

FileMaker Pro and Microsoft SQL Server Integration

By Bob Cusick
President, ClickWare, Inc.

FileMaker, Inc.'s Focus on Workgroups

FileMaker, Inc. is a leading database software vendor focused on the needs of workgroups of all kinds, ranging from departments within the enterprise to small businesses. FileMaker® software enables workgroups to solve many of their information-management challenges while connecting to corporate data sources, and fitting into IT requirements.

If you (or your customers) are working in a networked environment, chances are good that FileMaker isn't the only database system in use. Many midsize and large companies use Oracle, Microsoft SQL Server, Sybase, DB2 or any of a myriad of other SQL-based databases to keep track of everything from sales data to ERP (Enterprise Resource Planning) data.

FileMaker Pro 5.5 makes it easy to execute dynamic SQL statements to import data from corporate data sources. And the new Execute SQL ScriptMaker® script step automates the execution of any SQL query, including Data Manipulation Language (DML - i.e. INSERT, UPDATE, DELETE, etc.) and Data Definition Language (DDL - i.e. CREATE TABLE, CREATE INDEX, ALTER TABLE, etc.) commands, against any ODBC source. Solutions created in FileMaker Pro can easily be connected to Microsoft SQL, Oracle or other ODBC-compliant data sources.

Why link FileMaker with an external data source? The major benefit of FileMaker is its unique ease-of-use and with FileMaker Pro version 5.5, it's even easier for workgroups to build and deploy information solutions with links to back-end corporate databases. These links, controlled appropriately by internal IT staff, allow users to run their own reports, "upload" data into IS-approved data sources for use by other departments and applications, and share corporate data without having to re-enter the data into FileMaker. The end result is that users can take more of a "self serve" approach to data manipulation and storage while better conforming to IT standards and at the same time freeing valuable IT resources.

Looking At Microsoft SQL Server

Like Oracle, DB2, Informix, Sybase and other relational SQL database applications, Microsoft SQL Server 2000 is meant to do one thing and one thing only - store and retrieve data - quickly. We've chosen to look at SQL Server 2000 for the purposes of this white paper because it is an industry-standard database supported by highly skilled programmers and analysts.

SQL Server 2000 provides rich XML and Internet standard support that gives you the ability to store and retrieve data in XML format easily with built-in stored procedures. You can also use HTTP to send queries to the database, perform full-text search on documents stored in database, and run queries over the Web with natural language rather than writing SQL statements.

SQL Server 2000 is highly scalable and reliable and can meet the needs of demanding ecommerce and enterprise applications. SQL Server 2000 takes advantage of symmetrical multiprocessor (SMP) systems and can use up to 32 processors and 64 GB of RAM. SQL Server 2000 can distribute the database and data load across servers, and features failover clustering, log shipping, and new backup strategies. Because of this scalability, SQL Server 2000 is suitable for Very Large Databases (VLDB) in the millions or hundreds of millions of rows of data. Beyond the hardware scalability, each SQL Server 2000 database can have 2,147,483,647 objects (tables, columns, stored procedures, users, etc.) and each instance of SQL Server can have up to 32,767 databases, and there can be up to 16 instances of SQL Server can be running per server.

SQL-Based Database Limitations

If the speed and scalability of SQL-based databases are its biggest strengths - the lack of built-in tools to create a user interface to manipulate the data is one of its biggest weaknesses. It's completely up to the developer to create a separate user interface. In the case of SQL Server 2000 this is usually accomplished using Active Server Pages (ASP) for a browser-based interface, or Visual Basic for a client-based, "rich" interface (i.e. custom forms, master-child forms, reports, etc.) or a completely custom application written in Java, C# (C-Sharp), or C++. This limitation means that (usually) a completely different team of highly trained and experienced people is required to create the interface. It also means that there is a whole new development cycle required just for the interface - that inevitably impacts the data structures and composition of the backend database.

With the exception of a browser-based interface and a Java application - any application written to access the data is, by nature, platform-specific. For example, a Visual Basic or C# (C sharp) application won't run on Linux or Mac OS. A C++ application must be re-compiled - and in most cases - partially re-written to take advantage of each operating systems underlying services and toolbox calls. In multi-platform environments, this can add substantial development costs and increased development time to the overall project.

