Key Concepts in FileMaker 7

Mapping New Territory

Michael Harris
Senior Partner, Cerné Systems, Inc.

FileMaker 7 may be a work of genius. Time will tell. It certainly has those hallmarks of genius: simplicity and subtlety. The trouble is the subtlety bit. For most people, FileMaker 7 is not at all what it first appears to be. The simplicity is only available down the road somewhat, past the subtle curves.

The most challenging part of FileMaker 7, its new relational architecture, is the topic of this paper. This architecture is best understood as a gestalt, through seeing the pattern in a collection of new concepts working together. So, reading this document is necessarily a process of consecutive partial understandings and the gradual coming into focus of the whole. This paper is a learning map for that process, meant to be marked up and creased, recording the history of its usefulness.

On this map, key concepts, as defined by the "Vocabulary" and embodied in the "New Rules," are the landmarks. Like a real map, best use depends on the reader's starting point. For those with the least experience with FileMaker 7, read in the order written. Those with some FileMaker 7 experience might begin with the "Vocabulary" and "New Rules" sections. For the most experienced and the brave, start with the "Design Space" section. For all of us the "New Rules" are the road test. When the "New Rules" make complete sense, it is time to throw this map into the glove box.

For most people, FileMaker 7 is not at all what it first appears to be.

What Changed

The biggest structural change in FileMaker 7 is the loosening of the connection between **files** and **tables.** Before FileMaker 7, a file and a table were tightly bound – one file, one table. Files and tables were so inseparable that the term "table" was barely used in FileMaker circles, except to make an esoteric point about relational theory. One advantage of this overlap was that there was never any question about what table you were talking about when scripting or defining fields or doing other development work; the work could only be done in the file holding the table data you needed. If you required direct access to data (as opposed to access through relationships) from another file/table you had to move to that file.

FileMaker 7 loosens but does not break the bond between files and tables. Tables still may only be created from within a file, and every FileMaker **database** must have at least one file. Tables have a natural home in the file in which they are created. Only in this **home file** can table definitions be created and basic security managed. In other aspects of table usage, any FileMaker 7 file can use any suitable table, including those in other files and in other files at remote locations. This creates a new and wide-ranging need to tell FileMaker "What table are you talking about?" This question is answered by pointing layouts, scripts and field definitions to a particular table through a **Table Occurrence (TO)** in the **relationships graphs** in files (see Figure 3a).

Referencing a TO does much more than tell FileMaker which table to use; it also tells FileMaker what additional tables are available through **relationships** and what the relational conditions are for seeing that data. The same table may be referenced through many different TOs, in many different files, and while the same **primary table** data is available in every case, the related table data may be completely different. (See Figure 3a. The **table occurrences** "MailPerson", "PhonePerson" and "RidePerson" all point to the same primary table but have entirely different access to related tables.)

In previous versions of FileMaker, the one file/

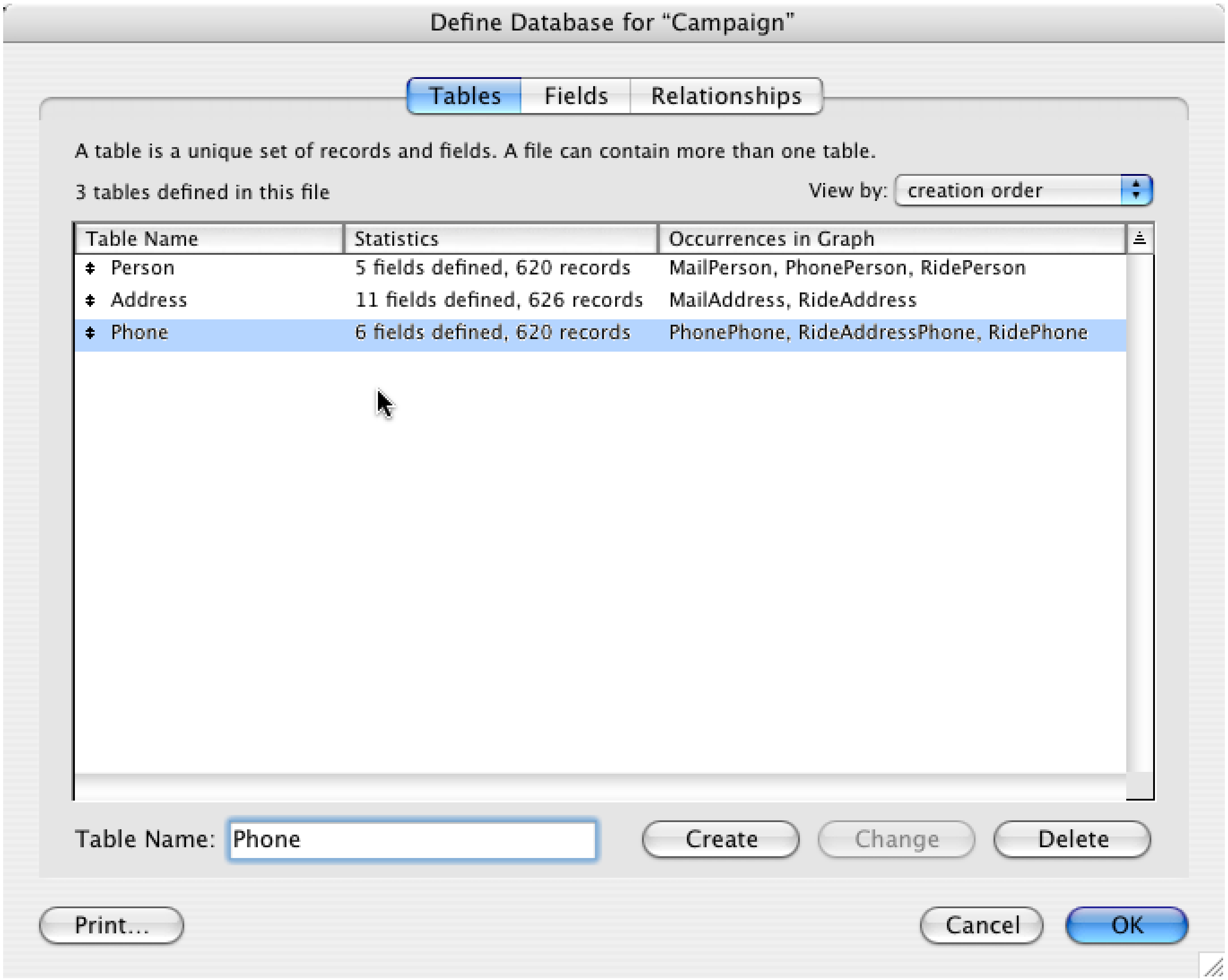
one table structure meant that any relationships established are universal in the sense that a business process must go to the appropriate file/table to use a relationship, and all relationships in that file are available to be used. In FileMaker 7, each file has its own relationships graph, which can reference any available table in the database (whether the current file is the table's home file or not) and the relationships established only apply when using that file. If, for example, a relationship between a Person table and an Address table is required in three different files, then that relationship must be re-specified in each of the three files. The good news is that a Person-to-Address relationship may be different in each file, giving a new level of control.

The result of all the loosening up of FileMaker structure is a new requirement to be specific about **context.** You must now answer the "what are you talking about" question much more often. Initially, a feeling of vertigo, or even existential nausea, is to be expected.

Case Study

Denise is running for public office. Her campaign needs a voter database. Three campaign activities are critical: mailings, phone calls, and rides for voters to the polling place. Since the campaign has untrained volunteers and is not an ongoing business, the database should be simple to use, robust, and flexible.

For mailings, the volunteers need to see the data from the point-of-view (POV) of an address because they want to send a single mail piece to every



A table is a unique set of records and fields. A file can contain more than one table.

