CHAPTER 7

Working with Tables

A *table* is a collection of information pertaining to some specific type of entity, made up of a collection of *records*. They are the central focal point of the database structure, or schema. Tables define the field structure of the data that will be stored within them (Chapter 8). Instances of tables can be interconnected to form relationships (Chapter 9). A table's fields can be displayed on layouts (Part 4) and manipulated with scripts (Part 5).

In this chapter, we will begin defining and building tables. Topics will include:

- Modeling tables
- Introducing the manage database window (Tables)
- Naming tables
- Managing tables

Modeling Tables

Data modeling is a term used to describe the process of planning and creating an abstract, virtual representation of the various properties, relationships and actions that describe a set of related entity classes that will be managed within a database. This term is used because the resulting table, field, and relational structures of a database are sometimes called a *virtual model* of the types of entities that are represented by the data structure.

The first step in planning a database model is to determine the names of the tables required. A table's object category might be a broad category (people or products) or a narrower subcategory (employees or cars), depending on the specific needs of the database you are building. In some cases, a table may even model some aspect or property of an entity or action, such as a quality of an object or process of an event.

In the early planning stages, a data model can simply be a list of table names. Later, you will want to sketch out a list of the fields that will be defined within each and indicate how the tables will interconnect to form a relational hierarchy.

M. C. Munro, *Learn FileMaker Pro 16*, DOI 10.1007/978-1-4842-2863-0_7

For now, our hypothetical data model can simply list the entitiy classes that will each be represented by a table. For example:

- Company
- Contact
- Project

As we expand into other areas of the data structure in later chapters, we will expand this model.

Introducing the Manage Database Dialog (Tables)

Tables are defined from the `Tables` tab of the `Manage Database` dialog. Once a database is open, this dialog can be opened by any of the following methods:

- In browse or layout mode, select the `File > Manage > Database` menu item.
- In layout mode, click the `Manage` icon in the toolbar and select `Database` from the pop-up menu.
- Running a script or clicking a button that uses the `Open Manage Database` script step.
- In the `Import Field Mapping` dialog (Chapter 5), click the `Manage Database` button.
- In most table selection pop-up menus throughout the programming interface, select the `Manage Database` option at the bottom.

Once open, make sure the `Tables` tab is selected, as shown in Figure 7-1.

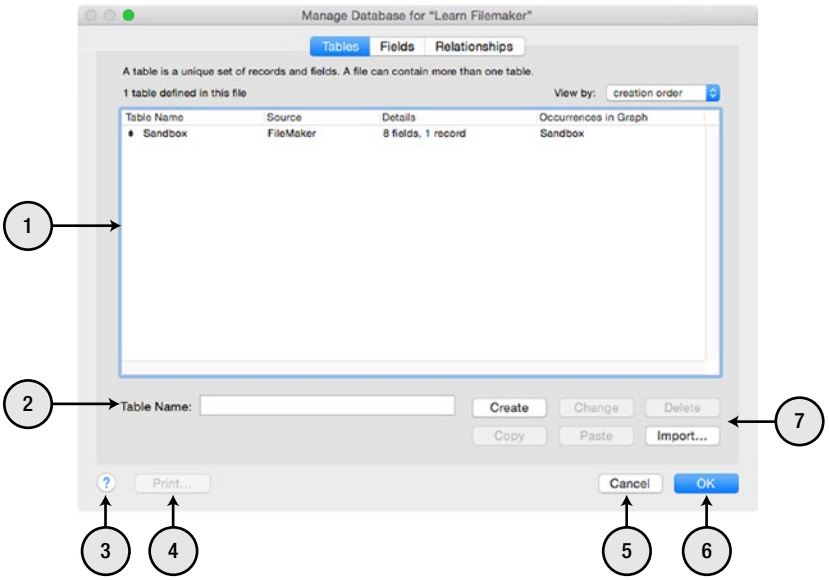

Figure 7-1. *The `Tables` tab of the `Manage Database` dialog*

The controls on the Tables tab are:

1. Table List — This area displays a list of all the tables that are defined within the current database file. The list can be sorted by either clicking on a column heading or choosing an option from the View by pop-up menu in the upper right corner. By default, a newly created file will automatically contain a single table with the same name as the file. However, in this case, it is different because we renamed that default table in the last chapter.
2. Table Name — This text area contains the name of the currently selected table. It is used to edit the table name or to enter a new table name.
3. Help — This button opens the FileMaker help guide to the "Defining Database Tables" page.
4. Print — This button will print the field definitions for the selected table(s).
5. Cancel — This button will close the dialog without saving changes in any tab after displaying a warning.
6. OK — This button will save changes and close the dialog.
7. Buttons — This group of six buttons is used to add tables and control table management functions described later in this chapter.