Another limitation of SQL-based databases is the lack of support for native calculated columns. This means it is up to the developer to create and manage totals and calculations as well as relationships (or joins) between tables via SQL statements. This can (and often does) become extremely complex. So a developer needs a robust knowledge of SQL programming to do anything beyond the most simple queries. While there are a number of commercial tools that provide more user-friendly "graphical" tools for reporting - some sort of programming is required to perform the insert/update/delete functions.

SQL Server 2000 includes a feature called "stored procedures." These stored procedures are programming code that is saved within the database and allow the developer to perform complex actions, calculations, multi-table joins, etc. However, in order to take advantage of these powerful stored procedures - the developer must learn the syntax of the programming language Transact-SQL (T-SQL).

Although T-SQL is a powerful procedural programming language, the learning curve can be quite steep.

To confuse matters even more - each SQL database vendor uses its proprietary programming language to extend the functionality of their database. For example, the stored procedure language in Oracle is called PL/SQL (Procedural Language SQL); in Sybase it's also T-SQL (SQL Server 2000 is based on Sybase) but has different syntax than SQL Server 2000; and in DB2 by IBM you can write stored procedures in C++, Java, JSQL, or procedural SQL. When using FileMaker, there is no need to program procedurally - and no procedural language to learn - because ScriptMaker provides a drag-and-drop "macro" interface for scripting complex actions between tables, adding and deleting data, performing validation, etc.

FileMaker is the Right Interface Tool

The biggest benefit of using FileMaker as the user interface tool is the fact that it is ready to use "right out of the box." Any user who can write a spreadsheet formula should have no problem in creating a useful solution with little or no instruction.

One of the most common user needs is reporting on data stored within a corporate data store. FileMaker makes it easy for novices to import data from the corporate data store, create custom reports with multiple levels of sorting and summarization, and easily manipulate the "look and feel" of the report without the assistance of over-burdened and scarce IT resources.

In addition to reporting, using FileMaker Pro 5.5, a user with moderate SQL knowledge can easily insert and update data into the corporate data store via ODBC. From an IT perspective, the corporate data store DBAs still retain complete control over the data because the user has to access the data via ODBC. Thus all the permissions for the login can be completely controlled based on user type, application type, department, etc.

Here are some at-a-glance reasons you should consider using FileMaker as your interface tool for viewing, reporting and updating a corporate data store:

- FileMaker is easy to use - even for non-technical content experts.
- The built-in ScriptMaker allows users to easily capture business rules via point-and-click interface rather than traditional code and debug cycle of a procedural language.
- FileMaker Pro 5.5 runs on Mac OS X, Mac OS 9, Windows XP/2000/NT/98/95/Me, and can be used to share data via a browser.
- FileMaker Server runs on Windows 2000/NT/98 (FileMaker is Windows 2000 logo certified), Red Hat Linux, Mac OS X and Mac OS Classic.
- FileMaker provides a rapid, WYSIWYG development environment that is great for creating functional prototypes as well as full-blown solutions.
- FileMaker provides password protection for structure as well as content.
- FileMaker can function in "disconnected" environments (i.e. not connected to central server).

- FileMaker features a single physical data file for data and structure (data and all forms for each table are encapsulated for easy portability).
- FileMaker provides a one-click Web translation of layouts via the built-in Web Companion.
- FileMaker has automatic record locking and concurrency controls.
- FileMaker makes it easy to modify table structure "on the fly."
- FileMaker is scalable up to 250 users (using FileMaker Server) without having to use SQL.

Dealing With SQL In FileMaker

FileMaker can communicate with external data stores in two different ways - either via the built-in ODBC import and Execute SQL features of FileMaker Pro 5.5 - or via a FileMaker SQL Plug-In (sold separately by a 3rd party vendor).

FileMaker version 4.1 and higher can import data (static SQL query) using ODBC while FileMaker Pro 5.5 can import data via ODBC with a dynamically generated SQL query or a static SQL query. In addition, FileMaker Pro 5.5 allows the execution of any valid SQL command via the "Execute SQL" ScriptMaker step. In order to use either of these features the client machine requires an ODBC driver that is client platform-specific. While native ODBC drivers are widely available for most SQL-based databases on the Windows platform, the situation is very different on the Mac OS platform. There are very few native ODBC drivers available on Mac OS.

