Introduction to External SQL Sources
TABLE OF CONTENTS

About This Technical Brief ................................................. 3
An Overview of ESS .......................................................... 3
ESS Design Goals: What It Is ................................................. 4
ESS Design Goals: What It Isn’t .............................................. 4
Supported Data Sources ....................................................... 5
How to Use ESS ............................................................... 5
Step 1: Establish database access and credentials ......................... 6
Step 2: Define a DSN on the host machine ................................ 6
Step 3: Define an External Data Source in FileMaker Pro ................. 6
Step 4: Add one or more tables to the FileMaker Pro Relationships Graph .... 6
Step 5: Create a relationship to the shadow table ......................... 6
Step 6: Use the external table as though it were a FileMaker Pro table ... 7
Establishing Access to an External Data Source ......................... 7
Defining a DSN .................................................................. 8
Defining a DSN in Mac OS X .................................................. 8
Defining a DSN in Windows .................................................... 12
Define an External Data Source in FileMaker Pro ....................... 14
Add Tables to the Relationships Graph .................................... 17
Manage the Field Structure of the New Table ............................. 19
Work With the New Table ..................................................... 20
ESS Tables vs. Native FileMaker Pro Tables ............................... 22
Supported Data Types .......................................................... 22
Concurrency Issues: How Fresh is My Data? ............................... 23
Indexing ........................................................................... 24
Schema Changes .................................................................. 24
ESS vs. Other Forms of SQL Connectivity in FileMaker Pro .......... 24
ESS Best Practices .............................................................. 26
Learn Some SQL ................................................................ 26
Learn About Views ............................................................. 26
Watch Your Indexes ............................................................ 27
Search Carefully ................................................................. 27
Sort Carefully ..................................................................... 27
Additional Resources .......................................................... 27
Terminology ........................................................................ 27
FileMaker TechNet Paper ...................................................... 28
Supported ESS Data Sources ................................................ 28
Supported ODBC Drivers ..................................................... 29
Books ................................................................................. 29
About This Technical Brief

It is the intent of this technical brief to help FileMaker® Pro developers and other information systems professionals better understand the new External SQL Data Sources (ESS) feature set introduced in the FileMaker Pro 9 product line and available in the FileMaker Pro 10 product line. Reading this document will assist developers and information architects in assessing the key benefits of the ESS feature set, and provide insights into the range of uses and integration possibilities for which the feature set was designed.

This paper covers the features and benefits of ESS to a depth that should assist developers and decision-makers in assessing how ESS may help them achieve their systems goals. Members of FileMaker Pro TechNet will have access to an in-depth technical brief, oriented toward developers seeking detailed technical information that explores the ESS feature set in deeper detail.

An Overview of ESS

FileMaker Pro has long had the ability to share data with a variety of external data sources and clients. The ability to import records from an ODBC-based data source, and the ability to run SQL commands on a variety of SQL back ends via the Execute SQL script step, offer developers a significant level of connectivity and interactivity with external, SQL-based data sources. Still, these connectivity features have certain limitations:

- Importing from an ODBC data source is a one-way operation that does not provide a “live” view of SQL-based data.
- Using Execute SQL to create, modify, and delete records in an SQL source requires creating handcrafted SQL statements.
- Both features require configuring ODBC drivers on the desktop of every client.

The ESS features in the FileMaker Pro 10 product line allow data tables from supported SQL sources to be “dropped into” a FileMaker Pro schema. The ESS tables behave very much like native FileMaker Pro tables. ESS overcomes the limitations of existing SQL connectivity methods in FileMaker Pro as follows:

- ESS views are highly dynamic and refreshable.
- Developers do not create SQL queries directly: the ESS feature set generates appropriate commands behind the scenes.
- When used with FileMaker Server, ESS requires only a single DSN per data source, created and maintained on the server.

In addition, ESS also offers the ability to define supplemental fields that extend the schema of an external SQL table. Supplemental fields consist of summary fields and unstored calculations that operate on data from an SQL table, but are defined locally within a FileMaker Pro solution.

The following sections of this paper explore the ESS feature set in detail.
ESS Design Goals: What It Is

In order to make the best use of the ESS feature set, it is important to understand what these features were designed for and how they may best be used.

ESS is designed as a means to better integrate a FileMaker Pro solution with existing SQL data sources, such as corporate data warehouses or SQL-based e-commerce systems. Here are some typical usage scenarios:

• A large company has a master personnel database that is Oracle-based, but has many line-of-business applications developed in FileMaker Pro. Using ESS, any FileMaker Pro application can have direct, up-to-date access to the corporation’s master list of people, departments, and divisions (with the necessary blessing of the corporate IT department, of course).