Table Properties

Tables are made up of five components, each listed as a column in the Table List on the Tables tab of the Manage Databases dialog.

- Table Name — This column contains the name of each table in the list.
- Source — This column contains the source type of each table, either "FileMaker" or the name of an ODBC data source.
- Fields — The Details column includes a count of the fields that are defined for each table (Chapter 8).
- Records — The Details column includes a count of the records that are stored in each table.
- Occurrences in Graph — This column contains a comma-separated list of every occurrence of the table in the relationship graph (Chapter 9).

Naming Tables

Table names can be flexible depending on a developer's preferences. Depending on the database's design, users may never even be aware of a table name so naming isn't typically important from an interface perspective. However, from the perspective of an often-overlooked *technical design,* naming can greatly impact your performance as a developer when working in the programming interface.

Table names should at least follow these rules:

- They should be a unique word or phrase.
- They should not contain the name of functions, especially those that have no parameters such as `Random` or `Pi`.
- Although many reserved symbols and words *can* be used in a name, avoiding them altogether is a good idea. Some will cause conflicts in calculation formulas and FileMaker will warn you about these and confirm that you want to continue.

Including spaces in table names is optional. For some web integrations, other than FileMaker Web Direct, you may want to avoid them completely. Generally, any of these name formats are perfectly fine for tables:

```
Project Resources
Project_Resources
ProjectResources
```

Beyond these, consider what constitutes a *good* table name. Ideally, the name of any object in a database should be carefully planned to be clearly descriptive, concise but not cryptic, and integrated with similar objects into groups but still differentiated from each another. All this while considering how the name affects sorting with other objects in a list.

Here are a few guidelines to consider:

- A table name should clearly indicate the class of entity being modeled by the table while being concise without being cryptic.
- In almost every case, using full words is preferred over abbreviations. If a word seems too long, consider using a thesaurus to find a replacement.
- Remember, the choice of naming should be considered in the full context of other tables in the file. For example, `Stuff` may be unclear but acceptable in a database that manages only one kind of stuff. However, as a database expands to include multiple tables containing data for several different kinds of stuff, it becomes prudent to use more descriptive names like `Inventory`, `Resources`, `Supplies`, etc.
- Consider using several words to increase clarity where applicable. When there are other tables that store data for similarly named entities, make sure that their names *differentiate* each table's uniqueness. This will help avoid developer confusion.
- Including prefixes to group tables will help to *integrate* them into subgroups by causing them to sort together alphabetically. This is especially helpful when selecting a table in pop-up menus throughout the developer interface. A prefix can be one or more words but shouldn't expand the name to the point where it causes the opposite of cryptic vagueness, cluttered overload. Sometimes a grouping prefix may require a reversal of common language order. For example, instead of `Company Property` and `Personal Property`, reversing these as `Property Company` and `Property Personal` allows the common term to become a sorting prefix as well as entity class identifier.
- Be consistent in usage. For example, consider keeping all names either singular or plural; `Contact` and `Company` or `Contacts` and `Companies` are both fine, while `Contacts` and `Company` is inconsistent.

Using the above advice, consider a few examples of good table naming below:

```
Company
Company Offices

Contact People
Contact Company
Contact Log

Project
Project Budget
Project Scheulde

Invoice
Invoice Line Item
Invoice Purchase Order
```

■ **Note** Remember, your preferences and needs for naming may be validly different than the isolated suggestions above. Never assume there is an absolute, context-less rule for naming that must be followed all the time.

Managing Tables

Tables can be selected, added, renamed, and deleted using the buttons on the `Tables` tab of the `Manage Database` dialog.

Selecting Tables

Many table management functions require a selection of at least one table. Selections can be made in the following ways:

- To select a *single table*, click on it in the list with the cursor. To change the selected table, click on a different table or use the `Up` and `Down` arrow keys to select the previous or next table.
- To select *multiple consecutive tables*, hold down the `Shift` key and click on two tables at the outside of the range desired.
- To select *multiple non-consecutive tables*, hold down the `Command` key and click on each desired table, one at a time.