3 tables defined in this file

View by: creation order

Table Name	Statistics	Occurrences in Graph
Person	5 fields defined, 620 records	MailPerson, PhonePerson, RidePerson
Address	11 fields defined, 626 records	MailAddress, RideAddress
Phone	6 fields defined, 620 records	PhonePhone, RideAddressPhone, RidePhone

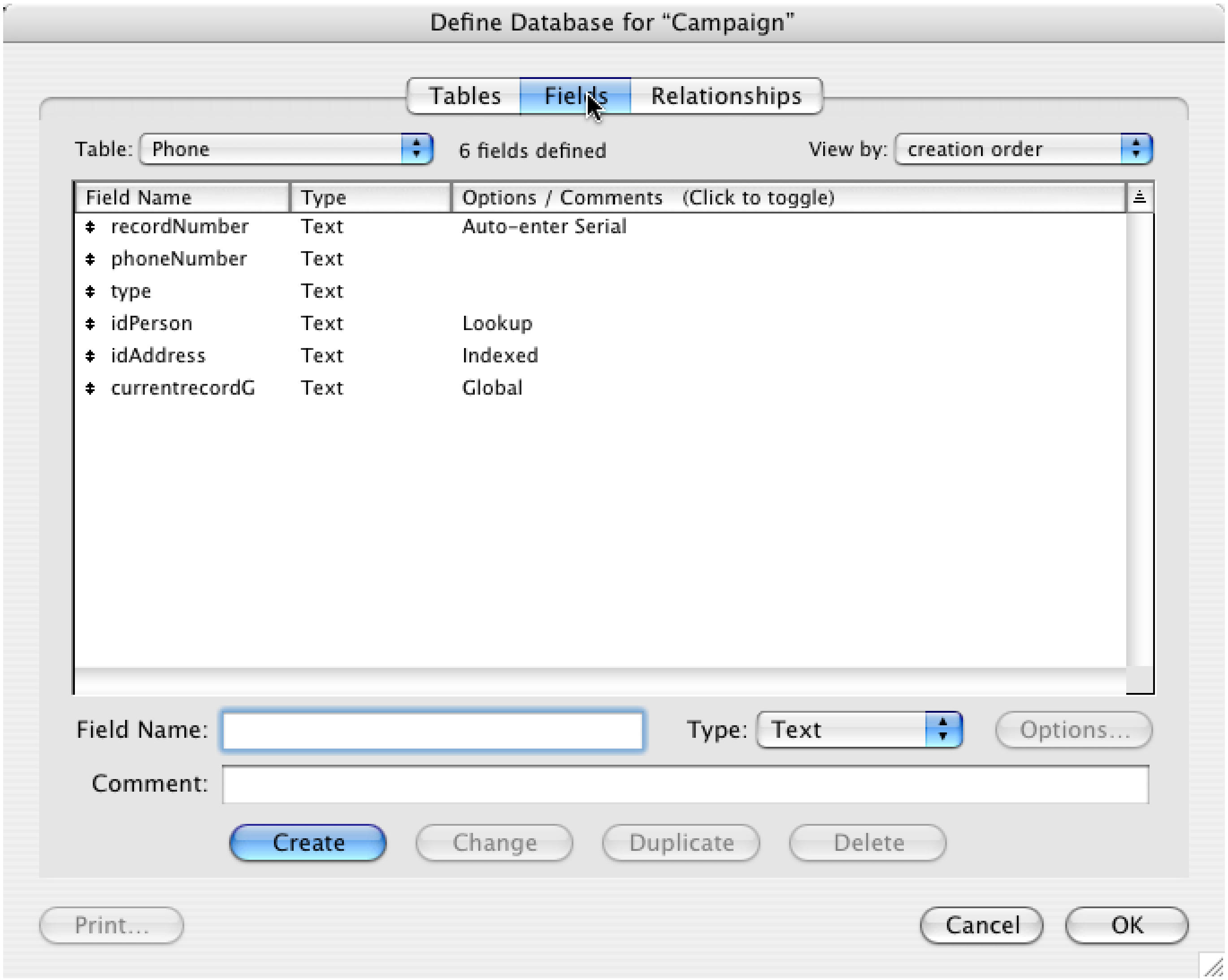
Table Name: Phone

Figure 1. Three tables in one file, each table with its own set of fields.

address, regardless of how many voters reside there. And, of course, they also want the mail pieces to be addressed to all voters at that location, so the Address table must have a one-to-many relationship to the Person table. For phone calls, they want to call each person at the phone number least likely to offend, so they want to see the database from the POV of the person. For rides for voters on election day, they want to see the phone numbers and voters from the POV of the rendezvous address, so the phone numbers need to be directly associated with addresses and only secondarily with people.

In previous versions of FileMaker, this database requires three files because a file and a table are, in effect, one and the same. In FileMaker 7, we must decide how many files to create, although the number of tables remains the same, three. In this small example, there is no reason to have more than one file, and one file simplifies security set up and other housekeeping, so we create a single new file, Campaign, to start the database. Field definitions are now associated with tables, not files, so each table has its own set of field definitions, defined one table at a time. Three different **layouts** can support the three main campaign activities.

We now need to specify three **data environments,** one for each POV of use of the Campaign file. A data environment is created in the **relationships graph.** A data environment is specified and graphically represented by a group of **Table Occurrences (TOs),** usually connected by relationships. This representation of the **data environment** we call a **Table Occurrence Group (TOG).** For Campaign, the **relationships graph** is shown in Figure 3a.



Define Database for "Campaign"

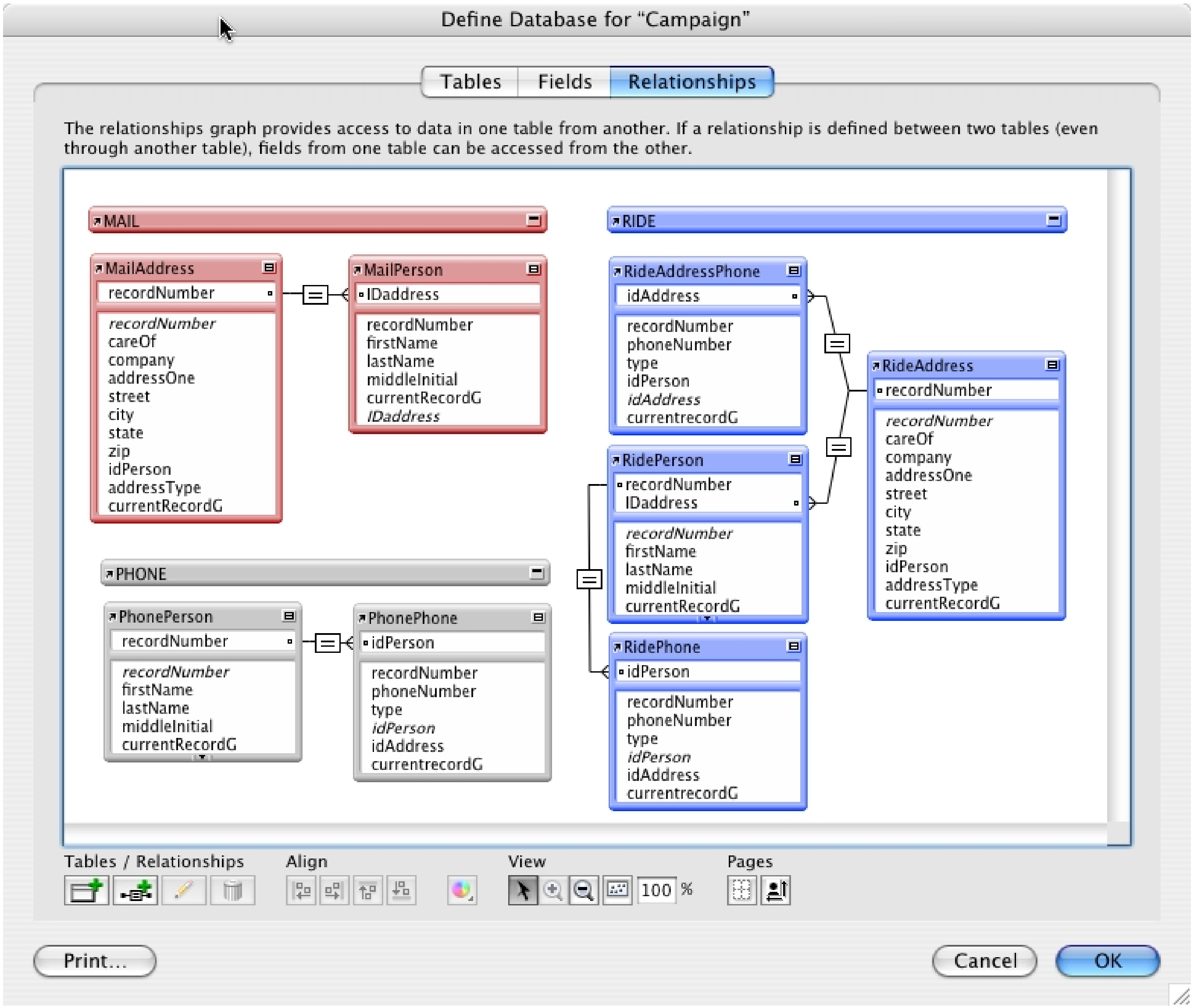
Table: Phone   6 fields defined   View by: creation order