• A reseller of computer hardware runs a MySQL-based e-commerce web site, but their customer management and shipping is handled by a FileMaker Pro application. Using ESS, a developer can “embed” the MySQL-based order and payment information into a FileMaker Pro solution. A customer’s record in the FileMaker Pro solution can easily show a portal that displays all of their MySQL-based orders, without any importing or synchronization.

• A museum has a Microsoft® SQL Server-based collections-and-exhibits management system, a FileMaker Pro based members-and-donation management system, and an Oracle-based ticketing-and-sales system. Using ESS, the museum can potentially:
  o Embed ticketing and sales information in the FileMaker Pro based members database, to get a better idea of the attendance and sales habits of individual members.
  o Create a FileMaker Pro based dashboard that gives an overall management view of the current status of collections, exhibits, membership, and sales.

ESS is primarily intended to allow FileMaker Pro solutions to share data with existing SQL-based data systems.

ESS Design Goals: What It Isn’t

The emphasis with ESS should be on integration. The ESS feature set is not intended to allow FileMaker Pro to act as a “front end” to SQL data sources. In particular:

• ESS does not allow a FileMaker Pro developer to compose their own SQL queries and pass them to the server: ESS creates all SQL queries behind the scenes. (Some such functionality is possible with ESS, though, by defining views within the SQL data source. FileMaker Pro’s Execute SQL script step, by contrast, does allow a developer to create and send customized SQL queries.).

• ESS does not allow a FileMaker Pro developer to alter or extend the schema of an SQL data source on the server side (though local extensions are possible with the ability to add supplemental fields to an ESS table within FileMaker Pro).
• ESS is not designed as a means to allow a FileMaker Pro solution to scale beyond the limits of a purely FileMaker Pro based solution. The ESS feature set is designed to emphasize the seamless integration of SQL-based tables into a FileMaker Pro solution, rather than to take specific advantage of the high-scalability features of SQL back ends.

It is important to keep these design goals in mind as you plan how to use ESS within your FileMaker Pro solutions.

Supported Data Sources

FileMaker Pro 10 supports the latest certified SQL data sources including:

• Oracle 9i
• Oracle 10g
• Oracle 11g*
• MS SQL Server 2000
• MS SQL Server 2005
• MS SQL Server 2008*
• MySQL 5.0 Community Edition
• MySQL 5.1 Community Edition*

* New to FileMaker Pro 10

Since the ESS feature set is based upon ODBC, FileMaker, Inc. may choose to add support for additional ODBC-capable data sources in future releases of the product line.

The supported data sources work with a selected set of supported ODBC drivers. These are listed in the Appendix at the end of this paper.

For further information on supported drivers and data sources; please consult the FileMaker Pro web site at http://www.filemaker.com/support/technologies/sql.html.

How to Use ESS

This section will walk through an example ESS usage scenario. Subsequent sections will provide additional detail and instruction on each step.

Tara Mayali is a FileMaker Pro developer who maintains a FileMaker Pro based solution for a large corporation. The FileMaker Pro solution needs access to the corporate personnel database. Currently, corporate IT sends Tara an export file every day that she imports into the appropriate tables in her FileMaker Pro solution. In addition to the ten data fields she receives from corporate IT, Tara’s FileMaker Pro file tracks an additional fifteen data fields for each person, data specific to her department. With ESS, Tara decides to try for a more direct integration with the master Oracle database of personnel.
Step 1: Establish database access and credentials

The Oracle DBA (database administrator) is unable to grant Tara direct access to the underlying tables, but is willing to define a database view that will grant Tara read-only access to the ten Oracle database fields she needs to work with. The DBA creates a new view for Tara and gives her a user name and password granting read-only access to that view.

Step 2: Define a DSN on the host machine

The DBA has given Tara everything she needs. Now it is up to Tara to integrate the new database view into her application. Tara’s application is running on FileMaker Server 10. Tara defines a single DSN (data source name) on the FileMaker Server machine. (This is a big improvement for Tara: in her past forays into ODBC, she had to configure a DSN on the desktop of each individual user). The DSN will permit Tara’s application to connect to the Oracle data source.