Adding Tables

There are *six* different methods for adding tables to a database file:

- Creating a new table.
- Adding a table from an ODBC data source.
- Duplicating an existing table.
- Importing a table.
- Automatically converting a data file to a database.
- Importing records (Chapter 5, "Selecting a Target Table").

Creating a New Table

Manually creating a new table from the `Tables` tab of the `Manage Database` dialog is a quick and easy two-step process:

1. Type the name of the new table into the `Table Name` text area.
2. Click the `Create` button.

The new table should appear in the list above with any other tables already defined.

Adding a Table from an ODBC Data Source

Open Database Connectivity (ODBC) is a standard application programming interface (API) that provides client applications a common language for interacting with other database systems written in the C language. *Java Database Connectivity* (JDBC), which FileMaker supports, is a similar API for accessing systems written in the Java language.

An ODBC or JDBC client application communicates with a *Driver Manager* application that uses a client driver to communicate with the external data source, as shown in Figure 7-2.

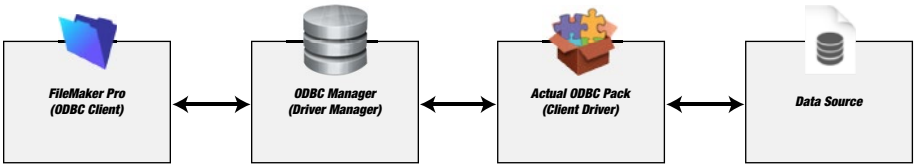

Figure 7-2. *The connection between a FileMaker and an external data source through an ODBC connection*

FileMaker can act as an ODBC client application or as an ODBC and JDBC data source. As a client application, FileMaker supports connections to external SQL data sources such as those from Oracle, Microsoft SQL, and MySQL. Once connected, tables from the external database can be added to a FileMaker database and, except for a few exceptions, can be used like a native FileMaker table.

The process of configuring a host computer and database for ODBC access is a relatively straightforward three-step process:

1. Prepare the host computer for an ODBC connection with a *Driver Manager* and *Client Driver*.
2. Connect the FileMaker database to the ODBC client driver.
3. Insert and use ODBC tables in the FileMaker database.

Preparing a Computer for an ODBC Connection

The first step to connecting a FileMaker database to an external ODBC source is to prepare the host computer for an ODBC Connection. This involves installing the ODBC Manager application (freeware for macOS), installing a client driver (such as Actual Technologies ODBC Pack), and then configuring the driver for a specific external data source.

The *host computer* is the computer upon which the database is installed (rather than a client computer that may be accessing it). This might be your computer with the database open in a client version of the FileMaker client application, used locally or shared across the network, or a dedicated FileMaker Server.

Installing the ODBC Manager Application

Download and install the freeware *ODBC Manager* application onto the host computer, by following these steps:

1. Download the *ODBC Manager* disk image from `http://www.odbcmanager.net`.
2. Locate the `ODBC_Manager_Installer.dmg` file in your `Downloads` folder and launch it.
3. The *ODBC Manager* disk image will appear on your desktop and open in a window.
4. Double-click on the `ODBC Manager.pkg` file to launch the installer.
5. Step through the installer panels to complete the installation.

Once the installer has run, the ODBC Manager application should be in your `/Applications/Utilities/` folder.

Installing the ODBC Driver

Next, download and install an ODBC driver, like the one described here from Actual Technologies, by following these steps:

1. Download the *Actual ODBC Pack* disk image, available at `http://www.actualtech.com/download.php`
2. Locate the `Actual_ODBC_Pack.dmg` file in your `Downloads` folder and launch it.
3. The *Actual ODBC Pack* disk image will appear on your desktop and open in a window.
4. Double-click on the `Actual ODBC Pack.pkg` file to launch the installer.
5. Step through the installer panels to complete the installation.

■ **Note** The driver is a fully functional demo, limited to display only the first three rows resulting from any query. Purchase a license key from Actual Technologies web store to remove this limit. They have two editions available, one for use on a single user database and another for shared databases.