Field Name	Type	Options / Comments (Click to toggle)
recordNumber	Text	Auto-enter Serial
phoneNumber	Text	
type	Text	
idPerson	Text	Lookup
idAddress	Text	Indexed
currentrecordG	Text	Global

Field Name: Type: Text

Comment:

Figure 2. Fields in the Phone table.


Define Database for "Campaign"

The relationships graph provides access to data in one table from another. If a relationship is defined between two tables (even through another table), fields from one table can be accessed from the other.

Figure 3a. The three data environments of the Campaign file.

The three **TOGs** are labeled with a long pseudo-TO for convenience (MAIL, PHONE, RIDE) and given different colors for clarity. Understanding the **relationships graph** is the key-to-the-kingdom of FileMaker 7.

The Relationships Graph

> *"A map is not the territory it represents…"*
> – Alford Korzybski

Korzybski's infamous observation warns against confusing our personal maps of the world with the world itself. Specifically, it admonishes those who see a mountain on their maps where none actually exists yet who insist that the mountain must be there because the map says so. The universality of this error is painfully clear. And it warns against bringing an obsolete or inappropriate map to FileMaker 7. A Zen no-mind, suspending without forgetting what you think you know about FileMaker is critical.

So, although an odd way to begin, let's start with what is *not* going on in the relationships graph in Figure 3a:

- this is *not* an ER (entity-relationship) diagram (see Figure 3b below)
- **TOs** are *not* the same as tables, even though each **TO** is an alias of one **primary table**

- the **relationships** are *not* the same objects they were in previous versions of FileMaker, and cannot be named or viewed in a list
- the **relationships graph** applies *only* when using this file and does *not* create any permanent or universal connections among tables

ER diagrams show tables (rectangles) and relationships (various flavors of lines connecting keys) to represent the fundamental data structure of a collection of **tables** in a **database**. The **relationships graph** is very different.

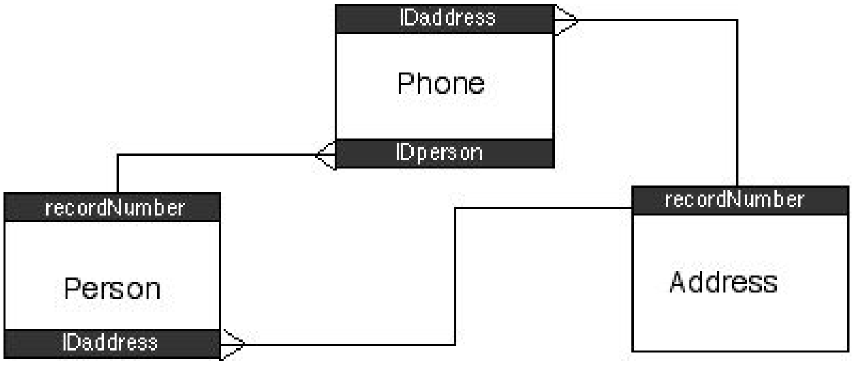


Figure 3b. Typical ER diagram for campaign example.

Not Your Mother's Relationship

A FileMaker 7 relationship is very different from what has heretofore been called a FileMaker relationship. This is much to the good, but it is quite confusing for the meaning of a key FileMaker word to change so radically. Consider this: in FileMaker 7 a relationship cannot be named, nor can relationships be viewed in a text list. Many of us initially thought this was a silly omission on the part of FileMaker, Inc. I now believe it is fundamental to the new architecture of FileMaker 7, and an indicator of how different FileMaker 7 really is.

So here is a bit of a teaser for what follows. What does it mean that a relationship in FileMaker 7 cannot be named? It means that a relationship is no longer an "object." What is it then? It is more like a "property" or a preference, or a set up. Of course, relationships are no less important – in fact they are much more powerful. But their status in the scheme of things FileMaker is changed dramatically.

A relationship can be created by dragging a relationship line from one TO to another on the relationships graph, or by using the Edit Relationship dialog. Dragging connects one field in one TO with a field in another TO. If one TO field is an auto-enter serial number field, then FileMaker assumes a one-to-many relationship and draws a relationship line ending in the usual crow's foot. Other sorts of relationships have different line endings and different symbol boxes in the middle of the line. Clicking on the relationship line opens the Edit Relationship dialog where the relationship can be refined or reconfigured completely. Figure 4 shows the Edit Relationship dialog for the relationship between two TOs in the Campaign file.

The Edit Relationship dialog shows two big differences in the relationships of FileMaker 7 versus the relationships of previous versions. First, the relationship can have many conditions using different key pairs. This allows relationship configuration to

Relationships Graph (Figure 3a)	**ER Diagram** (Figure 3b)
rectangles are table occurrences (TOs)	rectangles represent tables
a table may be represented many times	a table is usually represented once
each TO, even when referring to the same primary table, must have a different name	a table has one name
relationship lines are multipurpose: basic data, process logic, interface	relationship lines show data structure
relationship lines may involve many different key pairs and conditions	relationship lines connect 2 keys, 1 key per table

substitute for most of the complex key calculations previously required, as well as adding powerful new capabilities. Second, the relationship is inherently bidirectional. So, one relationship now replaces the two that may have been required in previous versions. You do not have to use the return direction but you cannot turn it off – it is there and is used if you do something that requires it, whether you intended that or not. We shall see that there is a good reason for this.