Step 3: Define an External Data Source in FileMaker Pro

With her DSN defined on the server. Tara opens a copy of FileMaker Pro Advanced and accesses her solution. (Tara does not need to use FileMaker Pro Advanced to work with ESS, but as a developer, she prefers the extended developer feature set). She goes to File > Manage > External Data Sources. She creates a new External Data Source and indicates that it should be ODBC-based. In defining the External Data Source, she points it to the DSN she just defined, and stores the Oracle user name and password within the data source definition. (The user name and password are encrypted within the FileMaker Pro file, so this is a secure choice, and it will save the individual FileMaker Pro users from having to remember and enter the Oracle credentials). That completes the configuration of the external data source.

Step 4: Add one or more tables to the FileMaker Pro Relationships Graph

Tara now goes into the Relationships Graph in her FileMaker Pro solution. She adds a new table occurrence, and chooses to add a table from an external source. She chooses her Oracle-based external data source, and sees a list of the available tables. She chooses the view that the Oracle DBA defined for her (she knows that her credentials won’t allow her to access the other tables).

When she chooses to drop the view into the Relationships Graph, several things happen:

• A new table occurrence is created
• A new layout is created based on the table occurrence
• A new shadow table is added to the list of tables in the solution, so that the Oracle table now appears in the FileMaker Pro table list

Step 5: Create a relationship to the shadow table

Tara now navigates to the Relationships Graph. She has fifteen additional data fields she wants to track for each person. It is not possible to add these files directly to the shadow table, but she can continue to store them in the personnel table she currently uses, and create a relationship from that table to the new SQL-based table.
This will make it possible to have both the SQL-based data and the extra data Tara maintains appear in the same views.

**Step 6: Use the external table as though it were a FileMaker Pro table**

With these steps completed, Tara can now access and work with the Oracle view as though it were a FileMaker Pro table. If she navigates to the new layout, she will see records from the Oracle table. She can add data to the fields in her related personnel table (though if she tries to change any of the Oracle fields, her privileges won’t let her do so). She can drop into Find mode and run a search against the Oracle table, searching on the Oracle fields, her own extra fields, or a combination of the two. She can create relationships between her FileMaker Pro tables and the Oracle view, displaying related data from Oracle on records from other table.

Going forward, Tara can do away with the daily import. Her application will have access to up-to-the-minute personnel data, with Tara's own data related alongside the master data. The integration is transparent to her users, and works within the limits of the access the Oracle DBA can grant.

Subsequent sections explore these steps in more detail.

**Establishing Access to an External Data Source**

In order to work with data tables in an external SQL source, you will need appropriate permissions. In some cases, you may have direct control over the SQL permissions, in which case you can establish appropriate permissions yourself. In other cases, you may need to request access from another individual or group. A determination of acceptable levels of access may be influenced by organizational policy, or compliance with external audit plans or regulations (such as the Sarbanes-Oxley act of 2002).

A DSN-based connection to an external SQL data source can store a user name and password. This is convenient for your DBA, since they need not establish credentials for each individual user. They can establish a single set of credentials for all FileMaker Pro access to a database. On the other hand, if the DBA already has more granular, user-level credentials in place, you can leave the user name and password unspecified when you define the DSN. A single user name and password can also be specified later on in the process, to be stored within the FileMaker Pro reference to the data source (perhaps the DSN is going to have non FileMaker Pro uses and it’s not appropriate to hard-code any access information). Alternatively, you can opt not to store the credentials anywhere. In this case, users will be prompted for a user name and password when they access the SQL data source. Depending on your circumstances, one scenario or the other might be more beneficial.

Regardless of whether you are establishing SQL database permissions for yourself, or working with a DBA to configure access, it is a good idea to limit FileMaker Pro’s interaction with the SQL database to the minimum necessary. This is a best practice for client access to any large data store: it is best to work only with the data you need, and configure permissions to allow only the necessary access while avoiding the possibility of costly errors. For example:

- If your FileMaker Pro solution only needs ten fields of data, find a way to access just those ten fields, and not the other thirty.
• If your users only need to read SQL data, and not to update SQL records, make sure they use credentials that prohibit write access to the SQL table.
• If your users only need to work with a few thousand out of hundreds of thousands of records, find a way to limit them to viewing just the records they need.

A common means of accomplishing all three of these goals is to define a database view within the SQL database. A view can be thought of as a sort of virtual table, with its own permissions. A view can combine data from several SQL tables, and limit the number of columns (fields) and records returned. If you need to work with a DBA to gain access, you may often find that DBAs prefer to grant access by creating appropriate views.

**Defining a DSN**

The next step in accessing an external SQL source is to define a DSN (Data Source Name) on the host machine. This may be a local computer, if the FileMaker Pro file is locally hosted, or it may be a server machine, if the file is being hosted using FileMaker Server. The ability to define and manage DSNs on the server is a major step forward in FileMaker Pro ODBC implementations.

