Moving from DBF to SQL Server
Pros and Cons

Overview

This document discusses the issues related to migrating a VFP DBF application to use SQL Server. A VFP DBF application uses native Visual FoxPro (VFP) database files (DBFs) for data storage. An alternative storage mechanism for data storage would be Microsoft SQL Server (SQLS). Both techniques have their advantages and drawbacks. This document hopes to clearly define those in a way that is easily digestible by the non-technical.

Note that VFP DBFs and MS SQL Server are not two competing types of the same thing; it is a bit of apples-and-oranges. DBFs are accessed from a *file server*; SQLS data is accessed from a *database server*. Visual FoxPro the client-development system can easily access data from both, either individually or in combination.

Visual FoxPro DBF

A FoxPro DBF application stores all of its data various (often hundreds) different VFP-format data files, plus related files for indexes and long-text. Each data file holds a particular table, such as clients, order header, order detail, and so forth. Some data files can be quite small, with only a few records. Others may have millions of records.

Benefits of the DBF

Speed

Without a doubt, Visual FoxPro using its native DBF data file format, using a local source (same machine or reasonably-fast network), is the fastest data system available on the PC platform, bar none. It is faster than any database server, including SQL Server, running on comparable hardware.

Simplicity

When VFP uses its native DBF files there is no intervening layer of software; it reads, writes, and manages the data files using its own built-in technologies – it is seamless and lightweight. This allows for the easy copying or moving of the entire application database simply by copying the a VFP DBF application directory to any other location, including a laptop computer. Multiple copies of the system are easy to create for testing and backup purposes. Further, a small portion of the data (say, just the client file) can be copied and given to someone.

Programming

Programs written in VFP that use the native DBF files can be structured or designed in several different ways, or a mixture of methods according to the needs or preferences of the programmer. A myriad of traditional (20+ year-old) techniques and ANSI standard SQL can be used and still achieve good performance.

Disadvantages of the DBF

Safety

While it occurs much less frequently with the more recent versions of VFP, individual data files can become corrupt due to network or workstation interruptions. When such corruptions occur, often a third-party utility must be used to correct the problem. Further, since typically all program users have full rights to the directory containing the data, an individual could (accidentally or maliciously) delete or overwrite one or more data files, or modify a data file from outside the VFP application, potentially introducing errors or corruption. There is no inherent protection of the DBF from outside application access.

Security

There is no provision for user-specific rights to DBF files other than that provided by the VFP application. Anyone with sufficient access to the application's data directory can copy any file or open it using an external tool (such as Access or Excel).

Data Integrity

While later versions of VFP have added the ability to enforce some forms of data integrity at the datafile level, these enhancements do not have the performance or robustness of those found in SQLS. In short, they are ways to tack-on these abilities to an existing data format. Therefore, a VFP DBF application system does not have any data integrity enforcement other than what the programmer remembers to code within the application. If someone accesses the data from outside the application then all data integrity checks that *might* be found in a VFP DBF application are bypassed.

Remote and Slow Access

Since DBFs reside on a file server, performance of the application is based partly on the speed of the file server's disk system and network, with the processing power of the local computer workstation being a significant additional factor.

For example: when a user requests a list of orders for the current month, the user's workstation must sort through all rows (assume a total of 500,000 orders in the file for this example). This will require, at best, index data for all 500,000 records to be dragged across the network from the file server to the user's workstation if there is an existing index on the requested date column. At worst, if the query involves non-indexed columns, a great deal of that many megabytes of data will have to come across the network and be processed by the user's workstation, where all of the processing is performed (none is performed on the file server).

This architecture works well on fast networks with moderately-powerful workstations. If the data is on a separate network or separated by a slow infrastructure, data access and apparent performance degrade substantially.

There is no extraordinary provision for remote access to DBF files, just as there isn't for Excel XLS files. Techniques such as RDC or terminal services can be used.

Technical Limitations

DBF files are limited by Microsoft to 2 gigabytes in size for each DBF.