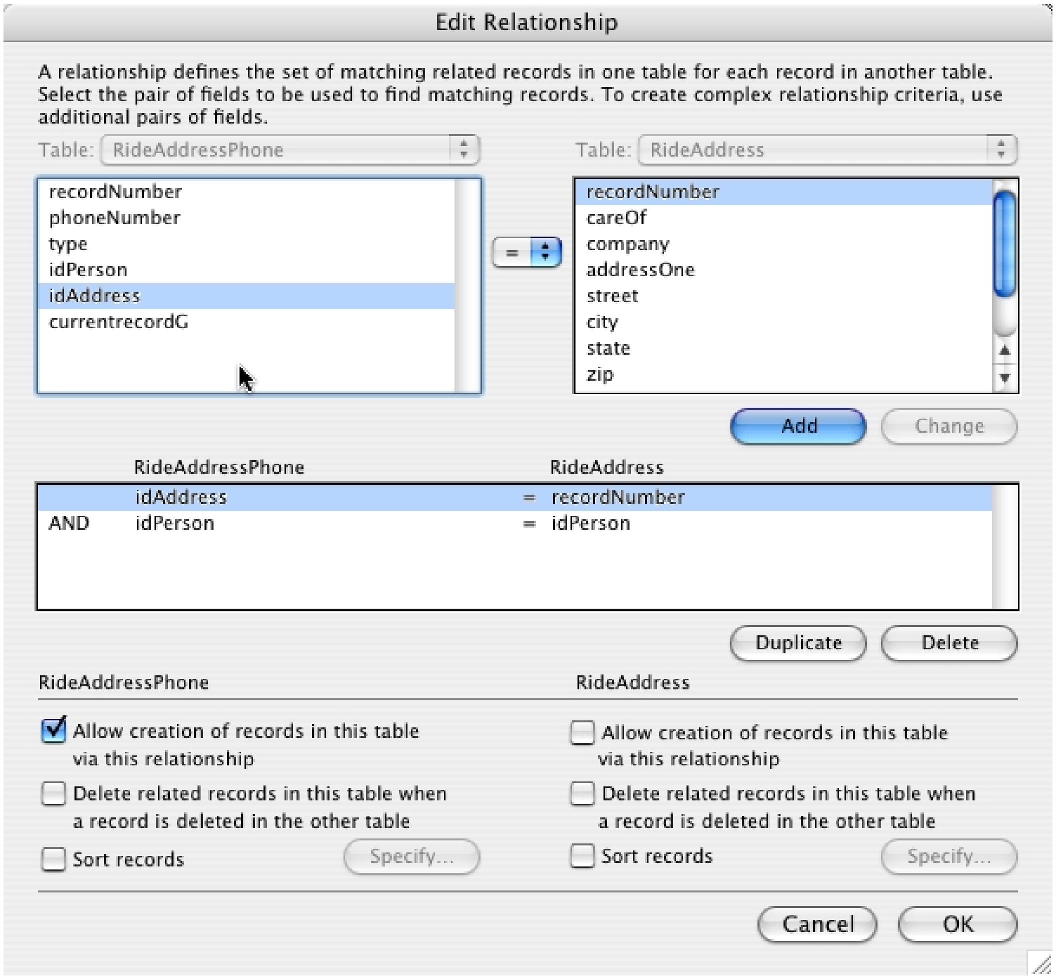
TO ≠ Table

I am considering having the title of this section put on a tee shirt: TO ≠ Table! This is the answer to the Zen koan that is FileMaker 7.

TOs point to tables, TOs are table aliases, so why is it important to distinguish strongly between TOs and tables? The reason is that TOs belong to a larger functional group: their **table occurrence group (TOG)**. A TOG specifies a **context** or **data environment** in which that TO is embedded. A given table may be aliased many times (have many TOs) in a relationships graph, and many more times in several other files' relationships graphs. Every one of these TOs has equal access to the data in the common primary table. However, every TO, each belonging to a different TOG, can have an entirely different set of relationships to other tables.

Consider our Campaign example (see Figure 3a). The Person table is aliased three times in the graph, with the TOs named: MailPerson, PhonePerson and RidePerson. When layouts are pointed to one of these TOs (see Figure 5a), they each have the same access to Person table data – direct and including all fields and records (except as restricted by security set-ups). However, a layout pointing to the PhonePerson TO cannot see any information from the Address table. A layout pointing to the MailPerson TO cannot see any phone numbers. Thus, each TO exists in its own data environment, or context, defined by the TOG of which it is a part.

Figure 5a shows a layout in layout mode. This layout points to the RidePerson TO. Any field from the Person table can be dragged directly onto the layout. However, to see data from, for example, the Address table requires a prefix to the Address table field name. In FileMaker before 7, this prefix was the name of the relationship to the table to be accessed. But relationships cannot be named! Instead, the field name prefix is a TO name. Which TO name? Any TO in the same TOG as the RidePerson TO, but no TOs



Edit Relationship

A relationship defines the set of matching related records in one table for each record in another table. Select the pair of fields to be used to find matching records. To create complex relationship criteria, use additional pairs of fields.

Table: RideAddressPhone

Table: RideAddress

	RideAddressPhone		RideAddress
	idAddress	=	recordNumber
AND	idPerson	=	idPerson

RideAddressPhone

☑ Allow creation of records in this table via this relationship

☐ Delete related records in this table when a record is deleted in the other table

☐ Sort records

RideAddress

☐ Allow creation of records in this table via this relationship

☐ Delete related records in this table when a record is deleted in the other table

☐ Sort records

Figure 4. This Edit Relationship dialog specifies two conditions, a variation on the Campaign example.

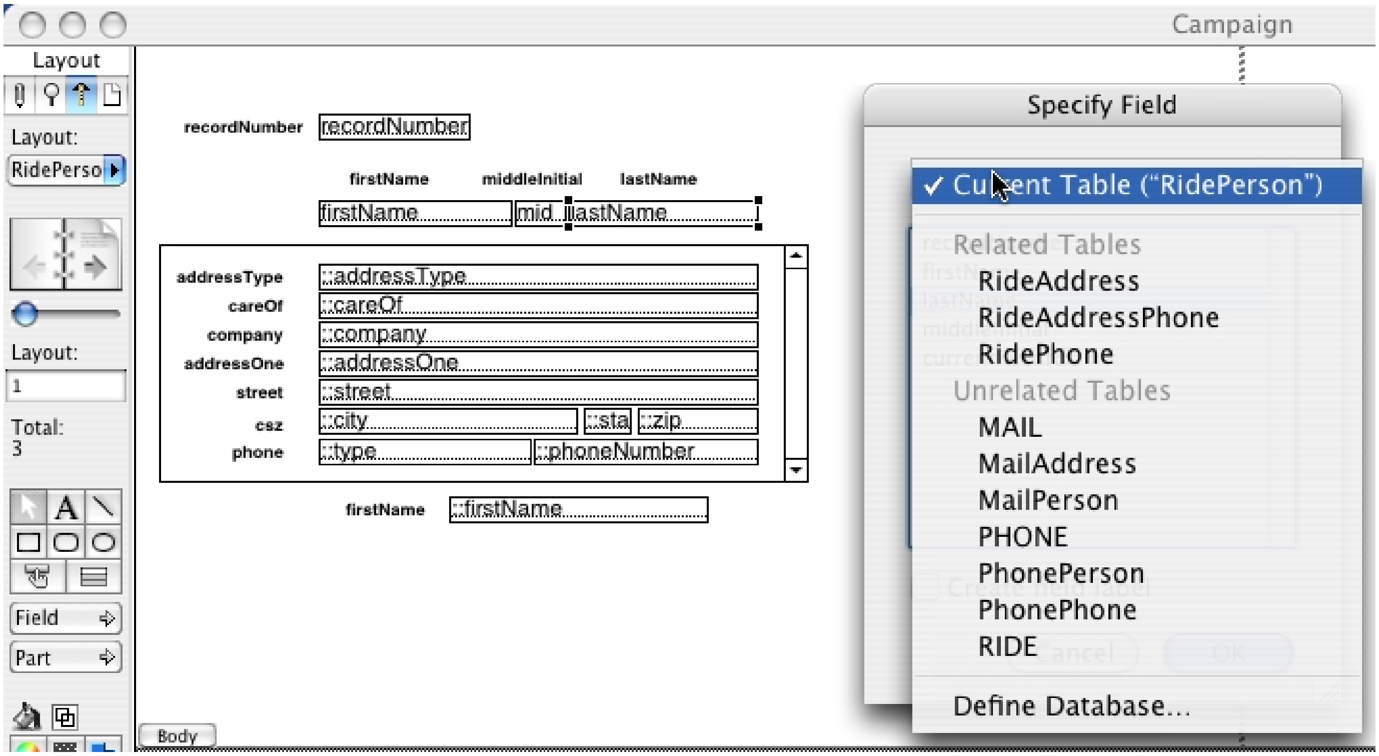
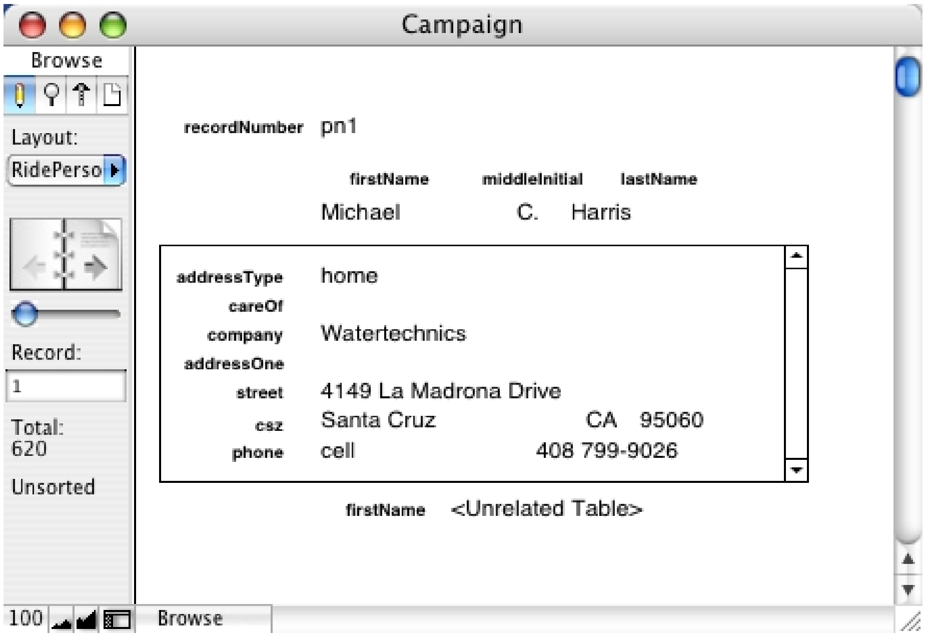