**Defining a DSN in Mac OS X**

This section covers creating a DSN for MySQL 5 in the Mac OS. In order to do so, you would follow these steps.

1. Download and install an appropriate, supported driver. In this case, you might choose version 2.7 or later of the ODBC driver for open source databases, from Actual Technologies: [http://www.actualtechnologies.com/product_opensourcedatabases.php](http://www.actualtechnologies.com/product_opensourcedatabases.php). (You could also use the MySQL Connector ODBC: see the Appendix for details of supported drivers).

2. Open the Mac OS X ODBC Administrator (in /Applications/Utilities). Select the System DSN tab and press the Add button.
(Note that FileMaker Pro ESS only supports System DSNs, which are usable by all users on a given machine.)

3. Choose the Actual Open Source Databases driver:

![Choose A Driver](image)

4. On the “Create a New Data Source to an Open Source Database” screen, click Continue.

![Create a New Data Source](image)
5. On the following screen, choose MySQL for the database type. Give your DSN a suitable name and description. If you initiated the process from the System DSN tab of the ODBC administrator, the DSN will be created as a system DSN, regardless of which type (system DSN or user DSN) is chosen on this screen.

![Image of ODBC data source creation](image)

6. On the next screen, choose the hostname or IP address of the MySQL server. In this case, the server is running on the same machine as the DSN. You will only need to change the port if the MySQL server is running on a port other than the default.

![Image of ODBC data source creation](image)

7. On the following screen, choose the name of the database to which you wish to connect. (You will need to know the exact name of the desired SQL database).
8. When you have set all the necessary options, you will see a screen describing the DSN you created. Press Done.

(Note that the screens for these ODBC drivers may appear slightly different in practice as the drivers evolve over time, but they should remain substantially similar, if not identical, to the screen shown here).
9. You should now be able to see the new DSN in the ODBC Administrator. Be sure to press the Apply button in order to save your changes. You can now quit the ODBC Administrator and begin using the new DSN.

Defining a DSN in Windows

The process of creating a DSN in Windows is very similar to the process under the Mac OS.

1. Download and install an appropriate, supported driver. In this case you might choose version 3.51.14 or later of the MySQL Connector ODBC: http://dev.mysql.com/downloads/connector/odbc/3.51.html#win32. (See the Appendix for details of other supported drivers).

2. Open the Windows ODBC administration tool by choosing Start > Control Panel > Administrative Tools > Data Sources (ODBC). Choose the System DSN tab and press the Add button.
3. Choose the MySQL ODBC 3.51 driver.

![MySQL ODBC 3.51 Driver Image]

4. Enter a name and description for the DSN. You will also need to enter the host name or IP address of the database server, as well as a user name and password, and the name of the database you wish to access.

![Connector/ODBC Image]
5. Once you have clicked OK, the new DSN should appear in the System DSN window, ready for use.

Define an External Data Source in FileMaker Pro

“External data source” is the FileMaker Pro 9 and 10 terminology for any data source that is not part of the current FileMaker Pro file. Such data sources can be “dropped into” the FileMaker Pro Relationships Graph, where they appear with italicized names that show their external nature. In FileMaker Pro 7 through 8.5, external data sources could only consist of tables from other FileMaker Pro files, and were managed via the Define File References dialog. In FileMaker Pro 9 and 10, external data sources can also be ODBC-based. All external data sources, both FileMaker Pro based and ODBC-based, are now managed via the Manage External Data Sources dialog.

Suppose you have a new FileMaker Pro file, in which you are going to begin building an application. Your application needs to have access to customer data from a MySQL data source. You have finished configuring a DSN that will allow you to connect to the customer database, as shown in the previous section. Your next steps would be as follows.
1. Choose File > Manage > External Data Sources.

![Manage External Data Sources]

2. Choose New. Designate the data source as an ODBC data source. Fill in a user name and password (or leave them blank to force users to authenticate directly.)

![Edit Data Source]

- Type: ODBC
- DSN: Specify
- Authentication: Specify user name and password (applies to all users)
- User name: 'root'
- Password: *********
3. Press Specify in order to choose a DSN. You will see a list of DSNs that are available on the host machine. If the file is locally hosted, you will see a list of DSNs available on the local machine. If the file is hosted remotely, you will see a list of the DSNs that have been defined on the remote host.

Add Tables to the Relationships Graph

Now than an external data source has been defined, it is possible to add tables from that data source to the Relationships Graph in FileMaker Pro. This process is very similar to the process of referencing tables from external FileMaker Pro files.