Configuring Driver

Next, add and configure a driver for the specific database to which a connection will be made by following these steps:

1. Open the ODBC Manager application, shown in Figure 7-3.

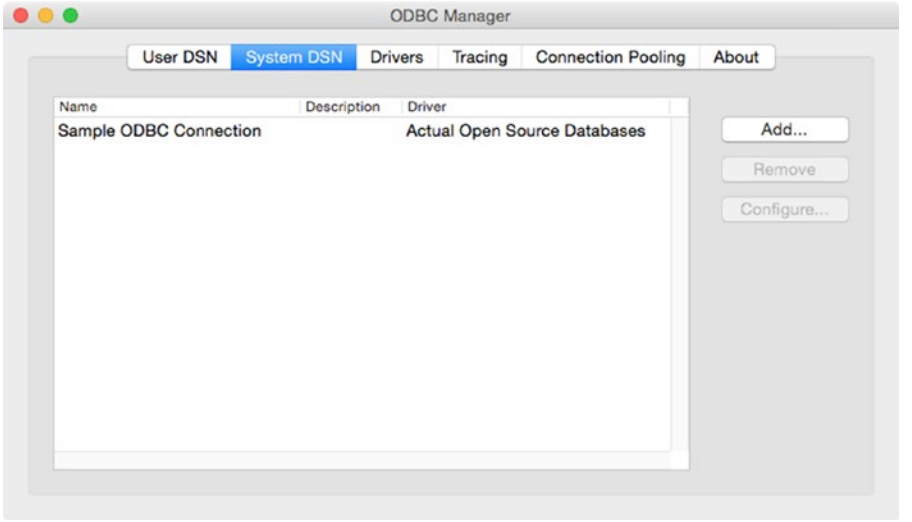

Figure 7-3. *The ODBC Manager application window with a single ODBC connection configured to a sample database*

2. Click on the `System DSN` tab.
3. Click the `Add` button.
4. A panel will appear attached to the dialog. Choose the appropriate driver from the list for the specific database you are connecting to. The list will include the following four driver options from Actual Technologies:
    - Actual access
    - Actual open source databases
    - Actual oracle
    - Actual SQL server

5. Enter settings into the multi-panel driver configuration dialog. The specific settings required will vary slightly depending on your choice of driver. However, generally, you must enter the server address, database name, username, and password.
6. When finished, click the `Done` button to close the configuration dialog.
7. The final panel of the configuration dialog allows you to test the connection to ensure that you have entered everything properly.
8. Quit the ODBC Manager application.

Connecting a FileMaker Database to the ODBC Client Driver

Next you must connect your FileMaker database to the ODBC driver you just configured in the ODBC Manager application. This is done by setting up an external data source in the database and then adding the table(s) from the external source into the FileMaker relationship graph.

Setting Up External Data Source

Select the `File > Manage > External Data Sources` menu item to open the `Manage External Data Source` dialog, shown in Figure 7-4, then click the `New` button.

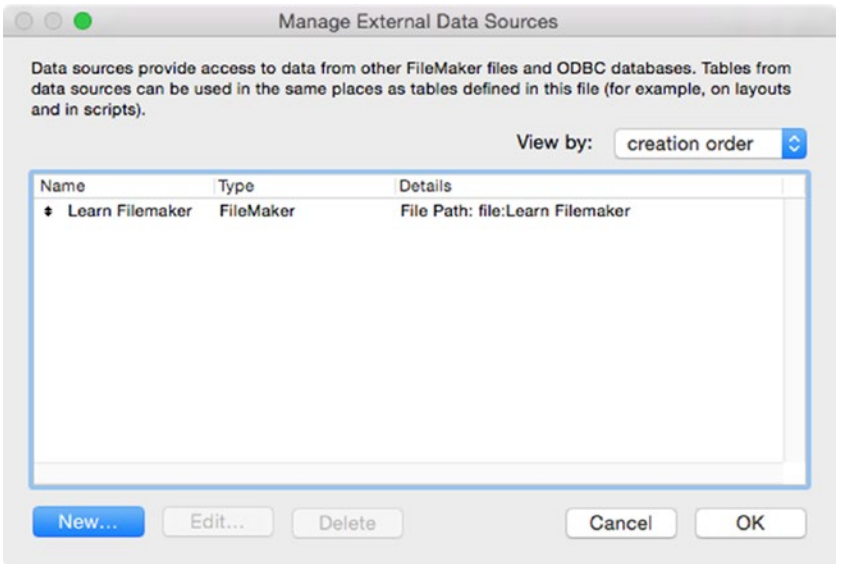

Figure 7-4. *The `Manage External Data Sources` dialog*

An empty `Edit Data Source` dialog will appear, like the one shown in Figure 7-5. Following the steps below to configure the connection to the Client Driver for the external data source.