Figure 5a. Specify fields dialog. Note that the pop-up list items in all uppercase ("MAIL") are dummy TOs used as TOG labels, and not meaningful choices.

outside this TOG. (There is an exception for Global fields, but it is a minor case.)

When specifying a new field for a layout we have a list of choices: the current table (in this case, Person), related tables, or unrelated tables. We may also go into Define Database and create something new to use. But wait a minute! The choices aren't really table names, they are TO names. The difference? Well the TOs MailPerson, PhonePerson, and RidePerson all point to the Person table, but we have no access to the "Unrelated Table" table data through the TOs Mail-Person and PhonePerson. If we place a field from those unrelated tables on the layout we get a "unrelated table" error. (See Figure 5b.)

Let's interpret the Specify Field pop-up window. First, we can disregard the dummy TOs used for labels in the relationship graph, all the items in all uppercase. We can also ignore the "Define Database" item. What's left? Well, the "related tables" are really the names of the TOs in the TOG to which our layout's TO belongs. In other words, TOs that are part of our current data environment or context. The "unrelated tables" are TOs outside the current TOG.



recordNumber pn1

firstName Michael
middleInitial C.
lastName Harris

addressType home
careOf
company Watertechnics
addressOne
street 4149 La Madrona Drive
csz Santa Cruz CA 95060
phone cell 408 799-9026

firstName <Unrelated Table>

Figure 5b. Fields from unrelated tables (outside the current TOG) can be placed on a layout but return an error value.

So, a TO is an alias of a table for the purposes of making relational connections to other tables' data. When you point a layout (or a script or field definition) to a TO you are getting this relational context as well as direct access to the table's data. A table per se has no connections to other tables, but is simply a repository for data. Therefore: TO ≠ Table!

TO ≠ Table!

New Schema Revealed

The characteristic differences of FileMaker 7 described so far add up to more than the sum of their parts. This is a new FileMaker architecture, or schema, or paradigm – pick your term. There is an entirely new yet central design object in FileMaker 7: the Table Occurrence Group (TOG). TOGs determine the data environment in which you operate at any given time. To use FileMaker 7, TOGs must be properly designed, carefully used, and constantly referred to (through one of their TOs). The relationship graph specifies the **data environments** in which a file operates and the current TOG selects the specific current environment from the TOGs available – and no file can use or is affected by the TOGs of another file.

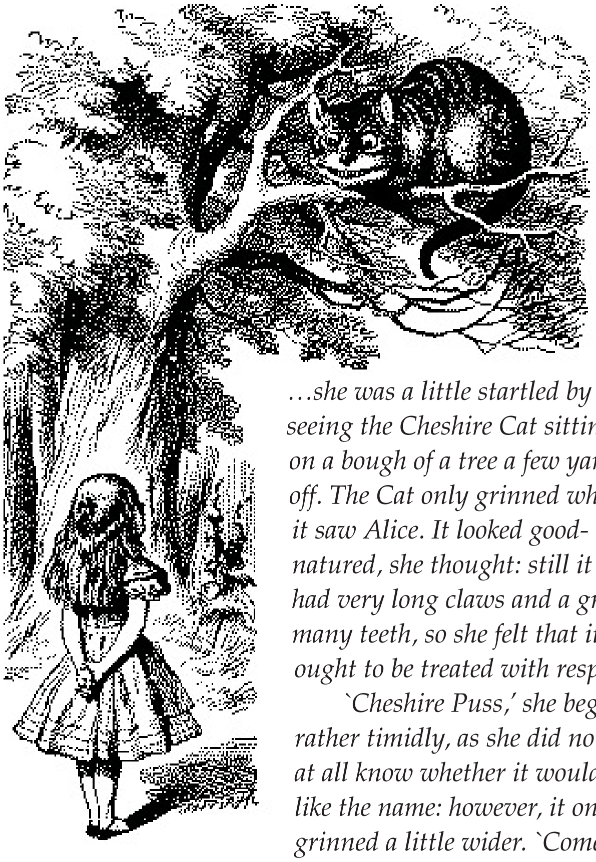
A more complete version of the quote from Korzybski is:

> *"A map is not the territory it represents, but if correct, it has a similar structure to the territory, which accounts for its usefulness."*

When you construct a "correct" TOG in a relationships graph, you are drawing a new map which creates a data environment, a territory, in which FileMaker can operate. And you may create many such territories, with each file a country of its own.

Another way to consider TOGs is to start from the point of view of a layout. Layouts are created with a specific purpose in mind: data entry, a list, reports, novel views of data, utility work, or some other business purpose. In our Campaign example, one layout might be used for mailing labels. The volunteers working on mailing do not need access to phone numbers and it may be a security issue to keep it that way. One way to manage this is to create a data environment for the mailing labels layout which makes it impossible to see phone numbers while still allowing the workers to fiddle with the label layout. This can be done by pointing the layout to a TO (MailAddress), which is part of a TOG that has no connections to a TO with the primary table Phone (see Figure 3a). Instead of working on a relationships graph in the abstract, it might be helpful to begin relationships specifications by considering the purposes of the layouts you need. Then create a given TOG for a specific layout and purpose, if a suitable TOG does not already exist.

TOG Anatomy



…she was a little startled by seeing the Cheshire Cat sitting on a bough of a tree a few yards off. The Cat only grinned when it saw Alice. It looked good-natured, she thought: still it had very long claws and a great many teeth, so she felt that it ought to be treated with respect.

`Cheshire Puss,' she began, rather timidly, as she did not at all know whether it would like the name: however, it only grinned a little wider. `Come, it's pleased so far,' thought Alice, and she went on. `Would you tell me, please, which way I ought to go from here?'

`That depends a good deal on where you want to get to,' said the Cat.

`I don't much care where –' said Alice.

`Then it doesn't matter which way you go,' said the Cat.

`– so long as I get somewhere,' Alice added as an explanation.

`Oh, you're sure to do that,' said the Cat, `if you only walk long enough.'