1. Open the Manage Database dialog and navigate to the Relationships Graph. Add a new table occurrence to the graph. As in previous versions of FileMaker Pro, you have a choice between referencing a table from the current file, or a table from an external source (formerly accessed via a file reference). Here, the customers_mysql data source has already been defined, so it is available as a data source for the graph.

![Specify Table dialog](image)

2. Choosing the customers_mysql data source gives access to the SQL tables defined in that source. In this example, the MySQL database only has a single table, called customer, which can be accessed.
3. Choosing the remote customer table drops a new table occurrence into the graph. As in previous versions of FileMaker Pro, the italicized name of the table occurrence tells you that the table occurrence is based on an external data source. Within the table occurrence appear the names of all fields defined in the SQL customer table.
Manage the Field Structure of the New Table

When you first add a table from an ODBC-based data source to your schema, FileMaker Pro creates a shadow table in the local file. The shadow table contains field definitions for all of the fields in the remote SQL table. It is not possible to change the definitions of these fields. It is possible to delete fields from the shadow table. This has no effect on the SQL database itself. It only affects the fields that are fetched and displayed in FileMaker Pro. Deleting from the shadow table any fields you are not interested in is an effective way to limit the amount of data you need to fetch from the SQL source. If you need to restore deleted shadow fields, or otherwise re-synchronize the structure of the shadow table in FileMaker Pro with the structure of the SQL table on the server, press the Sync button near the top of this screen.

It is also possible to add supplemental fields to the structure of the shadow table. Supplemental fields, as the figure below demonstrates, are unstored calculation or summary fields that are defined locally in FileMaker Pro. In this example, a developer is adding a calculated field to create a "Last, First" display style for customer names.
**Work With the New Table**

With the new table added to the Relationships Graph, it is now possible to work with the SQL-based table as though it were a regular FileMaker Pro table. It is possible, for example, to build relationships between FileMaker Pro tables and SQL-based tables (or even between different SQL-based tables if you choose to do so, though it might be more efficient to have your DBA create a database view on the SQL server that performs the same function).

As an example, suppose it is necessary to extend the SQL-based customer table by tracking some additional information about each customer, such as fields for District and Region. These need to be stored, editable fields, so it isn’t possible to define them as supplemental fields within the customer shadow table itself. However, it is possible to create a FileMaker Pro native table to hold this information, and link this table to the shadow table view via a relationship. To do this, one might follow these steps:

1. Define a new FileMaker Pro native table, `customer_additional`, to hold the additional fields.

2. Make sure that, in addition to the new data fields that it is necessary to store, the additional table has an appropriate match field with which to related to the SQL-based table (the field in this case is `id_customer`).
3. Create a relationship between the SQL-based table and the FileMaker Pro based table.

4. Create a data layout that shows data from both tables. In the figure below, the FileMaker Pro native columns are highlighted. From the user's perspective, this screen represents a fully editable view of basic and extended customer data (assuming the FileMaker Pro application has permission to write to the SQL fields as well as read them).
ESS Tables vs. Native FileMaker Pro Tables

The preceding examples have shown that it is straightforward to configure an external data source and add some or all of its tables to the schema of a FileMaker Pro application. These external tables are usable within FileMaker Pro in a nearly seamless fashion. However, there are some key differences between ESS tables and native FileMaker Pro tables, which are discussed in the following sections.

Supported Data Types

SQL data sources, in general, support a greater number of data types than does FileMaker Pro. Often these data types are more specific than FileMaker Pro data types, and they can be more limited in certain ways as well. For example, most SQL data sources have different data types for integer data and floating-point data, while FileMaker Pro handles both integers and floating-point numbers with a single data type. SQL integer types may be limited in size of data they can hold. The data sizes of text-type fields can vary too; for example, the MySQL "text" type can only hold 65,535 characters by default, unlike FileMaker Pro’s Text type, which can hold about a billion characters in theory. Accordingly, though a MySQL text column will be mapped to a FileMaker Pro Text field, the FileMaker Pro shadow field will contain validation that limits input to 65,535 characters.
In some cases, an SQL field may be able to hold less data than its FileMaker Pro counterpart. In a few instances, an SQL field may be able to work with a greater range of data. In all cases, FileMaker Pro will do its best to impose validation on the shadow fields that will enforce these limits.