However, FileMaker users of the Mac OS platform have options. Independent developers have created solutions that will allow Macintosh users to take advantage of FileMaker Pro 5.5 ODBC connectivity. Merant International, Ltd. has created Macintosh native ODBC drivers that support Microsoft SQL Server (7.0 and 2000), as well as Oracle, dBase, text files and Fox Pro. The drivers are available exclusively through FileMaker Solutions Alliance Partner Metro Technologies, Inc. (<http://www.metrotechnologies.com>).

The second option for connecting FileMaker to an external data source is to use the SQL Plug-In created by Professional Data Management and Rumora Automatisering and distributed in North America by ClickWare, Inc. (<http://www.clickware.com>). The SQL Plug-In works on all platforms that FileMaker client runs on (Mac OS Classic, Mac OS X, Windows 95, 98, 98SE, Me, NT4, 2000 Professional, 2000 Server, and XP) and allows connections via ODBC or JDBC. Because the FileMaker Plug-In API was introduced in FileMaker version 4.0 - you can use the SQL Plug-In with FileMaker 4.0, 4.1, 5.0 or 5.5 to programmatically communicate with any external data store via ODBC and/or JDBC. The Plug-In architecture also allows the developer to programmatically set the connection parameters, issue SQL statements (including direct calls to stored procedures) and receive back data that can then be parsed either column-by-column, row-by-row, or like FileMaker Pro 5.5, batch imported. Because the SQL Plug-In can also connect via any standard Type 3 or Type 4 JDBC driver (see <http://java.sun.com> for a list of 3rd party JDBC drivers and supported SQL databases), Macintosh compatibility is not issue.

Whether you choose FileMaker Pro 5.5 native capabilities or the SQL Plug-In, some technical skill is required to allow FileMaker to function as a user interface:

- Developer needs to understand some SQL
- Developer needs to understand FileMaker calculations
- Developer must be able to create FileMaker buttons and scripts to insert, update and delete data using SQL statements
- Developer must develop their own "data collision" checking and record locking

Ideal Uses For FileMaker as the User Interface

While it's possible to use FileMaker as the full user interface to import, insert, update, and delete data, there are three areas where using FileMaker as the user interface affords the greatest return on investment:

- Reporting
 - Use FileMaker powerful ODBC import (or the SQL Plug-In) functionality to import data for end-user customizable reports
 - Take advantage of powerful backend SQL-based servers to summarize large record sets and import into FileMaker only the summarized data
 - Import into FileMaker data that is "joined" several levels deep
 - Leverage existing reporting stored procedures written for other applications, while allowing end user to modify look and feel - thus freeing up valuable IS resources.
- End-User Interface
 - Lookup corporate data to ensure data integrity when developing FileMaker applications (i.e. staff directory, etc.) or to ensure data integrity within legacy FileMaker applications.
 - Make summarized data available to other legacy FileMaker applications
 - Easily "push" data from a FileMaker application to SQL databases used in high transaction ecommerce or enterprise applications
- Other
 - Easily "push" and "pull" data primarily stored within FileMaker into other SQL data source(s) for shared use between departments and workgroups or for use by other corporate applications.
 - Flexible solution for integrating or upsizing "legacy" FileMaker applications with other SQL sources - can do it bit by bit rather than all at once.
 - "Push" FileMaker data into an IS-approved SQL data store to conform to IS policies and procedures, backup schedules, and other IS-related issues.

FileMaker vs. SQL Terminology and Capabilities

Because FileMaker is a tool for use by non-programmers, workgroup knowledge workers and content experts (as well as IS staff), there are major differences in terminology used when working with FileMaker and with SQL-based databases. Below is a Quick Reference Guide to some of the key terminology differences between SQL Server 2000 and FileMaker:

- "Fields" in FileMaker = "Columns" in SQL Server
- "Records" in FileMaker = "Rows" in SQL Server
- A "Database" in FileMaker is ONE Table (including all forms, reports, and scripts)
- A "Database" in SQL Server can contain hundreds of Tables
- FileMaker maximum file size (per table) is 2GB
- SQL Server 2000 max file size (per database) is 1,048,515 TB
- FileMaker maximum columns per table is 5,900 (depending on field name length), and the physical field name is limited to 60 characters
- SQL Server maximum columns per table is 1,024, and physical column name is limited to 128 characters
- FileMaker Scripts are limited by disk space (or max file size) and the first 52 are displayable to the end user in the "Scripts" menu
- SQL Server Stored Procedures can be nested up to 32 levels deep and each database can have up to 2,147,483,647 objects - but has no user interface into the data
- A FileMaker "Text" field can contain up to 64K of text per field, per record
- SQL Server has multiple "Text" data types:
 - Char - up to 8,000 (8K) per record - fixed length
 - Varchar - up to 8,000 (8K) per record - variable length
 - Text - up to 2GB PER RECORD

IT staff and frequently asked questions about FileMaker

Developers and IT staff new to FileMaker often ask questions about its architecture. Here are answers to some frequent questions:

- Why does a single FileMaker file equal a single table?

It may not make traditional "sense" to a skilled IS database developer or DBA, but it makes sense for the non-programmer. Each file (table) contains only the data that's needed for the user to get the results they need. In addition, because FileMaker can import from any ODBC data source using a SQL query - the query itself can "flatten" multi-table data or query a pre-existing view to ensure that the user gets only the data they need.

- Why isn't FileMaker SQL-based?

FileMaker strives to deliver the best of both worlds: empowering

non-technical users with its point-and-click interface as an alternative to SQL programming, plus the ability to act as an ODBC data source by adhering to standards. FileMaker Pro 5.5 database act as an ODBC data source (via the built-in Local Data Access Companion [LDAC] or the Remote Data Access Companion [RDAC]). This capability allows IS personnel and other applications to query the data stored in FileMaker via their own SQL-based tool of choice. Note that advanced users may create SQL code within FileMaker if they choose.

- Why is the FileMaker "client" single-threaded, and why doesn't it support multiple processors?

Workgroups sharing databases enjoy greater performance by deploying FileMaker Server 5.5, which does support multiple processors per machine and is multi-threaded, and runs on Windows 2000/NT/98 (FileMaker is Windows 2000 logo certified), Red Hat® Linux, Mac OS X and Mac OS Classic.

- FileMaker is not as fast as SQL when sorting and summarizing large datasets. Will I see slow performance?

This is typically not an issue. When integrating a FileMaker interface with a corporate data store it is much more efficient for the "big iron" backend database to do the column selection, joins, sorting and grouping than bringing down just the "raw" data and relying upon FileMaker to sort and summarize the data.

- Why does FileMaker support only equijoins?

FileMaker is meant to be a tool used by non-programmers, workgroup knowledge workers and content experts. The advanced concepts of inner joins, outer joins, left inner joins, left outer joins, right inner joins, right outer joins, full joins and unions are best left to the highly skilled and experienced IS professionals. However, FileMaker can take advantage of these advanced joins and database-specific SQL when importing from an ODBC source, executing a SQL command against an ODBC source (with or without a data result), and when executing a SQL command to a stored procedure (with or without parameters - with or without a data result).

Conclusion

You can use FileMaker Pro 5.5 built-in ODBC import and Execute SQL ScriptMaker command or a the 3rd party SQL Plug-In to connect via ODBC or JDBC - to create an easy-to-use, flexible solution for bridging the gap between end users and corporate data stores. In addition, you can reduce the burden on valuable IS resources by allowing end users to modify the "look and feel" of reports, retain DBA-controlled access to the data and provide a way to consolidate workgroup data with corporate data in the SQL database of your choice.

About the Author

Bob Cusick is President of ClickWare, Inc., a member of the FileMaker Solutions Alliance, and a Technical Editor of the FileMaker Advisor Magazine. He has been creating custom FileMaker, SQL, and browser-based

solutions since 1990, and is an internationally recognized author, speaker and trainer. <http://www.clickware.com>, bob@clickware.com.

©2001 FileMaker, Inc. All rights reserved. FileMaker is a trademark of FileMaker, Inc., registered in the U.S. and other countries. Red Hat is a registered trademark of Red Hat, Inc., in the U.S. and other countries. All other trademarks are the property of their respective owners. Mention of third party products and companies is for informational purposes only and does not constitute an endorsement nor recommendation. The example companies, organization, products, domain names, e-mail addresses, people, places and events depicted are purely fictitious, and any resemblance to existing persons and companies is purely coincidental. Product specifications and availability subject to change without notice.