– Alice in Wonderland by Lewis Carroll

Alice could take any path, since she just wanted to get somewhere. **Paths** in FileMaker 7 cannot be treated so casually. As the central architectural objects, TOGs, as you might expect, have their own set of properties, rules and requisite anatomy. The primary and essential TOG rule is:

> *There must be a unique (or no) path between any two TOs in a relationships graph.*

Why is this? Since a related field reference uses only a TO name (for example, "MailAddress::Street"), FileMaker 7 must know the series of relationships (the **path**) to use to select data to show from the primary table of the related TO named. There is no way for the developer to make a choice (say, by specifying a series of named relationships) between two or more alternative paths between two TOs. Therefore the path must be unique so that FileMaker knows unambiguously which path to take.

In previous versions of FileMaker you named the "path"; in FileMaker 7 you name the destination.

Alice must say where she wants to get to. She does this by giving the name of the TO destination, and the TO named points to its primary table. Notice that in previous versions of FileMaker you named the "path" (the one step relationship to another file/table). In FileMaker 7 you name the destination and the path is automatically chosen since there can only be one.

TOGs also require two words of specialized vocabulary:

- **path** – a chain of relationships among a group of TOs, paths have a single beginning and single ending TO; there may be one or more relationship links in a path chain; there may be branching in the relationships of a group of TOs, but every path is both unique and unbranched;
- **path group** – often a subset group of a larger TOG, which consists of at least two TOs linked by a path.

TOG Surfin'

TOGs are their own data environments, but business processes may need to move from one data environment to another, perhaps invisibly to the user. This requires moving among layouts depending on where you are in a process, or "TOG surfing."

In our Campaign example, suppose a campaign worker is responsible for getting voters over 70 years old to rendezvous points for rides. Using a layout pointing to the RideAddress TO, the worker can call the ride rendezvous house and ask if all voters are ready to be picked up. If not, he can "surf" to a layout pointing to the PhonePerson TO and see all phone numbers associated with that person, including a cell phone number which was not available from the RideAddress layout.

It is easy to imagine that some business processes may benefit from, or even require, extensive TOG surfing. It may also turn out that TOG surfing accomplishes nothing that cannot be done better another way. This can only be determined through real world experience and may not be settled for some time.

Context

> *"A rose is a rose is a rose."* – Gertrude Stein

Even if your FileMaker work never requires anything as involved as TOG surfing, you will nevertheless find that keeping your place among multiple files and tables is difficult. You are required to specify "What are you talking about?" over and over again. In several dialogs the choices are confusing – it is not clear whether you are to specify a file, a layout, a table, or a table occurrence. Since it is possible to give the same name to all these objects, this invites troublesome confusion.

Several preliminary naming schemes for labeling these different classes of objects have been pro-

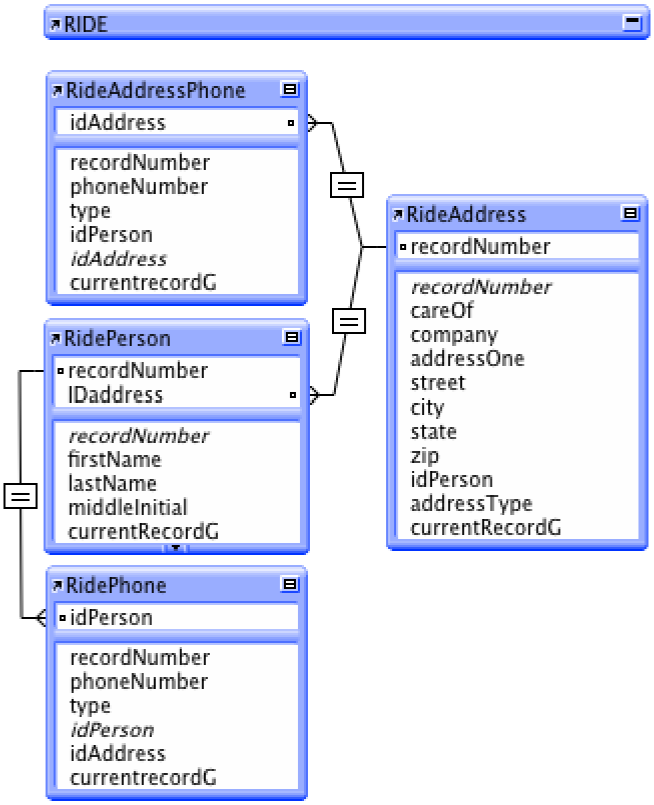


Figure 6a. A Table Occurrence Group (TOG), one of three from Figure 3a. Note the "RIDE" TO is a non-functional TO used as a TOG label.

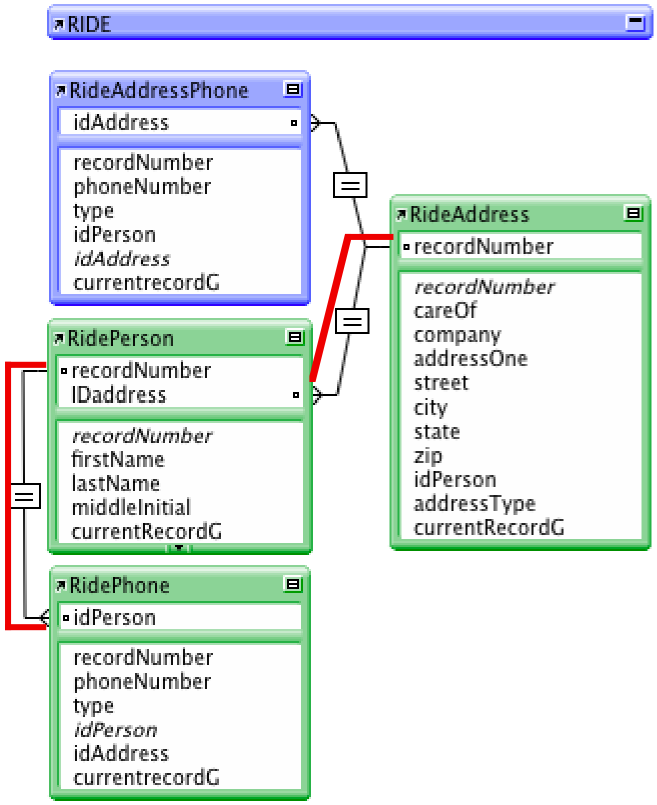


Figure 6b. A path (red lines) between the TO "RidePhone" and the TO "RideAddress" connects a path group (green): "RideAddress", "RidePerson" and "RidePhone".

posed. How you name objects should suit you and the problem at hand. However, you really must have a scheme that allows you to tell from a name itself whether a label identifies a file, a table, or a table occurrence. It is a good idea to also have additional naming systems for layouts and TOGs.

Stein's quote may be interpreted to mean that a rose (the flower) is a rose (the word) is a rose (the experience). But which word refers to which? (Yes, well, that's part of the art of the phrase.) In FileMaker 7 you had better know. For us maybe it should be "a roseFL is a roseWO is a roseEX".

Vocabulary

FileMaker 7 is a foreign country, and it speaks a different language – some words have gone away, some words have new meanings, and some new words are required. Some old structural terms with significantly different meanings in FileMaker 7 are **file**, **layout, relationship, window,** and **global**. Some new structural terms required for making sense of FileMaker 7 are **table, table occurrence (TO), table occurrence group (TOG), TOG surfing, path, path group,** and **context.** This paper uses some new terms which have been offered by various developers during discussions of FileMaker 7 (path, context) and coins some new terms (TOG, TOG surfing, path group, design space). In six months, a consensus vocabulary will evolve in the FileMaker community. Meanwhile, here are the terms this author finds helpful. (FileMaker, Inc. has its own official vocabulary, which does not always match that of the author.)

home file (AKA: mother file, creation file) – for a table, the file in which it was created; the only file where table field definitions can be added or edited, the file where basic security is managed

file (see home file) – (1) the primary document(s) of a FileMaker database; every FileMaker database must begin with the creation of a file and every database must contain at least one file; a place where the relationships among tables used by the file's layouts and scripts are specified;

(2) a functional manager and/or container of tables: can be for table management, business logic, presentation to users or other purposes;

(3) where tables, fields, scripts and relationships are created and used

table – where the data of a FileMaker database is stored in rows and columns; FileMaker 7 uses the term "table" in several design dialogs where "table occurrence" is meant (for example, see Figure 5a).