**Concurrency Issues: How Fresh is My Data?**

In a purely FileMaker Pro environment, each client’s view of the data is highly dynamic. If a FileMaker Pro user makes a change to a record, as soon as they submit the record, all other FileMaker Pro users will see the changed data. This is not the case for data from an ESS table. If a client changes ESS data, whether via FileMaker Pro or any other means, that change will not be immediately broadcast to other clients that may also be accessing that record. FileMaker Pro 10 periodically refreshes its view of SQL data, but there are times when the data visible in FileMaker Pro may be out of step with the most current SQL data.

In FileMaker Pro 10, the Refresh Window command is now available in the Records menu. Refresh Window can now flush any cached ODBC data and refresh the data from the ODBC source. Users (or your scripts) may need to use Refresh Window from time to time to be assured of working with the most current SQL data.

Record locking is another area in which ESS-based tables behave slightly differently. In FileMaker Pro based tables, when a user begins editing a record, other users are blocked from making changes to that record. This behavior does not exist for ESS records. Two users may edit an ESS record simultaneously. Whichever user submits the record first will be able to send their changes to the database. If the second user submits their copy of the record, they will see a warning like this one:
From here, they can decide whether to overwrite the other user’s changes, or undo their own.

**Indexing**

Databases need indexes in order to efficiently find data. In FileMaker Pro, indexes are used for searching, for sorting, and for creating relationships. Developers often do not need to worry about indexes in FileMaker Pro; unless directed otherwise, FileMaker Pro will create indexes as necessary in order to support any requested search, sort, or relationship choices.

The same is not true of ESS tables. FileMaker Pro cannot alter the schema of ESS tables, so it is unable to automatically create indexes to support requested operations. If a user chooses to run a search on an ESS field for which no SQL index exists, that search may run very slowly. Developers may wish to limit user searches to indexed SQL fields, and work closely with SQL DBAs to make certain that necessary indexes are in place.

**Schema Changes**

It’s been mentioned several times, but bears repeating: the ESS features do not afford FileMaker Pro developers any means to edit the schema of an SQL-based table or database. It is possible to edit the schema of the created shadow tables, by removing SQL fields from consideration, or adding supplemental unstored calculations or summary fields, but there is no direct means to change the SQL schema. In order to effect schema changes, you will need to change the SQL schema yourself, assuming appropriate access, or work with someone who can do so.

**ESS vs. Other Forms of SQL Connectivity in FileMaker Pro**

With the arrival of the ESS feature set, the FileMaker Product line now supports three distinct ways for FileMaker Pro based applications to interact with data from SQL data sources. The following table compares and contrasts these methods.
### Supported databases

<table>
<thead>
<tr>
<th></th>
<th>ODBC Import</th>
<th>Execute SQL script step</th>
<th>External SQL Data Source</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Supported databases</strong></td>
<td>Limited only by ODBC driver support</td>
<td>Limited only by ODBC driver support</td>
<td>Oracle 9i, Oracle 10g, Oracle 11g, MS SQL Server 2000, MS SQL Server 2005, MS SQL Server 2008, MySQL 5.0 and 5.1 Community Edition</td>
</tr>
</tbody>
</table>

- **Read records from ODBC sources**: X
- **Create, edit or delete records in ODBC data sources**: X
- **Dynamic, FileMaker Pro like interaction with ODBC data**: X
- **Send custom queries to an SQL database**: X
- **Create relationships between FileMaker Pro based and SQL-based data sets**: X
- **Extend SQL data sets with supplemental fields**: X
- **Integrate SQL tables into FileMaker Pro database structure**: X

The Execute SQL script step can potentially work with a much wider range of ODBC sources than ESS, limited only by available drivers. It cannot be used effectively to query for data, but only to create, edit or delete records on the server, or potentially to call stored procedures on the server. It affords no means to display ODBC-based data within the FileMaker Pro interface.

The ability to import from an ODBC data source is also potentially wide-ranging, but is almost the inverse of the Execute SQL script step. It can only be used to query for data, and cannot be used to create, edit, or delete records in the SQL source.

For supported ESS data sources, the ESS feature set offers the highest degree of interactivity and functionality. It does not, however, support sending custom SQL queries to a server, and its use is limited to three supported SQL products at present.
For data sources not supported by ESS, the ODBC import and Execute SQL capabilities of FileMaker Pro can be used together to provide easy and effective data sharing between FileMaker Pro and an ODBC-based data source.

**One important note about the ODBC Import and Execute SQL features is that they are not able to take advantage of server-side DSNs, as is the ESS feature set.** DSNs defined on a remote host are only available to ESS operations. Using ODBC Import and Execute SQL still necessitates configuring a DSN on each client desktop.