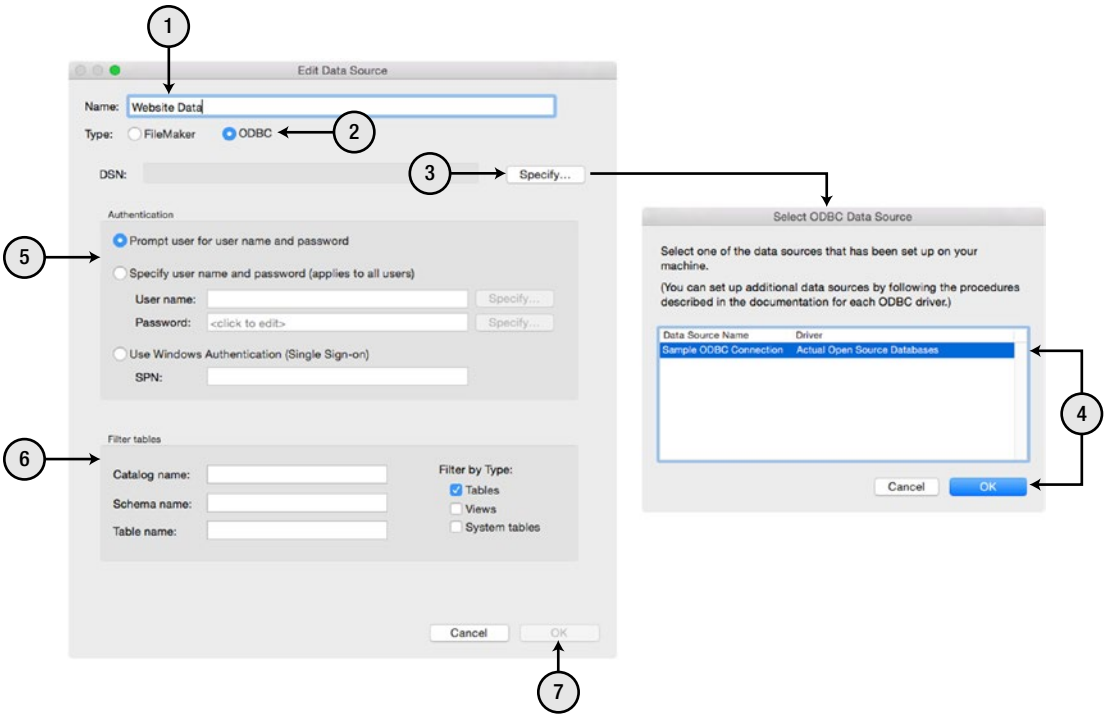

Figure 7-5. *The process for setting up an external data source*

1. Enter a name for the data source in the `Name` text area, which will be used to refer to the data source within the FileMaker database. It can be the same as the `Data Source Name` or a more descriptive name (as shown).
2. Select the `ODBC` radio button for the `Type`.
3. Click the `Specify` button on the right of the `DSN` text area.
4. In the `Select ODBC Data Source` dialog that opens, select from the available `ODBC Data Sources`. The list will include any you have previously defined in the ODBC Manager application. Once you select the desired driver, click the `OK` button to close the dialog.
5. Choose the `Authentication` option to indicate how a username and password should be entered at the user level when they attempt to access resources in the external database from within FileMaker. For the smoothest user experience, hard-code the user name and password directly in this area. For a more secure experience, have them prompted to enter these values at runtime.
6. Optionally, you can add filtering criteria to control which tables are displayed when a connection is made. This can be helpful when a database has many tables and you want to limit which are available from FileMaker.
7. Click the `OK` button to close the `Edit Data Source` dialog and save the current configuration.

The new external data source should appear in the list on the `Manage External Data Sources` dialog, shown in Figure 7-6. Click the `OK` button to close the dialog and save the configurations.