A vocabulary challenge: in one sentence, define a "file" in FileMaker 7.

table occurrence (TO, sometimes **TOC** or **table alias)** - in the relationships graph, an alias for a table; TOs determine the table data which a layout can access: the layout sees all the data from the primary table which the TO aliases; a layout also sees data from related tables as specified by the TO Group (TOG) to which the TO belongs

relationships graph – place for defining data relationships in a file (note: *not* equivalent to "in a database" unless the database consists of one file); a relationships graph contains one or more TOGs, each consisting of one of more TOs linked by relationships; a relationships graph may have more than one TO referencing the same primary table but every TO must have a unique name

relationship (or **relation) –** in a file's relationships graph, a conditional connection between two table occurrences (TOs), each representing the data in a primary table; a filter for specifying the conditions under which data from one table can be seen while using another table; in FileMaker 7 inherently bi-directional, multi-key, and multi-conditional; also new in FileMaker 7, relationships work together in chains (paths) where the net condition for data access in another table is the logical sum of the conditions of the relationships in the chain; (note: As far as I can tell "relation" and "relationship" mean the same thing. I prefer the shorter word, but I usually defer to the preference of FileMaker, Inc. for "relationship." Some developers have proposed that "relation" be used for the link between two tables and "relationship" for the path between TOs in a TOG, or even to the name of the TO. I am inclined to think "path" is a less-likely-to-confuse choice for "multi-relation" chains.)

primary table (also **base table) –** for a TO (or layout) the data for which the TO has unrestricted access, the table for which the TO is an alias

context (point-of-view [POV], data environment) – the current point of view of the data, script step, calculation or layout; usually specified by indicating a TO (and, therefore, a TOG)

table occurrence group (TOG) – a group of table occurrences linked by relations; a TOG obeys the rule: there must be a unique (or no) path between any two TOs on a given file's relationships graph

layout – a view of the database's tables; one layout is always directly connected to one and only one TO; a layout can see all the data of the primary table aliased by this TO and can see data of other tables through paths in the TOG to which the layout's TO belongs

layout data environment (see also equivalent, "context"**)** – for a given layout, the table(s) data it can see and the conditions (path connections through conditional relationships) through which the layout sees that data.

path – a chain of relationships among a group of TOs, paths have a single beginning and single ending TO; there may be one or more relationship links in a path chain; there may be branching in the relationships of a group of TOs, but every path between any two related TOs is both unique and unbranched

path group – often a subset group of a larger TOG, which consists of at least two TOs linked by a path

surfing TOGs (TOG surfing) – a business process technique of moving from one layout to another where successive layouts look at the same data through TOs in different TOGs (which equals different data environments); the result is that at different steps in the business process the user’s view of non-primary (related) tables is managed

> FileMaker 7 speaks a new language – some words have gone away, some words have new meanings, and some new words are required.

design space – an imaginary space representing a multi-dimensional range of design choices possible in FileMaker 7; a rough analogy to physical space except that it may have more than three or four dimensions

window - a view of data based on a layout in a file; a FileMaker file (or database) may have multiple windows open simultaneously, each viewing a different layout in the file or viewing the same layout with a different found set of records; can be individually named

database – a collection of tables in at least one file used together for some common purpose (Note: terminology in the FileMaker community is notoriously sloppy in the use of some words. “Database” is often used where “file” is meant. There is a real improvement in clarity when clear distinctions are made among the words table, file, and database. The author uses database as defined above – a personal preference.)

Rules for a New World

The FileMaker 7 Golden Rule
TO ≠ Table

The secrets of FileMaker 7 structure, the “aha” moments, lie in contemplating the new rules of FileMaker 7. Properly understood, these rules brightly illuminate the architecture of FileMaker 7.

1. Field definitions can only be added, removed, or edited within the home file in which a table was created.

2. Basic security can only be managed in the file in which a table was created.

3. Tables have no independent existence apart from the file in which they were created. Tables cannot be created except in a file. Tables cannot be moved (or duplicated) from one file to another. When a file is moved, the tables created by that file move with it.

4. Relationships are not objects but properties (set up options) of TOs. A TOG is a collection of TOs linked by relationships.

5. A TOG specifies relationships only for the file in which it is created and does not have any effect on tables other than the use of their data in that file.

6. Every TO must have a unique name in any given file.

7. There must be a unique (or no) path between any two TOs on a given file’s relationships graph.

8. One layout is always directly connected to one and only one TO.

9. A layout can see all the data of the primary table aliased by its TO. It can also see the data of other tables through paths in the TOG to which the layout's TO belongs. And the data in other tables the layout sees are selected and arranged by the path from its primary table along the path to the related TO.

10. A FileMaker file (or database) may have multiple windows open simultaneously, each viewing a different layout in the file or viewing the same layout with a different found set of records

Whether a complex project turns out good, bad or very ugly depends largely upon the skillful use of table occurrence groups.

11. The relationships graph of a given file establishes data relationships that apply only to that file (and its fields, layouts and scripts).

12. Scripts in a given file can only manipulate data in tables referenced by at least one TO in that file.

Baby Steps: Design Spaces

> *Yoshi went to his abbot to begin his FileMaker Zen studies. "When is a table occurrence a table?" said the master. "Thank you, Master," said Yoshi and retired to consider his koan. Over the next two years he went back once a week to hear the question repeated but could not satisfactorily answer it. Finally, on a bright spring day, his sensei said, "When is a table occurrence a table?" "When I may surf without water or board," Yoshi replied. The master then slapped Yoshi hard across the face, and the student was enlightened. "Now," said the master, "what is the right way to build a temple in a Design Space of more than four dimensions?" And Yoshi left to continue his studies.*

Understanding how FileMaker 7 can be used is a necessary first step, a relatively straightforward one. Coming to some understanding of best practices – how FileMaker 7 *should* be used – is a much longer haul. To get started the initial problem requires some grasp of the extent of the design options in FileMaker 7. How many different ways are there to map the structure of a client's business in FileMaker 7?

A **design space** is my attempt to imagine and roughly grasp the large number and huge range of new design possibilities in FileMaker 7. A design space with dimensions of interacting options helps me see how FileMaker 7 might be used to solve a client problem.

I have always seen FileMaker as having a natural structure, and my task as a developer was mapping FileMaker's strong structure to the structure of my client's uses. When done properly at the beginning of a project, this mapping process makes all subsequent development clean and easy. And getting the architecture wrong in the beginning of a project guarantees that the project is unnecessarily difficult and strangles on itself when it reaches a certain level of complexity.

So the overriding problem in FileMaker 7, after understanding what it can do, is: Given multiple ways to map the structure of my client's problem across the structure of FileMaker 7, how do I allocate key parts of my design to the right design spaces in FileMaker 7? (See Figure 7.)

Exploring the design options of FileMaker 7 will take years. Much of the work will be accomplished through trial and error, and the "By God, I'll never do it that way again!" method. Design space thinking may help. For instance, imagine a simple space of design options in three dimensions: How Many (and What Sort of) Tables Do I Create? How Many (and What Sort of) Files Do I Create? How Do I Distribute My Layouts Among My Files?

The table question is actually pretty straightforward. In previous versions of FileMaker the answer to the question: "What is a Record?" was a pretty good rough guide to what files/tables to create. Nothing substantial about this problem changed in FileMaker 7, so prior experience or some pass at relational theory tells you what tables are needed. Since professional practitioners usually come up with pretty much the same tables for a given client problem, this dimension of the design space may be considered to have a very short length. (I expect a few more tables to be created in a FileMaker 7 project because new tables can be created without the overhead of a new file for each new table.) The dimension of files is much longer and ranges from a minimum of one (a new option many a FileMaker developer has been aching for) through as many files as tables (the pre-FileMaker 7 way) to many more files than tables (perhaps where many user-interface (UI) files are created for different users of a common database engine). The third dimension of layout options has a wide range of logical (if not necessarily practical) possibilities. All layouts might go into a single UI file, layouts might be distributed among files on functional grounds – perhaps tracking file functional groups – or layouts might be distributed as they were in FileMaker 6, which is where they are after a conversion from FileMaker 6 to FileMaker 7.