**ESS Best Practices**

ESS is an extremely powerful new tool in the FileMaker Pro toolkit. The best way to use it is to understand clearly what its design aims were. Those design aims can be stated as allowing FileMaker Pro solutions to integrate with data tables based in popular SQL storage systems, while working as transparently as possible within existing FileMaker Pro metaphors and conventions. In other words, solutions should be able to incorporate data from SQL-based systems, but the data needs to look and act like FileMaker Pro data as far as possible.

It is important to note that “increasing the potential size and scalability of FileMaker Pro based solutions” is not an explicit design goal of the ESS feature set. It may be possible to use ESS in that way, but the feature set is designed to favor transparency and ease of use, not raw speed. Therefore, it is probably not wise to expect that a good use of ESS would be to replace your largest and most complex FileMaker Pro based tables with SQL-based tables. This might work perfectly well, and might even solve certain problems for certain systems, but it would be wise to proceed with caution and to remember that the feature designers were trying to achieve “smooth, FileMaker Pro like integration” not “bigger, faster FileMaker Pro based systems.”

With this in mind, as a developer, there are a number of important steps and practices that will benefit you in your use of the ESS features.

**Learn Some SQL**

Though one of the most impressive aspects of the ESS feature set is the way in which it hides the details of SQL queries from the developer, you will still find it helpful to be familiar with the basic workings of SQL. If for some reason you don’t get the results you expect from an SQL-based server, some investigation may be involved: you may need to look at the SQL query logs, or discuss the structure of the SQL database with a DBA. As in all other areas of technology, the more you know the better. All three supported SQL database systems (Oracle, MySQL, MS SQL Server) have free editions that you can download and install. It is well worth your time to choose one and gain some facility with the system, the available tools, and the basics of the SQL language.

**Learn About Views**

A particularly important feature of SQL back ends is their ability to store and manage database views. A view can be thought of as a sort of stored query, a virtual table that can have its own access permissions. For example, given a customer table with ten million records and sixty data fields, a DBA might create a view called eastern_address_book based on the SQL query “SELECT name_first, name_last,
address, city, state, zip FROM customer_master WHERE region='East'. The view can be queried (and incorporated into FileMaker Pro) just like any other SQL table. It has the beneficial property of only selecting a few of the many possible customer fields, and of selecting only a subset of the customer records. Limiting the amount of SQL data you work with at one time is an essential best practice for ESS.

Watch Your Indexes

Unlike FileMaker Pro, an SQL-based system will not automatically create indexes to support user queries. If a user submits a query that is not well supported by existing indexes, the query may run very slowly. Think carefully about what indexes you will need, and make sure they exist on the SQL side. In particular, you will need indexes to support the following operations:

- Relationships: if you want to build relationships with SQL tables, any SQL that you wish to use as a relationship match field should be indexed (the relationship will still function, but it may resolve slowly)
- Searches: if a user wants to search on a field, it will likely need some kind of index in SQL. The ESS feature tries to get the SQL server to do as much query processing as possible. Certain searches may not be fully supported in SQL, in which case FileMaker Pro will “finish” the query processing itself, but proper indexing on the SQL side is essential.

Search Carefully

As an extension of the point about indexing, it is probably useful to limit the ways in which users can perform searches against ESS data sources. You will want to make sure those searches are being performed in properly indexed fields, for example. Therefore, you may want to create special search forms for ESS data, or use some form of visual highlighting to indicate searchable fields. You may want use FileMaker Pro’s layout features to inhibit users from entering non-indexed ESS fields while in Find mode.

Sort Carefully

Because of the rich, multi-language support in FileMaker Pro, and the complexities of character sets and character encodings, FileMaker Pro is not able to rely on specific SQL systems to sort records in the same way that FileMaker Pro itself would sort them. Accordingly, unlike searches (which FileMaker Pro tries to offload to the SQL system as much as possible), all sorting of ESS data is performed within FileMaker Pro itself. Because of the way FileMaker Pro queries for ESS data, sorting a large ESS data set is likely to be inefficient, and should be avoided. If your users consistently need to work with a large data set sorted in a particular order, consider defining a database view that incorporates the desired sort order on the SQL side. In general, it is best to sort only small groups of ESS records: a few hundred to a few thousand is likely to give the best performance.

Additional Resources

Terminology

Here is a brief recap of important terms for understanding the ESS feature set.
ESS: External SQL Sources allows data from select SQL data sources to be referenced dynamically from a FileMaker Pro database.

ODBC: Open Database Connectivity is a widely used interface for communicating with databases. ESS works via ODBC connections.

ODBC driver: a piece of software that supports ODBC communication to a particular ODBC data source.