Microsoft SQL Server

The attributes here apply to most database server systems, including Oracle and Sybase. These are all *database servers*. Like Exchange Server, SQL Server is a process that runs on a server (preferably dedicated to the task, but not necessarily) and provides for interfaces for getting things in and out without direct access to the underlying storage files.

Benefits of SQL Server

Scalability

Unlike the DBF approach, all data for a given SQLS database can be stored in a single data file. A datafile can hold just a few records or many billions of records, depending on disk space. SQLS inherently uses multiple processors and, if ever needed, can be clustered onto multiple server machines (depending on which level of SQLS is used; there are several, from free to thousands of dollars). One SQL Server can hold several databases, so different applications can share the same server system.

Data Integrity

A database server really shines at ensuring (if so programmed) that data follows logical and business rules, regardless of how and by whom the data is accessed. For example, if the server is told never to allow deletes of a client if there are orders existing for that client, then it is an impossibility (absent database administrator override) for that to happen, even outside of a VFP DBF application. Data integrity gets enforced at the lowest level possible – at the database itself.

Triggers can also be programmed on the database server so that if a certain data event happens (say, a new client is added) certain additional actions occur. Server-level agents can monitor server activity and errors and notify supervisors if certain conditions occur. The system supports transactions, which allow the programmer to wrap multiple individual data events (add a client, an order header, and order detail lines, for example) in a transaction, so that if any part fails the whole is rolled-back, preventing a partial-data situation.

Data Security

In a proper SQLS installation no user has rights to the SQLS data directory. All access to the data is through SQLS. Users would not be able to copy or delete data files, even those who otherwise have access to the data through a VFP DBF application. In addition, user-based, role-based and application-based security can be implemented within the database.

Data Safety

SQLS data files can be backed up at any time, even while fully operational and in use. Further, all data operations are fully logged, allowing (at least in theory) a full to-the-second recovery of all data in the case of a system crash.

Data Access

As a database server, SQLS provides a number of ways for users to access its data via ODBC. Users could access via a FoxPro application; via MS Access or Excel (if allowed), or remotely from outside the organization (say, from a Web site) if so configured. For example, a user on the road or at home, running the VFP application with an Internet connection, could run the application using live data, just as if they were in the office (again, given sufficient permissions and configuration). Any client program that can use ODBC can be allowed to access the database if so desired. The server could be located in-house; at a local data center; or across the country – or across the world.

Further, SQLS allows several different mechanisms to access the data. For example, if a user only needs daily statistics or other compilations of data, they can call a process that isolates them from the individual direct data and only provides them the summary.

SQLS provides for optional replication, which allows data changes on one server to automatically migrate to another server.

Finally, since the vast majority of the filtering is done at the server rather than the workstation, the network speed and client workstation power are less-important.

Disadvantages of SQL Server

Complexity

The data is abstracted behind a "black box" and new layers (ODBC & SQL Server) are introduced between the client VFP application and the data. Database servers require more maintenance and tuning than a file server. The FoxPro application system would exist as two pieces rather than one: the client program (your .EXE) and the database server.

Most importantly, from a FoxPro application standpoint, is that large blocks of the application code would *probably* have to be rewritten. Since so much of a VFP application was possibly written using "old school" dBase-type data access programming, those pieces would have to be recoded to use the newer SQL-compatible VFP language and with a database server mindset. The application could support a mixed-scheme approach, where some data could be in DBFs and other data in SQLS.

Cost

SQL Server 2005 Express is a free version available from Microsoft that may meet the needs of the current application. Several other versions add management features, which can boost the price to several thousand dollars.

A database server has much greater performance needs than a file server; thus, the server hardware itself will likely be more expensive than a file server.

Summary

The single-greatest hindrance to moving to SQL Server as VFP application data storage system is the time and expense of rewriting parts of a VFP DBF application to support SQLS. This work could be done incrementally, one table/data file at a time.

However, the potential benefits of moving a VFP DBF application to SQL Server are clear, particularly in regards to much more flexible data access and much stronger data integrity.

Robert Bradley
Broad Lea LLC