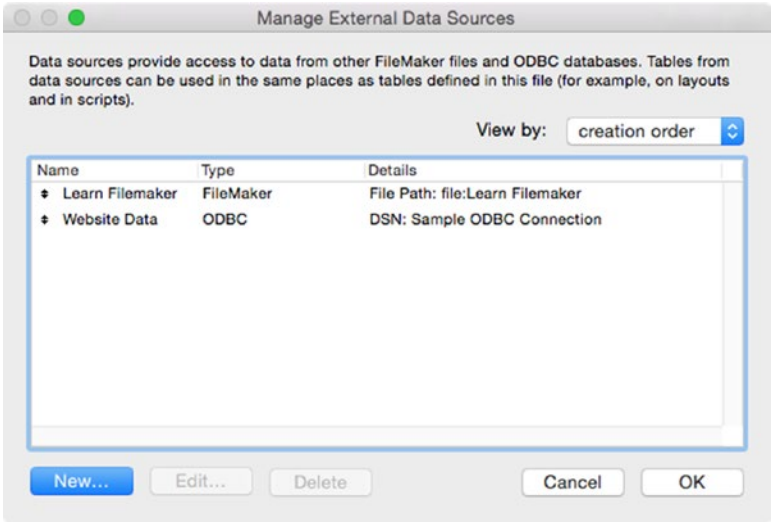

Figure 7-6. *The dialog after setting up the connection to the external data source*

Adding to Relationship Graph

Any table created directly within the `Tables` tab of the `Manage Databases` dialog will automatically be added to the Relationship graph, while an `ODBC` table must be manually added. To get started, open the `Manage Database` dialog and click on that tab. Then follow the steps shown in Figure 7-7.

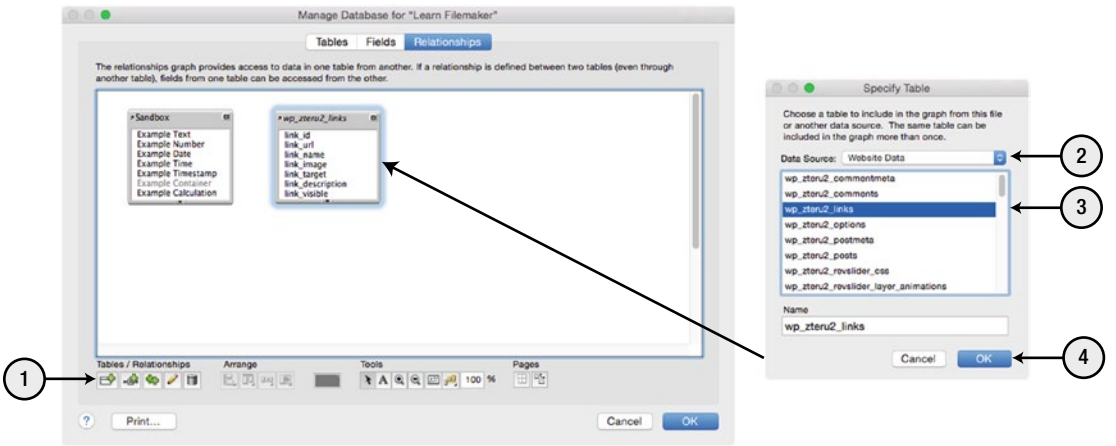

Figure 7-7. *The process for adding an external data source to the relationship graph*

1. Click the Add a table icon button, which is located on the far left of the row of icons.
2. In the Specify Table dialog that opens, click the Data Source pop-up and select the ODBC data source containing the table you want to add. This should refresh the list of available tables.
3. Click the table you want to add.
4. Click the OK button.

A new table occurrence box should appear in the relationship graph. If you switch to the Tables tab you will see the table in the list with italic formatting, which indicates an ODBC "shadow table."

Using an ODBC Table in FileMaker

Once an ODBC table occurrence is added to the relationship graph, it is available throughout the database in almost the same way as any native FileMaker occurrence. However, it is important to remember that the table really exists on the external server and is represented within FileMaker as a "shadow table" so there are a few notable differences. These include:

- *Schema Lock* — The structure of the external data source is not available for modification from within FileMaker. The shadow table created within FileMaker allows some modifications but they do not affect the remote table.
- *Deleting Fields* — Fields can be deleted from the shadow table to help thin out the amount of data queried but the fields remain intact in the remote table.
- *Modifying Fields* — Auto-Enter settings for remote fields can be modified in the shadow table since they essentially operate in the same way as users entering information into a field.
- *Adding Supplemental Fields* — You can add additional fields to the shadow table but are limited to unstored calculations and summary fields and the new fields are FileMaker-only and will not be added to the remote table.
- *Data Types* — SQL has separate data types for certain information, such as integers and floating-point data, which FileMaker handles as a single data type. In some cases, you may need to create a calculation field to convert data from the external table into a FileMaker data type using one of the GetAs functions (Chapter 13).
- *Data Entry Limitations* — When there are differences in the amount of data some field types can hold in the remote table, FileMaker will do its best to validate and enforce these limits to avoid any data loss or confusion. For example, MySQL fields tend to be more limited in content size than are native FileMaker fields.
- *Data Updates* — Updates to refresh record changes in the external table are less frequent across the network than changes to native FileMaker tables. Use the Refresh script step to force a window to update and ensure that the data the user sees is current.
- *Record Locking* — Unlike with FileMaker native tables, when a user begins editing a record in an external table, other users are not blocked from making changes to the same record. So, two users can be editing the same record simultaneously. The user who commits the record second, thereby submitting their changes after the other user, will see a warning dialog informing them that the record had been modified since they began editing and give them the ability to stop to avoid overwriting the other user's changes.
- *Indexing* — FileMaker can't index SQL fields so searches in external tables should be limited to those fields already indexed by the remote table to avoid long delays.

Getting More Information

FileMaker has a technical brief document available on its website entitled "Introduction to External SQL Sources" that provides more information about ODBC, JDBC, and the various connection options to and from FileMaker databases.

Duplicating an Existing Table

When an existing table is similar enough to the requirements of a new table, consider duplicating it rather than starting from scratch. While there is no `Duplicate` button, the `Copy` and `Paste` buttons serve this purpose. To duplicate a table, follow these steps:

1. Select the table(s) that will be duplicated.
2. Copy the table(s) by clicking the `Copy` button.
3. Paste a duplicate of the copied table(s) by clicking the `Paste` button.

The duplicate table(s) will appear in the list, highlighted as shown in Figure 7-8. Each new table will have the same name as the original with a numeric suffix added to ensure it is unique. Every field from the original table will be present as an exact duplicate in the new table, with the same names and settings. Furthermore, a similarly named occurrence will be placed on the relationship graph (Chapter 9). The only difference is that the new table will not copy records from the original.

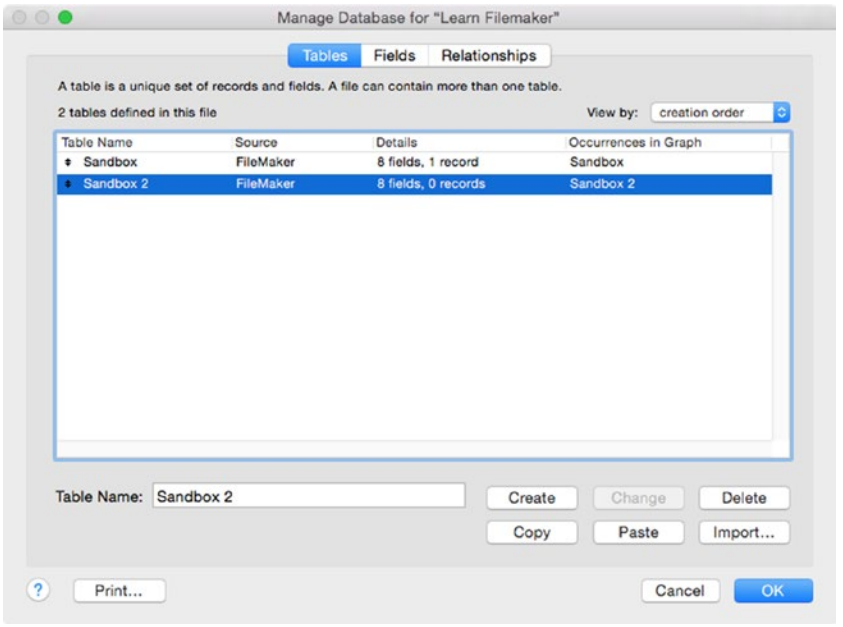

Figure 7-8. *A duplicate of the Sandbox table after pasting*

■ **Note** Tables can be copied and pasted *within* a file and *between* two files.