> "I trust I make myself obscure."
>
> – the character Sir Thomas More
> in a "Man for All Seasons"
> by Robert Bolt

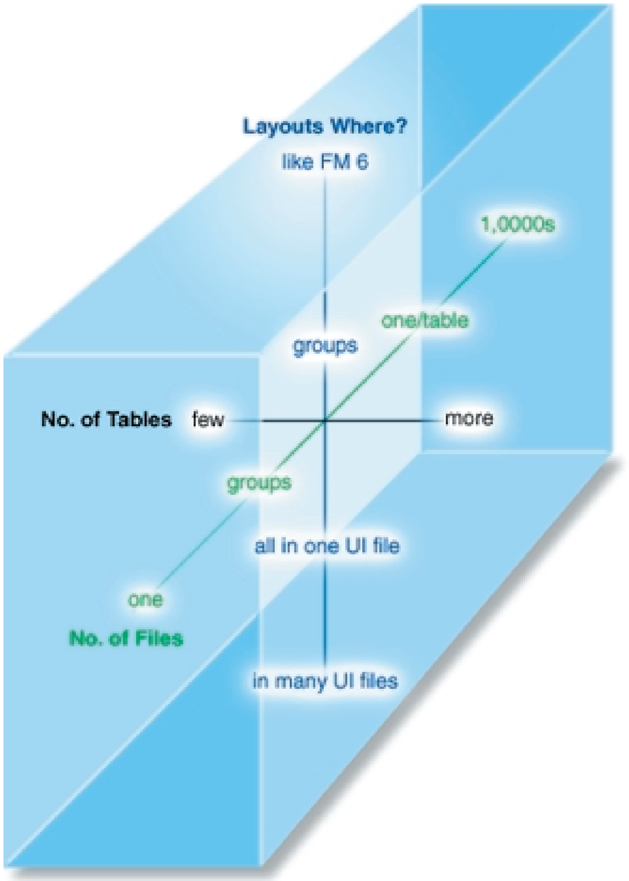


Figure 7. A simple 3D design space.

Figure 7 shows this simple design space, shaped like a thick slice of bread, because in the "tables" dimension little real variety is likely for a given project. So, roughly, there are four quadrants here with different combinations of design choices. If you choose to create one file for the entire database, all your layouts are necessarily in one file and you find yourself somewhere in the lower, nearest quadrant. If you group tables into functional groups and layouts likewise, you are in the upper nearest quadrant. Designing as you had to in FileMaker 6, you are wandering around in the upper further quadrant. Having many UI files, one for each of a hundred different work groups, for example, you might create many nearly identical layouts and place them in the appropriate customer UI files. You are now in the further lower quadrant.

The very rough description of the quadrants in this design space only suggests the options. It is easy enough to see the vast possibilities, but on what basis do you choose where to build your next database? Our company intends to cautiously explore the new ground of FileMaker 7. We will initially group tables into functional files when the tables are modified together and require joint security and development. One of our functional groups is likely to be a group of name and address tables – Person, Address, Telcom, Company, and Staff – inside a single Contacts (or maybe Customer) home file. For web-based application services, we may move to a functional group of files (a software "engine") surrounded by many unique UI files containing both the user interface and business logic for a given customer.

Even this simple example demonstrates that, with FileMaker 7, we enter an age of a genuine embarrassment of riches.

Table Space

- What fields do I need?
- What tables do I create?
- What keys do I use?
- Where do I store globals?
- Where do I store calculations?

File Space

- What is a File? What functions does it have?
- Which Files are home to which Tables?
- How does a business process move among Files? Why?
- Do I have functional layers of Files? If so, according to what scheme?

Layout Space

- Which layouts point to which TOs?
- Which layouts go in what Files?
- What is the "Data Environment" of my layouts?

TO Space

- What relations do I create? How complex?
- What paths?
- What TOs do I create?
- What TOGs do I create? To what end?
- Do I "surf" a business process through TOGs? To what end?
- In what File(s) do I put which TOs/TOGs?

Hyper Space

Given multiple ways to map the structure of my client's problem across the structure of FileMaker 7, how do I allocate key parts of my design to the right design spaces?

Putting on the Buckskins

Sometime in the next six months or year, an understanding of what FileMaker 7 *can* do is likely to become commonplace. Soon the FileMaker community will be flooded with classes and workshops, books and articles, presentations and competency tests, and on-line little-bits-of-everything. However, an understanding of best practices, of what *should* be done with FileMaker 7's new powers, will be much harder-won, a long trek into the tough wilderness of the real world of work.

The geniuses in the software salt mines at FileMaker, Inc. have given us an opportunity to join the Lewis and Clark expedition. Personally, I find this an irresistible offer and I am putting on my buckskins and keeping my powder dry. The trip may involve a few bears now and then but, I wouldn't miss it for the world.

A professor approached a new grad student in the lab and handed him a fresh fish. "Describe the fish," said the professor. The grad student went to the library and wrote a long report on the fish - its species and variety, its evolutionary history, its current population numbers, its parasites and so forth. He handed the report to the professor and the professor looked it over briefly and tossed it aside. "Try again," the prof said. After several more tries and rejections, the professor finally said in disgust, "Look at the fish!" Two weeks later the fish was totally decomposed, but the student had looked at the fish and his description accepted.

– traditional story for first year biology graduate students

When an ignorant person understands he may become a teacher. But a teacher may understand only by becoming an ignorant person again.

– paraphrase of saying by Zen sage Ekai

"If you understand … quantum mechanics, you don't understand quantum mechanics."

– Richard Feynman

Several early readers of this paper have said that reading it and then playing with FileMaker 7 for a while is the best way to nail down key concepts. The consensus is that three cycles of reading and fiddling allows you to toss this map aside and get on with your journey.

Thanks to Bill Richardson (billr@mac.com) for first bringing to my attention (a Zen moment) that a layout is attached to a TO not to a table; Kevin Frank (kfrank@kevinfrank.com) for excellent technical edits; to Joe Kroeger (joe@1-1.com) for layout; James Higgins (jwhiggins@mac.com) for general editing; and Russ Kohn (russ@chapsoft.com) for some enlightening early conversations on object-oriented programming and FileMaker 7. As is usual in the FileMaker community, a number of people contributed in various ways to this article. They did so with no motive other than to be helpful to other FileMaker users.

This is a living document and will continue to be corrected and enhanced until it is no longer useful. Please send all critiques and comments to the author at michaelharris@mac.com.

Michael Harris is Senior Partner at Cerné Systems, Inc., a custom database software consultancy and developer. Mike has been using FileMaker since before it was called FileMaker (it originally ran on the first IBM PC) and in twenty years of professional FileMaker practice has served more than 100 clients and been involved in more than 500 FileMaker projects. His customers include Apple, Claris (FileMaker, Inc.), Quantum, Lucasfilm, Esalen Institute and many other major companies, government agencies, and non-profits. He has published many articles on FileMaker, speaks frequently on FileMaker topics and has presented a number of high-end FileMaker classes and workshops. Mike can be contacted via e-mail at michaelharris@mac.com. He is current Chair and co-founder of the FileMaker user group, DIGFM, so he can also usually be found at the DIGFM meetings the third Tuesday evening of each month (except December and January). DIGFM meetings are web cast live and on-demand. For details on upcoming meetings and information on joining web casts, visit the DIGFM web site, http://www.digfm.org/

Cerné Systems, Inc.
PO Box 2307
Santa Cruz, CA 95063-2307
(408) 799-9026
fmannex@cruzio.com