the `SELECT` keyword), which makes life an awful lot easier. Also added are multi-threading in the server, better BLOB storage and sundry other improvements.

Leaping In

The first thing you must remember to do before getting to grips with FlashFiler is to start the server engine and set up at least one alias to point to a database file (such as the sample `mythicdb` database included with the product).

Unfortunately, the manual rather drops you in at the deep end: it is very much reference-oriented. A simple *'Create your first FlashFiler application'* chapter early in the manual would be a great help, along with more *'How to do xyz'* material.

Because FlashFiler uses its own proprietary data format, you will need to convert any existing databases. TurboPower provides various utilities to help with this. The main conversion route provided is the *BDE Transfer To FlashFiler* utility, which will open any BDE-compatible database and convert the tables within it to FlashFiler format. You can either use BDE aliases, or manually enter a directory and table name. Both BLOB fields and memos are supported. There does not seem to be any way of specifying how field types are converted, other than restructuring the data once it's in FlashFiler format.



Name	State	Clients	Sessions	Open Databases	Open Tables	Cursors	RAM (MB)
Chris	Started	0	0	0	0	0	0

Name	Address	State	Clients	Messages	Messages/Sec
IPX/SPX (FF)		Driver not installed	0	0	0.0000
Single User (FF)	Local	Started	0	154	0.2483
TCP/IP (FF)		Inactive	0	0	0.0000

➤ *Figure 1: FlashFiler Server.*

The *BDE Export To ASCII* utility will export any table which is supported by the BDE to fixed-length ASCII files, which can then be imported into FlashFiler format. This utility was provided in beta form in the version of FlashFiler I used. It does the job fine enough, but of course it won't work with BLOB or memo fields.

Another essential utility is the *FlashFiler Explorer*, which allows you to create, view and edit tables, as well as viewing and amending table structure and indexes. Talking of indexes, they can be unique or not, case-sensitive or not, and ascending or descending.

Developing applications with FlashFiler is fairly straightforward once you have got to grips with the way things work. A simple local server application (that is, the server is on the same machine as the application) is very easy. Using remote servers (on another machine, eg a LAN server) had me scratching my head: the help is very pithy, telling you only what each piece of the puzzle does, rather than how to assemble the whole thing; the manual is better but not idiot-level enough for ordinary mortals like me! It's a shame, as the software itself is clearly well thought out and implemented.

One of the facilities that FlashFiler has is to compile the server engine into your application's executable, so that you only have to ship your own .EXE file. The idea is, presumably, greater simplicity. Understandably, this limits your application to single-user status: as FlashFiler is a client/server engine not a file-sharing engine, you need a separate server for multi-user capabilities. I had expected that creating a single .EXE app would be very easy and that it might feature in some introductory part of the manual. In fact, the information on how to do this is on page 570 and to find it you have to know the magic word 'SingleEXE'. You actually have to start off by developing your application to use a separate server engine, then make some configuration changes to FlashFiler and to the database file, recompile it all, then copy some system tables to your application directory. Yes, well: not exactly simple, is it? My conclusion is that FlashFiler really is a client/server system and that is how it is best used. If you want a single .EXE solution, it's probably best to look elsewhere.

Conclusions

FlashFiler undoubtedly contains lots of clever stuff: the folks at TurboPower are certainly no slouches in terms of ideas or coding. The new SQL `SELECT` support is very useful, although other competing products offer fuller SQL support (ie not just limited to queries). The documentation has been beefed up, but could do with having more task-oriented material. Overall, users of FlashFiler still need to be prepared for something of a learning curve.

I think the big question is whether FlashFiler offers enough when compared to MySQL or Open Source InterBase, both of which are free and are also proper client/server databases. I'm not really qualified to give an authoritative answer (nor does the space I have here allow for an in-depth evaluation of the product), but some things in FlashFiler's favour are the ability to customise virtually anything you want and to do it in Delphi: if you want to tweak the server code, you can do that. As a general purpose database engine, I think FlashFiler will struggle against the established names (both commercial and free). As an engine which you can seamlessly embed into your own vertical market products, it's got a lot more going for it. If you need a client/server solution (that is, a shared file approach is not powerful enough), then FlashFiler should definitely be on your list of candidate products for evaluation.

For more information on FlashFiler visit www.turbopower.com or talk to your local dealer. Direct from TurboPower, FlashFiler costs $499; a typical UK price is around £300 plus VAT. This is a one-off cost: there are no royalties or client licences to buy. Supported versions are Delphi 3.0 or later and C++Builder 3.0 or later: we have no news yet on a Kylix version for Linux. The current version is 2.02, which fixes some bugs and adds minor enhancements over version 2.00.

Chris Frizelle (chrisf@itecuk.com) is Editor of *The Delphi Magazine.*

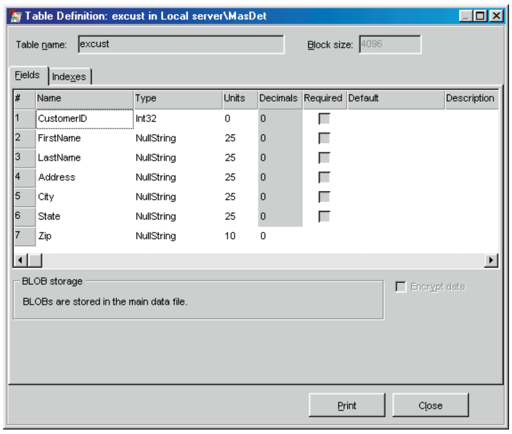


Table Definition: excust in Local server\MasDet

Table name: excust

Block size: 4096

#	Name	Type	Units	Decimals	Required	Default	Description
1	CustomerID	Int32	0	0	☐		
2	FirstName	NullString	25	0	☐		
3	LastName	NullString	25	0	☐		
4	Address	NullString	25	0	☐		
5	City	NullString	25	0	☐		
6	State	NullString	25	0	☐		
7	Zip	NullString	10	0			

BLOB storage

BLOBs are stored in the main data file.

☐ Encrypt data

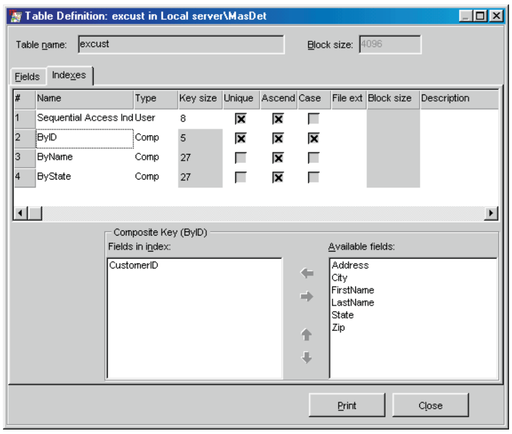


Table Definition: excust in Local server\MasDet

Table name: excust

Block size: 4096

#	Name	Type	Key size	Unique	Ascend	Case	File ext	Block size	Description
1	Sequential Access Ind	User	8	☒	☒	☐			
2	ByID	Comp	5	☒	☒	☒			
3	ByName	Comp	27	☐	☒	☐			
4	ByState	Comp	27	☐	☒	☐			

Composite Key (ByID)

Fields in index:
CustomerID

Available fields:
Address
City
FirstName
LastName
State
Zip

➤ *Figures 2 and 3: Table and index restructuring in FlashFiler Explorer.*