DSN: Data Source Name, a group of configuration settings that tell how to use a specific ODBC driver to talk to a specific ODBC-capable database.

Shadow table: a table definition, stored in FileMaker Pro that “shadows” a table in an external SQL source.

Shadow field: a field definition within a shadow table that corresponds to a field definition in an SQL data source. This field “shadows” the SQL-based field. Its definition cannot be edited, unlike that of a regular FileMaker Pro field, but the shadow field can be deleted from the shadow table, which prevents FileMaker Pro from working with or displaying that field.

Supplemental field: an unstored calculation or summary field added to a shadow table. Supplemental fields supplement the SQL schema with unstored fields that are defined in FileMaker Pro.

FileMaker TechNet Paper

This paper covers the ESS feature set for the benefit of those who may be considering adopting ESS. This paper is complemented by an in-depth technical brief, available to members of the FileMaker Technical Network (TechNet) that covers the ESS feature set in greater technical detail. The TechNet paper is intended as a guide for developers who are adopting and implementing ESS, and want to understand the specifics of how FileMaker Pro uses ESS to interact with SQL-based systems.

For information on becoming a member of TechNet, please visit http://www.filemaker.com/technet/.

Supported ESS Data Sources

ESS supports the following data sources at present. If you are working with any of these, consider downloading the vendor’s “light” or “express” version if you are not already familiar with the product.

Oracle Database Server (9i, 10g and 11g): http://www.oracle.com/database/index.html (Note the link to the Express version on that page)

MySQL Community Edition (5.0 and 5.1): http://dev.mysql.com/downloads/mysql/5.0.html (There’s no “light” or “express” version of MySQL: the Community Edition is downloadable at no charge. You might also want to look into the graphical administration tools at http://dev.mysql.com/downloads/gui-tools/5.0.html)

Supported ODBC Drivers

It is important to use ESS with supported ODBC drivers. Using unsupported drivers may give incorrect results.

Updated information on the latest supported drivers and data sources can be found at http://www.filemaker.com/support/technologies/sql.html.

Books

Even though ESS hides the details of SQL programming from users and developers, some familiarity with SQL is still useful when using ESS. Here is a reference to a free online book covering SQL in some detail. It is not the most current edition but the fundamentals of SQL have been quite constant and the material in the book is a solid starting point.


About the Author

Steve Lane is Vice President of Soliant Consulting, where he has led FileMaker and web development teams for the last eight years. Steve has co-authored five books on FileMaker, and is a frequent speaker at the FileMaker Developer Conference, where in 2003 he was awarded the FileMaker Fellowship award for “pushing the boundaries of FileMaker Pro.” He is an authorized FileMaker trainer and has been instrumental in the development of the last three versions of the FileMaker Professional Training Foundation Series materials. Steve has consulted with a wide variety of FileMaker clients, from small private companies to large public firms, and has specialized in integrating FileMaker systems with web-based and SQL-based systems. He holds a PhD in medieval history from the University of Chicago, where, years ago, he created his first relational database using the Macintosh-based Double Helix system.

This paper benefited considerably from the advice and insights of others. In particular, Galt Johnson and Clay Maeckel provided crucial technical critiques, and Ray Cologon called out and explored an important aspect of SQL Server behavior related to date/time fields.

---

©2009 FileMaker, Inc. All rights reserved. FileMaker is a trademark of FileMaker, Inc., registered in the U.S. and other countries, and the file folder logo is a trademark of FileMaker, Inc. All other trademarks are the property of their respective owners. FileMaker makes no warranties, express or implied, with respect to the performance or reliability of any products presented herein that are manufactured by independent vendors. All understandings, agreements, or warranties, if any, take place between the vendors and prospective users. Product specifications and availability subject to change without notice. (Docv2)
THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, AND FILEMAKER DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, OR THE WARRANTY OF NON-INFRINGEMENT. IN NO EVENT SHALL FILEMAKER OR ITS SUPPLIERS BE LIABLE FOR ANY DAMAGES WHATSOEVER INCLUDING DIRECT, INDIRECT, INCIDENTAL, CONSEQUENTIAL, LOSS OF BUSINESS PROFITS, PUNITIVE OR SPECIAL DAMAGES, EVEN IF FILEMAKER OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY. FILEMAKER MAY MAKE CHANGES TO THIS DOCUMENT AT ANY TIME WITHOUT NOTICE. THIS DOCUMENT MAY BE OUT OF DATE AND FILEMAKER MAKES NO COMMITMENT TO UPDATE THIS INFORMATION.