Importing a Table

The `Import` button on the `Tables` tab of the `Manage Database` dialog allows one or more tables to be imported from a different file directly. To import, follow these steps:

1. Click the `Import` button. An `Open File` dialog will appear.
2. Locate and open the database file containing the desired tables. This can be a file in a folder directory or hosted by a FileMaker Server. It can be the same file you are editing or a different file. Once opened, FileMaker will present an `Import Tables` dialog, shown in Figure 7-9, listing all the tables available for import from the selected database.

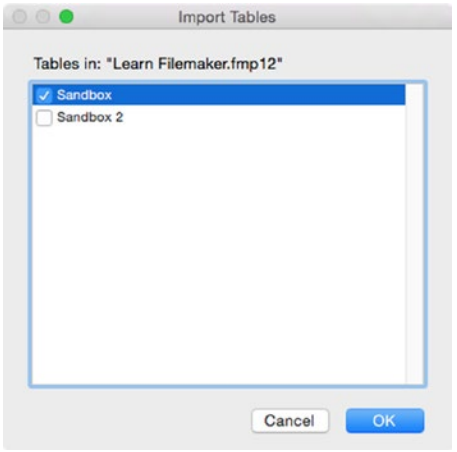

Figure 7-9. *The Import Tables dialog*

3. Check the check box next to the table(s) to import and click the `OK` button.

The selected table(s) will be copied into the current database and an `Import Summary` dialog will report the results, including:

- The number of items imported
- The number of items that were automatically renamed due to conflicts
- The number of errors that occurred

An `Import.log` file will be saved into the folder contains the destination file or onto the `Desktop`. If the destination file is remotely hosted.

Converting a Data File to Database

FileMaker can instantly convert a text or spreadsheet file into a new database when these are dropped on the application. The columns and rows of data from the source file will be automatically converted into a table, fields and records in the new file created by this process.

When the file is dropped onto the application, a dialog will offer two choices for field name handling, depending on what information the first row of incomiong data:

- `Field names` — This option indicates that the first row of data in the file being imported contains field names rather than actual data. If selected, the first row of data will be used to provide the names of fields into which the remaining information will flow.
- `Data` — This option indicates that the first row of data in the file being imported contains actual data and not field names. If selected, field names will be given a default value (`f1`, `f2`, `f3`, etc.) and the first row will be imported as a record.

Once this choice is made and the `OK` button clicked, the process will build a new file with the following resources:

- The new database's name will be the same as the data file with a suffix of "Converted" and the FileMaker file extension.
- A default table will be created with the same name as the new database file.
- A field will be created for each column of data with naming based on the choice made in a dialog.
- A record will be created for each row of incoming data, with each column of information entered in a corresponding field.
- Two layouts will be created: a form layout named "Layout #1" and a list layout named "Layout #2."

Modifying a Table Name

Changing the name of an existing table is a simple three-step process.

1. Select the table to be renamed.
2. In the `Table Name` text area, type the new name.
3. Click the `Change` button.

Once complete the name of the table in the list should be updated. Click the `OK` button to commit this change.

Deleting Tables

Depending on which of the previous examples you performed, the `Learn FileMaker` database may now have a bunch of duplicate tables. Fortunately, deleting tables is a simple process:

1. Select the table(s) to be deleted.
2. Click the `Delete` button.

3. A `Delete Tables` warning dialog will appear to confirm you want to continue, as shown in Figure 7-10.

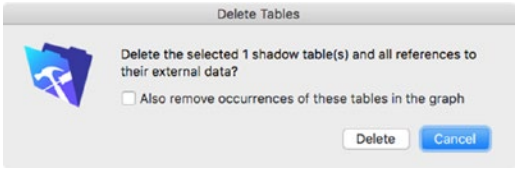

Figure 7-10. *The warning dialog when deleting tables*

4. The dialog includes a check box that, when selected, will delete both the tables *and* their corresponding occurrences in the relationship graph. This choice is optional and can vary depending on the reason for deleting the tables. For now, you should check the box. If you were deleting a table whose relationships will be repurposed with a new or existing table, you would want to uncheck the box to retain those connections. The table occurrences in the graph will become `<missing table>` indicating that they require reassignment.
5. Click the `Delete` button in the dialog to complete the process.
6. Manually delete and repurpose any layouts assigned to the deleted table(s).

Once complete the selected table(s) will disappear from the list. Click the `OK` button to commit this change.

■ **Caution** When deleting a table, all records that were contained within it will also be permanently deleted.

Summary

In this chapter, we discussed the basics of working with tables, including accessing external SQL data sources from within FileMaker.